Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

Τίτλος: Εφαρμογή Οικιακού Αυτοματισμού

Κορδαλής Παρασκευάς (ΑΜ : 2406)

Μπιλίκμπασι Ρενάτο (ΑΜ : 3403)

Επιβλέπων Καθηγητής : Παναγιωτάκης Σπυρίδων

Επιτροπή Αξιολόγησης :

Ημερομηνία Παρουσίασης :

Ευχαριστίες

Η εκπόνηση της παρούσας πτυχιακής εργασίας με θέμα «Εφαρμογή Οικιακού Αυτοματισμού» στο Τμήμα Μηχανικών Πληροφορικής, Τ.Ε.Ι Ηρακλείου υλοποιήθηκε με την υποστήριξη των καθηγητών Παναγιωτάκη Σπυρίδων και Βλησίδη Ανδρέα.

Πρώτα από όλα θα θέλαμε να ευχαριστήσουμε του καθηγητές της Σχολής Μηχανικών Πληροφορικής του ΤΕΙ Κρήτης κ. Παναγιωτάκη Σπυρίδων και κ, Βλησίδη Ανδρέα για την καθοδήγηση και τις υποδείξεις τους σε όλο το διάστημα εκπόνησης της πτυχιακής μας εργασίας και για την δυνατότητα που μας προσέφεραν να ασχοληθούμε με ένα τόσο ενδιαφέρον αντικείμενο.

Επίσης θα θέλαμε να ευχαριστήσουμε όλους τους καθηγητές με τους οποίους είχαμε την τύχη να συνεργαστούμε σε όλη την διάρκεια της ακαδημαϊκής μας πορείας για της γνώσεις και τις εμπειρίες που μας μετέδωσαν όλα αυτά τα χρόνια.

Τέλος θα θέλαμε να ευχαριστήσουμε τις οικογένειες μας για την υποστήριξη τους καθ’ όλη την διάρκεια των σπουδών μας.

Abstract

In this thesis, we created a device that controls the air conditioners of the T.E.I Crete, we have to achieve that wirelessly while also we check how much watts the A.C consumes.
The device uses the Arduino Uno R3 as the primary controller but in order to have a wireless connection we had to use the ESP8266 – 01 so we can connect to the internet and with the use of MQTT protocol we managed a 24/7 communication with the device, the device also used the SCT-013-000 sensor for the calculation of the watts and also the DHT11 sensor the temperature and humidity values.

Σύνοψη

Στόχος της πτυχιακής εργασίας είναι η δημιουργία μίας συσκευής θα ελέγχει τα κλιματιστικά του ΤΕΙ Κρήτης ασύρματα και απομακρυσμένα, ενώ παράλληλα θα παρατηρείτε και η κατανάλωση του εναλλασσόμενου ρεύματος που καταναλώνεται.

Η συσκευή χρησιμοποιεί το Arduino Uno R3 ως πρωτεύοντα ελεγκτή πάνω στον οποίο θα συνδεθούν και άλλες συσκευές για την καταγραφή δεδομένων αλλά και την ασύρματη επικοινωνία των συσκευών. Για την ασύρματη σύνδεση της συσκευής με το δίκτυο θα χρησιμοποιηθεί το Esp8266-01 και με την χρήση του πρωτοκόλλου MQTT επιτυγχάνουμε την επικοινωνία με την συσκευή. Η συσκευή χρησιμοποιεί επίσης τον αισθητήρα SCT-013-000 για τον υπολογισμό της κατανάλωσης των watt που καταναλώνονται καθώς επίσης και τον αισθητήρα DHT11 για τις τιμές της θερμοκρασίας και της υγρασίας.

Η εργασία πραγματοποιήθηκε στο εργαστήριο ΕΝΠΕΤ του ΤΕΙ Κρήτης.

Λέξεις κλειδιά : Arduino Uno, MQTT, SCT-013, DHT11, Esp8266, IR receiver, IR emitter

Πίνακας Περιεχομένων

Πίνακας Εικόνων

1.Εισαγωγή

1.1 Περίληψη

Στην καθημερινότητα μας αναζητούμε λύσεις για να αυτοματοποιήσουμε πολλές λειτουργίες και συσκευές που χειριζόμαστε καθημερινά, με σκοπό να εξοικονομούμε χρόνο και χρήματα. Αυτές οι προσπάθειες έχουν αποδώσει καρπούς και έχουν βρει εφαρμογή τόσο στην καθημερινότητα των ανθρώπων όσο και στην λειτουργία των εταιρειών.

Μία τεχνολογία που βρίσκει εφαρμογή σε όσα προαναφέραμε είναι το IoT (Internet of Things), το οποίο είναι ένα προηγμένο σύστημα αυτοματισμού και αναλύσεων το οποίο κάνει χρήση του δικτύου, της ανίχνευσης, των μεγάλων δεδομένων και της τεχνολογίας τεχνητής νοημοσύνης για να δώσει ολοκληρωμένα συστήματα για ένα προϊόν ή μία συσκευή. Τα συστήματα αυτά επιτρέπουν μεγαλύτερη διαφάνεια, έλεγχο και την απόδοση όταν εφαρμόζονται σε κάποιο σύστημα ή μια βιομηχανία.

IoT συστήματα έχουν εφαρμογές σε βιομηχανίες μέσω της μοναδικής ευελιξία τους και την ικανότητα τους να είναι κατάλληλα σε κάθε περιβάλλον. Ενισχύουν την συλλογή δεδομένων, την αυτοματοποίηση, τις λειτουργίες, και πολλά άλλα μέσω έξυπνων συσκευών και επιτρέπουν ισχυρή τεχνολογία.

Τα IoT συστήματα επιτρέπουν στους χρήστες να επιτύχουν βαθύτερη αυτοματοποίηση, ανάλυση, και την ενσωμάτωση μέσα σε ένα σύστημα. Βελτιώνουν την εμβέλεια και την ακρίβεια αυτών των περιοχών. Τα IoT συστήματα χρησιμοποιούν υπάρχουσες και αναδυόμενες τεχνολογίες για την ανίχνευση, τη δικτύωση και την ρομποτική.

Το IoT εκμεταλλεύεται τις πρόσφατες εξελίξεις στον τομέα του λογισμικού, την πτώση των τιμών στο κομμάτι των υλικών, και την σύγχρονη στάση απέναντι στην τεχνολογία. Είναι νέα και προηγμένα στοιχεία που επιφέρουν μεγάλες αλλαγές στην παράδοση προϊόντων, αγαθών και υπηρεσιών, οι οποίες αλλαγές έχουν πολύ μεγάλη επίδραση στο κοινωνικό, στο οικονομικό και πολιτικό κομμάτι μίας κοινωνίας.

Στα πιο σημαντικά χαρακτηριστικά του IoT περιλαμβάνονται η χρήση τεχνητής νοημοσύνης, δικτύωσης, αισθητήρων, ενεργού συμμετοχής και η χρήση μικρών συσκευών. Μία σύντομη επισκόπηση των χαρακτηριστικών αυτών παρατίθεται παρακάτω.

- Τεχνητή νοημοσύνη : Το IoT κάνει ουσιαστικά οτιδήποτε “έξυπνο”, που σημαίνει ότι βελτιώνει κάθε πτυχή της ζωής μας με την δύναμη της συλλογής πληροφοριών, των αλγορίθμων τεχνητής νοημοσύνης και της δικτύωσης.
- Συνδεσιμότητα : Νέες τεχνολογίες γενικής εφαρμογής για την δικτύωση, και συγκεκριμένα IoT δικτύωση, που σημαίνει ότι τα δίκτυα δεν είναι πλέον συνδεδεμένα αποκλειστικά και μόνο με μεγάλους παρόχους. Τα δίκτυα μπορούν να υπάρχουν σε μία μικρότερη και πολύ φθηνότερη κλίμακα ενώ εξακολουθούν να είναι πρακτικά. Το IoT δημιουργεί αυτά τα μικρά δίκτυα μεταξύ των συσκευών του συστήματος.
- Αισθητήρες : Το IoT χάνει τη διαφοροποίηση του χωρίς τους αισθητήρες. Ενεργούν ως καθοριστικά εργαλεία που μετατρέπουν το IoT από ένα τυπικό παθητικό δίκτυο συσκευών σε ένα ενεργό σύστημα ενσωμάτωσης στον πραγματικό κόσμο.
- Ενεργή συμμετοχή : Μεγάλο μέρος της σημερινής αλληλεπίδρασης γίνεται μέσω παθητικής συμμετοχής. Το IoT εισάγει ένα νέο πρότυπο για ενεργό περιεχόμενο,

προϊόντα, ή υπηρεσίες συμμετοχής.
- Μικρές συσκευές : Οι συσκευές, όπως αναμενόταν έχουν γίνει μικρότερες, φθηνότερες και πιο ισχυρές με την πάροδο του χρόνου. Το IoT εκμεταλλεύεται ειδικά κατασκευασμένες μικρές συσκευές για την παροχή ακρίβειας, επεκτασιμότητας και ευελιξίας.

Τα πλεονεκτήματα του IoT εκτείνονται σε κάθε τομέα του τρόπου ζωής και των επιχειρήσεων. Μερικά από αυτά αναφέρονται συνοπτικά στην συνέχεια :
- Βελτιωμένη Εμπλοκή Πελατών : Τα τρέχοντα αναλυτικά στοιχεία πάσχουν από τυφλά σημεία και σημαντικές ελλείψεις στην ακρίβεια και όπως επισημαίνεται η εμπλοκή παραμένει παθητική. Το IoT το μεταμορφώνει πλήρως ώστε να επιτευχθεί πιο πλούσια και αποτελεσματική εμπλοκή με το κοινό.
- Βελτιστοποίηση της τεχνολογίας : Οι ίδιες τεχνολογίες και τα δεδομένα τα οποία βελτιώνουν την εμπειρία του πελάτη, βελτιώνουν επίσης τη χρήση τη συσκευής, και βοηθούν στην ισχυρή βελτίωση της τεχνολογίας. Το IoT ξεκλειδώνει έναν κόσμο κρίσιμων λειτουργιών και πεδίων δεδομένων.
- Μειωμένη σπατάλη : Το IoT κάνει του τομείς βελτίωσης σαφής. Τα τρέχοντα αναλυτικά στοιχεία μας δίνουν επιφανειακή εικόνα, αλλά το IoT παρέχει πραγματικές πληροφορίες που θα οδηγήσουν σε αποτελεσματικότερη διαχείριση των πόρων.
- Βελτιωμένη συλλογή δεδομένων : Η σύγχρονη συλλογή δεδομένων πάσχει από περιορισμούς και είναι σχεδιασμένη για παθητική χρήση. Το IoT σπάει αυτά τα πλαίσια, και την τοποθετεί εκεί ακριβώς όπου οι άνθρωποι θέλουν να αναλύσουν τον κόσμο μας. Επιτρέπει μία ακριβή εικόνα των πάντων.

Αν το IoT προσφέρει μία σημαντική σειρά από πλεονεκτήματα, παρουσιάζει επίσης και ένα σημαντικό σύνολο προκλήσεων. Εν συντομία μερικά από αυτά :
- Ασφάλεια : Το IoT δημιουργεί ένα οικοσύστημα διαρκώς συνδεδεμένων συσκευών που επικοινωνούν μέσω δικτύων. Το σύστημα προσφέρει περιορισμένο έλεγχο παρά τα όποια μέτρα ασφαλείας. Γεγονός που αφήνει τους χρήστες εκτεθειμένους σε διάφορα είδη εισβολών.
- Προστασία προσωπικών δεδομένων : Η πολυπλοκότητα του IoT προσφέρει σημαντικά προσωπικά δεδομένα με ακραία λεπτομέρεια χωρίς την ενεργό συμμετοχή του χρήστη.
- Πολυπλοκότητα : Μερικοί βρίσκουν τα IoT συστήματα περίπλοκα όσον αφορά των σχεδιασμό, την ανάπτυξη και την συντήρηση δεδομένης της χρήσης πολλαπλών τεχνολογιών και ένα μεγάλο σύνολο επιτρεπόντων τεχνολογιών.
- Ευελιξία : Πολλοί ανησυχούν για την ευελιξία ενσωμάτωσης ενός IoT συστήματος εύκολα με ένα άλλο. Ανησυχούν μήπως βρουν τον εαυτό τοθς με αρκετά αλληλοσυγκρουόμενα ή κλειδωμένα συστήματα.
- Συμμόρφωση : Το IoT, όπως και κάθε άλλη τεχνολογία στον χώρο των επιχειρήσεων, πρέπει να συμμορφώνεται με τους κανονισμούς. Η πολυπλοκότητα του κάνει το θέμα της συμμόρφωσης να μοιάζει απίστευτα δύσκολο.

1.2 Κίνητρο διεξαγωγής εργασίας

Η προσπάθεια για των έλεγχο των κλιματιστικών και άλλων ηλεκτρικών συσκευών είναι από τα βασικά μελήματα των εταιριών, καθώς όσο εξελίσσονται οι τεχνολογίες οι εταιρίες προσαρμόζουν τις συσκευές πιο κοντά στις ανάγκες των καταναλωτών.

Σχεδόν όλες οι εταιρίες έχουν εξελίξει τις συσκευές τους ώστε να δίνουν την δυνατότητα στον καταναλωτή να έχει απομακρυσμένο έλεγχο των ηλεκτρικών συσκευών του, μέσω ασύρματου δικτύου. Το πρόβλημα εντοπίζεται σε παλαιότερες συσκευές οι οποίες δεν μπορούν να υποστηρίξουν αυτές τις δυνατότητες.

Το πρόβλημα αυτό μπορεί να ξεπεραστεί με την δημιουργία συσκευών, οι οποίες θα έχουν την δυνατότητα να επικοινωνούν με τις ηλεκτρικές συσκευές ακόμη και αν αυτές δεν αποτελούν μέρος των νέων "έξυπνων" συσκευών.

Στη παρούσα πτυχιακή εργασία δημιουργούμε μια συσκευή η οποία έχει την δυνατότητα επικοινωνίας με το διαδίκτυο, χρησιμοποιώντας ασύρματο δίκτυο, ώστε να επιτευχθεί ο απομακρυσμένος μίας μονάδας κλιματιστικού.

Το βασικό κίνητρο διεξαγωγής της εργασίας είναι ο απομακρυσμένος έλεγχος ενός Air Condition μέσω μιας ιστοσελίδας, χρησιμοποιώντας σχεδόν όλες τις λειτουργίες του χειριστηρίου και με όσο το δυνατόν μεγαλύτερη αξιοπιστία στην απόκριση της συσκευής. Για να το επιτύχουμε αυτό χρησιμοποιούμε το πρωτόκολλο επικοινωνίας MQTT.

1.3 Σκοπός και Στόχοι Εργασίας

Ο σκοπός αυτής της εργασίας είναι η έρευνα της λειτουργίας των συστημάτων IoT και των συσκευών που χρησιμοποιεί για την ανάπτυξη μίας συσκευής απομακρυσμένου ελέγχου και συλλογής δεδομένων, αρχικά των κλιματιστικών μονάδων ενός εργαστηρίου του ΤΕΙ Κρήτης και εν συνεχεία η ανάπτυξη του σε μεγαλύτερη κλίμακα.
Στόχος αυτή της εργασίας επίσης είναι η δημιουργία μίας συσκευής με το μικρότερο δυνατό κόστος, ώστε να είναι προσιτή η περαιτέρω έρευνα για ανάπτυξη του συγκεκριμένου project.

1.4 Δομή Εργασίας

Στο κεφάλαιο 2 γίνεται αναφορά στα εργαλεία software που χρησιμοποιήθηκαν για την υλοποίηση της εργασίας. Στο κεφάλαιο 3 γίνεται αναφορά στα εργαλεία hardware που χρησιμοποιήθηκαν για την δημιουργία της συσκευής. Περνώντας στο κεφάλαιο 4, παρουσιάζονται αναλυτικά τα βήματα που ακολουθήθηκαν για την δημιουργία της ιστοσελίδας μέσω της οποίας ελέγχουμε την συσκευή μας, καθώς επίσης και την συνδεσμολογία του hardware της συσκευής. Τέλος, το κεφάλαιο 5 αφορά την αναλυτική επεξήγηση του κώδικα και των τεχνολογιών που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής.

2.Μεθοδολογία υλοποίησης – Εργαλεία Software

Σε αυτό το κεφάλαιο θα παρουσιάσουμε τα προγράμματα και τις τεχνολογίες που χρησιμοποιήθηκαν για την επικοινωνία των συσκευών του project μας.

2.1 XAMPP

Εικόνα 1 - XAMPP

Το **XAMPP** είναι ένα πακέτο προγραμμάτων ελεύθερου λογισμικού, λογισμικού ανοικτού κώδικα και ανεξαρτήτου πλατφόρμας το οποίο περιέχει το εξυπηρετητή ιστοσελίδων http Apache, την βάση δεδομένων MySQL και ένα διερμηνέα για κώδικα γραμμένο σε γλώσσες προγραμματισμού PHP και Perl.

Το **XAMPP** είναι ακρωνύμιο και αναφέρεται στα παρακάτω αρχικά:

- X (αναφέρεται στο «cross-platform» που σημαίνει λογισμικό ανεξάρτητο πλατφόρμας)
- Apache HTTP εξυπηρετητής
- MySQL
- PHP
- Perl

Το **XAMPP** είναι ένα ελεύθερο λογισμικό το οποίο περιέχει ένα εξυπηρετητή ιστοσελίδων το οποίο μπορεί να εξυπηρετεί και δυναμικές ιστοσελίδες τεχνολογίας PHP/MySQL. Είναι ανεξάρτητο πλατφόρμας και τρέχει σε Microsoft Windows, Linux, Solaris, and Mac OS X και χρησιμοποιείται ως πλατφόρμα για την σχεδίαση και ανάπτυξη ιστοσελίδων με την τεχνολογίες όπως PHP, JSP και Servlets.

Όταν το XAMPP εγκατασταθεί στον τοπικό υπολογιστή διαχειρίζεται τον localhost ως ένα απομακρυσμένο κόμβο, ο οποίος συνδέεται με το πρωτόκολλο μεταφοράς αρχείων FTP. Η σύνδεση στον localhost μέσω του FTP μπορεί να γίνει με το όνομα χρήστη «newuser» και το κωδικό «wampp». Για την βάση δεδομένων MySQL υπάρχει ο χρήστης «root» χωρίς κωδικό πρόσβασης.

2.1.1 Apache HTTP

Εικόνα 2 – Apache Server

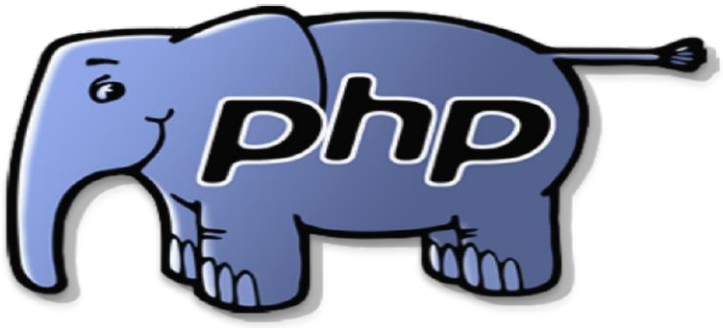
Ο **Apache HTTP** γνωστός και απλά σαν Apache είναι ένας εξυπηρετητής του παγκόσμιου ιστού (web). Όποτε ένας χρήστης επισκέπτεται ένα ιστότοπο το πρόγραμμα πλοήγησης (browser) επικοινωνεί με έναν διακομιστή (server) μέσω του πρωτοκόλλου HTTP, ο οποίος παράγει τις ιστοσελίδες και τις αποστέλλει στο πρόγραμμα πλοήγησης. Ο Apache είναι ένας από τους δημοφιλέστερους εξυπηρετητές ιστού, εν μέρει γιατί λειτουργεί σε διάφορες πλατφόρμες όπως τα Windows, το Linux, το Unix και το Mac OS X. Κυκλοφόρησε υπό την άδεια λογισμικού Apache και είναι λογισμικό ανοιχτού κώδικα. Συντηρείται από μια κοινότητα ανοικτού κώδικα με επιτήρηση από το Ίδρυμα Λογισμικού Apache (Apache Software Foundation).

2.1.2 MySQL

Εικόνα 3 – MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων που μετρά περισσότερες από 11 εκατομμύρια εγκαταστάσεις. Έλαβε το όνομά της από την κόρη του Μόντυ Βιντένιους, τη Μάι (αγγλ. My). Το πρόγραμμα τρέχει έναν εξυπηρετητή (server) παρέχοντας πρόσβαση πολλών χρηστών σε ένα σύνολο βάσεων δεδομένων.
Είναι δημοφιλής βάση δεδομένων για διαδικτυακά προγράμματα και ιστοσελίδες.

2.1.3 PHP



Εικόνα 4 - PHP

Η PHP (PHP: Hypertext Preprocessor) είναι μια γλώσσα προγραμματισμού για τη δημιουργία σελίδων web με δυναμικό περιεχόμενο. Μια σελίδα PHP περνά από επεξεργασία από ένα συμβατό διακομιστή του Παγκόσμιου Ιστού (π.χ. Apache), ώστε να παραχθεί σε πραγματικό χρόνο το τελικό περιεχόμενο, που είτε θα σταλεί στο πρόγραμμα περιήγησης των επισκεπτών σε μορφή κώδικα HTML ή θα επεξεργασθεί τις εισόδους δίχως να προβάλλει την έξοδο στο χρήστη, αλλά θα τις μεταβιβάσει σε κάποιο άλλο PHP script.

Η PHP αποτελεί μια από τις πιο διαδεδομένες τεχνολογίες στο Παγκόσμιο Ιστό, καθώς χρησιμοποιείται από πληθώρα εφαρμογών και ιστότοπων. Η ευρύτητα στη χρήση της είναι απόρροια της ευκολίας που παρουσιάζει ο προγραμματισμός με αυτή αλλά και στο γεγονός πως είναι μια γλώσσα η οποία βρίσκεται σχεδόν σε κάθε διακομιστή.

2.2 Javascript (JS)

Εικόνα 5 - Javascript

Η **JavaScript** (JS) είναι διερμηνευμένη γλώσσα προγραμματισμού για ηλεκτρονικούς υπολογιστές. Αρχικά αποτέλεσε μέρος της υλοποίησης των φυλλομετρητών Ιστού, ώστε τα σενάρια από την πλευρά του πελάτη (client-side scripts) να μπορούν να επικοινωνούν με τον χρήστη, να ανταλλάσσουν δεδομένα ασύγχρονα και να αλλάζουν δυναμικά το περιεχόμενο του εγγράφου που εμφανίζεται.

Η **JavaScript** είναι μια γλώσσα σεναρίων που βασίζεται στα πρωτότυπα (prototype-based), είναι δυναμική, με ασθενείς τύπους και έχει συναρτήσεις ως αντικείμενα πρώτης τάξης. Η

σύνταξή της είναι επηρεασμένη από τη C. Η JavaScript αντιγράφει πολλά ονόματα και συμβάσεις ονοματοδοσίας από τη Java, αλλά γενικά οι δύο αυτές γλώσσες δε σχετίζονται και έχουν πολύ διαφορετική σημασιολογία. Οι βασικές αρχές σχεδιασμού της JavaScript προέρχονται από τις γλώσσες προγραμματισμού Self και Scheme.Είναι γλώσσα βασισμένη σε διαφορετικά προγραμματιστικά παραδείγματα (multi-paradigm), υποστηρίζοντας αντικειμενοστρεφές, προστακτικό και συναρτησιακό στυλ προγραμματισμού.

Η **JavaScript** χρησιμοποιείται και σε εφαρμογές εκτός ιστοσελίδων — τέτοια παραδείγματα είναι τα έγγραφα PDF, οι εξειδικευμένοι φυλλομετρητές (site-specific browsers) και οι μικρές εφαρμογές της επιφάνειας εργασίας (desktop widgets). Οι νεότερες εικονικές μηχανές και πλαίσια ανάπτυξης για JavaScript (όπως το Node.js) έχουν επίσης κάνει τη JavaScript πιο δημοφιλή για την ανάπτυξη εφαρμογών Ιστού στην πλευρά του διακομιστή (server-side).

Δείγμα κώδικα Javascript

Ο κώδικας Javascript μιας σελίδας περικλείεται από τις ετικέτες της HTML <script type="text/javascript"> και </script>.

Για παράδειγμα, ο ακόλουθος κώδικας Javascript εμφανίζει ένα πλαίσιο διαλόγου με το κείμενο "Γεια σου, κόσμε!":

```
<script type="text/javascript">
alert('Γεια σου, κόσμε!');
</script>
```

Αν ο κώδικας Javascript περιέχει περισσότερες από μία εντολές, αυτές θα πρέπει να διαχωριστούν μεταξύ τους με το χαρακτήρα του ελληνικού ερωτηματικού ';' (δηλαδή της λατινικής άνω τελείας). Η χρήση του χαρακτήρα αυτού για την τελευταία εντολή δεν είναι απαραίτητη. Η διαχώριση των εντολών στους νεότερους φυλλομετρητές (browsers) δεν είναι απαραίτητη.
Μια άλλη βασική εντολή, η window.prompt("μήνυμα προς το χρήστη"), ζητάει από το χρήστη να συμπληρώσει ένα κομμάτι μιας αίτησης απευθείας ώστε τα δεδομένα να χρησιμοποιηθούν σαν κείμενο:

```
<script>
var FIRSTvariable = window.prompt("PLEASE FILL IN YOUR NAME")
alert("Your name is " + FIRSTvariable + ".")
</script>
```

2.3 JQuery

Εικόνα 6 - JQuery

Η jQuery είναι μια βιβλιοθήκη JavaScript σχεδιασμένη να απλοποιήσει την υλοποίηση σεναρίων (scripting) στη πλευρά του πελάτη (client-side) της HTML και υποστηρίζει πολλαπλούς φυλλομετρητές Ιστού. Κυκλοφόρησε τον Ιανουάριο του 2006 από τον Τζον Ρέριγκ (John Resig). Χρησιμοποιείται σε πάνω από το 65% των 10.000 ιστοτόπων με τη μεγαλύτερη επισκεψιμότητα.

Η jQuery είναι ελεύθερο λογισμικό, με άδεια MIT.

Η jQuery είναι ένα αρχείο JavaScript, που περιέχει όλες τις λειτουργίες. Μπορεί να συμπεριληφθεί σε μια ιστοσελίδα παρέχοντας το αρχείο τοπικά
<script type="text/javascript" src="jquery.js"></script>
ή έχοντας ένα σύνδεσμο σε ένα από τους πολλούς διακομιστές που τη φιλοξενούν.
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>

2.4 MQTT (MQ Telemetry Transport) Protocol

Εικόνα 7 – MQTT

Είναι ένα πρότυπο ISO (ISO / IEC 20922 PRF). Είναι ένα ελαφρύ πρωτόκολλο δημοσίευσης-εγγραφής (publish – subscribe) μηνυμάτων για χρήση πάνω από το πρωτόκολλο TCP / IP. Είναι σχεδιασμένο για συνδέσεις με απομακρυσμένες τοποθεσίες όπου απαιτείται ένα "μικρό ίχνος κώδικα " ή το εύρος ζώνης του δικτύου είναι περιορισμένο. Το πρότυπο απαιτεί έναν μεσίτη (broker). Ο broker είναι υπεύθυνος για τη διανομή μηνυμάτων σε ενδιαφερόμενους πελάτες με βάση το θέμα του μηνύματος.

Η προδιαγραφή του δεν προσδιορίζει την έννοια του μικρού κώδικα ή την έννοια του περιορισμένου εύρους ζώνης του δικτύου. Έτσι, η διαθεσιμότητα του πρωτοκόλλου για χρήση εξαρτάται από το πλαίσιο. MQTT-SN είναι μια παραλλαγή του κύριου πρωτοκόλλου που αποσκοπεί στην ενσωμάτωση συσκευών σε μη-TCP / IP δίκτυα, όπως το ZigBee.

Εναλλακτικά πρωτόκολλα περιλαμβάνουν το Advanced Message Queuing πρωτόκολλο, το IETF περιορισμένης εφαρμογής πρωτόκολλο, το XMPP και Web Application Messaging Protocol (WAMP).

2.4.1 MQTT εντολές

Το MQTT ορίζει μεθόδους για να υποδεικνύουν την επιθυμητή δράση που πρέπει να εκτελεστεί στην ταυτοποιημένη πηγή. Τι αντιπροσωπεύει αυτή η πηγή, είτε προϋπήρχαν δεδομένα ή δεδομένα που δημιουργούνται δυναμικά, εξαρτάται από την υλοποίηση του server. Συχνά η πηγή αντιστοιχεί σε ένα αρχείο ή την έξοδο ενός εκτελέσιμου που φιλοξενούνται στον server.

Connect
Περιμένει για μία σύνδεση που θα συσταθεί με τον server.

Disconnect
Περιμένει τον MQTT Client να τελειώσει να τελειώσει κάθε εργασία που έχει να κάνει, και για αποσύνδεση της συνεδρίας του TCP/IP.

Subscribe
Περιμένει την ολοκλήρωση μεθόδου εγγραφής ή διαγραφής.

UnSubscribe
Ζητά από τον server την διαγραφή του client από ένα η περισσότερα topics.

Publish
Μόλις κάνει την δημοσίευση στον MQTT Client επιστρέφει αμέσως στο νήμα της εφαρμογής.

2.5 Arduino Software IDE

Εικόνα 8 – Arduino Software IDE

Το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) του Arduino είναι μία εφαρμογή γραμμένη σε Java, που λειτουργεί σε πολλές πλατφόρμες, και προέρχεται από το IDE για τη γλώσσα προγραμματισμού Processing και το σχέδιο Wiring. Έχει σχεδιαστεί για να εισαγάγει τον προγραμματισμό στους καλλιτέχνες και τους νέους που δεν είναι εξοικειωμένοι με την ανάπτυξη λογισμικού. Περιλαμβάνει ένα πρόγραμμα επεξεργασίας κώδικα με χαρακτηριστικά όπως είναι η επισήμανση σύνταξης και ο συνδυασμός αγκύλων και είναι επίσης σε θέση να μεταγλωττίζει και να φορτώνει προγράμματα στην πλακέτα με ένα μόνο κλικ. Δεν υπάρχει συνήθως καμία ανάγκη να επεξεργαστείτε αρχεία make ή να τρέξετε προγράμματα σε ένα περιβάλλον γραμμής εντολών. Ένα πρόγραμμα ή κώδικας που γράφτηκε για Arduino ονομάζεται σκίτσο (sketch).
Τα Arduino προγράμματα είναι γραμμένα σε C ή C++.Το Arduino IDE έρχεται με μια βιβλιοθήκη λογισμικού που ονομάζεται «Wiring» από το πρωτότυπο σχέδιο Wiring γεγονός που καθιστά πολλές κοινές λειτουργίες εισόδου/εξόδου πολύ πιο εύκολες. Οι χρήστες πρέπει μόνο να ορίσουν δύο λειτουργίες για να κάνουν ένα πρόγραμμα κυκλικής εκτέλεσης:
-setup():μία συνάρτηση που τρέχει μία φορά στην αρχή του προγράμματος η οποία αρχικοποιεί τις ρυθμίσεις
-loop():μία συνάρτηση η οποία καλείται συνέχεια μέχρι η πλακέτα να απενεργοποιηθεί
Ένα τυπικό πρώτο πρόγραμμα για έναν μικροελεγκτή αναβοσβήνει απλά ένα LED. Στο περιβάλλον του Arduino, ο χρήστης μπορεί να γράψει ένα πρόγραμμα σαν αυτό:

```
#define LED_PIN 13

void setup () {
 pinMode (LED_PIN, OUTPUT); // enable pin 13 for digital output
}

void loop () {
 digitalWrite (LED_PIN, HIGH); // turn on the LED
 delay (1000); // wait one second (1000 milliseconds)
 digitalWrite (LED_PIN, LOW); // turn off the LED
 delay (1000); // wait one second
}
```

Είναι ένα χαρακτηριστικό των περισσότερων πλακετών Arduino ότι έχουν ένα LED και μία αντίσταση φορτίου που συνδέονται μεταξύ του pin 13 και του εδάφους, ένα βολικό χαρακτηριστικό για πολλά απλά τεστ. Ο προηγούμενος κώδικας δεν θα αναγνωριστεί από ένα κανονικό μεταγλωττιστή C + + ως έγκυρο πρόγραμμα, έτσι ώστε όταν ο χρήστης κάνει κλικ στο κουμπί «Upload to I / O board» στο IDE, ένα αντίγραφο του κώδικα θα γραφτεί σε ένα προσωρινό αρχείο με ένα παραπάνω include στην κορυφή και μία πολύ απλή συνάρτηση main() στο τέλος, για να φτιάξει ένα έγκυρο C++ πρόγραμμα.
Το IDE του Arduino χρησιμοποιεί το GNU toolchain και το AVR Libc για να μεταγλωττίζει προγράμματα και το avrdude για να φορτώνει προγράμματα στην πλακέτα.
Δεδομένου ότι η πλατφόρμα Arduino χρησιμοποιεί Atmel μικροελεγκτές, το περιβάλλον ανάπτυξης της Atmel, το AVR Studio ή το νεότερη έκδοση του Atmel Studio, μπορεί επίσης να χρησιμοποιηθεί για την ανάπτυξη λογισμικού για το Arduino.

2.6 Λειτουργικό σύστημα (Λ.Σ.) Windows 10

Τα Windows 10 είναι το νεότερο λειτουργικό σύστημα της Microsoft (μετά τα Windows 8.1) για υπολογιστές. Η πρώτη παρουσίαση των Windows 10 έγινε στις 30 Σεπτεμβρίου 2014. Ξεκίνησαν να κυκλοφορούν επισήμως στις 29 Ιουλίου του 2015, ενώ δοκιμαστική έκδοση είχε κυκλοφορήσει την 1 Οκτωβρίου 2014.

Το λειτουργικό σύστημα είναι διαθέσιμο σε 190 χώρες και σε 111 γλώσσες κατά την έναρξη διάθεσης, καθώς και στο πλαίσιο των προσπαθειών για την «επαναπροσέγγιση» των χρηστών στην Κίνα, η Microsoft ανακοίνωσε ότι θα συνεργαστεί με την Qihoo και την Tencent για να βοηθήσουν στην προώθηση και τη διανομή των Windows 10 στην Κίνα, και ότι η κινεζική εταιρεία κατασκευής PC Lenovo θα παρέχει βοήθεια στα κέντρα εξυπηρέτησης και στα καταστήματα λιανικής πώλησης για να βοηθήσει τους χρήστες κατά την αναβάθμιση σε Windows 10.

Ο λόγος που επιλέχτηκε αυτό το λειτουργικό σύστημα ήταν μόνο για την εύκολη χρήση που έχει, θα μπορούσαμε να χρησιμοποιήσουμε και κάποιο διαφορετικό Λ.Σ (π.χ. Ubuntu), χωρίς να χρειαστεί να αλλάξουμε πολλά πράγματα αλλά αν δεν υπάρχει κάποια προηγούμενη επαφή με αυτό το λειτουργικό τότε ο χρόνος υλοποίησης της εργασίας θα αυξανόταν.

3. Μεθοδολογία υλοποίησης – Εργαλεία Hardware

3.1 Arduino Uno R3

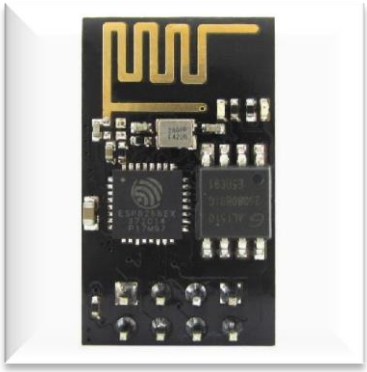
Εικόνα 9 – Arduino Uno R3

Μία πλακέτα Arduino αποτελείται από ένα μικροελεγκτή Atmel AVR (ATmega328 και ATmega168 στις νεότερες εκδόσεις, ATmega8 στις παλαιότερες) και συμπληρωματικά εξαρτήματα για την διευκόλυνση του χρήστη στον προγραμματισμό και την ενσωμάτωσή του σε άλλα κυκλώματα. Όλες οι πλακέτες περιλαμβάνουν ένα γραμμικό ρυθμιστή τάσης 5V και έναν κρυσταλλικό ταλαντωτή 16MHz (ή κεραμικό αντηχητή σε κάποιες παραλλαγές). Ο μικροελεγκτής είναι από κατασκευής προγραμματισμένος με ένα bootloader, έτσι ώστε να μην χρειάζεται εξωτερικός προγραμματιστής.
Σε εννοιολογικό επίπεδο, στην χρήση του Arduino software stack, όλα τα boards προγραμματίζονται με μία RS-232 σειριακή σύνδεση, αλλά ο τρόπος που επιτυγχάνεται αυτό διαφέρει σε κάθε hardware εκδοχή. Οι σειριακές πλάκες Arduino περιέχουν ένα απλό level shifter κύκλωμα για να μετατρέπει μεταξύ σήματος επιπέδου RS-232 και TTL. Τα τωρινά Arduino προγραμματίζονται μέσω USB· αυτό καθίσταται δυνατό μέσω της εφαρμογής προσαρμοστικών chip USB-to-Serial όπως το FTDI FT232. Κάποιες παραλλαγές, όπως το Arduino mini και το ανεπίσημο Boarduino, χρησιμοποιούν ένα αφαιρούμενο USB-to-Serial καλώδιο ή board, Bluetooth ή άλλες μεθόδους. (Όταν χρησιμοποιείται με παραδοσιακά εργαλεία microcontroller αντί για το Arduino IDE, χρησιμοποιείται πρότυπος προγραμματισμός AVR ISP).
Ο πίνακας Arduino εκθέτει τα περισσότερα microcontroller I/O pins για χρήση από άλλα κυκλώματα. Τα Diecimila, Duemilanove και το τρέχον Uno παρέχουν 14 ψηφιακά I/O pins, έξι από τα οποία μπορούν να παράγουν pulse-width διαμορφωμένα σήματα, και έξι αναλογικά δεδομένα. Αυτά τα pins βρίσκονται στην κορυφή του πίνακα μέσω female headers 0.1 ιντσών (2,2mm). Διάφορες εφαρμογές ασπίδων plug-in είναι εμπορικώς διαθέσιμες.
Το Arduino Nano και το Arduino-Compatible Bare Bones Board και Boarduino Board ενδέχεται να παρέχουν male header pins στο κάτω μέρος του board προκειμένου να συνδέονται σε Breadboards. Υπάρχουν πολλά boards συμβατά με και προερχόμενα από Arduino boards. Κάποια είναι λειτουργικά ισάξια με ένα Arduino και μπορεί να χρησιμοποιηθούν εναλλακτικά. Πολλοί είναι το βασικό Arduino με την προσθήκη καινοτόμων output drivers, συχνά για την χρήση σχολικής μόρφωσης για να απλοποιήσουν την κατασκευή buggies και μικρών robot. Άλλες είναι ηλεκτρικά ισάξιες αλλά αλλάζουν τον παράγοντα μορφής, επιτρέποντας κάποιες φορές την συνεχόμενη χρήση των Shields ενώ κάποιες όχι. Κάποιες παραλλαγές είναι τελείως διαφορετικοί επεξεργαστές, με ποικίλα επίπεδα συμβατότητας.

Χαρακτηριστικά του Arduino Uno

- Μικροελεγκτής : ATmega328
- Τάση Λειτουργίας : 5 V
- Τάση εισόδου : 5V
- Ψηφιακές Είσοδοι/Έξοδοι : 14 (6 εκ των οποίων παρέχουν εξόδους PWM)
- Αναλογικές Εισόδους : 6
- Συνεχές ρεύμα : 40mA
- Μνήμη Flash : 32 KB
- Flash Memory : 2KB
- EEPROM : 1KB
- Clock Speed : 16 MHz

3.2 ESP8266 – 01 Edition

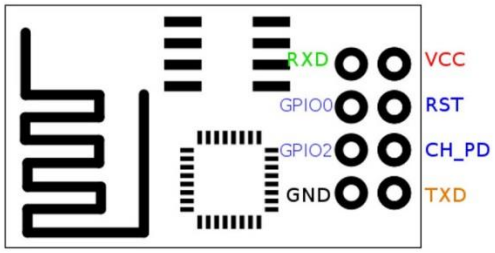


Εικόνα 10 – Esp8266-01

Το Esp8266 Wi-Fi Module είναι μια ολοκληρωμένη συσκεύη SOC (System On Chip) με ενσωματωμένο πρωτόκολλο TCP/IP που μπορεί να δώσει πρόσβαση στο Wi-Fi δίκτυο μας σε οποιονδήποτε μικροελεγκτή. Το Esp8266 είναι ικανό να φιλοξενήσει μία εφαρμογή είτε να εκφορτώσει όλες τις λειτουργίες δικτύωσης από έναν άλλο επεξεργαστή εφαρμογών. Κάθε Esp8266 έχει προ-προγραμματιστεί με ένα σύνολο εντολών AT, που σημαίνει, ότι μπορούμε εύκολα να το συνδέσουμε σε μία συσκευή Arduino και να έχουμε ικανότητα σύνδεσης στο Wi-Fi δίκτυο. Το Esp8266 είναι ένα εξαιρετικά χαμηλού κόστους αποτελεσματικό board με μία τεράστια και συνεχώς αναπτυσσώμενη κοινότητα.

Η συσκευή αυτή έχει αρκετά μεγάλη δυνατότητα επεξεργασίας και αποθήκευσης, που της επιτρέπει να ενσωματωθεί με σένσορες και άλλες συσκευές εφαρμογών μέσω των GPIOs

Το μοντέλο αυτό έχει αυτόνομη επεξεργαστική ισχύ καθώς και αποθηκευτική ικανότητα, αρκετή ώστε να οι οποίες του επιτρέπουν να ενσωματωθεί με αισθητήρες και ειδικές συσκευές εφαρμογών μέσω των GPIOs, με ελάχιστη εξέλιξη των αρχικών και ελάχιστη φόρτιση κατά την διάρκεια της εκτέλεσης. Το υψηλού βαθμού ολοκληρωμένο κύκλωμα του επιτρέπει την ελάχιστη χρήση εξωτερικών κυκλωμάτων, και έχει σχεδιαστεί για να καταλαμβάνει ελάχιστο χώρο σε μία PCB περιοχή. Το Esp8266 υποστηρίζει APSD για VoIP εφαρμογές και Bluetooth διεπαφές, περιέχει ένα αυτοβαθμονομημένο RF που του επιτρέπει να λειτουργεί υπό όλες τις συνθήκες, και δεν απαιτεί εξωτερικά μέρη RF.



Εικόνα 11 – Esp8266 GPIOs

Χαρακτηριστικά :

- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-Ap
- Ολοκληρωμένη στοίβα πρωτοκόλλου TCP/IP
- Ενσωματωμένος TR διακόπτης, balun, LNA, ενισχυτή ισχύος και αντιστοίχηση δικτύου
- Ενσωματωμένα PLLs, ρυθμιστές, DCXO και μονάδες διαχείρισης ισχύος
- +19.5dBm ισχύ εξόδου σε 802.11b λειτουργία
- Διακοπή λειτουργίας σε διαρροή ρεύματος <10μA
- 1MB Μνήμη Flash
- Ενσωματωμένο χαμηλής ισχύος 32-bit CPU που μπορεί να χρησιμοποιηθεί ως επεξεργαστής εφαρμογών
- SDIO 1.1 / 2.0, SPI, UART
- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4ms guard interval
- Wake up and transmit packets in < 2ms
- Κατανάλωση ενέργειας σε κατάσταση αναμονής < 1.0mW (DTIM3)

3.3 YHDC SCT-013-000 Current Transformer

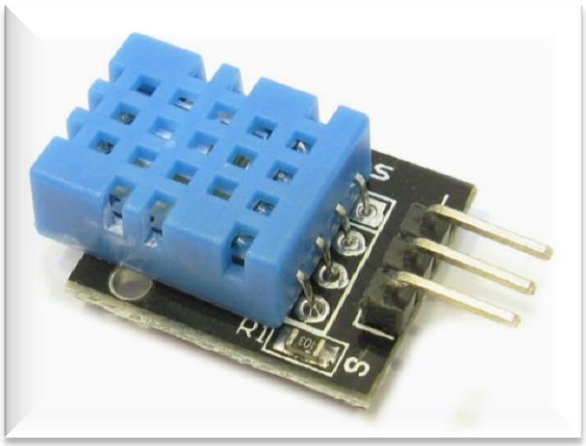
Εικόνα 12 – SCT-013

Ο αισθητήρας ρεύματος που χρησιμοποιήθηκε σε αυτή την εργασία είναι ο μετασχηματιστής ρεύματος SCT-013 της εταιρίας YHDC.
Ο πυρήνας του χωρίζεται και με αυτό τον τρόπο ο αισθητήρας αγκαλιάζει το καλώδιο που θέλουμε να μετρήσουμε το ρεύμα που το διαπερνά. Τέτοιου τύπου αισθητήρες ονομάζονται

non-invasive και είναι ιδανικοί για απλές εφαρμογές.

Όπως κάθε μετασχηματιστής έχει το πρωτεύον τύλιγμα, πυρήνα από φερρίτη και το δευτερεύον τύλιγμα. Το πρωτεύον πηνίο θα είναι η φάση ή ουδέτερος του καλωδίου που περνάει από την τρύπα του αισθητήρα, αλλά όχι και τα δύο μαζί διότι τα διαρρέουν ίσα ρεύματα αλλά σε διαφορετικές κατευθύνσεις, οπότε θα δημιουργηθούν δύο ίσα και αντίθετα μαγνητικά πεδία και θα ακυρώσουν το ένα το άλλο οπότε δεν θα έχουμε έξοδο στο δευτερεύον πηνίο. Για κάθε 100A στο πρωτεύον πηνίο επάγουν 50mA στο δευτερεύον, άρα το δευτερεύον πηνίο αποτελείται από 100/0.0050=2000 σπείρες.

3.4 DHT11 (Temperature – Humidity Sensor)



Εικόνα 13 – DHT11 Temperature-Humidity Sensor

Αυτός ο αισθητήρας DHT11 ελέγχει την θερμοκρασία και υγρασία καθώς διαθέτει αισθητήρα για μια βαθμονομημένη ψηφιακή έξοδο σήματος με τα αισθητήρια θερμοκρασίας και υγρασίας. Η τεχνολογία του εξασφαλίζει την υψηλή αξιοπιστία και εξαιρετική μακροχρόνια σταθερότητα. Ένας υψηλής απόδοσης 8-bit μικροελεγκτής είναι συνδεδεμένος. Αυτός ο αισθητήρας περιλαμβάνει ένα ωμικό στοιχείο και μια αίσθηση υγρής NTC μέτρησης της θερμοκρασίας. Έχει άριστη ποιότητα, γρήγορη απόκριση, ικανότητα αντί-παρεμβολών και τα πλεονεκτήματα του υψηλού κόστους απόδοσης.

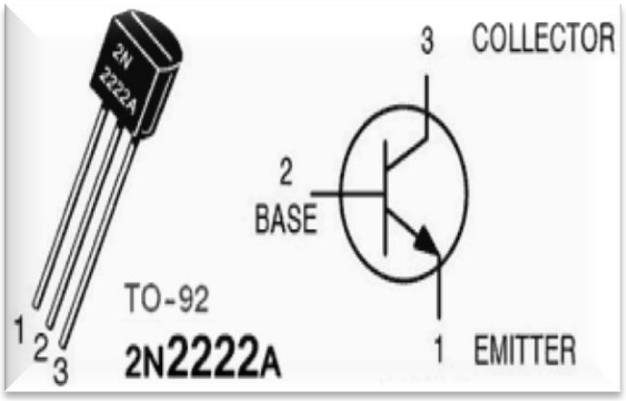
Χαρακτηριστικά:

- Λειτουργία : +5 V
- Πεδίο θερμοκρασίας : 0-50 °C σφάλμα ± 2°C
- Υγρασία : 20-90% RH ± 5% RH
- Σύνδεση : Ψηφιακή

3.5 Τρανζίστορ NPN 2N2222A

Το τρανζίστορ (transistor), κρυσταλλοτρίοδος και παλαιότερα κρυσταλλολυχνία, είναι διάταξη ημιαγωγών στερεάς κατάστασης, η οποία βρίσκει διάφορες εφαρμογές στην ηλεκτρονική, μερικές εκ των οποίων είναι η ενίσχυση, η σταθεροποίηση τάσης, η διαμόρφωση συχνότητας, η λειτουργία ως διακόπτης και ως μεταβλητή ωμική αντίσταση. Το τρανζίστορ μπορεί, ανάλογα με την τάση με την οποία πολώνεται, να ρυθμίζει την ροή του

ηλεκτρικού ρεύματος που απορροφά από συνδεδεμένη πηγή τάσης. Τα τρανζίστορ κατασκευάζονται είτε ως ξεχωριστά ηλεκτρονικά εξαρτήματα είτε ως τμήματα κάποιου ολοκληρωμένου κυκλώματος. Για εμάς το τρανζίστορ χρησιμοποιήθηκε ως διακόπτης που ανοιγοκλείνει την τροφοδοσία ενός led υπερύθρων ανάλογα με τις ανάγκες μας.



Εικόνα 14 – Transistor NPN Connections

3.6 I.R emitter (υπέρυθρος πομπός)

Εικόνα 15 – IR emitter

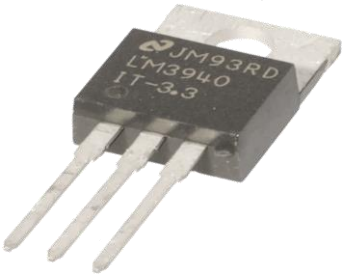
Ο υπέρυθρος πομπός χρησιμοποιείται για να μεταδίδει σήματα υπέρυθρων μέσω υπέρυθρων LED, ενώ υπάρχει ένας δέκτης υπέρυθρων για να πάρει τα σήματα από την άλλη πλευρά. Ένα υπέρυθρο LED είναι σαν οποιοδήποτε άλλο LED, με το χρώμα του επικεντρώνεται γύρω από 940nm.Ο πομπός μπορεί να χρησιμοποιηθεί για τη μετάδοση δεδομένων ή εντολών, αλλά και για να μιμηθεί τηλεχειριστήρια για τον έλεγχο της συσκευής στο σπίτι χρησιμοποιώντας ένα Arduino. Ο πομπός υπέρυθρων μπορεί να μεταδώσει σήματα αξιόπιστα μέχρι και 10 μέτρα. Πέρα από τα 10 μέτρα, ο δέκτης δεν μπορεί να πάρει τα σήματα.

3.7 IR receiver (υπέρυθρος δέκτης)

Εικόνα 16 – IR receiver

Ο δέκτης υπέρυθρων χρησιμοποιείται για τη λήψη υπέρυθρων σημάτων και χρησιμοποιείται επίσης για την ανίχνευση εξ αποστάσεως ελέγχου. Υπάρχει ένας ανιχνευτής IR στον δέκτη υπέρυθρων που χρησιμοποιείται για να πάρει το υπέρυθρο φως που εκπέμπεται από τον πομπό υπέρυθρων. Ο ανιχνευτής IR έχει αποδιαμορφωτή μέσα που αναζητά διαμορφωμένο IR σε 38 KHz. Ο δέκτης υπέρυθρων μπορεί να λάβει σήματα και σε απόσταση 10 μέτρων. Εάν είναι περισσότερα από 10 μέτρα, ο δέκτης δεν μπορεί να πάρει τα σήματα.

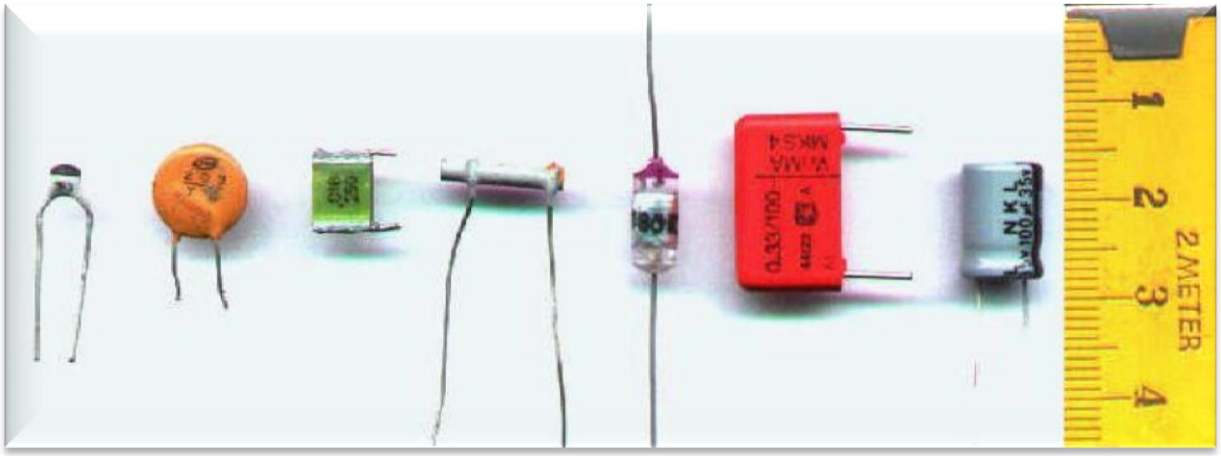
3.8 Ρυθμιστής Τάσης



Εικόνα 17 – Voltage Regulator

Ο ρυθμιστής τάσης είναι ηλεκτρικός ρυθμιστής σχεδιασμένος για να διατηρήσει αυτόματα μια τάση σταθερή.
Μπορεί να χρησιμοποιήσει έναν ηλεκτρομηχανικό μηχανισμό, ή ενεργητικά ή ενεργά ηλεκτρονικά συστατικά. Ανάλογα με το σχέδιο, μπορεί να χρησιμοποιηθεί για να ρυθμίσει Εναλλασσόμενο ρεύμα ή Συνεχές ρεύμα.
Με εξαίρεση τους ρυθμιστές διακλαδώσεων, όλοι οι ρυθμιστές τάσης λειτουργούν με τη σύγκριση της πραγματικής τάσης παραγωγής με κάποια εσωτερική σταθερή τάση αναφοράς. Οποιαδήποτε διαφορά ενισχύεται και χρησιμοποιείται για να ελέγξει το στοιχείο κανονισμού. Αυτό διαμορφώνει το α αρνητική ανατροφοδότηση σερβο βρόχος ελέγχου. Εάν η τάση παραγωγής είναι πάρα πολύ χαμηλή, το στοιχείο κανονισμού διατάζεται για να παραγάγει μια υψηλότερη τάση. Εάν η τάση παραγωγής είναι πάρα πολύ υψηλή, το στοιχείο κανονισμού διατάζεται για να παραγάγει μια χαμηλότερη τάση. Κατ' αυτό τον τρόπο, η τάση παραγωγής κρατιέται κατά προσέγγιση σταθερή. Ο βρόχος ελέγχου πρέπει να σχεδιαστεί προσεκτικά για να παραγάγει την επιθυμητή ανταλλαγή μεταξύ της σταθερότητας και της ταχύτητας της απάντησης.

3.9 Πυκνωτές



Εικόνα 18 - Capacitors

Πυκνωτής (συμβ. C) ονομάζεται ένα σύστημα δύο γειτονικών αγωγών ανάμεσα στους οποίους παρεμβάλλεται μονωτικό υλικό. Αυτό το μονωτικό υλικό μπορεί να είναι αέρας, πλαστικό, μίκα κ.α. Οι δύο αγωγοί ονομάζονται οπλισμοί του πυκνωτή, ενώ το παρεμβαλλόμενο υλικό ονομάζεται διηλεκτρικό του πυκνωτή. Βασικό χαρακτηριστικό κάθε πυκνωτή είναι η ιδιότητά του να αποθηκεύει ηλεκτρικό φορτίο, επομένως ηλεκτρική ενέργεια. Όταν ένας πυκνωτής είναι φορτισμένος, οι οπλισμοί του έχουν ηλεκτρικά φορτία κατά μέτρο ίσα και αντίθετα. Ονομάζουμε φορτίο του πυκνωτή (Qc) το φορτίο του θετικά φορτισμένου οπλισμού του.
Μεταξύ των οπλισμών ενός φορτισμένου πυκνωτή αναπτύσσεται διαφορά δυναμικού, την οποία ονομάζουμε τάση του πυκνωτή (Vc).

Το σταθερό πηλίκο του φορτίου ενός πυκνωτή προς την τάση του ονομάζεται χωρητικότητα του πυκνωτή και συμβολίζεται με το αγγλικό γράμμα C, που είναι το αρχικό γράμμα τη λέξης Capacity:

Ισχύει ότι: $C = Q / V$

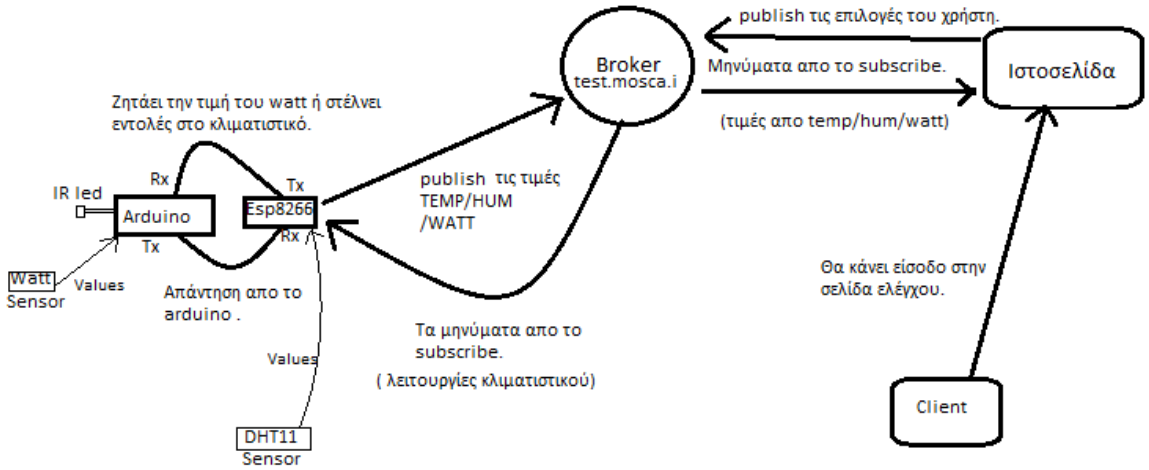
Μονάδα μέτρησης της χωρητικότητας του πυκνωτή είναι το 1 Φαράντ Farad (F). Πρόκειται όμως για μεγάλη μονάδα, που σπάνια χρησιμοποιείται στην πράξη. Συνήθως χρησιμοποιούνται τα υποπολλαπλάσιά του: μικροφαράντ (μF), νανοφαράντ (nF) και πικοφαράντ (pF).

3.10 Αντιστάτες

Εικόνα 19 - Transistor

Ο αντιστάτης είναι ένα ηλεκτρολογικό/ηλεκτρονικό εξάρτημα το οποίο χρησιμοποιείται σε διάφορα κυκλώματα για τον έλεγχο της ροής του ρεύματος. Ο αντιστάτης μερικές φορές λέγεται και ηλεκτρική αντίσταση, αν και ηλεκτρική αντίσταση είναι ένα φαινόμενο. Συχνές συνδεσμολογίες αντιστατών που συναντά κανείς στα ηλεκτρικά ή ηλεκτρονικά κυκλώματα είναι η σύνδεση αντιστάσεων σε σειρά και η σύνδεση αντιστάσεων παράλληλα. Η πρώτη συνδεσμολογία ονομάζεται διαιρέτης τάσης, ενώ η δεύτερη ονομάζεται διαιρέτης ρεύματος.

4. Υλοποίηση Εργασίας



Ξεκινώντας από δεξιά βλέπουμε ότι αρχικά είναι η σύνδεση του χρήστη με την ιστοσελίδα/σέρβερ μας. Εφόσον ο χρήστης έχει κάνει την σύνδεση του, τότε μέσω της βιβλιοθήκης mqtt ο σέρβερ κάνει εγγραφή στο μονοπάτι που έχουμε προκαθορίσει για τις τιμές των αισθητηρίων από τον client (υγρασίας, θερμοκρασίας και κατανάλωσης ισχύος). Αφού έχει κάνει την εγγραφή τότε είναι έτοιμος να δημοσιεύσει τις επιλογές του χρήστη.

Από την αριστερή μεριά βλέπουμε ότι το ESP8266 κάνει subscribe στο μονοπάτι που ο σέρβερ δημοσιεύει τις επιλογές του χρήστη και εφόσον χρειαστεί κάνει δημοσίευση τις τιμές των sensors που έχουμε. Καθώς το esp δεν διαθέτει αναλογική είσοδο, τις μετρήσεις για τα watt που καταναλώνει το κλιματιστικό το esp τις παίρνει από το Arduino. Παράλληλα, στέλνει στο Arduino, μέσω των pins που έχουμε δηλώσει (Rx , Tx), τις επιλογές του χρήστη ως προς τη λειτουργία του κλιματιστικού, καθώς το esp δεν διαθέτει pwm για να στείλουμε τις εντολές στον IR emitter κατ' ευθείαν.

Εφόσον έχουν εκτελεστεί ορθά όλες οι επικοινωνίες, το esp8266 δημοσιεύει τις τιμές που πήρε από τα αισθητήρια και παράλληλα ειδοποιεί τη σελίδα μας για το αν η εντολή εκτελέστηκε ή όχι. Όλες αυτές τις τιμές η σελίδα τις παίρνει από το μονοπάτι που έχει κάνει εγγραφή στις δημοσιεύσεις του client.

4.1 Εγκατάσταση MQTT Broker

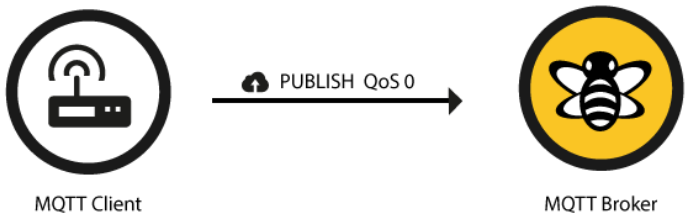
Παρακάτω θα δούμε πώς να κάνουμε εγκατάσταση του διακομιστή και τι προσφέρει ο διακομιστής που διαλέξαμε.

4.1.1 Επεξήγηση χρησιμότητας

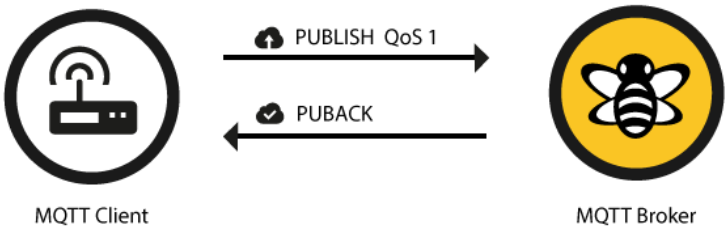
Η εγκατάσταση του διακοσμητή είναι αναγκαία για να γίνει η χρήση του πρωτοκόλλου , αν στέλναμε μηνύματα στον server χωρίς την χρήση του MQTT θα χρειαζόταν να στέλνουμε μηνύματα συνέχεια για να μας πει αν κάποια τιμή έχει αλλάξει ή όχι ,αυτό θα είχε σαν αποτέλεσμα ο server να δέχεται πάρα πολλά μηνύματα ανά λεπτό και κάποια από αυτά τα μηνύματα είναι πιθανό να χαθούν στην αποστολή η ακόμα και να μην φτάσουν ποτέ αν ο server μας δεν είναι σε λειτουργεί έστω και για λίγο.

Με την χρήση του πρωτοκόλλου MQTT έχουμε την δυνατότητα να εφαρμόσουμε την ποιότητα υπηρεσιών (QoS) το οποίο μας προσφέρει τρεις επιλογές :

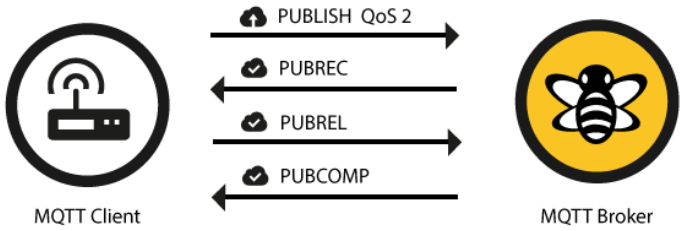
QoS (0) : Η επιλογή 0 είναι η μικρότερη και θα στείλει το μήνυμα μας το πολύ μια φορά αλλά δεν θα σιγουρευτεί για το αν έφτασε η όχι.



QoS (1) : Η επιλογή 1 το μήνυμα θα σιγουρευτεί ότι θα φτάσει τουλάχιστον μια φορά αλλά μπορούμε να στείλουμε το μήνυμα για πάνω από μια φορά . Μετά το Publish ο client μας περιμένει σαν απάντηση το PUBACK από τον Broker για να σιγουρευτεί ότι το μήνυμα στάλθηκε, αν το μήνυμα δεν έρθει μετά από κάποιο χρονικό διάστημα τότε ο client θα ξαναστείλει το μήνυμα.



QoS (2) : Η επιλογή 2 εγγυάται ότι κάθε μήνυμα θα το λάβει μόνο μια φορά . Είναι ο ασφαλέστερος τρόπος αλλά επίσης είναι και ο πιο αργός τρόπος . Η εγγύηση παρέχεται από δύο ροές μηνυμάτων μεταξύ αποστολέα και παραλήπτη.



Παράλληλα για τους πελάτες που είναι εγγεγραμμένοι σαν επίμονη συνεδρία (Persistent Session) αν είναι ανενεργοί εκείνη την στιγμή που θα έρθει ένα μήνυμα για αυτούς τότε το μηνύματα τους θα αποθηκευτεί σε μια σειρά για να τους σταλθεί ,στέλνονται μόνο τα μηνύματα που είναι QoS 1,2 .

4.1.2 Τρόπος εγκατάστασης

Η εγκατάσταση θα γίνει στα windows 10 , θα χρησιμοποιήσουμε τον διακοσμητή του Eclipse,τον Mosquito MQTT.

Για την εγκατάσταση θα πρέπει να :

πάμε στο http://mosquitto.org/download/ και κατεβάζουμε το αρχείο mosquitto-1.4.10-install-win32.exe το οποίο περιέχει όλα τα αρχεία που χρειάζονται για την εγκατάσταση.

Εφόσον έχουμε κάνει την εγκατάσταση τότε ο broker μας είναι έτοιμος και "ακούει" στις εξής θύρες :

1883 : MQTT, χωρίς κρυπτογράφηση.
8883 : MQTT, με κρυπτογράφηση.
8884 : MQTT, με κρυπτογράφηση , απαιτείται πιστοποιητικό πελάτη.
8080 : MQTT μέσω WebSockets, χωρίς κρυπτογράφηση.
8081 : MQTT μέσω WebSockets, με κρυπτογράφηση.

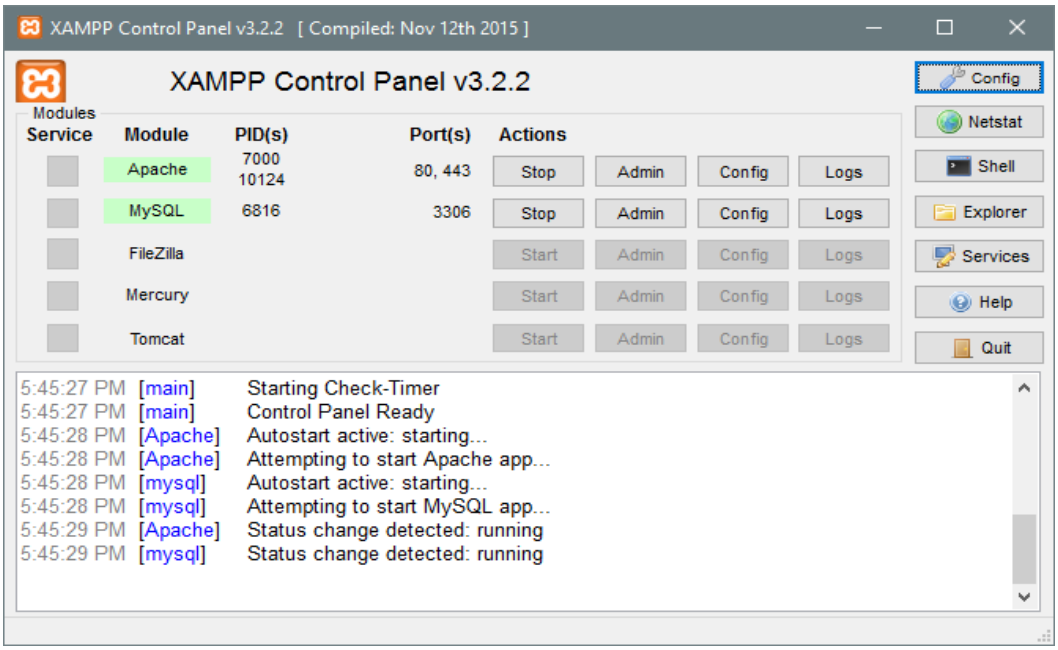
Οι κρυπτογραφημένες θύρες υποστηρίζουν TLS v1.2, v1.1 ή v1.0 με πιστοποιητικά X509 και απαιτούν υποστήριξη του πελάτη για να συνδεθείτε. Σε όλες τις περιπτώσεις θα πρέπει να επισκεφτούμε τις διευθύνσεις (mosquitto.org.crt (PEM format), ή mosquitto.org.der (DER format)) για να κατεβάσουμε τα πιστοποιητικά που χρειάζονται για να επαληθευτεί η σύνδεση.
Εμείς θα χρησιμοποιήσουμε την θύρα **1883** στην περίπτωσή μας.

4.2 Εγκατάσταση X.A.M.P.P

Η εγκατάσταση του XAMPP θα γίνει για να μπορέσουμε να στήσουμε την ιστοσελίδα και την βάση που χρειαζόμαστε.
Για την εγκατάσταση του θα πρέπει να επισκεφτούμε τον ιστότοπο του XAMPP (www.apachefriends.org) και να κατεβάσουμε την επιλογή για τα windows.
Εφόσον έχουμε κάνει την εγκατάσταση μπορούμε να ανοίξουμε τον πίνακα ελέγχου το προγράμματος , στην συνέχεια διαλέγουμε την επιλογή START στον Apache και MySQL ,μόλις οι υπηρεσίες αυτές ενεργοποιηθούν τότε αριστερά από την επιλογή μπορούμε να δούμε και ποια θύρα χρησιμο ,ο Apache θα χρησιμοποιηθεί για να φιλοξενήσει τον το ιστότοπο μας και ο προκαθορισμένος φάκελος του είναι:

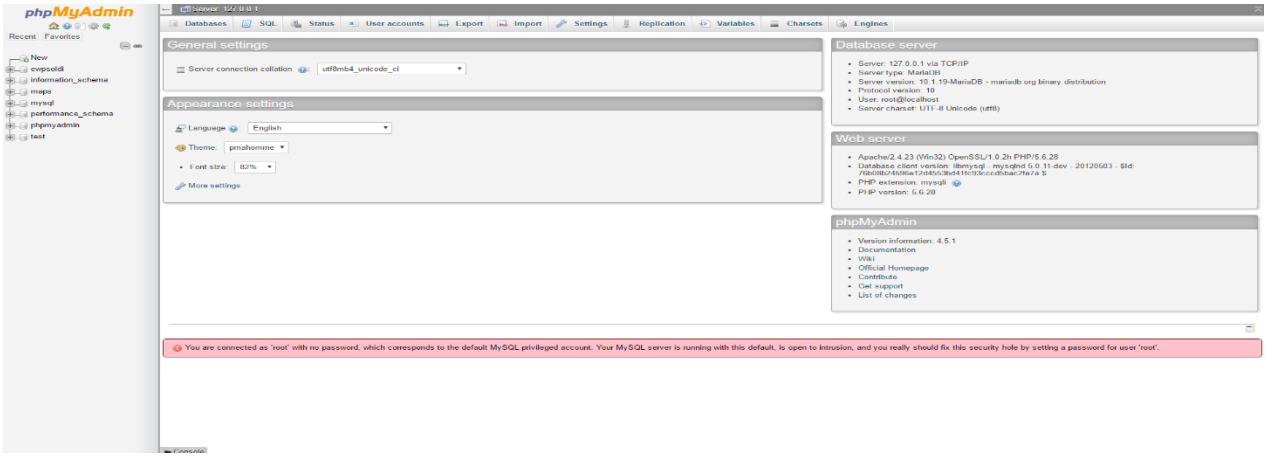
C:\xampp\htdocs
Για να επισκεφτούμε το αρχική σελίδα που έχουμε βάλει στο μονοπάτι από πάνω θα πρέπει να ανοίξουμε έναν browser και να πάμε στο:

localhost/



4.3 Στήσιμο Βάσης Δεδομένων

Για την δημιουργία της βάσης μας θα πρέπει να επισκεφτούμε την προκαθορισμένη τοποθεσία:
http://localhost/phpmyadmin/ στην οποία υπάρχει η διεπαφή για τον διαχειριστή, αν θελήσουμε να συνδεθούμε στην βάση μέσω PuTTY τότε θα φροντίσουμε να μην έχουμε κάνει κάποια αλλαγή στους φακέλους με τις ρυθμίσεις για την είσοδο ή αν έχουμε κάνει τότε να τις ξέρουμε καθώς μπορεί να έχει γίνει αλλαγή του προκαθορισμένου username και password τα οποία είναι αναγκαία.
Στην τοποθεσία θα δούμε το εξής:



Αριστερά έχουμε όλες τις βάσεις μας , για την δημιουργία νέας βάσης θα πρέπει να :

- Πάμε στην πρώτη επιλογή αριστερά – New
- Διαλέγουμε το όνομα που θα έχει η βάση μας στην επόμενη καρτέλα και πόσα γραμμές ,κλειδιά και ονόματα γραμμών.
- Έχουμε την επιλογή να δούμε την SQL μορφή των επιλογών μας με το κουμπί « Preview SQL » κάτω δεξιά στην σελίδα.

- Μπορούμε πάντα να προσθέσουμε και άλλη ή άλλες γραμμές στην βάση σε περίπτωση που δεν φτάνει ο αρχικός αριθμός που διαλέξαμε με την επιλογή «GO» από πάνω.
- Όταν έχουμε συμπληρώσει ότι μας χρειάζεται τότε διαλέγουμε την επιλογή « SAVE» κάτω δεξιά.

Η τελική μορφή που θα έχει η βάση θα είναι η εξής:

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	id	int(11)			No	None	AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
2	place	varchar(25)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
3	ac	varchar(25)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
4	x_value_start	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
5	x_value_end	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
6	y_value_start	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
7	y_value_end	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
8	value	varchar(5)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
9	x_control_start	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
10	x_control_end	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
11	y_control_start	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
12	y_control_end	int(11)			No	None		Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns
13	updated_at	timestamp		on update CURRENT_TIMESTAMP	No	0000-00-00 00:00:00	ON UPDATE CURRENT_TIMESTAMP	Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns

και το όνομα που διαλέξαμε για την βάση μας είναι : “ewpsoldi_arduino” .
Πιο αναλυτικά :

- id : θα είναι ένας ακέραιος που θα δίνεται αυτόματα από server
- place : θα είναι το κτίριο στο οποίο βρίσκονται στημένα τα Arduino με τα Esp8266.
- ac : θα είναι το δωμάτιο που βρίσκεται η κάθε συσκευή.
- x/y_value_start/end : Είναι οι συντεταγμένες για τα την αρχική σελίδα που θα δείχνει αν κάποιο κλιματιστικό είναι ανοιχτό, τα χ,y δείχνουν που ξεκινάει και που τελειώνει το κτίριο.
- value: Είναι η τιμή που θα έχει το κλιματιστικό κάθε φορά, ισχύει off για κλειστό και on για ανοιχτό.
- x/y_control_start/end: Είναι οι συντεταγμένες για να ζωγραφίσουμε το κάθε δωμάτιο σε περίπτωση που επιλέξουμε να το ελέγξουμε.
- update: θα κρατάει την τελευταία φορά που έγινε αλλαγή στην τιμή value , έχουμε βάλλει η τιμή αυτή να αλλάζει αυτόματα από τον σέρβερ.

Και όταν θα γίνεται είσοδο τιμών θα φαίνεται ως εξής:

	id	place	ac	x_value_start	x_value_end	y_value_start	y_value_end	value	x_control_start	x_control_end	y_control_start	y_control_end	updated_at
Edit Copy Delete	1	enpet	grafeio	20	50	70	100	off	14	277	12	132	2017-01-15 11:24:03
Edit Copy Delete	2	enpet	ergastirio	20	50	70	100	off	14	445	264	222	0000-00-00 00:00:00

Παράλληλα θα δημιουργήσουμε και μια βάση για τους χρήστες με όνομα “ users ” τις νέες

καταχωρίσεις στην βάση θα πρέπει να τις δημιουργήσουμε εμείς, οι δυνατότητες που θα έχει ο κάθε χρήστης είναι οι εξής:

- Ο κάθε χρήστης θα έχει ένα όνομα.
- Θα έχει έναν κωδικοποιημένο κωδικό ασφαλείας.
- Θα έχει ένα email σε περίπτωση που χρειαστεί επικοινωνία με κάποιον χρήστη για κάποιο λάθος.
- Θα έχει και τον ιστότοπο που θα μπορεί να επισκεφτεί.
- Θα μπορεί να κάνει έλεγχο σε συγκεκριμένο κτίριο μόνο και όχι όπου θελήσει αυτός.
- Θα κρατάμε το μέρος το οποίο μπορεί να publish στον MQTT διακομιστή μας.

Σύμφωνα με τα παραπάνω στοιχεία ο πίνακας μας θα έχει την παρακάτω μορφή :

4.4 Δημιουργία σελίδας ελέγχου

Παρακάτω θα δούμε πως θα γίνει το στήσιμο του site μας για να μπορούμε να ελέγχουμε τα κλιματιστικά.

4.4.1 Δημιουργία index.php

Η σελίδα index.php θα είναι η πρώτη σελίδα που θα βλέπει ο χρήστης ή ο διαχειριστής , θα είναι υπεύθυνη για να μας δείχνει αν κάποιο από τα κλιματιστικά που έχουμε συνδέσει είναι ανοιχτό και σε ποιο χώρο του Τ.Ε.Ι είναι, επίσης θα είναι υπεύθυνο για να μας καθοδηγεί στο επόμενο κομμάτι που θα είναι η σύνδεση στον χώρο που διαλέξαμε να ελέγξουμε.

4.4.1.1 Δημιουργία Canvas

Το στοιχείο canvas χρησιμοποιείται για να δημιουργήσουμε γραφικά στην σελίδα μας, μέσω JavaScript.
Ο canvas είναι ένα απλό δοχείο για τα γραφικά μας , χωρίς τον κώδικα σε javascript δεν υπάρχει κάτι που να φαίνεται από μόνο του. Παράλληλα ο έχει αρκετές μεθόδους που μπορούμε να καλέσουμε για να δημιουργήσουμε μονοπάτια ,κύκλους ,κείμενα και την πρόσθεση εικόνας.

Ένα παράδειγμα δημιουργίας ενός κενού canva είναι το εξής :

<canvas id="myCanvas" width="200" height="100"></canvas>
Πιο πάνω δημιουργήσαμε έναν canva με μήκος 200 pixels επί 100 pixel ο οποίος είναι κενός, ο λόγος που του δώσαμε και ένα id είναι για να μπορούμε στην συνέχεια να τον τροποποιήσουμε όπως θέλουμε εμείς.

4.4.2 Σύνδεση με την Βάση Δεδομένων

Για να μπορούμε να ελέγχουμε σε ποιο κτίριο και ποιο κλιματιστικό είναι ανοιχτό θα πρέπει να κάνουμε σύνδεση με την βάση δεδομένων πρώτα, εφόσον έχει γίνει η σύνδεση τότε πρέπει να ζητήσουμε τις τιμές που έχουν τα κλιματιστικά και έπειτα η απάντηση που θα πάρουμε κάθε φορά θα πρέπει να επεξεργαστεί έτσι ώστε να μπορέσουμε να βγάλουμε μια γενική τιμή για κάποιον από τους χώρους.

4.4.2.1 Δημιουργία dbconnect.php

Αρχικά θα πρέπει να δηλώσουμε τέσσερις μεταβλητές οι οποίες θα είναι οι εξής:

- **$servername** : εδώ να δηλώσουμε το μέρος στο οποίο βρίσκεται η βάση μας, δηλαδή την διεύθυνση που έχει. Αν τρέχουμε τον σέρβερ τοπικά τότε η τιμή που θα έχει είναι "**localhost**" , σε περίπτωση όμως που ο σέρβερ δεν είναι δικός μας και απλά πληρώνουμε κάπου αλλού για την φιλοξενία της ιστοσελίδας μας τότε θα έχει την μορφή **" https://gator4088.hostgator.com:2083 "** και αυτά τα στοιχεία μας δίνονται από τον ιδιοκτήτη.
- **$username** : είναι το όνομα που απαιτείτε για την είσοδο στην βάση του σέρβερ που έχουμε δηλώσει στο πεδίο $servername , όπως και πιο πάνω αυτό ή θα μας έχει δοθεί ή θα είναι κάτι που έχουμε βάλλει εμείς.
- **$password** : είναι το κωδικός που απαιτείτε για την είσοδο στην βάση και ισχύει ότι και για τον $username.
- **$dbname** : θα είναι το όνομα της βάσης στο οποίο θέλουμε να συνδεθούμε, εφόσον πιο πάνω δημιουργήσαμε τοπικά την βάση με το όνομα " **ewpsoldi_arduino** " τότε αυτό είναι και το όνομα της βάσης που θα χρησιμοποιήσουμε γιατί έχει τις τιμές για τα κλιματιστικά αλλά και τον χώρο στον οποίο βρίσκονται.

Στην περίπτωση μας θα φαίνονται ως εξής :

- $**servername** = "localhost";
- $**username** = "root";
- $**password** = "";
- $**dbname** = "ewpsoldi_arduino";

Στην συνέχεια θα πρέπει με την εντολή " mysqli_connect " να ξεκινήσουμε την σύνδεση στον σέρβερ με την βάση που έχουμε επιλέξει.
Έπειτα αυτή η σύνδεση πρέπει να καταχωρείτε σε κάποια μεταβλητή ώστε να μπορούμε να ζητήσουμε δεδομένα αλλά και για να μην χάσουμε την σύνδεση που δημιουργήσαμε, παράλληλα ελέγχουμε αν και η σύνδεση είναι εφικτή καθώς μπορεί τα στοιχεία που έχουμε δώσει να είναι λάθος ή ο σέρβερ να μην είναι προσβάσιμος εκείνη της στιγμή. Έτσι με αυτόν τον τρόπο θα μάθουμε αν μπορούμε να συνεχίσουμε στον επόμενο κομμάτι του κώδικα θα πρέπει να σταματήσουμε και περιμένουμε να γίνει η σύνδεση με τον σέρβερ, αυτό θα γίνει με τον παρακάτω κώδικα:

```
if (!($con= mysqli_connect($servername, $username, $password, $dbname))) {
   die('oops connection problem ! --> ' . mysql_error());
}
```

Εφόσον έχουμε κάνει όλα τα παραπάνω στην σελίδα dbconnect.php τότε είμαστε έτοιμη για να βάλουμε την σύνδεση στην index.php,αυτό θα γίνει με την εντολή :

```
include_once 'dbconnect.php';
```

αν η σελίδα dbconnect.php βρισκόταν σε άλλο φάκελο τότε έπρεπε να προσδιορίσουμε το μονοπάτι που βρίσκεται, έστω για παράδειγμα ότι ο βρισκόταν σε έναν άλλο φάκελο που λεγόταν " connect " το μονοπάτι θα γινόταν include_once '/connect/dbconnect.php';

4.4.3 Έλεγχος τιμών κλιματιστικού.

Εφόσον έχουμε φορτώσει την βάση θα πρέπει να κάνουμε ένα mysqli_query στο οποίο θα ζητάμε από τα κλιματιστικά την τιμή που έχουν, και θα φροντίσουμε να έρχονται με βάση το μέρος που βρίσκονται για να μπορέσουμε να καταλάβουμε αν κάποιο από τα κλιματιστικά που βρίσκονται εκεί είναι ανοιχτό ή αν όλα είναι κλειστά , αν κάποιο από τα κλιματιστικά

είναι ανοιχτό σε ένα κτίριο τότε γύρο από το κτίριο θα γίνεται πράσινο αλλιώς αν όλα είναι κλειστά τότε θα γίνεται κόκκινο.

Ξεκινάμε με το mysqli_query το οποίο θα είναι ως εξής:

$sql = mysqli_query($con, "SELECT * FROM klimatistika ORDER BY `klimatistika`.`place` ASC");

Η εντολή mysqli_query θέλει σαν παράμετρο την σύνδεση που έχουμε ανοίξει, στην περίπτωση μας το $con , παράλληλα θέλει και τον SQL κώδικα. Η απάντηση αποθηκεύεται στην μεταβλητή $sql .

4.4.3.1 Δημιουργία script για έλεγχο τιμών.

Εφόσον στον πιο πάνω κώδικα έχουμε ζητήσει από την βάση να φέρει τις τιμές πρέπει στην συνέχεια αυτές οι τιμές να ελέγχονται ώστε να διαπιστώσουμε αν το κλιματιστικό στον χώρο είναι ανοιχτό η κλειστό. Θα πρέπει σε μια While συνθήκη να ζητάμε κάθε μια τις γραμμές από την απάντηση που θα λάβουμε από την βάση μας σύμφωνα με το query το οποίο ζητήσαμε, αυτό θα γίνει με την εντολή mysqli_fetch_array($sql) επειδή όμως δεν ξέρουμε πόσες είναι ακριβώς οι γραμμές θα πάμε με μια while και η εντολή θα γίνει :

while ($row = mysqli_fetch_array($sql)) { }

όταν η εντολή mysqli_fetch_array($sql) δεν έχει άλλη γραμμή για να γυρίσει τότε θα επιστρέψει false και θα σταματήσει η while μας.

Αφού έχουμε δημιουργήσει την while μέσα θα πρέπει να κάνουμε κάποιους ελέγχους,
επειδή στην $sql οι τιμές έρχονται ανάλογα το μέρος στο οποίο βρίσκονται τα κλιματιστικά θα πρέπει να αποθηκεύσουμε το τελευταίο χώρο που ελέγξαμε σε μια μεταβλητή, μόλις αλλάξει αυτή η μεταβλητή καταλαβαίνουμε ότι παίρνουμε τιμές για διαφορετικό χώρο παράλληλα θα πρέπει να κρατάμε και τις τιμές που ξεκινάει και τελειώνει ο χώρος ώστε να μπορέσουμε να το ζωγραφίσουμε στον canva μας και σε μορφή κώδικα θα φαίνεται ως εξής :

```
$lastPlace = "enpet";
$placeValue = "random";
$lastPlace_x_start;
$lastPlace_x_end;
$lastPlace_y_start;
$lastPlace_y_end;
```

Και μέσα στην while:

Αρχικά θα ελέγχουμε μέρος που λάβαμε είναι το ίδιο με το προηγούμενο και αν είναι το ίδιο και η τιμή του είναι “ON” τότε η τιμή του μέρους θα γίνει “ON” αλλιώς θα παραμείνει “OFF” , εφόσον γίνει αλλαγή στο μέρος τότε θα αλλάζει η τιμή του $lastPlace και θα γίνεται ίση με το νέο μέρος και η τιμή του $placeValue θα γίνει ίση με την πρώτη τιμή του χώρου, όταν θα γίνεται αλλαγή του χώρου θα πρέπει να ζωγραφίσουμε και στον καμβά μας το κτίριο ανάλογα την τιμή που είχε το $placeValue.
Σε μορφή κώδικα θα φαίνεται ως εξής:

```
if ($row['place'] === $lastPlace) {
  if ($row['value'] === 'on') {
          $placeValue = "on";
       } else {
          if ($placeValue != "on") {
             $placeValue = "off";
          }
       }
       // HOLD LAST VALUE SO YOU CAN PAINT IT
       $lastPlace_x_start = $row['x_value_start'];
       $lastPlace_x_end = $row['x_value_end'];
       $lastPlace_y_start = $row['y_value_start'];
       $lastPlace_y_end = $row['y_value_end'];
    } else {
       if ($placeValue === "on") {
          $paint = "green";
       } else {
          $paint = "red";
       }
       if ($row['value'] === 'on') {
          $placeValue = "on";
       } else {
          $placeValue = "off";
       }
       $lastPlace = $row['place'];
       ?>
                    context.lineWidth = 6;
                    context.strokeStyle = "<?php echo $paint ?>";
                    context.strokeRect(<?php echo $lastPlace_x_start ?>,<?php echo $lastPlace_y_start ?>,
     <?php echo ($lastPlace_x_end - $lastPlace_x_start); ?>, <?php echo ($lastPlace_y_end -
$lastPlace_y_start);                ?>);
       <?php
    }
} if ($placeValue === "on") {
    $paint = "green";
} else {
    $paint = "red";
}
if ($row['value'] === 'on') {
    $placeValue = "on";
} else {
    $placeValue = "off";
}
?>
              context.lineWidth = 6;
              context.strokeStyle = "<?php echo $paint ?>";
              context.strokeRect(<?php echo $lastPlace_x_start ?>,<?php echo $lastPlace_y_start ?>,
   <?php echo ($lastPlace_x_end - $lastPlace_x_start); ?>, <?php echo ($lastPlace_y_end -
$lastPlace_y_start); ?>);
          }
       });
```

Οι εντολές κάνουν τα εξής :

- **context.lineWidth** : Εδώ θα δηλώσουμε το πάχος που θα έχει η γραμμή που θα ζωγραφίσουμε.
- **context.strokeStyle** : Εδώ δηλώνουμε το χρώμα που θέλουμε να έχει η γραμμή μας.
- **context.strokeRect** : Με αυτή την εντολή λέμε που θα ζωγραφίσουμε πάνω στον καμβά μας και παίρνει τέσσερις μεταβλητές σας ορίσματα :

    1) απο που ξεκινάμε να ζωγραφίζουμε στον X άξονα.
    2) από πού ξεκινάμε να ζωγραφίζουμε στον Y άξονα.
    3) Που θα σταματήσουμε να ζωγραφίζουμε στον X άξονα.
    4) Που θα σταματήσουμε να ζωγραφίζουμε στον Y άξονα.

4.4.5 Δημιουργία σελίδας εισόδου χρήστη (login.php)

Σε αυτή την σελίδα θα μεταφέρεται ο χρήστης εφόσον έχει κάνει επιλογή σε κάποιο από τα κτίρια που έχουν τοποθετηθεί τα Arduino με τα esp8266, θα την ονομάσουμε login.php.

Θα πρέπει να δημιουργήσουμε εμείς τους χρήστες που θέλουμε να έχουν πρόσβαση στους χώρους, εφόσον έχει γίνει αυτό το κομμάτι σύμφωνα με τον από πάνω τρόπο τότε μπορούμε να περάσουμε στον τρόπο επαλήθευσης τον στοιχείων , εφόσον θα γράψουμε κώδικα σε php για την επαλήθευση όλα θα γίνονται από την μεριά του σέρβερ κανείς δεν θα μπορεί να δει τον κώδικα που γράψαμε.

Αρχικά θα φτιάξουμε μια απλή φόρμα σε HTML κώδικα για να μπορεί ο χρήστης να βάλει τα στοιχεία του στα πεδία που θα του δώσουμε, θα έχει την εξής μορφή:

```
<form method="post">
        <input type="text" name="username" placeholder="Username" required="required" />
        <input type="password" name="pass" placeholder="Password" required="required" />
        <button type="submit" name="btn-login"  >Login</button>
</form>
```

Στην αρχή της φόρμας δηλώνουμε ότι είναι μορφή “post” δηλαδή τα στοιχεία που θα στείλουμε δεν θα φαίνονται πουθενά, γιαυτό και τα ονόματα που θα δώσουμε στα πεδία του input είναι πολύ σημαντικά για τον χειρισμό τον μεταβλητών. Στο πεδίο name θα πρέπει να βάλουμε το όνομα που θα έχει η μεταβλητή όταν την στείλουμε και με το required δηλώνουμε ότι αυτό το πεδίο πρέπει να εχει συμπληρωθεί για να μπορέσει ο χρήστης να πατήσει το κουμπί Login που βρίσκεται κάτω από την φόρμα.
Επόμενο βήμα είναι ο έλεγχος των τιμών που μας δόθηκαν , αυτό θα γίνει στην ίδια σελίδα.
Πρέπει πρώτα να δούμε αν έχουμε λάβει κάποια τιμή , αυτό θα γίνει με την εντολή:
isset($_POST[‘btn-login’])
και εφόσον έχουμε περάσει αυτόν τον έλεγχο θα πρέπει να κρατήσουμε σε δύο μεταβλητές τις τιμές που έχει βάλει ο χρήστης στα πεδία με όνομα “username” και “pass” :

```
$uname = mysqli_real_escape_string($con,$_POST['username']);
$upass = mysqli_real_escape_string($con,$_POST['pass']);
```

Το τελευταίο κομμάτι είναι να συγκρίνουμε τις τιμές που πήραμε με τις τιμές που έχουμε αποθηκευμένες στην βάση μας, για να πάρουμε τις τιμές από την βάση θα χρησιμοποιήσουμε την εντολή :

```
$res = mysqli_query($con,"SELECT * FROM users WHERE username='" . $uname . "'");
$row = mysqli_fetch_array($res);
```

Και στην συνέχεια πρέπει να ελέγχουμε αν ο κωδικός από το όνομα που ψάξαμε είναι ο ίδιος με τον κωδικό που είχε βάλει ο χρήστης το πεδίο “pass” :

```
if ($row['password'] == $upass) {
      $_SESSION['user'] = $row['user_id'];
      header("Location: " . $row['site'] . "");
   } else {
      ?>
      <script>alert('Wrong');</script>
      <?php
   }
```

Εφόσον ο κωδικός είναι σωστός τότε θα μεταφέρουμε τον χρήστη στον χώρο που του έχουμε δώσει πρόσβαση αλλιώς θα του εμφανίσουμε ένα μήνυμα που θα τον ειδοποιεί ότι ο τα στοιχεία που έβαλε είναι λάθος και θα παραμείνει στην ίδια σελίδα.

4.4.6 Δημιουργία σελίδας ελέγχου κλιματιστικών (controlpanel_enpet.php)

Για να δημιουργήσουμε την σελίδα θα πρέπει πρώτα να προσθέσουμε τις βιβλιοθήκες για την επικοινωνία του σέρβερ μας με τον mqtt διακομιστή. Έχουμε την επιλογή να προσθέσουμε τις βιβλιοθήκες μας σαν εξωτερικούς συνδέσμους αν θέλουμε η αν προτιμάμε μπορούμε να τα κάνουμε αντιγραφή σε ένα javascript αρχείο και να το βάλουμε στον φάκελο του σέρβερ μας, ο δεύτερος τρόπος είναι καλύτερος καθώς στον πρώτο τρόπο αν ο σέρβερ που έχει το αρχείο που θέλουμε έχει κάποιου είδους προβλήματος ή σε περίπτωση που χαθεί το πακέτο που ζητήσαμε στην μεταφορά τότε η ιστοσελίδα μας δεν θα λειτουργήσει σωστά.

4.4.6.1 Mqtt βιβλιοθήκη

Αρχικά θα πρέπει να επισπευτούμε τον ιστότοπο :

```
https://cdnjs.cloudflare.com/ajax/libs/paho-mqtt/1.0.1/mqttws31.js
```

για να μπορέσουμε να πάρουμε την βιβλιοθήκη θα κάνουμε αντιγραφή ότι έχει στον ιστότοπο και να τα βάλουμε σε ένα δικό μας αρχείο το οποίο θα το ονομάσουμε mqttws31.js .
Με αυτή την βιβλιοθήκη μπορούμε να έχουμε τις εξής μεθόδους :

Method Summary	
	connect(connectOptions) Θα συνδέσει τον πελάτη που θα δηλώσουμε στο “connectOptions" με τον διακοσμητή θέλουμε.
	disconnect() Θα κάνει αποσύνδεση από το διακοσμητή που έχουμε κάνει την

	εγγραφή μας.
	publish(payload,topic,Qos) Θα κάνει δημοσίευση ένα μήνυμα που θα το δηλώσουμε στο payload, θα δημοσιευτεί στο θέμα που θα του δηλώσουμε στο topic και με το Qos που θέλουμε σαν τρίτη παράμετρο.
	subscribe(filter, subscribeOptions) Θα κάνει εγγραφή σε ένα θέμα που θα δηλώσουμε , περιμένει από τον διακομιστή κάθε μήνυμα που θα δημοσιευτεί στο θέμα αυτό.
	unsubscribe(filter, unsubscribeOptions) Κάνει διαγραφή από ένα θέμα και τα μηνύματα από το θέμα αυτό σταματάνε να του έρχονται.

Field Summary

	onConnectionLost Θα καλεστεί σε περίπτωση που η σύνδεση που έχουμε κάνει με τον διακομιστή μας χαθεί.
	onMessageArrived Θα καλεστεί σε περίπτωση που έρθει κάποιο μήνυμα στο θέμα που έχουμε κάνει εγγραφή
	onMessageDelivered Θα καλεστεί όταν το μήνυμα που στείλαμε θα παραδωθεί,

Στην συνέχεια θα δημιουργήσουμε ένα αρχείο με όνομα “message.js” και μέσα θα δημιουργήσουμε την σύνδεση με τον διακομιστή που θέλουμε , αρχικά θα πρέπει να φτιάξουμε έναν νέο client ως εξής:

```
var client = new Messaging.Client("ο διακομιστής μας", πόρτα , "έναν κωδικό σαν ID");
```

Στην συνέχεια θα φτιάξουμε την μέθοδο onConnectionLost για να πούμε τι θα κάνει ο σέρβερ μας σε περίπτωση που χαθεί η σύνδεση με τον διακομιστή.

```
client.onConnectionLost = function (responseObject) {
        window.location.replace(" index.php");
};
```

Θα διαλέξουμε να πηγαίνει τον χρήστη ξανά στην αρχική σελίδα , καθώς και να δοκιμάσει τώρα ο χρήστης να στείλει μηνύματα δεν σταλθούν.

Στο επόμενο κομμάτι θα δηλώσουμε τα options του χρήστη για την σύνδεση και τι θα συμβαίνει σε περίπτωση που η πετύχει η αποτύχει η σύνδεση με τον διακομιστή.

```
var options = {
   timeout: 3,

   onSuccess: function () {
   }
   ,
   //Gets Called if the connection could not be established
   onFailure: function (message) {
      window.location.replace (" index.php");
   }
};
```

Όπως και πιο πάνω σε περίπτωση που η σύνδεση μας δεν ήταν επιτυχείς θα μεταφέρουμε τον χρήστη ξανά στην αρχική σελίδα.

Μπορούμε επίσης να δηλώσουμε και τις επιλογές της publish μεθόδου για να μην χρειάζεται να τις συμπληρώνουμε όλες κάθε φορά θέλουμε να κάνουμε δημοσίευση ενός μηνύματος.
Θα πρέπει αρχικά να δημιουργήσουμε μια μεταβλητή message στο οποίο message θα βάλουμε το payload μας, στο message επίσης θα δηλώσουμε τις ιδιότητες που θα έχει το μήνυμα μας.

```
var publish = function (payload, topic, QoS) {
   var message = new Messaging.Message(payload);
   message.isRetained = false;
   message.destinationName = topic;
   message.qos = QoS;
   message.cleanSession = false;
   client.send(message);
}
```

Το τελευταίο κομμάτι που θα προσθέσουμε είναι τα callback function δηλαδή συναρτήσεις οι οποίες θα σου γυρίσουν απάντηση εκεί από εκεί καλέστηκαν , αυτό μας χρειάζεται επειδή το Arduino πριν μας απαντήσει θα πρέπει να πάρει κάποιες μετρήσεις οπότε θα εμφανίσουμε την απάντηση εφόσον έχουμε πάρει τις μετρήσεις μας.
Θα δημιουργήσουμε μια callback συνάρτηση με όνομα connectRoom , όμως για να παίρνουμε τιμές από τον διακομιστή μας θα πρέπει με την επιλογή νέου δωματίου να γίνεται και νέα εγγραφή στον δωμάτιο που επιλέχτηκε και παράλληλα θα κάνουμε διαγραφή από το προηγούμενο χώρο, αυτή θα καλείτε όταν ο χρήστης επιλέξει ένα από τα δωμάτια στον χώρο που διάλεξε , μέσα στην συνάρτηση θα πρέπει να χειριζόμαστε τα μηνύματα που μας έρχονται ως εξής :

```
function connectRoom(site_publish, callback) {
   client.onMessageArrived = function (message) {
      if(!isNaN(message.payloadString)){
                              if (message.destinationName == site_publish + "/response/dht_hum") {
                                       document.getElementById("dht_hum").innerHTML                    =
message.payloadString + " %";
                              }
                              if (message.destinationName == site_publish + "/response/dht_temp") {
```

```
                                document.getElementById("dht_temp").innerHTML =
message.payloadString + "°C";
                            }
      }
      if (message.destinationName == site_publish + "/response/watt") {
            if (parseInt(message.payloadString) > 110) {
               onoff = 1;
               openclose = 1;
            }
            document.getElementById("watt").innerHTML = message.payloadString;
      }
      if (message.destinationName == site_publish + '/response') {
         if (message.payloadString == "connected") {
            isConnected = 1;
            callback(message.destinationName);
         } else {
            callback(message.payloadString);
         }
      }
   };
}
```

Με την εντολή message.payloadString μεταφέρει το μήνυμα που μας ήρθε και η εντολή message.destinationName είναι το μονοπάτι από το οποίο μας ήρθε το μήνυμα μας , επειδή κάνουμε εγγραφή σε ένα μονοπάτι χωρίς τέλος πρέπει να ελέγχουμε από ποιο μονοπάτι μας ήρθε για να μπορέσουμε να το μεταφέρουμε και στο ανάλογο μέρος στην ιστοσελίδα μας, οι τιμές “onoff” ,“openclose” και “isConnected” σημαίνουν :

- onoff : εδώ θα βάζουμε αν το κλιματιστικό είναι ανοιχτό η όχι , έχει αρχική τιμή 3 και θα αλλάζει μόνο αν πάρει σαν απάντηση ότι τα watt που καίει το κλιματιστικό είναι πάνω από 110 , τότε θα γίνεται 1 που θα σημαίνει ότι είναι ανοιχτό το κλιματιστικό μας .
- openclose: εδώ θα κρατάμε την τελευταία τιμή για το κουμπί που έχουμε να ελέγχει το άνοιγμα και το κλείσιμο του κλιματιστικού , επειδή είναι ένα μόνο κουμπί θα πρέπει να βλέπουμε τι τιμή είχε τελευταία φορά έτσι ώστε όταν ξαναπατηθεί να στέλνει την άλλη τιμή του.
- isConnected : εδώ θα κρατάμε αν είμαστε συνδεμένοι με το Arduino/esp8266 μας ,θα γίνεται 1 μόνο σε περίπτωση που το esp8266 μας απαντήσει “connected” , εφόσον η μεταβλητή αυτή γίνει 1 τότε μπορούμε να πατήσουμε και τα κουμπιά για τις λειτουργίες του κλιματιστικού μας , αλλιώς αυτά τα κουμπιά θα είναι απενεργοποιημένα .Για να καταλαβαίνει και ο χρήστης ότι είμαστε συνδεμένοι θα κάνουμε και τον χώρο στον οποίο διάλεξε στον canva πράσινο χρώμα.

4.4.6.2 Controlpanel_enpet.php

Για την δημιουργία της σελίδας ελέγχου του χώρου αρχικά θα πρέπει να δημιουργήσουμε ένα session με την εντολή session_start() , σε αυτό το session θα κρατάμε τον χρήστη που έχει κάνει σύνδεση κάθε φορά και εφόσον δεν έχει καταφέρει ο χρήστης να κάνει σύνδεση θα τον μεταφέρουμε στην σελίδα για την σύνδεση , αυτό το κομμάτι είναι για θέμα ασφαλείας καθώς κάποιος μπορεί να δει το μονοπάτι της σελίδας έλεγχου και να το επισκεφτεί χωρίς να

έχει πρόσβαση κανονικά και ο κώδικας θα φαίνεται ως εξής :

```
session_start();
include_once 'dbconnect.php';
if (!isset($_SESSION['user'])) {
   header("Location: index.php");
}
```

επόμενο βήμα είναι δημιουργήσουμε τις επιλογές του κλιματιστικού που θα έχει ο χρήστης , αυτές οι επιλογές θα αποτελούνται από :

- **4 Modes** : θα αποτελούνται από Auto, Heat, Cool και Dry Mode με τα οποία θα ελέγχουμε την λειτουργία που θα έχει το κλιματιστικό.
- **4 Fan speed** : θα αποτελούνται από τα Auto , Low , Medium , High speed και θα ελέγχουν την ταχύτητα του ανεμιστήρα.
- **2 Swings** : τα Auto και Manual θα ελέγχουν την διεύθυνση του αέρα .
- **Temperature set** : Εδώ θα καθορίζουμε την θερμοκρασία που θέλουμε να έχει το κλιματιστικό

παράλληλα θα πρέπει δείχνουμε την τιμή που επέλεξε ο χρήστης για την θερμοκρασία σε ένα label , θα δείχνουμε την κατανάλωση , την θερμοκρασία και την υγρασία που έχει το κλιματιστικό .
Τα κουμπιά για τα modes θα έχουν την εξής μορφή με μικρές διαφορές στο id τους καθώς η κάθε λειτουργία θα πρέπει να στέλνει διαφορετικό μήνυμα :

```
<div class="col-sm-3">
      <button class="btn btn-block  btn-primary" id="MD0"  width = "50" height = "50" value = "button" style= "color:white" />
        Auto Mode</button>
</div>
```

Τις τιμές για την θερμοκρασία και την υγρασία θα τις εμφανίζουμε ως εξής :

```
<div class='card temp'>
                    <div class='inner'>
                       <div class='icon'></div>
                       <div class='title'>
                          <div class='text' >TEMPERATURE | HUMIDITY</div>
                       </div>
                       <div class='number' id="dht_temp">Temperature</div>
                       <div class='number' id="dht_hum">Humidity</div>
                    </div>
</div>
```

Και τέλος τις τιμές για την κατανάλωση θα τις περάσουμε ως εξής :

```
<div class='card energy'>
                    <div class='inner'>
                       <div class='icon'></div>
                       <div class='title'>
                          <div class='text'>ENERGY</div>
                       </div>
                       <div class='number2' id='watt'>Watt</div>
                       <div class='measure'>Watt / HOUR</div>
                    </div>
```

```
</div>
```

Εφόσον έχουμε δημιουργήσει το γραφικό περιβάλλον της ιστοσελίδας μας θα πρέπει να τρέξουμε μερικά javascript για τις βασικές λειτουργίες μας. Στο τέλος του body μας θα πρέπει να προσθέσουμε το <script > για να μπορέσουμε μέσα να γράψουμε javascript . Αρχικά θα πρέπει να δημιουργήσουμε όσες μεταβλητές χρειάζονται τα μονοπάτια για κάνουμε εγγραφή και δημοσίευση , αυτές τις τιμές θα τις πάρουμε από την βάση μας την οποία την κάναμε include στην αρχή της σελίδας μας και το query που θα μας φέρει τις τιμές αυτές είναι το εξής :

```
$res = mysqli_query($con, "SELECT * FROM users WHERE user_id=" . $_SESSION['user']);
$userRow = mysqli_fetch_array($res);
```

Και η δήλωση τιμών στις μεταβλητές θα φαίνεται ως εξής :

```
var site_publish = "<?php echo $userRow['publish']; ?>";
```

στην συνέχεια θα πρέπει να δηλώσουμε τις μεταβλητές που θα κρατάνε αν το κλιματιστικό είναι ανοιχτό η κλειστό και αν ο χρήστης έχει συνδεθεί με την συσκευή μας η όχι , και οι δύο μεταβλητές θα έχουν αρχική τιμή 0. Έπειτα δημιουργούμε τον καμβά μας και μέσα στον καμβά θα βάλουμε την εικόνα που έχουμε δημιουργήσει για τον χώρο που έχουμε επιλέξει ,το μονοπάτι για την εικόνα το δηλώνουμε εμείς και θα είναι διαφορετικό για κάθε μέρος.

Με την βοήθεια του JQuery θα προσθέσουμε την εντολή $(function () { } η οποία θα τρέξει μόλις η ιστοσελίδα μας έχει φορτώσει , μέσα στην συνάρτηση θα βάλουμε τον mouseListener που χρειαζόμαστε , ο listener αυτός θα είναι υπεύθυνος για να καταλαβαίνει που πατήσαμε πάνω στον καμβά και την τιμή αυτή θα την παίρνει από τις εντολές :

```
var pos = getMousePos(canvas, e);
posx = pos.x;
posy = pos.y;
```

στην συνέχεια ελέγχουμε αν οι τιμές για τα x,y είναι μέσα σε κάποιες συγκεκριμένες τιμές για να καταλάβουμε σε ποιον χώρο έχει πατήσει ο χρήστης , εφόσον περάσει τον έλεγχο μας και καταλάβαμε που πάτησε τότε αρχικά θα κάνουμε την τιμή isConnected ίση με 0 και θα κάνουμε subscribe στον broker για τον χώρο αυτόν με την εντολή `client.subscribe(site_publish + '/response/#');`

Πριν συνεχίσουμε για τον νέο χώρο θα πρέπει να διαγράψουμε από τον καμβά οτιδήποτε είχαμε ζωγραφίσει πριν πάνω , ο μόνος τρόπος για να γίνει αυτό είναι με την διαγραφή όλου το καμβά με την εντολή context.clearRect στην συνέχεια θα πρέπει να του προσθέσουμε ξανά την εικόνα του χώρου με την εντολή context.drawImage , έπειτα με την εντολή subscribe πρέπει να κάνουμε δημοσίευση το μήνυμα connect στον νέο χώρο και μέσω της callback συνάρτησης θα περιμένουμε να μας απαντήσει η συσκευή μας , αν το μήνυμα που θα μας έρθει είναι connected τότε με την συνάρτηση paintCanv(1,2,3,4,5) θα ζωγραφίσουμε τον χώρο πράσινο.

Η εντολή paintCanv περιμένει 6 μεταβλητές σαν εισόδους :

1. Το χρώμα το οποίο θέλουμε να ζωγραφίσουμε τον χώρο.
2. Την αρχή από το άξονα X.
3. Το τέλος του άξονα X.
4. Αρχή άξονα Y.
5. Τέλος άξονα Y.

και μέσα στην paintCanv εφόσον είμαστε connected θα κάνουμε τα κουμπιά enabled και θα ζωγραφίσουμε τον καμβά με τρείς εντολές :

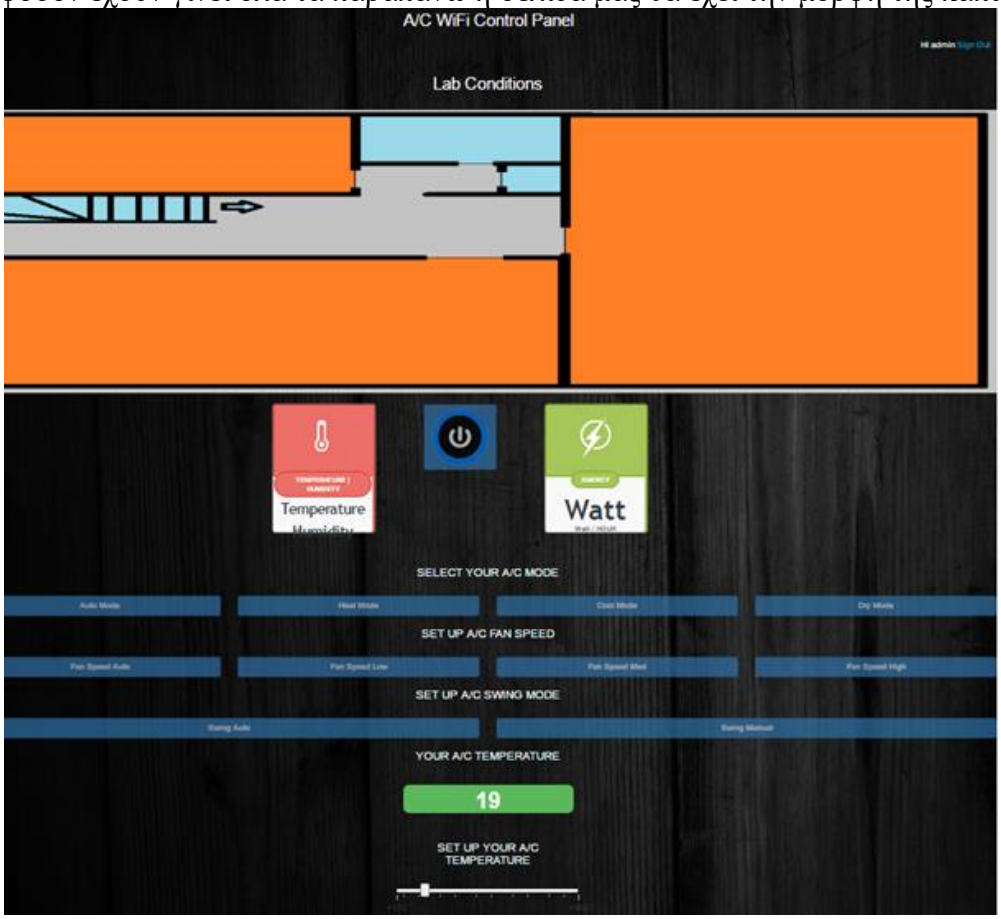
```
$("button").prop("disabled", false);
context.drawImage(imagz, 0, 0, canvas.width, canvas.height);
```

```
context.fillStyle = color;
context.fillRect(x_start, y_start, x_end, y_end);
```

το τελευταίο κομμάτι που μας μένει είναι μια μικρή ειδοποίηση πάνω δεξιά που τα δείχνει στον χρήστη ότι η εντολή του εκτελέστηκε στο κλιματιστικό , θα ειδοποιεί μόνο για τις επιλογές on/off και τα modes που μπορεί να διαλέξει. Ο τρόπος που θα παίρνουμε την απάντηση θα είναι πάλι μέσω callback συναρτήσεων δηλαδή ο χρήστης αρχικά θα στέλνει την τιμή και εφόσον γίνουν οι απαραίτητες μετρήσεις από την συσκευή μας και πάρουμε τις απαντήσεις μας θα λέμε στον χρήστη “success”. Όλα τα παραπάνω θα γίνουν με την συνάρτηση connectRoom και θα καλείται με κάθε κουμπί που θα πατάμε , η μορφή της θα είναι ως εξής :

```
connectRoom(site_publish, function (answer) {
                    if (answer == "done" && lastID == "AC") {
                        $.notify("Success", "success");
                        openclose = !openclose;
                    } else if (answer == "done") {
                        $.notify("Success", "success");

                    }
                });
```

Τέλος εφόσον έχουν γίνει όλα τα παραπάνω η σελίδα μας θα έχει την μορφή της κάπως έτσι :



Εικόνα 20 - Controlpanel_enpet

4.5 Arduino

Σε αυτό το κεφάλαιο θα αναλύσουμε τις συνδέσεις των αισθητήρων και των συσκευών ελέγχου με το Arduino.

4.5.1 Συνδεσμολογία Arduino με τις συσκευές ελέγχου

Η πλακέτα που χρησιμοποιούμε για την υλοποίηση της εφαρμογής συνδέεται με τις συσκευές που έχουμε αναφέρει στο κεφάλαιο 3 και πιο συγκεκριμένα με τα Esp8266-01 Wi-Fi Module , με τον αισθητήρα ρεύματος YHDC SCT-013-000 Current Transformer, με τον αισθητήρα θερμοκρασίας και υγρασίας DHT-11, με το τρανζίστορ NPN 2N2222A, με ένα IR emitter, με ένα IR receiver καθώς επίσης χρησιμοποιήσαμε ένας ρυθμιστή τάσης και μερικές αντιστάσεις και πυκνωτές.

Για να καταφέρουν να επικοινωνήσουν οι συσκευές και οι σένσορες που χρησιμοποιούμε για το project μας θα πρέπει να συμπεριλάβουμε στο προγραμματιστικό περιβάλλον του Arduino, κάποιες βιβλιοθήκες. Αυτές οι βιβλιοθήκες πολλές φορές αναλαμβάνουν να κάνουν κάποιες εργασίες για τις μετρήσεις οι τις λειτουργίες της συσκευής που θέλουμε να χρησιμοποιήσουμε. Επίσης οι βιβλιοθήκες αυτές στις περισσότερες των περιπτώσεων, συνοδεύονται από κάποια παραδείγματα, τα οποία δίνουν την ευκαιρία στους χρήστες της συσκευής να έχουν μια πρώτη ευκαιρία πειραματισμού με την συσκευή και εμφάνισης κάποιων αποτελεσμάτων.

4.5.2 Σύνδεση IR Decoder με το Arduino

Σε πρώτη φάση αυτό που πρέπει να κάνουμε είναι να βρούμε ένα τρόπο να αναλύσουμε το σήμα που στέλνει ένα τηλεχειριστήριο ενός κλιματιστικού, και ποιο συγκεκριμένα στην περίπτωση μας το κλιματιστικό που χρησιμοποιούμε είναι ένα κλιματιστικό της Midea. Από τα εργαλεία που έχουμε αναφέρει στο κεφάλαιο 3, αυτό που χρησιμοποιούμε είναι το IR receiver.

Το IR receiver έχει πολύ απλή συνδεσμολογία. Αποτελείται από 3 pins, όμως υπάρχουν κάποιες διαφορές ανάλογα το μοντέλο. Ενδεικτικά αναφέρονται κάποια χαρακτηριστικά για κάθε μοντέλο στον παρακάτω πίνακα :

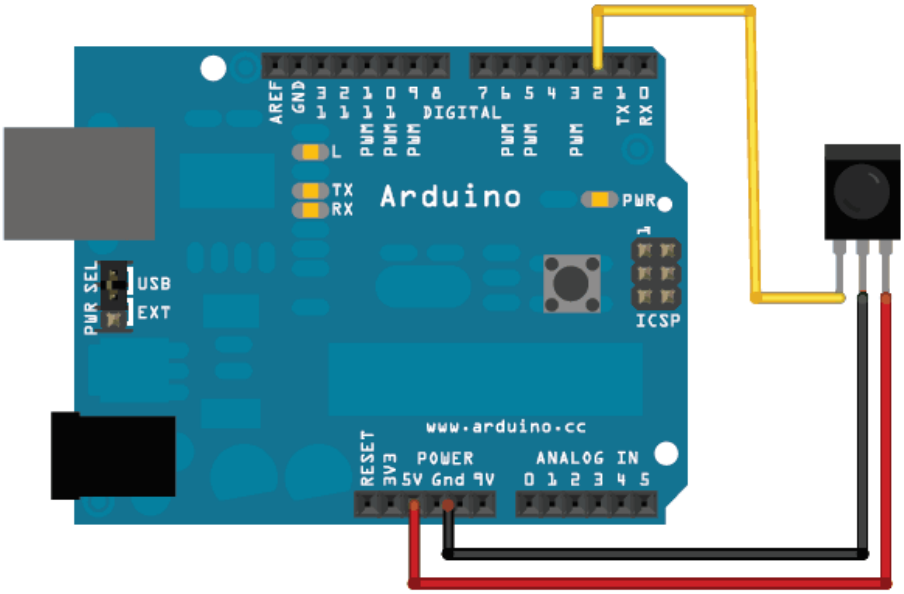
PARTS TABLE					
AGC		LEGACY, FOR LONG BURST REMOTE CONTROLS (AGC2)		RECOMMENDED FOR LONG BURST CODES (AGC4)	
Carrier frequency	30 kHz	TSOP4830	TSOP2230	TSOP4430	TSOP2430
	33 kHz	TSOP4833	TSOP2233	TSOP4433	TSOP2433
	36 kHz	TSOP4836	TSOP2236	TSOP4436 [1][2][3]	TSOP2436 [1][2][3]
	38 kHz	TSOP4838	TSOP2238	TSOP4438 [4][5][6]	TSOP2438 [4][5][6]
	40 kHz	TSOP4840	TSOP2240	TSOP4440	TSOP2440
	56 kHz	TSOP4856	TSOP2256	TSOP4456 [6][7]	TSOP2456 [6][7]
Package		Mold			
Pinning		1 = OUT, 2 = GND, 3 = V_S	1 = OUT, 2 = V_S, 3 = GND	1 = OUT, 2 = GND, 3 = V_S	1 = OUT, 2 = V_S, 3 = GND
Dimensions (mm)		6.0 W x 6.95 H x 5.6 D			
Mounting		Leaded			
Application		Remote control			
Best remote control code		[1] RC-5 [2] RC-6 [3] Panasonic [4] NEC [5] Sharp [6] r-step [7] Thomson RCA			

Εικόνα 21 - IR receiver Datasheet

Το μοντέλο του IR receiver που χρησιμοποιούμε για την συγκεκριμένη εργασία είναι το TSOP4840 το οποίο έχει φέρουσα συχνότητα 40 kHz και τα pins του συνδέονται όπως φαίνονται και στον πίνακα :

- αριστερό pin είναι η έξοδος
- μεσαίο pin είναι η γείωση
- δεξί pin είναι η τροφοδοσία.

Ενδεικτικό παράδειγμα της συνδεσμολογίας είναι η παρακάτω εικόνα, όπου παρουσιάζεται η σύνδεση ενός IR receiver με μία συσκευή Arduino.



Εικόνα 22 - IR receiver & Arduino

Για την εκπλήρωση του συγκεκριμένου project χρησιμοποιήθηκε μία βιβλιοθήκη ανοιχτού κώδικα, η οποία βρίσκεται στο GitHub, το οποίο είναι μια πλατφόρμα ανοιχτού κώδικα, η οποία χρησιμοποιείται από διάφορους developers ανά τον κόσμο, για τον διαμοιρασμό προγραμματιστικών εφαρμογών.

Σε αυτό το project που χρησιμοποιήσαμε από το GitHub γίνεται ανάλυση του IR σήματος με τον εξής τρόπο :

- Αρχικά επιλέγουμε μία από της τέσσερις επιλογές και ελέγχουμε εάν το πρωτόκολλο που επιλέξαμε είναι το σωστό
- Εν συνεχεία δοκιμάζουμε να στείλουμε ένα σήμα από το τηλεχειριστήριο και να δούμε τα αποτελέσματα στην σειριακή μας οθόνη, όπως φαίνεται στην παρακάτω εικόνα.

```
COM3 (Arduino/Genuino Uno)

Select model to decode (this affects the IR signal timings detection):
* '1' for Panasonic DKE>, Mitsubishi Electric, Fujitsu etc. codes
* '2' for Panasonic CKP, Midea etc. codes
* '3' for Mitsubishi Heavy etc. codes
* '4' for Hyyndai etc. codes
* '9' for entering the bit sequence on the serial monitor (instead of the IR receiver)

Enter choice: 4

Ready to decode IR for choice '4'



Number of symbols: 102
Symbols:
Hh10110010010011011001111101100000101011000101001 1hHh10110010010011011001111101100000101011000101001 1
Bytes:
4D,B2,F9,06,35,CA,4D,B2,F9,06,35,CA
Timings (in us):
PAUSE SPACE:  0
HEADER MARK:  4540
HEADER SPACE: 4680
BIT MARK:     500
ZERO SPACE:   500
ONE SPACE:    1614
Decoding known protocols...
Looks like a Carrier protocol #2
POWER: ON
FAN: 1
Temperature: 29
Checksum matches
```

Εικόνα 23 - IR decoder message

Παρατηρούμε ότι το συγκεκριμένο τηλεχειριστήριο χρησιμοποιεί το πρωτόκολλο carrier, και από την ανάλυση του σήματος παίρνουμε κάποιες πληροφορίες για την λειτουργία του κλιματιστικού.

- Εν συνεχεία θα τροποποιήσουμε την νέα βιβλιοθήκη που έχουμε πάρει από το GitHub, ώστε να προσαρμόσουμε τις λειτουργίες του κλιματιστικού μας.
- Το αμέσως επόμενο βήμα είναι να πάμε στην βιβλιοθήκη που έχουμε και να τοποθετήσουμε τις τιμές που μας επέστρεψε το IR decoder, ώστε να μπορέσουμε να κάνουμε μια αναπαραγωγή των σημάτων που στέλνει το led του τηλεχειριστηρίου μας.

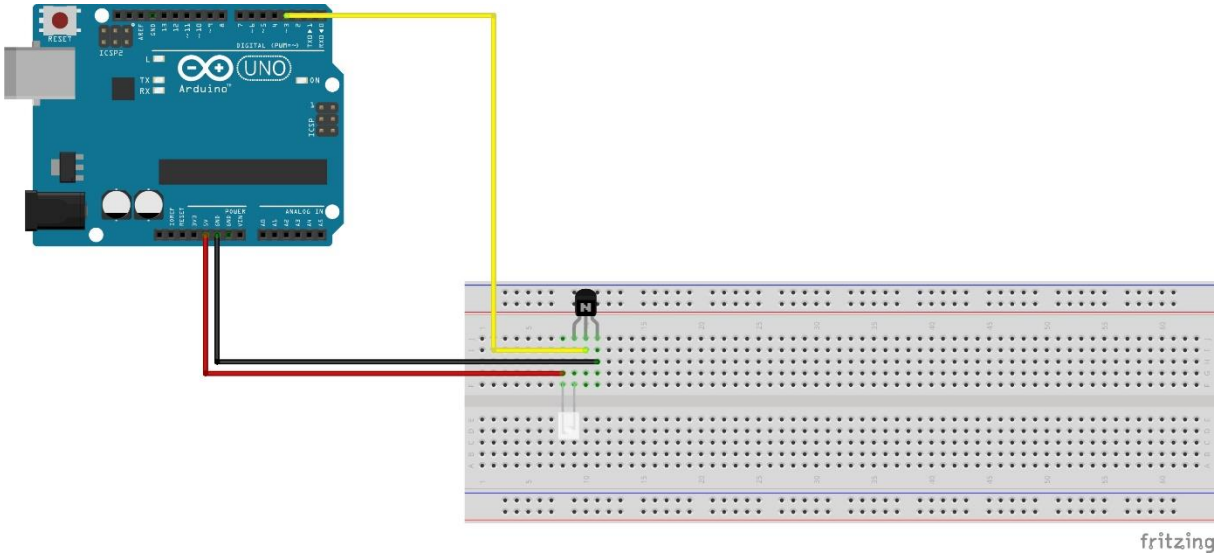
Όταν ολοκληρώσουμε αυτή την διαδικασία είμαστε σε θέση να πάμε στο επόμενο βήμα που είναι η αποστολή των εντολών του τηλεχειριστηρίου της κλιματιστικής μας μονάδας μέσω του IR emitter, κάτι το οποίο θα δούμε στην επόμενη παράγραφο αναλυτικά.

4.5.3 Σύνδεση IR emitter και αποστολή εντολών

Εφόσον έχουμε αναλύσει το σήμα που παίρνει το κλιματιστικό μας στο προηγούμενο βήμα, σειρά έχει τώρα η αναπαραγωγή του σήματος μέσω ενός led υπέρυθρων.

Η σύνδεση ενός IR emitter με την συσκευή του Arduino είναι πολύ απλή καθώς αποτελείται από 2 pins, από τα οποία το ένα είναι μεγαλύτερο και είναι αυτό που συνδέεται στα 5 volt και

το αριστερό είναι αυτό που πάει στην γείωση. Όμως στην περίπτωση μας επειδή χρησιμοποιούμε τρανζίστορ η σύνδεση διαφέρει λίγο, όπως φαίνεται και στην παρακάτω εικόνα.



Εικόνα 24 -IR emitter & Transistor

Δημιουργούμε μία μεταβλητή irSender τύπου **IRSenderPWM** και συνδέουμε τη βάση του τρανζίστορ στο pin 3 του Arduino, διότι θέλουμε το σήμα μας να είναι PWM. Έτσι, όταν το τρανζίστορ είναι σε λειτουργία κλειστού διακόπτη, το led στέλνει το σήμα μας με συχνότητα ανάλογη της λειτουργίας που θέλουμε να εκτελέσουμε.

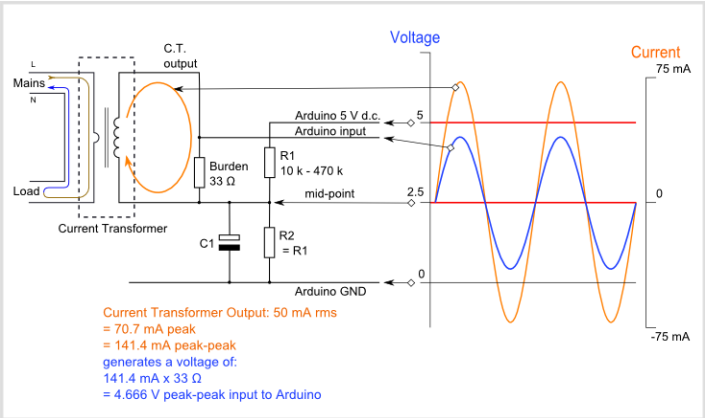
4.5.4 Σύνδεση SCT-013 με το Arduino

Ένα ακόμη εργαλείο που χρησιμοποιήσαμε για την υλοποίηση της πτυχιακής εργασίας, είναι ένας αισθητήρας εντάσεως ρεύματος.

Το παρακάτω σχήμα της εικόνας φαίνεται πως συνδέονται οι ακροδέκτες του αισθητήρα SCT-013 καθώς και ο ακροδέκτης τάσης, γείωσης και ο αναλογικός του Arduino για την μέτρηση της έντασης του ρεύματος.

Όπως αναφέρθηκε παραπάνω, η αντίσταση φορτίου είναι απαραίτητη για να κλείσει το δευτερεύον κύκλωμα του αισθητήρα SCT - 013. Ακόμα, για να συνδεθεί ο μετασχηματιστής εντάσεως ρεύματος στο Arduino, θα πρέπει το σήμα εξόδου του να είναι σύμφωνο με τις απαιτήσεις εισόδου των αναλογικών ακροδεκτών του Arduino. Συνεπώς, η είσοδος θα πρέπει να είναι μια θετική τάση μεταξύ 0 Volt και της τάσης αναφοράς ADC. Αυτό μπορεί να επιτευχθεί χρησιμοποιώντας το ακόλουθο κύκλωμα το οποίο αποτελείται από δύο κύρια μέρη:

- Τον αισθητήρα εντάσεως (C.T) και την αντίσταση φορτίου (Burden)
- Το διαιρέτη τάσης (R1 και R2)



Εικόνα 25 - Calculating Burden Resistor

Η ένταση του ρεύματος που μπορεί να αισθανθεί είναι από 0 έως 100 A. Ο συγκεκριμένος αισθητήρας υπερκαλύπτει της απαιτήσεις μίας οικίας ακόμη και αν αυτή τροφοδοτείται με ενισχυμένο μονοφασικό.

Για την συγκεκριμένη συσκευή που θέλουμε να ελέγξουμε η μέγιστη κατανάλωση ισχύος σύμφωνα με τα χαρακτηριστικά της συσκευής είναι 1500W. Επομένως, το μέγιστο ρεύμα που θα μπορούσε να περάσει μέσα από τον αισθητήρα είναι ίσο με :

$$i = \frac{p}{v} = \frac{1800}{230} = 7.82\ A$$

Στις μετρήσεις που έγιναν, η ένταση του ρεύματος δεν ξεπέρασε αυτή την τιμή και έτσι θεωρήθηκε η μέγιστη δυνατή και σύμφωνα με αυτή έγιναν οι υπολογισμοί. Όπως είναι γνωστό, στο εναλλασσόμενο ρεύμα, η τιμή που λαμβάνεται από όργανα μέτρησης είναι η ενεργός τιμή(rms –root mean square) του αντίστοιχου μεγέθους. Συνεπώς, η τιμή της κορυφής (peak) του πρωτεύοντος ρεύματος είναι ίση με :

$$i_{peak-primary} = i_{rms} * \sqrt{2} = 7.8 * 1.4142 = 11.03\ A$$

Αυτό είναι το μέγιστο ρεύμα που μπορεί να διαρρέει το πρωτεύον πηνίο του μετασχηματιστή. Για να βρεθεί το μέγιστο ρεύμα που διαρρέει το δευτερεύον πηνίο θα πρέπει να ληφθούν υπόψη τα χαρακτηριστικά του μετασχηματιστή. Συγκεκριμένα, ο λόγος σπειρών είναι ίσος :

$$n_{mumber-of-turns} = \frac{100A}{0.05A} = 2000$$

Οπότε το μέγιστο ρεύμα στο δευτερεύον πηνίο προκύπτει ίσο με :

$$i_{peak-secondary} = \frac{i_{peak-primary}}{n_{number-of-turns}} = \frac{11.03}{2000} = 0.0055\ A$$

Για την μεγιστοποίηση του φάσματος της μέτρησης, η τάση στην αντίσταση φορτίου όταν διέρχεται το ρεύμα κορυφής, θα πρέπει να είναι ίση με την αναλογική τάση αναφοράς του Arduino(AREF) διαιρεμένη με το 2. Για την τάση τροφοδοσίας 5 Volt θα είναι :

$$\frac{AREF}{2} = \frac{5}{2} = 2.5\ V$$

Τελικά, η ιδανική τιμή της αντίστασης φορτίου προκύπτει ίση με :

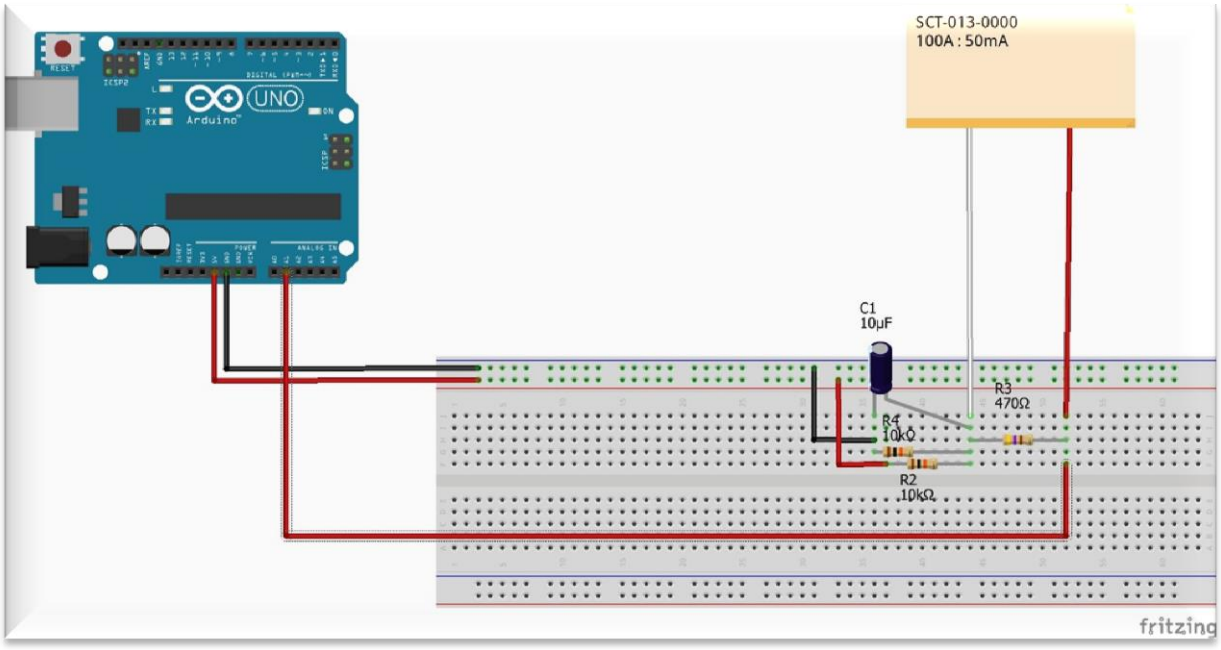
$$R_{ideal-burden} = \frac{\frac{AREF}{2}}{i_{peak-secondary}} = \frac{2.5}{0.0055} = 454\ \Omega$$

Η τιμή των 454 Ω δεν είναι μια κοινή τιμή αντίστασης στο εμπόριο. Υπάρχουν όμως παραπλήσιες αντιστάσεις στα 430 και στα 470 Ω. Επιλέχθηκε η τιμή των 470 Ω για την αντίσταση φορτίου.

Αφού χρησιμοποιήσαμε τους τύπους που δίνονται από την εταιρία που παράγει τον σένσορα για τον υπολογισμό του Burden Resistor, είμαστε έτοιμοι να ολοκληρώσουμε το κύκλωμα από το οποίο θα παίρνουμε τις μετρήσεις μας.

Όπως έχει προκύψει από την ανάλυση των τύπων θα χρειαστούμε έναν πυκνωτή 10μF, 2 αντιστάσεις των 10 kΩ και μία ακόμη αντίσταση των 470 Ω.

Στην παρακάτω εικόνα φαίνεται η συνδεσμολογία του αισθητήρα μας με το Arduino.



Εικόνα 26 - Arduino & SCT-013

Εφόσον έχουμε τελειώσει με την συνδεσμολογία της συσκευής μας μπορούμε να κάνουμε μια δοκιμή και να πάρουμε κάποιες μετρήσεις στην σειριακή οθόνη του Arduino από τον κώδικα που δίνεται παρακάτω.

```
// EmonLibrary examples openenergymonitor.org, Licence GNU GPL V3

#include "EmonLib.h"                   // Include Emon Library
EnergyMonitor emon1;                   // Create an instance

void setup()
{
  Serial.begin(9600);

  emon1.current(1, 111.1);             // Current: input pin, calibration.
}

void loop()
{
  double Irms = emon1.calcIrms(1480);  // Calculate Irms only

  Serial.print(Irms*230.0);            // Apparent power
  Serial.print(" ");
  Serial.println(Irms);                // Irms
}
```

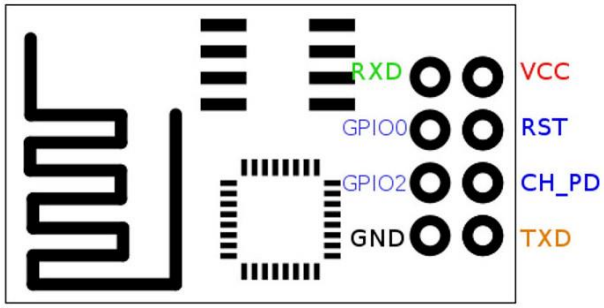
4.5.5 Σύνδεση Esp8266 με το Arduino

Όλες αυτές τις λειτουργίες και τις μετρήσεις που παίρνουμε από τους αισθητήρες θα πρέπει τώρα να τις μεταφέρουμε στον MQTT Broker, ο οποίος είναι υπεύθυνος για να κρατήσει τις

τιμές σε ένα topic και να ενημερώνει τα πεδία αυτά μεσώ των publish/subscribe. Για να το επιτύχουμε αυτό θα πρέπει οι τιμές των αισθητήρων και οι εντολές που θέλουμε να εκτελέσουμε μέσω του Arduino να ενημερώνονται μέσω ενός Wi-Fi module και στην συγκεκριμένη εργασία χρησιμοποιήθηκε ένα Esp8266-01.

Το Esp8266 χρειάζεται τροφοδοσία 3.3V, αλλά παρατηρήθηκε δυσλειτουργία της συσκευής όταν αυτή τροφοδοτούταν στο Arduino μαζί με όλες τις υπόλοιπες συσκευές και αισθητήρες. Γι αυτό τον λόγο προτιμήθηκε η χρήση ενός Voltage Regulator ο οποίος μα δίνει την δυνατότητα να σταθεροποιήσουμε την τάση στην είσοδο του Esp για μεγαλύτερη αξιοπιστία στην λειτουργία του.

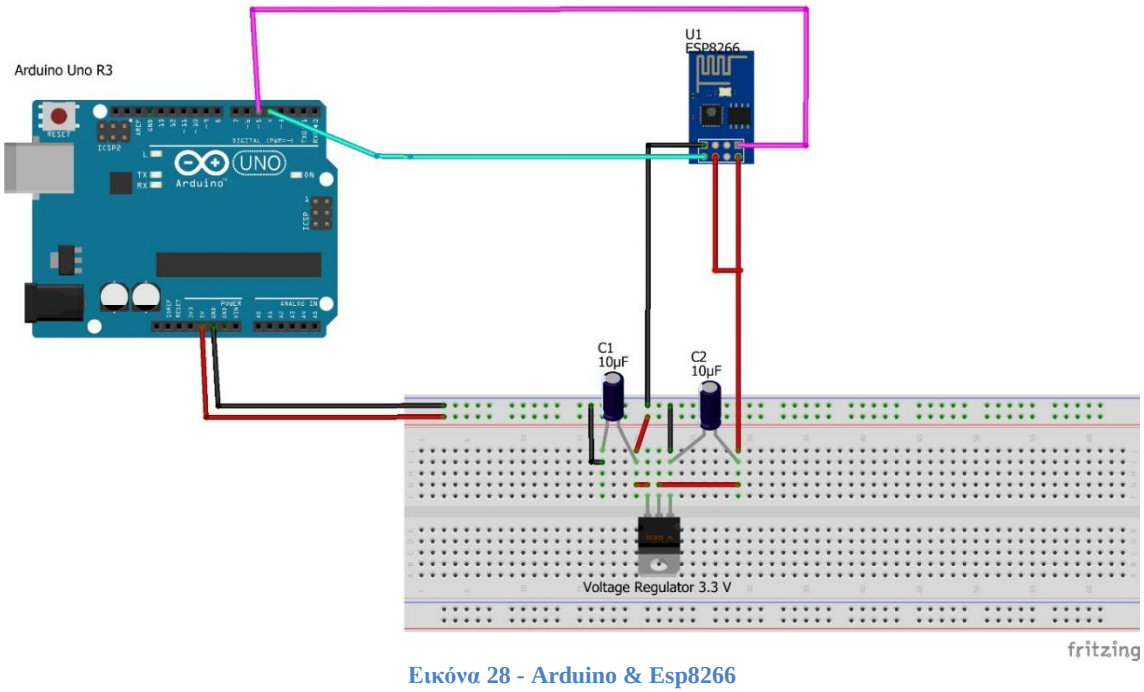
Στην παρακάτω εικόνα φαίνεται η λειτουργία του κάθε pin της συσκευής.



Εικόνα 27 - Esp8266 PinOut

Τα pins που θα χρησιμοποιήσουμε είναι το VCC και το CH_PD που θα τα τροφοδοτήσουμε με τάση 3.3V που θα πάρουμε από τον Voltage Regulator, το GND το οποίο θα συνδέσουμε στην γείωση, τα TX & RX τα οποία θα είναι υπεύθυνα για την ανταλλαγή μηνυμάτων και το GPI02 το οποίο θα διαβάζει τις τιμές του αισθητήρα DHT11.

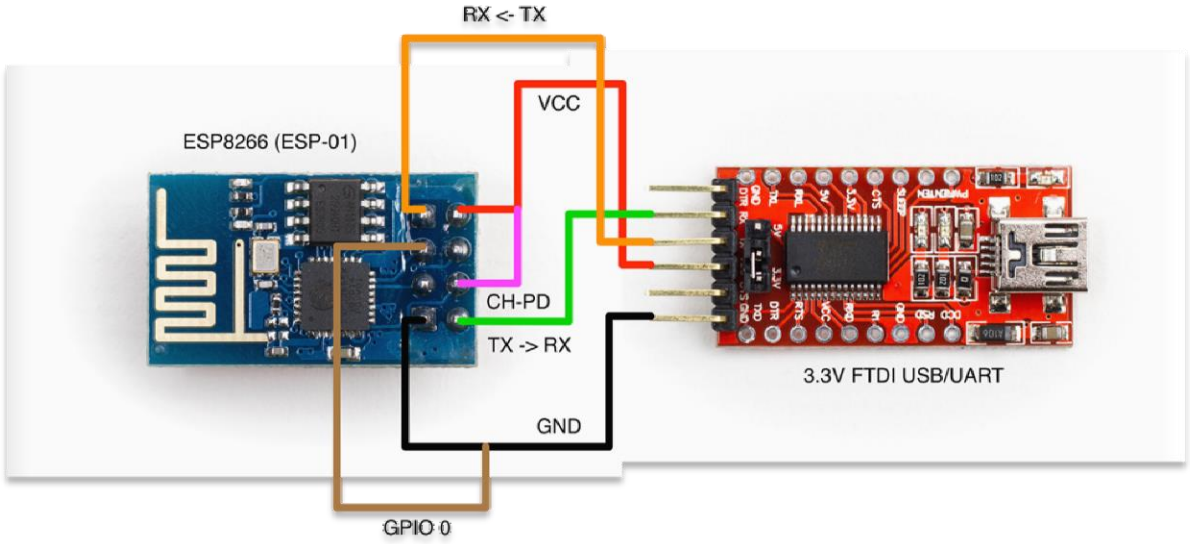
Στην παρακάτω εικόνα φαίνεται η συνδεσμολογία της συσκευής με το Arduino.



Εικόνα 28 - Arduino & Esp8266

Για να μπορέσουμε να εκμεταλλευτούμε όλες αυτές τις λειτουργίες που μας προσφέρουν οι αισθητήρες και οι συσκευές που έχουμε συνδέσει στην συσκευή μας θα πρέπει να έχουμε ανταλλαγή μηνυμάτων με τον Mqtt Broker μέσω της συσκευής Esp. Αυτό επιτυγχάνεται μέσω των pins Rx και Tx.

Παρατηρήθηκε κατά την διάρκεια της εργασίας μας, ότι μερικά Esp Modules ήθελαν update στο firmware για να καταφέρουμε να έχουμε σταθερή επικοινωνία. Για το επιτύχουμε αυτό χρειάστηκε να προγραμματίσουμε το Esp μέσω ενός FTDI programmer. Στην παρακάτω εικόνα φαίνεται η συνδεσμολογία ενός FTDI 3.3V USB/UART με ένα Esp8266-01.



Εικόνα 29 - FTDI & Esp8266-01

Για να καταφέρουμε να κάνουμε update το firmware του Esp Module χρησιμοποιήσαμε το NodeMCU Firmware Updater.

Για να προγραμματίσουμε το Esp Module, χρησιμοποιούμε τον FTDI programmer με την συνδεσμολογία που φαίνεται στην Εικόνα 28. Αρχικά θα πρέπει να ενσωματώσουμε στο Arduino IDE την βιβλιοθήκη της Adafruit και να δηλώσουμε κάποιες βασικές παραμέτρους ενεργοποίησης της.

```
#include <ESP8266WiFi.h>
#include <DHT.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

#define WLAN_SSID  "Enpet_epp"
#define WLAN_PASS  "smartlab2012"

#define arduino_pubsub "μονοπάτι όπου θα γίνεται η εγγραφή στον broker μας"
#define arduino_response "μονοπάτι όπου δημοσιεύουμε το μήνυμα μας στον broker "
#define arduino_temp "μονοπάτι όπου δημοσιεύουμε την τιμή της θερμοκρασίας"
#define arduino_hum "μονοπάτι όπου δημοσιεύουμε την τιμή της υγρασίας "
#define arduino_watt "μονοπάτι όπου δημοσιεύουμε την τιμή της κατανάλωσης "
```

```
#define MQTT_SERVER "test.mosca.io"        //Δηλώνουμε την τοποθεσία του broker
#define mqtt_port  1883                    //Δηλώνουμε την πόρτα του brokers
```

Στην συνέχεια δημιουργούμε μία συνάρτηση όπου συνδέεται η συσκευή μας με τον broker. Αρχικά θα δημιουργήσουμε τον mqtt Client, θα του δηλώσουμε με ποιον broker θέλουμε να συνδεθεί και σε ποια πόρτα. Εν συνεχεία θα πρέπει να δηλώσουμε την εγγραφή μας, η οποία παίρνει 3 ορίσματα, το πρώτο είναι ο client που δημιουργήσαμε ακριβώς από πάνω, το δεύτερο είναι το μονοπάτι όπου θα κάνουμε εγγραφή και το τρίτο το QoS.

```
// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.

Adafruit_MQTT_Client mqtt(&client, MQTT_SERVER, mqtt_port );

Adafruit_MQTT_Subscribe onoffbutton = Adafruit_MQTT_Subscribe(&mqtt, arduino_pubsub, 1);
```

Αμέσως μετά ακολουθεί η συνάρτηση void setup η οποία είναι η μία από τις δύο βασικές συναρτήσεις που χρησιμοποιεί το Arduino και στην οποία αρχικοποιούμε τις τιμές μας. Μέσα σε αυτή την συνάρτηση γίνεται προσπάθεια να συνδεθεί η συσκευή μας με το δίκτυο.

```
void setup() {
  Serial.begin(115200);
  delay(10);
  dht.begin();
  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
  }
  mqtt.subscribe(&onoffbutton);
}
```

Εν συνεχεία προσθέτουμε στον κώδικα μας την συνάρτηση void loop() η οποία θα μας ενημερώνει όταν γίνονται αλλαγή στις εγγραφές του broker.

```
void loop() {

  if (!mqtt.connected()) {
    MQTT_connect();
  }

  Adafruit_MQTT_Subscribe *subscription;

  while ((subscription = mqtt.readSubscription(3000))) {  /
```

```
   if (subscription == &onoffbutton) {
     if (strcmp((char *)onoffbutton.lastread, "connect") == 0) {
       mqtt.publish(arduino_response, "connected", 1);
       iscon(5);
     } else if (strcmp((char *)onoffbutton.lastread, "disconnect") == 0) {
     } else if (strcmp((char *)onoffbutton.lastread, "AC1") == 0) {
       Serial.print("AC1");
       delay(500);
       calcWatt(1);
     } else if (strcmp((char *)onoffbutton.lastread, "AC0") == 0) {
       Serial.print("AC0");
       delay(500);
       calcWatt(0);
     } else if (!(strcmp((char *)onoffbutton.lastread, "done") == 0)) {
       Serial.print((char *)onoffbutton.lastread);
       mqtt.publish(arduino_response, "done", 1);
     }
   }
 }

 if (! mqtt.ping()) {
   mqtt.disconnect();
 }


 unsigned long currentMillis = millis();
 if (currentMillis - previousMillis > interval) {
   previousMillis = currentMillis;
   calcWatt(4);
 }

 if(currentMillis - previousMillisESP > interval){
Serial.print("Reset");
 }
}
```

Αμέσως μετά δημιουργούμε μία συνάρτηση την void iscon() η οποία είναι υπεύθυνη για τις εγγραφές στον broker μας για την ενημέρωση των πεδίων της θερμοκρασίας και της υγρασίας από τον αισθητήρα, καθώς επίσης και η συνάρτηση calcWatt η οποία μας ενημερώνει για την κατανάλωση του ρεύματος της συσκευής.

```
void iscon(int mode) {
 float h = dht.readHumidity();
 char hum[1];
 dtostrf(h, 2, 0, hum);
 mqtt.publish(arduino_hum, hum, 1);

 float t = dht.readTemperature();
 char temp[1];
 dtostrf(t, 2, 0, temp);
```

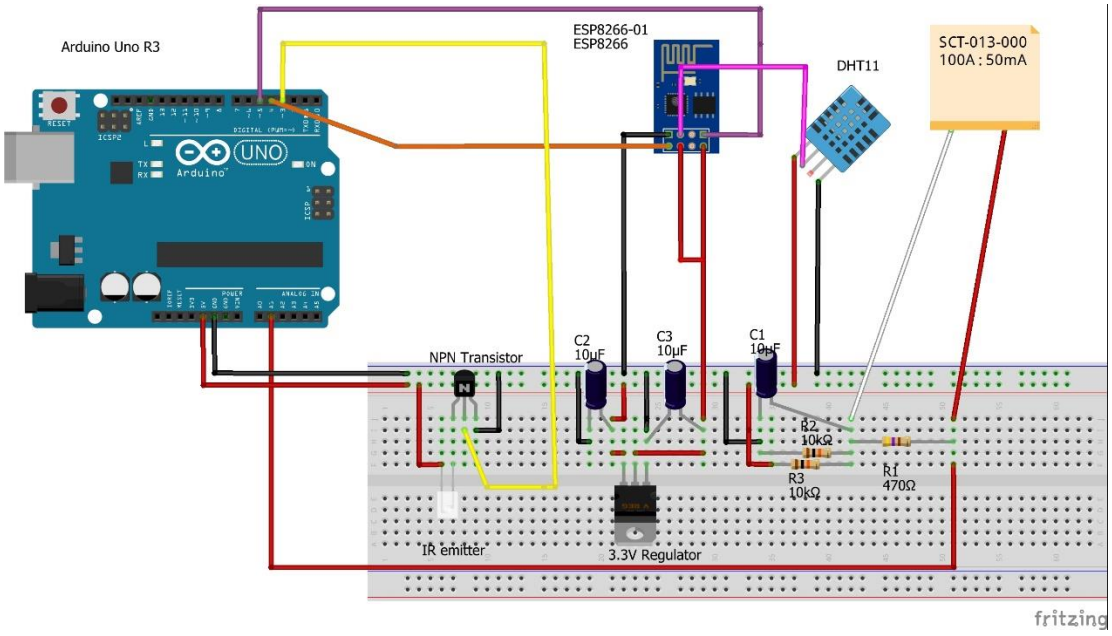
```
  mqtt.publish(arduino_temp, temp, 1);

  calcWatt(6);

}
```

Τέλος χρησιμοποιούμε ακόμη μία συνάρτηση την void MQTT_connect() η οποία είναι υπεύθυνη για να κρατάει την σύνδεση με τον broker ανοιχτή, και καλείται εάν χαθεί η σύνδεση με τον broker έως ότου βρεθεί πάλι η σύνδεση.

```
void MQTT_connect() {
  int8_t ret;
  uint8_t retries = 5;
  while ((ret = mqtt.connect()) != 0) {
    mqtt.disconnect();
    delay(5000);
    retries--;
    if (retries == 0) {
      // basically die and wait for WDT to reset me
      while (1);
    }
  }
}
```

4.5.6 Αποστολή πληροφοριών στην συσκευή μας μέσω του Arduino

Στο προηγούμενο υποκεφάλαιο δείξαμε πως θα στέλνουμε τα μηνύματα μας μέσω του Esp. Τώρα θα δούμε την τελική συνδεσμολογία όλων των συσκευών μαζί και πως μέσω του Arduino στέλνουμε το σήμα στην κλιματιστική μονάδα για ενεργοποίηση των λειτουργιών της.



Εικόνα 30 - Arduino Final Connection

Για να μπορέσουμε να στείλουμε ένα σήμα μέσω του IR emitter θα πρέπει να διαβάσουμε την

εντολή που μας στέλνει ο χρήστης από τον ιστότοπο. Όπως θα δείτε και στον κώδικα τον οποίο έχουμε ανεβάσει στο Arduino συμπεριλαμβάνει 4 βασικές βιβλιοθήκες οι οποίες μας εξυπηρετούν ώστε να έχουμε επικοινωνία μεταξύ του Esp και του Arduino. Χρησιμοποιούμε την SoftwareSerial για να μπορέσει το Arduino να δεχτεί τα μηνύματα από το Esp. Το pin 4 είναι ο receiver ο οποίος συνδέεται με τον transmitter του Esp και το pin 5 είναι ο transmitter ο οποίος συνδέεται με τον receiver του Esp. Επίσης συνδέουμε το pin 3 με τον IR emitter ο οποίος όταν ενεργοποιείται στέλνει ένα σήμα στο κλιματιστικό μας. Ακόμη δημιουργούμε μία μεταβλητή την heatpumpIR τύπου HeatpumpIR στο οποίο αποδίδουμε τις τιμές του MideaHeatpumpIR. Τέλος δημιουργούμε μία ακόμη μεταβλητή την emon1, η οποία είναι τύπου EnergyMonitor και θα την χρησιμοποιήσουμε ώστε να πάρουμε τις μετρήσεις από τον αισθητήρα του ρεύματος.

```
#include <SoftwareSerial.h>
#include <Arduino.h>
#include <MideaHeatpumpIR.h>
#include "EmonLib.h"

SoftwareSerial esp12(4, 5);       //RX,TX
IRSenderPWM irSender(3);
HeatpumpIR *heatpumpIR = new MideaHeatpumpIR();
EnergyMonitor emon1;
```

Έχουμε όμως να κάνουμε μερικές δηλώσεις μεταβλητών. Όπως φαίνεται και στο κώδικα που ακολουθεί δημιουργούμε την μεταβλητή Irms τύπου interger, η οποία μας επιστρέφει τον αριθμό των watt που καταναλώνει το κλιματιστικό μας. Δημιουργούμε επίσης μία μεταβλητή τύπου char την c, καθώς και μία μεταβλητή Boolean την gotMessage η οποία θα επιτρέπει στον IR emitter μας να στέλνει σήμα μόνο όταν έχουμε κάποιο νέο μήνυμα από το Esp. Δημιουργούμε ακόμη μία μεταβλητή τύπου String την message στην οποία θα αποθηκεύουμε το μήνυμα που μας έχει έρθει από το Esp και θα ενημερώνουμε τα πεδία των τιμών που θέλουμε να μεταβάλουμε. Ακόμη αρχικοποιούμε τα πεδία των μεταβλητών που χρειάζεται η συνάρτηση irSender, η οποία είναι υπεύθυνη για την ενεργοποίηση της συσκευής μας. Τέλος δημιουργούμε 5 πίνακες χαρακτήρων στους οποίους αποθηκεύουμε τις τιμές των πεδίων της συνάρτησης irSender.

```
int Irms;
char c;
boolean gotMessage = false;
String message;
int temp_value = 19, mode_value = 2, fanspeed_value = 0, swing_value = 0, ac_value ;
char AC[2], TP[2], MD[2], SM[2], FS[2];
```

Αμέσως μετά ακολουθεί η μία από τις δύο βασικές συναρτήσεις του Arduino, η void setup() στην οποία ανοίγουμε την επικοινωνία του Esp με την SoftwareSerial, καθώς επίσης και σε ποιον pin του Arduino θα παίρνουμε τις τιμές του αισθητήρα ρεύματος, πιο συγκεκριμένα στο A0.

```
void setup() {

  esp12.begin(115200);
  emon1.current(1, 111.1);
}
```

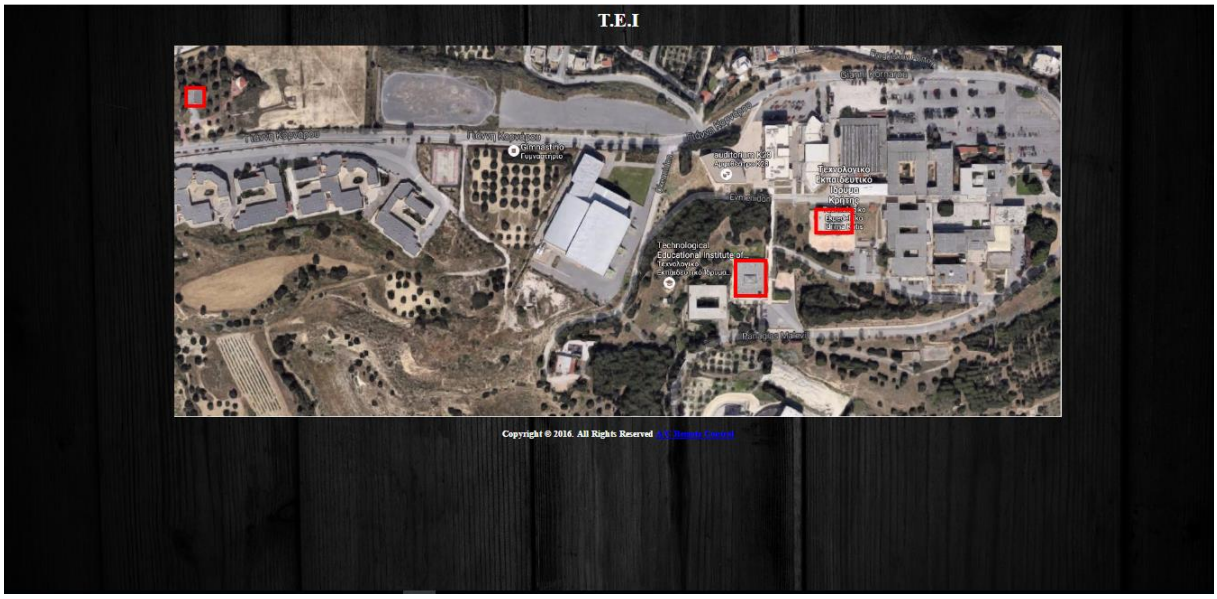
Το επόμενο κομμάτι του κώδικα μας αφορά την δεύτερη βασική συνάρτηση του Arduino, την void loop() μέσα στην οποία ενημερώνονται οι τιμές των μεταβλητών που έχουμε δημιουργήσει ώστε να σταλθεί το σήμα μας σε περίπτωση κάποιας αλλαγής. Πιο συγκεκριμένα όταν το Esp μας στείλει κάποια δεδομένα, δηλαδή η σειριακή μας θα γίνει available, οπότε θα περάσει τον έλεγχο της while,η gotMessage γίνεται true και αρχίζουμε να βάζουμε έναν-έναν τους χαρακτήρες στο String message μέχρι να σταματήσουν να έρχονται χαρακτήρες. Εν συνεχεία αναλύουμε το μήνυμα και μέσω τις συνάρτησης switch ελέγχουμε ποια τιμή έχει μεταβληθεί και ενημερώνουμε το πεδίο της. Η switch θα ελέγχει τον πρώτο χαρακτήρα του String message και ανάλογα την τιμή του θα παίρνει την τρίτη τιμή του message και θα τον μετατρέψει σε έναν ακέραιο με την εντολή atoi(). Τέλος, εφόσον στείλουμε το σήμα μας γίνεται η gotMessage false για να μην στέλνει συνέχεια την εντολή χωρίς να έχει αλλάξει κάποια τιμή. Ενδεικτικά φαίνονται στην συνέχεια αυτά που αναφέραμε σε ένα κομμάτι κώδικα.

```
void loop() {
  message = "";
  while (esp12.available()) {
    gotMessage = true;
    c = esp12.read();
    message += c;
    delay(10);
  }
  if (gotMessage) {
    switch (message[0]) {
      case 'A':
        AC[0] = message[2];
        AC[1] = '\0';
        AC[2] = '\0';
        ac_value = atoi(AC);
        break;
      default :
        gotMessage = false;
    }
  }
  if (gotMessage) {
    heatpumpIR->send(irSender, ac_value, mode_value, fanspeed_value , temp_value , swing_value, HDIR_AUTO);
    gotMessage = false;
  }
}
```

5. Χρήση του συστήματος

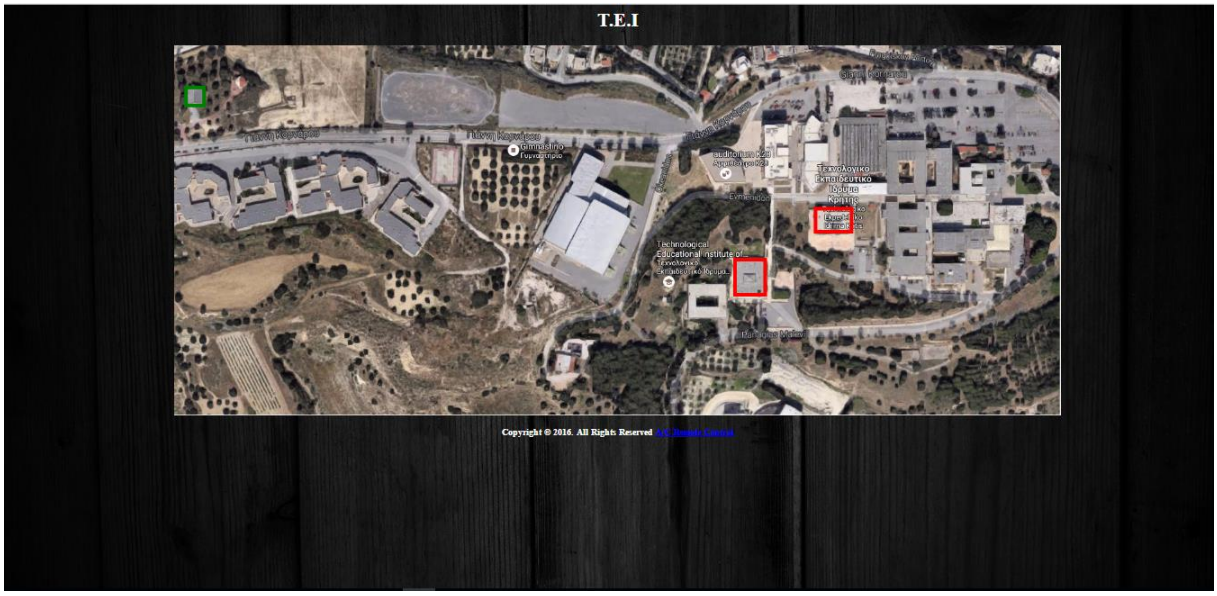
Σε αυτό το κεφάλαιο θα δείξουμε μερικά παραδείγματα της λειτουργίας του κλιματιστικού. Δηλαδή θα δείξουμε με εικόνες την αντίδραση που έχει το GUI της ιστοσελίδας μας σε διάφορα σενάρια λειτουργίας ή μη, μιας κλιματιστικής μονάδας.

Παράδειγμα πρώτο : Επισκέπτεται ένας admin την αρχική σελίδα της εφαρμογής και βλέπει στην αρχική οθόνη της εφαρμογής ότι όλες οι κλιματιστικές μονάδες του κτιρίου είναι κλειστές.



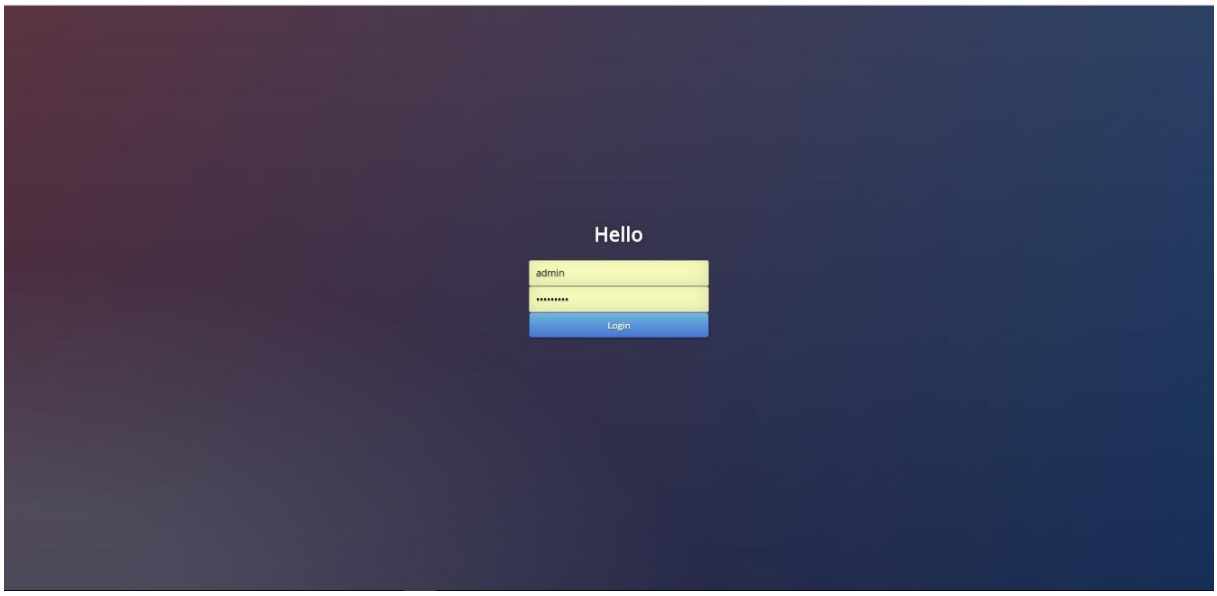
Εικόνα 31 - Scenario 1

Παράδειγμα δεύτερο : Επισκέπτεται ένας admin την αρχική σελίδα της εφαρμογής και παρατηρεί ότι κάποια κλιματιστική μονάδα του κτιρίου είναι ανοιχτή ενώ είναι βράδυ και δεν υπάρχει κάποιος στο κτίριο (προφανώς έχει μείνει ξεχασμένο από κάποιον που βρισκόταν στο κτίριο νωρίτερα). Το αποτέλεσμα που θα έχει η αρχική του οθόνη είναι :



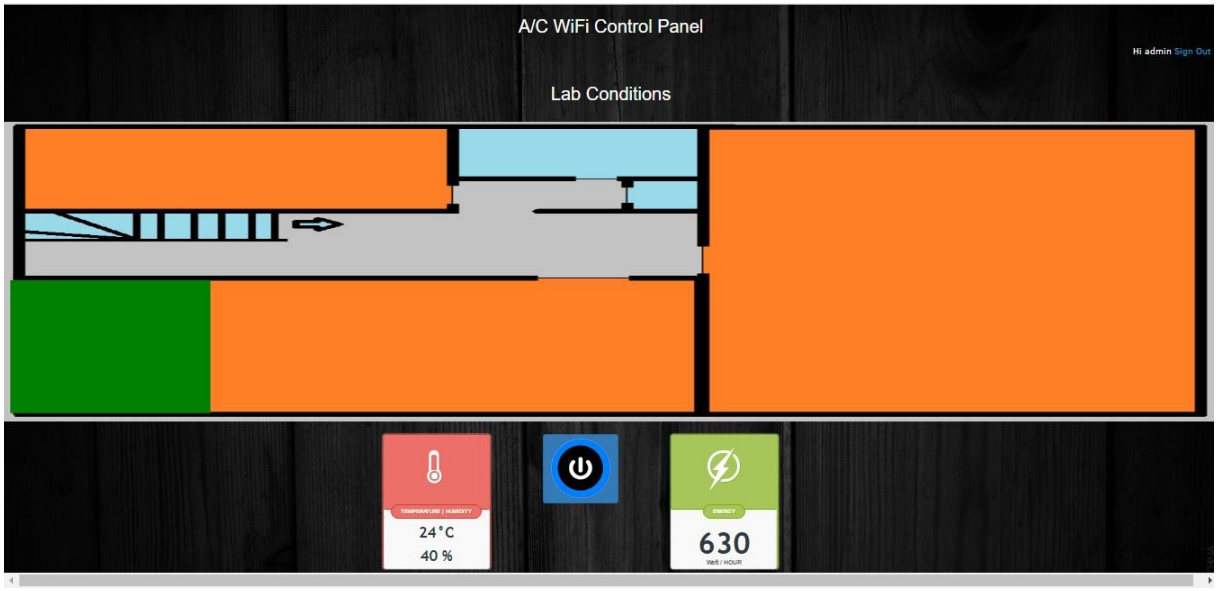
Εικόνα 32 - Scenario 2a

Στην συνέχεια ο admin θα πρέπει να κάνει είσοδο στο σύστημα επιλέγοντας το κτίριο στο οποίο φαίνεται να έχει ξεχαστεί κάποια κλιματιστική μονάδα ανοικτή. Αυτό θα επιτευχθεί κάνοντας click πάνω στο κτίριο το οποίο έχει πράσινο περίγραμμα, και πηγαίνει στην ιστοσελίδα εισόδου όπου πρέπει να κάνει χρήση των στοιχείων εισόδου που του έχουν δοθεί, όπως φαίνεται και στην παρακάτω εικόνα.



Εικόνα 33 – Login

Όταν κάνει είσοδο στο σύστημα ο admin μπορεί να δει ποιες μονάδες του κτιρίου είναι ανοικτές από την κατανάλωση των watt. Επιλέγοντας αρχικά τον χώρο του κτιρίου κάνει σύνδεση με την συσκευή του δωματίου και παίρνει δεδομένα από την συσκευή για την λειτουργία της και τις συνθήκες του χώρου. Όπως φαίνεται και στο στην εικόνα που ακολουθεί ο admin επιλέγει το δωμάτιο του κτιρίου και βλέπει ότι η κατανάλωση του κλιματιστικού είναι 630 Watt, κάτι το οποίο δηλώνει ότι το κλιματιστικό είναι ανοιχτό.



Εικόνα 34 - Scenario 2b

Εν συνεχεία έχει την δυνατότητα να κλείσει την συσκευή από το button On/Off ή να τροποποιήσει τις λειτουργίες της όπως φαίνεται στην εικόνα που ακολουθεί.

Εικόνα 35 - Scenario 2c

<u>Παράδειγμα τρίτο :</u> Ένα άλλο υποθετικό σενάριο εφαρμογής του συστήματος είναι ο έλεγχος των κλιματιστικών μονάδων ανάλογα τις καιρικές συνθήκες που επικρατούν για μεγαλύτερη οικονομία στην κατανάλωση. Για παράδειγμα συνδέεται ο admin και παρατηρεί ότι η κλιματιστική μονάδα δουλεύει σε πολύ ψηλή θερμοκρασία ενώ ήδη η εξωτερική θερμοκρασία της ατμόσφαιρας είναι ψιλά, επομένως έχει την δυνατότητα να επέμβει και να κατεβάσει την θερμοκρασία στα επίπεδα που αυτός θεωρεί σωστά ώστε να μειωθεί η κατανάλωση. Κάθε φορά που αλλάζει λειτουργία θα ενημερώνεται με ένα μήνυμα success πάνω δεξιά στην σελίδα μας.

Εικόνα 35 - Scenario 3a 1

Ξέρουμε ότι η θερμοκρασία είναι υψηλή όμως το κλιματιστικό λειτουργεί στους 28 βαθμούς και καταναλώνει 930 watt, οπότε ο admin θα επιλέξει να βάλει το κλιματιστικό στους 19 βαθμούς.

Εικόνα 35 - Scenario 3a 2

Βλέπουμε το success μήνυμα πάνω δεξιά και κατευθείαν την μείωση της κατανάλωσης του κλιματιστικού.

6. Αποτελέσματα

Η ανάπτυξη του λογισμικού έδωσε την ευκαιρία συνδυασμού πολλών τεχνολογιών, οι οποίες συνεργάστηκαν για την ανάπτυξη μιας πρωτότυπης εφαρμογής. Η τελική εφαρμογή περιλαμβάνει μια ιστοσελίδα ελέγχου στην οποία πάνω έχουμε βάλει όλες τις συνδέσεις mqtt που χρειάζονται για την απομακρυσμένη επικοινωνία με τις κλιματιστικές μονάδες και την πρότυπη κατασκευή με τον ελεγκτή που απαιτείται να τοποθετηθεί σε κάθε κλιματιστικό για τον έλεγχό του και την επικοινωνία με την ιστοσελίδα ελέγχου. Με το να στήσουμε τη βάση δεδομένων του συστήματος δυναμικά, η διαδικασία για να προστεθεί νέος χώρος στη βάση είναι εύκολος και δεν χρειάζεται να γίνουν όλα ξανά από την αρχή.

Με την χρήση του mqtt πρωτοκόλλου καταφέραμε να βγάλουμε ένα πολύ μεγάλο φορτίο από τον σέρβερ μας και από την συσκευή esp8266. Παράλληλα, ο κώδικας για τις αλλαγές των τιμών έγινε πιο εύκολος και πιο ξεκάθαρος και οι ανταλλαγες μυνημάων μέσω δικτύου περιορίζονται στις εντελώς απαραίτητες. Ο φόρτος εργασίας μεταφέρεται στον broker και τα μηνύματα που στέλνουμε δεν χάνονται.

6.1 Μελλοντική Εργασία και επεκτάσεις

Η εφαρμογή μπορεί να επεκταθεί στο μέλλον και πάνω στο Arduino να προστεθούν πιο πολλά αισθητήρια. Ένα αισθητήριο που θα μπορούσε να προσθέτει είναι ένα απλό photoresistor το οποίο θα μπορεί να μας ειδοποιεί για τον αν κάποιος βρίσκεται στο χώρο και έχει ανοίξει το φως. Εφόσον ο χρήστης το επιθυμεί, το σύστημα θα μπορούσε να ρυθμιστεί να στέλνει ειδοποιήσεις στο κινητό του, εφόσον το έχουμε ενεργοποιήσει.

Επιπλέον, το σύστημα θα μπορούσε παράλληλα να ελέγχει και άλλες συσκευές μέσα στο χώρο, οι οποίες χρησιμοποιούν IR σήματα, χωρίς να χρειαζόμαστε τηλεχειριστήριο για το κάθε ένα.

Τέλος, διάφοροι ευφυείς αλγόριθμοι δυναμικού ελέγχου των συσκευών θα μπορούσαν να εφαρμοστούν, π.χ. με βάση τις εξωτερικές καιρικές συνθήκες, ώστε να επιτυγχάνεται η μεγαλύτερη δυνατόν εξοικονόμηση ενέργειας και περιορισμός των ανθρώπινων αυθαιρεσιών.

7. Βιβλιογραφία

I. https://www.tutorialspoint.com/internet_of_things/index.htm
II. https://el.wikipedia.org/wiki/XAMPP
III. https://el.wikipedia.org/wiki/Apache_HTTP_%CE%B5%CE%BE%CF%85%CF%80%CE%B7%CF%81%CE%B5%CF%84%CE%B7%CF%84%CE%AE%CF%82
IV. https://el.wikipedia.org/wiki/MySQL
V. https://el.wikipedia.org/wiki/PHP
VI. https://el.wikipedia.org/wiki/JavaScript
VII. https://el.wikipedia.org/wiki/JQuery
VIII. https://en.wikipedia.org/wiki/MQTT
IX. https://el.wikipedia.org/wiki/Windows_10
X. https://el.wikipedia.org/wiki/Arduino
XI. https://www.sparkfun.com/products/13678
XII. https://el.wikipedia.org/wiki/%CE%A0%CF%85%CE%BA%CE%BD%CF%89%CF%84%CE%AE%CF%82
XIII. https://el.wikipedia.org/wiki/%CE%91%CE%BD%CF%84%CE%B9%CF%83%CF%84%CE%AC%CF%84%CE%B7%CF%82
XIV. http://www.hivemq.com/