ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη εφαρμογής σε κινητό για έξυπνο σπίτι

Διπλωματική Εργασία

Κατσαρός Κωνσταντίνος

Επιβλέπων: Τσαλαπάτα Χαρίκλεια

Βόλος 2021

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

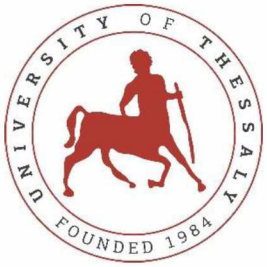
Ανάπτυξη εφαρμογής σε κινητό για έξυπνο σπίτι

Διπλωματική Εργασία

Κατσαρός Κωνσταντίνος

Επιβλέπων: Τσαλαπάτα Χαρίκλεια

Βόλος 2021



UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Smart home mobile application development

Diploma Thesis

Katsaros Konstantinos

Supervisor: Tsalapata Hariklia

Volos 2021

Εγκρίνεται από την Επιτροπή Εξέτασης:

Επιβλέπων/πουσα	**Τσαλαπάτα Χαρίκλεια** Ε.ΔΙ.Π. , Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Δασκαλοπούλου Ασπασία** Επίκουρος Καθηγήτρια, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Θάνος Γεώργιος** Ε.ΔΙ.Π , Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Ημερομηνία έγκρισης: 12-7-2021

ΕΥΧΑΡΙΣΤΙΕΣ ή ΣΧΟΛΙΑ

Θα ήθελα να ευχαριστήσω την επιβλέπουσα καθηγήτρια της διπλωματικής μου εργασίας κα. Τσαλαπάτα Χαρίκλεια για την ευκαιρία που μου έδωσε να ασχοληθώ με αυτό το θέμα διπλωματικής και για την άριστη συνεργασία και καθοδήγηση, καθ' όλη τη διάρκεια εκπόνησης της εργασίας.

Ευχαριστώ θερμά την οικογένεια μου για τη στήριξη όλα αυτά τα χρόνια των σπουδών μου, την υπομονή και την πολύτιμη βοήθεια τους. Τους ευχαριστώ που είναι δίπλα μου και με στηρίζουν με κάθε δυνατό τρόπο για να πετύχω τους στόχους και τα όνειρά μου.

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο Δηλών

Κατσαρός Κωνσταντίνος

12 Ιουλίου 2021

ΠΕΡΙΛΗΨΗ

Ο γρήγορος ρυθμός της εξέλιξης της τεχνολογίας και οι απαιτήσεις τις καθημερινότητας καθιστούν αναγκαία τη χρήση μιας συσκευής κινητού τηλεφώνου (smartphone). Μέσω των πολλών διαφορετικών λειτουργιών και των εφαρμογών που διαθέτει, ο χρήστης μπορεί να επικοινωνήσει μέσω διαδικτύου, να διαχειριστεί την ηλεκτρονική του αλληλογραφία, να κάνει πληρωμές και να ελέγχει τις λειτουργίες του έξυπνου σπιτιού του.

Ένα έξυπνο σπίτι έχει ως βασικό στόχο να προσφέρει άνεση, αποδοτικότητα, καλύτερη διαχείριση της ηλεκτρικής ενέργειας και ασφάλεια. Με το πέρασμα του καιρού η ιδέα ενός έξυπνου σπιτιού γίνεται όλο και πιο ελκυστική, καθώς πολλές εταιρίες επενδύουν σε πολλές και διαφορετικές λύσεις με στόχο την εξυπηρέτηση των καταναλωτών.

Στην παρούσα διπλωματική εργασία παρουσιάζεται η υλοποίηση συστήματος ενός έξυπνου σπιτιού με τη χρήση του μικροελεγκτή Arduino. Στόχος είναι η δημιουργία εφαρμογής σε περιβάλλον Android, η οποία προσομοιώνει τον έλεγχο ενός έξυπνου σπιτιού και ενσωματώνει πληροφορίες, που λαμβάνονται από αισθητήριες μονάδες συνδεδεμένες στον μικροελεγκτή Arduino.

ABSTRACT

The fast pace of technological evolution and the demands of everyday life, make the use of a mobile device (smartphone) a necessity. Through various functions and applications of a mobile device, a user is able to communicate via the internet, manage his electronic mail, make online payments and control the functions of their smart home.

The main goal of a smart home is to provide comfort, efficiency, better management of electrical power and security. As time passes, the idea of a smart home is becoming even more attractive, because a lot of companies invest in various solutions, with the main goal being the service of the consumers.

The present thesis demonstrates the implementation of a smart home's environment, with the use of the "Arduino" microcontroller. The main objective is the creation of an application in an Android environment, that simulates the control of a smart home and embodies information, which is received from sensors connected to the "Arduino" microcontroller.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 Έξυπνο σπίτι

Η ταχεία τεχνολογική πρόοδος και η αυξανόμενη ψηφιοποίηση έχει κάνει την ιδέα ενός ψηφιακού «έξυπνου» σπιτιού να είναι πιο επίκαιρη και δελεαστική από ποτέ. Η λογική είναι πως δημιουργείται ένα οικοσύστημα με έξυπνες συσκευές, οι οποίες είναι συνδεδεμένες και επικοινωνούν μεταξύ τους μέσα από το διαδίκτυο ή ένα οικιακό τοπικό δίκτυο. Επιπλέον, από αισθητήριες μονάδες λαμβάνουν ερεθίσματα από το περιβάλλον δημιουργώντας έτσι πολλές αυτοματοποιημένες διαδικασίες. Το έξυπνο σπίτι παρέχει στους ιδιοκτήτες άνεση καθώς και εξοικονόμηση κόστους, εξίσου σημαντική όμως είναι και η ασφάλεια του σπιτιού. Τα συστήματα smart home βοηθούν σημαντικά στην προστασία των κατοίκων και τις περιουσίας τους με όλο και καλύτερα και πιο σύνθετα συστήματα ασφαλείας.

Ο όρος «έξυπνο» σπίτι αναφέρεται σε μια κατοικία εξοπλισμένη με τεχνολογία που διευκολύνει, προάγει την ανεξαρτησία και αυξάνει την ποιότητα ζωής των κατοίκων. Η τεχνολογία ενσωματώνεται στην υποδομή της κατοικίας και δεν απαιτεί κάποια ιδιαίτερη εκπαίδευση από τον κάτοικο, μόνο κάποια εξοικείωση με ένα smartphone ή με έναν ηλεκτρονικό υπολογιστή.

Τα Smart Homes ή έξυπνα σπίτια, είναι μια σχετικά νέα τάση που με το πέρασμα του καιρού συνεχώς εξελίσσεται και με σταθερό ρυθμό όλο και περισσότερο ο κόσμος το επιλέγει. Αρχικά, η έξυπνη οικιακή τεχνολογία χρησιμοποιήθηκε για τον έλεγχο κυρίως περιβαλλοντικών συστημάτων, όπως ο φωτισμός και η θέρμανση της κατοικίας, αλλά η χρήση της έξυπνης τεχνολογίας έχει αναπτυχθεί έτσι ώστε σχεδόν οποιαδήποτε ηλεκτρική συσκευή εντός του σπιτιού να μπορεί να συμπεριληφθεί σε ένα έξυπνο οικοσύστημα. Ένα έξυπνο σπίτι μπορεί τώρα να παρακολουθεί τις δραστηριότητες του κατοίκου ενός σπιτιού και να διαχειρίζεται ανεξάρτητα συσκευές ανάλογα με τις απαιτήσεις του χρήστη [1].

Με την ανάπτυξη του internet και των ταχυτήτων του, όλο και περισσότερες και πιο περίπλοκες λειτουργίες εισάγονται σε ένα οικοσύστημα έξυπνου σπιτιού. Για παράδειγμα στον ιατρικό τομέα, ένα έξυπνο σπίτι παρέχει ιδανικά ένα περιβάλλον το οποίο

παρακολουθεί και βοηθά ηλικιωμένους ή άτομα με ειδικές ανάγκες ή παθήσεις. Ασχολείται κυρίως με τη βελτίωση της ποιότητας της υγείας και της υγειονομικής περίθαλψης [2].

1.2 Τεχνολογίες που χρησιμοποιούνται σε ένα έξυπνο σπίτι

1.2.1 Αισθητήρες

Χρησιμοποιείται ένα δίκτυο αισθητήρων για τη λήψη περιβαλλοντικών παραμέτρων και ερεθισμάτων εντός και εκτός του έξυπνου σπιτιού, όπως το φως, η θερμοκρασία, η ποιότητα αέρα, τις καιρικές συνθήκες, ατμοσφαιρική πίεση και άλλα δεδομένα. Αυτές οι πληροφορίες μπορούν στη συνέχεια να χρησιμοποιηθούν για τη λήψη έξυπνων και αυτοματοποιημένων αποφάσεων. Για παράδειγμα, ένας αισθητήρας φωτός μπορεί να ανιχνεύσει χαμηλά επίπεδα φωτός σε ένα δωμάτιο και να ανάψει τα φώτα, ή ένας αισθητήρας υγρασίας μπορεί να ενεργοποιήσει μια συσκευή αφυγραντήρα . Ακόμα, μπορεί να ανιχνευτεί αυτόματα για το αν κάποιος είναι στο σπίτι χρησιμοποιώντας μια ποικιλία αισθητήριων μέσων όπως υπέρυθρων (IR), αισθητήρες πίεσης. Επιπλέον, οι αισθητήρες χρησιμοποιούνται συχνά σε συνδυασμό με ενεργοποιητές, οι οποίοι παρέχουν ένα πλήρες σύστημα αντιδράσεων στα λαμβανόμενα ερεθίσματα των αισθητήρων.

Οι περιβαλλοντικοί αισθητήρες συχνά κρύβονται στο περιβάλλον του σπιτιού, που βρίσκονται σε εσωτερικούς και εξωτερικούς χώρους, καρέκλες, τοίχους, δάπεδα και σε διάφορα άλλα μέρη όπου συλλέγουν πληροφορίες μέσω κάποιου ενσύρματου(π.χ. Ethernet) ή ασύρματου πρωτοκόλλου(Wi-Fi) [3].

1.2.2 Ελεγκτές/Μικροελεγκτές

Τα δεδομένα που συλλέγονται από ένα σύστημα αισθητήρων συγκεντρώνονται σε μία κεντρική επεξεργαστική μονάδα, κάποιον μικροελεγκτή, στην παρούσα εργασία το Arduino. Εκεί θα ληφθούν οι αποφάσεις για το κάθε ερέθισμα και θα αποκριθούν σε πολύ μικρό χρονικό διάστημα. Πρόκειται δηλαδή για τον εγκέφαλο ο οποίος ελέγχει τις συσκευές και τους αισθητήρες και είναι υπεύθυνος για τη διαχείριση τους με βάση τον προγραμματισμό τους.

1.2.3 Λογισμικό

Το λογισμικό είναι το εργαλείο το οποίο θα χρησιμοποιήσει ο κάτοικος ενός έξυπνου σπιτιού για να διαχειριστεί τα πάντα μέσα στο σπίτι του. Το λογισμικό είναι εγκατεστημένο είτε σε ένα smartphone είτε ένα laptop. Ο χρήστης μπορεί να παρακολουθήσει το χώρο του ακόμα και απομακρυσμένα και να λαμβάνει οποιαδήποτε πληροφορία ή ειδοποίηση (παράδειγμα συναγερμός σπιτιού). Είναι σημαντικό η διεπαφή του χρήστη (UI) να έχει καλή σχεδίαση. Θα πρέπει η εφαρμογή να είναι εύχρηστη και εύκολη στη χρήση, ιδανικά και σε χρήστες με λιγότερη εξοικείωση με τα έξυπνα κινητά τηλέφωνα.

1.3 Δομή Διπλωματικής εργασίας

Σκοπός της παρούσας διπλωματικής εργασίας είναι η δημιουργία ενός συστήματος έξυπνου σπιτιού μέσα από αισθητήρες και έλεγχο μέσω android application. Πρόκειται κυρίως για ένα περιβάλλον επίβλεψης και επιτήρησης ενός σπιτιού, μέσω των ειδοποιήσεων στο κινητό. Επιπλέον δημιουργείται ένα πλήρες περιβάλλον μιας εφαρμογής ενός έξυπνου σπιτιού με διακόπτες και διάφορες επιλογές που λειτουργούν ως προσομοίωση. Η υλοποίηση βασίζεται στη χρήση της open-source πλακέτας Arduino Uno για την χρήση των αισθητήρων και στο android studio για τη δημιουργία της android εφαρμογής.

Στο Κεφάλαιο 2 παρουσιάζονται αναλυτικά κάποια android applications δημοφιλών τεχνολογικών εταιριών που είναι διαθέσιμα, τα οποία αποτελούν ολοκληρωμένη λύση για ένα smart home.

Στο Κεφάλαιο 3 γίνεται μια εισαγωγή για το hardware που χρησιμοποιήθηκε (Arduino Uno) καθώς και το λογισμικό (Arduino IDE) για τον προγραμματισμό του.

Το Κεφάλαιο 4 αφορά τον προγραμματισμό και την συρμάτωση του Arduino και των αισθητήρων.

Στο Κεφάλαιο 5 γίνεται μια εισαγωγή για το Android Studio, ένα πλήρες προγραμματιστικό περιβάλλον για τη δημιουργία Android εφαρμογών. Γίνεται μια μικρή επισκόπηση της γλώσσας προγραμματισμού Java και της γλώσσας σήμανσης (markup language) XML.

Στο Κεφάλαιο 6 παρουσιάζεται κάποιες σημαντικές έννοιες και λειτουργίες που χρησιμοποιήθηκαν κατά την υλοποίηση της εφαρμογής

Στο κεφάλαιο 7 παρουσιάζεται η εφαρμογή smart home που δημιουργήθηκε στα πλαίσια της διπλωματικής εργασίας, το MyHome. Γίνεται περιγραφή του περιβάλλοντος και των λειτουργιών που προσφέρει.

Στο Κεφάλαιο 8 αφορά το ανέβασμα του project στο GitHub.

Στο Κεφάλαιο 9 γίνεται ο επίλογος της διπλωματικής εργασίας καθώς και κάποιες επεκτάσεις που θα μπορούσαν να υλοποιηθούν στο μέλλον.

ΚΕΦΑΛΑΙΟ 2

ΠΑΡΟΜΟΙΕΣ ΕΦΑΡΜΟΓΕΣ

Στο κεφάλαιο 2 παρουσιάζεται πως λειτουργούν κάποιες android εφαρμογές smart home που υπάρχουν διαθέσιμες στην αγορά, τί δυνατότητες προσφέρουν, τους αυτοματισμούς όπως επίσης και το σχεδιασμό τους.

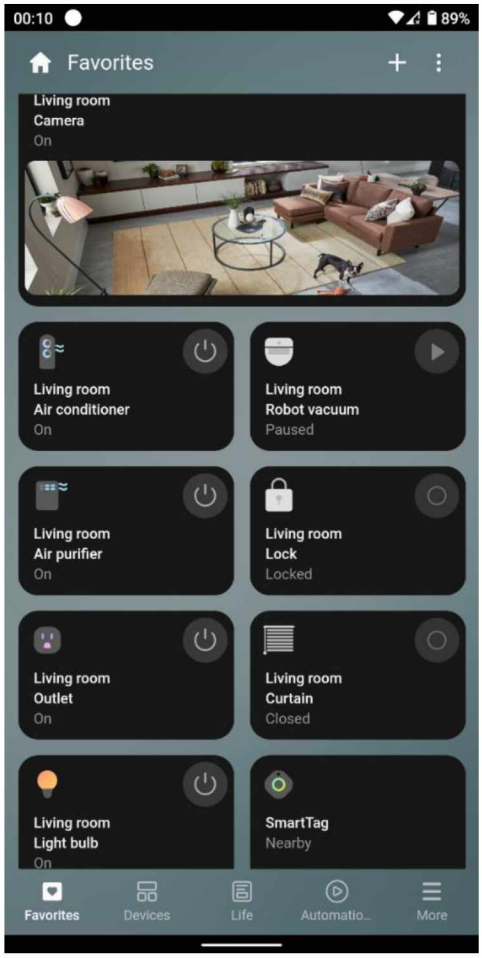
2.1 SmartThings

Ένα ολοκληρωμένο σύστημα έξυπνου σπιτιού είναι το SmartThings το οποίο προσφέρεται από την Samsung. Πρόκειται για λογισμικό μέσω του οποίου μπορούν όλες οι έξυπνες συσκευές της Samsung και άλλων επιλεγμένων εταιριών, να επικοινωνούν μεταξύ τους. Υπάρχει πληθώρα συσκευών οι οποίες είναι συμβατές όπως φωτιστικά, κάμερες, κλειδαριές, θερμοστάτες και άλλα, ενώ μέσω της εφαρμογής προσφέρονται ποικίλες λειτουργίες και αυτοματισμοί.

Χρησιμοποιώντας το SmartThings application προτείνεται η δημιουργία λογαριασμού Samsung έτσι ώστε να μπορεί ο χρήστης να δημιουργήσει το προσωπικό του Smart Home, στα μέτρα του και με βάση τις συσκευές που έχει εξοπλίσει το σπίτι του. Επίσης υπάρχει η λειτουργία για προβολή εικονικού smart home, στο οποίο δημιουργείται ένα εικονικό σπίτι με διάφορες συσκευές και αυτό θα χρησιμοποιηθεί για να μελετηθεί η εφαρμογή.

2.1.1 Favorites - Αγαπημένα

Στο έτοιμο εικονικό smart home, στην καρτέλα favorites (Εικόνα 2.1) υπάρχουν οι «αγαπημένες» λειτουργίες του χρήστη στις οποίες μπορεί ανά πάσα στιγμή να προσθέσει ή να αφαιρέσει κι άλλες. Πρόκειται για μια οθόνη στην οποία θα ο χρήστης θα προσθέσει συσκευές, αυτοματισμούς και άλλα για γρηγορότερη πρόσβαση. Στην οθόνη αυτή μπορεί να προστεθεί οποιαδήποτε συσκευή όπως το κλιματιστικό, το κλείδωμα/ξεκλείδωμα την κεντρικής πόρτας, μια ρομποτική σκούπα έτσι ώστε να γίνεται πιο άμεσα ο χειρισμός της εκάστοτε συσκευής ή λειτουργίας. Επιπλέον ο χρήστης μπορεί να προσθέσει τους αγαπημένους του αυτοματισμούς και σκηνές και να τους έχει γρήγορα διαθέσιμους σε μια καρτέλα.

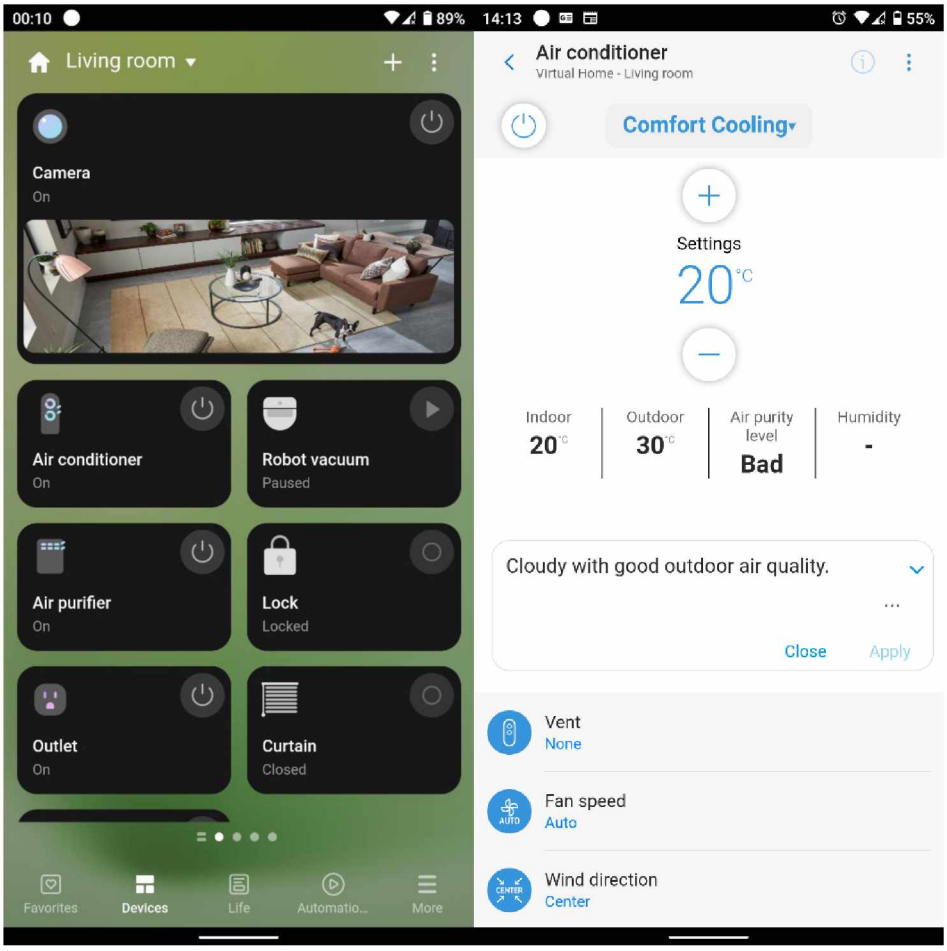


Εικόνα 2.1: Καρτέλα Favorites στην εφαρμογή SmartThings.

2.1.2 Devices

Συνεχίζοντας στην καρτέλα devices, είναι ένα μενού διαχείρισης των δωματίων και των συσκευών του. Ο χρήστης μπορεί να διαχειριστεί τις συνδεδεμένες συσκευές που είναι διαθέσιμες ανά δωμάτιο. Επίσης μέσα από αυτή την οθόνη, ο χρήστης μπορεί να δημιουργήσει τα διαφορετικά δωμάτια του σπιτιού του (κουζίνα, μπάνιο, υπνοδωμάτιο) και στη συνέχεια να προσθέσει οποιαδήποτε συμβατή συσκευή ή αισθητήρας στο επιθυμητό δωμάτιο. Για τη διαχείριση οποιασδήποτε συσκευής με λεπτομέρειες ο χρήστης την επιλέγει και στη συνέχεια θα αναδυθεί ένα μενού με τις δυνατότητες που υπάρχουν. Για παράδειγμα, στην καρτέλα devices στο σαλόνι (Living Room) υπάρχει συσκευή κλιματισμού την οποία επιλέγει ο χρήστης να διαχειριστεί. Αμέσως ανοίγει το μενού των ιδιοτήτων της συσκευής και υπάρχει δίνεται η δυνατότητα στο χρήστη να ενεργοποιήσει/απενεργοποιήσει τη συσκευή καθώς και όλες οι λειτουργίες που προσφέρει η συσκευή κλιματισμού (Εικόνα 2.2). Όπως φαίνεται στην εικόνα 2.2, ο χρήστης

μπορεί να αυξήσει ή να μειώσει τη θερμοκρασία, να επιλέξει το πρόγραμμα του air condition (cool, dry, purify,heat και άλλα), να ελέγξει την διεύθυνση του αέρα και πολλές ακόμα λειτουργίες. Επίσης μπορεί να λειτουργήσει αυτόματα εφόσον επιλεγεί και καθορίζει ανάλογα με τις συνθήκες εντός και εκτός του σπιτιού, το κατάλληλο πρόγραμμα λειτουργίας.

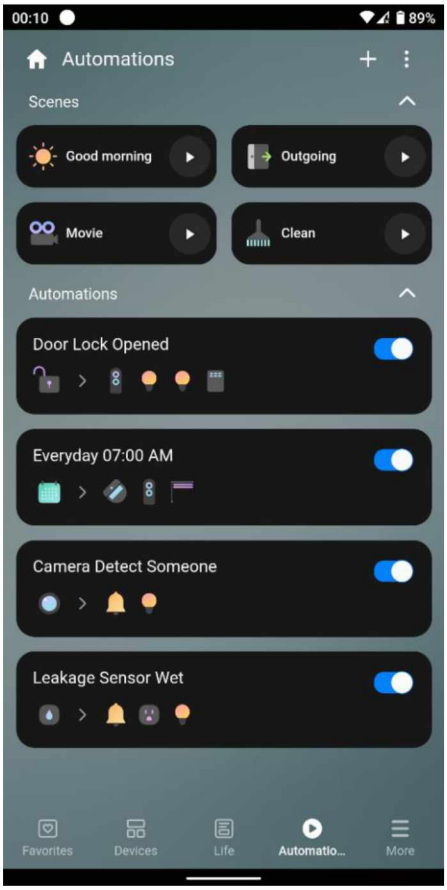


Εικόνα 2.2: (Αριστερά)Το σαλόνι με τις διαθέσιμες συσκευές. (Δεξιά) Επιλογή του Air conditioner και προβολή επιλογών και ρυθμίσεων.

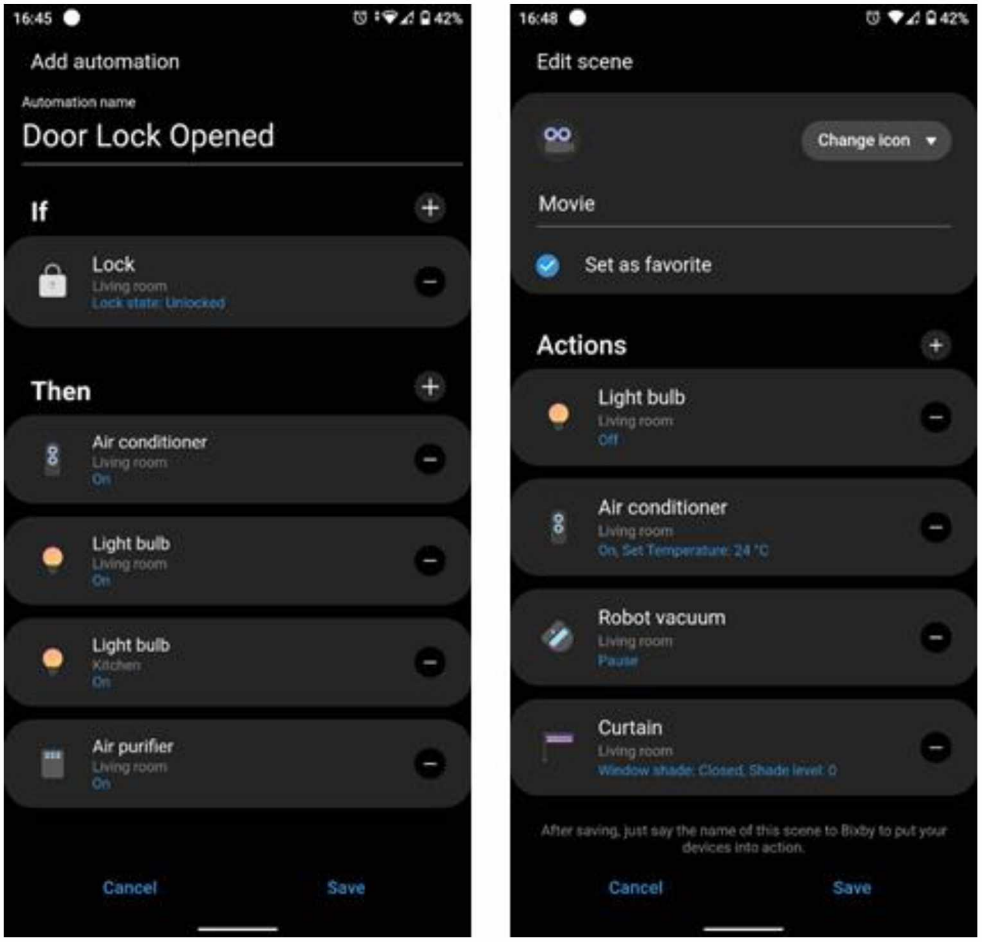
2.1.3 Automations

Στην καρτέλα automations δίνεται η δυνατότητα στο χρήστη να δημιουργήσει και να χρησιμοποιήσει αυτοματισμούς ή σκηνές που έχουν ήδη δημιουργηθεί (Εικόνα 2.3). Οι αυτοματισμοί είναι ουσιαστικά αυτοματοποιημένες διαδικασίες που ενεργοποιούνται συνήθως μετά από σήμα κάποιου αισθητήρα. Οι αυτοματισμοί δημιουργούνται σε ένα μενού και έχουν τη λογική if-then (Εικόνα 2.4), δηλαδή αν συμβεί ένα γεγονός τότε

γίνονται αντίστοιχα οι ενέργειες που θα επιλέξει ο χρήστης. Για παράδειγμα όταν ο χρήστης επιστρέφει σπίτι του και ξεκλειδώσει την πόρτα εισόδου, αυτόματα ανοίγει το κλιματιστικό και κάποια φώτα στο σαλόνι και στην κουζίνα. Η μπορεί να ορίσει συγκεκριμένη ώρα και μέρα της εβδομάδας που η ρομποτική σκούπα θα ενεργοποιείται. Σκηνές (scenes) είναι διαδικασίες που δε θα γίνουν αυτοματοποιημένα άλλα θα ενεργοποιηθούν μόνο εφόσον ο χρήστης το επιλέξει. Για να δημιουργηθεί μια σκηνή, εφόσον δοθεί το όνομα που επιθυμεί ο χρήστης, επιλέγεται πώς θα αντιδρούν οι διάφορες συσκευές στο σπίτι (Εικόνα 2.4). Για παράδειγμα, όταν ενεργοποιηθεί η σκηνή με όνομα “Outgoing” θα απενεργοποιηθούν συσκευές όπως κλιματιστικό και ο αφυγραντήρας ενώ θα ενεργοποιηθούν οι κάμερες ασφαλείας καθώς θα κλείσουν και τα ρολά των παραθύρων. Αν αντίστοιχα επιλεγεί η σκηνή “Good morning” θα ενεργοποιηθεί το κλιματιστικό, θα ανοίξουν τα ρολά του σπιτιού και θα απενεργοποιηθεί το σύστημα ασφαλείας (κάμερες, συναγερμός),ενώ η σκηνή “Movie” θα δημιουργήσει στο σαλόνι μια ιδανικότερη ατμόσφαιρα για την προβολή ταινίας χαμηλώνοντας τον φωτισμό, κατεβάζοντας τα ρολά των παραθύρων του σαλονιού.



Εικόνα 2.3: Απεικόνιση των σκηνών(πάνω) και των αυτοματισμών(κάτω).

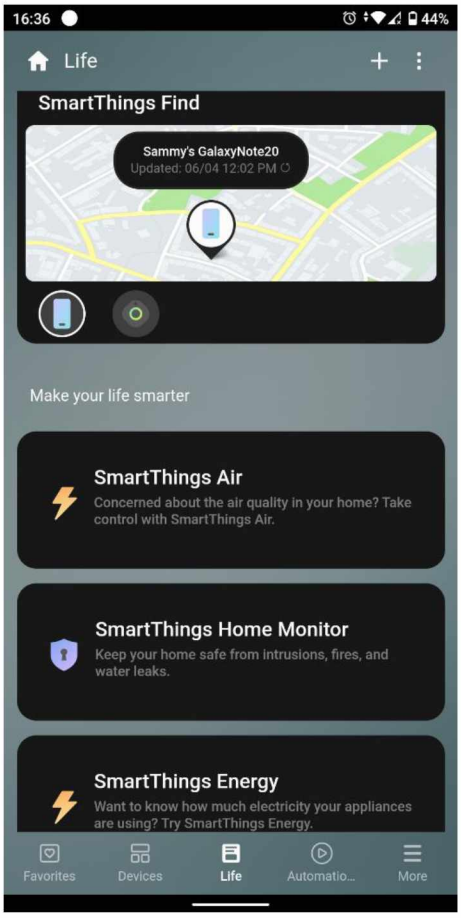


Εικόνα 2.4: (Αριστερά) Απεικόνιση της δημιουργίας ενός αυτοματισμού με τη λογική if-then. Στο συγκεκριμένο screenshot φαίνεται τι συμβαίνει (what) όταν (if) η κεντρική πόρτα του σπιτιού ξεκλειδώσει.(Δεξιά) Απεικόνιση της δημιουργίας μιας σκηνής (scene) με όνομα Movie το οποίο όταν ενεργοποιηθεί πραγματοποιούνται τα actions που φαίνονται στην φωτογραφία.

2.1.4 Life

Μέσα από την καρτέλα Life (Εικόνα 2.5), μπορούν να δημιουργηθούν υπηρεσίες που θα διευκολύνουν τη ζωή του χρήστη. Αρχικά είναι διαθέσιμη η υπηρεσία SmartThings Find που θα βοηθήσει στον εντοπισμό συσκευών με ακριβής τοποθεσία στο χάρτη και πολλές ακόμα λεπτομέρειες. Η υπηρεσία SmartThings Air, δίνει την δυνατότητα στο χρήστη να παρακολουθεί την ποιότητα του αέρα στο σπίτι του και σε κάθε δωμάτιο ξεχωριστά και δίνει συμβουλές για τη βελτίωση αν χρειάζεται. Στη συνέχεια, το SmartThings Home Monitor είναι ένα monitor του σπιτιού, στο οποίο ορίζονται ποιοι αισθητήρες λειτουργούν

όταν ο χρήστης βρίσκεται μέσα στο σπίτι και ποιοι όταν αυτός απουσιάζει. Αισθητήρες κίνησης, πυρκαγιάς ή κάμερες ασφαλείας ενεργοποιούνται όταν εντοπισθεί ότι ο χρήστης δεν βρίσκεται εντός κατοικίας και ενημερώνεται με αντίστοιχες ειδοποιήσεις στο κινητό του τηλέφωνο. Επίσης, διαθέσιμη είναι η υπηρεσία SmartThings Energy που μπορεί να δώσει πληροφορίες στο χρήστη σχετικά με την κατανάλωση των συσκευών του που είναι συνδεδεμένες στο έξυπνο σπίτι.



Εικόνα 2.5: Απεικόνιση των διαθέσιμων υπηρεσιών του SmartThings στην καρτέλα Life.

Τέλος, αξίζει να σημειωθεί ότι το application δίνει τη δυνατότητα πλήρη ελέγχου του σπιτιού με φωνητικές εντολές χρησιμοποιώντας κάποιον voice assistant είτε τον «Google Assistant» είτε την «Alexa» της Amazon. Επίσης, το SmartThings είναι συμβατό και μπορεί να χρησιμοποιηθεί ακόμα και από την οθόνη ενός αυτοκινήτου. Μοναδική προϋπόθεση το αυτοκίνητο να διαθέτει το Android Auto μια εφαρμογή που συνδέει το κινητό με την κεντρική οθόνη/infotainment του αυτοκινήτου. Ο χρήστης μπορεί να λάβει ειδοποιήσεις

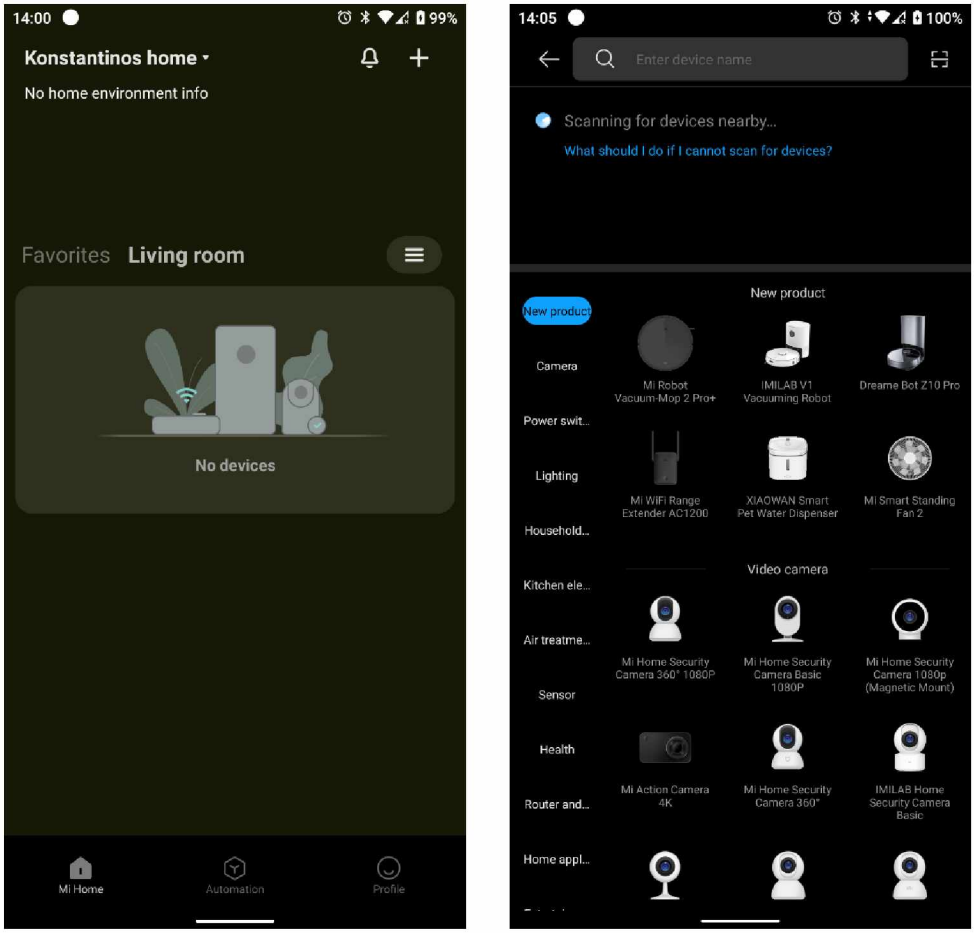
από το έξυπνο σπίτι ή ακόμα να ενεργοποιήσει την κεντρική θέρμανση πριν φτάσει σπίτι του.

2.2 Mi Home

Ένα ακόμα ολοκληρωμένο σύστημα έξυπνου σπιτιού είναι το Mi Home της Xiaomi. Πρόκειται για ένα σύστημα που υποστηρίζει συσκευές, αισθητήρες και πολλά ακόμα. Υποστηρίζονται κυρίως συσκευές τις ίδιας της Xiaomi αλλά υπάρχει συμβατότητα και με συσκευές από διάφορες ακόμα εταιρίες που ασχολούνται με συσκευές για έξυπνα σπίτια. Ομοίως όπως στο SmartThings απαιτείται από τον χρήστη να δημιουργήσει ένα λογαριασμό ώστε να κάνει πιο προσωποποιημένη την εμπειρία του με την εφαρμογή. Κάνοντας Log in στο λογαριασμό του ο χρήστης θα κληθεί να δώσει όνομα στο έξυπνο σπίτι του.

2.2.1 Mi Home

Στην καρτέλα Mi Home ο χρήστης μπορεί να διαχειριστεί το περιβάλλον του σπιτιού του. Μπορεί να δημιουργήσει και να ονομάσει τα δωμάτια του σπιτιού του και στη συνέχεια μπορεί σε κάθε δωμάτιο ξεχωριστά να προσθέτει τις συσκευές που διαθέτει και είναι συμβατές. Για παράδειγμα, δημιουργείται το σαλόνι (Εικόνα 2.6) και στη συνέχεια με το κουμπί πάνω δεξιά στην οθόνη γίνεται περιήγηση σε όλες τις διαθέσιμες συσκευές. Για την αναζήτηση, υπάρχει η δυνατότητα αναζήτησης πληκτρολογώντας το όνομα συσκευής ή ανά είδος όπως κάμερα, αισθητήρες, φωτισμός και άλλα (Εικόνα 2.6). Στην συνέχεια, αφού ο χρήστης αναζητήσει και βρει τη συσκευή την επιλέγει και ακολουθεί τις οδηγίες που θα εμφανιστούν στην οθόνη του κινητού του προκειμένου να την ενσωματώσει στο σύστημα του έξυπνου σπιτιού. Στη συνέχεια όλες οι συσκευές είναι διαθέσιμες στο μενού του αντίστοιχου δωματίου. Η λειτουργία Favorites/Αγαπημένα υπάρχει με τη μορφή δωματίου και όχι καρτέλας(tab), στην οποία ο χρήστης προσθέτει οποιαδήποτε συσκευή ανεξάρτητα σε ποιο δωμάτιο βρίσκεται.

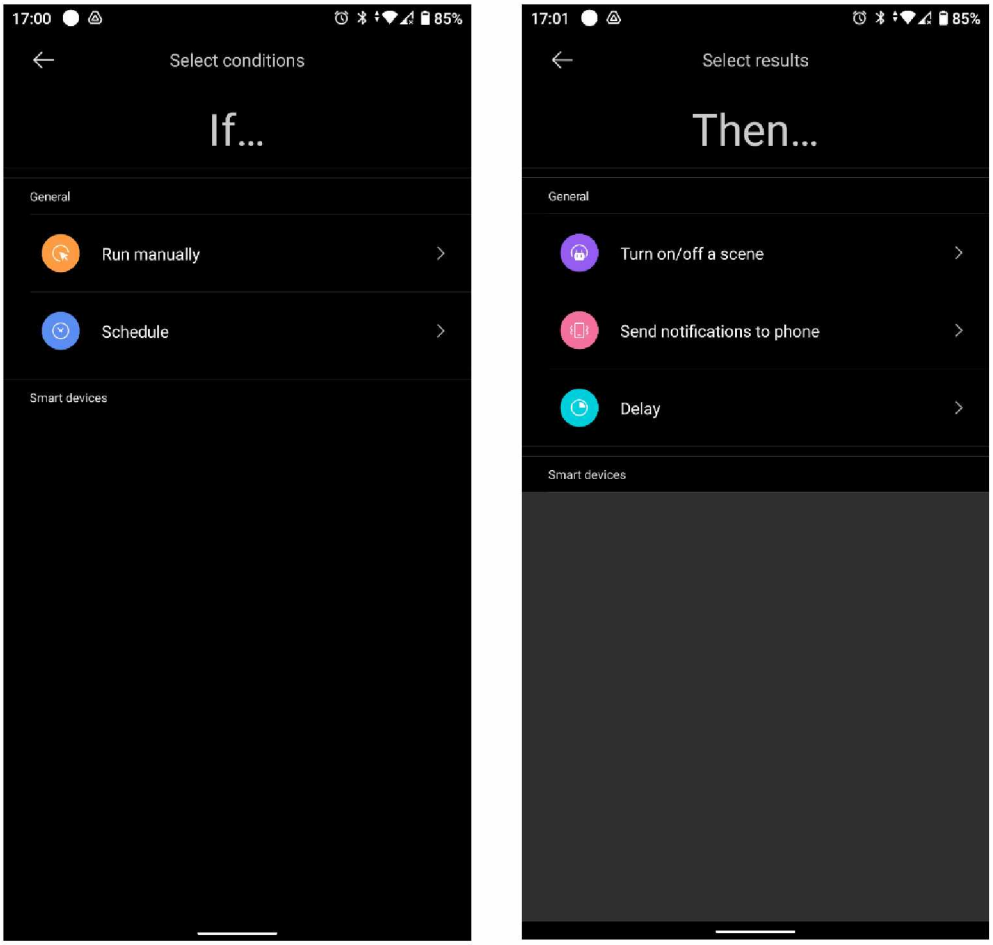


Εικόνα 2.6: (Αριστερά) Το σαλόνι με τις συσκευές. Στην συγκεκριμένη φωτογραφία δεν υπάρχουν συσκευές.(Δεξιά) Προσθήκη συσκευών στο σαλόνι με ονομαστική αναζήτηση της συσκευής.

2.2.2 Mi Home Automation

Στην καρτέλα Automation ο χρήστης θα δημιουργήσει τους αυτοματισμούς που θέλει για να τον διευκολύνει στην καθημερινότητά του. Η λογική είναι παρόμοια με αυτήν του application SmartThings που αναλύθηκε παραπάνω. Ο χρήστης δημιουργεί αυτοματισμούς με την λογική if – then, αν δηλαδή συμβεί κάποιο προεπιλεγμένο γεγονός τότε θα γίνουν αντίστοιχα κάποιες ενέργειες (Εικόνα 2.7). Για παράδειγμα αν στο οικοσύστημα του έξυπνου σπιτιού υπάρχει μια κάμερα ή ένας αισθητήρας κίνησης και ένας συναγερμός, ένας αυτοματισμός που θα μπορούσε να δημιουργήσει είναι αν ανιχνεύσει κίνηση να σταλεί ειδοποίηση στο κινητό του χρήστη και να ενεργοποιηθεί ο συναγερμός. Οι επιλογές του χρήστη είτε στο IF είτε στο Then αυξάνονται με τον αριθμό

των συσκευών. Τέλος, το Mi Home υποστηρίζει κάποιους voice assistants όπως τον «Google Assistant», την «Alexa» της Amazon και τον NAVER Clova (cloud virtual Assistant) που αναπτύχθηκε από την Naver Corporation. Με κατάλληλη ρύθμιση μπορεί να χρησιμοποιηθεί κάποιος assistant προκειμένου ο χρήστης να ελέγχει όλες τις συσκευές με φωνητικές εντολές.



Εικόνα 2.7: (Αριστερά) Δημιουργία αυτοματισμού με συνθήκη. Υπάρχουν λίγες επιλογές καθότι δεν υπάρχουν συνδεδεμένες συσκευές.(Δεξιά) Στην οθόνη αυτή επιλέγονται ποιες κινήσεις θα γίνουν αν η συνθήκη που όρισε ο χρήστης στην προηγούμενη οθόνη πραγματοποιηθεί.

2.3 Άλλες εφαρμογές

Γενικά, οι εφαρμογές που κυκλοφορούν στην αγορά έχουν παρόμοια λογική. Διευκολύνουν τη ζωή του χρήστη με τους αυτοματισμούς που προσφέρουν και την διαχείριση πολλών ή και όλων των συσκευών του σπιτιού. Η συσκευές αγοράζονται από τον χρήστη και συνδέονται στο οικοσύστημα του έξυπνου σπιτιού είτε χειροκίνητα είτε

αυτόματα μέσω δικτύου (Wi-Fi) και στη συνέχεια ανάλογα με τον εξοπλισμό που διαθέτει το σπίτι μπορούν να γίνουν περισσότεροι και πιο περίπλοκοι αυτοματισμοί.

Δημοφιλείς εφαρμογές για έξυπνα σπίτια είναι και η εφαρμογή Google Home της Google καθώς και η Smart Life της Volcano Technology Limited με εκατομμύρια downloads στο Play Store. Στα θετικά και των δύο εφαρμογών είναι η μεγάλη γκάμα συσκευών που μπορεί ο χρήστης να ενσωματώσει στο σύστημα έξυπνου σπιτιού. Μάλιστα στην περίπτωση του Google Home μπορεί ο χρήστης να συνδέσει και άλλες εφαρμογές έξυπνου σπιτιού (Mi Home, Smart Life και άλλες) κάνοντας Link τα διαφορετικά Accounts (π.χ. Google account και Mi account) και να διαχειρίζεται όλες τις συσκευές χρησιμοποιώντας μόνο την εφαρμογή της Google. Επίσης ο χρήστης επιλέγει την πλατφόρμα της επιλογής του για αναπαραγωγή μουσικής, βίντεο, podcasts μέσα από το YouTube, Spotify και άλλες.

ΚΕΦΑΛΑΙΟ 3

ΕΙΣΑΓΩΓΗ ΓΙΑ ΤΟΝ ΜΙΚΡΟΕΛΕΓΚΤΗ ARDUINO

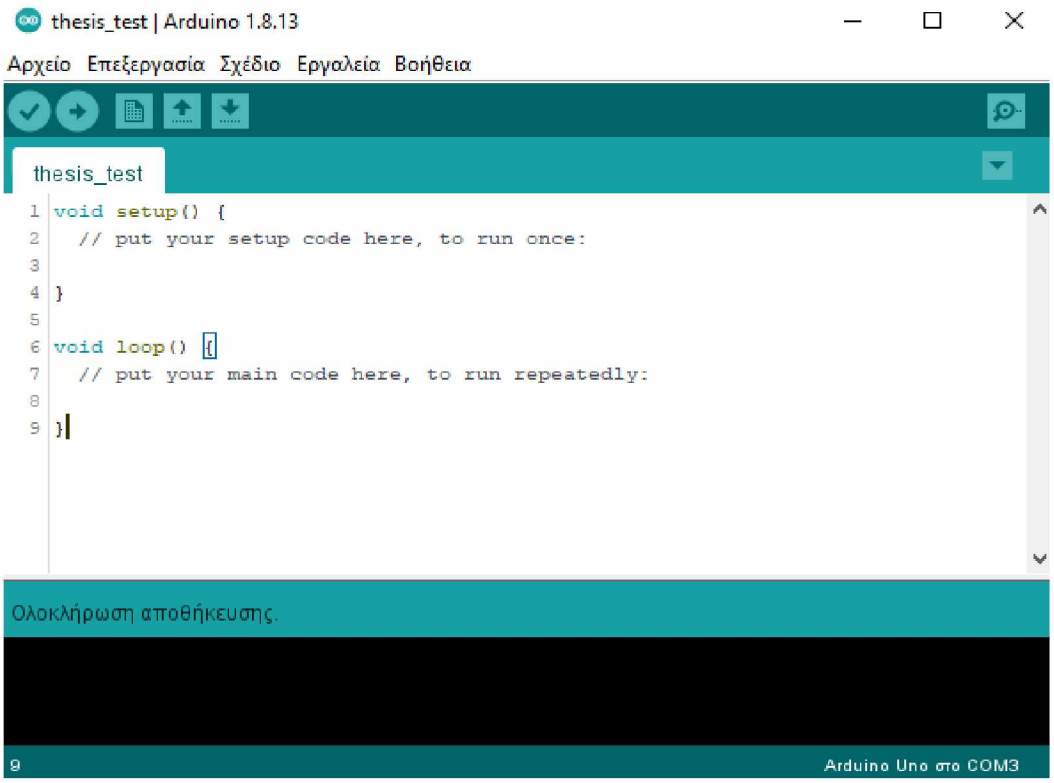
3.1 Μικροελεγκτές Arduino

Arduino είναι δημοφιλείς προγραμματιζόμενες πλακέτες που χρησιμοποιείται για την ανάπτυξη πολλών project. Πρόκειται μητρικές πλακέτες ανοικτού κώδικα με ενσωματωμένο μικροελεγκτή με ψηφιακές και αναλογικές εισόδους-εξόδους. Η γλώσσα προγραμματισμού που μπορεί να χρησιμοποιηθεί για τον προγραμματισμό ενός Arduino είναι ουσιαστικά μια απλοποιημένη μορφή της C++, εμπλουτισμένη με ένα σύνολο έτοιμων βιβλιοθηκών. Διαθέσιμες στην αγορά υπάρχουν πολλές πλακέτες Arduino με διάφορα χαρακτηριστικά και δυνατότητες η κάθε μια, διαφορές στον αριθμό εισόδων εξόδων, επεξεργαστική ισχύ [4].

Πληθώρα συσκευών είναι συμβατές με τις πλακέτες Arduino. Κάποιες από αυτές είναι αισθητήρες θερμοκρασίας, υγρασίας, απόστασης, ατμοσφαιρικής πίεσης και άλλα. Με ένα Arduino μπορεί κανείς να ανάψει ένα led φως, να ελέγξει μικρής ισχύος DC κινητήρες ακόμα και να στήσει ένα μικρό web server.

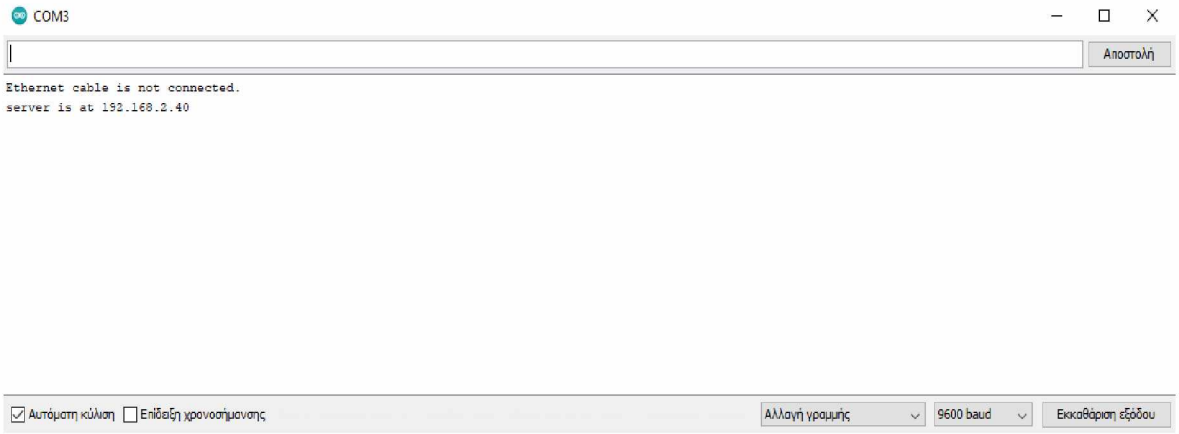
3.2 Το λογισμικό (IDE) του Arduino

Ο προγραμματισμός ενός Arduino γίνεται μέσα από το λογισμικό Arduino Software(IDE), το οποίο είναι διαθέσιμο για Windows, Mac OS X και Linux(Εικόνα 3.1). Πρόκειται για ένα ολοκληρωμένο περιβάλλον το οποίο διαθέτει επεξεργαστή κειμένου-κώδικα για τη συγγραφή και μεταγλωττιστή. Το Arduino συνδέεται στον υπολογιστή με καλώδιο USB type B (στην πλευρά του Arduino) και απόληξη USB type A (για τον υπολογιστή) και μέσω του λογισμικού ο κώδικας γίνεται upload στην πλακέτα.



Εικόνα 3.1: Το περιβάλλον του Arduino Software(IDE).

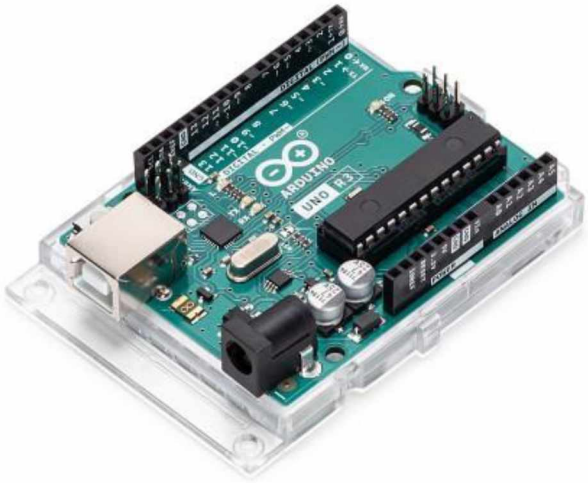
Το λογισμικό συνεχώς βελτιώνεται και αναβαθμίζεται. Εξαιρετικά σημαντική είναι η χρήση των βιβλιοθηκών του Arduino που είναι διαθέσιμες για download είτε μέσα από το λογισμικό είτε χειροκίνητη εγκατάσταση/import μιας custom βιβλιοθήκης. Custom βιβλιοθήκη μπορεί να γράψει οποιοσδήποτε ακολουθώντας τις οδηγίες στο site του Arduino. Επίσης οι βιβλιοθήκες εμπλουτίζονται, αναβαθμίζονται και εισέρχονται νέες. Το Arduino Software δίνει πολλές δυνατότητες και ευκολίες στο χρήστη- προγραμματιστή. Υπάρχουν διαθέσιμα μηνύματα κατά τον προγραμματισμό σε περίπτωση σφάλματος και στο upload – ανέβασμα του προγράμματος στην πλακέτα. Εμφανίζεται μήνυμα πόση μνήμη έχει καταληφθεί από τη συνολική διαθέσιμη της πλακέτας. Επίσης μπορεί να παρακολουθείται η σειριακή πύλη, η οποία χρησιμοποιείται συνήθως για εντοπισμό σφαλμάτων ή για άμεση επικοινωνία με το Arduino.



Εικόνα 3.2: Προβολή σειριακής θύρας. Μήνυμα σφάλματος καθώς το καλώδιο ethernet δεν είναι συνδεδεμένο και μήνυμα ενημέρωσης της διεύθυνσης του web server.

3.3 Arduino Uno

Η βάση αυτής της διπλωματικής εργασίας είναι το Arduino Uno Rev3(Εικόνα 3.2). Πρόκειται για έναν μικρό και οικονομικό μικροελεγκτή ο οποίος διαθέτει 6 αναλογικές εισόδους καθώς και 14 ψηφιακές εισόδους – εξόδους. Από τις 14 ψηφιακές, οι 6 μπορούν να λειτουργήσουν και σαν αναλογικές με το σύστημα Pulse-width modulation (PWM). Η τροφοδοσία του γίνεται μέσω σύνδεσης USB ή με έναν μετασχηματιστή ρεύματος ακόμα και μπαταρία, με τον κατασκευαστή να συνιστά είσοδο 7-12 volt για την προστασία της πλακέτας [5].



Εικόνα 3.2: Απεικόνιση του Arduino Uno Rev3 [5].

Τα τεχνικά χαρακτηριστικά της πλακέτας φαίνονται αναλυτικά στον παρακάτω πίνακα (Πίνακας 3.1):

Πίνακας 3.1: Τεχνικά χαρακτηριστικά του Arduino Uno Rev3 [5].

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Σημαντικό κατά τον προγραμματισμό ενός Arduino είναι να μην ξεπεραστεί το όριο της διαθέσιμης μνήμης Flash Memory και ιδιαίτερα της SRAM. Αν ξεπεραστεί το όριο, ή αν ακόμα η SRAM δεν επαρκεί (χωρίς απαραίτητα να έχει ξεπεραστεί το όριο) τότε το Arduino αποκτά μια ασταθή συμπεριφορά και υπάρχουν ανεπιθύμητα αποτελέσματα.

Συγκεκριμένα Βασισμένο στο chip ATmega328P το Arduino Uno διαθέτει τριών ειδών μνήμης:

i. Flash memory: Πρόκειται ουσιαστικά για την μνήμη στην οποία αποθηκεύεται ο κώδικας που χρησιμοποιείται. Σε περίπτωση διακοπής τροφοδοσίας η μνήμη διατηρεί τα δεδομένα.
ii. SRAM: Static random access memory.Χρησιμοποιείται από το πρόγραμμα που έχει γίνει upload στο Arduino για την αποθήκευση και τη διαχείριση των μεταβλητών ή πινάκων που χρησιμοποιούνται. Ακόμα και αν δεν ξεπεραστεί το όριο αρχικά(για το Arduino Uno 2KB) αν το πρόγραμμα είναι απαιτητικό με πολλές πράξεις (π.χ. Δημιουργία ενός πίνακα)είναι πιθανό να υπάρξει αστάθεια κατά τη λειτουργία του.
iii. EEPROM: Electrically erasable programmable read-only memory. Μνήμη η οποία θυμίζει σκληρό δίσκο υπολογιστή. Χρησιμοποιείται για να αποθηκευτούν πληροφορίες που μπορεί να είναι χρήσιμες μακροπρόθεσμα. Ομοίως όπως η flash memory η EEPROM δεν χάνει τα δεδομένα όταν η πλακέτα απενεργοποιηθεί. [6]

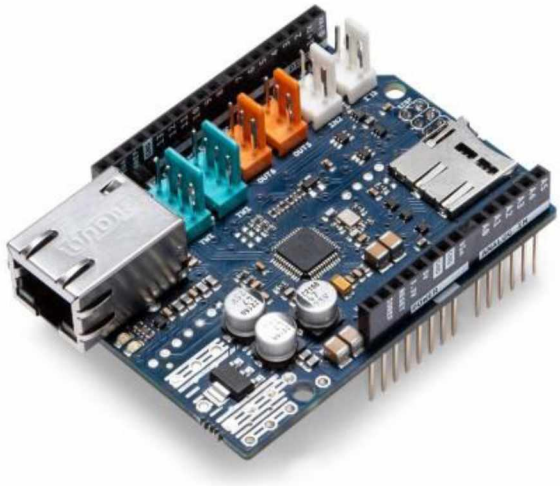
Οι διαφορές μεταξύ των πλακετών Arduino (π.χ. Arduino Mega 2560) πέρα από τις διαφορές στον αριθμό των αναλογικών εισόδων και ψηφιακών εισόδων/εξόδων εντοπίζεται κυρίως στην επεξεργαστική δύναμη του μικροελεγκτή καθώς και στη χωρητικότητα των μνημών. Έτσι, υπάρχει μια μεγάλη γκάμα πλακετών που θα καλύψει ένα μεγάλο φάσμα από projects.

3.4 Arduino Shields

Τα Arduino shields είναι πλακέτες- επεκτάσεις που μπορούν να συνδεθούν πάνω σε μία πλακέτα Arduino, με σκοπό να επεκτείνουν τις λειτουργίες και δυνατότητές του. Τοποθετούνται εύκολα πάνω στην πλακέτα Arduino σαν στοίβα, ενώ υπάρχει η δυνατότητα να συνδυαστούν περισσότερα του ενός shields ανάλογα με τις απαιτήσεις και την πολυπλοκότητα ενός project.

Πολλά Arduino shields είναι διαθέσιμα στην αγορά, μεταξύ αυτών είναι το Arduino Ethernet Shield 2 (Εικόνα 3.3) το οποίο έχει χρησιμοποιηθεί στην παρούσα διπλωματική εργασία. Πρόκειται για μια επέκταση η οποία μεταξύ άλλων, δίνει τη δυνατότητα στο Arduino να συνδεθεί στο διαδίκτυο μέσω καλωδίου ethernet, με ταχύτητες μέχρι και τα

100Mbps. Επιπλέον, διαθέτει υποδοχή για κάρτα μνήμης micro-SD που μπορεί να χρησιμοποιηθεί για την αποθήκευση αρχείων ή την μεταφορά τους μέσω του δικτύου. Μπορεί να χρησιμοποιηθεί για έναν real time data logger, δηλαδή να καταγράφει δεδομένα σε πραγματικό χρόνο. Για να χρησιμοποιηθεί η κάρτα μνήμης απαιτείται η χρήση της βιβλιοθήκης «SD.h» μέσα από το λογισμικό του Arduino.



Εικόνα 3.3: Απεικόνιση του Arduino Ethernet Shield 2 [7].

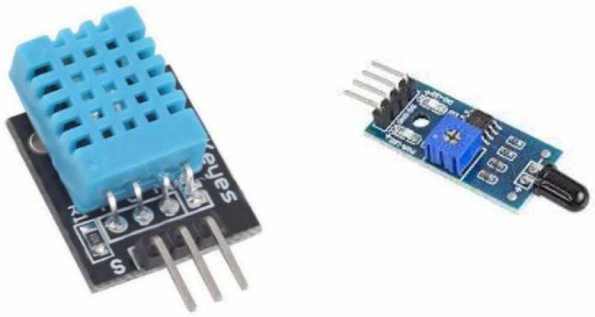
Για να λειτουργήσει το shield, απαραίτητη προϋπόθεση είναι να συμπεριληφθεί στον κώδικα η βιβλιοθήκη του Ethernet η οποία είναι διαθέσιμη για download μέσα από το software του Arduino. Επιπλέον, η παροχή των 5V που χρειάζεται για να λειτουργήσει το shield παρέχεται από την κεντρική πλακέτα που θα συνδεθεί [7].

3.5 Αισθητήρες

Υπάρχει μια μεγάλη γκάμα αισθητήρων τόσο αναλογικοί όσο και ψηφιακοί και μπορούν να καλύψουν ένα μεγάλο φάσμα από λειτουργίες. Ανάλογα με τους στόχους, μπορούν να χρησιμοποιηθούν πολλοί αισθητήρες ταυτόχρονα και η πλακέτα να διαχειριστεί τα ερεθίσματα που δίνουν ως εισόδους σύμφωνα με το πως έχει προγραμματιστεί. Οι ψηφιακοί αισθητήρες επιστρέφουν τιμή HIGH/LOW, 1 ή 0 αντίστοιχα. Οι αναλογικοί

αισθητήρες επιστρέφουν τιμή σε εύρος 0 έως 1023. Αν ένας αναλογικός αισθητήρας επιστρέφει τιμή 1023 σημαίνει ότι δεν ανιχνεύει τίποτα (τιμή ηρεμίας).

Οι αισθητήρες που χρησιμοποιήθηκαν σε αυτή τη διπλωματική εργασία, είναι χαμηλού κόστους παρόλα αυτά αρκετά ακριβείς στις μετρήσεις και πολύ αξιόπιστοι. Αρχικά, χρησιμοποιήθηκε ένας ψηφιακός αισθητήρας θερμοκρασίας και υγρασίας (Εικόνα 3.4) [8]. Στη συνέχεια ένας αισθητήρας φωτιάς με αναλογική και ψηφιακή έξοδο(Εικόνα 3.4) [9], ένας αισθητήρας βροχής με ψηφιακή και αναλογική έξοδο [10] καθώς και ένας αισθητήρα κίνησης(Εικόνα 3.5). Επίσης χρησιμοποιήθηκε ένα led το οποίο ανάβει όταν ο αισθητήρας κίνησης ανιχνεύσει κάποια κίνηση [11]. Όλοι οι αισθητήρες συνδέθηκαν με το Arduino με τη βοήθεια ενός breadboard, μια πλακέτα δοκιμών που επιτρέπει την προσωρινή σύνθεση κυκλωμάτων.



Εικόνα 3.4: (Από αριστερά προς τα δεξιά) DHT11 αισθητήρας θερμοκρασίας και υγρασίας και αισθητήρας φωτιάς.

Εικόνα 3.5: (Από αριστερά προς τα δεξιά) Αισθητήρας βροχής και HC-SR501 PIR Sensor αισθητήρας κίνησης

Σκοπός είναι να έχουμε δεδομένα τόσο για τους χώρους του σπιτιού (θερμοκρασία, υγρασία) όσο για το αν έξω βρέχει ή ψιχαλίζει και ταυτόχρονα να δημιουργηθεί ένα σύστημα ασφαλείας το οποίο θα ενημερώνει σε περίπτωση κινδύνου, είτε φωτιάς είτε παραβίασης του σπιτιού.

3.6 Πλεονεκτήματα Arduino

Στην αγορά είναι διαθέσιμοι πολλοί μικροελεγκτές και αναπτυξιακές πλακέτες, ωστόσο υπάρχουν κάποια πλεονεκτήματα που μπορούν να ξεχωρίσουν ένα Arduino:

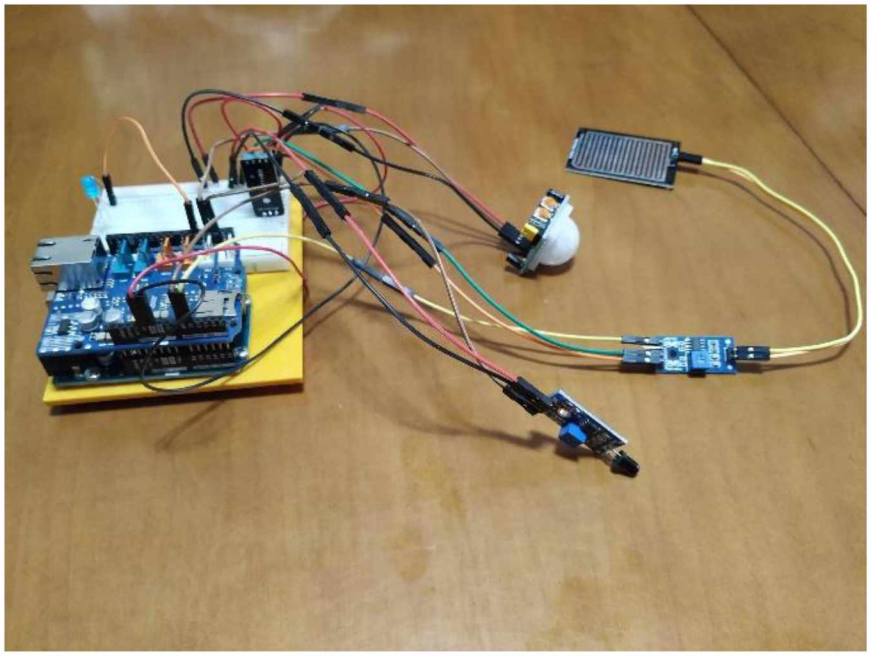
- Κόστος. Οι πλακέτες Arduino είναι σχετικά φθηνές ανάλογα με το μέγεθος του project που θα υλοποιηθεί, όπως επίσης και σε σχέση με άλλες πλατφόρμες μικροελεγκτών
- Το λογισμικό του είναι διαθέσιμο και μπορεί να χρησιμοποιηθεί από πολλές διαφορετικές πλατφόρμες λειτουργικών. Mac, Linux, Windows (Cross - platform).
- Ανοικτού κώδικα και λογισμικού. Ο μικροελεγκτής Arduino είναι επεκτάσιμος μέσω πολλών διαφορετικών βιβλιοθηκών.
- Μεγάλη online κοινότητα. Επαγγελματίες και ερασιτέχνες ανταλλάσσουν πληροφορίες και εμπειρίες online. Το επίσημο φόρουμ είναι πολύ ενεργό καθώς και πολλά άλλα site, όπου μοιράζονται ιδέες προς υλοποίηση, προβλήματα και λύσεις σε πολλά προβλήματα που μπορούν να προκύψουν.

ΚΕΦΑΛΑΙΟ 4

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΚΑΙ ΣΥΡΜΑΤΩΣΗ ARDUINO UNO

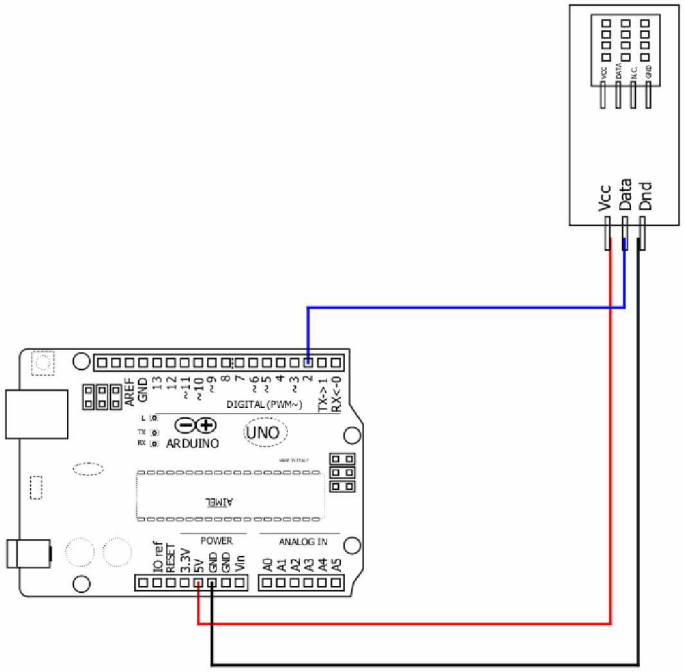
4.1 Συρμάτωση Arduino Uno

Στο Arduino τοποθετείται η Ethernet Shield 2 καθώς και οι αισθητήρες που αναφέρθηκαν παραπάνω. Στη συνέχεια θα δούμε τη σύνδεση των αισθητήρων και τον προγραμματισμό της πλακέτας. Στην (Εικόνα 4.1) φαίνεται το Arduino όπως είναι συνδεδεμένο με όλους τους αισθητήρες στον χώρο του σπιτιού.

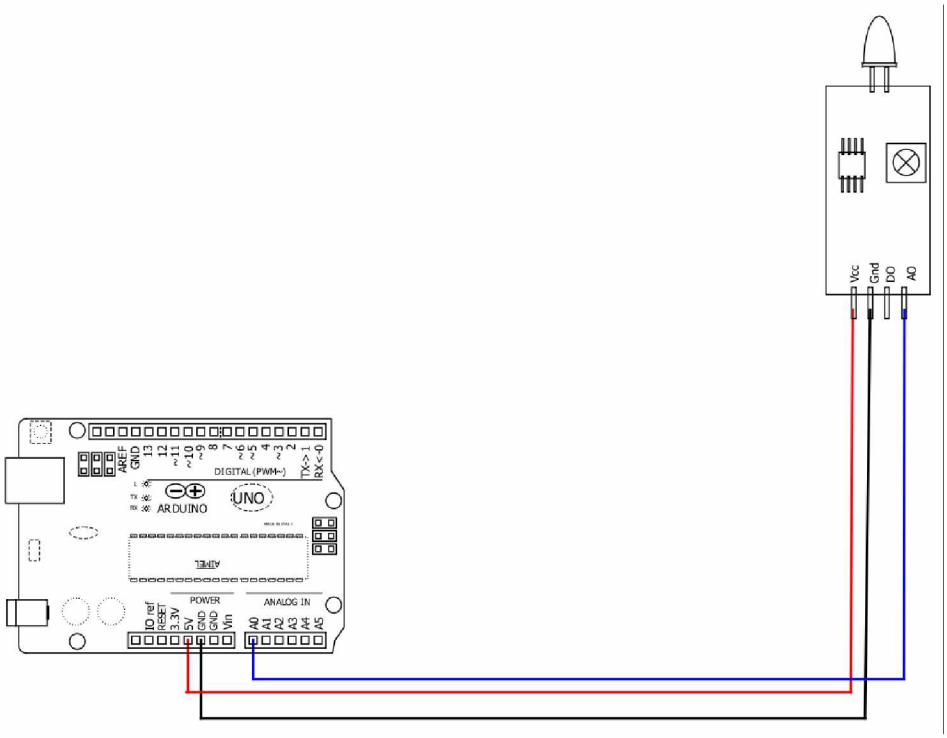


Εικόνα 4.1: Το Arduino συνδεδεμένο με τους αισθητήρες και το Ethernet Shield.

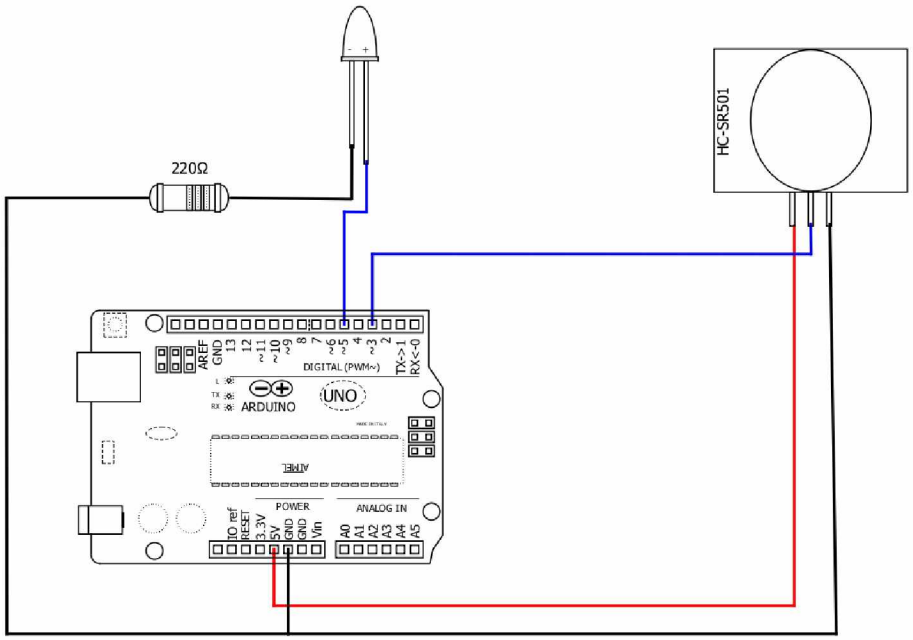
Για συρμάτωση του Arduino (συρμάτωση αισθητήρων) σχεδιάστηκαν τα διαγράμματα μέσω του λογισμικού QElectroTech διαθέσιμο για Windows, Mac OS και Linux. Μέσω του λογισμικού QElectroTech μπορούν να σχεδιαστούν ηλεκτρικά διαγράμματα καθώς και συρματώσεις. Τα σχέδια των αισθητήρων δεν ήταν διαθέσιμα στο λογισμικό και σχεδιάστηκαν από την αρχή. Για τον αισθητήρα θερμοκρασίας και υγρασίας (Εικόνα 4.2) [8], για τον αισθητήρα φωτιάς (Εικόνα 4.3) [9], για τον αισθητήρα ανίχνευσης κίνησης (Εικόνα 4.4) [11] και για τον αισθητήρα βροχής (Εικόνα 4.5) [10] παρουσιάζονται τα διαγράμματα στις αντίστοιχες εικόνες. Ο συνδυασμός των συρματώσεων θα οδηγήσει στο τελικό αποτέλεσμα.



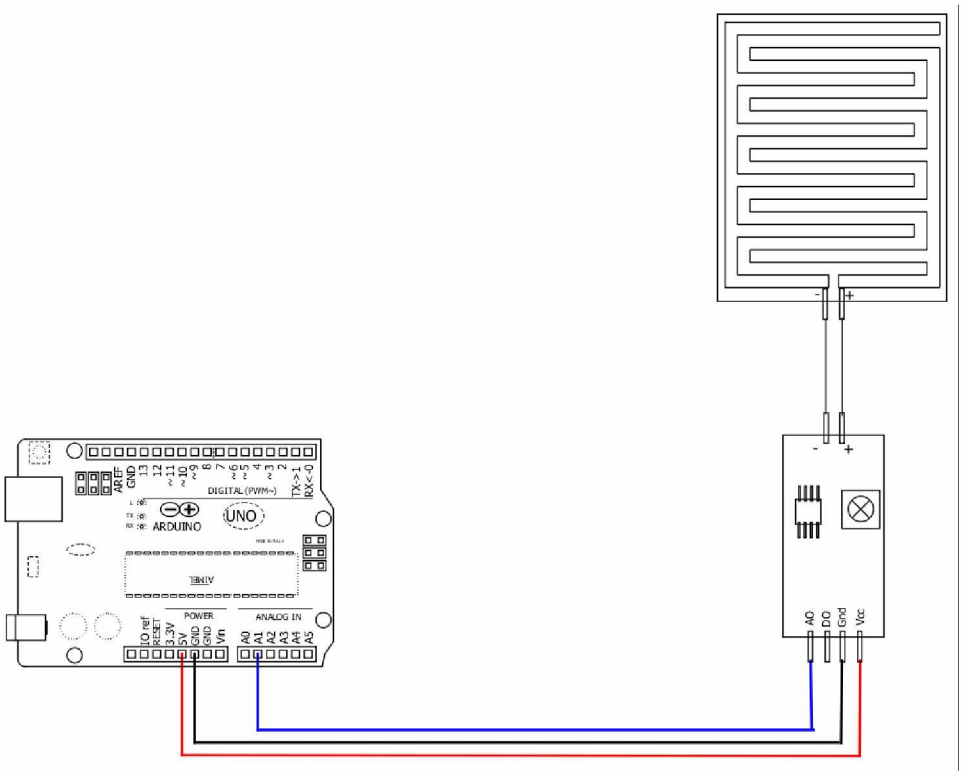
Εικόνα 4.2: Συρμάτωση αισθητήρα θερμοκρασίας και υγρασίας DHT11 με το λογισμικό QElectroTech.



Εικόνα 4.3: Συρμάτωση αισθητήρα φωτιάς (Fire sensor) με το λογισμικό QElectroTech.



Εικόνα 4.4: Συρμάτωση αισθητήρα ανίχνευσης κίνησης (HC-SR501) με το λογισμικό QElectroTech.



Εικόνα 4.5: Συρμάτωση αισθητήρα ανίχνευσης κίνησης (HC-SR501) με το λογισμικό QElectroTech.

4.2 Προγραμματισμός Arduino

Για τον προγραμματισμό του Arduino υπάρχουν δύο βασικές void συναρτήσεις, δηλαδή μετά το τέλος τους δεν επιστρέφουν κάποια τιμή. Αυτές οι συναρτήσεις είναι η setup() και η loop(). Η setup() θα εκτελεστεί μία μόνο φορά κατά το power on της πλακέτας και συνήθως σε αυτή τη συνάρτηση γίνονται τυχόν αρχικοποιήσεις που είναι αναγκαίες για ένα πρόγραμμα. Από την άλλη, η εκτέλεση της loop() γίνεται μετά την εκτέλεση της setup(), κατ' επανάληψη χωρίς διακοπές μέχρι η πλακέτα να τεθεί εκτός λειτουργίας (power off). Στην συνάρτηση loop() θα γραφτεί το κεντρικό κομμάτι του κώδικα και θα στηθεί ο web server.

Για το πρόγραμμα του Arduino χρησιμοποιήθηκαν βιβλιοθήκες που είναι διαθέσιμες μέσα από το Arduino IDE. Η βιβλιοθήκη SPI.h και Ethernet.h είναι απαραίτητες προκειμένου να λειτουργήσει η Ethernet Shield που έχει τοποθετηθεί στην μητρική πλακέτα Arduino Uno. Η βιβλιοθήκη SPI.h (Serial Peripheral Interface) χρησιμοποιείται για την επικοινωνία μιας πλακέτας Arduino με οποιαδήποτε περιφερειακή συσκευή στην περίπτωση αυτή την Ethernet Shield. Η βιβλιοθήκη Ethernet.h δίνει πρόσβαση στο διαδίκτυο. Αντίστοιχα η βιβλιοθήκη DHT.h κάνει εύκολη την διαχείριση οποιουδήποτε αισθητήρα DHT, μέσω των εντολών που είναι διαθέσιμες και θα δούμε στην συνέχεια.

Αρχικά, στο πρόγραμμα αρχικοποιούνται διάφορες τιμές και η τοπική διεύθυνση IP (http://192.168.2.40) την οποία θα λάβει η πλακέτα μέσα στο δίκτυο του σπιτιού. Πρέπει να είναι στατική, να μην μεταβάλλεται για να μη δημιουργηθεί δυσλειτουργία. Στη συνέχεια στη συνάρτηση setup() γίνονται αρχικοποιήσεις, καθορίζονται ποια pins του Arduino θα έχουν το ρόλο των εισόδων ή των εξόδων. Επίσης γίνονται έλεγχοι για τη σωστή λειτουργία της Ethernet Shield, αν είναι συνδεδεμένη στο Arduino, αν υπάρχει καλώδιο ethernet προκειμένου να ξεκινήσει ομαλά η του web server. Τα αντίστοιχα μηνύματα σφάλματος εμφανίζονται στην σειριακή θύρα, καθώς επίσης και μηνύματα επιτυχούς έναρξης του web server.

Στη συνάρτηση loop() δημιουργείται ο webserver, όπου απαντά με ένα http response όταν το ζήτα αντίστοιχα η εφαρμογή android την οποία θα μελετήσουμε σε επόμενο κεφάλαιο. Χρησιμοποιώντας την εντολή client.print() μπορούν να εμφανιστούν στον webserver οι τιμές όλων των αισθητήρων (Σχήμα 4.1). Επιλέγεται να εμφανίζονται ανά μία τιμή ένας

κενός χαρακτήρας, για να διαβαστούν ευκολότερα στη συνέχεια από την android εφαρμογή όπως φαίνεται και στο απόσπασμα κώδικα [12].

```
69    while (client.connected()) {
70      if (client.available()) {
71        char c = client.read();
72        Serial.write(c);
73        // if you've gotten to the end of the line (received a newline
74        // character) and the line is blank, the http request has ended,
75        // so you can send a reply
76        if (c == '\n' && currentLineIsBlank) {
77          // send a standard http response header
78          client.println("HTTP/1.1 200 OK");
79          client.println("Content-Type: text/html");
80          client.println("Connection: close");  // the connection will be closed after completion of the response
81          client.println();
82          client.print(dht.readTemperature());
83          client.print(" ");
84          client.print(dht.readHumidity());
85          client.print(" ");
86          client.print(range1 = map(analogRead(A0), sensorMin, sensorMax, 0, 3));
87          client.print(" ");
88          client.print(range2 = map(analogRead(A1), sensorMin, sensorMax, 0, 3));
89          client.print(" ");
90          client.print(pirValue = digitalRead(pirPin));
91          client.print(" ");
92          break;
93        }
```

Σχήμα 4.1: Απόσπασμα κώδικα από το Arduino IDE για την εγγραφή των τιμών των αισθητήρων στον webserver.

Ο αισθητήρας DHT11 είναι ένας ψηφιακός αισθητήρας θερμοκρασίας και υγρασίας. Με τη βιβλιοθήκη DHT.h και τις εντολές dht.readTemperature() καθώς και dht.readHumidity() διαβάζονται οι τιμές των αισθητήρων για μια δεδομένη στιγμή. Για τους αισθητήρες φωτιάς και βροχής επιλέγεται να χρησιμοποιηθούν οι αναλογικές τιμές έτσι ώστε να χρησιμοποιηθεί το εύρος 0 έως 1023 και μέσα από τις διαφορετικές τιμές να δημιουργήσουμε διαφορετικές ειδοποιήσεις. Συγκεκριμένα οι τιμές 0 έως 1023 τόσο για τον αισθητήρα βροχής όσο και για τον αισθητήρα φωτιάς αντιστοιχίζονται με τιμές από 0 έως 2 με την εντολή map [13]. Έτσι για παράδειγμα για τον αισθητήρα φωτιάς η τιμή 0 σημαίνει ότι υπάρχει φωτιά σε πολύ κοντινή απόσταση (από τη θέση του αισθητήρα), η τιμή 1 φωτιά σε σχετικά κοντινή απόσταση, ενώ η τιμή 2 σημαίνει καθόλου φωτιά. Αντίστοιχα για τον αισθητήρα βροχής η τιμή 0 σημαίνει ότι έξω βρέχει, η τιμή 1 σημαίνει ότι υπάρχει ελαφρά βροχόπτωση και η τιμή 2 ότι δε βρέχει. Οπότε υπάρχουν τρείς καταστάσεις για τους δύο αυτούς αισθητήρες. Στην (Εικόνα 4.6) φαίνονται οι τιμές που εμφανίζονται στον webserver. Η τιμή 30.90 είναι η τιμή της θερμοκρασίας σε βαθμούς κελσίου, η τιμή 44.00 είναι η υγρασία σε ποσοστό επί τοις εκατό, η πρώτη τιμή 2 είναι για

τον αισθητήρα φωτιάς και η δεύτερη για τον αισθητήρα βροχής. Τέλος η τιμή 0 είναι για τον ψηφιακό αισθητήρα κίνησης, όπου η τιμή 0 σημαίνει πως δεν ανίχνευσε κίνηση και τιμή 1 σημαίνει ότι ανίχνευσε.

Εικόνα 4.6: Τιμές που εμφανίζονται στον webserver. Από αριστερά προς τα δεξιά εμφανίζονται τιμή θερμοκρασίας, υγρασίας, φωτιάς, βροχής και κίνησης.

Οι τιμές των αισθητήρων ανανεώνονται συνεχώς καθώς το Arduino τρέχει τη συνάρτηση loop() ασταμάτητα. Το χρονικό διάστημα των ανανεώσεων αναλαμβάνει η εφαρμογή Android, αυτόματα χωρίς να χρειαστεί να κάνει κάτι ο χρήστης.

ΚΕΦΑΛΑΙΟ 5

ΕΠΙΣΚΟΠΗΣΗ ΤΟΥ ANDROID STUDIO

5.1 Εισαγωγή για το Android

Το Android είναι ένα λειτουργικό σύστημα ανοιχτού κώδικα σχεδιασμένο για συσκευές κινητής τηλεφωνίας, βασισμένο στον πυρήνα του Linux. Το Android αναπτύχθηκε από μια κοινοπραξία προγραμματιστών γνωστή ως Open Handset Alliance όπου συμμετέχουν πολλές διαφορετικές εταιρίες και χρηματοδοτείται από την Google. Για πρώτη φορά επιτεύχθηκε ο διαχωρισμός του hardware με το software το οποίο λειτουργεί με συνέπεια ενός πλουσιότερου οικοσυστήματος με περισσότερες συσκευές να είναι συμβατές με τις ίδιες εφαρμογές-applications [14].

Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλέφωνα και τα τάμπλετ, 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 σε κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, συνηθισμένους Η/Υ, προτζέκτορες και σε άλλες ηλεκτρονικές συσκευές.

Το λογισμικό Android αποτελεί το πιο διαδεδομένο λογισμικό παγκοσμίως, ιδιαίτερα στον χώρο της κινητής τηλεφωνίας (Smartphone). Πολλές εταιρίες έχουν δημιουργήσει το δικό τους λογισμικό βασισμένο στο Android όπως το One UI από την Samsung, το MI-UI από την Xiaomi, το Oxygen OS από την Oneplus και πολλά ακόμα.

Το λογισμικό συνεχώς εξελίσσεται και προσφέρει συνεχώς περισσότερες λειτουργίες, τεχνολογίες και υπηρεσίες στον χρήστη (Πίνακας 5.1). Για την έκδοση Android 10 (API 29) έχουν γίνει πολλές βελτιώσεις σε ότι αφορά την ασφάλεια, τη διαχείριση μπαταρίας και μνήμης, τα δικαιώματα και προσωπικά δεδομένα του χρήστη (καλύτερος έλεγχος στη χρήση δικαιωμάτων από εφαρμογές) ακόμα και βελτιώσεις στον χειρισμό μιας συσκευής. Τα παραδοσιακά πλήκτρα στο κάτω μέρος της οθόνης μπορούν να αντικατασταθούν με gestures έτσι ώστε ο χειρισμός να γίνει ευκολότερος, απελευθερώνοντας το κάτω μέρος της οθόνης από τα πλήκτρα. Κυριότερο ίσως χαρακτηριστικό του Android 10 και μεταγενέστερα είναι η υποστήριξη της τεχνολογίας 5G.

Πίνακας 5.1: Οι εκδόσεις του Android. Version/API.(Πηγή Android Studio)

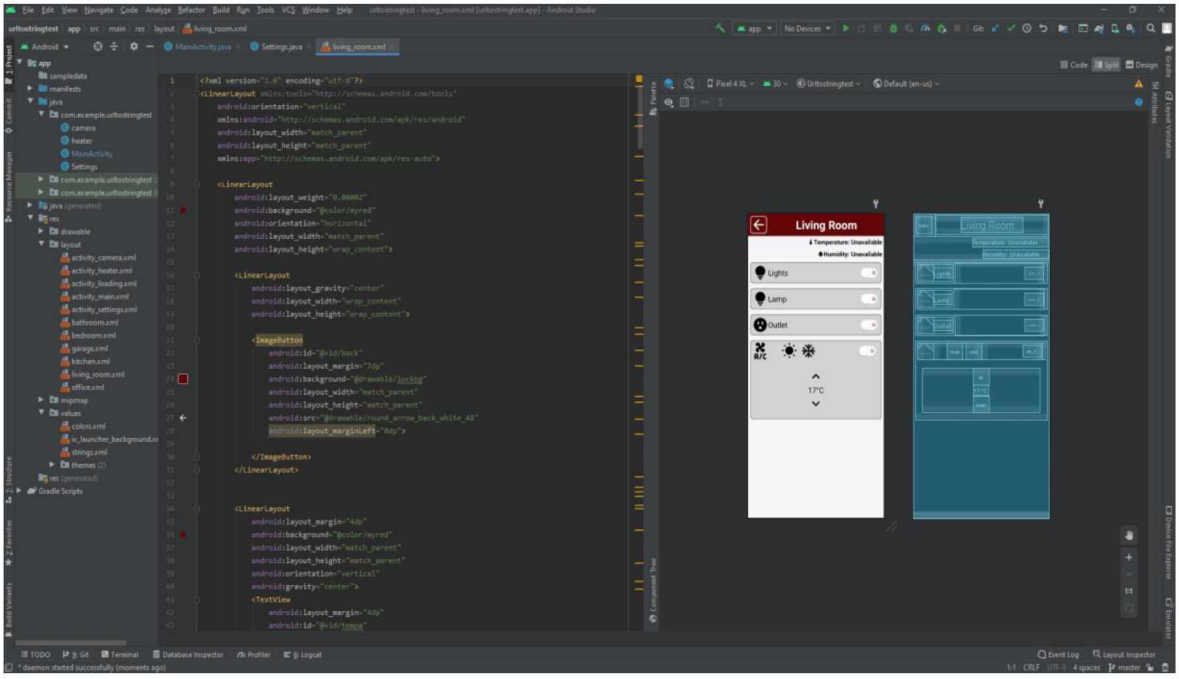
ANDROID PLATFORM VERSION	API LEVEL
4.0 Ice Cream Sandwich	15
4.1 Jelly Bean	16
4.2 Jelly Bean	17
4.3 Jelly Bean	18
4.4 KitKat	19
5.0 Lollipop	21
5.1 Lollipop	22
6.0 Marshmallow	23
7.0 Nougat	24
7.1 Nougat	25
8.0 Oreo	26
8.1 Oreo	27
9.0 Pie	28
10. Android 10	29

Το Application Programming Interface (API) είναι ένα σύνολο κανόνων, εργαλείων και ρυθμίσεων που χρησιμοποιούνται από τις εφαρμογές. Όσο μεγαλύτερη η βαθμίδα (API-Android) που είναι εγκατεστημένη σε μια συσκευή, τόσο νεότερη είναι. Εταιρίες και προγραμματιστές που δημιουργούν μια εφαρμογή πρέπει να κάνουν ένα πλάνο και να σκεφτούν ποια βαθμίδα API θα χρησιμοποιήσουν και αυτό εξαρτάται με τον αριθμό των χρηστών/συσκευών που θέλουν να χρησιμοποιούν την εφαρμογή τους [14]. Αν για παράδειγμα μια εφαρμογή φτιαχτεί για Android 10 (API 29) και νεότερη έκδοση ένας χρήστης που έχει Smartphone με έκδοση λογισμικού Android 8.1 Oreo (API 27) δε θα μπορεί να τη χρησιμοποιήσει. Η εφαρμογή Smart Home της διπλωματικής εργασίας μπορεί να τρέξει σε Android 5.0 (Lollipop) και μεταγενέστερες εκδόσεις.

5.2 Εισαγωγή στο Android Studio

5.2.1 Android Studio

Για τους προγραμματιστές το Android προσφέρει όλα τα εργαλεία που θα χρειαστεί για να δημιουργήσει μια εφαρμογή. Το Android Studio είναι ένα ολοκληρωμένο προγραμματιστικό περιβάλλον (IDE) που μπορεί κανείς να ξεκινήσει να χρησιμοποιεί για να φτιάξει μια εφαρμογή για οποιαδήποτε συσκευή κινητού, τάμπλετ, τηλεόρασης. Είναι συμβατό με τα λειτουργικά συστήματα των Windows, Mac OS, και Linux (Εικόνα 5.1). Η γλώσσα προγραμματισμού είναι η Java ή η Kotlin που είναι βασισμένη στην Java, κάτι που καθορίζεται στο ξεκίνημα ενός νέου project.

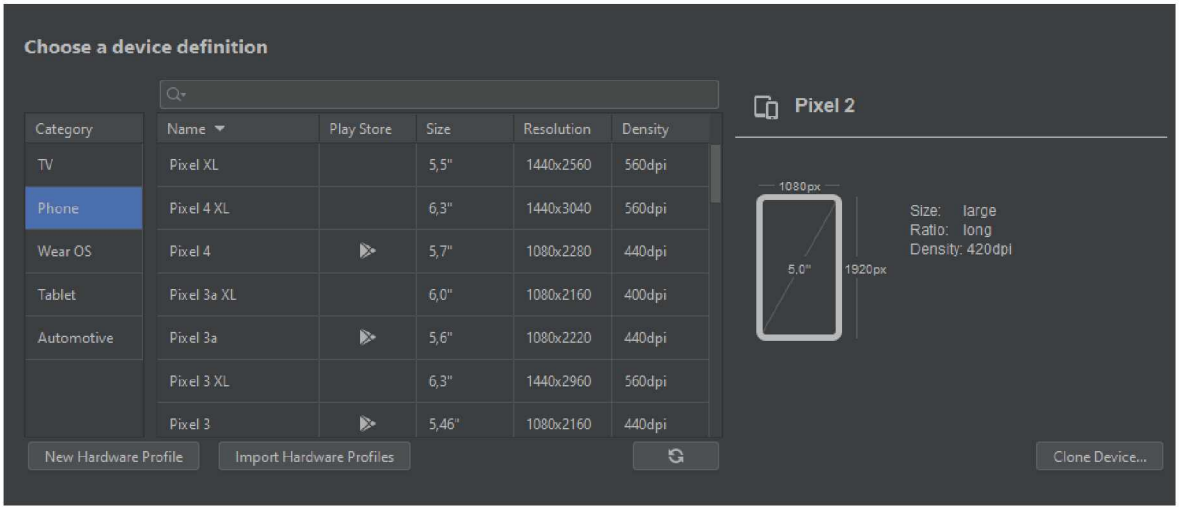


Εικόνα 5.1: Το περιβάλλον του Android Studio.

Στα θετικά, είναι ότι ένας προγραμματιστής δε χρειάζεται να έχει μια φυσική συσκευή για να προγραμματίσει την εφαρμογή του. Το Android Studio μπορεί να δημιουργήσει μια εικονική συσκευή (Android Emulator) η οποία προσομοιώνει το περιβάλλον μιας συσκευής Android, προκειμένου ο προγραμματιστής να δοκιμάζει την εφαρμογή που αναπτύσσει (Εικόνα 5.2). Απαραίτητη προϋπόθεση το hardware του ηλεκτρονικού υπολογιστή που χρησιμοποιεί ο προγραμματιστής (επεξεργαστική ισχύς, μνήμη RAM ηλεκτρονικού υπολογιστή) να είναι ικανό για την δημιουργία του Emulator. Κατά τη

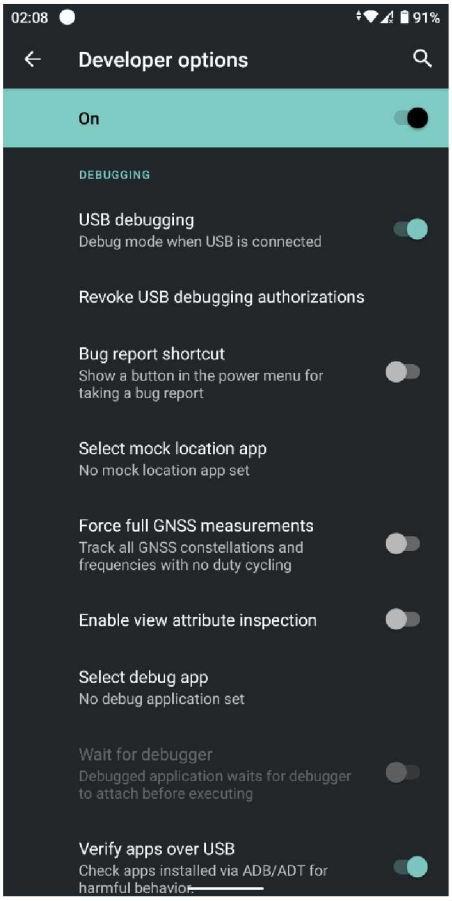
δημιουργία μιας εικονικής συσκευής καθορίζονται τα χαρακτηριστικά της συσκευής όπως μνήμη RAM, έκδοση λογισμικού Android και άλλα. Μπορεί ακόμα να επιλέξει μέσα από μια πληθώρα έτοιμων εικονικών συσκευών, έξυπνων κινητών τηλεφώνων, τηλεοράσεων, οθονών αυτοκινήτου (Εικόνα 5.3) [15].

Εικόνα 5.2: Εικονική συσκευή μέσω του Android Studio.



Εικόνα 5.3: Επιλογή εικονικής συσκευής μέσα από προκαθορισμένες επιλογές του Android Studio

Η χρήση μιας φυσικής συσκευής απαιτεί και αυτή κάποια βήματα πριν ο προγραμματιστής μπορεί να την χρησιμοποιήσει. Αρχικά συνδέεται η συσκευή μέσω USB στον υπολογιστή και πρέπει να εγκατασταθούν οι αντίστοιχοι οδηγοί (drivers) της συσκευής στον υπολογιστή. Στην συνέχεια στις ρυθμίσεις του smartphone στην επιλογή “Developer options” ενεργοποιείται το “USB debugging” και είναι έτοιμο για χρήση (Εικόνα 5.4) [16]. Για τις ανάγκες της διπλωματικής χρησιμοποιήθηκε μια φυσική συσκευή με Android 10.



Εικόνα 5.4: Απεικόνιση των επιλογών για προγραμματιστές και ενεργοποίηση του USB debugging στη φυσική συσκευή Android που χρησιμοποιήθηκε.

5.2.2 Java

Για την διπλωματική εργασία χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java. Η Java είναι μια class-based και object-oriented γλώσσα προγραμματισμού. Βασικός στόχος είναι να επιτρέπει στους προγραμματιστές ο κώδικας Java που γράφουν να μπορεί να τρέξει σε όλες τις πλατφόρμες που υποστηρίζουν την Java χωρίς να γίνεται compile ξανά ο κώδικας. Επίσης αποτελεί μια από τις ποιο διαδεδομένες και διάσημες γλώσσες προγραμματισμού. Στο Android Studio η Java χρησιμοποιείται για τα Activities.

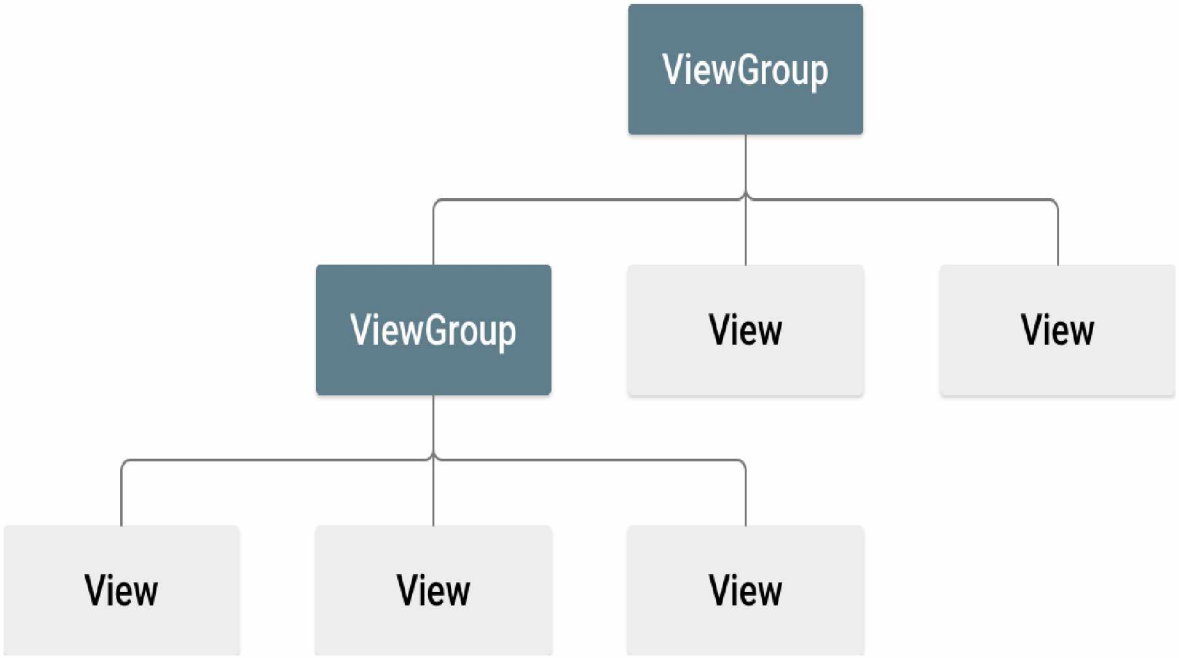
Ένα Activity είναι μια κλάση, ένα κρίσιμο συστατικό μιας εφαρμογής Android. Ένα Activity αποτελεί ουσιαστικά το παράθυρο στο οποίο η εφαρμογή προβάλλεται (UI-User Interface Design). Αυτό το παράθυρο συνήθως γεμίζει την οθόνη, ή μπορεί να είναι μικρότερο από την οθόνη και να αιωρείται πάνω από άλλα παράθυρα. Γενικά, ένα Activity είναι ουσιαστικά μία οθόνη σε μια εφαρμογή. Για παράδειγμα, ένα Activity μιας εφαρμογής

μπορεί να περιέχει μια οθόνη με ρυθμίσεις ή προτιμήσεις του χρήστη, ενώ ένα άλλο Activity να έχει μια επιλογή Select Photo, για ανέβασμα μιας φωτογραφίας στο διαδίκτυο. Για την εφαρμογή που αναπτύχθηκε activity είναι για παράδειγμα η οθόνη της κάμερας ή οι ρυθμίσεις στις οποίες ο χρήστης μπορεί να αλλάξει την διεύθυνση IP της κάμερας όπως θα δούμε στη συνέχεια.

Οι περισσότερες εφαρμογές περιέχουν πολλές οθόνες, πράγμα που σημαίνει ότι περιλαμβάνουν και πολλά Activities. Συνήθως, ένα Activity σε μια εφαρμογή ορίζεται ως main Activity, το οποίο αποτελεί την πρώτη οθόνη (main menu) που εμφανίζεται όταν ο χρήστης ξεκινά την εφαρμογή. Όλα ξεκινούν από εκεί και, αν η σχεδίαση είναι τέτοια, συνήθως τελειώνουν και εκεί. Κάθε Activity μπορεί στη συνέχεια να ξεκινήσει ένα άλλο Activity για να εκτελέσει διαφορετικές ενέργειες. Για παράδειγμα, το main Activity σε μια απλή εφαρμογή ηλεκτρονικού ταχυδρομείου μπορεί να εμφανίζει στην οθόνη τα εισερχόμενα email (main menu). Από εκεί, υπάρχουν κουμπιά (buttons) που ανοίγουν άλλα Activity και εμφανίζουν οθόνες για εργασίες όπως για παράδειγμα η σύνταξη email [17].

5.2.3 Layout – XML

Η XML είναι μία γλώσσα σήμανσης, που περιέχει ένα σύνολο κανόνων για την ηλεκτρονική κωδικοποίηση κειμένων. Κάθε Εφαρμογή Android έχει ένα κομμάτι σχεδίασης (UI) το οποίο περιλαμβάνει XML αρχεία (layout files), τα οποία συνδέονται μεταξύ τους αλλά και με τις Java κλάσεις για την γραφική απεικόνιση του ανάλογου Activity. Όλα τα στοιχεία σε ένα layout είναι φτιαγμένα σε μια ιεραρχία View και ViewGroup αντικειμένων. Τα View αποτελούν αντικείμενα με τα οποία ο χρήστης μπορεί να αλληλοεπιδράσει (συχνά αποκαλούμενα widgets), ενώ ένα ViewGroup που συνήθως λέγεται «layout» καθορίζει ουσιαστικά τη δομή και τη διάταξη των αντικειμένων, είτε αυτά είναι Views είτε ViewGroup αντικείμενα (Εικόνα 5.5). Για παράδειγμα, View αντικείμενα είναι ένα κουμπί στην οθόνη (Button), η εισαγωγή κειμένου (TextView), μια μπάρα αναζήτησης (search view) [18].



Εικόνα 5.5: Απεικόνιση της ιεραρχίας ενός Layout [18].

Για τα layouts (ViewGroups) υπάρχουν πολλών ειδών όπως το Linear Layout που χρησιμοποιήθηκε στην εργασία αυτή, το οποίο τοποθετεί τα αντικείμενα είτε οριζόντια σε μια γραμμή είτε κατακόρυφα σε μια στήλη [18]. Χρησιμοποιώντας εμφωλευμένα Linear Layout η εφαρμογή σχεδιάζεται έτσι ώστε να είναι συμβατή με όλα τα διαφορετικά μεγέθη οθονών και όχι μόνο για κινητά τηλέφωνα αλλά και για ταμπλέτες και τηλεοράσεις, χωρίς να δημιουργείται πρόβλημα και να καταστρέφεται ο σχεδιασμός.

Ένα layout μπορεί να δημιουργηθεί με δύο τρόπους:

- Καθορίζοντας τα αντικείμενα μέσω της XML. Το Android προσφέρει τη δυνατότητα χρησιμοποιώντας την XML να δημιουργηθούν τα αντικείμενα της διεπαφής του χρήστη τα οποία αντιστοιχούν σε αντικείμενα layout (π.χ. widgets, TextView)
- Δημιουργώντας αντικείμενα του layout προγραμματιστικά. Ο προγραμματιστής μπορεί να δημιουργήσει αντικείμενα View και ViewGroup και να χειριστεί τις ιδιότητες του προγραμματιστικά, δηλαδή με κώδικα σε γλώσσα προγραμματισμού Java για παράδειγμα.

Χρησιμοποιώντας την XML για να καθοριστεί η διεπαφή του χρήστη διαχωρίζει την εμφάνιση της εφαρμογής και τον σχεδιασμό της από τον κώδικα ο οποίος καθορίζει το πως συμπεριφέρεται και λειτουργεί η εφαρμογή. Επίσης χρησιμοποιώντας την XML είναι

ευκολότερο να σχεδιαστούν διαφορετικά layouts για διαφορετικού μεγέθους και προσανατολισμού οθονών. Δηλαδή, με την XML υπάρχει άμεσος τρόπος το layout που θα σχεδιαστεί να είναι συμβατό με όλες τις οθόνες κινητών τηλεφώνων, τάμπλετ, τηλεοράσεων όλων των μεγεθών. Είναι επίσης πιο εύκολο να σχεδιαστεί το layout με XML και να λειτουργεί για όλους τους πιθανούς προσανατολισμούς των οθονών, είτε οριζόντια είτε κατακόρυφα [18].

Τα εργαλεία του Android δίνουν στον προγραμματιστή την δυνατότητα να χρησιμοποιήσει όποιο τρόπο από τους δύο ή και τους δύο για το σχεδιασμό του UI της εφαρμογής του. Μπορεί για παράδειγμα να καθορίσει κάποια βασικά layout με XML και να επεμβαίνει στο σχεδιασμό τους προγραμματιστικά και να τον τροποποιεί.

ΚΕΦΑΛΑΙΟ 6

ΛΕΠΤΟΜΕΡΙΕΣ ΓΙΑ ΤΗΝ ΑΝΑΠΤΥΞΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

6.1 Εισαγωγή

Σε αυτό το σημείο θα παρουσιαστούν κάποιες βασικές έννοιες και τις λειτουργίες που πραγματοποιούν, όπως επίσης ο τρόπος που η Android εφαρμογή λαμβάνει τις πληροφορίες από τον web server του Arduino.

6.2 Shared Preferences

Στην εφαρμογή έχει γίνει χρήση της διεπαφής «Shared Preferences» [19]. Η διεπαφή αυτή αποτελεί μια μέθοδο αποθήκευσης τιμών (data) ως ζεύγη κλειδιών / τιμών σε ένα αρχείο στον χώρο αποθήκευσης της συσκευής ενός κινητού τηλεφώνου. Για παράδειγμα, μπορεί να δημιουργηθεί ένα κλειδί που είναι το "όνομα χρήστη" και για την τιμή, να αποθηκευτεί το όνομα χρήστη. Άλλα είδη δεδομένων που αποθηκεύονται είναι Strings, integers, floats, Booleans τα οποία αποθηκεύονται με σκοπό να χρησιμοποιηθούν με κάποιο τρόπο στη συνέχεια. Με αυτό τον τρόπο υπάρχει δυνατότητα δεδομένα να είναι διαθέσιμα σε όλη την εφαρμογή και σε όλα τα activities.

Χρησιμοποιώντας τον editor της διεπαφής «Shared Preferences» ο προγραμματιστής έχει τη δυνατότητα να καταχωρήσει κλειδιά με τις αντίστοιχες τιμές τους. Ανάλογα με το είδος μεταβλητής που θέλει να αποθηκεύσει, χρησιμοποιεί την αντίστοιχη μέθοδο. Αν παράδειγμα η μεταβλητή είναι ένα string, χρησιμοποιεί την putString() αν είναι ακέραιος αριθμός την putInt() και στη συνέχεια καλεί τη μέθοδο apply() για να αποθηκευτούν οι τιμές [19].

Για την υλοποίηση της εφαρμογής για Android στην διπλωματική εργασία, χρησιμοποιήθηκε η μέθοδος αποθήκευσης «Shared Preferences». Μέσω αυτής της διαδικασίας αποθηκεύονται επιλογές που κάνει ο χρήστης έτσι ώστε το περιβάλλον και η περιήγηση στην εφαρμογή να προσομοιώνουν το περιβάλλον παρόμοιων εφαρμογών που υπάρχουν διαθέσιμες.

Για παράδειγμα, για το δωμάτιο Living Room έχουν χρησιμοποιηθεί έξι τιμές. Τέσσερις από αυτές είναι Boolean και αποθηκεύεται η κατάσταση των διακοπτών του δωματίου (φώτα(lall), λαμπατέρ(l1), πρίζα(out), κλιματιστικό(ac)). Μια συμβολοσειρά (mode) χρησιμοποιείται για το πρόγραμμα του κλιματιστικού (κρύο, ζεστό) και τέλος ένας ακέραιος αριθμός (tempac) όπου αποθηκεύεται η επιλεχθείσα θερμοκρασία του κλιματιστικού (Σχήμα 6.1).

```
SharedPreferences.Editor editor = getSharedPreferences( name: "livingroom", MODE_PRIVATE).edit();
editor.putBoolean("lall", lall.isChecked());
editor.putBoolean("l1", l1.isChecked());
editor.putBoolean("out", out.isChecked());
editor.putBoolean("ac", ac.isChecked());
editor.putString("mode",modew);
editor.putInt("tempac",temparc);
editor.apply();
```

Σχήμα 6.1: Ο κώδικας για την αποθήκευση των επιλογών του χρήστη στην οθόνη του σαλονιού.

Έτσι, ακόμα και αν κλείσει την οθόνη του σαλονιού ή ελαχιστοποιήσει ακόμα και να απενεργοποιήσει την εφαρμογή, οι τελευταίες επιλογές που έκανε είναι αποθηκευμένες. Μόλις χρησιμοποιήσει ξανά την εφαρμογή οι τιμές αυτές θα φορτωθούν. Έτσι έρχεται πιο κοντά στην πραγματικότητα (simulation) η χρήση και η περιήγηση του χρήστη στο περιβάλλον της εφαρμογής.

6.3 Handlers

Ένας handler επιτρέπει να σταλθούν και να επεξεργαστούν διάφορα αντικείμενα, όπως μηνύματα ή κάποια άλλη εκτέλεση (π.χ. κλήση κάποιας κλάσης).

Υπάρχουν δύο βασικές χρήσεις για έναν handler: (1) για τον προγραμματισμό μηνυμάτων και runnables που θα εκτελεστούν κάποια στιγμή στο μέλλον. και (2) για να ενεργοποιήσετε μια ενέργεια που θα εκτελεστεί σε διαφορετικό νήμα (thread) [20].

Στην παρούσα πτυχιακή εργασία, handlers χρησιμοποιούνται και για τους δύο παραπάνω λόγους. Ένας handler αναλαμβάνει να καλεί την συνάρτηση sensor_val() η οποία

αναλύεται στην επόμενη ενότητα. Κάποιος άλλος handler αναλαμβάνει επίσης να κάνει διάφορες λειτουργίες σε κάποιο άλλο thread, και όχι το βασικό (main thread). Περισσότερα για τα threads στην ενότητα 6.5.

6.4 Τιμές αισθητήρων στην εφαρμογή Android

Η συνάρτηση που είναι υπεύθυνη για να διαβάζει τις τιμές από τον web server του Arduino είναι η sensor_val(). Ουσιαστικά πρόκειται για μια συνάρτηση η οποία καλείται ανά έξι δευτερόλεπτα με τη χρήση ενός handler. Αρχικά χρησιμοποιείται η κλάση URL. Η κλάση URL παίρνει την τοπική διεύθυνση του Arduino (http://192.168.2.40).Προαιρετικά μπορεί να καθορίσει η "θύρα" (port), οποία πραγματοποιείται η σύνδεση TCP. Στη συνέχεια διαβάζεται το κείμενο από μια ροή χαρακτήρων και αποθηκεύεται σε μια συμβολοσειρά (string) μέχρι να βρεθεί NULL χαρακτήρας.

Στην περίπτωση αυτή θα διαβαστούν και θα αποθηκευτούν σε ένα string τα περιεχόμενα του web server, δηλαδή οι τιμές των αισθητήρων. Το string μεταξύ των τιμών των αισθητήρων έχει ένα κενό χαρακτήρα, όπως είχε προγραμματιστεί στο κεφάλαιο 4. Έτσι χωρίζεται η συμβολοσειρά σε έναν πίνακα με τέσσερις τιμές με την εντολή split(“ “) η οποία χωρίζει ένα string με βάση την παράμετρο στην παρένθεση, στην προκειμένη περίπτωση τον κενό χαρακτήρα. Έτσι από αυτό το σημείο οι τιμές των αισθητήρων μπορούν να χρησιμοποιηθούν σε οποιοδήποτε σημείο της εφαρμογής.

6.5 Thread

Κάθε φορά που ξεκινά μια εφαρμογή, το σύστημα δημιουργεί ένα νήμα που ονομάζεται «main» για την εφαρμογή. Το main thread (επίσης γνωστό ως "UI Thread") είναι νήμα όπου μια εφαρμογή αλληλοεπιδρά με λειτουργικά στοιχεία της διεπαφής του χρήστη (User Interface). Για παράδειγμα, εάν ο χρήστης αγγίξει ένα κουμπί στην οθόνη, το νήμα διεπαφής χρήστη αποστέλλει την επιλογή αφής στο αντικείμενο, το οποίο στη συνέχεια ολοκληρώνει ένα αίτημα που έχει προγραμματιστεί να κάνει [21].

Η συνάρτηση sensor_val τρέχει σε ένα ξεχωριστό νήμα (Thread) καθότι το Android δεν επιτρέπει network requests στο βασικό νήμα (σφάλμα NetworkOnMainThreadException). Ο κυριότερος λόγος είναι ότι το main thread δε διαθέτει μεγάλη μνήμη για τέτοιες

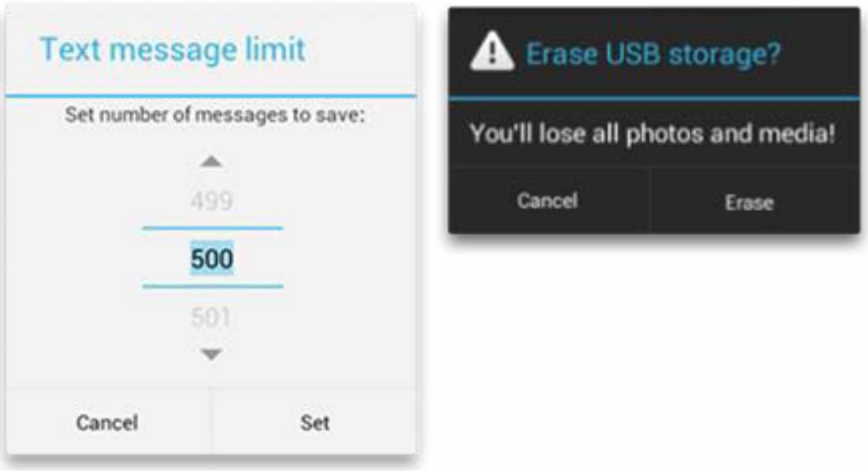
διαδικασίες . Επίσης η επιτυχία επικοινωνίας δεν είναι δεδομένη, άρα δε θα ήταν και αποδοτικό να γίνουν networks requests στο βασικό νήμα καθότι ενδέχεται να απορροφά πόρους άδικα [22].

Όταν χρησιμοποιείται ένα νήμα για όλες τις εργασίες αυτό θα οδηγήσει σε κακή γενική απόδοση της εφαρμογής. Συγκεκριμένα, εάν το UI thread ήταν υπεύθυνο για τα όλες τις λειτουργίες σε μια εφαρμογή εκτελώντας χρονοβόρες και μεγάλες λειτουργίες, όπως network requests ή αιτήματα σε μια βάση δεδομένων , τότε όλο το UI θα κατέρρεε. Δε θα δούλευε καμία λειτουργία σωστά και στο χρήστη θα φαινόταν απλά η οθόνη να έχει «παγώσει» [21].

6.6 Dialogs

6.6.1 Dialog

Ένα Dialog είναι συνήθως ένα μικρό παράθυρο που ζητά από το χρήστη να λάβει απόφαση ή να εισαγάγει πρόσθετες πληροφορίες όπως για παράδειγμα όνομα χρήστη και κωδικό πρόσβασης. Ένα Dialog μπορεί να καταλαμβάνει ολόκληρη την οθόνη ή κάποιο μέρος της(Εικόνα 6.1).



Εικόνα 6.1: Κάποια dialogs που ο χρήστης πρέπει να επιλέξει ή να συμπληρώσει κάποια πληροφορία ή να πάρει κάποια απόφαση για κάποιο ερώτημα [23].

Επίσης ο προγραμματιστής μπορεί να σχεδιάσει τα δικά του layout σε ένα dialog. Μπορεί δηλαδή να σχεδιάσει ένα dialog χρησιμοποιώντας XML όπως θα σχεδίαζε ένα Activity.

6.6.2 Alert Dialog

Η κλάση Alert Dialog επιτρέπει σε έναν προγραμματιστή να δημιουργήσει πολλών ειδών σχέδια dialog. Πολλές φορές είναι το μόνο που θα χρειαστεί μια εφαρμογή. Ένα Alert Dialog χωρίζεται σε τρία κομμάτια (Εικόνα 6.2) [23]:

1. Τίτλος

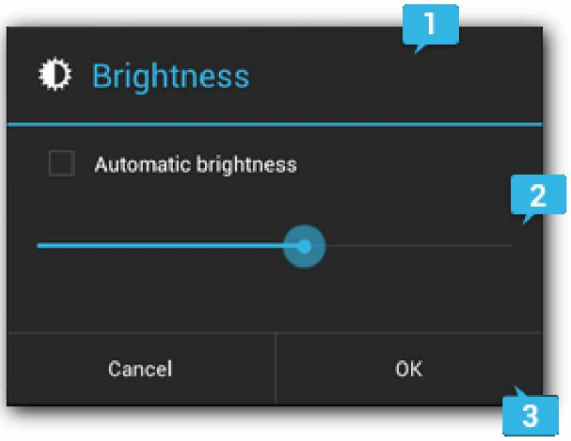
   Δεν είναι υποχρεωτικό να υπάρχει. Δίνει μια περίληψη της ερώτησης που υπάρχει στα περιεχόμενα. Βοηθά το χρήστη να πάρει απόφαση.

2. Περιεχόμενα

   Μπορεί να είναι ένα μήνυμα, μια λίστα ακόμα και ένα νέο layout.

3. Πλήκτρα

   Δεν θα πρέπει να υπάρχουν πάνω από τρία πλήκτρα σε ένα Alert Dialog.

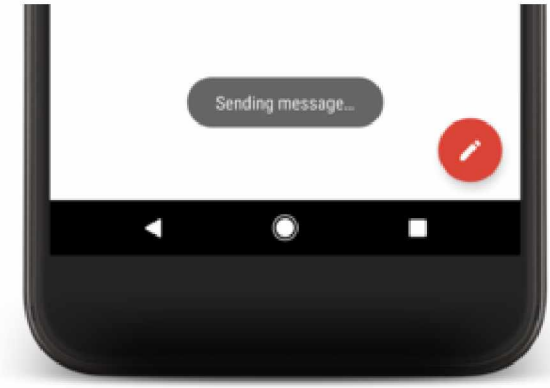


Εικόνα 6.2: Απεικόνιση ενός Alert Dialog [23].

6.7 Toast

Ένα toast δίνει πίσω στον χρήση μια ανάδραση (feedback) για μια λειτουργία. Πρόκειται για ένα μικρό αναδυόμενο μήνυμα. Χρησιμοποιεί όσο χώρο χρειάζεται (ανάλογα το μέγεθος του μηνύματος) και εμφανίζεται συνήθως στο κάτω μέρος της οθόνης. Έτσι η οθόνη της εφαρμογής που χρησιμοποιείται παραμένει το επίκεντρο και στο πρώτο πλάνο και συνεχίζει να είναι λειτουργική, δηλαδή το μήνυμα δεν διακόπτει τη λειτουργία της. Τα toasts εμφανίζονται για συγκεκριμένο χρονικό διάστημα το οποίο καθορίζεται από τον προγραμματιστή [24].

Για παράδειγμα πατώντας «Send» σε μια εφαρμογή ηλεκτρονικής αλληλογραφίας, αυτό θα πυροδοτήσει ένα toast «Sending message…» (Εικόνα 6.3)



Εικόνα 6.3: Απεικόνιση μηνύματος toast σε αποστολή email [24].

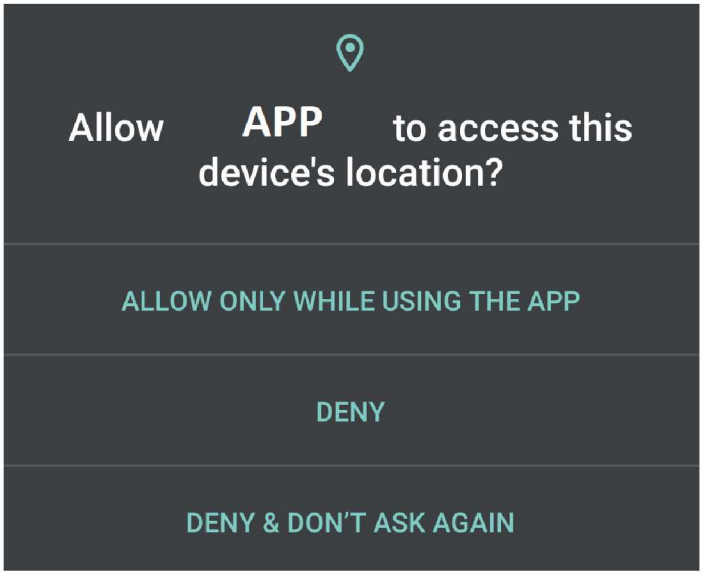
6.8 Permissions

Πολύ σημαντικό κεφάλαιο για τους χρήστες Android ή Apple smartphones και γενικότερα για τους καταναλωτές είναι τα δικαιώματα (permissions) που ζητούν οι εφαρμογές για να λειτουργήσουν. Κάθε εφαρμογή μπορεί να λειτουργήσει σε ένα κινητό τηλέφωνο με κάποιους περιορισμούς. Αν η εφαρμογή χρειάζεται περισσότερα στοιχεία από τον χρήστη για να λειτουργήσει όπως θα δούμε παρακάτω, ο καταναλωτής πρέπει να ενημερώνεται. Γενικότερα πρέπει να προστατεύεται με κάθε δυνατό τρόπο η ιδιωτικότητα των χρηστών.

Αν μια εφαρμογή χρησιμοποιεί τα λεγόμενα «επικίνδυνα» δικαιώματα (dangerous permissions ή runtime permissions) και η εφαρμογή εγκατασταθεί σε συσκευή με Android 6.0 (API level 23) και μεταγενέστερες εκδόσεις, τότε υποχρεωτικά ο χρήστης πρέπει να δώσει άδεια στην εφαρμογή κατά την έναρξη της εφαρμογής.

Τα runtime permissions δίνουν πρόσβαση σε μια εφαρμογή σε δεδομένα του χρήστη και επιτρέπουν στην εφαρμογή να πραγματοποιούν ενέργειες η οποίες επηρεάζουν το σύστημα ή και άλλες εφαρμογές. Έτσι, μια εφαρμογή πρέπει να ζητά την άδεια πρόσβασης από τον χρήστη καθώς θα μια εφαρμογή μπορεί να έχει πρόσβαση σε ευαίσθητα

δεδομένα του χρήστη όπως είναι η επαφές του, τη διεύθυνση ηλεκτρονικού ταχυδρομείου του ακόμα και την τοποθεσία του (Εικόνα 6.4) [25].

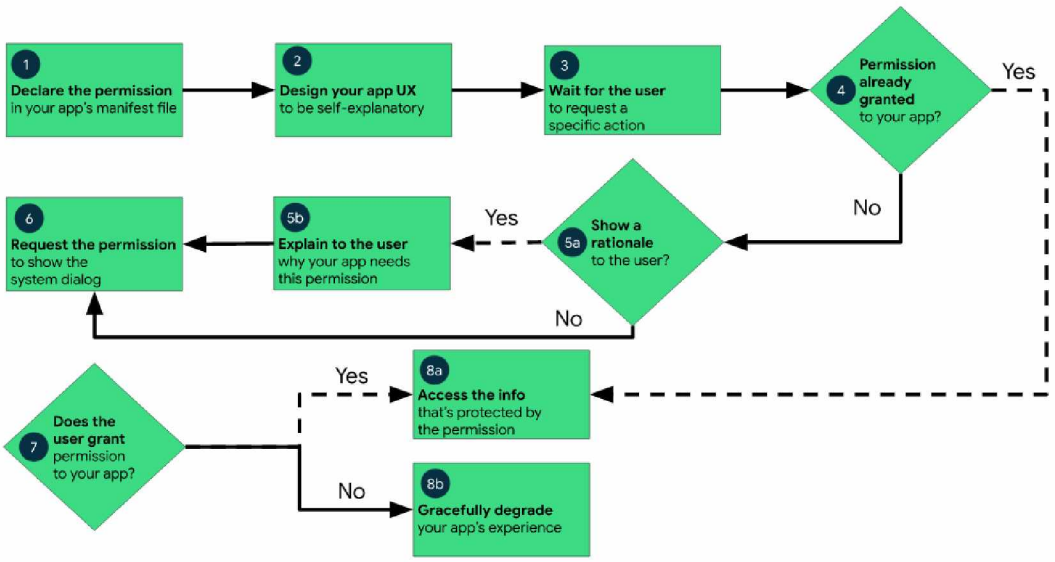


Εικόνα 6.4: Απεικόνιση μιας εφαρμογής που ζητά από το χρήστη αδειοδότηση για πρόσβαση σε ευαίσθητο περιεχόμενο, την τοποθεσία του.

Υπάρχουν κάποιες βασικές αρχές που καλό είναι να ακολουθούνται από τους προγραμματιστές σε ότι αφορά τα permissions [25].

- Η εφαρμογή ήταν καλό να σχεδιαστεί με τέτοιο τρόπο ώστε η περιήγηση σε αυτή να είναι ελεύθερη και τα αντίστοιχα permissions να ζητώνται όταν ο χρήστης θελήσει να χρησιμοποιήσει κάποια λειτουργία της εφαρμογής.
- Αν ο χρήστης δεν αποδεχτεί την αδειοδότηση κάποιου permission, τότε μπορεί να συνεχίσει να χρησιμοποιεί την εφαρμογή εκτός από την λειτουργία/λειτουργίες που δεν έδωσε την άδεια.

Επίσης θα πρέπει να γίνεται αξιολόγηση από τον προγραμματιστή κατά πόσο χρειάζονται πραγματικά κάποια permissions για να λειτουργήσει η εφαρμογή. Αν τελικά καταλήξει ότι το application πρέπει να έχει άδεια σε dangerous permissions πρέπει να γίνουν ορισμένα βήματα όπως φαίνεται στο Σχήμα 6.2 [25].



Σχήμα 6.2: Απεικόνιση ροής βημάτων που πρέπει να ακολουθούνται για τα permissions μιας εφαρμογής [25].

Για την εφαρμογή Smart Home της παρούσας διπλωματικής εργασίας τα dangerous permissions δεν χρειάστηκαν. Χρησιμοποιήθηκαν permission που δεν χρειάζεται πλέον ο χρήστης να ενημερώνεται. Συγκεκριμένα για την υλοποίηση της εφαρμογής χρησιμοποιήθηκαν τα εξής δικαιώματα:

1. Internet permission. Δίνει πρόσβαση στην εφαρμογή στο διαδίκτυο μέσω Wi-Fi.
2. Vibrate. Δίνει πρόσβαση στην εφαρμογή στην λειτουργία της δόνησης. Χρησιμοποιείται για τις ειδοποιήσεις.

ΚΕΦΑΛΑΙΟ 7

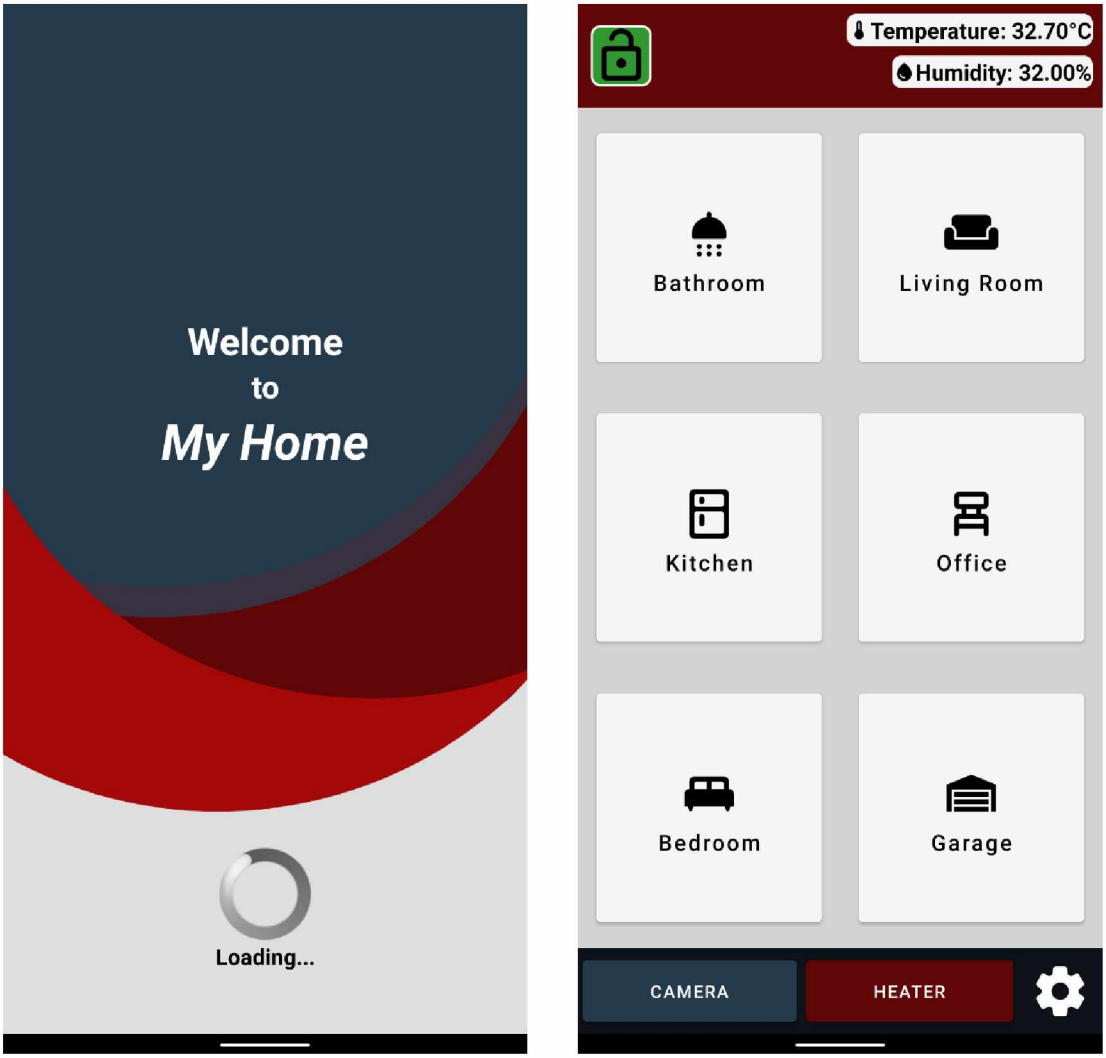
ΕΦΑΡΜΟΓΗ MY HOME

7.1 Περιβάλλον εφαρμογής

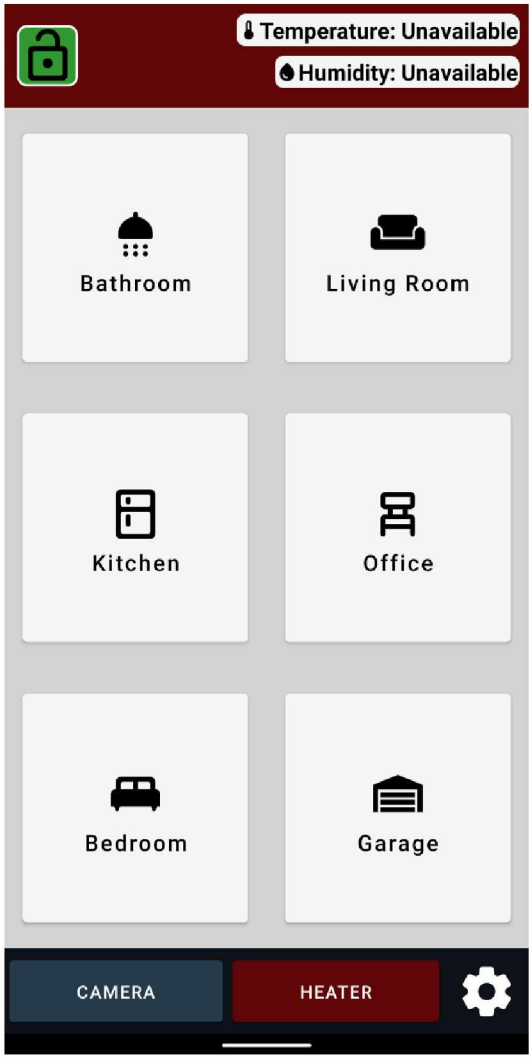
Για τις ανάγκες τις διπλωματικής εργασίας σχεδιάστηκε ένα logo (Εικόνα 7.1) για την εφαρμογή Smart Home (My Home) χρησιμοποιώντας το AdobeXD, ένα σχεδιαστικό εργαλείο το οποίο χρησιμοποιείται κυρίως για το σχεδιασμό εφαρμογών του διαδικτύου (web applications) και εφαρμογές για κινητές συσκευές.

Εικόνα 7.1: Logo της εφαρμογής σχεδιασμένο στο AdobeXD.

Ανοίγοντας την εφαρμογή ο χρήστης θα δει μια οθόνη καλωσορίσματος και στη συνέχεια θα βρεθεί στην κεντρική οθόνη της εφαρμογής (Εικόνα 7.2). Πάνω δεξιά βρίσκονται οι τιμές από τους αισθητήρες θερμοκρασίας και υγρασίας οι οποίες ανανεώνονται ανά έξι δευτερόλεπτα με τη χρήση ενός handler, οποίος καλεί τη συνάρτηση sensor_val(), υπεύθυνη για την επικοινωνία με το Arduino και τις πληροφορίες από τους αισθητήρες. Αν υπάρχει αδυναμία σύνδεσης με τον web server του Arduino στις τιμές της θερμοκρασίας και υγρασίας εμφανίζεται «Unavailable» (Εικόνα 7.3). Όταν πραγματοποιηθεί η σύνδεση τότε οι τιμές θα ανανεωθούν με τις τελευταίες τιμές.

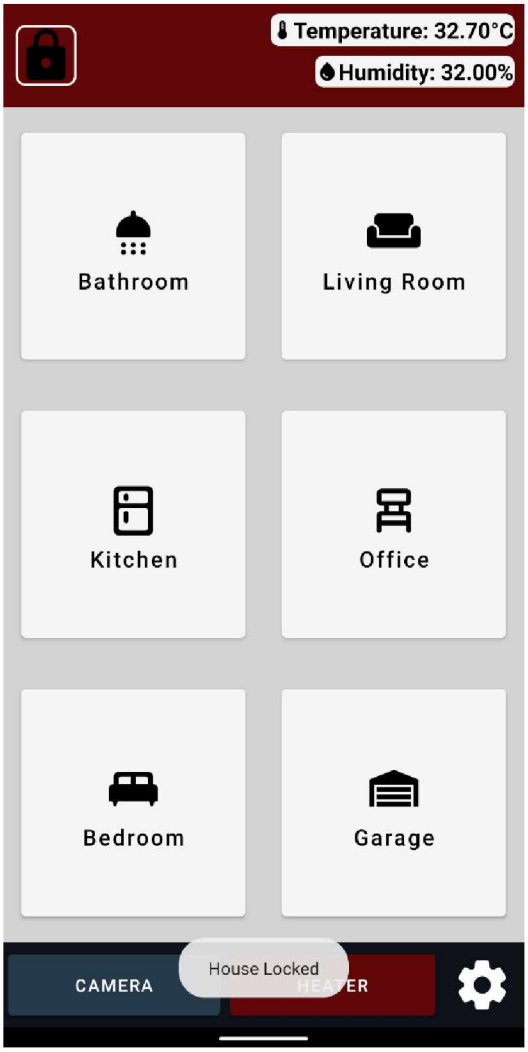


Εικόνα 7.2: (Αριστερά) Οθόνη καλωσορίσματος. (Δεξιά) Κεντρικό μενού εφαρμογής.



Εικόνα 7.3: Αδυναμία επικοινωνίας με τον web server. Οι τιμές θερμοκρασίας και υγρασίας είναι μη διαθέσιμες.

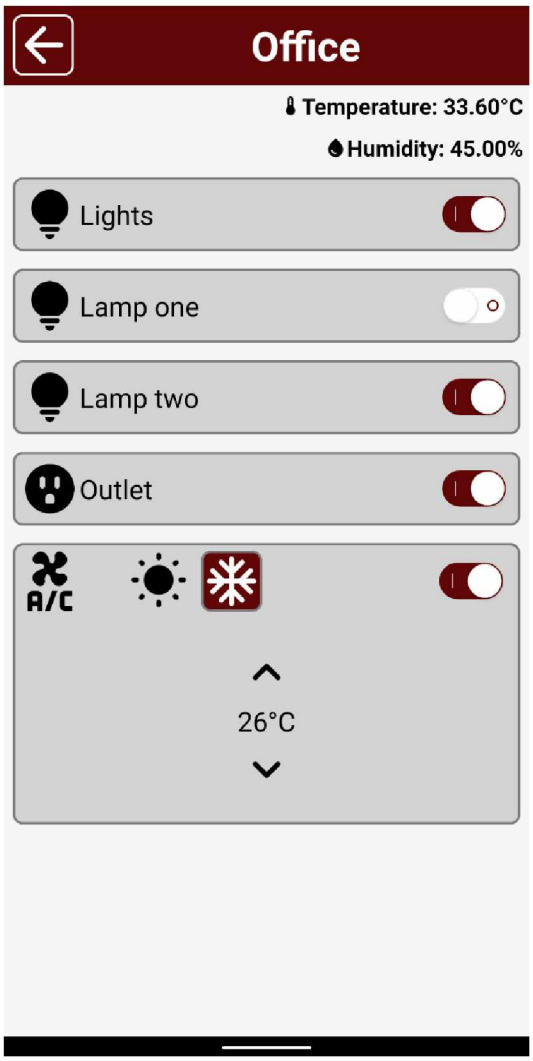
Επίσης, στο πάνω αριστερό μέρος βρίσκεται το κουμπί Lock/Unlock που κλειδώνει ή ξεκλειδώνει το σπίτι, ουσιαστικά ενεργοποιώντας ή απενεργοποιώντας αντίστοιχα τον αισθητήρα κίνησης που υπάρχει στην κεντρική είσοδο. Όταν πατηθεί το Lock/Unlock αλλάζει το εικονίδιο του λουκέτου ανάλογα με την επιλεγμένη κατάσταση και εμφανίζεται ένα αντίστοιχο μικρό αναδυόμενο μήνυμα στο κάτω μέρος της οθόνης (toast) (Εικόνα 7.4).



Εικόνα 7.4: Κλείδωμα του σπιτιού και εμφάνιση του αντίστοιχου μηνύματος (toast).

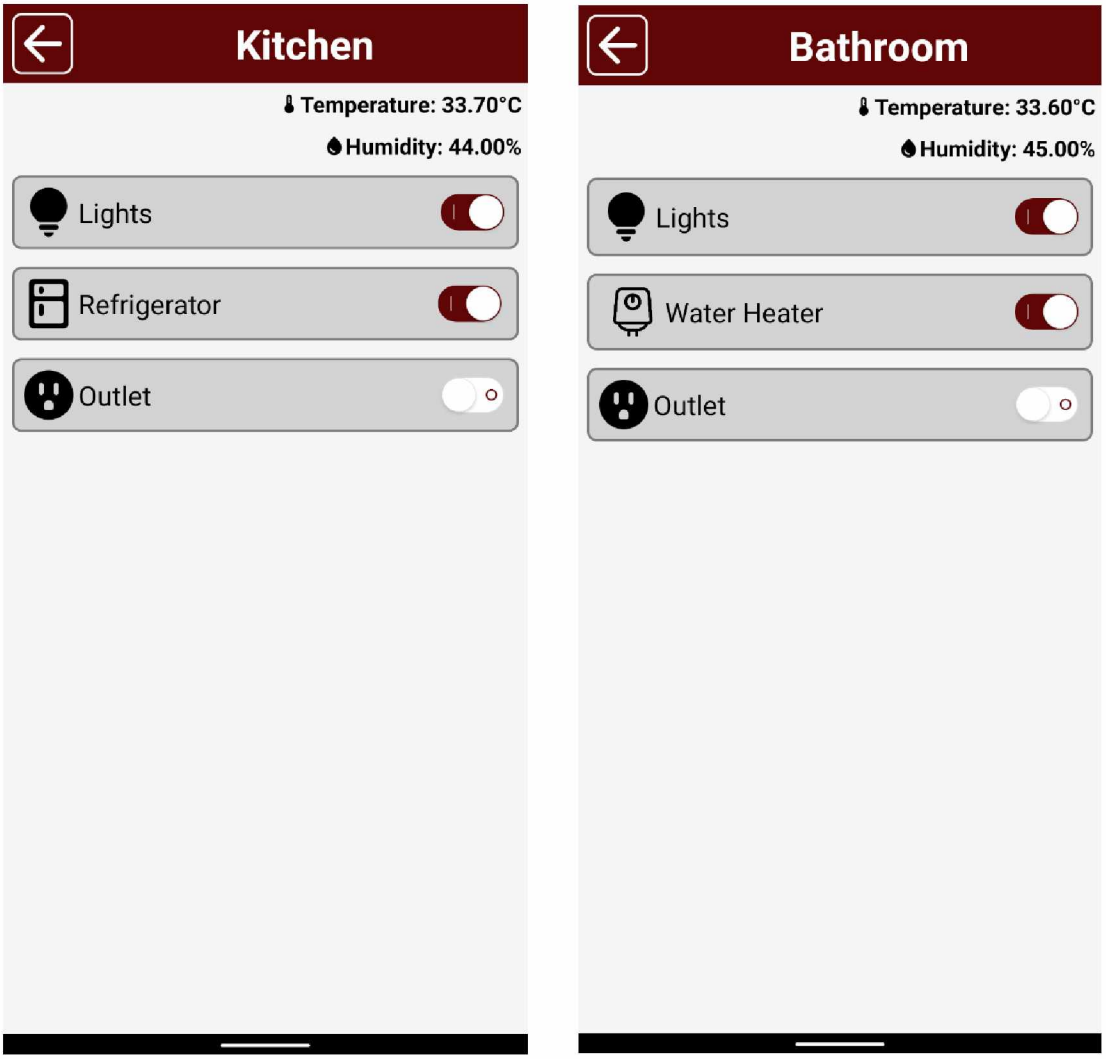
Από το κεντρικό μενού ο χρήστης μπορεί να περιηγηθεί σε όλους τους χώρους του σπιτιού όπως υπνοδωμάτιο, γραφείο, σαλόνι και να χρησιμοποιήσει τις διάφορες λειτουργίες σε κάθε δωμάτιο. Για παράδειγμα πατώντας το πλήκτρο του γραφείου θα βρεθεί στην αντίστοιχη οθόνη του γραφείου (Εικόνα 7.5). Εκεί μπορεί να ανοίξει τα φώτα ή μικρά λαμπατέρ που υπάρχουν στο δωμάτιο, να ενεργοποιήσει ή να απενεργοποιήσει τη λειτουργία μιας πρίζας. Ακόμα να χειριστεί το air conditioner σε λειτουργία κρύου ή ζεστού και να ρυθμίσει τους βαθμούς (κελσίου) λειτουργίας του. Επίσης στο δεξιά πάνω μέρος της οθόνης βρίσκονται οι ενδείξεις της θερμοκρασίας και υγρασίας του συγκεκριμένου δωματίου και υπάρχουν για όλα τα δωμάτια αντίστοιχες ενδείξεις θερμοκρασίας και υγρασίας. Ο διακόπτης (switch) που έχει χρησιμοποιηθεί σε όλα τα

δωμάτια του σπιτιού, είναι ένας custom διακόπτης από το GitHub ο οποίος ενσωματώθηκε στο application [26].



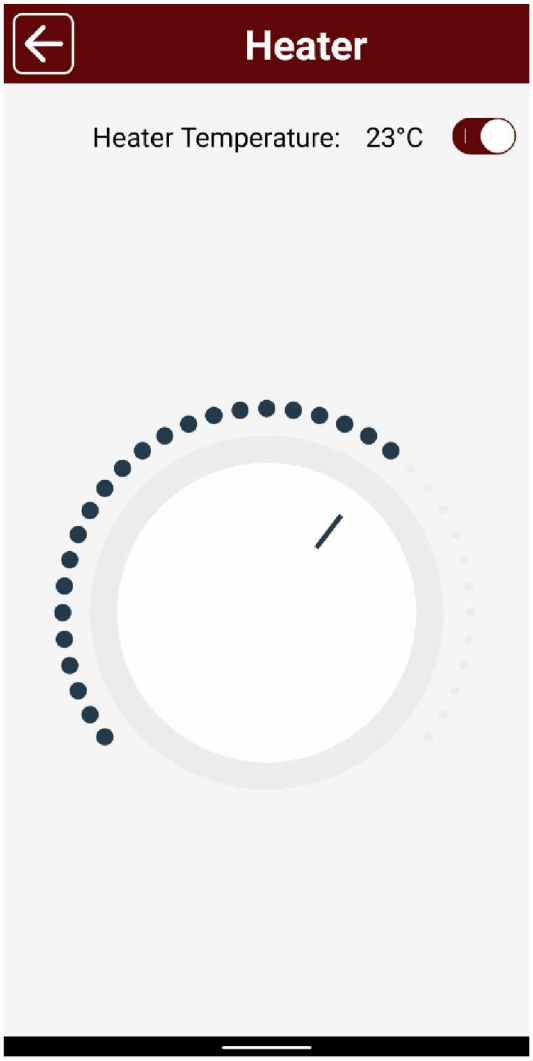
Εικόνα 7.5: Οθόνη γραφείου. Ενεργοποίηση κάποιων στοιχείων και ενεργοποίηση air conditioner.

Αντίστοιχες οθόνες υπάρχουν και για τα υπόλοιπα δωμάτια όπως το μπάνιο, η κουζίνα, το υπνοδωμάτιο, το σαλόνι και το γκαράζ με διάφορες συσκευές και πληροφορίες όπως η θερμοκρασία και η υγρασία. Στο μπάνιο ο χρήστης μπορεί να ανοίξει τα φώτα, το θερμοσίφωνα και να ενεργοποιήσει ή να απενεργοποιήσει τη λειτουργία μιας πρίζας. Στην κουζίνα αντίστοιχα να ανοίξει το φως, ενεργοποίηση ή απενεργοποίηση του ψυγείου και μιας πρίζας καθώς και η θερμοκρασία και η υγρασία (Εικόνα 7.6).



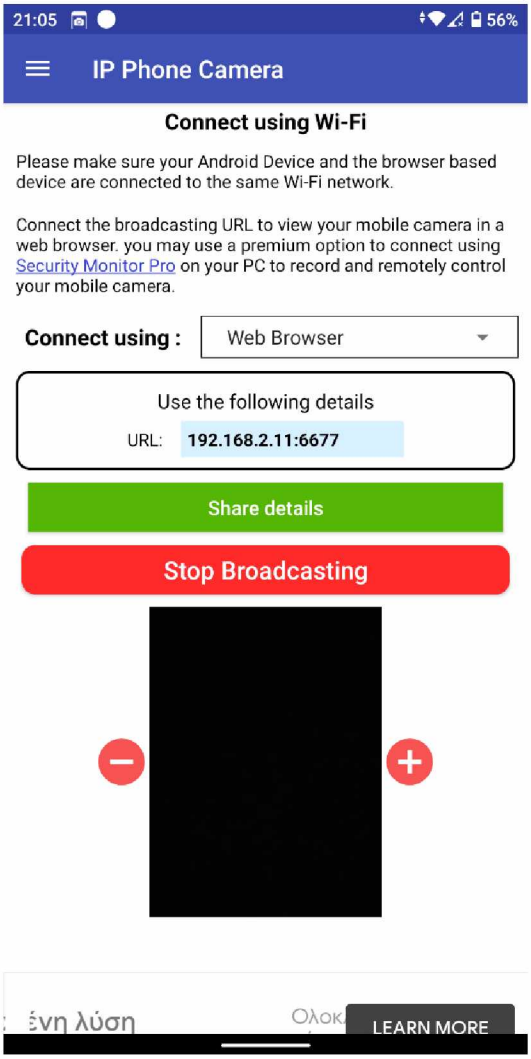
Εικόνα 7.6: (Αριστερά) Οθόνη κουζίνας. (Δεξιά) Οθόνη μπάνιου.

Επίσης, από την κεντρική οθόνη (κεντρικό μενού) ο χρήστης μπορεί να μεταβεί στην οθόνη της κεντρικής θέρμανσης (πλήκτρο Heater) και να την ενεργοποιήσει με το αντίστοιχο switch. Επίσης μπορεί να ρυθμίσει την θερμοκρασία. Χρησιμοποιήθηκε ένα custom Seekbar από το GitHub [27], το croller. Seekbar είναι μια ρυθμιζόμενη μπάρα από τον χρήστη που δείχνει την πρόοδο μιας τιμής, στην προκειμένη περίπτωση την επιλογή της θερμοκρασίας για την θέρμανση. Συγκεκριμένα, το croller είναι ένας κυκλικός εικονικός διακόπτης με τον οποίο ο χρήστης θα μπορεί να ρυθμίσει την θερμοκρασία (Εικόνα 7.7).



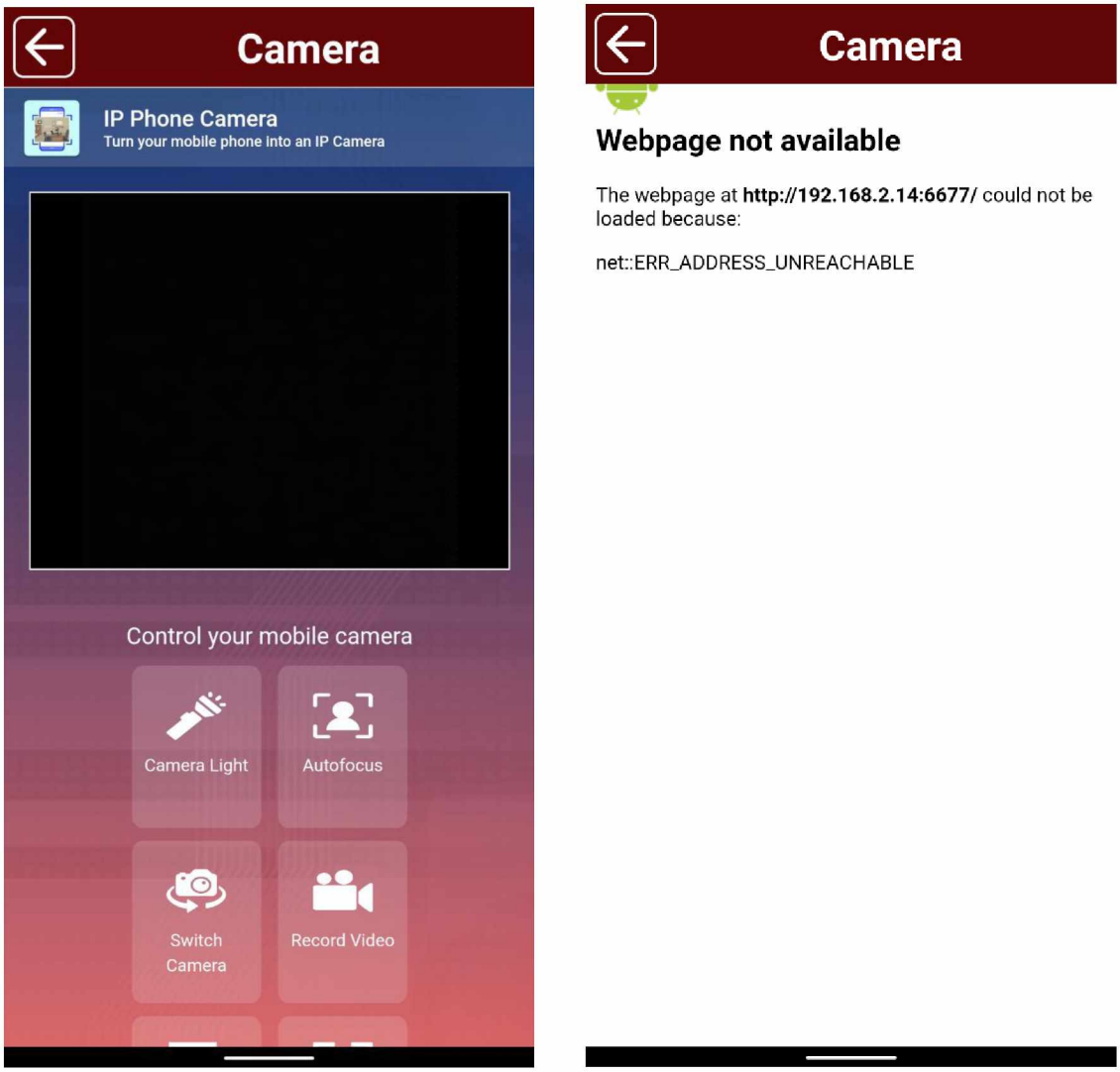
Εικόνα 7.7: Ρύθμιση κεντρικής θέρμανσης από το μενού Heater.

Για την λειτουργία της κάμερας έχει χρησιμοποιηθεί μια εξωτερική εφαρμογή από το Play Store, η «IP Phone Camera» [28]. Μέσω αυτής της εφαρμογής η οποία λειτουργεί στο παρασκήνιο, μετατρέπεται ένα smartphone σε μια IP camera και προβάλλει ζωντανά βίντεο από την κάμερα που διαθέτει σε τοπική διεύθυνση του δικτύου όπως φαίνεται στην εικόνα 7.8.



Εικόνα 7.8: Απεικόνιση IP Phone Camera application. Φαίνεται ένα preview του broadcast και το URL.

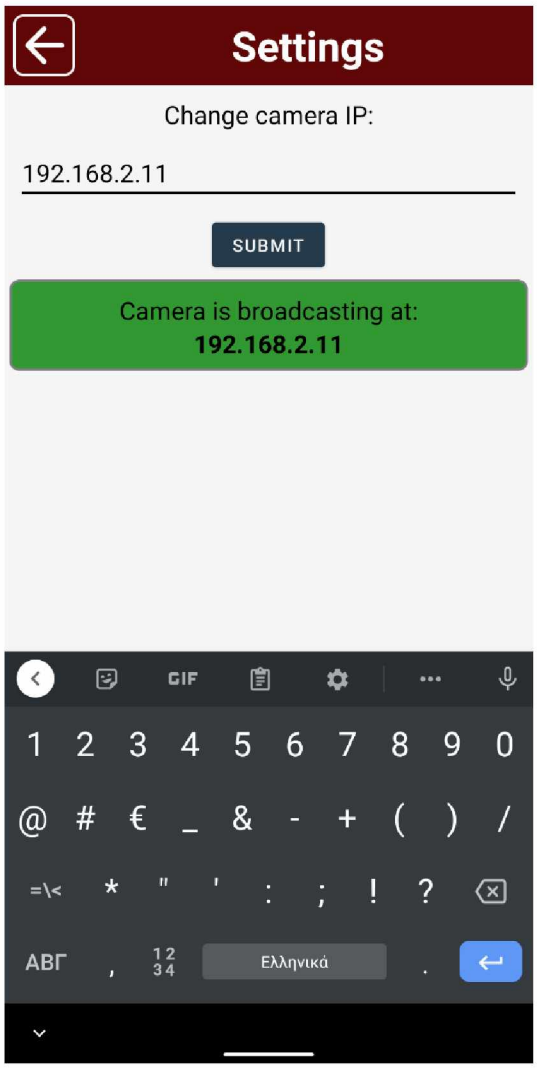
Έτσι, στο activity (οθόνη) της κάμερας χρησιμοποιείται ένα αντικείμενο WebView. Ένα αντικείμενο WebView επιτρέπει σε έναν προγραμματιστή να προβάλλει περιεχόμενο ιστού ως μέρος της διάταξης της οθόνης, αλλά τα αντικείμενα αυτά δεν διαθέτουν ορισμένες από τις δυνατότητες των πλήρως αναπτυγμένων προγραμμάτων περιήγησης. Έτσι, από το activity αυτό της κάμερας, φαίνεται η ζωντανή ροή της IP κάμερας και ο χρήστης έχει πρόσβαση σε ορισμένες λειτουργίες όπως να ανοίξει και να κλείσει τον φακό της IP κάμερας (smartphone), να κάνει autofocus, να εναλλάξει μεταξύ μπροστινής και οπίσθιας κάμερας του κινητού, να κάνει zoom και να ρυθμίσει τη φωτεινότητα. Επίσης, αν για οποιοδήποτε λόγο δεν υπάρχει πρόσβαση εμφανίζεται το μήνυμα σφάλματος (Εικόνα 7.9).



Εικόνα 7.9: (Αριστερά)Απεικόνιση του activity της κάμερας επιτυχής σύνδεση και λειτουργία. (Δεξιά) Απεικόνιση του activity της κάμερας με αποτυχημένη σύνδεση.

Επειδή μέσα από την εφαρμογή που χρησιμοποιήθηκε ώστε να μετατρέπεται ένα κινητό σε IP κάμερα, ο χρήστης δεν έχει τη δυνατότητα να επιλέξει τη διεύθυνση που θα προβάλλεται η ζωντανή ροή της κάμερας και δεν είναι στατική, μέσα από τις ρυθμίσεις της εφαρμογής Smart Home μπορεί να γράφει χειροκίνητα η διεύθυνση που καθορίζεται από την εφαρμογή «IP Phone Camera». Έτσι, ο χρήστης στο smartphone που θα έχει το ρόλο της IP κάμερας κατεβάζει την εφαρμογή «IP Phone Camera» από το Play Store. Την ανοίγει και ξεκινά τη ζωντανή ροή. Το URL που χρησιμοποιεί η IP κάμερα είναι για παράδειγμα το https://192.168.2.11:6677 . Έτσι ο χρήστης μέσα από τις ρυθμίσεις πληκτρολογεί το URL (192.168.2.11)και η κάμερα είναι έτοιμη για χρήση (Εικόνα 7.10).

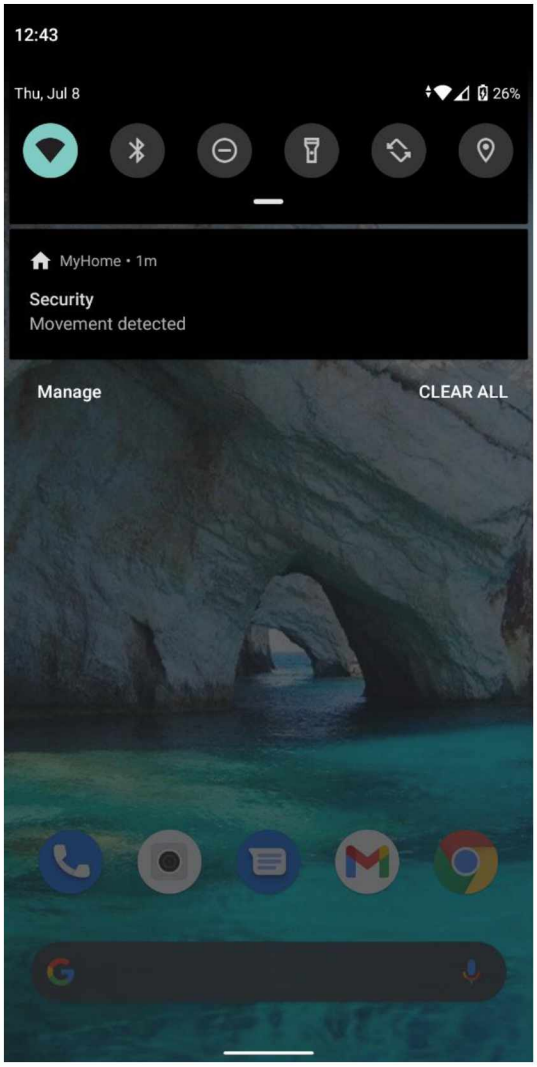
Επίσης εμφανίζεται ένα μικρό αναδυόμενο μήνυμα για την επιτυχή αλλαγή διεύθυνσης (toast).



Εικόνα 7.10: Απεικόνιση της οθόνης options. Αλλαγή IP της κάμερας.

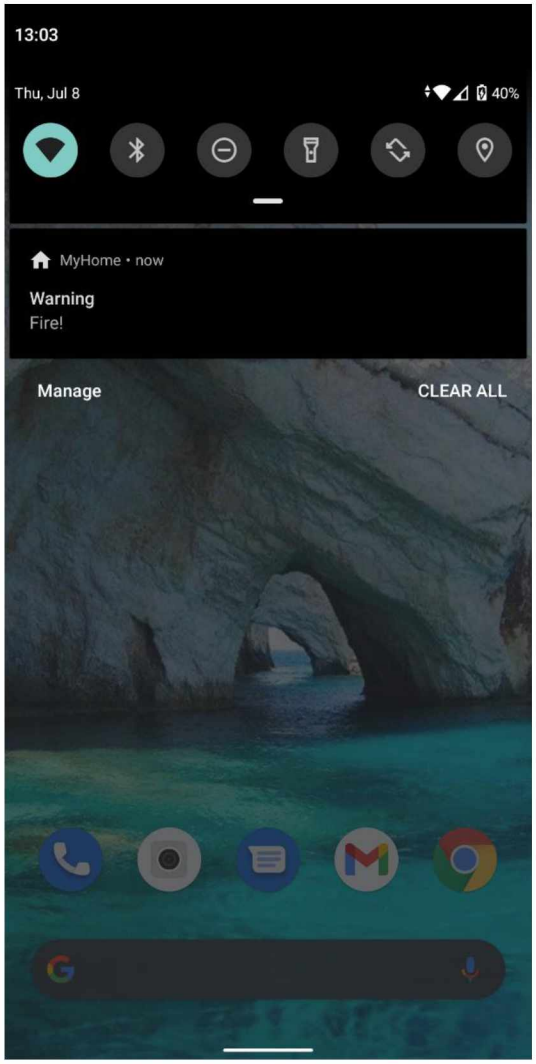
7.2 Ειδοποιήσεις - Notifications

Για την εφαρμογή υλοποιήθηκε και ένα σύστημα ειδοποιήσεων. Όταν κάποιος αισθητήρας ενεργοποιηθεί τότε, μέσα σε ένα μικρό χρονικό διάστημα ο χρήστης θα ενημερωθεί με μια αντίστοιχη ειδοποίηση. Για παράδειγμα, αν ο χρήστης έχει χρησιμοποιήσει τη λειτουργία Lock από την κεντρική οθόνη και έχει ενεργοποιηθεί ο αισθητήρας κίνησης, αν ανιχνευθεί κάποια κίνηση τότε ο χρήστης θα λάβει ειδοποίηση (Εικόνα 7.11). Αν ο χρήστης επιλέξει στην οθόνη την ειδοποίηση τότε αυτόματα θα ανοίξει η οθόνη της κάμερας προκειμένου ο χρήστης να μπορεί να δει τη ζωντανή ροή άμεσα χωρίς να χάσει χρόνο.



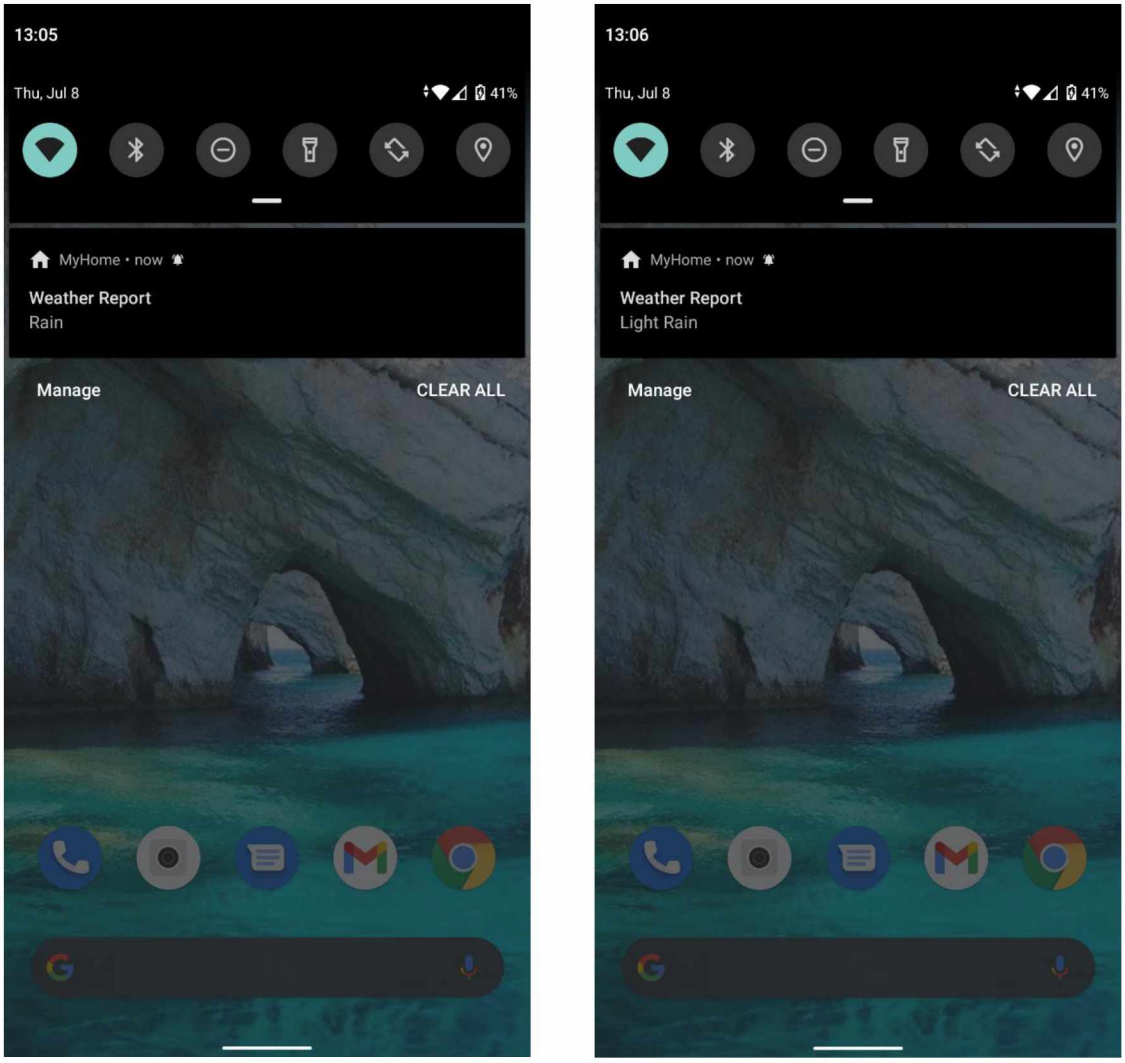
Εικόνα 7.11: Ειδοποίηση ώστε ο χρήστης να ενημερωθεί ότι ο αισθητήρας HC-SR501 ανίχνευσε κίνηση.

Αντίστοιχα υπάρχουν ειδοποιήσεις και για τους αισθητήρες φωτιάς ,βροχής ή ελαφριάς βροχής όπως φαίνεται στις εικόνες 7.12 και 7.13. Όλες οι ειδοποιήσεις για όλους τους αισθητήρες έχουν προγραμματιστεί ώστε να μην είναι συνεχείς. Για τον αισθητήρα φωτιάς, επειδή πρόκειται για την ασφάλεια των ενοίκων του σπιτιού, σχεδιάστηκε ώστε οι ειδοποιήσεις να είναι συχνές. Για τον αισθητήρα βροχής οι ειδοποιήσεις δεν είναι συχνές, θα περάσουν αρκετές ώρες μέχρι την επόμενη ειδοποίηση αν χρειαστεί (αν για παράδειγμα συνεχίζει και βρέχει). Ο αισθητήρας κίνησης, είναι ρυθμισμένος μέσω του αντίστοιχου ποτενσιόμετρου που βρίσκεται στο πίσω μέρος του να παραμένει ενεργοποιημένος (σήμα HIGH) για κάποια δευτερόλεπτα. Μόλις επέλθει σε κατάσταση ηρεμίας (σήμα LOW) είναι πάλι έτοιμος για χρήση και θα σταλεί ειδοποίηση.



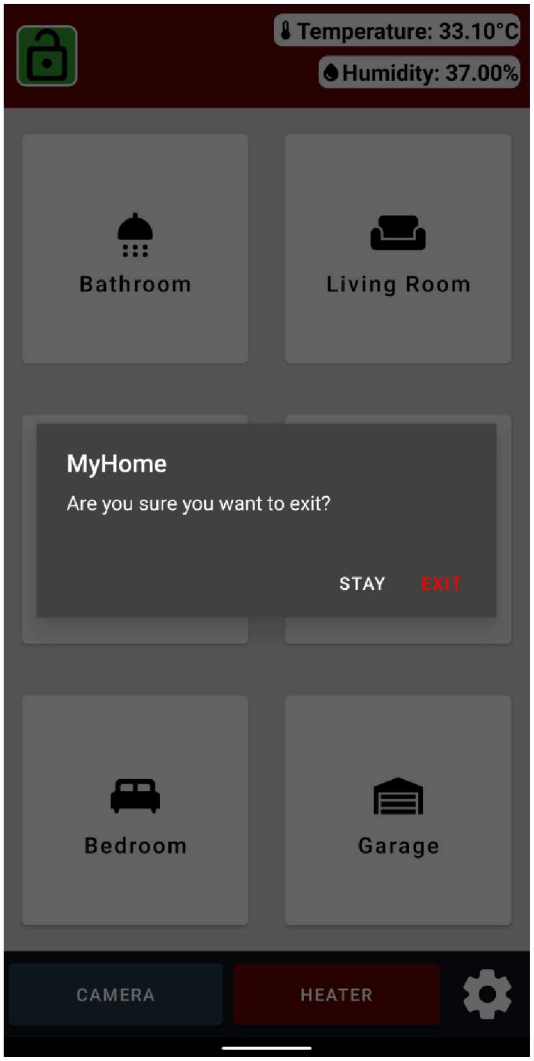
Εικόνα 7.12: Ειδοποίηση για φωτιά.

Η εφαρμογή δεν χρειάζεται να είναι σε πρώτο πλάνο ώστε ο χρήστης να λάβει ειδοποιήσεις. Αρκεί να δουλεύει στο παρασκήνιο και οι ειδοποιήσεις θα στέλνονται επιτυχώς. Επιπλέον τα notifications θα εμφανιστούν στον χρήστη ακόμα και αν το κινητό είναι κλειδωμένο, προκειμένου να ενημερωθεί έγκαιρα για την κατάσταση του περιβάλλοντος που διαμένει.



Εικόνα 7.13: (Αριστερά)Ειδοποίηση βροχής. (Δεξιά)Ειδοποίηση ελαφριάς βροχής.

Τέλος, ο χρήστης μπορεί να μεταφέρει την εφαρμογή στο παρασκήνιο χρησιμοποιώντας τα πλήκτρα περιήγησης Android που βρίσκονται στο κάτω μέρος της οθόνης του κινητού τηλεφώνου. Μπορεί επίσης όταν βρίσκεται στην αρχική οθόνη να επιλέξει να μεταφερθεί πίσω (πλήκτρο Go Back) και να επιλέξει «Exit» στο alert dialog που θα εμφανιστεί ή να διαλέξει την επιλογή «Stay» για να παραμείνει στο κεντρικό μενού της εφαρμογής και να συνεχίσει την περιήγηση (Εικόνα 7.14).



Εικόνα 7.14: Απεικόνιση alert dialog της εφαρμογής για έξοδο (ελαχιστοποίηση).

ΚΕΦΑΛΑΙΟ 8

GITHUB

Για τις ανάγκες της διπλωματικής εργασίας ο κώδικας της Android εφαρμογής και του μικροελεγκτή Arduino είναι διαθέσιμοι στο το GitHub Repository στην ηλεκτρονική διεύθυνση https://github.com/kokatsaros/Smart_Home_App-Thesis_UTH.

ΚΕΦΑΛΑΙΟ 9

ΕΠΙΛΟΓΟΣ

Υλοποιώντας την συγκεκριμένη διπλωματική έγινε μια εισαγωγή στον γοργά αναπτυσσόμενο κόσμο των έξυπνων σπιτιών καθώς και στον κόσμο της δημιουργίας μια εφαρμογής Android. Ολοκληρώνοντας την εργασία, ο αρχικός στόχος επετεύχθη δηλαδή να σχεδιαστεί και να δημιουργηθεί μια Android εφαρμογή για ένα έξυπνο σπίτι και σε αυτή να προστεθούν κάποιες λειτουργίες ενός ολοκληρωμένου πακέτου έξυπνου σπιτιού όπως αυτά παρουσιάστηκαν στο κεφάλαιο 2.

Τελικά αποδεικνύεται πως με μικρό κόστος σε ότι αφορά το hardware και χρήση εργαλείων open-source όπως το Arduino IDE ή για την ανάπτυξη εφαρμογών το Android Studio, είναι δυνατόν να δημιουργηθεί ένα σύστημα το οποίο θα κάνει έστω και λίγο καλύτερη και ευκολότερη τη ζωή των ενοίκων ενός σπιτιού. Στα θετικά είναι ότι ένα τέτοιο project επιδέχεται πολλές επεκτάσεις. Από την άλλη, απαιτείται χρόνος για την υλοποίηση ενός τέτοιου project, γνώσεις προγραμματισμού καθώς και εξοικείωση με λογισμικά όπως το Arduino IDE και το Android Studio.

Μελλοντικά θα μπορούσε η εφαρμογή να επεκταθεί και να υπάρχει αμφίδρομη επικοινωνία μεταξύ Android εφαρμογής και Arduino. Οι εντολές που σχεδιάστηκαν σαν simulation όπως άνοιγμα φώτων, ενεργοποίηση/απενεργοποίηση μιας πρίζας, άνοιγμα γκαραζόπορτας να λειτουργούν με κάποιες αλλαγές στον κώδικα του Arduino και του Android application καθώς και τη χρήση ρελέ. Στη συνέχεια μπορούν να δημιουργηθούν αυτοματισμοί και σκηνές όπως παρουσιάστηκαν στο κεφάλαιο 2 από τον ίδιο τον χρήστη. Να εμπλουτιστεί η εφαρμογή με περισσότερες προσωποποιημένες λειτουργίες, όπως η δημιουργία των δωματίων του σπιτιού.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] V. Ricquebourg, D. Menga, D. Durand, B. Marhic, L. Delahoche and C. Loge, "The Smart Home Concept : our immediate future," in *2006 1ST IEEE International Conference on E-Learning in Industrial Electronics*, 2006.

[2] S. Suresh and P. V. Sruthi, "A review on smart home technology," in *2015 Online International Conference on Green Engineering and Technologies (IC-GET)*, 2015.

[3] J. Bennett, O. Rokas and L. Chen, "Healthcare in the Smart Home: A Study of Past, Present and Future," *Sustainability,* vol. 9, 2017.

[4] "Arduino," [Online]. Available: https://www.arduino.cc/en/Guide/Introduction. [Accessed 20 6 2021].

[5] "Arduino Uno Rev3," [Online]. Available: https://store.arduino.cc/arduino-uno-rev3. [Accessed 20 6 2021].

[6] "Arduino Memory," [Online]. Available: https://www.arduino.cc/en/Tutorial/Foundations/Memory. [Accessed 20 6 2021].

[7] "Ethernet Shield," [Online]. Available: https://store.arduino.cc/arduino-ethernet-shield-2. [Accessed 20 6 2021].

[8] "DHT11 Sensor," [Online]. Available: https://components101.com/sensors/dht11-temperature-sensor. [Accessed 20 6 2021].

[9] "Flame Sensor," [Online]. Available: https://arduinomodules.info/ky-026-flame-sensor-module/. [Accessed 20 6 2021].

[10] "Rain Sensor," [Online]. Available: https://components101.com/sensors/rain-drop-sensor-module. [Accessed 21 6 2021].

[11] "PIR Motion Sensor," [Online]. Available: https://components101.com/sensors/hc-sr501-pir-sensor. [Accessed 20 5 2021].

[12] "Arduino Webserver," [Online]. Available: https://www.arduino.cc/en/Tutorial/LibraryExamples/WebServer. [Accessed 20 6 2021].

[13] "Map function Arduino," [Online]. Available: https://www.arduino.cc/reference/en/language/functions/math/map/. [Accessed 21 6 2021].

[14] M. Gargenta, Learning Android, O'Reilly Media, Inc., 2011.

[15] "Android Emulator," [Online]. Available: https://developer.android.com/studio/run/emulator. [Accessed 27 6 2021].

[16] "Setup android device," [Online]. Available: https://developer.android.com/studio/run/device. [Accessed 25 5 2021].

[17] "Activities," [Online]. Available: https://developer.android.com/guide/components/activities/intro-activities#ctm. [Accessed 25 5 2021].

[18] "Layouts-XML," [Online]. Available: https://developer.android.com/guide/topics/ui/declaring-layout. [Accessed 28 6 2021].

[19] "Shared Preferences," [Online]. Available: https://developer.android.com/reference/android/content/SharedPreferences. [Accessed 6 7 2021].

[20] "Handler," [Online]. Available: https://developer.android.com/reference/android/os/Handler. [Accessed 10 7 2021].

[21] "MainThread," [Online]. Available: https://www.androiddesignpatterns.com/2012/06/app-force-close-honeycomb-ics.html. [Accessed 10 7 2021].

[22] "NetworkOperations," [Online]. Available: https://developer.android.com/training/basics/network-ops. [Accessed 10 7 2021].

[23] "Dialogs," [Online]. Available: https://developer.android.com/guide/topics/ui/dialogs. [Accessed 7 7 2021].

[24] "Toast," [Online]. Available: https://developer.android.com/guide/topics/ui/notifiers/toasts. [Accessed 9 7 2021].

[25] "Permissions," [Online]. Available: https://developer.android.com/training/permissions/requesting. [Accessed 10 7 2021].

[26] "Custom Switch," [Online]. Available: https://github.com/zcweng/SwitchButton/blob/master/LICENSE. [Accessed 10 4 2021].

[27] "Croller-Circular Seekbar," [Online]. Available: https://github.com/harjot-oberai/Croller. [Accessed 10 4 2021].

[28] "IP Phone Camera," [Online]. Available: https://play.google.com/store/apps/details?id=com.ipphonecamera&hl=el&gl=US. [Accessed 9 7 2021].

ΠΑΡΑΡΤΗΜΑ

ΚΩΔΙΚΑΣ ANDROID STUDIO

Α) Η μέθοδος sensor_val(), με την οποία λαμβάνονται οι τιμές των αισθητήρων.

```
public void sensor_val (){


    new Thread(new Runnable() {
        public void run(){
            String urela1 = "http://192.168.2.40";
            URL url = null;
            try {
                url = new URL(urela1);
            } catch (MalformedURLException e) {
                e.printStackTrace();
            }
            BufferedReader in1 = null;
            try {
                in1 = new BufferedReader(
                        new InputStreamReader(
                                url.openStream()));
            } catch (IOException e) {
                e.printStackTrace();
            }

            if (in1 != null) {

                try {
                    fullString = in1.readLine();

                } catch (IOException e) {
                    e.printStackTrace();
                }

                try {
                    in1.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
```

```
runOnUiThread (new Runnable() {
    public void run(){
        // LogTable[0] -> temperature
        // LogTable[1] -> humidity
        // LogTable[2] -> fire sensor
        // LogTable[3] -> rain sensor
        // LogTable[4] -> motion sensor
        LogTable = fullString.split( regex: " ");
        if (LogTable.length>1) {
            //Jsontxt.setText(fullString);

            temperature.setText("Temperature: "+LogTable[0]+"°C");

            humid.setText("Humidity: "+LogTable[1]+"%");


            // an o aisthitiras fwtias anixnefsei 0,1 tote stelnw notification warning. me to flag stamatw tin sinexomeni apostoli notifications.
            if ((parseInt(LogTable[2]) == 0 || parseInt(LogTable[2]) == 1) && !fire_notify) {
                fire_notify = true;
                createNotification( aMessage: "Warning",  bMessage: "Fire!");
                fire_handler = new Handler();
                fire_handler.postDelayed(new Runnable() {
                    public void run() {
                        fire_notify = false;
                        //Toast.makeText(MainActivity.this, "Mhdenisa fwtia", Toast.LENGTH_SHORT).show();
                    }
                }, delayMillis: 10000);

            }
            else if (parseInt(LogTable[2]) == 2) {
                fire_notify = false;
            }

            // an o aisthitiras vroxis anixnefsei vroxoptosi stelnei ta antistixa notifications. Ana 3 wres stelnei notification(an vrexei)
            if (parseInt(LogTable[3]) == 0 && !rain_notify) {
                rain_notify = true;
                createNotification( aMessage: "Weather Report",  bMessage: "Rain");
                rain_handler = new Handler();
                rain_handler.postDelayed(new Runnable() {
                    public void run() {
                        fire_notify = false;
                        Toast.makeText( context: MainActivity.this,  text: "Mhdenisa vroxi", Toast.LENGTH_SHORT).show();
                    }
                }, delayMillis: 10800000);
```

```
                    }
                    else if (parseInt(LogTable[3]) == 1 && !rain_notify ) {
                        rain_notify = true;
                        rain_handler2 = new Handler();
                        rain_handler2.postDelayed(new Runnable() {
                            public void run() {
                                fire_notify = false;
                                //Toast.makeText(MainActivity.this, "Mhdenisa vroxi", Toast.LENGTH_SHORT).show();
                            }
                        }, delayMillis: 10800000);
                        createNotification( aMessage: "Weather Report", bMessage: "Light Rain");
                    }
                    else if (parseInt(LogTable[3]) == 2 ){
                        rain_notify = false; // epanafora tou flag se false otan stamatisei i vroxi wsta na mporw na to xrisimopoihsw ameswss.
                        //.makeText(MainActivity.this, "Mhdenisa vroxi", Toast.LENGTH_SHORT).show();
                    }



                    /// PIR Motion sensor ///
                    /// If house is locked and movement detected ///
                    if ((parseInt(LogTable[4]) == 1) && (lock == true)){
                        createNotification( aMessage: "Security", bMessage: "Movement detected");
                        Intent intent = new Intent( packageContext: MainActivity.this,camera.class);
                        startActivity(intent);
                    }

                }

            }});


        }
    }).start();



}
```

Β) Η μέθοδος createNotification(), με την οποία δημιουργούνται οι ειδοποιήσεις.

```
private NotificationManager notifManager;
public void createNotification(String aMessage, String bMessage) {
    final int NOTIFY_ID = 1002;

    // There are hardcoding only for show it's just strings
    String name = "my_package_channel";
    String id = "my_package_channel_1"; // The user-visible name of the channel.
    String description = "my_package_first_channel"; // The user-visible description of the channel.


    if (bMessage.equals("Movement detected")){
        Intent intent;
        PendingIntent pendingIntent;
        NotificationCompat.Builder builder;
        if (notifManager == null) {
            notifManager =
                    (NotificationManager)getSystemService(Context.NOTIFICATION_SERVICE);
        }

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            int importance = NotificationManager.IMPORTANCE_HIGH;
            NotificationChannel mChannel = notifManager.getNotificationChannel(id);
            if (mChannel == null) {
                mChannel = new NotificationChannel(id, name, importance);
                mChannel.setDescription(description);
                mChannel.enableVibration(true);
                mChannel.setLightColor(Color.GREEN);
                mChannel.setVibrationPattern(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
                notifManager.createNotificationChannel(mChannel);
            }
            builder = new NotificationCompat.Builder( context: this, id);

            intent = new Intent( packageContext: this, camera.class);
            intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_SINGLE_TOP);
            pendingIntent = PendingIntent.getActivity( context: this,  requestCode: 0, intent,  flags: 0);

            builder.setContentTitle(aMessage)  // required
                    .setSmallIcon(R.drawable.notification) // required
                    .setContentText(bMessage)  // required
                    .setDefaults(Notification.DEFAULT_ALL)
                    .setAutoCancel(true)
                    .setContentIntent(pendingIntent)
                    .setTicker(aMessage)
                    .setVibrate(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
```

```
    } else {

        builder = new NotificationCompat.Builder( context: this);

        intent = new Intent( packageContext: this, MainActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_SINGLE_TOP);
        pendingIntent = PendingIntent.getActivity( context: this,  requestCode: 0, intent,  flags: 0);

        builder.setContentTitle(aMessage)
                .setSmallIcon(R.drawable.notification) // required
                .setContentText(bMessage)  // required
                .setDefaults(Notification.DEFAULT_ALL)
                .setAutoCancel(true)
                .setContentIntent(pendingIntent)
                .setTicker(aMessage)
                .setVibrate(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400})
                .setPriority(Notification.PRIORITY_HIGH);
    } // else if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {

    Notification notification = builder.build();
    notifManager.notify(NOTIFY_ID, notification);

}else {
    Intent intent;
    PendingIntent pendingIntent;
    NotificationCompat.Builder builder;
    if (notifManager == null) {
        notifManager =
                (NotificationManager)getSystemService(Context.NOTIFICATION_SERVICE);
    }

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
        int importance = NotificationManager.IMPORTANCE_HIGH;
        NotificationChannel mChannel = notifManager.getNotificationChannel(id);
        if (mChannel == null) {
            mChannel = new NotificationChannel(id, name, importance);
            mChannel.setDescription(description);
            mChannel.enableVibration(true);
            mChannel.setLightColor(Color.GREEN);
            mChannel.setVibrationPattern(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
            notifManager.createNotificationChannel(mChannel);
        }
        builder = new NotificationCompat.Builder( context: this, id);
```

```
            intent = new Intent( packageContext: this, MainActivity.class);
            intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_SINGLE_TOP);
            pendingIntent = PendingIntent.getActivity( context: this,  requestCode: 0, intent,  flags: 0);

            builder.setContentTitle(aMessage)  // required
                    .setSmallIcon(R.drawable.notification) // required
                    .setContentText(bMessage)  // required
                    .setDefaults(Notification.DEFAULT_ALL)
                    .setAutoCancel(true)
                    .setContentIntent(pendingIntent)
                    .setTicker(aMessage)
                    .setVibrate(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
        } else {

            builder = new NotificationCompat.Builder( context: this);

            intent = new Intent( packageContext: this, MainActivity.class);
            intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_SINGLE_TOP);
            pendingIntent = PendingIntent.getActivity( context: this,  requestCode: 0, intent,  flags: 0);

            builder.setContentTitle(aMessage)
                    .setSmallIcon(R.drawable.notification) // required
                    .setContentText(bMessage)  // required
                    .setDefaults(Notification.DEFAULT_ALL)
                    .setAutoCancel(true)
                    .setContentIntent(pendingIntent)
                    .setTicker(aMessage)
                    .setVibrate(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400})
                    .setPriority(Notification.PRIORITY_HIGH);
        } // else if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {

        Notification notification = builder.build();
        notifManager.notify(NOTIFY_ID, notification);

    }

}
```

Γ) Η κλάση Java των ρυθμίσεων της εφαρμογής (options).

```java
package com.example.urltostringtest;

import ...

public class Settings extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Objects.requireNonNull(getDelegate().getSupportActionBar()).hide(); /* Hide action bar */
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,WindowManager.LayoutParams.FLAG_FULLSCREEN); /* Enable FULLSCREEN */
        setContentView(R.layout.activity_settings);

        TextView textView = (TextView) findViewById(R.id.defaultip);
        final EditText editText = (EditText) findViewById(R.id.edittext_ip);

        // keeping the value of textview
        SharedPreferences sharedPreferences = getSharedPreferences( name: "cameraIP", MODE_PRIVATE);
        String camIP = sharedPreferences.getString( key: "value", defValue: "");
        textView.setText(camIP);

        // when submit button is pressed ...
        Button Submit = (Button) findViewById(R.id.submit_IP);
        Submit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String value = editText.getText().toString().trim();
                SharedPreferences sharedPref = getSharedPreferences( name: "cameraIP", MODE_PRIVATE);
                SharedPreferences.Editor editor = sharedPref.edit();
                editor.putString("value", value);
                editor.apply();
                textView.setText(value);
                Toast.makeText( context: Settings.this, text: "Camera IP changed", Toast.LENGTH_SHORT).show();
            }
        });
```

```
        // when back button is pressed ...
        ImageButton back = (ImageButton) findViewById(R.id.settings_back);
        back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {finish();}
        });

    }
}
```

Δ) Η κλάση Java της κεντρικής θέρμανσης (heater).

```java
package com.example.urltostringtest;

import ...

public class heater extends AppCompatActivity {

    private TextView textView;
    private SwitchButton switchButton;
    private SeekBar seekBar;
    private Croller croller;

    public boolean heater = false;
    public static final String SHARED_PREFS = "sharedPrefs";
    public static final String TEXT = "text";
    public static final String SWITCH = "switch";

    private int seekBarProgress;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Objects.requireNonNull(getDelegate().getSupportActionBar()).hide(); /* Hide action bar */
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN); /* Enable FULLSCREEN */
        setContentView(R.layout.activity_heater);


        com.suke.widget.SwitchButton switchButton = (com.suke.widget.SwitchButton) findViewById(R.id.heater_onoff);
        TextView textView = (TextView) findViewById(R.id.heatertemp);
        Croller croller = (Croller) findViewById(R.id.croller);
        croller.setMax(35);
        croller.setMin(1);
        croller.setStartOffset(45);
```

```
croller.setOnCrollerChangeListener(new OnCrollerChangeListener() {
    @Override
    public void onProgressChanged(Croller croller, int progress) {
        if (progress>= 16) {
            textView.setText(progress + "°C");

        }
        else {
            textView.setText("Off");
        }
        SharedPreferences.Editor editor = getSharedPreferences( name: "save", MODE_PRIVATE).edit();
        editor.putInt("croller",croller.getProgress());
        editor.apply();
    }

    @Override
    public void onStartTrackingTouch(Croller croller) {

    }

    @Override
    public void onStopTrackingTouch(Croller croller) {

    }
});



SharedPreferences sharedPreferences = getSharedPreferences( name: "save",MODE_PRIVATE);
switchButton.setChecked(sharedPreferences.getBoolean( key: "value", defValue: false));
```

```
        switchButton.setOnCheckedChangeListener(new SwitchButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(SwitchButton view, boolean isChecked) {
                if (switchButton.isChecked()) {
                    // when switch checked
                    //save prefs for switch
                    SharedPreferences.Editor editor = getSharedPreferences( name: "save", MODE_PRIVATE).edit();
                    editor.putBoolean("value", true);
                    editor.apply();
                    switchButton.setChecked(true);
                } else {
                    //when switch unchecked
                    //save prefs for switch
                    SharedPreferences.Editor editor = getSharedPreferences( name: "save", MODE_PRIVATE).edit();
                    editor.putBoolean("value", false);
                    editor.apply();
                    switchButton.setChecked(false);
                }
            }
        });


        ImageButton back = (ImageButton) findViewById(R.id.heater_back);
        back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { finish(); }
        });

        // LOAD KAI UPDATE DATA PREFERENCES
        seekBarProgress = sharedPreferences.getInt( key: "croller", defValue: 0);
        croller.setProgress(seekBarProgress);
        textView.setText(croller +"°C");

    }
}
```

Ε) Η κλάση Java της IP κάμερας της εφαρμογής (Camera).

```java
package com.example.urltostringtest;

import ...

public class camera extends AppCompatActivity {

    private WebView webView;

    @SuppressLint("SetJavaScriptEnabled")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_camera);
        Objects.requireNonNull(getDelegate().getSupportActionBar()).hide(); /* Hide action bar */
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN); /* Enable FULLSCREEN */

        webView = (WebView) findViewById(R.id.webview);
        webView.setWebViewClient(new WebViewClient());
        SharedPreferences sharedPreferences = getSharedPreferences( name: "cameraIP", MODE_PRIVATE);
        String camIP = sharedPreferences.getString( key: "value",  defValue: "");
        webView.loadUrl("http://" + camIP + ":6677/");
        WebSettings webSettings = webView.getSettings();
        webSettings.setJavaScriptEnabled(true);


        ImageButton back = (ImageButton) findViewById(R.id.camera_back);
        back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { finish(); }
        });

    }

    public void onBackPressed() {
        if (webView.canGoBack()) {
            webView.goBack();
        } else {
            super.onBackPressed();
        }
    }

}
```