ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη εφαρμογής ηλεκτρονικής ατζέντας με καταγραφή μέτρησης πίεσης και σακχάρου σε περιβάλλον Android

Development of application "e-healthy calendar" for smartphones in Android environment

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Νικόλαου Σταμπόλου

Βόλος, Ιούνιος 2015

Η σελίδα αυτή είναι σκόπιμα λευκή.

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη εφαρμογής ηλεκτρονικής ατζέντας με καταγραφή μέτρησης πίεσης και σακχάρου σε περιβάλλον Android

Development of application "e-healthy calendar" for smartphones in Android environment

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Νικόλαου Σταμπόλου

Επιβλέποντες :

Ακρίτας Αλκιβιάδης
Καθηγητής Π.Θ.

Σταμούλης Γεώργιος
Καθηγητής Π.Θ.

Εγκρίθηκε από την διμελή εξεταστική επιτροπή την ημερομηνία εξέτασης.

(Υπογραφή)
.....
ΑκρίταςΑλκιβιάδης
Καθηγητής Π.Θ.

(Υπογραφή)
.....
Σταμούλης Γεώργιος
Καθηγητής Π.Θ.

Βόλος, Ιούνιος 2015

(Υπογραφή)

.....

Νικόλαος Σταμπόλος

Διπλωματούχος Μηχανικός Ηλεκτρονικών Υπολογιστών,
Τηλεπικοινωνιών και Δικτύων του Τμήματος Ηλεκτρολόγων
Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστημίου Θεσσαλίας

Η σελίδα αυτή είναι σκόπιμα λευκή.

Ευχαριστίες

Θα ήθελα να εκφράσω την βαθιά μου ευγνωμοσύνη στον κ. Αλκιβιάδη Ακρίτα , τον επιβλέπων καθηγητή μου για την καθοδήγηση και τον χρόνο που αφιέρωσε στην εκπόνηση της διπλωματικής αυτής εργασίας.

Τέλος , θέλω να ευχαριστήσω τους γονείς και τους φίλους μου για την υποστήριξη , την επιμονή και την ενθάρρυνση τους κατά τη διάρκεια των σπουδών μου.

Περίληψη

Τη σημερινή εποχή, η ραγδαία εξέλιξη της τεχνολογίας των "έξυπνων τηλεφώνων" (smartphones), έχει οδηγήσει στην ανάπτυξη πολυάριθμων εφαρμογών με χρησιμότητα σε όλους τους τομείς της καθημερινής μας ζωής.

Σκοπός της εργασίας είναι η ανάπτυξη μιας εφαρμογής σε λειτουργικό σύστημα Android, η οποία έχει ως κύριο στόχο να παρέχει εξειδικευμένες υπηρεσίες στο χρήστη, μέσα από την υλοποίηση ενός ηλεκτρονικού ημερολογίου και την δυνατότητα καταγραφής καθημερινών μετρήσεων σακχάρου και πίεσης. Η εφαρμογή παρέχει ακόμα τη δυνατότητα καταγραφής προσωπικών στοιχείων, οικείων προσώπων, ιατρών και αναλυτικούς πίνακες θερμιδομετρητή και φυσιολογικών τιμών εξετάσεων.

Abstract

Nowadays, the rapid improvement of technology of smartphones, has led to the development of numerous applications, used in all areas of our daily life.

Purpose of this thesis is to develop an application in Android operating system, which has a main objective to provide specialized services to the user, through the implementation of an electronic calendar and the ability to record daily blood sugar and pressure measurements. The application also offers the options to record personal data, relevant persons and doctors information and analytical calorimeter tables and physiological examinations prices.

Πίνακας περιεχομένων

1 Εισαγωγή

1.1 Σκοπός εργασίας

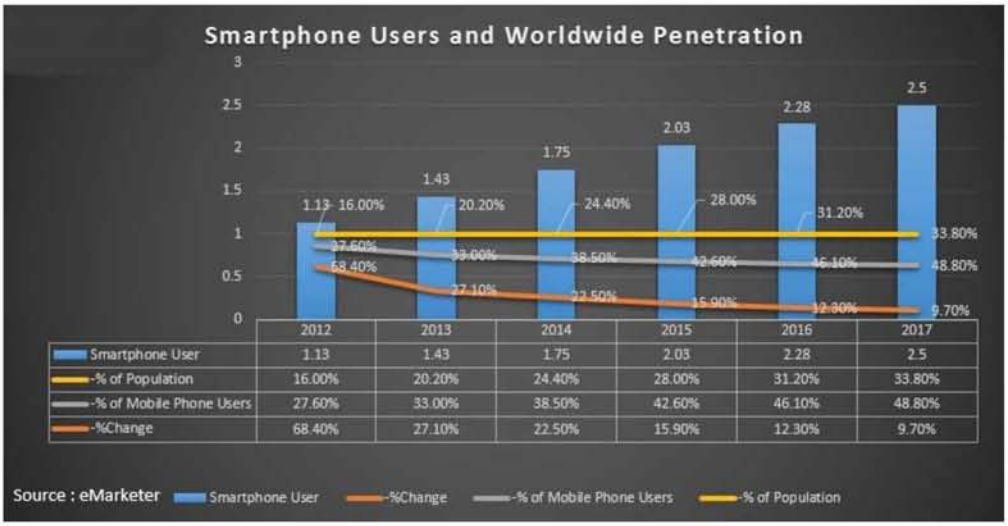
Η εργασία αυτή παρουσιάζει όλα τα βήματα για τον σχεδιασμό και την ανάπτυξη εφαρμογής για έξυπνα τηλέφωνα τελευταίας γενιάς (smartphones) σε λειτουργικό σύστημα Android. Η εφαρμογή στοχεύει στην οργάνωση της καθημερινής ζωής μέσα από ένα ημερολόγιο και στην καταγραφή χρήσιμων πληροφοριών για τον χρήστη, όπως συλλογή δεδομένων για μετρήσεις σακχάρου και πίεσης .

1.2 Κίνητρο

Το βασικό κίνητρο για την ανάπτυξη της εφαρμογής ήταν η ραγδαία εξάπλωση και χρήση των έξυπνων τηλεφώνων, γεγονός που κάνει επιτακτική την ανάγκη υλοποίησης χρήσιμων εφαρμογών. Σε αυτή λοιπόν την κατεύθυνση, η εφαρμογή αυτή έχει ως στόχο να καλύψει την ανάγκη των χρηστών να διατηρούν ιστορικό καταγραφής της κατάστασης της υγείας τους. Στη συνέχεια παρατίθεται μία εικόνα που δείχνει την εξάπλωση των έξυπνων κινητών τηλεφώνων παγκοσμίως.

Όπως φαίνεται και στην εικόνα 1.2.1, στα τέλη του 2014 ο αριθμός των παγκόσμιων χρηστών έξυπνων τηλεφώνων έφτασε τα 1.75 δισεκατομμύρια, δηλαδή περίπου στο 25% του παγκόσμιου πληθυσμού. Ο αριθμός αυτός αναμένεται να αυξηθεί στα 2.5 δισεκατομμύρια στα τέλη του 2017 αγγίζοντας το 34% του παγκόσμιου πληθυσμού.

Παγκόσμιοι χρήστες έξυπνων τηλεφώνων



Εικόνα 1.2.1: Παγκόσμιοι χρήστες έξυπνων τηλεφώνων

1.3 Δομή διπλωματικής εργασίας

Στο σημείο αυτό για την διευκόλυνση του αναγνώστη δίνεται μία αφηρημένη διάρθρωση της δομής της διπλωματικής εργασίας . Στο 2° κεφάλαιο παρουσιάζεται το θεωρητικό υπόβαθρο για την ανάπτυξη της εφαρμογής. Το 3° κεφάλαιο που ακολουθεί, περιγράφει την αρχιτεκτονική της εφαρμογής σε υψηλό επίπεδο με την αφηρημένη περιγραφή διαγράμματος (high-level diagram). Το 4° κεφάλαιο εμβαθύνει και παρουσιάζει τις λεπτομέρειες της υλοποίησης. Στη συνέχεια, στο 5° κεφάλαιο γίνεται παρουσίαση της εφαρμογής από την οπτική γωνία του χρήστη. Στο 6° κεφάλαιο αναλύεται το λογισμικό και τα εργαλεία που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής. Ακόμα, στο κεφάλαιο αυτό δίνονται οι απαιτήσεις εγκατάστασης και οδηγίες χρήσης της εφαρμογής. Τέλος, τα κεφάλαια 7 και 8 περιέχουν τον επίλογο και τις αναφορές σε βιβλιογραφία και περαιτέρω παραρτήματα

2 Θεωρητικό υπόβαθρο

2.1 Το λειτουργικό σύστημα Android

Το **Android** είναι ένα λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας, το οποίο χρησιμοποιεί τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την [Handset Alliance | Open Handset Alliance]. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google. Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλέφωνα και οι ταμπλέτες, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί και σε κονσόλες παιχνιδιών, ψηφιακές φυτογραφικές μηχανές, συνηθισμένους Η/Υ και σε άλλες ηλεκτρονικές συσκευές.
Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής υλικού (hardware), οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Irina Blok.

Εικόνα 2.1.1:Λογότυπο Android

Ιστορική ανάδρομη του Android

Η ιστορία εκδόσεων του λειτουργικού συστήματος Android των κινητών τηλεφώνων ξεκίνησε με την κυκλοφορία του Android beta, το Νοέμβριο του 2007. Η πρώτη εμπορική έκδοση ήταν το Android 1.0 που κυκλοφόρησε το Σεπτέμβριο του 2008. Το

Android είναι υπό συνεχή ανάπτυξη από την Google και την Open Handset Alliance (OHA), και έχει πραγματοποιηθεί μια σειρά από ενημερώσεις στην λειτουργία του συστήματος από την αρχική του κυκλοφορία μέχρι σήμερα.

Από τον Απρίλιο του 2009, οι εκδόσεις του Android έχουν θέμα από την ζαχαροπλαστική στην κωδική τους ονομασία και κυκλοφόρησαν σε αλφαβητική σειρά:

1. Apple Pie (1.0)
2. Banana Bread (1,1)
3. Cupcake (1,5)
4. Donut (1,6)
5. Eclair (2.0-2.1)
6. Froyo (2.2-2.2.3)
7. Gingerbread (2.3-2.3.7)
8. Honeycomb (3.0-3.2.6)
9. Ice Cream Sandwich (4.0-4.0.4)
10. Jelly Bean (4.1-4.3.1)
11. KitKat (4.4-4.4.4)
12. Lollipop (5.0-5.0.2)

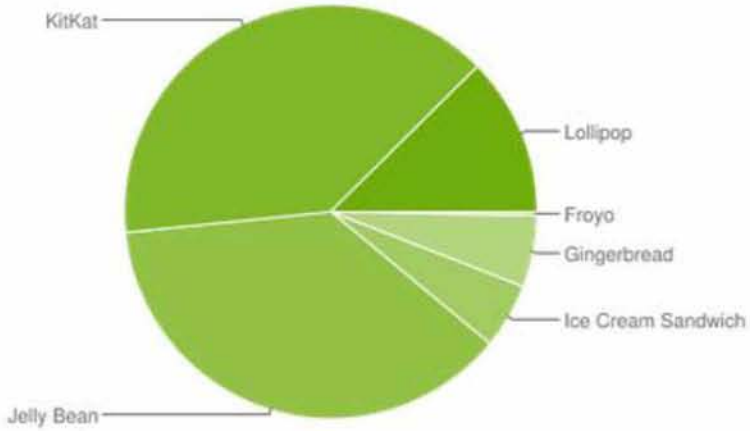
Εκδόσεις Android συστημάτων

Σε αυτό το τμήμα παρέχονται δεδομένα για τον σχετικό αριθμό των συσκευών με συγκεκριμένη έκδοση του λογισμικού Android. Τα δεδομένα συλλέχθηκαν σε διάστημα 7 ημερών μέχρι την 1η Ιουνίου 2015. Εκδόσεις με διανομή μικρότερη του 0,1% δεν αναφέρονται καθόλου. Τα δεδομένα αυτά συγκεντρώθηκαν από την νέα εφαρμογή Google Play Store, που υποστηρίζεται σε εκδόσεις Android 2.2 και μεταγενέστερες, έτσι συσκευές με παλαιότερες εκδόσεις δεν περιλαμβάνονται.

Στον πίνακα που ακολουθεί παρουσιάζονται οι εκδόσεις αυτές με το κωδικό όνομα τους στην 2η στήλη, την έκδοση API τους στην 3η στήλη και η τελευταία στήλη απεικονίζει την διανομή, η οποία δίνεται και με το ακολουθούμενο διάγραμμα.

Version	Codename	API	Distribution
2.2	Froyo	8	0.3%
2.3.3 - 2.3.7	Gingerbread	10	5.6%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	5.1%
4.1.x	Jelly Bean	16	14.7%
4.2.x		17	17.5%
4.3		18	5.2%
4.4	KitKat	19	39.2%
5.0	Lollipop	21	11.6%
5.1		22	0.8%

Πίνακας 2.1.1: Εκδόσεις Android



Εικόνα 2.1.2: Διανομή εκδόσεων Android

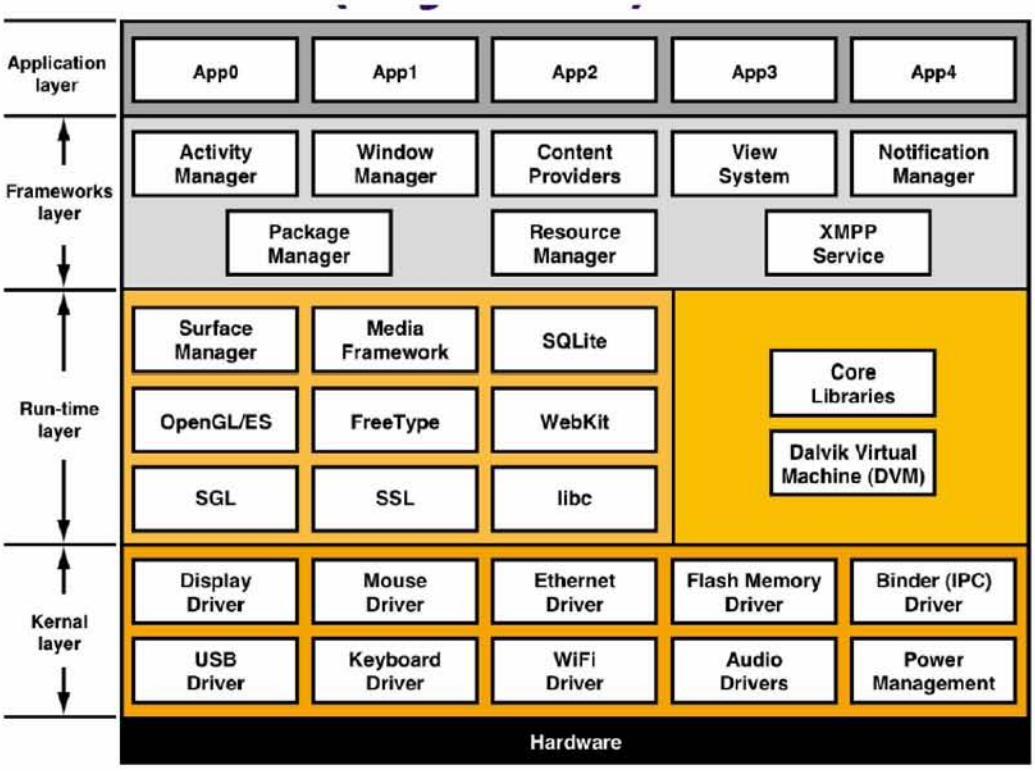
Υλικό

Η κύρια πλατφόρμα υλικού για το Android είναι η αρχιτεκτονική ARM (ARMv7 και ARMv8-A), καθώς και οι x86 και MIPS αρχιτεκτονικές (οι δύο τελευταίες κυρίως σε νεότερες εκδόσεις του Android). Από την έκδοση Android 5.0 "Lollipop", υποστηρίζονται 64-bit παραλλαγές από όλες τις πλατφόρμες. Επίσης από το 2012, έχουν κυκλοφορήσει στην αγορά Android συσκευές με Intel επεξεργαστές, όπως κινητά τηλέφωνα και ταμπλέτες (Lenovo K900, ASUS Padfone Mini). Ακόμη, από το Νοέμβριο του 2013 το Android 4.4 "συνιστά" τουλάχιστον 512 MB RAM.

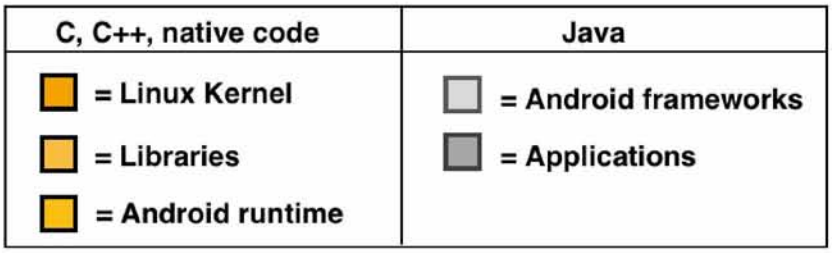
Οι συσκευές Android ενσωματώνουν πολλά προαιρετικά εξαρτήματα υλικού όπως βίντεο κάμερες, GPS, αισθητήρες προσανατολισμού, χειριστήρια παιχνιδιών, επιταχυνσιόμετρα, γυροσκόπια, βαρόμετρα, μαγνητόμετρα, αισθητήρες εγγύτητας, αισθητήρες πίεσης, θερμόμετρα, κ.α.

Αρχιτεκτονική

Το σύστημα Android ανήκει στην οικογένεια των λειτουργικών συστημάτων που βασίζονται στην προσέγγιση με επίπεδα – στρώματα (layered approach). Σύμφωνα με αυτήν την προσέγγιση, καθένα επίπεδο χτίζεται πάνω από υπάρχοντα επίπεδα. Το Android είναι μία στοίβα από κομμάτια λογισμικού, διαιρεμένα σε 5 τμήματα και 4 κυρίως στρώματα, τα οποία διακρίνονται και στις επόμενες εικόνες.



Εικόνα 2.1.3: Διάγραμμα της αρχιτεκτονικής του Android



Εικόνα 2.1.4: χρησιμοποιούμενες γλώσσες προγραμματισμού

Στη συνέχεια περιγράφονται τα τμήματα αυτά:

- **Linux kernel:** Η βάση της στοίβας είναι ο πυρήνας του Linux. Ο τροποποιημένος πυρήνας του συστήματος βασίζεται κυρίως στις εκδόσεις 3.4 ή 3.10 (Απρίλιος του 2014) του πυρήνα του Linux, οι οποίες υποστηρίζουν όλες τις κύριες λειτουργίες του συστήματος. Οι λειτουργίες αυτές αφορούν τη διαχείριση μνήμης, τη διαχείριση εργασιών, τις λειτουργίες δικτύου, την ασφάλεια του λειτουργικού, καθώς και ένα σύνολο από οδηγούς υλικού (hardware drivers). Συγκεκριμένα, οι οδηγοί αυτοί είναι υπεύθυνοι για την επικοινωνία του κομματιού του λογισμικού με το υλικό κομμάτι της συσκευής. Ο πυρήνας του Android μπορεί να βασίζεται στον πυρήνα του Linux, αλλά διαφέρει αρκετά από αυτόν. Αυτό οφείλεται στο γεγονός ότι η Google έχει κάνει διάφορες αλλαγές, ώστε να είναι ελαφρύτερος και βελτιστοποιημένος για χρήση σε κινητές συσκευές.

- **Libraries:** Στο δεύτερο επίπεδο υπάρχουν οι βιβλιοθήκες του Android. Αυτές ουσιαστικά αποτελούν τα APIs, τα οποία είναι διαθέσιμα στους προγραμματιστές για την ανάπτυξη των εφαρμογών. Αποτελούν ένα από τα δομικά υλικά των εφαρμογών, καθώς ενσωματώνονται και χρησιμοποιούνται για τις διάφορες λειτουργίες που παρέχει η καθεμία από αυτές. Το σύνολο σχεδόν των βιβλιοθηκών είναι γραμμένο σε C και C++ και έχουν μεταγλωττιστεί για τη χρήση τους από το λειτουργικό.

- **Dalvik Virtual Machine:** Σχεδόν όλο το σύνολο των APIs του Android βασίζεται στη γλώσσα προγραμματισμού Java. Στη Java ως γνωστόν υπάρχει η λεγόμενη Java εικονική μηχανή (Virtual Machine), στην οποία εκτελείται ο κώδικας bytecode των εφαρμογών. Στο Android υπάρχει αντίστοιχα η εικονική μηχανή Dalvik. Η Dalvik είναι η εικονική μηχανή μέσω της οποίας τρέχουν οι εφαρμογές του Android. Η κάθε εφαρμογή τρέχει μέσω της δικής της εικονικής μηχανής στη δικιά της διεργασία και γι αυτό το λόγο καμιά εφαρμογή δεν έχει επαφή με την άλλη, ενώ εκτελούνται παράλληλα. Η Dalvik δεν υποστηρίζει τον κώδικα bytecode, αντί αυτού οι κλάσεις της Java μεταγλωττίζονται (compile) σε αρχεία ".dex" για να τρέξουν στη VM. Τα

αρχεία dex ουσιαστικά αποτελούν συμπιεσμένα δεδομένα για εξοικονόμηση χώρου και χρόνου κατά την εκτέλεση. Το Android είναι από τη φύση του λειτουργικό σύστημα που υποστηρίζει την ταυτόχρονη εκτέλεση πολλών διεργασιών (multitasking) και έτσι επιτρέπει στις εφαρμογές του να τρέχουν σε πολλά νήματα ταυτόχρονα. Χάρη στο λιτό σχεδιασμό της Dalvik VM, αφού έχει σχεδιαστεί να έχει το ελάχιστο αντίκτυπο στη χρήση μνήμης, το σύστημα είναι σε θέση να τρέχει πολλές εικονικές μηχανές ταυτόχρονα.

- **Android runtime:** Ο χρόνος εκτέλεσης των εφαρμογών του Android βρίσκεται στο ίδιο επίπεδο με τις κύριες βιβλιοθήκες και την μηχανή Dalvik. Εδώ βρίσκεται το κοινό σημείο επαφής μεταξύ των δυνατοτήτων που παρέχουν οι βιβλιοθήκες και του χρόνου εκτέλεσης της Dalvik VM, τις λειτουργίες της οποίας περιγράψαμε παραπάνω.

- **Application framework:** Το Android παρέχει στους προγραμματιστές μια ανοικτού κώδικα πλατφόρμα ανάπτυξης, στην οποία μπορούν να δημιουργήσουν ιδιαίτερα καινοτόμες και πλούσιες σε υλικό εφαρμογές. Οι προγραμματιστές έχουν στην διάθεση τους τη δυνατότητα ελέγχου του υλικού της συσκευής και μέσω αυτής μπορούν να αποκτήσουν πρόσβαση σε υπηρεσίες εντοπισμού, εκτέλεση διεργασιών παρασκηνίου και πάρα πολλές ακόμη δυνατότητες, οι οποίες βασίζονται στα APIs που είναι διαθέσιμα. Μερικές από τις δυνατότητες αυτές είναι:

    - Ένα σύνολο από γραφικά στοιχεία (Views) για τη δημιουργία του γραφικού περιβάλλοντος, όπως λιστών (List View), κειμένων (Text View), κουμπιών (Buttons), κ.α.
    - Ένας διαχειριστής περιεχομένου (Content Manager), ο οποίος επιτρέπει το διαμοιρασμό δεδομένων μεταξύ των εφαρμογών.
    - Ένας διαχειριστής πόρων (Resource Manager) για την πρόσβαση σε πόρους όπως αλφαριθμητικά (strings), εικόνες, κ.α.
    - Ένας διαχειριστής ειδοποιήσεων (Notification Manager), ο οποίος επιτρέπει την προβολή ειδοποιήσεων στη μπάρα κατάστασης (status bar).
    - Ένας διαχειριστής δραστηριοτήτων (Activity Manager), ο οποίος διαχειρίζεται τον κύκλο ζωής των εφαρμογών.

- **Applications:** Σε αυτό το επίπεδο βρίσκονται όλες οι εφαρμογές Android, οι οποίες είναι εγκατεστημένες σε μία συσκευή.

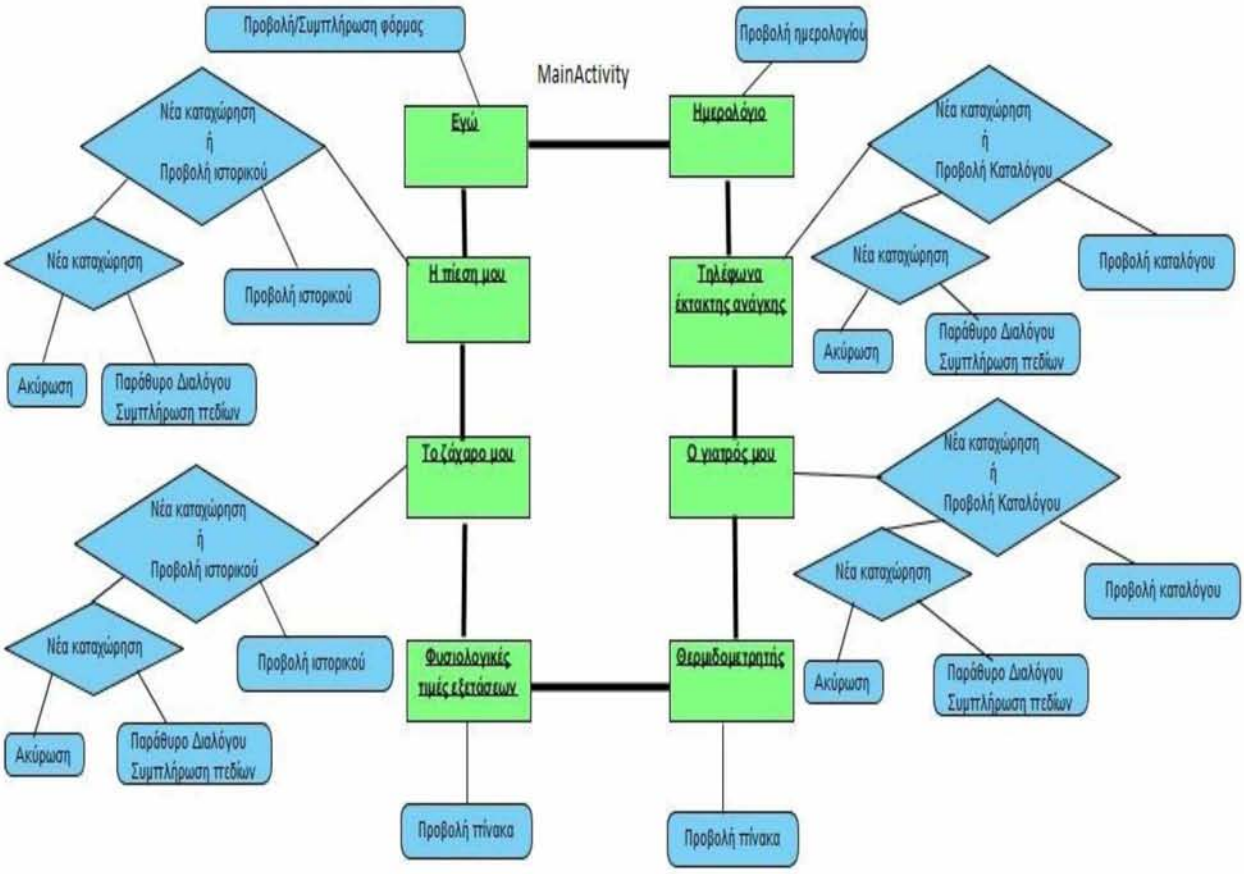
Εφαρμογές Android

Οι εφαρμογές (apps) που επεκτείνουν τη λειτουργικότητα των συσκευών, είναι γραμμένες κυρίως στη γλώσσα προγραμματισμού Java, χρησιμοποιώντας το Android Software Development Kit (SDK). Το SDK περιλαμβάνει ένα πρόγραμμα εντοπισμού σφαλμάτων (debugger), βιβλιοθήκες λογισμικού, εξομοιωτή (emulator), documentation και tutorials(εγχειρίδια χρήσης). Η Google υποστήριζε ως ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) το Eclipse με τη χρήση της προσθήκης Android Development Tool (ADT) μέχρι το Δεκέμβριο του 2014, όπου και κυκλοφόρησε το Android Studio ως κύριο IDE για την ανάπτυξη εφαρμογών Android. Το Google Play Store είναι ο κύριος πάροχος εφαρμογών, εγκατεστημένο σε κάθε συσκευή Android που συμμορφώνεται με τις απαιτήσεις συμβατότητας της Google και έχει άδεια χρήσης του λογισμικού Google Mobile Services. Το Google Play Store επιτρέπει στους χρήστες να αναζητούν, να κατεβάζουν και να ενημερώνουν τις εφαρμογές που είναι δημοσιευμένες από την Google ή από άλλους προγραμματιστές. Κατά το πρώτο τετράμηνο του 2015 υπάρχουν στο Play Store περισσότερες από 1,5 εκατομμύριο εφαρμογές διαθέσιμες για κατέβασμα.

3 Αρχιτεκτονική συστήματος

3.1 Ανάλυση επιμέρους μονάδων

Στο κεφάλαιο αυτό παρουσιάζεται η γενική αρχιτεκτονική της εφαρμογής σε υψηλό επίπεδο, μέσα από την αφηρημένη περιγραφή των επιμέρους μονάδων με τη βοήθεια του ακόλουθου διαγράμματος.



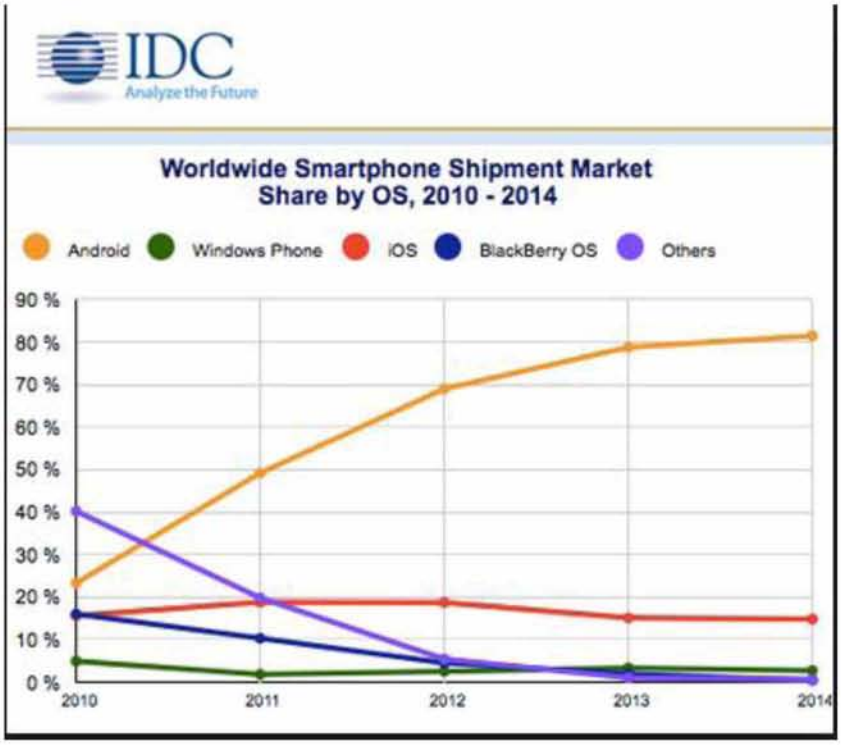
Εικόνα 3.1.1:Παρουσίαση δομής εφαρμογής

Η εφαρμογή αναπτύχθηκε στο λειτουργικό σύστημα Android. Ο κώδικας της εφαρμογής είναι γραμμένος σε Java, η οποία είναι γλώσσα προγραμματισμού υψηλού επιπέδου. Η εφαρμογή αναφέρεται σε χρήστες που επιθυμούν να διατηρούν ιστορικό της καθημερινής κατάστασης της υγείας τους, με την καταγραφή μετρήσεων πίεσης και σακχάρου. Οι υπηρεσίες που προσφέρει η εφαρμογή στους χρήστες περιγράφονται στην εικόνα 3.1.1 από τον κύριο κορμό του διαγράμματος (πράσινα ορθογώνια). Οι περαιτέρω λειτουργίες και ενέργειες αυτών των υπηρεσιών, δίνονται με τα επιμέρους κομμάτια που περιγράφουν τη ροή του προγράμματος για κάθε ξεχωριστή υπηρεσία (μπλε μονοπάτια).

3.2 Λογική σχεδιασμού

Το ερώτημα που τίθεται είναι, "Γιατί επιλέχθηκε το λειτουργικό σύστημα Android ;"

Η επιλογή του λειτουργικού συστήματος Android έγινε σκόπιμα, καθώς είναι το πιο ευρέως διαδεδομένο λογισμικό στον κόσμο. Χρησιμοποιείται σε πάνω από ένα δισεκατομμύριο συσκευές υλικού , σε όλο τον κόσμο , συμπεριλαμβανομένων των "έξυπνων κινητών τηλεφώνων"(smartphones), "έξυπνων τηλεοράσεων" (smart-tvs) , "έξυπνων ρολογιών" (smartwear) και πολλών επερχόμενων φορητών συσκευών . Το Android είναι ιδιαιτέρως φιλικό προς τον χρήστη και συνεργάζεται άψογα με διάφορες ήδη δημοφιλείς εφαρμογές της Google. Οι συσκευές με Android σύμφωνα με στατιστικές έρευνες, που έχουν διεξαχθεί, έχουν υψηλότερες πωλήσεις από όλες τις συσκευές BlackBerry,Windows phone και iOS μαζί. Η ακόλουθη εικόνα δείχνει ακριβώς αυτά τα στοιχεία, παρουσιάζοντας τα ποσοστά των πωλήσεων έξυπνων τηλεφώνων σε παγκόσμια κλίμακα από το έτος 2010 μέχρι το τέλος του 2014, διαχωρισμένες ανάλογα με το λογισμικό που χρησιμοποιούν οι συσκευές.



Εικόνα 3.2.1:Πωλήσεις έξυπνων τηλεφώνων παγκοσμίως, 2010-2014

4 Υλοποίηση

4.1 Ανάλυση λεπτομερειών υλοποίησης εφαρμογής

Γενικές έννοιες.

Στο κεφάλαιο αυτό θα γίνει μία λεπτομερής παρουσίαση της υλοποίησης της εφαρμογής. Πριν την περιγραφή του κώδικα της εφαρμογής, είναι σημαντικό να ορίσουμε κάποιες έννοιες και οντότητες που θα χρησιμοποιηθούν στη συνέχεια.

- **Activity:** Η κλάση Activity παρέχει μία οθόνη αλληλεπίδρασης με το χρήστη μέσω της μεθόδου setContentView(View). Η μέθοδος αυτή καλείται μέσα στη συνάρτηση onCreate(Bundle), η οποία αρχικοποιεί το activity μας.
- **Fragment:** Είναι ένα κομμάτι από την διεπαφή χρήστη μίας εφαρμογής, το οποίο τοποθετείται μέσα σε ένα activity και εξαρτάται άμεσα από αυτό . Η αλληλεπίδραση με τα fragments μέσα σε ένα activity γίνεται μέσω του FragmentManager. Στην εφαρμογή αυτή όλες οι κύριες επιλογές του χρήστη έχουν υλοποιηθεί ως διαφορετικά fragments, ο έλεγχος των διαφορετικών λειτουργιών γίνεται αλλάζοντας fragments αντί για να αλλάζουμε διαφορετικά activities.Αυτό την καθιστά πιο γρήγορη στην απόκριση.
- **ListView:** Είναι μία ομαδική προβολή (view group), η οποία αναπαριστά μία λίστα από αντικείμενα με δυνατότητα κύλισης. Τα αντικείμενα εισέρχονται αυτόματα στη λίστα μέσω ενός αντάπτορα (Adapter).
- **DrawerLayout:** Επιτρέπει "drawer" προβολές να εμφανίζονται από τη γωνία του παραθύρου.
- **TextView:** Παρουσιάζει κείμενο στο χρήστη.
- **ArrayAdapter:** Η κατηγορία ArrayAdapter μπορεί να χειριστεί μία λίστα ή πίνακα αντικειμένων Java ως είσοδο. Κάθε αντικείμενο αντιστοιχίζεται σε μία γραμμή και από προεπιλογή χαρτογραφεί τη μέθοδο toString() του αντικειμένου σε μία προβολή στο layout των γραμμών.
- **ListAdapter:** Περιέχει τον τρόπο που φαίνονται οι λίστες στον χρήστη.
- **JSON:** (JavaScript Object Notation) Είναι μία ανεξαρτήτου γλώσσας ανοιχτού προτύπου μορφή (open standard format), που χρησιμοποιεί αναγνωρίσιμο από τον

άνθρωπο (human readable) κείμενο για να μεταδώσει δεδομένα-αντικείμενα. Χρησιμοποιείται συνήθως για να μεταδώσει δεδομένα μεταξύ ενός εξυπηρετητή και μιας εφαρμογής ως ένα εναλλακτικό XML. Με το σύμβολο "{" ορίζεται ένα JSON αντικείμενο, ενώ με το σύμβολο "[" ορίζεται ένας JSON πίνακας.

- **GSON:** Η GSON είναι μία βιβλιοθήκη που χρησιμοποιείται για μετατροπή Java αντικειμένων στην JSON αναπαράσταση τους. Επίσης μπορεί να χρησιμοποιηθεί για μετατροπή ενός JSON string σε ένα αντίστοιχο Java αντικείμενο. Ως παράδειγμα εντοπίζουμε την χρήση του στην εφαρμογή για τους πίνακες των φυσιολογικών τιμών και του θερμιδομετρητή, που από πίνακες με δεδομένα τύπου JSON κάνουμε τα δεδομένα αυτά αντιστοίχιση (map) σε αντικείμενα κλάσεων.

Περιγραφή επιμέρους υπομονάδων.

Main.java

Στο αρχείο αυτό περιέχεται ο κώδικας που ορίζει την βασική δομή και το γενικό έλεγχο όλων των λειτουργιών της εφαρμογής. Αποτελεί την κύρια κλάση της εφαρμογής. Μέσα σε αυτήν την κλάση καλείται η συνάρτηση onCreate() για την αρχικοποίηση του MainActivity. Μέσα σε αυτό το αρχείο ορίζονται τα fragments που διαχειρίζονται τη συμπεριφορά, τις αλληλεπιδράσεις και την παρουσίαση του navigation drawer.

Με τη χρήση του Android NavigationDrawer είναι δυνατή η εναλλαγή μεταξύ των διαφορετικών fragments που έχουν υλοποιηθεί για την κάθε λειτουργία της εφαρμογής. Όπως φαίνεται στο παρακάτω κομμάτι κώδικα που παρατίθεται με την χρήση της μεθόδου onNavigationDrawerItemSelected(), ανάλογα με την διαφορετική επιλογή που θα γίνει, με ένα switch case statement δημιουργείται το αντίστοιχο fragment.

Ακόμα, στην κλάση αυτή ορίζονται και όλες οι βασικές μέθοδοι που χρησιμοποιούνται σε άλλα επιμέρους αρχεία.

```
01. public void onNavigationDrawerItemSelected(int position) {
02.         Fragment fragment = null;
03.         switch (position) {
04.             case 0:
05.                 mTitle = getString(R.string.section_me);
06.                 fragment = new Me();
07.                 break;
08.             case 1:
09.                 mTitle = getString(R.string.section_pressure);
10.                 fragment = new BloodPressure();
11.                 break;
12.             case 2:
13.                 mTitle = getString(R.string.section_sugar);
14.                 fragment = new BloodSugar();
15.                 break;
16.             case 3:
17.                 mTitle = getString(R.string.section_contacts);
18.                 fragment = new Contacts();
19.                 break;
20.             case 4:
21.                 mTitle = getString(R.string.section_doctors);
22.                 fragment = new Doctors();
23.                 break;
24.             case 5:
25.                 mTitle = getString(R.string.section_vitalstatistics);
26.                 fragment = new VitalStatistics();
27.                 break;
28.             case 6:
29.                 mTitle = getString(R.string.section_caloriemeter);
30.                 fragment = new Foods();
31.                 break;
32.             case 7:
33.                 mTitle = getString(R.string.section_calendar);
34.                 fragment = new Calendar();
35.                 break;
36.             default:
37.                 break;
38.         }
```

Κώδικας 4.1.1: Δημιουργία αντίστοιχου fragment

NavigationDrawerFragment.java

Το fragment αυτό χρησιμοποιείται για να διαχειριστεί τις αλληλεπιδράσεις και την παρουσίαση του navigation drawer, δηλαδή το πώς θα πλοηγηθεί ο χρήστης μέσα στην εφαρμογή. Στην πιο αφηρημένη του έννοια στο fragment αυτό υλοποιείται το πλαινό μενού και ορίζονται οι λειτουργίες που πρέπει να εκτελεστούν, έπειτα από μία συγκεκριμένη επιλογή που κάνει ο χρήστης από αυτό το μενού. Αυτό το fragment επηρεάζει άμεσα τις ενέργειες που γίνονται στην action bar. Επίσης κάθε φορά που επιλεγεί διαφορετικό fragment με τις αντίστοιχες λειτουργίες, θέτει και τον αντίστοιχο τίτλο στην αντίστοιχη οθόνη. Δίνει πρόσβαση στο χρήστη στις διαφορετικές οθόνες-λειτουργίες της εφαρμογής. Η action bar της εφαρμογής όπως φαίνεται και στο ακόλουθο κομμάτι κώδικα, είναι υλοποιημένη να αποκρίνεται σαν διαφορετικά μενού κάθε φορά που ο χρήστης ακουμπά κάποιο συγκεκριμένο αντικείμενο της.

```
01.  public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {
02.          // If the drawer is open, show the global app actions in the action bar. See also
03.          // showGlobalContextActionBar, which controls the top-left area of the action bar.
04.          if (mDrawerLayout != null && isDrawerOpen()) {
05.              inflater.inflate(R.menu.global, menu);
06.              showGlobalContextActionBar();
07.          } else {
08.              switch (mCurrentSelectedPosition) {
09.                  case 1:
10.                      inflater.inflate(R.menu.bloodpressuremenu, menu);
11.                      break;
12.                  case 2:
13.                      inflater.inflate(R.menu.bloodsugarmenu, menu);
14.                      break;
15.                  case 3:
16.                      inflater.inflate(R.menu.contactmenu, menu);
17.                      break;
18.                  case 4:
19.                      inflater.inflate(R.menu.doctormenu, menu);
20.                      break;
21.              }
22.          }
23.          super.onCreateOptionsMenu(menu, inflater);
24.      }
```

Κώδικας 4.1.2: Υλοποίηση επιμέρους μενού στην action bar

Σε κάθε διαφορετικό αντικείμενο που επιλέγεται από το μενού, χρησιμοποιώντας ένα switch case statement, δημιουργείται ένα παράθυρο διαλόγου και υλοποιείται οτιδήποτε αυτό περιέχει. Στον κώδικα 4.1.3 φαίνονται οι ενέργειες αυτές για την κάθε περίπτωση που μπορεί να επιλέξει ο χρήστης. Εάν επιλέξει για παράδειγμα η «πίεση μου» δημιουργείται ένα bloodPressureDialog και υλοποιείται το αντίστοιχο fragment.

```
@Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (mDrawerToggle.onOptionsItemSelected(item)) {
            return true;
        }
        switch (item.getItemId()) {
            case R.id.add_new_blood_pressure:
                BloodPressuresDialog bloodPressureDialog = new BloodPressuresDialog();
                bloodPressureDialog.show(getFragmentManager(), "NEW_BLOODPRESSURE");
                break;
            case R.id.add_new_blood_sugar:
                BloodSugarsDialog bloodSugarsDialog = new BloodSugarsDialog();
                bloodSugarsDialog.show(getFragmentManager(), "NEW_BLOODSUGAR");
                break;
            case R.id.add_new_contact:
                ContactDialog contactDialog = new ContactDialog();d
                contactDialog.show(getFragmentManager(), "NEW_CONTACT");
                break;
            case R.id.add_new_doctor:
                DoctorDialog doctorDialog = new DoctorDialog();
                doctorDialog.show(getFragmentManager(), "NEW_DOCTOR");
                break;
        }

        return super.onOptionsItemSelected(item);
    }
```

Κώδικας 4.1.3: Διαφορετικές ενέργειες ανάλογα με την επιλογή που θα γίνει

Dialogs

Στο κομμάτι αυτό θα περιγράψουμε τις ενέργειες που εκτελούνται και τον κώδικα του κάθε διαλόγου των επιμέρους λειτουργιών.

BloodPressuresDialog.java

Στην κλάση αυτή υλοποιείται κάθε κομμάτι που χρειάζεται για το διάλογο της εφαρμογής με το χρήστη, όσον αφορά τις μετρήσεις πίεσης. Χρησιμοποιείται η μέθοδος onCreateDialog(Bundle) και ορίζονται όλα τα πεδία τα οποία εμφανίζονται στο διάλογο αυτό. Εμφανίζεται κείμενο για όλα τα πεδία της φόρμας, που ζητείται να συμπληρωθεί από το χρήστη. Συμπεριλαμβάνονται το προβαλλόμενο κείμενο που ζητείται από το χρήστη να συμπληρώσει χαμηλή και υψηλή πίεση, ο επιλογέας(picker) ημερομηνίας και ο περιστροφέας(spinner) της ώρας της καταχώρησης της μέτρησης. Στο setPositiveButton ορίζονται οι ενέργειες που πρέπει να γίνουν αν ο χρήστης επιλέξει να αποθηκεύσει τις τιμές που εισήγαγε και ακόμη ορίζεται η ενέργεια αν πατηθεί το κουμπί Άκυρο, όπως φαίνεται στο κομμάτι κώδικα 4.1.4. Η κλάση αυτή ακόμα περιέχει συνάρτηση για γράψιμο σε αρχείο των δεδομένων αυτών για

αποθήκευση τους και για ανάγνωση από αρχείο, όπως φαίνονται στον κώδικα 4.1.5

```
@Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        AlertDialog.Builder dialogBuilder = new AlertDialog.Builder(this.getActivity());
        LayoutInflater inflater = getActivity().getLayoutInflater();
        View v = inflater.inflate(R.layout.new_blood_pressure_dialog, null);
        final EditText lowPressure = (EditText) v.findViewById(R.id.lowPressureValue);
        final EditText highPressure = (EditText) v.findViewById(R.id.highPressureValue);
        final DatePicker dateMeasure = (DatePicker) v.findViewById(R.id.pressureDatePicker);
        final Spinner timeMeasure = (Spinner) v.findViewById(R.id.pressureSpinner);
        dialogBuilder.setView(v)
                .setPositiveButton("Apothikeusi", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        File bloodPressuresInfo = getActivity().getApplicationContext().getFileStreamPath(bloodPressureFile);
                        if (bloodPressuresInfo.exists()) {
                            newMeasure = new Measure(dateMeasure.getDayOfMonth() + "/" +
 dateMeasure.getMonth() + "/" + dateMeasure.getYear(), timeMeasure.getSelectedItem().toString(),
lowPressure.getText().toString(), highPressure.getText().toString());
                            bloodPressures = readFromFile(bloodPressureFile);
                            measurementsArrayList = new ArrayList<>();
                            if (bloodPressures.length() > 2) {
                                measurementsArrayList = new Gson().fromJson(bloodPressures, new TypeToken<ArrayList<Measure>>() {
                                }.getType());
                            }
                            measurementsArrayList.add(newMeasure);
                            String x = new Gson().toJson(measurementsArrayList);
                            writeToFile(x, bloodPressureFile);
                            EventBus.getDefault().post(new UpdateBloodMeasuresUI());
                        }

                    }
                })
                .setNegativeButton("Akyro", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {

                    }
                });
        return dialogBuilder.create();
    }
```

Κώδικας 4.1.4 : onCreateDialog στην BloodPressuresDialog

```java
/**
     * Function to write to a file
     * @param data
     * @param file
     */
    private void writeToFile(String data, String file) {
        try {
            OutputStreamWriter outputStreamWriter = new OutputStreamWriter(getActivity().openFileOutput(file, Context.MODE_PRIVATE));
            outputStreamWriter.write(data);
            outputStreamWriter.close();
        } catch (IOException e) {
            Log.e("Exception", "File write failed: " + e.toString());
        }
    }

    /**
     * Function to read from a file
     * @param file
     * @return the file contents as a string
     */k
    private String readFromFile(String file) {

        String ret = "";

        try {
            InputStream inputStream = getActivity().openFileInput(file);

            if (inputStream != null) {
                InputStreamReader inputStreamReader = new InputStreamReader(inputStream);
                BufferedReader bufferedReader = new BufferedReader(inputStreamReader);
                String receiveString = "";
                StringBuilder stringBuilder = new StringBuilder();

                while ((receiveString = bufferedReader.readLine()) != null) {
                    stringBuilder.append(receiveString);
                }

                inputStream.close();
                ret = stringBuilder.toString();
            }
        } catch (FileNotFoundException e) {
            Log.e("login activity", "File not found: " + e.toString());
        } catch (IOException e) {
            Log.e("login activity", "Can not read file: " + e.toString());
        }

        return ret;
    }

}
```

Κώδικας 4.1.5: Εγγραφή & ανάγνωση αρχείου

BloodSugarDialog.java

Στην κλάση αυτή υλοποιείται κάθε κομμάτι που χρειάζεται για το διάλογο της εφαρμογής με το χρήστη όσον αφορά της μετρήσεις σακχάρου. Χρησιμοποιείται η μέθοδος onCreateDialog(Bundle) και ορίζονται όλα τα πεδία τα οποία εμφανίζονται στο διάλογο αυτό. Εμφανίζεται κείμενο για όλα τα πεδία της φόρμας, που ζητείται να συμπληρωθεί από το χρήστη. Συμπεριλαμβάνονται το προβαλλόμενο κείμενο που ζητείται από το χρήστη να συμπληρώσει την τιμή σακχάρου, ο επιλογέας(picker) ημερομηνίας της καταχώρησης της μέτρησης. Στο setPositiveButton ορίζονται οι ενέργειες που πρέπει να γίνουν αν ο χρήστης επιλέξει να αποθηκεύσει τις τιμές που

εισήγαγε και ακόμη ορίζεται η ενέργεια αν πατηθεί το κουμπί Άκυρο, όπως φαίνεται στο κομμάτι κώδικα 4.1.6. Η κλάση αυτή ακόμα περιέχει συνάρτηση για γράψιμο σε αρχείο των δεδομένων αυτών για αποθήκευση τους και συνάρτηση για ανάγνωση από αρχείο, όπως και προηγούμενα.

```
@Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        AlertDialog.Builder dialogBuilder = new AlertDialog.Builder(this.getActivity());
        LayoutInflater inflater = getActivity().getLayoutInflater();
        View v = inflater.inflate(R.layout.new_blood_sugar_dialog, null);
        final EditText sugarsMeasure = (EditText) v.findViewById(R.id.sugarsValue);
        final DatePicker dateMeasure = (DatePicker) v.findViewById(R.id.sugarsDatePicker);
        dialogBuilder.setView(v)
                .setPositiveButton("Apothikeusi", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        File bloodSugarsInfo = getActivity().getApplicationContext().getFileStreamPath(bloodSugarFile);
                        if (bloodSugarsInfo.exists()) {
                            newMeasure = new Measure(dateMeasure.getDayOfMonth() + "/" +
dateMeasure.getMonth() + "/" + dateMeasure.getYear(), "", sugarsMeasure.getText().toString(),
sugarsMeasure.getText().toString());
                            bloodSugars = readFromFile(bloodSugarFile);
                            measurementsArrayList = new ArrayList<>();
                            if (bloodSugars.length() > 2) {
                                measurementsArrayList = new Gson().fromJson(bloodSugars, new TypeToken<ArrayList<Measure>>() {
                                }.getType());
                            }
                            measurementsArrayList.add(newMeasure);
                            writeToFile(new Gson().toJson(measurementsArrayList), bloodSugarFile);
                            EventBus.getDefault().post(new UpdateBloodSugarsUI());
                        }

                    }
                })
                .setNegativeButton("Akyro", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {

                    }
                });
        return dialogBuilder.create();
    }
```

Κώδικας 4.1.6 : onCreateDialog στην BloodSugarDialog

ContactDialog.java

Στην κλάση αυτή υλοποιείται κάθε κομμάτι που χρειάζεται για το διάλογο της εφαρμογής με το χρήστη αντίστοιχα αυτή τη φορά για την καταχώρηση επαφών οικείων προσώπων. Χρησιμοποιείται η μέθοδος onCreateDialog(Bundle) και ορίζονται όλα τα πεδία τα οποία εμφανίζονται στο διάλογο αυτό. Εμφανίζεται κείμενο για όλα τα πεδία της φόρμας, που ζητείται να συμπληρωθεί από το χρήστη. Συμπεριλαμβάνονται το προβαλλόμενο κείμενο που ζητείται από το χρήστη να συμπληρώσει όνομα, επώνυμο, συγγένεια ή σχέση ,τηλέφωνο σπιτιού και γραφείου και κινητό. Στο setPositiveButton ορίζονται οι ενέργειες που πρέπει να γίνουν αν ο χρήστης επιλέξει να αποθηκεύσει τις τιμές που εισήγαγε και ακόμη ορίζεται η ενέργεια αν πατηθεί το κουμπί Άκυρο, όπως φαίνεται στο κομμάτι κώδικα 4.1.7. Η κλάση αυτή ακόμα περιέχει συνάρτηση για γράψιμο σε αρχείο των δεδομένων αυτών για αποθήκευση τους και για ανάγνωση από αρχείο, όπως και προηγούμενα.

```
@Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        AlertDialog.Builder dialogBuilder = new AlertDialog.Builder(this.getActivity());
        LayoutInflater inflater = getActivity().getLayoutInflater();
        View v = inflater.inflate(R.layout.new_contact_dialog_layout, null);
        final EditText firstName = (EditText) v.findViewById(R.id.contactFirstName);
        final EditText lastName = (EditText) v.findViewById(R.id.contactLastName);
        final EditText relationship = (EditText) v.findViewById(R.id.contactRelationship);
        final EditText homePhone = (EditText) v.findViewById(R.id.contactHomePhone);
        final EditText officePhone = (EditText) v.findViewById(R.id.contactOfficePhone);
        final EditText mobilePhone = (EditText) v.findViewById(R.id.contactMobilePhone);
        dialogBuilder.setView(v)
                .setPositiveButton("Apothikeusi", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        Contact newContact = new Contact();
                        File contactsInfo = getActivity().getApplicationContext().getFileStreamPath(contactsFile);
                        if (contactsInfo.exists()) {
                            newContact.setFirstName(firstName.getText().toString());
                            newContact.setLastName(lastName.getText().toString());
                            newContact.setRelationship(relationship.getText().toString());
                            newContact.setHomePhone(homePhone.getText().toString());
                            newContact.setWorkPhone(officePhone.getText().toString());
                            newContact.setMobile(mobilePhone.getText().toString());
                            contacts = readFromFile(contactsFile);
                            contactsArrayList = new ArrayList<>();
                            if (contacts.length() > 2) {
                                contactsArrayList = new Gson().fromJson(contacts, new TypeToken<ArrayList<Contact>>() {
                                }.getType());
                            }
                            contactsArrayList.add(newContact);
                            writeToFile(new Gson().toJson(contactsArrayList), contactsFile);
                            EventBus.getDefault().post(new UpdateContactsUI());
                        }

                    }
                })
                .setNegativeButton("Akyro", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {

                    }
                });
        return dialogBuilder.create();
    }
```

Κώδικας 4.1.7 : onCreateDialog στην ContactDialog

DoctorDialog.java

Στην κλάση αυτή υλοποιείται κάθε κομμάτι που χρειάζεται για το διάλογο της εφαρμογής με το χρήστη, αντίστοιχα, αυτή τη φορά για την καταχώρηση ιατρών. Χρησιμοποιείται η μέθοδος onCreateDialog(Bundle) και ορίζονται όλα τα πεδία τα οποία εμφανίζονται στο διάλογο αυτό. Εμφανίζεται κείμενο για όλα τα πεδία της φόρμας, που ζητείται να συμπληρωθεί από το χρήστη. Συμπεριλαμβάνονται το προβαλλόμενο κείμενο που ζητείται από το χρήστη να συμπληρώσει όνομα, επώνυμο, ειδικότητα, διεύθυνση, τηλέφωνο γραφείου και κινητό . Στο setPositiveButton ορίζονται οι ενέργειες που πρέπει να γίνουν αν ο χρήστης επιλέξει

να αποθηκεύσει τις τιμές που εισήγαγε και ακόμη ορίζεται η ενέργεια αν πατηθεί το κουμπί Άκυρο, όπως φαίνεται στο κομμάτι κώδικα 4.1.8. Η κλάση αυτή ακόμα περιέχει συνάρτηση για γράψιμο σε αρχείο των δεδομένων αυτών για αποθήκευση τους και συνάρτηση για ανάγνωση από αρχείο, όπως και προηγούμενα.

```
@Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        AlertDialog.Builder dialogBuilder = new AlertDialog.Builder(this.getActivity());
        LayoutInflater inflater = getActivity().getLayoutInflater();
        View v = inflater.inflate(R.layout.new_doctor_dialog, null);
        final EditText firstName = (EditText) v.findViewById(R.id.doctorsFirstName);
        final EditText lastName = (EditText) v.findViewById(R.id.doctorsLastName);
        final EditText specialty = (EditText) v.findViewById(R.id.doctorsSpecialty);
        final EditText officeAddress = (EditText) v.findViewById(R.id.doctorsOfficeAddress);
        final EditText officePhone = (EditText) v.findViewById(R.id.doctorsOfficePhone);
        final EditText mobilePhone = (EditText) v.findViewById(R.id.doctorsMobilePhone);
        dialogBuilder.setView(v)
                .setPositiveButton("Apothikeusi", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        Doctor newDoctor = new Doctor();
                        File doctorsInfo = getActivity().getApplicationContext().getFileStreamPath(doctorsFile);
                        if (doctorsInfo.exists()) {
                            newDoctor.setFirstName(firstName.getText().toString());
                            newDoctor.setLastName(lastName.getText().toString());
                            newDoctor.setSpecialty(specialty.getText().toString());
                            newDoctor.setOfficeAddress(officeAddress.getText().toString());
                            newDoctor.setWorkPhone(officePhone.getText().toString());
                            newDoctor.setMobile(mobilePhone.getText().toString());
                            doctors = readFromFile(doctorsFile);
                            doctorsArrayList = new ArrayList<>();
                            if (doctors.length() > 2) {
                                doctorsArrayList = new Gson().fromJson(doctors, new TypeToken<ArrayList<Doctor>>() {
                                }.getType());
                            }
                            doctorsArrayList.add(newDoctor);
                            writeToFile(new Gson().toJson(doctorsArrayList), doctorsFile);
                            EventBus.getDefault().post(new UpdateDoctorsUI());
                        }

                    }
                })
                .setNegativeButton("Akyro", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {

                    }
                });
        return dialogBuilder.create();
    }
```

Κώδικας 4.1.8 : onCreateDialog στην DoctorDialog

JSON αρχεία.

Στο σημείο αυτό θα δώσουμε ένα παράδειγμα αρχείου JSON ενδεικτικό από την υλοποίηση ώστε να κατανοηθεί η συγγραφή και η λειτουργία του. Όπως αναφέρθηκε και στην αρχή του κεφαλαίου με το σύμβολο "{" ορίζεται ένα JSON αντικείμενο, ενώ με το σύμβολο "[" ορίζεται ένας JSON πίνακας. Στην συγκεκριμένη εφαρμογή χρησιμοποιήθηκαν JSON πίνακες από αντικείμενα JSON για την περιγραφή των φυσιολογικών εξετάσεων και του θερμιδομετρητή, όπως φαίνεται στον παρακάτω κώδικα.

```
[ ⊖
   { ⊖
      "name":"ΣΑΚΧΑΡΟ",
      "value":"70  -  101"
   },
   { ⊖
      "name":"ΟΥΡΙΑ",
      "value":"10 - 50"
   },
   { ⊖
      "name":"ΚΡΕΑΤΙΝΙΝΗ ΑΝΤΡΕΣ",
      "value":"0.7 - 1.1"
   }
]
```

Κώδικας 4.1.9: Στιγμιότυπο vitals.json

Ακόμα υπάρχουν κλάσεις που ορίζουν τους adapters που επεκτείνουν τη χρήση του ArrayAdapter για την δημιουργία της διεπαφής με το χρήστη και την υλοποίηση του ListView. Οι κλάσεις αυτές είναι οι BloodPressureMeasurementsListAdapter.java, BloodSugarsMeasurementsListAdapter.java , ContactsListAdapter.java, DoctorListAdapter.java και InfoListAdapter.java. Καθεμία από τις οποίες με τη βοήθεια της getView() αρχικοποιεί και θέτει τον adapter που χρειάζεται(getView function to customize our adapter) για να προβάλλει και να κρατήσει τιμές για τα αντίστοιχα κελιά που θα συμπληρωθούν στις φόρμες διαλόγου με το χρήστη.

Στην υλοποίηση ακόμα υπάρχουν αρχεία XML τα οποία χρειάζονται για να οριστούν παράμετροι στα layouts των fragments, των list_item και των fragment_grids. Ακόμα όπως σε κάθε εφαρμογή Android υπάρχει και το αρχείο AndroidManifest.xml , στο οποίο ορίζονται άδειες (permissions) και περιορισμοί καθώς και συγκεκριμένες παράμετροι για κωδικοποιήσεις κειμένου.

5 Διεπαφή χρήστη εφαρμογής

5.1 Παρουσίαση εφαρμογής

Στο κεφάλαιο αυτό γίνεται μία παρουσίαση της εφαρμογής από την οπτική γωνία του χρήστη. Στις παραγράφους που ακολουθούν, δίνονται στιγμιότυπα εκτέλεσης της εφαρμογής, ώστε να περιγραφεί η διεπαφή του χρήστη, UI(user interface). Με τον όρο διεπαφή ή γραφικό περιβάλλον χρήστη καλείται στην πληροφορική ένα σύνολο γραφικών στοιχείων, τα οποία εμφανίζονται στην οθόνη κάποιας ψηφιακής συσκευής και χρησιμοποιούνται για την αλληλεπίδραση του χρήστη με κάποια δεδομένη εφαρμογή. Η αλληλεπίδραση έχει στόχο να επιτρέψει την αποτελεσματική λειτουργία και τον έλεγχο της συσκευής από την πλευρά του χρήστη, ενώ η συσκευή ανατροφοδοτεί ταυτόχρονα πληροφορίες για την διαδικασία λήψης αποφάσεων. Παρέχονται στο χρήστη, ενδείξεις και εργαλεία προκειμένου να φέρει εις πέρας τις προσφερόμενες, από την εφαρμογή λειτουργίες. Για το λόγο αυτό οι εφαρμογές δέχονται είσοδο από το χρήστη και αντιδρούν ανάλογα στα συμβάντα που αυτός πυροδοτεί.

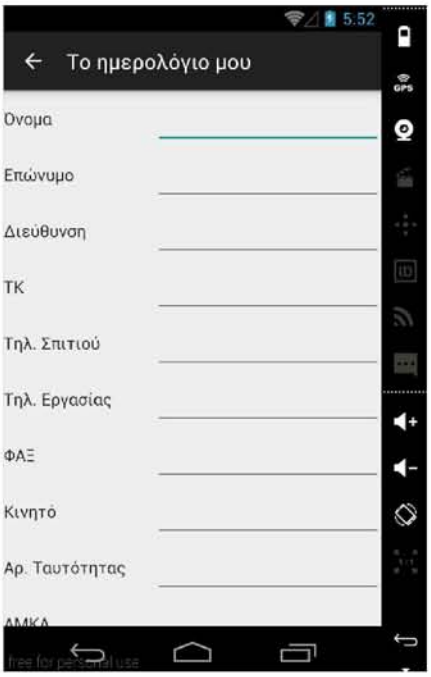
5.2 Εικονίδιο εφαρμογής

Η εικόνα 5.2.1 αποτελεί το λογότυπο της εφαρμογής.

Εικόνα 5.2.1: Εικονίδιο εφαρμογής

5.3 Εκκίνηση – Αρχική οθόνη εφαρμογής

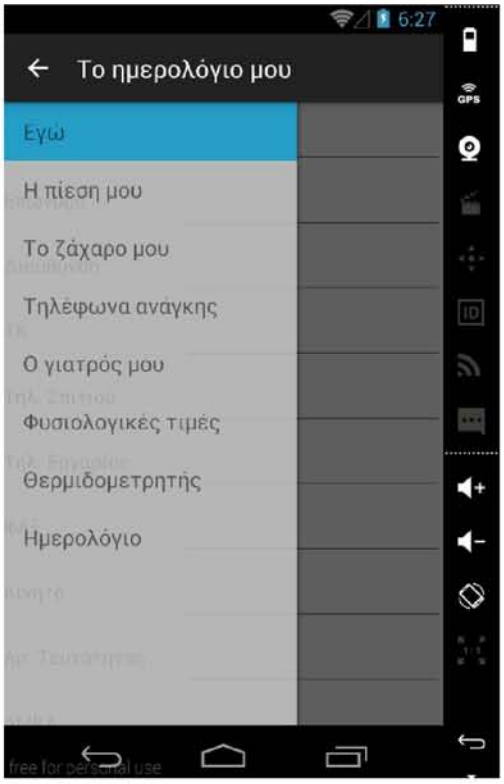
Η εικόνα 5.3.1 δείχνει την αρχική οθόνη που εμφανίζεται με το κλικ στο εικονίδιο της εφαρμογής. Η εφαρμογή με την εκκίνηση εμφανίζει την παρακάτω οθόνη, στην οποία ο χρήστης καλείται να συμπληρώσει τα προσωπικά του στοιχεία, εφόσον είναι η πρώτη φορά που χρησιμοποιεί την εφαρμογή. Εάν ο χρήστης έχει χρησιμοποιήσει στο παρελθόν την εφαρμογή και έχει αποθηκεύσει ήδη τα προσωπικά του δεδομένα, τότε η αρχική οθόνη εμφανίζει στα αντίστοιχα πεδία τα στοιχεία αυτά. Σέρνοντας με το δάκτυλο την οθόνη κάθετα, υπάρχει η δυνατότητα κύλισης που δίνει πρόσβαση σε όλα τα πεδία της φόρμας. Πατώντας πάνω σε ένα οποιοδήποτε κελί της φόρμας αναδύεται πληκτρολόγιο και ο χρήστης εισάγει δεδομένα στην εφαρμογή. Πατώντας το βέλος πάνω αριστερά στην οθόνη, δίπλα στην ετικέτα "Το ημερολόγιο μου", αναδύεται το μενού με τις κύριες επιλογές και λειτουργίες της εφαρμογής.



Εικόνα 5.3.1: Αρχική οθόνη εφαρμογής

5.4 Εμφάνιση πλευρικού μενού

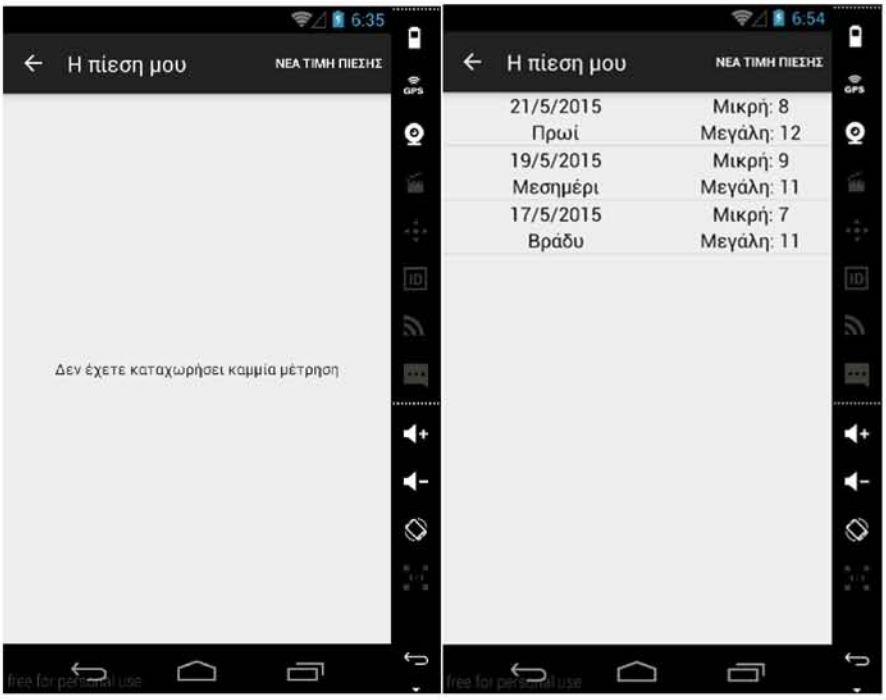
Η εικόνα 5.4.1 παρουσιάζει το πλευρικό μενού που αναδύεται, κάθε φορά που πατηθεί το βέλος επιστροφής, το οποίο εντοπίζεται δίπλα και αριστερά από την ετικέτα της κάθε επιμέρους λειτουργίας της εφαρμογής. Με αυτό τον τρόπο ο χρήστης μπορεί να πλοηγηθεί ανάμεσα στις διαφορετικές καρτέλες-υπηρεσίες, που του προσφέρει η εφαρμογή.



Εικόνα 5.4.1: Πλευρικό μενού πλοήγησης

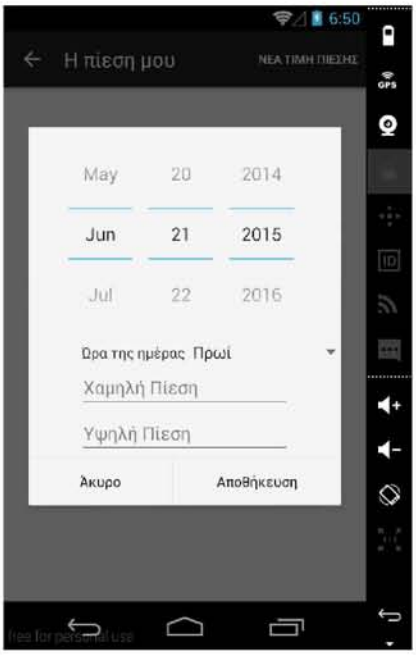
5.5 Η πίεση μου

Στην εικόνα 5.5.1 (αριστερά), παρουσιάζεται η επιλογή της καρτέλας καταγραφής μετρήσεων πίεσης του χρήστη. Εάν ο χρήστης χρησιμοποιεί την εφαρμογή για πρώτη φορά, τότε στο κυρίως πλαίσιο της οθόνης εμφανίζεται το μήνυμα προς το χρήστη «Δεν έχετε καταχωρήσει καμία μέτρηση». Σε αντίθετη περίπτωση, στο πλαίσιο εμφανίζεται το ιστορικό μετρήσεων πίεσης του χρήστη, όπως φαίνεται στην εικόνα 5.5.1(δεξιά).



Εικόνα 5.5.1:Προβολή καρτέλας «Η πίεση μου» χωρίς ιστορικό μετρήσεων (αριστερά), με ιστορικό (δεξιά)

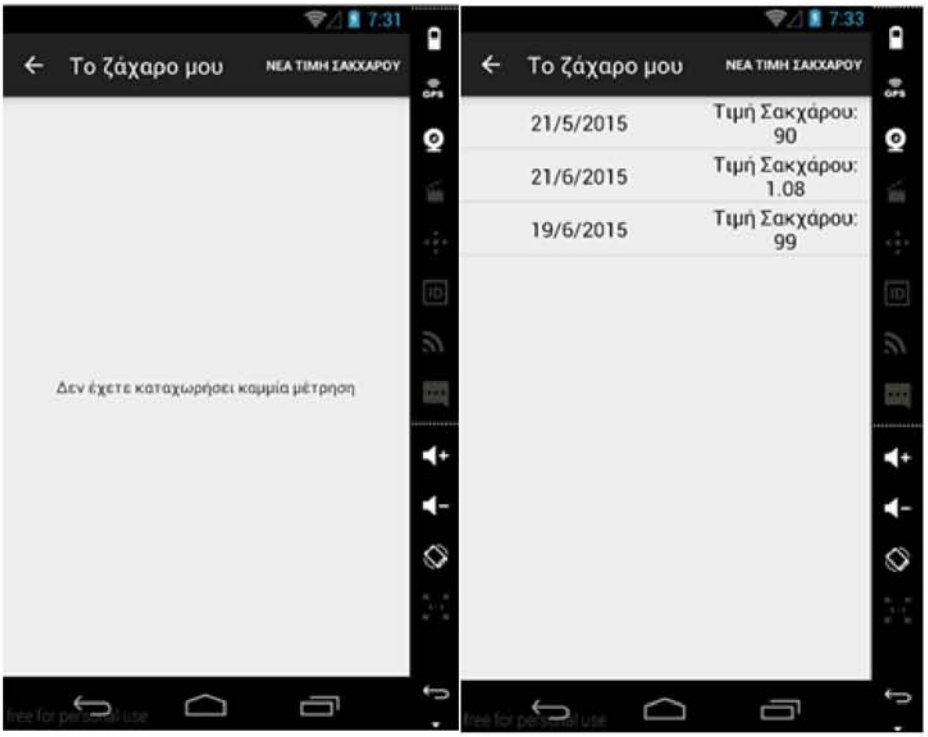
Εάν ο χρήστης επιθυμεί να καταχωρήσει νέα τιμή πίεσης, πατώντας στο αντίστοιχο κουμπί, αναδύεται το παράθυρο διαλόγου που φαίνεται στην εικόνα 5.5.2. Ο χρήστης χρησιμοποιώντας δυνατότητα κύλισης μπορεί να επιλέξει το μήνα, την ημέρα και το έτος αντίστοιχα της νέας καταχώρησης. Στο μέσο του παραθύρου διαλόγου ο χρήστης μπορεί πατώντας πάνω στο κατερχόμενο βέλος να επιλέξει ανάμεσα στις ώρες καταχώρησης, που ορίζονται ως «πρωί», «μεσημέρι» και «βράδυ» . Ακολουθούν δύο κελιά στα οποία με την ανάδυση ενός αριθμητικού πληκτρολογίου, ο χρήστης μπορεί να δηλώσει την μετρούμενη τιμή. Η επιλογή του κουμπιού «Αποθήκευση», αποθηκεύει τα δεδομένα αυτά και η αντίστοιχη επιλογή του κουμπιού «Άκυρο», τερματίζει την ενέργεια της καταχώρησης νέας τιμής πίεσης ως εσφαλμένη.



Εικόνα 5.5.2:Παράθυρο διαλόγου νέας καταχώρησης

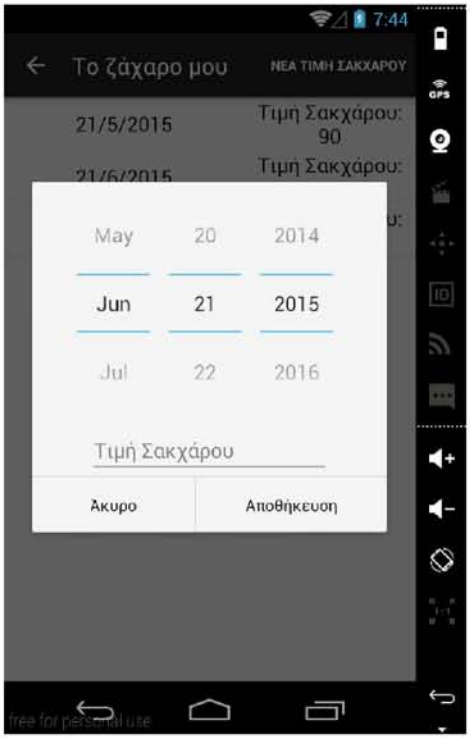
5.6 Το ζάχαρο μου

Στην εικόνα 5.6.1 (αριστερά), παρουσιάζεται η επιλογή της καρτέλας καταγραφής μετρήσεων σακχάρου του χρήστη. Εάν ο χρήστης χρησιμοποιεί την εφαρμογή για πρώτη φορά, τότε στο κυρίως πλαίσιο της οθόνης εμφανίζεται το μήνυμα προς το χρήστη «Δεν έχετε καταχωρήσει καμία μέτρηση». Σε αντίθετη περίπτωση, στο πλαίσιο εμφανίζεται το ιστορικό μετρήσεων σακχάρου του χρήστη, όπως φαίνεται στην εικόνα 5.6.1(δεξιά).



Εικόνα 5.6.1:Προβολή καρτέλας «Το ζάχαρο μου» χωρίς ιστορικό μετρήσεων (αριστερά), με ιστορικό (δεξιά)

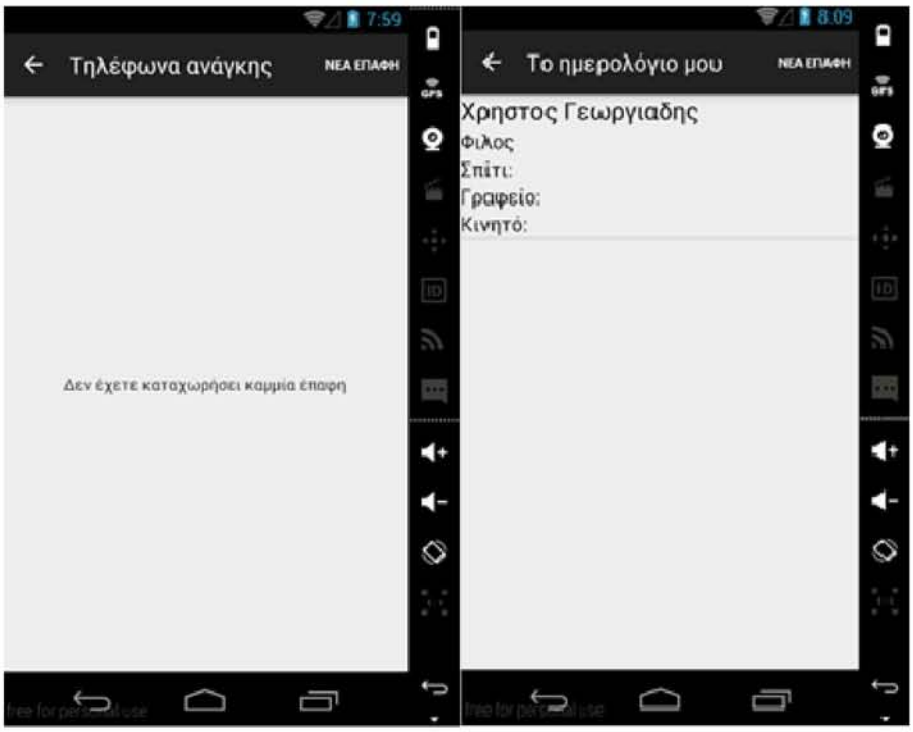
Εάν ο χρήστης επιθυμεί να καταχωρήσει νέα τιμή σακχάρου, πατώντας στο αντίστοιχο κουμπί, αναδύεται το παράθυρο διαλόγου που φαίνεται στην εικόνα 5.6.2. Ο χρήστης χρησιμοποιώντας δυνατότητα κύλισης μπορεί να επιλέξει το μήνα, την ημέρα και το έτος αντίστοιχα της νέας καταχώρησης. Ακολουθεί ένα κελί στο οποίο με την ανάδυση ενός αριθμητικού πληκτρολογίου, ο χρήστης μπορεί να δηλώσει την μετρούμενη τιμή. Η επιλογή του κουμπιού «Αποθήκευση», αποθηκεύει τα δεδομένα αυτά και η αντίστοιχη επιλογή του κουμπιού «Άκυρο», τερματίζει την ενέργεια της καταχώρησης νέας τιμής σακχάρου ως εσφαλμένη.



Εικόνα 5.6.2:Παράθυρο διαλόγου νέας καταχώρησης

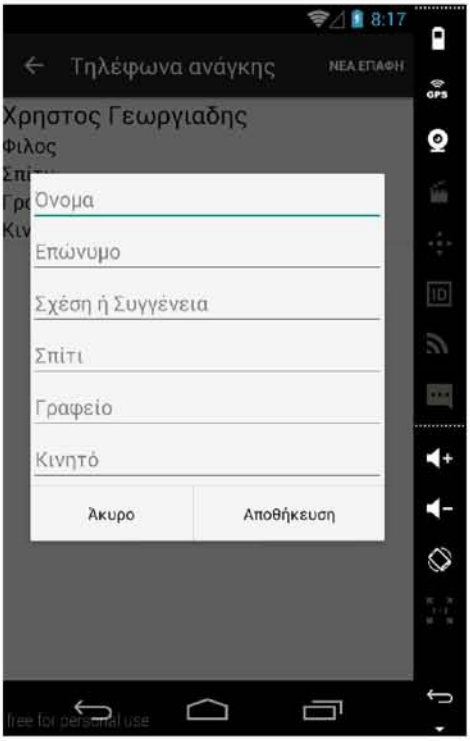
5.7 Τηλέφωνα ανάγκης

Στην εικόνα 5.7.1 παρουσιάζεται η επιλογή της καρτέλας καταγραφής τηλεφώνων έκτακτης ανάγκης οικείων προσώπων του χρήστη. Εάν ο χρήστης χρησιμοποιεί την εφαρμογή για πρώτη φορά, τότε στο κυρίως πλαίσιο της οθόνης εμφανίζεται το μήνυμα προς το χρήστη «Δεν έχετε καταχωρήσει καμία επαφή». Σε αντίθετη περίπτωση, στο πλαίσιο εμφανίζεται κατάλογος επαφών του χρήστη, όπως φαίνεται στην εικόνα 5.7.1(δεξιά).



Εικόνα 5.7.1:Προβολή καρτέλας «Τηλέφωνα ανάγκης» χωρίς προηγούμενες καταχωρήσεις (αριστερά) , προβολή καταλόγου (δεξιά)

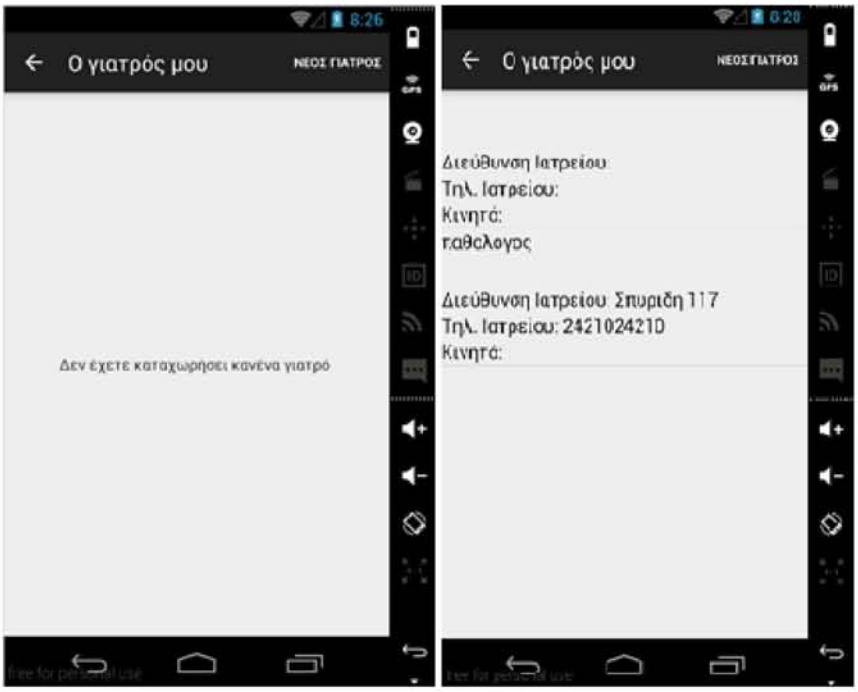
Εάν ο χρήστης επιθυμεί να καταχωρήσει νέα επαφή οικείου προσώπου, πατώντας στο αντίστοιχο κουμπί, αναδύεται το παράθυρο διαλόγου που φαίνεται στην εικόνα 5.7.2. Πατώντας πάνω σε ένα οποιοδήποτε κελί της φόρμας αναδύεται πληκτρολόγιο και ο χρήστης εισάγει δεδομένα στην εφαρμογή. Η επιλογή του κουμπιού «Αποθήκευση», αποθηκεύει τα δεδομένα αυτά και η αντίστοιχη επιλογή του κουμπιού «Άκυρο», τερματίζει την ενέργεια της καταχώρησης νέας επαφής ως εσφαλμένη.



Εικόνα 5.7.2:Παράθυρο διαλόγου νέας επαφής

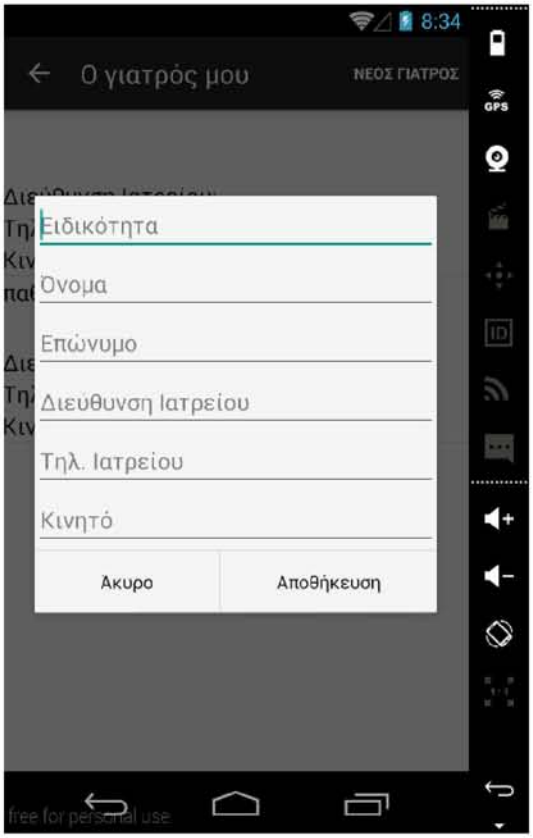
5.8 Ο γιατρός μου

Στην εικόνα 5.8.1 παρουσιάζεται η επιλογή της καρτέλας καταχώρησης των στοιχείων των προσωπικών ιατρών του χρήστη. Εάν ο χρήστης χρησιμοποιεί την εφαρμογή για πρώτη φορά, τότε στο κυρίως πλαίσιο της οθόνης εμφανίζεται το μήνυμα προς το χρήστη «Δεν έχετε καταχωρήσει κανένα γιατρό». Σε αντίθετη περίπτωση, στο πλαίσιο εμφανίζεται κατάλογος με τους προσωπικούς ιατρούς του χρήστη, όπως φαίνεται στην εικόνα 5.8.1(δεξιά).



Εικόνα 5.8.1:Προβολή καρτέλας «Ο γιατρός μου» χωρίς προηγούμενες καταχωρήσεις (αριστερά) , προβολή καταλόγου (δεξιά)

Εάν ο χρήστης επιθυμεί να καταχωρήσει νέα εγγραφή για προσωπικό ιατρό του, πατώντας στο αντίστοιχο κουμπί, αναδύεται το παράθυρο διαλόγου που φαίνεται στην εικόνα 5.8.2. Πατώντας πάνω σε ένα οποιοδήποτε κελί της φόρμας αναδύεται πληκτρολόγιο και ο χρήστης εισάγει δεδομένα στην εφαρμογή. Η επιλογή του κουμπιού «Αποθήκευση», αποθηκεύει τα δεδομένα αυτά και η αντίστοιχη επιλογή του κουμπιού «Άκυρο», τερματίζει την ενέργεια της καταχώρησης νέου ιατρού ως εσφαλμένη.



Εικόνα 5.8.2:Παράθυρο διαλόγου νέας καταχώρησης ιατρού

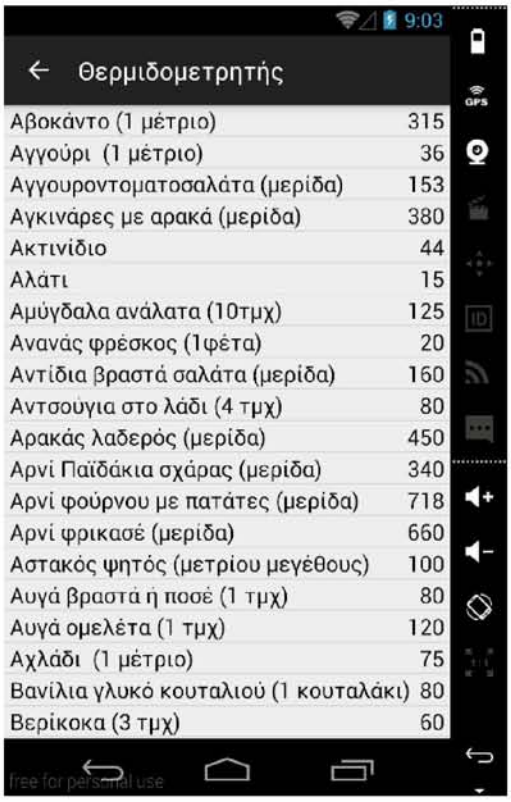
5.9 Φυσιολογικές τιμές εξετάσεων

Στην εικόνα 5.9.1 παρουσιάζεται η επιλογή της καρτέλας «Φυσιολογικές τιμές». Η καρτέλα αυτή προβάλλει έναν πίνακα, που αντιπροσωπεύει τις φυσιολογικές τιμές των αιματολογικών εξετάσεων, που πρέπει να έχει ένα υγιές άτομο. Στο αριστερό μέρος του πίνακα δίνονται τα στοιχεία και στο δεξί οι αντίστοιχες αριθμητικές τιμές τους, για άντρες και γυναίκες. Ο χρήστης έχει τη δυνατότητα κύλισης ώστε να έχει πρόσβαση σε όλο το εύρος του πίνακα.

Εικόνα 5.9.1: Φυσιολογικές τιμές εξετάσεων

5.10 Θερμιδομετρητής

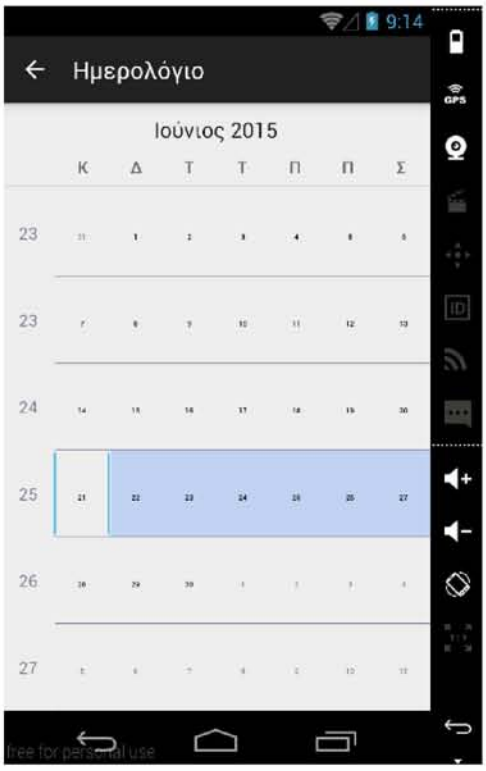
Στην εικόνα 5.10.1 παρουσιάζεται η επιλογή της καρτέλας «Θερμιδομετρητής». Η καρτέλα αυτή προβάλλει έναν πίνακα, που καταγράφει τροφές και την ενδεικτική θερμιδική αξία τους. Στο αριστερό μέρος του πίνακα δίνονται οι τροφές και διευκρινίσεις για τις ποσότητες τους, και στο δεξί οι αντίστοιχες αριθμητικές τιμές των θερμίδων. Ο πίνακας είναι οργανωμένος με αλφαβητική σειρά. Ο χρήστης έχει τη δυνατότητα κύλισης ώστε να έχει πρόσβαση σε όλο το εύρος του πίνακα.



Εικόνα 5.10.1: Θερμιδομετρητής

5.11 Ημερολόγιο

Στην εικόνα 5.11.1 παρουσιάζεται η επιλογή της καρτέλας «Ημερολόγιο». Εμφανίζεται ημερολόγιο με ενδείξεις για μήνα, έτος, ημέρα της εβδομάδας και ημερολογιακή εβδομάδα. Ο χρήστης μπορεί να περιηγηθεί σε οποιαδήποτε ημερομηνία επιθυμεί μέσω της κύλισης με το δάκτυλο. Η εκάστοτε ημέρα που διανύεται, υπογραμμίζεται με γραφικά.



Εικόνα 5.11.1: Ημερολόγιο

6 Εργαλεία ανάπτυξης εφαρμογής και απαιτήσεις συστήματος

6.1 Ανάπτυξη εφαρμογής

Η ανάπτυξη του κώδικα της εφαρμογής πραγματοποιήθηκε στο λογισμικό Android Studio, έκδοση 1.2.1. Το Android Studio είναι το επίσημο IDE(integrated development environment) για την ανάπτυξη εφαρμογών Android και στηρίζεται στο λογισμικό IntelliJ IDEA. Με τον όρο IDE καλείται ένα προγραμματιστικό περιβάλλον το οποίο έχει ομαδοποιηθεί σε ένα πακέτο σαν ένα ολοκληρωμένο πρόγραμμα-εφαρμογή. Τυπικά αποτελείται από έναν κειμενογράφο κώδικα(code editor), έναν μεταφραστή(compiler), ένα πρόγραμμα εντοπισμού σφαλμάτων(debugger) και έναν διαμορφωτή γραφικού περιβάλλοντος χρήστη (GUI builder). Το Android Studio είναι διαθέσιμο για χρήση σε περιβάλλον Windows, Mac OS καθώς και Linux και αντικατέστησε το Eclipse Android Development Tool(ADT), ως το πρωταρχικό για την Google λογισμικό IDE. Το Android Studio προσφέρει :

- Ευέλικτο σύστημα μετάφρασης και υποδομής. Με το Gradle είναι δυνατή η δημιουργία πολλαπλών αρχείων APKs για την ίδια εφαρμογή με διαφορετικά χαρακτηριστικά το καθένα χρησιμοποιώντας την ίδια εργασία(Project)
- Δυνατότητα χρήσης προτύπων έτοιμου κώδικα, για παράδειγμα χρήση στον σχεδιασμό της πλοήγησης (Navigation Drawer)
- Υποστήριξη για προσομοίωση συσκευών πολλαπλών διαφορετικών χαρακτηριστικών (π.χ ανάλυση οθόνης, μέγεθος οθόνης) μέσω του εργαλείου Virtual Device Manager
- Εργαλεία για βελτιστοποίηση της απόδοσης, της χρηστικότητας και της φορητότητας των εφαρμογών

Το Android Studio παρέχει τη δυνατότητα να γίνεται δοκιμή της εφαρμογής σε πραγματικό χρόνο κατά τη διάρκεια της ανάπτυξης της. Το εργαλείο αυτό ονομάζεται εξομοιωτής (emulator) και είναι ένας εύκολος τρόπος να προσομοιωθεί η λειτουργία της εφαρμογής σε μία πραγματική συσκευή. Ο εξομοιωτής που χρησιμοποιήθηκε

στην ανάπτυξη της εφαρμογής είναι το Genymotion ,με την βοήθεια της προσθήκης του Android Studio, Genymotion Device Manager. Στον πίνακα που ακολουθεί παρουσιάζονται τα χαρακτηριστικά της συσκευής που προσομοιώθηκε.

Όνομα συσκευής	Google Galaxy Nexus
Έκδοση λογισμικού (Android OS version)	4.1.1 (Jelly Bean) API 16
Μέγεθος/Ανάλυση οθόνης	4.65 ΄΄ /720x1280 pixels
Επεξεργαστής CPU	1 simple core
Μνήμη RAM	1024MB

Πίνακας 6.1.1: Χαρακτηριστικά εξομοιωτή

Για να γίνει εφικτή η χρήση του προγράμματος εξομοίωσης χρησιμοποιήθηκε το λογισμικό δημιουργίας εικονικών συσκευών Oracle VM Virtualbox 4.3.28.

6.2 Απαιτήσεις συστήματος

Οι ελάχιστες απαιτήσεις συστήματος για την εγκατάσταση της εφαρμογής από τους χρήστες, είναι να διαθέτουν συσκευή με έκδοση Android 4.0.3(Ice Cream Sandwich) με API 15 ή μεταγενέστερη αυτής.

6.3 Οδηγίες εγκατάστασης

Η εγκατάσταση της εφαρμογής μπορεί να γίνει με δύο τρόπους

- φορτώνοντας το αρχείο DiaryApp.apk της εφαρμογής στην συσκευή. Εντοπίζοντας το με έναν διαχειριστή αρχείων, κάνοντας διπλό κλικ ξεκινάει η εγκατάσταση της εφαρμογής.
- χρησιμοποιώντας το Android Studio, το οποίο παρέχει την δυνατότητα εκτέλεσης της εφαρμογής σε μία πραγματική συσκευή, πατώντας το κουμπί Run

στην πλατφόρμα. Για να είναι αυτό εφικτό πρέπει να έχουν ενεργοποιηθεί στην συσκευή οι επιλογές προγραμματιστή (Developer options)

6.4 Οδηγίες χρήσης εφαρμογής

Η εκτέλεση της εφαρμογής είναι πολύ εύκολη. Αφού, εγκατασταθεί η εφαρμογή, ο χρήστης μπορεί να βρει μία συντόμευση της εφαρμογής στο κυρίως μενού της συσκευής του. Αγγίζοντας το εικονίδιο της συντόμευσης εκκινείται αυτόματα η εφαρμογή.

6.5 Δοκιμές

Με την περάτωση της εφαρμογής πραγματοποιήθηκαν κάποιες δοκιμές σε πραγματικές συσκευές, για τον εντοπισμό πιθανών σφαλμάτων και δυσλειτουργιών. Οι δοκιμές που έλαβαν χώρα είχαν τα επιθυμητά αποτελέσματα. Εντοπίστηκε μία παράλειψη στην προβολή του πίνακα των φυσιολογικών τιμών και του θερμιδομετρητή σε διαφορετικά μεγέθη οθονών και σε αλλαγή του προσανατολισμού οθόνης των συσκευών, οπότε και επιδιορθώθηκε με την απαραίτητη αλλαγή στη στοίχιση των επιμέρους δεδομένων και τιμών. Στον πίνακα 6.5.1 δίνονται τα χαρακτηριστικά ορισμένων συσκευών στις οποίες έγιναν δοκιμές.

Συσκευή	LG Optimus L9 II D605	Samsung I9300 Galaxy S III	Samsung I9190 Galaxy S4 mini
Έκδοση λογισμικού (Android OS version)	Kit Kat v 4.4.2	Jelly Bean v 4.3	Jelly Bean v 4.3
Μέγεθος/Ανάλυση οθόνης	4.7 ΄΄ /720x1280 pixels	4.8΄΄ /720x1280 pixels	4.3΄΄ /540 x 960 pixels
Επεξεργαστής CPU	Qualcomm Snapdragon 400, Dual-core 1.4GHz Krait	Exynos 4412 Quad, Quad-core 1.4GHz Cortex-A9	Qualcomm MSM8930AB Snapdragon 400, Dual-core 1.7 GHz Krait 300
Μνήμη RAM	1024MB	1024MB	1536MB

Πίνακας 6.5.1: Χαρακτηριστικά συσκευών δοκιμών

7 Επίλογος

7.1 Συμπεράσματα

Με το τέλος της διπλωματικής εργασίας, ο αρχικός στόχος που είχε τεθεί, για τη δημιουργία μίας εφαρμογής που να καλύπτει τις ανάγκες για την οργάνωση της καθημερινής ζωής του χρήστη και την αποθήκευση ιστορικού μετρήσεων πίεσης και σακχάρου, έχει επιτευχθεί.

Η εφαρμογή είναι ιδιαιτέρως φιλική προς το χρήστη με ένα πολύ απλό και λειτουργικό γραφικό περιβάλλον, που την καθιστά επιπρόσθετα πολύ γρήγορη στην απόκριση και την απόδοση.

Ως πρώτη προσπάθεια για υλοποίηση εφαρμογής, μελετήθηκαν σε βάθος οι βασικές αρχές του λειτουργικού συστήματος Android και επιτεύχθηκε εξοικείωση με τα λογισμικά πακέτα υποστήριξης της ανάπτυξης εφαρμογών για έξυπνα κινητά τηλέφωνα.

7.2 Μελλοντικές επεκτάσεις και αναβαθμίσεις

Η ραγδαία εξέλιξη της τεχνολογίας και η εξάπλωση των έξυπνων τηλεφώνων, έχει οδηγήσει στην ανάπτυξη πολυάριθμων εφαρμογών με στόχο την βελτίωση της εμπειρίας του χρήστη μέσω των υπηρεσιών που του προσφέρουν. Οι ανάγκες των χρηστών διαρκώς αλλάζουν και επεκτείνονται, επομένως επιτακτική καθίσταται και η ανάγκη για συνεχή βελτίωση των εφαρμογών. Στην κατεύθυνση αυτή, θα μπορούσαν να υπάρξουν κάποιες περαιτέρω βελτιώσεις της εφαρμογής.

- Στην λειτουργία της καταχώρησης τηλεφώνων ανάγκης και προσωπικών ιατρών, θα μπορούσε να υπάρξει επέκταση της, ώστε να δημιουργηθεί μία επιλογή που να συνδέει την εφαρμογή με ιστοσελίδες (κάνοντας parsing τα δεδομένα των ιστότοπων). Έτσι, θα μπορούσε να δοθεί η δυνατότητα να παρέχονται πληροφορίες για εφημερεύοντα νοσοκομεία, φαρμακεία και συμβεβλημένους γιατρούς της περιοχής, αν συνδυαζόταν η χρήση της εφαρμογής με τον εντοπισμό της γεωγραφικής θέσης του χρήστη μέσω GPS της συσκευής.
- Στην καταγραφή και αποθήκευση ιστορικού μετρήσεων, μία περαιτέρω επέκταση θα μπορούσε να αποτελέσει η ομαδοποίηση των μετρήσεων και η αναπαράσταση

τους με γραφήματα ή διαγράμματα. Στα διαγράμματα να αποτυπώνονται στατιστικά στοιχεία (μέσος όρος μηνιαίων ή εβδομαδιαίων μετρήσεων). Αυτή η δυνατότητα θα μπορούσε να ανιχνεύει ανωμαλίες στην διακύμανση και να ειδοποιεί για επικίνδυνα επίπεδα για την υγεία του χρήστη.

- Τέλος, γενικά η εφαρμογή θα μπορούσε να συνεργαστεί σαν επιμέρους οντότητα με άλλες ήδη υπάρχουσες εφαρμογές, οι οποίες με τη βοήθεια εργαλείων που είναι διαθέσιμα στο Android(Google maps, αισθητήρες κίνησης κ.λ.π), να υπολογίζουν και να καταγράφουν πόσες θερμίδες έκαψε ο χρήστης με την απόσταση που διάνυσε και με την γυμναστική που έκανε σε πραγματικό χρόνο (τρέξιμο, περπάτημα, ποδήλατο)

8 Βιβλιογραφία

1. http://dazeinfo.com/2014/01/23/smartphone-users-growth-mobile-internet-2014-2017/
2. https://en.wikipedia.org/wiki/Android_%28operating_system%29
3. https://en.wikipedia.org/wiki/Android_version_history
4. https://el.wikipedia.org/wiki/Android
5. https://developer.android.com/about/dashboards/index.html?utm_source=suzunone
6. http://www.visual-paradigm.com
7. http://www.geekwire.com/2015/apple-takes-smartphone-market-share-from-android-thanks-to-strong-iphone-sales/
8. http://developer.android.com/reference/android/widget/ListAdapter.html
9. http://developer.android.com/guide/topics/ui/dialogs.html
10. http://stackoverflow.com/questions/20638967/how-to-change-fragments-using-android-navigation-drawer
11. http://stackoverflow.com/questions/18372450/how-to-set-arrayadapter-to-listview-android
12. https://en.wikipedia.org/wiki/JSON
13. http://jsonformatter.curiousconcept.com/
14. http://www.androidhive.info/
15. https://developer.android.com/sdk/index.html
16. http://developer.android.com/tools/device.html
17. https://www.genymotion.com/#!/