ΠΑΝΕΠΙΣΤΗΜΙΟ ΣΤΕΡΕΑΣ ΕΛΛΑΔΑΣ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΪΑΤΡΙΚΗ

Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων στις Νοσοκομειακές Μονάδες με χρήση Τεχνολογιών Διαδικτύου

ΠΑΠΑΔΗΜΗΤΡΙΟΥ ΧΡΗΣΤΟΣ (ΑΜ 90)

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων

Ευάγγελος Σακκόπουλος

Επίκ. Καθηγητής Π.Δ. 407/80

Λαμία, Αύγουστος 2010

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 18η Φεβρουαρίου 1999.

(Υπογραφή)	(Υπογραφή)	(Υπογραφή)
.....	.....	.....
Αλετράς Αντώνιος	Σανδαλίδης Χάρης	Ευάγγελος Σακκόπουλος
Καθηγητής ΠΑ.ΣΤ.ΕΛ	Λέκτορας ΠΑ.ΣΤ.ΕΛ	Επίκ. Καθηγητής Π.Δ. 407/80

Λαμία, Αύγουστος 2010

(Υπογραφή)

.....

ΠΑΠΑΔΗΜΗΤΡΙΟΥ ΧΡΗΣΤΟΣ

Πτυχιούχος Πληροφορικής με Εφαρμογές στη Βιοϊατρική ΠΑ.ΣΤ.ΕΛ.

Την παρούσα πτυχιακή εργασία αφιερώνω στους γονείς μου Άγγελο και Βικτωρία Παπαδημητρίου και στις αδελφές μου Μαριάννα και Χριστίνα, από τους οποίους αντλώ τις δυνάμεις μου.

Τους ευχαριστήσω από τα βάθη της καρδιάς μου για την αμέριστη στήριξη και πολύτιμη συμπαράσταση που έδειξαν καθ' όλη τη διάρκεια της φοιτητικής μου ζωής στο ΠΑ.ΣΤ.ΕΛ

Επίσης, θέλω να ευχαριστήσω τους φίλους μου για τη διαρκή συμπαράσταση τους και εμπιστοσύνη που έδειξαν και συνεχίζουν να δείχνουν στο πρόσωπό μου. Οι ειλικρινείς ευχαριστίες μου είναι το λιγότερο που μπορώ να τους απευθύνω.

Περίληψη

Η εφαρμογή μας έχει ως αντικείμενο τη βελτιστοποίηση της διαδικασίας ανάθεσης, διαμοιρασμού και τροποποίησης του προγράμματος των εφημεριών και περαιτέρω ιατρικών καθηκόντων στις μονάδες παροχής υγείας. Η ανάγκη ύπαρξης του συγκεκριμένου συστήματος πηγάζει από την πολυπλοκότητα των διαδικασιών συντονισμού, διαμοίρασης και τροποποίησης των επίσημων προγραμμάτων στις νοσοκομειακές μονάδες. Το σύστημα περιλαμβάνει την ανάπτυξη εφαρμογής για φορητή συσκευή και την ανάπτυξη Web Based εφαρμογής η οποία διαχειρίζεται τα δεδομένα της κύριας βάσης δεδομένων του συστήματος και η οποία βρίσκεται σε έναν κεντρικό υπολογιστή (server).

Για τη δημιουργία του συστήματος εφαρμόσαμε σύγχρονες τεχνολογίες διαδικτύου που περιελάμβανε τον προγραμματισμό σε τρείς διαφορετικές πλατφόρμες: Windows Mobile, Web Services, Web. Τις τεχνολογίες αυτές εφαρμόσαμε για την κατασκευή μίας εφαρμογής αποκλειστικά για φορητές συσκευές στον ρόλο του client, την κατασκευή Web Based εφαρμογής η οποία έχει τον ρόλο της διαχείρισης της Κεντρικής Βάσης Δεδομένων και της διαιτησίας του συστήματος και τέλος την κατασκευή Web Service για την πραγματοποίηση των ασύρματων συναλλαγών μεταξύ των εφαρμογών και των βάσεων δεδομένων.

Συγκεκριμένα, έγινε μελέτη του υπάρχοντος μη αυτόματου συστήματος διαχείρισης εφημεριών και καθηκόντων που υπάρχει σε δύο μεγάλα και σύγχρονα νοσοκομεία, πραγματοποιήθηκε καταγραφή των προβλημάτων που υπάρχουν και το κατά πόσο ένα αυτοματοποιημένο σύστημα διαχείρισης θα βοηθούσε τους εργαζομένους της νοσοκομειακής μονάδας (νοσηλευτικό, ιατρικό και διοικητικό προσωπικό) και θα έλυνε καίρια ζητήματα οργάνωσης. Πραγματοποιήθηκε σχεδίαση του συστήματος διαχείρισης εφημεριών/καθηκόντων βάσει των απαιτήσεων των χρηστών που απορρέουν μέσα από τα αποτελέσματα της στατιστικής μελέτης από ερωτηματολόγια και συνεντεύξεις.

Όλα τα παραπάνω οδήγησαν στη δημιουργία ενός συστήματος που εμπλέκεται προκειμένου να ρυθμίσει ζητήματα οργάνωσης και διαχείρισης των εφημεριών μεταξύ των εργαζομένων και των υπευθύνων προσωπικού. Η φορητή εφαρμογή προσφέρει στο χρήστη τη δυνατότητα ανάγνωσης, επεξεργασίας, αλλαγής και αποστολής των καθηκόντων προς τον κεντρικό υπολογιστή (server), καθώς και τη δυνατότητα διαπραγμάτευσης των καθηκόντων και των εφημεριών με άλλους χρήστες του συστήματος που διαθέτουν την εφαρμογή. Ένα θέμα το οποίο από πολλούς θεωρείται ταμπού καθώς είναι πολύ συχνή αφενός η εμπλοκή

τρίτων προσώπων και αφετέρου το φαινόμενο μη επίσημων αλλαγών που πραγματοποιούνται μεταξύ των εργαζομένων των νοσοκομειακών μονάδων. Η χρήση του συστήματος αυτού μπορεί να γίνεται εντός και εκτός του χώρου της νοσοκομειακής μονάδας προκειμένου οι χρήστες να παραμένουν ενήμεροι για οποιαδήποτε αλλαγή ή τροποποίηση προκύψει κατά τη διάρκεια της ημέρας.

Η γενικότητα της χρήσης του συγκεκριμένου συστήματος έγκειται στο γεγονός ότι οι υπηρεσίες διαδικτύου που προσφέρονται είναι επαναχρησιμοποιήσιμες, καθώς είναι δυνατή η πρόσβαση στις πληροφορίες που παρέχουν και έχουν δημοσιευθεί σε οποιοδήποτε μέρος του Διαδικτύου και από άλλες εφαρμογές.

Λέξεις Κλειδιά: Εφαρμογή Φορητών Συσκευών, Υπηρεσίες Διαδικτύου, Εφαρμογή Διαδικτύου, Εφαρμογή Διαχείρισης Προσωπικού

Abstract

This application was built in order to help improve the way shifts as well as farther medical duties are allocated, distributed and modified within health care units. The need for the specific system stems from the complexity of the actual process of coordination, distribution and modification of the official timetables in heath care units. The system includes the development of a mobile device application and the development of a web based application which administers the data of the main data base of the system located in a central computer (server).

For the development of the system modern web technology was applied which included programming on three different platforms: Windows Mobile, Web Services, Web. The above technology was used for the construction of an application strictly for portable devices in the role of a client, also for the construction of a Web Based application which administers the Central Data Base and arbitrates the system and finally for the construction of a Web Service which enables wireless transaction between the applications and the data bases.

A study of the existing, non-automatic shifts and duties timetable administration system was conducted in two big, modern hospitals. The problems that occurred due to the lack of an automatic personnel management system were taken into account while the solutions that such a system would offer in terms of administration (health care personnel, doctors and administrative personnel) were considered. The planning of the shifts/duties management system was based on the demands of the users as those were found in statistic analysis, questionnaires and interviews.

The above led to the construction of a system that operates in order to regulate and organize the administration of duties among the personnel and the personnel managers. The mobile device enables the user to see, process, alter and send the duties to the central computer (server) as well as to negotiate the duties and times with other users of the system and the application; a matter considered rather sensitive by many, since very often than not we have the interference of a third party as well as non official changes made in the times and which are kept between the staff of the health care units. The use of this system is possible both inside and outside a health care unit in order for the users to remain up to date with any changes to their timetable occurring during the day.

The possibility of a general use of the system is based on the fact that the web services offered can be reused, as access to information offered and published anywhere on the Web and/or by other applications is possible.

Keywords: Mobile Devices Application, Web Services, Web Application, Personnel Management Application

Πίνακας περιεχομένων

Πίνακας Εικόνων

Πίνακας Στατιστικών Στοιχείων

Πίνακες

1

Εισαγωγή

1.1 Οι φορητές συσκευές στον χώρο εργασίας των νοσοκομειακών μονάδων

Οι φορητές συσκευές χειρός επιτρέπουν στους ανθρώπους τη χρήση IT (Information Technologies) χωρίς να χρειάζεται να βρίσκονται σε μια συγκεκριμένη θέση. Εκτός από την παραδοσιακή λειτουργία φωνής, οι φορητές συσκευές, όπως notebooks, προσωπικοί ψηφιακοί βοηθοί (PDAs), palmtops, κινητά τηλέφωνα, ψηφιακές φωτογραφικές μηχανές, κλπ προσφέρουν υπηρεσίες μετάδοσης δεδομένων όπως:

Καθολικό σύστημα για κινητές επικοινωνίες (GSM) – επιτρέπει στα κινητά τηλέφωνα να στέλνουν και να λαμβάνουν δεδομένα, π.χ. σύνδεση με το διαδίκτυο με ένα ρυθμό παρόμοιο με του dial-up modem.

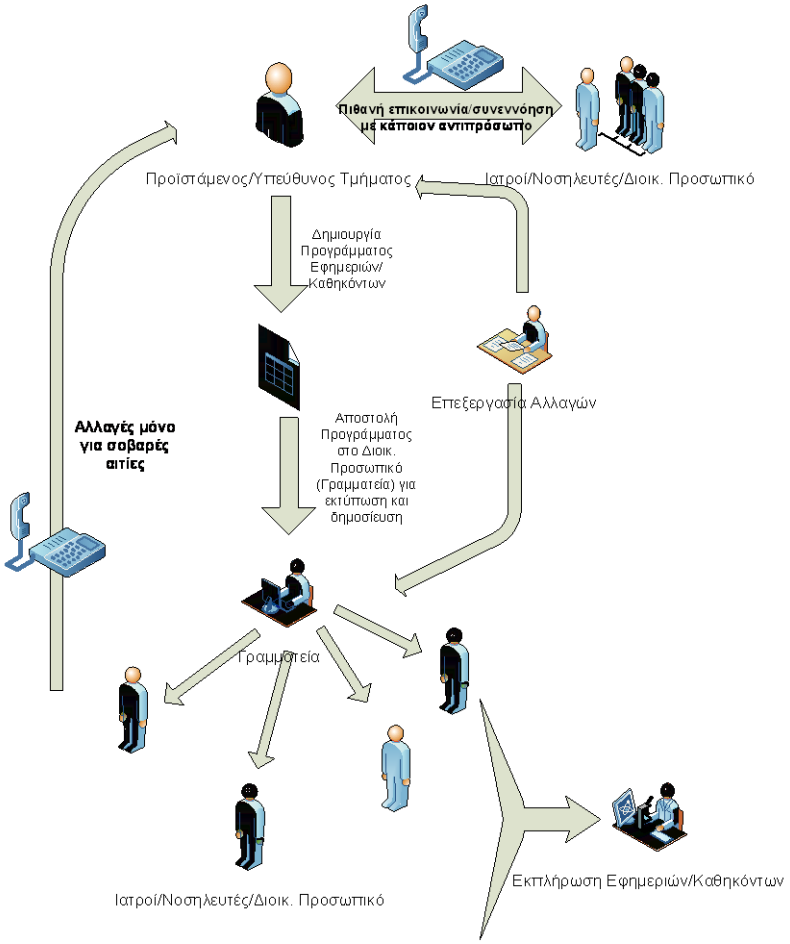
- General packet radio service (GPRS) – μια πάντα ενεργή υπηρεσία δεδομένων παρόμοια με την ευρυζωνική, αλλά σε μικρότερους ρυθμούς μετάδοσης.
- 3G – η "τρίτη γενιά" κυψελοειδών υπηρεσιών δεδομένων, επίσης προσφέρει πάντα ενεργές συνδέσεις σε ρυθμό συγκρίσιμο με τις ευρυζωνικές.
- Πρόσβαση στα e-mail και απλοποιημένες εφαρμογές γραφείου (office εφαρμογές).

Οι σύγχρονες εξελίξεις στους τομείς των τηλεπικοινωνιών και της πληροφορικής είναι προφανές ότι δημιουργούν ένα νέο περιβάλλον εργασίας και λειτουργίας στους χώρους παροχής υπηρεσιών υγείας. Χαρακτηριστικό παράδειγμα αποτελούν οι ασύρματες επικοινωνίες (για τον τομέα των τηλεπικοινωνιών) και οι φορητές υπολογιστικές συσκευές (Personal Digital Assistants - PDAs) για τον τομέα της πληροφορικής. Ο όρος των "κινητών υπολογιστών" (mobile computers) που σε άλλους επιχειρηματικούς τομείς έχει εδώ και χρόνια αξιοποιηθεί, υπήρξε για τον χώρο της υγείας μια έννοια παρεξηγημένη αφού δεν υπήρξαν μέχρι πρότινος ουσιαστικές εφαρμογές των τεχνολογιών αυτών. Με την υλοποίηση ασύρματων δικτύων ευρείας ζώνης (Wireless Local Area Networks - WLANs) εντός των νοσοκομείων αλλά και την προσαρμογή μεγάλου αριθμού Νοσοκομειακών Πληροφοριακών Συστημάτων ώστε να μπορούν να χρησιμοποιηθούν από φορητές υπολογιστικές συσκευές PDA, δημιουργεί νέα δεδομένα στην αυτοματοποίηση διαδικασιών, στο περιορισμό του κόστους και κυρίως στην αναβάθμιση της ποιότητας των παρεχόμενων υπηρεσιών (κυρίως μέσω της σημαντικής μείωσης των ιατρικών σφαλμάτων). Χαρακτηριστικά παραδείγματα της αξιοποίησης των παραπάνω τεχνολογιών αποτελούν η παραγγελιοδοσία εργαστηριακών εξετάσεων και φαρμάκων στο χώρο παροχής υπηρεσιών (point-of-care), η τηλεπαρακολούθηση ζωτικών βιοσημάτων, η ορθή αναγνώριση (ταυτοποίηση) ασθενών, κλπ.

1.2 Η ανάγκη για την ύπαρξη του συστήματος

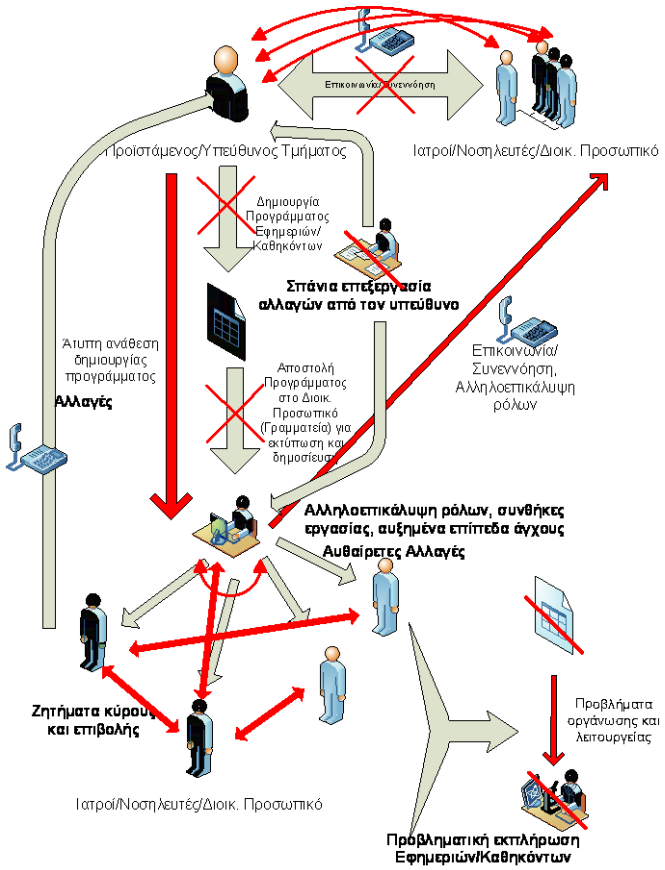
Η ανάγκη για την ύπαρξη του συγκεκριμένου συστήματος προκύπτει από την ιδιαιτερότητα των Νοσοκομειακών Μονάδων για ακριβή προγραμματισμό και συντονισμό του συνόλου των εργαζομένων και των οποιονδήποτε διαδικασιών που συντελούνται. Για το λόγο αυτό υπάρχουν προγράμματα που πρέπει να τηρούνται πιστά από τους εργαζομένους.

Οι τρείς μεγαλύτερες ομάδες εργαζομένων σε κάθε Νοσοκομειακή Μονάδα είναι το ιατρικό, το νοσηλευτικό και το διοικητικό προσωπικό. Για κάθε μία από αυτές υπάρχουν ξεχωριστά προγράμματα (εφημεριών, ιατρείων, βαρδιών, κ.α.) και υποχρεώσεις με ένα κοινό στόχο, την οργάνωση και το συντονισμό του προσωπικού για την ομαλή και εύρυθμη λειτουργία της Νοσοκομειακής Μονάδας.



Εικόνα 1: Εύρυθμη λειτουργία διαχείρισης εφημεριών και γενικών καθηκόντων

Όπως προαναφέραμε, οι Νοσοκομειακές Μονάδες είναι ένα πολύπλοκο οργανωτικό εργασιακό περιβάλλον στο οποίο κύρια συστατικά του είναι η αποτελεσματική επικοινωνία μεταξύ των εργαζομένων και η πειθαρχία στα προγράμματα εφημεριών/καθηκόντων. Στο τρέχων σύστημα διαχείρισης προσωπικού και γενικών καθηκόντων που λειτουργεί στα νοσοκομεία, οι ενέργειες του συντονισμού του προσωπικού, της διαμοίρασης, τροποποίησης και έγκρισης των επίσημων προγραμμάτων, αποτελούν εξαιρετικά χρονοβόρες και γραφειοκρατικές διαδικασίες. Στο ήδη βεβαρημένο και ευάλωτο σύστημα διαχείρισης προσωπικού προστίθενται οι άτυπες αλλαγές των προγραμμάτων εφημεριών και καθηκόντων, η άρνηση ευθυνών, τα προβλήματα επικοινωνίας με το προσωπικό, η αλληλοεπικάλυψη ρόλων, κ.α. Αποτέλεσμα είναι η έλλειψη συντονισμού και οργάνωσης του προσωπικού που οδηγούν σε προβλήματα παραγωγικότητας, αποτελεσματικότητας και κατ' επέκταση λειτουργικότητας των νοσοκομειακών μονάδων. Στο σημείο αυτό θα πρέπει να αναφερθούμε και στην εκδήλωση συγκρούσεων που προκύπτουν μεταξύ συναδέλφων κατά την διάρκεια εκτέλεσης των παραπάνω διαδικασιών, γεγονός που σπάνια λαμβάνεται υπόψη και έχει εξίσου σημαντικές επιπτώσεις στην λειτουργικότητα των νοσοκομειακών μονάδων.



Εικόνα 2: Προβληματική διαχείριση προγραμμάτων εφημεριών και γενικών καθηκόντων

1.3 Αντικείμενο πτυχιακής εργασίας

Στόχος της πτυχιακής εργασίας είναι η εκπαίδευση, έρευνα και εμβάθυνση σε τεχνολογίες διαδικτύου που αφορούν:

- Εφαρμογές για φορητές συσκευές
- Εφαρμογές Διαδικτύου
- Υπηρεσίες Διαδικτύου (XML Web Services)

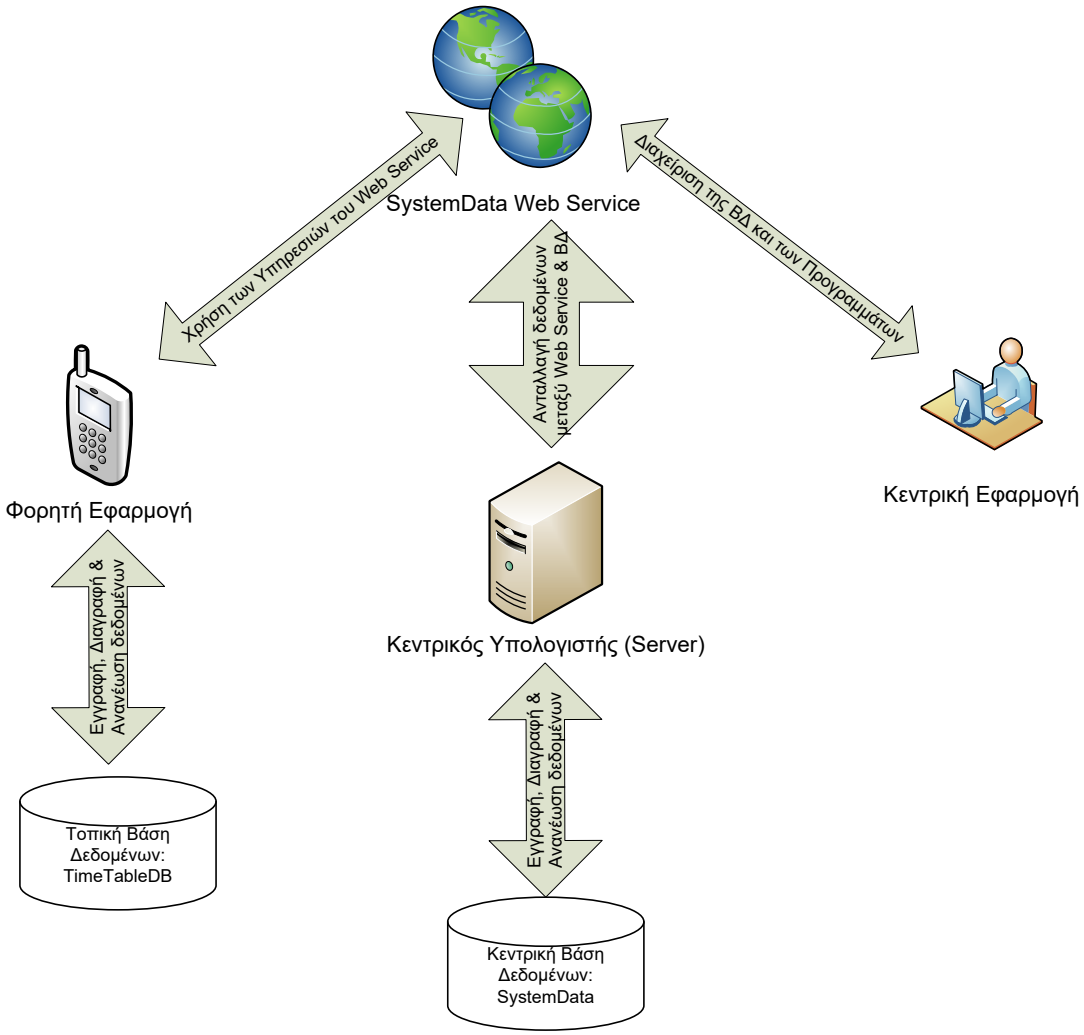
Στα πλαίσια της ανάγκης διεύρυνσης της λειτουργικότητας των φορητών συσκευών και των υπηρεσιών που προσφέρουν στους χώρους παροχής υγείας ώστε να ανταποκρίνονται στις απαιτήσεις των χρηστών, αντικείμενο της παρούσας πτυχιακής εργασίας είναι ο σχεδιασμός και υλοποίηση ενός συστήματος για τη βελτιστοποίηση της διαδικασίας ανάθεσης, διαμοιρασμού και τροποποίησης του προγράμματος των εφημεριών και περαιτέρω ιατρικών καθηκόντων στις μονάδες παροχής υγείας.

Το σύστημα περιλαμβάνει την ανάπτυξη εφαρμογής που υποστηρίζεται από κινητές συσκευές με λειτουργικό σύστημα Windows Mobile 6.5, την ανάπτυξη εφαρμογής Web Service για την ασύρματη επικοινωνία των φορητών συσκευών με τις επιμέρους εφαρμογές του συστήματος και την ανάπτυξη Web Based εφαρμογής με το όνομα «Κεντρική Εφαρμογή (Work Station)» η οποία διαχειρίζεται τα δεδομένα της Κεντρικής Βάσης Δεδομένων του συστήματος η οποία βρίσκεται σε έναν κεντρικό υπολογιστή (server).

Η φορητή εφαρμογή προσφέρει στο χρήστη την δυνατότητα ανάγνωσης, επεξεργασίας, αλλαγής και αποστολής των καθηκόντων προς τον κεντρικό υπολογιστή (server), καθώς και τη δυνατότητα διαπραγμάτευσης των καθηκόντων και των εφημεριών με άλλους χρήστες του συστήματος που διαθέτουν την εφαρμογή. Επομένως, μέσω αυτής της εφαρμογής πραγματοποιείται μια δυναμική διαχείριση του προσωπικού της νοσοκομειακής μονάδας από τους ίδιους τους χρήστες της εφαρμογής.

Η κεντρική εφαρμογή, έχει το ρόλο του συντονιστή, αναλαμβάνει την επεξεργασία των μηνυμάτων και των αιτήσεων των χρηστών για τροποποιήσεις των προγραμμάτων ή άλλου είδους έκτακτων ανακοινώσεων, προκειμένου στη συνέχεια να κατευθυνθεί προς την οδό έγκρισης από τους εκάστοτε υπευθύνους, επιταχύνοντας με αυτόν τον τρόπο την όλη διαδικασία καθώς και τον περιορισμό της αλληλοεπικάλυψης ρόλων και αρμοδιοτήτων λόγω προβληματικού καθορισμού καθηκόντων.

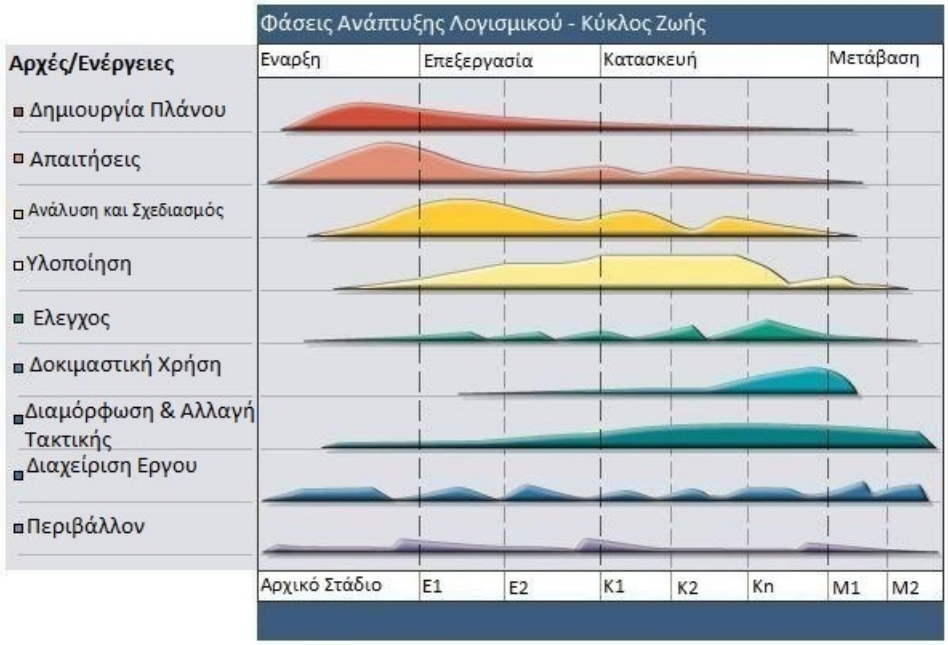
Ακόμη, η λειτουργία των υπηρεσιών του συστήματος βασίζεται στην τεχνολογία των υπηρεσιών Web services, οι οποίες επικοινωνούν με τη mobile και με την κεντρική εφαρμογή. Το πλεονέκτημα που προσφέρουν οι υπηρεσιών διαδικτύου είναι η επαναχρησιμοποίησή τους, καθώς είναι δυνατή η πρόσβαση στις πληροφορίες που παρέχουν και έχουν δημοσιευθεί σε οποιοδήποτε μέρος του διαδικτύου και από άλλες εφαρμογές.



Εικόνα 3: Λογικό Διάγραμμα Συστήματος

1.4 Οργάνωση της πτυχιακής εργασίας

Η οργάνωση της πτυχιακής εργασίας συμπίπτει με την εξελικτική πορεία του πρωτοκόλλου ανάπτυξης λογισμικού Rational Unified Process® (RUP®) της εταιρίας IBM [19]. Το RUP® είναι ένα σύνολο φιλοσοφιών και αρχών για την επιτυχή ανάπτυξη λογισμικού. Ο λόγος για τον οποίο επιλέξαμε το συγκεκριμένο πρωτόκολλο είναι επειδή εξασφαλίζει την παραγωγή υψηλής ποιότητας λογισμικού το οποίο ικανοποιεί τις ανάγκες των τελικών χρηστών του. Επίσης καθοδηγεί την διαδικασία ανάπτυξής του με οδηγίες, πρότυπα και συμβουλές για την χρήση των καταλληλότερων εργαλείων για όλες τις κρίσιμες δραστηριότητες του κύκλου ζωής ενός λογισμικού.



Εικόνα 4: Κύκλος ζωής δημιουργίας λογισμικού σύμφωνα με το πρώτυπο RUP

Στη συνέχεια δίνονται τα κεφάλαια που περιέχει η διπλωματική και γίνεται μια σύντομη περιγραφή των περιεχομένων του. Η σειρά και το περιεχόμενο των κεφαλαίων ακολουθούν τις φάσεις και τα στάδια ανάπτυξης λογισμικού του πρωτοκόλλου Rational Unified Process® (RUP®).

1.4.1 Τα κεφάλαια της πτυχιακής εργασίας και το περιεχόμενό τους

Στο **Κεφάλαιο 1** και κατά την φάση έναρξης της εργασίας στόχος ήταν να διαπιστώσουμε ότι το σύστημά μας είναι εφικτό να υλοποιηθεί και κυρίως ότι είναι χρήσιμο. Προκειμένου να διαπιστώσουμε τα παραπάνω, έπρεπε να γνωρίζουμε το πρόβλημα που θέλουμε να επιλύσουμε και στο οποίο θα στηρίξουμε την δημιουργία του συστήματός μας.

Το **Κεφάλαιο 2** αντιστοιχεί στην φάση της «Επεξεργασίας» σύμφωνα με το RUP. Κατά την φάση της επεξεργασίας, δώσαμε έμφαση στην ανάλυση του συστήματός μας. Συλλέξαμε πληροφορίες για το πρόβλημα που λύνουμε και μελετήσαμε το περιβάλλον του προβλήματος. Στόχος είναι να ορίσουμε με ακρίβεια τις απαιτήσεις του συστήματός μας, δηλαδή το τί ακριβώς θέλουμε να κάνει ή διαφορετικά ποιές από τις απαιτήσεις/ανάγκες των χρηστών θα εκπληρώνει.

Στην συνέχεια της φάσης αυτής ορίσαμε προτεραιότητες στις απαιτήσεις που έχουμε θέσει, προκειμένου να κατανοήσουμε ποιες είναι οι πιο σημαντικές και κατ' επέκταση που

πρέπει να δώσουμε μεγαλύτερη έμφαση. Πιο συγκεκριμένα, κατά τη διαδικασία της ανάλυσης έπρεπε:

- Να πραγματοποιηθεί μελέτη προκειμένου να διαπιστώσουμε τον τρόπο λειτουργίας του τρέχοντος συστήματος διαχείρισης εφημεριών/καθηκόντων που υπάρχει στις νοσοκομειακές μονάδες
- Να εντοπίσουμε τα προβλήματα και τις απαιτήσεις του τρέχοντος συστήματος
- Αφού εντοπισθούν τα προβλήματα του τρέχοντος συστήματος με ακρίβεια, να γίνουν σαφείς οι προσδοκίες για το νέο σύστημα διαχείρισης εφημεριών/καθηκόντων (τί περιμένουμε να κάνει)

Κατά τη διαδικασία του σχεδιασμού εκμεταλλευόμενοι τη γνώση από τη διαδικασία της ανάλυσης έπρεπε:

- Να πραγματοποιηθεί η μετάβαση από τον χώρο των προβλημάτων στον χώρο των λύσεων
- Να ορίσουμε τις απαιτήσεις των χρηστών
- Να ορίσουμε τις απαιτήσεις του συστήματος
- Να επικεντρωθούμε στον χώρο των λύσεων

Στο **Κεφάλαιο 3**, με σαφή πλέον επίγνωση του προβλήματος και του περιβάλλοντός του, αναλογιστήκαμε ποιός είναι ο πιο αποδοτικός τρόπος επίλυσής του και αν τα εργαλεία και οι τεχνολογίες που έχουμε στην διάθεσή μας, όντως θα οδηγήσουν στο επιθυμητό αποτέλεσμα. Προκειμένου να καταλήξουμε σε συμπεράσματα, χρειάστηκε να πειραματιστούμε με διάφορα εργαλεία και τρόπους επίλυσης του προβλήματος.

Στο **Κεφάλαιο 4** ορίζουμε λεπτομερώς:

- Τα επιμέρους τμήματα του συστήματος και τον τρόπο επικοινωνίας μεταξύ τους
- Τον τρόπο λειτουργίας του συστήματος από το προσωπικό της νοσοκομειακής μονάδας
- Τον τρόπο λειτουργίας του συστήματος εντός και εκτός των εγκαταστάσεων της νοσοκομειακής μονάδας
- Τον τρόπο αποθήκευσης όλων των απαραίτητων δεδομένων για την λειτουργία του συστήματος
- Τον τρόπο διασφάλισης της σωστής λειτουργίας του συστήματος
- Τον τρόπο διασφάλισης της μη απώλειας δεδομένων και της προστασίας τους

Στο **Κεφάλαιο 5** παρατίθεται αναλυτικά και διεξοδικά ο τρόπος υλοποίησης του συστήματος προκειμένου να μπορεί ο οποιοσδήποτε να επιχειρήσει αντίστοιχο εγχείρημα ή να λύσει απορίες που σχετίζονται με τον χώρο των windows mobile εφαρμογών και των Web Services.

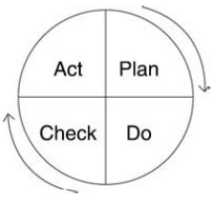
1.5 Αρχές της Ποιότητας Λογισμικού που χρησιμοποιήθηκαν για τη δημιουργία του συστήματος

Όλο και περισσότερες υπηρεσίες στηρίζονται σήμερα σε εφαρμογές λογισμικού. Αυτή η εξάρτηση απαιτεί το λογισμικό να λειτουργεί σωστά για πολύ καιρό (μακροπρόθεσμα) και να είναι εύκολο στη χρήση.

Η ποιότητα των εφαρμογών, πρέπει να τηρείται από την αρχή του κύκλου ζωής – ανάπτυξής τους, προκειμένου να αποφευχθούν ατέλειες που διαιωνίζονται κατά την πρόοδο ανάπτυξης του συστήματός μας.

Με την χρήση τεχνικών εξασφάλισης ποιότητας που τηρήσαμε από την αρχή της ανάπτυξης του συστήματός μας, μπορέσαμε να δημιουργήσουμε λογισμικό υψηλής ποιότητας που ανταποκρίνεται στις απαιτήσεις κάθε χρήστη (do the right things) χωρίς λάθη (do the things right) τα οποία μπορεί να αποβούν ζημιογόνα και μοιραία. Οι δραστηριότητες εξασφάλισης ποιότητας σχεδιάστηκαν και υλοποιήθηκαν αυστηρά σε συνεργασία με ιατρούς, νοσηλευτές, διοικητικό προσωπικό, διευθυντές προσωπικού και τους υπεύθυνους του πληροφορικού συστήματος του νοσοκομείου/κλινικής. Υποβάλλοντας το σύστημά μας σε συνεχείς δοκιμασίες για τον εντοπισμό σφαλμάτων δράσαμε αποφασιστικά για την πρόωρη επίλυσή τους. Μερικά από τα εργαλεία εξασφάλισης ποιότητας που χρησιμοποιήσαμε είναι τα εξής:

- Με την χρήση των χαρακτηριστικών ποιότητας του RUP® εξασφαλίσαμε για τον χρήστη:



  - Τη Λειτουργικότητα
  - Την Αξιοπιστία
  - Την Χρηστικότητα
  - Την Αποδοτικότητα
  - Και την ορθότητα (χωρίς λάθη) του λογισμικού

- Σχεδιασμός των δοκιμών (test) που υποβάλαμε για κάθε μέρος της εργασίας (unit testing) σύμφωνα με τους στόχους που είχαν οριστεί. Υλοποίηση των tests με πραγματικά σενάρια για την καταγραφή της συμπεριφοράς του συστήματος ακόμα και σε συνθήκες έντονης κινητικότητας της υγειονομικής μονάδας.

- Συγκεκριμένα για την υλοποίηση του κώδικα ακολουθήσαμε την μέθοδο ανάπτυξης (κώδικα) οδηγούμενη από test (Test Driven Development).

2

Ανάλυση Απαιτήσεων Συστήματος

Σύμφωνα με τον κύκλο ζωής του συστήματός μας, κατά την φάση της επεξεργασίας δίνουμε έμφαση στην ανάλυση του προβλήματος και έπειτα στον σχεδιασμό του συστήματος ο οποίος βασίζεται στη γνώση που προέρχεται από την διαδικασία της ανάλυσης. Σκοπός είναι να συλλέξουμε πληροφορίες για το πρόβλημα, την φύση του προβλήματος (π.χ. πρόβλημα οργάνωσης, πρόβλημα επικοινωνίας κ.τ.λ.) και το περιβάλλον του. Με τον τρόπο αυτό θα οδηγηθούμε στον ακριβή ορισμό των απαιτήσεων του συστήματός μας.

2.1 Η διαδικασία της ανάλυσης του περιβάλλοντος

Οι τρόποι που χρησιμοποιήσαμε προκειμένου να συλλέξουμε τις απαραίτητες πληροφορίες είναι:

- Χρήση ερωτηματολογίων, απευθυνόμενα στα άτομα που συμμετέχουν στο πρόβλημα [Παράρτημα Ι],
- τη διαδικασία των συνεντεύξεων με μελλοντικούς χρήστες του συστήματός μας,
- έρευνα για τυχών παρόμοιες εφαρμογές ή συστήματα που επιλύουν το ίδιο πρόβλημα [5]-[14].

2.1.1 Έρευνα σε Νοσοκομεία

Ο καλύτερος τρόπος για την ανάλυση του προβλήματος και την συγκέντρωση στοιχείων γύρω από αυτό είναι η διεξαγωγή έρευνας στο φυσικό περιβάλλον του προβλήματος, δηλαδή τα νοσοκομεία.

Με στόχο την δημιουργία ενός αποτελεσματικού συστήματος διαχείρισης εφημεριών και καθηκόντων, πραγματοποιήθηκε στατιστική έρευνα στα νοσοκομεία Α.Χ.Ε.Π.Α. - Πανεπιστημιακό Γενικό Νοσοκομείο Θεσσαλονίκης και ΓΕΝΙΚΟ ΣΤΡΑΤΙΩΤΙΚΟ ΝΟΣΟΚΟΜΕΙΟ ΑΘΗΝΩΝ (401 ΓΣΝΑ).

Σκοπός της έρευνας είναι:

- Η καταγραφή της τρέχουσας διαδικασίας βήμα προς βήμα της δημιουργίας, ανανέωσης και αλλαγής των προγραμμάτων εφημεριών και καθηκόντων.

- Ο εντοπισμός των εμπλεκομένων υπαλλήλων και η καταγραφή του ρόλου τους στη συνολική διαδικασία.

- Ο εντοπισμός και η καταγραφή των προβλημάτων, εάν υπάρχουν.

- Η καταγραφή των εμπειριών και των απόψεων για την τρέχουσα διαδικασία, από τους υπαλλήλους της νοσοκομειακής μονάδας

Τα στοιχεία και οι πληροφορίες που συλλέχθηκαν από την έρευνα, καθώς και η συνεργασία με τους υπαλλήλους των νοσοκομειακών μονάδων, βοήθησαν σε πρώτο στάδιο στην απόφαση για την δημιουργία του συγκεκριμένου συστήματος διαχείρισης εφημεριών και καθηκόντων και σε δεύτερο στάδιο, στο να κατανοήσουμε την συγκεκριμένη διαδικασία και να αποκτήσουμε μια πλήρη και εμπεριστατωμένη άποψη της τρέχουσα κατάστασης.

Με την δημιουργία κατάλληλων ερωτηματολογίων για κάθε ομάδα χρηστών και με συνεντεύξεις που πραγματοποιήθηκαν με εργαζομένους στον χώρο της υγείας, καθορίσαμε τα προβλήματα του τρέχοντος συστήματος. Βασιζόμενοι στα προβλήματα που προκύπτουν και στις απόψεις των εργαζομένων που εμφανίζονται στα αποτελέσματα των ερωτηματολογίων, οδηγούμαστε στις απαιτήσεις των χρηστών και στις απαιτήσεις του συστήματός μας.

Επιπλέον, πραγματοποιήθηκε η μοντελοποίηση με UML διαγράμματα, των σημαντικότερων απαιτήσεων που έχουν οι χρήστες από το σύστημά μας καθώς και η μοντελοποίηση των σημαντικότερων απαιτήσεων του ίδιου του συστήματος. Τίποτα όμως από τα παραπάνω δεν έχει αξία εάν ο χρήστης δεν έχει εύκολη πρόσβαση στις λειτουργίες που δίνουν λύση στα προβλήματά του. Για το λόγο αυτό δώσαμε μεγάλη προσοχή στο σχεδιασμό και τις λειτουργίες των διεπαφών ανθρώπου-Η/Υ.

2.1.1.1 Τρέχουσα διαδικασία ανανέωσης και αλλαγής του επίσημου Προγράμματος

Τυπικά και νομικά όλοι οι γιατροί του Ε.Σ.Υ. θα πρέπει να υπακούν στον νόμο ΥΠ'ΑΡΙΘ. 3754 περί Ρύθμιση Όρων Απασχόλησης. Οι διατάξεις του νόμου αυτού ορίζουν σε πρώτη φάση το ωράριο εργασίας (βάρδιας) και εφημεριών σύμφωνα με την ειδικότητα και την βαθμίδα του κάθε ιατρού.

Η δημιουργία και οργάνωση των εφημεριών γίνεται από τους υπεύθυνους του κάθε τμήματος ή κλινικής και τους υπεύθυνους ανθρώπινου δυναμικού. Η διαδικασία δημιουργίας του προγράμματος εφημεριών και καθηκόντων στις περισσότερες περιπτώσεις πραγματοποιείται χειρόγραφα, ενώ στην μειοψηφία των νοσοκομείων που χρησιμοποιούν Πληροφοριακό Σύστημα, πραγματοποιείται σε αρχεία του MS Excel ή της MS Access και στην συνέχεια τυπώνεται για να ανακοινωθεί.

Σε περίπτωση διαφωνίας με το πρόγραμμα, θα πρέπει να υπάρξει επικοινωνία του/των ενδιαφερόμενου/νων με τον υπεύθυνο του τμήματος προκείμενου να προβούν στις κατάλληλες αλλαγές - διαδικασία που δημιουργεί περαιτέρω προβλήματα. Κάποια από αυτά είναι η ασυμφωνία εργαζομένων με το νέο πρόγραμμα και η επιθυμία όλων των υπολοίπων εμπλεκομένων για την διαμόρφωση των καθηκόντων τους σύμφωνα με τις προσωπικές τους ανάγκες.

2.1.1.1.1 Ρόλοι των Χρηστών που συμμετέχουν στην τρέχουσα διαδικασία ανανέωσης και αλλαγής του επίσημου Προγράμματος

Ρόλος Προϊσταμένων: Είναι αρμόδιοι για την δημιουργία, τον έλεγχο, την εγκυρότητα και την έγκριση των προγραμμάτων εφημεριών και καθηκόντων των χρηστών.

Ρόλος διοικητικών υπαλλήλων που ασχολούνται με το πρόγραμμα εφημεριών-καθηκόντων: Είναι οι ομάδα που ασχολείται με την δημοσίευση των προγραμμάτων στους υπόλοιπους χρήστες. Επίσης, αρκετές φορές άτυπα συμμετέχουν στην δημιουργία και στην διαχείριση των αλλαγών του προγράμματος.

Ρόλος Υπαλλήλων (ιατροί, νοσηλευτές): Θα πρέπει να τηρούν τα προγράμματα εφημεριών καθηκόντων. Συχνά έρχονται σε συνεννόηση με τους προϊσταμένους ή με το διοικητικό προσωπικό προκειμένου να προβούν σε αλλαγές του προγράμματος για προσωπικούς ή άλλους λόγους. Εάν δεν το επιτύχουν προσπαθούν άτυπα να εξυπηρετηθούν από συναδέλφους τους, οι οποίοι καλύπτουν πιθανά κενά ή έρχονται σε συμφωνία ανταλλαγής εφημεριών-καθηκόντων.

2.1.1.2 Προβλήματα της τρέχουσας διαδικασία ανανέωσης και αλλαγής του επίσημου προγράμματος

Τα προβλήματα της τρέχουσας διαδικασίας διαφοροποιούνται σύμφωνα με το περιβάλλον στο οποίο βρίσκετε κάθε ομάδα χρήσης του συστήματος. Όπως έχουμε ήδη αναφέρει οι συμμετέχοντες στην διαδικασία χωρίζονται στις ομάδες των προϊσταμένων, της γραμματειακής υποστήριξης (συμπεριλαμβάνεται και το τμήμα πληροφορικής) και στο προσωπικό της νοσοκομειακής μονάδας (ιατροί και νοσηλευτές). Θα αναφέρουμε τα προβλήματα του τρέχοντος συστήματος αρχίζοντας από την ομάδα των προϊσταμένων και των υπευθύνων για την δημιουργία και έκδοση των προγραμμάτων εφημεριών/καθηκόντων.

Η συγκεκριμένη ομάδα είναι υπεύθυνη για την εύρυθμη λειτουργία της υγειονομικής μονάδας με συνέπεια να ασχολούνται με την επίλυση πολλών άλλων προβλημάτων που μπορούν να προκύψουν. Κατά την διαδικασία έκδοσης του προγράμματος στο τρέχων σύστημα, κάποιος προϊστάμενος θα πρέπει να έρθει σε επαφή με όλα τα μέλη του προσωπικού προκειμένου να διαπιστώσει ποιος είναι διαθέσιμος και πότε. Μια διαδικασία που απαιτεί **χρόνο** μέχρι να πραγματοποιηθεί η συνεννόηση με όλους τους υπαλλήλους, εξαιτίας του φόρτου εργασίας και των πολύπλοκων προγραμμάτων. Με αποτέλεσμα **να καθυστερεί** η έκδοση του προγράμματος εφημεριών και να **δημιουργείται εκνευρισμός**.

Επίσης λόγο προβλημάτων που μπορεί να προκύψουν στο εργατικό προσωπικό, γίνονται πολλές αλλαγές στο υπάρχων πρόγραμμα οι οποίες πρέπει να εγκριθούν από τους προϊσταμένους. Άρα, **δεν υπάρχει αμεσότητα** στην ανακοίνωση του προγράμματος εφημεριών/καθηκόντων και δεν αποτελεί εύκολη διαδικασία η αλλαγή του προγράμματος εφημεριών.

Το προσωπικό είναι δύσκολο να έρθει σε επικοινωνία με τον προϊστάμενο για μεγάλο πλήθος αλλαγών στο πρόγραμμα εφημεριών. Το γεγονός αυτό επιβαρύνει το προσωπικό της γραμματείας, το οποίο είναι υπεύθυνο για θέματα εύρεσης και επικοινωνίας ενώ στις περισσότερες περιπτώσεις διαδραματίζει τον ρόλο του μεσάζοντα στην επικοινωνία ιατρού-υπεύθυνου προσωπικού.

Τα άτομα των χειριστών Η/Υ και της γραμματείας δεν έχουν δικαίωμα να αλλάξουν τίποτα στο πρόγραμμα αν δεν εγκριθεί από τον προϊστάμενο. Η μόνη δικαιοδοσία που έχουν είναι να αναρτούν καθημερινά (αν υπάρχει) τη νέα έκδοση του προγράμματος εφημεριών και να «επιβλέπουν» σε ρόλο βοηθού οργανωτή την κατάσταση λειτουργίας της μονάδας.

Έχουν μεγάλο φόρτο εργασία αφού πρέπει να βρίσκουν και να παρουσιάσουν όλα τα προγράμματα των ιατρών στον προϊστάμενο ώστε να μπορεί συνέχεια να δημιουργήσει το πρόγραμμα.

Το προσωπικό της νοσοκομειακής μονάδας δίνει τον καλύτερο του εαυτό για την ομαλή λειτουργία μιας μονάδας, πολλές φορές όμως συμβαίνουν έκτακτα περιστατικά, προβλήματα εργασίας ή προσωπικά, τα οποία θέτουν εκτός προγράμματος ιατρούς και νοσηλευτές. Σε μεγάλες νοσοκομειακές μονάδες τα προβλήματα αυτά και οι απουσίες υπαλλήλων συσσωρεύονται και προκαλούν ένα ντόμινο προβλημάτων στην οργάνωση και την ομαλή λειτουργία της νοσοκομειακής μονάδας. Το ίδιο συμβαίνει στην περίπτωση που ορισμένοι υπάλληλοι έχουν ρεπό ή βρίσκονται σε άδεια.

Οι καταστάσεις οι οποίες αναφέραμε παραπάνω καθώς και αυτές που ακολουθούν είναι αιτίες συγκρούσεων στον χώρο των νοσοκομείων [17] οι οποίες εμφανίζονται και κατά την διαδικασία δημιουργίας, ανανέωσης και αλλαγής των προγραμμάτων εφημεριών και καθηκόντων.

- Ζητήματα κύρους και επιβολής (Richardson 1991)
- Συνθήκες εργασίας-αυξημένα επίπεδα άγχους
- Η άρνηση ευθυνών
- Η αλληλοεπικάλυψη ρόλων και αρμοδιοτήτων λόγω προβληματικού καθορισμού καθηκόντων καθώς και
- τα διαφορετικά επίπεδα εκπαίδευσης. (Jameson 2003)
- Προκλητική συμπεριφορά απέναντι στην ηγεσία, διακρίσεις (Singleton, 1999)
- Πολύπλοκο οργανωτικό εργασιακό περιβάλλον (Μπουραντάς 2002)
- Η καταλυτική συμμετοχή στην αύξηση στερεοτύπων σχετικά με το επάγγελμα των νοσηλευτών και των ιατρών, η αλλαγή ρόλου των νοσηλευτών (Jacinta 2006)
- Περιορισμένοι πόροι (έλλειψη προσωπικού, οικονομικοί και υλικοτεχνικοί πόροι)
- Έλλειψη οργάνωσης και προβλήματα διοίκησης
- Διαφορές νοοτροπίας, θέσης, επιπέδου μόρφωσης στις διάφορες ιεραρχικές βαθμίδες

Τα είδη των συγκρούσεων μπορεί να είναι:

- Διαπροσωπικές συγκρούσεις
- Ομαδικές συγκρούσεις
- Ιεραρχικές συγκρούσεις (π.χ. αυταρχικό μάνατζμεντ)
- Λειτουργικές συγκρούσεις – περισσότερο συχνές στο χώρο του νοσοκομείου
- Συγκρούσεις επιτελικών – γραμμικών στελεχών
- Συγκρούσεις μεταξύ τυπικής και άτυπης οργάνωσης

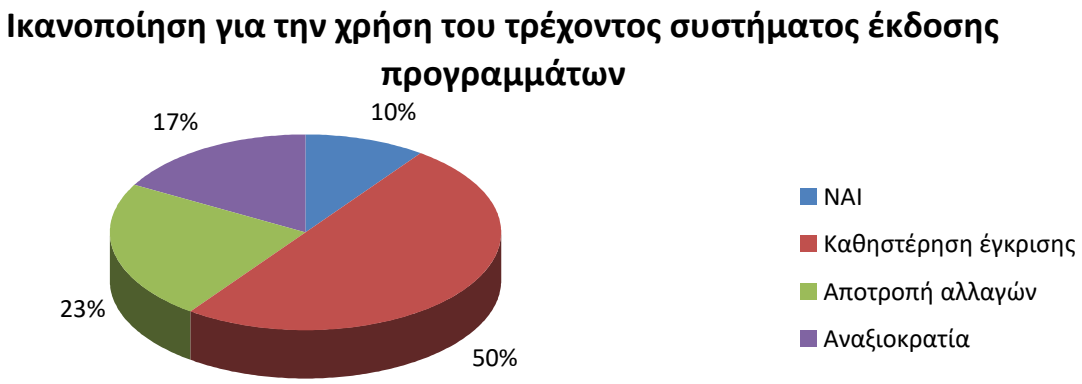
Η ύπαρξη συγκρούσεων σε μια ομάδα εργασίας, είναι ένας εξαιρετικά αρνητικός παράγοντας που υπονομεύει επί μονίμου βάσεως την παραγωγικότητα και την αποτελεσματικότητά της και κατ' επέκταση της νοσοκομειακής μονάδας.

2.1.1.3 Στατιστικά Αποτελέσματα ερωτηματολογίων

Προκειμένου να αποκτήσουμε την πλήρη εικόνα της τρέχουσας διαδικασίας και να βρούμε λύσεις στα προβλήματα, δημιουργήσαμε 3 διαφορετικά ερωτηματολόγια, 1 για κάθε ομάδα υπαλλήλων.

- Ερωτηματολόγιο που απευθύνεται σε όλους τους συμμετέχοντες της διαδικασίας που εξετάζουμε, ιατρούς, νοσηλευτικό προσωπικό, προϊσταμένους-διευθυντές και γραμματεία-χειριστές Η/Υ.
- Ερωτηματολόγιο που απευθύνεται αποκλειστικά στο προσωπικό της γραμματείας και τους χειριστές Η/Υ.
- Ερωτηματολόγιο που απευθύνεται αποκλειστικά στους προϊσταμένους, διευθυντές τμημάτων και υπευθύνους προσωπικού.

Τα ερωτηματολόγια ήταν ανώνυμα και η συμπλήρωσή τους προαιρετική εξαιτίας του φόρτου εργασίας που υπάρχει στις νοσοκομειακές μονάδες. Ο αριθμός των ερωτηματολογίων που δόθηκαν προς συμπλήρωση ήταν ανάλογος του πλήθους των ατόμων σε κάθε ομάδα. Έτσι μοιράστηκαν συνολικά 50 ερωτηματολόγια, 30 ερωτηματολόγια σε ιατρούς, νοσηλευτικό προσωπικό, προϊσταμένους-διευθυντές και γραμματεία-χειριστές Η/Υ, 12 ερωτηματολόγια για το προσωπικό της γραμματείας και τους χειριστές Η/Υ, 8 ερωτηματολόγια σε προϊσταμένους. Στη συνεχεία παρουσιάζουμε ορισμένα ενδεικτικά αποτελέσματα από τις απαντήσεις των ερωτηθέντων:



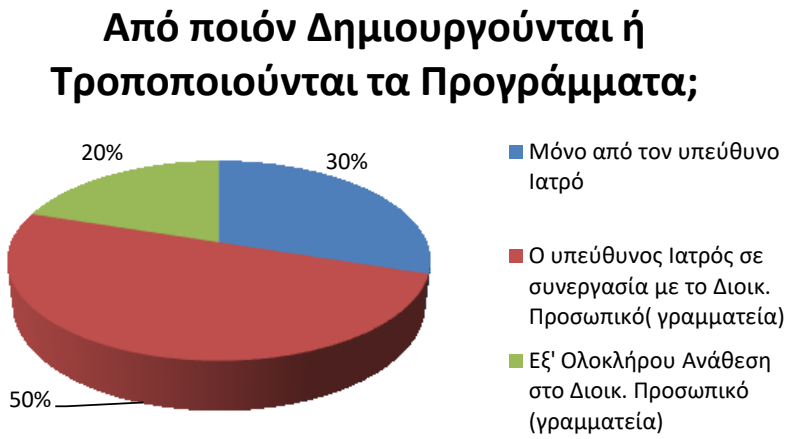
Στατιστικά Στοιχεία 1: Ικανοποίηση των χρηστών από το τρέχων σύστημα διαχείρισης και έκδοσης προγραμμάτων

Στην ερώτηση «Είστε ευχαριστημένος από τον τρόπο διαμόρφωσης, έκδοσης και ανακοίνωσης του προγράμματος εφημεριών-καθηκόντων;» μεγάλο ποσοστό ανέφερε ότι δεν είναι ευχαριστημένο και οι λόγοι είναι: καθυστέρηση έγκρισης, αποτροπή αλλαγών, αναξιοκρατία.

Μεγάλο Ποσοστό στην ερώτηση «Νομίζετε ότι θα σας διευκόλυνε εάν η διαδικασία της έκδοσης των προγραμμάτων πραγματοποιούταν με αυτοματοποιημένο τρόπο και ηλεκτρονικά (π.χ. «το πρόγραμμα να εμφανίζεται αυτόματα στην οθόνη σας και η έγκριση να γίνεται με το πάτημα ενός κουμπιού»);» Η απάντηση ήταν ΝΑΙ.

Στατιστικά Στοιχεία 2: Αποτελέσματα ικανοποίησης των χρηστών στην περίπτωση του αυτοματοποιημένου τρόπου διαχείρισης και έκδοσης προγραμμάτων

Το μεγαλύτερο ποσοστό από τους προϊσταμένους δεν αναλαμβάνουν εξ' ολοκλήρου την δημιουργία του προγράμματος και εμπλέκονται διαφορετικά πρόσωπα για την εκπλήρωση της συγκεκριμένης διαδικασίας.

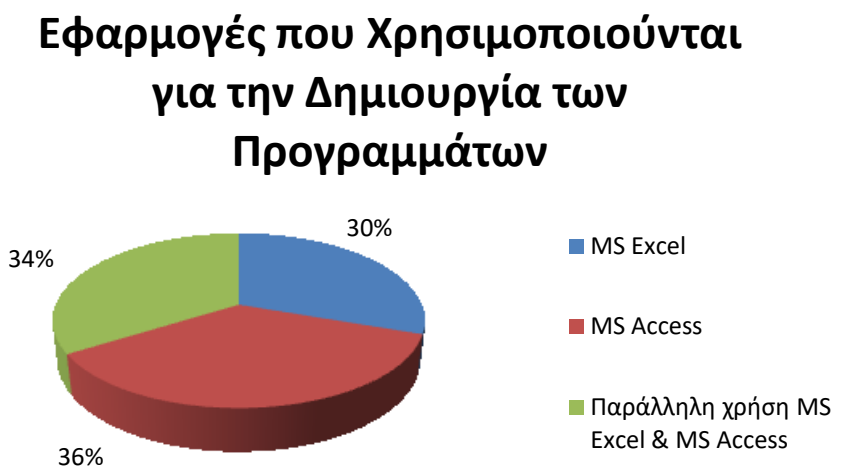


Στατιστικά Στοιχεία 3: Ποιοί δημιουργούν ή τροποποιούν τα προγράμματα εφημεριών/καθηκόντων

Μεγάλο ποσοστό από τα άτομα που αναμιγνύονται στο πρόγραμμα έχουν την δυνατότητα έγκρισης αλλαγών χωρίς να ρωτήσουν τους υπευθύνους.

Στατιστικά Στοιχεία 4: Ποσοστά αυθαίρετων πραγματοποιήσεων αλλαγών στα προγράμματα εφημεριών/καθηκόντων

Ρωτήσαμε την γραμματεία αν χρησιμοποιεί κάποια εμπορική εφαρμογή για τη δημιουργία του προγράμματος και μας απάντησαν ότι χρησιμοποιούν τα MS Excel και MS Access. Στις απαντήσεις περιλαμβάνονται και οι περιπτώσεις παράλληλης χρήσης των εφαρμογών.



Στατιστικά Στοιχεία 5: Εφαρμογές που χρησιμοποιούνται για την δημιουργία των προγραμμάτων

2.1.1.4 Λύσεις που παρέχει το σύστημα διαχείρισης εφημεριών-καθηκόντων

Το σύστημά μας θα μπορούσε να ενσωματωθεί σε ένα Ολοκληρωμένο Πληροφοριακό Σύστημα Υγείας (ΟΠΣΥ) ή να αποτελέσει μια ξεχωριστή, αυτόνομη υπηρεσία της νοσοκομειακής μονάδας.

- Άμεση ενημέρωση όλων των ενδιαφερομένων για αλλαγές και ανανεώσεις των προγραμμάτων
- Άμεση ειδοποίηση ιατρικού και νοσηλευτικού προσωπικού σε περιπτώσεις εκτάκτου ανάγκης
- Μείωση των καθηκόντων εργατικού δυναμικού - μείωση αλληλοεπικάλυψης ρόλων και καθηκόντων
- Ελαχιστοποίηση του χρόνου ενασχόλησης με την διαδικασία (δημιουργία, αλλαγή, έγκριση, ενημέρωση, ανακοίνωση) των προγραμμάτων εφημεριών-καθηκόντων.
- Εξάλειψη σφαλμάτων λόγο κακής συνεννόησης ή επικοινωνίας

- Ικανοποίηση των εργαζομένων μέσω της αποφυγής συμμετοχής στην διαδικασία υποβολής αλλαγών ή άλλων αιτήσεων

- Αμεσότητα στην επικοινωνία μεταξύ συναδέλφων χωρίς κόστος

- Ευκολία στη δημιουργία προγραμμάτων και ενημερώσεων και στην ανακοίνωσή τους

- Αποφυγή εντάσεων και συγκρούσεων μεταξύ των υπαλλήλων της νοσοκομειακής μονάδας

Όλα τα παραπάνω οδηγούν σε αναβάθμιση του επιπέδου οργάνωσης κάθε νοσοκομειακής μονάδας και τη διαμόρφωση θετικού κλίματος μεταξύ των συναδέλφων. Τα αποτελέσματα είναι η αύξηση της αποδοτικότητας και της αποτελεσματικότητας στο σύνολο του εργατικού δυναμικού (ιατροί, γραμματείς, υπεύθυνη τμημάτων, διευθυντές και χειριστές Η/Υ).

2.1.2 Έρευνα για τον εντοπισμό και την μελέτη εφαρμογών οι οποίες επικεντρώνονται στην διαχείριση προγραμμάτων και εργασιών σε νοσοκομειακές μονάδες

Η έρευνα που έγινε προκειμένου να εντοπίσουμε συστήματα ή υπηρεσίες παρόμοιες με τις δικές μας ήταν άκαρπη. Οι πλειονότητα των νοσοκομειακών εφαρμογών, ασχολούνται με λειτουργίες Ηλεκτρονικών Ιατρικών Φακέλων, αυτοματοποιημένης ηλεκτρονικής χρέωσης των ασθενών και της αποτελεσματικής διαχείρισης των ραντεβού των ασθενών. Με βάση τα παραπάνω, οι εφαρμογές που επικεντρώνονται στη διαχείριση των ωρολογίων προγραμμάτων των νοσοκομειακών υπαλλήλων είναι περιορισμένες και μπορούν να χωριστούν σε 2 μεγάλες κατηγορίες.

Την πρώτη κατηγορία αποτελούν εφαρμογές αυτόματης δημιουργίας ωρολογίων προγραμμάτων και αποτελεσματικής διαχείρισης του χρόνου όπως τα :

- ScheduFlow της εταιρίας Duoserve [5]
- Time Tracker της εταιρίας Asgard systems [6]

Την δεύτερη κατηγορία πλαισιώνουν κυρίως εφαρμογές (τύπου Practice Management) που συνδυάζουν τις εξής λειτουργίες:

- Την διαχείριση των προσωπικών ραντεβού των ιατρών με τους ασθενείς τους,
- Τις χρεώσεις των ασθενών σύμφωνα με τις υπηρεσίες που τους έχουν παρασχεθεί και σπανιότερα,
- Λειτουργίες μισθοδοσίας των υπαλλήλων της νοσοκομειακής μονάδας βάση του ωρολογίου προγράμματος των εφημεριών και διαφόρων καθηκόντων.

Παραδείγματα τέτοιων εφαρμογών είναι:

- Τα LytecMD, Lytec 2010 της εταιρίας Complete Medical Billing [7]
- Το CSC Appointment Scheduling [8]
- Τα προϊόντα Medical Scheduling της Claricode [9]

Και τις δυο παραπάνω ομάδες-οικογένειες εφαρμογών μπορούμε πλέον να τις βρούμε ενσωματωμένες στα σύγχρονα Ολοκληρωμένα Πληροφοριακά Σύστημα Υγείας. Το ίδιο συμβαίνει και στην περίπτωση εφαρμογών για φορητές συσκευές. Εφαρμογές για την δυναμική διαχείριση του προγράμματος των εφημεριών και των καθηκόντων για κάθε νοσοκομειακό υπάλληλο ξεχωριστά δεν υπάρχουν.

Υπάρχουν δύο εφαρμογές που ασχολούνται με την διαχείριση προσωπικού στον ελληνικό δημόσιο τομέα της υγείας χωρίς όμως να διαθέτουν κάποια εφαρμογή ή demo προς κατέβασμα (download) και χωρίς να αναφέρονται καθόλου σε φορητές συσκευές και διαχείριση σε πραγματικό χρόνο. Αυτές είναι οι:

- AppSoft, Ειδικές εφαρμογές για Δημόσιο-ΝΠΔΔ-ΔΕΚΟ, Διαχείριση προσωπικού ΝΠΔΔ [13] και
- Ειδική εφαρμογή Διαχείρισης Προσωπικού ΝΠΔΔ της εταιρίας Uniplan [14]

Εφαρμογές όπως το PatientKeeper [10] και το MedicMate [12] ειδικεύονται κυρίως στην διαχείριση των ιατρικών δεδομένων και τη χρέωση των ασθενών. Οι λειτουργίες διαχείρισης του προγράμματος απευθύνονται στην τακτοποίηση των ραντεβού που έχουν οι ιατροί με τους ασθενείς τους και την καταγραφή των προσωπικών τους ημερησίων προγραμμάτων.

2.1.3 Διαφοροποίηση της εφαρμογής μας από τις υπόλοιπες

Είναι γεγονός ότι όλες σχεδόν οι παραπάνω εφαρμογές έχουν σχεδιαστεί και αναπτυχθεί στο εξωτερικό και τις περισσότερες φορές στοχεύουν στην εξυπηρέτηση

ιδιωτικών ιατρών και κλινικών. Επίσης θα πρέπει να αναφέρουμε ότι καμία από τις υπάρχουσες εφαρμογές δημιουργίας νοσοκομειακών προγραμμάτων δεν πραγματοποιεί ούτε προβλέπει τη δυναμική διαχείριση των αλλαγών που μπορεί να προκύψουν στο πρόγραμμα, με αποτέλεσμα να μην υπάρχει ευελιξία και πιθανά λάθη να επιφέρουν ντόμινο αλλαγών.

Το σύστημά μας πάει ένα βήμα πιο μπροστά από τις ήδη υπάρχουσες εφαρμογές, χωρίς να προσπαθεί να τις επισκιάσει. Έχει σχεδιαστεί και δημιουργηθεί με βάση έρευνα που πραγματοποιήθηκε σε δημόσια νοσοκομεία και με βάση μελέτη πάνω στην οργάνωση των νοσοκομείων στην χώρα μας σε πρακτικό και νομικό επίπεδο. Το σύστημα της αυτόματης διαχείρισης των νοσοκομειακών καθηκόντων στοχεύει κατά κύριο λόγο στην εξάλειψη της χρονικής και γραφειοκρατικής καθυστέρησης που μέχρι στιγμής απαιτούν οι αλλαγές των προγραμμάτων εφημεριών και καθηκόντων (π.χ. βάρδιες), γεγονός που αποτελεί μέρος της οργάνωσης του Ε.Σ.Υ. Ο τρόπος με τον οποίον το σύστημά μας σκοπεύει να διευθετήσει το πρόβλημα αυτό είναι μέσα από τη μαζική επικοινωνία σε πραγματικό χρόνο μεταξύ των εργαζομένων της νοσοκομειακής μονάδας και μέσω της δυναμικής διαχείρισης και ενημέρωσης των αλλαγών που προκύπτουν.

2.2 Η διαδικασία σχεδιασμού του συστήματος Διαχείρισης Εφημεριών και Καθηκόντων

Τα αποτελέσματα των αναλύσεων και των ερευνών που προηγήθηκαν έχουν σκοπό τον ορισμό των απαιτήσεων των χρηστών αλλά και του συστήματός μας. Δηλαδή ορίζουμε το τι περιμένει ο χρήστης από το σύστημά μας να κάνει και τα μέσα με τα οποία το σύστημά μας θα πραγματοποιήσει τις ανάγκες των χρηστών. Ο ορισμός των απαιτήσεων αποτελεί μια επισκόπηση των δομών δεδομένων που βρίσκονται στο υπόβαθρο του συστήματος που δημιουργούμε.

2.2.1 Λειτουργικές Απαιτήσεις του συστήματος

Στις λειτουργικές απαιτήσεις αναφέρουμε το σύνολο εκείνων των λειτουργιών που αποδίδουν στο σύστημά μας την συμπεριφορά του. Οι λειτουργικές απαιτήσεις μπορεί να είναι υπολογισμοί, τεχνικές λεπτομέρειες, χειρισμός ή επεξεργασία στοιχείων και άλλες συγκεκριμένες λειτουργίες.

2.2.1.1 Απαιτήσεις Χρηστών Κεντρικής Εφαρμογής

- Δυνατότητα δημιουργίας προγράμματος εφημεριών-καθηκόντων για το προσωπικό της γραμματείας και των χειριστών Η/Υ μέσω της κεντρικής εφαρμογής (Work Station).

- Δυνατότητα του χρήστη να ανατρέχει σε παλαιότερα προγράμματα και εργασίες

- Δυνατότητα ανταλλαγής εφημεριών, καθηκόντων και γενικών προγραμματισμένων εργασιών μεταξύ των χρηστών του συστήματος

- Δυνατότητα αυτόματης ανανέωσης (ανακατασκευής) του προγράμματος εφημεριών σύμφωνα με τις πληροφορίες που λαμβάνει το σύστημα από τις φορητές συσκευές των υπαλλήλων

- Πλήρη γνωστοποίηση των ατόμων που υποβάλουν οποιασδήποτε μορφής αίτηση

- Ύπαρξη περιορισμών τους οποίους θα πρέπει να ακολουθεί κάθε πρόγραμμα εφημεριών, είτε κατά την δημιουργία του, είτε κατά την αυτόματη ανανέωσή του (π.χ. την Τετάρτη 25/9/2010 θα πρέπει υποχρεωτικά να είναι σε βάρδια τουλάχιστον 2 παθολόγοι) .

- Η εφαρμογή θα είναι σε θέση να εντοπίζει ασυμφωνίες στα πρόγραμμα εφημεριών, είτε κατά την δημιουργία του, είτε κατά την αυτόματη ανανέωσή του και να ενημερώνει με αντίστοιχα μηνύματα τους χρήστες του συστήματος.

- Στην εφαρμογή του κεντρικού υπολογιστή θα πραγματοποιούνται οι εγγραφές των χρηστών που θα χρησιμοποιούν το σύστημα αυτόματης ενημέρωσης και τροποποίησης εφημεριών.

- Το σύστημα θα χωρίζει τους χρήστες σε ομάδες σύμφωνα με την ειδικότητά τους, το τμήμα εργασίας τους και την βαθμίδα τους.

- Ο χρήστης της κεντρικής εφαρμογής θα έχει την δυνατότητα πλοήγησης στις λίστες των εγγεγραμμένων μελών χρηστών.

- Η κεντρική εφαρμογή θα είναι υπεύθυνη μόνο για τον έλεγχο και την αξιολόγηση αιτήσεων που αφορούν την αλλαγή στοιχείων της εργασίας (π.χ. ώρα έναρξης ή λήξης) και όχι για τις ανταλλαγές εργασιών μεταξύ των χρηστών.

- Οι αιτήσεις αλλαγής, μεταφοράς ή ακύρωσης των εργασιών που αποστέλλονται από τους χρήστες, θα γίνονται αυτόματα ορατές και έτοιμες για επεξεργασία από τον χρήστη της κεντρικής εφαρμογής

- Ο χρήστης της κεντρικής εφαρμογής θα έχει την δυνατότητα αξιολόγησης, έγκρισης ή και απόρριψης των αιτήσεων, αλλαγής ή μεταφοράς των εργασιών που αποστέλλουν οι χρήστες των φορητών εφαρμογών .

- Κατά την πλοήγηση στις λίστες των εγγεγραμμένων μελών του συστήματος, ο χρήστης της εφαρμογής θα μπορεί να επιλέξει όποιο άτομο θελήσει (ανάλογα με την ειδικότητά του) προκειμένου να τον εισάγει σε πόστο (εφημερία) του προγράμματος.

- Δυνατότητα αποκλεισμού μεμονωμένων ατόμων ή ομάδων ατόμων (ειδικότητες) για τους οποίους δε θα εισέρχονται στο σύστημα οι πληροφορίες που αφορούν αλλαγές στο πρόγραμμα εφημεριών.

- Ο χρήστης της εφαρμογής θα μπορεί να επιλέγει σε ποιά άτομα ή ομάδες ατόμων θα αποστέλλεται το πρόγραμμα εφημεριών και οι ανανεώσεις του.

- Εξαιτίας του σχεδιασμού και των περιορισμών της ΒΔ και της ίδιας της εφαρμογής, είναι δυνατός ο εντοπισμός όχι μόνο ασυμφωνιών που αφορούν ημ/νίες και ώρες, αλλά και προσώπων (ονομαστικά) και ειδικοτήτων.

- Η εφαρμογή θα μπορεί να εντοπίζει αυτόματα και να εμφανίζει τα άτομα του προσωπικού που βρίσκονται στους χώρους του νοσοκομείου.

- Μια ακόμη λειτουργία που θα παρέχει η εφαρμογή είναι η εμφάνιση της κατάστασης στην οποία βρίσκεται κάθε χρήστης (μαζί με τα βασικά στοιχεία ταυτοπροσωπίας του). Η κατάσταση (mode) αυτή θα μπορεί να αλλάζει αυτόματα σύμφωνα με τις πληροφορίες που περιέχουν τα προγράμματα εφημεριών. Οι καταστάσεις μπορεί να είναι: «εργάζεται», «σε ρεπό», «σε αναμονή», «σε άδεια» κτλ.

- Ο χρήστης της κεντρικής εφαρμογής θα έχει τη δυνατότητα να γράφει και να ανακοινώνει προς τα εγγεγραμμένα μέλη, μηνύματα και πληροφορίες με σκοπό την αρτιότερη λειτουργία της μονάδας.

- Η εφαρμογή θα έχει δυνατότητες επικοινωνίας με τις φορητές συσκευές κάθε μέλους ανεξαρτήτως της περιοχής όπου βρίσκεται (απαιτείται η σύνδεση της φορητής συσκευής με ασύρματο δίκτυο επικοινωνίας: wi-fi, 3G, HSPDA)

- Η έγκριση προγράμματος, σε περιπτώσεις όπου κρίνεται αναγκαία, να αποτελεί για τους προϊσταμένους απλή διαδικασία.

2.2.1.2 Απαιτήσεις Χρηστών Φορητής Εφαρμογής

- Η εφαρμογή θα είναι σε θέση να λαμβάνει με επιτυχία τα δεδομένα αποστολής του κεντρικού υπολογιστή και να τα προβάλει στην οθόνη του χρήστη.

- Ικανότητα ενημέρωσης, ανανέωσης και αλλαγής όχι μόνο εφημεριών αλλά και βαρδιών και γενικών προγραμματισμένων καθηκόντων.

- Δυνατότητα ενημέρωσης, ανανέωσης και αλλαγής γενικών προγραμματισμένων καθηκόντων από περισσότερα του ενός προγράμματα είτε αυτά προέρχονται από την ίδια είτε από διαφορετική νοσοκομειακή μονάδα.

- Οι χρήστες που δημοσιεύουν εργασίες προς ανταλλαγή να έχουν ενημέρωση για το άτομο που τις αποδέχεται στο πρόγραμμά τους

- Οι δημοσιευμένες εργασίες είναι ορατές και μπορούν να γίνουν αποδεκτές μόνο από συναδέλφους που ανήκουν στην ίδια ομάδα εργασίας.

- Όσοι επιθυμούν να εξυπηρετήσουν κάποιον συνάδελφό τους με την αποδοχή δημοσιευμένης εργασίας θα πρέπει να ενημερώνονται για το άτομο που την δημοσίευσε

- Αναλυτική και κατανοητή διεπαφή της εφαρμογής που αφορά το συγκεκριμένο χρήστη της φορητής εφαρμογής.

- Δυνατότητα αποθήκευσης και εμφάνισης στην φορητή συσκευή παλαιότερων προγραμμάτων.

- Ο χρήστης έχει τη δυνατότητα μερικής τροποποίησης της εμφάνισης στην οθόνη των προγραμμάτων και των πληροφοριών που τον αφορούν.

- Ο χρήστης της φορητής εφαρμογής έχει την δυνατότητα να ανταλλάσει, να ακυρώνει, να μεταφέρει και να αλλάζει τις εργασίες που επιθυμεί (στα πλαίσια οργάνωσης κάθε νοσοκομειακής μονάδας).

- Σε περίπτωση διαφωνίας με το πρόγραμμα εφημεριών, ο χρήστης θα υποβάλει αιτήσεις αλλαγής για τα πεδία του προγράμματος που τον αφορούν και έρχονται πιθανόν σε σύγκρουση με τις υποχρεώσεις του.

- Οι αιτήσεις θα είναι ορατές από τα άτομα ή την ομάδα ατόμων που αφορά το πρόγραμμα προς έγκριση.

- Θα υπάρχει ορατότητα μεταξύ των χρηστών της εφαρμογής που είναι συνδεδεμένοι με το σύστημα,

- ενώ παράλληλα θα εμφανίζεται και η κατάσταση κάθε χρήστη («εργάζεται», «σε ρεπό», «σε αναμονή», «σε άδεια» κτλ).

2.2.2 Μη λειτουργικές απαιτήσεις

Στις μη λειτουργικές απαιτήσεις αναφέρουμε τα γενικά χαρακτηριστικά του συστήματός μας και όχι συγκεκριμένες συμπεριφορές που μπορεί να έχει.

- Το σύστημα έχει την δυνατότητα να αποτελεί κομμάτι ενός Ολοκληρωμένου Πληροφοριακού Συστήματος Υγείας (ΟΠΣΥ) ή μεμονωμένο σύστημα παροχής υπηρεσιών.

- Το σύστημά μας θα αποτελείται από δυο εφαρμογές, μία η οποία θα είναι εγκατεστημένη στις φορητές συσκευές των χρηστών και μία Web Based εφαρμογή η οποία θα διαχειρίζεται τα δεδομένα και τις πληροφορίες του συστήματος.

- Χρήση κεντρικού υπολογιστή (Server) με δυνατότητα λήψης και αποστολής δεδομένων προς και από το World Wide Web (www).

- Ο κεντρικός Υπολογιστής θα πρέπει να έχει την δυνατότητα αποθήκευσης μεγάλου όγκου δεδομένων.

- Χρήση Βάσης Δεδομένων (ΒΔ) για την αποθήκευση των δεδομένων που διαχειρίζεται το σύστημα.

- Χρήση Βάσης Δεδομένων ειδική για φορητές συσκευές για την αποθήκευση των δεδομένων που διαχειρίζεται η φορητή εφαρμογή.

- Οι ασύρματες δυναμικές υπηρεσίες που παρέχει το σύστημά μας θα πραγματοποιούνται με την βοήθεια υπηρεσιών Web Service οι οποίες διαχειρίζονται την ανταλλαγή δεδομένων μεταξύ Κεντρικής Εφαρμογής (Work Station) - Web Service, Web Service - Φορητή Εφαρμογής και το αντίστροφο.

- Από την περιγραφή της διαδικασίας έκδοσης προγραμμάτων προκύπτει η γενική απαίτηση άμεσης επικοινωνίας όλων των χρηστών μεταξύ τους.

- Ευκολία στην χρήση και στην ανάγνωση των προγραμμάτων μέσω της φορητής συσκευής

- Ευελιξία στις δυνατότητες αλλαγής και μεταφοράς των προγραμματισμένων εργασιών

2.2.3 Ασφάλεια και περιορισμοί

Η χρήση των υπηρεσιών του συστήματος απαιτεί την ταυτοποίηση του χρήστη με τα αντίστοιχα στοιχεία ταυτοποίησης, όνομα χρήστη (username) και μοναδικό κωδικό πρόσβασης (password). Ο χρήστης θα λαμβάνει τα στοιχεία ταυτοποίησής του κατά την εγγραφή του στο σύστημα.

Κάθε χρήστης θα ανήκει σε μια ομάδα χρηστών ανάλογα με τις αρμοδιότητές του και την ειδικότητά του. Ο διαχωρισμός αυτός πραγματοποιείται για τη σαφή κατοχύρωση των δικαιωμάτων πρόσβασης κάθε χρήστη. Αυτό συνεπάγεται ότι κανένας χρήστης δε μπορεί να έχει πρόσβαση ούτε ενημέρωση για δεδομένα και πληροφορίες που αφορούν διαφορετικές ομάδες χρηστών.

Ο τρόπος και η συχνότητα χρήσης του συστήματος υπόκειται στις αρχές ορθής χρήσης στα πλαίσια της οργάνωσης κάθε νοσοκομειακής μονάδας. Για τη λειτουργία του συστήματος είναι υποχρεωτική η δυνατότητα επικοινωνίας με το WWW και κατ' επέκταση με τις υπηρεσίες Web Service τόσο της Κεντρικής όσο και της Φορητής Εφαρμογής.

Η κεντρική εφαρμογή είναι αδύνατον να λειτουργήσει χωρίς τη δυνατότητα επικοινωνίας με τις υπηρεσίες του Web Service. Σε περίπτωση που η φορητή εφαρμογή δεν διαθέτει προσωρινά πρόσβαση στο WWW και στις υπηρεσίες του Web Service, τότε ο χρήστης μπορεί να χειρίζεται τα δεδομένα που ήδη είναι αποθηκευμένα στην τοπική ΒΔ της συσκευής χωρίς όμως να είναι σε θέση να κάνει χρήση των λειτουργιών: δημοσίευσης, αποδοχής και υποβολής αίτησης των εφημεριών/εργασιών.

2.2.4 Αλληλεπιδράσεις με άλλα συστήματα

Η δημιουργία και η κατασκευή του συστήματος θα αναλάβει καθολικά την διεργασία που έχει προαναφερθεί. Αυτό σημαίνει την αντικατάσταση των χειρόγραφων καταγραφών και εγγραφών.

Σε περίπτωση χρήσης λογισμικών (π.χ. MS Excel) από τους διοικητικούς υπαλλήλους για τη δημιουργία των προγραμμάτων και των καθηκόντων, έχει προβλεφθεί

ώστε ο τύπος των αρχείων αυτών να είναι συμβατός με τη λειτουργία της δημιουργίας προγράμματος που εκτελεί η κεντρική εφαρμογή (Work Station).

Όπως έχουμε ήδη αναφέρει το σύστημα μας είναι αρκετά ευέλικτο στην ενσωμάτωσή του σε ένα οποιοδήποτε άλλο πληροφοριακό σύστημα. Ο λόγος είναι ότι εξαρτάται αποκλειστικά από Τεχνολογίες Εφαρμογών Διαδικτύου. Και οι δυο εφαρμογές κεντρική και φορητή καθώς και οι υπηρεσίες Web Service βασίζονται στο πρωτόκολλο XML, γεγονός που προσφέρει ευελιξία σε αλλαγές και σε διορθώσεις, προκειμένου να συνεργαστεί και με τμήματα διαφορετικών συστημάτων. Το ίδιο ισχύει και στον τρόπο της μεταξύ τους ανταλλαγής δεδομένων, πρόκειται για αποστολή και λήψη XML αρχείων. Θα αναφερθούμε αναλυτικότερα στα συγκεκριμένα χαρακτηριστικά σε επόμενη παράγραφο (XML Web Services 3.4).

2.2.5 Ορισμός προτεραιοτήτων

Οι προτεραιότητες που έχουμε θέσει για το συγκεκριμένο σύστημα αφορούν το σύνολο της φορητής εφαρμογής (λειτουργίες, σχεδιασμός, εργαλεία, τεχνολογίες, ΒΔ) και την επικοινωνία (ανταλλαγή δεδομένων) της φορητής συσκευής με μια απομακρυσμένη Βάση Δεδομένων μέσω Web Service.

3

Οι Τεχνολογίες του Συστήματος Διαχείρισης Εφημεριών & Καθηκόντων

3.1 Windows Mobile 6

Τα Windows Mobile 6 είναι μια πλατφόρμα για φορητές συσκευές, βασισμένη στα Windows CE 5.0, και χρησιμοποιείται κατά κόρον σε Personal digital assistants (PDAs) και smart phones. Το Microsoft Visual Studio 2008 και το Windows Mobile SDK, καθιστούν δυνατή τη δημιουργία λογισμικού για τη Windows Mobile πλατφόρμα, τόσο σε native (Visual C++), όσο και σε managed (Visual C#, Visual Basic .NET) κώδικα.

Χρησιμοποιώντας τη Windows Mobile πλατφόρμα, μπορούμε να δημιουργήσουμε καινοτόμες εφαρμογές για φορητές συσκευές. Η πλατφόρμα προσφέρει χαρακτηριστικά, όπως σύνδεση δεδομένων, ασφάλεια, πλούσια υποστήριξη API, όπως Bluetooth και Pocket Outlook Object Model (POOM), μεγάλη συλλογή από προγραμματιστικά μοντέλα που περιλαμβάνουν κώδικα native και managed, mobile web development, και πόρους συσκευών, όπως πολλαπλά νήματα (multithreating).

Το λειτουργικό σύστημα Windows Mobile 6 παρέχει άριστη συμβατότητα με τις εφαρμογές Windows Mobile 5.0. Η συμβατότητα με το Windows Mobile 5.0 είναι ένας ρητός στόχος του Windows Mobile 6 και έχει εξεταστεί εξονυχιστικά σε όλη τη διαδικασία

ανάπτυξης του Windows Mobile 6. Η συντριπτική πλειοψηφία των εφαρμογών Windows Mobile 5.0 θα τρέξουν σε συσκευές Windows Mobile 6 χωρίς τροποποίηση. Αυτή η ισχυρή συμβατότητα μας επιτρέπει να δημιουργήσουμε ένα μόνο εκτελέσιμο, το οποίο να απευθύνεται τόσο σε Windows Mobile 5.0 συσκευές, όσο και σε Windows Mobile 6 χρησιμοποιώντας είτε Windows Mobile 5.0 SDK, είτε Windows Mobile 6 Professional SDK ή Windows Mobile 6 Standard SDK. Εντούτοις, μερικά νέα χαρακτηριστικά και αλλαγές στο Windows Mobile 6 θα επηρεάσουν την ανάπτυξή μας.

3.1.1 Νέα χαρακτηριστικά των Windows Mobile 6

Τα ακόλουθα χαρακτηριστικά τηλεφώνου είναι νέα στο Windows Mobile 6. Στον πίνακα αναφέρονται οποιεσδήποτε επιπλέον απαιτήσεις λογισμικού και υλικού.

Χαρακτηριστικά	Περιγραφή
Compact Framework	Η έκδοση 2 του NET Compact Framework SP2 περιλαμβάνεται τώρα στη ROM των Windows Mobile 6 εφαρμογών. Οι Device Emulators περιλαμβάνουν επίσης SP2.
Microsoft SQL Server 2005 Compact Edition	Ο SQL Server 2005 Compact Edition είναι μια ελαφριά έκδοση του SQL Server 2005 για να χρησιμοποιείται ως μια συμπαγής τοπική βάση δεδομένων σε ένα PC, Tablet, Windows Mobile ή Windows Embedded CE συσκευή. Οι υπεύθυνοι για την ανάπτυξη θα έχουν τη δυνατότητα να χρησιμοποιήσουν τη βάση δεδομένων για να αναπτύξουν με συνέπεια τις περιστασιακά, συνδεδεμένες εφαρμογές σε όλες τις Microsoft πλατφόρμες πελατών. Η βάση δεδομένων μπορεί να χρησιμοποιηθεί για τοπική αποθήκευση, καθώς επίσης και για συγχρονισμό δεδομένων με άλλες εκδόσεις SQL Server 2005. Αυτό διευκολύνει τους υπεύθυνους να αναπτύξουν εφαρμογές που έχουν εκδόσεις για tablet/laptop/desktop αλλά και για κινητές συσκευές. Ο SQL Server Compact Edition συμπεριλαμβάνεται στη ROM των Windows Mobile 6 συσκευών.
Νέα προεπιλεγμένη βάση δεδομένων	Η EDB έχει αντικαταστήσει την CEDB σαν προεπιλεγμένη βάση δεδομένων.
Νέα APIs για την αναπαραγωγή αρχείων ήχου.	Βελτιωμένη υποστήριξη για αναπαραγωγή αρχείων ήχου.
IFRAME XHTML Element	Internet Explorer Mobile τώρα υποστηρίζει το IFRAME στοιχείο.
Windows Mobile Marketplace	Το Windows Mobile Marketplace είναι ένα framework που επιτρέπει

	στους χρήστες να αναζητούν, να αγοράζουν, να κατεβάζουν και να εγκαθιστούν εφαρμογές λογισμικού τρίτων απευθείας από τη συσκευή τους. Υπάρχουν δύο βασικά συστατικά στον Marketplace framework. Τα βασισμένα στα καταστήματα web που φιλοξενούνται από έναν ή περισσότερους συνεργάτες διανομής λογισμικού καθώς επιλέγονται από το χειριστή ή τον OEM και χρησιμοποιούνται από την Marketplace εφαρμογή πελάτη που βρίσκεται στη συσκευή. Η εφαρμογή πελατών παρέχει μια τυποποιημένη διεπαφή στα online καταστήματα και διαχειρίζεται τις συναλλαγές αγορών και εγκαταστάσεων.
Security, GPS and Resolution Awareness Tools	Εργαλεία και utilities σχεδιασμένα με σκοπό να βοηθήσουν τους υπεύθυνους για την ανάπτυξη Windows Mobile 6 εφαρμογών. Τα εργαλεία που συμπεριλαμβάνονται υποστηρίζουν GPS, Security και resolution-awareness. Το CabSignTool καθιστά ευκολότερη την υπογραφή πολλαπλών αρχείων για τη διανομή.
Cellular Emulator	Ο Cellular Emulator, ή CellEmu, διαθέτει τα ακόλουθα χαρακτηριστικά γνωρίσματα: • Ο CellEmu είναι μια UI-driven desktop εφαρμογή που είναι εύκολο να λειτουργήσει. • Τα CellEmu device images ενσωματώνουν έναν driver με πραγματικό ρυθμό παραγωγής. • Ο CellEmu υποστηρίξει PPP σύνδεση δεδομένων. • Με το Cellular Emulator, μπορεί να εξομοιωθεί πραγματική τηλεφωνική δραστηριότητα στο Device Emulator.
Device Emulator for Windows Mobile	Ο Device Emulator έχει αναβαθμιστεί στην έκδοση 2, με πολλές βελτιώσεις και νέα χαρακτηριστικά γνωρίσματα.
Local Server Framework	Αυτός ο εξομοιωτής επιτρέπει στους χρήστες για αναπτύξουν server εφαρμογές που τρέχουν στο localhost.
USB OTG Drivers	Το Windows Mobile τώρα υποστηρίζει το καθολικό serial bus (USB) On-The-Go (OTG). Το OTG standard σχεδιάζει μια μέθοδο με την οποία οι φορητές συσκευές μπορούν να αλληλεπιδράσουν άμεσα με άλλες φορητές συσκευές.

Πίνακας 1: Νέα χαρακτηριστικά των Windows Mobile 6

3.1.2 Νέες συμβάσεις ονομάτων στα Windows Mobile 6

Με την εισαγωγή του Windows Mobile 6, η ονοματολογία έχει αλλάξει για να αντιστοιχεί καλύτερα τις επωνυμίες και τα προϊόντα με την πραγματικότητα των σημερινών φορητών συσκευών που κυκλοφορούν στην αγορά. Ο ιστορικός διαχωρισμός μεταξύ των Smart phones και της Pocket PC Phone έκδοσης μειώνεται δραματικά, και στόχος είναι η ανανέωση της ορολογίας ώστε να αντανακλά καλύτερα την εξέλιξη στη βιομηχανία των φορητών συσκευών.

Ο παρακάτω πίνακας συνοψίζει τις αλλαγές:

Προηγούμενες κατηγορίες	Νέες κατηγορίες
Windows Mobile for Pocket PC	Windows Mobile 6 Classic
Windows Mobile for Pocket PC Phone Edition	Windows Mobile 6 Professional
Windows Mobile for Smartphone	Windows Mobile 6 Standard

Πίνακας 2: Νέες συμβάσεις ονομάτων στα Windows Mobile 6

Το Windows Mobile SDKs απεικονίζει αυτές τις αλλαγές στην ονομασία "Project Templates" και "Emulator Images".

3.1.3 Εγκατάσταση εργαλείων ανάπτυξης για τα Windows Mobile 6

Το λογισμικό ανάπτυξης που προαπαιτείται εξαρτάται από το λειτουργικό σύστημα του χρήστη. Για Windows XP, που χρησιμοποιήθηκαν στα πλαίσια υλοποίησης της πτυχιακής απαιτούνται:

- Microsoft Windows XP SP3.
- Microsoft Visual Studio 2008, Standard Edition ή άνω.
- Microsoft .NET Compact Framework v2 SP1. Συνιστάται το SP2.
- ActiveSync 5.
- Windows Mobile 6 Professional SDK, or Windows Mobile 6 Standard SDK ή και τα δύο.

Ο Device Emulator είναι ένα εργαλείο που μιμείται τη συμπεριφορά μιας πλατφόρμας hardware βασισμένης σε Microsoft Windows. Παρέχει μια εικονική πλατφόρμα υλικού που μπορεί να χρησιμοποιηθεί για δοκιμή των εφαρμογών σε πολλαπλά εικονικά περιβάλλοντα. Με τον Device Emulator, οι δοκιμές γίνονται χρησιμοποιώντας software που μιμείται hardware. Ο Device Emulator τρέχει κώδικα που γίνεται compiled για ARM μικροεπεξεργαστές. Όταν διαμορφώνεται με ρεαλισμό και αρχικοποιείται με ένα κατάλληλο

run-time image, παρέχει υψηλό βαθμό ομοιότητας με μια πραγματική καταναλωτική συσκευή-στόχο. Ο εξομοιωτής συσκευών υποστηρίζει τα ακόλουθα χαρακτηριστικά γνωρίσματα:

- Διαμορφώσιμη ανάλυση οθόνης
- Ευέλικτος προσανατολισμός επίδειξης για υποστήριξη παιχνιδιών
- Host-key συνδυασμοί που υποστηρίζουν την ειδική λειτουργία
- Αντιστοίχηση serial port
- Εξομοίωση καρτών αποθήκευσης
- Υποστήριξη δικτύου

Επιπλέον, ο εξομοιωτής υποστηρίζει πολλαπλά περιβάλλοντα ανάπτυξης, όπως το Visual Studio .NET 2005, το Visual Studio .NET 2003,τα Windows CE 5.0 και μεταγενέστερα. Ο χρήστης μπορεί να αναπτύξει μια εφαρμογή στο περιβάλλον που προτιμά, και να χρησιμοποιήσει έπειτα τον εξομοιωτή για να δοκιμάσει την εφαρμογή του. Τα ακόλουθα emulator images συμπεριλαμβάνονται στα SDKs:

- Windows Mobile 6 Standard SDK
- Windows Mobile 6 Standard (176x220 pixels - 96 dpi)
- Windows Mobile 6 Standard Landscape QVGA (240x320 pixels - 131 dpi)
- Windows Mobile 6 Standard QVGA (320x240 pixels - 131 dpi)
- Windows Mobile 6 Professional SDK
- Windows Mobile 6 Classic (240x320 pixels - 96 dpi)
- Windows Mobile 6 Professional (240x320 pixels - 96 dpi)
- Windows Mobile 6 Professional Square (240x240 pixels - 96 dpi)
- Windows Mobile 6 Professional Square QVGA (320x320 pixels - 128 dpi)
- Windows Mobile 6 Professional Square VGA (480x480 pixels - 192 dpi)
- Windows Mobile 6 Professional VGA (480x640 pixels - 192 dpi)

Η έκδοση 2 του Device Emulator είναι προς τα πίσω-συμβατή με την έκδοση 1. Η έκδοση 2 υποστηρίζει τα υπάρχοντα OS και τα υπάρχοντα save state αρχεία, και ενσωματώνει τις υπάρχουσες εκδόσεις του Visual Studio. Τα χαρακτηριστικά γνωρίσματα που είναι νέα στην έκδοση 2 του emulator περιλαμβάνουν:

- Βελτιώσεις στην εκτέλεση και στην απόδοση για debug/deploy.
- Βελτιωμένη υποστήριξη hardware που περιλαμβάνει προειδοποιητικά LEDs, υποστήριξη μικροτηλεφώνων, μπαταρία και backlight ρύθμιση αντίθεσης.
- ActiveSync-over-RNDIS για να βελτιώσουν τη συμβατότητα με τις βασισμένες συσκευές.
- Επιταχυνόμενη υποστήριξη video για Direct3D Mobile.
- Υποστήριξη Backlight.
- Βελτιωμένη διανομή φακέλων.
- Εξομοίωση της μπαταρίας και των προειδοποιητικών LEDs.

3.2. NET Framework 2.0

Το .NET Framework [11] είναι ένα αναπόσπαστο component των Windows που υποστηρίζει τη δημιουργία και εκτέλεση της επόμενης γενιάς εφαρμογών και υπηρεσιών διαδικτύου (Web services). Το .NET Framework έχει σχεδιαστεί για να εκπληρώσει τους ακόλουθους στόχους :

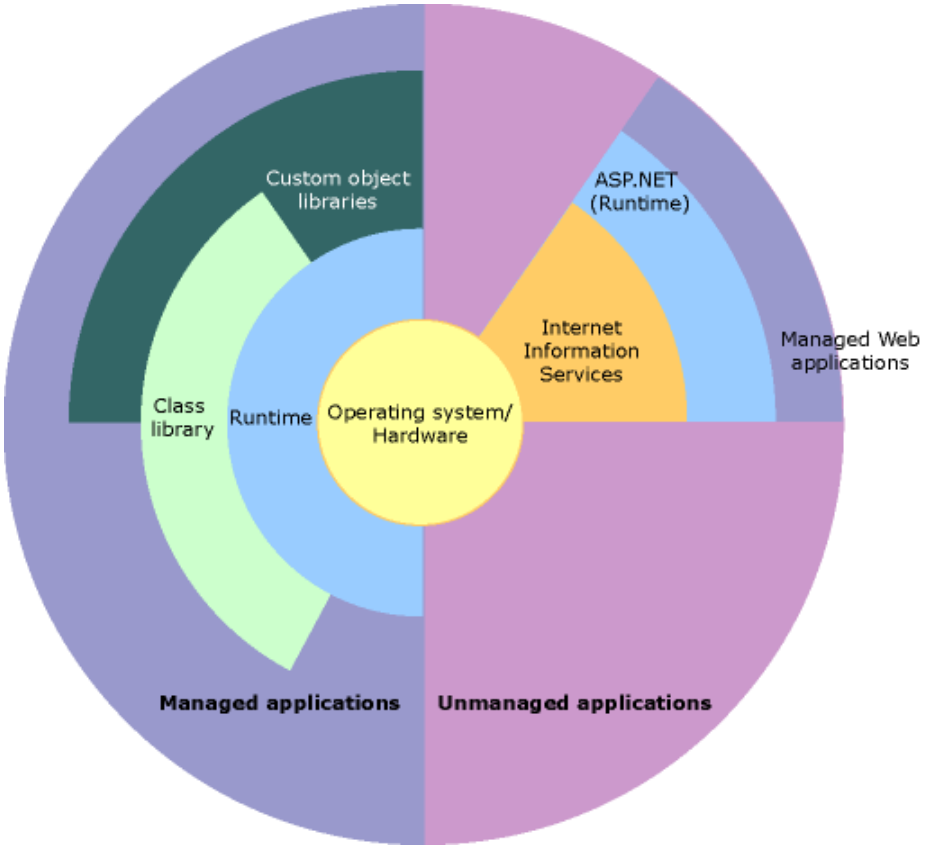
- Να παρέχει ένα συνεπές αντικειμενοστραφές περιβάλλον προγραμματισμού είτε ο κώδικας αποθηκεύεται και εκτελείται τοπικά, είτε εκτελείται τοπικά αλλά είναι κατανεμημένος στο διαδίκτυο, ή τέλος εκτελείται από μακριά.
- Να παρέχει ένα περιβάλλον εκτέλεσης κώδικα που ελαχιστοποιεί την επέκταση του λογισμικού και τον ανταγωνισμό των εκδόσεων.
- Να παρέχει ένα περιβάλλον που προωθεί την ασφαλή εκτέλεση του κώδικα, συμπεριλαμβανομένου του κώδικα που δημιουργείται από έναν άγνωστο ή όχι πλήρως έμπιστο τρίτο πρόσωπο.
- Να παρέχει ένα περιβάλλον εκτέλεσης κώδικα που αποβάλλει τα προβλήματα απόδοσης των scripted ή interpreted περιβαλλόντων.
- Να παρέχει στον υπεύθυνο για την ανάπτυξη εμπειρία από ποικίλους τύπους εφαρμογών, όπως εφαρμογές βασισμένες στα WINDOWS και στο WEB.
- Να χτίσει όλη την επικοινωνία σε πρότυπα βιομηχανίας που εξασφαλίζουν ότι ο κώδικας βασισμένος σε .NET Framework μπορεί να ενσωματωθεί με οποιοδήποτε άλλο κώδικα.

Το .NET Framework έχει δύο κύρια συστατικά: την common language runtime και τη .NET Framework class library. Η common language runtime είναι η βάση του .NET Framework. Μπορείτε να σκεφτείτε το χρόνο εκτέλεσης ως πράκτορα που διαχειρίζεται τον

κώδικα στο χρόνο εκτέλεσης, που παρέχει υπηρεσίες πυρήνα όπως η διαχείριση μνήμης ή η διαχείριση νημάτων και επιβάλλει ασφάλεια τύπων και άλλες μορφές ακρίβειας κώδικα που προωθούν την ασφάλεια και την ευρωστία. Στην πραγματικότητα, η έννοια της διαχείρισης κώδικα είναι μια θεμελιώδης αρχή του χρόνου εκτέλεσης. Ο κώδικας που στοχεύει στο χρόνο εκτέλεσης είναι γνωστός ως διοικούμενος κώδικας, ενώ ο κώδικας που δεν στοχεύει στο χρόνο εκτέλεσης είναι γνωστός όπως unmanaged κώδικας. Η βιβλιοθήκη κλάσεων (class library), το άλλο κύριο συστατικό του .NET Framework, είναι μια περιεκτική, αντικειμενοστραφής συλλογή επαναχρησιμοποιήσιμων τύπων που μπορεί κανείς να χρησιμοποιήσει για να αναπτύξει από παραδοσιακές command-line ή graphical user interface (GUI) εφαρμογές ως εφαρμογές βασισμένες στις πιο πρόσφατες καινοτομίες που παρέχονται από το ASP.NET, όπως οι μορφές διαδικτύου (Web Forms) και οι υπηρεσίες διαδικτύου (XML Web services).

Το .NET Framework μπορεί να φιλοξενηθεί από components που φορτώνουν την common language runtime στις διεργασίες τους και αρχικοποιούν την εκτέλεση του διοικούμενου κώδικα, δημιουργώντας έτσι ένα περιβάλλον λογισμικού που μπορεί να εκμεταλλευτεί τόσο τα διοικούμενα όσο και τα μη διοικούμενα χαρακτηριστικά γνωρίσματα. Το .NET Framework όχι μόνο παρέχει διάφορους runtime hosts, αλλά υποστηρίζει και την ανάπτυξη runtime hosts τρίτων προσώπων.

Το ακόλουθο σχήμα παρουσιάζει τη σχέση common language runtime και της βιβλιοθήκης κλάσεων στις εφαρμογές μας και στο γενικό σύστημα. Η απεικόνιση δείχνει επίσης, πώς ο διοικούμενος κώδικας λειτουργεί μέσα σε μια μεγαλύτερη αρχιτεκτονική.



Εικόνα 5: Περιεχόμενα του .NET Framework

Οι ενότητες που ακολουθούν περιγράφουν αναλυτικά τα κύρια συστατικά και χαρακτηριστικά του .NET

3.2.1 Common Language Runtime

Η common language runtime διαχειρίζεται τη μνήμη, την εκτέλεση των νημάτων, την εκτέλεση του κώδικα, την πιστοποίηση ασφάλειας του κώδικα και άλλες υπηρεσίες του συστήματος. Αυτά τα χαρακτηριστικά είναι βασικά για τον managed code που τρέχει στην common language runtime.

Όσον αφορά την ασφάλεια, τα διοικούμενα components διακρίνονται από διαφορετικούς βαθμούς εμπιστοσύνης, ανάλογα με τους παράγοντες που περιλαμβάνουν την προέλευσή τους (όπως το Διαδίκτυο, το επιχειρηματικό δίκτυο ή ο τοπικός υπολογιστής). Αυτό σημαίνει ότι ένα διοικούμενο component μπορεί ή όχι να είναι σε θέση να πραγματοποιήσει λειτουργίες πρόσβασης σε αρχεία, σε εγγραφές ή άλλες λεπτές λειτουργίες ακόμα κι αν χρησιμοποιείται στην ίδια ενεργό εφαρμογή. Τα χαρακτηριστικά της common language runtime είναι τα εξής:

- Επιβάλλει την ασφάλεια πρόσβασης κώδικα
- Ενδυναμώνει επίσης την ευρωστία κώδικα υλοποιώντας μια αυστηρή type-and-code-verification υποδομή, που λέγεται κοινό σύστημα τύπων (CTS).
- Το διοικούμενο περιβάλλον του χρόνου εκτέλεσης εξαλείφει πολλά κοινά ζητήματα λογισμικού. Παραδείγματος χάριν, ο χρόνος εκτέλεσης χειρίζεται αυτόματα τις αναφορές στα αντικείμενα, απελευθερώνοντας τες όταν δεν χρησιμοποιούνται πλέον. Αυτή η αυτόματη διαχείριση μνήμης επιλύει τα δύο πιο κοινά λάθη που συναντώνται στις εφαρμογές τις διαρροές μνήμης και τις άκυρες αναφορές μνήμης.
- Επιταχύνει την παραγωγικότητα των ανθρώπων που αναπτύσσουν τις εφαρμογές. Παραδείγματος χάριν, οι προγραμματιστές μπορούν να γράψουν τις εφαρμογές στη γλώσσα ανάπτυξης της επιλογής τους, εκμεταλλευόμενοι πλήρως το χρόνο εκτέλεσης, τη βιβλιοθήκη κλάσεων και τα components που γράφονται σε άλλες γλώσσες από άλλους υπεύθυνους για την ανάπτυξη.
- Ενώ σχεδιάστηκε για το λογισμικό του μέλλοντος, υποστηρίζει επίσης το λογισμικό του παρόντος και του παρελθόντος.
- Ο χρόνος εκτέλεσης έχει ως σκοπό να ενισχύσει την απόδοση.
- Μπορεί να φιλοξενηθεί από υψηλής απόδοσης server-side εφαρμογές, όπως Microsoft SQL Server και Internet Information Services (IIS).

3.2.2 .NET Framework Class Library

Η .NET Framework class library είναι μια συλλογή από επαναχρησιμοποιήσιμους τύπους που ενοποιούνται με τη common language runtime. Η class library είναι αντικειμενοστραφής, παρέχοντας τύπους από τους οποίους ο managed code του χρήστη μπορεί να αντλήσει λειτουργικότητα. Αυτό όχι μόνο κάνει τους τύπους του .NET Framework εύκολους στη χρήση, αλλά μειώνει και το χρόνο που σχετίζεται με την εκμάθηση νέων χαρακτηριστικών του .NET Framework. Επιπλέον, η συλλογή των κλάσεων του .NET Framework υλοποιούν ένα σύνολο διασυνδέσεων τις οποίες μπορεί ο προγραμματιστής να χρησιμοποιήσει για να αναπτύξει τις δικές του συλλογές κλάσεων.

Όπως είναι αναμενόμενο από μια αντικειμενοστραφή class library, οι τύποι του .NET Framework δίνουν τη δυνατότητα πραγματοποίησης μιας συλλογής από προγραμματιστικές ενέργειες συμπεριλαμβανομένων ενεργειών όπως διαχείριση αλφαριθμητικών, συλλογή δεδομένων, σύνδεση με βάση δεδομένων και πρόσβαση σε αρχεία. Εκτός από αυτές τις

κοινές ενέργειες η class library συμπεριλαμβάνει τύπους που υποστηρίζουν μια συλλογή από εξειδικευμένα σενάρια ανάπτυξης. Για παράδειγμα, το .NET Framework μπορεί να χρησιμοποιηθεί για να την ανάπτυξη των ακόλουθων τύπων εφαρμογών και υπηρεσιών:

- Console applications.
- Windows GUI applications (Windows Forms).
- ASP.NET applications.
- XML Web services.
- Windows services.

Για παράδειγμα, οι κλάσεις των Windows Forms είναι ένα περιεκτικό σύνολο από επαναχρησιμοποιήσιμους τύπους που απλοποιούν σε μεγάλο βαθμό την ανάπτυξη Windows GUI. Για να γράψει κανείς μια ASP.NET Web Form εφαρμογή, μπορεί να χρησιμοποιήσει τις Web Forms κλάσεις.

3.2.3 Ανάπτυξη Εφαρμογών Πελατών

Οι εφαρμογές πελατών είναι οι πλησιέστερες σε ένα παραδοσιακό είδος εφαρμογών στον προγραμματισμό βασισμένο στα Windows. Πρόκειται για ένα τύπο εφαρμογών που εμφανίζουν παράθυρα σε φόρμες, δίνοντας τη δυνατότητα στο χρήστη να πραγματοποιήσει μια ενέργεια. Οι εφαρμογές πελατών περιλαμβάνουν εφαρμογές όπως επεξεργαστές κειμένου και λογιστικά φύλλα καθώς επίσης και τις συνήθεις επιχειρησιακές εφαρμογές όπως τα εργαλεία εισαγωγής δεδομένων, εργαλεία αναφοράς κλπ. Οι εφαρμογές πελατών συνήθως περιλαμβάνουν παράθυρα, μενού, κουμπιά και άλλα GUI στοιχεία και συχνά έχουν πρόσβαση σε τοπικούς πόρους, όπως σύστημα αρχείων και περιφερειακούς όπως εκτυπωτές.

Ένα άλλο είδος εφαρμογών πελατών είναι το παραδοσιακό ActiveX control (τώρα έχει αντικατασταθεί από το Windows Form Control) που επεκτείνεται μέσω του Διαδικτύου ως ιστοσελίδα. Αυτή η εφαρμογή είναι σαν τις άλλες εφαρμογές πελατών: εκτελείται natively, έχει πρόσβαση στους τοπικούς πόρους, και περιλαμβάνει γραφικά στοιχεία.

Στο παρελθόν, οι προγραμματιστές δημιουργούσαν τέτοιες εφαρμογές C/C++ σε συνδυασμό με Microsoft Foundation Classes (MFC) ή με ένα γρήγορο περιβάλλον ανάπτυξης όπως (RAD) Microsoft Visual Basic. Το .NET Framework ενσωματώνει στοιχεία από τα υπάρχοντα προϊόντα σε ένα, απλοποιώντας έτσι την ανάπτυξη εφαρμογών πελατών.

Οι Windows Forms classes που περιέχονται στο .NET Framework σχεδιάστηκαν για να χρησιμοποιούνται για GUI ανάπτυξη. Ο χρήστης μπορεί εύκολα να δημιουργήσει παράθυρα εντολών, κουμπιά, μενού, γραμμές εργαλείων και άλλα στοιχεία οθόνης με την ευελιξία που χρειάζεται για να εξυπηρετηθούν μεταβαλλόμενες επιχειρηματικές ανάγκες.

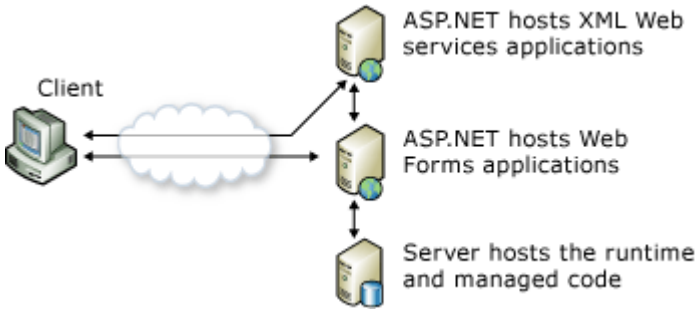
Για παράδειγμα, το .NET Framework παρέχει απλές ιδιότητες για τη ρύθμιση χαρακτηριστικών που σχετίζονται με τις φόρμες. Σε μερικές περιπτώσεις το βασικό λειτουργικό σύστημα δεν υποστηρίζει την άμεση αλλαγή αυτών των χαρακτηριστικών οπότε σε αυτές τις περιπτώσεις το .NET Framework αυτόματα επαναδημιουργεί τις φόρμες. Αυτός είναι ένας από τους πολλούς τρόπους που το .NET Framework ενσωματώνει τη διεπαφή των υπεύθυνων για την ανάπτυξη, κάνοντας τον κώδικα απλούστερο και πιο συμπαγή.

Αντίθετα από τα ActiveX controls, τα Windows Forms δεν έχουν πλήρη πρόσβαση στον υπολογιστή ενός χρήστη. Αυτό σημαίνει ότι ο δυαδικός ή natively εκτελέσιμος κώδικας μπορεί να έχει πρόσβαση σε μερικούς από τους πόρους στο σύστημα του χρήστη (όπως τα στοιχεία GUI και περιορισμένη πρόσβαση αρχείων) χωρίς να είναι σε θέση να προσεγγίσει άλλους πόρους. Λόγω της ασφαλούς πρόσβασης κώδικα, πολλές εφαρμογές που άλλοτε έπρεπε να εγκατασταθούν στο σύστημα ενός χρήστη, μπορούν τώρα να επεκταθούν μέσω του διαδικτύου. Οι εφαρμογές μπορούν να υλοποιήσουν τα χαρακτηριστικά μιας τοπικής εφαρμογής, έχοντας επεκταθεί σαν ιστοσελίδες.

3.2.4 Ανάπτυξη εφαρμογών Κεντρικού Υπολογιστή

Οι εφαρμογές από την πλευρά του server στο managed world υλοποιούνται μέσω runtime hosts. Unmanaged εφαρμογές host σε common language runtime, που επιτρέπει στο διοικούμενο κώδικα να ελέγξει τη συμπεριφορά του κεντρικού υπολογιστή. Αυτό το πρότυπο παρέχει στους χρήστες όλα τα χαρακτηριστικά γνωρίσματα της common language runtime και της βιβλιοθήκης κλάσεων, κερδίζοντας παράλληλα την απόδοση και την εξελιξιμότητα του host server.

Στο επόμενο σχήμα παρουσιάζεται ένα βασικό σχήμα δικτύου με managed code να τρέχει σε διαφορετικούς servers. Αντίστοιχοι Servers όπως ο IIS και ο SQL Server μπορούν να πραγματοποιήσουν δεδομένες λειτουργίες, ενώ η λογική της εφαρμογής του χρήστη μπορεί να εκτελεστεί μέσω του managed code (διαχειρίσιμου κώδικα).



Εικόνα 6: Σχήμα δικτύου με managed code να τρέχει σε διαφορετικούς Servers

ASP.NET είναι το hosting environment που επιτρέπει στους προγραμματιστές να χρησιμοποιούν το .NET Framework για να στοχεύσουν Web-based εφαρμογές. Παρόλα αυτά,

το ASP.NET είναι κάτι περισσότερο από ένας runtime host, είναι μια ολοκληρωμένη αρχιτεκτονική για την ανάπτυξη Web sites και Internet-distributed objects χρησιμοποιώντας managed code. Τόσο οι Web Forms όσο και οι XML Web services χρησιμοποιούν IIS και ASP.NET σαν μηχανισμό δημοσίευσης για τις εφαρμογές, ενώ και οι δύο υποστηρίζουν μια συλλογή από κλάσεις στο .NET Framework.

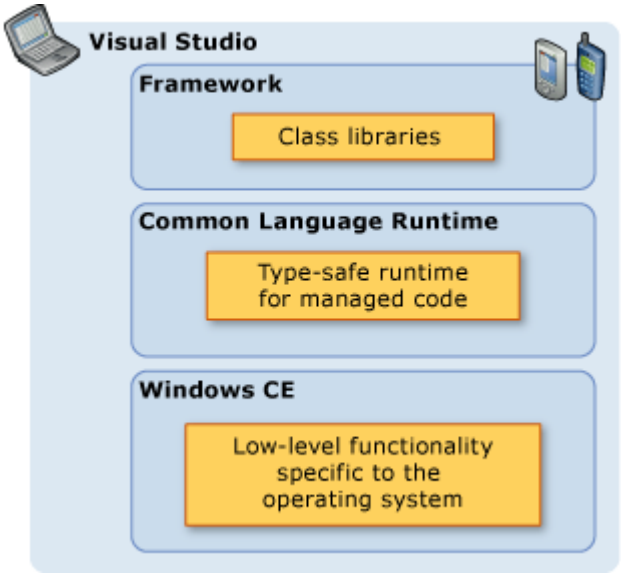
3.3. NET Compact Framework

Το .NET Compact Framework είναι ένα περιβάλλον ανεξάρτητο από το υλικό για την εκτέλεση προγραμμάτων σε συσκευές, όπως personal digital assistants (PDAs), κινητά τηλέφωνα και μετασχηματιστές (set-top boxes). Τρέχει πάνω από το λειτουργικό σύστημα Microsoft Windows CE και στηρίζεται σε ένα rebuilt της common language runtime (CLR), που έχει σχεδιαστεί για να λειτουργεί αποτελεσματικά όταν τρέχει προγράμματα για συσκευές με περιορισμένους πόρους. Το .NET Compact Framework φέρνει το managed code και τις XML Web Services στις συσκευές και παρέχει οφέλη όπως η ασφάλεια, η συλλογή απορριμμάτων, ο χειρισμός εξαιρέσεων και η προστασία.

Το .NET Compact Framework είναι ένα υποσύνολο της βιβλιοθήκης κλάσεων του .NET Framework και περιέχει επίσης κλάσεις αποκλειστικά σχεδιασμένες για συσκευές με περιορισμένους πόρους. Υλοποιεί περίπου το 30 % της βιβλιοθήκης κλάσεων του πλήρους. .NET Framework. Κληρονομεί την πλήρη .NET Framework του χρόνου εκτέλεσης κοινής γλώσσας και της διοικούμενης εκτέλεσης κώδικα . Το .NET Framework υποστηρίζει ανάπτυξη σε Visual Basic και Visual C#, αλλά όχι σε C++.

3.3.1 Αρχιτεκτονική του .NET Compact Framework

Η ακόλουθη απεικόνιση παρουσιάζει την αρχιτεκτονική της πλατφόρμας .NET Compact Framework.



Εικόνα 7: Αρχιτεκτονική της πλατφόρμας .NET Compact Framework

3.3.1.1 Windows CE

Το .NET Compact Framework χρησιμοποιεί το λειτουργικό σύστημα Windows CE για τη λειτουργικότητα και για διάφορα συγκεκριμένα χαρακτηριστικά των συσκευών. Διάφοροι τύποι και assemblies, όπως για Windows Forms, γραφικά, σχέδιο και Web Services, ξαναδημιουργήθηκαν για να εκτελούνται αποτελεσματικά στις συσκευές, παρόλο που είχαν αντιγραφεί από το πλήρες .NET Framework.

3.3.1.2 Common Language Runtime

Το .NET Compact Framework common language runtime (CLR) ξαναδημιουργήθηκαν επίσης για να επιτρέψει σε περιορισμένους πόρους να τρέξουν στην περιορισμένη μνήμη και για να χρησιμοποιήσουν αποτελεσματικά την ισχύ της μπαταρίας.

Υπάρχει ένα επίπεδο προσαρμογής της πλατφόρμας, που δεν παρουσιάζεται στο παραπάνω σχήμα, μεταξύ του Windows CE και του common language runtime για να αντιστοιχήσει τις υπηρεσίες και τις διεπαφές συσκευών που απαιτούνται από το CLR και το Framework επάνω στις υπηρεσίες Windows CE και τις διεπαφές.

3.3.1.3 Framework

Το .NET Compact Framework είναι ένα υποσύνολο του .NET Framework και περιέχει επίσης χαρακτηριστικά που έχουν σχεδιαστεί αποκλειστικά για το .NET Compact Framework. Παρέχει τα χαρακτηριστικά και την ευκολία της χρήσης, πράγμα που

προσελκύει καθαρά υπεύθυνους για την ανάπτυξη εφαρμογής συσκευών στο .NET Framework και προγραμματιστές desktop εφαρμογών στις συσκευές.

3.3.1.4 Visual Studio

Η εμπειρία με το Microsoft Visual studio 2008 για την ανάπτυξη εφαρμογών έξυπνων συσκευών είναι τόσο εύκολη όσο και με τις desktop εφαρμογές. Η smart device ανάπτυξη στο Visual studio περιλαμβάνει ένα σύνολο τύπων project και εξομοιωτών που στοχεύουν στην ανάπτυξη Pocket PC, Smartphone, και ενσωματωμένο Windows CE εφαρμογών.

3.4 XML Web Services

Οι υπηρεσίες διαδικτύου (web services) είναι XML αναπαραστάσεις προγραμμάτων, αντικειμένων ή κειμένων που είναι προσπελάσιμα μέσω Internet για απ' ευθείας αλληλεπίδραση μεταξύ εφαρμογών. Οι υπηρεσίες διαδικτύου μπορούν να προσπελαστούν με χρήση φυλλομετρητών αλλά δεν απαιτείται η χρήση ούτε φυλλομετρητή ούτε HTML. Οι υπηρεσίες διαδικτύου παρέχουν έναν ανεξάρτητο από δεδομένα μηχανισμό παρουσίασης των υπηρεσιών της επιχείρησης, με χρήση στάνταρντ XML πρωτοκόλλων τεχνολογίες που χρησιμοποιούνται και συμπεριλαμβάνουν:

- XML, που περιλαμβάνει βασική XML, XML schemas και XML parsers.
- SOAP (Simple Object Access Protocol), που αποτελεί ένα πρωτόκολλο επικοινωνίας εφαρμογών βασισμένο σε XML.
- WSDL (Web Services Description Languages), που είναι ένα XML schema για περιγραφή των μηνυμάτων, λειτουργιών και αντιστοιχήσεις πρωτοκόλλων των υπηρεσιών διαδικτύου.
- UDDI (Universal Description Discovery and Integration), που είναι ο χώρος αποθήκευσης για καταχώρηση και αναζήτηση περιγραφών υπηρεσιών διαδικτύου.

Ένα από τα πρωταρχικά πλεονεκτήματα της αρχιτεκτονικής των XML Web services είναι ότι επιτρέπει σε προγράμματα που γράφονται σε διαφορετικές γλώσσες και για διαφορετικές πλατφόρμες να επικοινωνούν με τρόπο βασισμένο σε πρότυπα. Το άλλο σημαντικό πλεονέκτημα που έχουν οι XML Web services σε σχέση με προηγούμενες προσπάθειες είναι ότι λειτουργούν με τυποποιημένα Web πρωτόκολλα—XML, HTTP και TCP/IP . Ένας σημαντικός αριθμός επιχειρήσεων έχει ήδη μια υποδομή Ιστού και ανθρώπους με τη γνώση και την εμπειρία να τη διατηρήσουν, ώστε και πάλι, το κόστος καταχώρησης XML Web services να είναι σημαντικά λιγότερο από τις προηγούμενες τεχνολογίες.

Έχουμε ορίσει μια XML Web service ως υπηρεσία λογισμικού που διατίθεται στον παγκόσμιο ιστό μέσω του SOAP, περιγράφεται με ένα αρχείο WSDL και καταχωρείται στο UDDI. Η επόμενη λογική ερώτηση αφορά τη λειτουργικότητα που μπορούν να παρέχουν. Οι πρώτες XML Web services έτειναν να είναι πηγές πληροφοριών που μπορούσαν εύκολα να ενσωματωθούν σε εφαρμογές— αναφορά αποθεμάτων, καιρικές προβλέψεις, αποτελέσματα αθλημάτων κ.λπ. Η έκθεση εφαρμογών που ήδη υπάρχουν ως XML Web services θα επιτρέψει στους χρήστες να αναπτύξουν νέες, ισχυρότερες εφαρμογές που χρησιμοποιούν τις XML Web services ως δομικές μονάδες. Για παράδειγμα, ένας χρήστης μπορεί να αναπτύξει μια εφαρμογή αγοράς στην οποία να λαμβάνονται αυτόματα πληροφορίες για τις τιμές από ποικίλους προμηθευτές και να επιτρέπει στο χρήστη να επιλέξει έναν προμηθευτή, να υποβάλει την παραγγελία και να παρακολουθήσει την εξέλιξή της έως ότου γίνει η παραλαβή. Η εφαρμογή αγοράς, εκτός από την έκθεση των υπηρεσιών της στον Web, στη συνέχεια μπορεί να χρησιμοποιήσει XML Web services για να ελέγξει τη πιστωτική του πελάτη, να χρεώσει το λογαριασμό του και να κανονίσει την αποστολή με μια μεταφορική εταιρία.

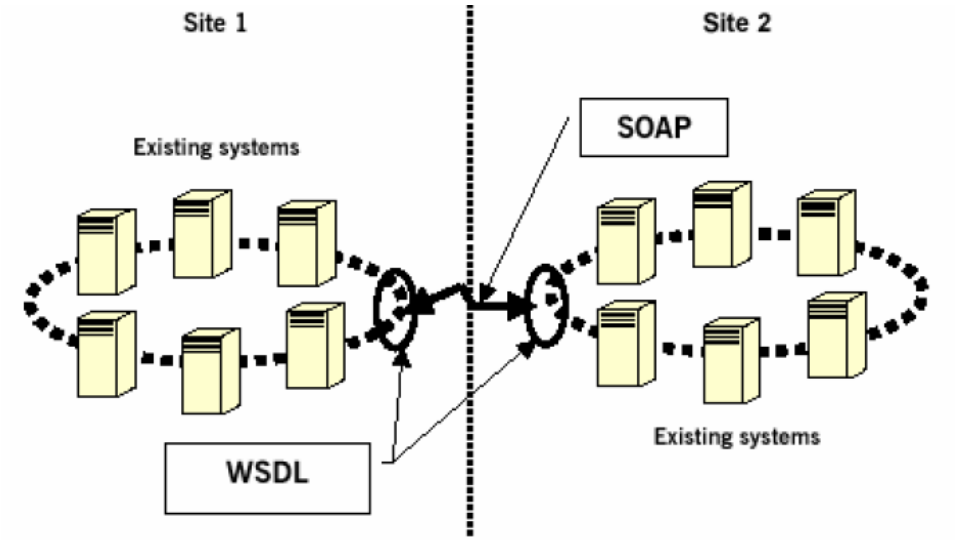
Στο μέλλον, μερικές από τις πιο ενδιαφέρουσες XML Web services θα υποστηρίζουν εφαρμογές που χρησιμοποιούν το διαδίκτυο για να κάνουν πράγματα που δεν μπορούν να γίνουν σήμερα. Για παράδειγμα, μια από τις υπηρεσίες που οι XML Web Services θα καθιστούν δυνατή είναι η ημερολογιακή υπηρεσία. Εάν ο οδοντίατρος και ο μηχανικός σας εξέθεταν τα ημερολόγιά τους μέσω αυτής της XML Web service, θα μπορούσατε να κανονίσετε τις συναντήσεις σας on line.

3.4.1 Simple Object Access Protocol (SOAP)

Το SOAP είναι ένα πλαίσιο ανταλλαγής μηνυμάτων βασισμένο σε XML. Είναι ειδικά σχεδιασμένο για την ανταλλαγή μηνυμάτων μέσω διαδικτύου. Είναι απλό στη χρήση και εντελώς ανεξάρτητο από λειτουργικό σύστημα, γλώσσα προγραμματισμού ή πλατφόρμα κατανεμημένων συστημάτων.

Εκτός από το να παρέχει μια αντιστοίχηση σε ένα επίπεδο μεταφοράς για την ανταλλαγή XML μηνυμάτων μέσω του διαδικτύου, με το SOAP μια επιχείρηση μπορεί να:

- Δημοσιοποιήσει τις υπηρεσίες της για ανταλλαγή XML εταιρικών δεδομένων.
- Ανακαλύψει την τοποθεσία και την μορφή υπηρεσιών άλλων επιχειρήσεων.
- Καθορίσει ιδιότητες των ανταλλασσόμενων μηνυμάτων που σχετίζονται με την ποιότητα της υπηρεσίας.



Εικόνα 8: Soap μηνύματα για διασύνδεση απομακρυσμένων δικτυακών τόπων

Όπως φαίνεται στο σχήμα παραπάνω, το SOAP παρέχει έναν ανεξάρτητο και γενικό πρωτόκολλο επικοινωνίας για την σύνδεση δύο ή περισσότερων πυλών ή εταιρικών δικτυακών τόπων. Τα σημερινά συστήματα αποτελούνται από έναν συνδυασμό πολλών διαφορετικών κατηγοριών υλικού και λογισμικού. Το SOAP και η XML βοηθούν στην συμφωνία ενός κοινού τρόπου ανταλλαγής δεδομένων μεταξύ ετερογενών συστημάτων. Το WSDL χρησιμοποιείται για την περιγραφή των υπηρεσιών και το SOAP για την μετάδοση της πληροφορίας.

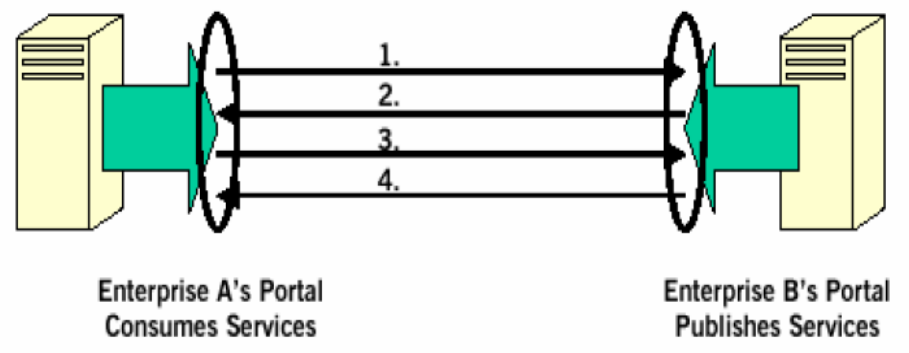
Η εικόνα 5 δείχνει τον μηχανισμό με τον οποίο επιτυγχάνεται η ανακάλυψη των υπηρεσιών μιας εταιρείας με χρήση SOAP.

1. Η επιχείρηση Α χρησιμοποιεί ένα URL που παρέχεται από την επιχείρηση Β για να ανακτήσει μια λίστα με τις υπηρεσίες που δημοσιεύει η Β.

2. Η επιχείρηση Α «κατεβάζει» τα XML schemas (συνήθως σε WSDL) που περιγράφουν την μορφή των μηνυμάτων που αναμένονται από τις υπηρεσίες της εταιρείας Β.

3. Η επιχείρηση Α σχηματίζει το ανάλογο XML μήνυμα και το αποστέλλει μέσω SOAP στην επιχείρηση Β.

4. Η επιχείρηση Β στέλνει μια απάντηση, μέσω SOAP, την οποία η επιχείρηση Α ερμηνεύει χρησιμοποιώντας την πληροφορία για το XML schema που έλαβε στο βήμα 2.



Εικόνα 9: Ανακάλυψη Soap μηνυμάτων

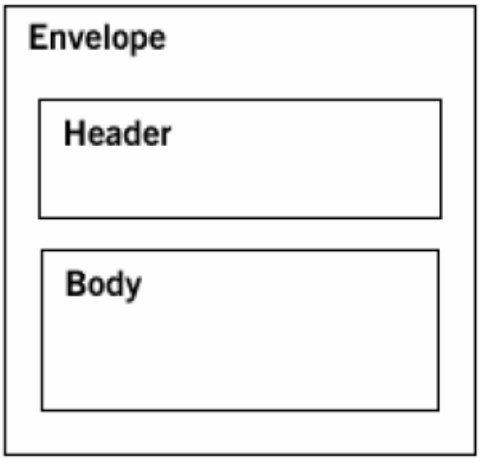
Με αυτόν το τρόπο, οι δύο επιχειρήσεις μπορούν να ανταλλάξουν πληροφορίες για τις υπηρεσίες που επιθυμούν να παρέχουν και να καταναλώσουν. Μία άλλη μέθοδος ανακάλυψης υπηρεσιών είναι με χρήση του UDDI.

3.4.2 Μορφή SOAP μηνύματος

Τα μηνύματα SOAP αποτελούνται από τρία βασικά μέρη:

- Τον Φάκελο (Envelope).
- Την Επικεφαλίδα (Header), που είναι προαιρετική.
- Το Σώμα (Body).

Η εικόνα 16 απεικονίζει τα τρία μέρη από τα οποία αποτελείται ένα SOAP μήνυμα.



Εικόνα 10: Μέρη ενός Soap μηνύματος

Ο φάκελος είναι υποχρεωτικός και ουσιαστικά σηματοδοτεί την αρχή και το τέλος του μηνύματος (αν και τα μηνύματα μπορεί να περιέχουν συνδέσμους σε αντικείμενα εκτός του φακέλου). Περιέχει πληροφορίες για το περιεχόμενο του μηνύματος και πώς να γίνει η επεξεργασία του. Η επικεφαλίδα είναι προαιρετική και περιέχει επιπλέον πληροφορίες που σχετίζεται με την ασφάλεια, τις συναλλαγές και την ποιότητα των υπηρεσιών. Περιλαμβάνει έναν μηχανισμό με τον οποίο τα συναλλασσόμενα μέρη διαπραγματεύονται συμφωνία για υποστήριξη μιας συγκεκριμένης επικεφαλίδας ή συνόλου επικεφαλίδων. Ένα SOAP μήνυμα μπορεί να έχει περισσότερες από μια επικεφαλίδες. Το σώμα περιέχει τα δεδομένα του πραγματικού μηνύματος.

3.4.3 Web Services Description Languages (WSDL)

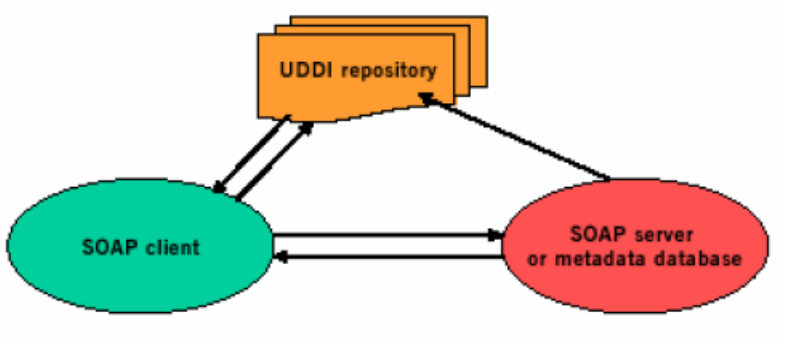
Αφού το SOAP έγινε διαθέσιμο σαν ένας μηχανισμός ανταλλαγής XML μηνυμάτων μεταξύ επιχειρήσεων (ή ξεχωριστών εφαρμογών μέσα στην ίδια επιχείρηση), παρουσιάστηκε η ανάγκη ενός καλύτερου τρόπου για την περιγραφή των μηνυμάτων καθώς και του τρόπου με τον οποίο γίνεται η ανταλλαγή τους.

Το WSDL είναι ένα XML schema, που αναπτύχθηκε από την Microsoft και την IBM με σκοπό να ορίσει το XML μήνυμα, τη λειτουργία και το πρωτόκολλο αντιστοίχησης μιας υπηρεσίας διαδικτύου που προσπελαύνεται χρησιμοποιώντας SOAP ή κάποιο άλλο XML πρωτόκολλο. Το συντακτικό του WSDL επιτρέπει τον αφαιρετικό ορισμό τόσο των μηνυμάτων όσο και των λειτουργιών των μηνυμάτων, έτσι ώστε να μπορούν να αντιστοιχηθούν σε πολλαπλές φυσικές υλοποιήσεις.

3.4.4 Universal Description, Discovery και Integration (UDDI)

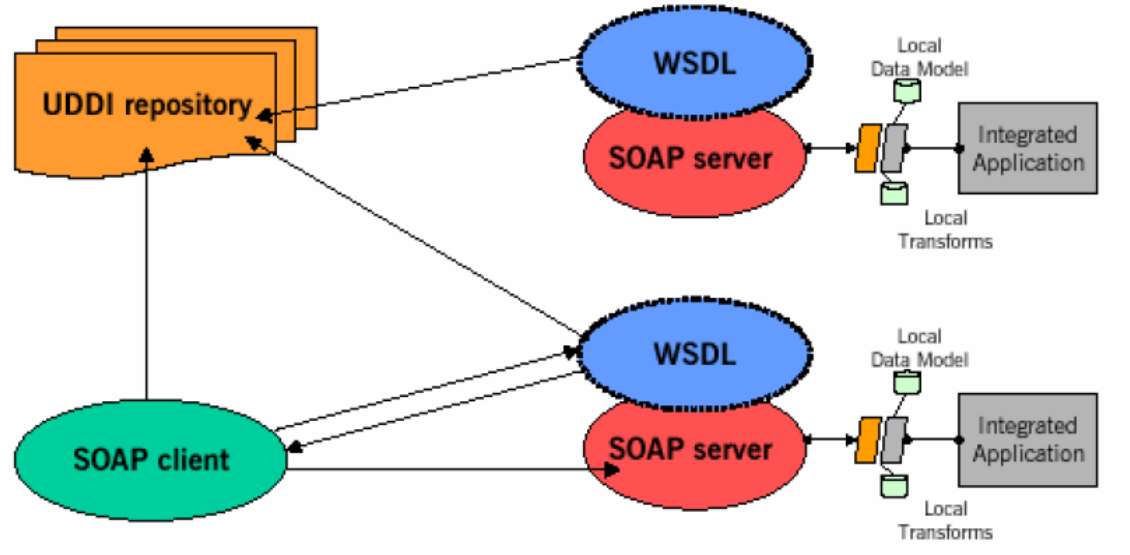
Το UDDI ορίζει ένα μοντέλο δεδομένων (σε XML) και SOAP APIs για καταχώρηση και αναζήτηση πληροφορίας μιας επιχείρησης, συμπεριλαμβανομένης της πληροφορίας που σχετίζεται με τις υπηρεσίες που παρέχει η επιχείρηση στο διαδίκτυο. Η βασική ιδέα είναι ότι οι επιχειρήσεις χρησιμοποιούν τα SOAP APIs για να καταχωρούν τις υπηρεσίες που παρέχουν στο UDDI. Άλλες επιχειρήσεις ψάχνουν στο UDDI όταν θέλουν να ανακαλύψουν έναν εμπορικό συνεργάτη. Η πληροφορία στο UDDI κατηγοριοποιείται σύμφωνα με τον τύπο των εταιρειών και την γεωγραφική τους θέση. Από τη στιγμή που βάσει των παραπάνω κριτηρίων βρεθεί η επιθυμητή επιχείρηση, το UDDI μπορεί να παρέχει πληροφορία εύρεσης των υπηρεσιών που παρέχει η επιχείρηση, δίνοντας ουσιαστικά έναν «δείκτη» στο WSDL αρχείο που περιγράφει τις υπηρεσίες διαδικτύου που παρέχει η συγκεκριμένη επιχείρηση.

Η εικόνα 7 δείχνει πως μια επιχείρηση καταχωρεί το WSDL αρχείο της στο UDDI,



Εικόνα 11: Καταχώρηση WSDL στο UDDI

και η εικόνα 8 πως μια άλλη επιχείρηση μπορεί να έχει πρόσβαση σε αυτή την πληροφορία.



Εικόνα 12: Ανακάλυψη υπηρεσιών διαδικτύου (WSDL) περιγραφή μέσω UDDI

4

Σχεδίαση Συστήματος

Το σύστημά μας αποτελείται από την Κεντρική Βάση Δεδομένων (ΚΒΔ) η οποία είναι εγκατεστημένη στον κεντρικό υπολογιστή (Server) του συστήματος, την φορητή εφαρμογή η οποία εκτελείται στην φορητή συσκευή του χρήστη, την κεντρική εφαρμογή και τις υπηρεσίες Web Service. Προκειμένου ο χρήστης να κάνει χρήση των λειτουργιών του συστήματος απαιτείται η επικοινωνία είτε μεταξύ των φορητών εφαρμογών είτε μεταξύ της φορητής εφαρμογής και της Κεντρικής Εφαρμογής με στόχο την καταγραφή δεδομένων στην ΚΒΔ. Στη συνεχεία θα αναφερθούμε αναλυτικά στον σχεδιασμό και τις λειτουργίες του συστήματος για κάθε τμήμα του ξεχωριστά αλλά και σε συνδυασμό μεταξύ τους.

4.1 Οι Βάσεις Δεδομένων

Για την αποτελεσματική λειτουργία του συστήματος αναπτύχθηκαν δύο βάσεις δεδομένων, η Κεντρική Βάση Δεδομένων SystemData και η τοπική Βάση Δεδομένων της φορητής συσκευής TimeTableDB που λειτουργούν κάτω από τον MS SQL Server (SQLEXPRESS) και τον MS SQL Server Compact Edition (SQL Server Mobile) αντίστοιχα. Η βάση SystemData είναι εγκαταστημένη στον κεντρικό υπολογιστή του συστήματος και χρησιμοποιείται για την επικοινωνία με τα Web Services. Για τον λόγο αυτό στη συνέχεια θα τον αποκαλούμε και ως «απομακρυσμένο» κεντρικό υπολογιστή, αφού η πρόσβαση σε αυτόν θα γίνεται μέσω Web Services από διάφορα σημεία. Η βάση δεδομένων TimeTableDB αποθηκεύεται τοπικά μαζί με την εφαρμογή στην φορητή συσκευή, η οποία διαθέτει

λειτουργικό σύστημα Windows Mobile Professional 6.5. Η βάση αυτή χρησιμεύει για την ανταλλαγή δεδομένων με την εφαρμογή αλλά και με την βάση SystemData μέσω των Web Services.

4.1.1 Η Κεντρική Βάση Δεδομένων SystemData

Ο σχεδιασμός και στη συνέχεια η δημιουργία της Βάση Δεδομένων SystemData αποτελεί αποτέλεσμα έρευνας που πραγματοποιήθηκε πάνω στις απαιτήσεις των χρηστών και στις απαιτήσεις του συστήματός μας. Πιστοί στα πρότυπα της τεχνολογίας και της τεχνολογίας λογισμικού, ακολουθήσαμε όλες τις προβλεπόμενες διαδικασίες για το σχεδιασμό και την υλοποίηση μιας λειτουργικής και αποτελεσματικής ΒΔ που εξυπηρετεί το είδος και τις σχέσεις που αναπτύσσονται μεταξύ των δεδομένων που πρόκειται να αποθηκεύσουμε.

4.1.1.1 Πίνακες της Κεντρικής Βάσης Δεδομένων SystemData

Η ΒΔ SystemData αποτελείται από τους εξής πίνακες:

Users: Στο πίνακα Users αποθηκεύονται τα γενικά και ειδικά στοιχεία των χρηστών του συστήματος που είναι σχεδόν ίδια για κάθε χρήστη (π.χ. κωδικός χρήστη (User_ID), κωδικός πρόσβασης (password), κ.α.). Οι χρήστες του συστήματος είναι συγκεκριμένοι: Ιατρικό προσωπικό, Νοσηλευτικό προσωπικό, Διοικητικό προσωπικό.

Address: Στον συγκεκριμένο πίνακα αποθηκεύονται μία ή περισσότερες διευθύνσεις των χρηστών του συστήματος.

Phone_Numbers: Στον συγκεκριμένο πίνακα αποθηκεύονται ένας ή περισσότεροι τηλεφωνικοί αριθμοί των χρηστών του συστήματος.

Declared_Duties: Σε αυτό τον πίνακα αποθηκεύονται οι τύποι των εργασιών στις οποίες μπορεί να συμμετέχει οποιοσδήποτε χρήστης του συστήματος. Το σύστημά μας παρέχει την δυνατότητα καθορισμού των «τύπων εργασίας» (π.χ. Εφημερίες, Καθημερινά Ιατρεία, Βάρδιες, κ.α.) από την γραμματεία ή και την Διεύθυνση σύμφωνα με τον τρόπο οργάνωσης της εκάστοτε Νοσοκομειακής Μονάδας. Για κάθε τύπο εργασίας που δηλώνει ο χρήστης, αυτή εντάσσεται στο πρόγραμμά του και ενημερώνεται για τις τροποποιήσεις που μπορούν να συμβούν.

Declared_Locations: Σε αυτό τον πίνακα αποθηκεύονται οι «Τοποθεσίες» στις οποίες μπορεί να εργάζεται οποιοσδήποτε χρήστης του συστήματος. Μερικά παραδείγματα τοποθεσιών είναι τα εξής: Νοσοκομεία τα οποία επισκέπτεται ο εργαζόμενος στα πλαίσια τις εργασίας του, τμήματα της Νοσοκομειακής μονάδας (με την χωροταξική έννοια), κ.α. Το σύστημά μας παρέχει την δυνατότητα καθορισμού των «Τοποθεσιών» από την γραμματεία ή και την Διεύθυνση σύμφωνα με τον τρόπο οργάνωσης της εκάστοτε Νοσοκομειακής Μονάδας. Για κάθε «Τοποθεσία» που δηλώνει ο χρήστης, κατά την εγγραφή του στο σύστημα, αυτή εντάσσεται στο πρόγραμμά του και ενημερώνεται για τις τροποποιήσεις που μπορούν να συμβούν.

ChangeList: Ο πίνακας αυτός περιέχει τα χαρακτηριστικά στοιχεία των εργασιών που έχουν δημοσιευτεί από τους χρήστες. Ουσιαστικά πρόκειται για μια λίστα με εργασίες τις οποίες οι κάτοχοί τους δεν μπορούν να εκπληρώσουν και επιθυμούν να τις παραχωρήσουν σε κάποιον από τους συναδέλφους τους. Αυτοί με την σειρά τους μπορούν να εντάξουν στο πρόγραμμά τους μια ή περισσότερες δημοσιευμένες εργασίες που έχουν την δυνατότητα και ικανότητα να εκπληρώσουν, προκειμένου να εξυπηρετήσουν τους αρχικούς κατόχους των εργασιών.

Department: Εδώ αποθηκεύονται όλα τα τμήματα στα οποία μπορεί να ανήκει και να εργάζεται κάποιος εγγεγραμμένος χρήστης (π.χ. Καρδιολογικό τμήμα, Τμήμα Επειγόντων Περιστατικών, Λογιστικό Τμήμα, κ.α.).

User_Team: Εδώ αποθηκεύονται όλες οι ομάδες των εργαζομένων στις οποίες μπορεί να ανήκει κάποιος εγγεγραμμένος χρήστης (π.χ. Γιατροί Καρδιολόγοι, νοσηλευτές, διοικητικοί υπάλληλοι.

Item: Ο πίνακας Item χρησιμοποιείται για την αποθήκευση των χαρακτηριστικών στοιχείων που είναι κοινά για κάθε αντικείμενο που μπορεί να μεταφερθεί μεταξύ των χρηστών. Τα αντικείμενα αυτά είναι συγκεκριμένα: Program (Το πρόγραμμα εφημεριών-καθηκόντων), Message (Μηνύματα Επικοινωνίας), Comment.

Program: Σε αυτό τον πίνακα αποθηκεύονται τα εξειδικευμένα στοιχεία που αφορούν και χαρακτηρίζουν μόνο τα αντικείμενα τύπου «Πρόγραμμα» (Program).

Message: Σε αυτό τον πίνακα αποθηκεύονται τα εξειδικευμένα στοιχεία που αφορούν και χαρακτηρίζουν μόνο τα αντικείμενα τύπου «Μήνυμα» (Message).

Comment: Σε αυτό τον πίνακα αποθηκεύονται τα εξειδικευμένα στοιχεία που αφορούν και χαρακτηρίζουν μόνο τα αντικείμενα τύπου «Σχόλιο» (Message).

Applications: Στον συγκεκριμένο πίνακα αποθηκεύονται οι αιτήσεις αλλαγής ή ακύρωσης των εφημεριών/εργασιών που καταθέτουν οι χρήστες μέσω των φορητών τους συσκευών.

Modifications: Ο πίνακας αυτός αποθηκεύει τα στοιχεία των αλλαγών που προτείνουν οι χρήστες του συστήματος για την πραγματοποίηση της αλλαγής των εργασιών τους.

Locations: Ο συγκεκριμένος πίνακας αποθηκεύει μια λίστα από τοποθεσίες που μπορούν να επιλέξουν οι χρήστες. Η λίστα αυτή εξαρτάται από την εμβέλεια της νοσοκομειακής μονάδας και του συστήματος.

Eidikothtes: Ο πίνακας αυτός αποθηκεύει τις ειδικότητες των εργαζομένων που συμπεριλαμβάνονται στο σύστημα της αυτόματης διαχείρισης εφημεριών και καθηκόντων.

Bathmides: Ο συγκεκριμένος πίνακας αποθηκεύει το σύνολο των βαθμίδων στο οποίο μπορεί να ανήκει κάποιος από τους εγγραμμένους χρήστες του συστήματος.

Sum_Duties: Ο συγκεκριμένος πίνακας αποθηκεύει το σύνολο των εργασιών που μπορεί να εκτελεί κάποιος από τους εγγραμμένους χρήστες του συστήματος.

4.1.1.2 Σχεσιακό Μοντέλο της Κεντρικής Βάσης Δεδομένων

Στα στάδια της σχεδίασης του συστήματός μας και κατά την ανάλυση των απαιτήσεών του, δημιουργήσαμε το σχεσιακό μοντέλο, την κανονικοποίηση και το μοντέλο οντοτήτων-συσχετίσεων της ΒΔ, τα οποία θα παρουσιαστούν στη συνέχεια μαζί με τα πεδία κάθε πίνακα.

Το σχεσιακό μοντέλο περιγράφει τη Βάση Δεδομένων και οργανώνει τις εγγραφές σύμφωνα με τις μεταξύ τους σχέσεις. Γι αυτό το λόγο μια Βάση Δεδομένων σχεδιασμένη

σύμφωνα με ένα σχεσιακό μοντέλο, μπορεί εύκολα να υλοποιηθεί με ένα μοντέλο Οντοτήτων - Συσχετίσεων. Στις σχεσιακές Βάσεις Δεδομένων όπως είναι και η SystemData, οι εγγραφές οργανώνονται σε πίνακες. Οι πίνακες σε μια σχεσιακή Βάση Δεδομένων, αποτελούνται από μια ή περισσότερες στήλες που αντιστοιχούν σε τιμές πεδίων (ή στα χαρακτηριστικά των μοντέλων Οντοτήτων - Συσχετίσεων) και από γραμμές που αντιστοιχούν σε εγγραφές για αυτά τα πεδία.

Τα ονόματα των πινάκων και των στηλών, αναγράφονται με την ονομασία που υπάρχει στην ΒΔ (αγγλική γλώσσα), για αποδοτικότερη όμως ανάγνωση, αναγράφονται τα ονόματα των στηλών κάθε πίνακα και στην ελληνική γλώσσα. Στη συνέχεια ακολουθεί το σχεσιακό μοντέλο της ΒΔ SystemData στο οποίο παρουσιάζονται και οι πίνακες της ΒΔ με τα πεδία τους (στήλες) καθώς και οι περιορισμοί ακεραιότητας που διέπουν το μοντέλο αυτό.

Users (User_ID/κωδικό όνομα χρήστη, User_Team/ομάδα χρήστη, name/όνομα, surname/επώνυμο,username/όνομα εισόδου, password/κωδικός εισόδου, ΑΜΚΑ, status/κατάσταση σύνδεσης, Department/τμήμα, Position/θέση, Specialty/Ειδικότητα, Rung/Βαθμίδα)

Address (User_ID/κωδικός χρήστη, νομός, δήμος, city/πόλη, address/οδός και αριθμός, Τ.Κ., περιοχή, χώρα)

Phone Numbers (User_ID /κωδικός χρήστη, αριθμός τηλεφώνου, κηνιτό, FAX)

Department (Department _Name/όνομα τμήματος, Department_ID)

User_Team (Team_Name/όνομα ομάδας, Team_ID)

Modifications (Program_ID, Date/ημερομηνία, Duty Type/τύπος εργασίας, Duty Start Time/ώρα έναρξης, Duty End Time/ώρα λήξης, User ID /κωδικός χρήστη)

Applications (Program_ID, User ID /κωδικός χρήστη,App_Type/τύπος αίτησης)

Locations (Location_Name)

Sum_Duties (Duty_Type_Name)

Eidikothtes (Eidikothta_Name)

Bathmides (Onoma_Bathmidas)

Item (Item_ID/κωδικός αντικειμένου, Creator_ID/κωδικός δημιουργού, Receiver_ID/κωδικός παραλήπτη)

Program (Item_ID/κωδικός αντικειμένου, Date/ημερομηνία, Duty Type/τύπος εργασίας, Duty Start Time/ώρα έναρξης, Duty End Time/ώρα λήξης, Location/τοποθεσία, User ID /κωδικός χρήστη, Program Name/όνομα προγράμματος)

Change List (User_ID /κωδικός χρήστη, Item_ID/κωδικός αντικειμένου)

Declared Duties (User_ID /κωδικός χρήστη, Duty Type/τύπος εργασίας)

Declared Locations (User_ID /κωδικός χρήστη, Location/τοποθεσία)

Message (Item_ID/κωδικός αντικειμένου, Creator_ID/κωδικός δημιουργού, content/περιεχόμενο)

Comment (Item_ID κωδικός αντικειμένου, Creator_ID/κωδικός δημιουργού, commented_item_ID/κωδικός σχολιασμένου αντικειμένου, comment text/περιεχόμενο)

Υπόμνημα:

Πρωτεύον κλειδί

Εναλλακτικό κλειδί

Ξένο κλειδί

4.1.1.3 Περιορισμοί ακεραιότητας

Οι περιορισμοί ακεραιότητας (integrity constraints) αποτελούν αναπόσπαστο κομμάτι του σχεσιακού μοντέλου και πρέπει να ισχύουν πάντα για κάθε στιγμιότυπο της ΒΔ (σε κάθε σχέση της ΒΔ). Τότε ένα στιγμιότυπο ονομάζεται έγκυρο (valid). Οι περιορισμοί ακεραιότητας επαληθεύονται κάθε φορά που πραγματοποιούνται αλλαγές στα δεδομένα (εισαγωγή, διαγραφή, ενημέρωση).

Για το συγκεκριμένο σχεσιακό μοντέλο οι περιορισμοί ακεραιότητας έχουν ως εξής:

Κενές τιμές: για την ιδανική λειτουργία του συστήματος, σε κανένα πεδίο δεν επιτρέπεται η ύπαρξη κενών τιμών. Στην φάση όμως της δημιουργίας και των δοκιμών θα είμαστε λιγότερο αυστηροί προκειμένου να εξασφαλίσουμε και να είμαστε σίγουροι για τις υπόλοιπες λειτουργίες της Βάσης Δεδομένων.

Ακεραιότητα οντοτήτων: ο σχεδιασμός του σχεσιακού μοντέλου είναι τέτοιος ώστε κάθε γραμμή, κάθε πίνακας, να προσδιορίζεται μοναδικά από το πρωτεύον κλειδί.

Έχουμε λάβει υπ' όψιν μας την ακεραιότητα αναφορών, προκειμένου κάθε κλειδί K ενός πίνακα (A) που αποτελεί χαρακτηριστικό ενός άλλου πίνακα (B), να είναι και ξένο κλειδί του (B).

4.1.1.4 Κανονικοποίηση της Κεντρικής Βάσης Δεδομένων

Τα προβλήματα που είναι πιθανό να παρουσιασθούν κατά τη διαδικασία της υλοποίησης του σχεδιασμού μιας βάσης δεδομένων είναι η περιττή (άσκοπη) επανάληψη πληροφοριών, που είναι γνωστή με τον όρο redundancy, καθώς και δυσκολίες στην ενημέρωση της βάσης δεδομένων. Τα παραπάνω προβλήματα είναι γνωστά ως πλεονασμοί δεδομένων και ανωμαλίες ενημέρωσης και για να αντιμετωπιστούν με επιτυχία, θα πρέπει να διασπάσουμε τις μεγάλες σχέσεις σε μικρότερες. Αυτό γίνεται με τη διαδικασία της κανονικοποίησης, έτσι ώστε η βάση δεδομένων να είναι έτοιμη για καταχώριση στοιχείων.

Η κανονικοποίηση (normalization) είναι μια τεχνική που ασχολείται με την ανάλυση των σχέσεων (συσχετίσεων) σε μια βάση δεδομένων όπου κάνουμε μετατροπή των αρχικών μεγάλων σχέσεων σε μικρότερες. Στη συνέχεια προσπαθούμε να καταγράψουμε ορισμένες από τις συναρτησιακές εξαρτήσεις που προκύπτουν:

Username, password → κωδικός χρήστη, ομάδα χρήστη, όνομα, επώνυμο, ΑΜΚΑ, status, τμήμα

User_ID/Κωδικός χρήστη → νομός, δήμος, πόλη, οδός, αριθμός, Τ.Κ., περιοχή, χώρα, αριθμός τηλεφώνου, θέση, όνομα ομάδας, τμήμα, ειδικότητα, βαθμίδα, κωδικός αντικειμένου (που δημιούργησε), Date/ημερομηνία, Duty Type/τύπος εργασίας, Duty Start Time/ώρα έναρξης, Duty End Time/ώρα λήξης, Location/τοποθεσία, Program Name/όνομα προγράμματος

οδός, αριθμός, Τ.Κ. → User ID/κωδικός χρήστη, νομός, δήμος, πόλη, περιοχή, χώρα

Item ID/κωδικός αντικειμένου → κωδικός χρήστη/δημιουργού, ημερομηνία δημιουργίας, κωδικός παραλήπτη, ομάδα ενδιαφερομένων, κωδικός σχολιασμένου αντικειμένου, App_Type

κωδικός αντικειμένου, κωδικός δημιουργού→ παραλήπτης, περιεχόμενο αντικειμένου (αρχείο)

Το παραπάνω σχεσιακό μοντέλο βρίσκεται σε 1ΚΜ (Κανονική Μορφή) αφού δεν υπάρχει ως τιμή γνωρίσματος μια πλειάδα, ένα σύνολο τιμών ή συνδυασμός των δύο, ώστε

να προκύψουν σύνθετα χαρακτηριστικά ή εμφωλευμένες σχέσεις. Επίσης το μοντέλο μας ανήκει και στην 2ΚΜ αφού για κάθε πίνακα ισχύει ένα από τα εξής:

- το πρωτεύον κλειδί αποτελείται από ένα και μόνο χαρακτηριστικό,
- ο πίνακας δεν έχει χαρακτηριστικά που δεν αποτελούν κλειδί

  (all-key relation), ή
- κάθε χαρακτηριστικό που δεν είναι κλειδί, είναι πλήρως συναρτησιακά εξαρτώμενο από το πρωτεύον κλειδί.

Επιπλέον υπάρχουν αρκετές μεταβατικές εξαρτήσεις, όπως:

κωδικός χρήστη → κωδικός αντικειμένου

κωδικός αντικειμένου → ημερομηνία δημιουργίας

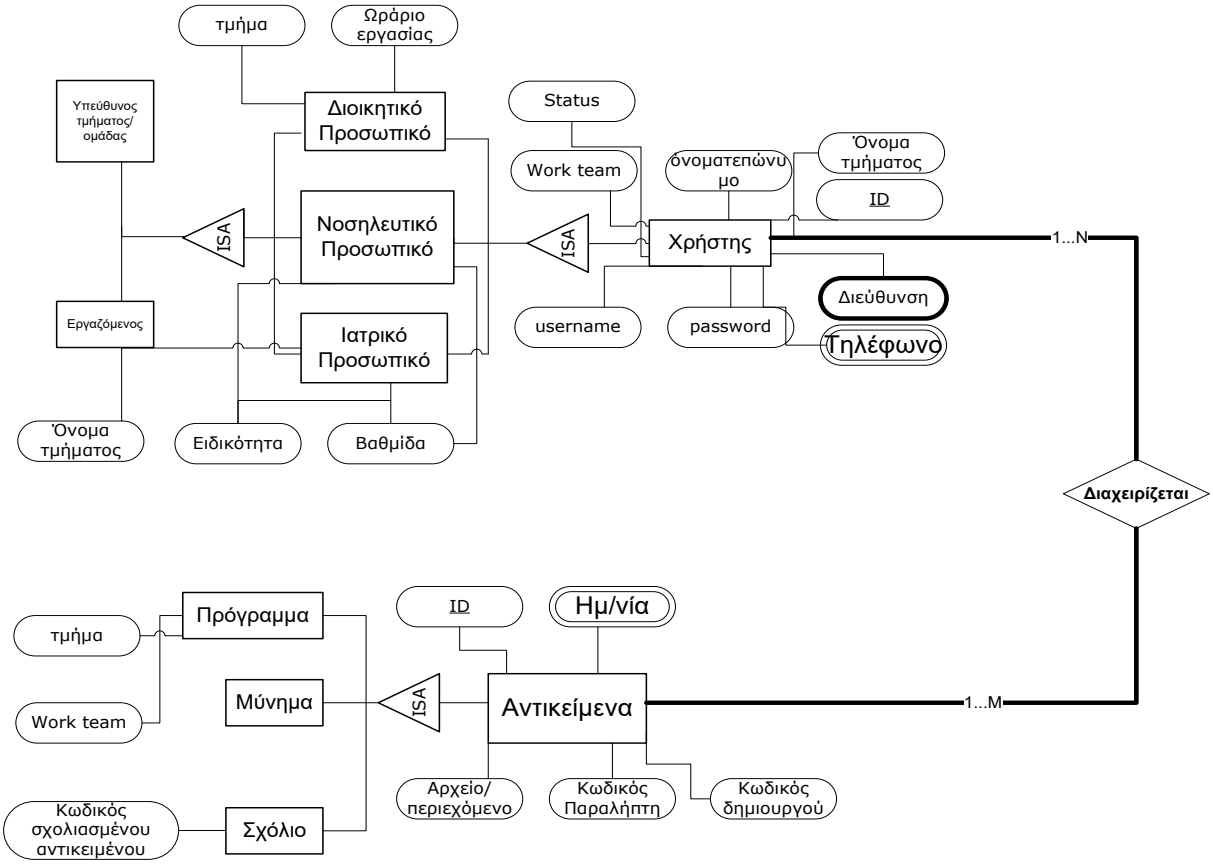
Χαρακτηριστικό είναι το γεγονός ότι ορισμένοι από τους πίνακες ανήκουν και στην **3ΚΜ.** Μια ακόμη μορφή κανονικοποίησης, είναι η κανονική μορφή **Boyce-Codd**, στην οποία βρίσκονται οι περισσότεροι από τους πίνακες του σχήματός μας. Εξετάζοντας όλες τις παραπάνω συναρτησιακές εξαρτήσεις, προκύπτει ότι για κάθε **X→Y** εξάρτηση, το **X** αποτελεί κάποιο κλειδί.

4.1.1.5 Μοντέλο Οντοτήτων – Συσχετίσεων

Το μοντέλο οντοτήτων-συσχετίσεων (μοντέλο Ο/Σ - ER model) είναι ένα αφαιρετικό μοντέλο δεδομένων του συστήματός μας. Σκοπός του είναι να περιγράψει τα δεδομένα που πρόκειται να αποθηκευτούν στη Βάση Δεδομένων.

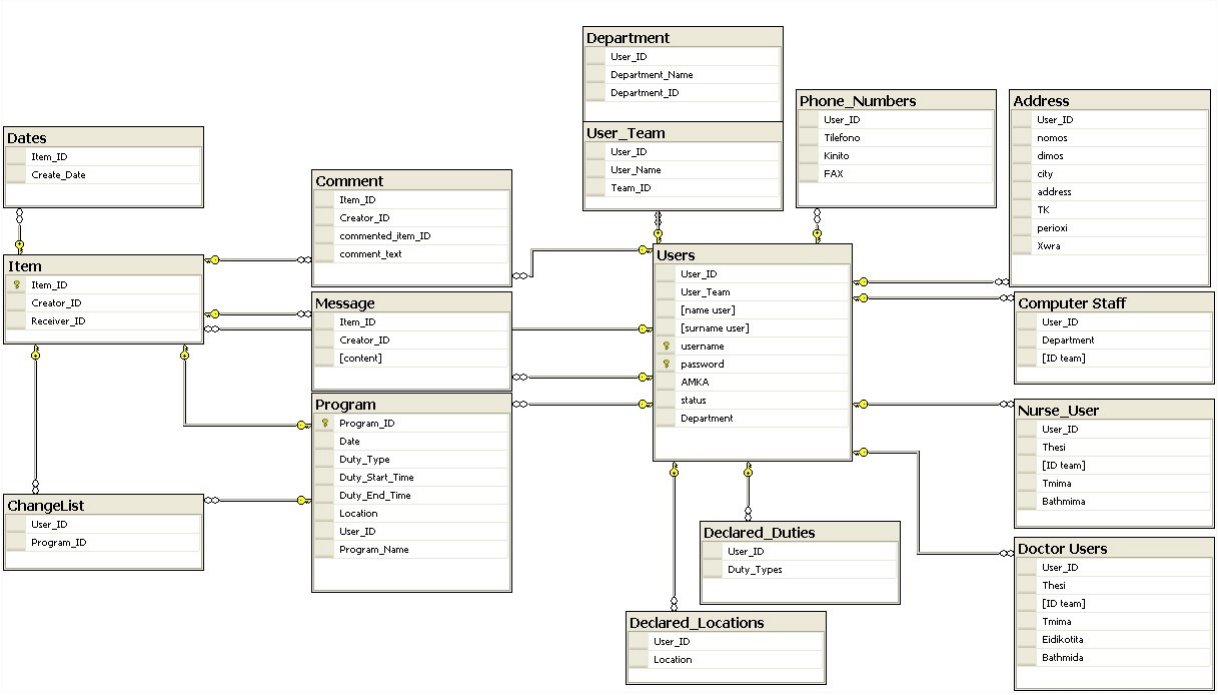
Το διάγραμμα οντοτήτων-συσχετίσεων που παρουσιάζεται στη συνέχεια ακολουθεί το Σχεσιακό Μοντέλο και την κανονικοποίηση των δεδομένων που θα αποθηκεύονται στην ΒΔ. Με τον τρόπο αυτό προσπαθούμε:

- Να εξαλειφθεί ο πλεονασμός των δεδομένων
- Να εξασφαλιστεί η σύνδεση μεταξύ των πινάκων ΒΔ χωρίς απώλειες δεδομένων και
- Να διατηρηθούν οι εξαρτήσεις πινάκων και στηλών της ΒΔ



Εικόνα 13: Διάγραμμα Οντοτήτων-Συσχετίσεων της Κεντρικής Βάσης Δεδομένων SystemData

Αποτέλεσμα της φάσης του σχεδιασμού που προηγήθηκε είναι η δημιουργία μιας ΒΔ που ακολουθεί πιστά τις τεχνικές και θεωρητικές προδιαγραφές του. Το παρακάτω διάγραμμα είναι το τελικό διάγραμμα της Βάσης Δεδομένων που δημιουργήθηκε.



Εικόνα 14: Τελικό διάγραμμα των πινάκων της Κεντρικής Βάσης Δεδομένων SystemData

4.1.2 Η Τοπική Βάση Δεδομένων TimeTableDB

Η Βάση Δεδομένων TimeTableDB βρίσκεται αποθηκευμένη στην φορητή συσκευή και για τον λόγο αυτό σε αρκετές περιπτώσεις θα γίνεται αναφορά σε αυτή ως «τοπική ΒΔ». Τα στοιχεία που πρέπει να αποθηκευτούν στην φορητή συσκευή είναι κυρίως αυτά που αφορούν τον κάθε χρήστη ξεχωριστά και έχουν σχέση μόνο με το προσωπικό του πρόγραμμα εφημεριών-καθηκόντων, συνεπώς διαθέτει λιγότερους πίνακες από αυτούς που υπάρχουν στον απομακρυσμένο κεντρικό υπολογιστή του συστήματος.

4.1.2.1 Οι πίνακες της Τοπικής Βάσης Δεδομένων

Ορισμένοι από τους πίνακες της τοπικής ΒΔ είναι κοινοί, όπως οι πίνακες Program, Declared_Duties, Declared_Locations, ChangeList περιέχουν όμως δεδομένα που σχετίζονται μόνο με τον χρήστη της εφαρμογής. Στην κατηγορία με τους κοινούς πίνακες μπορούμε να εντάξουμε και τον πίνακα UserData ο οποίος αντιστοιχεί στον πίνακα Users της ΒΔ SystemData. Όμοια με τους υπολοίπους, ο πίνακας UserData αποθηκεύει την εγγραφή του πίνακα Users που αντιστοιχεί στον συγκεκριμένο χρήστη της εφαρμογής. Οι υπόλοιποι πίνακες είναι οι:

CurrentState: Ο συγκεκριμένος πίνακας αποθηκεύει τις ρυθμίσεις του χρήστη που έχουν σχέση με την εμφάνιση ή την απόκρυψη στηλών από τον πίνακα του προγράμματος εφημεριών-καθηκόντων.

ViewChanges: Ο πίνακας αυτός αποθηκεύει τα απαραίτητα στοιχεία προκειμένου ο εκάστοτε χρήστης της εφαρμογής να γνωρίζει ποιός συνάδελφός του έχει αποδεχτεί την εφημερία που προηγουμένως είχε ο ίδιος δημοσιεύσει.

4.1.2.2 Σχεσιακό μοντέλο της Τοπικής Βάσης Δεδομένων

Για τους λόγους που αναφέραμε στην περίπτωση της ΒΔ SystemData και που παρουσιάσαμε στην προηγούμενη παράγραφο, έτσι και για την ΒΔ TimeTableDB της φορητής συσκευής έχουμε δημιουργήσει το σχεσιακό μοντέλο καθώς και την κανονικοποίηση της Βάσης.

UserData (User_ID/κωδικό όνομα χρήστη, User_Team/ομάδα χρήστη, name/όνομα, surname/επώνυμο, username/όνομα εισόδου, password/κωδικός εισόδου, AMKA, status/κατάσταση σύνδεσης, Department/τμήμα)

Program (Item_ID/κωδικός αντικειμένου, Date/ημερομηνία, Duty Type/τύπος εργασίας, Duty Start Time/ώρα έναρξης, Duty End Time/ώρα λήξης, Location/τοποθεσία, User ID /κωδικός χρήστη, Program Name/όνομα προγράμματος)

Change List (User_ID /κωδικός χρήστη, Item_ID/κωδικός αντικειμένου)

Declared Duties (User_ID /κωδικός χρήστη, Duty Type/τύπος εργασίας)

Declared Locations (User_ID /κωδικός χρήστη, Location/τοποθεσία)

CurrentState (Element/αντικείμενο, Value/τιμή)

ViewChanges (Item_ID/κωδικός αντικειμένου, Receiver_ID/κωδικός παραλήπτη)

Υπόμνημα:

Πρωτεύον κλειδί

Εναλλακτικό κλειδί

Ξένο κλειδί

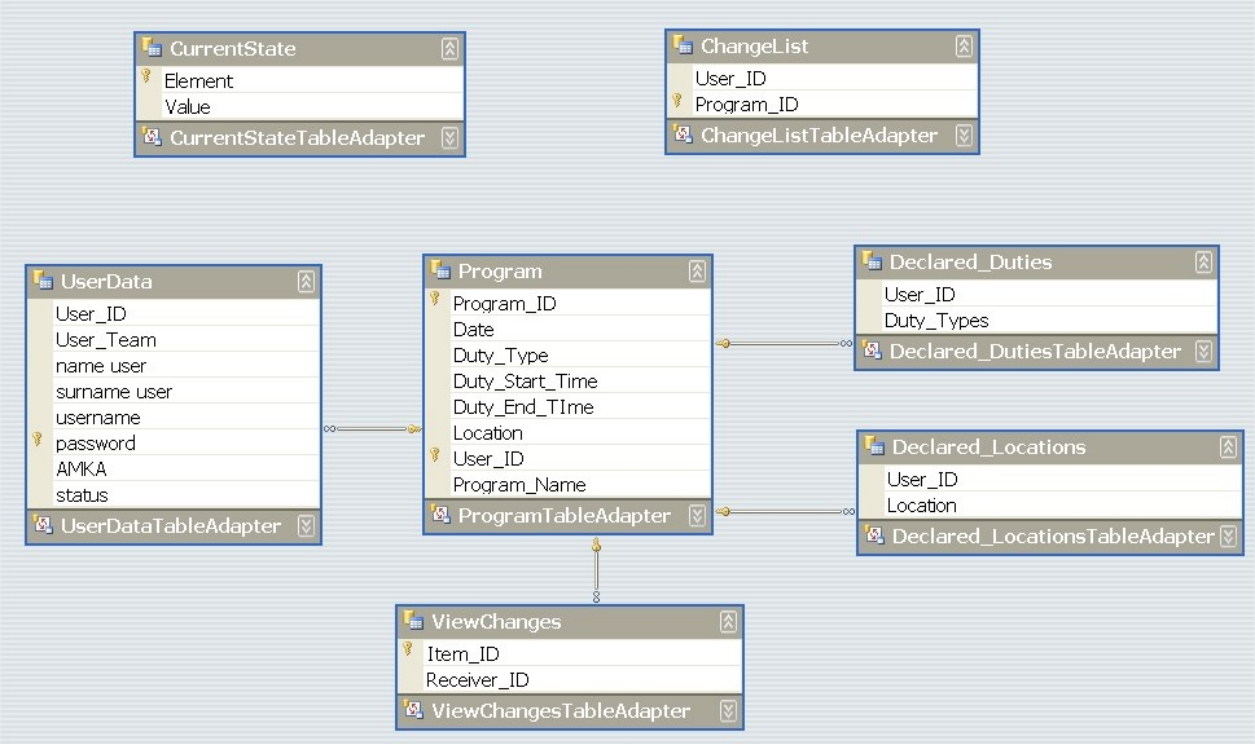
4.1.2.3 Κανονικοποίηση Τοπικής Βάσης Δεδομένων

Password → κωδικός χρήστη, ομάδα χρήστη, όνομα, επώνυμο,username, AMKA, status

User_ID/Κωδικός χρήστη → ομάδα χρήστη, τμήμα, ειδικότητα, βαθμίδα, κωδικός αντικειμένου/προγράμματος , Date/ημερομηνία, Duty Type/τύπος εργασίας, Duty Start Time/ώρα έναρξης, Duty End Time/ώρα λήξης, Location/τοποθεσία, Program Name/όνομα προγράμματος, Receiver_ID/κωδικός παραλήπτη

Item ID/κωδικός αντικειμένου → κωδικός χρήστη/δημιουργού, κωδικός χρήστη/παραλήπτη, Date/ημερομηνία, Duty Type/τύπος εργασίας, Duty Start Time/ώρα έναρξης, Duty End Time/ώρα λήξης, Location/τοποθεσία, Program Name/όνομα προγράμματος

Στη συνέχεια ακολουθεί το διάγραμμα της ΒΔ TimeTableDB όπως προκύπτει από το σχεσιακό μοντέλο και την διαδικασία της κανονικοποίησης.



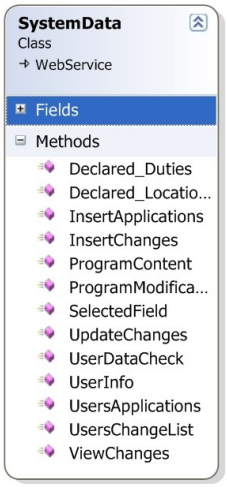
Εικόνα 15: Διάγραμμα της τοπικής Βάσης Δεδομένων TimeTableDB που βρίσκεται στην φορητή συσκευή

4.2 Το Web Service του συστήματος

Η επικοινωνία των εφαρμογών που απαρτίζουν το σύστημα πραγματοποιείται από τις υπηρεσίες του Web Service. Το Web Service είναι εγκατεστημένο στον κεντρικό υπολογιστή

(Server) του συστήματος. Ρόλος του είναι να εκτελεί τις εντολές που προέρχονται από την φορητή και την κεντρική εφαρμογή. Οι εντολές αυτές έχουν σχέση με τη διαγραφή, την ανάκτηση, την αποθήκευση και την ανανέωση των δεδομένων από και προς την Κεντρική Βάση Δεδομένων (ΚΒΔ) που είναι επίσης εγκατεστημένη στον κεντρικό υπολογιστή. Δηλαδή, το Web Service αποτελεί τον μεσάζοντα στην διαδικασία επικοινωνίας των εκάστοτε εφαρμογών με την ΚΒΔ.

Το Web Service του συστήματος ονομάζεται SystemData_WebService και αποτελείται από μεθόδους οι οποίες πραγματοποιούν τις απαραίτητες ενέργειες προκειμένου να επιτευχθούν με τον σωστό τρόπο και χωρίς λάθη οι ανταλλαγές των δεδομένων μεταξύ των εφαρμογών και της ΚΒΔ. Το Web Service αποτελείται από 13 μεθόδους (Web Methods) οι οποίες είναι υπεύθυνες για το σύνολο των υπηρεσιών του συστήματος. Οι μέθοδοι αυτοί βρίσκονται στην κλάση SystemData.



Εικόνα 16: Η κλάση SystemData με τις Web Methods

4.2.1 Υπηρεσίες και μέθοδοι του Web Service

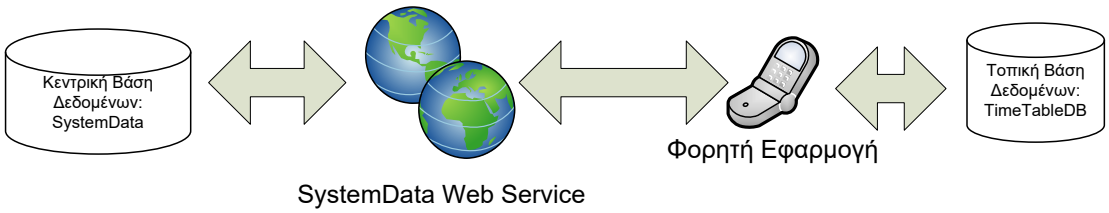
4.2.1.1 Υπηρεσία αρχικοποίησης και ανανέωσης προγράμματος-Η μέθοδος ProgramContent

Η μέθοδος ProgramContent καλείται από την φορητή εφαρμογή κατά την αρχικοποίηση της φόρμας «Πρόγραμμα». Πραγματοποιεί την ανάκτηση των προγραμμάτων από τον πίνακα Program της ΚΒΔ τα οποία επιστρέφει στην φορητή εφαρμογή με σκοπό την παρουσίαση τους στον χρήστη και κατ' επέκταση τη διαχείρισή τους. Η κλήση της μεθόδου αυτής πραγματοποιείται για την αρχικοποίηση και την ανανέωση των στοιχείων του προγράμματος εφημεριών και γενικών καθηκόντων που παρουσιάζονται στο χρήστη της

φορητής εφαρμογής. Τα δεδομένα αυτά στη συνέχεια θα καταγράψουν τις κατάλληλες μεθόδους της φορητής εφαρμογής, στον αντίστοιχο πίνακα Program της τοπικής ΒΔ.

4.2.1.1.1 Υπηρεσία αρχικοποίησης και ανανέωσης στοιχείων χρήστη- Οι μέθοδοι Declared_Locations και Declared_Duties

Ομοίως με την μέθοδο ProgramContent, οι μέθοδοι Declared_Locations και Declared_Duties καλούνται από την φορητή εφαρμογή κατά την αρχικοποίηση της φόρμας «Πρόγραμμα». Οι συγκεκριμένες μέθοδοι είναι υπεύθυνες για την ανάκτηση των αντίστοιχων πινάκων Declared_Locations και Declared_Duties που υπάρχουν στην ΚΒΔ και την επιστροφή των δεδομένων αυτών στην φορητή εφαρμογή. Με τον τρόπο αυτό γνωρίζουμε τα προγράμματα για τα οποία ενδιαφέρεται να λαμβάνει ενημερώσεις κάθε χρήστης της φορητής εφαρμογής.



Εικόνα 17: Διάγραμμα ροής δεδομένων στο σύστημα διαχείρησης εφημεριών/καθηκόντων

4.2.1.1.2 Υπηρεσία παρακολούθησης αλλαγών- Η μέθοδος ViewChanges

Η μέθοδος ViewChanges καλείται κατά την αρχικοποίηση της φόρμας «Παρακολούθηση Αλλαγών» της φορητής εφαρμογής. Σκοπός της συγκεκριμένης μεθόδου είναι να ανακτά δεδομένα από τον πίνακα Item της ΚΒΔ και να τα επιστρέφει στην φορητή εφαρμογή. Η λειτουργία αυτή αποσκοπεί στην ενημέρωση των χρηστών των οποίων οι δημοσιευμένες προς ανταλλαγή εργασίες έχουν γίνει αποδεκτές, για το ποιός χρήστης-συνάδελφος έκανε αποδεκτό το αίτημά τους [Εικόνα 19]..

4.2.1.2 Υπηρεσία δημοσίευσης και ανταλλαγής εργασιών

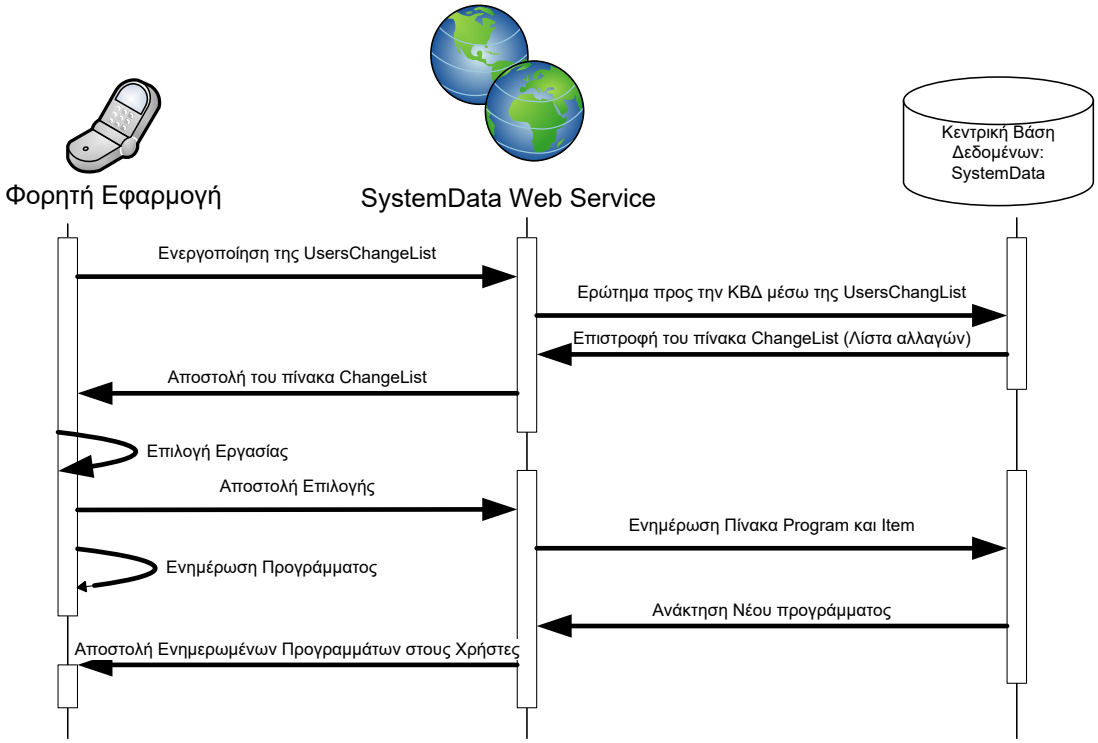
4.2.1.2.1 Η μέθοδος Insert Changes

Η μέθοδος Insert Changes είναι υπεύθυνη για την καταγραφή των ανταλλαγών που επιθυμούν οι χρήστες να πραγματοποιήσουν μεταξύ τους. Η μέθοδος αυτή καλείται από την φόρμα «Κοινοποίηση Εργασίας» της φορητής συσκευής όταν ο χρήστης επιθυμεί να

δημοσιεύσει την εργασία του με σκοπό να την ανταλλάξει με έναν συνάδελφό του. Σε πρώτη φάση η μέθοδος καταγράφει την δημοσίευση της εργασίας στον αντίστοιχο πίνακα ChangeList της ΚΒΔ και στη συνέχεια καταγράφει τον χρήστη που πραγματοποίησε την δημοσίευση ανανεώνοντας τον πίνακα Item της ΚΒΔ [Εικόνα 18],[Εικόνα 19].

4.2.1.2.2 Η μέθοδος UsersChangeList

Η μέθοδος UsersChangeList αποτελεί συνέχεια της μεθόδου InsertChanges και καλείται από την φόρμα «Change List» της φορητής εφαρμογής όταν ο χρήστης επιθυμεί να αποδεχτεί και να συμπεριλάβει στο πρόγραμμά του, μία από τις δημοσιευμένες προς ανταλλαγή εργασίες των συναδέλφων του. Με την κλήση της συνάρτησης αυτής επιστρέφεται στον χρήστη μια λίστα με όλα τα στοιχεία των δημοσιευμένων εργασιών καθώς και τα στοιχεία των χρηστών που τις δημοσίευσαν [Εικόνα 18],[Εικόνα 19].

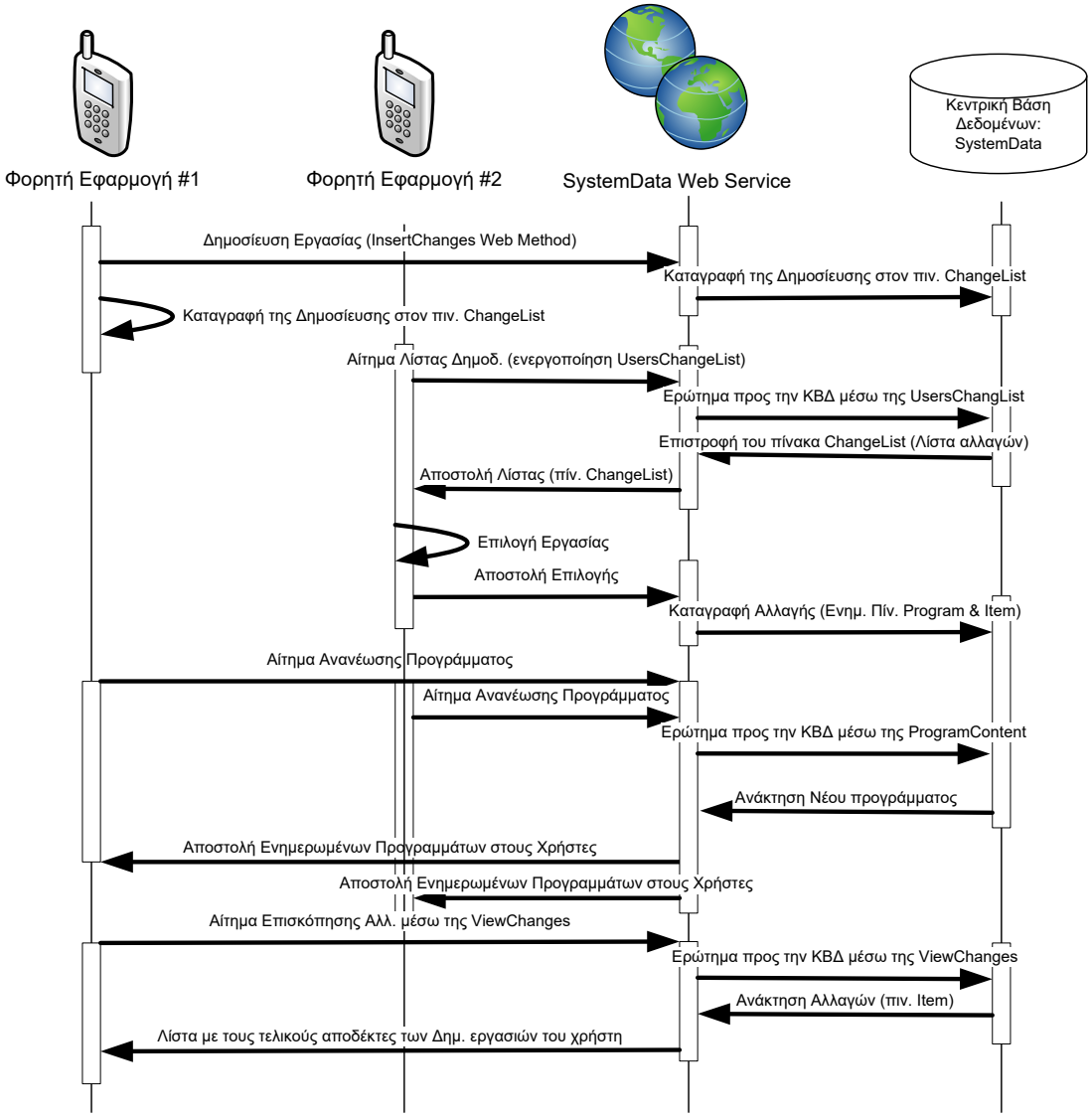


Εικόνα 18: Διάγραμμα της σειράς εκτέλεσης των διαδικασιών για την λειτουργία αποδοχής των δημοσιευμένων εργασιών

4.2.1.2.3 Η μέθοδος UpdateChanges

Η μέθοδος UpdateChanges ολοκληρώνει την υπηρεσία ανταλλαγής εφημεριών μεταξύ συναδέλφων. Όταν ο χρήστης επιλέξει την εργασία που θέλει να αποδεχτεί και να συμπεριλάβει στο πρόγραμμά του, μέσα από την λίστα των δημοσιευμένων εργασιών που εμφανίζεται στην οθόνη της φορητής του εφαρμογής, τότε καλείται η μέθοδος UpdateChanges. Η συγκεκριμένη μέθοδος αναδιαμορφώνει το πρόγραμμα των καθηκόντων

θέτοντας υπεύθυνο για την εκτέλεση μίας εφημερίας ή βάρδιας τον χρήστη που έκανε αποδεκτή την δημοσιευμένη εργασία. Στη συνέχεια διαγράφει από την λίστα των δημοσιευμένων αλλαγών του πίνακα ChangeList (της ΚΒΔ) την εργασία που μόλις έγινε αποδεκτή και τέλος αναδιαμορφώνει τον πίνακα Item ορίζοντας ως τελικό αποδέκτη της εργασίας τον χρήστη που την αποδέχτηκε, μετά την δημοσίευσή της [Εικόνα 19].

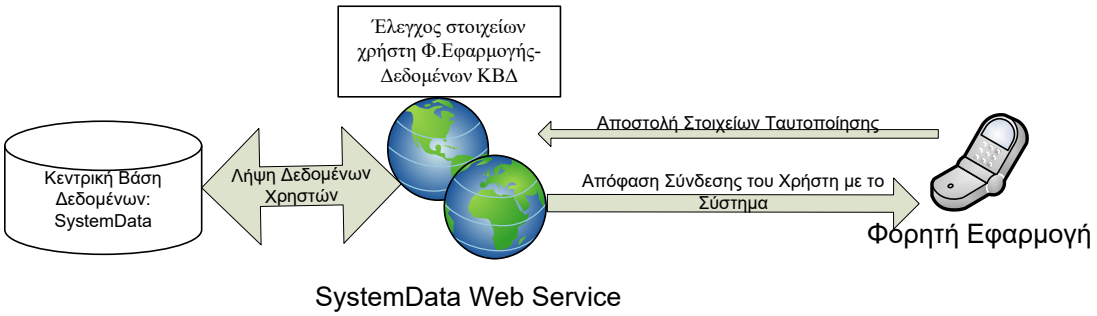


Εικόνα 19: Διάγραμμα υπηρεσίας δημοσίευσης και ανταλλαγής εργασιών

4.2.1.3 Υπηρεσία ταυτοποίησης χρήστη

Η μέθοδος UserDataCheck καλείται από την φόρμα «Σύνδεση Χρήστη» της φορητής εφαρμογής όταν ο χρήστης δοκιμάσει να πραγματοποιήσει την σύνδεση της συσκευής του με το σύστημα. Σκοπός της συνάρτησης UserDataCheck είναι ο έλεγχος των στοιχείων ταυτοποίησης του χρήστη της φορητής εφαρμογής με αυτά που βρίσκονται αποθηκευμένα

στον πίνακα Users της ΚΒΔ. Εάν τα στοιχεία ταυτοποιούνται τότε μπορεί να πραγματοποιηθεί η σύνδεση του χρήστη και της φορητής εφαρμογής με το σύστημα αυτόματης διαχείρισης εφημεριών και γενικών καθηκόντων.



Εικόνα 20: Διάγραμμα ροής δεδομένων κατά τη διαδικασία ταυτοποίησης του χρήστη φορητής εφαρμογής

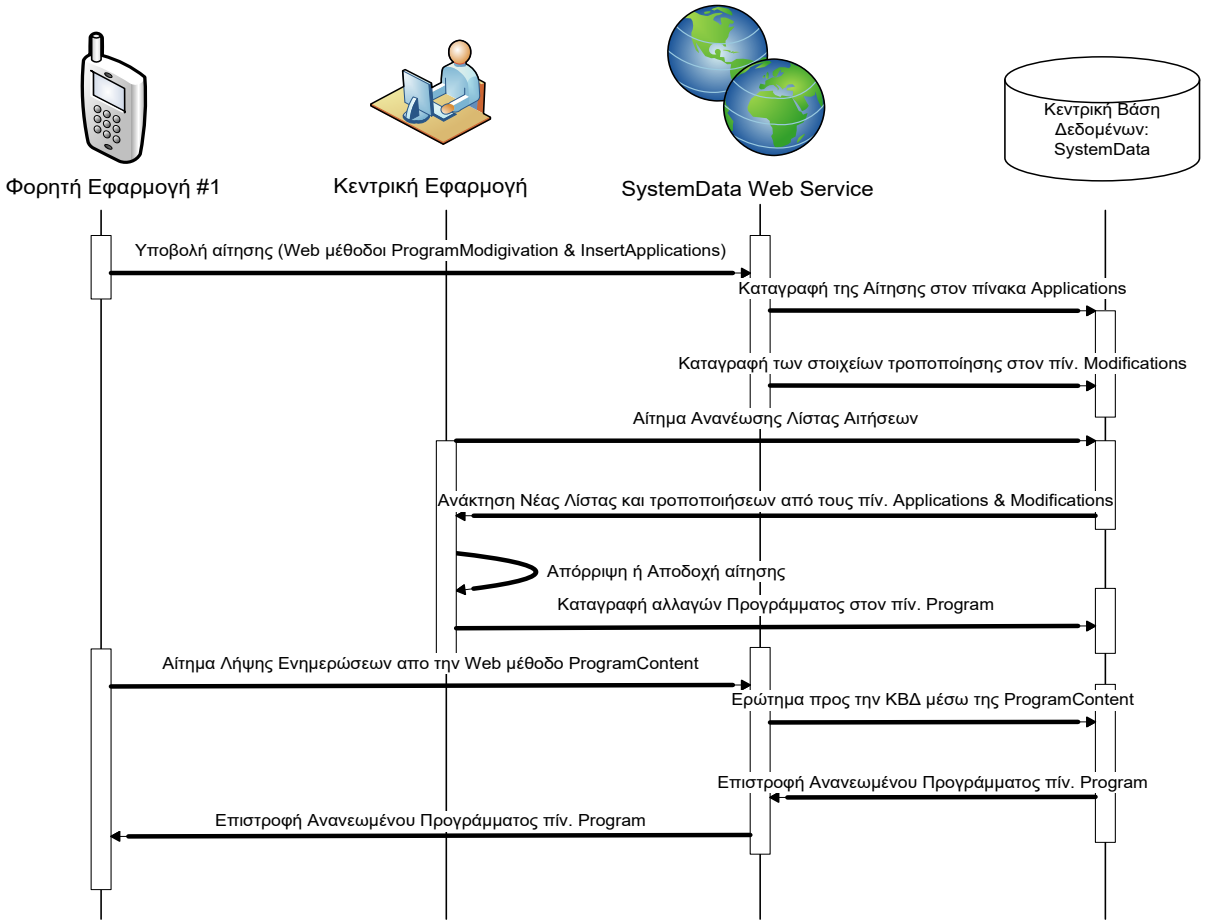
4.2.1.4 Υπηρεσία υποβολής αιτήσεων

4.2.1.4.1 Η μέθοδος InsertApplications

Η μέθοδος InsertApplications καλείται από την φορητή εφαρμογή του χρήστη όταν αυτός επιθυμεί να υποβάλλει αίτηση προς την γραμματεία, δηλαδή του ατόμου που διαχειρίζεται τις αλλαγές των προγραμμάτων, για αλλαγή της ώρας έναρξης ή και λήξης της εργασίας, ακύρωση της εργασίας ή μεταφορά της εργασίας σε άλλη ημερομηνία ή και ώρα. Η συγκεκριμένη μέθοδος πραγματοποιεί την καταγραφή των στοιχείων υποβολής της αίτησης (χρήστης, πρόγραμμα, τύπος αίτησης) στον αντίστοιχο πίνακα (Applications) της ΚΒΔ [Εικόνα 21].

4.2.1.4.2 Η μέθοδος ProgramModification

Η μέθοδος ProgramModification αποτελεί συνέχεια της μεθόδου InsertApplications και ομοίως καλείται όταν ο χρήστης επιθυμεί να υποβάλλει αίτηση αλλαγής των στοιχείων της εργασίας του προς την γραμματεία. Η μέθοδος ProgramModification καταγράφει στον πίνακα Modifications της ΚΒΔ τις αλλαγές των στοιχείων της εργασίας που επιθυμεί/προτείνει ο χρήστης έτσι ώστε να μπορεί στη συνέχεια ο χρήστης της Κεντρικής Εφαρμογής να γνωρίζει τα στοιχεία αυτά προτού απορρίψει ή κάνει αποδεκτή την αίτηση αλλαγής του χρήστη [Εικόνα 21].

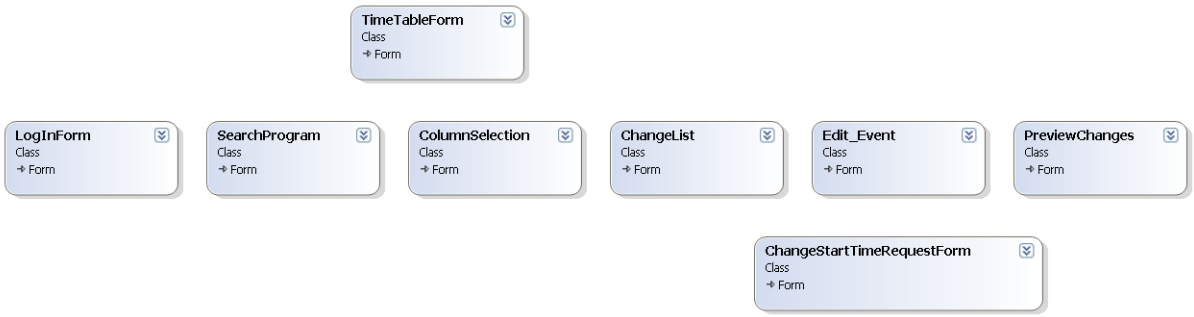


Εικόνα 21: Διάγραμμα διαδικασιών της υπηρεσίας υποβολής αιτήσεων

4.3 Φορητή Εφαρμογή

Στα πλαίσια αυτής της πτυχιακής εργασίας υλοποιήθηκε μια εφαρμογή για Pocket PCs και σύγχρονες φορητές συσκευές όπως κινητά τηλέφωνα. Η εφαρμογή αναπτύχθηκε με την πλατφόρμα Windows Mobile 6.5 Professional και απευθύνεται σε χρήστες με φορητές συσκευές οι οποίες υποστηρίζουν το συγκεκριμένο λειτουργικό σύστημα.

Η εφαρμογή αποτελείται από 8 Windows Forms αντικείμενα (ή αλλιώς φόρμες), όλες οι φόρμες και οι κλάσεις της εφαρμογής βρίσκονται υπό το `namespace: TimeTable_V2`. Οι φόρμες είναι οι εξής: TimeTableForm, LogInForm, SearchProgram, ColumnSelection, ChangeList, Edit_Event, PreviewChanges και ChangeStartTimeRequestForm, οι οποίες υλοποιούνται από τις ομώνυμες κλάσεις. Στη συνέχεια παρουσιάζεται ένα διάγραμμα με τις φόρμες της εφαρμογής.

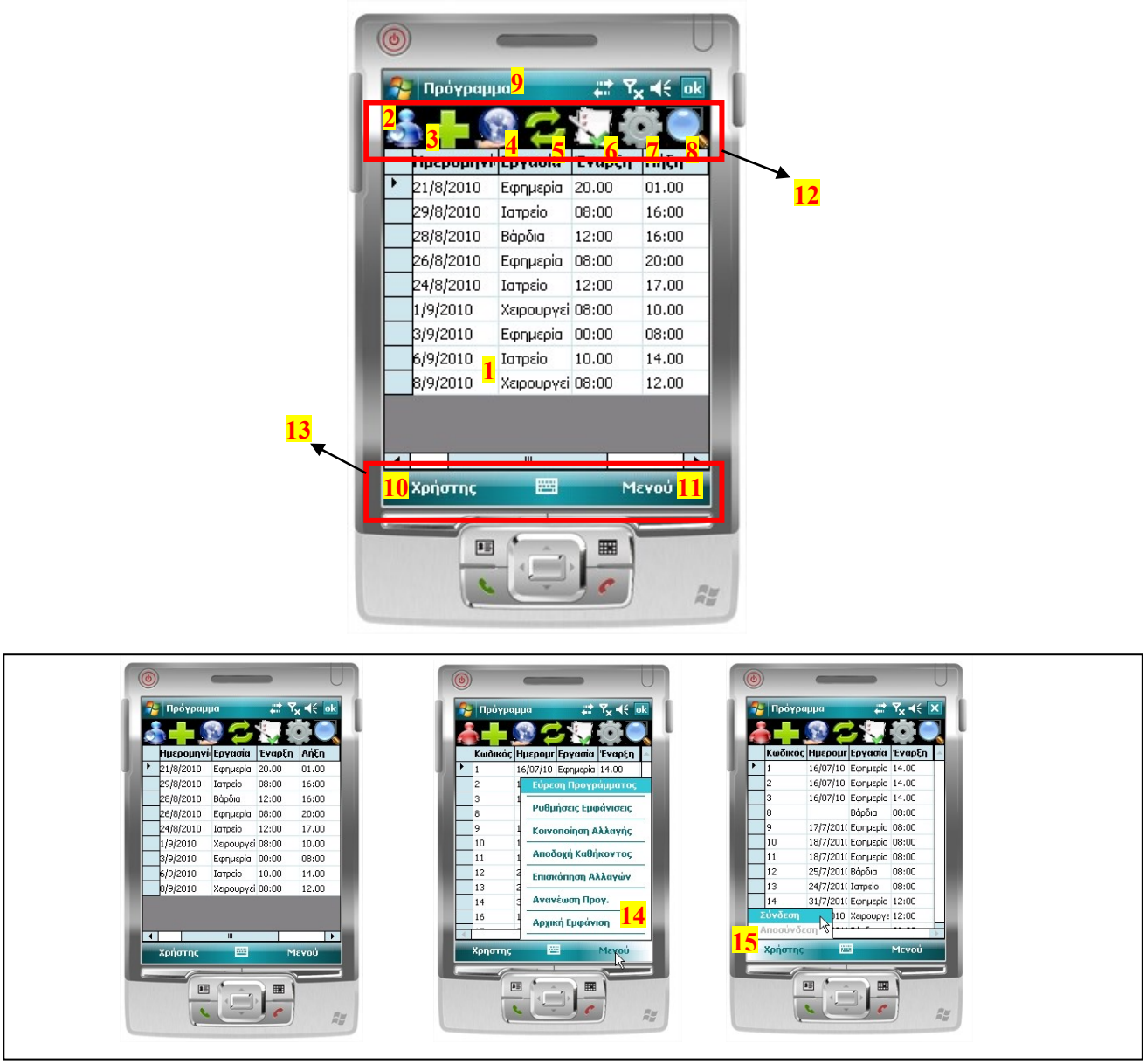


Εικόνα 22: Διάγραμμα με τις φόρμες που αποτελούν τη φορητή εφαρμογή

4.3.1 Φόρμα «TimeTableForm»

Η φόρμα με όνομα TimeTableForm.cs [Εικόνα 23] ή αλλιώς φόρμα: Πρόγραμμα, είναι η πρώτη που εμφανίζεται στον χρήστη κατά την έναρξη της εφαρμογής από την φορητή συσκευή έχοντας τον ρόλο της «Κεντρικής Σελίδας» και αποτελεί την κύρια φόρμα επικοινωνίας του χρήστη με την εφαρμογή. Επίσης είναι η μόνη φόρμα υπεύθυνη για την οπτικοποίηση του προγράμματος εφημεριών-καθηκόντων του χρήστη.

Αποτελείται από μια μπάρα εργαλείων (Tool Bar) στην κορυφή της οθόνης, ένα μεγάλο πλαίσιο Data Grid (πλέγμα δεδομένων) στο κέντρο και μια «Menu Bar» στο κάτω μέρος της οθόνης.



Εικόνα 23: Κεντρική οθόνη φορητής εφαρμογής

Βασικά Χαρακτηριστικά:

1: Πίνακας (Data Grid) εμφάνισης του προγράμματος.

2: Εικονίδιο κατάστασης σύνδεσης χρήστη.

3: Εικονίδιο ενεργοποίησης υπηρεσίας αποδοχής εφημεριών και καθηκόντων.

4: Εικονίδιο ενεργοποίησης υπηρεσίας κοινοποίησης αλλαγών.

5: Εικονίδιο για την ενεργοποίηση της διαδικασίας ανανέωσης των δεδομένων του πίνακα.

6: Εικονίδιο ενεργοποίησης υπηρεσίας επισκόπησης αλλαγών.

7: Εικονίδιο ενεργοποίησης λειτουργίας για την διαμόρφωση των δεδομένων εμφάνισης του προγράμματος.

8: Εικονίδιο για την ενεργοποίηση της υπηρεσίας αναζήτησης.

9: Τίτλος φόρμας.

10: Πλήκτρο για την σύνδεση ή αποσύνδεση του χρήστη.

11: Πλήκτρο για την χρήση του μενού λειτουργιών και υπηρεσιών.

12: Γραμμή (ή «μπάρα») εργαλείων.

13: Γραμμή μενού.

14: Λίστα (ή «Μενού») λειτουργιών και υπηρεσιών της φορητής εφαρμογής.

15: Λίστα επιλογών καταστάσεων σύνδεσης.

Στη μπάρα εργαλείων είναι συγκεντρωμένες οι λειτουργίες που έχουν σχέση με την διαχείριση του προγράμματος εφημεριών-καθηκόντων του χρήστη. Κάθε εικονίδιο είναι μοναδικό και αποτελεί χαρακτηριστικό της λειτουργίας-ιδιότητας που αντιπροσωπεύει. Με τον τρόπο αυτό η πλοήγηση του χρήστη στις λειτουργίες της εφαρμογής πραγματοποιείται εύκολα και γρήγορα.

Στο κέντρο της οθόνης εμφανίζεται το πρόγραμμα εφημεριών-καθηκόντων του χρήστη. Η απεικόνιση των δεδομένων είναι όμοια με αυτή των λογιστικών φύλλων, δηλαδή ένα πλέγμα με κελιά που περιέχει τα δεδομένα του προγράμματος. Ο χρήστης έχει τη δυνατότητα να πλοηγηθεί στα δεδομένα του προγράμματος χρησιμοποιώντας τις μπάρες κύλισης, μία για κάθετη και μία για οριζόντια ολίσθηση.

Στο κάτω μέρος της οθόνης εμφανίζεται η γραμμή κουμπιών «menu» ή αλλιώς «Menu Bar». Τα δύο κουμπιά μενού της γραμμής αντιστοιχούν στα δυο κουμπιά «hardware» που διαθέτουν οι περισσότερες φορητές συσκευές, επιτυγχάνοντας έτσι την πλοήγηση στις λειτουργίες τις εφαρμογής είτε μέσω των πλήκτρων της συσκευής, είτε μέσω των εικονιδίων που εμφανίζονται στην οθόνη. Τα πλήκτρα της Menu Bar δεν αντιστοιχούν σε κάποια συγκεκριμένη λειτουργία της εφαρμογής, όπως συμβαίνει με τα πλήκτρα-εικονίδια της γραμμής εργαλείων. Με το πάτημα αυτών των πλήκτρων εμφανίζεται στον χρήστη μια λίστα με τα ονόματα των διαθέσιμων λειτουργιών από τις οποίες ο χρήστης μπορεί να επιλέξει αυτή που επιθυμεί.

Τα χαρακτηριστικά της φόρμας TimeTableForm όπως και οι λειτουργίες της, υλοποιούνται με την βοήθεια κλάσεων, συναρτήσεων και μεθόδων, οι οποίες θα παρουσιαστούν και θα αναλυθούν στο κεφάλαιο της υλοποίησης. Οι λειτουργίες που εκτελούνται με το πάτημα πλήκτρων ή εικονιδίων, αποτελούν ξεχωριστές οντότητες και

υλοποιούνται στις αντίστοιχες ξεχωριστές φόρμες, μια για κάθε λειτουργία. Η φόρμα TimeTableForm είναι υπεύθυνη για τις εξής λειτουργίες:

- Υπηρεσία αρχικοποίησης και ανανέωσης προγράμματος και προσωπικών δεδομένων.
- Προβολή του προγράμματος εφημεριών-καθηκόντων.
- Προσδιορισμός της κατάστασης σύνδεσης του χρήστη.
- Πλοήγηση του χρήστη στις φόρμες και τις λειτουργίες της εφαρμογής

4.3.1.1 Υπηρεσία αρχικοποίησης και ανανέωσης προγράμματος και προσωπικών δεδομένων

Σκοπός μας είναι να μεταφέρουμε τα δεδομένα που αφορούν τον χρήστη, από τον κεντρικό υπολογιστή, μέσω του Web Service και να τα αποθηκεύσουμε στην τοπική βάση δεδομένων SQL Server Compact Edition της συσκευής μας, που έχει το όνομα TimeTableDB. Οι Web μέθοδοι Program, Declared_Duties και Declared_Locations ανακτούν τα δεδομένα από τους αντίστοιχους πίνακες της ΚΒΔ τα οποία στην συνέχεια λαμβάνονται από την Φορητή Εφαρμογή. Τέλος, η μέθοδος Populate (που θα αναλήσουμε στο κεφάλαιο της υλοποίησης) της φορητής εφαρμογής αποθηκεύει τα ληφθέντα δεδομένα στην τοπική ΒΔ TimeTableDB.

4.3.1.2 Προβολή του προγράμματος εφημεριών-καθηκόντων

Στην προηγούμενη παράγραφο παρακολουθήσαμε την διαδικασία με την οποία τα δεδομένα του προγράμματος εφημεριών και καθηκόντων εισέρχονται στην τοπική ΒΔ TimeTableDB της φορητής συσκευής. Πλέον τα δεδομένα του προγράμματος είναι έτοιμα για επεξεργασία και διαχείρηση από τις μεθόδους της φορητής εφαρμογής καθώς και από τον ίδιο τον χρήστη.

Στα πλαίσια της διαχείρισης και επεξεργασίας των δεδομένων είναι και η προβολή των δεδομένων αυτών που αφορούν το χρήστη της φορητής εφαρμογής. Η προβολή των δεδομένων του προγράμματος εφημεριών και καθηκόντων πραγματοποιείται με την χρήση του πλέγματος δεδομένων (DataGrid). Ο συγκεκριμένος τρόπος εμφάνισης παρέχει στο χρήστη τη δυνατότητα να βλέπει αναλυτικά τα στοιχεία κάθε προγράμματος στην ίδια σειρά [Εικόνα 23].

4.3.1.3 Υπηρεσία προσδιορισμού της κατάστασης σύνδεσης του χρήστη

Σε αυτή την παράγραφο θα διευκρινίσουμε τί εννοούμε με τον όρο *κατάσταση σύνδεσης του χρήστη* και πώς πραγματοποιούμε τον αντίστοιχο έλεγχο. Η εφαρμογή μας προκειμένου να λειτουργεί σύμφωνα με τις προδιαγραφές που έχουμε ορίσει θα πρέπει να πραγματοποιεί συχνές ενημερώσεις στους πίνακες που είναι αποθηκευμένοι στην τοπική ΒΔ. Οι ενημερώσεις των δεδομένων πραγματοποιούνται μέσω των Web Services, γεγονός που απαιτεί την αποτελεσματική επικοινωνία/σύνδεση μεταξύ της φορητής εφαρμογής και της ΚΒΔ που βρίσκεται εγκατεστημένη στον Κεντρικό υπολογιστή (server) του συστήματος και τροφοδοτεί την εφαρμογή με δεδομένα. Οι απαραίτητες προϋποθέσεις σύνδεσης και δικτύωσης που πρέπει να ισχύουν είναι οι εξής:

- Πρόσβαση της φορητής εφαρμογής στο World Wide Web
- Ικανότητα επικοινωνίας και ανταλλαγής δεδομένων μεταξύ της φορητής εφαρμογής και του κεντρικού υπολογιστή με την βοήθεια των Web Services.

Όταν ισχύουν **και** οι δύο παραπάνω προϋποθέσεις, τότε λέμε ότι ο χρήστης «*έχει συνδεθεί επιτυχώς με το σύστημα*» ή πιο απλά «*είναι συνδεδεμένος*» και έχει πρόσβαση σε όλες τις υπηρεσίες που παρέχει η εφαρμογή μας. Η ενημέρωση του χρήστη για την κατάσταση της σύνδεσής του, πραγματοποιείται με ένα χαρακτηριστικό εικονίδιο στην πάνω αριστερή γωνία της οθόνης. Στην προκειμένη περίπτωση το εικονίδιο είναι το : «  ».

Όταν δεν ισχύουν μια ή και οι δυο από τις παραπάνω υποχρεωτικές προϋποθέσεις πρόσβασης, λέμε ότι ο χρήστης «*δεν είναι συνδεδεμένος με το σύστημα*» έχοντας έτσι περιορισμένη πρόσβαση στις υπηρεσίες που παρέχει η εφαρμογή. Τότε στην θέση του εικονιδίου που αντιπροσωπεύει την επιτυχημένη σύνδεση, εμφανίζεται το εικονίδιο «  », δηλώνοντας την έλλειψη πρόσβασης στο WWW ή την ανικανότητα επικοινωνίας με τον κεντρικό υπολογιστή. Οι υπηρεσίες στις οποίες ο χρήστης δεν έχει πρόσβαση είναι: η δημοσίευση υποψήφιων εφημεριών για αλλαγή, η αποδοχή δημοσιευμένων εφημεριών και η ανανέωση της τοπικής βάσης δεδομένων και κατά συνέπεια των προγραμμάτων.

Σε αυτό το σημείο θα πρέπει να τονίσουμε ότι η κατάσταση πρόσβασης που περιγράφουμε και ο έλεγχος αυτής **δεν** έχει καμία σχέση με την ταυτοποίηση των στοιχείων κατά την είσοδο του χρήστη στην εφαρμογή (Log In).

4.3.1.4 Πλοήγηση του χρήστη στις φόρμες και τις λειτουργίες της εφαρμογής

Όπως έχουμε ήδη αναφέρει η φόρμα TimeTableForm αποτελεί την κεντρική φόρμα της εφαρμογής. Αυτό την καθιστά ως το σημείο πρόσβασης για τις υπόλοιπες λειτουργίες της εφαρμογής. Η διεπαφή (interface) της φόρμας έχει συγκεντρωμένες τις λειτουργίες που προορίζονται για τον χρήστη της φορητής εφαρμογής στην γραμμή ή μπάρα λειτουργιών προκειμένου ο χρήστης να έχει ευκολότερη πρόσβαση σε αυτές σε περίπτωση που χρησιμοποιεί συσκευή αφής. Επίσης είναι διαθέσιμος και ο κλασικός τρόπος για την πλοήγηση στις λειτουργίες της εφαρμογής μέσο της χρήσης της γραμμής μενού και της λίστας των λειτουργιών που εμφανίζονται στον χρήστη.

4.3.2 Φόρμα «LogInForm»

Η φόρμα «LogInForm» έχει τίτλο «Είσοδος Χρήστη» και διαχειρίζεται την είσοδο του χρήστη στην φορητή εφαρμογή. Προκειμένου ένας εγγεγραμμένος χρήστης της εφαρμογής να έχει πρόσβαση και να κάνει χρήση των υπηρεσιών της θα πρέπει να εισάγει τα στοιχεία που επαληθεύουν την ταυτότητά του.

4.3.2.1 Υπηρεσία ταυτοποίησης χρήστη

Ο χρήστης κατά την εγγραφή του στο σύστημα είναι υποχρεωμένος να δηλώσει ένα συνθηματικό όνομα χρήστη (username) και έναν μυστικό κωδικό (password). Τα στοιχεία αυτά ταυτοποιούν τον χρήστη καθώς είναι μοναδικά για κάθε χρήστη. Τα username και password είναι στην αρχή (κατά την εγγραφή του χρήστη) αποθηκευμένα μόνο στην ΚΒΔ που βρίσκεται στον κεντρικό απομακρυσμένο υπολογιστή. Προκειμένου να γίνει χρήση των υπηρεσιών και των λειτουργιών του συστήματος μέσω της φορητής του εφαρμογής, θα πρέπει πρώτα να ταυτοποιηθεί. Η ταυτοποίηση εξυπηρετεί δύο σκοπούς:

- Την λήψη των δεδομένων που αφορούν προσωπικά τον χρήστη και για τα οποία ενδιαφέρεται.
- Την προστασία από την λήψη δεδομένων που αφορούν άλλους χρήστες και αποτελούν προσωπικά στοιχεία.

Ο χρήστης κατά την είσοδό του στην εφαρμογή δεν είναι σε θέση να δει οποιαδήποτε δεδομένα ή να κάνει χρήση οποιασδήποτε υπηρεσίας χωρίς πρώτα να έχει ταυτοποιηθεί. Προκειμένου να πραγματοποιηθεί η ταυτοποίηση του χρήστη, αυτός είναι υποχρεωμένος να εισάγει στα κατάλληλα πεδία της φόρμας «LogInForm» τα στοιχεία ταυτοποίησής του, δηλαδή το Όνομα Χρήστη (username) και τον Κωδικό Χρήστη (password).

Εικόνα 24: Φόρμα «LogInForm»

Βασικά Χαρακτηριστικά:

1: Τίτλος φόρμας.

2: Πεδίο εισαγωγής ονόματος χρήστη (username).

3: Πεδίο εισαγωγής κωδικού χρήστη (password).

4: Πλήκτρο πραγματοποίησης σύνδεσης.

5: Πλήκτρο ακύρωσης σύνδεσης.

Όταν ο χρήστης εισάγει τα απαραίτητα στοιχεία θα πρέπει να πατήσει το πλήκτρο «Σύνδεση», που εμφανίζεται στην αριστερή πλευρά της γραμμής menu, προκειμένου να ολοκληρωθεί η διαδικασία ταυτοποίησης και σύνδεσής του με το σύστημα.

Με το πάτημα του πλήκτρου «Σύνδεση» τα στοιχεία που μόλις εισήγαγε ο χρήστης της φορητής εφαρμογής αποστέλλονται μέσω του Web Service και της Web μεθόδου «UserDataCheck» για διασταύρωση με τα στοιχεία των χρηστών που βρίσκονται αποθηκευμένα στην ΚΒΔ. Η εξακρίβωση της ταυτοποίησης των στοιχείων πραγματοποιείτε εκείνη την στιγμή στο ίδιο Web Service και στην ίδια Web μέθοδο. Στη συνέχεια το αποτέλεσμα του ελέγχου αποστέλλεται στη φορητή εφαρμογή. Εάν το αποτέλεσμα σηματοδοτεί την ταυτοποίηση των στοιχείων του χρήστη τότε όλες οι υπηρεσίες και λειτουργίες της εφαρμογής και του συστήματος είναι στην διάθεσή του. Εάν το αποτέλεσμα του ελέγχου των στοιχείων δεν σηματοδοτεί την ταυτοποίηση του χρήστη, τότε η φορητή εφαρμογή δεν εκτελεί ούτε εξυπηρετεί καμία εντολή του χρήστη, ούτε είναι δυνατόν ο

χρήστης να έχει οποιαδήποτε πρόσβαση σε δεδομένα ή υπηρεσίες της φορητής εφαρμογής ή του συστήματος.

4.3.3 Φόρμα «SearchProgram»

Η φόρμα αυτή είναι υπεύθυνη για την εκτέλεση λειτουργιών που εξυπηρετούν:

- Τη γρήγορη και εύκολη αναζήτηση των προγραμματισμένων εφημεριών ή καθηκόντων

- Τη μεμονωμένη εμφάνιση εφημεριών-καθηκόντων που ανήκουν σε μια συγκεκριμένη κατηγορία π.χ. Προγραμματισμένες χειρουργικές επεμβάσεις.

Εικόνα 25: Φόρμα «SearchProgram»

Βασικά Χαρακτηριστικά:

1: Τίτλος της φόρμας

2: Αντικείμενο «DateTimePicker» για την επιλογή ημερομηνίας.

3: Αντικείμενο «CheckBox».

4: Αντικείμενο «ComboBox» για την επιλογή των δηλωμένων από τον χρήστη τύπων εργασίας.

5: Αντικείμενο «ComboBox» για την επιλογή των δηλωμένων από τον χρήστη τοποθεσιών.

6: Αντικείμενο «ComboBox» για την επιλογή των αποθηκευμένων (στην τοπική ΒΔ) προγραμμάτων.

7: Πλήκτρο ενεργοποίησης της διαδικασίας αναζήτησης.

8: Πλήκτρο επιστροφής στην κεντρική φόρμα με ταυτόχρονη ακύρωση της διαδικασίας αναζήτησης.

Η φόρμα περιλαμβάνει ένα αντικείμενο «DateTimePicker», τέσσερα αντικείμενα «CheckBox» και τρία αντικείμενα «ComboBox». Κάθε ένα από αυτά τα αντικείμενα χρησιμοποιούνται για την επιλογή των κριτηρίων αναζήτησης από τον χρήστη. Αναλυτικότερα, στο αντικείμενο «DateTimePicker» ο χρήστης επιλέγει μία ημερομηνία προκειμένου να εμφανιστούν οι εφημερίες που αντιστοιχούν σε αυτή. Ενώ με την ενεργοποίηση του αντικειμένου «CheckBox» που βρίσκεται από κάτω, η αναζήτηση γενικεύεται με αποτέλεσμα να επιλέγονται για εμφάνιση οι μελλοντικά προγραμματισμένες εφημερίες ή καθήκοντα. Στη συνέχεια, ο χρήστης μπορεί να επιλέξει ως κριτήριο αναζήτησης τον τύπο της εργασίας, δηλαδή αν πρόκειται για εφημερία, για βάρδια, για χειρουργείο κλπ, την τοποθεσία για την οποία έχει προγραμματιστεί η εργασία-εφημερία, π.χ. όνομα νοσοκομείου, τμήμα εργασίας, κλινική κλπ. Τέλος ο χρήστης έχει την δυνατότητα να επιλέξει προς εμφάνιση εργασίες οι οποίες ανήκουν σε κάποιο καθορισμένο από την γραμματεία πρόγραμμα όπως για παράδειγμα το πρόγραμμα εφημεριών για τον μήνα Αύγουστο ή το πρόγραμμα βάρδιας της 32^{ης} εβδομάδας.

Θα πρέπει επίσης να αναφέρουμε ότι τα κριτήρια προς επιλογή των «ComboBox» που αφορούν τον Τύπο Εργασίας και την Τοποθεσία είναι διαφορετικά για κάθε χρήστη. Κάθε χρήστης, ανάλογα με την ομάδα στην οποία ανήκει, μπορεί να επιλέξει κατά την εγγραφή του έναν αριθμό από διαθέσιμους Τύπους Εργασιών και διαθέσιμες Τοποθεσίες που αντιστοιχούν σε εφημερίες για τις οποίες επιθυμεί να ενημερώνεται από το σύστημα. Για παράδειγμα ένας χρήστης που ανήκει στην «Ομάδα: Γιατρός» και στο «Καρδιολογικό Τμήμα» μπορεί να επιλέξει να ενημερώνεται για «τύπους Εργασιών» όπως χειρουργεία, βάρδιες, εφημερίες κ.α. και τοποθεσίες όπως Νοσοκομείο x, Κλινική x, τμήμα επειγόντων περιστατικών κ.α. Ενώ κάποιος χρήστης που ανήκει στην «Ομάδα: Διοικητικό Προσωπικό» μπορεί να επιλέξει να ενημερώνεται για «τύπους Εργασιών» όπως βάρδιες, εφημερίες κ.α. και τοποθεσίες όπως Γραμματειακή Υποστήριξη, Τμήμα Πληροφορικής, Οικονομικό Τμήμα.

Για την υποβολή των κριτηρίων αναζήτησης, ο χρήστης θα πρέπει να πατήσει το πλήκτρο «Άνοιγμα» της menu bar και θα εμφανιστεί στη φόρμα «Πρόγραμμα» το πρόγραμμα

εφημεριών-καθηκόντων που αντιστοιχεί στα κριτήρια αναζήτησης που ο ίδιος έχει θέσει. Εάν ο χρήστης δεν θέλει να κάνει χρήση της λειτουργίας αναζήτησης, τότε έχει την δυνατότητα να επιστρέψει στην προηγούμενη οθόνη που βρισκόταν, με το πάτημα του πλήκτρου «Επιστροφή» της menu bar.

4.3.4 Φόρμα «ChangeList»

Η ανταλλαγή των εφημεριών και των διαφόρων καθηκόντων δεν θα είχε κανένα νόημα εάν οι χρήστες μόνο δημοσίευαν τις εργασίες τους προς αλλαγή και δεν δέχονταν να εκπληρώσουν εργασίες συναδέλφων τους. Η φόρμα «`ChangeList`» είναι αυτή που επιτρέπει στους χρήστες να εντάσσουν στο πρόγραμμά τους μια ή περισσότερες δημοσιευμένες εφημερίες/εργασίες συναδέλφων τους.

4.3.4.1 Υπηρεσία δημοσίευσης και ανταλλαγής εργασιών-Λειτουργία αποδοχής δημοσιευμένων εργασιών

Η συγκεκριμένη υπηρεσία ενεργοποιείτε με το πάτημα του πλήκτρου «  » που βρίσκετε στην γραμμή λειτουργιών, στο επάνω μέρος της κεντρικής φόρμας. Όταν ο χρήστης πατήσει το πλήκτρο αυτό, ένα αίτημα αποστέλλεται από την φορητή εφαρμογή προς το Web Service μέσω της Web μεθόδου «UsersChangeList» η οποία όπως αναφέραμε και στο προηγούμενο κεφάλαιο, επιστρέφει μια λίστα με τις δημοσιευμένες εργασίες των χρηστών του συστήματος. Η λίστα με τις δημοσιευμένες εργασίες των χρηστών, εμφανίζεται στην οθόνη με τη βοήθεια ενός αντικειμένου «DataGrid». Στη συνέχεια ο χρήστης είναι σε θέση να επιλέξει την εφημερία που θέλει να εντάξει στο πρόγραμμά του, μέσα από την λίστα εφημεριών και εργασιών που εμφανίζεται στην οθόνη της συσκευής του.

Εικόνα 26: Η φόρμα "ChangeList"

Βασικά Χαρακτηριστικά:

1: Τίτλος φόρμας.

2: Πλήκτρο αποδοχής επιλεγμένης εφημερίας.

3: Πίνακας με τις δημοσιευμένες προς ανταλλαγή εργασίες/εφημερίες.

4: Πλήκτρο ακύρωσης της υπηρεσίας αποδοχής εργασιών/εφημεριών και επιστροφής στην κεντρική οθόνη.

Εφόσον ο χρήστης επιλέξει την εφημερία που θέλει να εντάξει στο πρόγραμμά του, με το πάτημα του πλήκτρου «Αποδοχή» ενεργοποιούνται οι μηχανισμοί, που καλούν την Web μέθοδο UpdateChanges η οποία, όπως αναφέραμε και στην παρουσίαση του Web Service, εντάσσει αυτόματα την επιλεγμένη εργασία στο πρόγραμμα του χρήστη ενημερώνοντας αυτόματα την ΚΒΔ καθώς και την τοπική ΒΔ, ενώ παράλληλα ενημερώνεται για την αλλαγή και ο χρήστης που είχε δημοσιεύσει την συγκεκριμένη εργασία. Εάν ο χρήστης δεν επιθυμεί να χρησιμοποιήσει τη λειτουργία «Αποδοχή Καθήκοντος», μπορεί απλά να πατήσει το πλήκτρο «Ακύρωση» που βρίσκεται στο δεξιό τμήμα της menu bar.

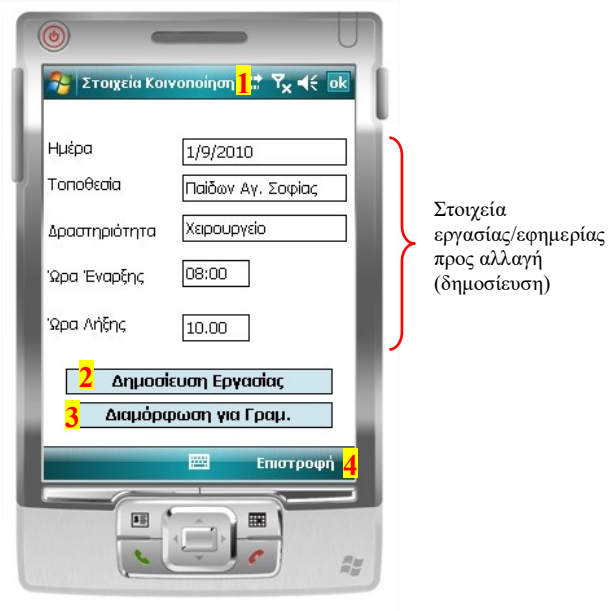
Τα δεδομένα που εμφανίζονται στο «DataGrid» προέρχονται από τον πίνακα ChangeList. Όπως ήδη έχουμε αναφέρει στην όλη διαδικασία παρεμβάλλονται υπηρεσίες του Web Service μεταξύ της εφαρμογής και του απομακρυσμένου κεντρικού υπολογιστή. Για τον λόγο αυτό ο χρήστης θα πρέπει να είναι «συνδεδεμένος» με το σύστημα κατά την χρήση της

συγκεκριμένης υπηρεσίας. Η συγκεκριμένη υπηρεσία έχει σχεδιαστεί έτσι ώστε η λίστα των εργασιών που αφορά τον κάθε χρήστη ξεχωριστά να είναι η πιο πρόσφατα ενημερωμένη, αποφεύγοντας με τον τρόπο αυτό την δημιουργία αντικρουόμενων αιτήσεων για την αποδοχή εργασιών.

4.3.5 Φόρμα «Edit_Event»

Στην προηγούμενη παράγραφο είδαμε τον τρόπο με τον οποίο πραγματοποιείται η αποδοχή των δημοσιευμένων εργασιών, μέρος της υπηρεσίας δημοσίευσης και ανταλλαγής εφημεριών. Σε αυτή την παράγραφο θα παρουσιάσουμε τον τρόπο με τον οποίο πραγματοποιούνται οι δημοσιεύσεις. Τον ρόλο αυτό διαδραματίζει η φόρμα «`Edit_Event`». Επίσης, η φόρμα αυτή αποτελεί προπύργιο για την υπηρεσία αποστολής αιτήσεων τροποποίησης του προγράμματος εφημεριών/καθηκόντων.

Προκειμένου ο χρήστης να δημοσιεύσει μια εργασία του προγράμματός του με σκοπό να την αναλάβει κάποιος άλλος συνάδελφός του, θα πρέπει πρώτα να επιλέξει από τον πίνακα των προγραμμάτων που εμφανίζεται στην οθόνη της κεντρικής φόρμας την εργασία ή εφημερία που θέλει να δημοσιεύσει και στη συνέχεια να πατήσει το πλήκτρο «  » που βρίσκεται στην γραμμή λειτουργιών. Με το πάτημα του πλήκτρου εμφανίζεται στον χρήστη η φόρμα «Edit_Event».



Εικόνα 27: Η φόρμα "Edit_Event"

Βασικά Χαρακτηριστικά:

1: Τίτλος φόρμας.

2: Πλήκτρο ενεργοποίησης υπηρεσίας δημοσίευσης εφημερίας προς ανταλλαγή.

3: Πλήκτρο μορφοποίησης της φόρμας με σκοπό την δημιουργία αιτήσεων.

4: Πλήκτρο επιστροφής στην αρχική οθόνη με ταυτόχρονη ακύρωση της διαδικασίας.

Στην φόρμα εμφανίζονται τα στοιχεία της εργασίας που έχει σκοπό να δημοσιεύσει ο χρήστης. Έτσι ο χρήστης ελέγχει τα στοιχεία της εργασίας προκειμένου να σιγουρευτεί ότι όντως επιθυμεί την δημοσίευση της συγκεκριμένης εργασίας. Εάν τελικά ο χρήστης δεν θέλει να συνεχίσει την διαδικασία δημοσίευσης της εργασίας μπορεί να ακυρώσει την διαδικασία και να επιστρέψει στην κεντρική φόρμα πατώντας το πλήκτρο «Επιστροφή» που βρίσκεται στο δεξί μέρος της γραμμής menu.

4.3.5.1 Υπηρεσία δημοσίευσης και ανταλλαγής εφημεριών-Λειτουργία Δημοσίευσης Εργασιών

Όπως έχουμε αναφέρει η λειτουργία της Δημοσίευσης Εργασιών που αποτελεί τμήμα της Υπηρεσίας Δημοσίευσης και Ανταλλαγής Εφημεριών εκτελείται από την φόρμα «Edit_Event». Εφόσον ο χρήστης είναι σίγουρος για την δημοσίευση της εργασίας που έχει επιλέξει, αρκεί να πατήσει το πλήκτρο «Δημοσίευση Εργασίας» προκειμένου να αρχίσει η λειτουργία του «μηχανισμού». Με το πάτημα του συγκεκριμένου πλήκτρου καλείται η Web μέθοδος InsertChanges η οποία συλλέγει τα στοιχεία της εργασίας προς δημοσίευση με σκοπό την ενημέρωση της λίστας των δημοσιευμένων εργασιών που βρίσκεται στον πίνακα ChangeList της ΚΒΔ. Με τον τρόπο αυτό οι χρήστες θα είναι ενήμεροι για τις νέες εργασίες που είναι διαθέσιμες προς αποδοχή.

Με το πάτημα του πλήκτρου «Δημοσίευση Εργασίας» ενεργοποιείται και η ενημέρωση του πίνακα «ChangeList» της τοπικής ΒΔ, με την καταγραφή της νέας εργασίας προς δημοσίευση. Ο πίνακας «ChangeList» της τοπικής ΒΔ περιέχει τις εφημερίες που έχει επιλέξει ο χρήστης προς δημοσίευση.

Εφόσον ολοκληρωθεί η λειτουργία της Δημοσίευσης της εργασίας, η φόρμα «Edit_Event» κλείνει και παρουσιάζεται πάλι στον χρήστη η οθόνη με το πρόγραμμα εφημεριών-καθηκόντων.

4.3.5.2 Υπηρεσία υποβολής αιτήσεων - Εκκίνηση διαδικασίας

Στη συνέχεια θα παρουσιάσουμε την περίπτωση οπού ο χρήστης δεν θέλει απλά να προβεί σε δημοσίευση της εφημερίας του με σκοπό την ανταλλαγή, αλλά επιθυμεί να προβεί σε τροποποίηση του προγράμματος εφημεριών-καθηκόντων αλλάζοντας τα στοιχεία της εφημερίας του. Η συγκεκριμένη τροποποίηση μπορεί να πραγματοποιηθεί μόνο μετά από αίτηση στον υπεύθυνο για την δημιουργία και την σύνταξη του προγράμματος.

Η εκκίνηση της διαδικασίας για την υπηρεσία υποβολής αιτήσεων πραγματοποιείται στην φόρμα «Edit_Event», ενώ η διαδικασία συνεχίζεται και ολοκληρώνεται στην φόρμα «ChangeStartTimeRequestForm» που θα παρουσιάσουμε στην συνέχεια.

Ο χρήστης, πατώντας το πλήκτρο «Διαμόρφωση για Γραμματεία» παρακολουθεί την φόρμα «Edit_Event» να αλλάζει μορφή. Ορισμένα πλήκτρα εμφανίζονται ενώ άλλα τροποποιούνται. Σκοπός της συγκεκριμένης διαμόρφωσης είναι η παρουσίαση στον χρήστη, των επιλογών τροποποίησης των στοιχείων της εφημερίας.

Εικόνα 28: Μορφοποίηση "Διαμόρφωση για Γραμματεία" της φόρμας "Edit_Event"

Οι επιλογές αυτές είναι:

- Η δυνατότητα αλλαγής της ώρας έναρξης της εφημερίας, της ώρας λήξης της εφημερίας ή και των δύο (πλήκτρα «Αλλαγή» 5, 6).

- Μεταφορά της εργασίας σε άλλη ημερομηνία και με διαφορετικές ώρες (πλήκτρο «Μεταφορά Εφημερίας» 7).

- Ολική ακύρωση της εφημερίας, χωρίς αναπλήρωσή της (πλήκτρο «Ακύρωση Εφημερίας» 8).

4.3.5.2.1 Αίτηση αλλαγής ώρας – Εκκίνηση Διαδικασίας

Με το πάτημα των πλήκτρων «Αλλαγή», παρουσιάζεται στον χρήστη η φόρμα «ChangeStartTimeRequestForm» με σκοπό την συνέχιση της διαδικασίας η οποία περιλαμβάνει την σύνταξη και την υποβολή, προς την γραμματεία, της αίτησης αλλαγής ώρας έναρξης ή και λήξης της εφημερίας-εργασίας.

4.3.5.2.2 Αίτηση μεταφοράς εφημερίας/εργασίας – Εκκίνηση Διαδικασίας

Συμπεριφορά όμοια με αυτή που αναφέρθηκε στην προηγούμενη παράγραφο παρουσιάζει το πάτημα του πλήκτρου «Μεταφορά Εφημερίας». Όταν ο χρήστης επιθυμεί να κάνει μεταφορά της εργασίας του πατά το πλήκτρο «Μεταφορά Εφημερίας» και πραγματοποιείται η μετάβαση από την φόρμα «Edit_Event» στην φόρμα «Change Start Time Request Form» όπου και ολοκληρώνεται η διαδικασία με την σύνταξη και την υποβολή της αίτησης αλλαγής της ημερομηνίας ή και της ώρας της εφημερίας-εργασίας.

4.3.5.2.3 Αίτηση ολικής ακύρωσης εφημερίας

Αντίθετα από την παραπάνω περίπτωση υποβολής αίτησης, στην περίπτωση της αίτησης για Ολική ακύρωση της εφημερίας, η διαδικασία αρχίζει και ολοκληρώνεται στην φόρμα «Edit_Event». Όταν ο χρήστης είναι σίγουρος ότι επιθυμεί την ακύρωση της εργασίας του, τότε μπορεί απλά να πατήσει το πλήκτρο «Ακύρωση Εφημερίας» που εμφανίζεται στο κάτω μέρος της φόρμας «Edit_Event». Με το πάτημα του πλήκτρου δεν παρουσιάζεται κάποια ξεχωριστή ειδική φόρμα στην οθόνη του χρήστη. Αυτό που πραγματοποιείται είναι μια αποστολή αιτήματος διαγραφής της εφημερίας προς την γραμματεία με την ενεργοποίηση των Web μεθόδων Insert Applications και Program Modification, των οποίων τη λειτουργία έχουμε εξηγήσει στην παράγραφο Web Service. Στη συνέχεια, ο χρήστης θα ενημερωθεί για τη διεκπεραίωση της αίτησής του μέσα από την ανανέωση/αλλαγή του προγράμματος εφημεριών/καθηκόντων.

4.3.6 Φόρμα «ChangeStartTimeRequestForm»

Προκειμένου ο χρήστης να τροποποιήσει την εφημερία-εργασία που του αναλογεί ή ορισμένα από τα χαρακτηριστικά της, θα πρέπει να υποβάλει την αντίστοιχη αίτηση προς την υπηρεσία που είναι αρμόδια για την σύνταξη και τις αλλαγές του προγράμματος εφημεριών-καθηκόντων. Η υπηρεσία αυτή τις περισσότερες φορές αποτελείται από διοικητικό προσωπικό που έχει τον ρόλο της γραμματείας της νοσοκομειακής μονάδας (για λόγους συντομίας, θα αναφέρουμε την υπηρεσία αυτή απλά ως γραμματεία). Για την αυτοματοποίηση της παραπάνω διαδικασίας έχουμε δημιουργήσει την φόρμα «Change Start Time Request Form» η οποία πραγματοποιεί τις αιτήσεις:

- Αλλαγής ώρας έναρξης της εργασίας, ώρας λήξης της εργασίας ή και των δύο.
- Μεταφοράς εργασίας σε άλλη ημερομηνία και με διαφορετικές ώρες.

Θα πρέπει σε αυτό το σημείο να υπενθυμίσουμε ότι για την αίτηση ολικής ακύρωσης της εφημερίας-εργασίας, η αίτηση υποβάλλεται αυτόματα στην γραμματεία με το πάτημα του πλήκτρου «Ακύρωση Εφημερίας» που βρίσκεται στην φόρμα «Edit_Event». Με αποτέλεσμα να μην παρεμβάλλεται καθόλου η φόρμα «Change Start Time Request Form».

Για την αποτελεσματική σύνταξη και υποβολή προς την γραμματεία των παραπάνω αιτήσεων έχουμε δημιουργήσει για τον χρήστη ένα λειτουργικό περιβάλλον το οποίο αποτελείται από τρεις διαφορετικές αλλά όμοιες μεταξύ τους διεπαφές. Σε αυτές, καλείται ο χρήστης να συμπληρώσει τα στοιχεία για την τροποποίησης της εφημερίας του. Και οι τρεις διεπαφές προκύπτουν από τη φόρμα «Change Start Time Request Form» και κάθε μία από αυτές προκύπτει μετά από διαμόρφωση των χαρακτηριστικών-αντικειμένων της φόρμας αυτής («ChangeStartTimeRequestForm»). Έτσι, προκειμένου να είμαστε πιο ακριβείς και τυπικοί στα λεγόμενά μας, θα αναφερόμαστε σε αυτές τις διεπαφές, με τον χαρακτηρισμό *μορφοποιήσεις* ή *μορφές*. Κάθε μορφή λοιπόν, αντιστοιχεί στην δημιουργία μίας αίτησης τροποποίησης της εφημερίας-εργασίας του χρήστη προς την γραμματεία, η οποία απαιτεί να συμπληρωθούν στοιχεία από τον χρήστη. Ανάλογα με την αίτηση τροποποίησης που θέλει να υποβάλει ο χρήστης, πατά το αντίστοιχο πλήκτρο στην φόρμα «Edit_Event» που θα έχει ως συνέπεια την εμφάνιση της κατάλληλης προς συμπλήρωση αίτησης.

4.3.6.1.1 Αίτηση αλλαγής ώρας

Όταν ο χρήστης πατά ένα από τα πλήκτρα «Αλλαγή» που βρίσκονται στην φόρμα «Edit_Event» , στην οθόνη εμφανίζεται η ακόλουθη μορφοποίηση:

Εικόνα 29: Μορφοποίηση "Αίτηση αλλαγής ώρας" της φόρμας "ChangeStartTimeRequestForm"

Βασικά Χαρακτηριστικά:

1: Τίτλος φόρμας.

2: Πλαίσιο καταγραφής τρέχουσας ώρας (της εφημερίας).

3: Πλαίσια εισαγωγής της νέας ώρας σε μορφή ωω:λλ.

4: Πλήκτρο επικύρωσης νέας ώρας.

5: Πλήκτρο μετάβασης στην μορφοποίηση με σκοπό την υποβολή αίτησης μεταφοράς.

Η συγκεκριμένη μορφή δίνει στο χρήστη τη δυνατότητα τροποποίησης της ώρας έναρξης, της ώρας λήξης ή και των δύο με έναν αρκετά λειτουργικό και εύχρηστο τρόπο. Με σαφή διαχωρισμό μεταξύ των λειτουργιών αλλαγής ώρας, παρουσιάζεται στον χρήστη η τρέχουσα Ώρα Έναρξης και Λήξης και ζητείται από αυτόν να εισάγει την επιθυμητή (Νέα Ώρα) Ώρα Έναρξης ή/και Λήξης της εργασίας-εφημερίας σε μορφή «ωω:λλ». Για την επιβεβαίωση της αλλαγής της ώρας της εφημερίας, ο χρήστης θα πρέπει να πατήσει το πλήκτρο «OK» που αντιστοιχεί στην ώρα που θέλει να τροποποιήσει (ώρα λήξης ή έναρξης) και βρίσκεται στο δεξιό τμήμα της φόρμας. Εάν ο χρήστης δεν εισάγει την ώρα στην οποία επιθυμεί να μεταφερθεί η εφημερία του, τότε αυτή παραμένει η ίδια, δηλαδή ισχύει η τρέχουσα ώρα Έναρξης ή και Λήξης. Στη συνέχεια, πατώντας το πλήκτρο «Υποβολή»,

ενεργοποιούνται οι κατάλληλοι μηχανισμοί και μέθοδοι του Web Service προκειμένου να μεταφερθεί η αίτηση αλλαγής ώρας στην αρμόδια υπηρεσία της Νοσοκομειακής μονάδας όπου και εξετάζεται η δυνατότητα έγκρισής της και η τροποποίησης της συγκεκριμένης εφημερίας.

Όπως είδαμε και σε προηγούμενη παράγραφο, οι Web μέθοδοι είναι υπεύθυνες για την έκβαση της υπηρεσίας υποβολής αιτήσεων είναι οι Program Modigication και Insert Application.

Θα παραλείψουμε σκόπιμα τη λειτουργία του πλήκτρου «Μεταφορά Εφημερίας» που εμφανίζεται στο κέντρο της οθόνης, την οποία θα μελετήσουμε αργότερα.

4.3.6.1.2 Αίτηση μεταφοράς εφημερίας/εργασίας

Όταν ο χρήστης πατά το πλήκτρο «Μεταφορά Εφημερίας», που βρίσκεται στην φόρμα «Edit_Event» , στην οθόνη εμφανίζεται η ακόλουθη μορφοποίηση:

Εικόνα 30: Διαμόρφωση "Μεταφορά εφημερίας" της φόρμας "ChangeStartTimeRequestForm"

Βασικά Χαρακτηριστικά:

6: Πλαίσιο καταγραφής της τρέχουσας ημερομηνίας (της εφημερίας).

7: Αντικείμενο «DateTimePicker» για την επιλογή της νέας ημερομηνίας (της εφημερίας).

8: Πλήκτρο επικύρωσης νέας ημερομηνίας.

9: Πλήκτρο μετάβασης στην μορφοποίηση με σκοπό την υποβολή αίτησης αλλαγής ώρας.

10: Πλήκτρο υποβολής αίτησης.

11: Πλήκτρο επιστροφής στη κεντρική οθόνη με ταυτόχρονη ακύρωση της διαδικασίας.

Η συγκεκριμένη μορφή δίνει στον χρήστη την δυνατότητα τροποποίησης της ημερομηνίας που πρέπει να πραγματοποιηθεί η εφημερία-εργασία. Ουσιαστικά δηλαδή αναφερόμαστε σε μεταφορά του καθήκοντος σε διαφορετική ημέρα. Στην φόρμα παρατηρούμε δύο πλαίσια κειμένου: στο πρώτο παρουσιάζεται η τρέχουσα ημερομηνία της εφημερίας ενώ στο δεύτερο ο χρήστης καλείται μέσα από ένα αντικείμενο «Time Day Picker», το οποίο προσομοιώνει ένα ημερολόγιο, να επιλέξει την ημερομηνία στην οποία επιθυμεί να μεταφερθεί η εφημερία-εργασία του. Ο χρήστης προκειμένου να επικυρώσει την εισαγωγή της νέας ημερομηνίας, θα πρέπει να πατήσει το πλήκτρο «OK» που βρίσκεται δεξιά από το πλαίσιο εισαγωγής της νέας ημερομηνίας. Εάν για οποιοδήποτε όμως λόγω δεν επιθυμεί να συνεχίσει τη διαδικασία με την υποβολή της αίτησης, του παρέχεται η δυνατότητα να την ακυρώσει με το πάτημα του πλήκτρου «Επιστροφή» από το γραμμή menu. Για την αποστολή της αίτησης προς την γραμματεία ο χρήστης πατά το πλήκτρο «Υποβολή» ενεργοποιώντας τους κατάλληλους μηχανισμούς και μεθόδους του Web Service.

4.3.6.1.2.1 Αίτηση Ολικής μεταφοράς εφημερίας/εργασίας

Τα πλήκτρα «Αλλαγή Ώρας» και «Μεταφορά Εφημερίας» που εμφανίζονται στο κέντρο της φόρμας κατά την υποβολή αίτησης αλλαγής ώρας και κατά την υποβολή αίτησης μεταφοράς της εργασίας και που σκόπιμα δεν παρουσιάσαμε στις προηγούμενες παραγράφους, οδηγούν στην τρίτη και τελευταία διαμόρφωση της φόρμας «ChangeStartTimeRequestForm».

Εικόνα 31: Διαμόρφωση "Ολική μεταφορά εφημερίας" της φόρμας "ChangeStartTimeRequestForm"

Η διαμόρφωση αυτή είναι μια παραλλαγή της υποβολής αίτησης για μεταφορά της εργασίας και αποτελεί συγχώνευση των δύο προηγούμενων διαμορφώσεων (συμπλήρωση και υποβολή αίτησης αλλαγής ώρας και μεταφοράς εργασίας) συμπεριλαμβανομένων και των λειτουργιών τους. Για τον λόγο αυτό, μπορούμε να αναφερθούμε στα πλήκτρα «Αλλαγή Ώρας» και «Μεταφορά Εφημερίας» που εμφανίζονται στις προηγούμενες διαμορφώσεις, με τον όρο: *πλήκτρα συγχώνευσης*. Ουσιαστικά, πρόκειται για ένα πλήκτρο με δύο διαφορετικά ονόματα - εξαρτάται από τον τύπο της αίτησης που εμφανίζεται στην οθόνη του χρήστη.

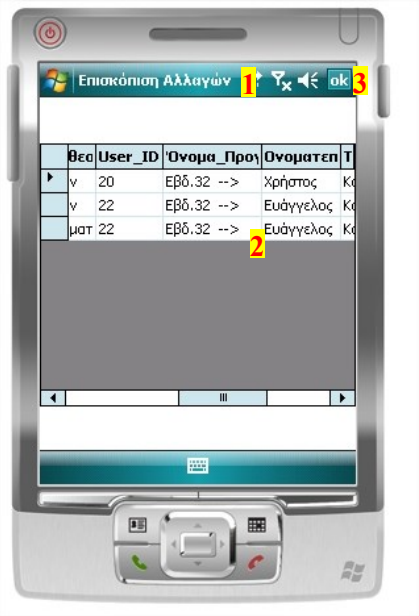
Έχοντας ήδη θέσει σαφή όρια των αιτήσεων με την ύπαρξη ξεχωριστών μορφοποιήσεων για κάθε μία, θελήσαμε να αποδώσουμε μια πιο ουσιαστική έννοια στον όρο «Μεταφορά Εφημερίας». Δηλαδή η συγκεκριμένη διαμόρφωση αποτελεί μία ολική αίτηση μεταφοράς της εφημερίας του χρήστη αφού του δίνεται η δυνατότητα παράλληλης τροποποίησης της ώρας και της ημερομηνίας της εφημερίας του. Μπορούμε επίσης να παρατηρήσουμε ότι παρόλη την συγκέντρωση αρκετών στοιχείων τροποποίησης στην ίδια διαμόρφωση, υπάρχει σαφής διαχωρισμός μεταξύ τους. Το γεγονός αυτό προσδίδει λειτουργικότητα και ευχρηστία προσφέροντας στο χρήστη μια σχετική ευελιξία (με τα πλήκτρα συγχώνευσης) η οποία του επιτρέπει και να διορθώσει τυχόν σφάλματα κατά την πλοήγησή του στην εφαρμογή.

4.3.7 Φόρμα «PreviewChanges»

Κάθε χρήστης που δημοσιεύει μία εφημερία-εργασία για ανταλλαγή, θα πρέπει να έχει την δυνατότητα να γνωρίζει το άτομο που την αποδέχεται στο προσωπικού του πρόγραμμα. Η συγκεκριμένη δυνατότητα πραγματοποιείται από την φόρμα «PreviewChanges».

Η φόρμα παρουσιάζει σε ένα πίνακα τα ακριβή στοιχεία της εφημερίας ή της εργασίας που δημοσίευσε ο κάτοχος της φορητής εφαρμογής καθώς και τα πλήρη στοιχεία ταυτοποίησης του ατόμου που την συμπεριέλαβε στο πρόγραμμά του.

Όλα τα παραπάνω δεδομένα καταγράφονται στους αντίστοιχους πίνακες της ΚΒΔ προς αποφυγή διενέξεων και προβλημάτων απόδοσης ευθυνών.



Εικόνα 32: Φόρμα "PreviewChanges"

Βασικά Χαρακτηριστικά:

1: Τίτλος φόρμας.

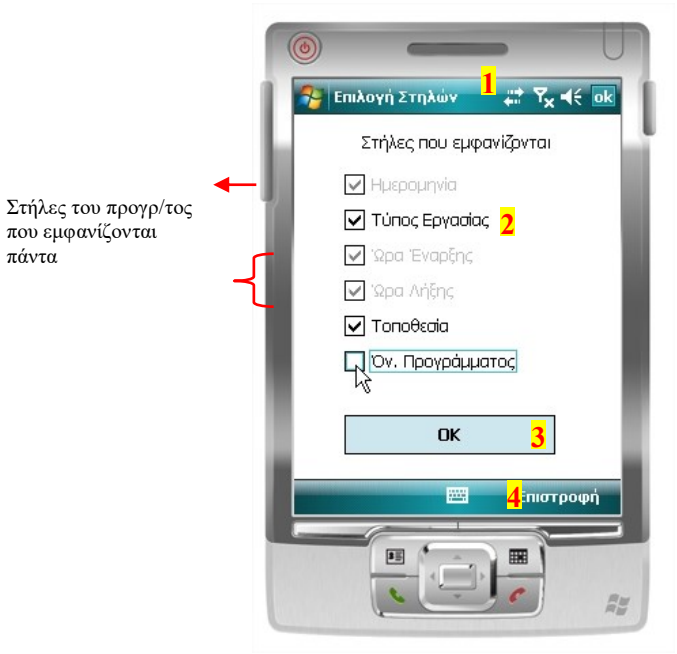
2: Πίνακας επισκόπησης αλλαγών.

3: Πλήκτρο επιστροφής στην κεντρική οθόνη.

4.3.8 Φόρμα «ColumnSelection»

Θέλοντας να δώσουμε στον χρήστη την δυνατότητα να προσαρμόζει ο ίδιος τα δεδομένα εμφάνισης του προγράμματος εφημεριών-καθηκόντων, σχεδιάσαμε και δημιουργήσαμε την φόρμα «Column Selection». Η φόρμα αυτή είναι υπεύθυνη για την

εμφάνιση ή την απόκρυψη στηλών από τον πίνακα του προγράμματος που εμφανίζεται στην οθόνη του χρήστη.



Εικόνα 33: Η φόρμα "Column Selection"

Βασικά Χαρακτηριστικά:

1: Τίτλος φόρμας.

2: Αντικείμενο «CheckBox».

3: Πλήκτρο ενεργοποίησης των επιλογών του χρήστη.

4: Πλήκτρο επιστροφής στην κεντρική οθόνη και ταυτόχρονη ακύρωση τυχόν επιλογών.

Ο χρήστης έχει την δυνατότητα να επιλέξει τις στήλες που θα εμφανίζονται μέσω των αντικειμένων «Check Box» που εμφανίζονται στην οθόνη. Εάν το κουτάκι που αντιπροσωπεύει κάθε στήλη είναι επιλεγμένο «☑», τότε τα δεδομένα της στήλης αυτής θα εμφανίζονται στον χρήστη. Η «by default» εμφάνιση του προγράμματος περιλαμβάνει όλες τις στήλες. Ο χρήστης έχει την δυνατότητα να απενεργοποιήσει το κουτάκι και την στήλη που αντιπροσωπεύει, πατώντας πάνω του. Όταν ο χρήστης ολοκληρώσει την επιλογή των στηλών τότε πατά το πλήκτρο «OK» για την υποβολή των ρυθμίσεων.

Μέσα από την λειτουργία αυτής της φόρμας επιθυμούμε να προσφέρουμε στον χρήστη την δυνατότητα προσαρμογής της εμφάνισης του προγράμματός του στις προσωπικές του ανάγκες και απαιτήσεις. Οι ομάδες των ιατρών και των νοσηλευτών έχουν ιδιαίτερες

απαιτήσεις, εργάζονται κάτω από ένα δύσκολο και πολλές φορές συνεχές ωράριο εργασίας, γεγονός που απαιτεί την εύκολη και γρήγορη ανάγνωση των δεδομένων. Η εφαρμογή έχει την δυνατότητα μέσω της φόρμας «Column Selection» και της λειτουργίας της «επιλογής στηλών», να παρουσιάζει με λειτουργικό τρόπο τα δεδομένα ώστε να μην κουράζουν τον χρήστη.

Επίσης ο όγκος των δεδομένων του προγράμματος εφημεριών-καθηκόντων με τις πολλές γραμμές και στήλες μπορούν να μπερδέψουν τον χρήστη - συγκεκριμένα λάθη μπορεί να προκαλέσουν μεγάλο αντίκτυπο στην σωστή εκτέλεση του προγράμματος και την λειτουργία του εκάστοτε χώρου εργασίας. Αντίστοιχα λάθη μπορούν εύκολα να περιοριστούν με την λειτουργία της «επιλογής στηλών», ελαχιστοποιώντας τα δεδομένα εμφάνισης του προγράμματος στην οθόνη του χρήστη.

4.4 Κεντρική Εφαρμογή Work Station

Η Κεντρική Εφαρμογή που αποκαλούμε «Work Station», αποτελεί μια Web Based εφαρμογή. Χρήστες της εφαρμογής αυτής μπορεί να είναι μόνο διοικητικοί υπάλληλοι, ιατροί ή νοσηλευτές αρμόδιοι για την δημιουργία, τροποποίηση, έκδοση και ανανέωση των προγραμμάτων εφημεριών/καθηκόντων, που αφορούν μια ή και περισσότερες ομάδες εργαζομένων. Κάθε χρήστης της εφαρμογής «Work Station» συνδέεται με διαφορετικό λογαριασμό, γεγονός που σημαίνει ότι ανάλογα με την ιδιότητα και τις αρμοδιότητες του, κάθε χρήστης διαθέτει διαφορετικά δικαιώματα χρήσης των λειτουργιών της εφαρμογής. Έτσι εάν σε έναν διοικητικό υπάλληλο δεν έχει δοθεί η αρμοδιότητα της έγκρισης ή απόρριψης των αιτήσεων αλλαγής των εργαζομένων, δεν μπορεί ο συγκεκριμένος υπάλληλος να κάνει χρήση της συγκεκριμένης λειτουργίας και θα περιοριστεί μόνο στην εγγραφή και επεξεργασία των χρηστών του συστήματος.

Μία από τις λειτουργίες της συγκεκριμένης εφαρμογής αποτελεί η διαχείριση των δεδομένων που βρίσκονται αποθηκευμένα στην Κεντρική Βάση Δεδομένων (ΚΒΔ). Από την συγκεκριμένη ιδιότητά της έλαβε και τον χαρακτηρισμό της «Κεντρικής Εφαρμογής». Δηλαδή μέσω της εφαρμογής δίνεται η δυνατότητα στον εκάστοτε αρμόδιο χρήστη να διαχειρίζεται τα δεδομένα που αφορούν τους χρήστες (εγγραφή, διαγραφή, τροποποίηση στοιχείων κ.α.), τα δεδομένα αντικειμένων (προγράμματα, μηνύματα κ.α.) και γενικά ότι πρέπει και μπορεί να εισαχθεί από ένα χρήστη.

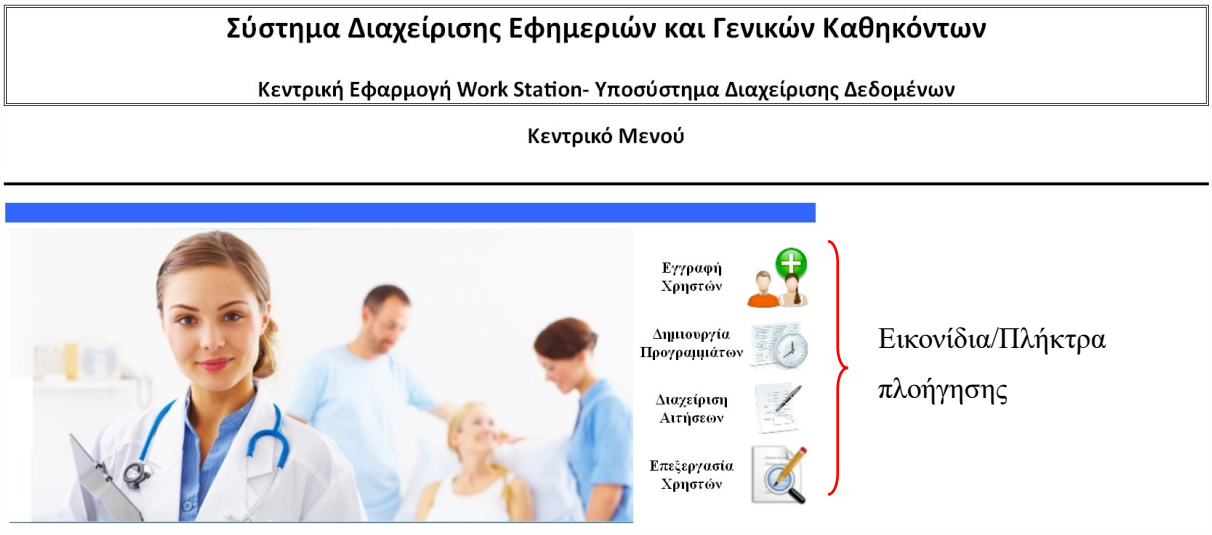
Μία ακόμη ιδιότητα της συγκεκριμένης εφαρμογής είναι ο ρόλος του οργανωτή του συστήματος. Σκοπός είναι η δημιουργία προγραμμάτων εργασίας και εφημεριών για τους

χρήστες του συστήματος καθώς και η ανανέωση ή τροποποίησή τους μέσω της διαδικασίας έγκρισης ή απόρριψης των αλλαγών που προτείνουν οι χρήστες των φορητών εφαρμογών.

Το γεγονός ότι η συγκεκριμένη εφαρμογή είναι Web Based εξυπηρετεί την λειτουργία της από οποιοδήποτε μέρος και σε οποιοδήποτε ηλεκτρονικό υπολογιστή, εντός ή εκτός των χώρων της νοσοκομειακής μονάδας. Επίσης αποτελεί λύση για την εξουσιοδότηση των αρμόδιων ατόμων που ασχολούνται με την δημιουργία ή την τροποποίηση των προγραμμάτων.

4.4.1 Αρχική Σελίδα Κεντρικής Εφαρμογής-Work Station

Η αρχική σελίδα της Κεντρικής Εφαρμογής-Work Station είναι η πρώτη σελίδα που εμφανίζεται στον χρήστη με την έναρξη της εφαρμογής. Χρησιμοποιείται για την πλοήγηση στις λειτουργίες και τις υπηρεσίες της εφαρμογής με την βοήθεια χαρακτηριστικών εικονιδίων που οδηγούν τον χρήστη σε κάθε μία από αυτές.



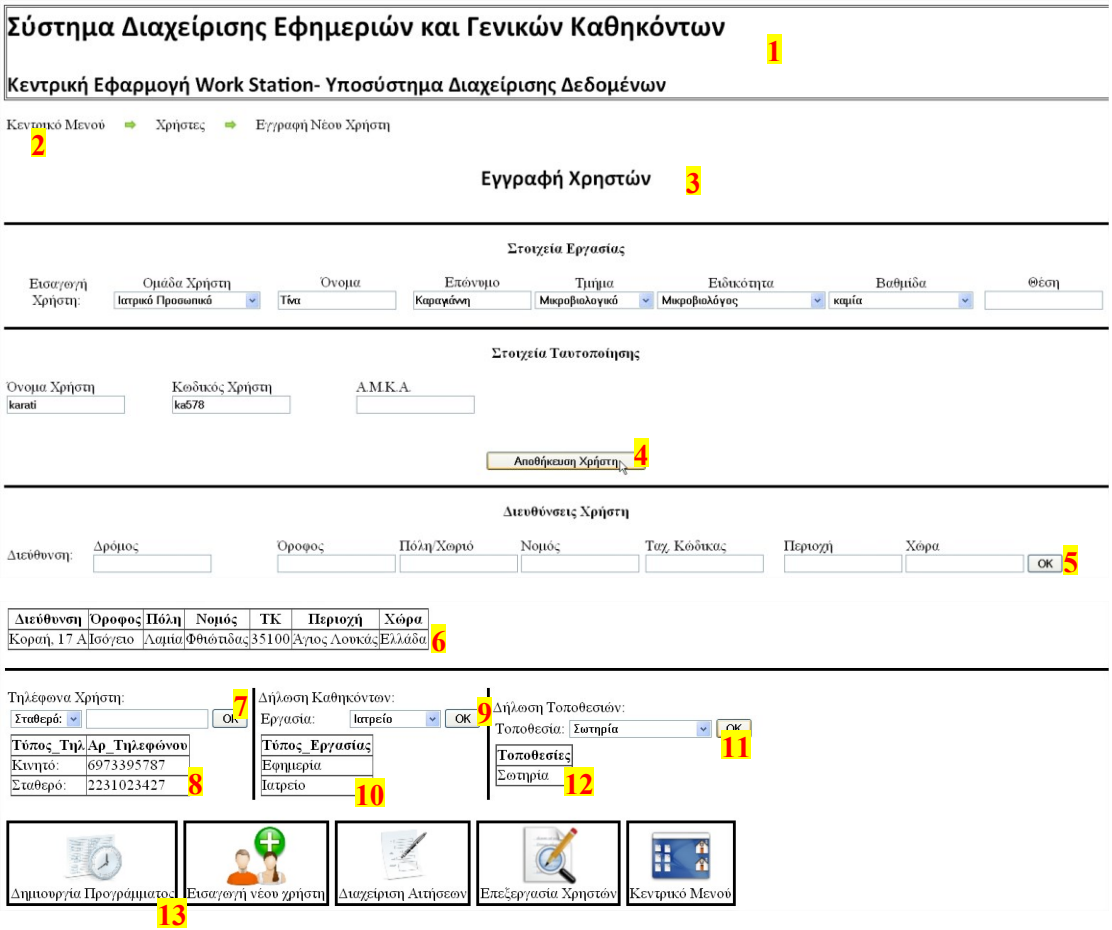
Εικόνα 34: Αρχική σελίδα της Κεντρικής Εφαρμογής-Work Station

4.4.2 Εγγραφή νέου χρήστη στο σύστημα αυτόματης διαχείρισης εφημεριών και καθηκόντων

Η εγγραφή των νέων χρηστών του συστήματος πραγματοποιείται από μία φόρμα συμπλήρωσης των προσωπικών στοιχείων του χρήστη, των στοιχείων επικοινωνίας με τον χρήστη καθώς και όλων εκείνων των στοιχείων που είναι απαραίτητα για την λειτουργία του συστήματος. Τα στοιχεία αυτά είναι: το όνομα χρήστη, ο μυστικός κωδικός, η δήλωση εργασιών και τοποθεσιών προς ενημέρωση.

Ο αρμόδιος χρήστης ο οποίος θα χειρίζεται την ιστοσελίδα, σε πρώτη φάση συμπληρώνει όλα εκείνα τα δεδομένα που χαρακτηρίζουν έναν χρήστη ως οντότητα της νοσοκομειακής μονάδας και τα οποία είναι: η ομάδα χρήστη (Ιατρικό, Νοσηλευτικό ή Διοικητικό Προσωπικό), όνομα, επώνυμο, τμήμα της νοσοκομειακής μονάδας στην οποία θα ανήκει ο χρήστη, η ειδικότητα του χρήστη, η βαθμίδα του στην ιεραρχία και τέλος η θέση του ανάμεσα στους εργαζομένους ή στην νοσοκομειακή μονάδα. Με το πάτημα του πλήκτρου «Αποθήκευση Χρήστη», ο χρήστης της Κεντρικής Εφαρμογής ενεργοποιεί όλους εκείνους του μηχανισμούς που πυροδοτούν την έναρξη της εγγραφής των στοιχείων αυτών στους κατάλληλους πίνακες της Κεντρικής Βάσης Δεδομένων.

Στη συνέχεια ο χρήστης της εφαρμογής συμπληρώνει επιπλέον στοιχεία για την εγγραφή του και τα οποία είναι: τηλέφωνα και διευθύνσεις επικοινωνίας καθώς και η δήλωση των εργασιών και τοποθεσιών για τις οποίες θα ενημερώνεται. Η εισαγωγή καθενός από τα παραπάνω στοιχεία απαιτεί την επικύρωσή τους με το πάτημα του πλήκτρου «OK» που βρίσκεται στα δεξιά κάθε πλαισίου συμπλήρωσης.



Εικόνα 35: Φόρμα εγγραφής νέων χρηστών του συστήματος

Βασικά Χαρακτηριστικά:

1: Τίτλος εφαρμογής.

2: Επισκόπηση πλοήγησης.

3: Τίτλος φόρμας.

4,5,7,9,11: Πλήκτρα επικύρωσης εισαγωγής δεδομένων.

6,8,10,12: Πίνακες επισκόπησης των νέων εγγραφών.

13: Πλήκτρα πλοήγησης στις φόρμες της εφαρμογής.

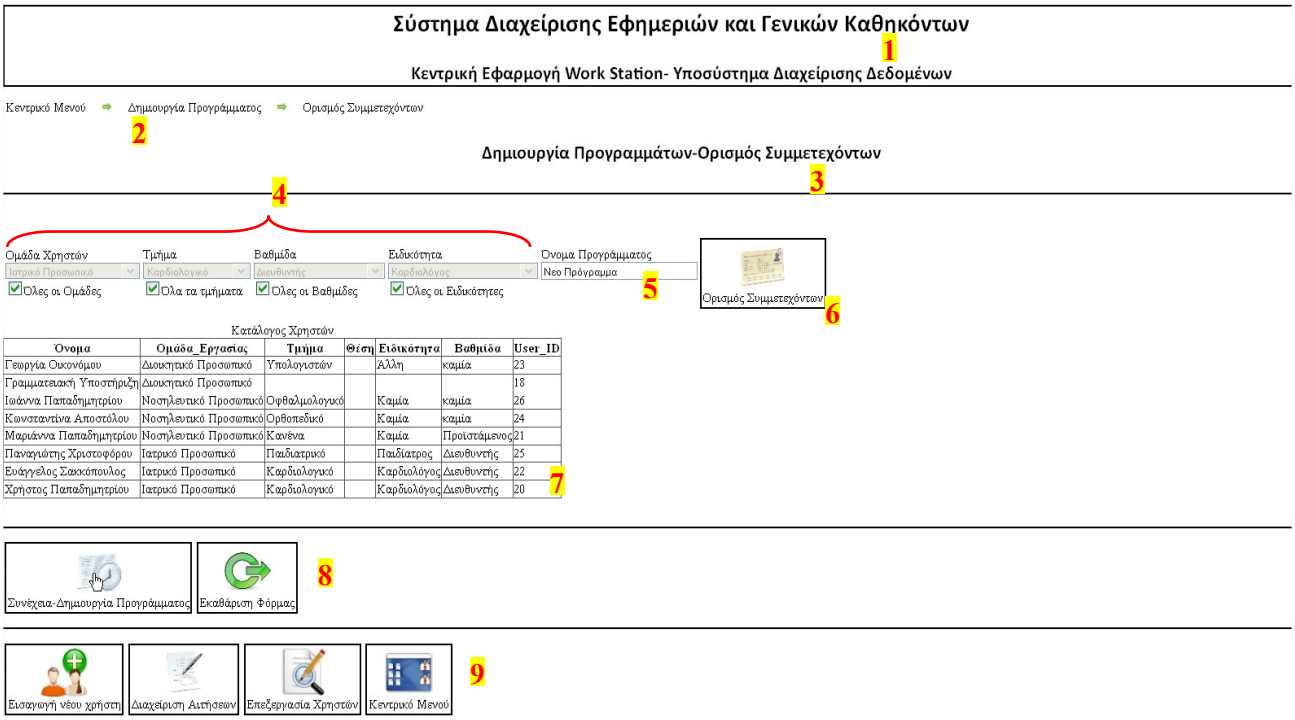
4.4.3 Διαδικασία δημιουργίας προγράμματος εφημεριών/καθηκόντων

Ο σχεδιασμός της εφαρμογής για την πραγματοποίηση της δημιουργίας προγραμμάτων εφημεριών και καθηκόντων έχει γίνει με γνώμονα την γρήγορη, εύκολη και χωρίς λάθη διεκπεραίωση της συγκεκριμένης λειτουργίας. Προκειμένου ο στόχος αυτός να είναι υλοποιήσιμος, ο χρήστης της εφαρμογής θα πρέπει να έχει πλήρη επίγνωση του προγράμματος που θέλει να δημιουργήσει καθώς και των εργαζομένων για τους οποίους δημιουργείται. Η διαδικασία της δημιουργίας προγράμματος εφημεριών/καθηκόντων αποτελείται από δύο στάδια. Στο πρώτο στάδιο ο χρήστης της Κεντρικής Εφαρμογής ορίζει τα άτομα ή τις ομάδες ατόμων για τους οποίους προορίζεται το πρόγραμμα εφημεριών/καθηκόντων, ενώ στο δεύτερο στάδιο η διαδικασία εξατομικεύεται δίνοντας στον χρήστη την δυνατότητα να καθορίσει τα στοιχεία του προγράμματος για κάθε εργαζόμενο ξεχωριστά.

Τα άτομα στα οποία απευθύνεται κάθε πρόγραμμα που δημιουργείται είναι και αυτά που θα το λαμβάνουν στις φορητές τους εφαρμογές και στα οποία θα αποστέλλονται οι τυχών ενημερώσεις ή τροποποιήσεις του.

4.4.3.1 Πρώτο στάδιο της διαδικασίας δημιουργίας προγράμματος εφημεριών/καθηκόντων

Στο πρώτο στάδιο της διαδικασίας για την δημιουργία προγράμματος ο χρήστης της Κεντρικής Εφαρμογής (Work Station) καθορίζει τα άτομα ή τις ομάδες ατόμων για τα οποία προορίζεται το πρόγραμμα που πρόκειται να δημιουργήσει στην συνέχεια.



Εικόνα 36: Φόρμα Δημιουργία Προγραμμάτων-Ορισμός Συμμετεχόντων

Βασικά Χαρακτηριστικά:

1: Τίτλος εφαρμογής.

2: Επισκόπηση πλοήγησης.

3: Τίτλος φόρμας.

4: Πλαίσια εισαγωγής χαρακτηριστικών αναζήτησης.

5: Πλαίσιο εισαγωγής ονόματος του προγράμματος που πρόκειται να δημιουργηθεί.

6: Πλήκτρο ενεργοποίησης της αναζήτησης εργαζομένων βάση χαρακτηριστικών.

7: Πίνακας εργαζομένων που ανταποκρίνονται στα χαρακτηριστικά αναζήτησης.

8: Πλήκτρα λειτουργιών φόρμας.

9: Πλήκτρα πλοήγησης στις φόρμες της εφαρμογής.

Ο αρμόδιος χρήστης έχει την δυνατότητα δημιουργίας προγράμματος σύμφωνα με την ομάδα στην οποία ανήκουν οι εργαζόμενοι (Ιατρικό, Νοσηλευτικό ή Διοικητικό Προσωπικό), το τμήμα του νοσοκομείου ή της κλινικής για την οποία προορίζεται το πρόγραμμα καθώς επίσης και κατά ειδικότητα ήκαι βαθμίδα (ιεραρχία) των εργαζομένων. Οι επιλογές αυτές μπορούν να εφαρμοστούν κατά αποκλειστικότητα των επιλογών, δηλαδή δημιουργία προγράμματος μόνο για το ιατρικό προσωπικό ή μόνο για τους Ιατρούς

Καρδιολόγους ή να εφαρμοστούν σε συνδυασμό, π.χ. δημιουργία προγράμματος για Ιατρούς Καρδιολόγους που είναι Επιμελητές.

Αφού ο χρήστης καθορίσει τους εργαζόμενους για τους οποίους προορίζεται το πρόγραμμα εφημεριών/καθηκόντων, θα πρέπει να εισάγει ένα όνομα για το πρόγραμμα που πρόκειται να δημιουργήσει. Το όνομα του προγράμματος αποτελεί θετικό παράγοντα για την αποτελεσματικότερη λειτουργία του συστήματος ενημερώσεων. Το όνομα του προγράμματος θα πρέπει όμως να είναι σχετικό με το περιεχόμενό του ή με την ομάδα των εργαζομένων στους οποίος απευθύνεται, π.χ. πρόγραμμα νοσηλευτών καρδιολογικής κλινικής 32^{ης} εβδομάδας.

Πατώντας το πλήκτρο «Δημιουργία» εμφανίζεται στο χρήστη ένας κατάλογος με τα ονόματα των εργαζομένων οι οποίοι ανταποκρίνονται στα χαρακτηριστικά τα οποία έχει εισάγει ο χρήστης της εφαρμογής. Η συγκεκριμένη λίστα παρουσιάζει τα ονόματα των εργαζομένων για τους οποίους προορίζεται το πρόγραμμα που πρόκειται να δημιουργηθεί. Η λογική για την εμφάνιση της λίστας αυτής είναι ο έλεγχος για το αν τα άτομα που εμφανίζονται ανταποκρίνονται στα κριτήρια που έχει εισάγει ο χρήστης και αντιστρόφως αν τα κριτήρια που έχει εισάγει ο χρήστης συμπεριλαμβάνουν τα επιθυμητά άτομα. Με το πάτημα του πλήκτρου «Συνέχεια» η διαδικασία της δημιουργίας προγράμματος εφημεριών/καθηκόντων συνεχίζεται ανοίγοντας μια νέα φόρμα συμπλήρωσης στην οθόνη του χρήστη στην οποία και πραγματοποιείται η εξατομίκευση της διαδικασίας εισάγοντας τα στοιχεία του προγράμματος που αφορούν κάθε εργαζόμενο ξεχωριστά.

4.4.3.2 Δεύτερο στάδιο, ολοκλήρωση της διαδικασίας δημιουργίας προγράμματος εφημεριών/καθηκόντων

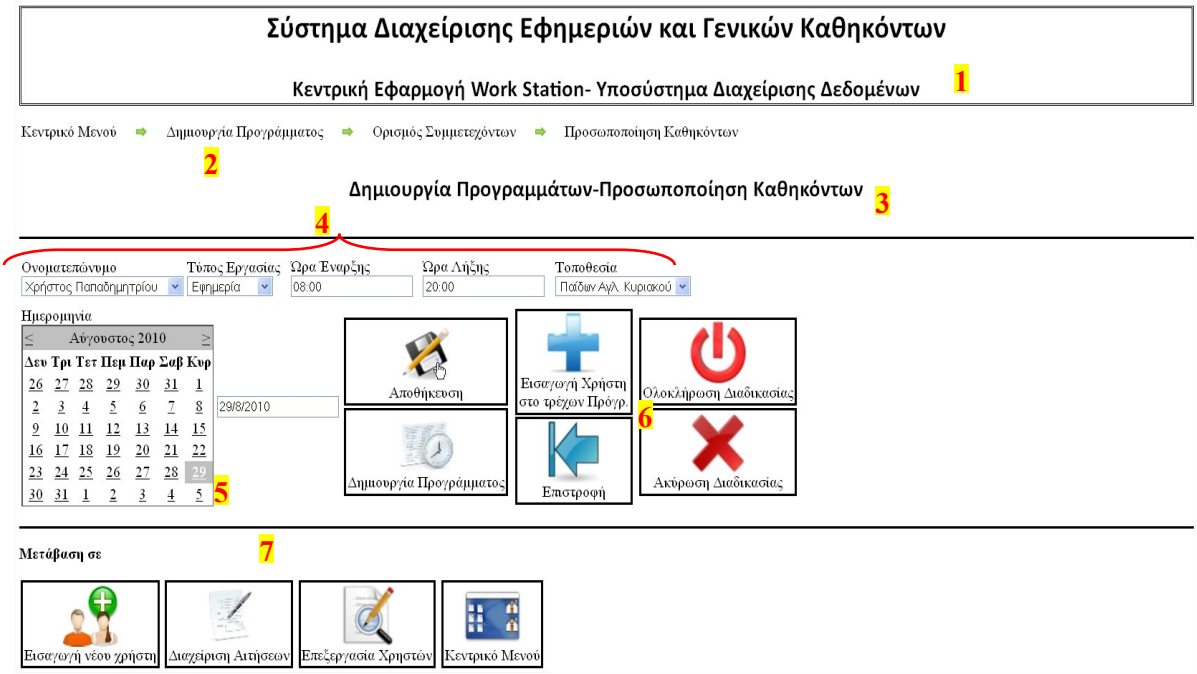
Έχοντας εισάγει τα στοιχεία και το όνομα του προγράμματος που πρόκειται να δημιουργήσει κατά το πρώτο στάδιο της διαδικασίας, ο χρήστη πατώντας το πλήκτρο «Συνέχεια» καλείται να ολοκληρώσει την διαδικασία δημιουργίας προγράμματος με την επιλογή των εργαζομένων στους οποίους απευθύνεται το πρόγραμμα και με τις λεπτομέρειες των εφημεριών/καθηκόντων που αντιστοιχούν σε κάθε εργαζόμενο.

Ο χρήστης της Κεντρικής Εφαρμογής από την νέα φόρμα που εμφανίζεται στην οθόνη του, επιλέγει τους εργαζομένους μέσα από μία λίστα με ονοματεπώνυμα, ίδια με αυτή που παρουσιάστηκε στον χρήστη κατά το πρώτο βήμα της διαδικασίας δημιουργίας του προγράμματος. Ο χρήστης εξακολουθεί να εισάγει τις λεπτομέρειες του προγράμματος για κάθε εργαζόμενο ξεχωριστά καθορίζοντας το είδος της εργασίας, την ώρα έναρξης, την ώρα λήξης, την τοποθεσία της εργασίας και την ημερομηνία στην οποία θα πρέπει να πραγματοποιηθεί η εργασία. Τα περισσότερα από αυτά συμπληρώνονται με

αυτοματοποιημένο τρόπο καθώς το περιεχόμενο των τύπων εργασίας και των τοποθεσιών είναι συγκεκριμένο και καθορίζεται από τον τρόπο οργάνωσης και την κάλυψη υπηρεσιών που προσφέρει κάθε νοσοκομειακή μονάδα.

Προκειμένου να γίνει η επικύρωση των στοιχείων που εισάγει ο χρήστης πρέπει να πατάει το πλήκτρο «ΟΚ» που βρίσκεται στα δεξιά κάθε πλαισίου συμπλήρωσης. Τα στοιχεία του προγράμματος για κάθε εργαζόμενο εισέρχονται στο υπό δημιουργία πρόγραμμα με το πάτημα του πλήκτρου «Αποθήκευση», το οποίο ενεργοποιεί τους μηχανισμούς εγγραφής του προγράμματος στην Κεντρική Βάση Δεδομένων και την δημοσίευσή του στους χρήστες του συστήματος για τους οποίους απευθύνεται. Η διαδικασία της δημιουργίας του προγράμματος ολοκληρώνεται με το πάτημα του πλήκτρου «Ολοκλήρωση Δημιουργίας Προγράμματος». Ταυτόχρονα στο τέλος της σελίδας εμφανίζεται προεπισκόπηση του προγράμματος που δημιουργεί ο χρήστης.

Στον χρήστη, με την ύπαρξη κατάλληλων πλήκτρων, παρέχεται η δυνατότητα Αυτόματης Εισαγωγής Νέου Χρήστη, Δημιουργίας Νέου Προγράμματος, Επιστροφής στην Προηγούμενη Σελίδα για την τυχών διόρθωση σφαλμάτων και τέλος δυνατότητα Ολικής Ακύρωσης της διαδικασίας δημιουργίας προγράμματος, συμπεριλαμβανομένης και της διαγραφής των καταγραφών που υπάρχουν στο υπό δημιουργία πρόγραμμα.



Εικόνα 37: Φόρμα δημιουργίας προγραμμάτων-προσωποποίηση καθηκόντων

Βασικά Χαρακτηριστικά:

1: Τίτλος εφαρμογής.

2: Επισκόπηση πλοήγησης.

3: Τίτλος φόρμας.

4: Πλαίσια εισαγωγής χαρακτηριστικών εφημερίας/εργασίας.

5: Αντικείμενο «Calendar» για τον καθορισμό της ημερομηνίας πραγματοποίησης της εφημερίας.

6: Πλήκτρα λειτουργιών φόρμας.

7: Πλήκτρα πλοήγησης στις φόρμες της εφαρμογής.

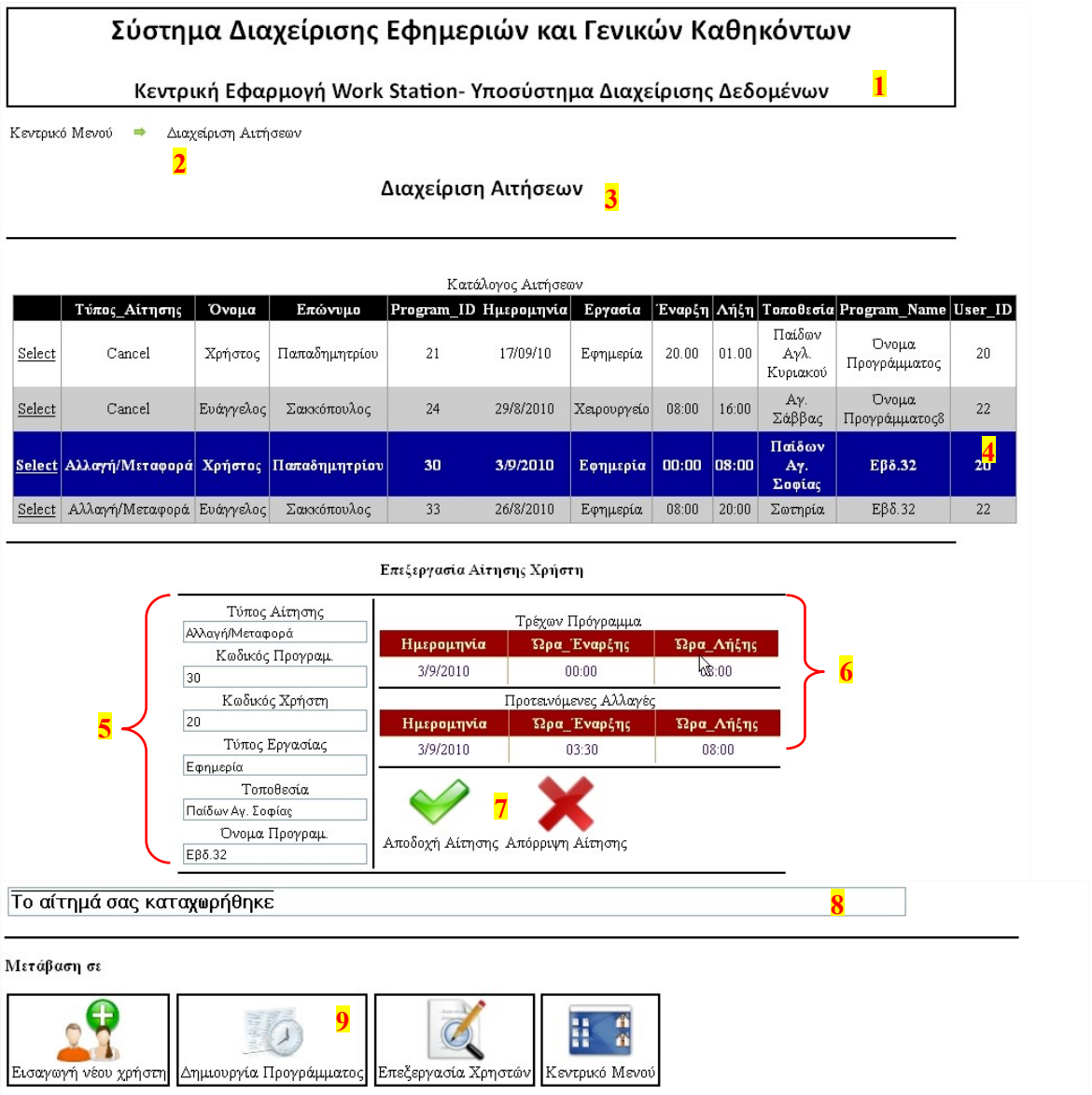
4.4.4 Διαδικασία τροποποίησης προγράμματος μετά από αίτηση χρήστη φορητής εφαρμογής

Μια από τις πιο σημαντικές υπηρεσίες του συστήματος είναι η δυνατότητα τροποποίησης του προγράμματος εφημεριών/καθηκόντων ύστερα από την υποβολή αίτησης των χρηστών της φορητής εφαρμογής. Η ιστοσελίδα «Change Application Management» είναι υπεύθυνη για την διαχείριση των αιτήσεων και την τροποποίηση του προγράμματος εφημεριών/καθηκόντων.

Στο χρήστη της Κεντρικής Εφαρμογής δίνεται η δυνατότητα παρακολούθησης και διαχείρισης των αιτήσεων των χρηστών μέσα από έναν «Κατάλογο Αιτήσεων» που εμφανίζεται στην οθόνη. Ο κατάλογος αυτός περιέχει όλες εκείνες τις αιτήσεις που έχουν υποβάλει οι χρήστες του συστήματος μέσω των φορητών εφαρμογών, ανεξάρτητα από τον τύπο της αίτησης (αίτηση ακύρωσης, αλλαγής ώρας, μεταφοράς εργασίας). Με το πάτημα του πλήκτρου «Select» που εμφανίζεται αριστερά από κάθε αίτηση, ο χρήστης επιλέγει την αντίστοιχη αίτηση για περεταίρω μελέτη και επεξεργασία. Στο σημείο αυτό θα πρέπει να υπενθυμίσουμε ότι ο χρήστης της Κεντρικής Εφαρμογής μπορεί να είναι ένας από τους υπεύθυνους προσωπικού, διευθυντής τμήματος, προϊστάμενος ή και διοικητικός υπάλληλος που έχει τη δικαιοδοσία για την πραγματοποίηση αλλαγών και τροποποιήσεων του προγράμματος εφημεριών/καθηκόντων.

Η μελέτη της αίτησης περιλαμβάνει την συγκέντρωση του ενδιαφέροντος του χρήστη στην συγκεκριμένη αίτηση μέσα από την εμφάνιση των λεπτομερειών που την περικλείουν. Ο χρήστης είναι σε θέση να βλέπει τα απαραίτητα χαρακτηριστικά στοιχεία για κάθε αίτηση και να «συγκρίνει» τις αλλαγές που προτείνουν οι χρήστες μέσω των αιτήσεων τους, με το τρέχων πρόγραμμα που εμπεριέχει την συγκεκριμένη εργασία. Έτσι ο χρήστης της Κεντρικής Εφαρμογής μπορεί να μεταβεί στην φάση της επεξεργασίας και να αποφασίσει εάν θα κάνει δεκτή ή θα απορρίψει την αίτηση του χρήστη. Όποια και να είναι η απόφαση του χρήστη της

Κεντρικής Εφαρμογής, ο χρήστης της φορητής εφαρμογής, ο οποίος υπέβαλε την συγκεκριμένη αίτηση, θα ενημερωθεί για την έκβαση της αίτησής του μέσω των ενημερώσεων του προγράμματός του.



Εικόνα 38: Φόρμα επεξεργασίας αιτήσεων

Βασικά Χαρακτηριστικά:

1: Τίτλος εφαρμογής.

2: Επισκόπηση πλοήγησης.

3: Κατάλογος αιτήσεων προς επεξεργασία.

4: Πλαίσια εισαγωγής χαρακτηριστικών εφημερίας/εργασίας.

5: Στοιχεία ταυτοποίησης της υπό επεξεργασίας αίτησης.

6: Πίνακες σύγκρισης-επεξεργασίας στοιχείων αίτησης.

7: Πλήκτρα λειτουργιών της φόρμας.

8: Πλαίσιο έκβασης της αίτησης.

9: Πλήκτρα πλοήγησης σε φόρμες της εφαρμογής.

4.4.5 Διαδικασία τροποποίησης των στοιχείων εγγραφής του χρήστη

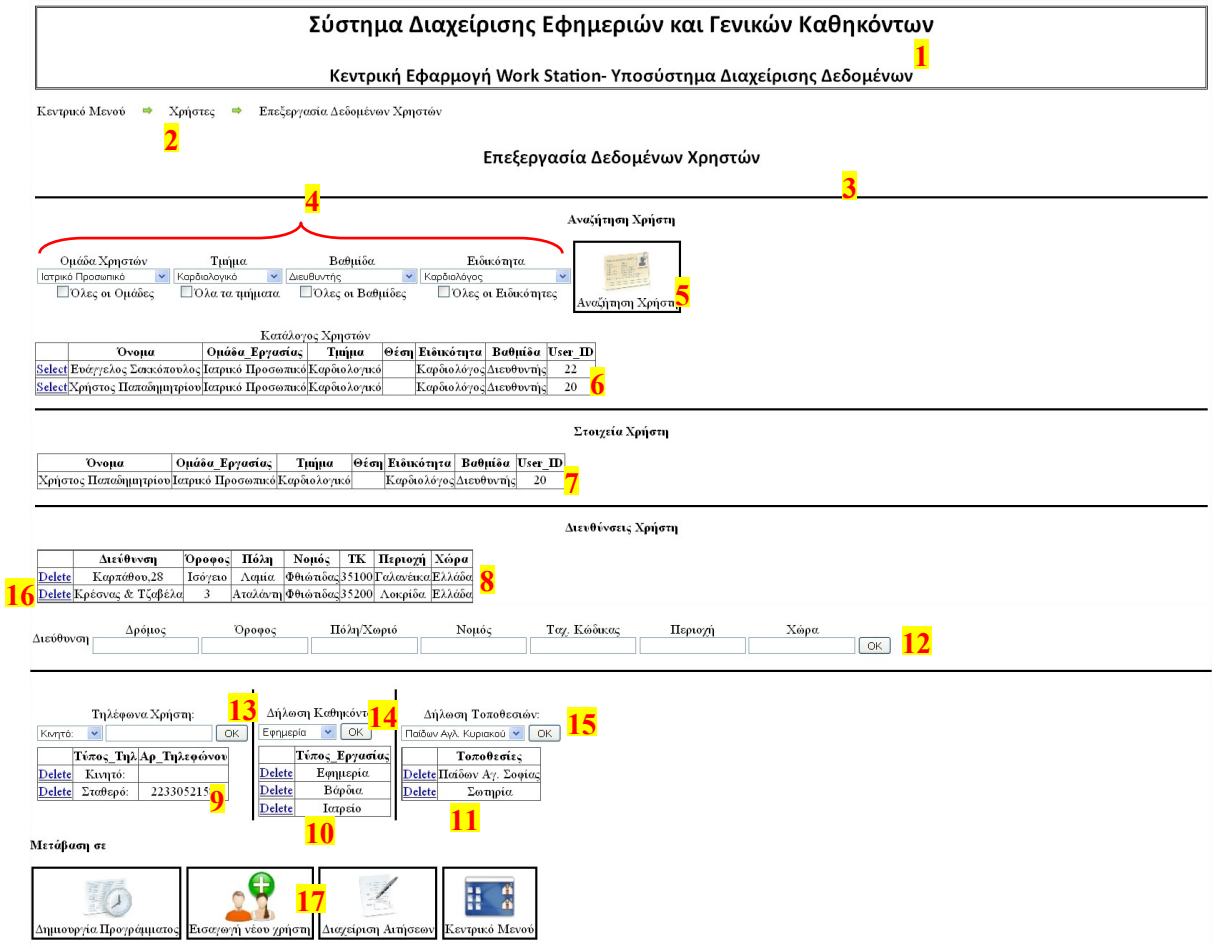
Μια ακόμη ιδιότητα της κεντρικής εφαρμογής «Work Station» είναι η δυνατότητα τροποποίησης των στοιχείων που δήλωσε ο χρήστης κατά την εγγραφή του στο σύστημα. Η ιστοσελίδα «Edit Users» είναι υπεύθυνη για την επεξεργασία και την τροποποίηση των στοιχείων αυτών. Υπάρχουν πολλοί λόγοι για την ύπαρξη της συγκεκριμένης φόρμας, όπως:

- Η διόρθωση πιθανών λαθών στα χαρακτηριστικά που συνέβησαν κατά την διαδικασία εγγραφής του χρηστή,
- πρόσθεση ή και η αφαίρεση στοιχείων όπως τηλέφωνα και διευθύνσεις επικοινωνίας,
- αναλυτική επισκόπηση όλων των στοιχείων ενός συγκεκριμένου εγγραμμένου χρήστη του συστήματος,
- ολική διαγραφή ενός χρήστη από το σύστημα

Προκειμένου να είναι εύκολη η διαδικασία ανεύρεσης ενός συγκεκριμένου ατόμου ή ομάδας ατόμων των οποίων μας ενδιαφέρουν τα χαρακτηριστικά τους, μία αναζήτηση στον πίνακα των εργαζομένων με βάση συγκεκριμένα χαρακτηριστικά που ορίζει ο εκάστοτε χρήστης της κεντρικής εφαρμογής. Οι εργαζόμενοι που ανταποκρίνονται στα χαρακτηριστικά εμφανίζονται σε έναν πίνακα/κατάλογο. Ο κατάλογος αυτός περιέχει μόνο τα χαρακτηριστικά αυτά που βοηθούν στην ταυτοποίηση ενός χρήστη και τον διαχωρισμό του από έναν άλλον. Με το πάτημα του πλήκτρου «Select» που εμφανίζεται αριστερά από κάθε χρήστη/εργαζόμενο που παρουσίασε η ρουτίνα της αναζήτησης, ο χρήστης της κεντρικής εφαρμογής επιλέγει την αντίστοιχη εγγραφή για περαιτέρω μελέτη και επεξεργασία. Στο σημείο αυτό θα πρέπει να υπενθυμίσουμε ότι ο χρήστης της Κεντρικής Εφαρμογής μπορεί να είναι ένας από τους υπευθύνους προσωπικού, διευθυντής τμήματος, προϊστάμενος ή και διοικητικός υπάλληλος που έχει την δικαιοδοσία για την πραγματοποίηση αλλαγών και τροποποιήσεων του προγράμματος εφημεριών/καθηκόντων.

Η μελέτη μίας εγγραφής περιλαμβάνει την παρουσίαση όλων των στοιχείων εκείνων που σχετίζονται με τον αντίστοιχο χρήστη, όπως τηλέφωνα, διευθύνσεις και τις δηλώσεις των καθηκόντων και των τοποθεσιών, έτσι ο χρήστης της κεντρικής εφαρμογής είναι σε θέση να βλέπει τα απαραίτητα χαρακτηριστικά στοιχεία για κάθε εγγραφή και να τα προσαρμόζει

ανάλογα. Ο χρήστης της φορητής εφαρμογής ενημερώνεται άμεσα από την έκβαση των αλλαγών μέσω πινάκων που εμφανίζονται σε κάθε ομάδα χαρακτηριστικών.



Εικόνα 39: Φόρμα επεξεργασίας στοιχείων χρηστών

Βασικά Χαρακτηριστικά:

1: Τίτλος εφαρμογής.

2: Επισκόπηση πλοήγησης.

3: Τίτλος φόρμας.

4: Πλαίσια εισαγωγής χαρακτηριστικών αναζήτησης.

5: Πλήκτρο ενεργοποίησης της αναζήτησης εργαζομένων βάση χαρακτηριστικών

6: Πίνακας εργαζομένων που ανταποκρίνονται στα χαρακτηριστικά αναζήτησης.

7: Πίνακας με τα στοιχεία ταυτοποίησης του επιλεγμένου προς επεξεργασία χρήστη.

8,9,10,11: Πίνακες επισκόπησης υπαρχόντων στοιχείων και αλλαγών.

12,13,14,15: Πλήκτρα επικύρωσης εισαγωγής δεδομένων.

16: Πλήκτρο διαγραφής επαφής.

17: Πλήκτρα πλοήγησης στις φόρμες της εφαρμογής.

5

Υλοποίηση Συστήματος

5.1 Υλοποίηση Συναλλαγών του Web Service με τις Βάσεις Δεδομένων

Η πρόσβαση των Web services και των εφαρμογών στις βάσεις δεδομένων είναι ένα χαρακτηριστικό του συστήματος εφόσον σε ένα μεγάλο μέρος του κώδικά μας υπάρχουν συναλλαγές με αυτή. Το ίδιο σημαντικό είναι και η επικοινωνία των ΒΔ με την ίδια την εφαρμογή. Κρίνεται σκόπιμο επομένως, να παρουσιάσουμε πώς επικοινωνούμε με την SQL Server βάση δεδομένων, πώς θέτουμε ερωτήματα και πώς παίρνουμε τα αντίστοιχα αποτελέσματα.

5.1.1 Επικοινωνία Web Service - Απομακρυσμένου Κεντρικού Υπολογιστή

Προκειμένου να μεταφερθούν δεδομένα από τον κεντρικό υπολογιστή στην εφαρμογή που υπάρχει στην φορητή συσκευή και αντίστροφα, μεσολαβούν οι υπηρεσίες Web Service. Οι υπηρεσίες αυτές αντλούν δεδομένα από τη ΒΔ ***System Data*** που βρίσκεται στον απομακρυσμένο κεντρικό υπολογιστή και τα μεταβιβάζουν στην μέθοδο η οποία τις κάλεσε. Για να κάνουμε χρήση των υπηρεσιών αυτών, αρχικά πρέπει να συμπεριλάβουμε τον SQL Server στην εφαρμογή της Web service υπηρεσίας, το οποίο γίνεται προσθέτοντας τις

κατάλληλες αναφορές (references) που είναι στην πραγματικότητα βοηθητικές βιβλιοθήκες (dll), εδώ System.Data.SqlClient. Επιπλέον ορίζουμε στην αρχή του κώδικα τους χώρους ονομάτων τους ώστε να διευκολυνθεί η πρόσβαση σε αυτές:

using System.Data.SqlClient;

Κάθε φορά που απαιτείται η ανταλλαγή δεδομένων του Web Service με τη ΒΔ SystemData, εκτελείται το ακόλουθο τμήμα κώδικα που χρησιμοποιεί το connection string προκειμένου να ανοίξει μια σύνδεση με τη βάση.

```
string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;UserId=xpchris;Password=123456";
SqlConnection conn = new SqlConnection(sConn2);
conn.Open();
```

Πλαίσιο Κειμένου 1: Χρήση του connection string για την δημιουργία σύνδεσης με την ΚΒΔ

Στην προκειμένη περίπτωση πηγή δεδομένων αποτελεί η ΒΔ SystemData. Απαραίτητη προϋπόθεση προκειμένου να ορίσουμε στο αντικείμενο SqlConnection αποτελεί η υπόδειξη της τοποθεσίας (το μονοπάτι) στην οποία βρίσκεται η πηγή δεδομένων με την οποία επιθυμούμε να συνδεθούμε και ονομάζεται connection string.

Επίσης, για να πραγματοποιήσουμε τα ερωτήματα προς την ΒΔ πρέπει να δημιουργήσουμε μία μεταβλητή τύπου SqlCommand η οποία περιέχει το ερώτημα προς την βάση καθώς και την μεταβλητή της σύνδεσης με την ΒΔ (SqlConnection). Ακολουθεί παράδειγμα ερωτήματος (πλαίσιο κειμένου 2) με το οποίο η υπηρεσία Web Service αντλεί από τη ΒΔ SystemData το όνομα του Προγράμματος στο οποίο ανήκει η εργασία/εφημερία με κωδικό αντικειμένου 'Pr01'.

```
string sConn2 = "data source=FERRARI-4000\\SQLEXPRESS;Database=SystemData;User
Id=xpchris;Password=123456";

SqlConnection conn = new SqlConnection(sConn2);
conn.Open();

System.Data.SqlClient.SqlCommand cmd = new SqlCommand("SELECT Program_Name FROM
Program WHERE (Item_ID = 'Pr01')", conn);

System.Data.SqlClient.SqlDataReader dr = cmd.ExecuteReader();
dr.Read();

return (dr["Program_Name"].ToString());
```

Πλαίσιο Κειμένου 2: Παράδειγμα για την υποβολή ερωτήματος από το Web Service στην ΚΒΔ με σκοπό την ανάκτηση δεδομένων

5.1.2 Επικοινωνία Mobile Εφαρμογής- SQL Server Compact Edition

Όπως είδαμε και προηγουμένως, στην περίπτωση της επικοινωνίας Web Service-Απομακρυσμένου Κεντρικού Υπολογιστή αρχικά πρέπει να συμπεριλάβουμε τον SSCE (SQL Server Compact Edition) στην εφαρμογή της φορητής συσκευής μας, το οποίο γίνεται προσθέτοντας τις κατάλληλες αναφορές (references). Στην συγκεκριμένη περίπτωση η

«Reference» που πρέπει να προσθέσουμε είναι η «System.Data.SqlServerCe». Επαναλαμβάνουμε ότι ο ορισμός των αναφορών πραγματοποιείται στην αρχή του κώδικα, μαζί με τα υπόλοιπα «namespaces» (χώρους ονομάτων) , ώστε να διευκολυνθεί η πρόσβαση σε αυτές: using System.Data.SqlServerCe;

Αντίστοιχα με τα αντικείμενα SqlConnection και SqlCommand που χρησιμοποιούνται για την επικοινωνία με την ΒΔ SystemData (SQL Server), στην περίπτωση της ΒΔ TimeTableDB (SSCE) χρησιμοποιούμε τα αντικείμενα SqlCeConnection και SqlCeCommand. Ακολουθεί παράδειγμα (πλαίσιο κειμένου 3) ερωτήματος με το οποίο μια συνάρτηση της εφαρμογής αντλεί από τη ΒΔ TimeTableDB (SSCE) τις εγγραφές (εργασίες) του πίνακα «Program» με ένα συγκεκριμένο κωδικό (ProgramID).

```
string s = string.Format("Program_ID='{0}'", ProgramID);

string cmd = "SELECT * FROM Program WHERE " + s;

SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
Files\TimeTable_V2\TimeTableDB.sdf");
connection.Open();

SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
SqlCeDataReader dr = SqlCmd.ExecuteReader();
dr.Read();
SqlCmd.ExecuteNonQuery();
```

Πλαίσιο Κειμένου 3: Παράδειγμα ερωτήματος από την φορητή εφαρμογή προς την τοπική ΒΔ

5.1.3 Επικοινωνία Web Service-Mobile Εφαρμογής

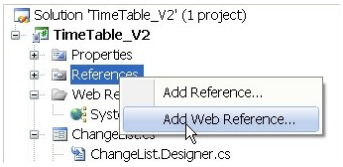
Για την επικοινωνία και την μεταφορά των δεδομένων από το Web Service προς την εφαρμογή και αντίστροφα, θα πρέπει να εισάγουμε στην εφαρμογή μας ένα Web Reference καθώς και τις απαραίτητες αναφορές (βιβλιοθήκες). Με την εισαγωγή του Web Reference ορίζουμε την υπηρεσία Web Service με την οποία θα επικοινωνεί η εφαρμογή μας. Ο ορισμός των αναφορών πραγματοποιείται στην αρχή του κώδικα, μαζί με τα υπόλοιπα «namespaces» (χώρους ονομάτων), ώστε να διευκολυνθεί η πρόσβαση σε αυτές:

```
using System.Web;
using System.Web.Services;
```

Πλαίσιο Κειμένου 4: Ορισμός αναφορών

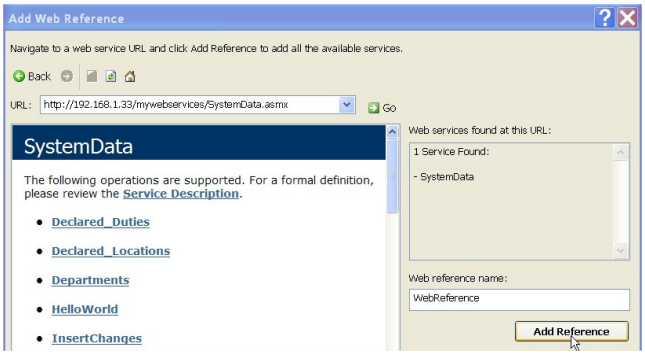
Προκειμένου να κάνουμε την εισαγωγή του Web Reference θα πρέπει να εκτελεστούν τα ακόλουθα βήματα:

- Στα δεξιά του παραθύρου της Visual Studio (στο υπο-παράθυρο Solution Explorer) κάνουμε δεξί κλικ στο Reference και μετά Add Web Reference.



Εικόνα 40: Εισαγωγή Web Reference

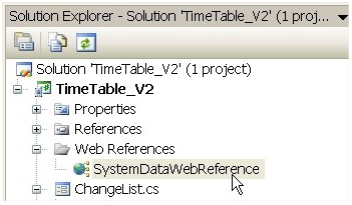
- Στο νέο παράθυρο με όνομα «Add Web Reference» εισάγουμε την URL διεύθυνση του Web Service που παρέχει τις υπηρεσίες που θέλουμε προσθέτοντας στη συνέχεια το όνομα του Web Service (.asmx) και πατάμε GO.



Εικόνα 41: Παράθυρο Add Web Reference

- Αν όλα τα βήματα δημοσίευσης του Web Service έχουν γίνει σωστά θα πρέπει να εμφανίζονται στο παράθυρο «Add Web Reference» οι συναρτήσεις-υπηρεσίες του Web Service. Έπειτα, στο ίδιο παράθυρο, καλούμαστε να δώσουμε ένα σχετικό όνομα στο Web Reference, π.χ. SystemDataWebReference. Ολοκληρώνουμε την διαδικασία πατώντας το πλήκτρο Add Reference.

- Παρατηρούμε ότι στο παράθυρο του Solution Explorer έχει προστεθεί το Web Reference με το όνομα που του έχουμε δώσει.



Εικόνα 42: Παράθυρο Solution Explorer

- Τώρα μπορούμε να προχωρήσουμε στη δημιουργία αντικειμένων με τα οποία αντλούμε τα δεδομένα από το Web Service.

- Για την πραγματοποίηση της επικοινωνίας Web Service - Mobile Εφαρμογής, αρχικοποιούμαι ένα αντικείμενο της κλάσης SystemData (=το όνομα του Web Service .asmx) με όνομα «TableContentWebService». Η κλάση αυτή ανήκει στο Web Reference «SystemDataWebReference» που έχουμε προσθέσει στην εφαρμογή.

```
SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();
```

Πλαίσιο Κειμένου 5: Αρχικοποίηση αντικειμένου Web Service

Σκοπός μας είναι να μεταφέρουμε τα δεδομένα που αφορούν τον χρήστη από τον κεντρικό υπολογιστή, μέσω του Web Service και να τα χρησιμοποιήσουμε με οποιονδήποτε τρόπο στις συναρτήσεις της mobile εφαρμογής. Η κλήση πλέον των υπηρεσιών του Web Service πραγματοποιείται με τον ίδιο τρόπο που καλούμε τις μεθόδους των βιβλιοθηκών. Δηλαδή το όνομα της κλάσης ακολουθούμενο από την μέθοδο που επιθυμούμε. Στην περίπτωση των Web Service, η κλάση αποτελείται από το αντικείμενο του Web Reference και η μέθοδος αντιστοιχεί στην υπηρεσία του Web Service που θέλουμε να χρησιμοποιήσουμε. Στο πλαίσιο που ακολουθεί παρουσιάζουμε παράδειγμα στο οποίο θα αποθηκεύσουμε στην μεταβλητή «ProgramDataTable» την τιμή που μας επιστρέφει το Web Service μέσω της συνάρτησης/υπηρεσίας «ProgramContent()».

```
ProgramDataTable = TableContentWebService.ProgramContent();
```

Πλαίσιο Κειμένου 6: Αρχικοποίηση μεταβλητής με τιμή που προέρχεται από το Web Sevice

Με αντίστοιχο τρόπο χρησιμοποιούμε τις κλάσεις και τις μεθόδους όταν θέλουμε να μεταβιβάσουμε δεδομένα από την mobile εφαρμογή στο Web Service. Στο ακόλουθο παράδειγμα τροφοδοτούμε την υπηρεσία «UpdateChanges» με τον κωδικό ενός χρήστη και τον κωδικό μιας εργασίας-εφημερίας ("xpchris", Program_ID.ToString()).

```
TableContentWebService.UpdateChanges("xpchris", Program_ID.ToString());
```

Πλαίσιο Κειμένου 7: Μεταβίβαση δεδομένων από την φορητή εφαρμογή προς το Web Service

5.2 Υλοποίηση Λειτουργιών Φορητής Εφαρμογής

Πριν την παρουσίαση του κώδικα της εφαρμογής θεωρούμε απαραίτητο να γίνει αναφορά και να δώσουμε τον ορισμό κάποιων κλάσεων και των αντικειμένων τους. Τα συγκεκριμένα αντικείμενα έχουν ιδιαίτερη σημασία για την εφαρμογή και θα πρέπει να γίνουν κατανοητά από τον αναγνώστη.

DataTable : Τα αντικείμενα της κλάσης DataTable αποτελούν σημεία αναφοράς για την βιβλιοθήκη της ADO.NET, καθώς χρησιμοποιούνται από άλλα αντικείμενα όπως το DataSet,

που θα αναλύσουμε στην επόμενη παράγραφο. Κάθε αντικείμενο της κλάσης DataTable αντιπροσωπεύει έναν πίνακα δεδομένων με γραμμές και στήλες.

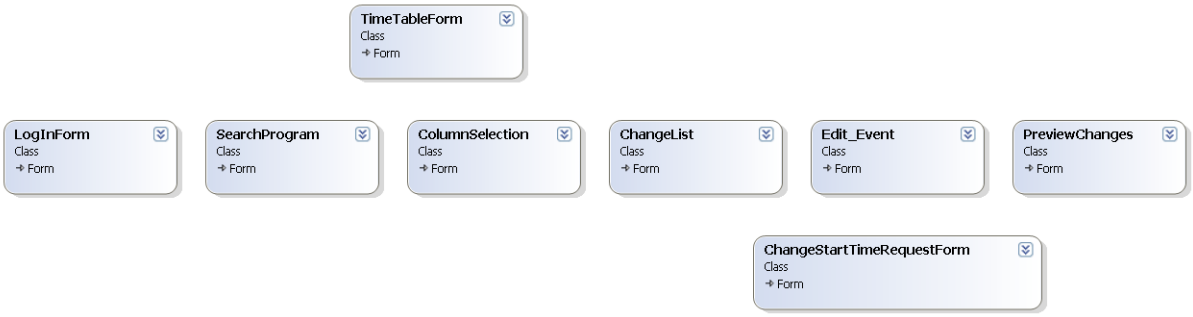
DataSet : Η κλάση **DataSet** αποτελεί ένα είδος μνήμης που συγκεντρώνει στοιχεία τα οποία ανακτώνται από μια πηγή δεδομένων όπως για παράδειγμα μια βάση δεδομένων. Ένα αντικείμενο της κλάσης DataSet, αποτελείται από μία συλλογή αντικειμένων της κλάσης DataTable. Ένα αντικείμενο DataSet έχει την δυνατότητα να μεταβιβάζει ή και να αντλεί δεδομένα, συμπεριλαμβανομένου του σχήματος (schema) που έχουν. Η καταγραφή ή άντληση δεδομένων πραγματοποιείται με την χρήση XML αρχείων. Τα XML αρχεία έχουν την ιδιότητα να συγκρατούν το σχήμα (schema) των δεδομένων που περιέχουν. Άλλη μια σημαντική ιδιότητα των XML αρχείων είναι ότι μπορούν να μεταφερθούν μέσω του πρωτοκόλλου HTTP και να χρησιμοποιηθούν από οποιαδήποτε εφαρμογή και σε οποιαδήποτε πλατφόρμα που υποστηρίζει την γλώσσα XML.

DataAdapter : Η κλάση **DataAdapter (SqlDataAdapter για την ΒΔ SystemData τύπου SqlServer ή SqlCeDataAdapter για την ΒΔ TimeTableDB τύπου Sql Server Compact Edition)** χρησιμεύει ως μια γέφυρα επικοινωνίας μεταξύ ενός αντικειμένου **DataSet** και μιας πηγής δεδομένων (data source). Χρησιμοποιείται για την ανάκτηση και αποθήκευση δεδομένων από και προς την πηγή δεδομένων. Η κλάση DataAdapter παρέχει αυτή τη γέφυρα μέσω ενός συνόλου εντολών για την διαχείριση των δεδομένων καθώς και της διασύνδεσης με την ΒΔ. Οι εντολές αυτές χρησιμοποιούνται για να γεμίσουν το **DataSet** και να ενημερώσουν την πηγή των δεδομένων (data source).

```
SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
SqlCeDataReader dr = SqlCmd.ExecuteReader();
```

Πλαίσιο Κειμένου 8: Τρόπος χρήσης των αντικειμένων "SqlCeDataReader"

Στο σημείο αυτό θα πρέπει να επαναλάβουμε ότι η εφαρμογή αποτελείται από 8 Windows Forms αντικείμενα (ή αλλιώς φόρμες), όλες οι φόρμες και οι κλάσεις της εφαρμογής βρίσκονται υπό το `namespace: TimeTable_V2` . Οι φόρμες είναι οι εξής: TimeTableForm, LogInForm, SearchProgram, ColumnSelection, ChangeList, Edit_Event, PreviewChanges και ChangeStartTimeRequestForm, οι οποίες υλοποιούνται από τις ομώνυμες κλάσεις. Κάθε φόρμα που προστίθεται από τον σχεδιαστή ή τον προγραμματιστή (developer) είναι μια GUI εφαρμογή πάνω στην οθόνη και παρέχει τη δυνατότητα προσθήκης Controls από το Toolbox του Visual Studio, όπως Buttons, Labels, Textboxes κλπ. Στη συνέχεια παρουσιάζεται ένα διάγραμμα με τις φόρμες της εφαρμογής.



Εικόνα 43: Διάγραμμα με τις φόρμες που περιέχει η φορητή εφαρμογή

5.2.1 Επικοινωνία των μεθόδων της φορητής εφαρμογής με την τοπική βάση δεδομένων SQL Server Compact Edition (SSCE)

Πριν περάσουμε στην αναλυτική επεξήγηση των μεθόδων της εφαρμογής, θα θέλαμε να παρουσιάσουμε πώς επικοινωνούμε με την SQL Server Compact Edition (SSCE) βάση δεδομένων, δηλαδή πώς θέτουμε ερωτήματα και πώς παίρνουμε τα αντίστοιχα αποτελέσματα.

Αρχικά πρέπει να συμπεριλάβουμε τον SSCE στην εφαρμογή μας, το οποίο γίνεται προσθέτοντας τις κατάλληλες αναφορές (references). Στην πραγματικότητα οι αναφορές αυτές είναι βοηθητικές βιβλιοθήκες (dll). Στην περίπτωση μας η «Reference» που πρέπει να προσθέσουμε στην εφαρμογή μας είναι η «`System.Data.SqlServerCe`». Ο ορισμός των Αναφορών πραγματοποιείται στην αρχή του κώδικα, μαζί με τα υπόλοιπα «namespaces» (χώρους ονομάτων) , ώστε να διευκολυνθεί η πρόσβαση σε αυτές:

```
using System.Data.SqlServerCe;
```

Κάθε φορά που απαιτείται η ανταλλαγή δεδομένων της εφαρμογής με τη τοπική ΒΔ ***TimeTableDB***, εκτελείται το ακόλουθο τμήμα κώδικα που χρησιμοποιεί το ***connection string*** προκειμένου να ανοίξει μια σύνδεση με τη βάση.

```
SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
Files\TimeTable_V2\TimeTableDB.sdf");

connection.Open();
```

Πλαίσιο Κειμένου 9: Δημιουργία σύνδεσης της φορητής εφαρμογής με την τοπική Βάση Δεδομενων

Ένα αντικείμενο **`SqlCeConnection`** αντιπροσωπεύει την αποκλειστική σύνδεση με μια πηγή δεδομένων. Απαραίτητη προϋπόθεση προκειμένου να ορίσουμε στο αντικείμενο

SqlCeConnection αποτελεί η υπόδειξη της τοποθεσίας (το μονοπάτι) στην οποία βρίσκεται η πηγή δεδομένων με την οποία επιθυμούμε να συνδεθούμε. Επίσης, για να πραγματοποιήσουμε τα ερωτήματα προς την ΒΔ πρέπει να δημιουργήσουμε μία μεταβλητή τύπου `SqlCeCommand` η οποία περιέχει το ερώτημα προς την βάση καθώς και τη μεταβλητή της σύνδεσης με την ΒΔ (`SqlCeConnection`).

```
SqlCeCommand cmd = new SqlCeCommand("query_string", connection);
```

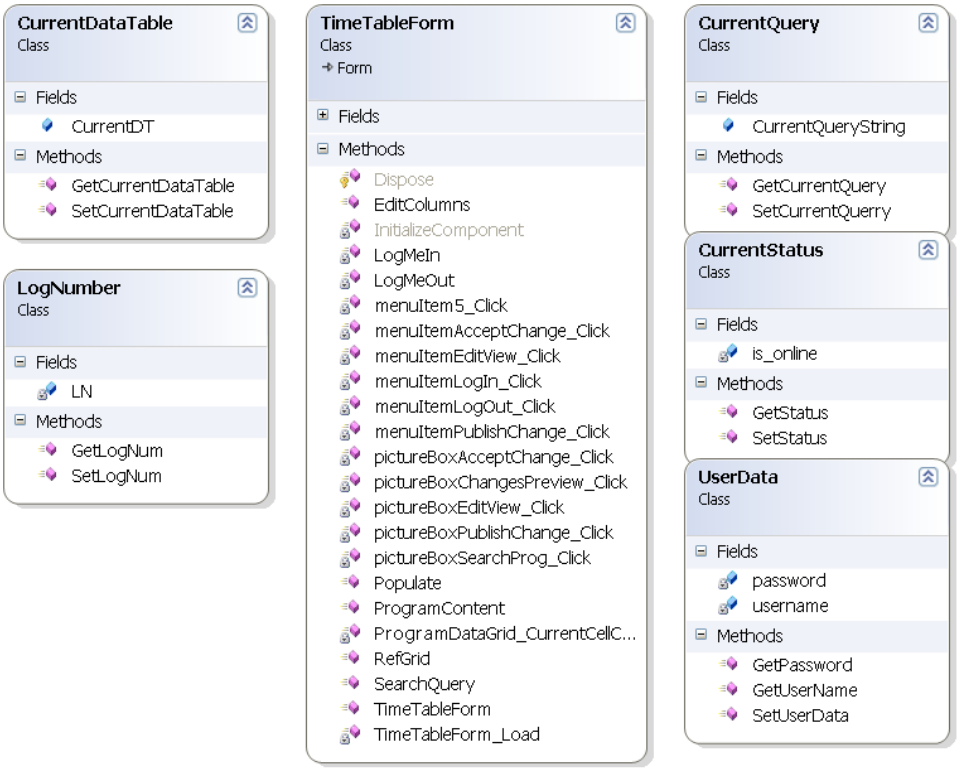
Πλαίσιο Κειμένου 10: Αρχικοποίηση αντικειμένου "SqlCeCommand"

5.2.2 Υλοποίηση της φόρμας «TimeTableForm»

5.2.2.1 Παρουσίαση κλάσεων της φόρμας «Πρόγραμμα»

Όπως συμβαίνει σε κάθε φόρμα, έτσι και στη φόρμα «TimeTableForm» υπάρχουν δεδομένα τα οποία δεν αλλάζουν καθ' όλη την διάρκεια λειτουργίας της εφαρμογής, όπως για παράδειγμα τα στοιχεία σύνδεσης του χρήστη. Υπάρχουν όμως και δεδομένα τα οποία αλλάζουν συνεχώς τιμές κατά την διάρκεια χρήσης της εφαρμογής. Σε τέτοιου τύπου δεδομένα μας ενδιαφέρει η πιο πρόσφατη τιμή τους, όπως για παράδειγμα, το πιο πρόσφατο ερώτημα που πραγματοποίησε η εφαρμογή προς την τοπική ΒΔ. Προκειμένου να γίνεται άμεση χρήση αντίστοιχων δεδομένων από διάφορα σημεία της εφαρμογής χρησιμοποιούμε κλάσεις στις οποίες αποθηκεύουμε τις τιμές των δεδομένων αυτών.

Οι κλάσεις που χρησιμοποιούμε στην φόρμα «Πρόγραμμα» αποτελούνται από την μεταβλητή που θέλουμε να αποθηκεύσουμε και από συναρτήσεις που είτε ανανεώνουν την τιμή της μεταβλητής, είτε μας επιστρέφουν την πιο πρόσφατα αποθηκευμένη τιμή τους. Στο πλαίσιο που ακολουθεί παρουσιάζονται σχηματικά οι κλάσεις και οι μέθοδοι (συναρτήσεις) της φόρμας «TimeTableForm».



Εικόνα 44: Διάγραμμα κλάσεων και μεθόδων της φόρμας "TimeTableForm"

Οι κλάσεις CurrentDataTable, CurrentQuery και CurrentStatus δημιουργήθηκαν με σκοπό να συγκρατούν πληροφορίες για την τρέχουσα κατάσταση της εφαρμογής. Πιο συγκεκριμένα, η CurrentQuery διαχειρίζεται (αποθηκεύει ή μας επιστρέφει) το τελευταίο ερώτημα της εφαρμογής προς την ΒΔ που σχετίζεται με τον πίνακα του προγράμματος εφημεριών-καθηκόντων. Με αυτό τον τρόπο μπορούμε να γνωρίζουμε κάθε στιγμή την όψη του πίνακα «Program» που εμφανίζεται στην οθόνη του χρήστη. Στη μεταβλητή τύπου «string», της ίδιας κλάσης και με όνομα `CurrentQueryString`, αποθηκεύουμε το πιο πρόσφατο ερώτημα (query string) που αποστέλλεται στην ΒΔ. Στη συγκεκριμένη περίπτωση η αρχική τιμή της μεταβλητής όνομα `CurrentQueryString`, δηλαδή το ερώτημα που αντιπροσωπεύει την αρχική (default) εμφάνιση κατά την έναρξη της εφαρμογής, είναι η εξής:

```
public string CurrentQueryString = "SELECT Program_ID AS Κωδικός, Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program WHERE User_ID=
'xpchris'";
```

Πλαίσιο Κειμένου 11: Ερώτημα προς την τοπική ΒΔ που επιστρέφει το πρόγραμμα το οποίο εμφανίζεται κατά την έναρξη της εφαρμογής

Ο μοναδικός σκοπός της συνάρτησης `GetCurrentQuery()` είναι να επιστρέφει την αποθηκευμένη τιμή της μεταβλητής `CurrentQueryString` σε οποιοδήποτε αντικείμενο τύπου «string» την καλέσει.

```
public string GetCurrentQuery()
        {
            return CurrentQueryString;
        }
```

Πλαίσιο Κειμένου 12: Η συνάρτηση που επιστρέφει την τιμή "CurrentQueryString" της κλάσης "CurrentQuery"

Ενώ η συνάρτηση `SetCurrentQuerry` έχει ως όρισμα ένα αντικείμενο τύπου «string» το οποίο χρησιμοποιεί προκειμένου να αλλάξει την ήδη αποθηκευμένη τιμή της μεταβλητής `CurrentQueryString`.

```
public void SetCurrentQuerry(string currentQuery)
        {
            CurrentQueryString = currentQuery;
        }
```

Πλαίσιο Κειμένου 13: Η συνάρτηση ανάθεσης τιμής στην μεταβλητή "CurrentQueryString" της κλάσης "CurrentQuery"

Εκμεταλλευόμενοι την δυνατότητα που μας παρέχει η `SetCurrentQuerry` την καλούμε σε σημεία του κώδικα στα οποία πραγματοποιούνται τα ερωτήματα προς την ΒΔ, με σκοπό η μεταβλητή `CurrentQueryString` να ανανεώνει την τιμή της με το πιο πρόσφατο ερώτημα.

```
public class CurrentQuery
    {
        public string CurrentQueryString = "SELECT Program_ID AS Κωδικός, Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program WHERE User_ID=
'xpchris'";

        public string GetCurrentQuery()
        {
            return CurrentQueryString;
        }

        public void SetCurrentQuerry(string currentQuery)
        {
            CurrentQueryString = currentQuery;
        }
    }
```

Πλαίσιο Κειμένου 14: Η κλάση "CurrentQuery"

Ίδια λογική και υλοποίηση με την `CurrentQuery` παρουσιάζουν και οι κλάσεις `CurrentDataTable` και `CurrentStatus`. Η διαφορά τους έγκειται στο γεγονός ότι η μεταβλητή που διαχειρίζονται δεν είναι τύπου «string» αλλά τύπου «DataTable» και «Boolean» αντίστοιχα.

Στην περίπτωση της CurrentDataTable αποθηκεύουμε τον πίνακα που αντλούμαι από την ΒΔ και αντιστοιχεί στο ερώτημα που διαχειρίζεται η κλάση «CurrentQuery». Ενώ στην περίπτωση της «CurrentStatus» η «boolean» μεταβλητή με όνομα «is_online» εναλλάσσεται μεταξύ των τιμών «false» όταν ο χρήστης «δεν είναι συνδεδεμένος με το σύστημα» (θα το εξηγήσουμε στις επόμενες παραγράφους) και «true» όταν ο χρήστης έχει πρόσβαση στις υπηρεσίες του συστήματος.

5.2.2.2 Αναλυτική περιγραφή της φόρμας «TimeTableForm»

Όταν καλείται η φόρμα TimeTableForm πραγματοποιείται η δημιουργία των αντικειμένων των κλάσεων στη φόρμα και που θα χρησιμοποιηθούν στην συνέχεια από άλλα στοιχεία της φόρμας αυτής. Επίσης, πραγματοποιείται η αρχικοποίηση των «TableAdapters» αντικειμένων, τα οποία «γεμίζουν» με δεδομένα από το Data Set με όνομα `timeTableDBDataSet1`, της βάσης TimeTableDB που βρίσκεται στην φορητή συσκευή μας.

```
        //αρχικοποίηση αντικειμένων (κλάσεων)

        CurrentQuery CurrentQr = new CurrentQuery();
        CurrentDataTable CurrentDT = new CurrentDataTable();
        CurrentStatus CurrentST = new CurrentStatus();
        LogNumber Logins = new LogNumber();

        public TimeTableForm()
               {
                InitializeComponent();

        this.declared_DutiesTableAdapter1.Fill(this.timeTableDBDataSet1.Declared_Duties);

        this.declared_LocationsTableAdapter1.Fill(this.timeTableDBDataSet1.Declared_Locations);
               this.programTableAdapter1.Fill(this.timeTableDBDataSet1.Program);

                int currentLog = Logins.GetLogNum();
                if (currentLog == 0)
                    {
                       LogMeOut();
                       Logins.SetLogNum(currentLog++);
                     }

                }
```

Πλαίσιο Κειμένου 15: Αρχικοποίηση των αντικειμένων κλάσεων κατά την φόρτωση της φόρμας "TimeTableForm"

5.2.2.2.1 Η συνάρτηση TimeTableForm_Load

Έπειτα ακολουθεί η συνάρτηση `TimeTableForm_Load`. Η συγκεκριμένη συνάρτηση εκτελείται μόνο κατά την αρχικοποίηση της φόρμας και είναι υπεύθυνη για:

- Την αρχικοποίηση και ανανέωση του προγράμματος και προσωπικών δεδομένων.
- Την προβολή του προγράμματος εφημεριών-καθηκόντων.

- Τον προσδιορισμό της κατάστασης σύνδεσης του χρήστη.

5.2.2.2.2 Αρχικοποίηση και ανανέωση του προγράμματος και προσωπικών δεδομένων

Για την επικοινωνία και τη μεταφορά των δεδομένων μεταξύ της εφαρμογής και του Web Service, αρχικοποιείται ένα στιγμιότυπο της κλάσης `SystemData` με όνομα «`TableContentWebService`». Η κλάση αυτή ανήκει στο Web Reference «`SystemDataWebReference`» που έχουμε προσθέσει στην εφαρμογή.

```
SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();
```

Πλαίσιο Κειμένου 16: Αρχικοποίηση στιγμιότυπου της κλάσης "SystemData"

Σκοπός μας είναι να μεταφέρουμε τα δεδομένα που αφορούν τον χρήστη από τον κεντρικό υπολογιστή, μέσω του Web Service και να τα αποθηκεύσουμε στην τοπική βάση δεδομένων SQL Server Compact Edition της συσκευής μας που έχει όνομα TimeTableDB. Για τον σκοπό αυτό δημιουργούμε τρία αντικείμενα τύπου Data Table όσοι και οι πίνακες που θέλουμε να γεμίσουμε με δεδομένα.

```
DataTable ProgramDataTable = new DataTable();
DataTable LocationsDataTable = new DataTable();
DataTable DutiesDataTable = new DataTable();

ProgramDataTable = TableContentWebService.ProgramContent();
LocationsDataTable = TableContentWebService.Declared_Locations();
DutiesDataTable = TableContentWebService.Declared_Duties();
```

Πλαίσιο Κειμένου 17: Δημιουργία και αρχικοποίηση αντικειμένων "DataTable"

Στη συνέχεια αρχικοποιούμε τα τρία αντικείμενα Data Table `ProgramDataTable` , `LocationsDataTable` και `DutiesDataTable` και καλώντας τις μεθόδους του «`TableContentWebService`» τροφοδοτούμαι τα αντικείμενα DataTables με δεδομένα. Οι μέθοδοι που χρησιμοποιούμαι είναι οι `ProgramContent`, `Declared_Locations` και `Declared_Duties`.

Τα αντικείμενα Data Table `ProgramDataTable` , `LocationsDataTable` και `DutiesDataTable` περιέχουν προσωρινά τα δεδομένα του χρήστη που προορίζονται για τους πίνακες Program, Declared_Duties και Declared_Locations της τοπικής βάσης δεδομένων. Η μεταφορά των δεδομένων από τα αντικείμενα DataTables στους αντίστοιχους πίνακες της ΒΔ πραγματοποιήται με χρήση της συνάρτησης Populate.

```
Populate(ProgramDataTable, "Program");
Populate(LocationsDataTable, "Declared_Locations");
Populate(DutiesDataTable, "Declared_Duties");
```

Πλαίσιο Κειμένου 18: Τρόπος χρήσης της συνάρτησης "Populate"

Η συνάρτηση Populate έχει ως ορίσματα ένα αντικείμενο τύπου DataTable με τα δεδομένα του πίνακα στον οποίο πρόκειται να μεταφερθούν καθώς και το όνομα του πίνακα αυτού. Η υλοποίηση της συνάρτησης Populate ακολουθεί σε επόμενη παράγραφο.

5.2.2.2.2.1 Προβολή του προγράμματος εφημεριών-καθηκόντων

Για την προβολή και εμφάνιση του προγράμματος εφημεριών-καθηκόντων καλούμε την συνάρτηση ProgramContent, η οποία έχει ως ορίσματα ένα αντικείμενο τύπου string με το ερώτημα προς την τοπική ΒΔ και ένα αντικείμενο τύπου Data Set για την προσωρινή αποθήκευση των δεδομένων που θα ληφθούν από την τοπική ΒΔ.

```
string cmd = CurrentQr.GetCurrentQuery();
```

Πλαίσιο Κειμένου 19: Εισαγωγή στην μεταβλητή "cmd" του τρέχοντος "querystring"

Η συνάρτηση ProgramContent επιστρέφει ένα αντικείμενο τύπου DataTable γεμάτο με δεδομένα από τον πίνακα Program. Τα δεδομένα αυτά αντιστοιχούν στο ερώτημα που τέθηκε προς την ΒΔ και θέλουμε να προβάλουμε στην οθόνη του χρήστη.

Η προβολή των δεδομένων του πίνακα, πραγματοποιείται με την χρήση του πλέγματος δεδομένων (DataGrid). Για τον λόγο αυτό, ορίζουμε ως πηγή δεδομένων για το αντικείμενο `ProgramDataGrid` τον πίνακα που περιέχει τα δεδομένα προς εμφάνιση.

```
ProgramDataGrid.DataSource =ProgramContent(cmd, dsProgramContent);
```

Πλαίσιο Κειμένου 20: Καθορισμός της πηγής δεδομένων του αντικειμένου "Data Grid"

5.2.2.2.2.2 Προσδιορισμός της κατάστασης σύνδεσης του χρήστη

Ο έλεγχος για την κατάσταση της επικοινωνίας μεταξύ εφαρμογής και κεντρικού υπολογιστή εκτελείται μέσα στην συνάρτηση `TimeTableForm_Load` με την χρήση της διαχείρισης εξαιρέσεων `try` - `catch`. Ο έλεγχος πραγματοποιείται ως εξής:

Οι εντολές που βρίσκονται μέσα στο «block» της εντολής `try` εκτελούνται μόνο αν ισχύουν οι δύο υποχρεωτικές απαιτήσεις σύνδεσης που είδαμε σε προηγούμενη παράγραφο. Ενεργοποιείται έτσι μια αλυσίδα εντολών καθώς και η κλήση των διαδοχικών συναρτήσεων. Μεταξύ αυτών είναι και οι εντολές για την ενεργοποίηση του εικονιδίου και την αποθήκευση της κατάστασης του χρήστη.

```
pictureBox5.Visible = true;
pictureBox5.BringToFront();
pictureBox1.SendToBack();
pictureBox1.Visible = false;
bool online=true;
CurrentST.SetStatus(online);
```

Πλαίσιο Κειμένου 21: Εντολές για την ενεργοποίηση του εικονιδίου σύνδεσης και την αποθήκευση της κατάστασης σύνδεσης του χρήστη

Στην περίπτωση που δεν εκτελεστεί λόγω κάποιου σφάλματος μία ή περισσότερες από τις εντολές που βρίσκονται μέσα στο «block» της `try` σημαίνει ότι το σύστημα δεν είναι σε θέση να πραγματοποιήσει τη σύνδεση με το Web Service, δηλαδή δεν ισχύουν οι υποχρεωτικές απαιτήσεις πρόσβασης. Τότε ενεργοποιούνται αυτόματα οι εντολές που βρίσκονται μέσα στο «block» της εντολής `catch`, θέτοντας την εφαρμογή σε κατάσταση «εκτός σύνδεσης» και ενημερώνοντας το χρήστη για την ελλιπή πρόσβαση στις υπηρεσίες της εφαρμογής.

```
MessageBox.Show("Δεν έχετε πρόσβαση στο Ίντερνετ "+
"το Profil σας τήθεται αυτόματα σε εργασία χωρίς Σύνδεση. "+
"Οι υπηρεσίες Αλλαγής και Αποδοχής καθηκόντων είναι ΕΚΤΟΣ Λειτουργίας" );
pictureBox5.Visible = false;
pictureBox5.SendToBack();
pictureBox1.BringToFront();
pictureBox1.Visible = true;
```

Πλαίσιο Κειμένου 22: Εντολές που ενεργοποιούνται όταν ο χρήστης δεν έχει πρόσβαση στο διαδίκτυο

Ο κώδικας της συνάρτησης `TimeTableForm_Load` που πραγματοποιεί τον παραπάνω έλεγχο καθώς και τις υπόλοιπες λειτουργίες, δίνεται στη συνέχεια.

```
private void TimeTableForm_Load(object sender, EventArgs e)
        {
            try
            {
                SystemDataWebReference.SystemData TableContentWebService = new
               TimeTable_V2.SystemDataWebReference.SystemData();

                DataTable ProgramDataTable = new DataTable();
                DataTable LocationsDataTable = new DataTable();
                DataTable DutiesDataTable = new DataTable();
                ProgramDataTable = TableContentWebService.ProgramContent();
                LocationsDataTable = TableContentWebService.Declared_Locations();
                DutiesDataTable = TableContentWebService.Declared_Duties();

                DataView dv = new DataView();
                dv.Table = ProgramDataTable;

                Populate(ProgramDataTable, "Program");
                Populate(LocationsDataTable, "Declared_Locations");
                Populate(DutiesDataTable, "Declared_Duties");

                pictureBox5.Visible = true;
                pictureBox5.BringToFront();
                pictureBox1.SendToBack();
                pictureBox1.Visible = false;
                bool online=true;
                CurrentST.SetStatus(online);
            }

            catch (Exception ex)
            {
                //throw (ex);
                MessageBox.Show("Δεν έχετε πρόσβαση στο Ίντερνετ "+
                "το Profil σας τήθεται αυτόματα σε εργασία χωρίς Σύνδεση. "+
                "Οι υπηρεσίες Αλλαγής και Αποδοχής καθηκόντων είναι ΕΚΤΟΣ
               Λειτουργίας" );
                pictureBox5.Visible = false;
                pictureBox5.SendToBack();
                pictureBox1.BringToFront();
                pictureBox1.Visible = true;

            }

            string cmd = CurrentQr.GetCurrentQuery();
            DataSet dsProgramContent = new DataSet();
            ProgramDataGrid.DataSource =ProgramContent(cmd, dsProgramContent);
            ProgramDataGrid.Refresh();


        }
```

Πλαίσιο Κειμένου 23: Κώδικας της συνάρτησης TimeTableForm_Load

5.2.2.2.3 Η συνάρτηση Populate

Η συνάρτηση ***populate*** είναι μια ευφυής συνάρτηση η οποία καλείται μόνο από την συνάρτηση `TimeTableForm_Load` κατά την αρχικοποίηση ή «φόρτωμα» της κεντρικής φόρμας TimeTableForm. Έργο της είναι να αντιγράψει στην τοπική ΒΔ τα δεδομένα όπως ακριβώς έχουν ληφθεί από το Web Service. Το αποτέλεσμα που προκύπτει είναι μια όψη από τη βάση δεδομένων SystemData (του κεντρικού υπολογιστή) με τα δεδομένα του χρήστη, στην τοπική βάση δεδομένων που βρίσκεται στην φορητή συσκευή.

Απαραίτητη προϋπόθεση για την σωστή λειτουργία της συνάρτησης ***populate*** αποτελεί το σχήμα της βάσης δεδομένων. Το σχήμα που έχουν οι πίνακες της τοπικής βάσης

δεδομένων ***TimeTableDB***, θα πρέπει να είναι ακριβώς το ίδιο με αυτό της βάση δεδομένων στον κεντρικό υπολογιστή. Η συνάρτηση ***populate*** αναγνωρίζει τα πεδία του πίνακα δεδομένων που έχουν ληφθεί από το Web Service (αντίστοιχα το Web Service έχει κάνει λήψη των δεδομένων από την βάση δεδομένων του κεντρικού υπολογιστή) και αποθηκεύει τα δεδομένα στην τοπική ΒΔ πραγματοποιώντας μια αντιστοίχιση αυτών των πεδίων και των ονομάτων τους. Τα πεδία αυτά είναι τα ονόματα των στηλών, ο τύπος δεδομένων κ.α. Η συνάρτηση ***populate*** αποτελείται από δύο μέρη.

- Στο πρώτο μέρος πραγματοποιείται η αναγνώριση του αριθμού και των ονομάτων των στηλών που βρίσκονται στον πίνακα του αντικειμένου Data Table.
- Στο δεύτερο μέρος πραγματοποιείται η εισαγωγή των τιμών κάθε κελιού στον αντίστοιχο πίνακα της ΒΔ.

Αναλυτικότερα, η συνάρτηση ***populate*** έχει ως ορίσματα ένα αντικείμενο τύπου ***Data Table*** και ένα αντικείμενο ***string***. Το αντικείμενο ***DataTable*** περιέχει τον πίνακα με τα δεδομένα που πρέπει να αντιγράψει η συνάρτηση στη ΒΔ. Το αντικείμενο τύπου ***string*** με όνομα «`tablename`» αναφέρετε στο όνομα του πίνακα της τοπικής ΒΔ στον οποίο θα πραγματοποιηθεί η αντιγραφή των δεδομένων.

```
public void Populate(DataTable TableContent, string tablename)
```

Πλαίσιο Κειμένου 24: Τα ορίσματα της συνάρτησης "Populate"

5.2.2.2.3.1 Πρώτο μέρος της συνάρτησης Populate

Όπως αναφέραμε στην προηγούμενη παράγραφο, η συνάρτηση ***populate*** αποτελείται από δύο τμήματα/μέρη. Στο πρώτο τμήμα, πρώτος στόχος της συνάρτησης είναι η αυτόματη δημιουργία ενός ερωτήματος εισαγωγής (***insert query***) SQL το οποίο θα χρησιμοποιηθεί από το δεύτερο τμήμα της συνάρτησης. Στο σημείο αυτό θα πρέπει να υπενθυμίσουμε ότι ένα ερώτημα SQL που χρησιμοποιείται για την εισαγωγή δεδομένων σε έναν πίνακα ΒΔ, έχει την εξής μορφή:

```
insert into onoma_pinaka (onoma_sthlhs1,onoma_sthlhs2...onoma_sthlhsx)
values(timh1,timh2...timhx);
```

Πλαίσιο Κειμένου 25: Δομή SQL ερωτήματος

Προκειμένου να υλοποιήσουμε το παραπάνω εγχείρημα, χρησιμοποιούμε την κλάση `StringBuilder`. Αυτή η κλάση αντιπροσωπεύει ένα αντικείμενο τύπου ***string*** του οποίου η τιμή (περιεχόμενο της μεταβλητής), μπορεί να αλλάζει. Δηλαδή, είναι μια μεταβλητή ακολουθία χαρακτήρων. Λέμε ότι η τιμή του είναι μεταβλητή επειδή μπορεί να τροποποιηθεί μόλις δημιουργηθεί με την επισύναψη, την αφαίρεση, την αντικατάσταση ή την παρεμβολή άλλων χαρακτήρων.

```
public void Populate(DataTable TableContent, string tablename)
        {
            StringBuilder sql = new StringBuilder();
            sql.Append("insert into " + tablename + "(");

            StringBuilder fields = new StringBuilder();
            StringBuilder parameters = new StringBuilder();

            foreach (DataColumn col in TableContent.Columns)
            {
                fields.Append(col.ColumnName);
                parameters.Append("@" + col.ColumnName.ToLower());

              if (col.ColumnName !=
              TableContent.Columns[TableContent.Columns.Count -  1].ColumnName)

                {
                    fields.Append(",");
                    parameters.Append(",");
               }
            }

            sql.Append(fields.ToString() + ")");
            sql.Append("values(");
            sql.Append(parameters.ToString() + ")");
```

Πλαίσιο Κειμένου 26: Πρώτο μέρος της συνάρτησης "Populate"

Στο παραπάνω πλαίσιο με τον κώδικα βλέπουμε τον τρόπο με τον οποίο σχηματίζεται το υποψήφιο προς αποστολή «query string». Στην υλοποίηση παρατηρούμε τρία (3) αντικείμενα τύπου ***StringBuilder*** και μία επαναληπτική διαδικασία «foreach».

```
parameters.Append("@" + col.ColumnName.ToLower();
```

Η μεταβλητή με όνομα «sql» χρησιμοποιείται για την αποθήκευση του «query string», ως σύνολο. Οι μεταβλητές «fields» και «parameters» χρησιμοποιούνται αντίστοιχα για την αποθήκευση του ονόματος κάθε στήλης που βρίσκεται στο ***Data Table*** και για την αποθήκευση της τιμής που βρίσκεται σε κάθε κελί της στήλης. Και οι δύο αυτές μεταβλητές, στο τέλος επισυνάπτονται στην τιμή της μεταβλητής «sql» προκειμένου να δημιουργήσουν ένα «query string» σχεδόν έτοιμο για αποστολή σε μία βάση δεδομένων. Η εισαγωγή των δεδομένων στις μεταβλητές «fields» και «parameters» πραγματοποιείται με την βοήθεια της επαναληπτικής διαδικασίας «foreach», η οποία για κάθε στήλη που βρίσκεται στον πίνακα, εισάγει στη μεταβλητή «fields» το όνομα της αντίστοιχης στήλης, ενώ στην μεταβλητή «parameters» εισάγεται μια τιμή-παράμετρος η οποία στη συνέχεια (στο δεύτερο τμήμα της συνάρτησης) θα αντικατασταθεί από την πραγματική τιμή του πίνακα. Επίσης, προστίθενται τα κόμματα ανάμεσα στα ονόματα των στηλών, όπου χρειάζονται. Η εισαγωγή των κομμάτων πραγματοποιείται εξετάζοντας αν κάθε στήλη του πίνακα είναι η τελευταία ή όχι σε σειρά. Εάν είναι, τότε δεν προστίθεται άλλο κόμμα. Το αποτέλεσμα που προκύπτει από το πρώτο μέρος της συνάρτησης «populate» είναι ένα query string της μορφής.

```
insert   into   onoma_pinaka   (onoma_sthlhs1,onoma_sthlhs2...onoma_sthlhsx)
values(@parametros1,@parametros2...@parametrosx);
```

Πλαίσιο Κειμένου 27: Το Query String που προκύπτει από το πρώτο μέρος της συνάρτησης "Populate"

5.2.2.2.3.2 Δεύτερο μέρος της συνάρτησης Populate

Αρχικά, στο δεύτερο μέρος της συνάρτησης ***populate*** ορίζουμε το ***connection string*** και στην συνέχεια την μεταβλητή που ανοίγει τη σύνδεση με τη βάση. Έπειτα, με δύο εμφωλευμένες επαναληπτικές διαδικασίες «**foreach**», μεταφερόμαστε σε κάθε κελί του πίνακα της μεταβλητής ***DataTable***, διαβάζουμε την τιμή του κελιού και την εισάγουμε στη θέση της παραμέτρου του ***query string*** που δημιουργήθηκε στο πρώτο τμήμα της συνάρτησης.

```
SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
Files\TimeTable_V2\TimeTableDB.sdf");

foreach (DataRow row in TableContent.Rows)
      {
        if (connection.State == ConnectionState.Closed)
           {
             connection.Open();
           }

         SqlCeCommand cmd = new SqlCeCommand(sql.ToString(), connection);

         foreach (DataColumn col in TableContent.Columns)
           {
             cmd.Parameters.AddWithValue("@" + col.ColumnName.ToLower(),
             row[col.ColumnName]);
           }

         try
           {
              cmd.ExecuteNonQuery();
           }
         catch (Exception ex)
           {
                         }

      }
```

Πλαίσιο Κειμένου 28: Δεύτερο μέρος της συνάρτησης "Populate"

Το αποτέλεσμα είναι ένα πλήρες διαμορφωμένο «query string» το οποίο αντιγράφει γραμμή προς γραμμή τα δεδομένα του πίνακα τύπου Data Table στον αντίστοιχο πίνακα που βρίσκεται στην τοπική ΒΔ της συσκευής μας. Παράλληλα, πριν από κάθε εισαγωγή δεδομένων εκτελείται ένας έλεγχος προκείμενου να διαπιστώσουμε εάν η σύνδεση με την ΒΔ είναι ανοικτή. Στην περίπτωση που για κάποιον λόγο έχει διακοπεί η σύνδεση, την ανοίγουμε ξανά. Επίσης, μετά από την αποστολή κάθε ερωτήματος προς την ΒΔ πραγματοποιείται ένας έλεγχος σφαλμάτων για τις εντολές που απευθύνονται στην ΒΔ. Σε περίπτωση σφάλματος κατά την εκτέλεση των εντολών, επιστρέφεται ένα μήνυμα εξηγώντας τι παρεμπόδισε την σωστή διεκπεραίωση του ερωτήματος. Και τα δύο μέρη της συνάρτησης «populate» μαζί με τον υπόλοιπο κωδικό της φόρμας παρουσιάζονται στο παράρτημα με τον κώδικα της φορητής εφαρμογής.

5.2.2.2.4 Η συνάρτηση ProgramContent

Κατά την επεξήγηση της συνάρτησης `TimeTableForm_Load` συναντήσαμε τη συνάρτηση `ProgramContent` η οποία τροφοδοτούσε με δεδομένα το `ProgramDataGrid`. Έχοντας αυτό στο μυαλό μας, μπορούμε να πούμε πως έργο της `ProgramContent` είναι να στέλνει ένα ερώτημα προς την τοπική βάση δεδομένων και να της επιστρέφεται το αποτέλεσμα σαν πίνακας. Στη συνέχεια, ο πίνακας αυτός μέσω του αντικειμένου `ProgramDataGrid` εμφανίζεται στην οθόνη του χρήστη. Χαρακτηριστικό στοιχείο της `ProgramContent` είναι ότι χρησιμοποιείται από πολλές ακόμα συναρτήσεις και φόρμες αφού αποτελεί και επιτελεί μια από τις σημαντικότερες ρουτίνες της εφαρμογής.

Θα αρχίσουμε την παρουσίαση της υλοποίησης αναφερόμενοι στα ορίσματα της συνάρτησης, τα οποία είναι: μια μεταβλητή τύπου «string» με το ερώτημα προς την βάση και μια κενή μεταβλητή τύπου «DataSet» η οποία θα χρησιμοποιηθεί για την δημιουργία του πίνακα τύπου DataTable. Ο πίνακας αυτός, αφού επιστραφεί γεμάτος με δεδομένα, θα αποτελεί το αντικείμενο εμφάνισης από το `ProgramDataGrid`.

```
public DataTable ProgramContent(string cmd, DataSet dsProgramContent)
```

Στη συνέχεια ορίζουμε το «string» με το ερώτημα προς την βάση, ως το τρέχον ερώτημα.

```
CurrentQr.SetCurrentQuerry(cmd);
```

Έπειτα, δημιουργούμε και ανοίγουμε τη σύνδεση με την ΒΔ χρησιμοποιώντας ένα αντικείμενο **`SqlCeConnection`**, όπως ακριβώς είδαμε και στην προηγούμενη παράγραφο.

```
SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
Files\TimeTable_V2\TimeTableDB.sdf");
            connection.Open();
```

Έχοντας πλέον αποκτήσει πρόσβαση στην ΒΔ, εγκαθιδρύουμε την γέφυρα επικοινωνίας με ένα αντικείμενο **`SqlCeDataAdapter`** μεταξύ ενός κενού DataSet με όνομα `dsProgramContent` και της ΒΔ. Χρησιμοποιώντας την συνάρτηση «Fill» αντλούμαι τα δεδομένα από την ΒΔ και γεμίζουμε το DataSet.

```
SqlCeDataAdapter daProgramContent = new SqlCeDataAdapter(cmd, connection);
daProgramContent.Fill(dsProgramContent,"Program");
```

Στη συνέχεια με την εντολή «`DataTable myDataTable = dsProgramContent.Tables[0];`» δημιουργούμαι ένα πίνακα δια μέσου του DataSet που περιλαμβάνει όλα τα δεδομένα που ανακτήσαμε από τη ΒΔ. Τέλος, αποθηκεύουμε τον πίνακα `myDataTable` ως τον τρέχοντα πίνακα εμφάνισης και τον επιστρέφουμε προκειμένου να

χρησιμοποιηθεί ως πηγή δεδομένων από άλλες συναρτήσεις ή αντικείμενα, όπως το ProgramDataGrid.

```
public DataTable ProgramContent(string cmd, DataSet dsProgramContent)
        {
 CurrentQr.SetCurrentQuerry(cmd);

SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
Files\TimeTable_V2\TimeTableDB.sdf");
connection.Open();

SqlCeDataAdapter daProgramContent = new SqlCeDataAdapter(cmd, connection);
daProgramContent.Fill(dsProgramContent,"Program");

 DataTable myDataTable = dsProgramContent.Tables[0];
 connection.Close();

 CurrentDT.SetCurrentDataTable(myDataTable);

 return myDataTable;
        }
```

Πλαίσιο Κειμένου 29: Ο κώδικας της συνάρτησης "ProgramContent"

5.2.2.2.5 Η συνάρτηση menuItemPublishChange_Click

Η συνάρτηση menuItemPublishChange_Click, σε συνεργασία με τις συναρτήσεις τις φόρμας «Edit_Event» εκτελεί τη λειτουργία της ανταλλαγής εφημεριών μεταξύ των χρηστών. Η συνάρτηση που παρουσιάζουμε σε αυτή την παράγραφο ενεργοποιείται με το πάτημα του κουμπιού «Κοινοποίηση Αλλαγής». Η συγκεκριμένη επιλογή βρίσκεται στο menu λειτουργιών και αντιστοιχεί στην εκτέλεση της λειτουργίας αυτής, αφού πρώτα ο χρήστης έχει επιλέξει την επιθυμητή προς δημοσίευση εφημερία.

Πρώτος στόχος της συνάρτησης «menuItemPublishChange_Click» είναι ο εντοπισμός της εφημερίας που έχει επιλέξει ο χρήστης από το πλέγμα δεδομένων που εμφανίζεται στην οθόνη. Επίσης είναι υπεύθυνη για την τροφοδότηση της φόρμας «Edit_Event» με τα στοιχεία της υποψήφιας προς δημοσίευση εφημερίας μέσω της συνάρτησης «SetChangeOptions». Η συνάρτηση «SetChangeOptions» ανήκει στην φόρμα «Edit_Event» και θα την αναλύσουμε σε επόμενη ενότητα.

Προκειμένου να εντοπίσουμε την εφημερία που έχει επιλέξει ο χρήστης πάνω στο πλέγμα των δεδομένων, θα πρέπει να γνωρίζουμε τις συντεταγμένες του κελιού που αντιστοιχεί στην συγκεκριμένη εφημερία.

```
int colNum = ProgramDataGrid.CurrentCell.ColumnNumber;
int rowNum = ProgramDataGrid.CurrentCell.RowNumber;
```

Πλαίσιο Κειμένου 30: Ορισμός συντεταγμένων

Για τον λόγο αυτό χρησιμοποιούμε δυο μεταβλητές τύπου «int» προκειμένου να αποθηκεύσουμε σε αυτές τον αριθμό της επιλεγμένης στήλης και σειράς που θα μας επιστρέψουν οι συναρτήσεις «CurrentCell.ColumnNumber» και «CurrentCell.RowNumber» του

αντικειμένου «ProgramDataGrid». Στη συνέχεια αποθηκεύουμε το χαρακτηριστικό στοιχείο «ProgramID» της εφημερίας που αντιστοιχεί στην επιλεγμένη σειρά. Σκοπός είναι η δημιουργία ενός ερωτήματος προς την ΒΔ προκειμένου να μας επιστρέψει τα πεδία-στήλες που αντιστοιχούν στην εφημερία προς δημοσίευση.

```
object ProgramID = ProgramDataGrid[rowNum, 0];
string s = string.Format("Program_ID='{0}'", ProgramID);
string cmd = "SELECT * FROM Program WHERE " + s;
```

Τα στοιχεία αυτά αντλούνται από την ΒΔ με τη χρήση ενός αντικειμένου «SqlCeDataReader» το οποίο παρέχει έναν σειριακό τρόπο ανάγνωσης των πεδίων που βρίσκονται στην ίδια σειρά της πηγής δεδομένων.

```
SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");

connection.Open();
SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
SqlCeDataReader dr = SqlCmd.ExecuteReader();
dr.Read();

SqlCmd.ExecuteNonQuery();
```

Εάν δεν προκύψει κάποιο πρόβλημα κατά την εκτέλεση των εντολών, το περιεχόμενο του αντικειμένου «SqlCeDataReader» με όνομα «dr» θα μοιάζει με τον παρακάτω πίνακα:

Program_ID	Date	Duty_Type	Duty_Start_Time	Duty_End_Time	Location	User_ID	Program_Name
Pr02	2010-04-06	Efhmereia	14.00	16.00	Aglaia Kyriakou	xpchris	Programma 1

Όπου η τιμή του πεδίου «Program_ID» αντιστοιχεί στην τιμή «Program_ID» της εφημερίας που έχει επιλέξει ο χρήστης προς δημοσίευση. Έπειτα, δημιουργούμε ένα αντικείμενο της φόρμας «Edit_Event» με όνομα «formlog» και καλούμε την συνάρτηση «SetChangeOptions» που χρησιμοποιεί ως ορίσματα της μεταβλητής τύπου «SqlCeDataReader» με τα δεδομένα της εφημερίας προς δημοσίευση. Τέλος, κλείνουμε την σύνδεση με την ΒΔ και εμφανίζουμε στην οθόνη την φόρμα «Edit_Event» η οποία στη συνέχεια θα προωθήσει την επιλογή μας προς δημοσίευση.

```
var formlog = new Edit_Event();
formlog.SetChangeOptions(dr);

connection.Close();
formlog.ShowDialog();
```

Σε περίπτωση σφάλματος κατά την εκτέλεση των παραπάνω εντολών, πραγματοποιείται μια επανεκκίνηση των δεδομένων εμφάνισης, με τον τρόπο που έχουμε παρουσιάσει και στις παραπάνω συναρτήσεις.

```
string currentquery = CurrentQr.GetCurrentQuery();
DataSet dsProgramContent = new DataSet();
ProgramDataGrid.DataSource = ProgramContent(currentquery, dsProgramContent);
```

Για τους λόγους που έχουμε αναφέρει στην παρουσίαση της συνάρτησης «TimeTableForm_Load», η λειτουργία της δημοσίευσης εφημεριών δεν είναι διαθέσιμη όταν ο χρήστης «δεν είναι συνδεδεμένος με το σύστημα». Ο συγκεκριμένος έλεγχος πραγματοποιείται με την βοήθεια της κλάσης «CurrentStatus» που αναλύσαμε στην αρχή της παρουσίασης.

```
if (CurrentST.GetStatus())
{…}
else
{
  MessageBox.Show("Η υπηρεσία δεν είναι Διαθέσιμή. Δεν έχετε πρόσβαση στο Δίκτυο."+
  "Μεταβείτε σε σημείο με δυνατότητα σύνδεσης και πραγματοποιήστε επαναδύνδεση με
το     σύστημα");
}
```

Στην περίπτωση οπού ικανοποιείται η συνθήκη ελέγχου, εκτελούνται όλες οι παραπάνω λειτουργίες της συνάρτησης «menuItemPublishChange_Click». Ενώ, στην αντίθετη περίπτωση, εμφανίζεται ένα μήνυμα που ενημερώνει τον χρήστη για την κατάσταση της σύνδεσής του. Στο παρακάτω πλαίσιο εμφανίζεται ο πλήρης κωδικός της συνάρτησης «menuItemPublishChange_Click»

```
private void menuItemPublishChange_Click(object sender, EventArgs e)
        {
            if (CurrentST.GetStatus())
            {
                int colNum = ProgramDataGrid.CurrentCell.ColumnNumber;
                int rowNum = ProgramDataGrid.CurrentCell.RowNumber;

                try
                {
                    object ProgramID = ProgramDataGrid[rowNum, 0];
                    string s = string.Format("Program_ID='{0}'", ProgramID);
                    string cmd = "SELECT * FROM Program WHERE " + s;

                      SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
                      Files\TimeTable_V2\TimeTableDB.sdf");
                    connection.Open();
                    SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
                    SqlCeDataReader dr = SqlCmd.ExecuteReader();
                    dr.Read();
                    SqlCmd.ExecuteNonQuery();

                    var formlog = new Edit_Event();
                    formlog.SetChangeOptions(dr);

                    connection.Close();
                    formlog.ShowDialog();
                }

                catch (Exception ex)
                {
                    string currentquery = CurrentQr.GetCurrentQuery();
                    DataSet dsProgramContent = new DataSet();
                      ProgramDataGrid.DataSource = ProgramContent(currentquery,
                      dsProgramContent);
                }

            }

            else
            {
               MessageBox.Show("Η υπηρεσία δεν είναι Διαθέσιμή. Δεν έχετε πρόσβαση στο Δίκτυο");
            }
    }
```

Πλαίσιο Κειμένου 31: Η συνάρτηση menuItemPublishChange_Click

5.2.2.2.6 Οι συναρτήσεις menuItemLogIn_Click και LogMeIn

Προκειμένου ο χρήστης να έχει πρόσβαση στις υπηρεσίες της εφαρμογής, θα πρέπει να γίνει επαλήθευση της ταυτότητάς του. Για τον λόγο αυτό έχουμε δημιουργήσει την φόρμα «LogInForm» η οποία είναι υπεύθυνη για τον έλεγχο της ταυτότητας του χρήστη. Η συνάρτηση «`menuItemLogIn_Click`» έχει σκοπό την ενεργοποίηση της φόρμας «LogInForm» καθώς και την διαχείριση του αποτελέσματος που προκύπτει από τον έλεγχο της ταυτοποίησης.

Η «`menuItemLogIn_Click`» ενεργοποιείται όταν ο χρήστης πατήσει το κουμπί της σύνδεσης κατά την είσοδό του στην εφαρμογή. Με το πάτημα του κουμπιού δημιουργείται ένα αντικείμενο της φόρμας «LogInForm». Καλώντας την συνάρτηση «`ShowDialog`» του αντικειμένου που δημιουργήθηκε, ενεργοποιούνται οι εντολές αρχικοποίησης της φόρμας η οποία στη συνέχεια εμφανίζεται στην οθόνη του χρήστη.

```
private void menuItemLogIn_Click(object sender, EventArgs e)
        {
     var loginform = new LogInForm();
            loginform.ShowDialog();

            if (loginform.DialogResult == DialogResult.OK)
            {LogMeIn();}
         }
```

Όπως αναφέραμε και προηγουμένως, από τον έλεγχο ταυτοποίησης χρήστη που πραγματοποιείται στην φόρμα «LogInForm» παράγεται ένα αποτέλεσμα. Εάν το αποτέλεσμα του ελέγχου είναι αληθές, δηλαδή ο χρήστης έχει εισάγει τα σωστά στοιχεία ταυτοποίησης, τότε το αποτέλεσμα είναι «`DialogResult.OK`» και καλείται η συνάρτηση «`LogMeIn`» της φόρμας «TimeTableForm» η οποία ενεργοποιεί όλες τις λειτουργίες τις εφαρμογής και τις θέτει έτοιμες για χρήση.

Η συνάρτηση «`LogMeIn`» [Πλαίσιο Κειμένου 32] καλείται όταν ελεγχθεί και επιβεβαιωθεί η ταυτότητα του χρήστη. Σκοπός της είναι να κάνει τις λειτουργίες τις εφαρμογής διαθέσιμες προς χρήση. Ο τρόπος που το πραγματοποιεί είναι είτε με την ενεργοποίηση είτε με την απενεργοποίηση στοιχείων της φόρμας «TimeTableForm». Συγκεκριμένα ενεργοποιεί:

- το πλέγμα της εμφάνισης των δεδομένων «`ProgramDataGrid`»
- την μπάρα εργαλείων με τις λειτουργίες της εφαρμογής και
- το πλήκτρο της αποσύνδεσης-εξόδου από την εφαρμογή

ενώ απενεργοποιεί το πλήκτρο εισόδου στην εφαρμογή αφού αυτή έχει ήδη πραγματοποιηθεί.

```
private void LogMeIn()
        {
            menuItemLogOut.Enabled = true;
            menuItemLogIn.Enabled = false;
            ProgramDataGrid.Enabled = true;
            StatusBarPanel.Enabled = true;
        }
```

Πλαίσιο Κειμένου 32

5.2.2.2.7 Οι συναρτήσεις menuItemLogOut_Click και LogMeOut

Μετά από μια επιτυχημένη είσοδο του χρήστη στην εφαρμογή, ενεργοποιείται το πλήκτρο της αποσύνδεσης επιτρέποντας έτσι στο χρήστη να εξέλθει από την εφαρμογή και να γίνει παύση της δυνατότητας χρήσης των λειτουργιών της. Με το πάτημα του πλήκτρου της αποσύνδεσης εκτελείται η «`menuItemLogOut_Click`» [Πλαίσιο Κειμένου 33], η οποία με την σειρά της καλεί την συνάρτηση «`LogMeOut`».

```
private void menuItemLogOut_Click(object sender, EventArgs e)
        {
            LogMeOut();
        }
```

Πλαίσιο Κειμένου 33

Όπως η συνάρτηση «LogMeIn» έτσι και η «LogMeOut» [Πλαίσιο Κειμένου 34] εκτελεί μια σειρά ενεργοποιήσεων και απενεργοποιήσεων στοιχείων της φόρμας. Πιο συγκεκριμένα εκτελεί τις αντίστροφες εντολές από την «LogMeIn» , δηλαδή απενεργοποιεί:

- το πλέγμα της εμφάνισης των δεδομένων «ProgramDataGrid»
- την μπάρα εργαλείων με τις λειτουργίες της εφαρμογής και
- το πλήκτρο της αποσύνδεσης-εξόδου από την εφαρμογή

ενώ ενεργοποιεί το πλήκτρο εισόδου στην εφαρμογή, αφού πλέον ο χρήστης έχει αποσυνδεθεί. Επίσης εκτελεί μια σειρά εντελών που έχουν σκοπό την αλλαγή του εικονιδίου της κατάστασης σύνδεσης του χρήστη από «  » σε «  », προκειμένου να δηλώνει ότι ο χρήστης βρίσκεται εκτός σύνδεσης.

```
private void LogMeOut()
        {
            menuItemLogOut.Enabled = false;
            menuItemLogIn.Enabled = true;
            ProgramDataGrid.Enabled = false;
            StatusBarPanel.Enabled = false;
            pictureBox5.Visible = false;
            pictureBox5.SendToBack();
            pictureBox1.BringToFront();
            pictureBox1.Visible = true;
        }
```

Πλαίσιο Κειμένου 34

5.2.2.2.8 Η συνάρτηση «menuItemAcceptChange_Click»

Η «menuItemAcceptChange_Click» διαχειρίζεται το πλήκτρο που αντιστοιχεί στην λειτουργία της «αποδοχής εφημεριών» που έχουν δημοσιεύσει οι υπόλοιποι χρήστες της εφαρμογής. Με το πάτημα του πλήκτρου «Αποδοχή Καθήκοντος» που βρίσκεται στο menu λειτουργιών, ενεργοποιείται η «menuItemAcceptChange_Click» και πραγματοποιείται ο έλεγχος για την κατάσταση της σύνδεσης του χρήστη. Εάν ισχύουν οι απαραίτητες προϋποθέσεις [Προσδιορισμός της κατάστασης σύνδεσης του χρήστη] για την επικοινωνία κεντρικού υπολογιστή-mobile εφαρμογής, τότε εκτελούνται οι εντολές για την παρουσίαση της φόρμας «ChangeList» η οποία όπως προαναφέραμε εκτελεί την λειτουργία της «αποδοχής εφημεριών», λεπτομέρειες αναφέρονται στην αντίστοιχη [Υλοποίηση της φόρμας «ChangeList]. Εάν η σύνδεση της εφαρμογής με τον κεντρικό υπολογιστή δεν είναι εφικτή,

τότε παρουσιάζεται στον χρήστη ένα μήνυμα. Το μήνυμα αυτό ενημερώνει τον χρήστη για την αιτία που δεν εκτελείται η συγκεκριμένη λειτουργία [Πλαίσιο Κειμένου 35].

```
private void menuItemAcceptChange_Click(object sender, EventArgs e)
        {
            if (CurrentST.GetStatus())
            {
                var formlog = new ChangeList();
                formlog.ShowDialog();
            }
            else
            {
MessageBox.Show("Η υπηρεσία δεν είναι Διαθέσιμή. Δεν έχετε πρόσβαση στο
Δίκτυο");
            }
        }
```

Πλαίσιο Κειμένου 35

5.2.2.2.9 Οι συναρτήσεις pictureBoxSearchProg_Click και menuItem5_Click

Οι συναρτήσεις `pictureBoxSearchProg_Click` και `menuItem5_Click` είναι υπεύθυνες για την εμφάνιση και ενεργοποίηση της φόρμας «`SearchProgram`» [Πλαίσιο Κειμένου 36]. Η «`pictureBoxSearchProg_Click`» διαχειρίζεται το πάτημα του εικονιδίου που αντιστοιχεί στην αναζήτηση και εύρεση προγράμματος ενώ η «`menuItem5_Click`» διαχειρίζεται το πάτημα του πλήκτρου «Εύρεση Προγράμματος» από το menu επιλογών στην οθόνη του χρήστη.

```
private void menuItem5_Click(object sender, EventArgs e)
        {
            var formlog = new SearchProgram();
            formlog.ShowDialog();
        }

private void pictureBoxSearchProg_Click(object sender, EventArgs e)
        {
            menuItem5_Click(sender, e);
 }
```

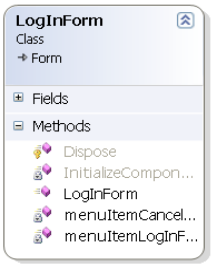
Πλαίσιο Κειμένου 36

Κάτι αντίστοιχο συμβαίνει και με τα ζεύγη συναρτήσεων `menuItemEditView_Click`, `pictureBoxEditView_Click` και `pictureBoxChangesPreview_Click`, `menuItemChangesPreview_Click`. Τα ζεύγη αυτά διαχειρίζονται την ενεργοποίηση και παρουσίαση για τις φόρμες «`ColumnSelection`» και «`PreviewChanges`» αντίστοιχα, τις οποίες θα αναλύσουμε σε επόμενες παραγράφους.

Οι συναρτήσεις `pictureBoxPublishChange_Click` και `pictureBoxAcceptChange_Click` διαχειρίζονται τα εικονίδια «  » και «  » αντίστοιχα, με το πάτημα των οποίων καλούνται οι συναρτήσεις «`menuItemPublishChange_Click`» και «`menuItemAcceptChange_Click`» τις οποίες αναλύσαμε στις προηγούμενες παραγράφους.

5.2.3 Υλοποίηση της φόρμας «LogInForm»

Η φόρμα «LogInForm» αποτελείται από τις συναρτήσεις `LogInForm`, `menuItemLogInForm_Click` και `menuItemCancelLogInForm_Click`. Οι συναρτήσεις `Dispose` και `InitializeComponent`, που φαίνονται στο διπλανό διάγραμμα, αποτελούν «by default» τμήματα της φόρμας και πραγματοποιούν τις διαδικασίες καταστροφής και δημιουργίας της.



Εικόνα 45: Διάγραμμα κλάσεων της φόρμας LogInForm

5.2.3.1 Η συνάρτηση «`LogInForm`»

Θα αρχίσουμε την παρουσίαση της υλοποίησης των συναρτήσεων ξεκινώντας από τις πιο απλές. Η συνάρτηση «`LogInForm`» η οποία έχει το ίδιο όνομα με τη φόρμα, είναι αυτή που καλεί την «by default» συνάρτηση «`InitializeComponent`» [Πλαίσιο Κειμένου 37], δημιουργώντας με αυτό τον τρόπο το γραφικό περιβάλλον της φόρμας.

```
public LogInForm()
        {
            InitializeComponent();
               }
```

Πλαίσιο Κειμένου 37

5.2.3.2 Η συνάρτηση «menuItemCancelLogInForm_Click»

Η συνάρτηση «`menuItemCancelLogInForm_Click`» ενεργοποιείται με το πάτημα του πλήκτρου «Ακύρωση» που βρίσκεται στο αριστερό μέρος του «menu bar». Οι εντολές που εκτελεί ακυρώνουν την διαδικασία ελέγχου ταυτοποίησης του χρήστη πριν ολοκληρωθεί

```
this.DialogResult = DialogResult.Cancel;
```

και κλείνουν την φόρμα «`LogInForm`»

```
this.Close();
```

με αποτέλεσμα να εμφανίζεται στην οθόνη η φόρμα που ήταν ενεργοποιημένη πριν την κλήση της «`LogInForm`» .

```
private void menuItemCancelLogInForm_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
            this.Close();
        }
```

5.2.3.3 Η συνάρτηση menuItemLogInForm_Click

Για το τέλος αφήσαμε την πιο σημαντική συνάρτηση της «LogInForm». Αυτή είναι η «menuItemLogInForm_Click» η οποία ενεργοποιείται με το πάτημα του πλήκτρου «Σύνδεση» που βρίσκεται στη «menu bar» της οθόνης. Η συγκεκριμένη συνάρτηση [Πλαίσιο Κειμένου 38] κάνει χρήση του WebService αντικειμένου «UserCheck» και της συνάρτησης «UserDataCheck» προκειμένου να στείλει στον κεντρικό υπολογιστή τα στοιχεία σύνδεσης του χρήστη και να λάβει πίσω την τιμή μιας μεταβλητής με το αποτέλεσμα της αναζήτησης.

```
SystemDataWebReference.SystemData UserCheck = new
TimeTable_V2.SystemDataWebReference.SystemData();

String UCheck =
UserCheck.UserDataCheck(textBoxUserName.Text.ToString(),
textBoxPassword.Text.ToString());
```

Πλαίσιο Κειμένου 38

Εάν τα στοιχεία σύνδεσης του χρήστη υπάρχουν στον πίνακα της ΒΔ των εγγεγραμμένων χρηστών, επιστρέφεται η τιμή «ok» με την οποία αρχικοποιούμε την συμβολοσειρά «UCheck». Στην αντίθετη περίπτωση η τιμή που επιστρέφει το WebService και θα ανατεθεί στην συμβολοσειρά «UCheck» είναι η «fail».

Εάν ο έλεγχος της ταυτοποίησης είναι αληθής, δηλαδή η μεταβλητή «UCheck» έχει την τιμή «ok», ενεργοποιείται η διαδικασία εγγραφής του «Ονόματος Χρήστη» και του μυστικού «Κωδικού Χρήστη» στην τοπική ΒΔ. Ο πίνακας της ΒΔ στον οποίο καταχωρούνται τα στοιχεία ταυτοποίησης του χρήστη ονομάζεται «UserData». Η ενέργεια αυτή εξυπηρετεί δύο σκοπούς:

- Τη χρήση των στοιχείων από άλλες κλάσεις και συναρτήσεις της εφαρμογής.
- Την πραγματοποίηση του ελέγχου ταυτοποίησης όταν η εφαρμογή δεν μπορεί να συνδεθεί με τον απομακρυσμένο κεντρικό υπολογιστή.

Πριν από την καταχώρηση των στοιχείων πραγματοποιείται ένας έλεγχος με χρήση των block «try-catch» [Πλαίσιο Κειμένου 39]. Στον έλεγχο αυτό προσπαθούμε να διαπιστώσουμε αν ο χρήστης της εφαρμογής είναι νέος, οπότε πραγματοποιείται νέα εγγραφή ή αν έχει γίνει η εγγραφή του χρήστη στο παρελθόν ώστε να αποτρέψουμε τη διπλή εγγραφή του.

```
if (UCheck == "ok")
    {
      string usrnm = textBoxUserName.Text.ToString();
      string pswd = textBoxPassword.Text.ToString();

       SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
       Files\TimeTable_V2\TimeTableDB.sdf");
      connection.Open();

      try
        {
              string cmd1 = string.Format("SELECT * FROM UserData WHERE
              Username='{0}' AND        Password='{1}'",
              textBoxUserName.Text.ToString(), textBoxPassword.Text.ToString());

              SqlCeCommand SqlCmd1 = new SqlCeCommand(cmd1, connection);
               SqlCmd1.ExecuteNonQuery();
        }
      catch (Exception ex)
        {
             string UserInfo =
             UserCheck.UserInfo(textBoxUserName.Text.ToString(),
             textBoxPassword.Text.ToString());
             string cmd2 = string.Format("INSERT INTO UserData
             (User_ID,username,password) values ({0},{1},{2})", UserInfo,
             textBoxUserName.Text.ToString(), textBoxPassword.Text.ToString());

             SqlCeCommand SqlCmd2 = new SqlCeCommand(cmd2, connection);
              SqlCmd2.ExecuteNonQuery();
        }
        MessageBox.Show("Εχετε συνδεθεί επιτυχώς");
        this.DialogResult = DialogResult.OK;
    }
```

Πλαίσιο Κειμένου 39

Σε περίπτωση αποτυχίας σύνδεσης εμφανίζεται στον χρήστη ένα μήνυμα που τον ενημερώνει για το σφάλμα που προέκυψε [Πλαίσιο Κειμένου 40].

```
else
  {
     MessageBox.Show("Δέν έχετε εισάγει σωστά στοιχεία σύνδεσης");
  }
```

Πλαίσιο Κειμένου 40

Όπως αναφέραμε και σε προηγούμενο σημείο της παραγράφου, ένας χρήστης που έχει συνδεθεί μέσω της φορητής του συσκευής έστω και για μία φορά στο σύστημα μπορεί να κάνει χρήση των περιορισμένων λειτουργιών της ακόμα και όταν βρίσκεται εκτός σύνδεσης. Ο έλεγχος για την ταυτοποίηση χρήστη σε κατάσταση χωρίς σύνδεση πραγματοποιείται με blocks «try-catch» και παρουσιάζεται στο παρακάτω πλαίσιο [Πλαίσιο Κειμένου 41].

```
catch (Exception ex)
  {
     //εκτελούνται οι εντολές του block μόνο αν ο χρήστης είναι ίδη εγγεγραμμένος
   try
     {
       SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
       Files\TimeTable_V2\TimeTableDB.sdf");
        connection.Open();

       string cmd1 = string.Format("SELECT * FROM UserData WHERE Username='{0}' AND
       Password='{1}'", textBoxUserName.Text.ToString(),
       textBoxPassword.Text.ToString());
        SqlCeCommand SqlCmd1 = new SqlCeCommand(cmd1, connection);
        SqlCmd1.ExecuteNonQuery();

        MessageBox.Show("Εχετε συνδεθεί επιτυχώς");
        this.DialogResult = DialogResult.OK;
      }

   catch
      {
         MessageBox.Show("Δέν έχετε εισάγει σωστά στοιχεία σύνδεσης"
          + "ή εισέρχεται νέος χρήστης για πρώτη φορά"+
          "χωρίς η εφαρμογή να συνδέεται με το σύστημα ενημέρωσης εφημεριών");
      }
 }
```

Πλαίσιο Κειμένου 41

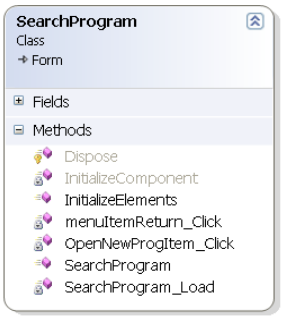
Ο συνολικός κώδικας της συνάρτησης menuItemLogInForm και της φόρμας LogInForm παρουσιάζονται στο παράρτημα με τον κώδικα της φορητής εφαρμογής.

5.2.4 Υλοποίηση της φόρμα «SearchProgram»

Οι λειτουργίας της φόρμα `SearchProgram` ή «Αναζήτηση Προγράμματος» πραγματοποιούνται από τις συναρτήσεις:

- InitializeElements
- menuItemReturn_Click
- OpenNewProgItem_Click
- SearchProgram και
- SearchProgram_Load

τις οποίες θα αναλύσουμε στην συνέχεια της ενότητας.



Εικόνα 46: Διάγραμμα κλάσεων της φόρμας ServhProgram

Η φόρμα περιλαμβάνει ένα αντικείμενο «DateTimePicker», ένα αντικείμενο «CheckBox» και τρία αντικείμενα «ComboBox». Κάθε ένα από αυτά τα αντικείμενα χρησιμοποιούνται για την επιλογή των κριτηρίων αναζήτησης από τον χρήστη. Αναλυτικότερα, στο αντικείμενο «DateTimePicker» ο χρήστης επιλέγει μία ημερομηνία προκειμένου να εμφανιστούν οι εφημερίες που αντιστοιχούν σε αυτή. Ενώ με την ενεργοποίηση του αντικειμένου «CheckBox» που βρίσκεται από κάτω, η αναζήτηση

γενικεύεται με αποτέλεσμα να επιλέγονται για εμφάνιση οι μελλοντικά προγραμματισμένες εφημερίες ή καθήκοντα.

5.2.4.1 Η συνάρτηση «InitializeElements»

Η αρχικοποίηση των παραπάνω αντικειμένων με δεδομένα πραγματοποιείται από την συνάρτηση «InitializeElements» η οποία καλείται από την συνάρτηση «SearchProgram_Load» κατά την δημιουργία της φόρμας `SearchProgram`. Σκοπός της «InitializeElements» είναι να συνδεθεί με την τοπική ΒΔ και να τροφοδοτήσει με δεδομένα τα αντικείμενα «ComboBox». Τα δεδομένα που τροφοδοτούν τα τρία 3 «ComboBox» είναι αποθηκευμένα σε τρείς πίνακες (ένας για κάθε αντικείμενο) οι οποίοι έχουν δημιουργηθεί και έχουν «γεμίσει» κατά την αρχικοποίηση της ΒΔ στην φόρμα «Πρόγραμμα».

Προκειμένου να πετύχουμε την τροφοδοσία των αντικειμένων με δεδομένα, θα χρειαστούμε για κάθε «ComboBox» να κάνουμε χρήση των αντικειμένων `DataSet`, `string` και `SqlCeDataAdapter`. Στην μεταβλητή τύπου `string` θα αποθηκεύσουμε το ερώτημα προς την ΒΔ που αντιστοιχεί στον πίνακα με τα δεδομένα που θέλουμε να περιέχει το αντίστοιχο «ComboBox» [Πλαίσιο Κειμένου 42].

Πλαίσιο Κειμένου 42

```
string cmdDutiesList = "SELECT Duty_Types FROM Declared_Duties";
string cmdLocationsList = "SELECT Location FROM Declared_Locations";
string cmdProgramNameList = "SELECT Program_Name FROM Program";
```

Στις μεταβλητές τύπου `DataSet` [Πλαίσιο Κειμένου 43] θα αποθηκευτούν οι πίνακες με τα δεδομένα που αντιστοιχούν στα παραπάνω ερωτήματα.

```
DataSet dsDeclareDutiesList = new DataSet();
DataSet dsDeclareLocationsList = new DataSet();
DataSet dsProgramNameList=new DataSet();
```

Πλαίσιο Κειμένου 43

Οι μεταβλητές `SqlCeDataAdapter` χρησιμοποιούνται και αυτές ως γέφυρα επικοινωνίας μεταξύ της ΒΔ και των `DataSet` . Αφού ανοίξουμε την σύνδεση με την τοπική ΒΔ, κάνουμε χρήση της συνάρτησης «Fill» κάθε αντικειμένου `SqlCeDataAdapter` η οποία τροφοδοτεί το αντίστοιχο `DataSet` με τα δεδομένα του αντίστοιχου πίνακα της ΒΔ [Πλαίσιο Κειμένου 44].

```
SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
Files\TimeTable_V2\TimeTableDB.sdf");
connection.Open();

SqlCeDataAdapter daDeclareDutiesList = new SqlCeDataAdapter(cmdDutiesList,connection);
SqlCeDataAdapter daDeclareLocationsList = new SqlCeDataAdapter(cmdLocationsList, connection);
SqlCeDataAdapter daProgramNameList = new SqlCeDataAdapter(cmdProgramNameList, connection);

daDeclareDutiesList.Fill(dsDeclareDutiesList, "Declared_Duties");
daDeclareLocationsList.Fill(dsDeclareLocationsList, "Declared_Locations");
daProgramNameList.Fill(dsProgramNameList, "Program");
```

Πλαίσιο Κειμένου 44

Έπειτα ορίζουμε ως πηγή δεδομένων για τα «ComboBox» [Πλαίσιο Κειμένου 45] το κατάλληλο `DataSet` που πλέον περιέχει τα κριτήρια επιλογής της αντίστοιχης κατηγορίας.

```
comboBoxDutyTypes.DataSource = dsDeclareDutiesList.Tables[0].DefaultView;
comboBoxDutyTypes.DisplayMember = "Duty_Types";
comboBoxLocations.DataSource = dsDeclareLocationsList.Tables[0].DefaultView;
comboBoxLocations.DisplayMember = "Location";
comboBoxProgramName.DataSource = dsProgramNameList.Tables[0].DefaultView;
comboBoxProgramName.DisplayMember = "Program_Name";
```

Πλαίσιο Κειμένου 45

Ολόκληρη η συνάρτηση InitializeElements μαζί με την συνάρτηση SearchProgram_Load που την καλεί, παρουσιάζονται στο παράρτημα με τον κωδικό της φορητής εφαρμογής.

5.2.4.2 Οι συναρτήσεις «OpenNewProgItem_Click» και «menuItemReturn_Click»

Για την υποβολή των κριτηρίων αναζήτησης, ο χρήστης θα πρέπει να πατήσει το πλήκτρο «Άνοιγμα» της menu bar και θα εμφανιστεί στην φόρμα «Πρόγραμμα» το πρόγραμμα εφημεριών-καθηκόντων που αντιστοιχεί στα κριτήρια αναζήτησης που έχει θέσει. Εάν ο χρήστης δεν θέλει να κάνει χρήση της λειτουργίας αναζήτησης, τότε έχει την δυνατότητα να επιστρέψει στην προηγούμενη οθόνη που βρισκόταν, με το πάτημα του πλήκτρου «Επιστροφή» της menu bar.

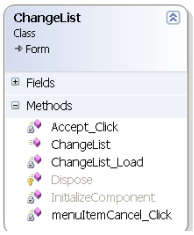
Με το πάτημα του πλήκτρου «Άνοιγμα» ενεργοποιείται η συνάρτηση OpenNewProgItem_Click. Λειτουργία της συνάρτησης είναι να τροποποιεί το τρέχων ερώτημα προς την βάση δεδομένων προκειμένου να εμφανίζονται στον πίνακα του προγράμματος οι εργασίες εκείνες που ανταποκρίνονται στα συγκεκριμένα χαρακτηριστικά που έχει ορίσει ο χρήστης.

5.2.5 Υλοποίηση της φόρμας «ChangeList»

Οι λειτουργίες της φόρμας εκτελούνται από τις συναρτήσεις:

- ChangeList_Load,
- Accept_Click,
- menuItemCancel_Click,

τις οποίες θα αναλύσουμε στην συνέχεια της ενότητας.



Εικόνα 47: Διάγραμμα κλάσσης της φόρμας ChangeList

Η λίστα με τις δημοσιευμένες εργασίες των χρηστών, εμφανίζεται στην οθόνη με την χρήση ενός αντικειμένου «DataGrid». Τα δεδομένα που εμφανίζονται στο «DataGrid» προέρχονται από έναν συγκεντρωτικό πίνακα με τις δημοσιευμένες εργασίες όλων των χρηστών που βρίσκεται αποθηκευμένος στον απομακρυσμένο κεντρικό υπολογιστή.

5.2.5.1 Η συνάρτηση «ChangeList_Load»

Πηγή δεδομένων για το «DataGrid» αποτελεί μία μεταβλητή `DataSet` με όνομα «`ChangeListDataSet`», ενώ τροφοδοτείται με δεδομένα από την συνάρτηση «`UsersChangeList`» του WebService αντικειμένου `TableContentWebService` [Πλαίσιο Κειμένου 46].

```
private void ChangeList_Load(object sender, EventArgs e)
        {
       SystemDataWebReference.SystemData TableContentWebService = new
       TimeTable_V2.SystemDataWebReference.SystemData();

        DataSet ChangeListDataSet = new DataSet("UsersChangeList");
        ChangeListDataSet = TableContentWebService.UsersChangeList();

       ChangeListDataGrid.DataSource = ChangeListDataSet.Tables["UsersChangeList"];
        }
```

Πλαίσιο Κειμένου 46

5.2.5.2 Η συνάρτηση «Accept_Click»

Όταν ο χρήστης επιλέξει από την λίστα την εργασία που επιθυμεί να εντάξει στο πρόγραμμά του, στην συνέχεια πρέπει να πατήσει το πλήκτρο «Αποδοχή» προκειμένου να πραγματοποιηθεί η ανάθεση της εργασίας. Με το πάτημα του πλήκτρου ενεργοποιείται η συνάρτηση «`Accept_Click`» και ο αλγόριθμος εντοπίζει τις συντεταγμένες στις οποίες βρίσκεται η εργασία που έχει αποδεχτεί ο χρήστης [Πλαίσιο Κειμένου 47] και πραγματοποιεί την αποθήκευση κάθε πεδίου-στήλης που αντιστοιχεί στην συγκεκριμένη εργασία. Η αποθήκευση των πεδίων γίνεται σε ξεχωριστές μεταβλητές προκειμένου να επιτύχουμε αποδοτικότερη διαχείριση των δεδομένων.

```
int colNum = ChangeListDataGrid.CurrentCell.ColumnNumber;
int rowNum = ChangeListDataGrid.CurrentCell.RowNumber;

object Program_ID = ChangeListDataGrid[rowNum, 2];
object Date = ChangeListDataGrid[rowNum, 3];
object Duty_Type = ChangeListDataGrid[rowNum, 4];
object Duty_Start_Time = ChangeListDataGrid[rowNum, 5];
object Duty_End_Time = ChangeListDataGrid[rowNum, 6];
object Location = ChangeListDataGrid[rowNum, 7];
object Program_Name = ChangeListDataGrid[rowNum, 8];
object User_ID = ChangeListDataGrid[rowNum, 9];
```

Πλαίσιο Κειμένου 47

Γνωρίζοντας πλέον την εργασία που έχει αποδεχτεί ο χρήστης για να ενθέσει στο πρόγραμμά του, θα πρέπει να προβούμε σε μια σειρά ενημερώσεων. Πρώτα θα πρέπει να διαγράψουμε από την λίστα των δημοσιευμένων (προς ανάθεση) εφημέριων, την εφημερία-εργασία που μόλις ο χρήστης έκανε δεκτή στο πρόγραμμά του. Αυτή η λίστα βρίσκεται στην ΒΔ «SystemData» του απομακρυσμένου κεντρικού υπολογιστή και υπάρχει αντίγραφό της στην τοπική ΒΔ, με την όψη που ταυτίζεται με τα χαρακτηριστικά (Ομάδα Χρήστη, Τμήμα) του χρήστη. Αρχικά η διαγραφή γίνεται τοπικά [Πλαίσιο Κειμένου 48]:

```
string cmd = "DELETE FROM ChangeList WHERE Program_ID=@Program_ID";
SqlCeConnection connection = new SqlCeConnection(@"Data Source=\Program
Files\TimeTable_V2\TimeTableDB.sdf");

connection.Open();
SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
SqlCmd.ExecuteNonQuery();
```

Πλαίσιο Κειμένου 48

Και στη συνέχεια με την χρήση της μεθόδου «UpdateChanges» του Web Service πραγματοποιούμε την διαγραφή της εργασίας και από την ΒΔ του απομακρυσμένου υπολογιστή:

```
SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();

TableContentWebService.UpdateChanges("xpchris", Program_ID.ToString());
```

Εάν ο χρήστης κάνει λάθος και πατήσει το πλήκτρο της αποδοχής χωρίς να έχει επιλέξει πρώτα την εργασία που επιθυμεί, η εφαρμογή θα τον ενημερώσει με το ακόλουθο μήνυμα:

```
MessageBox.Show("Δεν έχετε επιλέξει αντικείμενο για Αποδοχή");
```

Ολόκληρη η συνάρτηση «Accept_Click» παρουσιάζεται στο παρακάτω πλαίσιο [Πλαίσιο Κειμένου 49].

```
private void Accept_Click(object sender, EventArgs e)
        {
            try
            {
                int colNum = ChangeListDataGrid.CurrentCell.ColumnNumber;
                int rowNum = ChangeListDataGrid.CurrentCell.RowNumber;

                object Program_ID = ChangeListDataGrid[rowNum, 2];
                object Date = ChangeListDataGrid[rowNum, 3];
                object Duty_Type = ChangeListDataGrid[rowNum, 4];
                object Duty_Start_Time = ChangeListDataGrid[rowNum, 5];
                object Duty_End_Time = ChangeListDataGrid[rowNum, 6];
                object Location = ChangeListDataGrid[rowNum, 7];
                object Program_Name = ChangeListDataGrid[rowNum, 8];
                object User_ID = ChangeListDataGrid[rowNum, 9];

              SystemDataWebReference.SystemData TableContentWebService = new
              TimeTable_V2.SystemDataWebReference.SystemData();

                string cmd = "DELETE FROM ChangeList WHERE Program_ID=@Program_ID";
              SqlCeConnection connection = new SqlCeConnection(@"Data
              Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");

              connection.Open();
                SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
                SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
                SqlCmd.ExecuteNonQuery();

               TableContentWebService.UpdateChanges("xpchris",
               Program_ID.ToString());

                var formlog = new TimeTableForm();
                this.Close();
            }
            catch
            {
                MessageBox.Show("Δεν έχετε επιλέξει αντικείμενο για Αποδοχή");
            }
        }
```

Πλαίσιο Κειμένου 49

Τέλος ο χρήστης εάν δεν επιθυμεί να κάνει χρήση της υπηρεσίας, πατώντας το πλήκτρο «Επιστροφή» επιστρέφει στην προηγούμενη οθόνη:

```
private void menuItemCancel_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
        }
```

5.2.6 Υλοποίηση της φόρμας «Edit_Event»

5.2.6.1 Η κλάση «CurrentDataReader»

Η φόρμα «Edit_Event» προκειμένου να χρησιμοποιήσει σε πάνω από μία συναρτήσεις δεδομένα τα οποία δεν τροποποιούνται κατά την διαδικασία της δημοσίευσης της επιλεγμένης εφημερίας, διαθέτει την κλάση «CurrentDataReader». Η κλάση αυτή αποθηκεύει τα δεδομένα «User_ID» και «Program_ID» τα οποία θα χρησιμοποιηθούν αντίστοιχα για την ταυτοποίηση του χρήστη και της επιλεγμένης προς δημοσίευση εργασίας. Η διαχείριση των δεδομένων αυτών, όπως έχουμε δει και σε προηγούμενες κλάσεις,

πραγματοποιείται με την ύπαρξη των αντίστοιχων για την κλάση συναρτήσεων «Get» και «Set». Παρακάτω παρατηρούμε τον κώδικα της κλάσης «CurrentDataReader».

```
public class CurrentDataReader
    {
        public string User_ID;
        public string Program_ID;

        public string GetUser()
        {
            return User_ID;
        }

        public string GetProgram()
        {
            return Program_ID;
        }

        public void SetCurrentDr(string UserID, string ProgramID)
        {
            User_ID = UserID;
            Program_ID = ProgramID;
        }
    }
```

5.2.6.2 Η συνάρτηση «SetChangeOptions»

Όπως έχουμε ήδη προαναφέρει, προκειμένου ο χρήστης να προβεί σε δημοσίευση εργασίας αρκεί να την επιλέξει επάνω στο πλέγμα το οποίο εμφανίζεται και να πατήσει το πλήκτρο «  ». Με το πάτημα του πλήκτρου τα δεδομένα μεταβιβάζονται από την φόρμα «Πρόγραμμα» στη φόρμα «Edit_Event» μέσω της συνάρτησης «SetChangeOptions». Η συνάρτηση αυτή ανήκει στη φόρμα «Edit_Event» και δέχεται ως όρισμα ένα αντικείμενο «SqlCeDataReader» το οποίο περιέχει όλα τα πεδία-στήλες της εφημερίας που επέλεξε ο χρήστης για δημοσίευση. Πρώτη εντολή της συνάρτησης είναι η κλήση της κλάσης για την αποθήκευση των χαρακτηριστικών στοιχείων της εφημερίας. Η αρχικοποίηση των μεταβλητών της κλάσης «CurrentDataReader» με τις τιμές των δεδομένων που θέλουμε να διαφυλάξουμε, πραγματοποιείται με την συνάρτηση «SetCurrentDr».

```
CDR.SetCurrentDr(SpecificRow["User_ID"].ToString(),SpecificRow["Program_ID"].ToString())
;
```

Μια ακόμη λειτουργία της συνάρτησης «SetChangeOptions» είναι και η αρχικοποίηση των αντικειμένων «Text Box» με τις πληροφορίες της επιλεγμένης εφημερίας. Η λειτουργία αυτή αφενός αποσκοπεί στον επανέλεγχο των στοιχείων της εφημερίας από τον χρήστη και αφετέρου στην δυνατότητα τροποποίησής τους όταν ο χρήστης επιθυμεί να πραγματοποιήσει αίτηση αλλαγής προς την γραμματεία. Στο παρακάτω πλαίσιο παρουσιάζεται ο κώδικας της συνάρτησης «SetChangeOptions».

```
public void SetChangeOptions(SqlCeDataReader SpecificRow)
{
CDR.SetCurrentDr(SpecificRow["User_ID"].ToString(),
SpecificRow["Program_ID"].ToString());

            ChangedDateTextBox.Text = SpecificRow["Date"].ToString();
            ChangedLocTextBox.Text = SpecificRow["Location"].ToString();
            ChangedDutyTextBox.Text = SpecificRow["Duty_Type"].ToString();
            ChangedSTtextBox.Text = SpecificRow["Duty_Start_Time"].ToString();
            ChangedETtextBox.Text = SpecificRow["Duty_End_Time"].ToString();
        }
```

5.2.6.3 Η συνάρτηση «ShareDutyButton_Click»

Προκειμένου τώρα ο χρήστης να προβεί σε δημοσίευση της εφημερίας, αρκεί να πατήσει το πλήκτρο «Δημοσίευση Εργασίας». Με το πάτημα του πλήκτρου ενεργοποιείται η διαδικασία ενημέρωσης της «Λίστας Αλλαγών» που βρίσκεται στην ΒΔ του κεντρικό υπολογιστή, καθώς και η ενημέρωση του πίνακα «ChangeList» της τοπικής ΒΔ που περιέχει τις εφημερίες που έχει επιλέξει ο χρήστης προς δημοσίευση. Η συνάρτηση που εκτελεί τις εντολές για την παραπάνω διαδικασία είναι η «ShareDutyButton_Click».

Τα δεδομένα με τα οποία ενημερώνονται αυτοί οι δύο πίνακες είναι τα στοιχεία που αποθηκεύουμε στην μοναδική κλάση της φόρμας, δηλαδή τα «User_ID» και «Program_ID». Η χρήση αυτών των δύο χαρακτηριστικών πληροφοριών δηλώνει ποιός χρήστης δημοσίευσε την εργασία-εφημερία με κωδικό = «Program_ID». Όπως κάθε φορά που επιθυμούμε να πραγματοποιήσουμε συναλλαγή με την ΒΔ του απομακρυσμένου κεντρικού υπολογιστή, έτσι και τώρα κάνουμε χρήση υπηρεσιών Web Service. Αυτή την φορά χρησιμοποιούμε την συνάρτηση «InsertChanges» του Web Service αντικείμενου «TableContentWebService».

```
SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();

TableContentWebService.InsertChanges(User_ID, Program_ID);
```

Το «query string» είναι κοινό και για τις δύο ενημερώσεις που θέλουμε να πραγματοποιήσουμε.

```
string cmd="INSERT INTO ChangeList (User_ID,Program_ID) values
(@User_ID,@Program_ID)";
```

Εφόσον ολοκληρωθεί η λειτουργία της Δημοσίευσης της Εργασίας, η φόρμα «Edit_Event» κλείνει και παρουσιάζεται πάλι στον χρήστη η οθόνη με το πρόγραμμα εφημεριών-καθηκόντων. Στη συνέχεια παρουσιάζεται ο κώδικας της συνάρτησης

«ShareDutyButton_Click».

```
private void ShareDutyButton_Click(object sender, EventArgs e)
        {
               SystemDataWebReference.SystemData TableContentWebService = new
               TimeTable_V2.SystemDataWebReference.SystemData();

            string User_ID=CDR.GetUser();
            string Program_ID=CDR.GetProgram();
              string cmd="INSERT INTO ChangeList (User_ID,Program_ID) values (
              @User_ID,@Program_ID)";

              SqlCeConnection connection = new SqlCeConnection(@"Data
              Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
            connection.Open();

            SqlCeCommand SqlCmd=new SqlCeCommand(cmd,connection);
            SqlCmd.Parameters.AddWithValue("@User_ID",User_ID);
            SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd.ExecuteNonQuery();

            TableContentWebService.InsertChanges(User_ID, Program_ID);

            this.Close();
        }
```

5.2.6.4 Η συνάρτηση «Edit_Event_Load»

Όπως προαναφέραμε η φόρμα «Edit_Event» αποτελεί προπύργιο για την αποστολή αίτησης τροποποίησης της εργασίας στην γραμματεία και για ορισμένους ακόμα λόγους που θα αναφέρουμε στη συνέχεια, αφού όμως πρώτα μελετήσουμε τον τρόπο με τον οποίο πραγματοποιείται η διαμόρφωση της φόρμας «Edit_Event». Κατά την αρχικοποίηση της φόρμας «Edit_Event» τα πλήκτρα αλλαγής ώρας καθώς και τα πλήκτρα «Μεταφορά Εφημερίας» και «Ακύρωση Εφημερίας» έχουν προγραμματιστεί ώστε να μην είναι ορατά.

```
private void Edit_Event_Load(object sender, EventArgs e)
        {
            buttonChangeHour2.Visible = false;
            ChangeStartTimeButton.Visible = false;
            buttonMoveDuty.Visible = false;
            buttonCancelDuty.Visible = false;
        }
```

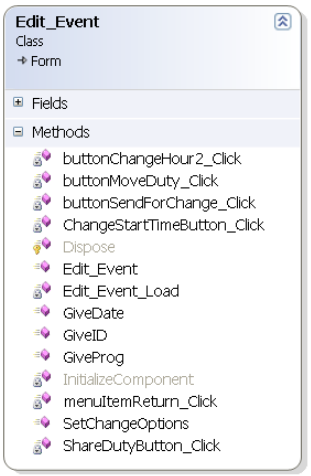
5.2.6.5 Η συνάρτηση «buttonSendForChange_Click»

Για την διαμόρφωση της φόρμας κατά το πάτημα του πλήκτρου «Διαμόρφωση για Γραμματεία» εκτελούνται οι εντολές που αποκρύπτουν το πλήκτρο αυτό και εμφανίζουν τα υπόλοιπα. Οι εντολές φαίνονται στο πλαίσιο που ακολουθεί.

```
private void buttonSendForChange_Click(object sender, EventArgs e)
        {
            ShareDutyButton.Visible = false;
            buttonChangeHour2.Visible = true;
            ChangeStartTimeButton.Visible = true;
            buttonMoveDuty.Visible = true;
            buttonCancelDuty.Visible = true;
            buttonSendForChange.Visible = false;
        }
```

5.2.6.6 Η «κρυφή» υποστήριξη της φόρμας «Edit_Event» στην φόρμα «ChangeStartTimeRequestForm»

Εκτός των προαναφερθέντων, η φόρμα «Edit_Event» παρέχει «κρυφή» υποστήριξη στην «ChangeStartTimeRequestForm». Στο διπλανό διάγραμμα παρατηρούμε τις συναρτήσεις GiveDate, GiveID, GiveProg. Οι συναρτήσεις αυτές έχουν στις μεταβλητές τους αποθηκευμένες τιμές οι οποίες θα χρησιμοποιηθούν προκειμένου να κληθούν από την φόρμα «ChangeStartTimeRequestForm» και να την τροφοδοτήσουν με τα αντίστοιχα δεδομένα. Τα δεδομένα αυτά είναι:



Εικόνα 48: Διάγραμμα κλάσεων της φόρμας Edit_Event

- Η τρέχουσα ημερομηνία της εργασίας,
- το ID του χρήστη που πραγματοποιεί την αίτηση και
- το ID της εφημερίας-εργασίας η οποία θα τροποποιηθεί.

Στα πλαίσια της υποστήριξης της φόρμας «ChangeStartTimeRequestForm» είναι και οι συναρτήσεις «ChangeStartTimeButton_Click» και «buttonMoveDuty_Click» οι οποίες ενεργοποιούνται αντίστοιχα με το πάτημα των πλήκτρων αλλαγής ώρας και του πλήκτρου «Μεταφορά Εφημερίας». Ο πλήρως κώδικας της φόρμας «Edit_Event» και της «ChangeStartTimeRequestForm» παρουσιάζονται στο αντίστοιχο παράρτημα που περιέχει τον κώδικα της εφαρμογής [Κώδικας φορητής εφαρμογής].

5.2.7 Υλοποίηση της φόρμας «ChangeStartTimeRequestForm»

5.2.7.1 Οι συναρτήσεις «SetHours» και «SetDate»

Όπως έχουμε ήδη αναφέρει, κύριο χαρακτηριστικό της φόρμας «ChangeStartTimeRequestForm» είναι οι διαμορφώσεις στις οποίες υποβάλλεται. Για την διαμόρφωση που αντιστοιχεί στην αίτηση αλλαγής ώρας, τον πρώτο λόγο έχει η συνάρτηση «SetHours» της φόρμας «ChangeStartTimeRequestForm». Η συνάρτηση αυτή αρχικοποιεί τα αντικείμενα «Text Box» που αντιπροσωπεύουν τις τρέχουσες ώρες έναρξης και λήξης της εκάστοτε εφημερίας [Πλαίσιο Κειμένου 50].

```
CurrentStartHourTextBox.Text = StartHour;
CurrentEndHourTextBox.Text = EndHour;
```

Πλαίσιο Κειμένου 50

ξης και λήξης των εφημεριών είναι μέρος των ορισμάτων της συνάρτησης [Πλαίσιο Κειμένου 51]. Η μεταβλητή που συμπληρώνει την λίστα των ορισμάτων είναι η συμβολοσειρά «Flag».

```
public void SetHours(string StartHour, string EndHour,string Flag)
```

Πλαίσιο Κειμένου 51

Όπως είναι εμφανές και από το όνομά της, η μεταβλητή «Flag» λειτουργεί σαν σημαία για την συγκεκριμένη συνάρτηση. Έχουμε ορίσει ότι αν η τιμή της μεταβλητής αυτής ισούται με την λέξη «hour» [Πλαίσιο Κειμένου 52], τότε θα εκτελεστούν οι κατάλληλες εντολές ώστε η φόρμα «ChangeStartTimeRequestForm» να διαμορφωθεί αντίστοιχα προκειμένου να τηρεί τις προδιαγραφές για την σύνταξη για την αίτηση αλλαγής ώρας.

```
if (Flag == "hour")
   {
     panelChangeDate.Visible = false;
     panelSplit.Visible = true;
     panelSplit.BackColor = Color.White;
     buttonComboChange.Visible = true;
     buttonComboChange.BringToFront();
     buttonComboChange.Text = "Μεταφορά Εφημερίας";
    }
```

Πλαίσιο Κειμένου 52

Όπως έχουμε δει και σε προηγούμενες περιπτώσεις, οι εντολές που εκτελούνται μέσα στο block του `if` διαχειρίζονται την εμφάνιση και την θέση των αντικειμένων της φόρμας διαμορφώνοντάς την κατάλληλα [Πλαίσιο Κειμένου 53].

```
public void SetHours(string StartHour, string EndHour,string Flag)
        {
            CurrentStartHourTextBox.Text = StartHour;
            CurrentEndHourTextBox.Text = EndHour;

            if (Flag == "hour")
            {
                panelChangeDate.Visible = false;
                panelSplit.Visible = true;
                panelSplit.BackColor = Color.White;
                buttonComboChange.Visible = true;
                buttonComboChange.BringToFront();
                buttonComboChange.Text = "Μεταφορά Εφημερίας";
            }
        }
```

Πλαίσιο Κειμένου 53

Την ίδια λογική και υλοποίηση ακολουθεί και η συνάρτηση «SetDate» που παρουσιάζεται παρακάτω [Πλαίσιο Κειμένου 54]. Η συνάρτηση «SetDate» τροποποιεί με τις εντολές της την φόρμα «ChangeStartTimeRequestForm» προκειμένου να την διαμορφώσει κατάλληλα για την σύνταξη της αίτησης «Μεταφορά Εφημερίας».

```
public void SetDate(string OldDate, string Flag)
        {
            textBoxOldDate.Text = OldDate;

            if (Flag == "date")
            {
                panelChangeHour.Visible = false;
                panelChangeDate.Location=new Point(0,0);
                panelSplit.Visible = true;
                buttonComboChange.Visible = true;
                panelSplit.BackColor = Color.White;
                buttonComboChange.BringToFront();
                buttonComboChange.Text = "Αλλαγή Ωρας";
            }
        }
```

Πλαίσιο Κειμένου 54

Η συνάρτηση «SetHours» της φόρμας «ChangeStartTimeRequestForm» που παρουσιάστηκε παραπάνω, καλείται από την συνάρτηση «ChangeStartTimeButton_Click» [Πλαίσιο Κειμένου 55] της φόρμας «Edit_Event» και η οποία ενεργοποιείται όταν ο χρήστης πατήσει το πλήκτρο «Αλλαγή (της ώρας)» στη φόρμα «Edit_Event». Αντίστοιχα και η συνάρτηση «SetDate» της φόρμας «ChangeStartTimeRequestForm» καλείται από την συνάρτηση «buttonMoveDuty_Click» της φόρμας «Edit_Event» η οποία ενεργοποιείται όταν ο χρήστης πατήσει το πλήκτρο «Μεταφορά Εφημερίας» στην φόρμα «Edit_Event».

```
private void ChangeStartTimeButton_Click(object sender, EventArgs e)
        {
            var formlog = new ChangeStartTimeRequestForm();
            string HourFlag = "hour";

               formlog.SetHours(ChangedSTtextBox.Text.ToString(),
               ChangedETtextBox.Text.ToString(),HourFlag);
               formlog.SetDate(ChangedDateTextBox.Text.ToString(), HourFlag);
               formlog.ShowDialog();
        }

        private void buttonMoveDuty_Click(object sender, EventArgs e)
        {
            var formlog = new ChangeStartTimeRequestForm();
            string FlagDate = "date";
            formlog.SetDate(ChangedDateTextBox.Text.ToString(), FlagDate);
              formlog.SetHours(ChangedSTtextBox.Text.ToString(),
              ChangedETtextBox.Text.ToString(), FlagDate);
            formlog.ShowDialog();
        }
```

Πλαίσιο Κειμένου 55

Αν παρατηρήσουμε προσεκτικά τα παραπάνω πλαίσια με τις υλοποίηση των συναρτήσεων «ChangeStartTimeButton_Click» και «buttonMoveDuty_Click» θα διαπιστώσουμε ότι ενώ ενεργοποιούνται με διαφορετικά πλήκτρα από την φόρμα «Edit_Event», καλούν τις ίδιες συναρτήσεις της φόρμας «ChangeStartTimeRequestForm» τροφοδοτώντας την με δεδομένα.

```
formlog.SetHours(ChangedSTtextBox.Text.ToString(),ChangedETtextBox.Text.ToString(),HourFlag)
;
```

```
formlog.SetDate(ChangedDateTextBox.Text.ToString(), HourFlag);
```

Ο λόγος για τον οποίο ενεργοποιούνται και οι δύο συναρτήσεις είναι προκειμένου να έχουμε στη διάθεση μας όλα τα στοιχεία τροποποίησης της εφημερίας (ώρες + ημερομηνία). Έτσι δίνεται στο χρήστη η δυνατότητα να μεταβεί από την αίτηση αλλαγής ώρας στην αίτηση μεταφοράς της ημερομηνίας μέσω των πλήκτρων «συγχώνευσης».

Δηλαδή, τα δεδομένα που μεταβιβάζονται κάθε φορά από την φόρμα «Edit_Event» στην φόρμα «ChangeStartTimeRequestForm» είναι τα ίδια, άσχετα με το αν ο χρήστης πατήσει το πλήκτρο για αλλαγή ώρας ή για μεταφορά της εφημερίας. Το στοιχείο που καθορίζει την μορφή της διαμόρφωσης που θα έχει η φόρμα «ChangeStartTimeRequestForm» είναι η μεταβλητή «Flag». Όπως έχουμε αναφέρει και παραπάνω, εάν η τιμή της «Flag» ισούται με «hour» τότε θα ενεργοποιηθούν οι εντολές που διαμορφώνουν την φόρμα για την αίτηση αλλαγής ώρας της εφημερίας, ενώ αν η τιμή της «Flag» ισούται με «date» τότε θα ενεργοποιηθούν οι εντολές που διαμορφώνουν την φόρμα για την αίτηση μεταφοράς της εφημερίας.

5.2.7.2 Τα πλήκτρα συγχώνευσης

Όπως είδαμε στην προηγούμενη παράγραφο, τα δεδομένα που δέχεται η φόρμα «ChangeStartTimeRequestForm» είναι τα ίδια ασχέτως της αίτησης που επιθυμεί να υποβάλει ο χρήστης. Για τη μετάβαση από την μία μορφή υποβολής αίτησης στην άλλη, χρησιμοποιούμε τα πλήκτρα συγχώνευσης. Η λειτουργία των πλήκτρων αυτών είναι να ελέγχουν την τιμή (όνομα) που έχει το πλήκτρο συγχώνευσης. Εάν το όνομα του πλήκτρου είναι «Μεταφορά Εφημερίας», σημαίνει ότι ο χρήστης βρίσκεται ήδη στην αίτηση αλλαγής ώρας και θέλει να μεταβεί την αίτηση «Μεταφορά Εφημερίας», ενώ το αντίθετο συμβαίνει όταν το όνομα του πλήκτρου είναι «Αλλαγή Ωρας».

```
private void buttonComboChange_Click(object sender, EventArgs e)
        {
            if (buttonComboChange.Text == "Μεταφορά Εφημερίας")
            {
                panelChangeDate.Visible = true;
                panelSplit.Visible = true;
                panelSplit.BackColor = Color.Gray;
                buttonComboChange.Visible = false;
                panelChangeDate.Visible = true;
            }

            if (buttonComboChange.Text == "Αλλαγή Ωρας")
            {
                panelChangeDate.Location = new Point(3, 159);
                panelChangeHour.Location = new Point(0, 0);
                panelSplit.Visible = true;
                panelSplit.BackColor = Color.Gray;
                panelSplit.BringToFront();
                buttonComboChange.Visible = false;
                panelChangeHour.Visible = true;
            }
        }
```

Πλαίσιο Κειμένου 56: Κώδικας που πραγματοποιεί τις λειτουργίες των πλήκτρων συγχώνευσης

Το όνομα που θα έχει το πλήκτρο συγχώνευσης εξαρτάται κάθε φορά από τον τύπο της αίτηση που εμφανίζεται στην οθόνη του χρήστη, δηλαδή από τις εκάστοτε εντολές που ενεργοποιεί η «Flag». Στο παραπάνω πλαίσιο παρουσιάζεται η υλοποίηση της συνάρτησης «buttonComboChange_Click» η οποία ενεργοποιείται με το πάτημα του πλήκτρου συγχώνευσης και εκτελεί τις απαραίτητες εντολές για την δημιουργία της κοινής αίτησης «Αλλαγής Ώρας» και «Μεταφορά Εφημερίας».

5.2.7.3 Πλήκτρα επικύρωσης της εισαγωγής των νέων στοιχείων του προγράμματος

Προκειμένου ο χρήστης να επικυρώσει τα νέα στοιχεία μεταφοράς της εργασίας θα πρέπει μετά από κάθε αλλαγή να πατά τα πλήκτρα «OK» που βρίσκονται στην δεξιά μεριά των πλαισίων συμπλήρωσης. Με το πάτημα των πλήκτρων αυτών, τα νέα στοιχεία που έχει εισάγει ο χρήστης αποθηκεύονται στην κλάση «ModificationData» [Πλαίσιο Κειμένου 57] η οποία συγκεντρώνει όλα τα στοιχεία που θα πρέπει να αποσταλούν στην γραμματεία για να πραγματοποιηθεί η υποβολή της αίτησης αλλαγής. Τα στοιχεία αυτά είναι: οι νέες ώρες έναρξης και λήξης της εργασίας, η νέα ημερομηνία μεταφοράς, ο κωδικός της υποψήφιας για αλλαγή εργασίας και ο κωδικός του χρήστη που υποβάλει την αίτηση. Η κλάση «ModificationData» παρουσιάζεται στο ακόλουθο πλαίσιο.

```
public class ModificationData
    {
        public string NewStartHour = "0";
        public string NewEndHour = "0";
        public string NewDate = "0";
        public string UserID = "0";
        public int ProgramID = 0;

        public void SetSHour(string SHour)
        {
            NewStartHour = SHour;
        }

        public void SetEHour(string EHour)
        {
            NewEndHour = EHour;
        }

        public void SetNDate(string NDate)
        {
            NewDate = NDate;
        }

        public void SetIDs(string User_ID,
int Program_ID)
        {
            UserID = User_ID;
            ProgramID = Program_ID;
        }

        public string GetSHour()
        {
            return NewStartHour;
        }

        public string GetEHour()
        {
            return NewEndHour;
        }

        public string GetNDate()
        {
            return NewDate;
        }

        public string GetUID()
        {
            return UserID;
        }

        public int GetPID()
        {
            return ProgramID;
        }
    }
```

Πλαίσιο Κειμένου 57

Ο κώδικας των πλήκτρων επικύρωσης των αλλαγών που εισήγαγε ο χρήστης παρουσιάζεται στη συνέχεια.

5.2.7.4 Η συνάρτηση «menuItemSubmit_Click»

Σκοπός όλων όσων έχουμε δημιουργήσει παραπάνω είναι η αποστολή αίτησης στην γραμματεία της Νοσοκομειακής Μονάδας. Έχοντας αναλύσει τον τρόπο με τον οποίο συμπληρώνει ο χρήστης κάθε μια από τις φόρμες, αλλά και τον τρόπο με τον οποίο λειτουργεί κάθε μία από αυτές, απομένει να μελετήσουμε τον τρόπο αποστολής τους. Εφόσον ο χρήστης έχει συμπληρώσει την φόρμα που επιθυμεί προκειμένου να τροποποιήσει την εφημερία του, πατά το πλήκτρο «Υποβολή» για να αποσταλεί στην γραμματεία. Η συνάρτηση που εκτελείται με το πάτημα του πλήκτρου «Υποβολή» είναι η «`menuItemSubmit_Click`». Κύριο αντικείμενο της συνάρτησης αυτής είναι η κλήση των κατάλληλων Web μεθόδων των υπηρεσιών του Web Service για την εγγραφή των αλλαγών στην ΚΒΔ.

Πιο συγκεκριμένα, καλώντας την Web μέθοδο «`ProgramModification`» πραγματοποιείται η αποστολή των νέων στοιχείων αλλαγής της εργασίας που είναι αποθηκευμένα στην κλάση «ModificationData», ενώ καλώντας την αντίστοιχη Web μέθοδο πραγματοποιείται η αποστολή των στοιχείων ταυτοποίησης του χρήστη (User_ID), της υποψήφιας προς αλλαγή εργασίας (Program_ID) και του τύπου αίτησης που αποστέλλει ο χρήστης (Application Type= αλλαγή εργασίας/εφημερίας). Στη συνέχεια ακολουθεί ο κώδικας της συνάρτησης «`menuItemSubmit_Click`» [Πλαίσιο Κειμένου 58].

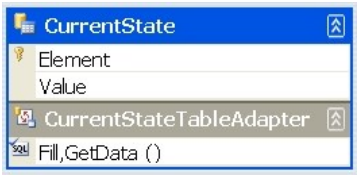
```
private void menuItemSubmit_Click(object sender, EventArgs e)
        {
       string App_Type = "Αλλαγή Εργασίας/Εφημερίας";
       SystemDataWebReference.SystemData TableContentWebService = new
       TimeTable_V2.SystemDataWebReference.SystemData();

       TableContentWebService.ProgramModification(ModifyProgram.GetUID(),ModifyProg
       ram.GetPID(), ModifyProgram.GetSHour(), ModifyProgram.GetEHour(),
       ModifyProgram.GetNDate());

       TableContentWebService.InsertApplications(ModifyProgram.GetUID(),ModifyProgr
       am.GetPID(), App_Type);

       this.Close();
       }
```

Πλαίσιο Κειμένου 58

5.2.8 Υλοποίηση της φόρμας «ColumnSelection»

Ο χρήστης έχει την δυνατότητα να επιλέξει τις στήλες που θα εμφανίζονται μέσω των αντικειμένων «Check Box» που εμφανίζονται στην οθόνη. Εάν το κουτάκι που αντιπροσωπεύει κάθε στήλη είναι επιλεγμένο «☑», τότε τα δεδομένα της στήλης αυτής θα εμφανίζονται στο χρήστη. Η «by default» εμφάνιση του προγράμματος περιλαμβάνει όλες τις στήλες. Ο χρήστης έχει την δυνατότητα να απενεργοποιήσει το κουτάκι και την στήλη που αντιπροσωπεύει, πατώντας πάνω του. Όταν ο χρήστης ολοκληρώσει την επιλογή των στηλών

τότε πατά το πλήκτρο «ok» για την υποβολή των ρυθμίσεων. Για την αποτελεσματική υλοποίηση όλων των λειτουργιών που εκτελούνται μέσω της φόρμας «ColumnSelection» έπρεπε να κάνουμε χρήση της τοπικής ΒΔ ώστε να αποθηκεύουμε τις ρυθμίσεις που έχει πραγματοποιήσει ο χρήστης. Στον πίνακα «CurrentState» αποθηκεύουμε την κατάσταση κάθε αντικειμένου «Check Box». Με τον τρόπο αυτό η εφαρμογή έχει την δυνατότητα να ανατρέξει οποιαδήποτε στιγμή στην ΒΔ και ελέγχοντας τον πίνακα «CurrentState» να εμφανίσει το πρόγραμμα εφημεριών σύμφωνα με τις ρυθμίσεις του χρήστη.



Από το «σχήμα (schema)» του πίνακα «CurrentState» αντιλαμβανόμαστε ότι ο πίνακας έχει δύο στήλες. Στη στήλη με όνομα «Element» αποθηκεύουμε το όνομα που αντιστοιχεί στο αντικείμενο «Check Box», ενώ στη στήλη «Value» αποθηκεύουμε την τιμή που έχει το αντικείμενο «Check Box». Έχουμε ορίσει να αποθηκεύουμε στη στήλη «Value» την τιμή «`enabled`» για κάθε επιλεγμένο (`xCheckBox.CheckState = CheckState.Checked;`) από τον χρήστη αντικείμενο «Check Box» και στην αντίθετη περίπτωση την τιμή «`unable`».

Η φόρμα «`ColumnSelection`» κατά την αρχικοποίηση της (συνάρτηση «`ColumnSelection_Load`») εκτελεί εντολές που ελέγχουν από την ΒΔ τις αποθηκευμένες ρυθμίσεις του χρήστη και τις εφαρμόζουν «τσεκάροντας» τα αντικείμενα που αντιστοιχούν στις προς εμφάνιση στήλες.

Επίσης με χρήση των εντολών διαχείρισης εξαιρέσεων «`try-catch`» ελέγχουμε εάν είναι η πρώτη φορά που ο χρήστης εφαρμόζει τις ρυθμίσεις του, όπου και πραγματοποιείται η αρχικοποίηση των αντικειμένων «Check Box» καθώς και η αντίστοιχη αποθήκευση της κατάστασής τους στην ΒΔ. Θα πρέπει να αναφέρουμε στο σημείο αυτό ότι η αρχικοποίηση των αντικειμένων (η by default κατάσταση) προβλέπει σε ενεργοποίηση όλων των στηλών του προγράμματος εφημεριών-καθηκόντων.

5.2.8.1 Η συνάρτηση «ApplyColChanges_Click»

Στην συνέχεια η συνάρτηση «`ApplyColChanges_Click`» που ενεργοποιείται με το πάτημα του πλήκτρου «OK» επιτελεί το πολύπλοκο έργο της αυτόματης δημιουργίας ενός «query string». Το συγκεκριμένο ερώτημα καθορίζει τις στήλες του προγράμματος που θα ανακτηθούν από την ΒΔ και θα εμφανιστούν στην κεντρική οθόνη του χρήστη.

Το έργο αυτό είναι πολύπλοκο επειδή το «query string» ακολουθεί συγκεκριμένους κανόνες σύνταξης. Για τον λόγο αυτό θα πρέπει να προβλεφθούν ορισμένοι συνδυασμοί ταυτόχρονα ενεργοποιημένων αντικειμένων «Check Box» προκειμένου να τηρηθούν με

ακρίβεια οι συντακτικοί κανόνες του «query string», όπως για παράδειγμα η χρήση του «,» μεταξύ των επιλεγμένων στηλών του προγράμματος .

Για την λύση του συγκεκριμένου προβλήματος έπρεπε να προγραμματίσουμε όλους τους πιθανούς συνδυασμούς επιλογών που θα μπορούσε να κάνει ο χρήστης της φορητής εφαρμογής. Ο χρήστης μπορεί να επέμβει στην εμφάνιση μόνο των τριών από των 6 βασικών στηλών αφού οι στήλες : ημερομηνία, ώρα έναρξης και ώρα λήξης της εφημερίας θα πρέπει να εμφανίζονται συνεχώς. Έτσι λοιπόν όλοι οι πιθανοί συνδυασμοί για τρείς στήλες είναι 2^3=8 διαφορετικές υλοποίησης «if». Σε κάθε ένα «if» μπλοκ εκτελείται και ένα διαφορετικό string που αποστέλλεται προς την ΒΔ προκειμένου να εμφανιστεί στον πίνακα του προγράμματος η προσαρμοσμένη μορφή εμφάνισης που έχει ορίσει ο χρήστης. Ο κώδικας της συνάρτησης παρουσιάζεται στο παράρτημα του κώδικα της φορητής εφαρμογής

5.2.8.2 Η συνάρτηση «menuItemReturn_Click»

Τέλος, εάν ο χρήστης δεν επιθυμεί να κάνει χρήση της λειτουργίας «Επιλογή Στηλών» έχει την δυνατότητα να μεταφερθεί την κεντρική οθόνη της εφαρμογής με το πάτημα του πλήκτρου «Επιστροφή». Η συνάρτηση που διαχειρίζεται το πάτημα του συγκεκριμένου πλήκτρου είναι η «`menuItemReturn_Click`» η οποία παρουσιάζονται στο παράρτημα με τον κώδικα ολόκληρης της εργασίας μαζί με τις υπόλοιπες συναρτήσεις της φόρμας «`ColumnSelection`».

5.3 Υλοποίηση Λειτουργιών Κεντρικής Εφαρμογής

5.3.1 Υλοποίηση της φόρμας «UserManagement»

Η φόρμα «UserManagement» έχει δημιουργηθεί με σκοπό την εγγραφή των χρηστών στο σύστημα εφημεριών/καθηκόντων. Αποτελεί μια φόρμα συμπλήρωσης των στοιχείων του χρήστη με αντικείμενα textbox και DropDownList. Πηγή δεδομένων για τα αντικείμενα DropDownList είναι συγκεκριμένοι πίνακες της Κεντρικής Βάσης Δεδομένων.

Η ολοκλήρωση της εγγραφής του χρήστη αποτελείται από δύο στάδια. Στο πρώτο στάδιο πραγματοποιείται η εγγραφή των στοιχείων εκείνων που προορίζονται για τον πίνακα Users, ενώ κατά το δεύτερο στάδιο πραγματοποιείται οι εγγραφή των στοιχείων στους πίνακες Address, Declared Duties, Declared Locations και Phone Numbers. Ο λόγος για την ύπαρξη δύο σταδίων αποθήκευσης είναι ότι με την εγγραφή του χρήστη στον πίνακα Users ο χρήστης που εγγράφεται αποκτά τον χαρακτηριστικό κωδικό ID που τον αντιπροσωπεύει σε οποιαδήποτε συναλλαγή. Έχοντας λοιπόν ο χρήστης αποκτήσει τον κωδικό ID

πραγματοποιείται στη συνέχεια και η ολοκλήρωση της εγγραφής των στοιχείων στους υπόλοιπους πίνακες με την χρήση του ID του χρήστη.

5.3.1.1 Εγγραφή των στοιχείων του χρήστη στον πίνακα «Users»

Η εγγραφή των στοιχείων του χρήστη πραγματοποιείται με το πάτημα του πλήκτρου «Αποθήκευση Χρήστη». Με τον τρόπο αυτό ενεργοποιείται η συνάρτηση «ButtonSaveUser_Click» η οποία πραγματοποιεί την σύνδεση με την Κεντρική Βάση Δεδομένων και εκτελεί το query της εγγραφής των δεδομένων στην Βάση.

Παράλληλα πραγματοποιείται και έλεγχος διπλο-εγγραφής του χρήστη. Ο έλεγχος αυτός βασίζεται στο αποτέλεσμα της εκτέλεσης των εντολών προς την ΚΒΔ. Η βάση δεδομένων είναι έτσι σχεδιασμένη ώστε εάν η εγγραφή των δεδομένων δεν πραγματοποιηθεί με επιτυχία, ο μόνος λόγος αποτυχίας της είναι ότι η εγγραφή αυτή υπάρχει ήδη στη βάση δεδομένων.

```
protected void ButtonSaveUser_Click(object sender, EventArgs e)
        {
       string sConn2 = "data source=FERRARI-
       4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
       string cmd = "INSERT INTO Users
       (User_Team,name,surname,username,password,AMKA,Thesi,Eidikothta,Bathmida)
       values
       (@User_Team,@name_user,@surname_user,@username,@password,@AMKA,@Thesi,@Eidik
       othta,@Bathmida)";

       SqlConnection conn = new SqlConnection(sConn2);
       conn.Open();
            try
            {
                SqlCommand SqlCmd = new SqlCommand(cmd, conn);
              SqlCmd.Parameters.AddWithValue("@User_Team",
              DropDownListUserTeam.Text);
              SqlCmd.Parameters.AddWithValue("@name_user", TextBoxName.Text);
              SqlCmd.Parameters.AddWithValue("@surname_user",
              TextBoxSurname.Text);
              SqlCmd.Parameters.AddWithValue("@username", TextBoxUserName.Text);
              SqlCmd.Parameters.AddWithValue("@password", TextBoxPassword.Text);
                SqlCmd.Parameters.AddWithValue("@AMKA", TextBoxAMKA.Text);
                SqlCmd.Parameters.AddWithValue("@Thesi", TextBoxThesi.Text);
              SqlCmd.Parameters.AddWithValue("@Eidikothta",
              DropDownListEidikothta.Text);
              SqlCmd.Parameters.AddWithValue("@Bathmida",
              DropDownListBathmida.Text);
                SqlCmd.ExecuteNonQuery();
            }

            catch
            {
                Console.Write("Ο χρήστης είναι ήδη εγγεγραμένος");
            }
```

Πλαίσιο Κειμένου 59: Η συνάρτηση "ButtonSaveUser_Click"

5.3.1.2 Εγγραφή των στοιχείων του χρήστη στους υπόλοιπους πίνακες

Έχοντας αποκτήσει τον χαρακτηριστικό κωδικό ID συνεχίζουμε την διαδικασία εγγραφής των στοιχείων του χρήστη στους πίνακες Address, Declared Duties, Declared

Locations και Phone Numbers στους οποίους πραγματοποιείται η αποθήκευση των διευθύνσεων, τηλεφώνων, δηλωμένων προς ενημέρωση τύπων εργασίας και τοποθεσιών του χρήστη. Η αποθήκευση καθ' ενός από τα παραπάνω στοιχεία πραγματοποιείται με το πάτημα του πλήκτρου «OK» που βρίσκεται στα δεξιά κάθε πλαισίου συμπλήρωσης. Με το πάτημα του συγκεκριμένου πλήκτρου εκτελείται η αντίστοιχη συνάρτηση με στόχο την ανάκτηση του ID του χρήστη του οποίου γίνεται η εγγραφή και στη συνέχεια την αποθήκευση του στοιχείου στον κατάλληλο πίνακα. Για παράδειγμα εάν ο χρήστης της Κεντρικής Εφαρμογής επιθυμεί να εισάγει μία διεύθυνση (έστω τη διεύθυνση ένα), τότε μετά την συμπλήρωση των στοιχείων διεύθυνση, νομός, Τ.Κ. κτλ. πατά το πλήκτρο «OK» στα δεξιά. Αμέσως ενεργοποιείται η συνάρτηση «ButtonDieuth1_Click» η οποία όπως προαναφέραμε πραγματοποιεί την ανάκτηση του ID του χρήστη του οποίου γίνεται η εγγραφή και έπειτα την αποθήκευση των στοιχείων της διεύθυνσης στον πίνακα Address.

```
protected void ButtonDieuth1_Click(object sender, EventArgs e)
        {

            string sConn2 = "data source=FERRARI-
            4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Address
            (User_ID,nomos,dimos,city,address,TK,perioxi,Xwra) VALUES
            (@User_ID,@nomos,@dimos,@city,@address,@TK,@perioxi,@Xwra)";
            string cmd3 = "Select User_ID From Users Where User_Team=@User_Team AND
            name=@name_user AND surname=@surname_user AND username=@username AND
            password=@password";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd3 = new SqlCommand(cmd3, conn);
            SqlCmd3.Parameters.AddWithValue("@User_Team",
            DropDownListUserTeam.Text);
             SqlCmd3.Parameters.AddWithValue("@name_user", TextBoxName.Text);
            SqlCmd3.Parameters.AddWithValue("@surname_user", TextBoxSurname.Text);
            SqlCmd3.Parameters.AddWithValue("@username", TextBoxUserName.Text);
            SqlCmd3.Parameters.AddWithValue("@password", TextBoxPassword.Text);
            SqlCmd3.ExecuteNonQuery();
             System.Data.SqlClient.SqlDataReader dr = SqlCmd3.ExecuteReader();
             dr.Read();

            string userid = dr["User_ID"].ToString();

            dr.Close();

               System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
               conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", userid);
            SqlCmd2.Parameters.AddWithValue("@nomos", TextBoxNom1.Text);
            SqlCmd2.Parameters.AddWithValue("@orofos", TextBoxDum1.Text);
            SqlCmd2.Parameters.AddWithValue("@city", TextBoxCity1.Text);
            SqlCmd2.Parameters.AddWithValue("@address", TextBoxDrom1.Text);
            SqlCmd2.Parameters.AddWithValue("@TK", TextBoxTKK.Text);
            SqlCmd2.Parameters.AddWithValue("@perioxi", TextBoxLoc1.Text);
            SqlCmd2.Parameters.AddWithValue("@Xwra", TextBoxCun1.Text);
            SqlCmd2.ExecuteNonQuery();
        }
```

Πλαίσιο Κειμένου 60: Ο κώδικας της συνάρτησης "ButtonDieuth1_Click"

5.3.2 Υλοποίηση της φόρμας «InitProgramCreation»

Όπως έχουμε ήδη προαναφέρει η διαδικασία της δημιουργίας προγράμματος εφημεριών/καθηκόντων πραγματοποιείται σε δύο βήματα. Στο πρώτο βήμα ο χρήστης της Κεντρικής Εφαρμογής ορίζει τα άτομα ή τις ομάδες ατόμων για τα οποία προορίζεται το πρόγραμμα εφημεριών/καθηκόντων, ενώ στο δεύτερο βήμα η διαδικασία εξατομικεύεται δίνοντας στο χρήστη την δυνατότητα να καθορίσει τα στοιχεία του προγράμματος για κάθε εργαζόμενο ξεχωριστά.

5.3.2.1 Πρώτο βήμα της διαδικασίας δημιουργίας προγράμματος εφημεριών/καθηκόντων

Κατά πρώτο βήμα της διαδικασίας για την δημιουργία προγράμματος ο χρήστης της Κεντρικής Εφαρμογής (Work Station) καθορίζει τα άτομα ή τις ομάδες ατόμων για τα οποία προορίζεται το πρόγραμμα που πρόκειται να δημιουργήσει στην συνέχεια. Ο χρήστης της Κεντρικής Εφαρμογής έχει την δυνατότητα να καθορίσει την ομάδα ενδιαφερόντων για το συγκεκριμένο πρόγραμμα μέσα από μία διαδικασία όμοια της αναζήτησης. Ο χρήστης εισάγει τα χαρακτηριστικά στοιχεία των ατόμων ή της ομάδας των ατόμων που επιθυμεί και έπειτα πατά το πλήκτρο «Δημιουργία». Πατώντας το πλήκτρο αυτό πραγματοποιείται αναζήτηση των χρηστών που διαθέτουν τα συγκεκριμένα χαρακτηριστικά στοιχεία και τα οποία είναι: η ομάδα εργαζομένων στην όποια θα πρέπει να ανήκει το άτομα ή τα άτομα για τα οποία προορίζεται το πρόγραμμα, το τμήμα της νοσοκομειακής μονάδας στο οποίο ανήκουν, η βαθμίδα και η θέση που κατέχουν. Υποχρεωτική είναι η ονομασία του νέου προγράμματος με τρόπο χαρακτηριστικό του περιεχομένου του.

Με το πάτημα του πλήκτρου «Δημιουργία» ενεργοποιούνται οι εντολές που πραγματοποιούν την αναζήτηση των εργαζομένων με τα χαρακτηριστικά γνωρίσματα που εισήγαγε ο χρήστης της κεντρικής εφαρμογής τα οποία στην συνέχεια εμφανίζονται σε ένα συγκεντρωτικό πίνακα. Η συνάρτηση που εκτελεί τις ενέργειες αυτές είναι η «ButtonCreate_Click». Ο χρήστης της Κεντρικής Εφαρμογής συνεχίζει την διαδικασία της δημιουργίας του νέου προγράμματος σε νέα φόρμα στην οποία μεταβαίνει πατώντας το πλήκτρο «Συνέχεια» το οποίο πλήκτρο ενεργοποιεί την συνάρτηση «ButtonContinue_Click».

```
protected void ButtonCreate_Click(object sender, EventArgs e)
{
    string sConn2 = "data source=FERRARI-4000\\SQLEXPRESS;Database=SystemData;User
    Id=xpchris;Password=123456";

    string Department = string.Format("Department='{0}' AND",
    DropDownListDepartment.Text);
    string Bathmida = string.Format("Bathmida='{0}'", DropDownListBathmida.Text);
    string UserTeam = string.Format("User_Team='{0}'", DropDownListUserTeam.Text);
    string Eidikothta = string.Format("Eidikothta='{0}' AND", D
       ropDownListEidikothta.Text);
              string and1 = " AND ";
    string sSQL = "select name+ ' ' +surname AS Όνομα,Department AS Τμήμα,Thesi AS
    Θέση,Eidikothta AS Ειδικότητα,Bathmida AS Βαθμίδα,User_ID from Users Where ";
               string TotalQuery = "";

    if (CheckBoxUserTeam.Checked == CheckBoxDepartment.Checked ==
    CheckBoxEidikothta.Checked == CheckBoxBathmida.Checked == true)
    {
        sSQL = "select name+ ' ' +surname AS Όνομα,User_Team AS
       Ομάδα_Εργασίας,Department AS Τμήμα,Thesi AS Θέση,Eidikothta AS
       Ειδικότητα,Bathmida AS Βαθμίδα,User_ID from Users";
    }

    if (CheckBoxUserTeam.Checked == true)
    {
        TotalQuery = sSQL + Department + Eidikothta + Bathmida;
        if (CheckBoxDepartment.Checked == true)
        {
            TotalQuery = sSQL + Eidikothta + Bathmida;
            if (CheckBoxEidikothta.Checked == true)
            {
                TotalQuery = sSQL + Bathmida;
                if (CheckBoxBathmida.Checked == true)
                {
                    TotalQuery = sSQL;
                }
            }
        }

    }
    else
    {
       TotalQuery = sSQL + UserTeam + and1 + Department + Eidikothta + Bathmida;
        if (CheckBoxDepartment.Checked == true)
        {
            TotalQuery = sSQL + UserTeam + and1 + Eidikothta + Bathmida;
            if (CheckBoxEidikothta.Checked == true)
            {
                TotalQuery = sSQL + UserTeam + and1 + Bathmida;
                if (CheckBoxBathmida.Checked == true)
                {
                    TotalQuery = sSQL + UserTeam;
                }
            }
        }
    }

    TextBoxQuery.Text = TotalQuery;
    SqlConnection conn = new SqlConnection(sConn2);

    DataSet dsCreate = new DataSet();
    SqlDataAdapter daCreate = new SqlDataAdapter(TotalQuery, sConn2);
    daCreate.Fill(dsCreate, "Users");

    Session["myDataSet"] = dsCreate;
    Session["NewProgramName"] = TextBoxProgName.Text;

    CurrDS.SetDS(dsCreate);

    DataTable myDataTable = dsCreate.Tables[0];

    Specific_Users_List.DataSource = myDataTable;
    Specific_Users_List.DataBind();

}
```

Πλαίσιο Κειμένου 61: Ο κώδικας της συνάρτησης "ButtonCreate_Click"

5.3.3 Υλοποίηση της φόρμας «ContinueProgCreation»

5.3.3.1 Δεύτερο βήμα, ολοκλήρωση της διαδικασίας δημιουργίας προγράμματος εφημεριών/καθηκόντων

Έχοντας εισάγει τα στοιχεία και το όνομα του προγράμματος που έχει δημιουργηθεί κατά το πρώτο βήμα της διαδικασίας, πατώντας ο χρήστης το πλήκτρο «Συνέχεια» καλείται να ολοκληρώσει την διαδικασία δημιουργίας προγράμματος με την επιλογή των εργαζομένων για τους οποίους απευθύνεται το πρόγραμμα και με τις λεπτομέρειες των εφημεριών/καθηκόντων που αντιστοιχούν σε κάθε εργαζόμενο.

Αποστολή της φόρμας «ContinueProgCreation», η οποία εκτελεί την διαδικασία της εξατομίκευσης του νέου προγράμματος, είναι η απόκτηση των στοιχείων των χρηστών που ανταποκρίνονται στα χαρακτηριστικά γνωρίσματα του χρήστη της κεντρικής εφαρμογής και τα οποία ανακτήθηκαν κατά την διαδικασία του πρώτο βήματος από την φόρμα «InitProgCreation». Τα στοιχεία τα οποία ενδιαφερόμαστε να ανακτήσουμε από την προηγούμενη φόρμα είναι το αντικείμενο DataSet που περιέχει τα στοιχεία των εργαζομένων που παρουσιάστηκαν στον πίνακα «Κατάλογος Χρηστών» της φόρμας «InitProgCreation» καθώς και το όνομα του νέου προγράμματος που εισήγαγε ο χρήστης στην ίδια φόρμα («InitProgCreation»).

Η ανάκτηση των δεδομένων αυτών πραγματοποιείται με μεταβλητές τύπου Session οι οποίες ορίζονται στην φόρμα «InitProgCreation» και χρησιμοποιούνται από την φόρμα «ContinueProgCreation».

```
Session["myDataSet"] = dsCreate;
Session["NewProgramName"] = TextBoxProgName.Text;
```

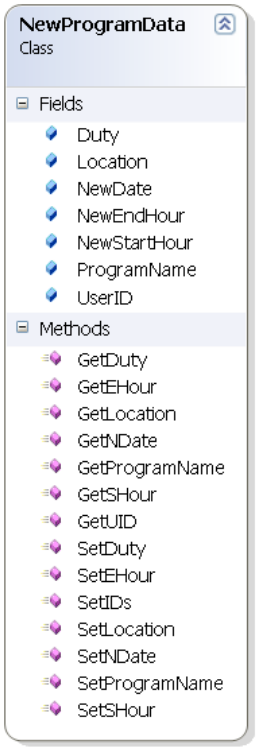
Κειμένου 62: Ορισμός "Session" μεταβλητών στην φόρμα "InitProgCreation"

```
if (Session["NewProgramName"] != null)
            {
                NewProgram.SetProgramName((string)Session["NewProgramName"]);
            }
            if (Session["myDataSet"] != null)
            {
                DataSet datasetCreate = new DataSet();
                datasetCreate = (DataSet)Session["myDataSet"];
               …
           }
```

Πλαίσιο Κειμένου 63: Χρήση των "Session" μεταβλητών από την φόρμα "ContinueProgCreation"

Τα δεδομένα του DataSet χρησιμοποιούνται ως πηγή δεδομένων για το DropDownList στο οποίο ο χρήστης της Κεντρικής Εφαρμογής επιλέγει τους εργαζομένους

που θα συμπεριλάβει στο νέο πρόγραμμα που δημιουργεί. Στη συνέχεια εισάγει τις υπόλοιπες λεπτομέρειες του προγράμματος για κάθε χρήστη με το πάτημα του πλήκτρου «OK» που βρίσκεται στα δεξιά κάθε πλαισίου συμπλήρωσης. Οι λεπτομέρειες αυτές, εκτός από το ονοματεπώνυμο του εργαζόμενου, είναι ο τύπος της εργασίας, οι ώρες έναρξης και λήξης της εργασίας, η ημερομηνία και την τοποθεσία στην οποία θα λάβει χώρα η εργασία. Όλα αυτά τα δεδομένα αποθηκεύονται προσωρινά στην κλάση «NewPragramData» προκειμένου στη συνέχεια να αποθηκευτούν ομαδικά στον πίνακα «Program».



Εικόνα 49: Διάγραμμα της κλάσης "NewProgramData"

Κάθε φορά που ο χρήστης πατά ένα από τα πλήκτρα «OK», πραγματοποιείται η αποθήκευση του συγκεκριμένου στοιχείου στην κλάση «NewProgramData».

```
protected void ButtonSTCon_Click(object sender, EventArgs e)
        {
            NewProgram.SetSHour(TextBoxStartTime.Text);
        }
```

Πλαίσιο Κειμένου 64: Συνάρτηση που εκτελείται με το πάτημα του πλήκτρου "OK" και αποθηκεύει την ώρα έναρξης της εργασίας στην κλάση "NewProgramData"

Όταν ο χρήστης της κεντρικής εφαρμογής (αρμόδιος για την δημιουργία του προγράμματος) ολοκλήρωσει την συμπλήρωση της φόρμας θα πρέπει να πατήσει το πλήκτρο «Αποθήκευση» προκειμένου το πρόγραμμα του εργαζομένου που επεξεργαζόταν να αποθηκευτεί στην ΚΒΔ. Με το πάτημα του πλήκτρου «Αποθήκευση» εκτελούνται οι εντολές της συνάρτησης «ButtonSave_Click» οι οποίες πραγματοποιούν την αποθήκευση των στοιχείων του προγράμματος στην ΚΒΔ.

```
protected void ButtonSave_Click(object sender, EventArgs e)
        {
            string sConn2 = "data source=FERRARI-
            4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "INSERT INTO Program
            (Date,Duty_Type,Duty_Start_Time,Duty_End_Time,Location,User_ID,Program_
            Name) values
            (@Date,@Duty_Type,@Duty_Start_Time,@Duty_End_Time,@Location,@User_ID,@P
            rogram_Name)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            try
            {
                SqlCommand SqlCmd = new SqlCommand(cmd, conn);
                SqlCmd.Parameters.AddWithValue("@Date", NewProgram.GetNDate());
                SqlCmd.Parameters.AddWithValue("@Duty_Type", NewProgram.GetDuty());
                SqlCmd.Parameters.AddWithValue("@Duty_Start_Time",
                NewProgram.GetSHour());
                SqlCmd.Parameters.AddWithValue("@Duty_End_Time",
                NewProgram.GetEHour());
                SqlCmd.Parameters.AddWithValue("@Location",
                NewProgram.GetLocation());
                SqlCmd.Parameters.AddWithValue("@User_ID", );
                SqlCmd.Parameters.AddWithValue("@Program_Name",
                NewProgram.GetProgramName());
                SqlCmd.ExecuteNonQuery();
            }

            catch
            {
               Console.Write("ο χρήστης που προσπαθείται να εισάγεται υπάρχει
               ήδη");
            }
        }
```

Πλαίσιο Κειμένου 65: Η συνάρτηση "ButtonSave_Click" πραγματοποιεί την αποθληκευση των στοιχείων του προγράμματος στην ΚΒΔ

Ο χρήστης έχει την δυνατότητα επέκτασης των δυνατοτήτων του με την χρήση των πλήκτρων «Ολοκλήρωση Δημιουργίας Προγράμματος», «Εισαγωγή Χρήστη στο τρέχον Πρόγραμμα», «Δημιουργία Νέου Προγράμματος», «Επιστροφή», «Ακύρωση Διαδικασίας».

Το πλήκτρο «Δημιουργία Νέου Προγράμματος» χρησιμοποιείται προκειμένου ο χρήστης να πλοηγηθεί γρήγορα και εύκολα στην φόρμα δημιουργίας νέου προγράμματος. Η φόρμα αυτή δεν είναι άλλη από την «InitProgramCreation»

```
protected void ButtonNewProgram_Click(object sender, EventArgs e)
        {
            ProgramDisplay formlog = new ProgramDisplay();
            Response.Redirect("InitProgramCreation.aspx");
        }
```

Πλαίσιο Κειμένου 66: Η συγκεκριμένη συνάρτηση ενεργοποιείται με το πάτημα του πλήκτρου "Δημιουργία Νέου Προγράμματος" και καλεί την φόρμα "InitProgramCreation"

Το πλήκτρο «Εισαγωγή Χρήστη στο Τρέχον Πρόγραμμα» χρησιμοποιείται για να «καθαρίσει» η φόρμα στην οποία βρίσκεται εκείνη τη στιγμή ο χρήστης ώστε να είναι σε θέση να εισάγει νέο χρήστη στο πρόγραμμα που δημιουργεί. Πρέπει να υπενθυμίσουμε όμως πως προκειμένου να πραγματοποιηθεί η συγκεκριμένη ενέργεια, η φόρμα έχει την απαίτηση

των «Session» μεταβλητών «`Session["NewProgramName"]`», η οποία εμπεριέχει το όνομα του προγράμματος που δημιουργεί ο χρήστης καθώς επίσης και το όνομα της «`Session["myDataSet"]`» η οποία περιέχει την λίστα των εργαζομένων για τους οποίους προορίζεται το υπό κατασκευή πρόγραμμα. Οι συναρτήσεις αυτές αποστέλλεται από την φόρμα «InitProgramCreation» προς την φόρμα «ContinueProgCreation» με την εναλλαγή κατά την πλοήγηση από τη μία στην άλλη (με την σειρά που αναφέρθηκαν).

Στην ουσία το πάτημα του πλήκτρου «Εισαγωγή Χρήστη στο Τρέχον Πρόγραμμα» προκαλεί ξανά την αρχικοποίηση της φόρμας «ContinueProgCreation» προκειμένου να πραγματοποιηθεί η εισαγωγή νέου εργαζομένου στο πρόγραμμα. Τώρα όμως δεν μπορούμε να πάρουμε τις μεταβλητές από τη φόρμα «InitProgramCreation». Η λύση στο πρόβλημα αυτού είναι η δημιουργία «Session» μεταβλητών στη φόρμα «ContinueProgCreation» με τα ίδια ονόματα και τις ίδιες τιμές της «InitProgramCreation» που έχουμε ήδη πάρει κατά την πρώτη κλήση της «ContinueProgCreation». Η συγκεκριμένη διαδικασία πραγματοποιείται κατά την αρχικοποίηση της φόρμας «ContinueProgCreation».

Επίσης το πάτημα του πλήκτρου «Εισαγωγή Χρήστη στο Τρέχον Πρόγραμμα» ενεργοποιεί της εντολές που εκτελούνται κατά την επαναφόρτωση (αρχικοποίηση) της φόρμας «ContinueProgCreation» ώστε να δημιουργηθεί ένας πίνακας που εμφανίζει μια προεπισκόπηση του μέχρι τώρα προγράμματος που έχει δημιουργήσει ο χρήστης.

```
protected void Page_Load(object sender, EventArgs e)
        {
            if (Session["NewProgramName"] != null)
            {
                //αποθήκευση του ονόματος που έχει
                //το νέο πρόγραμμα στην κλάση
                //προκείμένου να το αποθηκεύσουμε
                //στην συνέχεια στην ΚΒΔ
                NewProgram.SetProgramName((string)Session["NewProgramName"]);

                //δημιουργία ίδιας session μεταβλητής που θα
                //χρησιμοποιηθεί κατά την επαναφώρτωση της φόρμας
                Session["NewProgramName"] = NewProgram.GetProgramName();

                try
                {
                    //δημιουργία πίνακα προεπισκόπησης
                    //νέου προγράμματος
                     string sConn2 = "data source=FERRARI-
                     4000\\SQLEXPRESS;Database=SystemData;User
                     Id=xpchris;Password=123456";
                    SqlConnection conn = new SqlConnection(sConn2);
                    conn.Open();
                       string.Format("select * from Program where
                      Program_Name='{0}'", NewProgram.GetProgramName());
                    DataSet dsNewProg = new DataSet();
                      SqlDataAdapter daNewProg = new SqlDataAdapter(cmd2, sConn2);
                    daNewProg.Fill(dsNewProg, "Program");
                    DataTable myDataTable = dsNewProg.Tables[0];
                    GridViewPreviewProg.DataSource = myDataTable;
                    GridViewPreviewProg.DataBind();
                }
                catch
                {

                }
            }

            if (Session["myDataSet"] != null)
            {
                //χρήση της session μεταβλητής
                //με την λίστα των εργαζομένων
                //για τους οποίους προορίζεται
                //το νέο πρόγραμμα
                DataSet datasetCreate = new DataSet();
                datasetCreate = (DataSet)Session["myDataSet"];

                //δημιουργία ίδιας session μεταβλητής που θα
                //χρησιμοποιηθεί κατά την επαναφώρτωση της φόρμας
                Session["myDataSet"] = datasetCreate;

                //φόρτωση του πλαισίου λίστας με τα
                //ονόματα των εργαζομένων
                DropDownListNames.DataSource = datasetCreate.Tables[0];
                DropDownListNames.DataValueField = "Ονομα";
                DropDownListNames.DataTextField = "Ονομα";
                DropDownListNames.DataBind();

                string testq = string.Format("Ονομα='{0}'",
                DropDownListNames.Text);
                datasetCreate.Tables[0].Constraints.Add("pk_User_ID",
                datasetCreate.Tables[0].Columns[0], true);
                DataRow drow =
                datasetCreate.Tables[0].Rows.Find(DropDownListNames.Text);
                TextBoxTest.Text = drow[0].ToString() + "'" + drow[1].ToString();
            }
        }
```

Πλαίσιο Κειμένου 67: Οι εντολές της συνάρτησης "Page_Load" εκτελούνται κατά την αρχικοποίηση της φόρμας "ContinueProgCreation"

Με το πάτημα του πλήκτρου «Επιστροφή» πραγματοποιείται η διαγραφή των στοιχείων που έχουν συμπληρωθεί μόνο στην συγκεκριμένη φόρμα που βρισκόταν πριν πατήσει το πλήκτρο και μεταβαίνει αυτόματα στην φόρμα «InitProgramCreation» η οποία περιέχει τις ρυθμίσεις και τα στοιχεία των χρηστών που είχε εισάγει ο χρήστης κατά το ξεκίνημα της διαδικασίας δημιουργίας του προγράμματος.

Με το πάτημα του πλήκτρου «Ολοκλήρωση Δημιουργίας Προγράμματος» ολοκληρώνεται η διαδικασία της δημιουργίας του προγράμματος, αποθηκεύεται το νέο πρόγραμμα στην Κεντρική Βάση Δεδομένων και αρχίζει η διαδικασία δημοσίευση του προγράμματος στους εργαζομένους και παράλληλα χρήστες του συστήματος και οι οποίοι περιλαμβάνονται στο νέο πρόγραμμα που δημιουργήθηκε.

Με το πάτημα του πλήκτρου «Ακύρωση Διαδικασίας» ο χρήστης έχει την δυνατότητα ολικής διαγραφής των όσων δεδομένων δημιουργήθηκαν κατά την διαδικασία σύνταξης του νέου προγράμματος.

```
string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
string cmd = "DELETE FROM Program WHERE Program_Name='@Program_Name'";
SqlConnection conn = new SqlConnection(sConn2);
conn.Open();
try
{
    SqlCommand SqlCmd = new SqlCommand(cmd, conn);
  SqlCmd.Parameters.AddWithValue("@Program_Name",
  NewProgram.GetProgramName());
    SqlCmd.ExecuteNonQuery();
}
catch
{
    Console.Write("παρουσιάστηκε σφάλμα κατά την διαγραφή");
}
```

Πλαίσιο Κειμένου 68: Με το πάτημα του πλήκτρου "Ακύρωση Διαδικασίας" διαγράφονται τα δεδομένα από την ΚΒΔ που καταγράφηκαν κατά την διαδικασία της σύνταξης του προγράμματος

5.3.4 Υλοποίηση της φόρμας «ChangeApplicationManagement»

Η φόρμα «ChangeApplicationManagement» εκτελεί την υπηρεσία της τροποποίησης προγράμματος μετά από την αίτηση αλλαγής που υπέβαλε χρήστης της φορητής εφαρμογής. Ο κατάλογος των αιτήσεων αλλαγής που έχουν υποβάλει οι χρήστες προκύπτει από την άντληση των δεδομένων του πίνακα «Applications» της ΚΒΔ σε συνδυασμό με στοιχεία άλλων πινάκων όπως ο πίνακας «Users» προκειμένου ο χρήστης της κεντρικής εφαρμογής να έχει πλήρη ενημέρωση όσον αφορά τα άτομα που έχουν υποβάλει αίτηση αλλαγής της εργασίας τους. Η συγκεκριμένη διαδικασία της εμφάνισης της λίστας αιτήσεων πραγματοποιείται κατά την αρχικοποίηση της φόρμας.

```
protected void Page_Load(object sender, EventArgs e)
        {
string sConn2 = "data source=FERRARI-4000\\SQLEXPRESS;Database=SystemData;User
Id=xpchris;Password=123456";
string sSQL = "select A.[App_Type] AS Τύπος_Αίτησης,E.* from Applications A inner
join (select U.name AS Όνομα, U.surname AS Επώνυμο,C.* from Users U inner join
(select p.Program_ID AS Κωδικός,p.Date AS Ημερομηνία,p.Duty_Type AS
Εργασία,p.Duty_Start_Time AS Έναρξη,p.Duty_End_Time AS Λήξη,p.Location AS
Τοποθεσία,p.Program_Name,p.User_ID from Program p inner join Applications l on
l.Program_ID=p.Program_ID)C on C.User_ID=U.User_ID)E on E.User_ID=A.User_ID";

            SqlConnection conn = new SqlConnection(sConn2);
            DataSet dsApplications = new DataSet();
            SqlDataAdapter daApplications = new SqlDataAdapter(sSQL, sConn2);
            daApplications.Fill(dsApplications, "Applications");
            DataTable myDataTable = dsApplications.Tables[0];
            GridViewApplications.DataSource = myDataTable;
            GridViewApplications.DataBind();
        }
```

Πλαίσιο Κειμένου 69: Οι εντολές για την εμφάνηση της λίστας των αιτήσεων εκτελούνται κατά την αρχικοποίηση της φόρμας από την συνάρτηση "Page_Load"

Ο χρήστης επιλέγει μία αίτηση για επεξεργασία από την λίστα πατώντας το πλήκτρο «Select (επιλογή)» που βρίσκεται στα αριστερά κάθε εγγραφής. Τα στοιχεία του χρήστη και της εργασίας παρουσιάζονται αναλυτικά και ξεκάθαρα στα πλαίσια που υπάρχουν κάτω από την λίστα των αιτήσεων προκειμένου ο χρήστης να συγκρίνει το τρέχον πρόγραμμα με τις αλλαγές που προτείνει ο χρήστη και να αποφασίσει στη συνέχεια αν θα κάνει δεκτή ή θα απορρίψει την αίτηση του χρήστη. Τα στοιχεία αυτά αντλούνται από την ΚΒΔ.

Ορισμένα από τα στοιχεία που υπάρχουν στην λίστα αιτήσεων εμφανίζονται και στα πλαίσια αναλυτικής περιγραφής των αιτήσεων. Η εμφάνιση αυτή οφείλεται στη μεταφορά των στοιχείων αυτών από τον πίνακα των αιτήσεων στα αντίστοιχα πλαίσια. Τα στοιχεία αυτά είναι: ο Τύπος Αίτησης, Κωδικός Προγράμματος, Κωδικός Χρήστη, Τύπος Εργασίας, Τοποθεσία, Όνομα Προγράμματος. Έχει ενδιαφέρον να εστιάσουμε στον τρόπο με τον οποία τα παραπάνω δεδομένα εμφανίζονται στα αντίστοιχα πλαίσια.

Επιλέγοντας για επεξεργασία μία από τις αιτήσεις που εμφανίζονται, αποθηκεύουμε την συγκεκριμένη εγγραφή (σειρά του πίνακα) σε μία μεταβλητή:

```
GridViewRow row = GridViewApplications.SelectedRow;
```

Στη συνέχεια ενθέτουμε τα δεδομένα που θέλουμε και τα οποία αντιστοιχούν στα στοιχεία (κελιά) της εγγραφής στα κατάλληλα πλαίσια «textbox».

```
protected void GridViewApplications_SelectedIndexChanged(object sender, EventArgs e)
        {
            GridViewRow row = GridViewApplications.SelectedRow;

            TextBoxApp_Type.Text=row.Cells[1].Text;
            TextBoxProgID.Text=row.Cells[4].Text;
            TextBoxUserID.Text=row.Cells[11].Text;
              …
```

Πλαίσιο Κειμένου 70: Εντολές για την άντληση δεδομένων από την επιλεγμένη εγγραφή του πίνακα των αιτήσεων

Ο χρήστης της Κεντρικής Εφαρμογής αποφασίζει μετά από τη μελέτη των αλλαγών που προτείνει ο αποστολέας της αίτησης εάν θα κάνει δεκτή την αίτηση αλλαγής του ή εάν το πρόγραμμα εφημεριών/καθηκόντων θα παραμείνει ως έχει. Στην περίπτωση που ο χρήστης της Κεντρικής Εφαρμογής αποδεχτεί την αίτηση που του υποβλήθηκε, τότε θα πρέπει να πατήσει το πλήκτρο «Αποδοχή Αίτησης». Εάν η αίτηση του χρήστη αφορά την ολική διαγραφή της εργασίας, τότε εκτελούνται οι κατάλληλες εντολές που διαγράφουν από τον πίνακα των προγραμμάτων (Program) της ΚΒΔ την συγκεκριμένη εργασία.

```
if (TextBoxApp_Type.Text == "Ακύρωση")
            {
               string cmd = "Delete * From Program Where Program_ID='@Program_ID'";
               System.Data.SqlClient.SqlCommand SqlCmd = new SqlCommand(cmd, conn);
                SqlCmd.Parameters.AddWithValue("@User_Team", TextBoxProgID.Text);
                TextBoxResult.Text = "Το αίτημά σας καταχωρήθηκαι";
            }
```

Πλαίσιο Κειμένου 71: Εάν γίνει δεκτή η αίτηση διαγραφής τότε το αντίστοιχο πρόγραμμα διαγράφεται από την ΚΒΔ

Ο χρήστης της φορητής εφαρμογής που είχε υποβάλει την αίτηση ενημερώνεται σχετικά με την έκβασή της από την ανανέωση του προγράμματος. Στο νέο πρόγραμμα δεν θα συμπεριλαμβάνεται η εργασία που διαγράφηκε.

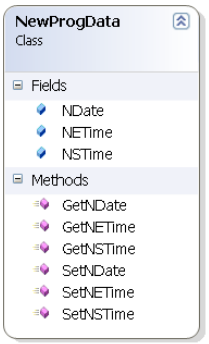
Εάν η αίτηση αφορά την αλλαγή της εργασίας η οποία περιλαμβάνει αλλαγή ώρας ή/και ημερομηνίας, τότε με το πάτημα του πλήκτρου «Αποδοχή Αίτησης» ενεργοποιούνται οι εντολές που προκαλούν την ανανέωση του τρέχοντος προγράμματος με τα χαρακτηριστικά εκείνα που πρότεινε ο χρήστης της φορητής εφαρμογής κατά την υποβολή της αίτησης αλλαγής της εργασίας του. Τα στοιχεία αυτά υπάρχουν ήδη αποθηκευμένα στην ΚΒΔ στον πίνακα «Modifications» και είναι ήδη στην διάθεσή μας αφού έχει πραγματοποιηθεί η άντληση τους κατά την δημιουργία του πίνακα «Προτεινόμενες Αλλαγές» που εμφανίζεται στην οθόνη του χρήστη της Κεντρικής Εφαρμογής.

```
string cmd2 = "UPDATE Program Set
Date='@Date',Duty_Start_Time='@Duty_Start_Time',Duty_End_Time='@Duty_End_Time'
where Program_ID='@Program_ID'";
System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2, conn);
SqlCmd2.Parameters.AddWithValue("@Date", NPD.GetNDate());
SqlCmd2.Parameters.AddWithValue("@Duty_Start_Time", NPD.GetNSTime());
SqlCmd2.Parameters.AddWithValue("@Duty_End_Time", NPD.GetNETime());
SqlCmd2.ExecuteNonQuery();
```

Πλαίσιο Κειμένου 72:Οι εντολές που πραγματοποιούν την ανανέωση του πίνακα "Program" αντλούν τα δεδομένα από την κλάση "NewProgData"

Για την περαιτέρω διευκόλυνση κατά την χρήση των στοιχείων ανανέωσης (νέα ημερομηνία, νέα ώρα έναρξης και λήξης) δημιουργήσαμε και την κλάση «NewProgData»

στην οποία αποθηκεύουμε τα δεδομένα αυτά προκειμένου να προβούμε σε ομαδική ανανέωση των στοιχείων της ΚΒΔ.



Εικόνα 50: Η κλάση "NewProgData" περιέχει τα στοιχεία που πρότεινε ο χρήστης κατά την υποβολή της αίτησης αλλαγής και είναι αυτά με τα οποία θα ανανεωθεί ο πίνακας της ΚΒΔ "Program"

Ο χρήστης της κεντρικής εφαρμογής ενημερώνεται για τις ενέργειες του συστήματος από ένα μήνυμα που εμφανίζεται στην οθόνη του κατά την επιτυχή ολοκλήρωση των διαδικασιών. Το μήνυμα αυτό αναφέρει «Το αίτημα σας καταχωρήθηκε».

5.3.5 Υλοποίηση της φόρμας «EditUsers»

Όπως αναφέραμε και στο προηγούμενο κεφάλαιο, η συγκεκριμένη φόρμα είναι υπεύθυνη για την επεξεργασία ή/και την παρουσίαση των στοιχείων ενός χρήστη. Προκειμένου να γίνει η διαχείριση των στοιχείων αυτών πρέπει να προηγηθεί μια αναζήτηση στον πίνακα των χρηστών βάσει ορισμένων χαρακτηριστικών προκειμένου να μας οδηγήσει στα άτομα ή στο άτομο για το οποίο ενδιαφερόμαστε να επεξεργαστούμε τα στοιχεία του. Τα χαρακτηριστικά της αναζήτησης είναι : η ομάδα των εργαζομένων στην οποία ανήκει, το τμήμα του νοσοκομείο στο οποίο ανήκει, βαθμίδα και ειδικότητα που κατέχει. Τα αποτέλεσμα της αναζήτησης είναι ένας κατάλογος με τους εργαζομένους της νοσοκομειακής μονάδας που ανταποκρίνονται στα χαρακτηριστικά αναζήτησης που έχει υποβάλει ο χρήστης της κεντρικής εφαρμογής.

Η προγραμματιστική υλοποίηση της αναζήτησης βασίζεται στην διαμόρφωση ενός query string που αποστέλλεται στην Κεντρική Βάση Δεδομένων. Η διαμόρφωση του query string αρχίζει αποθηκεύοντας το query string που δεν περιέχει κανένα χαρακτηριστικό αναζήτησης, στην μεταβλητή «sSQL» [Πλαίσιο Κειμένου 73].

```
string sSQL = "select name+' '+surname AS Όνομα,Department AS Τμήμα,Thesi AS
Θέση,Eidikothta AS Ειδικότητα,Bathmida AS Βαθμίδα,User_ID from Users";
```

Πλαίσιο Κειμένου 73: Η αποθήκευση του query string στην μεταβλητή sSQL

Στη συνέχεια για κάθε χαρακτηριστικό αναζήτησης που πρόκειται να προστεθεί στο query string πραγματοποιείται ένας έλεγχος. Με τον συγκεκριμένο έλεγχο διαπιστώνουμε αν το χαρακτηριστικό που θα εισάγουμε είναι το μοναδικό που έχει εισαχθεί μέχρι στιγμής, ή εάν έχει προηγηθεί η εισαγωγή και άλλων χαρακτηριστικών. Ο λόγος για τον οποίο μας ενδιαφέρει αυτή η πληροφορία, αφορά την σύνταξη του query string που θα σταλεί στην ΚΒΔ. Εάν το χαρακτηριστικό αναζήτησης που πρόκειται να εισαχθεί είναι το πρώτο, τότε στο αρχικό query string που είναι αποθηκευμένο στην μεταβλητή sSQL θα προστεθεί η κατάληξη: `WHERE χαρακτηριστικο_x='επιλεγμένο_χαρακτηριστικό'` και στη συνέχεια θα αποθηκευθεί ως τρέχων query string στην μεταβλητή sSQL. Εάν στη συνέχεια δεν προστεθεί επιπλέον χαρακτηριστικό τότε το συγκεκριμένο query string θα είναι αυτό που θα εκτελεστεί. Εάν όμως το χαρακτηριστικό αναζήτησης δεν είναι το μοναδικό που έχει υποβληθεί, τότε στο query string, που πλέον περιέχει το αρχικό query συν την προσθήκη κάποιου χαρακτηριστικού επιπλέων (όπως είδαμε στις προηγούμενες σειρές της ίδιας παραγράφου), θα προστεθεί η κατάληξη: `AND χαρακτηριστικο_x = 'επιλεγμένο_χαρακτηριστικό'`.

Το στοιχείο που ελέγχουμε προκειμένου να διαπιστώσουμε εάν το χαρακτηριστικό είναι το μοναδικό ή όχι, είναι τα αντικείμενο «Check Box» που βρίσκονται στην φόρμα. Εάν ένα αντικείμενο «Check Box» είναι ενεργοποιημένο σημαίνει ότι δεν ενδιαφέρει τον χρήστη της Κεντρικής Εφαρμογής, ο εργαζόμενος τον οποίο αναζήτα να ανταποκρίνεται στο συγκεκριμένο χαρακτηριστικό. Με αποτέλεσμα το χαρακτηριστικό αυτό να μην προστίθεται στο query string. Έτσι με πιο απλά και κατανοητά λόγια, ο αλγόριθμος που ακολουθούν οι έλεγχοι είναι: εάν ένα τουλάχιστον από τα αντικείμενα «Check Box» είναι απενεργοποιημένα, τότε σημαίνει ότι έχει προστεθεί κάποιο χαρακτηριστικό αναζήτησης στο query string με αποτέλεσμα η διαμόρφωση του να είναι η προσθήκη της κατάληξης: `AND χαρακτηριστικο_x = 'επιλεγμένο_χαρακτηριστικό'`. Στην αντίθετη περίπτωση, δηλαδή εάν μέχρι στιγμής όλα τα αντικείμενα «Check Box» είναι ενεργοποιημένα τότε το χαρακτηριστικό που πρόκειται να προστεθεί θα είναι το πρώτο στη σειρά, με αποτέλεσμα την προσθήκη στο query string την κατάληξη: `WHERE χαρακτηριστικο_x = 'επιλεγμένο_χαρακτηριστικό'`. Η συνάρτηση που πραγματοποιεί την αναζήτηση και τους ελέγχους που προαναφέραμε είναι η: `ImageButtonCreation_Click` και θα την βρείτε στο παράρτημα με τον κώδικα της Κεντρικής Εφαρμογής.

Στη συνέχεια ο χρήστης της Κεντρικής εφαρμογής με την βοήθεια του πλήκτρου «Select» επιλέγει από τον κατάλογο των εργαζομένων, που είναι αποτέλεσμα της αναζήτησης που προηγήθηκε, τον εργαζόμενο για τον οποίο ενδιαφέρεται. Με τον τρόπο αυτό η

εφαρμογή στο παρασκήνιο αποκτά το «User_ID» του συγκεκριμένου χρήστη και ανασύρει από την βάση δεδομένων όσα στοιχεία εγγραφής τον αφορούν (στοιχεία ταυτοποίησης, στοιχεία διευθύνσεων και τηλεφώνων και δηλωμένες εργασίες και τοποθεσίες). Έπειτα ο χρήστης μπορεί να κάνει επισκόπηση των δεδομένων αυτών ή να προβεί και σε αλλαγές των στοιχείων όπως διαγραφή υπαρχόντων ή προσθήκη νέων, με τον τρόπο που έχουμε ήδη παρουσιάσει στην φόρμα «UserManagement». Μπορείτε να βρείτε ολόκληρο τον κώδικα της φόρμας στο παράρτημα με τον κώδικα της Κεντρικής Εφαρμογής.

6

Επίλογος

6.1 Συμπεράσματα

Αποτέλεσμα της πτυχιακής εργασίας είναι η δημιουργία του συστήματος διαχείρισης εφημεριών/καθηκόντων. Τα συμπεράσματα που προκύπτουν από την ολοκλήρωση της εργασίας έχουν να κάνουν τόσο με τις μεθοδολογίες και τεχνολογίες που χρησιμοποιήσαμε όσο και με τον τομέα της οργάνωσης και της διαχείρισης προσωπικού των νοσοκομειακών μονάδων.

Η χρήση των υπηρεσιών XML Web Service είναι πολύ αποτελεσματική καθώς αποτελεί λύση σε προβλήματα διασύνδεσης μεταξύ εφαρμογών και συστημάτων. Επίσης οι υπηρεσίες αυτές προσφέρουν ευελιξία στους χρήστες τους καθώς είναι προσβάσιμες από οποιονδήποτε υπολογιστή με σύνδεση στο διαδίκτυο.

Από την έρευνα που πραγματοποιήθηκε στο χώρο των νοσοκομειακών μονάδων και τη μελέτη των αποτελεσμάτων της προκύπτει ότι η οργάνωση του προσωπικού των νοσοκομειακών μονάδων μπορεί να βελτιστοποιηθεί σε ποσοστό έως και 100% με τη δημιουργία αποτελεσματικών προγραμμάτων εργασίας και με αυστηρή τήρηση τους από το προσωπικό.

Το σύστημα μας ασχολείται με τη νευραλγικής σημασίας διαδικασία της διαχείρισης των προγραμμάτων βάζοντας τάξη και καταγράφοντας τις κινήσεις του προσωπικού σε ότι έχει να κάνει με την δημιουργία, την έκδοση, τη διαμοίραση, την αλλαγή και την

τροποποίηση των προγραμμάτων εφημεριών και γενικών καθηκόντων· βελτιώνοντας με τον τρόπο αυτό την οργάνωση του προσωπικού και συμβάλλοντας κατά συνέπεια στην ομαλοποίηση της λειτουργίας και την αύξηση της αποτελεσματικότητας των νοσοκομειακών μονάδων.

Η χρήση εφαρμογών για φορητές συσκευές στον χώρο των νοσοκομειακών μονάδων δεν αποτελεί πλέον καινοτομία από μόνη της. Καινοτομία αποτελεί η διαχείριση σε πραγματικό χρόνο των αλλαγών και των τροποποιήσεων που προκύπτουν στα προγράμματα των νοσοκομειακών μονάδων με τη βοήθεια φορητών συσκευών. Όλοι οι χρήστες του συστήματος ανεξαρτήτως του τομέα απασχόλησης είναι ενήμεροι ενώ ταυτόχρονα μπορούν οι ίδιοι να ενημερώνουν άμεσα μέσω της φορητής τους συσκευής για αλλαγές που επιθυμούν να γίνουν. Επίσης, έχουν οι ίδιοι τη δυνατότητα μερικής διαμόρφωσης του προγράμματος με την διαδικασία των ανταλλαγών.

6.2 Μελλοντικές επεκτάσεις

Το σύστημα μας έχει σχεδιαστεί εξ' αρχής προκειμένου να μπορεί υποστηρίξει μελλοντικές επεκτάσεις χωρίς την ανάγκη αρχιτεκτονικών και σχεδιαστικών τροποποιήσεων. Το γεγονός αυτό οφείλεται εν' μέρη στον πυρήνα των μεθοδολογιών και των τεχνολογιών που έχουν χρησιμοποιηθεί για την υλοποίηση του συστήματος, όπως: XML, HTML, C# και κατ' επέκταση στην σημασία που δόθηκε στον σχεδιασμό του συστήματος και ιδιαίτερα στις Βάσεις Δεδομένων και στο Web Service.

6.2.1 Το σύστημα διαχείρισης εφημεριών/καθηκόντων ως κομμάτι ενός ΟΠΣΥ

Το σύστημα μας πολύ εύκολα θα μπορούσε να ενοποιηθεί με ένα Ολοκληρωμένο Πληροφοριακό Σύστημα Υγείας. Η ευκολία της ενοποίησης εντοπίζεται όχι μόνο από τεχνολογικής άποψης άλλα και από άποψη λειτουργικότητας.

Το σύστημα διαχείρισης εφημεριών/καθηκόντων μπορεί να αποτελέσει αντικείμενο επέκτασης της λειτουργικότητας ενός ΟΠΣΥ. Πολλά από τα δεδομένα που χρησιμοποιούνται για την λειτουργία του συστήματος μας όπως: τα στοιχεία του προσωπικού της νοσοκομειακής μονάδας και τα προγράμματα εφημεριών/καθηκόντων είναι πολλές φορές κοινά με αυτά που βρίσκονται αποθηκευμένα στις Βάσεις Δεδομένων των ΟΠΣΥ.

Από τεχνολογικής άποψης, ένα σύστημα του οποίου οι συναλλαγές με το εξωτερικό περιβάλλον πραγματοποιούνται από ένα ήδη έτοιμο και ανεξάρτητο Web Service και από ανεξάρτητες φορητές εφαρμογές μέσω μηνυμάτων XML μπορεί πολύ γρήγορα και εύκολα να ενοποιηθεί με ένα άλλο, ανεξαρτήτως της δομής και της αρχιτεκτονικής του. Οποιεσδήποτε συναλλαγές μεταξύ των δύο συστημάτων μπορούν να πραγματοποιηθούν με ανταλλαγή

μηνυμάτων XML στα οποία μπορούν να πραγματοποιηθούν οι οποιεσδήποτε τροποποιήσεις των δεδομένων τους, αποφεύγοντας με αυτόν τον τρόπο οποιαδήποτε φυσική σύνδεση η οποία θα ήταν δυνατόν να απαιτεί παραμετροποίηση των δεδομένων συναλλαγής. Επίσης θα μπορούσαν να προστεθούν περιορισμοί ακεραιότητας όπως:

- Νόμοι του κράτους που διέπουν τις διατάξεις εργασιακών κανόνων και υπόλοιπων παραμέτρων που ισχύουν στις Νοσοκομειακές μονάδες [18].

- Άτυποι και τυπικοί κανόνες που ισχύουν σε μία Νοσοκομειακή Μονάδα μεταξύ των λειτουργικών τμημάτων εργασίας, καθώς επίσης και μεταξύ των εργαζομένων.

6.2.2 Το σύστημα διαχείρισης εφημεριών ως αντικείμενο επικοινωνίας του προσωπικού νοσοκομειακής μονάδας

Επέκταση του συστήματος διαχείρισης εφημεριών/καθηκόντων θα μπορούσαν να αποτελέσουν λειτουργίες και υπηρεσίες που εξυπηρετούν την επικοινωνία μεταξύ των χρηστών του συστήματος και κατά συνέπεια του προσωπικού της νοσοκομειακής μονάδας. Η συγκεκριμένη ιδέα αποκτά περισσότερο ενδιαφέρον αν σκεφτούμε ότι όλοι οι χρήστες του συστήματος αποτελούν μέλη μιας κοινότητας ανθρώπων με πολλά κοινά στοιχεία όπως χώρος και ωράρια εργασίας, προβλήματα και επιδιώξεις.

Ο τρόπος επικοινωνίας μεταξύ των χρηστών μπορεί να έχει τα χαρακτηριστικά των chat rooms με κατάλογο επαφών για κάθε χρήστη κ.α. Επίσης, μπορεί να έχει τα χαρακτηριστικά ανταλλαγής μηνυμάτων τα οποία θα ενδιαφέρουν συγκεκριμένες ομάδες χρηστών κάθε φορά, ή και τα δύο αυτά χαρακτηριστικά να συνυπάρχουν στην φορητή εφαρμογή.

Η ανταλλαγή μηνυμάτων σε πραγματικό χρόνο μεταξύ των εργαζομένων μίας νοσοκομειακής μονάδας θα μπορούσε να προσφέρει λύσεις σε αρκετά προβλήματα της οργάνωσης και της στελέχωσης των μονάδων. Επίσης η υπηρεσία ανταλλαγής μηνυμάτων μπορεί να χρησιμοποιηθεί και ως εργαλείο ειδοποίησης για έκτακτα περιστατικά ή συνθηκών που χρήζουν προσοχής.

6.2.3 Επέκταση του συστήματος για χρήση σε διάφορα είδη περιβάλλοντος εργασίας

Το γεγονός ότι το σύστημα μας είναι σχεδιασμένο για την διαχείριση εφημεριών/καθηκόντων νοσοκομειακής μονάδας οφείλεται στην εξειδίκευση του τμήματος

στον τομέα της «Πληροφορικής με Εφαρμογές στην Βιοϊατρική». Ένα αντίστοιχο σύστημα θα μπορούσε να διαχειρίζεται εφημερίες, καθήκοντα και άλλα δεδομένα που έχουν σχέση με την οργάνωση προσωπικού και προγραμμάτων εργασίας σε τομείς όπως η αστυνομία, η πυροσβεστική, ο στρατός, η εκπαίδευση κ.α.

Για την επέκταση τους συστήματος μας ώστε να καλύπτει ένα ευρύτερο φάσμα επαγγελμάτων και απαιτήσεων χρησιμοποιούμε την ίδια φιλοσοφία και τις ίδιες αρχές ανάλυσης και σχεδίασης με το παρόν σύστημα, της διαχείρισης εφημεριών/καθηκόντων νοσοκομειακής μονάδας. Οι αλλαγές που απαιτούνται για την πραγματοποίηση της συγκεκριμένης επέκτασης εντοπίζονται στα δεδομένα αποθήκευσης και στις απαιτήσεις των χρηστών.

7

Βιβλιογραφία

[1] Eyhab Al-Masri and Qusay H. Mahmoud , Discovering the Best Web Service, WWW 2007, May 8-12, Canada.

[2] Ioannis V. Papaioannou, Dimitrios T. Tsesmetzis, Ioanna G. Roussaki, Miltiades E. Anagnostou, A QoS Ontology Language for Web-Services, School of Electrical and Computer Engineering, National Technical University of Athens,Greece.

[3] Windows Embedded Developer Center, http://msdn2.microsoft.com/en-us/library/bb158532.aspx, Jule 2007

[4] Lionel M. Ni, Baijian Yang, Pei Zheng, Professional Microsoft Smartphone Programming, 2007 by Wiley Publishing

[5] Duoserve SheduFlow (desktop & online edition), Ιούλιος 2010, link: http://www.duoserve.com/1002/1106/Appointment-Scheduler.htm

[6] Time Tracker, version 5.1 , by Asgard Systems, Ιούλιος 2010, link: http://www.asgardsystems.com/

[7] Lytec 2010 & Lytec MD, DABBS Computer Consultant , LLC, Ιούλιος 2010, link: http://www.dabbsco.com/lytec_main.htm , http://www.dabbsco.com/lytec_md.htm

[8] CSC 's Appointment Scheduling, CSC, Ιούλιος 2010, link: http://www.cscpaperlessoffice.com/Appointment-Scheduling.htm

[9] Medical scheduling software, claricode, Ιούλιος 2010, link: http://www.claricode.com/software-development/medical-scheduling-software/

[10] PatientKeeper, Ιούλιος 2010, Link: http://www.claricode.com/software-development/medical-scheduling-software/

[11] http://msdn2.microsoft.com/en-us/library/zw4w595w(VS.80).aspx (.NET Framework)

[12] MedicMate, Ιούλιος 2010, link: http://www.medicmate.com/

[13] AppSoft, Ειδικές εφαρμογές για Δημόσιο-ΝΠΔΔ-ΔΕΚΟ, Διαχείριση προσωπικού ΝΠΔΔ, Αύγουστος 2010, link: http://www.appsoft.gr/index.php?option=com_content&view=article&id=117&Itemid=102

[14] Ειδική εφαρμογή Διαχείρισης Προσωπικού ΝΠΔΔ, Uniplan, Αύγουστος 2010, link: http://www.uniplan.gr/solutions/deko.htm

[15] Web Services: .NET Development, Web Services, Technical Articles, XML Web Services Basics

[16] Διαφάνειες μαθήματος «Τεχνολογίες Διαδικτύου», Καθηγητής: Ευάγγελος Σακκόπουλος, ακαδημαϊκό έτος 2008-2009

[17] Η διαχείριση των συγκρούσεων στο Νοσηλευτικό και Ιατρικό Προσωπικό στα Δημόσια Παιδιατρικά Νοσοκομεία, Αλεξάνδρα Κοντογιάννη, Διπλωματική Εργασία τμήμα Νοσηλευτικής, Πανεπιστήμιο Αθηνών, Επιβλέπουσα Καθηγήτρια: Δάφνη Καϊτελίδου.

[18] Νόμος υπ' αριθ. 3754, Ρύθμιση όρων απασχόλησης ιατρών του ΕΣΥ σύμφωνα με το π.δ 76/25 και άλλες διατάξεις, www.mof-glk.gr

[19] Rational Unified Process, Best practices for Software Development Teams, Rational Software White Paper, Rev 11/01, link: http://www-01.ibm.com/software/rational/

[20] http://msdn2.microsoft.com/en-us/library/system.web.services.description.servicedescriptionimporter%28VS.80%29.aspx

[21] http://www.gotdotnet.com/Community/UserSamples/Details.aspx?SampleGuid=65A1D4EA-0F7A-41BD-8494-E916EBC4159C

[22] http://msdn2.microsoft.com/en-us/library/system.net.httpwebrequest.getresponse.aspx

[23] http://msdn2.microsoft.com/en-us/library/system.net.webrequest.create.aspx

[24] http://msdn2.microsoft.com/en-us/library/system.net.httpwebresponse.getresponsestream.aspx

[25] http://msdn2.microsoft.com/en-us/library/bb278114.aspx (connect cradle)

[26] http://www.businesslink.gov.uk/bdotg/action/detail?type=RESOURCES&itemId=1074298285

[27] http://www.ercim.org/publication/Ercim_News/enw54/siau.html

[28] http://en.wikipedia.org/wiki/Windows_Mobile

[29] http://authors.aspalliance.com/brettb/RandomNumbersInCSharp.asp

[30] http://www.geekpedia.com/Question75_How-to-select-a-random-value-from-an-array.html

[31] http://www.c-sharpcorner.com/UploadFile/dpatra/425/Default.aspx

[32] http://dotnetperls.com/datagridview-tips

[33] http://www.akadia.com/services/dotnet_databinding.html

[34] http://www.west-wind.com/presentations/dotnetwebservices/DotNetWebServices.asp

[35] http://msdn.microsoft.com/en-us/library/bb158486.aspx

[36] http://www.codeproject.com/KB/mobile/MOB4DEVS05.aspx

[37] http://support.microsoft.com/kb/310107

[38] http://www.dotnetfordevices.com/articles/50.html

[39] http://breathingtech.com/2009/testing-and-debugging-sql-database-on-windows-mobile-devices/

[40] http://www.databasejournal.com/features/mssql/article.php/3694086/Programming-SQL-Server-2005-Compact-Edition-with-ADO.htm

[41] http://msdn.microsoft.com/en-us/library/ms838168.aspx

[42] http://www.christec.co.nz/blog/archives/136

Παράρτημα Ι: Ερωτηματολόγια

Ερωτηματολόγιο Για το προσωπικό του τμήματος

1. Είστε ευχαριστημένος από τον τρόπο διαμόρφωσης, έκδοσης και ανακοίνωσης του προγράμματος εφημεριών-καθηκόντων; ☐ Ναι ☐ Όχι
   a. Εάν όχι, γιατί; _____
   _____
2. Έχετε να προτείνετε κάποιες αλλαγές στην παραπάνω διαδικασία;
   _____
   _____
3. Εάν διαφωνείτε με το πρόγραμμα (δεν σας βολεύει η ώρα, η μέρα κτλ) απευθύνεστε στους αρμόδιους για να προβούν σε αλλαγές; ☐ Ναι ☐ Όχι
   a. Έρχεστε σε επικοινωνία με συναδέλφους σας με σκοπό την ανταλλαγή καθηκόντων ή εφημεριών ; ☐ Ναι ☐ Όχι
4. Δημιουργούνται προβλήματα (διαφωνίες) εξαιτίας του προγράμματος των εφημεριών μεταξύ των μελών του προσωπικού (ιατροί, νοσηλευτές); ☐ Ναι ☐ Όχι
   a. Θέλετε να μας αναφέρετε κάποιο παράδειγμα;
   _____
   _____
5. Πιστεύεται πως η δυνατότητα ανανέωσης και ενημέρωσης του προγράμματος, σε πραγματικό χρόνο και από οπουδήποτε θα σας ωφελούσε (αύξηση αποδοτικότητας/παραγωγικότητας); ☐ Ναι ☐ Όχι
6. Χρησιμοποιείτε φορητές συσκευές (π.χ. iPhone, mp3 players, PDAs) ή άλλα gadgets στην καθημερινότητα σας; ☐ Ναι ☐ Όχι

ΠΑΝΕΠΙΣΤΗΜΙΟ ΣΤΕΡΕΑΣ ΕΛΛΑΔΑΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗΝ ΒΙΟΪΑΤΡΙΚΗ

Ερωτηματολόγιο Για Την Γραμματεία

1. Το πρόγραμμα εφημεριών-καθηκόντων διαφορφώνεται απο εσάς;
   _____
   _____
2. Πόσες ώρες την βδομάδα ασχολείστε (προσωπικά) με το πρόγραμμα εφημεριών-καθηκόντων;
   ☐
3. Χρησιμοποιείται κάποια εφαρμογή του υπολογιστή για την δημιουργία (εγγραφή και οργάνωση) του προγράμματος εφημεριών;
   a. Εάν ναι, ποιο πρόγραμμα είναι αυτό; _____
   b. Εάν όχι, θέλετε να μας προτείνεται κάποιο; _____
4. Η γραμματεία διαθέτει την δικαιοδοσία για την πραγματοποίηση των αλλαγών του προγράμματος εφημεριών-καθηκόντων; ☐ Ναι ☐ Όχι
5. Γνωρίζεται περίπου τον αριθμό των ατόμων συνολικά (γραμματεία + διοικητικοί) που ασχολούνται με την διαδικασία έκδοσης και ανακοίνωσης του προγράμματος; ☐
6. Το πρόγραμμα υποβάλλετε σε πολλές τροποποιήσεις μέχρι να πάρει την τελική του μορφή και την έγκριση; ☐ Ναι ☐ Όχι
7. Εάν κάποιος από το προσωπικό (ιατροί, νοσηλευτές) επιθυμεί αλλαγές στο πρόγραμμα, πρέπει να έρθει σε επικοινωνία με την γραμματεία ή με κάποιον άλλον αρμόδιο;
   a. Αν ναι, ποία είναι η διαδικασία; _____
   _____
   _____
   _____
   b. Αν πρέπει να αναφερθεί σε κάποιον αρμόδιο ποιος είναι αυτός; _____

Ερωτηματολόγιο Για υπευθύνους τμήματος

1. Πιο είναι το πόστο σας; _____
2. Το πρόγραμμα διαμορφώνεται από εσάς ή προτιμάται να ασχολούνται άλλοι (πχ γραμματεία) για την διαμόρφωση του;
   a. Εάν ναι, ασχολείστε προσωπικά από την αρχή μέχρι και την έγκριση του ή το αναθέτετε σε άλλους μετά την πρώτη έκδοση του; ☐ Ναι ☐ Όχι
   b. Εάν όχι, ποιός αναλαμβάνει την διαμόρφωσή του;
      _____
3. Πόσες ώρες την βδομάδα ασχολείστε με το πρόγραμμα εφημεριών-καθηκόντων; ☐
4. Γνωρίζετε-ενημερώνεστε εάν το πρόγραμμα δέχεται επιπλέον αλλαγές από μέχρι την τελική έκδοσή του;
   a. Εάν ναι, γνωρίζετε πόσες φορές γίνονται τροποποιήσεις; _____
   b. Εάν όχι, θα θέλατε να γνωρίζετε-ενημερώνεστε για την διαδικασία των αλλαγών; ☐ Ναι ☐ Όχι
5. Το πρόγραμμα παίρνει έγκριση από εσάς μόνο όταν έχει την τελική του μορφή ή και κατά την διαδικασία των τροποποιήσεων; ☐ Ναι ☐ Όχι
   a. Εάν όχι, γνωρίζετε ή θα θέλατε να γνωρίζετε από ποιόν; ☐ Ναι ☐ Όχι
6. Είστε ευπρόσδεκτος σε αλλαγές που ζητάνε τα μέλη του υπολοίπου προσωπικού(γιατροί,νοσηλευτές, κτλ); ☐ Ναι ☐ Όχι
   a. Εάν όχι, γιατί;
      _____
7. Γνωρίζεται πόση ώρα μεσολαβεί από την τελική έγκριση του προγράμματος μέχρι την ανακοίνωση του; Εάν ναι, πόσες; ☐
   a. Εάν όχι, ποιός νομίζετε ότι είναι ο λόγος; _____
8. Νομίζετε ότι θα σας διευκόλυνε εάν η παραπάνω διαδικασία πραγματοποιούταν με αυτοματοποιημένο τρόπο και ηλεκτρονικά (π.χ. το πρόγραμμα να εμφανίζεται αυτόματα στην οθόνη σας και η έγκριση να γίνεται με το πάτημα ενός κουμπιού); ☐ Ναι ☐ Όχι
9. Είστε ευχαριστημένος/η από την ισχύουσα διαδικασία; ☐ Ναι ☐ Όχι
10. Έχετε να προτείνετε κάποιες αλλαγές στην τρέχουσα διαδικασία έκδοσης των προγραμμάτων εφημεριών-καθηκόντων;
    _____
    _____

ΠΑΝΕΠΙΣΤΗΜΙΟ ΣΤΕΡΕΑΣ ΕΛΛΑΔΑΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗΝ ΒΙΟΪΑΤΡΙΚΗ

Παράρτημα ΙΙ: Κώδικας C#

Κώδικας φορητής εφαρμογής

Κώδικας Web Service

Κώδικας Κεντρικής Εφαρμογής

ΠΑΝΕΠΙΣΤΗΜΙΟ ΣΤΕΡΕΑΣ ΕΛΛΑΔΑΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΪΑΤΡΙΚΗ

Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων στις Νοσοκομειακές Μονάδες με χρήση Τεχνολογιών Διαδικτύου

ΠΑΠΑΔΗΜΗΤΡΙΟΥ ΧΡΗΣΤΟΣ (ΑΜ 90)

Πηγαίος Κώδικας Πτυχιακής Εργασίας

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ
Επιβλέπων
Ευάγγελος Σακκόπουλος
Επίκ. Καθηγητής Π.Δ. 407/80
Λαμία, Αύγουστος 2010

Περιεχόμενα

1 Κώδικας φορητής εφαρμογής C#

1.1 Φόρμα «TimeTableForm/Form1.cs»

```
using System;
using System.IO;
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{


    public partial class TimeTableForm : Form
    {
        //αρχικοποίηση κλάσεων
        CurrentQuery CurrentQr = new CurrentQuery();
        CurrentDataTable CurrentDT = new CurrentDataTable();
        CurrentStatus CurrentST = new CurrentStatus();
        LogNumber Logins = new LogNumber();

        public TimeTableForm()
        {
            InitializeComponent();


this.declared_DutiesTableAdapter1.Fill(this.timeTableDBDataSet1.Declared_Du
ties);

this.declared_LocationsTableAdapter1.Fill(this.timeTableDBDataSet1.Declared
_Locations);

this.programTableAdapter1.Fill(this.timeTableDBDataSet1.Program);

            int currentLog = Logins.GetLogNum();
```

```
            var formlog = new TemporaryForm();

            if (formlog.GiveLogNumber() == 0)
            {
                LogMeOut();

                string Initcmd = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program";

                formlog.SetRunningQuery(Initcmd);

                Logins.SetLogNum(currentLog++);

            }

        }



        private void TimeTableForm_Load(object sender, EventArgs e)
        {
            SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");

            try
            {
                SystemDataWebReference.SystemData TableContentWebService =
new TimeTable_V2.SystemDataWebReference.SystemData();


                DataTable ProgramDataTable = new DataTable();
                DataTable LocationsDataTable = new DataTable();
                DataTable DutiesDataTable = new DataTable();

                var formlog = new TemporaryForm();
                //MessageBox.Show(formlog.GiveID());
                ProgramDataTable =
TableContentWebService.ProgramContent(formlog.GiveID());
                LocationsDataTable =
TableContentWebService.Declared_Locations(formlog.GiveID());
                DutiesDataTable =
TableContentWebService.Declared_Duties(formlog.GiveID());

                connection.Open();

                string cmdDelPr = "Delete From Program";
                string cmdDelDD = "Delete From Declared_Duties";
                string cmdDelDL = "Delete From Declared_Locations";

                SqlCeCommand SqlcmdDelPr = new SqlCeCommand(cmdDelPr,
connection);
                SqlcmdDelPr.ExecuteNonQuery();
                SqlCeCommand SqlcmdDelDD = new SqlCeCommand(cmdDelDD,
connection);
                SqlcmdDelDD.ExecuteNonQuery();
                SqlCeCommand SqlcmdDelDL = new SqlCeCommand(cmdDelDL,
connection);
                SqlcmdDelDL.ExecuteNonQuery();

                DataView dv = new DataView();
```

```
                dv.Table = ProgramDataTable;

                Populate(ProgramDataTable, "Program");
                Populate(LocationsDataTable, "Declared_Locations");
                Populate(DutiesDataTable, "Declared_Duties");

                pictureBox5.Visible = true;
                pictureBox5.BringToFront();
                pictureBox1.SendToBack();
                pictureBox1.Visible = false;
                bool online=true;
                CurrentST.SetStatus(online);
            }

            catch (Exception ex)
            {
                //throw (ex);
                MessageBox.Show("Δεν έχετε πρόσβαση στο Ίντερνετ "+
                    "το Profil σας τήθεται αυτόματα σε εργασία χωρίς
Σύνδεση. "+
                    "Οι υπηρεσίες Αλλαγής και Αποδοχής καθηκόντων είναι
ΕΚΤΟΣ Λειτουργίας" );
                pictureBox5.Visible = false;
                pictureBox5.SendToBack();
                pictureBox1.BringToFront();
                pictureBox1.Visible = true;

            }

            var formlog2 = new TemporaryForm();

            string cmd = formlog2.GiveRunningQuery();

            DataSet dsProgramContent = new DataSet();
            ProgramDataGrid.DataSource =ProgramContent(cmd,
dsProgramContent);

        }


        public void Populate(DataTable TableContent, string tablename)
        {
            StringBuilder sql = new StringBuilder();
            sql.Append("insert into " + tablename + "(");

            StringBuilder fields = new StringBuilder();
            StringBuilder parameters = new StringBuilder();

            foreach (DataColumn col in TableContent.Columns)
            {
                fields.Append(col.ColumnName);
                parameters.Append("@" + col.ColumnName.ToLower());

                if (col.ColumnName !=
TableContent.Columns[TableContent.Columns.Count - 1].ColumnName)
                {
                    fields.Append(",");
                    parameters.Append(",");

                }
            }
```

```
            sql.Append(fields.ToString() + ")");
            sql.Append("values(");
            sql.Append(parameters.ToString() + ")");

            string tatalRows = TableContent.Rows.Count.ToString();

            SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
            foreach (DataRow row in TableContent.Rows)
            {
                if (connection.State == ConnectionState.Closed)
                {
                    connection.Open();
                }

                SqlCeCommand cmd = new SqlCeCommand(sql.ToString(),
connection);

                foreach (DataColumn col in TableContent.Columns)
                {
                    cmd.Parameters.AddWithValue("@" +
col.ColumnName.ToLower(), row[col.ColumnName]);
                }

                try
                {
                    cmd.ExecuteNonQuery();
                }
                catch (Exception ex)
                {
                    //throw ex;
                }

            }

        }

        public DataTable ProgramContent(string cmd, DataSet
dsProgramContent)
        {
            CurrentQr.SetCurrentQuerry(cmd);

            SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
            connection.Open();

            SqlCeDataAdapter daProgramContent = new SqlCeDataAdapter(cmd,
connection);
            daProgramContent.Fill(dsProgramContent,"Program");

            DataTable myDataTable = dsProgramContent.Tables[0];
            connection.Close();

            CurrentDT.SetCurrentDataTable(myDataTable);

            return myDataTable;
        }

        private void ProgramDataGrid_CurrentCellChanged(object sender,
EventArgs e)
```

```
        {

        }


        private void menuItemPublishChange_Click(object sender, EventArgs
e)
        {
            if (CurrentST.GetStatus())
            {
                int colNum = ProgramDataGrid.CurrentCell.ColumnNumber;
                int rowNum = ProgramDataGrid.CurrentCell.RowNumber;

                try
                {
                    object ProgramID = ProgramDataGrid[rowNum, 0];

                    string s = string.Format("Program_ID='{0}'",
ProgramID);

                    string cmd = "SELECT * FROM Program WHERE " + s;

                    SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
                    connection.Open();
                    SqlCeCommand SqlCmd = new SqlCeCommand(cmd,
connection);
                    SqlCeDataReader dr = SqlCmd.ExecuteReader();
                    dr.Read();
                    SqlCmd.ExecuteNonQuery();

                    var formlog = new Edit_Event();
                    formlog.SetChangeOptions(dr);

                    connection.Close();
                    formlog.ShowDialog();

                }

                catch (Exception ex)
                {
                    string currentquery = CurrentQr.GetCurrentQuery();
                    DataSet dsProgramContent = new DataSet();
                    ProgramDataGrid.DataSource =
ProgramContent(currentquery, dsProgramContent);
                    //this.Close();
                }

            }

            else
            {
                MessageBox.Show("Η υπηρεσία δεν είναι Διαθέσιμή. Δεν έχετε
πρόσβαση στο Δίκτυο");
            }


        }

        private void menuItemAcceptChange_Click(object sender, EventArgs e)
        {
```

```
            if (CurrentST.GetStatus())
            {
                var formlog = new ChangeList();
                formlog.ShowDialog();
            }
            else
            {
                MessageBox.Show("Η υπηρεσία δεν είναι Διαθέσιμή. Δεν έχετε
πρόσβαση στο Δίκτυο");
            }

        }

        private void menuItemLogIn_Click(object sender, EventArgs e)
        {
            var loginform = new LogInForm();
            loginform.ShowDialog();

            if (loginform.DialogResult == DialogResult.OK)
            {
                LogMeIn();
            }
        }

        private void LogMeIn()
        {
            menuItemLogOut.Enabled = true;
            menuItemLogIn.Enabled = false;
            ProgramDataGrid.Enabled = true;
            StatusBarPanel.Enabled = true;
        }

        private void menuItemLogOut_Click(object sender, EventArgs e)
        {
            LogMeOut();
        }

        private void LogMeOut()
        {
            menuItemLogOut.Enabled = false;
            menuItemLogIn.Enabled = true;
            ProgramDataGrid.Enabled = false;
            StatusBarPanel.Enabled = false;
            pictureBox5.Visible = false;
            pictureBox5.SendToBack();
            pictureBox1.BringToFront();
            pictureBox1.Visible = true;
        }

        private void pictureBoxSearchProg_Click(object sender, EventArgs e)
        {
            menuItem5_Click(sender, e);
        }

        private void menuItem5_Click(object sender, EventArgs e)
        {
            var formlog = new SearchProgram();
            formlog.ShowDialog();
        }

        private void pictureBoxEditView_Click(object sender, EventArgs e)
```

```
        {
            menuItemEditView_Click(sender, e);
        }
        private void menuItemEditView_Click(object sender, EventArgs e)
        {
            var formlog = new ColumnSelection();
            formlog.ShowDialog();
        }

        private void pictureBoxPublishChange_Click(object sender, EventArgs
e)
        {
            menuItemPublishChange_Click(sender, e);
        }
        private void pictureBoxAcceptChange_Click(object sender, EventArgs
e)
        {
            menuItemAcceptChange_Click(sender, e);
        }

        private void pictureBoxChangesPreview_Click(object sender,
EventArgs e)
        {
            menuItemChangesPreview_Click(sender, e);
        }
        private void menuItemChangesPreview_Click(object sender, EventArgs
e)
        {
            var formlog = new PreviewChanges();
            formlog.ShowDialog();
        }

        private void menuItemRefresh_Click(object sender, EventArgs e)
        {
            //this.ShowDialog();
            var TempForm = new TemporaryForm();
            string Initcmd = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program";
            TempForm.SetRunningQuery(Initcmd);
            TimeTableForm_Load(sender, e);
        }

        private void pictureBoxRefresh_Click(object sender, EventArgs e)
        {
            menuItemRefresh_Click(sender,e);
        }

        private void menuItemInit_Click(object sender, EventArgs e)
        {
            menuItemRefresh_Click(sender, e);
        }


    }

    public class CurrentDataTable
    {
        public DataTable CurrentDT = new DataTable();

        public DataTable GetCurrentDataTable()
```

```
    {
        return CurrentDT;
    }

    public void SetCurrentDataTable(DataTable CurrentDTable)
    {
        CurrentDT = CurrentDTable;
    }
}

public class CurrentQuery
{

    public string CurrentQueryString = "";
    public string GetCurrentQuery()
    {
        return CurrentQueryString;
    }

    public void SetCurrentQuerry(string currentQuery)
    {
        CurrentQueryString = currentQuery;
    }
}

public class CurrentStatus
{
    bool is_online = false;

    public bool GetStatus()
    {
        return is_online;
    }

    public void SetStatus(bool status)
    {
        is_online = status;
    }

}

public class UserData
{
    string username;
    string password;

    public string GetUserName()
    {
        return username;
    }

    public string GetPassword()
    {
        return password;
    }

    public void SetUserData(string UserN, string Pswrd)
    {
        username = UserN;
        password = Pswrd;
```

```
        }
    }

    public class LogNumber
    {
        int LN=0;

        public int GetLogNum()
        {
            return LN;
        }

        public void SetLogNum(int Logs)
        {
            LN = Logs;
        }
    }
}
```

1.2 Φόρμα «ChangeList.cs»

```
using System;
using System.IO;
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{
    public partial class ChangeList : Form
    {
        public ChangeList()
        {
            InitializeComponent();
        }

        private void ChangeList_Load(object sender, EventArgs e)
        {

            SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();

            DataSet ChangeListDataSet = new DataSet("UsersChangeList");
            ChangeListDataSet = TableContentWebService.UsersChangeList();

            ChangeListDataGrid.DataSource =
ChangeListDataSet.Tables["UsersChangeList"];
        }

        private void Accept_Click(object sender, EventArgs e)
```

```
        {
            try
            {
                int colNum = ChangeListDataGrid.CurrentCell.ColumnNumber;
                int rowNum = ChangeListDataGrid.CurrentCell.RowNumber;


                object Program_ID = ChangeListDataGrid[rowNum, 1];
                object Date = ChangeListDataGrid[rowNum, 2];
                object Duty_Type = ChangeListDataGrid[rowNum, 3];
                object Duty_Start_Time = ChangeListDataGrid[rowNum, 4];
                object Duty_End_Time = ChangeListDataGrid[rowNum, 5];
                object Location = ChangeListDataGrid[rowNum, 6];
                object Program_Name = ChangeListDataGrid[rowNum, 7];
                object User_ID = ChangeListDataGrid[rowNum, 8];


                SystemDataWebReference.SystemData TableContentWebService =
new TimeTable_V2.SystemDataWebReference.SystemData();

                string cmd = "DELETE FROM ChangeList WHERE
Program_ID=@Program_ID";
                SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
                connection.Open();
                SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
                SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
                SqlCmd.ExecuteNonQuery();

                int ProgramID = Convert.ToInt32(Program_ID.ToString(), 10);

                var formlog = new TemporaryForm();
                string UID = formlog.GiveID();

                TableContentWebService.UpdateChanges(UID, ProgramID);

                var formlog2 = new TimeTableForm();
                formlog2.ShowDialog();
                this.DialogResult = DialogResult.OK;

                this.Close();
            }
            catch
            {
                MessageBox.Show("Δεν έχετε επιλέξει αντικείμενο για
Αποδοχή");
            }
        }

        private void menuItemCancel_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
        }
    }
}
```

1.3 Φόρμα «LogInForm.cs»

```
using System;
using System.IO;
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{
    public partial class LogInForm : Form
    {
        public LogInForm()
        {
            InitializeComponent();
        }

        private void menuItemLogInForm_Click(object sender, EventArgs e)
        {
            // αν ο χρήστης είναι συνδεδεμένος στο σύστημα εκτελούνται οι
εντολες στο block



            try
            {

                SystemDataWebReference.SystemData UserCheck = new
TimeTable_V2.SystemDataWebReference.SystemData();

                string UCheck =
UserCheck.UserDataCheck(textBoxUserName.Text.ToString(),
textBoxPassword.Text.ToString());

                if (UCheck == "ok")
                {
                    string usrnm = textBoxUserName.Text.ToString();
                    string pswd = textBoxPassword.Text.ToString();

                    SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
                    connection.Open();

                    //έλεγχος για διπλο-εγγραφή χρήστη
                    try
                    {
                        string cmd1 = string.Format("SELECT * FROM UserData
WHERE Username='{0}' AND Password='{1}'", textBoxUserName.Text.ToString(),
textBoxPassword.Text.ToString());
                        SqlCeCommand SqlCmd1 = new SqlCeCommand(cmd1,
connection);
```

```
                        SqlCeDataReader dr = SqlCmd1.ExecuteReader();
                        dr.Read();
                        string UserID=dr["User_ID"].ToString();

                        var formlog = new TemporaryForm();
                        formlog.SetID(UserID);
                        formlog.SetLogNumber(1);
                    }
                    catch (Exception ex)
                    {

                        string UserInfo =
UserCheck.UserInfo(textBoxUserName.Text.ToString(),
textBoxPassword.Text.ToString());

                        string cmd2 ="INSERT INTO UserData
(User_ID,username,password) values (@User_ID,@username,@password)";
                        SqlCeCommand SqlCmd2 = new SqlCeCommand(cmd2,
connection);
                        SqlCmd2.Parameters.AddWithValue("@User_ID",
UserInfo);
                        SqlCmd2.Parameters.AddWithValue("@username",
textBoxUserName.Text.ToString());
                        SqlCmd2.Parameters.AddWithValue("@password",
textBoxPassword.Text.ToString());
                        SqlCmd2.ExecuteNonQuery();

                        var formlog = new TemporaryForm();
                        formlog.SetID(UserInfo);
                        formlog.SetLogNumber(1);

                    }
                    MessageBox.Show("Εχετε συνδεθεί επιτυχώς");
                    var formlog2 = new TimeTableForm();
                    formlog2.ShowDialog();
                    this.DialogResult = DialogResult.OK;
                }

                else
                {
                    MessageBox.Show("Δέν έχετε εισάγει σωστά στοιχεία
σύνδεσης");
                }
            }

            // εαν ο χρήστης δεν είναι συνδεδεμένος με το σύστημα
εκτελούνται οι ενοτλές του
            // block
            catch (Exception ex)
            {
                //εκτελούνται οι εντολές του block μόνο αν ο χρήστης είναι
ίδη εγγεγραμμένος
                try
                {
                    SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
                    connection.Open();
                    string cmd1 = string.Format("SELECT * FROM UserData
WHERE Username='{0}' AND Password='{1}'", textBoxUserName.Text.ToString(),
textBoxPassword.Text.ToString());
```

```
                    SqlCeCommand SqlCmd1 = new SqlCeCommand(cmd1,
connection);
                    SqlCmd1.ExecuteNonQuery();

                    MessageBox.Show("Έχετε συνδεθεί επιτυχώς");
                    var formlog = new TimeTableForm();
                    formlog.ShowDialog();
                    this.DialogResult = DialogResult.OK;
                }

                catch
                {
                    MessageBox.Show("Δέν έχετε εισάγει σωστά στοιχεία
σύνδεσης"
                        + "ή εισέρχεται νέος χρήστης για πρώτη φορά"+
                        "χωρίς η εφαρμογή να συνδέεται με το σύστημα
ενημέρωσης εφημεριών");
                }
            }
        }

        private void menuItemCancelLogInForm_Click(object sender, EventArgs
e)
        {
            this.DialogResult = DialogResult.Cancel;
            this.Close();
        }

    }
}
```

1.4 Φόρμα «SearchProgram.cs»

```
using System;
using System.IO;
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{
    public partial class SearchProgram : Form
    {
        public SearchProgram()
        {
            InitializeComponent();
        }
```

```
        private void SearchProgram_Load(object sender, EventArgs e)
        {
            InitializeElements();
        }

        public void InitializeElements()
        {
            DataSet dsDeclareDutiesList = new DataSet();
            DataSet dsDeclareLocationsList = new DataSet();
            DataSet dsProgramNameList=new DataSet();

            string cmdDutiesList = "SELECT Duty_Types FROM
Declared_Duties";
            string cmdLocationsList = "SELECT Location FROM
Declared_Locations";
            string cmdProgramNameList = "SELECT Program_Name FROM Program";

            SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
            connection.Open();

            SqlCeDataAdapter daDeclareDutiesList = new
SqlCeDataAdapter(cmdDutiesList,connection);
            SqlCeDataAdapter daDeclareLocationsList = new
SqlCeDataAdapter(cmdLocationsList, connection);
            SqlCeDataAdapter daProgramNameList = new
SqlCeDataAdapter(cmdProgramNameList, connection);

            daDeclareDutiesList.Fill(dsDeclareDutiesList,
"Declared_Duties");
            daDeclareLocationsList.Fill(dsDeclareLocationsList,
"Declared_Locations");
            daProgramNameList.Fill(dsProgramNameList, "Program");

            connection.Close();

            comboBoxDutyTypes.DataSource =
dsDeclareDutiesList.Tables[0].DefaultView;
            comboBoxDutyTypes.DisplayMember = "Duty_Types";
            comboBoxLocations.DataSource =
dsDeclareLocationsList.Tables[0].DefaultView;
            comboBoxLocations.DisplayMember = "Location";
            comboBoxProgramName.DataSource =
dsProgramNameList.Tables[0].DefaultView;
            comboBoxProgramName.DisplayMember = "Program_Name";

        }

        private void OpenNewProgItem_Click(object sender, EventArgs e)
        {
            var Form1Log = new TimeTableForm();
            var TempForm = new TemporaryForm();
            string Initcmd = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program";
            TempForm.SetRunningQuery(Initcmd);


            string newquery = TempForm.GiveRunningQuery();

            dateTimePicker1.Format = DateTimePickerFormat.Custom;
```

```
            dateTimePicker1.CustomFormat = "dd/mm/yyyy";

            if (CheckboxTodayAndOn.Checked == true)
            {
                dateTimePicker1.Enabled = false;
                //CheckBoxDate = true;

            }
            else
            {
                newquery = TempForm.GiveRunningQuery();
                newquery = string.Format("{0} where Date='{1}'", newquery,
textBoxDate.Text);

                TempForm.SetRunningQuery(newquery);
            }

            if (checkBoxWEDuty.Checked == true)
            {

                comboBoxDutyTypes.Enabled = false;
                //CheckBoxDuty = true;
            }
            else
            {
                if (CheckboxTodayAndOn.Checked == true)
                {
                    newquery = TempForm.GiveRunningQuery();
                    newquery = string.Format("{0} AND Duty_Type=N'{1}'",
newquery, comboBoxDutyTypes.Text);
                    TempForm.SetRunningQuery(newquery);
                }
                else
                {
                    newquery = TempForm.GiveRunningQuery();
                    newquery = string.Format("{0} WHERE Duty_Type=N'{1}'",
newquery, comboBoxDutyTypes.Text);
                    TempForm.SetRunningQuery(newquery);
                }

            }

            if (checkBoxWELoc.Checked == true)
            {
                comboBoxLocations.Enabled = false;
                //CheckBoxLocation = true;
            }
            else
            {
                if ((CheckboxTodayAndOn.Checked == true) &&
(checkBoxWEDuty.Checked == true))
                {
                    newquery = TempForm.GiveRunningQuery();
                    newquery = string.Format("{0} WHERE Location=N'{1}'",
newquery, comboBoxLocations.Text);
                    TempForm.SetRunningQuery(newquery);
                }
                else
                {
                    newquery = TempForm.GiveRunningQuery();
```

```
                    newquery = string.Format("{0} AND Location=N'{1}'",
newquery, comboBoxLocations.Text);
                    TempForm.SetRunningQuery(newquery);
                }

            }

            if (checkBoxPN.Checked == true)
            {
                comboBoxProgramName.Enabled = false;
                //CheckBoxProgName = true;
            }
            else
            {
                if ((CheckboxTodayAndOn.Checked == true) &&
(checkBoxWEDuty.Checked == true) && (checkBoxWELoc.Checked == true))
                {
                    newquery = TempForm.GiveRunningQuery();
                    newquery = string.Format("{0} WHERE
Program_Name=N'{1}'", newquery, comboBoxProgramName.Text);
                    TempForm.SetRunningQuery(newquery);
                }
                else
                {
                    newquery = TempForm.GiveRunningQuery();
                    newquery = string.Format("{0} AND Program_Name=N'{1}'",
newquery, comboBoxProgramName.Text);
                    TempForm.SetRunningQuery(newquery);
                }
            }

            this.DialogResult = DialogResult.OK;
            Form1Log.ShowDialog();
        }

        private void menuItemReturn_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
        }

        private void dateTimePicker1_ValueChanged(object sender, EventArgs
e)
        {
            textBoxDate.Text = dateTimePicker1.Value.ToString("d/M/yyyy");
        }

        private void CheckboxTodayAndOn_CheckStateChanged(object sender,
EventArgs e)
        {

        }
    }
}
```

1.5 Φόρμα «TemporaryForm.cs»

```
using System;
using System.IO;
using System.Net;
```

```
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{
    public partial class TemporaryForm : Form
    {

        public TemporaryForm()
        {
            InitializeComponent();
        }


        public static string UserIdData = "";


        public string GiveID()
        {
            return UserIdData;
        }

        public void SetID(string UserID)
        {

            UserIdData = UserID;
        }


        public static int LogNumber = 0;

        public int GiveLogNumber()
        {
            return LogNumber;
        }

        public void SetLogNumber(int LogNum)
        {

            LogNumber = LogNum;
        }

        public static string RunningQuery = "";

        public string GiveRunningQuery()
        {
            return RunningQuery;
        }

        public void SetRunningQuery(string query)
        {
```

```
            RunningQuery = query;
        }
    }
}
```

1.6 Φόρμα «PreviewChanges.cs»

```
using System;
using System.IO;
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{
    public partial class PreviewChanges : Form
    {
        public PreviewChanges()
        {
            InitializeComponent();
        }

        private void PreviewChanges_Load(object sender, EventArgs e)
        {
            SystemDataWebReference.SystemData ViewChanges = new
TimeTable_V2.SystemDataWebReference.SystemData();

            var formlog = new TemporaryForm();
            string UID = formlog.GiveID();
            DataTable VChanges = new DataTable();
            VChanges = ViewChanges.ViewChanges(UID);
            dataGridPreviewChanges.DataSource = VChanges;

            Populate(VChanges,"ViewChanges");


        }

        public void Populate(DataTable TableContent, string tablename)
        {
            StringBuilder sql = new StringBuilder();
            sql.Append("insert into " + tablename + "(");

            StringBuilder fields = new StringBuilder();
            StringBuilder parameters = new StringBuilder();

            foreach (DataColumn col in TableContent.Columns)
            {
```

```
                fields.Append(col.ColumnName);
                parameters.Append("@" + col.ColumnName.ToLower());

                if (col.ColumnName !=
TableContent.Columns[TableContent.Columns.Count - 1].ColumnName)
                {
                    fields.Append(",");
                    parameters.Append(",");

                }
            }

            sql.Append(fields.ToString() + ")");
            sql.Append("values(");
            sql.Append(parameters.ToString() + ")");

            //int rowCnt = 0;
            string tatalRows = TableContent.Rows.Count.ToString();


            SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
            foreach (DataRow row in TableContent.Rows)
            {
                if (connection.State == ConnectionState.Closed)
                {
                    connection.Open();
                }

                SqlCeCommand cmd = new SqlCeCommand(sql.ToString(),
connection);

                foreach (DataColumn col in TableContent.Columns)
                {
                    cmd.Parameters.AddWithValue("@" +
col.ColumnName.ToLower(), row[col.ColumnName]);
                }

                try
                {
                    cmd.ExecuteNonQuery();
                }
                catch (Exception ex)
                {
                    //throw ex;
                }


            }

        }
    }
}
```

1.7 Φόρμα «ColumnSelection.cs»

```
using System;
using System.IO;
```

```
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;


namespace TimeTable_V2
{
    public partial class ColumnSelection : Form
    {
        public ColumnSelection()
        {
            InitializeComponent();
        }

        private void ColumnSelection_Load(object sender, EventArgs e)
        {
            SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
            connection.Open();

            SqlCeCommand cmdDuty = new SqlCeCommand("SELECT Value FROM
CurrentState WHERE Element='Duty_Type'", connection);
            SqlCeCommand cmdLoc = new SqlCeCommand("SELECT Value FROM
CurrentState WHERE Element='Location'", connection);
            SqlCeCommand cmdPrN = new SqlCeCommand("SELECT Value FROM
CurrentState WHERE Element='Program_Name'", connection);

            SqlCeDataReader Dutydr = cmdDuty.ExecuteReader();
            SqlCeDataReader LoCdr = cmdLoc.ExecuteReader();
            SqlCeDataReader PrNdr = cmdPrN.ExecuteReader();

            Dutydr.Read();
            LoCdr.Read();
            PrNdr.Read();


            try
            {


                if (Dutydr["Value"].ToString() == "enabled")
                {
                    DutyTypeCheckBox.CheckState = CheckState.Checked;
                }
                else
                {
                    DutyTypeCheckBox.CheckState = CheckState.Unchecked;
                }

                if (LoCdr["Value"].ToString() == "enabled")
```

```
                {
                    LocationColcheckBox.CheckState = CheckState.Checked;
                }
                else
                {
                    LocationColcheckBox.CheckState = CheckState.Unchecked;
                }

                if (PrNdr["Value"].ToString() == "enabled")
                {
                    ProgNameColcheckBox.CheckState = CheckState.Checked;
                }
                else
                {
                    ProgNameColcheckBox.CheckState = CheckState.Unchecked;
                }
            }

            catch
            {
                string StcmdDuty = string.Format("INSERT INTO CurrentState
(Element,Value) values ('{0}','{1}')", "Duty_Type", "enabled");
                string StcmdLoc = string.Format("INSERT INTO CurrentState
(Element,Value) values ('{0}','{1}')", "Location", "enabled");
                string StcmdPrN = string.Format("INSERT INTO CurrentState
(Element,Value) values ('{0}','{1}')", "Program_Name", "enabled");


                SqlCeCommand InitcmdDuty = new SqlCeCommand(StcmdDuty,
connection);
                SqlCeCommand InitcmdLoc = new SqlCeCommand(StcmdLoc,
connection);
                SqlCeCommand InitcmdPrN = new SqlCeCommand(StcmdPrN,
connection);

                InitcmdDuty.ExecuteNonQuery();
                InitcmdLoc.ExecuteNonQuery();
                InitcmdPrN.ExecuteNonQuery();
            }

        }



        private void ApplyColChanges_Click(object sender, EventArgs e)
        {
            var Form1Log = new TimeTableForm();
            var TempForm = new TemporaryForm();
            string Initcmd ="SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program";
            TempForm.SetRunningQuery(Initcmd);
            string newquery = TempForm.GiveRunningQuery();

            SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
            connection.Open();

            // t t t
```

```
            if ((ProgNameColcheckBox.CheckState == CheckState.Checked) &&
(DutyTypeCheckBox.CheckState == CheckState.Checked) &&
(LocationColcheckBox.CheckState == CheckState.Checked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery ="SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program";
                TempForm.SetRunningQuery(newquery);

                ProgNameColcheckBox.CheckState = CheckState.Checked;
                DutyTypeCheckBox.CheckState = CheckState.Checked;
                LocationColcheckBox.CheckState = CheckState.Checked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='enabled' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }

            //t t f
            if ((ProgNameColcheckBox.CheckState == CheckState.Checked) &&
(DutyTypeCheckBox.CheckState == CheckState.Checked) &&
(LocationColcheckBox.CheckState == CheckState.Unchecked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Program_Name AS ΌνΠρογράμ FROM Program";
                TempForm.SetRunningQuery(newquery);

                ProgNameColcheckBox.CheckState = CheckState.Checked;
                DutyTypeCheckBox.CheckState = CheckState.Checked;
                LocationColcheckBox.CheckState = CheckState.Unchecked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='enabled' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }

            // t f t
            if ((ProgNameColcheckBox.CheckState == CheckState.Checked) &&
(DutyTypeCheckBox.CheckState == CheckState.Unchecked) &&
(LocationColcheckBox.CheckState == CheckState.Checked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS Λήξη,Location AS
Τοποθεσία,Program_Name AS ΌνΠρογράμ FROM Program";
```

```
                TempForm.SetRunningQuery(newquery);

                ProgNameColcheckBox.CheckState = CheckState.Checked;
                DutyTypeCheckBox.CheckState = CheckState.Unchecked;
                LocationColcheckBox.CheckState = CheckState.Checked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='unable' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }

            //t f f
            if ((ProgNameColcheckBox.CheckState == CheckState.Checked) &&
(DutyTypeCheckBox.CheckState == CheckState.Unchecked) &&
(LocationColcheckBox.CheckState == CheckState.Unchecked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS Λήξη,Program_Name AS
ΌνΠρογράμ FROM Program";
                TempForm.SetRunningQuery(newquery);

                ProgNameColcheckBox.CheckState = CheckState.Checked;
                DutyTypeCheckBox.CheckState = CheckState.Unchecked;
                LocationColcheckBox.CheckState = CheckState.Unchecked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='unable' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }

            //f t t
            if ((ProgNameColcheckBox.CheckState == CheckState.Unchecked) &&
(DutyTypeCheckBox.CheckState == CheckState.Checked) &&
(LocationColcheckBox.CheckState == CheckState.Checked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη,Location AS Τοποθεσία FROM Program";
                TempForm.SetRunningQuery(newquery);

                ProgNameColcheckBox.CheckState = CheckState.Unchecked;
                DutyTypeCheckBox.CheckState = CheckState.Checked;
                LocationColcheckBox.CheckState = CheckState.Checked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
```

```
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='enabled' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }

            //f t f
            if ((ProgNameColcheckBox.CheckState == CheckState.Unchecked) &&
(DutyTypeCheckBox.CheckState == CheckState.Checked) &&
(LocationColcheckBox.CheckState == CheckState.Unchecked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Type AS Εργασία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS
Λήξη FROM Program";
                TempForm.SetRunningQuery(newquery);

                ProgNameColcheckBox.CheckState = CheckState.Unchecked;
                DutyTypeCheckBox.CheckState = CheckState.Checked;
                LocationColcheckBox.CheckState = CheckState.Unchecked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='enabled' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }

            //f f t
            if ((ProgNameColcheckBox.CheckState == CheckState.Unchecked) &&
(DutyTypeCheckBox.CheckState == CheckState.Unchecked) &&
(LocationColcheckBox.CheckState == CheckState.Checked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS Λήξη,Location AS
Τοποθεσία FROM Program";
                TempForm.SetRunningQuery(newquery);


                ProgNameColcheckBox.CheckState = CheckState.Unchecked;
                DutyTypeCheckBox.CheckState = CheckState.Unchecked;
                LocationColcheckBox.CheckState = CheckState.Checked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='unable' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='enabled' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }
```

```
            // f f f
            if ((ProgNameColcheckBox.CheckState == CheckState.Unchecked) &&
(DutyTypeCheckBox.CheckState == CheckState.Unchecked) &&
(LocationColcheckBox.CheckState == CheckState.Unchecked))
            {

                newquery = TempForm.GiveRunningQuery();
                newquery = "SELECT Program_ID AS Κωδικός,Date AS
Ημερομηνία,Duty_Start_Time AS Έναρξη,Duty_End_Time AS Λήξη FROM Program";
                TempForm.SetRunningQuery(newquery);

                ProgNameColcheckBox.CheckState = CheckState.Unchecked;
                DutyTypeCheckBox.CheckState = CheckState.Unchecked;
                LocationColcheckBox.CheckState = CheckState.Unchecked;
                SqlCeCommand cmdPrN = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Program_Name'", connection);
                cmdPrN.ExecuteNonQuery();
                SqlCeCommand cmdDType = new SqlCeCommand("update
CurrentState set Value='unable' where Element='Duty_Type'", connection);
                cmdDType.ExecuteNonQuery();
                SqlCeCommand cmdLoc = new SqlCeCommand("update CurrentState
set Value='unable' where Element='Location'", connection);
                cmdLoc.ExecuteNonQuery();
            }


            this.DialogResult = DialogResult.OK;
            Form1Log.ShowDialog();
        }

        private void menuItemReturn_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
        }
    }
}
```

1.8 Φόρμα «Edit_Event.cs»

```
using System;
using System.IO;
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{
    public partial class Edit_Event : Form
    {
```

```
        CurrentDataReader CDR= new CurrentDataReader();

         public Edit_Event()
         {
             InitializeComponent();
         }

         private void Edit_Event_Load(object sender, EventArgs e)
         {
             buttonChangeHour2.Visible = false;
             ChangeStartTimeButton.Visible = false;
             buttonMoveDuty.Visible = false;
             buttonCancelDuty.Visible = false;
         }

         public void SetChangeOptions(SqlCeDataReader SpecificRow)
         {

CDR.SetCurrentDr(SpecificRow["User_ID"].ToString(),SpecificRow["Program_ID"
].ToString());

             ChangedDateTextBox.Text = SpecificRow["Date"].ToString();
             ChangedLocTextBox.Text = SpecificRow["Location"].ToString();
             ChangedDutyTextBox.Text = SpecificRow["Duty_Type"].ToString();
             ChangedSTtextBox.Text =
SpecificRow["Duty_Start_Time"].ToString();
             ChangedETtextBox.Text =
SpecificRow["Duty_End_Time"].ToString();
         }

         private void ShareDutyButton_Click(object sender, EventArgs e)
         {
             SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();

             string User_ID = CDR.GetUser();
             int Program_ID = CDR.GetProgram();
             string cmd = "INSERT INTO ChangeList (User_ID,Program_ID)
values (@User_ID,@Program_ID)";

             SqlCeConnection connection = new SqlCeConnection(@"Data
Source=\Program Files\TimeTable_V2\TimeTableDB.sdf");
             connection.Open();

             SqlCeCommand SqlCmd = new SqlCeCommand(cmd, connection);
             SqlCmd.Parameters.AddWithValue("@User_ID", User_ID);
             SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
             SqlCmd.ExecuteNonQuery();

             TableContentWebService.InsertChanges(User_ID, Program_ID);

             this.Close();
         }

         private void buttonCancelDuty_Click(object sender, EventArgs e)
         {
             SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();

             string User_ID = CDR.GetUser();
             int Program_ID = CDR.GetProgram();
```

```
            string App_Type = "Cancel";


TableContentWebService.InsertApplications(User_ID,Program_ID,App_Type);
            TableContentWebService.ProgramModification(User_ID, Program_ID,
"Cancel", "Cancel", "Cancel");

            this.Close();
        }

        private void buttonSendForChange_Click(object sender, EventArgs e)
        {
            ShareDutyButton.Visible = false;
            buttonChangeHour2.Visible = true;
            ChangeStartTimeButton.Visible = true;
            buttonMoveDuty.Visible = true;
            buttonCancelDuty.Visible = true;
            buttonSendForChange.Visible = false;
        }

        private void ChangeStartTimeButton_Click(object sender, EventArgs
e)
        {
            var formlog = new ChangeStartTimeRequestForm();
            string HourFlag = "hour";
            formlog.SetHours(ChangedSTtextBox.Text.ToString(),
ChangedETtextBox.Text.ToString(),HourFlag);
            formlog.SetDate(ChangedDateTextBox.Text.ToString(), HourFlag);
            formlog.IDs(CDR.GetUser(), CDR.GetProgram());

            formlog.ShowDialog();
            this.Close();
        }

        private void buttonChangeHour2_Click(object sender, EventArgs e)
        {
            ChangeStartTimeButton_Click(sender, e);
        }

        private void buttonMoveDuty_Click(object sender, EventArgs e)
        {
            var formlog = new ChangeStartTimeRequestForm();
            string FlagDate = "date";
            formlog.SetDate(ChangedDateTextBox.Text.ToString(), FlagDate);
            formlog.SetHours(ChangedSTtextBox.Text.ToString(),
ChangedETtextBox.Text.ToString(), FlagDate);
            formlog.IDs(CDR.GetUser(), CDR.GetProgram());

            formlog.ShowDialog();
            this.Close();
        }

        public string GiveID()
        {
            string id = CDR.GetUser();
            return id;
        }

        public int GiveProg()
        {
            int prog = CDR.GetProgram();
```

```
            return prog;
        }

        public string GiveDate()
        {
            string Date = ChangedDateTextBox.Text.ToString();
            return Date;
        }

        private void menuItemReturn_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;

        }

    }

    public class CurrentDataReader
    {
        public string User_ID;
        public int Program_ID;

        public string GetUser()
        {
            return User_ID;
        }
        public int GetProgram()
        {
            return Program_ID;
        }
        public void SetCurrentDr(string UserID, string ProgramID)
        {
            User_ID = UserID;
            Program_ID = Convert.ToInt32(ProgramID, 10);
        }
    }
}
```

1.9 Φόρμα «ChangeStartTimeRequestForm.cs»

```
using System;
using System.IO;
using System.Net;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.Web;
using System.Web.Services;
using System.Configuration;
using System.Collections;
using System.Data.SqlServerCe;

namespace TimeTable_V2
{
```

```
public partial class ChangeStartTimeRequestForm : Form
{
    ModificationData ModifyProgram = new ModificationData();

    public void IDs(string UID, int PID)
    {
        ModifyProgram.SetIDs(UID, PID);
    }

    public ChangeStartTimeRequestForm()
    {
        InitializeComponent();
    }

    public void SetHours(string StartHour, string EndHour,string Flag)
    {
        CurrentStartHourTextBox.Text = StartHour;
        ModifyProgram.SetSHour(StartHour);

        CurrentEndHourTextBox.Text = EndHour;
        ModifyProgram.SetEHour(EndHour);

        if (Flag == "hour")
        {
            panelChangeDate.Visible = false;
            panelSplit.Visible = true;
            panelSplit.BackColor = Color.White;
            buttonComboChange.Visible = true;
            buttonComboChange.BringToFront();
            buttonComboChange.Text = "Μεταφορά Εφημερίας";
        }
    }

    public void SetDate(string OldDate, string Flag)
    {
        textBoxOldDate.Text = OldDate;
        ModifyProgram.SetNDate(OldDate);

        if (Flag == "date")
        {
            panelChangeHour.Visible = false;
            panelChangeDate.Location=new Point(0,0);
            panelSplit.Visible = true;
            buttonComboChange.Visible = true;
            panelSplit.BackColor = Color.White;
            buttonComboChange.BringToFront();
            buttonComboChange.Text = "Αλλαγή Ώρας";
        }
    }

    private void buttonComboChange_Click(object sender, EventArgs e)
    {
        if (buttonComboChange.Text == "Μεταφορά Εφημερίας")
        {
            panelChangeDate.Visible = true;
            panelSplit.Visible = true;
            panelSplit.BackColor = Color.Gray;
            buttonComboChange.Visible = false;
            panelChangeDate.Visible = true;
        }
```

```
            if (buttonComboChange.Text == "Αλλαγή Ωρας")
            {
                panelChangeDate.Location = new Point(3, 159);
                panelChangeHour.Location = new Point(0, 0);
                panelSplit.Visible = true;
                panelSplit.BackColor = Color.Gray;
                panelSplit.BringToFront();
                buttonComboChange.Visible = false;
                panelChangeHour.Visible = true;
            }
        }



        private void menuItemSubmit_Click(object sender, EventArgs e)
        {
            string App_Type = "Αλλαγή/Μεταφορά";
            SystemDataWebReference.SystemData TableContentWebService = new
TimeTable_V2.SystemDataWebReference.SystemData();


TableContentWebService.ProgramModification(ModifyProgram.GetUID(),ModifyPro
gram.GetPID(), ModifyProgram.GetSHour(), ModifyProgram.GetEHour(),
ModifyProgram.GetNDate());

TableContentWebService.InsertApplications(ModifyProgram.GetUID(),ModifyProg
ram.GetPID(), App_Type);

            this.Close();
        }



        private void buttonSTLock_Click(object sender, EventArgs e)
        {
            string NewStaHour = textBoxNewStartHour.Text;
            string NewStaMin = textBoxNewStartMin.Text;

            if ((NewStaHour == "hh") && (NewStaMin == "mm"))
            {
                //εγγραφή νέων στοιχείων στην κλάση
                //ModificationData
                ModifyProgram.SetSHour(CurrentStartHourTextBox.Text);

                CurrentStartHourTextBox.Text =
CurrentStartHourTextBox.Text;
                CurrentStartHourTextBox.Enabled = false;
                textBoxNewStartHour.Enabled = false;
                textBoxNewStartMin.Enabled = false;

            }
            else
            {

                int NewStartHour = Convert.ToInt32(NewStaHour, 10);
                int NSM = Convert.ToInt32(NewStaMin, 10);
                string NStartHour = NewStaHour + ":" + NewStaMin;
                //έλεγχος ορθής εισαγωγής στοιχείων ώρας
                if ((NewStartHour > 24) || (NSM > 60))
                {
```

```
                    MessageBox.Show("Δεν έχετε εισάγει έγκυρα στοιχεία
ώρας");
                }
                else
                {
                    //εγγραφή νέων στοιχείων στην κλάση
                    //ModificationData
                    ModifyProgram.SetSHour(NStartHour);

                    CurrentStartHourTextBox.Text = NStartHour;
                    CurrentStartHourTextBox.Enabled = false;
                    textBoxNewStartHour.Enabled = false;
                    textBoxNewStartMin.Enabled = false;
                }
            }
        }

        private void buttonETLock_Click(object sender, EventArgs e)
        {
            string NewEndHour = textBoxNewEndHour.Text;
            string NewEndMin = textBoxNewEndMin.Text;

            if ((NewEndHour == "hh") && (NewEndMin == "mm"))
            {
                //εγγραφή νέων στοιχείων στην κλάση
                //ModificationData
                ModifyProgram.SetEHour(CurrentEndHourTextBox.Text);

                CurrentEndHourTextBox.Text = CurrentEndHourTextBox.Text;
                CurrentEndHourTextBox.Enabled = false;
                textBoxNewEndHour.Enabled = false;
                textBoxNewEndMin.Enabled = false;

            }
            else
            {

                int NEH = Convert.ToInt32(NewEndHour, 10);
                int NEM = Convert.ToInt32(NewEndMin, 10);
                string NEndHour = NewEndHour + ":" + NewEndMin;
                //έλεγχος ορθής εισαγωγής στοιχείων ώρας
                if ((NEH > 24) || (NEM > 60))
                {
                    MessageBox.Show("Δεν έχετε εισάγει έγκυρα στοιχεία
ώρας");
                }
                else
                {
                    //εγγραφή νέων στοιχείων στην κλάση
                    //ModificationData
                    ModifyProgram.SetEHour(NEndHour);

                    CurrentEndHourTextBox.Text = NEndHour;
                    CurrentEndHourTextBox.Enabled = false;
                    textBoxNewEndHour.Enabled = false;
                    textBoxNewEndMin.Enabled = false;
                }
            }
        }

        private void buttonDateLock_Click(object sender, EventArgs e)
```

```
        {
            DateTime DateQ = dateTimePicker1.Value;
            //ορισμός format αποθήκευσης της ημερομηνίας
            string NewDate = DateQ.ToString("dd/MM/yy");
            //εγγραφή νέων στοιχείων στην κλάση
            //ModificationData
            ModifyProgram.SetNDate(NewDate);

            textBoxOldDate.Text = NewDate;
            textBoxOldDate.Enabled = false;
            dateTimePicker1.Enabled = false;
        }


    }

    public class ModificationData
    {
        public string NewStartHour = "0";
        public string NewEndHour = "0";
        public string NewDate = "0";
        public string UserID = "0";
        public int ProgramID = 0;

        public void SetSHour(string SHour)
        {
            NewStartHour = SHour;
        }

        public void SetEHour(string EHour)
        {
            NewEndHour = EHour;
        }

        public void SetNDate(string NDate)
        {
            NewDate = NDate;
        }

        public void SetIDs(string User_ID, int Program_ID)
        {
            UserID = User_ID;
            ProgramID = Program_ID;
        }

        public string GetSHour()
        {
            return NewStartHour;
        }

        public string GetEHour()
        {
            return NewEndHour;
        }

        public string GetNDate()
        {
            return NewDate;
        }

        public string GetUID()
```

```
        {
            return UserID;
        }

        public int GetPID()
        {
            return ProgramID;
        }
    }

}
```

1.10 Reference.cs

```
namespace TimeTable_V2.SystemDataWebReference {
    using System.Diagnostics;
    using System.Web.Services;
    using System.ComponentModel;
    using System.Web.Services.Protocols;
    using System;
    using System.Xml.Serialization;
    using System.Data;


    /// <remarks/>
    [System.Diagnostics.DebuggerStepThroughAttribute()]
    [System.ComponentModel.DesignerCategoryAttribute("code")]
    [System.Web.Services.WebServiceBindingAttribute(Name="SystemDataSoap",
Namespace="http://tempuri.org/")]
    public partial class SystemData :
System.Web.Services.Protocols.SoapHttpClientProtocol {

        /// <remarks/>
        public SystemData() {
            this.Url = "http://192.168.1.6/mywebservices/SystemData.asmx";
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/ProgramContent", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public System.Data.DataTable ProgramContent(string UserId) {
            object[] results = this.Invoke("ProgramContent", new object[] {
                        UserId});
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginProgramContent(string UserId,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("ProgramContent", new object[] {
                        UserId}, callback, asyncState);
        }

        /// <remarks/>
        public System.Data.DataTable EndProgramContent(System.IAsyncResult
asyncResult) {
```

```
            object[] results = this.EndInvoke(asyncResult);
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/Declared_Locations", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public System.Data.DataTable Declared_Locations(string UserId) {
            object[] results = this.Invoke("Declared_Locations", new
object[] {
                        UserId});
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginDeclared_Locations(string UserId,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("Declared_Locations", new object[] {
                        UserId}, callback, asyncState);
        }

        /// <remarks/>
        public System.Data.DataTable
EndDeclared_Locations(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/Declared_Duties", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public System.Data.DataTable Declared_Duties(string UserId) {
            object[] results = this.Invoke("Declared_Duties", new object[]
{
                        UserId});
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginDeclared_Duties(string UserId,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("Declared_Duties", new object[] {
                        UserId}, callback, asyncState);
        }

        /// <remarks/>
        public System.Data.DataTable EndDeclared_Duties(System.IAsyncResult
asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>
```

```
[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/ViewChanges", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public System.Data.DataTable ViewChanges(string UserID) {
            object[] results = this.Invoke("ViewChanges", new object[] {
                        UserID});
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginViewChanges(string UserID,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("ViewChanges", new object[] {
                        UserID}, callback, asyncState);
        }

        /// <remarks/>
        public System.Data.DataTable EndViewChanges(System.IAsyncResult
asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/InsertChanges", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public void InsertChanges(string User, int Program) {
            this.Invoke("InsertChanges", new object[] {
                        User,
                        Program});
        }

        /// <remarks/>
        public System.IAsyncResult BeginInsertChanges(string User, int
Program, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("InsertChanges", new object[] {
                        User,
                        Program}, callback, asyncState);
        }

        /// <remarks/>
        public void EndInsertChanges(System.IAsyncResult asyncResult) {
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/UpdateChanges", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public void UpdateChanges(string User, int Program) {
            this.Invoke("UpdateChanges", new object[] {
```

```
                        User,
                        Program});
        }

        /// <remarks/>
        public System.IAsyncResult BeginUpdateChanges(string User, int
Program, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("UpdateChanges", new object[] {
                        User,
                        Program}, callback, asyncState);
        }

        /// <remarks/>
        public void EndUpdateChanges(System.IAsyncResult asyncResult) {
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/UsersChangeList", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public System.Data.DataSet UsersChangeList() {
            object[] results = this.Invoke("UsersChangeList", new
object[0]);
            return ((System.Data.DataSet)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult
BeginUsersChangeList(System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("UsersChangeList", new object[0],
callback, asyncState);
        }

        /// <remarks/>
        public System.Data.DataSet EndUsersChangeList(System.IAsyncResult
asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((System.Data.DataSet)(results[0]));
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/UserDataCheck", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public string UserDataCheck(string username, string password) {
            object[] results = this.Invoke("UserDataCheck", new object[] {
                        username,
                        password});
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginUserDataCheck(string username,
string password, System.AsyncCallback callback, object asyncState) {
```

```
            return this.BeginInvoke("UserDataCheck", new object[] {
                        username,
                        password}, callback, asyncState);
        }

        /// <remarks/>
        public string EndUserDataCheck(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((string)(results[0]));
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/InsertApplications", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public void InsertApplications(string User, int Program, string
Application) {
            this.Invoke("InsertApplications", new object[] {
                        User,
                        Program,
                        Application});
        }

        /// <remarks/>
        public System.IAsyncResult BeginInsertApplications(string User, int
Program, string Application, System.AsyncCallback callback, object
asyncState) {
            return this.BeginInvoke("InsertApplications", new object[] {
                        User,
                        Program,
                        Application}, callback, asyncState);
        }

        /// <remarks/>
        public void EndInsertApplications(System.IAsyncResult asyncResult)
{
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/ProgramModification", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public void ProgramModification(string User, int Program, string
StHour, string EnHour, string NeDate) {
            this.Invoke("ProgramModification", new object[] {
                        User,
                        Program,
                        StHour,
                        EnHour,
                        NeDate});
        }

        /// <remarks/>
```

```
        public System.IAsyncResult BeginProgramModification(string User,
int Program, string StHour, string EnHour, string NeDate,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("ProgramModification", new object[] {
                        User,
                        Program,
                        StHour,
                        EnHour,
                        NeDate}, callback, asyncState);
        }

        /// <remarks/>
        public void EndProgramModification(System.IAsyncResult asyncResult)
{
            this.EndInvoke(asyncResult);
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/UsersApplications", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public System.Data.DataTable UsersApplications() {
            object[] results = this.Invoke("UsersApplications", new
object[0]);
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult
BeginUsersApplications(System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("UsersApplications", new object[0],
callback, asyncState);
        }

        /// <remarks/>
        public System.Data.DataTable
EndUsersApplications(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/UserInfo", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public string UserInfo(string username, string password) {
            object[] results = this.Invoke("UserInfo", new object[] {
                        username,
                        password});
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginUserInfo(string username, string
password, System.AsyncCallback callback, object asyncState) {
```

```
            return this.BeginInvoke("UserInfo", new object[] {
                        username,
                        password}, callback, asyncState);
        }

        /// <remarks/>
        public string EndUserInfo(System.IAsyncResult asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((string)(results[0]));
        }

        /// <remarks/>

[System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.
org/SelectedField", RequestNamespace="http://tempuri.org/",
ResponseNamespace="http://tempuri.org/",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        public System.Data.DataTable SelectedField() {
            object[] results = this.Invoke("SelectedField", new object[0]);
            return ((System.Data.DataTable)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginSelectedField(System.AsyncCallback
callback, object asyncState) {
            return this.BeginInvoke("SelectedField", new object[0],
callback, asyncState);
        }

        /// <remarks/>
        public System.Data.DataTable EndSelectedField(System.IAsyncResult
asyncResult) {
            object[] results = this.EndInvoke(asyncResult);
            return ((System.Data.DataTable)(results[0]));
        }
    }
}
```

2 Κώδικας Web Service C#

2.1 SystemData.asmx.cs

```
using System;
using System.IO;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using System.Xml.Serialization;
using System.Text;
using System.Collections;
using System.ComponentModel;
using System.Data.OleDb;
using System.Drawing;

namespace SystemData_WebService
{
    /// <summary>
    /// Summary description for SystemData
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET
AJAX, uncomment the following line.
    [System.Web.Script.Services.ScriptService]
    public class SystemData : System.Web.Services.WebService
    {
        #region Declarations

        public string mSelectedTable;
        private bool mTableSelected;
        ArrayList arrViews;
        ArrayList arrTables;

        #endregion


        //
```

```
        //Υπηρεσία αρχικοποίησης και ανανέωσης Προγράμματος και στοιχείων
χρήστη εφαρμογής
        //

        [WebMethod]
        public DataTable ProgramContent(string UserId)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";

            string sSQL = string.Format("select * from Program Where
User_ID='{0}'",UserId);

            SqlConnection conn = new SqlConnection(sConn2);

            DataSet dsProgramContent = new DataSet();

            SqlDataAdapter daProgramContent = new SqlDataAdapter(sSQL,
sConn2);
            daProgramContent.Fill(dsProgramContent,"Program");

            DataTable myDataTable = dsProgramContent.Tables[0];

            return myDataTable;
        }

        [WebMethod]
        public DataTable Declared_Locations(string UserId)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";


            string sSQL =string.Format("select * from Declared_Locations
Where User_ID='{0}'",UserId);

            SqlConnection conn = new SqlConnection(sConn2);

            DataSet dsDeclared_Locations = new DataSet();

            SqlDataAdapter daDeclared_Locations = new SqlDataAdapter(sSQL,
sConn2);
            daDeclared_Locations.Fill(dsDeclared_Locations,
"Declared_Locations");

            DataTable myDataTable = dsDeclared_Locations.Tables[0];

            return myDataTable;

        }

        [WebMethod]
        public DataTable Declared_Duties(string UserId)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";


            string sSQL = string.Format("select * from Declared_Duties
Where User_ID='{0}'", UserId);
```

```
            SqlConnection conn = new SqlConnection(sConn2);

            DataSet dsDeclared_Duties = new DataSet();

            SqlDataAdapter daDeclared_Duties = new SqlDataAdapter(sSQL,
sConn2);
            daDeclared_Duties.Fill(dsDeclared_Duties, "Declared_Duties");

            DataTable myDataTable = dsDeclared_Duties.Tables[0];

            return myDataTable;

        }

        //
        //Υπηρεσία Επισκόπησης Αλλαγών
        //

        [WebMethod]
        public DataTable ViewChanges(string UserID)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string sSQL = string.Format("select C.*,U.name +' '+U.surname
AS Ονοματεπώνυμο,Department AS Τμήμα,Eidikothta AS Ειδικότητα,Bathmida AS
Βαθμήδα From Users U inner join"
            + " (Select p.Program_ID,p.Date AS Ημερομηνία,p.Duty_Type AS
Εργασία,p.Duty_Start_Time AS Έναρξη,p.Duty_End_Time AS Λήξη,p.Location AS
Τοποθεσία,p.User_ID,p.Program_Name+'  -->' AS Όνομα_Προγρ from Program p
inner join"
            + " (select Item_ID,Receiver_ID from Item WHERE
Creator_ID='{0}') K on K.Item_ID=p.Program_ID) C on
C.User_ID=U.User_ID",UserID);

            SqlConnection conn = new SqlConnection(sConn2);

            //SqlCommand cmd = new SqlCommand(sSQL, conn);

            DataSet dsViewChanges = new DataSet();

            SqlDataAdapter daViewChanges = new SqlDataAdapter(sSQL,sConn2);
            daViewChanges.Fill(dsViewChanges, "Item");

            DataTable myDataTable = dsViewChanges.Tables[0];

            return myDataTable;
        }

        //
        // Υπηρεσία ανταλλαγής εφημεριών μεταξύ συναδέλφων
        //

        [WebMethod]
        public void InsertChanges(string User, int Program)
        {
            string User_ID = User;
            int Program_ID = Program;

            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
```

```
            string cmd = "INSERT INTO ChangeList (User_ID,Program_ID)
values (@User_ID,@Program_ID)";
            string cmd2 = "UPDATE Item Set Creator_ID=@User_ID WHERE
Item_ID=@Program_ID";
            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            SqlCommand SqlCmd = new SqlCommand(cmd, conn);
            SqlCmd.Parameters.AddWithValue("@User_ID", User_ID);
            SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd.ExecuteNonQuery();

            SqlCommand SqlCmd2 = new SqlCommand(cmd2, conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", User_ID);
            SqlCmd2.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd2.ExecuteNonQuery();
        }

        [WebMethod]
        public void UpdateChanges(string User, int Program)
        {
            string User_ID = User;
            int Program_ID = Program;

            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "UPDATE Program SET User_ID=@User_ID WHERE
Program_ID=@Program_ID";
            string cmd2 = "DELETE FROM ChangeList WHERE
Program_ID=@Program_ID";
            string cmd3 = "UPDATE Item SET Receiver_ID=@User_ID WHERE
Item_ID=@Program_ID";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            SqlCommand SqlCmd = new SqlCommand(cmd, conn);
            SqlCmd.Parameters.AddWithValue("@User_ID", User_ID);
            SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd.ExecuteNonQuery();

            SqlCommand SqlCmd2 = new SqlCommand(cmd2, conn);
            SqlCmd2.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd2.ExecuteNonQuery();

            SqlCommand SqlCmd3 = new SqlCommand(cmd3, conn);
            SqlCmd3.Parameters.AddWithValue("@User_ID", User_ID);
            SqlCmd3.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd3.ExecuteNonQuery();

        }


        [WebMethod]
        public DataSet UsersChangeList()
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
```

```
            string sSQL = "select U.name +' '+U.surname AS
Ονοματεπώνυμο,C.* from Users U inner join"
            +"(select p.Program_ID,p.Date AS Ημερομηνία,p.Duty_Type AS
Εργασία,p.Duty_Start_Time AS Έναρξη,p.Duty_End_Time AS Λήξη,p.Location AS
Τοποθεσία,p.Program_Name AS Όνομα_Προγρ,p.User_ID from Program p inner join
ChangeList l on l.Program_ID=p.Program_ID)"
            +" C on C.User_ID=U.User_ID";

            SqlConnection conn = new SqlConnection(sConn2);

            DataSet dsProgramContent = new DataSet();

            SqlDataAdapter daProgramContent = new SqlDataAdapter(sSQL,
sConn2);
            daProgramContent.Fill(dsProgramContent, "UsersChangeList");

            return dsProgramContent;
        }

        //
        //Υπηρεσία Ταυτοποίησης Χρήστη
        //

        [WebMethod]
        public string UserDataCheck(string username, string password)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "SELECT * FROM Users where username='@username'
AND password='@password'";
            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            string check;

            try
            {
                SqlCommand SqlCmd = new SqlCommand(cmd, conn);
                SqlCmd.Parameters.AddWithValue("@username", username);
                SqlCmd.Parameters.AddWithValue("@password", password);
                SqlCmd.ExecuteNonQuery();


                check= "ok";
                return check;
            }

            catch (Exception ex)
            {

                check="fail";
                return check;
            }

        }



        //
        //Υπηρεσία Υποβολής Αιτήσεων
        //
```

```
        [WebMethod]
        public void InsertApplications(string User, int Program, string
Application)
        {
            string User_ID = User;
            int Program_ID = Program;
            string App_Type = Application;

            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "INSERT INTO Applications
(User_ID,Program_ID,App_Type) values (@User_ID,@Program_ID,@App_Type)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            SqlCommand SqlCmd = new SqlCommand(cmd, conn);
            SqlCmd.Parameters.AddWithValue("@User_ID", User_ID);
            SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd.Parameters.AddWithValue("@App_Type", App_Type);
            SqlCmd.ExecuteNonQuery();
        }

        [WebMethod]
        public void ProgramModification(string User, int Program, string
StHour, string EnHour, string NeDate)
        {
            string User_ID = User;
            int Program_ID = Program;
            string Date = NeDate;
            string StartHour = StHour;
            string EndHour = EnHour;

            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "INSERT INTO Modifications
(User_ID,Program_ID,Date,Duty_Start_Time,Duty_End_Time) values
(@User_ID,@Program_ID,@Date,@Duty_Start_Time,@Duty_End_Time)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            SqlCommand SqlCmd = new SqlCommand(cmd, conn);
            SqlCmd.Parameters.AddWithValue("@User_ID", User_ID);
            SqlCmd.Parameters.AddWithValue("@Program_ID", Program_ID);
            SqlCmd.Parameters.AddWithValue("@Date", Date);
            SqlCmd.Parameters.AddWithValue("@Duty_Start_Time", StartHour);
            SqlCmd.Parameters.AddWithValue("@Duty_End_Time", EndHour);
            SqlCmd.ExecuteNonQuery();

        }



        /// <summary>
        /// ////////////////////////////////////////////////
        /// </summary>
        /// <returns></returns>

        [WebMethod]
```

```
        public DataTable UsersApplications()
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";

            string sSQL = "select A.[App_Type] AS Τύπος_Αίτησης,E.* from
Applications A inner join (select U.[name user] AS Όνομα, U.[surname user]
AS Επώνυμο,C.* from Users U inner join (select p.Program_ID AS
Κωδικός,p.Date AS Ημερομηνία,p.Duty_Type AS Εργασία,p.Duty_Start_Time AS
Έναρξη,p.Duty_End_Time AS Λήξη,p.Location AS
Τοποθεσία,p.Program_Name,p.User_ID from Program p inner join Applications l
on l.Program_ID=p.Program_ID)C on C.User_ID=U.User_ID)E on
E.User_ID=A.User_ID";

            SqlConnection conn = new SqlConnection(sConn2);

            DataSet dsApplications = new DataSet();
            SqlDataAdapter daApplications = new SqlDataAdapter(sSQL,
sConn2);
            daApplications.Fill(dsApplications, "Applications");

            DataTable myDataTable = dsApplications.Tables[0];

            return myDataTable;
        }

        [WebMethod]
        public string UserInfo(string username, string password)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = string.Format("SELECT * FROM Users where
username='{0}' AND password='{1}'", username, password);
            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand Sqlcmd = new SqlCommand(cmd,
conn);

            System.Data.SqlClient.SqlDataReader dr =
Sqlcmd.ExecuteReader();
            dr.Read();

            return (dr["User_ID"].ToString());

        }

        [WebMethod]
        public DataTable SelectedField()
        {
            DataTable schemaTable;


            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";

            SqlConnection conn = new SqlConnection(sConn2);

            conn.Open();

           //Retrieve records from the Employees table into a DataReader.
```

```
            System.Data.SqlClient.SqlCommand cmd = new SqlCommand("SELECT *
FROM  Program", conn);

            System.Data.SqlClient.SqlDataReader myReader =
cmd.ExecuteReader(CommandBehavior.KeyInfo);

            //Retrieve column schema into a DataTable.
             schemaTable = myReader.GetSchemaTable();

            myReader.Close();
            conn.Close();
            return schemaTable;


        }

    }
}
```

2.2 Web.config

```
<?xml version="1.0"?>
<configuration>
      <configSections>
            <sectionGroup name="system.web.extensions"
type="System.Web.Configuration.SystemWebExtensionsSectionGroup,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35">
                  <sectionGroup name="scripting"
type="System.Web.Configuration.ScriptingSectionGroup,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35">
                        <section name="scriptResourceHandler"
type="System.Web.Configuration.ScriptingScriptResourceHandlerSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/>
                        <sectionGroup name="webServices"
type="System.Web.Configuration.ScriptingWebServicesSectionGroup,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35">
                              <section name="jsonSerialization"
type="System.Web.Configuration.ScriptingJsonSerializationSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="Everywhere"/>
                              <section name="profileService"
type="System.Web.Configuration.ScriptingProfileServiceSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/>
                              <section name="authenticationService"
type="System.Web.Configuration.ScriptingAuthenticationServiceSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/>
                              <section name="roleService"
type="System.Web.Configuration.ScriptingRoleServiceSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
```

```
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/>
                        </sectionGroup>
                  </sectionGroup>
            </sectionGroup>
       <sectionGroup name="applicationSettings"
type="System.Configuration.ApplicationSettingsGroup, System,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" >
   <section name="SystemData_WebService.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false"
/>
  </sectionGroup>
 </configSections>
      <appSettings/>
      <connectionStrings/>
      <system.web>
            <!--
            Set compilation debug="true" to insert debugging
            symbols into the compiled page. Because this
            affects performance, set this value to true only
            during development.
        -->
            <compilation debug="true">
                  <assemblies>
                        <add assembly="System.Core, Version=3.5.0.0,
Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
                        <add assembly="System.Data.DataSetExtensions,
Version=3.5.0.0, Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
                        <add assembly="System.Web.Extensions,
Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
                        <add assembly="System.Xml.Linq, Version=3.5.0.0,
Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
                  </assemblies>
            </compilation>
            <!--
            The <authentication> section enables configuration
            of the security authentication mode used by
            ASP.NET to identify an incoming user.
        -->
            <authentication mode="Windows"/>
            <!--
            The <customErrors> section enables configuration
            of what to do if/when an unhandled error occurs
            during the execution of a request. Specifically,
            it enables developers to configure html error pages
            to be displayed in place of a error stack trace.

        <customErrors mode="RemoteOnly"
defaultRedirect="GenericErrorPage.htm">
            <error statusCode="403" redirect="NoAccess.htm" />
            <error statusCode="404" redirect="FileNotFound.htm" />
        </customErrors>
        -->
            <pages>
                  <controls>
                        <add tagPrefix="asp" namespace="System.Web.UI"
assembly="System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
```

```
                        <add tagPrefix="asp"
namespace="System.Web.UI.WebControls" assembly="System.Web.Extensions,
Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
                  </controls>
            </pages>
            <httpHandlers>
                  <remove verb="*" path="*.asmx"/>
                  <add verb="*" path="*.asmx" validate="false"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                  <add verb="*" path="*_AppService.axd" validate="false"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                  <add verb="GET,HEAD" path="ScriptResource.axd"
type="System.Web.Handlers.ScriptResourceHandler, System.Web.Extensions,
Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"
validate="false"/>
            </httpHandlers>
            <httpModules>
                  <add name="ScriptModule"
type="System.Web.Handlers.ScriptModule, System.Web.Extensions,
Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
            </httpModules>
      </system.web>
      <system.codedom>
            <compilers>
                  <compiler language="c#;cs;csharp" extension=".cs"
warningLevel="4" type="Microsoft.CSharp.CSharpCodeProvider, System,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089">
                        <providerOption name="CompilerVersion"
value="v3.5"/>
                        <providerOption name="WarnAsError" value="false"/>
                  </compiler>
            </compilers>
      </system.codedom>
      <!--
        The system.webServer section is required for running ASP.NET AJAX
under Internet
        Information Services 7.0.  It is not necessary for previous version
of IIS.
    -->
      <system.webServer>
            <validation validateIntegratedModeConfiguration="false"/>
            <modules>
                  <remove name="ScriptModule"/>
                  <add name="ScriptModule" preCondition="managedHandler"
type="System.Web.Handlers.ScriptModule, System.Web.Extensions,
Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
            </modules>
            <handlers>
                  <remove name="WebServiceHandlerFactory-Integrated"/>
                  <remove name="ScriptHandlerFactory"/>
                  <remove name="ScriptHandlerFactoryAppServices"/>
                  <remove name="ScriptResource"/>
                  <add name="ScriptHandlerFactory" verb="*" path="*.asmx"
preCondition="integratedMode"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
```

```
                  <add name="ScriptHandlerFactoryAppServices" verb="*"
path="*_AppService.axd" preCondition="integratedMode"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                  <add name="ScriptResource" preCondition="integratedMode"
verb="GET,HEAD" path="ScriptResource.axd"
type="System.Web.Handlers.ScriptResourceHandler, System.Web.Extensions,
Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
            </handlers>
      </system.webServer>
      <runtime>
            <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
                  <dependentAssembly>
                        <assemblyIdentity name="System.Web.Extensions"
publicKeyToken="31bf3856ad364e35"/>
                        <bindingRedirect oldVersion="1.0.0.0-1.1.0.0"
newVersion="3.5.0.0"/>
                  </dependentAssembly>
                  <dependentAssembly>
                        <assemblyIdentity
name="System.Web.Extensions.Design" publicKeyToken="31bf3856ad364e35"/>
                        <bindingRedirect oldVersion="1.0.0.0-1.1.0.0"
newVersion="3.5.0.0"/>
                  </dependentAssembly>
            </assemblyBinding>
      </runtime>
 <applicationSettings>
  <SystemData_WebService.Properties.Settings>
   <setting name="SystemData_WebService_WebReference_SystemData"
    serializeAs="String">
    <value>http://192.168.1.33/MyWebServices/SystemData.asmx</value>
   </setting>
   <setting name="SystemData_WebService_LocalHost_WebReference_SystemData"
    serializeAs="String">
    <value>http://127.0.0.1/MyWebServices/SystemData.asmx</value>
   </setting>
  </SystemData_WebService.Properties.Settings>
 </applicationSettings>
</configuration>
```

3 Κεντρική Εφαρμογή

3.1 Φόρμα MainMenu.aspx.cs

3.1.1 Κώδικας C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WorkStation_V1
{
    public partial class MainMenu : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void ImageButtonAddUser_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("UserManagement.aspx");
        }

        protected void ImageButtonCreateProgram_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void ImageButtonApplicationManagement_Click(object
sender, ImageClickEventArgs e)
        {
            Response.Redirect("ChangeApplicationManagement.aspx");
        }

        protected void ImageButtonEditUsers_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("EditUsers.aspx");
        }
    }
}
```

3.1.2 Κώδικας HTML

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="MainMenu.aspx.cs"
Inherits="WorkStation_V1.MainMenu" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
            <div style="border-style: double; text-align: center;">

        <span lang="el"
            style="font-family: Calibri; font-size: xx-large; font-weight:
bold; font-style: normal; clip: rect(auto, auto, auto, auto); text-align:
center;">
        Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων</span><br />
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        <br />
 Κεντρική Εφαρμογή Work Station-
            Υποσύστημα Διαχείρισης Δεδομένων</span></div>

    <p style="text-align: center">
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        Κεντρικό Μενού</span></p>
    <p style="border-bottom-style: solid">
         </p>

       <table style="text-align: center"><tr>
           <td style="padding: 10px; color: #3366FF; background-color:
#3366FF; width: 10px;"></td></tr>
       <tr>
       <td>

        <table>

        <tr>

        <td>
            <asp:Image ID="Image1" runat="server"
                ImageUrl="~/Icons/newhospitalmanagementsoftware.jpg" />
        </td>

        <td>
         <table style="font-size: medium; font-weight: bold">

          <tr>
          <td><span lang="el">Εγγραφή Χρηστών</span></td>
           <td>
             <asp:ImageButton ID="ImageButtonAddUser" runat="server"
              ImageUrl="~/Icons/new-user-group-icone-6256-128.png"
                                  onclick="ImageButtonAddUser_Click" />
```

```
                </td>
               </tr>

                <tr>

                <td><span lang="el">Δημιουργία Προγραμμάτων</span></td>
                   <td>
                    <asp:ImageButton ID="ImageButtonCreateProgram"
                   runat="server"
                   ImageUrl="~/Icons/ChronologicalReview.png"
                   onclick="ImageButtonCreateProgram_Click" />
                  </td>
                  </tr>

                  <tr>
                  <td><span lang="el">Διαχείριση Αιτήσεων</span></td>
                   <td>
                  <asp:ImageButton ID="ImageButtonApplicationManagement"
runat="server"   ImageUrl="~/Icons/Application.png"

onclick="ImageButtonApplicationManagement_Click" />
                  </td>
                  </tr>
                     <tr>
                     <td><span lang="el">Επεξεργασία Χρηστών</span></td>
                      <td>
                             <asp:ImageButton ID="ImageButtonEditUsers"
      runat="server"
                       ImageUrl="~/Icons/edit-find-replace-old-icone-8806-
64.png"
                       onclick="ImageButtonEditUsers_Click" />
                        </td>
                     </tr>

                     </table>
        </td>

        </tr>
        </table>

        </td>
        </tr>
        </table>

    </form>
</body>
</html>
```

3.2 Φόρμα UserManagement.aspx.cs

3.2.1 Κώδικας C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
```

```
using System.IO;
using System.Web.Services;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using System.Xml.Serialization;
using System.Text;
using System.Collections;
using System.ComponentModel;
using System.Data.OleDb;
using System.Drawing;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

namespace WorkStation_V1
{
    public partial class UserManagement : System.Web.UI.Page
    {


        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void ButtonSaveUser_Click(object sender, EventArgs e)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "INSERT INTO Users
(User_Team,name,surname,username,password,AMKA,Thesi,Eidikothta,Department,
Bathmida) values
(@User_Team,@name_user,@surname_user,@username,@password,@AMKA,@Thesi,@Eidi
kothta,@Department,@Bathmida)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            try
            {
                SqlCommand SqlCmd = new SqlCommand(cmd, conn);
                SqlCmd.Parameters.AddWithValue("@User_Team",
DropDownListUserTeam.Text);
                SqlCmd.Parameters.AddWithValue("@name_user",
TextBoxName.Text);
                SqlCmd.Parameters.AddWithValue("@surname_user",
TextBoxSurname.Text);
                SqlCmd.Parameters.AddWithValue("@username",
TextBoxUserName.Text);
                SqlCmd.Parameters.AddWithValue("@password",
TextBoxPassword.Text);
                SqlCmd.Parameters.AddWithValue("@AMKA", TextBoxAMKA.Text);
                SqlCmd.Parameters.AddWithValue("@Thesi",
TextBoxThesi.Text);
                SqlCmd.Parameters.AddWithValue("@Eidikothta",
DropDownListEidikothta.Text);
                SqlCmd.Parameters.AddWithValue("@Department",
DropDownListDepartments.Text);
```

```
                SqlCmd.Parameters.AddWithValue("@Bathmida",
DropDownListBathmida.Text);
                SqlCmd.ExecuteNonQuery();
            }

            catch
            {
                Console.Write("Ο χρήστης είναι ήδη εγγεγραμένος");
            }

        }

        public string ReturnID()
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd3 = "Select User_ID From Users Where
User_Team=@User_Team AND name=@name_user AND surname=@surname_user AND
username=@username AND password=@password";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd3 = new SqlCommand(cmd3,
conn);
            SqlCmd3.Parameters.AddWithValue("@User_Team",
DropDownListUserTeam.Text);
            SqlCmd3.Parameters.AddWithValue("@name_user",
TextBoxName.Text);
            SqlCmd3.Parameters.AddWithValue("@surname_user",
TextBoxSurname.Text);
            SqlCmd3.Parameters.AddWithValue("@username",
TextBoxUserName.Text);
            SqlCmd3.Parameters.AddWithValue("@password",
TextBoxPassword.Text);
            SqlCmd3.ExecuteNonQuery();
            System.Data.SqlClient.SqlDataReader dr =
SqlCmd3.ExecuteReader();
            dr.Read();

            string userid = dr["User_ID"].ToString();

            dr.Close();

            return userid;

        }

        protected void ButtonDieuth1_Click(object sender, EventArgs e)
        {
            string UserID = ReturnID();

            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Address
(User_ID,nomos,orofos,city,address,TK,perioxi,Xwra) VALUES
(@User_ID,@nomos,@orofos,@city,@address,@TK,@perioxi,@Xwra)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();
```

```
            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@nomos", TextBoxNom1.Text);
            SqlCmd2.Parameters.AddWithValue("@orofos", TextBoxDum1.Text);
            SqlCmd2.Parameters.AddWithValue("@city", TextBoxCity1.Text);
            SqlCmd2.Parameters.AddWithValue("@address", TextBoxDrom1.Text);
            SqlCmd2.Parameters.AddWithValue("@TK", TextBoxTKK.Text);
            SqlCmd2.Parameters.AddWithValue("@perioxi", TextBoxLoc1.Text);
            SqlCmd2.Parameters.AddWithValue("@Xwra", TextBoxCun1.Text);
            SqlCmd2.ExecuteNonQuery();

            string sSQL3 = string.Format("select address AS
Διεύθυνση,orofos AS Όροφος,city AS Πόλη,nomos AS Νομός,TK,perioxi AS
Περιοχή,Xwra AS Χώρα FROM Address where User_ID='{0}'", UserID);
            DataSet dsAddress = new DataSet();
            SqlDataAdapter daAddress = new SqlDataAdapter(sSQL3, sConn2);
            daAddress.Fill(dsAddress, "Address");
            DataTable myAddress = dsAddress.Tables[0];
            GVAdr.DataSource = myAddress;
            GVAdr.DataBind();
            TextBoxNom1.Text = TextBoxDum1.Text = TextBoxCity1.Text =
TextBoxDrom1.Text = TextBoxTKK.Text = TextBoxLoc1.Text = TextBoxCun1.Text =
"";

        }


        protected void ButtonTelOk_Click(object sender, EventArgs e)
        {
            string UserID = ReturnID();
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Phone_Numbers
(User_ID,Tupos_Til,Til) VALUES (@User_ID,@Tupos_Til,@Til)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@Tupos_Til",
DropDownListTelType1.Text);
            SqlCmd2.Parameters.AddWithValue("@Til", TextBoxTelNum1.Text);
            SqlCmd2.ExecuteNonQuery();

            string sSQL2 = string.Format("select Tupos_Til AS Τύπος_Τηλ,Til
AS Αρ_Τηλεφώνου FROM Phone_Numbers where User_ID='{0}'", UserID);

            DataSet dsPhones = new DataSet();
            SqlDataAdapter daPhones = new SqlDataAdapter(sSQL2, sConn2);
            daPhones.Fill(dsPhones, "Phone_Numbers");
            DataTable myPhones = dsPhones.Tables[0];
            GridViewPhones.DataSource = myPhones;
            GridViewPhones.DataBind();

            TextBoxTelNum1.Text = "";
        }
```

```
        protected void ButtonDuty1_Click(object sender, EventArgs e)
        {
            string UserID = ReturnID();
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Declared_Duties (User_ID,Duty_Types)
VALUES (@User_ID,@Duty_Types)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@Duty_Types",
DropDownListDuty1.Text);

            SqlCmd2.ExecuteNonQuery();

            string sSQL4 = string.Format("select Duty_Types AS
Τύπος_Εργασίας FROM Declared_Duties where User_ID='{0}'", UserID);
            DataSet dsDD = new DataSet();
            SqlDataAdapter daDD = new SqlDataAdapter(sSQL4, sConn2);
            daDD.Fill(dsDD, "Declared_Duties");
            DataTable myDD = dsDD.Tables[0];
            GridViewDD.DataSource = myDD;
            GridViewDD.DataBind();
        }


        protected void ButtonLoc1_Click(object sender, EventArgs e)
        {
            string UserID = ReturnID();
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Declared_Locations
(User_ID,Location) VALUES (@User_ID,@Location)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@Location",
DropDownListLoc1.Text);
            SqlCmd2.ExecuteNonQuery();

            string sSQL5 = string.Format("select Location AS Τοποθεσίες
FROM Declared_Locations where User_ID='{0}'", UserID);
            DataSet dsDL = new DataSet();
            SqlDataAdapter daDL = new SqlDataAdapter(sSQL5, sConn2);
            daDL.Fill(dsDL, "Declared_Locations");
            DataTable myDL = dsDL.Tables[0];
            GridViewDL.DataSource = myDL;
            GridViewDL.DataBind();

        }
```

```
        protected void ImageButtonAddUser_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("UserManagement.aspx");
        }

        protected void ImageButtonProgramCreation_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void ImageButtonApplications_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("ChangeApplicationManagement.aspx");
        }

        protected void ImageButton2_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("EditUsers.aspx");
        }

        protected void ImageButtonGoToMenu_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("MainMenu.aspx");
        }


    }

    public class User
    {
        public string User_ID="";

        public void SetUserID(string Userid)
        {
            User_ID = Userid;
        }

        public string GetUserID()
        {
            return User_ID;
        }
    }
}
```

3.2.2 Κώδικας HTML

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="UserManagement.aspx.cs"
Inherits="WorkStation_V1.UserManagement" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
```

```
<head runat="server">
    <title></title>
    <style type="text/css">
        .style2
        {
            width: 238px;
        }
        .style3
        {
            width: 147px;
        }
        .style4
        {
            width: 132px;
        }
        .style5
        {
            width: 177px;
        }
        .style6
        {
            width: 197px;
        }
        .style7
        {
            width: 22px;
        }
        .style10
        {
            width: 90px;
        }
        .style11
        {
            width: 241px;
        }
        .style14
        {
            width: 113px;
        }
        .style15
        {
            width: 91px;
        }
        .style16
        {
            width: 159px;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div style="border-style: double">

        <span lang="el"
            style="font-family: Calibri; font-size: xx-large; font-weight:
bold; font-style: normal; clip: rect(auto, auto, auto, auto); text-align:
center;">
        Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων</span><br />
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
```

```
        <br />
 Κεντρική Εφαρμογή Work Station-
            Υποσύστημα Διαχείρισης Δεδομένων</span></div>
            <table>
        <tr>
        <td><span lang="el">Κεντρικό Μενού</span></td>
        <td>
            <asp:ImageButton ID="ImageButton1" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
        <td><span lang="el">Χρήστες</span></td>
        <td>
            <asp:ImageButton ID="ImageButton2" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
            <td>
                <span lang="el">Εγγραφή Νέου Χρήστη</span></td>
        </tr>
        </table>
    <p style="text-align: center">
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        Εγγραφή Χρηστών </span></p>
    <p style="border-bottom-style: solid">
         </p>
    <p style="text-align: center">
        <p style="font-size: medium; font-weight: bold; vertical-align:
middle; text-align: center"><span lang="el"><td class="style14"
align="center" style="font-weight: bold"
                    width="30"><span lang="el">Στοιχεία</span>
Εργασίας</span>
        </p>
        <Table>


        <td class="style14" align="center" width="30" style="text-align:
center"><span lang="el">Εισαγωγή Χρήστη:  </span></td>
            <td class="style4" style="text-align: center">
                <asp:Label ID="Label3" runat="server" Text="Ομάδα
Χρήστη"></asp:Label>
                <br />
                <asp:DropDownList ID="DropDownListUserTeam" runat="server"
                    DataSourceID="SumUserTeams"
DataTextField="Onoma_Omadas"
                    DataValueField="Onoma_Omadas">
                </asp:DropDownList>
            </td>
        <td class="style16" style="text-align: center">
              <span lang="en-us" style="text-align: center"> </span>
<asp:Label ID="Label1" runat="server" Text="Όνομα"></asp:Label>
            <br />
            <asp:TextBox ID="TextBoxName" runat="server"
                ></asp:TextBox>
            </td>
            <td class="style7" style="text-align: center">
                <asp:Label ID="Label2" runat="server"
Text="Επώνυμο"></asp:Label>
                <br />
                <asp:TextBox ID="TextBoxSurname"
runat="server"></asp:TextBox>
```

```
            </td>
            <td style="text-align: center">
                <span lang="el">
                <asp:Label ID="Label4" runat="server"
Text="Τμήμα"></asp:Label>
                <br />
                <asp:DropDownList ID="DropDownListDepartments"
runat="server"
                    DataSourceID="Departments1"
DataTextField="Onoma_Tmimatos"
                    DataValueField="Onoma_Tmimatos" >
                </asp:DropDownList>
                </span>
                <br />
            </td>
            <td style="text-align: center">
                <span lang="el">
                <asp:Label ID="Label5" runat="server"
Text="Ειδικότητα"></asp:Label>
                <br />
                <asp:DropDownList ID="DropDownListEidikothta"
runat="server" DataSourceID="Specifity1"
                    DataTextField="Eidikothta_Name"
DataValueField="Eidikothta_Name">
                </asp:DropDownList>
            </td>

            <td class="style4" style="text-align: center">
                <asp:Label ID="Label6" runat="server"
Text="Βαθμίδα"></asp:Label>
                <br />
                <asp:DropDownList ID="DropDownListBathmida" runat="server"
DataSourceID="Scales1"
                    DataTextField="Onoma_Bathmidas"
DataValueField="Onoma_Bathmidas"
                    >
                </asp:DropDownList>
            </td>
            <td class="style3" style="text-align: center">
                <asp:Label ID="Label7" runat="server"
Text="Θέση"></asp:Label>
                <br />
                <asp:TextBox ID="TextBoxThesi"
runat="server"></asp:TextBox>
            </td>
            </tr>

        </table>
        <p style="border-bottom-style: solid"></p>
        <p style="font-size: medium; font-weight: bold; vertical-align:
middle; text-align: center"><span lang="el"><td class="style14"
align="center" style="font-weight: bold"
                    width="30"><span lang="el">Στοιχεία</span>
Ταυτοποίησης</span></p>
        <table>
        <tr><td class="style5"><span lang="el">Όνομα Χρήστη<br />
            <asp:TextBox ID="TextBoxUserName" runat="server"></asp:TextBox>
            </span></td>
            <td class="style6"><span lang="el">Κωδικός Χρήστη<br />
                <asp:TextBox ID="TextBoxPassword"
runat="server"></asp:TextBox>
```

```
                </span></td>
            <td><span lang="el">A.M.K.A.<br />
                <asp:TextBox ID="TextBoxAMKA" runat="server"></asp:TextBox>
                </span></td>
            <td class="style7"> </td>
            </tr>

            <tr><td>
                 </td></tr>
                </table>
                <p style="text-align: center">
                <asp:Button ID="ButtonSaveUser" runat="server"
onclick="ButtonSaveUser_Click"
                    Text="Αποθήκευση Χρήστη" Width="175px" />
                </p>
                <p style="border-bottom-style: solid">
                </p>
                <p style="font-size: medium; font-weight: bold; vertical-
align: middle; text-align: center"><span lang="el"><td class="style14"
align="center" style="font-weight: bold"
                    width="30">Διευθύνσεις Χρήστη</span></p>

                <table>


            <tr><td class="style15"><span lang="el">Διεύθυνση:</span></td>
            <td class="style6"><span lang="el">Δρόμος<br />
                <asp:TextBox ID="TextBoxDrom1"
runat="server"></asp:TextBox>
                </span></td>
            <td><span lang="el">Όροφος<br />
                <asp:TextBox ID="TextBoxDum1" runat="server"></asp:TextBox>
                </span></td>
            <td class="style7">Π<span lang="el">όλη/Χωριό<br />
                <asp:TextBox ID="TextBoxCity1"
runat="server"></asp:TextBox>
                </span></td>
            <td class="style4"><span lang="el">Νομός<br />
                <asp:TextBox ID="TextBoxNom1" runat="server"></asp:TextBox>
                </span></td>
            <td class="style3"><span lang="el">Ταχ. Κώδικας<br />
                <asp:TextBox ID="TextBoxTKK" runat="server"></asp:TextBox>
                </span></td>
                <td><span lang="el">Περιοχή<br />
                    <asp:TextBox ID="TextBoxLoc1"
runat="server"></asp:TextBox>
                    </span></td>
                <td><span lang="el">Χώρα<br />
                    <asp:TextBox ID="TextBoxCun1"
runat="server"></asp:TextBox>
                    </span></td>
                    <td>
                        <br />
                        <asp:Button ID="ButtonDieuth1" runat="server"
Text="OK"
                            onclick="ButtonDieuth1_Click" />
                </td>
            </tr>

        </Table>
```

```
        <p></p>

        <table>
        <tr>
        <td>

            <asp:GridView ID="GVAdr" runat="server">
            </asp:GridView>

        </td>
        </tr>
        </table>
        <p style="border-bottom-style: solid">
        </p>
        <table>

        <tr>

        <td class="style2" style="border-right-style: solid"><span
lang="el"
                style="font-size: medium; text-align: center">Τηλέφωνα
Χρήστη:</span><br />
            <table>
            <tr><td>
                <asp:DropDownList ID="DropDownListTelType1" runat="server">
                    <asp:ListItem>Κινητό:</asp:ListItem>
                    <asp:ListItem>Fax:</asp:ListItem>
                    <asp:ListItem>Σταθερό:</asp:ListItem>
                </asp:DropDownList>
                </td><td>
                    <asp:TextBox ID="TextBoxTelNum1"
runat="server"></asp:TextBox>
                </td>
                <td><span lang="el">
                        <asp:Button ID="ButtonTelOk" runat="server"
Text="OK"
                        onclick="ButtonTelOk_Click" />
                    </td>
                </tr>
                </table>
                <table>
                <tr>
                <td>
                 <asp:GridView ID="GridViewPhones" runat="server">
            </asp:GridView>
                </td>
                </tr>
                </table>

            </td>
            <td class="style11" style="border-right-style: solid">

            <span lang="el" style="font-size: medium; text-align:
center">Δήλωση Καθηκόντων:</span><br />
            <table>
            <tr><td class="style10">
                <span lang="el">
            <asp:Label ID="Label8" runat="server"
Text="Εργασία:"></asp:Label>
    </span>
                </td><td>
```

```
                    <span lang="el">
                <asp:DropDownList ID="DropDownListDuty1" runat="server"
DataSourceID="DutyType1"
                        DataTextField="Duty_Type_Name"
DataValueField="Duty_Type_Name">
                </asp:DropDownList>
    </span>
                </td>
                <td><span lang="el">
                        <asp:Button ID="ButtonDuty1" runat="server"
Text="OK"
                        onclick="ButtonDuty1_Click" />
                    </td>
                </tr>
                </table>
                <table>
                <tr>
                <td>
                 <asp:GridView ID="GridViewDD" runat="server">
                </asp:GridView>
                </td>
                </tr>
                </table>

            </td>
            <td class="style2">

            <span lang="el" style="font-size: medium; text-align:
center">Δήλωση Τοποθεσιών:</span><br />
            <table>
            <tr><td class="style10">
                <span lang="el">
            <asp:Label ID="Label12" runat="server"
Text="Τοποθεσία:"></asp:Label>
    </span>
                </td><td>
                    <span lang="el">
                <asp:DropDownList ID="DropDownListLoc1" runat="server"
DataSourceID="Locations1"
                        DataTextField="Location_Name"
DataValueField="Location_Name">
                </asp:DropDownList>
    </span>
                </td>
                <td><span lang="el">
                        <asp:Button ID="ButtonLoc1" runat="server"
Text="OK"
                        onclick="ButtonLoc1_Click" />
                    </td>
                </tr>
                </table>
                <table>
                <tr>
                <td>
                <asp:GridView ID="GridViewDL" runat="server">
                </asp:GridView>
                </td>
                </tr>
                </table>
            </td>
            </tr>
```

```
        </table>
        </td>
        <td>
        <table>

             </tr>
        </table>
         </td>
        </tr>
        </table>
    </p>
    <table><tr><td align="center" style="border-style: solid">
        <asp:ImageButton ID="ImageButtonProgramCreation" runat="server"
            ImageUrl="~/Icons/ChronologicalReview.png"
            onclick="ImageButtonProgramCreation_Click" />
        <br />
        <span lang="el">Δημιουργία Προγράμματος</span></td>
        <td></td>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonAddUser" runat="server"
ImageAlign="Middle"
                ImageUrl="~/Icons/new-user-group-icone-6256-128.png"
                onclick="ImageButtonAddUser_Click" />
            <br />
            <span lang="el">Εισαγωγή νέου χρήστη </span>
        </td>
        <td></td>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonApplications" runat="server"
                ImageUrl="~/Icons/Application.png"
onclick="ImageButtonApplications_Click" />
            <br />
            <span lang="el">Διαχείριση Αιτήσεων</span></td>
            <td>
            </td>
            <td style="text-align: center; border-style: solid">
                <asp:ImageButton ID="ImageButton3" runat="server"
                    ImageUrl="~/Icons/edit-find-replace-old-icone-8806-
64.png"
                    onclick="ImageButton2_Click" />
                <br />
                <span lang="el">Επεξεργασία Χρηστών</span></td>
            <td></td>
            <td align="center" style="border-style: solid">
                <asp:ImageButton ID="ImageButtonGoToMenu" runat="server"
                    ImageUrl="~/Icons/office-icone-8240-128.png"
                    onclick="ImageButtonGoToMenu_Click" />
                <br />
                <span lang="el">Κεντρικό Μενού</span></td>
        </tr></table>
    </span>
    <span lang="el">
    <asp:SqlDataSource ID="DutyType1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Duty_Type_Name] FROM [Sum_Duties]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="Locations1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
```

```
        SelectCommand="SELECT [Location_Name] FROM
[Locations]"></asp:SqlDataSource>
    <asp:SqlDataSource ID="Departments1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Tmimatos] FROM [Sum_Departments]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="Specifity1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Eidikothta_Name] FROM [Eidikothtes]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="Scales1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Bathmidas] FROM [Bathmides]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="SumUserTeams" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Omadas] FROM [Sum_User_Teams]">
    </asp:SqlDataSource>
    </form>
    </body>
</html>
```

3.3 Φόρμα EditUser.aspx.cs

3.3.1 Κώδικας C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.IO;
using System.Web.Services;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using System.Xml.Serialization;
using System.Text;
using System.Collections;
using System.ComponentModel;
using System.Data.OleDb;
using System.Drawing;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

namespace WorkStation_V1
{
    public partial class EditUsers : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
```

```
        }

        protected void ImageButtonCreation_Click(object sender,
ImageClickEventArgs e)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";

            string sSQL = "select name+' '+surname AS Όνομα,Department AS
Τμήμα,Thesi AS Θέση,Eidikothta AS Ειδικότητα,Bathmida AS Βαθμίδα,User_ID
from Users";

            if (CheckBoxUserTeam.Checked == CheckBoxDepartment.Checked ==
CheckBoxEidikothta.Checked == CheckBoxBathmida.Checked == true)
            {
                sSQL = "select name+ ' ' +surname AS Όνομα,User_Team AS
Ομάδα_Εργασίας,Department AS Τμήμα,Thesi AS Θέση,Eidikothta AS
Ειδικότητα,Bathmida AS Βαθμίδα,User_ID from Users";
            }

            Session["CurrentQuery"] = sSQL;
            string newquery = (string)Session["CurrentQuery"];

            if (CheckBoxUserTeam.Checked == true)
            {
                DropDownListUTeam.Enabled = false;
            }
            else
            {

                newquery = string.Format("{0} where User_Team=N'{1}'",
newquery, DropDownListUTeam.Text);

            }

            if (CheckBoxDepartment.Checked == true)
            {
                DropDownLDepartment.Enabled = false;
            }
            else
            {
                if (CheckBoxUserTeam.Checked == true)
                {
                    newquery = string.Format("{0} WHERE Department=N'{1}'",
newquery, DropDownLDepartment.Text);

                }
                else
                {
                    newquery = string.Format("{0} AND Department=N'{1}'",
newquery, DropDownLDepartment.Text);

                }
            }


            if (CheckBoxBathmida.Checked == true)
            {
                DropDownLBathmida.Enabled = false;
            }
```

```
            else
            {
                if ((CheckBoxUserTeam.Checked == true) &&
(CheckBoxDepartment.Checked == true))
                {
                    newquery = string.Format("{0} WHERE Bathmida=N'{1}'",
newquery, DropDownLBathmida.Text);

                }
                else
                {
                    newquery = string.Format("{0} AND Bathmida=N'{1}'",
newquery, DropDownLBathmida.Text);

                }
            }

            if (CheckBoxEidikothta.Checked == true)
            {
                DropDownLEidikothta.Enabled = false;
            }
            else
            {
                if ((CheckBoxUserTeam.Checked == true) &&
(CheckBoxDepartment.Checked == true) && (CheckBoxBathmida.Checked == true))
                {
                    newquery = string.Format("{0} WHERE Eidikothta=N'{1}'",
newquery, DropDownLEidikothta.Text);

                }
                else
                {
                    newquery = string.Format("{0} AND Eidikothta=N'{1}'",
newquery, DropDownLEidikothta.Text);

                }
            }

            Session["CurrentQuery"] = newquery;


            Console.Write((string)Session["CurrentQuery"]);
            SqlConnection conn = new SqlConnection(sConn2);
            DataSet dsCreate = new DataSet();
            SqlDataAdapter daCreate = new
SqlDataAdapter((string)Session["CurrentQuery"], sConn2);
            daCreate.Fill(dsCreate, "Users");

            Session["myDataSet"] = dsCreate;


            DataTable myDataTable = dsCreate.Tables[0];
            Specific_Users_List.DataSource = myDataTable;
            Specific_Users_List.DataBind();
        }

        protected void Specific_Users_List_SelectedIndexChanged(object
sender, EventArgs e)
        {
            GridViewRow row = Specific_Users_List.SelectedRow;
            string UserId = "";
```

```
            UserId = row.Cells[7].Text;
            Session["UserID"] = UserId;



            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";

            string sSQL = string.Format("select name+ ' ' +surname AS
Όνομα,User_Team AS Ομάδα_Εργασίας,Department AS Τμήμα,Thesi AS
Θέση,Eidikothta AS Ειδικότητα,Bathmida AS Βαθμίδα,User_ID FROM Users where
User_ID='{0}'", UserId);
            string sSQL2 = string.Format("select Tupos_Til AS Τύπος_Τηλ,Til
AS Αρ_Τηλεφώνου FROM Phone_Numbers where User_ID='{0}'", UserId);
            string sSQL3 = string.Format("select address AS
Διεύθυνση,orofos AS Όροφος,city AS Πόλη,nomos AS Νομός,TK,perioxi AS
Περιοχή,Xwra AS Χώρα FROM Address where User_ID='{0}'", UserId);
            string sSQL4 = string.Format("select Duty_Types AS
Τύπος_Εργασίας FROM Declared_Duties where User_ID='{0}'", UserId);
            string sSQL5 = string.Format("select Location AS Τοποθεσίες
FROM Declared_Locations where User_ID='{0}'", UserId);


            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            DataSet dsUser = new DataSet();
            DataSet dsPhones = new DataSet();
            DataSet dsAddress = new DataSet();
            DataSet dsDD = new DataSet();
            DataSet dsDL = new DataSet();

            SqlDataAdapter daUser = new SqlDataAdapter(sSQL, sConn2);
            SqlDataAdapter daPhones = new SqlDataAdapter(sSQL2, sConn2);
            SqlDataAdapter daAddress = new SqlDataAdapter(sSQL3, sConn2);
            SqlDataAdapter daDD = new SqlDataAdapter(sSQL4, sConn2);
            SqlDataAdapter daDL = new SqlDataAdapter(sSQL5, sConn2);


            daUser.Fill(dsUser, "User");
            daPhones.Fill(dsPhones, "Phone_Numbers");
            daAddress.Fill(dsAddress, "Address");
            daDD.Fill(dsDD, "Declared_Duties");
            daDL.Fill(dsDL, "Declared_Locations");

            DataTable myUser = dsUser.Tables[0];
            DataTable myPhones = dsPhones.Tables[0];
            DataTable myAddress = dsAddress.Tables[0];
            DataTable myDD = dsDD.Tables[0];
            DataTable myDL = dsDL.Tables[0];

            GridViewUser.DataSource = myUser;
            GridViewPhones.DataSource = myPhones;
            GridViewAd.DataSource = myAddress;
            GridViewDD.DataSource = myDD;
            GridViewDL.DataSource = myDL;

            GridViewUser.DataBind();
            GridViewPhones.DataBind();
            GridViewAd.DataBind();
            GridViewDD.DataBind();
```

```
            GridViewDL.DataBind();

            System.Data.SqlClient.SqlCommand SqlCmd3 = new
SqlCommand(sSQL2, conn);
            System.Data.SqlClient.SqlDataReader dr =
SqlCmd3.ExecuteReader();
            dr.Read();

            dr.Close();
        }



        protected void ButtonDieuth1_Click(object sender, EventArgs e)
        {
            string UserID = (string)Session["UserID"];

            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Address
(User_ID,nomos,orofos,city,address,TK,perioxi,Xwra) VALUES
(@User_ID,@nomos,@orofos,@city,@address,@TK,@perioxi,@Xwra)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@nomos", TextBoxNom1.Text);
            SqlCmd2.Parameters.AddWithValue("@orofos", TextBoxDum1.Text);
            SqlCmd2.Parameters.AddWithValue("@city", TextBoxCity1.Text);
            SqlCmd2.Parameters.AddWithValue("@address", TextBoxDrom1.Text);
            SqlCmd2.Parameters.AddWithValue("@TK", TextBoxTKK.Text);
            SqlCmd2.Parameters.AddWithValue("@perioxi", TextBoxLoc1.Text);
            SqlCmd2.Parameters.AddWithValue("@Xwra", TextBoxCun1.Text);
            SqlCmd2.ExecuteNonQuery();

        }


        protected void ButtonTelOk_Click(object sender, EventArgs e)
        {
            string UserID = (string)Session["UserID"];
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Phone_Numbers
(User_ID,Tupos_Til,Til) VALUES (@User_ID,@Tupos_Til,@Til)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@Tupos_Til",
DropDownListTelType1.Text);
            SqlCmd2.Parameters.AddWithValue("@Til", TextBoxTelNum1.Text);
            SqlCmd2.ExecuteNonQuery();
        }
```

```
        protected void ButtonDuty1_Click(object sender, EventArgs e)
        {
            string UserID = (string)Session["UserID"];
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Declared_Duties (User_ID,Duty_Types)
VALUES (@User_ID,@Duty_Types)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@Duty_Types",
DropDownListDuty1.Text);

            SqlCmd2.ExecuteNonQuery();

        }


        protected void ButtonLoc1_Click(object sender, EventArgs e)
        {
            string UserID = (string)Session["UserID"];
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd2 = "INSERT INTO Declared_Locations
(User_ID,Location) VALUES (@User_ID,@Location)";

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();


            System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2,
conn);
            SqlCmd2.Parameters.AddWithValue("@User_ID", UserID);
            SqlCmd2.Parameters.AddWithValue("@Location",
DropDownListLoc1.Text);
            SqlCmd2.ExecuteNonQuery();
        }


        protected void ImageButtonAddUser_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("UserManagement.aspx");
        }

        protected void ImageButtonProgramCreation_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void ImageButtonApplications_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("ChangeApplicationManagement.aspx");
        }
```

```
        protected void ImageButtonGoToMenu_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("MainMenu.aspx");
        }


    }
}
```

3.3.2 Κώδικας HTML

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="EditUsers.aspx.cs" Inherits="WorkStation_V1.EditUsers" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title></title>
</head>
<body>
    <form id="EditUsers" runat="server" style="text-align: center">
        <div style="border-style: double">
            <span lang="el"
                style="font-family: Calibri; font-size: xx-large; font-
weight: bold; font-style: normal; clip: rect(auto, auto, auto, auto); text-
align: center;">
            Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων</span><br
/>
            <span lang="el"
                style="font-family: Calibri; font-size: x-large; font-
weight: bold; font-style: normal; text-align: center;">
            <br />
            Κεντρική Εφαρμογή Work Station- Υποσύστημα Διαχείρισης
Δεδομένων</span></div>
            <table>
        <tr>
        <td><span lang="el">Κεντρικό Μενού</span></td>
        <td>
            <asp:ImageButton ID="ImageButton1" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
        <td><span lang="el">Χρήστες</span></td>
        <td>
            <asp:ImageButton ID="ImageButton2" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
            <td>
                <span lang="el">Επεξεργασία Δεδομένων Χρηστών</span></td>
        </tr>
        </table>
    <p>
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        Επεξεργασία Δεδομένων Χρηστών </span>
        </p>
```

```
        <p style="border-bottom-style: solid"> </p>
        <p style="font-size: medium; text-align: center; font-weight:
bold;">
        <span lang="el">Αναζήτηση Χρήστη</span></p>

    <table >
    <tr>
    <td>Ομάδα Χρηστών<br />
        <asp:DropDownList ID="DropDownListUTeam" runat="server"
            DataSourceID="Sum_User_Teams" DataTextField="Onoma_Omadas"
            DataValueField="Onoma_Omadas" >
            <asp:ListItem>Iatriko Prosopiko</asp:ListItem>
            <asp:ListItem>Nosileutiko Prosopiko</asp:ListItem>
            <asp:ListItem>Doiikitiko Prosopiko</asp:ListItem>
            <asp:ListItem Selected="True"> </asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxUserTeam" runat="server" Text="Όλες οι
Ομάδες" />
        </td>
    <td>Τμήμα<br />
        <span lang="en-us">
        <asp:DropDownList ID="DropDownLDepartment" runat="server"
            DataSourceID="Sum_Departments" DataTextField="Onoma_Tmimatos"
            DataValueField="Onoma_Tmimatos" >
            <asp:ListItem>Οτιδήποτε</asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxDepartment" runat="server" Text="Όλα τα
τμήματα" />
        </span>
        </td>
    <td>Βαθμίδα<br />
        <span lang="en-us">
        <asp:DropDownList ID="DropDownLBathmida" runat="server"
            DataSourceID="Sum_Bathmides" DataTextField="Onoma_Bathmidas"
            DataValueField="Onoma_Bathmidas">
            <asp:ListItem>Οτιδήποτε</asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxBathmida" runat="server" Text="Όλες οι
Βαθμίδες" />
        </span>
        </td>
    <td>Ειδικότητα<br />
        <span lang="en-us">
        <asp:DropDownList ID="DropDownLEidikothta" runat="server"
            DataSourceID="SumEidikothtes" DataTextField="Eidikothta_Name"
            DataValueField="Eidikothta_Name">
            <asp:ListItem>Οτιδήποτε</asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxEidikothta" runat="server"
            Text="Όλες οι Ειδικότητες" />
        </span>
        </td>

    <td align="center" style="border-style: solid">
        <asp:ImageButton ID="ImageButtonCreation" runat="server"
            ImageUrl="~/Icons/PatientData.png"
onclick="ImageButtonCreation_Click" />
```

```
        <br />
        <span lang="el" style="font-size: medium; text-align:
center">Αναζήτηση Χρήστη</span></td>

    </tr>

    </table>
    <p>
        <asp:GridView ID="Specific_Users_List" runat="server"

onselectedindexchanged="Specific_Users_List_SelectedIndexChanged"
            Caption="Κατάλογος Χρηστών">
            <Columns>
                <asp:CommandField ShowSelectButton="True" />
            </Columns>
        </asp:GridView>
        </p>
    <p style="border-bottom-style: solid"></p>



        <p style="font-size: medium; font-weight: bold; text-align:
center">
            <span lang="el">Στοιχεία Χρήστη</span></p>

        <table>
        <tr>
        <td>
            <asp:GridView ID="GridViewUser" runat="server">
            </asp:GridView>
            </td>
        </tr>
        </table>
             <p style="font-size: medium; font-weight: bold; text-align:
center; border-bottom-style: solid">
                <span lang="el"></span></p>
                <p style="font-size: medium; font-weight: bold; text-
align: center;">
                <span lang="el">Διευθύνσεις Χρήστη</span></p>

                        <table>
        <tr>

            <td>

                <asp:GridView ID="GridViewAd" runat="server">
                    <Columns>
                        <asp:CommandField ShowDeleteButton="True" />
                    </Columns>
                </asp:GridView>

            </td>

        </tr>
        </table>
        <p></p>

        <table>
            <tr><td class="style5"><span lang="el">Διεύθυνση </span></td>
            <td class="style6"><span lang="el">Δρόμος<br />
```

```
                <asp:TextBox ID="TextBoxDrom1"
runat="server"></asp:TextBox>
                </span></td>
            <td><span lang="el">Οροφος<br />
                <asp:TextBox ID="TextBoxDum1" runat="server"></asp:TextBox>
                </span></td>
            <td class="style7">Π<span lang="el">όλη/Χωριό<br />
                <asp:TextBox ID="TextBoxCity1"
runat="server"></asp:TextBox>
                </span></td>
            <td class="style4"><span lang="el">Νομός<br />
                <asp:TextBox ID="TextBoxNom1" runat="server"></asp:TextBox>
                </span></td>
            <td class="style3"><span lang="el">Ταχ. Κώδικας<br />
                <asp:TextBox ID="TextBoxTKK" runat="server"></asp:TextBox>
                </span></td>
                <td><span lang="el">Περιοχή<br />
                    <asp:TextBox ID="TextBoxLoc1"
runat="server"></asp:TextBox>
                    </span></td>
                <td><span lang="el">Χώρα<br />
                    <asp:TextBox ID="TextBoxCun1"
runat="server"></asp:TextBox>
                    </span></td>
                    <td>
                        <br />
                        <asp:Button ID="ButtonDieuth1" runat="server"
Text="OK"
                            onclick="ButtonDieuth1_Click" />
                </td>
            </tr>
        </table>
        <p style="border-bottom-style: solid"></p>

        <table>
        <tr><td class="style9">

        <table>

        <tr><td class="style2" style="border-right-style: solid"><span
lang="el">Τηλέφωνα Χρήστη:</span><br />
            <table>
            <tr><td>
                <asp:DropDownList ID="DropDownListTelType1" runat="server">
                    <asp:ListItem>Κινητό:</asp:ListItem>
                    <asp:ListItem>Fax:</asp:ListItem>
                    <asp:ListItem>Σταθερό:</asp:ListItem>
                </asp:DropDownList>
                </td><td>
                    <asp:TextBox ID="TextBoxTelNum1"
runat="server"></asp:TextBox>
                </td>
                <td><span lang="el">
                        <asp:Button ID="ButtonTelOk" runat="server"
Text="OK"
                        onclick="ButtonTelOk_Click" />
                    </td>
                </tr>

                </table>
                <table>
```

```
                <tr>
                <td>
                <asp:GridView ID="GridViewPhones" runat="server">
                <Columns>
                    <asp:CommandField ShowDeleteButton="True" />
                </Columns>
            </asp:GridView>
                </td>
                </tr>
                </table>
        </td>
            <td class="style11" style="border-right-style: solid">

            <span lang="el">
                <br />
                Δήλωση Καθηκόντων:</span><br />
            <table>
            <tr>
            <td>
                    <span lang="el">
                <asp:DropDownList ID="DropDownListDuty1" runat="server"
DataSourceID="DutyType1"
                    DataTextField="Duty_Type_Name"
DataValueField="Duty_Type_Name">
                </asp:DropDownList>
    </span>
                </td>
                <td lang="el">
                    <asp:Button ID="ButtonDuty1" runat="server"
Text="OK"
                    onclick="ButtonDuty1_Click" />
                </td>
                </tr>
                </table>
                <table>
                <tr>
                <td>
                <asp:GridView ID="GridViewDD" runat="server">
                    <Columns>
                        <asp:CommandField ShowDeleteButton="True" />
                    </Columns>
                </asp:GridView>
                </td>
                </tr>
                </table>

            </td>


            <td class="style2">

            <span lang="el">Δήλωση Τοποθεσιών:</span><br />
            <table>
            <tr>
            <td>
                    <span lang="el">
                <asp:DropDownList ID="DropDownListLoc1" runat="server"
DataSourceID="Locations1"
                    DataTextField="Location_Name"
DataValueField="Location_Name">
                </asp:DropDownList>
```

```
    </span>
                </td>
                <td><span lang="el">
                        <asp:Button ID="ButtonLoc1" runat="server"
Text="OK"
                        onclick="ButtonLoc1_Click" />
                    </td>
                </tr>

            </table>


            <table>
                <tr>
                <td>
                  <asp:GridView ID="GridViewDL" runat="server">
                    <Columns>
                        <asp:CommandField ShowDeleteButton="True" />
                    </Columns>
                </asp:GridView>
                </td>
                </tr>
                </table>
            </td>
            </tr>
        </table>
        </table>
        <p style="font-size: medium; font-weight: bold; text-align:
left;"el">
            Μετάβαση σε</span></p>
    <table><tr><td align="center" style="border-style: solid">
        <asp:ImageButton ID="ImageButtonProgramCreation" runat="server"
            ImageUrl="~/Icons/ChronologicalReview.png"
            onclick="ImageButtonProgramCreation_Click" />
        <br />
        <span lang="el">Δημιουργία Προγράμματος</span></td>
        <td></td>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonAddUser" runat="server"
ImageAlign="Middle"
                ImageUrl="~/Icons/new-user-group-icone-6256-128.png"
                onclick="ImageButtonAddUser_Click" />
            <br />
            <span lang="el">Εισαγωγή νέου χρήστη </span>
        </td>
        <td></td>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonApplications" runat="server"
                ImageUrl="~/Icons/Application.png"
onclick="ImageButtonApplications_Click" />
            <br />
            <span lang="el">Διαχείριση Αιτήσεων</span></td>
            <td>
            </td>

            <td align="center" style="border-style: solid">
                <asp:ImageButton ID="ImageButtonGoToMenu" runat="server"
                    ImageUrl="~/Icons/office-icone-8240-128.png"
                    onclick="ImageButtonGoToMenu_Click" />
                <br />
                <span lang="el">Κεντρικό Μενού</span></td>
```

```
        </tr></table>


        <span lang="en-us">
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />
        <br />

    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT * FROM [Program]" ></asp:SqlDataSource>
        <asp:SqlDataSource ID="Sum_User_Teams" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Omadas] FROM [Sum_User_Teams]">
    </asp:SqlDataSource>
        <asp:SqlDataSource ID="Sum_Departments" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Tmimatos] FROM [Sum_Departments]">
    </asp:SqlDataSource>
        <br />
    <asp:SqlDataSource ID="Sum_Bathmides" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Bathmidas] FROM [Bathmides]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="SumEidikothtes" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Eidikothta_Name] FROM [Eidikothtes]">
    </asp:SqlDataSource>
        <br />
    <span lang="el">
    <asp:SqlDataSource ID="DutyType1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Duty_Type_Name] FROM [Sum_Duties]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="Locations1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Location_Name] FROM
[Locations]"></asp:SqlDataSource>
        </span>
```

```
    </form>
</body>
</html>
```

3.4 Φόρμα InitProgramCreation.aspx.cs

3.4.1 Κώδικας C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.IO;
using System.Web.Services;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using System.Xml.Serialization;
using System.Text;
using System.Collections;
using System.ComponentModel;
using System.Data.OleDb;
using System.Drawing;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

namespace WorkStation_V1
{
    public partial class ProgramDisplay : System.Web.UI.Page
    {
        public CurrentDataSet CurrDS = new CurrentDataSet();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (Session["NewProgramName"] != null)
            {
                string NewProgName = (string)Session["NewProgramName"];
                TextBoxProgName.Text = NewProgName;

            }
            if (Session["myDataSet"] != null)
            {
                //χρήση της session μεταβλητής
                //με την λίστα των εργαζομένων
                //για τους οποίους προορίζεται
                //το νέο πρόγραμμα
                DataSet datasetCreate = new DataSet();
                datasetCreate = (DataSet)Session["myDataSet"];

                //δημιουργία ίδιας session μεταβλητής που θα
                //χρησιμοποιηθεί κατά την επαναφώρτωση της φόρμας
                Session["myDataSet"] = datasetCreate;
```

```
                Session["myDataSet"] = datasetCreate;
                Session["NewProgramName"] = TextBoxProgName.Text;

                CurrDS.SetDS(datasetCreate);

                DataTable myDataTable = datasetCreate.Tables[0];

                Specific_Users_List.DataSource = myDataTable;
                Specific_Users_List.DataBind();
            }
            if ((DropDownListUserTeam.Text == "Νοσηλ. Πρωσοπικό") ||
(DropDownListUserTeam.Text == "Διοικ. Πρωσοπικό"))
            {
                DropDownListEidikothta.Visible = false;
            }
        }


        public DataSet GiveDS()
        {
            DataSet CurrentDataSet = new DataSet();
            return CurrDS.GetDS();
        }

        protected void ImageButtonCreation_Click(object sender,
ImageClickEventArgs e)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";

            string sSQL = "select name+' '+surname AS Όνομα,Department AS
Τμήμα,Thesi AS Θέση,Eidikothta AS Ειδικότητα,Bathmida AS Βαθμίδα,User_ID
from Users";

            if (CheckBoxUserTeam.Checked == CheckBoxDepartment.Checked ==
CheckBoxEidikothta.Checked == CheckBoxBathmida.Checked == true)
            {
                sSQL = "select name+ ' ' +surname AS Όνομα,User_Team AS
Ομάδα_Εργασίας,Department AS Τμήμα,Thesi AS Θέση,Eidikothta AS
Ειδικότητα,Bathmida AS Βαθμίδα,User_ID from Users";
            }

            Session["CurrentQuery"] = sSQL;
            string newquery = (string)Session["CurrentQuery"];

            if (CheckBoxUserTeam.Checked == true)
            {
                DropDownListUserTeam.Enabled = false;
            }
            else
            {

                newquery = string.Format("{0} where User_Team=N'{1}'",
newquery, DropDownListUserTeam.Text);

            }

            if (CheckBoxDepartment.Checked == true)
            {
                DropDownListDepartment.Enabled = false;
```

```
            }
            else
            {
                if (CheckBoxUserTeam.Checked == true)
                {
                    newquery = string.Format("{0} WHERE Department=N'{1}'",
newquery, DropDownListDepartment.Text);

                }
                else
                {
                    newquery = string.Format("{0} AND Department=N'{1}'",
newquery, DropDownListDepartment.Text);

                }
            }


            if (CheckBoxBathmida.Checked == true)
            {
                DropDownListBathmida.Enabled = false;
            }
            else
            {
                if ((CheckBoxUserTeam.Checked == true) &&
(CheckBoxDepartment.Checked == true))
                {
                    newquery = string.Format("{0} WHERE Bathmida=N'{1}'",
newquery, DropDownListBathmida.Text);

                }
                else
                {
                    newquery = string.Format("{0} AND Bathmida=N'{1}'",
newquery, DropDownListBathmida.Text);

                }
            }

            if (CheckBoxEidikothta.Checked == true)
            {
                DropDownListEidikothta.Enabled = false;
            }
            else
            {
                if ((CheckBoxUserTeam.Checked == true) &&
(CheckBoxDepartment.Checked == true) && (CheckBoxBathmida.Checked == true))
                {
                    //newquery = (string)Session["CurrentQuery"];
                    newquery = string.Format("{0} WHERE Eidikothta=N'{1}'",
newquery, DropDownListEidikothta.Text);
                    //Session["CurrentQuery"] = newquery;
                }
                else
                {
                    //newquery = (string)Session["CurrentQuery"];
                    newquery = string.Format("{0} AND Eidikothta=N'{1}'",
newquery, DropDownListEidikothta.Text);
                    //Session["CurrentQuery"] = newquery;
                }
            }
```

```
            Session["CurrentQuery"] = newquery;


System.Diagnostics.Debug.Write((string)Session["CurrentQuery"]);
            SqlConnection conn = new SqlConnection(sConn2);
            DataSet dsCreate = new DataSet();
            SqlDataAdapter daCreate = new
SqlDataAdapter((string)Session["CurrentQuery"], sConn2);
            daCreate.Fill(dsCreate, "Users");

            Session["myDataSet"] = dsCreate;
            Session["NewProgramName"] = TextBoxProgName.Text;

            CurrDS.SetDS(dsCreate);

            DataTable myDataTable = dsCreate.Tables[0];
            Specific_Users_List.DataSource = myDataTable;
            Specific_Users_List.DataBind();
        }

        protected void ImageButtonClear_Click(object sender,
ImageClickEventArgs e)
        {
            Session["NewProgramName"] = null;
            Session["CurrentQuery"] = null;
            Session["myDataSet"] = null;

            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void ImageButtonContinue_Click(object sender,
ImageClickEventArgs e)
        {
            ContinueProgCreation formlog = new ContinueProgCreation();
            Session["NewProgramName"] = TextBoxProgName.Text;

            Response.Redirect("ContinueProgCreation.aspx");
        }

        protected void ImageButtonAddUser_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("UserManagement.aspx");
        }

        protected void ImageButtonApplications_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("ChangeApplicationManagement.aspx");
        }

        protected void DropDownListUserTeam_SelectedIndexChanged(object
sender, EventArgs e)
        {

        }

        protected void ImageButton2_Click(object sender,
ImageClickEventArgs e)
        {
```

```
            Response.Redirect("EditUsers.aspx");
        }

        protected void ImageButtonGoToMenu_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("MainMenu.aspx");
        }




    }

    public class CurrentDataSet
    {
        public DataSet CurrentDs;

        public void SetDS(DataSet CDS)
        {
            CurrentDs = CDS;
        }

        public DataSet GetDS()
        {
            return CurrentDs;
        }
    }
}
```

3.4.2 Κώδικας HTML

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="InitProgramCreation.aspx.cs"
Inherits="WorkStation_V1.ProgramDisplay" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title></title>
    <style type="text/css">
        .style1
        {
            height: 22px;
        }
        </style>
</head>
<body>
    <form id="form1" runat="server">
        <div style="border-style: double; text-align: center;">

        <span lang="el"
            style="font-family: Calibri; font-size: xx-large; font-weight:
bold; font-style: normal; clip: rect(auto, auto, auto, auto); text-align:
center;">
        Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων</span><br />
        <span lang="el"
```

```
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        <br />
 Κεντρική Εφαρμογή Work Station-
            Υποσύστημα Διαχείρισης Δεδομένων</span></div>
            <table>
        <tr>
        <td><span lang="el">Κεντρικό Μενού</span></td>
        <td>
            <asp:ImageButton ID="ImageButton1" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
        <td><span lang="el">Δημιουργία Προγράμματος</span></td>
        <td>
            <asp:ImageButton ID="ImageButton2" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
            <td>
                <span lang="el">Ορισμός Συμμετεχόντων</span></td>
        </tr>
        </table>
    <p style="text-align: center">
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        Δημιουργία Προγραμμάτων-Ορισμός Συμμετεχόντων </span></p>
    <p style="border-bottom-style: solid">
         </p>
    <p class="style1">
        <span lang="en-us">   </span></p>


    <table>
    <tr>
    <td>Ομάδα Χρηστών<br />
        <asp:DropDownList ID="DropDownListUserTeam" runat="server"
            DataSourceID="Sum_User_Teams" DataTextField="Onoma_Omadas"
            DataValueField="Onoma_Omadas"

onselectedindexchanged="DropDownListUserTeam_SelectedIndexChanged">
            <asp:ListItem>Iatriko Prosopiko</asp:ListItem>
            <asp:ListItem>Nosileutiko Prosopiko</asp:ListItem>
            <asp:ListItem>Doiikitiko Prosopiko</asp:ListItem>
            <asp:ListItem Selected="True"> </asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxUserTeam" runat="server" Text="Όλες οι
Ομάδες" />
        </td>
    <td>Τμήμα<br />
        <span lang="en-us">
        <asp:DropDownList ID="DropDownListDepartment" runat="server"
            DataSourceID="Sum_Departments" DataTextField="Onoma_Tmimatos"
            DataValueField="Onoma_Tmimatos">
            <asp:ListItem>Οτιδήποτε</asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxDepartment" runat="server" Text="Όλα τα
τμήματα" />
        </span>
        </td>
```

```
    <td>Βαθμίδα<br />
        <span lang="en-us">
        <asp:DropDownList ID="DropDownListBathmida" runat="server"
            DataSourceID="Sum_Bathmides" DataTextField="Onoma_Bathmidas"
            DataValueField="Onoma_Bathmidas">
            <asp:ListItem>Οτιδήποτε</asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxBathmida" runat="server" Text="Όλες οι
Βαθμίδες" />
        </span>
        </td>
    <td>Ειδικότητα<br />
        <span lang="en-us">
        <asp:DropDownList ID="DropDownListEidikothta" runat="server"
            DataSourceID="SumEidikothtes" DataTextField="Eidikothta_Name"
            DataValueField="Eidikothta_Name">
            <asp:ListItem>Οτιδήποτε</asp:ListItem>
        </asp:DropDownList>
        <br />
        <asp:CheckBox ID="CheckBoxEidikothta" runat="server"
            Text="Όλες οι Ειδικότητες" />
        </span>
        </td>
    <td>Όνομα Προγράμματος<br />
        <span lang="en-us">
        <asp:TextBox ID="TextBoxProgName" runat="server"
Width="207px">Όνομα Προγράμματος</asp:TextBox>
        <br />
        <table><tr><td> </td></tr></table>

        </span>
        </td>
    <td align="center" style="border-style: solid">
        <asp:ImageButton ID="ImageButtonCreation" runat="server"
            ImageUrl="~/Icons/PatientData.png"
onclick="ImageButtonCreation_Click" />
        <br />
        <span lang="el">Ορισμός Συμμετεχόντων</span></td>

    </tr>


    </table>


    <p>
        <span lang="en-us">
        <asp:GridView ID="Specific_Users_List" runat="server"
            Caption="Κατάλογος Χρηστών">
        </asp:GridView>
        </span>
        <br /></p>
        <p style=" border-bottom-style: solid"></p>
        <p style="font-size: medium; font-weight: bold"><span
lang="el">Γραμμή Εργαλείων</span></p>
        <table>
        <tr>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonContinue" runat="server"
                ImageUrl="~/Icons/ChronologicalReview.png"
```

```
                onclick="ImageButtonContinue_Click" />
            <br />
            <span lang="el">Συνέχεια-Δημιουργία Προγράμματος</span>
            </td>
        <td></td>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonClear" runat="server"
                ImageUrl="~/Icons/out-session-icone-6378-128.png"
                onclick="ImageButtonClear_Click" />
            <br />
            <span lang="el">Εκαθάριση Φόρμας</span></td>

        </tr>
        </table>
        <p style=" border-top-style: solid"></p>
        <p style="font-size: medium; font-weight: bold"><span
lang="el">Μετάβαση σε</span></p>

    <table>
    <tr>
    <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonAddUser" runat="server"
ImageAlign="Middle"
                ImageUrl="~/Icons/new-user-group-icone-6256-128.png"
                onclick="ImageButtonAddUser_Click" />
            <br />
            <span lang="el">Εισαγωγή νέου χρήστη </span>
        </td>
    <td>
    </td>
    <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonApplications" runat="server"
                ImageUrl="~/Icons/Application.png"
onclick="ImageButtonApplications_Click" />
            <br />
            <span lang="el">Διαχείριση Αιτήσεων</span>
            </td>
            <td></td>
            <td style="text-align: center; border-style: solid">
                <asp:ImageButton ID="ImageButton3" runat="server"
                    ImageUrl="~/Icons/edit-find-replace-old-icone-8806-
64.png"
                    onclick="ImageButton2_Click" />
                <br />
                <span lang="el">Επεξεργασία Χρηστών</span></td>
            <td></td>

    <td align="center" style="border-style: solid">
                <asp:ImageButton ID="ImageButtonGoToMenu" runat="server"
                    ImageUrl="~/Icons/office-icone-8240-128.png"
                    onclick="ImageButtonGoToMenu_Click" />
                <br />
                <span lang="el">Κεντρικό Μενού</span></td>
    </tr>
    </table>



    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
```

```
        SelectCommand="SELECT * FROM [Program]"></asp:SqlDataSource>
        <asp:SqlDataSource ID="Sum_User_Teams" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Omadas] FROM [Sum_User_Teams]">
    </asp:SqlDataSource>
        <span lang="en-us">
        <asp:SqlDataSource ID="Sum_Departments" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Tmimatos] FROM [Sum_Departments]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="Sum_Bathmides" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Onoma_Bathmidas] FROM [Bathmides]">
    </asp:SqlDataSource>
    <asp:SqlDataSource ID="SumEidikothtes" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString
%>"
        SelectCommand="SELECT [Eidikothta_Name] FROM [Eidikothtes]">
    </asp:SqlDataSource>
        </span>
    </form>
</body>
</html>
```

3.5 Φόρμα ContinueProgCreation.aspx.cs

3.5.1 Κώδικας C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.IO;
using System.Web.Services;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using System.Xml.Serialization;
using System.Text;
using System.Collections;
using System.ComponentModel;
using System.Data.OleDb;
using System.Drawing;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.Xml.Linq;



namespace WorkStation_V1
{
```

```
    public partial class ContinueProgCreation : System.Web.UI.Page
    {
        public NewProgramData NewProgram = new NewProgramData();
        public CurrentNameDataSet CurrDataSet = new CurrentNameDataSet();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!Page.IsPostBack)
            {

                if (Session["NewProgramName"] != null)
                {
                    //αποθήκευση του ονόματος που έχει
                    //το νέο πρόγραμμα στην κλάση
                    //προκείμένου να το αποθηκεύσουμε
                    //στην συνέχεια στην ΚΒΔ

NewProgram.SetProgramName((string)Session["NewProgramName"]);

                    //δημιουργία ίδιας session μεταβλητής που θα
                    //χρησιμοποιηθεί κατά την επαναφώρτωση της φόρμας
                    Session["NewProgramName"] =
NewProgram.GetProgramName();

                    try
                    {
                        //δημιουργία πίνακα προεπισκόπησης
                        //νέου προγράμματος
                        string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
                        SqlConnection conn = new SqlConnection(sConn2);
                        conn.Open();
                        string cmd2 = string.Format("select * from Program
where Program_Name='{0}'", NewProgram.GetProgramName());
                        DataSet dsNewProg = new DataSet();
                        SqlDataAdapter daNewProg = new SqlDataAdapter(cmd2,
sConn2);
                        daNewProg.Fill(dsNewProg, "Program");
                        DataTable myDataTable = dsNewProg.Tables[0];
                        GridViewPreviewProg.DataSource = myDataTable;
                        GridViewPreviewProg.DataBind();
                    }
                    catch
                    {

                    }
                }

                if (Session["myDataSet"] != null)
                {
                    //χρήση της session μεταβλητής
                    //με την λίστα των εργαζομένων
                    //για τους οποίους προορίζεται
                    //το νέο πρόγραμμα
                    DataSet datasetCreate = new DataSet();
                    datasetCreate = (DataSet)Session["myDataSet"];
                    CurrDataSet.SetDS(datasetCreate);

                    //δημιουργία ίδιας session μεταβλητής που θα
                    //χρησιμοποιηθεί κατά την επαναφώρτωση της φόρμας
                    Session["myDataSet"] = datasetCreate;
```

```
                    //φόρτωση του πλαισίου λίστας με τα
                    //ονόματα των εργαζομένων
                    DropDownListNames.DataSource = datasetCreate.Tables[0];
                    DropDownListNames.DataValueField = "User_ID";
                    DropDownListNames.DataTextField = "Όνομα";
                    DropDownListNames.DataBind();


                    string testq = string.Format("Όνομα='{0}'",
DropDownListNames.Text);
                    datasetCreate.Tables[0].Constraints.Add("pk_User_ID",
datasetCreate.Tables[0].Columns[0], true);
                    DataRow drow =
datasetCreate.Tables[0].Rows.Find(DropDownListNames.Text);
                    //TextBoxTest.Text = drow[0].ToString() + "'" +
drow[1].ToString();
                }
            }
        }

        protected void ButtonDutyCon_Click(object sender, EventArgs e)
        {
            NewProgram.SetDuty(DropDownListDuty.Text);
        }

        protected void ButtonSTCon_Click(object sender, EventArgs e)
        {
            NewProgram.SetSHour(TextBoxStartTime.Text);
        }

        protected void ButtonETCon_Click(object sender, EventArgs e)
        {
            NewProgram.SetEHour(TextBoxEndTime.Text);
        }

        protected void ButtonLocCon_Click(object sender, EventArgs e)
        {
            NewProgram.SetLocation(DropDownListLocation.Text);
        }

        protected void ButtonDateCon_Click(object sender, EventArgs e)
        {

            NewProgram.SetNDate(TextBoxCalendar.Text);
        }

        protected void ButtonSave_Click(object sender, ImageClickEventArgs
e)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "INSERT INTO Program
(Date,Duty_Type,Duty_Start_Time,Duty_End_Time,Location,User_ID,Program_Name
) values
(@Date,@Duty_Type,@Duty_Start_Time,@Duty_End_Time,@Location,@User_ID,@Progr
am_Name)";
            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            try
```

```
            {
                SqlCommand SqlCmd = new SqlCommand(cmd, conn);
                SqlCmd.Parameters.AddWithValue("@Date",
TextBoxCalendar.Text);
                SqlCmd.Parameters.AddWithValue("@Duty_Type",
DropDownListDuty.Text);
                SqlCmd.Parameters.AddWithValue("@Duty_Start_Time",
TextBoxStartTime.Text);
                SqlCmd.Parameters.AddWithValue("@Duty_End_Time",
TextBoxEndTime.Text);
                SqlCmd.Parameters.AddWithValue("@Location",
DropDownListLocation.Text);
                SqlCmd.Parameters.AddWithValue("@User_ID",
DropDownListNames.SelectedValue);
                SqlCmd.Parameters.AddWithValue("@Program_Name",
(string)Session["NewProgramName"]);
                SqlCmd.ExecuteNonQuery();
            }

            catch
            {
                Console.Write("ο χρήστης που προσπαθείται να εισάγεται
υπάρχει ήδη");
            }

            string cmd2 = string.Format("select U.name +' '+U.surname AS
Ονοματεπώνυμο,C.* from Users U inner join"
            +" (select p.Date AS Ημερομηνία,p.Duty_Type AS
Εργασία,p.Duty_Start_Time AS Έναρξη,p.Duty_End_Time AS Λήξη,p.Location AS
Τοποθεσία,p.Program_Name AS Όνομα_Προγρ,p.User_ID from Program p where
Program_Name=N'{0}')"
            +" C on C.User_ID=U.User_ID",
(string)Session["NewProgramName"]);
            DataSet dsNewProg = new DataSet();
            SqlDataAdapter daNewProg = new SqlDataAdapter(cmd2, sConn2);
            daNewProg.Fill(dsNewProg, "Program");

            DataTable myDataTable = dsNewProg.Tables[0];
            GridViewPreviewProg.DataSource = myDataTable;
            GridViewPreviewProg.DataBind();


            string cmdItem = "INSERT INTO Item
(Item_ID,Creator_ID,Receiver_ID) values
(@Item_ID,@Creator_ID,@Receiver_ID)";
            SqlCommand SqlCmdItem = new SqlCommand(cmdItem, conn);
            SqlCmdItem.Parameters.AddWithValue("@Item_ID", ReturnID());
            SqlCmdItem.Parameters.AddWithValue("@Creator_ID", "18");
            SqlCmdItem.Parameters.AddWithValue("@Receiver_ID",
DropDownListNames.SelectedValue);
            SqlCmdItem.ExecuteNonQuery();
        }

        public string ReturnID()
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd3 = "Select Program_ID From Program Where Date=@Date
AND Duty_Type=@Duty_Type AND Duty_Start_Time=@Duty_Start_Time AND
Duty_End_Time=@Duty_End_Time AND Location=@Location AND
Program_Name=@Program_Name";
```

```
            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            System.Data.SqlClient.SqlCommand SqlCmd3 = new SqlCommand(cmd3,
conn);
            SqlCmd3.Parameters.AddWithValue("@Date", TextBoxCalendar.Text);
            SqlCmd3.Parameters.AddWithValue("@Duty_Type",
DropDownListDuty.Text);
            SqlCmd3.Parameters.AddWithValue("@Duty_Start_Time",
TextBoxStartTime.Text);
            SqlCmd3.Parameters.AddWithValue("@Duty_End_Time",
TextBoxEndTime.Text);
            SqlCmd3.Parameters.AddWithValue("@Location",
DropDownListLocation.Text);
            SqlCmd3.Parameters.AddWithValue("@Program_Name",
(string)Session["NewProgramName"]);
            SqlCmd3.ExecuteNonQuery();
            System.Data.SqlClient.SqlDataReader dr =
SqlCmd3.ExecuteReader();
            dr.Read();

            string userid = dr["Program_ID"].ToString();

            dr.Close();

            return userid;

        }



        protected void ButtonNewProgram_Click(object sender, EventArgs e)
        {
            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void Calendar1_SelectionChanged(object sender, EventArgs
e)
        {
            TextBoxCalendar.Text =
Calendar1.SelectedDate.ToShortDateString();
        }


        protected void ButtonNameCon_Click(object sender, EventArgs e)
        {
            DataSet nameds = new DataSet();
            nameds = (DataSet)Session["myDataSet"];

            DataTable ProgUsers = new DataTable();
            ProgUsers = nameds.Tables["Users"];

            //IQueryable<>

            var query = from o in ProgUsers.AsEnumerable() where
o.Field<string>("Όνομα") == DropDownListNames.Text select new { User_ID =
o.Field<int>("User_ID") };

            GridViewUserId.DataSource = query;
            GridViewUserId.DataBind();
```

```
            //TextBoxTest.Text = GridViewUserId.Rows[0].Cells[0].Text;

           NewProgram.SetIDs(DropDownListNames.SelectedValue.ToString());

System.Diagnostics.Debug.WriteLine(DropDownListNames.SelectedValue);
        }

        protected void ImageButtonProgramCreation_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void ImageButtonReturn_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void ImageButtonCancel_Click(object sender,
ImageClickEventArgs e)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string cmd = "DELETE FROM Program WHERE
Program_Name='@Program_Name'";
            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();
            try
            {
                SqlCommand SqlCmd = new SqlCommand(cmd, conn);
                SqlCmd.Parameters.AddWithValue("@Program_Name",
NewProgram.GetProgramName());
                SqlCmd.ExecuteNonQuery();
            }
            catch
            {
                Console.Write("παρουσιάστηκε σφάλμα κατά την διαγραφή");
            }
        }

        protected void ImageButtonAddUser_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("UserManagement.aspx");
        }

        protected void ImageButtonApplications_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("ChangeApplicationManagement.aspx");
        }

        protected void ImageButtonGoToMenu_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("MainMenu.aspx");
        }

        protected void ImageButtonAddProgUser_Click(object sender,
ImageClickEventArgs e)
```

```
        {
            TextBoxCalendar.Text = "";
            TextBoxEndTime.Text = "";
            TextBoxStartTime.Text = "";
        }

        protected void ImageButton2_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("EditUsers.aspx");
        }

    }

    public class NewProgramData
    {
        public string NewStartHour = "0";
        public string NewEndHour = "0";
        public string NewDate = "0";
        public string UserID = "0";
        public string Duty = "";
        public string ProgramName = "";
        public string Location = "";

        public void SetSHour(string SHour)
        {
            NewStartHour = SHour;
        }

        public void SetEHour(string EHour)
        {
            NewEndHour = EHour;
        }

        public void SetNDate(string NDate)
        {
            NewDate = NDate;
        }

        public void SetIDs(string User_ID)
        {
            UserID = User_ID;

        }
        public void SetDuty(string DutyType)
        {
            Duty = DutyType;
        }
        public void SetProgramName(string NewProgName)
        {
            ProgramName = NewProgName;
        }
        public void SetLocation(string Loc)
        {
            Location = Loc;
        }
        public string GetSHour()
        {
            return NewStartHour;
        }
```

```
            public string GetEHour()
            {
                return NewEndHour;
            }

            public string GetNDate()
            {
                return NewDate;
            }

            public string GetUID()
            {
                return UserID;
            }
            public string GetDuty()
            {
                return Duty;
            }
            public string GetProgramName()
            {
                return ProgramName;
            }
            public string GetLocation()
            {
                return Location;
            }

        }

        public class CurrentNameDataSet
        {
            public DataSet CurrentDataSet;

            public void SetDS(DataSet CDS)
            {
                CurrentDataSet = CDS;
            }

            public DataSet GetDS()
            {
                return CurrentDataSet;
            }
        }
    }
```

3.5.2 Κώδικας HTML

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="ContinueProgCreation.aspx.cs"
Inherits="WorkStation_V1.ContinueProgCreation" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
```

```
      <div style="border-style: double; text-align: center;">

        <span lang="el"
            style="font-family: Calibri; font-size: xx-large; font-weight: bold; font-
style: normal; clip: rect(auto, auto, auto, auto); text-align: center;">
        Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων</span><br />
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold; font-style:
normal; text-align: center;">
        <br />
 Κεντρική Εφαρμογή Work Station-
            Υποσύστημα Διαχείρισης Δεδομένων</span></div>
            <table>
        <tr>
        <td><span lang="el">Κεντρικό Μενού</span></td>
        <td>
            <asp:ImageButton ID="ImageButton2" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
        <td><span lang="el">Δημιουργία Προγράμματος</span></td>
        <td>
            <asp:ImageButton ID="ImageButton3" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
            <td>
                <span lang="el">Ορισμός Συμμετεχόντων</span></td>
                <td>
            <asp:ImageButton ID="ImageButton4" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
            <td>
                <span lang="el">Προσωποποίηση Καθηκόντων</span></td>
        </tr>
        </table>
    <p style="text-align: center">
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold; font-style:
normal; text-align: center;">
        Δημιουργία Προγραμμάτων-Προσωποποίηση Καθηκόντων</span></p>
    <p style="border-bottom-style: solid">
         </p>
            <table>
            <tr>
            <td>

        <span lang="el"><asp:Label ID="Label7" runat="server"
Text="Ονοματεπώνυμο"></asp:Label>
                <br />

        <asp:DropDownList ID="DropDownListNames" runat="server">
        </asp:DropDownList>
    </span></td>

            <td><span lang="el">
        <asp:Label ID="Label8" runat="server" Text="Τύπος Εργασίας"></asp:Label>
                <br />
                <asp:DropDownList ID="DropDownListDuty" runat="server"
DataSourceID="Sum_Duties"
                    DataTextField="Duty_Type_Name" DataValueField="Duty_Type_Name">
                </asp:DropDownList>
    </span></td>
```

```
    <td>
         </td>
            <td><span lang="el">
        <asp:Label ID="Label9" runat="server" Text="Ώρα Έναρξης"></asp:Label>
                <br />
        <asp:TextBox ID="TextBoxStartTime" runat="server"></asp:TextBox>
    </span></td>
    <td>
         </td>
            <td><span lang="en-us">
        <asp:Label ID="Label10" runat="server" Text="Ώρα Λήξης"></asp:Label>
                <br />
                <span lang="el">
        <asp:TextBox ID="TextBoxEndTime" runat="server"></asp:TextBox>
        </span>
    </span>

                </td>
                <td>
                     </td>
            <td><span lang="en-us">
        <asp:Label ID="Label11" runat="server" Text="Τοποθεσία"></asp:Label>
                <br />
                <span lang="el">
        <asp:DropDownList ID="DropDownListLocation" runat="server"
DataSourceID="Locations"
                    DataTextField="Location_Name" DataValueField="Location_Name">
        </asp:DropDownList>
        </span>
    </span>
                </td>
                <td>
                     </td>
            <td></td>
            </tr>
            <tr>
            <td></td>
            </tr>
            </table>
            <table>
            <tr>
            <td><span lang="el">Ημερομηνία</span><asp:Calendar ID="Calendar1"
runat="server" onselectionchanged="Calendar1_SelectionChanged"
                    ></asp:Calendar>
                </td>
                <td>

                    <asp:TextBox ID="TextBoxCalendar" runat="server"></asp:TextBox>
                    <br />
                </td>
                <td style="text-align: center">
                    <table>
                    <tr>
                    <td style="border-style: solid">
                    <asp:ImageButton ID="ImageButton1" runat="server"
                        ImageUrl="~/Icons/disk-pen-register-record-in-writing-icone-9080-
64.png"
                        onclick="ButtonSave_Click" BorderStyle="Solid" />
                    <br />
                    <span lang="el">Αποθήκευση</span>
                    </td>
```

```
                </tr>
                <tr>
                <td style="text-align: center; border-style: solid">
                <asp:ImageButton ID="ImageButtonProgramCreation" runat="server"
            ImageUrl="~/Icons/ChronologicalReview.png"
            onclick="ImageButtonProgramCreation_Click" />
        <br />
        <span lang="el">Δημιουργία Προγράμματος</span>
                </td>
                </tr>
                </table>
                </td>
                <td style="text-align: center">
                <table>
                <tr>
                <td style="border-style: solid">
                    <asp:ImageButton ID="ImageButtonAddProgUser" runat="server"
                        ImageUrl="~/Icons/add-more-icone-4053-64.png"
                        onclick="ImageButtonAddProgUser_Click" />
                    <br />
                    Εισαγωγή Χρήστη
                    <br />
                    στο τρέχων Πρόγρ.
                </td>
                </tr>
                <tr>
                <td style="text-align: center; border-style: solid">
                <asp:ImageButton ID="ImageButtonReturn" runat="server"
            ImageUrl="~/Icons/first-go-icone-5042-64.png"
            onclick="ImageButtonReturn_Click" />
        <br />
        <span lang="el">Επιστροφή</span>
                </td>
                </tr>
                </table>
                   </td>
            <td style="text-align: center">
                <table>
                <tr>
                <td style="border-style: solid">
                 <asp:ImageButton ID="ImageButtonFinal" runat="server"
                    ImageUrl="~/Icons/release-icone-3911-64.png" BorderStyle="Solid"
/>
                <br />
                Ολοκλήρωση Δ<span lang="el">ιαδικασίας</span>
                </td>
                </tr>
                <tr>
                <td style="text-align: center; border-style: solid">
                    <asp:ImageButton ID="ImageButtonCancel" runat="server"
                        ImageUrl="~/Icons/cancel-remove-icone-5993-64.png"
                        onclick="ImageButtonCancel_Click" />
                    <br />
                    <span lang="el">Ακύρωση Διαδικασίας</span></td>
                </tr>
                </table>
               </td>
        </tr>
        </table>

    <span lang="el">
```

```
        </span>
        <p>
            <asp:GridView ID="GridViewPreviewProg" runat="server">
            </asp:GridView>
            <asp:GridView ID="GridViewUserId" runat="server" Visible="False">
            </asp:GridView>
    </p>
    <p style="border-bottom-style: solid"></p>
    <p style="font-size: medium; font-weight: bold; text-align: left;>

           <span lang="el"> Μετάβαση σε</span></p>
    <table><tr>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonAddUser" runat="server" ImageAlign="Middle"
                ImageUrl="~/Icons/new-user-group-icone-6256-128.png"
                onclick="ImageButtonAddUser_Click" />
            <br />
            <span lang="el">Εισαγωγή νέου χρήστη </span>
        </td>
        <td></td>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonApplications" runat="server"
                ImageUrl="~/Icons/Application.png" onclick="ImageButtonApplications_Click"
/>
            <br />
            <span lang="el">Διαχείριση Αιτήσεων</span></td>
            <td>
            </td>
            <td style="text-align: center; border-style: solid">
                <asp:ImageButton ID="ImageButton5" runat="server"
                    ImageUrl="~/Icons/edit-find-replace-old-icone-8806-64.png"
                    onclick="ImageButton2_Click" />
                <br />
                <span lang="el">Επεξεργασία Χρηστών</span></td>
            <td></td>
            <td align="center" style="border-style: solid">
                <asp:ImageButton ID="ImageButtonGoToMenu" runat="server"
                    ImageUrl="~/Icons/office-icone-8240-128.png"
                    onclick="ImageButtonGoToMenu_Click" />
                <br />
                <span lang="el">Κεντρικό Μενού</span></td>
        </tr></table>

    <span lang="el">
        <asp:SqlDataSource ID="Sum_Duties" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString %>"
        SelectCommand="SELECT [Duty_Type_Name] FROM [Sum_Duties]">
    </asp:SqlDataSource>
    </span>
                <span lang="el">
        <asp:SqlDataSource ID="Locations" runat="server"
        ConnectionString="<%$ ConnectionStrings:SystemDataConnectionString %>"
        SelectCommand="SELECT [Location_Name] FROM [Locations]"></asp:SqlDataSource>
        </span>

    </form>
</body>
</html>
```

3.6 Φόρμα ChangeApplicationManagement.aspx.cs

3.6.1 Κώδικας C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.IO;
using System.Web.Services;
using System.Data;
using System.Configuration;
using System.Data.SqlClient;
using System.Data.Sql;
using System.Xml.Serialization;
using System.Text;
using System.Collections;
using System.ComponentModel;
using System.Data.OleDb;
using System.Drawing;
using System.Web.Security;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

namespace WorkStation_V1
{
    public partial class _Default : System.Web.UI.Page
    {
        public NewProgData NPD = new NewProgData();

        protected void Page_Load(object sender, EventArgs e)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            string sSQL = "select R.Τύπος_Αίτησης,R.Όνομα, R.Επώνυμο,C.* from" +
                          " (select A.[App_Type] AS Τύπος_Αίτησης,A.Program_ID,E.* from
Applications A inner join" +
                          " (select U.name AS Όνομα, U.surname AS Επώνυμο, U.User_ID
from Users U)E on E.User_ID=A.User_ID) R inner join" +
                          " (select p.Program_ID,p.Date AS Ημερομηνία,p.Duty_Type AS
Εργασία,p.Duty_Start_Time AS Έναρξη," +
                          "p.Duty_End_Time AS Λήξη,p.Location AS
Τοποθεσία,p.Program_Name,p.User_ID from Program p inner join Applications l on
l.Program_ID=p.Program_ID)C on C.Program_ID=R.Program_ID";

            SqlConnection conn = new SqlConnection(sConn2);
            DataSet dsApplications = new DataSet();
            SqlDataAdapter daApplications = new SqlDataAdapter(sSQL, sConn2);
            daApplications.Fill(dsApplications, "Applications");
            DataTable myDataTable = dsApplications.Tables[0];
            GridViewApplications.DataSource = myDataTable;
            GridViewApplications.DataBind();
        }


        protected void GridViewApplications_SelectedIndexChanged(object sender,
EventArgs e)
```

```
        {
            GridViewRow row = GridViewApplications.SelectedRow;

            TextBoxApp_Type.Text=row.Cells[1].Text;
            TextBoxProgID.Text=row.Cells[4].Text;
            TextBoxUserID.Text=row.Cells[11].Text;
            TextBoxDt.Text = row.Cells[6].Text;
            TextBoxLocation.Text = row.Cells[9].Text;
            TextBoxProgramName.Text = row.Cells[10].Text;

            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";

            string sSQL = string.Format("select Date AS Ημερομηνία,Duty_Start_Time AS
Ώρα_Έναρξης,Duty_End_Time AS Ώρα_Λήξης from Program where Program_ID='{0}'",
TextBoxProgID.Text);
            string sSQL2 = string.Format("select Date AS Ημερομηνία,Duty_Start_Time AS
Ώρα_Έναρξης,Duty_End_Time AS Ώρα_Λήξης from Modifications where Program_ID='{0}'",
TextBoxProgID.Text);

            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            DataSet dsOldProgram = new DataSet();
            DataSet dsNewProgram = new DataSet();

            SqlDataAdapter daOldProgram = new SqlDataAdapter(sSQL, sConn2);
            SqlDataAdapter daNewProgram = new SqlDataAdapter(sSQL2, sConn2);

            daOldProgram.Fill(dsOldProgram, "Program");
            daNewProgram.Fill(dsNewProgram, "Modifications");

            DataTable myOldProgram = dsOldProgram.Tables[0];
            DataTable myNewProgram = dsNewProgram.Tables[0];

            GridViewCurrentProgram.DataSource = myOldProgram;
            GridViewNewProgram.DataSource = myNewProgram;

            GridViewCurrentProgram.DataBind();
            GridViewNewProgram.DataBind();

            System.Data.SqlClient.SqlCommand SqlCmd3 = new SqlCommand(sSQL2, conn);
            System.Data.SqlClient.SqlDataReader dr = SqlCmd3.ExecuteReader();
            dr.Read();

            TextBoxNewDate.Text=dr["Ημερομηνία"].ToString();
            TextBoxNewStartTime.Text =dr["Ώρα_Έναρξης"].ToString();
            TextBoxNewEndTime.Text=dr["Ώρα_Λήξης"].ToString();
            dr.Close();
        }

        protected void ImageButtonAccept_Click(object sender, ImageClickEventArgs e)
        {
            string sConn2 = "data source=FERRARI-
4000\\SQLEXPRESS;Database=SystemData;User Id=xpchris;Password=123456";
            SqlConnection conn = new SqlConnection(sConn2);
            conn.Open();

            if (TextBoxApp_Type.Text == "Cancel")
            {
```

```
                System.Diagnostics.Debug.Write("mphke sto cancel:" +
TextBoxNewDate.Text + TextBoxNewStartTime.Text + TextBoxNewEndTime.Text +
TextBoxProgID.Text);

                string cmd = "DELETE From Program where Program_ID=@Program_ID";
                System.Data.SqlClient.SqlCommand SqlCmd = new SqlCommand(cmd, conn);
                SqlCmd.Parameters.AddWithValue("@Program_ID", TextBoxProgID.Text);

                string sSQL3 = "DELETE  from Applications where
Program_ID=@Program_ID";
                string sSQL4 = "DELETE  from Modifications where
Program_ID=@Program_ID";

                System.Data.SqlClient.SqlCommand SqlCmd3 = new SqlCommand(sSQL3, conn);
                SqlCmd3.Parameters.AddWithValue("@Program_ID", TextBoxProgID.Text);
                SqlCmd3.ExecuteNonQuery();

                System.Data.SqlClient.SqlCommand SqlCmd4 = new SqlCommand(sSQL4, conn);
                SqlCmd4.Parameters.AddWithValue("@Program_ID", TextBoxProgID.Text);
                SqlCmd4.ExecuteNonQuery();

                TextBoxResult.Text = "Το αίτημά σας καταχωρήθηκαι";
            }
            else
            {
                string cmd2 = "UPDATE Program Set
Date=@Date,Duty_Start_Time=@Duty_Start_Time,Duty_End_Time=@Duty_End_Time where
Program_ID=@Program_ID";

                System.Diagnostics.Debug.Write("apo klash:" + TextBoxNewDate.Text +
TextBoxNewStartTime.Text + TextBoxNewEndTime.Text + TextBoxProgID.Text);


                System.Data.SqlClient.SqlCommand SqlCmd2 = new SqlCommand(cmd2, conn);
                SqlCmd2.Parameters.AddWithValue("@Date", TextBoxNewDate.Text);
                SqlCmd2.Parameters.AddWithValue("@Duty_Start_Time",
TextBoxNewStartTime.Text);
                SqlCmd2.Parameters.AddWithValue("@Duty_End_Time",
TextBoxNewEndTime.Text);
                SqlCmd2.Parameters.AddWithValue("@Program_ID", TextBoxProgID.Text);
                SqlCmd2.ExecuteNonQuery();

                string sSQL3 = "DELETE  from Applications where
Program_ID=@Program_ID";
                string sSQL4 = "DELETE  from Modifications where
Program_ID=@Program_ID";

                System.Data.SqlClient.SqlCommand SqlCmd3 = new SqlCommand(sSQL3, conn);
                SqlCmd3.Parameters.AddWithValue("@Program_ID", TextBoxProgID.Text);
                SqlCmd3.ExecuteNonQuery();

                System.Data.SqlClient.SqlCommand SqlCmd4 = new SqlCommand(sSQL4, conn);
                SqlCmd4.Parameters.AddWithValue("@Program_ID", TextBoxProgID.Text);
                SqlCmd4.ExecuteNonQuery();

                TextBoxResult.Text = "Το αίτημά σας καταχωρήθηκε";

            }
        }

        protected void ImageButtonCancel_Click(object sender, ImageClickEventArgs e)
```

```
        {

        }

        protected void ImageButtonAddUser_Click(object sender, ImageClickEventArgs e)
        {
            Response.Redirect("UserManagement.aspx");
        }

        protected void ImageButtonProgramCreation_Click(object sender,
ImageClickEventArgs e)
        {
            Response.Redirect("InitProgramCreation.aspx");
        }

        protected void ImageButton2_Click(object sender, ImageClickEventArgs e)
        {
            Response.Redirect("EditUsers.aspx");
        }

        protected void ImageButtonGoToMenu_Click(object sender, ImageClickEventArgs e)
        {
            Response.Redirect("MainMenu.aspx");
        }

    }

    public class NewProgData
    {
        public string NDate="";
        public string NSTime = "";
        public string NETime = "";

        public void SetNDate(string Date)
        {
            NDate = Date;
        }
        public void SetNSTime(string STime)
        {
            NSTime = STime;
        }
        public void SetNETime(string ETime)
        {
            NETime = ETime;
        }
        public string GetNDate()
        {
            return NDate;
        }
        public string GetNSTime()
        {
            return NSTime;
        }
        public string GetNETime()
        {
            return NETime;
        }
    }
}
```

3.6.2 Κώδικας HTML

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="ChangeApplicationManagement.aspx.cs"
Inherits="WorkStation_V1._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title></title>
    <style type="text/css">
        #form1
        {
            height: 1662px;
            width: 1118px;
        }
        .style1
        {
            height: 1420px;
            width: 999px;
            margin-top: 0px;
            margin-bottom: 0px;
            margin-right: 135px;
        }
        .style2
        {
            margin-left: 0px;
        }
        </style>
</head>
<body>
    <form id="ChangeApplicationManagement" runat="server" class="style1">
          <div style="border-style: double; text-align: center;">

        <span lang="el"
            style="font-family: Calibri; font-size: xx-large; font-weight:
bold; font-style: normal; clip: rect(auto, auto, auto, auto); text-align:
center;">
        Σύστημα Διαχείρισης Εφημεριών και Γενικών Καθηκόντων</span><br />
        <span lang="el"
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        <br />
 Κεντρική Εφαρμογή Work Station-
            Υποσύστημα Διαχείρισης Δεδομένων</span></div>
            <table>
        <tr>
        <td><span lang="el">Κεντρικό Μενού</span></td>
        <td>
            <asp:ImageButton ID="ImageButton3" runat="server"
                ImageUrl="~/Icons/left-arrow-mini-icone-6194-48.png" />
            </td>
        <td><span lang="el">Διαχείριση Αιτήσεων</span></td>
        </tr>
        </table>
    <p style="text-align: center">
        <span lang="el"
```

```
        style="font-family: Calibri; font-size: x-large; font-weight: bold;
font-style: normal; text-align: center;">
        Διαχείριση Αιτήσεων</span></p>
    <p style="border-bottom-style: solid">
         </p>

    <br />
    <table>
    <tr>
    <td style="text-align: center">
     <asp:GridView ID="GridViewApplications" runat="server"
BackColor="White"
        BorderColor="#999999" BorderStyle="Solid" BorderWidth="1px"
CellPadding="3"
        ForeColor="Black" GridLines="Vertical"
    style="margin-left: 3px; margin-top: 0px" Width="487px"
            AutoGenerateSelectButton="True" Caption="Κατάλογος Αιτήσεων"
CaptionAlign="Top"
            Height="241px"

onselectedindexchanged="GridViewApplications_SelectedIndexChanged">
            <FooterStyle BackColor="#CCCCCC" />
            <PagerStyle BackColor="#999999" ForeColor="Black"
HorizontalAlign="Center" />
            <SelectedRowStyle BackColor="#000099" Font-Bold="True"
ForeColor="White" />
            <HeaderStyle BackColor="Black" Font-Bold="True"
ForeColor="White" />
            <AlternatingRowStyle BackColor="#CCCCCC" />
        </asp:GridView>
    </td>
    </tr>
    </table>
    <p style="border-bottom-style: none; border-top-style: solid;"></p>
    <p style="text-align: center; font-size: medium; font-weight:
bold"><span lang="el">
        Επεξεργασία Αίτησης Χρήστη</span></p>
    <table style="border-style: solid none solid none;"
             title="Επεξεργασία Αίτησης Χρήστη" align="center">
    <tr>
    <td>

    <table style="text-align: center">
    <tr>

    <td style="text-align: center">
    <asp:Label ID="Label9" runat="server" Text="Τύπος Αίτησης"></asp:Label>
    <br />
        <asp:TextBox ID="TextBoxApp_Type" runat="server"
Width="189px"></asp:TextBox>
    </td>
    </tr>
    <tr>
    <td style="text-align: center">
    <asp:Label ID="Label3" runat="server" Text="Κωδικός
Προγραμ."></asp:Label>
        <br />
        <asp:TextBox ID="TextBoxProgID" runat="server"
Width="189px"></asp:TextBox>
        <br />
    </td>
```

```
    </tr>
    <tr>
    <td>
    <asp:Label ID="Label5" runat="server" Text="Κωδικός
Χρήστη"></asp:Label>
        <br />
        <asp:TextBox ID="TextBoxUserID" runat="server" style="margin-left:
0px"
            Width="189px"></asp:TextBox>
    </td>
    </tr>
    <tr>
    <td>
    <asp:Label ID="Label6" runat="server" Text="Τύπος
Εργασίας"></asp:Label>
        <br />

        <asp:TextBox ID="TextBoxDt" runat="server"  style="margin-left:
0px"
            Width="189px"></asp:TextBox>
    </td>
    </tr>
    <tr>
    <td style="text-align: center">
    <asp:Label ID="Label7" runat="server" Text="Τοποθεσία"></asp:Label>
        <br />
        <span lang="en-us"></span>
        <asp:TextBox ID="TextBoxLocation" runat="server"
Width="189px"></asp:TextBox>
    </td>
    </tr>
    <tr>
    <td>
    <asp:Label ID="Label8" runat="server" Text="Όνομα
Προγραμ."></asp:Label>
        <br /><span lang="el">
        <asp:TextBox ID="TextBoxProgramName" runat="server"
Width="189px"></asp:TextBox>
        </span>
    </td>
    </tr>
    </table>


    </td>

    <td style="text-align: center; border-left-style: solid">
    <table>
    <tr style="border-bottom-style: solid"><td style="text-align: center">
        <asp:GridView ID="GridViewCurrentProgram" runat="server"
BackColor="White"
            BorderColor="#CC9966" BorderStyle="None" BorderWidth="1px"
            Caption="Τρέχων Πρόγραμμα" CellPadding="4" CssClass="style2"
Width="419px">
            <RowStyle BackColor="White" ForeColor="#330099" />
            <FooterStyle BackColor="#FFFFCC" ForeColor="#330099" />
            <PagerStyle BackColor="#FFFFCC" ForeColor="#330099"
HorizontalAlign="Center" />
            <SelectedRowStyle BackColor="#FFCC66" Font-Bold="True"
ForeColor="#663399" />
```

```
            <HeaderStyle BackColor="#990000" Font-Bold="True"
ForeColor="#FFFFCC" />
        </asp:GridView>
    </td>
    </tr>
    <tr style="border-top-style: solid">
    <td style="border-top-style: solid">
    <asp:GridView ID="GridViewNewProgram" runat="server" BackColor="White"
        BorderColor="#CC9966" BorderStyle="None" BorderWidth="1px"
        Caption="Προτεινόμενες Αλλαγές" CellPadding="4" Width="421px"
CaptionAlign="Top"
        >
        <RowStyle BackColor="White" ForeColor="#330099" />
        <FooterStyle BackColor="#FFFFCC" ForeColor="#330099" />
        <PagerStyle BackColor="#FFFFCC" ForeColor="#330099"
HorizontalAlign="Center" />
        <SelectedRowStyle BackColor="#FFCC66" Font-Bold="True"
ForeColor="#663399" />
        <HeaderStyle BackColor="#990000" Font-Bold="True"
ForeColor="#FFFFCC" />
    </asp:GridView>
    </td></tr>
    <tr>
        <td style="text-align: center; border-top-style: solid;">
            <table>
                <tr>
                    <td>
        <asp:ImageButton ID="ImageButtonAccept" runat="server"

            ImageUrl="~/Icons/accept-this-fact-so-successful-green-ok-
icone-9341-64.png"
            onclick="ImageButtonAccept_Click" />
        <br />
        <span lang="el">Αποδοχή Αίτησης</span></td>
                    <td></td>
    <td>
        <asp:ImageButton ID="ImageButtonCancel" runat="server"
            ImageUrl="~/Icons/cancel-remove-icone-5993-64.png"
            onclick="ImageButtonCancel_Click" />
        <br />
        <span lang="el">Απόρριψη Αίτησης</span></td>
             </tr>
            </table>
    </td>
    </tr>
    </table>
    </td>
    </tr>
  </table>
    <p ></p>

    <table><tr><td>
        <asp:TextBox ID="TextBoxResult" runat="server" Font-Bold="True"
            Font-Overline="True" Font-Size="Large"
Width="880px"></asp:TextBox>
        </td></tr></table>

        <p style="border-bottom-style: solid"></p>
    <p style="font-size: medium; font-weight: bold; text-align: left;>

          <span lang="el"> Μετάβαση σε</span></p>
```

```
    <table><tr>
        <td align="center" style="border-style: solid">
            <asp:ImageButton ID="ImageButtonAddUser" runat="server"
ImageAlign="Middle"
                ImageUrl="~/Icons/new-user-group-icone-6256-128.png"
                onclick="ImageButtonAddUser_Click" />
            <br />
            <span lang="el">Εισαγωγή νέου χρήστη </span>
        </td>
        <td></td>
        <td align="center" style="border-style: solid">
        <asp:ImageButton ID="ImageButtonProgramCreation" runat="server"
            ImageUrl="~/Icons/ChronologicalReview.png"
            onclick="ImageButtonProgramCreation_Click" />
        <br />
        <span lang="el">Δημιουργία Προγράμματος</span></td>
            <td>
            </td>
            <td style="text-align: center; border-style: solid">
                <asp:ImageButton ID="ImageButton2" runat="server"
                    ImageUrl="~/Icons/edit-find-replace-old-icone-8806-
64.png"
                    onclick="ImageButton2_Click" />
                <br />
                <span lang="el">Επεξεργασία Χρηστών</span></td>
                <td></td>
            <td align="center" style="border-style: solid">
                <asp:ImageButton ID="ImageButtonGoToMenu" runat="server"
                    ImageUrl="~/Icons/office-icone-8240-128.png"
                    onclick="ImageButtonGoToMenu_Click" />
                <br />
                <span lang="el">Κεντρικό Μενού</span></td>

        </tr></table>
        <p>
            <asp:TextBox ID="TextBoxNewDate" runat="server"
Visible="False"></asp:TextBox>
            <span lang="en-us"> <asp:TextBox ID="TextBoxNewStartTime"
runat="server"
                Visible="False"></asp:TextBox>
 <asp:TextBox ID="TextBoxNewEndTime" runat="server"
Visible="False"></asp:TextBox>
            </span></p>

  </form>
</body>
</html>
```

4 Κώδικας SQL της Κεντρικής Βάσης Δεδομένων «SystemData»

4.1 Πίνακας Address

```
CREATE TABLE [dbo].[Address](
	[User_ID] [nvarchar](20) NULL,
	[nomos] [nvarchar](50) NULL,
	[orofos] [nvarchar](50) NULL,
	[city] [nvarchar](50) NULL,
	[address] [nvarchar](80) NULL,
	[TK] [nvarchar](10) NULL,
	[perioxi] [nvarchar](50) NULL,
	[Xwra] [nvarchar](50) NULL
) ON [PRIMARY]
```

4.2 Πίνακας Application

```
CREATE TABLE [dbo].[Applications](
	[User_ID] [nvarchar](20) NULL,
	[Program_ID] [int] NOT NULL,
	[App_Type] [nvarchar](20) NULL,
 CONSTRAINT [PK_Applications] PRIMARY KEY CLUSTERED
(
	[Program_ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

4.3 Πίνακας Bathmides

```
CREATE TABLE [dbo].[Bathmides](
	[Onoma_Bathmidas] [nvarchar](50) NULL
) ON [PRIMARY]
```

4.4 Πίνακας ChangeList

```
REATE TABLE [dbo].[ChangeList](
	[Program_ID] [int] NOT NULL,
	[User_ID] [varchar](20) NULL,
 CONSTRAINT [PK_ChangeList] PRIMARY KEY CLUSTERED
```

```
(
      [Program_ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

4.5 Πίνακας Comment

```
CREATE TABLE [dbo].[Comment](
      [Item_ID] [int] IDENTITY(1,1) NOT NULL,
      [Creator_ID] [varchar](20) NULL,
      [commented_item_ID] [varchar](20) NULL,
      [comment_text] [varchar](255) NULL
) ON [PRIMARY]
```

4.6 Πίνακας Declared_Duties

```
CREATE TABLE [dbo].[Declared_Duties](
      [User_ID] [varchar](20) NULL,
      [Duty_Types] [nvarchar](50) NULL
) ON [PRIMARY]
```

4.7 Πίνακας Declared_Locations

```
CREATE TABLE [dbo].[Declared_Locations](
      [User_ID] [varchar](20) NULL,
      [Location] [nvarchar](50) NULL
) ON [PRIMARY]
```

4.8 Πίνακας Eidikothtes

```
CREATE TABLE [dbo].[Eidikothtes](
      [Eidikothta_Name] [nvarchar](60) NULL
) ON [PRIMARY]
```

4.9 Πίνακας Item

```
CREATE TABLE [dbo].[Item](
      [Item_ID] [int] NOT NULL,
      [Creator_ID] [varchar](20) NULL,
      [Receiver_ID] [varchar](20) NULL
) ON [PRIMARY]
```

4.10 Πίνακας Locations

```
CREATE TABLE [dbo].[Locations](
      [Location_Name] [nvarchar](50) NULL
) ON [PRIMARY]
```

4.11 Πίνακας Message

```
CREATE TABLE [dbo].[Message](
      [Item_ID] [int] IDENTITY(1,1) NOT NULL,
      [Creator_ID] [varchar](20) NULL,
      [content] [varchar](255) NULL
) ON [PRIMARY]
```

4.12 Πίνακας Modifications

```
CREATE TABLE [dbo].[Modifications](
	[Program_ID] [int] NOT NULL,
	[Date] [varchar](20) NULL,
	[Duty_Start_Time] [varchar](20) NULL,
	[Duty_End_Time] [varchar](20) NULL,
	[User_ID] [varchar](20) NULL
) ON [PRIMARY]
```

4.13 Πίνακας Phone_Numbers

```
CREATE TABLE [dbo].[Phone_Numbers](
	[User_ID] [nvarchar](50) NOT NULL,
	[Tupos_Til] [nvarchar](50) NULL,
	[Til] [nvarchar](50) NOT NULL
) ON [PRIMARY]
```

4.14 Πίνακας Program

```
CREATE TABLE [dbo].[Program](
	[Program_ID] [int] IDENTITY(1,1) NOT NULL,
	[Date] [nvarchar](50) NULL,
	[Duty_Type] [nvarchar](50) NULL,
	[Duty_Start_Time] [nvarchar](10) NULL,
	[Duty_End_Time] [nvarchar](10) NULL,
	[Location] [nvarchar](50) NULL,
	[User_ID] [varchar](20) NULL,
	[Program_Name] [nvarchar](30) NULL,
 CONSTRAINT [PK_Program] PRIMARY KEY CLUSTERED
(
	[Program_ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

4.15 Πίνακας Department

```
CREATE TABLE [dbo].[Department](
	[Department_Name] [varchar](50) NOT NULL,
	[Department_ID] [varchar](50) NULL
) ON [PRIMARY]
```

4.16 Πίνακας Sum_Duties

```
CREATE TABLE [dbo].[Sum_Duties](
	[Duty_Type_Name] [nvarchar](50) NULL
) ON [PRIMARY]
```

4.17 Πίνακας User_Teams

```
CREATE TABLE [dbo].[User_Team](
	[User_Name] [varchar](50) NOT NULL,
	[Team_ID] [varchar](50) NULL
) ON [PRIMARY]
```

4.18 Πίνακας Users

```
CREATE TABLE [dbo].[Users](
	[User_ID] [int] IDENTITY(1,1) NOT NULL,
	[User_Team] [nvarchar](50) NULL,
	[name] [nvarchar](50) NULL,
	[surname] [nvarchar](50) NULL,
	[username] [nvarchar](50) NOT NULL,
	[password] [nvarchar](50) NOT NULL,
	[AMKA] [nvarchar](50) NULL,
	[status] [nvarchar](50) NULL,
	[Department] [nvarchar](50) NULL,
	[Thesi] [nvarchar](50) NULL,
	[Eidikothta] [nvarchar](50) NULL,
	[Bathmida] [nvarchar](50) NULL,
 CONSTRAINT [username] PRIMARY KEY CLUSTERED
(
	[username] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY],
 CONSTRAINT [some_name] UNIQUE NONCLUSTERED
(
	[User_ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY],
 CONSTRAINT [user_id] UNIQUE NONCLUSTERED
(
	[User_ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

4.19 Πίνακας Doctor Users

```
CREATE TABLE [dbo].[Doctor Users](
	[User_ID] [varchar](20) NULL,
	[Thesi] [varchar](50) NULL,
	[ID team] [smallint] NULL,
	[Department] [varchar](50) NULL,
	[Eidikotita] [varchar](50) NULL,
	[Bathmida] [varchar](50) NULL
) ON [PRIMARY]
```

4.20 Πίνακας Nurse User

```
CREATE TABLE [dbo].[Nurse_User](
	[User_ID] [varchar](20) NULL,
	[Thesi] [varchar](50) NULL,
	[ID team] [smallint] NULL,
	[Department] [varchar](50) NULL,
	[Bathmima] [varchar](50) NULL,
	[Eidikotita] [varchar](50) NULL
) ON [PRIMARY]
```