



Πανεπιστήμιο Πελοποννήσου

Τμήμα Ψηφιακών Συστημάτων

**Σχολή Οικονομίας και Τεχνολογίας  
Τμήμα Ψηφιακών Συστημάτων**

**ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ**

*«Συστήματα ασφαλείας IoT στην θεωρία και στην πράξη»*

**ΓΙΩΡΓΟΣ ΜΑΝΤΑΣΑΣ**

**ΑΜ:2017062**

**Επιβλέπων : Παναγιώτης Κόκκινος**

**ΣΠΑΡΤΗ**

**2023**

## ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία αναφέρεται στα συστήματα ασφαλείας με βασικό γνώμονα το διαδίκτυο των πραγμάτων IoT Internet of Things, με έμφαση στην θεωρία και στην πράξη. Περιλαμβάνει τρία κεφάλαια, με το πρώτο να περιγράφει το θέμα αναφερόμενο στο πρώτο σύστημα ασφαλείας που κατασκευάστηκε ποτέ και αναλύει τα κίνητρα και τα οφέλη των εξ χρονισμένων συστημάτων ασφαλείας IoT. Στο δεύτερο κεφάλαιο, εξετάζονται οι λειτουργίες και οι διασυνδέσεις των συστημάτων ασφαλείας IoT, με εστίαση στην αρχιτεκτονική τους με παράδειγμα τα συστήματα της κατασκευάστρια εταιρία Ajax, καθώς και στο λογισμικό που χρησιμοποιεί με τα σενάρια αυτοματισμού. Επίσης περιέχονται διαγράμματα ροής και μπλοκ για την καλύτερη κατανόηση του τρόπου λειτουργίας των συστημάτων ασφαλείας βασιζόμενα στα συστήματα Ajax. Το τρίτο κεφάλαιο παρουσιάζει την υλοποίηση ενός συστήματος ασφαλείας IoT χρησιμοποιώντας την πλακέτα μικροηλεκτρονικών ESP32 μαζί με διάφορους αισθητήρες. Συμπεριλαμβάνονται προτάσεις για την υλοποίησης ενός τέτοιου συστήματος, ο τρόπος λειτουργίας του IoT συστήματος ασφαλείας που κατασκευάστηκε, το κόστος και τα υλικά που χρησιμοποιήθηκαν, τα προγράμματα που βοήθησαν στην υλοποίηση του, ο τρόπο κατασκευής του κυκλώματος από την αρχή μέχρι το τέλος, το λογισμικό και ο κώδικας που κατασκευάστηκε, τα προβλημάτων και οι περιορισμών που προέκυψαν κατά την υλοποίηση, καθώς και τα τελικά συμπεράσματα μιας εξολοκλήρου κατασκευής. Τέλος, βρίσκεται μια ενότητα με την βιβλιογραφία και άρθρα που χρησιμοποιήθηκαν κατά την διάρκεια της έρευνας και υλοποίησης της παρούσας πτυχιακής εργασίας.


## ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

"Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάση επιστημονικής παράφρασης.

Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων.

Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δε μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας."

Όνομα και Επώνυμο Συγγραφέα (Με Κεφαλαία): ΓΙΩΡΓΟΣ ΜΑΝΤΑΣΑΣ

Υπογραφή (Ολογράφως, χωρίς μονογραφή): 

Ημερομηνία (Ημέρα – Μήνας – Έτος): 25-05-2023

# ΠΕΡΙΕΧΟΜΕΝΑ

<b>ΚΕΦΑΛΕΟ 1 ΕΙΣΑΓΩΓΗ.....</b>	<b>8</b>
<b>1.1 Γενική περιγραφή .....</b>	<b>8</b>
Εικόνα 1.1.1: Πρώτο σύστημα συναγερμού .....	8
Εικόνα 1.1.2: Σύγχρονο σύστημα συναγερμού .....	9
Εικόνα 1.1.3: Διαδίκτυο των πραγμάτων IoT .....	10
<b>1.2 Κίνητρα και οφέλη .....</b>	<b>11</b>
Εικόνα 1.2.1: Διαδίκτυο των πραγμάτων IoT συναγερμός.....	12
<b>ΚΕΦΑΛΕΟ 2 ΛΕΙΤΟΥΡΓΙΕΣ ΚΑΙ ΔΙΑΣΥΝΔΕΣΕΙΣ .....</b>	<b>12</b>
<b>2.1 Αρχιτεκτονική στα IoT συστήματα ασφαλείας.....</b>	<b>12</b>
Εικόνα 2.1.1: Κεντρικός πίνακας hub .....	13
Εικόνα 2.1.2: Αισθητήρες κινήσεις .....	15
Εικόνα 2.1.3: Αισθητήρες πόρτας .....	16
Εικόνα 2.1.4: Αισθητήρες φωτιάς.....	17
Εικόνα 2.1.5: αισθητήρες ανίχνευσης διαρροής νερού .....	18
<b>2.2 Λογισμικά και σενάρια αυτοματισμού .....</b>	<b>19</b>
Εικόνα 2.2.1: Παράθυρο εφαρμογής και διαχείρισης Ajax Systems.....	20
Εικόνα 2.2.2: Ειδοποιήσεις, έλεγχος και δωμάτια εφαρμογής Ajax systems .....	21
Εικόνα 2.2.3: Σενάρια αυτοματισμού .....	23
Εικόνα 2.2.4: Πολλαπλά μέλη με περιορισμένη πρόσβαση Ajax systems.....	24
Εικόνα 2.2.5: Πρόγραμμα για έξυπνα ρολόγια .....	25

<b>2.3 Διαγράμματα ροής και διαγράμματα μπλοκ.....</b>	<b>27</b>
Διάγραμμα 2.3.1: Αρχιτεκτονικό διάγραμμα IoT συστήματος ασφαλείας .....	27
Διάγραμμα ροής 2.3.2: διάγραμμα ροής μεταφοράς πληροφοριών.....	28
Διάγραμμα ροής 2.3.3: διάγραμμα ροής σεναρίου αισθητήρα.....	29
Διάγραμμα ροής 2.3.4: διάγραμμα ροής σεναρίου αισθητήρα πόρτας.....	29
Διάγραμμα ροής 2.3.5: διάγραμμα ροής σεναρίου αισθητήρα διαρροής .....	30
Διάγραμμα ροής 2.3.6: διάγραμμα ροής σεναρίου σύστημα πυρανίχνευσης.....	31
Διάγραμμα μπλοκ 2.3.7: διάγραμμα μπλοκ αισθητήρας κίνησης .....	32
Διάγραμμα μπλοκ 2.3.8: διάγραμμα μπλοκ αισθητήρας πόρτας .....	32
Διάγραμμα μπλοκ 2.3.9: διάγραμμα μπλοκ αισθητήρας φωτιάς .....	33
Διάγραμμα μπλοκ 2.3.10: διάγραμμα μπλοκ αισθητήρας διαρροής νερού.....	33
Διάγραμμα μπλοκ 2.3.11: διάγραμμα μπλοκ κεντρικού Hub .....	34
<b>ΚΕΦΑΛΕΟ 3 ΥΛΟΠΟΙΗΣΗ.....</b>	<b>35</b>
<b>3.1 Προτάσεις για υλοποίηση .....</b>	<b>35</b>
<b>3.2 Υλοποίηση συστήματος ασφαλείας IoT .....</b>	<b>36</b>
Εικόνα 3.2.1: Έξυπνο σύστημα ασφαλείας σε Arduino .....	36
<b>3.3 Τρόπος λειτουργίας.....</b>	<b>38</b>
Εικόνα 3.3.1: IoT σύστημα ασφαλείας, διαδικτυακή εφαρμογή σε έξυπνο τηλέφωνο .....	38
Εικόνα 3.3.2: IoT σύστημα ασφαλείας, διαδικτυακή εφαρμογή σε υπολογιστή .....	39
Εικόνα 3.3.3: IoT σύστημα ασφαλείας, email ειδοποιήσεις. ....	40
Εικόνα 3.3.4: IoT σύστημα ασφαλείας, συνδεδεμένο κύκλωμα. ....	42
Διάγραμμα στοιχείων 3.3.5: IoT σύστημα ασφαλείας, διάγραμμα στοιχείων. ....	43
<b>3.4 Υλικά και κόστος κατασκευής.....</b>	<b>43</b>

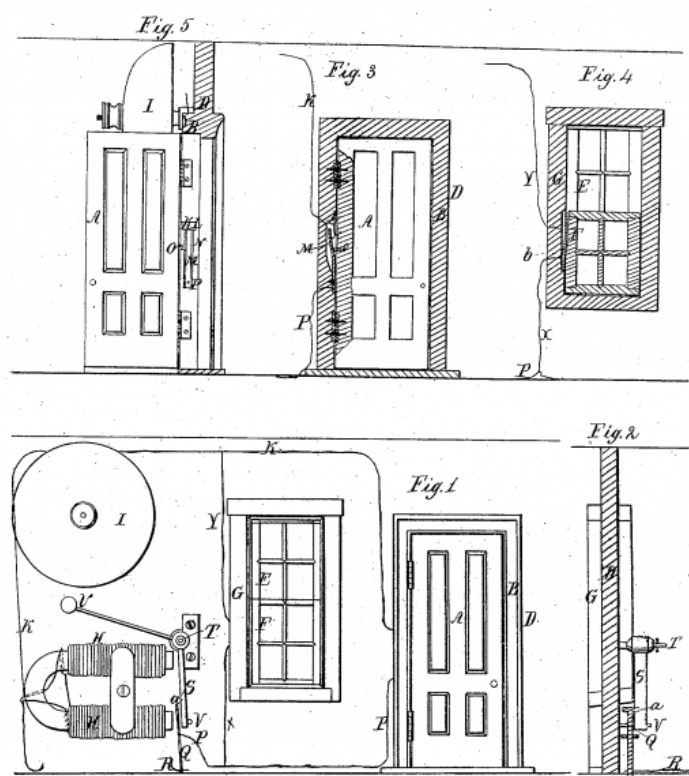
Εικόνα 3.4.1: Υλικά κατασκευής.....	45
<b>3.5 Προγράμματα που χρησιμοποιήθηκαν.....</b>	<b>46</b>
Εικόνα 3.5.1: Google Drive.....	47
Εικόνα 3.5.2: Trello .....	48
Εικόνα 3.5.3: Tinkercad.....	49
Εικόνα 3.5.4: Arduino IoT Cloud .....	50
Εικόνα 3.5.5: Arduino IDE .....	51
Εικόνα 3.5.6: Arduino Client Agent.....	52
Εικόνα 3.5.7: Notepad++ .....	52
Εικόνα 3.5.8: IFTTT .....	53
<b>3.6 Τρόπος κατασκευής κυκλώματος.....</b>	<b>54</b>
Εικόνα 3.6.1: Αρχικό σχέδιο Tinkercad.....	55
Εικόνα 3.6.2: Τελική μορφή κυκλώματος.....	56
<b>3.7 Λογισμικό και κώδικας .....</b>	<b>56</b>
Εικόνα 3.7.1: Arduino IoT Cloud Select Device .....	57
Εικόνα 3.7.2: Arduino IoT Cloud Configure.....	58
Εικόνα 3.7.3: Arduino IoT Cloud Sketch .....	59
Εικόνα 3.7.4: Arduino IoT Cloud Dashboards .....	60
Εικόνα 3.7.5: Arduino IoT Cloud Cloud Variables .....	61
Εικόνα 3.7.6: IFTTT Code.....	62
Εικόνα 3.7.7: Συνάρτηση Setup().....	65
Εικόνα 3.7.8: Συνάρτηση Loop() .....	66
<b>3.8 Προβλήματα και περιορισμοί .....</b>	<b>67</b>

Εικόνα 3.8.1: Βελτιστοποίηση κώδικα.....	67
Εικόνα 3.8.2: Global και Local μεταβλητές .....	68
Εικόνα 3.8.3: Google API Services φωνητικές εντολές .....	69
<b>3.9 Τελικά συμπεράσματα.....</b>	<b>71</b>
<b>Βιβλιογραφία και άρθρα.....</b>	<b>72</b>

# ΚΕΦΑΛΕΟ 1 ΕΙΣΑΓΩΓΗ

## 1.1 Γενική περιγραφή

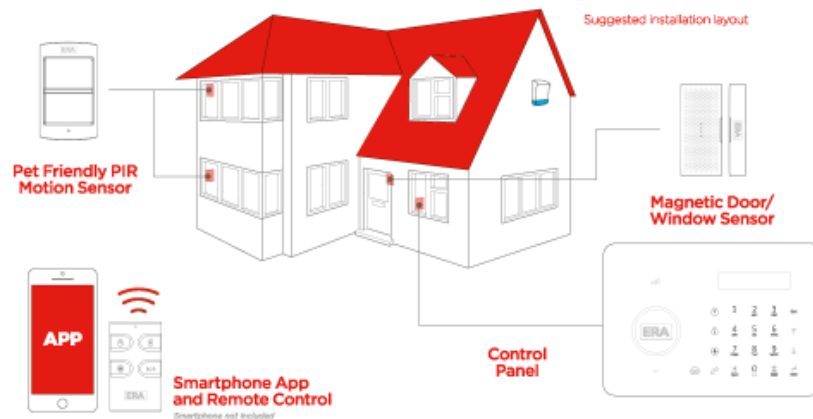
Ο συναγερμός που χρησιμοποιείτε για την αποφυγή διαρρήξεων υπάρχει πολύ περισσότερο καιρό από όσο πιστεύει ο περισσότερος κόσμος. Σύμφωνα με την διαδικτυακή ελευθέρου περιεχομένου εγκυκλοπαίδεια wikipedia.org το 1858 ο Αμερικανός επιχειρηματίας Έντουιν Χόλμς έβγαλε στην ελεύθερη αγορά το πρώτο ηλεκτρομαγνητικό σύστημα συναγερμού που μπορούσε να προσαρμοστεί σε πόρτες και παράθυρα σπιτιών.



Εικόνα 1.1.1: Πρώτο σύστημα συναγερμού



Από τότε έως σήμερα η τεχνολογία έχει εξελιχθεί ραγδαία. Σύντομα κατασκευάστηκαν έξυπνα συστήματα συναγερμού που επιτρέπουν στους χρήστες να διαχειρίζονται και να έχουν πρόσβαση σε αυτό απομακρυσμένα μέσω μιας έξυπνης συσκευής που μπορεί να είναι ένα τηλέφωνο ή ένα ρολόι. Καθώς μπορούν να καλούν βοήθεια πατώντας ένα κουμπί ή ακόμη και αυτόματα όταν αυτό είναι απαραίτητο. Επίσης οι χρήστες λαμβάνουν ιδιοποιήσεις ανάλογα με την κατάσταση του σπιτιού τους καθώς σε ένα τέτοιο σύστημα μπορούν να προσαρμοστούν αμέτρητοι αισθητήρες έχοντας την δυνατότητα να αντλούν δεδομένα από το πραγματικό περιβάλλον. Με αυτόν τον τρόπο ένα απλό σπίτι ευάλωτο σε παραβάτες μπορεί ευκολά να αναβαθμιστεί σε ένα απόρθητο φρούριο. Κάπως έτσι στους σύγχρονους συναγερμούς αφαιρέθηκαν τα πηνία και οι μεταλλικές κατασκευές που βοηθούν στην παραγωγή ηλεκτρομαγνητικών πεδίων και αντικαταστάθηκαν από τσιπάκια και επεξεργαστές που βοηθούν στην καλύτερη διαχείριση και επεξεργασία των πληροφοριών που λαμβάνονται από διάφορους αισθητήρες.



Εικόνα 1.1.2: Σύγχρονο σύστημα συναγερμού

Οι αισθητήρες που μπορούν να προσαρμοστούν σε έναν έξυπνο συναγερμό ποικίλουν ανάλογα με τις ανάγκες των χρηστών. Υπάρχουν αισθητήρες που ανιχνεύουν κινήσεις, που ελέγχουν αν μια πόρτα ή ένα παράθυρο είναι ανοικτό, που καταλαβαίνουν αν έχει σπάσει κάποιο τζάμι ή έστω να έχει ραγίσει, κάμερες που καταγράφουν όταν ενεργοποιηθεί κάποια ανεπιθύμητη ενέργεια και διάφορα άλλα. Μόλις το σύστημα ανιχνεύσει κάποια πληροφορία τότε στέλνεται ένα σήμα στο νέφος cloud και από εκεί στέλνεται ακαριαία μια ειδοποίηση στο τηλέφωνο του χρήστη με τις πληροφορίες που κατέγραψε το σύστημα συναγερμού. Έτσι ο κάτοχος έχει άμεση πρόσβαση στο σπίτι του ανεξαρτήτως της περιοχής και της χώρας που βρίσκεται την συγκεκριμένη χρονική στιγμή και μπορεί άμεσα να δράσει μέσω του έξυπνου τηλεφώνου του με τρόπο που εκείνος κρίνει πως είναι ο καταλληλότερος. Η τεχνολογία που ορίζει ένα έξυπνο σύγχρονο σύστημα ασφαλείας ονομάζεται διαδίκτυο των πραγμάτων IoT Internet of things, καθώς όλα τα υποσυστήματα του συναγερμού επικοινωνούν μεταξύ τους και είναι άμεσα συνδεδεμένα με υπηρεσίες νέφους cloud.



Εικόνα 1.1.3: Διαδίκτυο των πραγμάτων IoT

## 1.2 Κίνητρα και οφέλη

Ένα έξυπνο σύστημα συναγερμού πλέον είναι το βασικότερο μέτρο ασφαλείας που χρειάζεται να υπάρχει σε μια επιχείρηση ή σε ένα σπίτι. Σύμφωνα με μελέτη που δημοσίευσε η Αμερικάνικη ιδιωτική εταιρεία Knoema που εξειδικεύεται στην συλλογή και οπτικοποίηση επίσημων πόρων στο site knoema.com, οι καταγεγραμμένες διαρρήξεις που έγιναν στην Ελλάδα ήταν 13,892 και αυτές αφορούν μόνο το έτος 2020 και μόνο διαρρήξεις σε σπίτια. Σύμφωνα με άλλες πηγές οι διαρρήκτες επιλέγουν σπίτια με εύαλωτα χαρακτηριστικά που μπορούν να εκμεταλλευτούν. Επίσης είναι σύνηθες φαινόμενο ένα σπίτι να παραβιάζεται πολλαπλές φορές από τους ίδιους διαρρήκτες επειδή ο ιδιοκτήτης δεν αναβάθμισε την ασφάλεια του σπιτιού του. Εγκαθιστώντας ένα IoT έξυπνο σύστημα συναγερμού σε ένα σπίτι ή σε μια επιχείρηση ο ιδιοκτήτης εξασφαλίζει την ασφάλεια του και την ασφάλεια της περιουσίας του, έχει τον πλήρη έλεγχο ανά πάσα στιγμή και δεν βασίζεται απαραίτητα σε δευτερογενείς ή τριτογενείς μεθόδους ασφαλείας. Ένα ακόμα πλεονέκτημα είναι πως πάνω σε ένα έξυπνο σύστημα ασφαλείας μπορούν να προστεθούν αισθητήρες ανίχνευσης φωτιάς, αισθητήρες διαρροής νερού και επικίνδυνων αερίων όπως υγραέριο και φυσικό αέριο. Πράγματα που πλέον υπάρχουν και χρησιμοποιούνται σε κάθε σπίτι. Με κάθε ανίχνευση υπάρχει και η συγκεκριμένη ιδιοποίηση σε ένα τηλέφωνο ή ακόμη και σε ένα έξυπνο ρολόι. Με αυτό τον τρόπο οι χριστές έχουν την ευχέρεια να δημιουργήσουν το δικό τους οικοσύστημα ασφαλείας και αυτοματισμού. Ένα μοναδικό διαδικτυακό δίκτυο που όλα τα τσιπάκια και οι αισθητήρες επικοινωνούν και αλληλοεπιδρούν μεταξύ τους αυτόματα παίρνοντας προκαθορισμένες και αυτοματοποιημένες αποφάσεις για την καλύτερη διαχείριση της ασφάλειας και αποφυγής κινδύνου.



*Εικόνα 1.2.1: Διαδίκτυο των πραγμάτων IoT συναγερμός*

## **ΚΕΦΑΛΕΟ 2 ΛΕΙΤΟΥΡΓΙΕΣ ΚΑΙ ΔΙΑΣΥΝΔΕΣΕΙΣ**

### **2.1 Αρχιτεκτονική στα IoT συστήματα ασφαλείας**

Συνήθως κατά την αγορά ενός IoT έξυπνου συστήματος ασφαλείας εκτός από τον βασικό πυρήνα του που είναι το κεντρικό σημείο διασύνδεσης hub συμπεριλαμβάνεται και ένα σύνολο από αισθητήρες. Αν ο χρήστης θεωρήσει πως χρειάζεται το σύστημα του να παρέχει επιπρόσθετη ασφάλεια και λειτουργίες μπορεί να αναζητήσει για την αγορά επιπλέον αισθητήρων και εξαρτημάτων αρκεί αυτά να είναι συμβατά με το συνολικό σύστημα ασφαλείας που έχει εγκαταστήσει. Τα περισσότερα έτοιμα IoT έξυπνα συστήματα ασφαλείας που κυκλοφορούν στην αγορά αποτελούνται από έναν κεντρικό πίνακα hub και ένα σύνολο από αισθητήρες και ανιχνευτές.

Ο κεντρικός πίνακας hub είναι καρδιά ενός έξυπνου συστήματος ασφαλείας και απαιτούν σύνδεση στο δίκτυο ασύρματα ή ενσύρματα. Ο κεντρικός πίνακας κάνει πράξει την επικοινωνία με τον διακομιστή ή αλλιώς νέφος της εκάστοτε εταιρείας ώστε να διαμορφωθεί και να ελεγχθεί το σύστημα από όλα τα σημεία του κόσμου. Μέσο του νέφους cloud μεταφέρονται οι ειδοποιήσεις του συστήματος στον χρήστη και ελέγχεται αν κάποιος αισθητήρας χρειάζεται να ενημερωθεί. Τα δεδομένα και οι πληροφορίες καταγραφής μεταφέρονται στο νέφος cloud με ασφάλεια μέσω ενός κρυπτογραφημένου καναλιού χρησιμοποιώντας συνήθως κρυπτογράφηση ιδιωτικού κλειδιού.



Εικόνα 2.1.1: Κεντρικός πίνακας hub

Εκτός από τον κεντρικό πίνακα hub προσφέρονται αισθητήρες κίνησης που μερικοί από αυτούς ενσωματώνουν τεχνολογίες ανίχνευσης σπασμένου τζαμιού. Σε βασικό επίπεδο ένας αισθητήρας κίνησης λειτουργεί χρησιμοποιώντας θετική διαφορική αλλαγή. Υπάρχουν δύο υποδοχές στον αισθητήρα.

Ένας παθητικός αισθητήρα υπέρυθρων και ένας όρασης ή αλλιώς vision. Όταν δεν υπάρχει κάποια κίνηση μπροστά στον αισθητήρα οι δύο υποδοχές του ανιχνεύουν την ίδια ποσότητα υπέρυθρων.

Μόλις περάσει ένα ζεστό σώμα όπως ένας άνθρωπος ή ζώο προκαλείται μια θετική αλλαγή διαφοράς μεταξύ υποδοχών και ενεργοποιείται ένας ρελές δημιουργώντας ένα κλιστώ κύκλωμα. Αμέσως μετά μέσω του κυκλώματος στέλνετε ένα σήμα στο κύριο hub του συναγερμού και εκεί γίνεται η επεξεργασία και διαχείριση του σήματος. Όλα τα συστήματα του συναγερμού επικοινωνούν μεταξύ τους χρησιμοποιώντας ένα πρωτόκολλο επικοινωνίας που ορίζεται από την κατασκευάστρια εταιρία. Παράδειγμα τα IoT έξυπνα συστήματα ασφαλείας της εταιρίας Ajax systems χρησιμοποιούν αμφίδρομης επικοινωνίας Jeweller Radio για την μεταφορά των γεγονότων, και το πρωτόκολλο Wings που είναι υπεύθυνο για την μεταφορά οπτικού μέσου όπως η εικόνα μιας κάμερας.

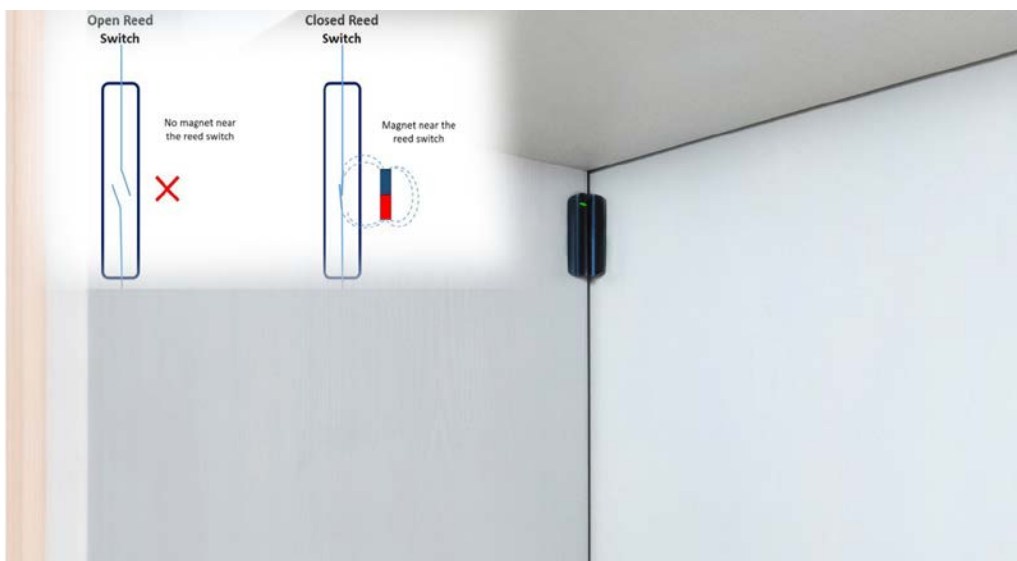
Όταν ένας αισθητήρας θέλει να επικοινωνήσει με το κεντρικό Hub τότε αποστέλλεται ένα αίτημα επικοινωνίας μέσω του πρωτοκόλλου επικοινωνίας που χρησιμοποιείται. Αν υπάρχει ανταπόκριση από το Hub τότε πραγματοποιείται η μεταφορά των δεδομένων. Με τον ίδιο τρόπο πραγματοποιείται η επικοινωνία μεταξύ του κεντρικού hub με το νέφος cloud. Τις περισσότερες φορές κάθε υποσύστημα του συστήματος ασφαλείας τροφοδοτείται ξεχωριστά με την βοήθεια μπαταριών που αντικαθίστανται από τον χρήστη με το πέρας του χρόνου.

Η τεχνολογία ανίχνευσης σπασμένων παραθύρων που περιέχετε σε μερικούς αισθητήρες αποτελείται από ένα ευαίσθητο μικρόφωνο που ανιχνεύει ένα περιορισμένο εύρος συχνοτήτων του ήχου, παρόμοιο με αυτό που εκπέμπετε από ένα τζάμι όταν σπάει. Όταν ανιχνεύσει μια τέτοια συχνότητα ενεργοποιείται ένας ρελές και στέλνετε σήμα ραδιοσυχνοτήτων στον κεντρικό πίνακα του συστήματος.



Εικόνα 2.1.2: Αισθητήρες κινήσεις

Ένας ακόμη βασικός αισθητήρας ενός IoT έξυπνου συστήματος ασφαλείας είναι ο αισθητήρας πόρτας που ανιχνεύει αν μια πόρτα ή ένα παράθυρο είναι ανοικτό ή κλειστό. Ο αισθητήρας αυτός αποτελείται από δύο μέρη. Το ένα μέρος περιέχει έναν μαγνητικό διακόπτη Reed switch και το άλλο είναι ένας καθαρός μαγνήτης. Ο μαγνητικός διακόπτης περιέχει ένα κύκλωμα που τα ακροφύσια του είναι κατασκευασμένα με τρόπο ώστε να μοιάζουν με μεταλλικοί ράβδοι. Οι μεταλλικοί ράβδοι βρίσκονται σε πολύ κοντινή απόσταση μεταξύ τους χωρίς ωστόσο ο ένας να ακουμπάει τον άλλον ώστε το κύκλωμα να παραμένει ανοικτό. Οι δύο ράβδοι είναι περικυκλωμένοι από ένα αδρανές αέριο και είναι κλισμένοι μέσα σε μια γυάλινη αμπούλα. Όταν πλησιάσει ένας μαγνήτης αρκετά κοντά, τα ακροφύσια μαγνητίζονται και ακουμπάνε μεταξύ τους κλίνοντας το κύκλωμα. Με τον τρόπο αυτό ο αισθητήρας γνωρίζει πότε μια πόρτα είναι ανοικτή ή κλειστή. Στέλνετε το αντίστοιχο σήμα στον κεντρικό πίνακα hub και αυτό με την σειρά του στέλνει σήμα στο νέφος cloud με τα δεδομένα που έχει συλλέξει ο αισθητήρας από την κατάσταση της πόρτας.



Εικόνα 2.1.3: Αισθητήρες πόρτας

Είναι σημαντικό σε ένα Ιοτ έξυπνο σύστημα ασφαλείας να υπάρχουν και συστήματα πυρασφάλειας. Τέτοιου ίδιους αισθητήρες ανίχνευσης φωτιάς τοποθετούνται στο ταβάνι ενός σπιτιού καθώς με βάση τους νόμους της φύσης ο ζεστός αέρας είναι ελαφρύτερος έτσι κατευθύνεται πάντα προς τα πάνω και μαζί του κατευθύνει την φλόγα και τους καπνούς. Οι περισσότεροι αισθητήρες ανίχνευσης φωτιάς αποτελούνται από ένα λέιζερ που η ακτίνα του φωτός του είναι πάντα προσανατολισμένη να ακουμπάει έναν αισθητήρα ανίχνευσης φωτός. Μόλις ο καπνός εισβάλει στην συσκευή περνάει μέσα από την ακτίνα φωτός που εκπέμπετε από το λέιζερ με αποτέλεσμα η ακτίνα να διακόπτεται ή να αλλάζει πορεία. Μόλις ο αισθητήρας φωτός σταματήσει να λαμβάνει την ακτίνα έστω και για ένα κλάσμα του δευτερολέπτου, προκειμένου να επιβεβαιωθεί πως πρόκειται για καπνός φωτιάς και όχι για καπνό διαφορετικού παράγοντα όπως καπνός από ένα ηλεκτρονικό τσιγάρο. Ελέγχεται την θερμοκρασία με την βοήθεια ενός αισθητήρα.



Ο αισθητήρας μέτρησης θερμότητας που αποτελείται από ευαίσθητα οξείδια μετάλλων βασιζόμενα σε ημιαγωγούς με επιμεταλλωμένα ή συντηγμένα καλώδια σύνδεσης ώστε να ανιχνεύεται η κατάσταση της θερμοκρασίας στον χώρο. Αν η θερμοκρασία είναι μεγαλύτερη από ένα προκαθορισμένο όριο τότε ενεργοποιείται αυτόματα ένας ρελές δημιουργώντας κλιστώ κύκλωμα με σκοπό να σταλεί άμεσα σήμα στον κύριο πίνακα hub του συστήματος και από εκεί να ειδοποιηθεί ο χρήστης. Όλοι αυτή η διαδικασία μπορεί να πραγματοποιηθεί μέσα σε ελάχιστα δευτερόλεπτα.



Εικόνα 2.1.4: Αισθητήρες φωτιάς

Οι αισθητήρες ανίχνευσης διαρροής νερού συμπεριλαμβάνονται πολύ συχνά στα IoT έξυπνα συστήματα της αγοράς. Αυτοί οι αισθητήρες τοποθετούνται σε περιοχές που υπάρχει ο κίνδυνος κάποια διαρροής όπως ένα μπάνιο, κουζίνα ή μια αποθήκη. Η λογική πίσω από αυτήν την τεχνολογία είναι απλή. Ο αισθητήρας στο κάτω μέρος του περιέχει δύο μεταλλικά ακροφύσια όπου το ένα βρίσκεται δίπλα στο άλλο χωρίς ωστόσο να έρχονται σε επαφή μεταξύ τους.

Μόλις εισέλθει νερό στην συσκευή και βραχούν τα δύο ακροφύσια τότε με την βοήθεια του νερού που είναι καλός αγωγός και επιτρέπει την μεταφορά των ηλεκτρονίων, μεταφέρεται ρεύμα στα ακροφύσια. Αυτό έχει ως αποτέλεσμα να υπάρξει βραχυκύκλωμα στην πλακέτα του αισθητήρα. Αφού γίνει η βραχυκύκλωση ενεργοποιείται ένας μικροεπεξεργαστής του κυκλώματος και στέλνει σήμα στον κεντρικό πίνακα με μια ένδειξη που αναφέρει πως κάπου στο δωμάτιο υπάρχει μια διαρροή νερού. Με αυτήν την λογική οι αισθητήρες ανίχνευσης διαρροής νερού τοποθετούνται πάντα στο έδαφος ώστε να ανιχνεύσουν τον κίνδυνο γρηγορότερα.

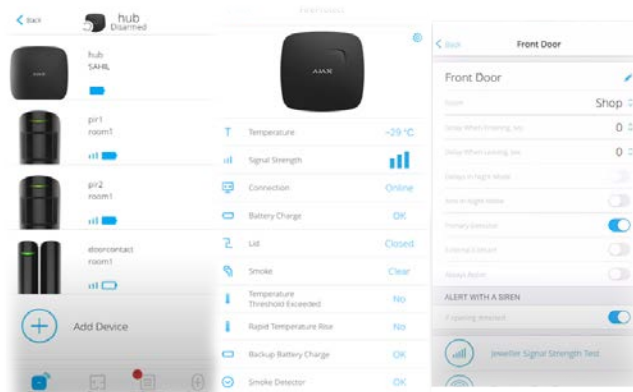


Εικόνα 2.1.5: αισθητήρες ανίχνευσης διαρροής νερού

## 2.2 Λογισμικά και σενάρια αυτοματισμού

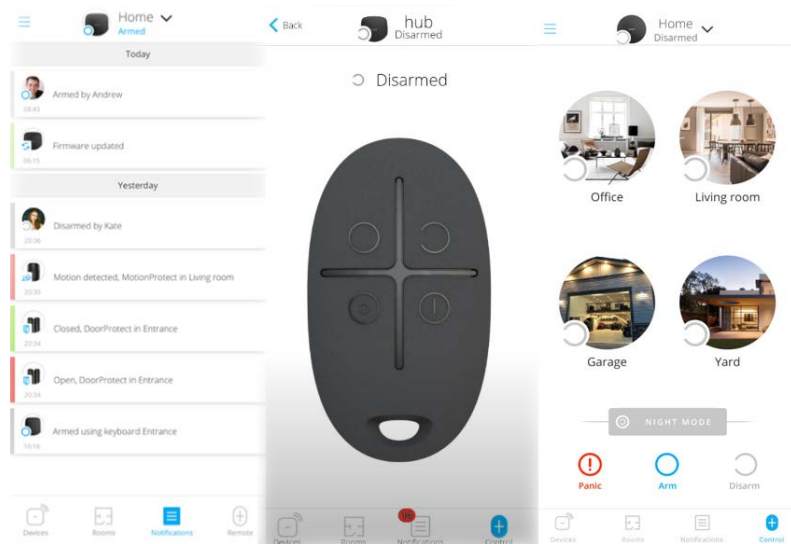
Οι εταιρίες που κατασκευάζουν IoT έξυπνα συστήματα ασφαλείας μαζί με το υλικό προσφέρουν και το κατάλληλο λογισμικό για την σωστή διαχείριση του συστήματος και την δημιουργία σκηνικών αυτοματισμού. Τις περισσότερες φορές τα προγράμματα διατίθενται δωρεάν σε συσκευές Android, Apple και windows. Μετά την εγκατάσταση ενός τέτοιου προγράμματος σε μια κινητή συσκευή θα ζητηθείς από τον χρήστη να δημιουργήσει τον δικό του προσωπικό λογαριασμό. Στην συνέχεια της εγγραφής ο χρήστης καλείται να συνδέσει στην εφαρμογή τον κεντρικό πίνακα Hub και των αισθητήρων του συστήματος ασφαλείας ώστε να μπορέσει να γίνει η επικοινωνία με το νέφος cloud της αντίστοιχης εταιρίας.

Αφού προστεθούν όλα τα υποσυστήματα του συναγερμού εμφανίζονται στην στο αντίστοιχο παράθυρο με τις συνδεδεμένες συσκευές του προγράμματος. Κατά την επιλογή μιας συνδεδεμένης συσκευής από το πρόγραμμα εμφανίζονται πληροφορίες σχετικά με την κατάσταση του αισθητήρα όπως την διάρκεια της μπαταρίας και αν βρίσκεται σε λειτουργία. Επίσης σε αυτό το σημείο γίνεται η μεμονωμένη διάχυση ενός αισθητήρα. Παρέχονται επιλογές όπως ο ορισμός της ονομασίας του δωματίου που βρίσκεται εγκατεστημένο το συγκεκριμένο σύστημα, η προσθήκη χρόνου καθυστέρησης κατά την έναρξη και το τέλος της ανίχνευσης, ο επιθυμητός χρόνος ενεργοποίησης του αισθητήρα και προκαθορισμένες από τον χρήστη ρυθμίσεις όπως λειτουργία νύχτας και ημέρας. Παρακάτω απεικονίζεται το κεντρικό παράθυρο και το παράθυρο διαχείρισης συνδεδεμένων αισθητήρων του προγράμματος που συνοδεύεται από τα έξυπνα συστήματα ασφαλείας Ajax systems



Εικόνα 2.2.1: Παράθυρο εφαρμογής και διαχείρισης Ajax Systems

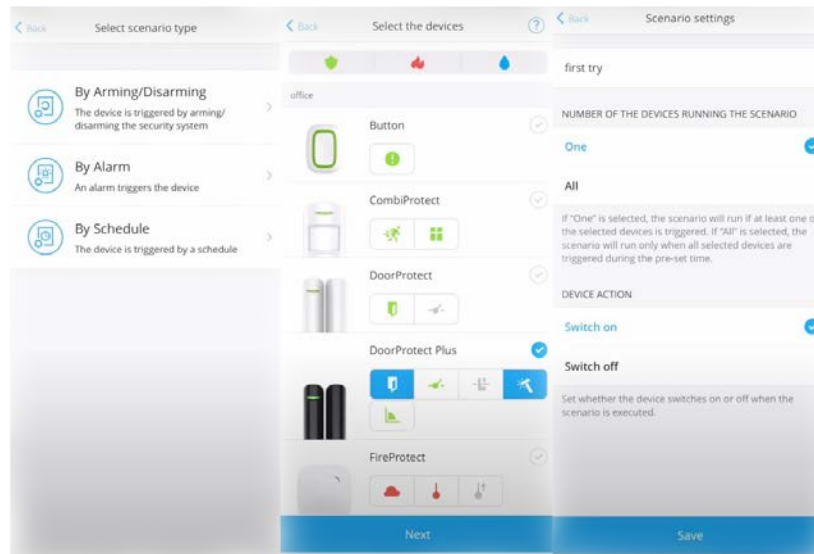
Κάθε πρόγραμμα που συνοδεύεται από ένα IoT έξυπνο σύστημα ασφαλείας παρέχονται οι ενότητες ειδοποιήσεων και ελέγχου. Στο εσωτερικό της ενότητας ειδοποιήσεων βρίσκεται όλο το ιστορικό της καταγραφής του συστήματος όπως το πότε άνοιξε μια πόρτα ενεργοποιώντας τον αντίστοιχο αισθητήρα αλλά και το ποια ήταν η τελευταία φορά που πραγματοποιήθηκε μια ενημέρωση λογισμικού. Στην ενότητα ελέγχου ο χρήστης μπορεί να θέσει το σύστημα του ως λειτουργία αδράνειας, έκτακτης ανάγκης, λειτουργία κλειδώματος και την επιλογή του να γίνει η απενεργοποίηση ολόκληρου του συστήματος. Τέλος παρέχετε η δυνατότητα στον χρήστη μέσω του προγράμματος να ομαδοποιεί τους αισθητήρες ανάλογα το δωμάτιο του σπιτιού που βρίσκονται. Με αυτόν τον τρόπο επιλέγοντας μια ομάδα οι αισθητήρες που ανήκουν σε αυτή ενεργοποιούνται, απενεργοποιούνται, και τίθενται σε αδράνεια σχετικά με τις απαιτούμενες ανάγκες. Η παρακάτω εικόνα απεικονίζει τις ενότητες που αναφέρθηκαν μέσω του προγράμματος Ajax systems.



Εικόνα 2.2.2: Ειδοποιήσεις, έλεγχος και δωμάτια εφαρμογής Ajax systems

Ένα βασικό χαρακτηριστικό των προγραμμάτων είναι πως παρέχετε η δυνατότητα στον χρήστη να δημιουργήσει τα δικά του μοναδικά σενάρια αυτοματισμού. Μέσο των ρυθμίσεων ενός υποσυστήματος για παράδειγμα έναν έξυπνο ρελέ υπάρχει η επιλογή για την εισαγωγή σεναρίου. Αρχικά πρέπει να επιλεγθεί ο τρόπος με τον οποίο θα ενεργοποιείτε το σενάριο. Με την ενεργοποίηση ή την απενεργοποίηση του συστήματος συναγερμού, με την ανίχνευση κάποιας απειλής μέσω ενός αισθητήρα ή μέσω συγκεκριμένης ώρας και ημέρας που καθορίζεται από την χρήστη. Διαλέγοντας μια από της παραπάνω επιλογές όπως ενεργοποίηση με την ανίχνευση κάποιας απειλής μέσω ενός αισθητήρα, εμφανίζονται στην οθόνη όλα τα συνδεδεμένα υποσυστήματα ομαδοποιημένα σε κατηγορίες. Ο χρήστης καλείται να επιλέξει τον αισθητήρα που επιθυμεί να ενεργοποιεί το συγκεκριμένο σενάριο αυτοματισμού. Στην συνέχεια είναι σημαντικό να επιλεγθεί ο τρόπος με τον οποίο θα ενεργοποιήσει ο αισθητήρας το σενάριο.

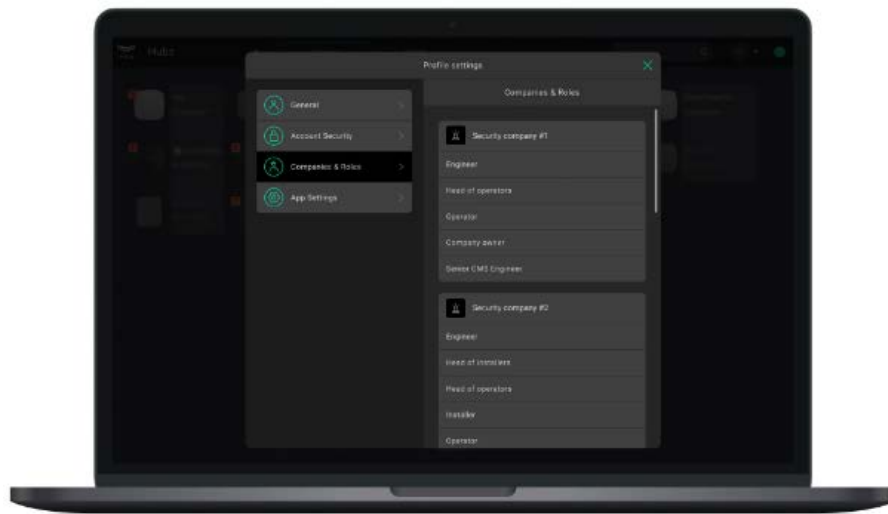
Αν επιλεγθεί για παράδειγμα ο αισθητήρας ανίχνευσης φωτιάς πρέπει να επιλεγθεί και η συνθήκη του. Οι επιλογές ποικίλουν ανάλογα με τον τύπο του αισθητήρα. Στον αισθητήρα ανίχνευσης φωτιάς υπάρχει η δυνατότητα στο σενάριο να ενεργοποιηθεί μόλις ανιχνευθεί φωτιά ή όταν ο αισθητήρας θεωρήσει πως η φωτιά έχει σταματήσει να βρίσκεται στον χώρο. Είναι σημαντικό να αναφερθεί πως σε αυτό το σημείο μπορεί να επιλεγθεί πάνω από μια διαφορετική συνθήκη. Στο τελευταίο βήμα δίνεται από τον χρήστη ένα αντιπροσωπευτικό όνομα του σεναρίου ώστε να υπάρχει καλύτερη κατανόηση για το τι διαχειρίζεται ο συγκεκριμένος αυτοματισμός. Στην συνέχεια προστίθενται τα αποτελέσματα που θα έχει η εκτέλεση του σεναρίου. Για παράδειγμα όταν εκτελεσθούν όλες οι κατάλληλες συνθήκες να απενεργοποιηθεί ο έξυπνος ρελέ και να σταματήσει την κεντρική πυγή ρεύματος σε ένα σπίτι ώστε να αποφευχθεί μια ενδεχόμενη μεγαλύτερης πυρκαγιά. Με αυτόν τον τρόπο ο χρήστης μόλις έχει φτιάξει ένα σενάριο που λέει πως όταν ένας συγκεκριμένος αισθητήρας καταλάβει πως υπάρχει φωτιά τότε αυτόματα απενεργοποίησε έναν έξυπνο ρελέ με αποτέλεσμα να τεθεί σε παύση η διανομή ρεύματος στο σπίτι. Η παρακάτω εικόνα αποτελεί ένα παράδειγμα της κατηγορίας σεναρίων αυτοματισμού μέσω του προγράμματος Ajax systems.



Εικόνα 2.2.3: Σενάρια αυτοματισμού

Όλες οι ρυθμίσεις, τα σενάρια αυτοματισμού, οι ειδοποιήσεις και οι λειτουργίες που βρίσκονται στην εγκατεστημένο πρόγραμμα του χρήστη δεν αποθηκεύονται στην μνήμη του τηλεφώνου αλλά σε μια βάση δεδομένων που ανήκει στο νέφος cloud της εταιρείας που κατασκεύασε το αντίστοιχο IoT έξυπνο σύστημα ασφαλείας. Με αυτόν τον τρόπο εξασφαλίζεται πως σε περίπτωση που χαθεί ή κατεστράφη η έξυπνη τηλεφωνική συσκευή του χρήστη τα δεδομένα δεν θα χαθούν και θα παραμείνουν ασφαλές. Με μια απλή σύνδεση στον λογαριασμό του χρήστη ανεξάρητου συσκευής τα δεδομένα θα επανέλθουν στην παρούσα κατάσταση τους. Τα προγράμματα που τρέχει σε υπολογιστές Windows και Apple συνήθως διαφέρει από το αντίστοιχο των android και iOS συσκευών.

Απευθύνονται περισσότερο σε επιχρίσεις και προσφέρουν λειτουργίες όπως η εισαγωγή πολλαπλών μελών με περιορισμένη πρόσβαση. Αυτές οι λειτουργίες επιτρέπουν σε έναν ιδιοκτήτη μιας επιχείρησης να δώσει πρόσβαση στο σύστημα ασφαλείας της εταιρείας του σε έναν ή περισσότερους υπαλλήλους του. Μπορεί να ρυθμιστεί η πρόσβαση που θα έχει κάθε εργαζόμενος και τα δικαιώματα διαχείρισης του συστήματος. Με αυτόν τον τρόπο μπορεί να οριστεί μια ομάδα που θα έχει πρόσβαση μόνο στις κάμερες της εταιρείας και θα είναι υπεύθυνη για την συντήρηση τους. Μια ομάδα υπεύθυνη μόνο για τους αισθητήρες κίνησης και άλλη μια με μοναδική πρόσβαση στα συστήματα πυρασφάλειας. Ο ιδιοκτήτης έχοντας πρόσβαση σε όλο το σύστημα μπορεί να επιβλέπει ολόκληρη την λειτουργία του ή να επιτρέψει την πρόσβαση όλου του συστήματος σε κάποιον γενικό υπεύθυνο ασφαλείας. Ένα παράδειγμα ενός τέτοιου προγράμματος αποτελεί η παρακάτω εικόνα που απεικονίζει το πρόγραμμα της εταιρίας Ajax systems σε υπολογιστή με λειτουργικό σύστημα Windows.



Εικόνα 2.2.4: Πολλαπλά μέλη με περιορισμένη πρόσβαση Ajax systems



Ένα μικρό ποσοστό των εταιριών που κατασκευάζουν IoT έξυπνα συστήματα ασφαλείας προσφέρουν ένα αντίστοιχο πρόγραμμα που απευθύνετε σε έξυπνα ρολόγια. Εκεί ο χρήστης μπορεί να έχει πρόσβαση σε πολύ βασικές λειτουργίες χωρίς να χρειαστεί να ανοίξει το τηλέφωνο ή τον υπολογιστή του χάνοντας πολύτιμο χρόνο. Αυτές οι λειτουργίες είναι η άμεση απόπλιση του συναγερμού, η ενεργοποίηση του και η έναρξη κάποιας ρουτίνας όπως η ρουτίνα με όνομα λειτουργία νυκτός. Συνήθως εκεί βρίσκεται και ένα κουμπί έκτακτης ανάγκης που ενεργοποιεί την σειρήνα και επικοινωνεί με συγκεκριμένους ανθρώπους που έχουν προκαθοριστεί από τον χρήστη. Τέλος στο ρολόι εμφανίζονται όλες οι ιδιοποιήσεις τους συστήματος ασφαλείας ώστε να ενημερώνεται ο χρήστης ακαριαία. Η σημαντική προϋπόθεση για να λαμβάνονται οι ιδιοποιήσεις στο ρολόι είναι να υπάρχει σύνδεση μέσω Bluetooth με το τηλέφωνο. Όλες οι πληροφορίες μεταφέρονται στο ρολόι από το τηλέφωνο μέσω τεχνολογίας Bluetooth.



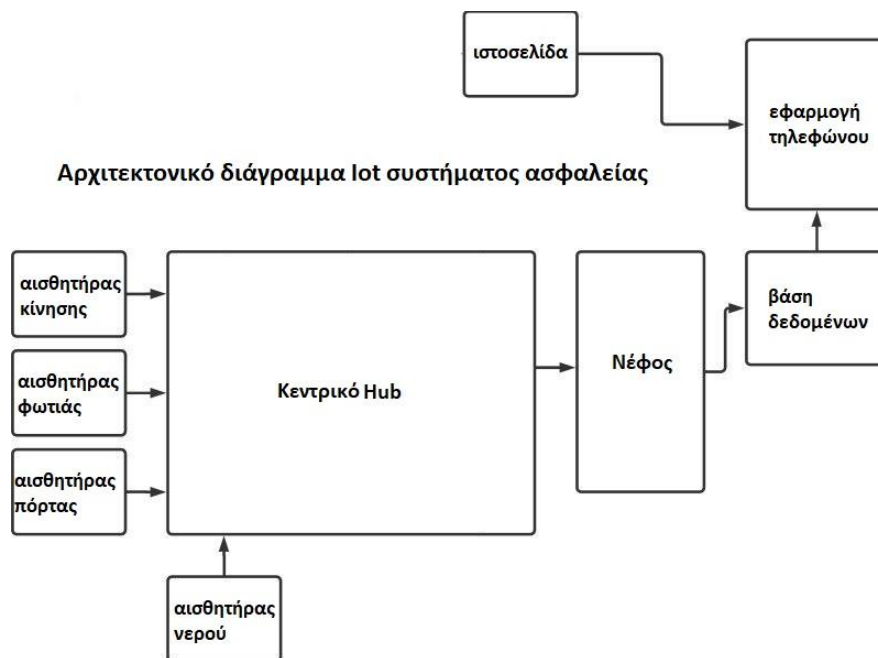
Εικόνα 2.2.5: Πρόγραμμα για έξυπνα ρολόγια

Ο κάτοχος ενός έξυπνου συστήματος συναγερμού έχει τον πλήρη έλεγχο από το τηλέφωνο ή τον υπολογιστή του. Για αυτό είναι εξίσου σημαντικό να υπάρχει ενεργοποιημένο ένα μέτρο ασφαλείας στην εφαρμογή όπως η μέθοδος ταυτοποίησης δύο παραγόντων Two factor authentication 2FA.

Ενεργοποιώντας μια δεύτερη μέθοδο ταυτοποίησης ο χρήστης εξασφαλίζει πως κανένας άλλος δεν μπορεί να εισβάλει στον λογαριασμό του ώστε να έχει πρόσβαση στο σύστημα συναγερμού μέσω της εφαρμογής εκτός από τον ίδιο. Έχοντας ενεργοποιημένο τον παράγοντα 2FA ο χρήστης κατά την σύνδεση του λογαριασμού που έχει δημιουργήσει για το έξυπνο σύστημα ασφαλείας, στέλνετε ένας προσωρινός κωδικός σε μορφή sms σε έναν αριθμό τηλεφώνου που έχει οριστεί. Εισαγάγω ντας τον στο απαιτούμενο πεδίο της εφαρμογής εξασφαλίζεται πως η σύνδεση είναι ασφαλές επιτρέποντας τον χρήστη να εισέλθει και να διαχειριστεί το σύστημα ασφαλείας του. Ο κωδικός αυτός έχει ισχύ μόνο ένα λεπτό. Αν δεν προλάβει να γίνει η σύνδεση μέχρι τότε στέλνετε ένας καινούριος και ο παλιός κωδικός παύει να ισχύ. Ο προσωρινός κωδικός μπορεί να σταλεί στον χρήστη με πολλαπλούς τρόπους. Σε μορφή sms, σε μορφή ηχογραφημένης κλήσεις ή μέσω κάποιας τρίτης εφαρμογής όπως είναι ο επαληθευτής Google ή το Microsoft authenticator. Τέλος το 2FA προσφέρει μια μέθοδο ταυτοποίησης χωρίς ωστόσο να χρειάζεται ο χρήστης να πληκτρολογήσει κάποιον προσωρινό κωδικό. Κατά την ενεργοποίηση του ο χρήστης μπορεί να αποθηκεύσει την διεύθυνση IP ενός τηλεφώνου ή μιας ταμπλέτας που έχει στην κατοχή του. Με αυτήν την μέθοδο κάθε φορά που συνδέεται κάποιος στον λογαριασμό που χρησιμοποιείται στο έξυπνο σύστημα ασφαλείας, αυτόματα θα εμφανίζεται ένα παράθυρο στην συσκευή που ανήκει η αποθηκευμένη διεύθυνση IP, και θα ζητάει από τον κάτοχο να επαληθεύσει την σύνδεση του πατώντας ένα κουμπί επιβεβαίωσης.

## 2.3 Διαγράμματα ροής και διαγράμματα μπλοκ

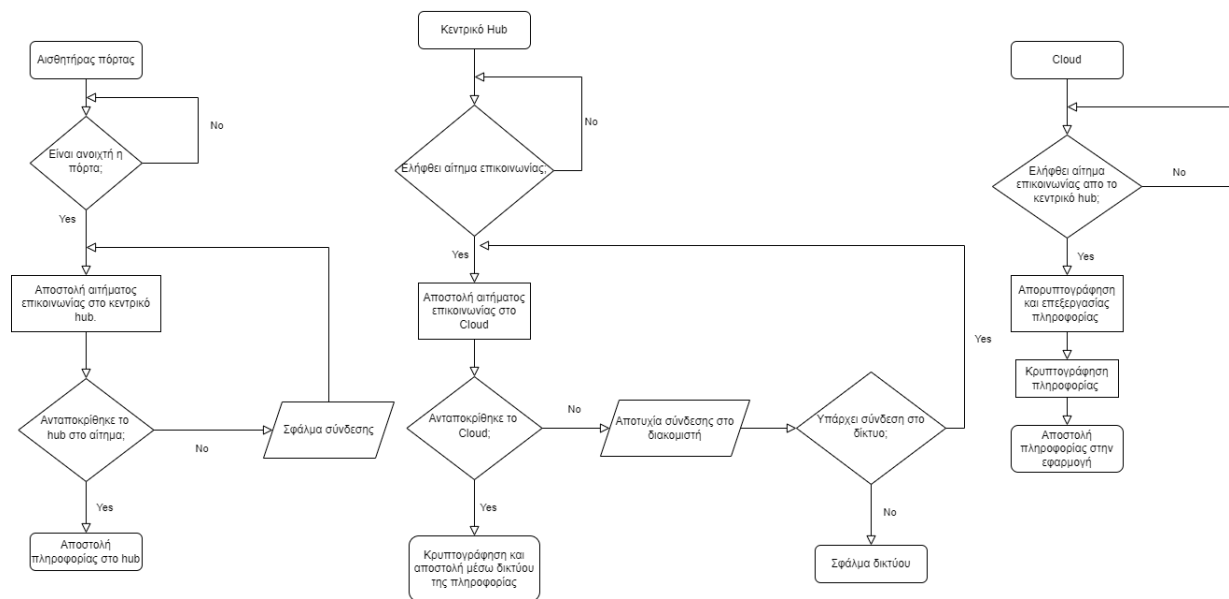
Στα παρακάτω διαγράμματα ροής εξετάζεται η λειτουργία της μεταφοράς πληροφοριών μεταξύ των υποσυστημάτων του έξυπνου συστήματος ασφαλείας αλλά και την συνδεσιμότητα του προγράμματος με το νέφος cloud που ανήκει στην εκάστοτε εταιρεία. Επίσης εξετάζεται το σενάριο αυτοματισμού που αναφέρθηκε στην προηγούμενη παράγραφο και μερικά επιπλέον παραδείγματα από δημοφιλές σενάρια αυτοματισμού. Τέλος στα διαγράμματα μπλοκ που ακολουθούν, παρουσιάζεται ο τρόπος λειτουργίας και διασύνδεσης μεταξύ του εσωτερικού υλικού που αποτελούν κάθε αισθητήρα που αναφέρθηκε συμπεριλαμβανομένου του κεντρικού πίνακα hub. Τα διαγράμματα αυτά δημιουργήθηκαν βασιζόμενα στα τεχνικά χαρακτηριστικά και τις αρχιτεκτονικές από τα δεδομένα και τις πληροφορίες που παρουσιάστηκαν στην ενότητα Αρχιτεκτονική στα IoT συστήματα ασφαλείας.



Διάγραμμα 2.3.1: Αρχιτεκτονικό διάγραμμα IoT συστήματος ασφαλείας

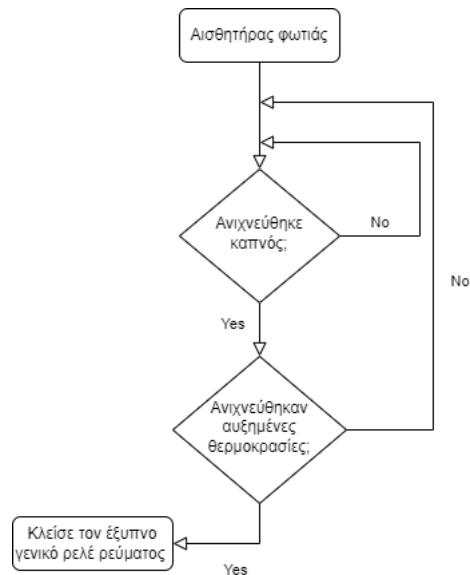
Στο παραπάνω διάγραμμα απεικονίζεται η αρχιτεκτονική ενός απλού IoT συστήματος ασφαλείας.

Στο επόμενο διάγραμμα απεικονίζεται η διαδικασία με την οποία ένας απλός αισθητήρας πόρτας στέλνει την πληροφορία πως μια πόρτα είναι ανοιχτή στο κεντρικό hub, και από εκεί η πληροφορία στέλνεται κρυπτογραφημένη στο Cloud της εταιρείας και στην συνέχεια μεταφέρεται στην εγκατεστημένη εφαρμογή που έχει ο χρήστης στο έξυπνο τηλέφωνο του.



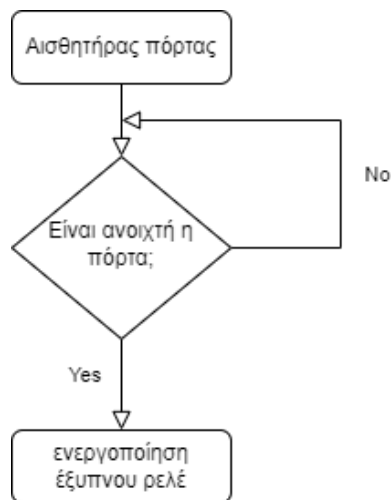
Διάγραμμα ροής 2.3.2: διάγραμμα ροής μεταφοράς πληροφοριών

Στο επόμενο διάγραμμα ροής αναπαρίσταται το σενάριο αυτοματισμού που αναφέρθηκε στο παραπάνω παράδειγμα. Κατά την ανίχνευση μιας πυρκαγιάς ανοικτής πόρτας ενεργοποιείται αυτόματα ένας έξυπνος ρελές.



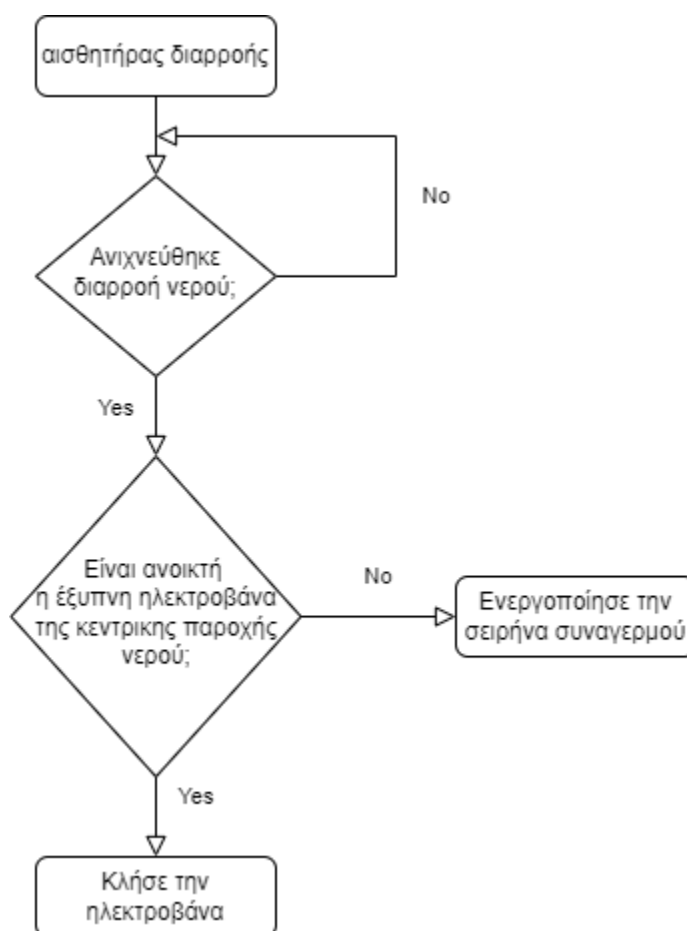
Διάγραμμα ροής 2.3.3: διάγραμμα ροής σεναρίου αισθητήρα

Στο επόμενο διάγραμμα ροής αναπαρίσταται ένα σενάριο αυτοματισμού από ένα αισθητήρα ανίχνευσης κατάστασης πόρτας. Κατά την ανίχνευση μιας ανοικτής πόρτας ενεργοποιείται αυτόματα ένας έξυπνος ρελές.



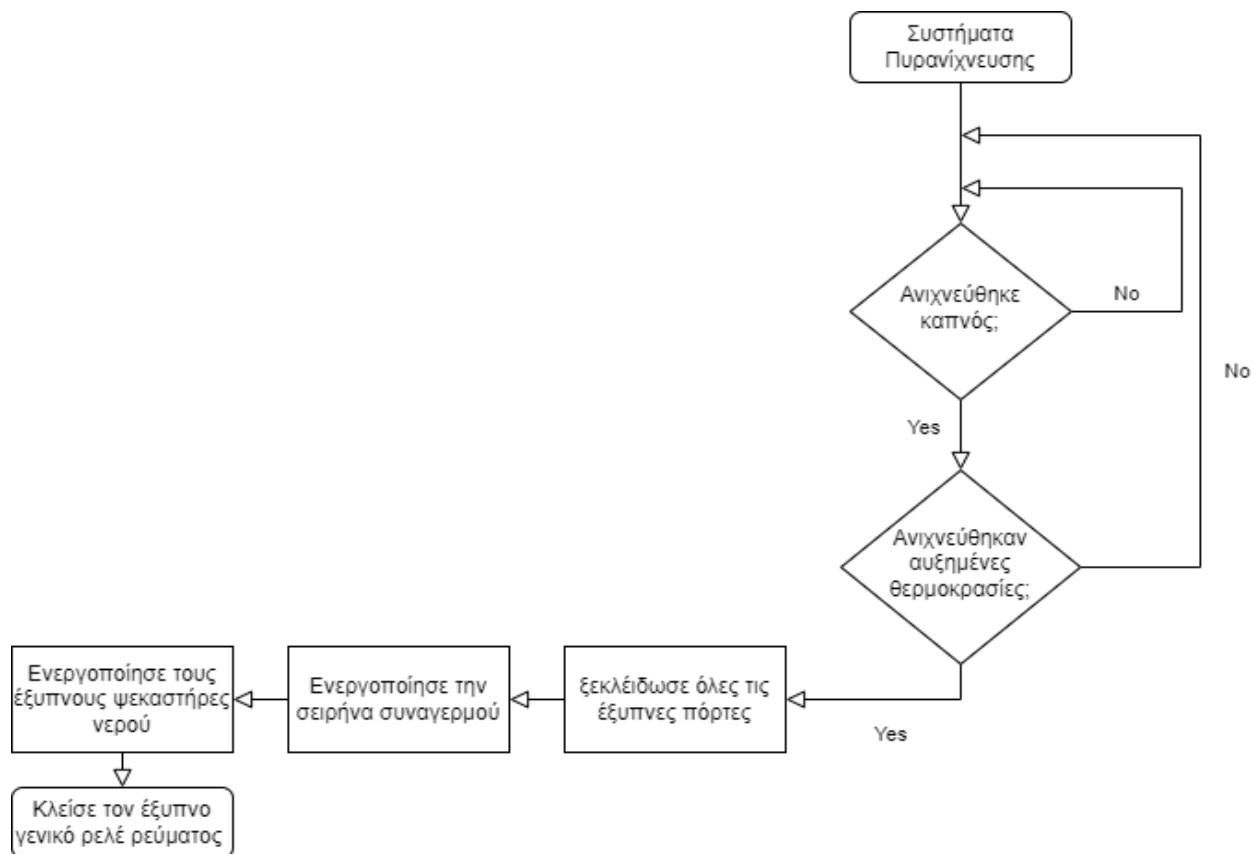
Διάγραμμα ροής 2.3.4: διάγραμμα ροής σεναρίου αισθητήρα πόρτας

Στο παρακάτω διάγραμμα εξετάζεται ένα σενάριο αυτοματισμού που εξυπηρετεί της ανάγκες μιας διαρροής νερού. Η διαδικασία ξεκινάει μόλις ένας αισθητήρας διαρροής ανιχνεύσει νερό. Στην συνέχεια ελέγχεται αν η έξυπνη ηλεκτροβάννα που είναι συνδεδεμένη στον κεντρικό σωλήνα παροχής νερού είναι ανοιχτή. Αν είναι ανοιχτή τότε ο αυτοματισμός την κλείνει ώστε να σταματήσει προσωρινά η διαρροή. Σε περίπτωση που είναι ήδη κλειστή τότε το σύστημα ενεργοποιεί την σειρά του συναγερμού ώστε να ενημερωθεί εγκαίρως ο ιδιοκτήτης.

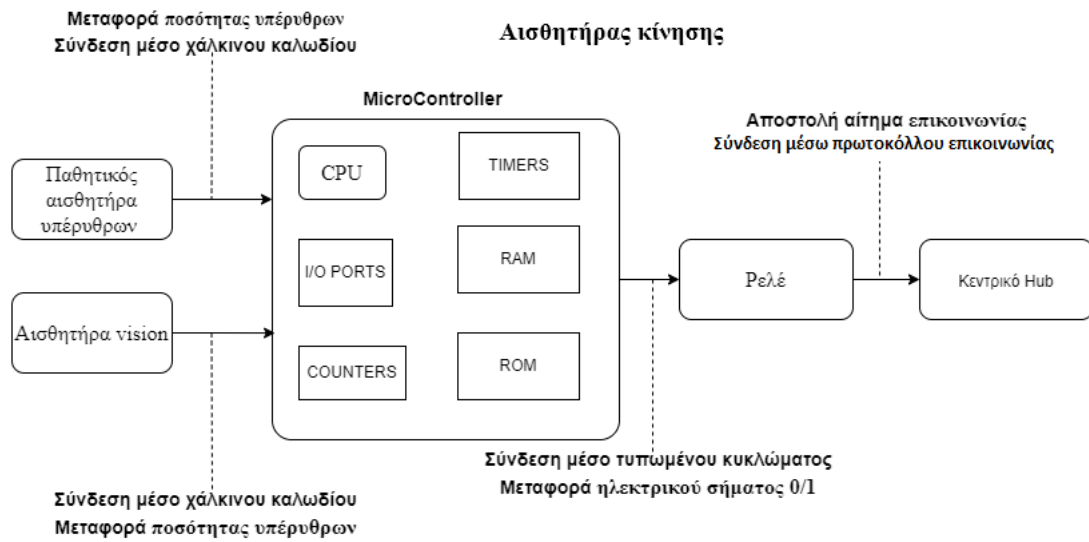


Διάγραμμα ροής 2.3.5: διάγραμμα ροής σεναρίου αισθητήρα διαρροής

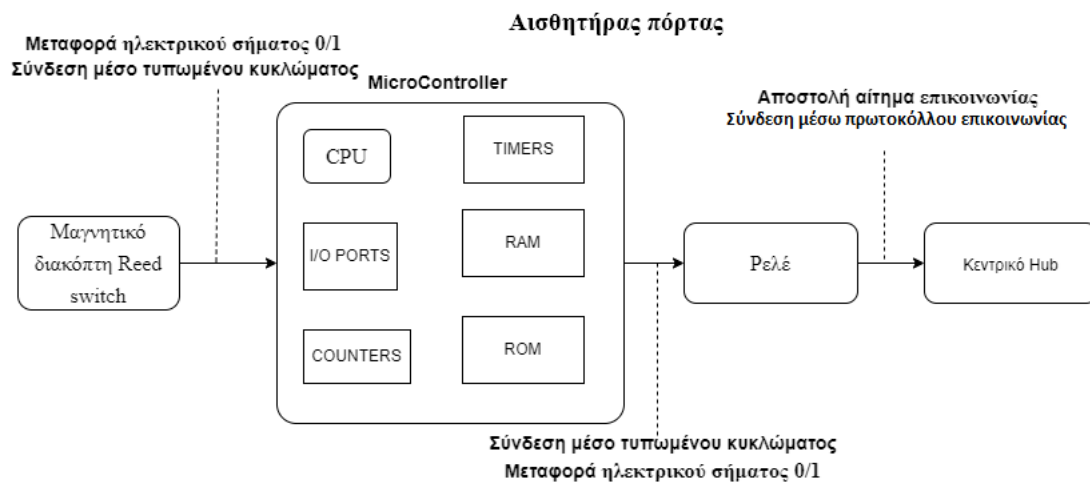
Στο τελευταίο διάγραμμα ροής υλοποιείται ένα σενάριο αυτοματισμού που είναι υπεύθυνο για την αντιμετώπιση μιας ενδεχόμενης πυρκαγιάς. Αρχικά ελέγχεται αν υπάρχει ανίχνευση καπνού και απότομη αλλαγή αυξημένης θερμοκρασίας στον χώρο. Σε περίπτωση που ισχύουν και οι δύο συνθήκες τότε αυτόματα ξεκλειδώνουν όλες οι πόρτες που έχουν έξυπνη κλειδαριά ώστε να γίνει ευκολότερη η απεγκλώβιση και η εγκατάλειψη του κτηρίου. Ενεργοποιείται η σειρήνα του συναγερμού ώστε να ειδοποιηθούν όλοι για την πυρκαγιά και ενεργοποιούνται οι έξυπνοι ψεκαστήρες νερού που βρίσκονται στον χώρο. Τέλος απενεργοποιείται ο έξυπνος ρελές γενικού ρεύματος για να αποφευχθεί η γρήγορη εξάπλωση της πυρκαγιάς.



Διάγραμμα ροής 2.3.6: διάγραμμα ροής σεναρίου σύστημα πυρανίχνευσης

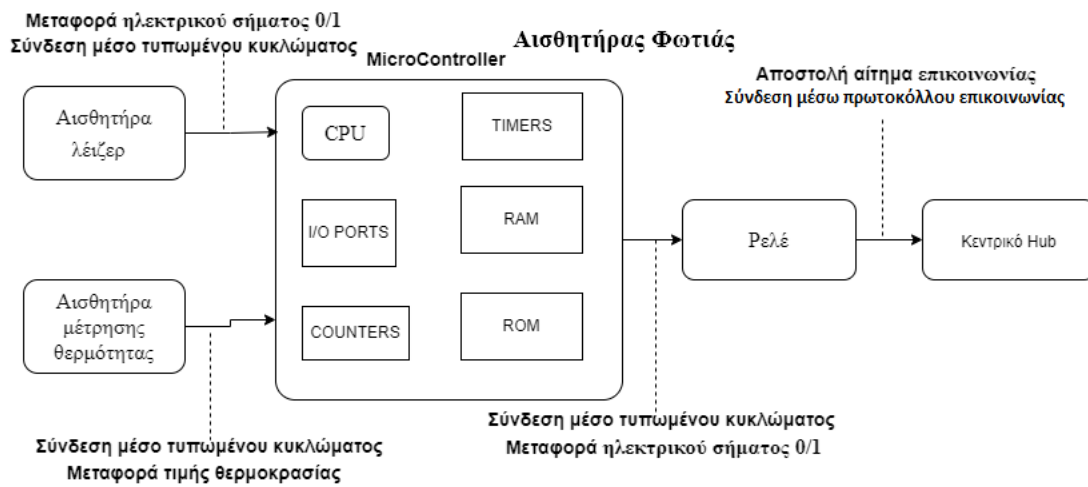


Διάγραμμα μπλοκ 2.3.7: διάγραμμα μπλοκ αισθητήρας κίνησης

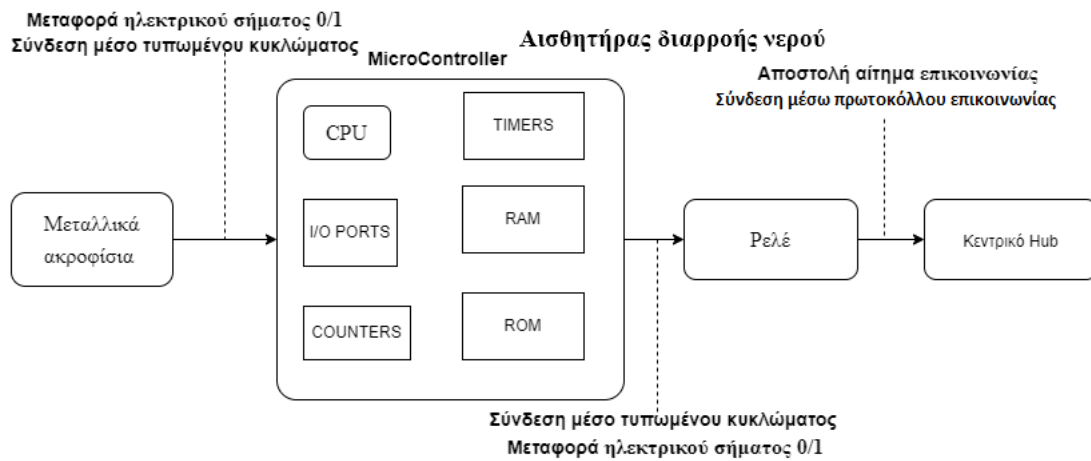


Διάγραμμα μπλοκ 2.3.8: διάγραμμα μπλοκ αισθητήρας πόρτας

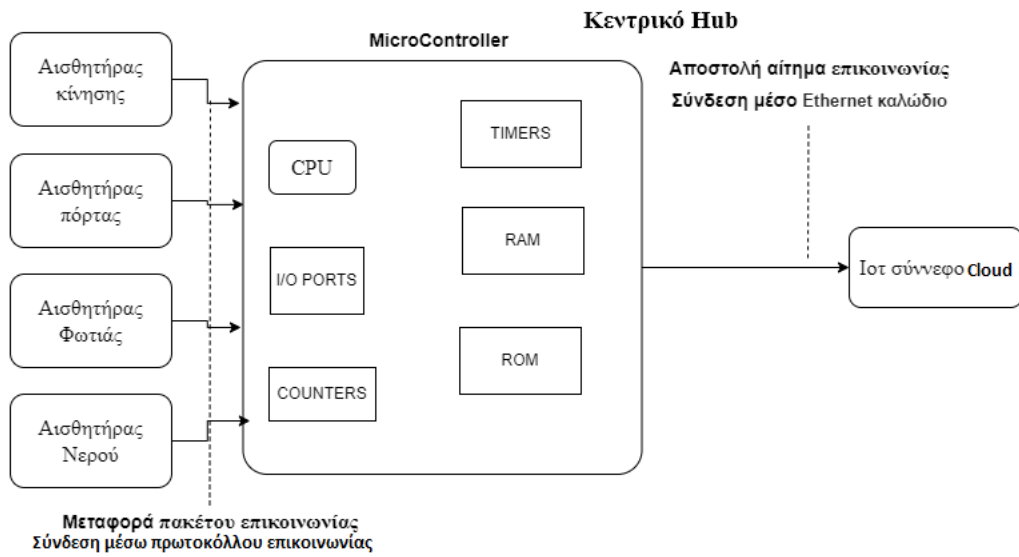




Διάγραμμα μπλοκ 2.3.9: διάγραμμα μπλοκ αισθητήρας φωτιάς



Διάγραμμα μπλοκ 2.3.10: διάγραμμα μπλοκ αισθητήρας διαρροής νερού



Διάγραμμα μπλοκ 2.3.11: διάγραμμα μπλοκ κεντρικού Hub

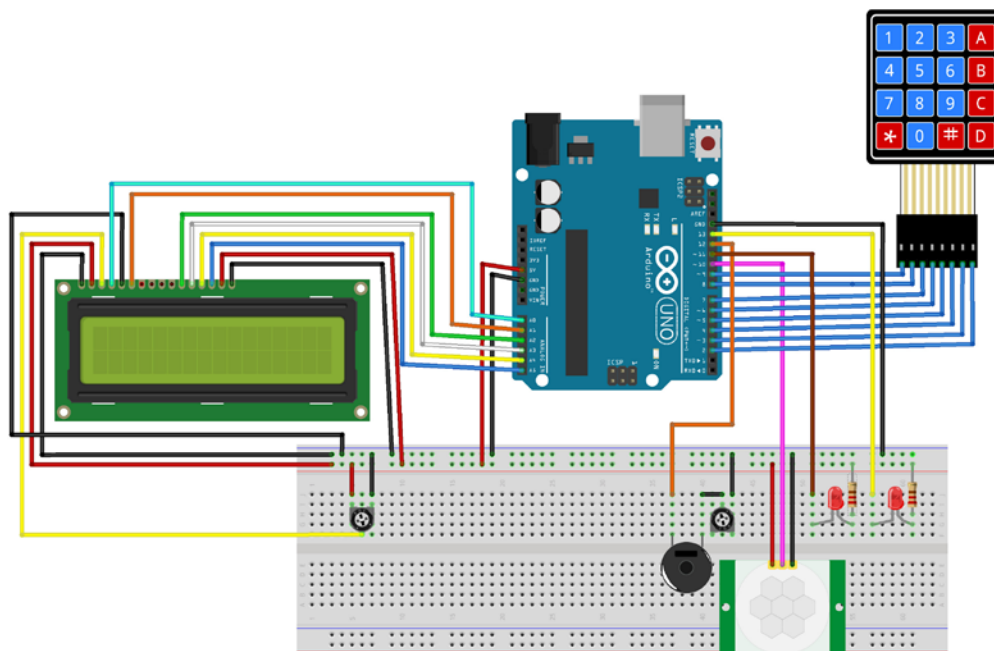
## ΚΕΦΑΛΕΟ 3 ΥΛΟΠΟΙΗΣΗ

### 3.1 Προτάσεις για υλοποίηση

Η Ajax systems που αναφέρθηκε στο προηγούμενο κεφάλαιο δεν είναι η μοναδική εταιρεία όσο αφορά την κατασκευή έξυπνων συστημάτων ασφαλείας. Ο κορυφαίος Γαλλικός όμιλος Somfy εξειδικεύεται στα έξυπνα συστήματα ασφαλείας και αυτοματισμού με τα συστήματα Somfy protect. Έχει την δικιά της εφαρμογή για την διαχείριση των συστημάτων και την δημιουργία σεναρίων αυτοματισμού. Παρέχει συστήματα όπως αισθητήρες κίνησης και ανίχνευσης, κάμερες ασφαλείας, σειρήνες συναγερμού καθώς και κάμερες με ενσωματωμένες σειρήνες. Η παγκόσμιος γνωστή εταιρεία Amazon, από το έτος 2013 κατασκευάζει τα δικά της κορυφαία έξυπνα συστήματα ασφαλείας με όνομα Amazon ring. Παρέχουν όλες της παραπάνω δυνατότητες αλλά έχουν ένα σημαντικό πλεονέκτημα. Το Amazon ring υποστηρίζεται άψογα από την ψηφιακή βοηθό Alexa που κατασκευάζεται από την ίδια εταιρεία. Αυτό έχει ως αποτέλεσμα το σύστημα συναγερμού να υποστηρίζει φωνητικές εντολές. Στα συστήματα ασφαλείας άλλων εταιρειών που δεν υποστηρίζεται ένας ψηφιακός βοηθός ο χρήστης προκειμένου να διαχειριστή τους αισθητήρες και ολόκληρο τον συναγερμού θα χρειαστεί να χειριστεί την αντίστοιχη εφαρμογή της εταιρείας. Στην περίπτωση του Amazon ring αρκεί απλά ο χρήστης να πει αυτό που θέλει. Με απλές εντολές όπως ‘Alexa, Turn on the alarm system’ και ‘Alexa, Disable the sensors of bedroom for tonight’ γίνεται η διαχείριση του συστήματος χωρίς την χρήση της εφαρμογής. Αυτό γίνεται με ασφαλές τρόπο καθώς η ψηφιακή βοηθός έχει την δυνατότητα να αναγνωρίζει την φωνή του ιδιοκτήτη.

### 3.2 Υλοποίηση συστήματος ασφαλείας IoT

Οι χρήστες εκτός από το να πάρουν ένα έτοιμο σύστημα ασφαλείας έχουν την δυνατότητα να κατασκευάσουν τα δικά τους μοναδικά συστήματα ασφαλείας σε επίπεδο υλικού αλλά και λογισμικού χρησιμοποιώντας ένα σύνολο από συνδεδεμένες αντιστάσεις, πυκνωτές, αισθητήρες και μιας πλακέτας μικροηλεκτρονικών όπως είναι το Arduino. Θα είναι εξ ολοκλήρου σχεδιασμένο και κατασκευασμένο από τον χρήστη με όλες τις επιθυμητές προδιαγραφές. Ωστόσο αυτή η διαδικασία είναι αρκετά χρονοβόρα και απαιτεί προχωρημένες γνώσεις προγραμματισμού, δικτύων, ηλεκτρονικών συστημάτων, κυκλωμάτων και Cloud.

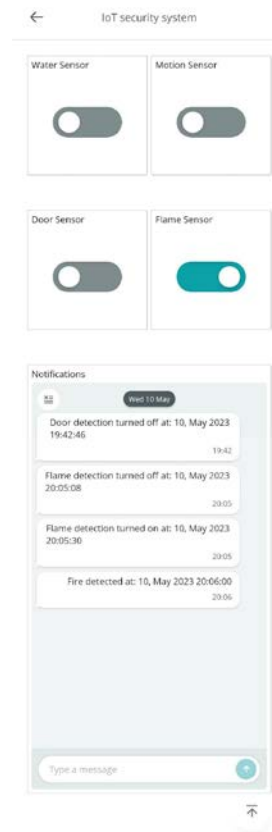


Εικόνα 3.2.1: Έξυπνο σύστημα ασφαλείας σε Arduino

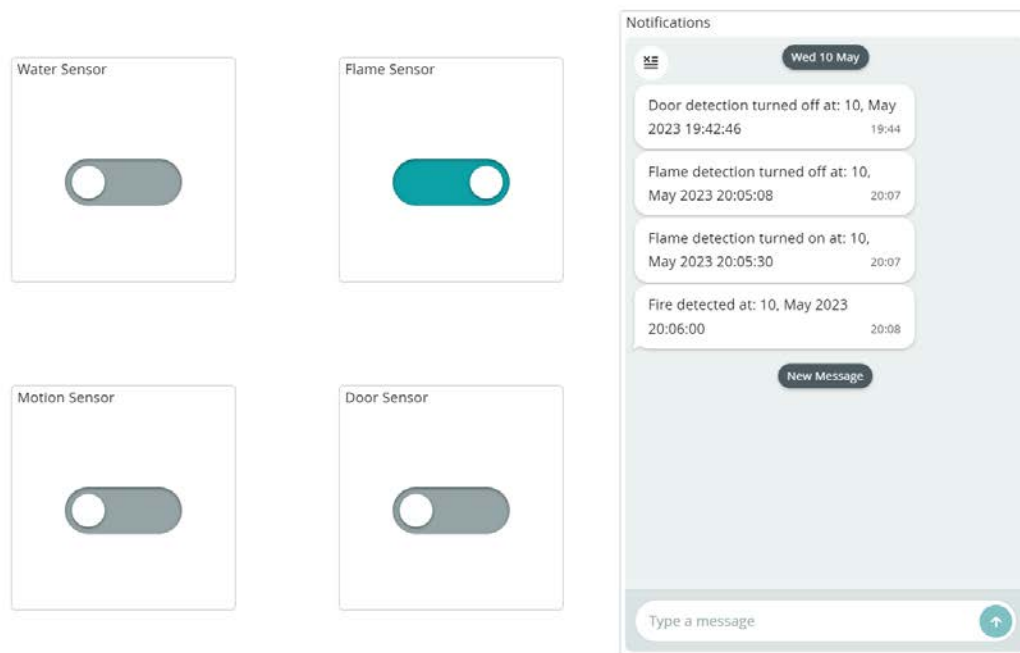
Στα πλαίσια της παρούσας πτυχιακής εργασίας κατασκευάστηκε ένα έξυπνο IoT σύστημα ασφαλείας που ενσωματώνει λειτουργίες όπως ανίχνευση πυρκαγιάς και διαρροής υγρών, αναγνώριση ανοικτής ή κλειστής πόρτας, ανίχνευση κίνησης, μια φωτεινή ένδειξη και ένα μικρό προειδοποιητικό ηχείο. Εκτός από τους αισθητήρες και τα ηλεκτρονικά του IoT συστήματος σχεδιάστηκε και υλοποιήθηκε το αντίστοιχο λογισμικό με τρόπο ώστε να είναι εύκολο και φιλικό στην χρήση. Με ενσωματωμένο σύστημα ειδοποιήσεων εντός της εφαρμογής, ειδοποιήσεις μέσω e-mail και σύστημα διαχείρισης αισθητήρων. Το λογισμικό που κατασκευάστηκε ορίζεται ως διαδικτυακή εφαρμογή με αποτέλεσμα να είναι συμβατό με τηλέφωνα και ταμπλέτες Android, iOS. Αλλά και με υπολογιστές που χρησιμοποιούν λειτουργικό σύστημα Windows, MacOS και Linux. Επίσης επειδή η εφαρμογή βασίζεται στο διαδίκτυο δεν απαιτείται από τον χρήστη να κάνει κάποια εγκατάσταση στην συσκευή του και όλα τα δεδομένα του έξυπνου συστήματος ασφαλείας αποθηκεύονται στο νέφος Cloud. Το μόνο που απαιτείται για την χρήση της εφαρμογής είναι να υπάρχει σύνδεση στο δίκτυο και ο χρήστης να έχει συνδέσει τον λογαριασμό του e-mail στην εφαρμογή. Στις επόμενες ενότητες αναφέρετε αναλυτικά όλη η διαδικασία της υλοποίησης του έξυπνου συστήματος ασφαλείας. Από την επιλογή των υλικών και το κόστος κατασκευής του μέχρι την δημιουργία του λογισμικού και όλα τα προβλήματα που αντιμετωπίστηκαν.

### 3.3 Τρόπος λειτουργίας

Ολόκληρη η κατασκευή του IoT έξυπνο σύστημα ασφαλείας έχει πραγματοποιηθεί με γνώμονα την ευχρηστία και την ευκολία. Η διαδικτυακή εφαρμογή που συνοδεύεται με το σύστημα περιλαμβάνει τέσσερις επιλογές ενεργοποίησης και απενεργοποίησης των αισθητήρων του συστήματος και μια ξεχωριστή ενότητα που λαμβάνονται όλες οι ειδοποιήσεις.



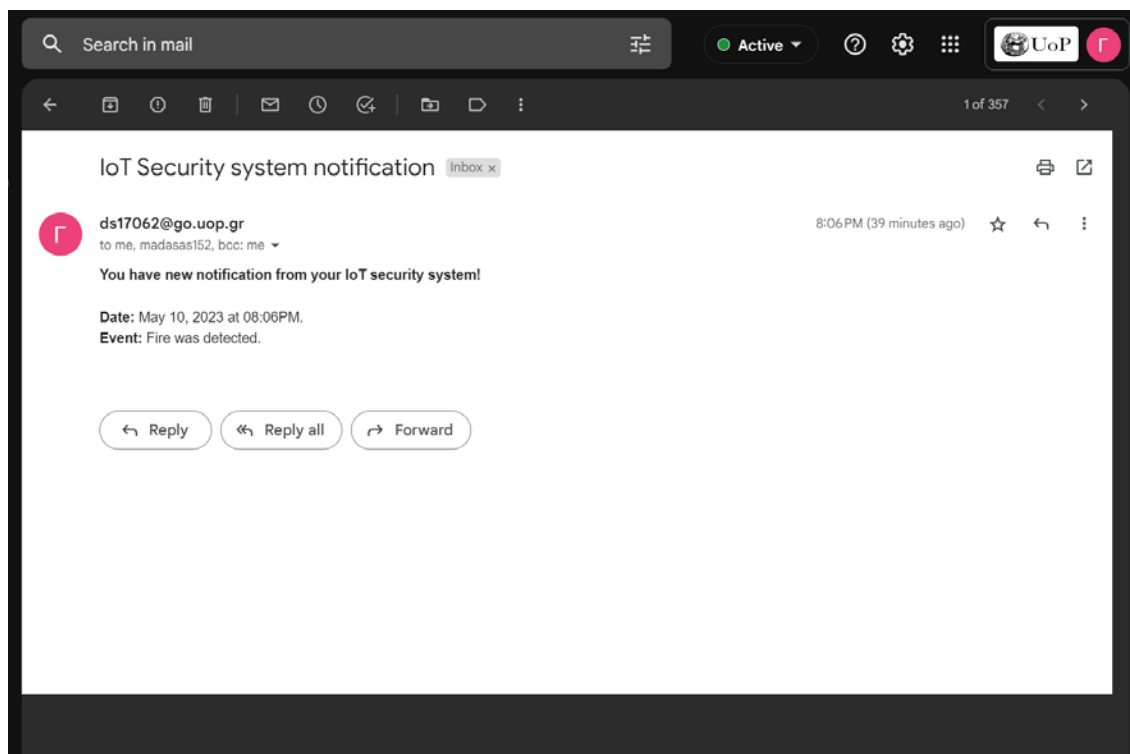
Εικόνα 3.3.1: IoT σύστημα ασφαλείας, διαδικτυακή εφαρμογή σε έξυπνο τηλέφωνο



Εικόνα 3.3.2: IoT σύστημα ασφαλείας, διαδικτυακή εφαρμογή σε υπολογιστή

Στις παραπάνω εικόνες απεικονίζεται η εφαρμογή που κατασκευάστηκε να τρέχει μέσω ενός φυλομετρητή σε ένα έξυπνο τηλέφωνο με λειτουργικό σύστημα Android και σε ένα υπολογιστικό σύστημα με εγκατεστημένο λειτουργικό Windows. Στο αριστερό μέρος βρίσκονται οι επιλογές με τα ονόματα των αισθητήρων που τους αντιστοιχούν. Το Water Sensor αντιστοιχεί στο σύστημα ανίχνευσης διαρροής υγρών, Flame Sensor για το σύστημα ανίχνευσης φωτιάς, Motion Sensor για το σύστημα ανίχνευσης κίνησης και Door Sensor που αντιστοιχεί στο σύστημα που αναγνωρίζει την κατάσταση μιας πόρτας. Την στιγμή που τραβήχτηκε η φωτογραφία οι αισθητήρας ανίχνευσης διαρροής υγρών, ανίχνευσης κίνησης και ανίχνευσης κατάστασης πόρτας έχουν απενεργοποιηθεί από τον χρήστη ενώ ο αισθητήρας ανίχνευσης φωτιάς είναι ενεργοποιημένος και έτοιμος να ανιχνεύσει μια πιθανή πυρκαγιά.

Στην δεξιά πλευρά της εφαρμογής βρίσκετε ο πίνακας με όλες τις ειδοποιήσεις του συστήματος ασφαλείας. Οι ειδοποιήσεις περιλαμβάνουν γεγονότα όπως την ακριβές χρονική περίοδο που ένας αισθητήρας ενεργοποιήθηκε ή απενεργοποιήθηκε και το αν κάποιος αισθητήρας έχει ανιχνεύσει κάποια δραστηριότητα. Αν υπάρξει κάποια καινούρια ειδοποίηση στο σύστημα τότε σχεδόν ακαριαία εμφανίζεται στον πίνακα ειδοποιήσεων της εφαρμογής και ταυτόχρονα αποστέλλεται στο ηλεκτρονικό ταχυδρομείο του χρήστη σε μορφή γραπτού email.

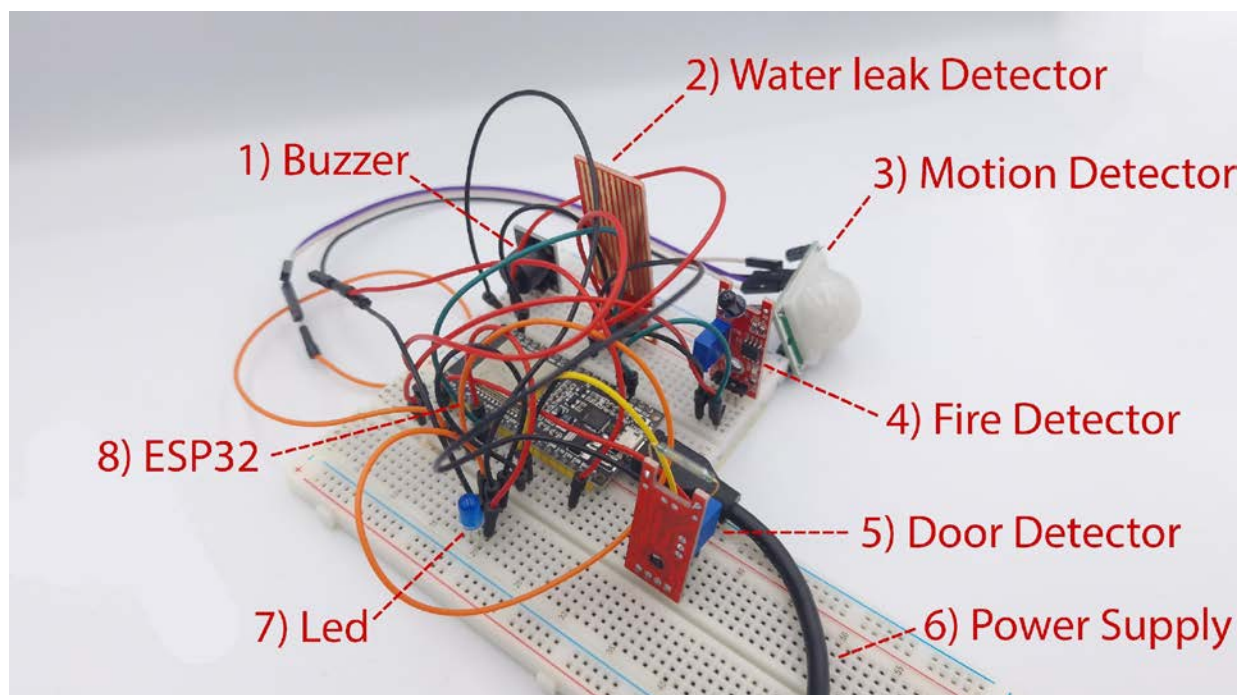


Εικόνα 3.3.3: IoT σύστημα ασφαλείας, email ειδοποιήσεις.

Η παρακάτω εικόνα απεικονίζει την κεντρική πλακέτα με όλους τους αισθητήρες συνδεδεμένους, πάνω σε δύο πλακέτες δοκιμών Breadboard. Κάθε εξάρτημα του κυκλώματος έχει αριθμηθεί με σκοπό την καλύτερη κατανόηση του συστήματος.

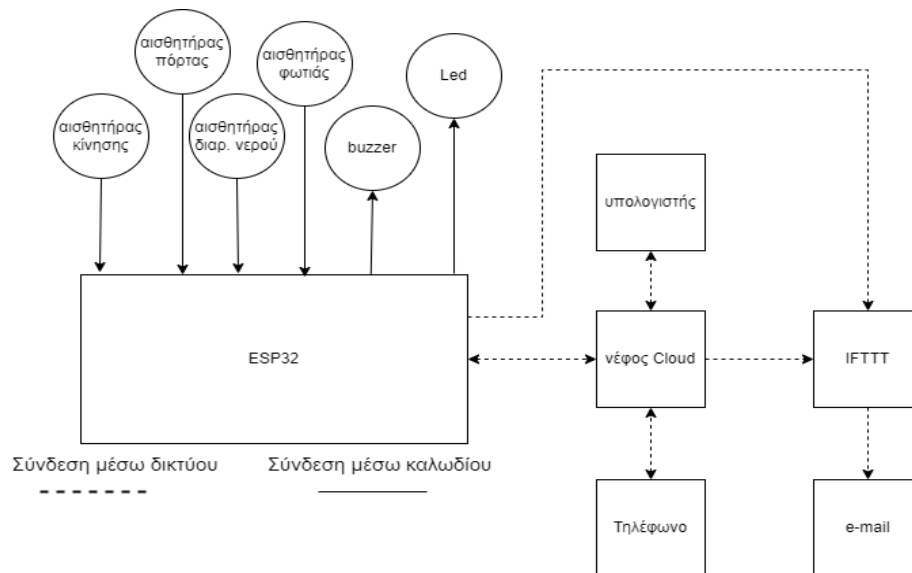


Πιο αναλυτικά στο 1) βρίσκεται το ηχείο του συναγερμού που ενεργοποιείται όταν υπάρξει μια ανίχνευση από κάποιον αισθητήρα. Στο 2) απεικονίζεται ο ανιχνευτής διαρροής υγρών που ενεργοποιείται όταν έρθει σε επαφή με κάποιο υγρό όπως το νερό. Το 3) αντιστοιχεί στον ανιχνευτή κίνησης που σαρώνει τον χώρο προκειμένου να διαπιστώσει κάποια κίνηση. Στο 4) βρίσκεται ο ανιχνευτής φωτιάς και με την βοήθεια του δέκτη που έχει ενσωματωμένο είναι ικανό να ανιχνεύσει την φλόγα. Το 5) απεικονίζει τον αισθητήρα που ανιχνεύει την κατάσταση μιας πόρτας. Ο συγκεκριμένος αισθητήρας για να λειτουργήσει ως ανιχνευτής πόρτας θα πρέπει να τοποθετηθεί πάνω στην κάσα της επιθυμητής πόρτας και στην συνέχεια να προστεθεί ένας απλός μαγνήτης πάνω στην πόρτα. Με αυτόν τον τρόπο όταν η πόρτα είναι κλειστή τότε ο μαγνήτης θα πλησιάσει τον αισθητήρα και έτσι το σύστημα θα καταλάβει ότι η πόρτα είναι κλειστή. Ενώ όταν η πόρτα θα ανοίξει τότε απομακρύνεται ο μαγνήτης από τον αισθητήρα και έτσι εμφανίζεται στον χρήστη η ειδοποίηση πως η συγκεκριμένη πόρτα είναι ανοικτή. Το 6) αφορά την πυγή ρεύματος που χρειάζεται το σύστημα για να λειτουργήσει και αυτό ανέρχεται στα 5 με 12 Volt. Στο 7) απεικονίζεται η λυχνία Led που ενεργοποιείται όταν υπάρξει κάποια αλλαγή στο σύστημα ενώ στο 8) βρίσκεται η κεντρική πλακέτα ESP32 που είναι ο εγkéφαλος του συστήματος και είναι υπεύθυνος για την επεξεργασία των δεδομένων που λαμβάνει από όλους τους αισθητήρες.



Εικόνα 3.3.4: IoT σύστημα ασφαλείας, συνδεδεμένο κύκλωμα.

Στο επακόλουθο διάγραμμα απεικονίζονται όλα τα συνδεδεμένα στοιχεία του IoT συστήματος ασφαλείας που κατασκευάστηκε και η μεταξύ τους επικοινωνία. Αναλυτικότερα οι αισθητήρες κίνησης, ανίχνευσης πόρτας, διαρροής νερού και φωτιάς που είναι συνδεδεμένοι στην πλακέτα ESP32 μέσω καλωδίου μεταφέρουν δεδομένα από το περιβάλλον στην πλακέτα ενώ η λυχνία Led και ο ηχείος buzzer λαμβάνουν δεδομένα από την πλακέτα. Η ESP32 στέλνει και λαμβάνει δεδομένα από το νέφος Cloud μέσω δικτύου και με την σειρά του το νέφος επικοινωνεί με τον υπολογιστή, με το έξυπνο τηλέφωνο. Τέλος η υπηρεσία IFTTT λαμβάνει δεδομένα απευθείας από την πλακέτα αλλά και από το νέφος μέσω δικτύου. Στην συνέχεια στέλνει της κατάλληλες πληροφορίες στο ηλεκτρονικό ταχυδρομείο του χρήστη.



Διάγραμμα στοιχείων 3.3.5: IoT σύστημα ασφαλείας, διάγραμμα στοιχείων.

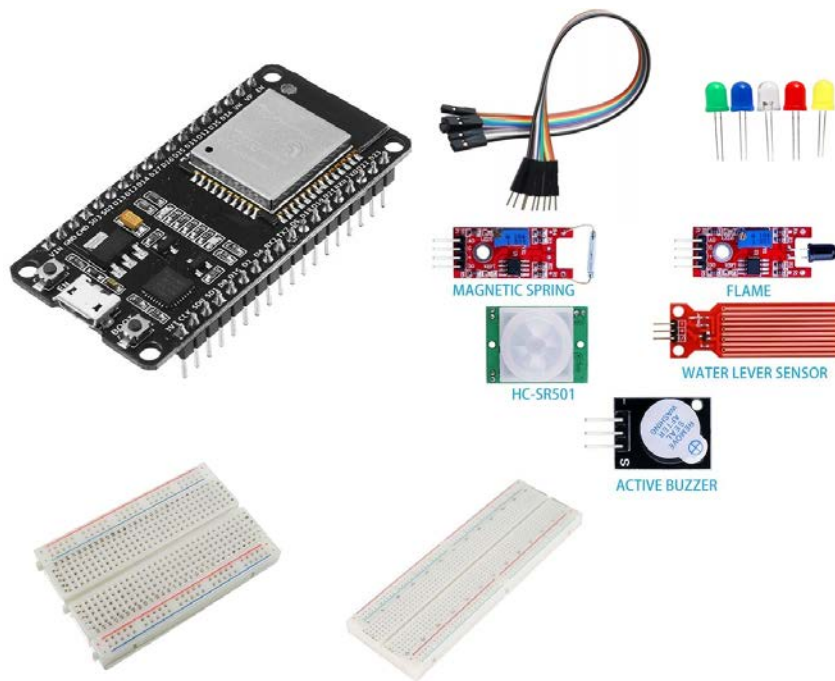
### 3.4 Υλικά και κόστος κατασκευής

Προκυμμένον να κατασκευαστεί το IoT έξυπνο σύστημα ασφαλείας που αναφέρεται στην προηγούμενη ενότητα χρησιμοποιήθηκε ένας αριθμός από αισθητήρες, καλώδια, φωτάκια Led, πλακέτες και διάτρητες πλακέτες. Όλοι οι αισθητήρες που χρησιμοποιήθηκαν κατασκευάζονται από την Κινέζικη εταιρία κολοσσό Elegoo. Συγκεκριμένα για το σύστημα ανίχνευσης φωτιάς χρησιμοποιήθηκε ο αισθητήρας Elegoo Flame sensor που έχει κόστος αγοράς 7 ευρώ. Ο βασικός λόγος που επιλέχθηκε ο Elegoo Flame sensor είναι γιατί είναι απλός στον προγραμματισμό και υπάρχει μεγάλο απόθεμα στην Ελληνική αγορά. Για την κατασκευή του ανιχνευτή διαρροής υγρών χρησιμοποιήθηκε ο αισθητήρας Elegoo Water level sensor και κοστολογείται στα 6 ευρώ. Ο Elegoo Water level sensor είναι κατασκευασμένος για να ανιχνεύει την πληρότητα ενός δοχείου που περιέχει κάποιο υγρό και όχι για την ανίχνευση υγρών όπως έχει χρησιμοποιηθεί στο συγκεκριμένο έξυπνο σύστημα ασφαλείας.

Ωστόσο ο λόγος που έχει χρησιμοποιηθεί ο συγκεκριμένος αισθητήρας είναι γιατί είναι από τους ελάχιστους αισθητήρες αυτού του είδους που διανέμονται σε Ελληνικά καταστήματα. Ο Elegoo Water level sensor έχει προγραμματιστεί με τον κατάλληλο τρόπο ώστε να λειτουργήσει ως ανιχνευτής υγρών και όχι έτσι όπως έχει ορίσει η κατασκευάστρια εταιρία. Προκειμένου να υλοποιηθεί ο ανιχνευτής πόρτας χρησιμοποιήθηκε ένας Reed αισθητήρας και συγκεκριμένα ο Magnetic spring της Elegoo και κοστολογείται στα 6 ευρώ. Καθώς και ένας συμβατικός μαγνήτης που χρησιμοποιήθηκε για την ενεργοποίηση και απενεργοποίηση του συγκεκριμένου αισθητήρα. Για την κατασκευή του συστήματος ανίχνευσης κίνησης χρησιμοποιήθηκε ο αισθητήρας κίνησης HC-SR501 PIR Motion sensor. Ο συγκεκριμένος αισθητήρας είναι ένας παθητικός αισθητήρας υπέρυθρων. Το κόστος αγοράς του HC-SR501 PIR Motion sensor είναι στα 10 ευρώ. Η επιλογή του HC-SR501 PIR Motion sensor όπως και του προηγούμενου αισθητήρα δεν έγινε με κάποιο συγκεκριμένο κριτήριο καθώς υπάρχει μεγάλο απόθεμα στην Ελληνική αγορά και όλα μοιράζονται παρόμοιες ιδιότητες. Εκτός από τους αισθητήρες αγοράστηκαν και χρησιμοποιήθηκαν δύο πλακέτες δοκιμών Breadboard διαφορετικών διαστάσεων. Μια μικρή και μια μεγάλη. Η μικρή πλακέτα κοστολογείται στα 5 ευρώ ενώ η μεγάλη στα 10 ευρώ. Βασικό ρόλο στην κατασκευή του IoT συστήματος ασφαλείας έπαιξαν τα καλώδια Jumper, λυχνίες Led ένδειξης ειδοποιήσεων και ένα μικρό μεμονωμένο ηχείο Elegoo Passive buzzer που χρησιμοποιήθηκε ως ακουστική ένδειξη σειρήνας.

Το κόστος των καλωδίων, των λυχνιών και του ηχείου ορίζεται στα 5 ευρώ. Τέλος η κεντρική πλακέτα που χρησιμοποιήθηκε είναι η ESP32 DEVKIT 1 που κατασκευάζεται από την Espressif Systems και κοστολογείται στα 10 ευρώ. Γιατί όμως χρησιμοποιήθηκε πλακέτα ESP32 και όχι μια πλακέτα Arduino που είναι ευρέως γνωστή;

Ο βασικότερος λόγος είναι πως το Esp32 φέρει ενσωματωμένη κάρτα δικτύου που εξυπηρετεί στην διασύνδεση της πλακέτας με δίκτυο. Το Arduino προκειμένου να συνδεθεί στο δίκτυο είναι απαραίτητο να προσαρμοστεί ξεχωριστεί κάρτα δικτύου. Το συνολικό κόστος για αγορά όλων των υποσυστημάτων που αναφέρθηκαν πλησιάζει το ποσό των 60 ευρώ. Το κόστος αγοράς και εγκατάστασης ενός έτοιμου συστήματος ασφαλείας που προσφέρει παρόμοιες λειτουργίες ανέρχεται περίπου στα 200 με 400 ευρώ.



Εικόνα 3.4.1: Υλικά κατασκευής.

### 3.5 Προγράμματα που χρησιμοποιήθηκαν

Τα προγράμματα που χρησιμοποιήθηκαν για την σχεδίαση και ολοκλήρωση του έργου παρέχονται δωρεάν από τις επίσημες ιστοσελίδες της εκάστοτε εταιρίας. Για την σωστή οργάνωση ολόκληρης της υλοποίησης χρησιμοποιήθηκαν οι διαδικτυακές εφαρμογές Google Drive και Trello.

Για την προσομοίωση της συμβατότητας και της συνδεσμολογίας των υλικών χρησιμοποιήθηκε η δικτυακή εφαρμογή Tinkercad. Για τον σχεδιασμό του λογισμικού και την σύνδεση του κώδικα στην πλακέτα ESP32 χρησιμοποιήθηκαν οι δικτυακές εφαρμογές Arduino IoT cloud και IFTTS καθώς και το πρόγραμμα Arduino Client Agent. Τέλος για την συγγραφή κώδικα χρησιμοποιήθηκαν τα προγράμματα notepad++ και Arduino IDE.

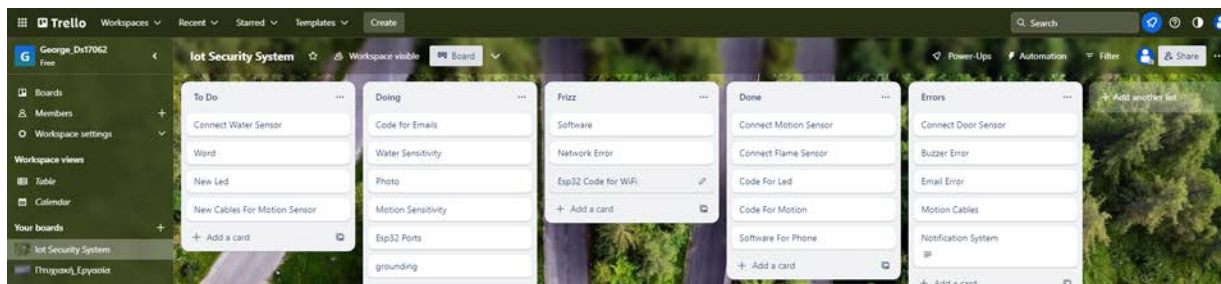
Το Google drive είναι μια αρκετά γνωστή υπηρεσία νέφους cloud αποθήκευσης και συγχρονισμού αρχείων που κυκλοφόρησε από την Google το έτος 2012. Με το Google drive οι χρήστες μπορούν να αποθηκεύουν τα δεδομένα τους με την ασφάλεια της Google στο Cloud και να αποκτούν πρόσβαση σε αυτά από οποιαδήποτε συσκευή με μια σύνδεση στο δίκτυο και την χρήση ενός λογαριασμού. Επιπλέον, δίνεται η δυνατότητα στους χρήστες να διαμοιράσουν τα αρχεία που επιθυμούν σε τρίτα άτομα σε πραγματικό χρόνο. Η υπηρεσία είναι διαθέσιμη δωρεάν με περιορισμένο αποθηκευτικό χώρο, ενώ υπάρχουν επίσης επιλογές πληρωμής με μεγαλύτερο αποθηκευτικό χώρο και επιπλέον χαρακτηριστικά. Το Google drive έπαιξε σημαντικό ρόλο στην διαχείριση δεδομένων. Συγκεκριμένο επιλέχθηκε γιατί παρέχεται η δυνατότητα της μεταφοράς και επεξεργασίας δεδομένων από οποιαδήποτε συσκευή χωρίς ο χρήστης να περιορίζεται σε έναν συγκεκριμένο υπολογιστή ή σε ένα συγκεκριμένο τηλέφωνο.

Συγκεκριμένα στο Google Drive αποθηκευόντουσαν κομμάτια κώδικα που βρισκόντουσαν ακόμα υπό ανάπτυξη και στην συλλογή ψηφιακών δεδομένων.



Εικόνα 3.5.1: Google Drive

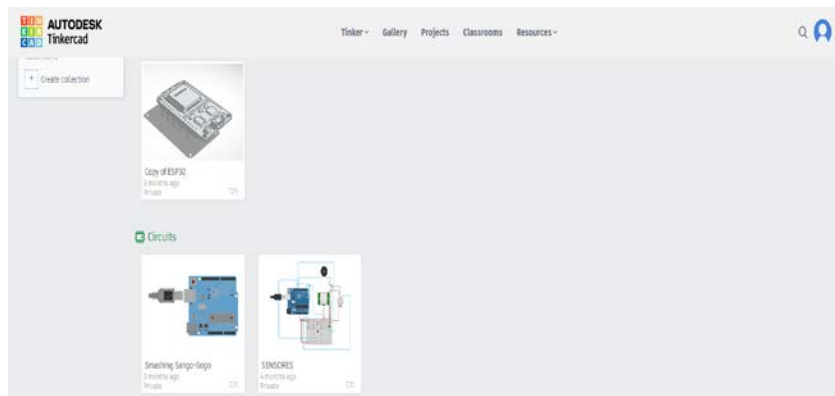
Το Trello είναι ένα πρόγραμμα διαχείρισης έργων και εργασιών που αναπτύχθηκε από την Trello Enterprise το 2011. Είναι μια διαδικτυακή εφαρμογή που βασίζεται στο σύστημα καρτελών και πινάκων για τη διαχείριση και την οργάνωση διαφόρων εργασιών και ενεργειών. Με το Trello, οι χρήστες μπορούν να δημιουργήσουν και να οργανώσουν θεματικούς πίνακες που στο εσωτερικό τους περιέχουν διεργασίες και δεδομένα από το εκάστοτε έργο. Επίσης είναι ευέλικτο, καθώς υποστηρίζει συνεργατική εργασία και διαμοιρασμό καρτελών με άλλους χρήστες. Η εφαρμογή είναι διαθέσιμη δωρεάν για απλή χρήση, αλλά υπάρχουν επίσης επιλογές πληρωμής για επιπλέον χαρακτηριστικά και λειτουργίες. Το Trello υπήρξε πολύ χρήσιμο στην καταγραφή σφαλμάτων από κομμάτια κώδικα αλλά και σε προβλήματα συνδεσμολογίας και ασυμβατότητας του υλικού. Η δημιουργία των πινάκων μέσω της εφαρμογής Trello έγινε με τρόπο ώστε να υπάρχει μια καλύτερη οργάνωση σφαλμάτων για να είναι ευκολότερο να επιλυθούν.



Εικόνα 3.5.2: Trello

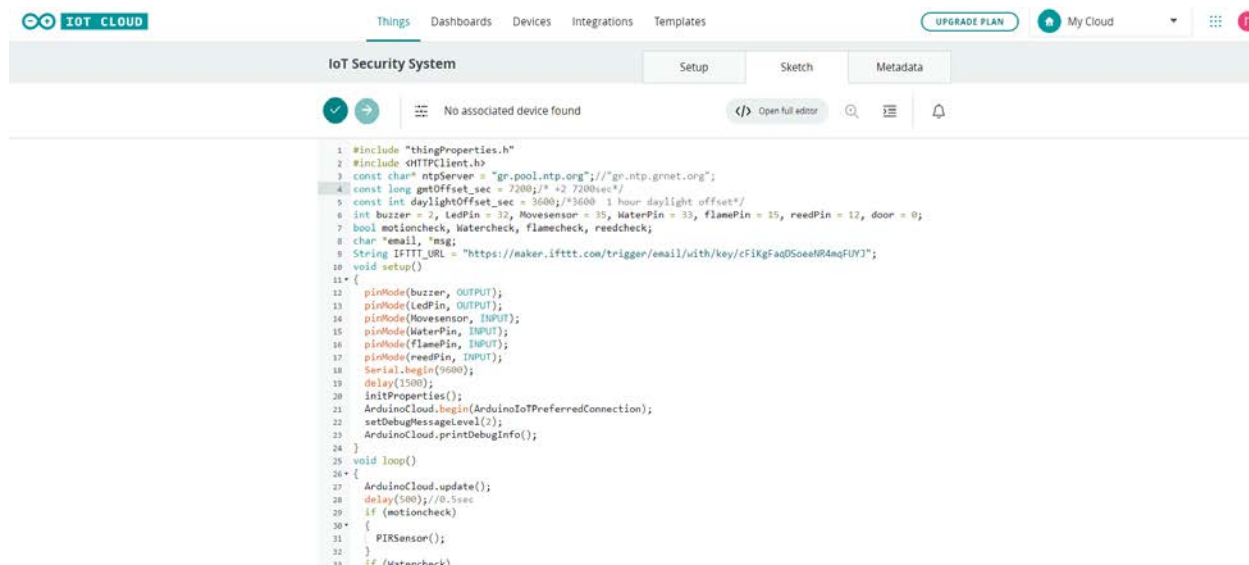
Πριν πραγματοποιηθεί η διαδικασία της αγοράς αισθητήρων, πλακέτας και την διαδικασία της συνδεσμολογίας τους ήταν αναγκαίο να χρησιμοποιηθεί ένας προσομοιωτής κυκλώματος ώστε να αποφευχθούν ανεπιθύμητες καταστάσεις όπως η βραχυκύκλωση και η ασυμβατότητα υλικού. Το Tinkercad είναι ένα διαδικτυακό εργαλείο σχεδίασης τρισδιάστατων μοντέλων που κατασκευάστηκε από την γνωστή εταιρία Autodesk το 2011. Πρόκειται για ένα εύκολο εργαλείο στην χρήση που επιτρέπει στους χρήστες να σχεδιάζουν αντικείμενα σε τρεις διαστάσεις, χρησιμοποιώντας πρότυπα σχήματα και εξειδικευμένα εργαλεία. Ωστόσο, το Tinkercad δεν περιορίζεται μόνο στο σχεδιασμό τρισδιάστατων μοντέλων. Οι χρήστες μπορούν επίσης να χρησιμοποιήσουν την εφαρμογή για τη δημιουργία ηλεκτρολογικών κυκλωμάτων, χρησιμοποιώντας ειδικά εργαλεία που παρέχονται από την ίδια εταιρία. Τα εργαλεία αυτά περιλαμβάνουν διάφορα είδη αντιστάσεων, πηνίων, συνδέσμων, διακοπών, ολοκληρωμένων κυκλωμάτων καθώς και εξειδικευμένες ρυθμίσεις για τη δημιουργία λογικών πυλών και κυκλωμάτων. Η παρούσα εφαρμογή χρησιμοποιήθηκε ως προσομοιωτής της συνδεσμολογίας του IoT συστήματος ασφαλείας που κατασκευάστηκε επειδή δεν απαιτεί κάποια εγκατάσταση και η χρήση του μπορεί να γίνει από οποιαδήποτε συσκευή.





Εικόνα 3.5.3: Tinkercad

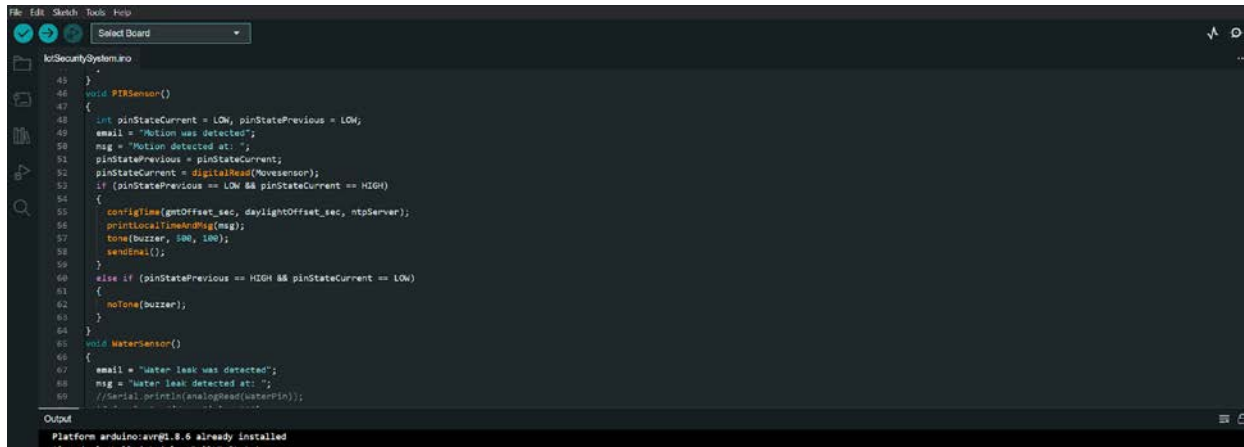
Το Arduino IoT Cloud είναι μια υπηρεσία νέφους cloud που κατασκευάστηκε Ιταλία από τους Massimo Banzi και David Cueartielles το 2005. Πρόκειται για μια διαδικτυακή εφαρμογή που επιτρέπει στους χρήστες να συνδέσουν πλακέτες Arduino αλλά και άλλων εταιριών στο νέφους cloud με αποτέλεσμα να προγραμματίζονται και να ελέγχονται απομακρυσμένα μέσω διαδικτύου. Το Arduino IoT Cloud υποστηρίζει επίσης τη δημιουργία εφαρμογών IoT και μπορούν να συνδεθούν απομακρυσμένα με την εκάστοτε πλακέτα που οι χρήστες επιθυμούν να προγραμματίσουν. Η δημιουργία των εφαρμογών γίνεται χρησιμοποιώντας μια εύχρηστη διεπαφή χρήστη που παρέχεται από την εταιρία Arduino. Η δημιουργία της εφαρμογής που συνοδεύει το έξυπνο σύστημα ασφαλείας προγραμματίστηκε και κατασκευάστηκε εξ ολοκλήρου από το Arduino IoT Cloud καθώς μέσω αυτής πραγματοποιήθηκε η σύνδεση της πλακέτας ESP32 με το νέφος cloud, ο βασικός προγραμματισμός της και η σύνδεση με μηχανισμούς όπως το webhook για την αποστολή των ειδοποιήσεων σε μορφή email.



Εικόνα 3.5.4: Arduino IoT Cloud

Ένα εναλλακτικό πρόγραμμα που χρησιμοποιήθηκε για την συγγραφή του κώδικα είναι το Arduino IDE που όπως και το Arduino IoT Cloud σχεδιάστηκε και κατασκευάστηκε στην Ιταλία από τους Massimo Banzi και David Cueartielles το 2005. Το Arduino IDE Integrated Development Environment είναι μια πλατφόρμα ανοιχτού κώδικα που χρησιμοποιείται για την ανάπτυξη και τον προγραμματισμό των πλακετών Arduino καθώς και άλλων πλακετών που βασίζονται στην κατασκευαστική λογική των πλακετών Arduino. Το πρόγραμμα αυτό προσφέρει έναν εύχρηστο περιβάλλον ανάπτυξης, το οποίο επιτρέπει στους χρήστες να γράφουν, να μεταγλωττίζουν και να τον μεταφορτώνουν κώδικα στις πλακέτες τους. Το πρόγραμμα περιλαμβάνει έναν πλούσιο επεξεργαστή κειμένου με πολλές δυνατότητες όπως τονισμό σύνταξης, αυτόματη συμπλήρωση κώδικα και αναδίπλωση κώδικα.

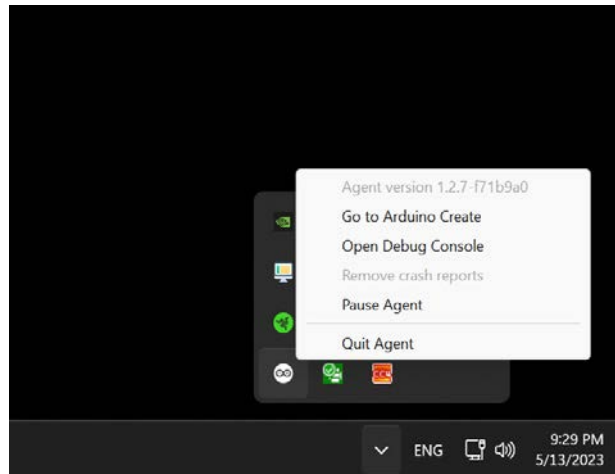
Επίσης, παρέχει την δυνατότητα να προστεθούν εξωτερικές βιβλιοθήκες καθώς ενσωματώνει και έναν προσομοιωτή για την δοκιμή του κώδικα χωρίς τη χρήση πραγματικού υλικού. Η γραφή κώδικα στο περιβάλλον Arduino IDE βασίζεται στην γλώσσα προγραμματισμού C/C++.



Εικόνα 3.5.5: Arduino IDE

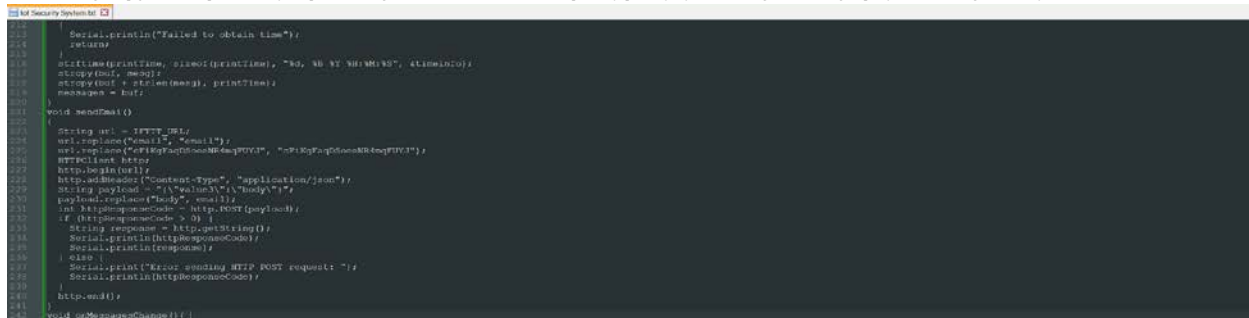
Το Arduino Client είναι ένα πρόγραμμα λογισμικού που χρησιμοποιείται από την δικτυακή εφαρμογή Arduino IoT Cloud αλλά και από το πρόγραμμα Arduino IDE. Κατασκευάστηκε από τους δημιουργούς του Arduino και χρησιμοποιείται για τη σύνδεση και αναγνώριση των πλακετών σε ένα υπολογιστικό σύστημα. Χωρίς την χρήση του Arduino Client Agent είναι αδύνατον οι εφαρμογές και τα προγράμματα ενός υπολογιστή να αναγνωρίσουν τις πλακέτες όπως η ESP32 που χρησιμοποιήθηκε στο έξυπνο σύστημα ασφαλείας. Είναι ο βασικός μεσάζοντας για την αναγνώριση και μεταφορά κώδικα και βιβλιοθηκών από ένα πρόγραμμα σε μια πλακέτα. Επιπλέον, το Arduino Client επιτρέπει την παρακολούθηση της σειριακής επικοινωνίας με την πλακέτα, κάτι που μπορεί να βοηθήσει στη διάγνωση προβλημάτων και στην αποσφαλμάτωση του κώδικα.

Το Arduino Client Agent είναι συμβατό με Windows, Mac OS X και Linux και είναι διαθέσιμο δωρεάν για λήψη από τον επίσημο ιστότοπο.



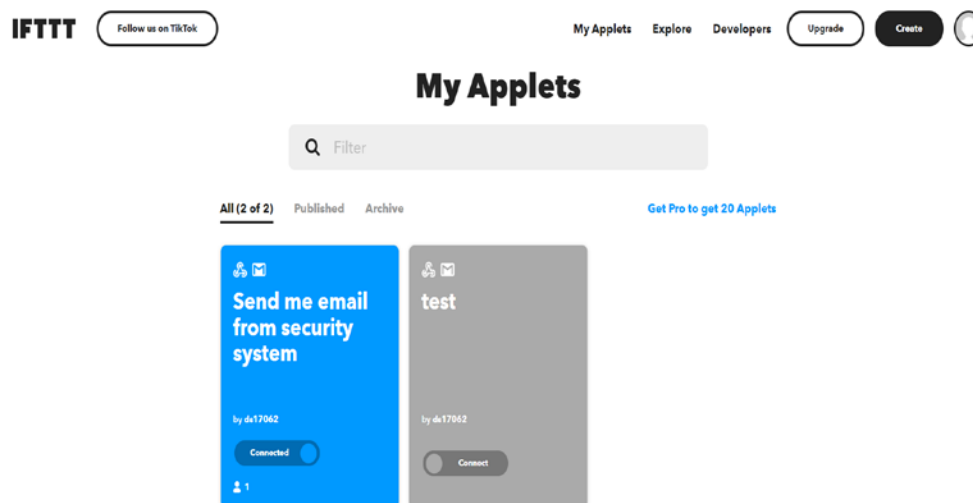
Εικόνα 3.5.6: Arduino Client Agent

Ένα ακόμη εναλλακτικό πρόγραμμα που χρησιμοποιήθηκε για την συγγραφή του κώδικα είναι το Notepad++. Είναι ένα πρόγραμμα επεξεργασίας ελευθέρου κειμένου και κώδικα που αναπτύχθηκε το 2003 από τον Don Ho. Υποστηρίζει και αναγνωρίζει πολλές γλώσσες προγραμματισμού, καθώς παρέχει πληθώρα χρήσιμων λειτουργιών, όπως αυτόματη συμπλήρωση κώδικα, αλλαγή κωδικοποίησης, αναδιαμόρφωση κώδικα, απομακρυσμένη επεξεργασία αρχείων, μετατροπή χαρακτήρων ASCII σε δεκαδικό σύστημα γραφής και άλλα. Είναι απλό στην χρήση καθώς έχει την δομή ενός κλασικού προγράμματος επεξεργασίας κειμένου.



Εικόνα 3.5.7: Notepad++

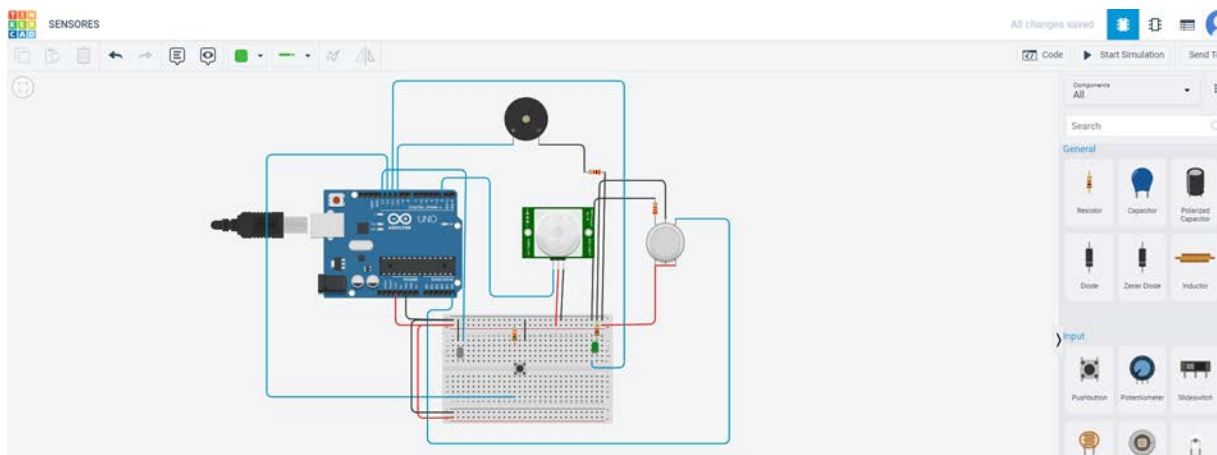
Το IFTTT If This Then That είναι μια ιδιωτική εμπορική εταιρεία που ιδρύθηκε το 2011. Η εταιρεία παρέχει μια διαδικτυακή εφαρμογή που επιτρέπει στους χρήστες να συνδέουν υπηρεσίες ιστού και συσκευές IoT μεταξύ τους, χρησιμοποιώντας ένα σύστημα εφαρμογίδιο συντακτικών κανόνων γνωστό ως Applets. Με το IFTTT, οι χρήστες μπορούν να δημιουργήσουν διάφορες αυτοματοποιημένες διαδικασίες Applets με βάση συγκεκριμένων συνθηκών που ορίζονται από αυτούς, όπως η ώρα, οι καιρικές συνθήκες, η εισερχόμενη αλληλογραφία, οι εντολές φωνητικού ελέγχου ή μια συνθήκη από κάποιον κώδικα. Για παράδειγμα, μπορεί να δημιουργηθεί ένα Applet που θα σας ειδοποιεί τον χρήστη με μήνυμα στο κινητό όταν υπάρχει ένα email που δεν έχει διαβαστεί από ένα συγκεκριμένο αποστολέα. Το IFTTT είναι ένα πολύ χρήσιμο εργαλείο για την αυτοματοποίηση καθημερινών εργασιών και την εξοικονόμηση χρόνου και ενέργειας. Στο συγκεκριμένο IoT σύστημα ασφαλείας η διαδικτυακή εφαρμογή IFTTT χρησιμοποιήθηκε για την αποστολή των ειδοποιήσεων στον χρήστη σε μορφή ηλεκτρονικού ταχυδρομείου email.



Εικόνα 3.5.8: IFTTT

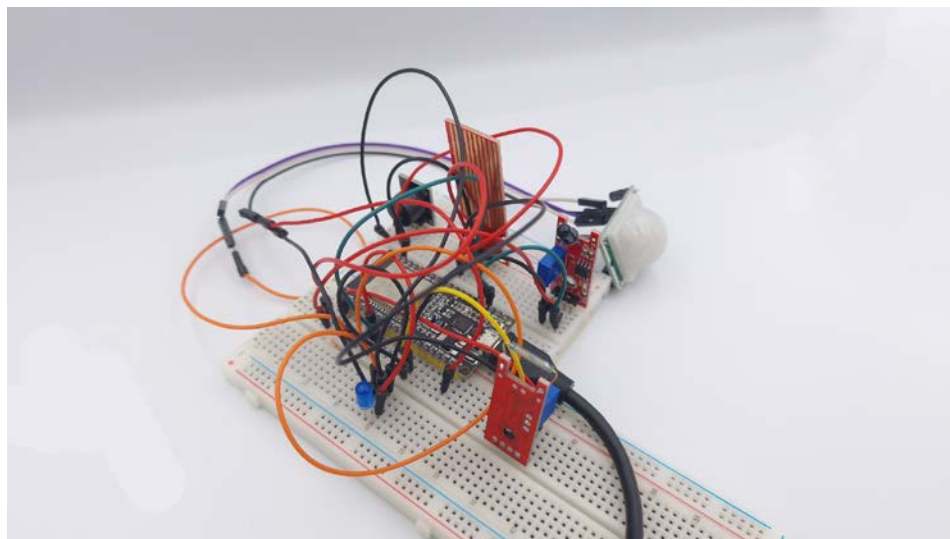
### 3.6 Τρόπος κατασκευής κυκλώματος

Κατά την έναρξη της κατασκευής του κυκλώματος είναι σημαντικό να έχει παρθεί η απόφαση για το τι λειτουργίες θα ενσωματώνει το έξυπνο σύστημα ασφαλείας. Στην προκειμένη περίπτωση η λειτουργίες που επιθυμούμε να έχει το σύστημα είναι να μπορεί να ανιχνεύει τις κινήσεις σε έναν χώρο, να παρέχει ασφάλεια πυρκαγιάς και διαρροής υγρών, να ανιχνεύει την κατάσταση μιας πόρτας και τέλος θέλουμε να έχει ένα τρόπο με τον οποίο θα ιδιοποιείται άμεσα ο χρήστης για κάποια ανίχνευση και αλλαγή στο σύστημα. Αυτό πραγματοποιήθηκε με την χρήση μιας λυχνίας Led και ενός μικρού ηχείου ώστε να υπάρχει οπτικοακουστική ιδιοποίηση. Με αυτόν τον τρόπο γνωρίζουμε πως θα χρειαστεί να προμηθευτούμε έναν αισθητήρα κίνησης, έναν αισθητήρα ανίχνευσης φωτιάς, έναν για την ανίχνευση διαρροής υγρών, έναν μαγνητικός αισθητήρας για την ανίχνευση της κατάστασης μιας πόρτας, μια λυχνία Led, ένα μικρό ηχείο, αρκετά καλώδια Jumper για την συνδεσμολογία των αισθητήρων, μια ή δύο διάτρητες δοκιμαστικές πλακέτες ώστε να πατήσει ολόκληρο το κύκλωμα και τέλος την κεντρική πλακέτα μικροηλεκτρονικών που θα παίζει τον ρόλο του εγκεφάλου του συστήματος. Πριν την πραγματοποίηση της έρευνα αγοράς για τα συγκεκριμένα υποσυστήματα χρειάστηκε να πραγματοποιηθεί μια προσομοίωση του κυκλώματος μέσω ενός προγράμματος όπως είναι το Tinkercad. Με αυτόν τον τρόπο εξασφαλίζεται πως όλα τα υποσυστήματα είναι συμβατά μεταξύ τους και πως η πλακέτα μικροηλεκτρονικών που θα χρησιμοποιηθεί ταιριάζει στις ανάγκες του συστήματος. Η παρακάτω εικόνα απεικονίζει την αρχική προσομοίωση της συνδεσμολογίας του κυκλώματος χρησιμοποιώντας το Tinkercad.



Εικόνα 3.6.1: Αρχικό σχέδιο Tinkercad

Στην συνέχεια πραγματοποιήθηκε η έρευνα αγοράς και η προμήθεια όλων των υποσυστημάτων μέσω διαδικτύου. Μόλις παραλήφθηκαν τα υποσυστήματα ξεκίνησε η διαδικασία της συνδεσμολογίας των αισθητήρων πάνω στην πλακέτα. Για κάθε αισθητήρα που συνδεόταν στην πλακέτα πραγματοποιούνταν έλεγχος ώστε να επιβεβαιωθεί πως έχει συνδεθεί σωστά. Οι έλεγχοι γινόντουσαν με την χρήση ενός δοκιμαστικού κώδικα και μιας λυχνίας Led. Όταν ένας αισθητήρας λάμβανε κάποιο ερέθισμα από το περιβάλλον τότε ενεργοποιούνταν η λυχνία Led και στην οθόνη του υπολογιστή παρέχονταν οι πληροφορίες της κατάστασης του αισθητήρα. Η παραπάνω διαδικασία είχε διάρκεια αρκετών εβδομάδων καθώς απαιτούνταν η σχολαστική μελέτη για κάθε αισθητήρα ξεχωριστά καθώς ο καθένας από αυτούς έχει τον δικό του τρόπο λειτουργίας και έναν μοναδικό τρόπο προγραμματισμού. Αυτό είχε σαν αποτέλεσμα να δημιουργείται κάθε φορά καινούριος κώδικας προκειμένου να πραγματοποιείται ο έλεγχος της σωστής λειτουργίας σε κάθε αισθητήρα. Σε περίπτωση που ένα συνδεδεμένο υποσύστημα δεν λειτουργούσε σωστά έπρεπε να μελετηθεί από την αρχή η συνδεσμολογία του και ο αντίστοιχος κώδικας του. Με αυτόν τον τρόπο το τελικό αποτέλεσμα του κυκλώματος παρουσιάζεται στην επακόλουθη εικόνα.



Εικόνα 3.6.2: Τελική μορφή κυκλώματος

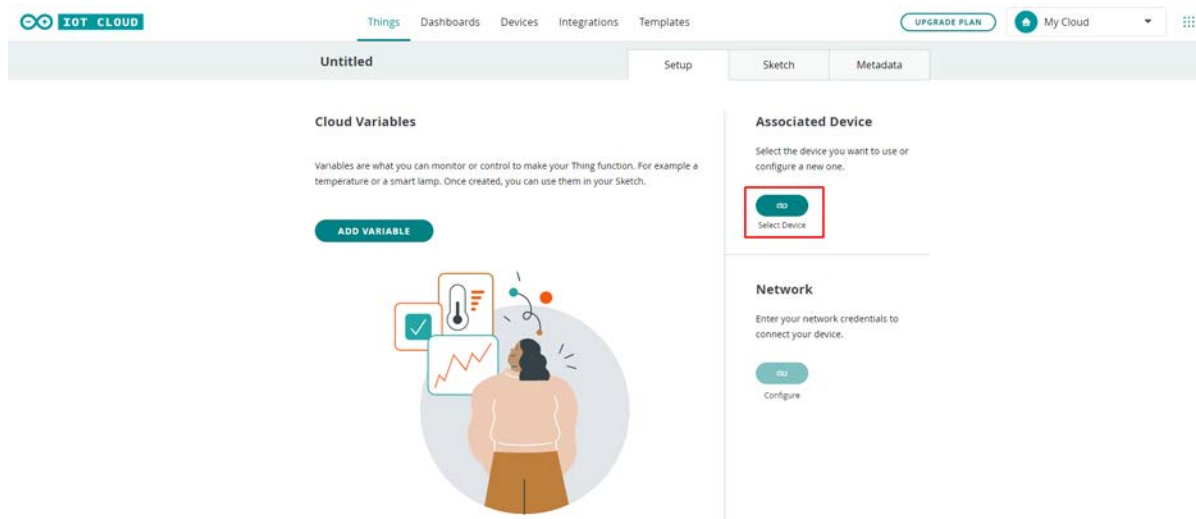
### 3.7 Λογισμικό και κώδικας

Προκειμένου να κατασκευαστή το λογισμικό του συστήματος και ο κώδικας του κατά κύρια βάση χρησιμοποιήθηκε η διαδικτυακή εφαρμογή Arduino IoT Cloud που αναφέρθηκε στην ενότητα με τα προγράμματα που χρησιμοποιήθηκαν. Παρακάτω αναλύεται η διαδικασία της κατασκευής του λογισμικού και της συγγραφής κώδικα, χρησιμοποιώντας εξολοκλήρου την εφαρμογή Arduino IoT Cloud, καθώς τα υπόληπτα προγράμματα που αναφέρθηκαν χρησιμοποιήθηκαν ως βοηθητικά και όχι ως το κύριο πρόγραμμα για την ολοκληρωτική υλοποίηση του συστήματος που κατασκευάστηκε. Τέλος θα ακολουθήσουν και θα επεξηγηθούν μερικά αποσπάσματα από την τελική μορφή του κώδικα που απαρτίζουν το IoT σύστημα ασφαλείας.

Η διαδικασία ξεκινά κάνοντας μια εγγραφή στην διαδικτυακή πλατφόρμα Arduino IoT Cloud. Στην συνέχεια είναι σημαντικό να πραγματοποιηθεί η προσθήκη της πλακέτα μικροηλεκτρονικών EPS32 με το περιβάλλον του Arduino IoT Cloud,



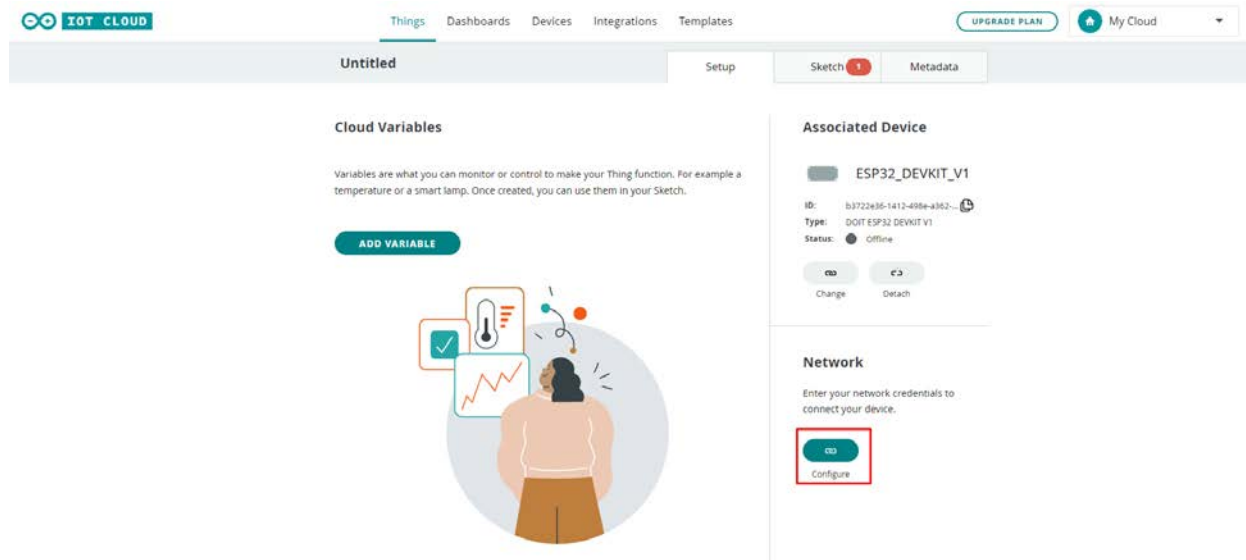
και να γίνει η σύνδεση της με το δίκτυο του σπιτιού πριν ακόμα ξεκινήσει η διαδικασία της συγγραφής του κώδικα. Με αυτόν τον τρόπο προβλέπονται μελλοντικά προβλήματα που θα προκύψουν κατά την μεταφοράς του κώδικα στην πλακέτα. Η διαδικασία σύνδεσης της πλακέτας με το Arduino IoT Cloud και το οικιακό δίκτυο πραγματοποιείται συνδέοντας αρχικά την πλακέτα με τον υπολογιστή χρησιμοποιώντας ένα καλώδιο μεταφοράς δεδομένων USB. Στην συνέχεια κάνοντας κλικ στην επιλογή Create a Thing που εμφανίζεται στην αρχική οθόνη και τέλος επιλέγοντας το κουμπί Select Device που εμφανίζεται στην δεξιά περιοχή του προγράμματος.



Εικόνα 3.7.1: Arduino IoT Cloud Select Device

Πατώντας το παραπάνω κουμπί εμφανίζεται ένα καινούριο παράθυρο στην οθόνη που περιέχονται διάφορα μοντέλα των υποστηριζόμενων πλακετών. Μόλις πραγματοποιηθεί η επιλογή της πλακέτας που στην παρούσα κατάσταση είναι η ESP32 DEVKIT V1, καλείτε να πραγματοποιηθεί η αποθηκεύσει σε ένα μέρος του υπολογιστή το μυστικό κλειδί και ο κωδικό ταυτοποίησης που θα εμφανιστούν στην οθόνη.

Τέλος προκειμένου να συνδεθεί η πλακέτα μικροηλεκτρονικών στο δίκτυο θα χρειαστεί να επιλεγεί το ονομαζόμενο κουμπί Configure που θα εμφανιστεί μόλις πραγματοποιηθεί η διαδικασία της προθήκης πλακέτας στο σύστημα.

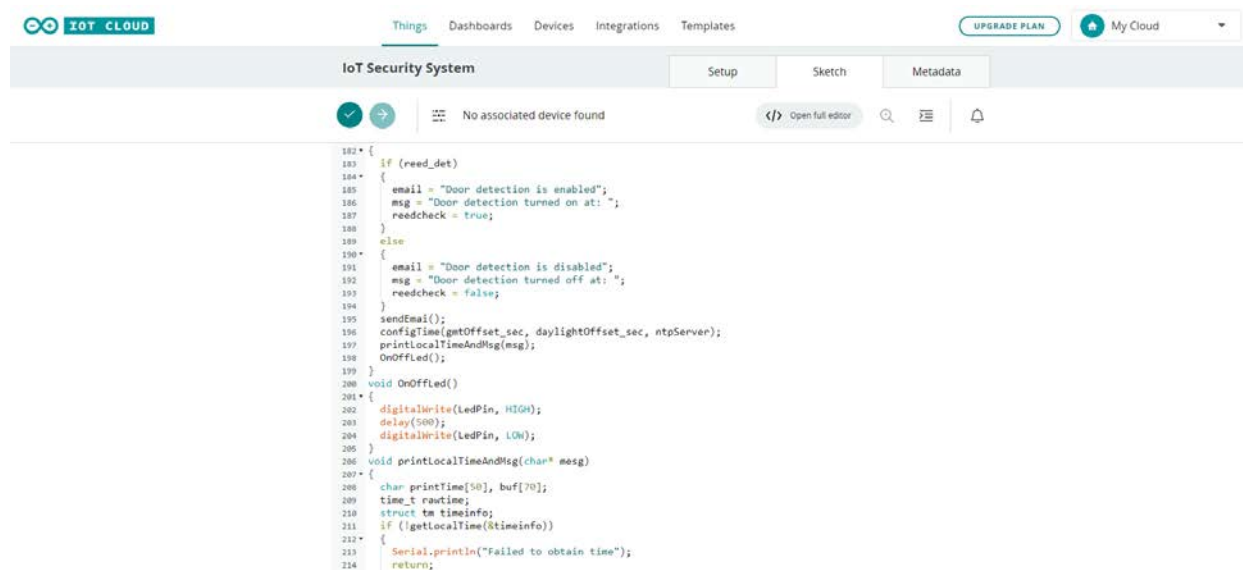


Εικόνα 3.7.2: Arduino IoT Cloud Configure

Μέσο ενός νέου παραθύρου που πρόκειται να εμφανιστεί στην οθόνη δίνεται η επιλογή της προσθήκης του ονόματος και του κωδικού του δικτύου μαζί με το μυστικό κλειδί της πλακέτας που του είχε ζητηθεί να αποθηκεύσει.

Αφού ολοκληρώθηκε η διαδικασία της σύνδεσης του δικτύου ξεκίνησε η συγγραφή του κώδικα. Στο περιεχόμενο της δεύτερης καρτέλας του Arduino IoT Cloud που φέρει το όνομα Sketch βρίσκεται ένα περιβάλλον στο οποίο πραγματοποιείται η σύνταξη του κώδικα και η μεταφορά του στην επιθυμητή πλακέτα. Πιο συγκεκριμένα ενσωματώνει έναν επεξεργαστή κειμένου, μια σειρά από βιβλιοθήκες και εργαλεία για την σύνταξη κώδικα.

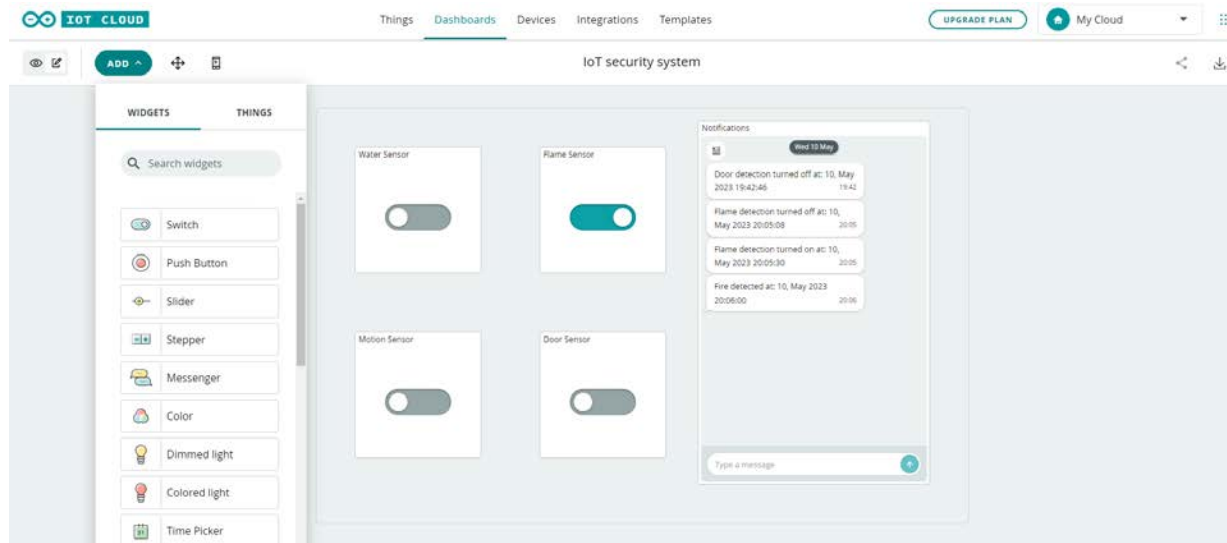
Η γλώσσα προγραμματισμού που υποστηρίζεται από το σύστημα είναι η Arduino η οποία βασίζεται στην γνωστή C/C++. Τέλος βρίσκεται η επιλογή Verify που επιβεβαιώνει την ορθότητα του κώδικα και η επιλογή Upload που πραγματοποιεί την μεταφορά του κώδικα στην πλακέτα μικροηλεκτρονικών.



```
182 * {
183   if (reed_det)
184   {
185     email = "Door detection is enabled";
186     msg = "Door detection turned on at: ";
187     reedcheck = true;
188   }
189   else
190   {
191     email = "Door detection is disabled";
192     msg = "Door detection turned off at: ";
193     reedcheck = false;
194   }
195   sendEmail();
196   configTime(getOffset_sec, daylightOffset_sec, ntpServer);
197   printLocalTimeAndMsg(msg);
198   OnOffLed();
199 }
200 void OnOffLed()
201 {
202   digitalWrite(LedPin, HIGH);
203   delay(500);
204   digitalWrite(LedPin, LOW);
205 }
206 void printLocalTimeAndMsg(char* msg)
207 {
208   char printTime[50], buf[70];
209   time_t rawtime;
210   struct tm timeinfo;
211   if (!getLocalTime(&timeinfo))
212   {
213     Serial.println("Failed to obtain time");
214     return;
215   }
```

Εικόνα 3.7.3: Arduino IoT Cloud Sketch

Κατά το τελευταίο στάδιο της υλοποίησης κατασκευάστηκε το λογισμικό που συνοδεύει το IoT σύστημα ασφαλείας μέσω της ενότητας dashboards που παρέχεται από το Arduino IoT Cloud. Εκεί βρίσκεται ένα περιβάλλον στο οποίο προσφέρετε η δυνατότητα της δημιουργίας διαφορετικών πινάκων ελέγχου για τις συσκευές που είναι συνδεδεμένες στο IoT Cloud. Συγκεκριμένα ο χρήστης μπορεί να δημιουργήσει διάφορα στοιχεία ελέγχου, όπως κουμπιά, διακόπτες, γραφήματα, δείκτες, ετικέτες οργάνωσης και άλλα.



Εικόνα 3.7.4: Arduino IoT Cloud Dashboards

Το κάθε στοιχείο που προστίθεται στο γραφικό περιβάλλον dashboards προγραμματίζεται στην καρτέλα Sketch που αναφέρθηκε παραπάνω. Ωστόσο προκειμένου να προγραμματιστεί ένα γραφικό στοιχείο είναι απαραίτητο να δημιουργηθεί μια μοναδική μεταβλητή που συνοδεύει και περιγράφει το εκάστοτε γραφικό στοιχείο. Αυτό το είδος των μεταβλητών η Arduino το ονομάζει μεταβλητές νέφους Cloud Variables. Ορίζονται και αρχικοποιούνται στην καρτέλα Setup που αποτελεί την κεντρική καρτέλα της εφαρμογής Arduino IoT Cloud, ενώ η σύνδεση των μεταβλητών με το αντίστοιχο γραφειακό στοιχείο πραγματοποιείται από τις ιδιότητες των εκάστοτε στοιχείων στην ενότητα Dashboards.

Things
Dashboards
Devices
Integrations
Templates

UPGRADE PLAN

IoT Security System

Setup

Sketch

Metadata

Cloud Variables

ADD

Name ↓	Last Value	Last Update
<input type="checkbox"/> <b>flame_det</b> bool flame_det;	true	10 May 2023 20:05:40
<input type="checkbox"/> <b>messages</b> String messages;	Fire detected at: 10,...	10 May 2023 20:06:02
<input type="checkbox"/> <b>motion_det</b> bool motion_det;	false	10 May 2023 19:42:56
<input type="checkbox"/> <b>reed_det</b> bool reed_det;	false	10 May 2023 19:42:56
<input type="checkbox"/> <b>water_det</b> bool water_det;	false	10 May 2023 19:42:56

Associated Device

iot

ID: 61c126fc-2723-4934-889f-7...

Type: DOIT ESP32 DEVKIT V1

Status: Offline

Change

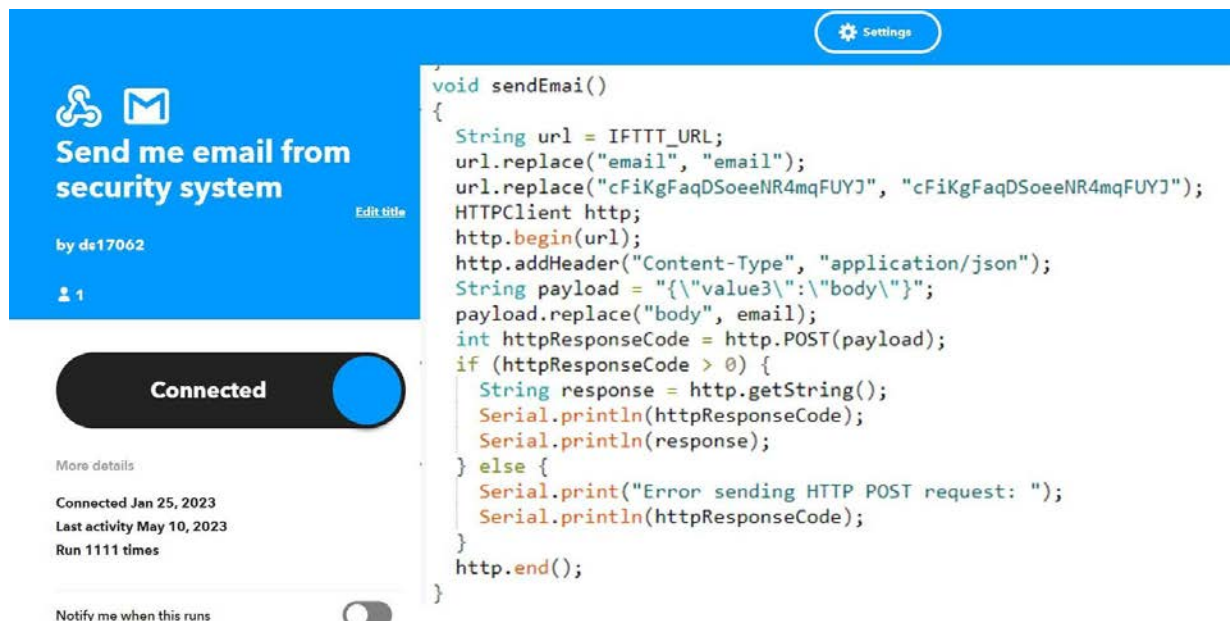
Detach

Network

Εικόνα 3.7.5: Arduino IoT Cloud Cloud Variables

Μετά την ολοκλήρωση των γραφικών στοιχείων και την συγγραφή ενός μεγάλου κομματιού του κώδικα, προκειμένου να υλοποιηθεί η διαδικασία αποστολής ιδιοποιήσεων μέσω email χρησιμοποιήθηκε η διαδικτυακή υπηρεσία IFTT If This Then That. Αυτή η υπηρεσία επιτρέπει στους χρήστες να δημιουργούν αυτοματισμούς μεταξύ διαφορετικών υπηρεσιών και συσκευών. Οι χρήστες μπορούν να συνδέουν διάφορες εφαρμογές, όπως ηλεκτρονικό ταχυδρομείο, μέσα κοινωνικής δικτύωσης, φωτογραφίες, και αισθητήρες για να δημιουργήσουν αυτοματισμούς που εκτελούν δράσεις αυτόματα. Στην περίπτωση του συστήματος ασφαλείας που κατασκευάστηκε η υπηρεσία IFTTT χρησιμοποιήθηκε για την δημιουργία ενός σεναρίου αυτοματισμού που ενεργοποιεί αυτόματα με ένα συγκεκριμένο ερέθισμα που λαμβάνει μέσω του κώδικα του συστήματος ασφαλείας. Αυτός ο αυτοματισμός μόλις εκτελεστεί στέλνει ένα μήνυμα στο ηλεκτρονικό ταχυδρομείο σε μια προκαθορισμένη διεύθυνση email με τα περιεχόμενα πολλών μεταβλητών που βρίσκονται στον κεντρικό κώδικα.

Αυτές οι μεταβλητές ανανεώνονται συνεχώς και περιλαμβάνουν δεδομένα όπως η τρέχουσα ώρα, η κατάσταση των αισθητήρων και τα δεδομένα μιας ανίχνευσης του συστήματος ασφαλείας. Η υπηρεσία IFTTT είναι άμεσα συνδεδεμένη με τον κώδικα και εκτελείται αλάνθαστα και ακαριαία.



Εικόνα 3.7.6: IFTTT Code

Κατά την έναρξη της συγγραφής του κώδικα είναι απαραίτητο να δημιουργηθεί η συνάρτηση με όνομα `Setup()`. Η συνάρτηση `Setup()` τοποθετείται πάντα στην αρχή του κώδικα κάτω από το σημείο που πραγματοποιείται η δήλωση μεταβλητών. Σκοπός της συνάρτησης είναι η αρχικοποίηση των ακροδεκτών της πλακέτας, η αρχικοποίηση εξωτερικών βιβλιοθηκών σε περίπτωση που αυτό είναι απαραίτητο, η ρύθμιση της αρχικής κατάστασης της πλακέτας που θα ενσωματώσει τον κώδικα και τέλος η αρχικοποίηση μιας πιθανής επικοινωνίας όπως η σύνδεση ενός δικτύου.

Η συνάρτηση `Setup()` που απεικονίζεται στην επόμενη εικόνα κατασκευάστηκε και χρησιμοποιείται στο IoT σύστημα ασφαλείας. Συγκεκριμένα στο εσωτερικό της συνάρτησης χρησιμοποιείται η εντολή `pinMode` για την αρχικοποίηση των ακροδεκτών της πλακέτα. Η `pinMode` χρησιμοποιεί δύο διαφορετικά ορίσματα. Στο πρώτο όρισμα αποθηκεύεται ο αριθμός του ακροδέκτη που έχει συνδεθεί ένα υποσύστημα, ενώ στο δεύτερο όρισμα δηλώνετε ο τύπος επικοινωνίας του εκάστοτε υποσυστήματος. Στην προκειμένη περίπτωση δημιουργείται η μεταβλητή με το όνομα `buzzer` και αποθηκεύετε ο αριθμός 2. Στην συνέχεια με την εντολή `pinMode(buzzer, OUTPUT)` ο κώδικας καταλαβαίνει πως το `buzzer` δεν είναι μια απλή μεταβλητή αλλά αντιπροσωπεύει τον ακροδέκτη με αριθμό 2 που βρίσκεται στο ESP32. Στον ακροδέκτη με νούμερο 2 είναι συνδεδεμένος ο ένας από τους τρεις πόλους του μεμονωμένου ηχείου που χρησιμοποιείται για να λαμβάνει δεδομένα από την πλακέτα. Με το όρισμα `OUTPUT` ο κώδικας αναγνωρίζει πως στον ακροδέκτη με αριθμό 2 του ESP32 θα χρησιμοποιηθεί ως έξοδος και όχι ως είσοδος δεδομένων. Το ηχείο λαμβάνει πληροφορίες από το σύστημα όπως το πότε να ήχησε, τον χρόνο που θα είναι ενεργοποιημένο καθώς και την ένταση του. Ωστόσο με την εντολή `pinMode(MoveSensor, INPUT)` ο αλγόριθμός καταλαβαίνει πως ο ακροδέκτης με αριθμό 35, που περιέχεται στο εσωτερικό της μεταβλητής `MoveSensor`, θα χρησιμοποιηθεί ως είσοδος. Ο αισθητήρας κίνησης που έχει τοποθετηθεί στον ακροδέκτη 35 θα στείλει της πληροφορίες ανίχνευσης στην πλακέτα EPS32.

Η εντολή `Serial.begin(9600)` που περιέχεται στην συνάρτηση `Setup()` χρησιμοποιείται για την έναρξη μια συριακής επικοινωνίας της πλακέτας με τον υπολογιστή με αριθμό μετάδοσης 9600. Αυτή η εντολή βοηθάει τον προγραμματιστή να παρακολουθεί την λειτουργία του κώδικα σε πραγματικό χρόνο μέσω του υπολογιστή που είναι συνδεδεμένη η πλακέτα. Η `Serial.begin(9600)` δεν επηρεάζει το αποτέλεσμα της λειτουργίας του κώδικα.

Άλλη μια σημαντική εντολή που βρίσκεται στην `Setup()` είναι η `Delay(1500)`. Μόλις φτάσει η στιγμή της εκτέλεσης του κώδικα `Delay(1500)` ο αλγόριθμός θα κάνει μια παύση αξίας 1,5 δευτερολέπτων. Αυτή η εντολή χρησιμοποιείται για να γίνει μια μικρή προκαθορισμένη καθυστέρηση της εκτέλεσης του κώδικα.

Στις τέσσερις τελευταίες γραμμές κώδικα στο εσωτερικό της `Setup()` που απεικονίζονται στην παρακάτω εικόνα καλούνται τέσσερις διαφορετικές συνάρτησης που αρχικοποιούν την κατάσταση της πλακέτας. Οι συναρτήσεις αυτές καλούνται από εξωτερικές βιβλιοθήκες και όχι από συναρτήσεις που έχουν γραφεί στον κώδικα. Η συνάρτηση `initProperties()` χρησιμοποιείται για την αρχικοποίηση των ιδιοτήτων της πλακέτας ESP32 στα πλαίσια του Arduino IoT Cloud. Η `arduinoCloud.begin(ArduinoIoTPreferredConnection)` χρησιμοποιείται για να ξεκινήσει η σύνδεση της πλακέτας ESP32 με το Arduino IoT Cloud. Η `setDebugMessageLevel(2)` χρησιμοποιείται για τον καθορισμό του επιπέδου αναφοράς αποσφαλμάτωσης. Στην περίπτωση αυτή, ορίζεται το επίπεδο 2, που σημαίνει ότι θα εμφανίζονται αναφορές αποσφαλμάτωσης με μέτριο επίπεδο λεπτομέρειας. Τέλος η συνάρτηση `ArduinoCloud.printDebugInfo()` χρησιμοποιείται για την εκτύπωση πληροφοριών αποσφαλμάτωσης σχετικά με τη σύνδεση της διαδικτυακής εφαρμογής Arduino IoT Cloud.



```

const char* ntpServer = "gr.pool.ntp.org";//"gr.ntp.grnet.org";
const long gmtOffset_sec = 7200; /* +2 7200sec*/
const int daylightOffset_sec = 3600; /*3600 1 hour daylight offset*/
int buzzer = 2, LedPin = 32, Movesensor = 35, WaterPin = 33, flamePin = 15, reedPin = 12, door = 0;
bool motioncheck, Watercheck, flamecheck, reedcheck;
char *email, *msg;
String IFTTT_URL = "https://maker.ifttt.com/trigger/email/with/key/cFiKgFaqDSoeeNR4mqFUYJ";

void setup()
{
  pinMode(buzzer, OUTPUT);
  pinMode(LedPin, OUTPUT);
  pinMode(Movesensor, INPUT);
  pinMode(WaterPin, INPUT);
  pinMode(flamePin, INPUT);
  pinMode(reedPin, INPUT);
  Serial.begin(9600);
  delay(1500);
  initProperties();
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

```

Εικόνα 3.7.7: Συνάρτηση Setup()

Η συνάρτηση Loop() χρησιμοποιείται και αυτή από κάθε κώδικα που απευθύνετε σε πλακέτες μικροηλεκτρονικών. Οι γραμμές που βρίσκονται στο εσωτερικό της συνάρτησης Loop() εκτελούνται συνεχώς και ασταμάτητα. Η συνάρτηση σταματά να εκτελείτε μόνο όταν το πρόγραμμα τερματιστεί ή αν η πλακέτα αποσυνδεθεί από το ρεύμα. Παρακάτω απεικονίζετε ο κώδικας που βρίσκεται στο εσωτερικό της Loop() του συστήματος ασφαλείας που κατασκευάστηκε. Συγκεκριμένα έχει προστεθεί ο κώδικας που χρειάζεται να εκτελείται ασταμάτητα. Όπως ο έλεγχος για το πότε ένας αισθητήρας έχει ενεργοποιηθεί μέσω του γραφικού περιβάλλοντος του λογισμικού που αναπτύχθηκε και η άμεση αποστολή των δεδομένων από τους αισθητήρες στην πλακέτα EPS32.

```
void loop()
{
  ArduinoCloud.update();
  delay(500); // 0.5sec
  if (motioncheck)
  {
    PIRSensor();
  }
  if (Watercheck)
  {
    WaterSensor();
  }
  if (flamecheck)
  {
    flameSensor();
  }
  if (reedcheck)
  {
    reedSensor();
  }
}
```

Εικόνα 3.7.8: Συνάρτηση Loop()

Στο συνοδευτικό αρχείο με όνομα IoT\_Security\_System\_Code βρίσκεται η τελική μορφή ολόκληρου του κώδικα του IoT συστήματος ασφαλείας που κατασκευάστηκε μαζί με τις μεταβλητές νέφος Cloud Variables που είναι άμεσα συνδεδεμένες με το γραφικό περιβάλλον.

## 3.8 Προβλήματα και περιορισμοί

Ένα από τα βασικότερα προβλήματα που προέκυψαν ήταν ο περιορισμένος χώρος που ενσωματώνει η πλακέτα ESP32. Συγκεκριμένα στην παρούσα πλακέτα ο κώδικας του χρήστη δεν θα πρέπει να ξεπερνάει το μέγεθος των 30720 bytes ή 30.72 kilobytes. Ο αρχικός κώδικας που κατασκευάστηκε για το IoT σύστημα ασφαλείας μαζί με όλες τις βιβλιοθήκες που χρησιμοποιήθηκαν ξεπερνούσε αρκετά το επιτρεπτό όριο των 30.72 kilobytes. Προκειμένου να αντιμετωπιστεί αυτό το πρόβλημα κάποια κομμάτια κώδικα αφαιρέθηκαν ενώ κάποια άλλα αναπρογραμματίστηκαν με τρόπο ώστε να λειτουργούν εξίσου καλά καταλαμβάνοντας λιγότερο χώρο στο σύστημα. Ένα τέτοιο παράδειγμα αποτελούν οι παρακάτω εικόνες.

```
/*-----Old-----*/
void reedSensor()
{
    int proximity = digitalRead(reedPin);
    if (proximity == LOW)
    {
        if (door == 2) {
            handleDoorClosed();
        } else if (door == 0) {
            handleDoorClosed();
        }
    } else if (proximity == HIGH) {
        if (door == 1) {
            handleDoorOpened();
        } else if (door == 0) {
            handleDoorOpened();
        }
    }
}

void handleDoorClosed() {
    msg = ("Door closed at: ");
    email = "The door closed";
    tone(buzzer, 500, 50); //100
    door = 1;
    sendEmail();
    configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
    printLocalTimeAndMsg(msg);
}

void handleDoorOpened() {
    email = "The door opened";
    msg = ("Door opened at: ");
    noTone(buzzer);
    door = 2;
    sendEmail();
    configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
    printLocalTimeAndMsg(msg);
}

/*-----New-----*/
void reedSensor()
{
    int proximity = digitalRead(reedPin);
    if ((proximity == LOW && door == 2) or (proximity == LOW && door == 0))
    {
        msg = ("Door closed at: ");
        email = "The door closed";
        tone(buzzer, 500, 50); //100
        door = 1;
        sendEmail();
        configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
        printLocalTimeAndMsg(msg);
    }
    else if ((proximity == HIGH && door == 1) or (proximity == HIGH && door == 0))
    {
        email = "The door opened";
        msg = ("Door opened at: ");
        noTone(buzzer);
        door = 2;
        sendEmail();
        configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
        printLocalTimeAndMsg(msg);
    }
}
```

Εικόνα 3.8.1: Βελτιστοποίηση κώδικα

Επίσης αφαιρέθηκαν οι περισσότερες εξωτερικές βιβλιοθήκες συναρτήσεων και αντικαταστάθηκαν με κώδικα που χρησιμοποιεί ενσωματωμένες βιβλιοθήκες της εταιρίας Arduino. Αφαιρέθηκαν οι περισσότερες καθολικές μεταβλητές Global και μετατράπηκαν σε τοπικές Local. Με αυτόν τον τρόπο οι μεταβλητές που έχουν οριστεί ως τοπικές χρησιμοποιούνται μόνο κατά την διάρκεια που εκτελείται η συναρτήσεις που έχουν οριστεί. Μόλις ολοκληρωθεί η συνάρτηση τότε αυτές οι μεταβλητές καταστρέφονται. Έτσι εξοικονομείται χώρος στο σύστημα και ο κώδικας εκτελείται με μεγαλύτερη ευκολία.

<pre>#include "thingProperties.h" #include &lt;HTTPClient.h&gt; int pinStateCurrent = LOW, pinStatePrevious = LOW; String email, msg;  /*-----Global variables-----*/ void PIRSensor() {     email = "Motion was detected";     msg = "Motion detected at: ";     pinStatePrevious = pinStateCurrent;     pinStateCurrent = digitalRead(Movesensor);     if (pinStatePrevious == LOW &amp;&amp; pinStateCurrent == HIGH)     {         configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);         printLocalTimeAndMsg(msg);         tone(buzzer, 500, 100);         sendEmail();     }     else if (pinStatePrevious == HIGH &amp;&amp; pinStateCurrent == LOW)     {         noTone(buzzer);     } }</pre>	<pre>#include "thingProperties.h" #include &lt;HTTPClient.h&gt;  /*-----Local variables-----*/ void PIRSensor() {     int pinStateCurrent = LOW, pinStatePrevious = LOW;     String email = "Motion was detected";     String msg = "Motion detected at: ";     pinStatePrevious = pinStateCurrent;     pinStateCurrent = digitalRead(Movesensor);     if (pinStatePrevious == LOW &amp;&amp; pinStateCurrent == HIGH)     {         configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);         printLocalTimeAndMsg(msg);         tone(buzzer, 500, 100);         sendEmail();     }     else if (pinStatePrevious == HIGH &amp;&amp; pinStateCurrent == LOW)     {         noTone(buzzer);     } }</pre>
---	---

Εικόνα 3.8.2: Global και Local μεταβλητές

Πλέον το μέγεθος του κώδικα ανέρχεται στο 90% του συνολικού χώρου της πλακέτας ESP32. Αρχικά το σύστημα ασφαλείας είχε κατασκευαστεί με περεταίρω λειτουργίες όπως ή επιλογή διαφορετικών γλωσσών, η αποστολή ειδοποιήσεων σε μορφή SMS, αυτόματες τηλεφωνικές κλήσεις έκτακτης ανάγκης. Ακόμη είχε ξεκινήσει η ανάπτυξη κώδικα για την υποστήριξη φωνητικών εντολών μέσω των Google API Services ώστε ο χρήστης να μπορεί να αλληλοεπιδράσει με το σύστημα ασφαλείας μέσω φωνητικών εντολών σε διάφορες γλώσσες, καθώς ενσωμάτωνε λειτουργίες μετάφρασης της Google.

Η παρακάτω εικόνα απεικονίζει ένα απόσπασμα από τον κώδικα που βρισκόταν στο στάδιο δοκιμών της παραπάνω τεχνολογίας προγραμματισμένο σε γλώσσα Python.

```

124 def speech_to_text(): # pip install audio
125     speech_client = speech.SpeechClient()
126     media_file_name_mp3 = "G:\My Drive\GoogleTextToSpeech\Test_mav_en.wav"
127     with open(media_file_name_mp3, "rb") as f:
128         bytes_data_mp3 = f.read()
129     audio_mp3 = speech.RecognitionAudio(content=bytes_data_mp3)
130     config_mp3 = speech.RecognitionConfig(audio_format=speech.AudioFormat.MP3,
131     sample_rate_hertz=48000,
132     enable_automatic_punctuation=True,
133     language_codes="en-US")
134     response_audio_mp3 = speech_client.recognize(
135     config=config_mp3, audio=audio_mp3)
136     auto_audio_mp3 =
137     ""
138     for result in response_audio_mp3.results:
139         print(result.alternatives[0].transcript)
140     print(audio_mp3)
141
142 def live_speech(): # pip install audio, read csv
143     audio_content = ps.audiocontent(mode="wav")
144     ps.showPlot(content, name="synthase.csv")
145     while True:
146         from audio.get()
147
148     speech_to_text()
149     #google_tts()
150     #live_speech()
151     #ask()
152     #live_tts()
153     #text_to_speech()
154     #auto_correct()
155     #text_tag()
156     #translation()
157
158     #live_speech()
159     #live_tts()
160     #live_speech()
161     #live_tts()
162     #live_speech()
163     #live_tts()
164     #live_speech()
165     #live_tts()
166     #live_speech()
167     #live_tts()
168     #live_speech()
169     #live_tts()
170     #live_speech()
171     #live_tts()
172     #live_speech()
173     #live_tts()
174     #live_speech()
175     #live_tts()
176     #live_speech()
177     #live_tts()
178     #live_speech()
179     #live_tts()
180     #live_speech()
181     #live_tts()
182     #live_speech()
183     #live_tts()
184     #live_speech()
185     #live_tts()
186     #live_speech()
187     #live_tts()
188     #live_speech()
189     #live_tts()
190     #live_speech()
191     #live_tts()
192     #live_speech()
193     #live_tts()
194     #live_speech()
195     #live_tts()
196     #live_speech()
197     #live_tts()
198     #live_speech()
199     #live_tts()
200     #live_speech()
201     #live_tts()
202     #live_speech()
203     #live_tts()
204     #live_speech()
205     #live_tts()
206     #live_speech()
207     #live_tts()
208     #live_speech()
209     #live_tts()
210     #live_speech()
211     #live_tts()
212     #live_speech()
213     #live_tts()
214     #live_speech()
215     #live_tts()
216     #live_speech()
217     #live_tts()
218     #live_speech()
219     #live_tts()
220     #live_speech()
221     #live_tts()
222     #live_speech()
223     #live_tts()
224     #live_speech()
225     #live_tts()
226     #live_speech()
227     #live_tts()
228     #live_speech()
229     #live_tts()
230     #live_speech()
231     #live_tts()
232     #live_speech()
233     #live_tts()
234     #live_speech()
235     #live_tts()
236     #live_speech()
237     #live_tts()
238     #live_speech()
239     #live_tts()
240     #live_speech()
241     #live_tts()
242     #live_speech()
243     #live_tts()
244     #live_speech()
245     #live_tts()
246     #live_speech()
247     #live_tts()
248     #live_speech()
249     #live_tts()
250     #live_speech()
251     #live_tts()
252     #live_speech()
253     #live_tts()
254     #live_speech()
255     #live_tts()
256     #live_speech()
257     #live_tts()
258     #live_speech()
259     #live_tts()
260     #live_speech()
261     #live_tts()
262     #live_speech()
263     #live_tts()
264     #live_speech()
265     #live_tts()
266     #live_speech()
267     #live_tts()
268     #live_speech()
269     #live_tts()
270     #live_speech()
271     #live_tts()
272     #live_speech()
273     #live_tts()
274     #live_speech()
275     #live_tts()
276     #live_speech()
277     #live_tts()
278     #live_speech()
279     #live_tts()
280     #live_speech()
281     #live_tts()
282     #live_speech()
283     #live_tts()
284     #live_speech()
285     #live_tts()
286     #live_speech()
287     #live_tts()
288     #live_speech()
289     #live_tts()
290     #live_speech()
291     #live_tts()
292     #live_speech()
293     #live_tts()
294     #live_speech()
295     #live_tts()
296     #live_speech()
297     #live_tts()
298     #live_speech()
299     #live_tts()
300     #live_speech()
301     #live_tts()
302     #live_speech()
303     #live_tts()
304     #live_speech()
305     #live_tts()
306     #live_speech()
307     #live_tts()
308     #live_speech()
309     #live_tts()
310     #live_speech()
311     #live_tts()
312     #live_speech()
313     #live_tts()
314     #live_speech()
315     #live_tts()
316     #live_speech()
317     #live_tts()
318     #live_speech()
319     #live_tts()
320     #live_speech()
321     #live_tts()
322     #live_speech()
323     #live_tts()
324     #live_speech()
325     #live_tts()
326     #live_speech()
327     #live_tts()
328     #live_speech()
329     #live_tts()
330     #live_speech()
331     #live_tts()
332     #live_speech()
333     #live_tts()
334     #live_speech()
335     #live_tts()
336     #live_speech()
337     #live_tts()
338     #live_speech()
339     #live_tts()
340     #live_speech()
341     #live_tts()
342     #live_speech()
343     #live_tts()
344     #live_speech()
345     #live_tts()
346     #live_speech()
347     #live_tts()
348     #live_speech()
349     #live_tts()
350     #live_speech()
351     #live_tts()
352     #live_speech()
353     #live_tts()
354     #live_speech()
355     #live_tts()
356     #live_speech()
357     #live_tts()
358     #live_speech()
359     #live_tts()
360     #live_speech()
361     #live_tts()
362     #live_speech()
363     #live_tts()
364     #live_speech()
365     #live_tts()
366     #live_speech()
367     #live_tts()
368     #live_speech()
369     #live_tts()
370     #live_speech()
371     #live_tts()
372     #live_speech()
373     #live_tts()
374     #live_speech()
375     #live_tts()
376     #live_speech()
377     #live_tts()
378     #live_speech()
379     #live_tts()
380     #live_speech()
381     #live_tts()
382     #live_speech()
383     #live_tts()
384     #live_speech()
385     #live_tts()
386     #live_speech()
387     #live_tts()
388     #live_speech()
389     #live_tts()
390     #live_speech()
391     #live_tts()
392     #live_speech()
393     #live_tts()
394     #live_speech()
395     #live_tts()
396     #live_speech()
397     #live_tts()
398     #live_speech()
399     #live_tts()
400     #live_speech()
401     #live_tts()
402     #live_speech()
403     #live_tts()
404     #live_speech()
405     #live_tts()
406     #live_speech()
407     #live_tts()
408     #live_speech()
409     #live_tts()
410     #live_speech()
411     #live_tts()
412     #live_speech()
413     #live_tts()
414     #live_speech()
415     #live_tts()
416     #live_speech()
417     #live_tts()
418     #live_speech()
419     #live_tts()
420     #live_speech()
421     #live_tts()
422     #live_speech()
423     #live_tts()
424     #live_speech()
425     #live_tts()
426     #live_speech()
427     #live_tts()
428     #live_speech()
429     #live_tts()
430     #live_speech()
431     #live_tts()
432     #live_speech()
433     #live_tts()
434     #live_speech()
435     #live_tts()
436     #live_speech()
437     #live_tts()
438     #live_speech()
439     #live_tts()
440     #live_speech()
441     #live_tts()
442     #live_speech()
443     #live_tts()
444     #live_speech()
445     #live_tts()
446     #live_speech()
447     #live_tts()
448     #live_speech()
449     #live_tts()
450     #live_speech()
451     #live_tts()
452     #live_speech()
453     #live_tts()
454     #live_speech()
455     #live_tts()
456     #live_speech()
457     #live_tts()
458     #live_speech()
459     #live_tts()
460     #live_speech()
461     #live_tts()
462     #live_speech()
463     #live_tts()
464     #live_speech()
465     #live_tts()
466     #live_speech()
467     #live_tts()
468     #live_speech()
469     #live_tts()
470     #live_speech()
471     #live_tts()
472     #live_speech()
473     #live_tts()
474     #live_speech()
475     #live_tts()
476     #live_speech()
477     #live_tts()
478     #live_speech()
479     #live_tts()
480     #live_speech()
481     #live_tts()
482     #live_speech()
483     #live_tts()
484     #live_speech()
485     #live_tts()
486     #live_speech()
487     #live_tts()
488     #live_speech()
489     #live_tts()
490     #live_speech()
491     #live_tts()
492     #live_speech()
493     #live_tts()
494     #live_speech()
495     #live_tts()
496     #live_speech()
497     #live_tts()
498     #live_speech()
499     #live_tts()
500     #live_speech()
501     #live_tts()
502     #live_speech()
503     #live_tts()
504     #live_speech()
505     #live_tts()
506     #live_speech()
507     #live_tts()
508     #live_speech()
509     #live_tts()
510     #live_speech()
511     #live_tts()
512     #live_speech()
513     #live_tts()
514     #live_speech()
515     #live_tts()
516     #live_speech()
517     #live_tts()
518     #live_speech()
519     #live_tts()
520     #live_speech()
521     #live_tts()
522     #live_speech()
523     #live_tts()
524     #live_speech()
525     #live_t
```

Εικόνα 3.8.3: Google API Services φωνητικές εντολές

Μια ακόμη λειτουργία που ακυρώθηκε κατά την ανάπτυξη της ήταν η δυνατότητα που δινόταν στον χρήστη να μπορεί να δημιουργήσει τα δικά του σενάρια αυτοματισμού και τα δικά του χρονοδιαγράμματα πάνω στο σύστημα ασφαλείας. Όπως να ενεργοποιούνται οι αισθητήρες μια συγκεκριμένη ώρα με προκαθορισμένη διάρκεια για όλες τις μέρες της εβδομάδας. Ένα άλλο σενάριο αυτοματισμού που θα μπορούσε να κατασκευαστεί από τον χρήστη είναι πως μόλις το σύστημα ανιχνεύσει την πόρτα να ανοίγει τότε αυτόματα θα γίνετε η ενεργοποίηση του αισθητήρα κίνησης και η έναρξη της καταγραφής κάποιας εξωτερικής κάμερας ασφαλείας. Δυστυχώς λόγω της μικρής χωρητικότητας του ESP32 οι ανάπτυξη αυτών των λειτουργιών σταμάτησε και δεν ενσωματώθηκαν στην τελική μορφή του συστήματος ασφαλείας που κατασκευάστηκε.

Άλλο ένα πρόβλημα που προέκυψε ήταν η συνδεσμολογία των αισθητήρων στην πλακέτα. Αυτό συνέβη γιατί η διαδικτυακή εφαρμογή Tinkercad που χρησιμοποιήθηκε ως προσομοιωτής κυκλώματος δεν περιλάμβανε τους ίδιους αισθητήρες που χρησιμοποιήθηκαν κατά την κατασκευή του IoT συστήματος ασφαλείας. Αυτό είχε σαν αποτέλεσμα σε μερικούς από τους αισθητήρες που αγοράστηκαν να ενσωματώνουν διαφορετικές επαφές σύνδεσης σε σχέση με τον αντίστοιχο αισθητήρα που παρέχει το Tinkercad. Η επίλυση του συγκεκριμένου προβλήματος ήταν απλή αλλά αρκετά χρονοβόρα καθώς απαιτούσε μεγάλη προσοχή και πολλές δοκιμές.

Ένας ακόμη περιορισμός ήταν ο προϋπολογισμός. Προκειμένου το συνολικό κόστος όλου του συστήματος να παραμείνει χαμηλό έπρεπε να γίνουν μερική συμβιβασμοί. Στα αρχικά σχέδια του συστήματος που κατασκευάστηκε ήταν η ύπαρξη δύο ESP32 πλακετών και όχι μιας. Στην πρώτη πλακέτα θα υπήρχαν συνδεδεμένοι όλοι οι αισθητήρες και θα επεξεργαζόταν τα δεδομένα από αυτούς. Η δεύτερη πλακέτα θα λάμβανε τα δεδομένα από την πρώτη μέσω ασύρματης επικοινωνίας και στην συνέχεια θα τα έστελνε στο νέφος Cloud που βρίσκεται το λογισμικό του συστήματος που κατασκευάστηκε. Με αυτόν τον τρόπο το κόστος του συστήματος θα αυξανόταν αρκετά καθώς θα απαιτούνταν η αγορά μιας επιπλέον πλακέτας ESP32 και μιας ενδεχόμενης συνδρομής στην διαδικτυακή εφαρμογή Arduino IoT Cloud, καθώς έπρεπε να χρησιμοποιηθούν επιπλέον υπηρεσίες που δεν παρέχονται από το δωρεάν πλάνο που προσφέρει η εφαρμογή. Ωστόσο με αυτόν τον τρόπο η μια πλακέτα θα περιείχε τον κώδικα για την επεξεργασία των αισθητήρων και η δεύτερη πλακέτα θα είχε μόνο τον κώδικα για την επικοινωνία του με το νέφος cloud. Αυτό θα επέτρεπε την προσθήκη περισσότερου κώδικα όπως την ενσωμάτωση των φωνητικών εντολών που αναφέρθηκε παραπάνω.

### 3.9 Τελικά συμπεράσματα

Συνοψίζοντας η διαδικασία της εξολοκλήρου κατασκευής ενός IoT συστήματος ασφαλείας είναι ευχάριστη αλλά δεν είναι εύκολη υπόθεση. Απαιτεί αρκετό χρόνο, γνώσεις, αναζήτηση πληροφοριών, ενδεχομένως να χρειαστεί η βοήθεια ενός ειδικού πάνω στον τομέα ενώ το βασικότερο είναι να υπάρχει αντοχή στις αποτυχίες. Το αποτέλεσμα είναι πολύ θετικό καθώς δημιουργείτε ένα μοναδικό σύστημα φτιαγμένο από την αρχή και βασιζόμενο στις ανάγκες του χρήστη ωστόσο δεν σταματάει εκεί. Μετά την ολοκλήρωση της κατασκευής ο χρήστης μπορεί να τροποποιήσει τον κώδικα του συστήματος και να προσθέσει επιπρόσθετες λειτουργίες. Σε ένα συμβατικό σύστημα ασφαλείας δεν παρέχετε η πρόσβαση στον πυγαίο κώδικα του συστήματος. Στην αρχή το σύστημα ασφαλείας που έχει κατασκευαστεί από τον χρήστη μπορεί να μοιάζει μικρό και να παρέχει τις βασικές λειτουργίες, ωστόσο σε μελλοντικό επίπεδο μπορεί να εξελιχθεί σε κάτι μεγάλο. Το σύστημα ασφαλείας που κατασκευάστηκε για τις ανάγκες της παρούσας πτυχιακής εργασίας ενσωματώνει βασικές λειτουργίες. Σε μελλοντικό στάδιο θα πραγματοποιηθούν διαδικασίες εξέλιξης με σκοπό να τοποθετηθεί σε ένα έξυπνο σπίτι και να συνεργαστεί με κάποιον έξυπνο ψηφιακό όπως είναι το Amazon echo dot της εταιρίας Amazon και το Google nest της Google.

## Βιβλιογραφία και άρθρα

### Κεφάλαιο 1

Εικόνα 1.1.1 <https://herringtechnology.com>

Εικόνα 1.1.2 <https://www.erahomesecurity.com>

Εικόνα 1.1.3 <https://www.intuz.com/blog/iot-smart-home-security-benefits-use-cases-and-top-devices>

Εικόνα 1.2.1 <https://www.frontpointsecurity.com>

[https://en.wikipedia.org/wiki/Edwin\\_Holmes\\_\(inventor\)](https://en.wikipedia.org/wiki/Edwin_Holmes_(inventor))

<https://herringtechnology.com/news/the-history-of-the-modern-alarm-system/>

<https://www.erahomesecurity.com/security-products/other-security-systems/era-valiant-and-%20%20invincible-alarm/era-invincible-smart-home-alarm-system/>

<https://knoema.com/atlas/Greece/Burglary-rate>

<https://www.met.police.uk/cp/crime-prevention/protect-home-crime/residential-burglary-facts/>

<https://www.frontpointsecurity.com/blog/smart-home-security-systems-and-smart-home-devices>

<https://www.intuz.com/blog/iot-smart-home-security-benefits-use-cases-and-top-devices>



## Κεφάλαιο 2

Εικόνα 2.1.1 <https://ajax.systems>

Εικόνα 2.1.2 <https://ajax.systems>

Εικόνα 2.1.3 <https://ajax.systems>

Εικόνα 2.1.4 <https://ajax.systems>

Εικόνα 2.1.5 <https://ajax.systems>

Εικόνα 2.2.1 <https://ajax.systems>

Εικόνα 2.2.2 <https://ajax.systems>

Εικόνα 2.2.3 <https://ajax.systems>

Εικόνα 2.2.4 <https://ajax.systems>

Εικόνα 2.2.5 <https://ajax.systems>

<https://ajax.systems/>

<https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work>

<https://randomnerdtutorials.com/monitor-your-door-using-magnetic-reed-switch-and-arduino/>

[https://en.wikipedia.org/wiki/Smoke\\_detector](https://en.wikipedia.org/wiki/Smoke_detector)

<https://www.youtube.com/@ajaxsystems8076>

<https://app.diagrams.net/>

[https://en.wikipedia.org/wiki/Multi-factor\\_authentication](https://en.wikipedia.org/wiki/Multi-factor_authentication)

## Κεφάλαιο 3

Εικόνα 3.2.1 <https://www.tinkercad.com/>

Εικόνα 3.3.1

Εικόνα 3.3.2

Εικόνα 3.3.3

Εικόνα 3.3.4

Εικόνα 3.4.1 <https://www.elegoo.com/>

Εικόνα 3.5.1 <https://drive.google.com>

Εικόνα 3.5.2 <https://trello.com/>

Εικόνα 3.5.3 <https://www.tinkercad.com/>

Εικόνα 3.5.4 <https://create.arduino.cc/>

Εικόνα 3.5.5 <https://www.arduino.cc/en/software>

Εικόνα 3.5.6 <https://support.arduino.cc/hc/en-us/articles/360014869820-Install-the-Arduino-Create-Agent>

Εικόνα 3.5.7 <https://notepad-plus-plus.org/>

Εικόνα 3.5.8 <https://ifttt.com/>

Εικόνα 3.6.1 <https://www.tinkercad.com/>

Εικόνα 3.6.2

Εικόνα 3.7.1 <https://cloud.arduino.cc>

Εικόνα 3.7.2 <https://cloud.arduino.cc>

Εικόνα 3.7.3 <https://cloud.arduino.cc>

Εικόνα 3.7.4 <https://cloud.arduino.cc>

Εικόνα 3.7.5 <https://cloud.arduino.cc>

Εικόνα 3.7.6 <https://ifttt.com/>

Εικόνα 3.7.7 <https://cloud.arduino.cc>

Εικόνα 3.7.8 <https://cloud.arduino.cc>

Εικόνα 3.8.1

Εικόνα 3.8.2

Εικόνα 3.8.3

<https://www.elegoo.com/>

<https://www.tinkercad.com/>

<https://app.diagrams.net/>

<https://aws.amazon.com/>

<https://cloud.google.com/>

<https://www.amazon.com/alexa>

<https://cloud.arduino.cc/>

<https://www.electronics-tutorials.ws/io/thermistors.html>

<https://www.linquip.com/blog/difference-between-microprocessor-and-microcontroller/>

[https://en.wikipedia.org/wiki/Google\\_Drive](https://en.wikipedia.org/wiki/Google_Drive)

<https://en.wikipedia.org/wiki/Trello>

<https://en.wikipedia.org/wiki/Tinkercad>

<https://en.wikipedia.org/wiki/Arduino>

<https://en.wikipedia.org/wiki/Notepad%2B%2B>

<https://en.wikipedia.org/wiki/IFTTT>

<https://en.wikipedia.org/wiki/Applet>