ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΕΦΑΡΜΟΣΜΕΝΑ ΗΛΕΚΤΡΟΝΙΚΑ ΣΥΣΤΗΜΑΤΑ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΚΑΤΑΣΚΕΥΗ ΕΝΟΣ ΑΥΤΟΜΑΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΕΛΕΓΧΟΥ ΕΝΥΔΡΕΙΟΥ»

Του φοιτητή Καμπουρίδη Ηλία
Αρ. Μητρώου:Elem51906m

Επιβλέπων
Κος Άγγελος Γιακουμής
Βαθμίδα Λέκτορας

Σεπτέμβριος 2021

Τίτλος Δ.Ε.: Σχεδιασμός και κατασκευή ενός αυτόματου συστήματος ελέγχου ενυδρείου.
Κωδικός Δ.Ε. 21155
Ονοματεπώνυμο φοιτητή: Ηλίας Καμπουρίδης
Ονοματεπώνυμο εισηγητή: Άγγελος Γιακουμής
Ημερομηνία ανάληψης Δ.Ε. 25/8/2020
Ημερομηνία περάτωσης Δ.Ε. 22/9/2021

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της μεταπτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Μεταπτυχιακό Πρόγραμμα Σπουδών «Εφαρμοσμένα Ηλεκτρονικά Συστήματα» στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

«Αφιέρωση»

Η παρούσα διπλωματική εργασία είναι αφιερωμένη στην σύζυγό μου και τους γονείς μου για την ηθική υποστήριξη που μου παρείχαν κατά την διάρκεια των σπουδών μου.

Πρόλογος

Η ενασχόληση με τα οικιακά ενυδρεία αποτελεί ένα αρκετά γνωστό και διαδεδομένο χόμπι σε ολόκληρο τον κόσμο. Όλο και περισσότεροι άνθρωποι προσπαθούν να προσομοιώσουν ένα τμήμα από το υδάτινο περιβάλλον (ψάρια, υδρόβια φυτά) και να το μεταφέρουν μέσω του δοχείου του ενυδρείου (γυάλα) στον προσωπικό τους χώρο. Η ενασχόληση όμως με το συγκεκριμένο αντικείμενο, απαιτεί την συστηματική παρακολούθηση και συντήρηση ενός τέτοιου συστήματος καθώς αποτελεί ζωτικής σημασίας για τους οργανισμούς που φιλοξενούνται στο εσωτερικό του ενυδρείου. Για τον σκοπό αυτό υπάρχουν διάφορες συσκευές στην αγορά όπως οι ηλεκτρονικές ταΐστρες για την περιοδική προσθήκη συγκεκριμένης ποσότητας τροφής, συσκευές μέτρησης της θερμοκρασίας ή του ph του νερού καθώς και άλλες σχετικές συσκευές μέτρησης. Παρόλο που η ύπαρξη τέτοιον βοηθητικών συσκευών κάνει πιο εύκολη την συντήρηση του ενυδρείου, δεν αλλάζει το γεγονός ότι υπάρχουν περιορισμοί στους συγκεκριμένους αυτοματισμούς, όπως ο απομακρυσμένος έλεγχος ή η μη έγκαιρη ενημέρωση όταν κάποια από τις παραμέτρους αλλάζει και απαιτείται κάποια διορθωτική ενέργεια. Για παράδειγμα η εξάτμιση του νερού στο δοχείο, από την υψηλή θερμοκρασία τους καλοκαιρινούς μήνες έχει ως αποτέλεσμα η στάθμη να πέφτει και να απαιτείται η συμπλήρωση ποσότητας νερού. Με σκοπό τα παραπάνω μέσω της παρούσας διπλωματικής εργασίας επιχειρείται η δημιουργία ενός αυτόματου συστήματος ελέγχου ενυδρείου.

Περίληψη

Στην παρούσα διπλωματική εργασία σχεδιάστηκε και υλοποιήθηκε ένα «έξυπνο» σύστημα ενυδρείου. Σκοπός ήταν η διαχείριση και ο έλεγχος του ενυδρείου να πραγματοποιείται μέσω μιας ολοκληρωμένης εφαρμογής android. Η ενασχόληση με το συγκεκριμένο αντικείμενο, απαιτεί την συστηματική παρακολούθηση και συντήρηση του συστήματος, καθώς αποτελεί ζωτικής σημασίας για τους οργανισμούς που φιλοξενούνται στο εσωτερικό του. Λειτουργίες όπως η συνεχής μέτρηση και ενημέρωση της θερμοκρασίας και της στάθμης του νερού στο εσωτερικό δοχείο του ενυδρείο, η χειροκίνητη ή προγραμματισμένη απελευθέρωση τροφής στο ενυδρείο και η χειροκίνητη ή αυτόματη ενεργοποίηση του φως αποτελούν τους κύριους αυτοματισμούς του συστήματος με κεντρικό σύστημα ελέγχου τον μικροελεγκτή STM32H743ZI.

Για τον σχεδιασμό και την υλοποίηση του «έξυπνου» ενυδρείου ήταν απαραίτητο να οριστούν ορισμένες προδιαγραφές όσον αφορά στη λειτουργία και τους στόχους του συστήματος ώστε να επιτευχθούν συγκεκριμένοι αυτοματισμοί. Με τον τρόπο αυτό, η επιλογή συγκεκριμένων, αυτοματοποιημένων λειτουργιών είχε ως αποτέλεσμα την έρευνα των κατάλληλων αισθητήρων και του κατάλληλου μικροελεγκτή. Η επιλογή των κατάλληλων αισθητήρων είχε σαν αποτέλεσμα την επιλογή των κατάλληλων ενεργοποιητών.

Στα πλαίσια επομένως της υλοποίησης του «έξυπνου ενυδρείου», το σύστημα χωρίστηκε σε δύο βασικά μέρη. Το πρώτο μέρος αφορά την κατασκευή που αποτελείται από το ενυδρείο, τα αισθητήρια και τον μικροελεγκτή ενώ το δεύτερο μέρος την ανάπτυξη μιας ολοκληρωμένης εφαρμογής android. Η δομή του συστήματος αποτελείται από την κατασκευή, τον server και το app. Στην κατασκευή ανήκουν το ενυδρείο, τα αισθητήρια και ο μικροελεγκτής. Ο server βρίσκεται στην πλατφόρμα Amazon EC2 και χρησιμοποιώντας το πρωτόκολλο Mqtt, δίνεται η δυνατότητα επικοινωνίας με τις συσκευές που χρησιμοποιούνται στην κατασκευή. Ενώ το app έχει εγκατασταθεί σε ένα κινητό τηλέφωνο νέας τεχνολογίας (smart phone) με λειτουργικό android. Τέλος η αρχή λειτουργίας του συστήματος είναι η αποστολή σημάτων μεταξύ της κατασκευής, του server και του app ώστε να πραγματοποιούνται οι προκαθορισμένες ενέργειες με σκοπό την αυτοματοποίηση των λειτουργιών.

«Design and construction of an automatic aquarium control system»

«Ilias Kampouridis»

Abstract

This dissertation draws and implements a smart aquarium system. Its aim is to manage and control the aquarium through a complete android application. This particular object demands the systematic observation and system maintenance as it is vital for the organisms that exist in it. Several functions, such as the constant temperature measurement and update, the manual food dispenser in the aquarium or the automatic light activation consist both the main system automation and the microcontroller STM32H743ZI which is the main control system.

The design and implementation of the smart aquarium requires several specifications regarding the function and the goals that we would like to achieve through this system. Therefore, the choice of specific automated functions resulted in the search of the appropriate sensors and microcontroller which also led to the choice of the most appropriate activators.

As a consequence, the implementation of the smart aquarium divided the system into two main parts. The first one involves the aquarium construction, the sensors and the microcontroller while the second one focuses on the development of a complete android application. Moreover, the system structure is composed of the construction, the server and the app. The aquarium, the sensors and the microcontroller belong to the construction. The server is on the Amazon EC2 platform and runs the Mqtt, providing communication with the devices that are used on the construction. On the other hand, the app is installed on a new technology mobile phone (smart phone) with android software. Lastly, the system principal function involves the signal transmission among the construction, the server and the app in order to carry out the scheduled tasks that will automate all the functions.

Ευχαριστίες

Οφείλω να ευχαριστήσω θερμά τον κύριο Γιακούμη Άγγελο, Καθηγητή του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΠΜΣ Εφαρμοσμένα Ηλεκτρονικά Συστήματα για την εμπιστοσύνη, την ανάθεση και την επίβλεψη της διπλωματικής εργασίας αλλά και όλους τους καθηγητές του Τμήματος. Επίσης θα ήθελα να ευχαριστήσω την σύζυγό μου και την οικογένειά μου για την υποστήριξη και την αγάπη τους.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
IoT	Internet of Things
ΔτΠ	Διαδίκτυο των Πραγμάτων
IAB	Internet Architecture Board
IP	Internet Protocol
ITU	International Telecommunication Union
M2M	Machine to Machine
MQTT	Message Queuing Telemetry Trasport
QoS	Quality of Service
LTE	Long Term Evolution

Κεφάλαιο 1ο: Αντικείμενο της Διπλωματικής Εργασίας

1.1 Εισαγωγή

Ολοένα και περισσότερο στις μέρες μας, αν και η ενασχόληση με τα οικιακά ενυδρεία αποτελεί εδώ και χρόνια ένα αρκετά γνωστό και διαδεδομένο χόμπι σε ολόκληρο τον κόσμο, λόγω της τεχνολογικής ανάπτυξης η οποία έδωσε την δυνατότητα να αναπτυχθούν διάφοροι αυτοματισμοί στην λειτουργία ενός τέτοιου συστήματος, είχε ως αποτέλεσμα, η ενασχόληση με τα ενυδρεία να αποκτά θετικό πρόσημο από διάφορους καταναλωτές. Έτσι, όλο και περισσότεροι άνθρωποι προσπαθούν να προσομοιώσουν ένα τμήμα από το υδάτινο περιβάλλον (ψάρια, υδρόβια φυτά) και να το μεταφέρουν μέσω των δοχείων του ενυδρείου στον προσωπικό τους χώρο (σπίτι, χώρος εργασίας).

Η ενασχόληση όμως με το συγκεκριμένο αντικείμενο, απαιτεί την συστηματική παρακολούθηση και συντήρηση ενός τέτοιου συστήματος καθώς αποτελεί ζωτικής σημασίας για τους οργανισμούς που φιλοξενούνται στο εσωτερικό δοχείο του ενυδρείου. Επομένως το ζήτημα της συντήρησης των ενυδρείων αποτελεί αν όχι το σημαντικότερο, τουλάχιστον ένα από τα σημαντικότερα κομμάτια της ενασχόλησης με αυτά. Η διαδικασία συντήρησης ενός τέτοιου συστήματος έχει αρκετές παραμέτρους που πρέπει κάποιος να λάβει σοβαρά υπόψη του. Είναι σημαντικό η σωστή λειτουργία ενός ενυδρείου να εξασφαλίζεται ακόμα και όταν ο κάτοχος του ενυδρείου είτε απουσιάζει, είτε βρίσκεται στο διπλανό δωμάτιο από το σημείο όπου έχει εγκατασταθεί το σύστημα, έτσι ώστε να ειδοποιείται έγκαιρα όταν κάποια από τις παραμέτρους του αλλάξει με σκοπό αν είναι δυνατόν να προβεί στις απαραίτητες διορθωτικές ενέργειες. Για τον σκοπό αυτό υπάρχουν διάφορες συσκευές στην αγορά όπως οι ηλεκτρονικές ταΐστρες για την περιοδική προσθήκη συγκεκριμένης ποσότητας τροφής στο δοχείο του ενυδρείο, συσκευές μέτρησης της θερμοκρασίας ή μέτρησης του ph του νερού καθώς και άλλες σχετικές συσκευές μέτρησης. Παρόλο που η ύπαρξη τέτοιον βοηθητικών συσκευών κάνει πιο εύκολη την συντήρηση του ενυδρείου, δεν αλλάζει το γεγονός ότι υπάρχουν περιορισμοί στους συγκεκριμένους αυτοματισμούς.

Για τον λόγο αυτό, μέσω της παρούσας Δ.Ε. επιχειρείται η δημιουργία ενός αυτόματου συστήματος ελέγχου ενυδρείου, έτσι ώστε το σύστημα να έχει τον πλήρη έλεγχο των λειτουργιών μέσω εφαρμογής android. Η χειροκίνητη ή προγραμματισμένη απελευθέρωση τροφής, η διατήρηση της σωστής λειτουργίας του ενυδρείου ελέγχοντας την στάθμη του νερού, την θερμοκρασία και η χειροκίνητη ή αυτόματη ενεργοποίηση του φως αποτελούν τους κύριους αυτοματισμούς του συστήματος με κεντρικό σύστημα ελέγχου τον μικροελεγκτή STM32H743ZI της σειράς STM32 με επεξεργαστή Arm καθιστώντας το ένα «έξυπνο» σύστημα ενυδρείου.

1.2 Αντικείμενο και σκοπός της Διπλωματικής Εργασίας

Το επίπεδο της τεχνολογίας έχει δώσει την δυνατότητα στις μέρες μας να αναπτυχθούν πολλές εφαρμογές Internet of Things. Η τεχνολογία IoT αποτελεί το επόμενο μεγάλο βήμα στον χώρο της τεχνολογίας και είναι αρκετά διαδεδομένη σε όλο τον κόσμο. Η χρήση της, καθιστά ένα σύστημα «έξυπνο», όπως ένα «έξυπνο σπίτι» ή ένα «έξυπνο κτήριο». Επίσης η τεχνολογία IoT μπορεί να εφαρμοστή και σε συσκευές που χρησιμοποιούνται καθημερινά από τον άνθρωπο. Λύνοντας με αυτόν τον τρόπο πολλά προβλήματα, όπως τον απομακρυσμένο έλεγχο και την διαχείριση μιας κατάστασης χωρίς την ανθρώπινη φυσική παρουσία στον ίδιο χώρο. Η τεχνολογία IoT είναι αυτή που μπορεί επίσης να εφαρμοστεί στην παρούσα εργασία για την δημιουργία ενός «έξυπνου» συστήματος ενυδρείου ψαριών.

Με την παρούσα Δ.Ε. παρατίθενται, ο σχεδιασμός και η κατασκευή ενός «έξυπνου» συστήματος ελέγχου ενυδρείου, όπου αφενός επιδιώκεται η αυτοματοποίηση των λειτουργιών του χωρίς την ανθρώπινη παρουσία, με την χρήση του μικροελεγκτή STM32H743ZI, αφετέρου ο απομακρυσμένος έλεγχος, η ενημέρωση αλλαγής μιας παραμέτρου, η αλλαγή μιας κατάστασης και η εκτέλεση μιας λειτουργίας χειροκίνητα από τον χρήστη ή αυτόματα εκτελώντας μια συγκεκριμένη ενέργεια μέσα από την ανάπτυξη μιας ολοκληρωμένης εφαρμογής android που σκοπό έχει την αυτοματοποίηση των λειτουργιών ενός τέτοιου συστήματος.

1.3 Οργάνωση των κεφαλαίων της Διπλωματικής Εργασίας

Στην συγκεκριμένη Δ.Ε. αναλύεται τόσο η μεθοδολογία σχεδιασμού της κατασκευής και της εφαρμογής android όσο και η υλοποίηση τους. Στα πλαίσια της υλοποίησης του «έξυπνου ενυδρείου», το σύστημα χωρίστηκε σε δύο βασικά μέρη. Το πρώτο μέρος αφορά την κατασκευή που αποτελείται από το ενυδρείο, τα αισθητήρια και τον μικροελεγκτή ενώ το δεύτερο μέρος αφορά την ανάπτυξη μιας ολοκληρωμένης εφαρμογής android. Συνολικά αποτελείται από έξι κεφάλαια.

Στα κεφάλαια που θα ακολουθήσουν, θα αναλυθούν τα στοιχεία της εργασίας τόσο σε θεωρητικό επίπεδο όσο και σε επίπεδο σχεδίασης και υλοποίησης. Συγκεκριμένα το 1ο κεφάλαιο αποτελεί μια σύντομη εισαγωγή στο αντικείμενο και τον σκοπό της Δ.Ε. Στο 2ο και στο 3ο κεφάλαιο της εργασίας γίνεται μια θεωρητική αναφορά. Σε αυτό περιλαμβάνονται θεωρητικές αναφορές σχετικά με τα στοιχεία που χρησιμοποιούνται για την υλοποίηση του συστήματος. Στο 4ο κεφάλαιο περιγράφεται η υλοποίηση του συστήματος, αναλύονται τα χαρακτηριστικά των αισθητήρων και ο τρόπος λειτουργίας τους στο σύστημα αλλά και η εφαρμογή android ώστε να γίνει κατανοητό η αλληλεπίδραση του συστήματος με τον χρήστη. Στο 5ο κεφάλαιο παρουσιάζονται τα προγραμματιστικά εργαλεία, οι γλώσσες προγραμματισμού και επεξήγεται το λογισμικό τόσο του μικροελεγκτή όσο και της εφαρμογής μέσα από διαγράμματα ροής. Στο 6ο και τελευταίο κεφάλαιο παρουσιάζονται τα συμπεράσματα της παρούσας διπλωματικής εργασίας και τυχόν μελλοντικές επεκτάσεις που θα μπορούσαν να γίνουν με σκοπό τη βελτίωση του συγκεκριμένου συστήματος.

Η ανάπτυξη ολόκληρου του λογισμικού (κώδικες), αναφορές σε βιβλία, σε διαδικτυακές σελίδες και σε επιστημονικά άρθρα που κατέστησαν δυνατή την ολοκλήρωση της εργασίας αυτής τόσο όσον αφορά στη σύνταξή της όσο και στην υλοποίησή της υπάρχουν στο τέλος της εργασίας.

1.4 Επίλογος

Στο κεφάλαιο αυτό παρατίθενται μια σύντομη εισαγωγή στο αντικείμενο της Δ.Ε. Σκοπός της δημιουργίας του κεφαλαίου αυτού είναι ο αναγνώστης να κατανοήσει το περιεχόμενο του θέματος, το οποίο αφορά τον σχεδιασμό και την υλοποίηση ενός σύγχρονου και «έξυπνου» συστήματος ενυδρείου με κεντρικό σύστημα ελέγχου τον μικροελεγκτή της σειράς STM32 αλλά και την δημιουργία μιας ολοκληρωμένης εφαρμογής android για την διαχείριση των αυτοματισμών του συστήματος από τον χρήστη, χρησιμοποιώντας σύγχρονες τεχνολογίες και προγραμματιστικά εργαλεία με σκοπό την αυτοματοποίηση των λειτουργιών ενός τέτοιου συστήματος. Επίσης αναφέρεται το περιεχόμενο των κεφαλαίων της Δ.Ε που θα ακολουθήσουν έτσι ώστε ο αναγνώστης να σχηματίσει μια πρώτη εικόνα για τα κεφάλαια που έπονται.

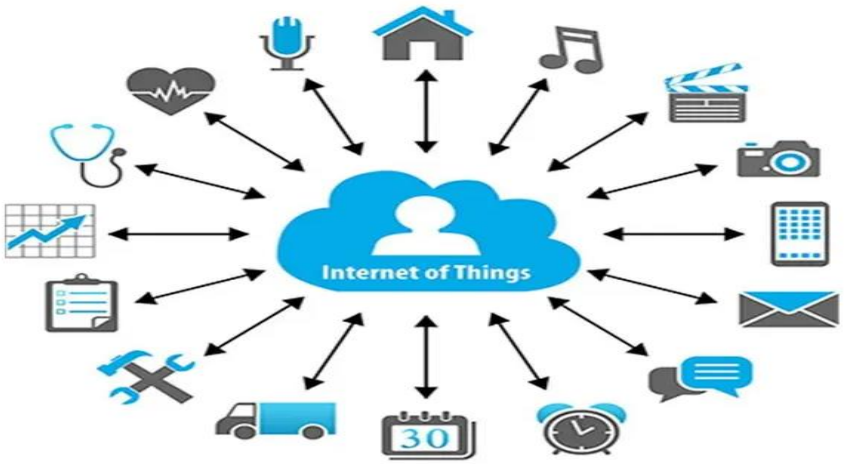
Κεφάλαιο 2ο: Διαδίκτυο των Πραγμάτων

2.1 Εισαγωγή

Οι μέρες μας θα μπορούσαν να χαρακτηριστούν ως οι μέρες ανάπτυξης του Διαδικτύου των Πραγμάτων καθώς οι απαραίτητες τεχνολογίες υπάρχουν και ολοένα και βελτιώνονται, καθιστώντας πλέον περισσότερο από ορατό την εξέλιξη της τεχνολογίας αυτής. Το ΔτΠ αποτελεί μία από τις πολλά υποσχόμενες τεχνολογίες, η οποία θα απασχολήσει την παγκόσμια αγορά της τεχνολογίας τα επόμενο χρόνια. Καινούργιες τεχνολογίες ασύρματης μετάδοσης δεδομένων έχουν αναπτυχθεί και συνεχίζουν να εξελίσσονται με το πέρας του χρόνου. Κάποιες από αυτές αναπτύχθηκαν για την κινητή τηλεφωνία και έχουν τη δυνατότητα κάλυψης μεγάλων αποστάσεων όπως το GPRS, LTE, LoRaWan ενώ άλλες έχουν δημιουργηθεί για τη κάλυψη τοπικών ασύρματων δικτύων όπως το Wifi, Bluetooth, ZigBee, 6LoWPAN. Από την άλλη πλευρά στο επίπεδο του hardware, οι επεξεργαστές γίνονται ολοένα μικρότεροι και ισχυρότεροι όπως τα σύγχρονα και «έξυπνα» smart phones ή smart watch με πρόσβαση στο διαδίκτυο. Ακόμα μικροεπεξεργαστές (Arm, Arduino, Raspberry Pi) μικροί σε μέγεθος, χαμηλού κόστους, με πολλές δυνατότητες όπως σύνδεσης σε ασύρματα δίκτυα και στο διαδίκτυο είναι διαθέσιμοι στην αγορά και δίνουν την δυνατότητα σε ολόκληρη την τεχνολογική κοινότητα (φοιτητές, ερευνητές) να δημιουργήσουν και να αναπτύσσουν έξυπνες ιδέες βασιζόμενοι στις νέες αυτές τεχνολογίες. Με αυτό τον τρόπο ωθούν το όραμα του ΔτΠ περισσότερο από κάθε άλλη φορά καθώς η τεχνολογία είναι πλέον ώριμη, προσιτή και σε ένα αρκετά καλό επίπεδο ώστε η τεχνολογική κοινότητα να διαθέτει όλα εκείνα τα εργαλεία που είναι απαραίτητα για να υλοποιήσει δίκτυα από καθημερινά αντικείμενα που συνδέονται μεταξύ τους και είναι διαχειρίσιμα χωρίς την ανθρώπινη παρουσίαση αλλά από απόσταση. Καθώς το ΔτΠ διεισδύει στην καθημερινότητα των ανθρώπων έχει σαν αποτέλεσμα να γίνεται περισσότερο δημοφιλής ως τεχνολογία, όμως πλέον, μπορεί να αποτελέσει έναν ξεχωριστό επιστημονικό κλάδο παρά μία απλή τεχνολογία.

2.2 Internet of Things (IoT)

Ενώ ο όρος Διαδίκτυο των πραγμάτων είναι σχετικά καινούργιος, η έννοια του συνδυασμού υπολογιστών και των δικτύων για παρακολούθηση και έλεγχο συσκευών υπάρχει εδώ και δεκαετίες. Το 1999 ο Βρετανός Kevin Ashton χρησιμοποίησε για πρώτη φορά τον όρο Διαδίκτυο των πραγμάτων (Internet of Things, IoT) για να περιγράψει ένα σύστημα στο οποίο τα αντικείμενα του φυσικού κόσμου θα μπορούσαν να συνδεθούν με το διαδίκτυο χρησιμοποιώντας αισθητήρες **[1]**.



Εικόνα 1: Internet of Things, (www.justcreative.com)

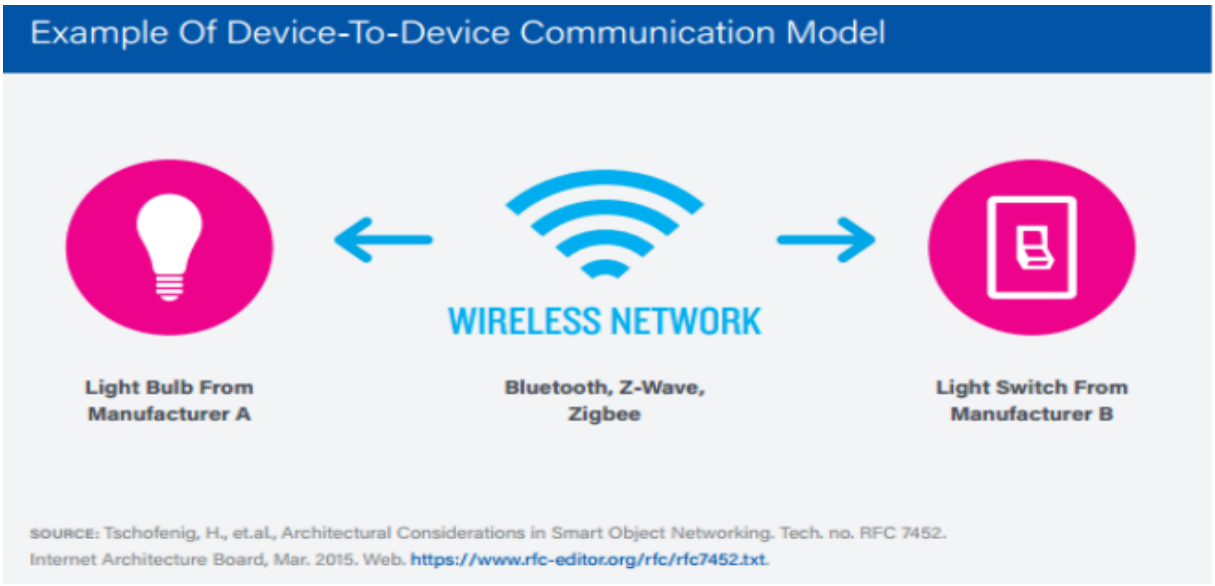
Αυτό που εννοείται με τον όρο ΔτΠ και αυξάνεται με πολύ γρήγορους ρυθμούς, έχει να κάνει με την χρήση των «έξυπνα» συνδεμένων συσκευών και συστημάτων για την συλλογή δεδομένων που αντλούνται από αισθητήρες και ενεργοποιητές σε μηχανήματα και άλλα φυσικά αντικείμενα. Τα περισσότερα φυσικά αντικείμενα όπως τα μηχανήματα, οι μεταφορές, οι υποδομές και οι συσκευές είναι εξοπλισμένες με δικτυωμένους αισθητήρες και ενεργοποιητές που τους επιτρέπουν να παρακολουθούν το περιβάλλον τους, να αναφέρουν την κατάστασή τους, να λαμβάνουν οδηγίες και ακόμη να αναλαμβάνουν δράση με βάση τις πληροφορίες που λαμβάνουν. Επομένως για τους ανθρώπους, το ΔτΠ έχει τη δυνατότητα να προσφέρει λύσεις που βελτιώνουν σημαντικά την ενεργειακή απόδοση, την ασφάλεια, την υγεία, την εκπαίδευση και πολλές άλλες πτυχές της καθημερινής ζωής. Πρόκειται επομένως για ένα δίκτυο καθημερινών αντικειμένων τα οποία βρίσκονται σε σύνδεση μεταξύ τους και συλλέγουν πληροφορίες με σκοπό να εκτελέσουν μία ενέργεια. Τέτοιου είδους συσκευές έχουν εφαρμογή σε διάφορους τομείς της καθημερινότητας, καθώς οι δυνατότητες για δημιουργία νέων και «έξυπνων» εφαρμογών μπορεί να θεωρηθεί ότι είναι άπειρες, όπως στα «έξυπνα» σπίτια, στην υγεία, στην αγροτική παραγωγή και γεωργία, στις μεταφορές και στις μετακινήσεις, φέροντας τον σχετικό ηλεκτρονικό εξοπλισμό δίνουν την δυνατότητα να αντιλαμβάνονται τιμές του περιβάλλοντος μέσω αισθητήρων που διαθέτουν, να έχουν δικτυακές δυνατότητες μέσω σχετικού ηλεκτρονικού εξοπλισμού και να μπορούν να επικοινωνούν με άλλες συσκευές του χώρου χρησιμοποιώντας υπάρχουσες τεχνολογίες δικτύωσης **[2]**. Βασικό χαρακτηριστικό ενός τέτοιου δικτύου είναι ότι επιτρέπει την παρακολούθηση συσκευών είτε από μικρή ή είτε από μεγάλη απόσταση μέσα από το υπάρχον διαδικτυακό σύστημα δίνοντας έτσι τη δυνατότητα για γρηγορότερη και αποτελεσματικότερη ένωση του πραγματικού κόσμου, με τον κόσμο των υπολογιστών. Με αυτό τον τρόπο το ΔτΠ αποτελεί μια πλατφόρμα μέσω της οποίας είναι δυνατή η επικοινωνία μεταξύ μεγάλου εύρους συσκευών.

2.2.1 Μοντέλα επικοινωνίας στο IoT

Τον Μάρτιο του 2015 το Συμβούλιο Αρχιτεκτονικής του Διαδικτύου (Internet Architecture Board, IAB) κυκλοφόρησε ένα κατευθυντήριο αρχιτεκτονικό έγγραφο για τη δικτύωση των «έξυπνων» αντικειμένων, που περιγράφει ένα πλαίσιο τεσσάρων κοινών μοντέλων επικοινωνίας που χρησιμοποιείται από συσκευές IoT **[3]**.

Device-to-Device

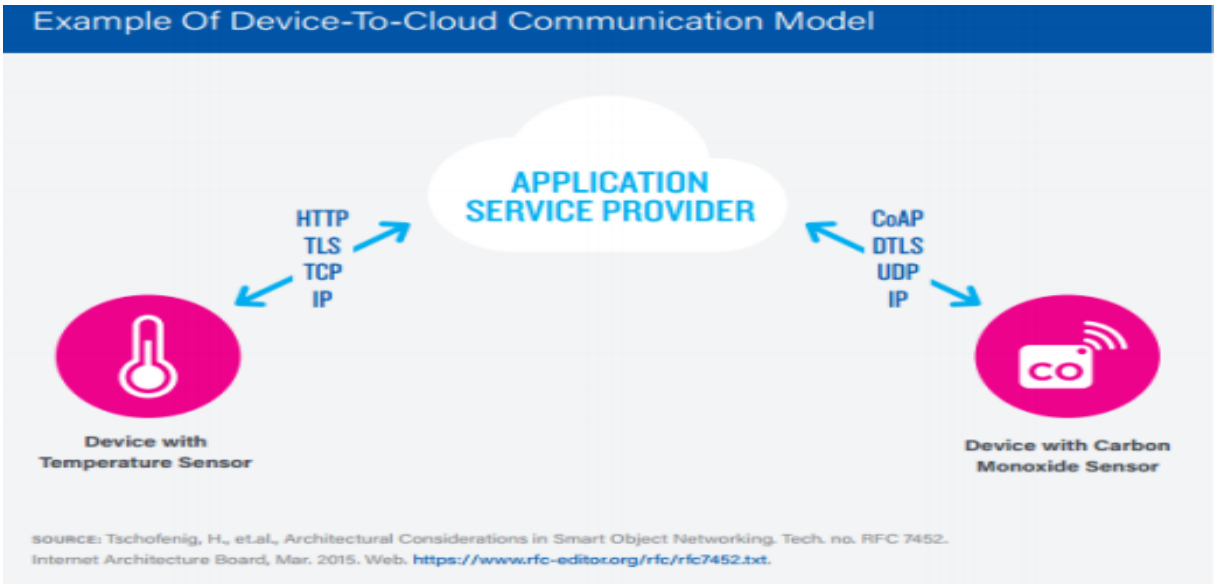
Το μοντέλο επικοινωνίας device-to-device αντιπροσωπεύει δύο ή περισσότερες συσκευές που συνδέονται άμεσα και επικοινωνούν μεταξύ τους, χωρίς τη χρήση ενός ενδιάμεσου server. Το μοντέλο αυτό επιτρέπει σε συσκευές που συμμορφώνονται με ένα συγκεκριμένο πρωτόκολλο επικοινωνίας να επικοινωνούν και να ανταλλάσσουν μηνύματα μεταξύ τους. Οι συσκευές χρησιμοποιούν πρωτόκολλα όπως το Bluetooth, Z-Wave, ή ZigBee για την μεταξύ τους επικοινωνία. Χρησιμοποιείται κυρίως σε εφαρμογές που στέλνονται μικρά πακέτα δεδομένων πληροφορίας μεταξύ των συσκευών, όπως σε συστήματα οικιακών αυτοματισμών (λάμπες, διακόπτες κ.α.).



Εικόνα 2: Μοντέλο Device to Device

Device-to-Cloud

Στο μοντέλο επικοινωνίας device-to-cloud, η συσκευή συνδέεται απευθείας σε μια διαδικτυακή υπηρεσία cloud, για την ανταλλαγή δεδομένων και τη διαχείριση της κίνησης των μηνυμάτων. Αυτή η προσέγγιση χρησιμοποιεί τους ήδη υπάρχοντες μηχανισμούς επικοινωνίας, όπως το ενσύρματο Ethernet ή το ασύρματο WiFi, ώστε να δημιουργηθεί μια σύνδεση μεταξύ της συσκευής και του δικτύου IP, το οποίο τελικά θα συνδεθεί με την cloud υπηρεσία. Το μοντέλο αυτό χρησιμοποιείται από κάποιες συσκευές οι οποίες συνδέονται στο διαδίκτυο και μεταδίδουν δεδομένα σε μια cloud υπηρεσία για ανάλυση, επεξεργασία ή αποθήκευση. Επίσης παρέχουν στον χρήστη απομακρυσμένο έλεγχο της συσκευής μέσω κάποιου smart phone, tablet ή web site.

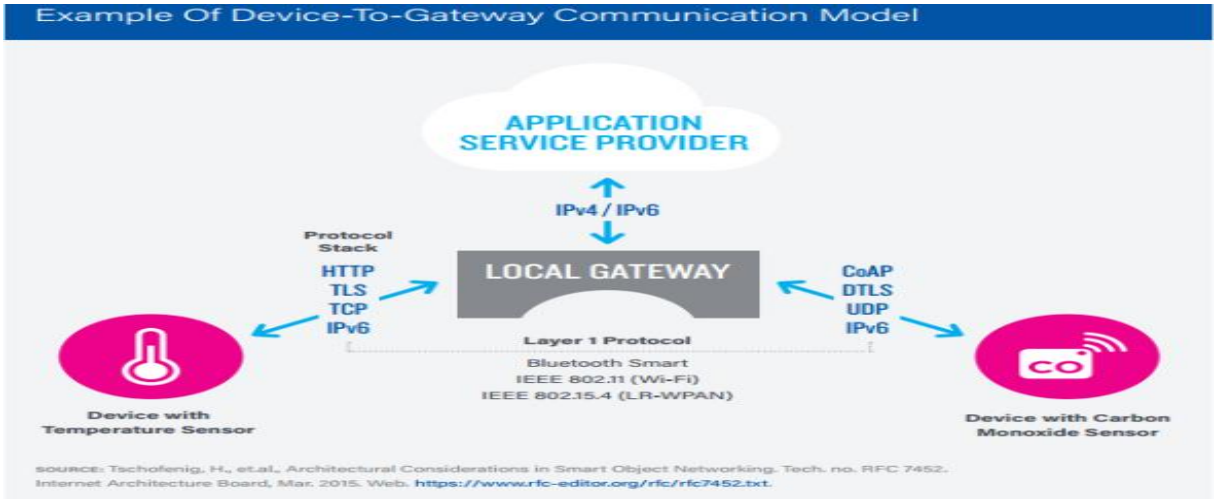


Εικόνα 3: Μοντέλο Device to Cloud

Device-to-Gateway

Στο μοντέλο device-to-gateway η συσκευή συνδέεται με μια cloud υπηρεσία μέσω ενός διαύλου (gateway). Μια gateway συσκευή μπορεί να απορρίπτει, να αθροίζει και να ελέγχει τη μορφή των

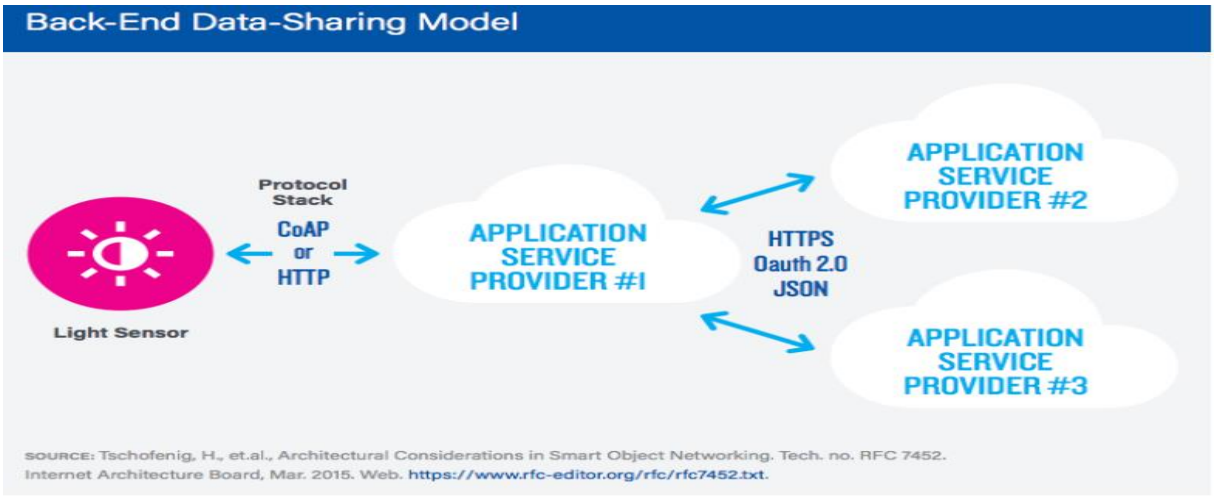
δεδομένων από μια ομάδα απλών αισθητήρων δρώντας ουσιαστικά ως μεσολαβητής πριν τα στείλει κάπου αλλού. Επομένως αποτελεί έναν μεσολαβητή μεταξύ της IoT συσκευής και της cloud υπηρεσίας παρέχοντας ασφάλεια. Μια μορφή του μοντέλου device-to-gateway που συναντάται καθημερινά σε καταναλωτικές συσκευές είναι τα smart phones τα οποία λειτουργούν ως gateways ώστε συσκευές που από μόνες τους δεν έχουν πρόσβαση στο διαδίκτυο να έχουν την δυνατότητα να συνδεθούν σε μια cloud υπηρεσία.



Εικόνα 4: Μοντέλο Device to Gateway

Back-End-Data-Sharing

Το μοντέλο αυτό αναφέρεται στην αρχιτεκτονική που επιτρέπει στους χρήστες να εξάγουν και να αναλύσουν τα δεδομένα που προέρχονται από μια cloud υπηρεσία σε συνδυασμό με δεδομένα από άλλες πηγές. Το μοντέλο αυτό αποτελεί μια προέκταση του μοντέλου επικοινωνίας device-to-cloud, στο οποίο οι IoT συσκευές ανεβάζουν τα δεδομένα μόνο σε έναν πάροχο υπηρεσιών. Το μοντέλο back-end-data-sharing υποδηλώνει ότι χρειάζεται μια πιο συνολική προσέγγιση των cloud υπηρεσιών για την επίτευξη της χρήσης των δεδομένων των συσκευών που φιλοξενούνται στο cloud.



Εικόνα 5: Μοντέλο Back End Data Sharing

2.2.2 Ορισμοί του Internet of Things

Παρά το μεγάλο παγκόσμιο ενδιαφέρον γύρω από το ΔτΠ δεν υπάρχει ένας διεθνώς αποδεκτός ορισμός για τον όρο. Οι διαφορετικοί ορισμοί χρησιμοποιούνται από διάφορες ομάδες έτσι ώστε να περιγράψουν μια συγκεκριμένη άποψη για το τι σημαίνει IoT και ποια είναι τα σημαντικά χαρακτηριστικά του. Μερικούς από τους πιο διαδεδομένους ορισμούς είναι:

- Internet Architecture Board (IAB)

«Ο όρος 'Διαδίκτυο των πραγμάτων' (IoT) υποδηλώνει μια τάση όπου ένας μεγάλος αριθμός ενσωματωμένων συσκευών χρησιμοποιούν τις υπηρεσίες επικοινωνίας που προσφέρονται από τα πρωτόκολλα του Διαδικτύου. Πολλές από αυτές τις συσκευές, που συχνά ονομάζονται 'έξυπνα αντικείμενα', δεν λειτουργούν άμεσα από τον άνθρωπο, αλλά υπάρχουν ως συνιστώσες σε κτίρια ή οχήματα ή είναι διασκορπισμένα στο περιβάλλον» **[4]**.

- International Telecommunication Union (ITU)

«Διαδίκτυο των πραγμάτων (IoT): Μια παγκόσμια υποδομή για την κοινωνία της πληροφορίας που επιτρέπει προηγμένες υπηρεσίες μέσω της διασύνδεσης (φυσικών και εικονικών) πραγμάτων με βάση της υφιστάμενη και την σε εξέλιξη διαλειτουργικότητα των τεχνολογιών της πληροφορίας και της επικοινωνίας.

Σημείωση 1- Μέσω της αξιοποίησης των δυνατοτήτων αναγνώρισης, καταγραφής δεδομένων, την επεξεργασία και την επικοινωνία, το IoT κάνει πλήρη χρήση των πραγμάτων έτσι ώστε να προσφέρουν υπηρεσίες σε όλα τα είδη των εφαρμογών, εξασφαλίζοντας παράλληλα ότι οι απαιτήσεις της ασφάλειας και της προστασίας της ιδιωτικής ζωής πληρούνται.

Σημείωση 2- Από μια ευρύτερη προοπτική, το IoT μπορεί να γίνει αντιληπτό ως ένα όραμα με τεχνολογίες και κοινωνικές επιπτώσεις» **[5]**.

- IEE Communication Magazine

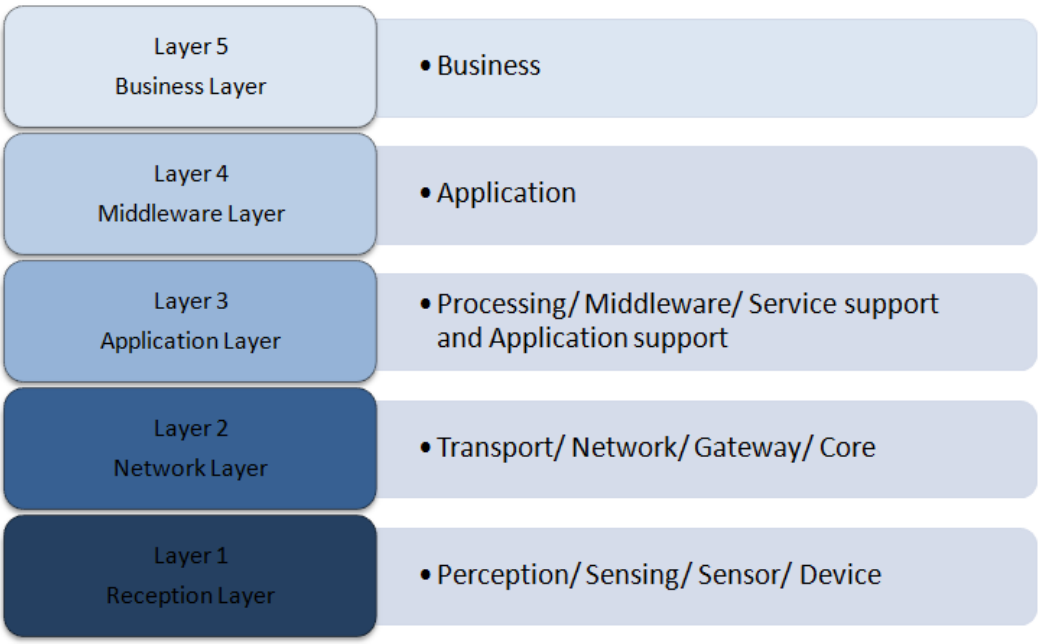
«Το Διαδίκτυο των πραγμάτων (IoT) αποτελεί ένα πλαίσιο στο οποίο όλα τα πράγματα έχουν μια αντιπροσώπευση και μια παρουσία στο Internet. Πιο συγκεκριμένα το Διαδίκτυο των Πραγμάτων έχει ως στόχο να προσφέρει νέες εφαρμογές και υπηρεσίες γεφυρώνοντας τον φυσικό και τον εικονικό κόσμο, στον οποίο οι Machine-to-Machine (M2M) επικοινωνίες αντιπροσωπεύουν την βασική επικοινωνία που επιτρέπει τις αλληλεπιδράσεις μεταξύ των πραγμάτων και των εφαρμογών στο cloud» **[6]**.

- Wikipedia

«Το Διαδίκτυο των πραγμάτων (IoT) είναι το δίκτυο των φυσικών αντικειμένων ή 'πραγμάτων', ενσωματωμένο με ηλεκτρονικά, λογισμικό, αισθητήρες και συνδεσιμότητα έτσι ώστε να μπορέσει να επιτευχθεί η μεγαλύτερη αξία και η καλύτερη εξυπηρέτηση μέσω ανταλλαγής δεδομένων μεταξύ του κατασκευαστή, του χειριστή ή και άλλων συνδεδεμένων συσκευών. Κάθε πράγμα είναι μοναδικά αναγνωρίσιμο μέσω του ενσωματωμένου συστήματος, αλλά είναι σε θέση να συνεργάζεται με την υφιστάμενη υποδομή του Διαδικτύου» **[7]**.

2.2.3 Αρχιτεκτονική Internet of Things

Η βασικότερη αρχιτεκτονική είναι η αρχιτεκτονική τριών επιπέδων και αποτελείται από τα επίπεδα αντίληψης, δικτύου και εφαρμογής και προστίθενται δύο επιπλέον επίπεδα, ενδιάμεσου λογισμικού και επιχείρησης, δημιουργώντας την αρχιτεκτονική πέντε επιπέδων ώστε το IoT να εστιάσει σε πιο λεπτομερή επίπεδα **[8]**.



Εικόνα 6: Αρχιτεκτονική επιπέδου του IoT

- Perception Layer (Επίπεδο Αντίληψης)

Το επίπεδο αυτό περιλαμβάνει όλα τα αντικείμενα όπως αισθητήρες, κάμερες και ενεργοποιητές που συλλέγουν και συγκεντρώνουν την απαραίτητη πληροφορία.

- Network Layer (Επίπεδο Δικτύου)

Η λειτουργία του επιπέδου αυτού είναι η ορθή και γρήγορη επικοινωνία με άλλα αντικείμενα, δικτυακές συσκευές και εξυπηρετητές για την μετάδοση, αποθήκευση και επεξεργασία μεγάλου αριθμού δεδομένων σε ένα δίκτυο. Στο επίπεδο Δικτύου περιλαμβάνονται τεχνολογίες δικτύων, όπως ασύρματα ή ενσύρματα δίκτυα και τοπικά δίκτυα (LAN).

- Application Layer (Επίπεδο Εφαρμογής)

Αυτό το επίπεδο είναι υπεύθυνο για την επεξεργασία των δεδομένων μέσω διάφορων εφαρμογών που έχουν ληφθεί από το επίπεδο δικτύου. Ο χρήστης έχει την δυνατότητα να έρθει σε επαφή με τα «έξυπνα» αντικείμενα που παρέχει το επίπεδο αντίληψης.

- Middleware Layer (Επίπεδο Ενδιάμεσου Λογισμικού)

Με το επίπεδο αυτό συνδέεται το επίπεδο δικτύου με το επίπεδο εφαρμογής καθώς αποτελεί το λογισμικό, το οποίο δίνει την δυνατότητα στις εφαρμογές να έχουν πρόσβαση στα δεδομένα που λαμβάνουν τα αντικείμενα.

- Business Layer (Επίπεδο Επιχείρησης)

Το επίπεδο αυτό έχει να κάνει με την οικονομική διαχείριση των παρεχόμενων υπηρεσιών καθώς οι επιχειρήσεις παρέχουν πληροφορίες και δεδομένα μέσω των εφαρμογών τους με σκοπό τα χρηματικά κέρδη.

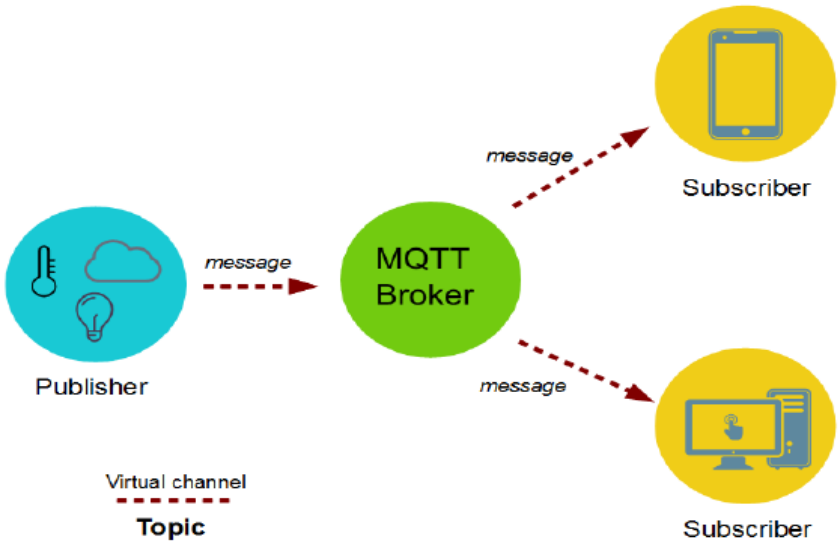
2.3 Message Queuing Telemetry Transport (MQTT)

Το πρωτόκολλο MQTT εφευρέθηκε το 1999 από τους Dr.Andy Stanfor-Clark των IBM και Arlen Nipper της Arcom **[9]**. Ο στόχος ήταν να δημιουργηθεί ένα πρωτόκολλο επικοινωνίας χαμηλών απαιτήσεων σε ενέργεια και δικτυακό εύρος (bandwidth) ώστε να παρακολουθείται και να

συντονίζεται η λειτουργία αγωγών πετρελαίου δια μέσου δορυφόρου. Τα αρχικά MQTT προέρχονται από τα αρχικά των λέξεων MQ Telemetry Transport. Το MQ προέρχεται από το MQ Series, ένα προϊόν που αναπτύχθηκε από την IBM και υποστηρίζει το MQTT.

Το MQTT είναι ένα απλό, ελαφρύ και εύχρηστο στην εφαρμογή, πρωτόκολλο ανταλλαγής μηνυμάτων για M2M επικοινωνίες. Ουσιαστικά πρόκειται για ένα ελαφρύ σύστημα «εγγραφής» (subscribe) και «δημοσίευσης» (publish) στο οποίο οι χρήστες μπορούν να στείλουν και να λάβουν μηνύματα ως client. Είναι βασισμένο στην αρχιτεκτονική publish-subscribe, η οποία διευκολύνει σε μεγάλο βαθμό το σχεδιασμό IoT εφαρμογών. Οι αρχές σχεδίασης που ακολουθεί το πρωτόκολλο είναι η ελαχιστοποίηση του εύρους ζώνης (bandwidth) του δικτύου, καθώς και των απαιτούμενων πόρων (ενέργειας, μνήμης). Αυτό οφείλετε στο γεγονός ότι διαθέτει κεφαλίδα πακέτου σταθερή και ίση με δύο bytes. Η υψηλή αποτελεσματικότητά του όσον αφορά στο εύρος ζώνης και της χαμηλής κατανάλωσης ισχύος το καθιστούν ως το κύριο προτεινόμενο πρωτόκολλο για εφαρμογές IoT. Λειτουργεί πάνω από το TCP/IP επίπεδο με προεπιλεγμένη θύρα τη 1883 παρέχοντας αξιόπιστες ροές πληροφορίας και εξασφαλίζοντας την μεταφορά των μηνυμάτων από τον MQTT Client στον MQTT Server (αρχιτεκτονική client-server).

Το MQTT είναι ένα ασύγχρονο πρωτόκολλο, αποτελείται από τρία βασικά στοιχεία ή μηχανισμούς: τον Subscriber, τον Publisher και τον Broker και ζευγνύει έναν client που δημοσιεύει ένα μήνυμα (publisher) σε άλλους clients που λαμβάνουν το μήνυμα (subscribers). Με τον όρο MQTT client συμπεριλαμβάνεται τόσο ο publisher όσο και ο subscriber. Στην γενική περίπτωση ένας MQTT client μπορεί να είναι και τα δύο. Το έργο του Publisher είναι η γένεση και η μετάδοση δεδομένων στον Subscriber με τη βοήθεια του Broker. Η εξασφάλιση της ασφάλειας αποτελεί το έργο του Broker και την επιτυγχάνει με τον έλεγχο και τον επανέλεγχο της εξουσιοδότησης των Subscribers και των Publishers. Δηλαδή ο broker λαμβάνει μηνύματα από τους Publisher και τα στέλνει στους Subscribers αφού φιλτράρει τους clients με βάση το topic του μηνύματος. Το topic είναι ένα αλφαριθμητικό που ορίζει σε ποια θεματική θα δημοσιευθεί ένα μήνυμα. Ο broker φροντίζει όσοι clients είναι εγγεγραμμένοι στο παρόν topic να λάβουν το μήνυμα επιτυχώς **[10]**.



Εικόνα 7: Message Queuing Telemetry Transport

2.3.1 Επίπεδα Quality of Service (QoS)

Το επίπεδο QoS αποτελεί ουσιαστικά μια συμφωνία ανάμεσα στον αποστολέα και τον παραλήπτη για την εγγυημένη παράδοση ενός μηνύματος. Η διαδρομή του μηνύματος από τον publisher στο subscriber μπορεί να εξεταστεί από δύο πλευρές. Δηλαδή το QoS που αφορά την αποστολή μηνύματος από τον Publisher στον broker, όπως έχει καθοριστεί από την εντολή publish και το QoS για την αποστολή μηνύματος από τον broker στο subscriber, όπως έχει καθοριστεί από την εντολή subscribe. Το MQTT διαθέτει 3 QoS επίπεδα και κατηγοριοποιούνται ως εξής **[11]**:

- QoS=0

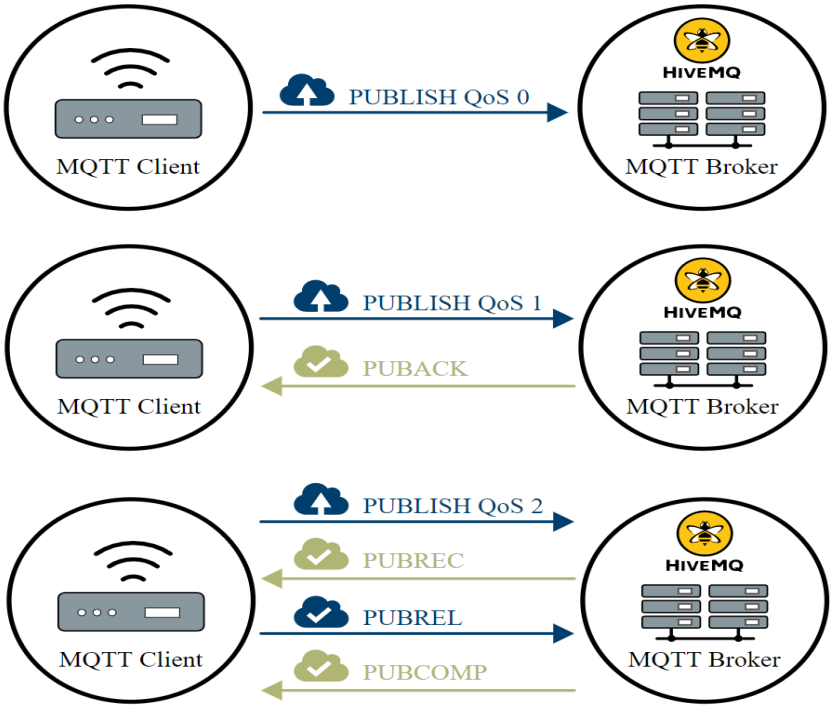
At most once (το πολύ μία φορά): Το μήνυμα στέλνεται μόνο μία φορά και ο client και ο broker δεν λαμβάνουν άλλα μέτρα για την αναγνώριση της παράδοσης του μηνύματος ή της αποστολής (fire and forget). Είναι αξιόπιστο τόσο όσο είναι το TCP πρωτόκολλο πάνω από το οποίο πραγματοποιείται η αποστολή και επιλέγεται όταν υπάρχει σταθερή σύνδεση broker και client ή όταν δεν υπάρχει πρόβλημα να χαθεί λίγη πληροφορία.

- QoS=1

At least once (τουλάχιστον μία φορά): Το μήνυμα στέλνεται από τον αποστολέα πολλαπλές φορές μέχρι να ληφθεί η αναγνώριση παράδοσης (PUBACK). Αν περάσει συγκεκριμένο χρονικό διάστημα και δεν το λάβει ο δέκτης τότε ο αποστολέας προσπαθεί να αποστείλει το μήνυμα ξανά και έτσι μπορεί να φτάσει περισσότερες από μία φορές στον δέκτη.

- QoS=2

Exactly once (ακριβώς μία φορά): Ο παραλήπτης λαμβάνει ένα και μόνο ένα μήνυμα από τον αποστολέα και στέλνει επιβεβαίωση. Είναι ο ασφαλέστερος αλλά και ο πιο αργός τρόπος να σταλεί ένα μήνυμα.



Εικόνα 8: Επίπεδα Quality of Service

2.3.2 Μοντέλο Publish-Subscribe στο MQTT

Η τεχνική Publish-Subscribe αποτελεί ένα πρότυπο αποστολής μηνυμάτων το οποίο έχει τρεις ρόλους **[12]**:

Publishers

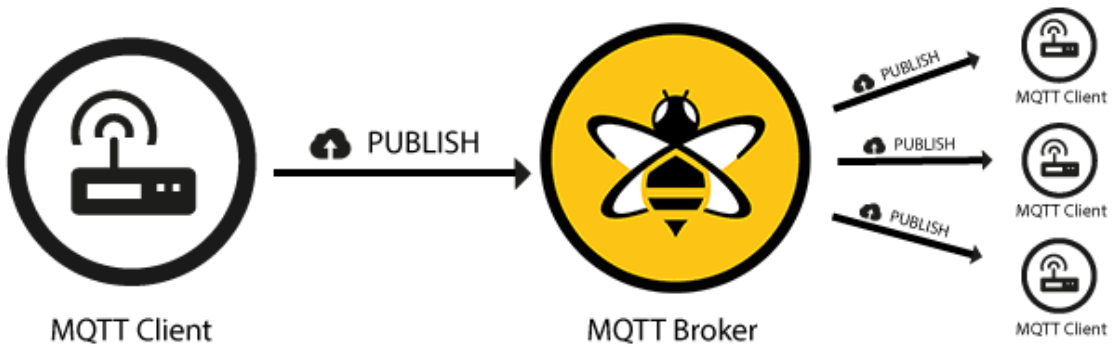
Οι Publishers αποτελούν αυτούς που αποστέλλουν τα μηνύματα. Κατά την αποστολή μηνυμάτων δεν καθορίζουν ποιος θα είναι ο παραλήπτης του μηνύματος, αλλά το μήνυμα αποστέλλεται σε κάποιο θέμα (topic) στον Broker.

Subscribers

Οι Subscribers αποτελούν τους παραλήπτες των μηνυμάτων. Ένας subscriber δηλώνει στον Broker το ενδιαφέρον του σε ένα συγκεκριμένο θέμα. Οποιοδήποτε μήνυμα σταλεί στο συγκεκριμένο θέμα, αποστέλλεται αυτόματα και στους subscribers του θέματος.

Broker

Ο Broker στην ουσία αποτελεί τον ενδιάμεσο κόμβο που συνδέει όλους τους χρήστες (publishers και subscribers). Είναι ένας δρομολογητής ο οποίος είναι υπεύθυνος να επιτρέπει την εγγραφή των subscribers σε θέματα και να δρομολογεί τα εισερχόμενα γεγονότα από τους publishers στους subscribers ανάλογα με το θέμα.



Εικόνα 9: Μοντέλο Publish-Subscribe

2.3.3 Ασφάλεια στο πρωτόκολλο MQTT

Το πρωτόκολλο MQTT μπορεί να χρησιμοποιηθεί και σε μη ασφαλή περιβάλλοντα και τα πακέτα που μεταδίδονται να τύχουν κακόβουλης μεταχείρισης. Αυτό οδηγεί στην ύπαρξη κάποιου μηχανισμού που θα αυθεντικοποιεί, θα εξουσιοδοτεί και θα κρυπτογραφεί **[13]**.

Αυθεντικοποίηση (authentication)

Το MQTT πρωτόκολλο παρέχει τα πεδία username και password για πιστοποίηση κατά την σύνδεση στον broker. Τα στοιχεία στέλνονται σε απλό κείμενο χωρίς κάποιας μορφής κρυπτογράφησης με αποτέλεσμα με οποιαδήποτε κλοπή των δεδομένων κατά την μεταφορά τους, τα στοιχεία μπορούν να καταγραφούν και να χρησιμοποιηθούν.

Εξουσιοδότηση

Η εξουσιοδότηση και η αυθεντικοποίηση είναι δύο αλληλένδετα στοιχεία καθώς δεν έχει νόημα να επιτραπεί μια ενέργεια (εξουσιοδότηση) αν δεν έχει προηγηθεί έλεγχος της αυθεντικοποίησης. Σε συστήματα επικοινωνίας τα οποία χρησιμοποιούν το MQTT πρωτόκολλο η εξουσιοδότηση μπορεί να φανεί πολύ χρήσιμη. Η εξουσιοδότηση που δίνεται στον client αφορά τα δικαιώματά του στα topics και θα πρέπει να μπορεί να ρυθμιστεί. Όπως το επίπεδο QoS που έχει δικαίωμα να χρησιμοποιήσει.

Κρυπτογράφηση

Τα TCP πακέτα περνούν από διάφορες υποδομές (routers, firewalls) πριν φτάσουν στον προορισμό τους. Έτσι τα δεδομένα μπορούν να τύχουν κακόβουλης επεξεργασίας από κάθε σημείο. Για τον λόγο αυτό υπάρχουν τα πρωτόκολλα TLS (Transport Layer Security) και SSL (Secure Sockets Layer) που χρησιμοποιούν μηχανισμούς χειραψίας ώστε να ρυθμίσουν κατάλληλες παραμέτρους για να δημιουργήσουν μια ασφαλή σύνδεση μεταξύ client και server.

2.4 Επίλογος

Το παρόν κεφάλαιο αναφέρεται στην έννοια Διαδίκτυο των Πραγμάτων και διατυπώνονται σχετικές πληροφορίες ώστε να γίνει κατανοητή η νέα αυτή τεχνολογία καθώς και οι δυνατότητες που παρέχει στον σύγχρονο κόσμο της τεχνολογίας. Σκοπός του κεφαλαίου αυτού ήταν ο αναγνώστης να λάβει μία «εικόνα» σχετικά με το ΔτΠ μέσα από τους ορισμούς, τα τέσσερα μοντέλα επικοινωνίας που χρησιμοποιούνται από συσκευές IoT και την αρχιτεκτονική του IoT. Επίσης παρατίθενται θεωρητικά στοιχεία σχετικά με το πρωτόκολλο επικοινωνίας MQTT όσο αφορά τα QoS επίπεδα, το μοντέλο publish/subscribe αλλά και τα ζητήματα ασφαλείας του.

Κεφάλαιο 3ο: Cloud Computing και Amazon EC2

3.1 Εισαγωγή

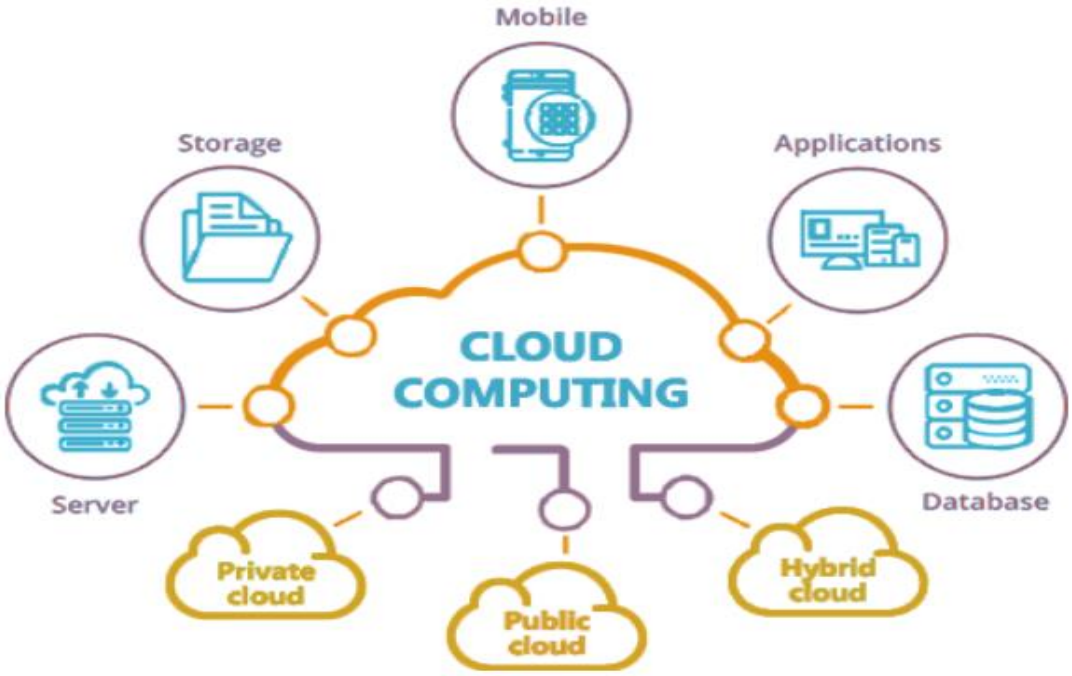
Τον τελευταίο καιρό οι δυνατότητες που προσφέρουν οι νέες τεχνολογίες πληροφοριών με σχετικά μικρό κόστος είναι αρκετά διαδεδομένες ως εκ τούτου γίνεται όλο και περισσότερο αναφορά για το cloud computing καθώς αποτελεί μία από αυτές τις τεχνολογίες. Η τεχνολογία cloud computing, ή αλλιώς υπολογιστικό νέφος αποτελεί ένα νέο κομμάτι της πληροφορικής, το οποίο παρέχει επιπλέον προοπτικές σε τεχνολογίες δικτύωσης. Δίνει νέες δυνατότητες ως προς την αρχιτεκτονική, τον σχεδιασμό και την υλοποίηση των υπαρχόντων δικτύων και κέντρων δεδομένων. Μια τέτοια υπηρεσία αποτελεί και το Amazon Elastic Cloud. Το Amazon Elastic Cloud είναι μια διαδικτυακή υπηρεσία η οποία αποτελεί ένα κεντρικό κόμβο της cloud computing πλατφόρμας της εταιρίας Amazon, και παρέχει την δυνατότητα χρησιμοποίησης υπολογιστικής δύναμης εφαρμόζοντας την τεχνική των εικονικών εξυπηρετητών.

3.2 Cloud Computing

Πιο συγκεκριμένα το νέφος υπολογιστών είναι ένα μοντέλο το οποίο παρέχει εύκολη πρόσβαση σε μια πληθώρα από υπολογιστικούς πόρους μέσω των οποίων διατίθεται υπηρεσίες διαμοιρασμού υπολογιστικών πόρων (δίκτυο, υπολογιστές, αποθηκευτικός χώρος, περιβάλλοντα ανάπτυξης εφαρμογών, εφαρμογές και υπηρεσίες). Ο χρήστης έχει την δυνατότητα χρήσης, ανάπτυξης λογισμικού, πρόσβασης σε πληροφορίες και αποθήκευσης δεδομένων τα οποία βρίσκονται σε ένα cloud και μπορούν να δεσμευτούν και να απελευθερωθούν γρήγορα με την ελάχιστη δυνατή

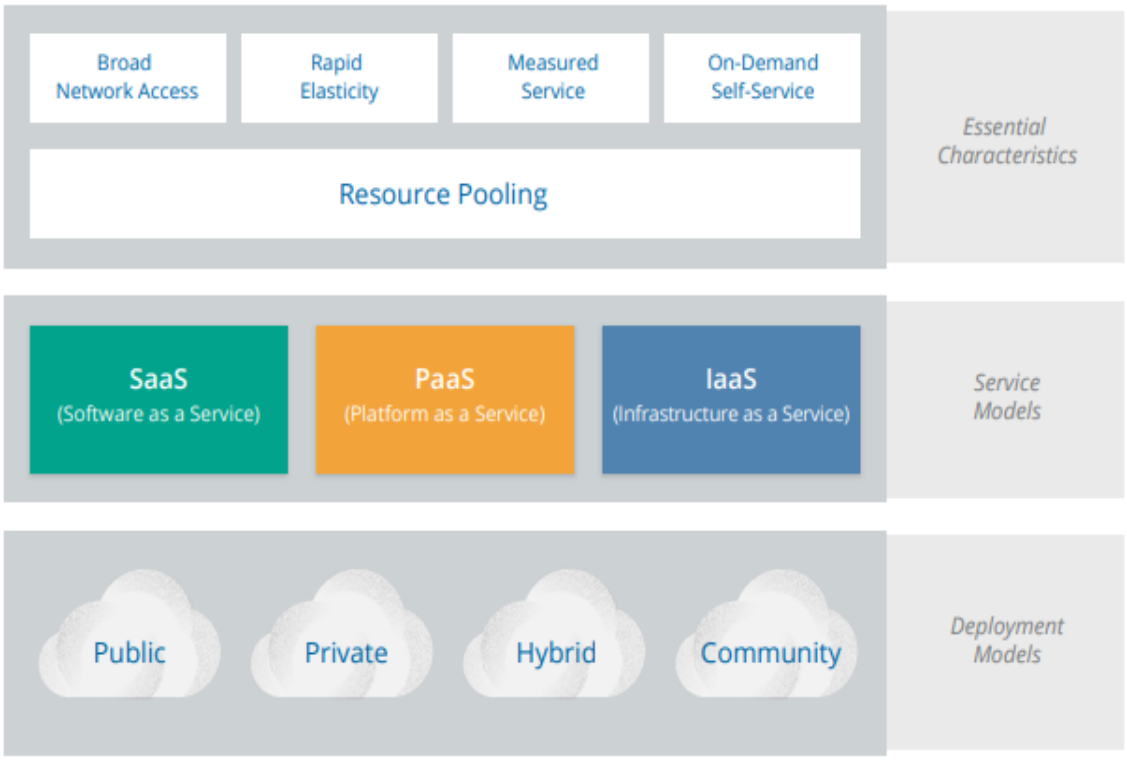
προσπάθεια και αλληλεπίδραση **[14]**. Επίσης αυτοί οι υπολογιστικοί πόροι που παρέχονται μέσω του cloud computing, μπορούν να χρησιμοποιηθούν ώστε να εξυπηρετήσουν πολλούς καταναλωτές και να καλύψουν τις απαιτήσεις των καταναλωτών.

Ουσιαστικά οι ενδιαφερόμενοι χρήστες cloud ζητούν την πρόσβαση από ένα σύνολο υπηρεσιών που είναι διαθέσιμοι μέσω του διαδικτύου και είναι προσβάσιμοι μέσω τυποποιημένων μηχανισμών. Η χρήση των διαθέσιμων υπολογιστικών πόρων δεν απαιτεί την ανθρώπινη αλληλεπίδραση με το πάροχο της κάθε υπηρεσίας. Στην περίπτωση που αποδοθεί ένα τμήμα πόρων σε κάποιον χρήστη cloud, το τμήμα αυτό είναι δεσμευμένο σε αυτόν τον χρήστη μέχρι αυτός να το απελευθερώσει. Οι υπολογιστικοί αυτοί πόροι βρίσκονται σε ένα cloud και δεσμεύονται ανά πάσα στιγμή **[15]** και σε οποιαδήποτε ποσότητα από τον εκάστοτε χρήστη όταν το ζητήσει, έπειτα επιστρέφουν στο cloud όταν αποδεσμευτούν από τον χρήστη και οι υπολογιστικοί πόροι είναι γρήγορα και πάλι διαθέσιμοι να χρησιμοποιηθούν από κάποιον άλλον ενδιαφερόμενο χρήστη. Ο χρήστης στην πραγματικότητα δεν μπορεί να προσδιορίσει και να καταλάβει που βρίσκονται ακριβώς οι υποδομές που χρησιμοποιεί ή υπηρεσίες που έχει δεσμεύσει και έχει ζητήσει για να χρησιμοποιήσει.



Εικόνα 10: Cloud Computing **[16]**

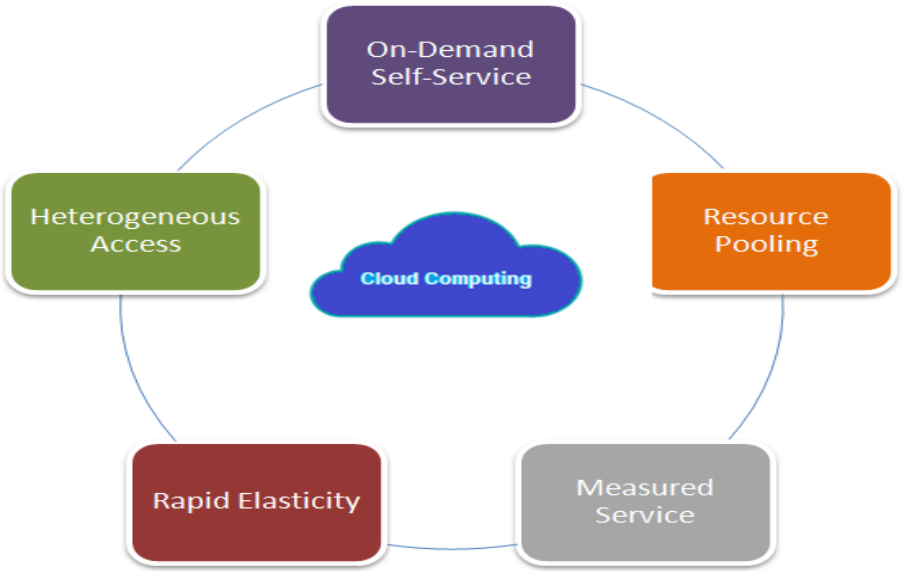
Στην προσπάθεια να δοθεί ορισμός για το cloud computing, το Εθνικό Ινστιτούτο Τυποποιήσεων και Τεχνολογίας (NIST-National Institute of Standards Technology) **[17]** διατύπωσε ότι το cloud computing είναι ένα μοντέλο που επιτρέπει ευέλικτη, κατά απαίτηση (On-Demand) δικτυακή πρόσβαση σε ένα κοινόχρηστο σύνολο υπολογιστικών πόρων, το οποίο μπορεί να παρέχεται γρήγορα και να είναι διαθέσιμο με την ελάχιστη προσπάθεια διαχείρισης και αλληλεπίδρασης από τον πάροχο της υπηρεσίας. Αυτό το cloud μοντέλο αποτελεί σύνθεση πέντε σημαντικών χαρακτηριστικών, τριών μοντέλων παροχής υπηρεσιών, καθώς και τεσσάρων μοντέλων ανάπτυξης **[18]**.



Εικόνα 11: Απεικόνιση ορισμού NIST για την αρχιτεκτονική του Cloud Computing

3.2.1 Χαρακτηριστικά του Cloud Computing

Όπως αναφέρθηκε στην ενότητα 3.2 υπάρχουν πέντε σημαντικά χαρακτηριστικά τα οποία διακρίνουν τις υπηρεσίες νέφους συγκριτικά με τις απλές υπολογιστικές μεθόδους **[19]**.



Εικόνα 12: Χαρακτηριστικά του Cloud Computing

Το πρώτο χαρακτηριστικό είναι η αυτό-εξυπηρέτηση κατά απαίτηση (On-Demand Self-Service). Ο χρήστης έχει την δυνατότητα να δέχεται ή να απορρίπτει την παροχή υπηρεσιών (χρόνος δέσμευσης

του server, μέγεθος αποθηκευτικού χώρου) άμεσα και χωρίς καθυστέρηση όταν το επιθυμεί χωρίς να απαιτείται η ανθρώπινη παρέμβαση με τον πάροχο της εκάστοτε υπηρεσίας.

Το δεύτερο χαρακτηριστικό είναι η ευρεία πρόσβαση στο δίκτυο (Heterogeneous Access) όπου οι δυνατότητες παρέχονται σε όλο το δίκτυο και η πρόσβαση μπορεί να γίνει μέσω τυποποιημένων μηχανισμών και με οποιαδήποτε συνδεδεμένη συσκευή.

Το τρίτο χαρακτηριστικό είναι η διάθεση πόρων (Resource Pooling). Οι υπολογιστικοί πόροι του παρόχου συνδυάζουν δυναμικά φυσικούς και εικονικούς πόρους όπως υπολογιστικό χρόνο και αποθηκευτικό χώρο ώστε να μοιράζονται σε πολλούς χρήστες ανταποκρινόμενοι στην ζήτηση των χρηστών.

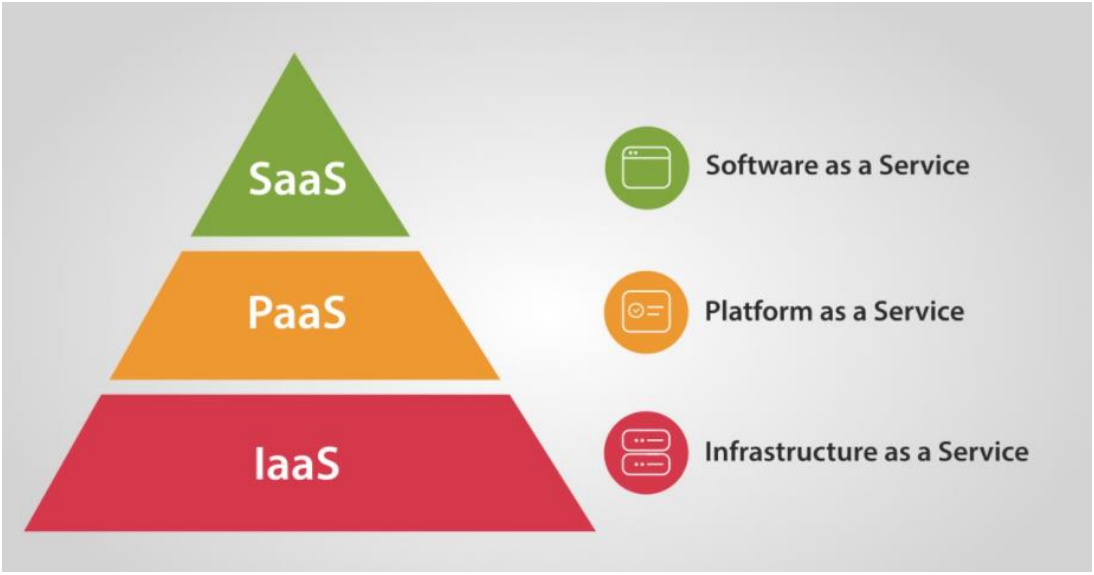
Το τέταρτο χαρακτηριστικό είναι η ταχεία ελαστικότητα (Rapid Elasticity) όπου η υπηρεσία μπορεί να παρέχεται γρήγορα, ελαστικά και αυτόματα χωρίς πρόβλημα ακόμα και σε περιόδους αυξημένου φόρτου.

Το πέμπτο και τελευταίο χαρακτηριστικό είναι η μετρούμενη υπηρεσία (Measured Service) στο οποίο τα συστήματα cloud computing καταγράφουν τη χρήση των πόρων ώστε να οργανώνουν και να βελτιστοποιούν αυτόματα τη διάθεση των πόρων και ανάλογα με τη χρήση να πραγματοποιείται η χρέωση.

3.2.2 Μοντέλα παροχής υπηρεσίας του Cloud Computing

Τα τρία μοντέλα παροχής υπηρεσίας νέφους σύμφωνα και με την εικόνα 13 είναι **[20]**,**[21]**:

- το λογισμικό νέφους ως υπηρεσία (Software as a Service, SaaS)
- η πλατφόρμα νέφους ως υπηρεσία (Platform as a Service, PaaS)
- η δομή νέφους ως υπηρεσία (Infrastructure as a Service, IaaS)



Εικόνα 13: Μοντέλα παροχής του Cloud Computing (www.litslink.com)

Το πρώτο μοντέλο αποτελεί το λογισμικό νέφους ως υπηρεσία (SaaS). Το SaaS είναι μια δυνατότητα που παρέχεται στους χρήστες ώστε να χρησιμοποιούν τις εφαρμογές που είναι διαθέσιμες στο νέφος. Ουσιαστικά ο χρήστης έχει πρόσβαση σε εφαρμογές οι οποίες είναι τυπικά διαθέσιμες μέσω διεπαφών διαδικτύου όπως ένας φυλλομετρητής από διάφορα ήδη συσκευών (προσωπικοί υπολογιστές, tablets,

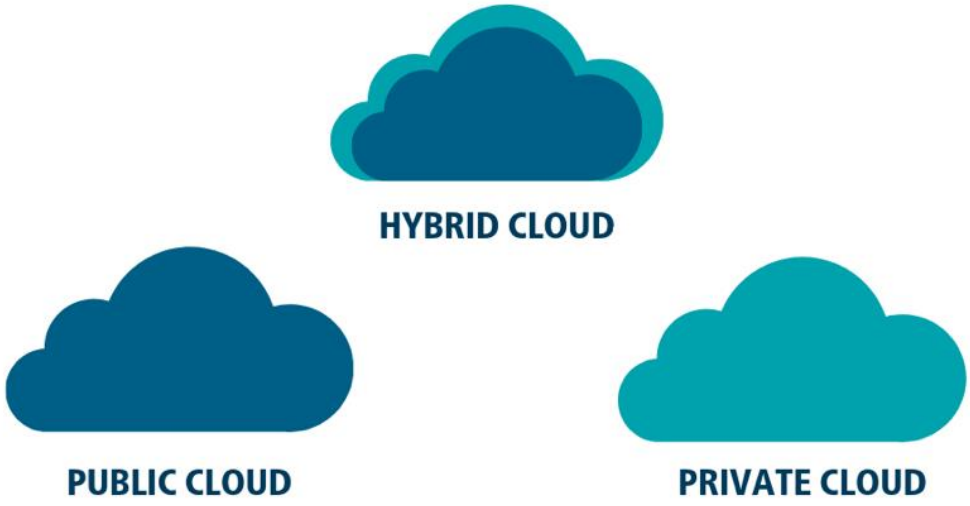
smart phones και άλλες φορητές συσκευές) μέσω μιας υποδομής cloud computing. Ο χρήστης δεν επηρεάζει ούτε έχει πρόσβαση στην υποκείμενη υποδομή του cloud computing συστήματος, δηλαδή τις δικτυακές υποδομές, τους διακομιστές, τα λειτουργικά συστήματα ή τους αποθηκευτικούς χώρους, στο οποίο τρέχουν οι εφαρμογές, αλλά μόνο στην ίδια την εφαρμογή.

Το δεύτερο μοντέλο είναι η πλατφόρμα νέφους ως υπηρεσία (PaaS). Ο χρήστης μπορεί να αναπτύξει ο ίδιος εφαρμογές χρησιμοποιώντας γλώσσες προγραμματισμού ή εργαλεία τα οποία παρέχονται από τον πάροχο της υπηρεσίας στη δομή του νέφους. Ο χρήστης δεν διαχειρίζεται ούτε έχει πρόσβαση στην υποκείμενη υποδομή του cloud computing συστήματος, δηλαδή τις δικτυακές υποδομές, τους διακομιστές, τα λειτουργικά συστήματα ή τους αποθηκευτικούς χώρους, στο οποίο τρέχουν οι εφαρμογές, αλλά έχει τον πλήρη έλεγχο των εφαρμογών που έχει αναπτύξει και έχει την δυνατότητα να διαμορφώσει τις ρυθμίσεις του περιβάλλοντος στο οποίο πρόκειται να «τρέξει» η εφαρμογή του.

Το τελευταίο μοντέλο το αποτελεί η δομή νέφους ως υπηρεσία (IaaS). Στο μοντέλο αυτό ο χρήστης μπορεί να εφοδιάζεται με λειτουργίες επεξεργασίας, αποθήκευσης, δικτύου και άλλους υπολογιστικούς πόρους στους οποίους μπορεί να αναπτύξει και να «τρέξει» οποιασδήποτε μορφής λογισμικού. Δίνεται η δυνατότητα στον χρήστη να μπορεί να ελέγχει την κατανομή των υπολογιστικών πόρων, των χώρων αποθήκευσης, τη διαμόρφωση των δικτύων, και να ρυθμίζει γενικά όλες τις θεμελιώδεις υπολογιστικές δομές μιας cloud υποδομής. Ο χρήστης έχει την δυνατότητα να αναπτύξει και να εκτελέσει το δικό του λογισμικό χωρίς περιορισμούς, έχοντας πλήρη πρόσβαση μέχρι και στο λειτουργικό σύστημα που θα εκτελέσει την εφαρμογή του. Επίσης ο χρήστης δεν διαχειρίζεται ούτε έχει πρόσβαση στην υποκείμενη υποδομή του cloud συστήματος, αλλά έχει τον πλήρη έλεγχο των λειτουργικών συστημάτων και των χώρων αποθήκευσης. Όλες αυτές οι υποδομές παρέχονται ως υπηρεσία νέφους και κοστολογούνται με βάση τη χρήση.

3.2.3 Μοντέλα ανάπτυξης του Cloud Computing

Σύμφωνα με αυτό που αναφέρθηκε στην ενότητα 3.2 υπάρχουν τέσσερα μοντέλα ανάπτυξης νέφους.



Εικόνα 14: Μοντέλα ανάπτυξης του Cloud Computing **[22]**

Το δημόσιο νέφος (public cloud), στο οποίο η δομή του είναι διαθέσιμη σε όλο το κοινό. Αποτελείται από ένα σύνολο πόρων (υπολογιστές και δίκτυα υπολογιστών) οι οποίοι βασίζονται στο πρότυπο cloud computing και είναι διαθέσιμοι μέσω διαδικτύου ενώ τις περισσότερες φορές παρέχονται από ένα πάροχο. Το μοντέλο public cloud χαρακτηρίζεται από πολλά πλεονεκτήματα καθώς οι υπηρεσίες

προσφέρονται στους χρήστες με ασφάλεια, ελαστικότητα και συνεχή διαθεσιμότητα, αλλά και η ευελιξία λόγω της άμεσης διάθεσης υπηρεσιών και η χρέωση η οποία αφορά μόνο τις υπηρεσίες που θα χρησιμοποιηθούν αποτελούν ένα ακόμη από πλεονεκτήματά του **[23]**.

Το ιδιωτικό νέφος (private cloud), όπου η δομή του είναι κοινή για έναν και μόνο οργανισμό. Ουσιαστικά πρόκειται για ένα σύνολο υπολογιστικών πόρων που προσφέρονται με τέτοιο τρόπο ώστε να σχεδιάζονται, να καθορίζονται και να ελέγχονται από έναν συγκεκριμένο οργανισμό **[24]**.

Το νέφος κοινότητας (community cloud), στο οποίο η δομή του είναι κοινή για μερικούς οργανισμούς υποστηρίζοντας μία συγκεκριμένη κοινότητα με κοινά ενδιαφέροντα και ανάγκες. Διαθέτει υποδομή η οποία είναι διαμοιρασμένη από πολλούς οργανισμούς και εξυπηρετεί συγκεκριμένη κοινότητα. Η κοινότητα αυτή έχει ως κοινό τόπο κάποιο συγκεκριμένο στόχο ή ενδιαφέρον **[25]**.

Το τελευταίο μοντέλο ανάπτυξης συμπληρώνει το υβριδικό νέφος (hybrid cloud). Το νέφος αυτό αποτελείται από δύο ή περισσότερα νέφη διαφορετικού είδους. Με άλλα λόγια το μοντέλο αυτό συνδυάζει τους πόρους που προέρχονται από ένα ή περισσότερα private cloud ή κάνοντας συνδυασμό αυτών των δύο. Το hybrid cloud έχει την δυνατότητα να προσφέρει στους χρήστες επεκτασιμότητα, εξοικονόμηση κόστους, ασφάλεια και ευελιξία **[26]**.

3.3 Amazon Elastic Cloud Computing (Amazon EC2)

Όπως αναφέρθηκε και στην εισαγωγή του κεφαλαίου το Amazon Elastic Cloud Computing (EC2) **[27]** είναι μια διαδικτυακή υπηρεσία η οποία αποτελεί ένα κεντρικό κόμβο της cloud computing πλατφόρμας της Amazon, και παρέχει την δυνατότητα χρησιμοποίησης και επέκτασης υπολογιστικών πόρων γρήγορα και άμεσα εφαρμόζοντας την τεχνική των εικονικών εξυπηρετητών. Παρέχει στον χρήστη την δυνατότητα να αποκτήσει και να προσαρμόσει τους απαιτούμενους υπολογιστικούς πόρους για τους οποίους ενδιαφέρεται μειώνοντας τον χρόνο που χρειάζεται ώστε να τους αποκτήσει ή αποδεσμεύσει, να έχει τον πλήρη έλεγχο των πόρων που χρησιμοποιεί αλλά και να κοστολογηθεί ανάλογα με την χρήση των πόρων που δεσμεύει **[28]**.

Τα βασικά χαρακτηριστικά του EC2 είναι **[29]**:

Amazon Elastic Block Store (EBS), παρέχει στους χρήστες ισχυρές δυνατότητες και προσφέρει σταθερό και αξιόπιστο χώρο αποθήκευσης για τα EC2 instances.

Πολλαπλές περιοχές, όπου το Amazon EC2 δίνει την δυνατότητα να τοποθετηθούν τα instances σε διαφορετικές τοποθεσίες. Οι τοποθεσίες αυτές αποτελούνται από ζώνες και περιοχές διαθεσιμότητας και με αυτό τον τρόπο προστατεύονται οι εφαρμογές από την αποτυχία μιας ενιαίας θέσης.

Ελαστικές IP διευθύνσεις (Elastic IP Addressing), αφορά στατικές IP διευθύνσεις που έχουν σχεδιαστεί για δυναμικό cloud computing. Μια ελαστική διεύθυνση IP συνδέεται με τον λογαριασμό του χρήστη και ο χρήστης έχει τον έλεγχο αυτής της διεύθυνσης IP μέχρι να την αποδεσμεύσει.

Ελαστική εξισορρόπηση φόρτου (Elastic Load Balancing), μοιράζει αυτόματα την εισερχόμενη κίνηση εφαρμογών σε πολλά instances επιτρέποντας στον χρήστη περισσότερη ανοχή σφαλμάτων στις εφαρμογές του.

Αυτόματη κλιμάκωση (Auto Scaling), με την αυτόματη κλιμάκωση, ο χρήστης μπορεί να εξασφαλίσει ότι ο αριθμός των instances που χρησιμοποιεί κλιμακώνεται αυτόματα ανάλογα με τις συνθήκες που έχει ορίσει. Κλιμακώνεται περισσότερο κατά τη διάρκεια της αιχμής των απαιτήσεων για να διατηρείται η απόδοση και λιγότερο κατά τη διάρκεια ύφεσης των απαιτήσεων για την ελαχιστοποίηση του κόστους.

VM Εισαγωγή/Εξαγωγή, με το VM Εισαγωγή/Εξαγωγή επιτρέπεται η εύκολη εισαγωγή εικόνων, εικονικής μηχανής από το περιβάλλον του χρήστη στο Amazon EC2 αλλά και η εξαγωγή τους ανά πάσα στιγμή.

3.3.1 Στοιχεία υπηρεσίας του Amazon EC2

Τα στοιχεία που χαρακτηρίζουν το Amazon EC2 είναι **[30]**:

Ελαστικότητα: Το Amazon EC2 δίνει την δυνατότητα στον χρήστη να χρησιμοποιήσει άμεσα από έναν μέχρι χιλιάδες servers ταυτόχρονα ώστε να καλύψει τις ανάγκες του και να κοστολογηθεί μόνο για τους υπολογιστικούς πόρους που χρησιμοποίησε.

Πλήρης έλεγχος: Το Amazon EC2 δίνει την δυνατότητα στον χρήστη να αποκτήσει τον πλήρη έλεγχο των instances, να έχει πρόσβαση σε αυτά και να μπορεί να αλληλεπιδράσει με αυτά, όπως ακριβώς θα έκανε και με οποιοδήποτε μηχάνημα.

Ευελιξία: Το Amazon EC2 προσφέρει μια ευελιξία στον χρήστη δίνοντάς του την επιλογή πολλαπλών instances, λειτουργικών συστημάτων και πακέτων λογισμικού και επιτρέποντάς του να επιλέξει και να διαμορφώσει το μέγεθος του αποθηκευτικού χώρου.

Σχεδιασμός: Το Amazon EC2 είναι σχεδιασμένο για χρήση και με άλλες δικτυακές υπηρεσίες της Amazon, όπως το Amazon Storage Service (υπηρεσία δικτυακού χώρου αποθήκευσης), την υπηρεσία βάσης δεδομένων της amazon.com (Amazon RDS) και την Amazon Simple Queue Service (Amazon SQS), ώστε να προσφέρει μια ολοκληρωμένη λύση υπολογιστικών υπηρεσιών.

Αξιοπιστία: Το Amazon EC2 διαθέτει ένα πλήρες και αξιόπιστο περιβάλλον καθώς η υπηρεσία λειτουργεί μέσα στην δικτυακή υποδομή και τα κέντρα δεδομένων της Amazon.

Ασφάλεια: Το Amazon EC2 προσφέρει πολλούς μηχανισμούς για τη διασφάλιση των υπολογιστικών πόρων, εντός του Virtual Private Cloud (VPC), ο χρήστης έχει την δυνατότητα να απομονώσει τα υπολογιστικά instances.

3.4 Επίλογος

Το cloud computing διαφορετικά υπολογιστικό νέφος αποτελεί μία από τις τεχνολογίες πληροφοριών το οποίο παρέχει επιπλέον προοπτικές σε τεχνολογίες δικτύωσης καθώς δίνει νέες δυνατότητες ως προς την αρχιτεκτονική, τον σχεδιασμό και την υλοποίηση των υπαρχόντων δικτύων και κέντρων δεδομένων, με χαμηλό κόστος. Όπως το Amazon Elastic Cloud το οποίο είναι μια διαδικτυακή υπηρεσία η οποία αποτελεί ένα κεντρικό κόμβο της cloud computing πλατφόρμας της εταιρίας Amazon, και παρέχει την δυνατότητα χρησιμοποίησης υπολογιστικής δύναμης εφαρμόζοντας την τεχνική των εικονικών εξυπηρετητών. Στο παρόν κεφάλαιο γίνεται αναφορά τόσο στα χαρακτηριστικά του cloud computing, στα μοντέλα παροχής υπηρεσίας και στα μοντέλα ανάπτυξης του cloud computing όσο και στα χαρακτηριστικά του Amazon EC2 και στα στοιχεία υπηρεσίας του.

Κεφάλαιο 4ο: Σχεδιασμός συστήματος

4.1 Εισαγωγή

Όπως είναι λογικό για να σχεδιαστεί και να υλοποιηθεί ένα «έξυπνο» και σύγχρονο σύστημα θα πρέπει πρώτα θα θέσουμε ορισμένες προδιαγραφές όσον αφορά στη λειτουργία και τους στόχους τους οποίους επιθυμούμε να πετύχουμε μέσω αυτού.

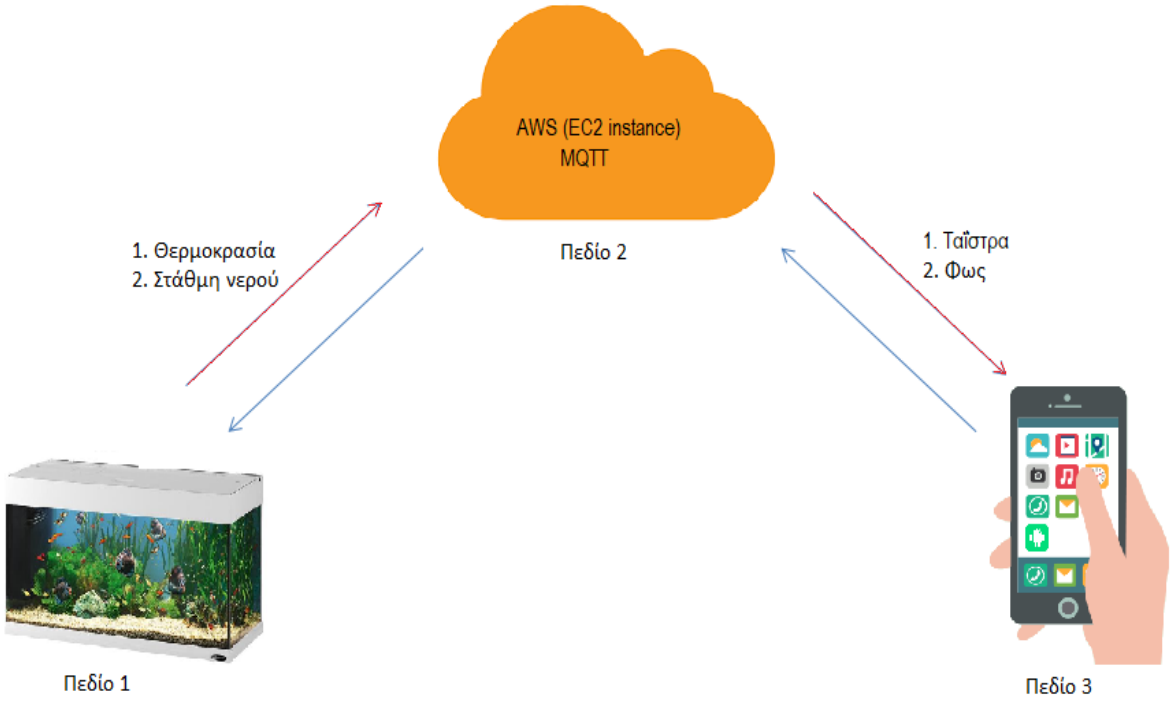
Βασικοί στόχοι του συστήματος αποτελούν η συνεχής και έγκυρη παρακολούθηση και ενημέρωση της θερμοκρασίας αλλά και της στάθμης του νερού στο εσωτερικό δοχείο του ενυδρείου. Ο έλεγχος και η ενεργοποίηση της τροφής των ψαριών και της φωτεινότητας του ενυδρείου αλλά και η δημιουργία μιας κατάλληλης και ολοκληρωμένης εφαρμογής android για τον χειρισμό των λειτουργιών του συστήματος όπως επίσης και η δυνατότητα του απομακρυσμένου ελέγχου.

Οι λειτουργίες για τις οποίες θα χρησιμοποιήσουμε τον εξοπλισμό που θα αναλύσουμε στο κεφάλαιο αυτό είναι η συνεχής μέτρηση και απεικόνιση της θερμοκρασίας του νερού στο δοχείο του ενυδρείο, η έγκυρη ενημέρωση της στάθμης του νερού, η χειροκίνητη και αυτόματη ενεργοποίηση της ταΐστρας και της φωτεινότητας του ενυδρείου. Στο κεφάλαιο αυτό επίσης θα αναλυθεί η αρχή λειτουργίας του συστήματος αλλά και η αλληλεπίδραση του χρήστη με αυτό μέσω της εφαρμογής android.

4.2 Η δομή του συστήματος

Το σύστημα το οποίο έχει σχεδιαστή, σύμφωνα και με το σχήμα 4.1 αποτελείται από τρία πεδία, το ενυδρείο, τον server και την εφαρμογή android (app). Στο πεδίο ένα βρίσκεται η κατασκευή, η οποία αποτελείται από το ενυδρείο, τα αισθητήρια και τον μικροελεγκτή, στο πεδίο δύο βρίσκεται ο server ο οποίος βρίσκεται στην πλατφόρμα Amazon Web Services (AWS) και είναι EC2 instance και στον οποίο τρέχει το MQTT δίνοντας έτσι την δυνατότητα επικοινωνίας με τις συσκευές που χρησιμοποιούνται στο σύστημα. Το πεδίο τρία αποτελείται από το κινητό τηλέφωνο στο οποίο έχει εγκατασταθεί η εφαρμογή android.

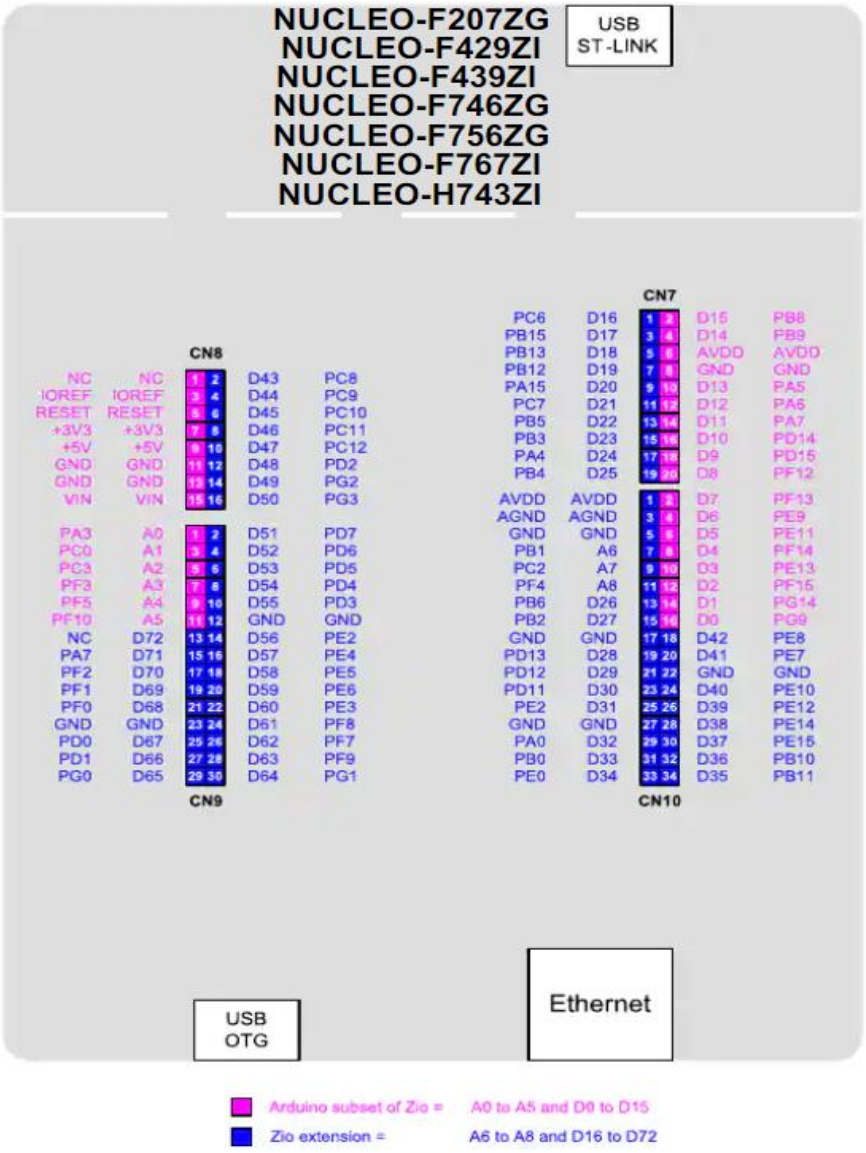
Από το πεδίο ένα (ενυδρείο) πραγματοποιείται η αποστολή δύο σημάτων, αυτό της θερμοκρασίας και της στάθμης νερού προς το πεδίο δύο (server). Το κάθε σήμα στέλνει ένα topic με payload τις τιμές οι οποίες λαμβάνονται από το αισθητήριο θερμοκρασίας και στάθμης νερού αντίστοιχα, στην διεύθυνση 18.220.230.XXX στην πόρτα 1883 του MQTT. Έπειτα ο server στέλνει το κάθε σήμα στην συσκευή που είναι συνδεδεμένη. Στην συγκεκριμένη περίπτωση όπως φαίνεται γίνεται χρήση του κινητού τηλεφώνου (smart phone) με λειτουργικό android, στο οποίο είναι εγκατεστημένη η εφαρμογή android. Αντίστοιχα από το πεδίο τρία (app) πραγματοποιείται η αποστολή ακόμη δύο σημάτων αυτό της ταΐστρας και της φωτεινότητας. Στην περίπτωση αυτή η αποστολή topic γίνεται από το κινητό τηλέφωνο προς τον server. Περιεχόμενο του payload είναι οι επιλογές οι οποίες έχουμε ορίσει στην εφαρμογή. Δηλαδή χειροκίνητο τάισμα (μία δόση), τάισμα κάθε 5 δευτερόλεπτα ή τάισμα κάθε 10 δευτερόλεπτα, ενώ για την επιλογή της φωτεινότητας υπάρχει η επιλογή αυτόματη ενεργοποίηση της led ταινίας ή χειροκίνητη ενεργοποίηση με την επιλογή on/off. Έπειτα ο server στέλνει το αντίστοιχο μήνυμα στην συσκευή η οποία είναι συνδεδεμένη από την άλλη πλευρά, δηλαδή στο ενυδρείο και με αυτή την διαδικασία ενεργοποιείται η λειτουργία που έχουμε επιλέξει από το app.



Σχήμα 4.1 : Δομή του συστήματος

4.3 Ο Μικροελεγκτής STM32H743ZI

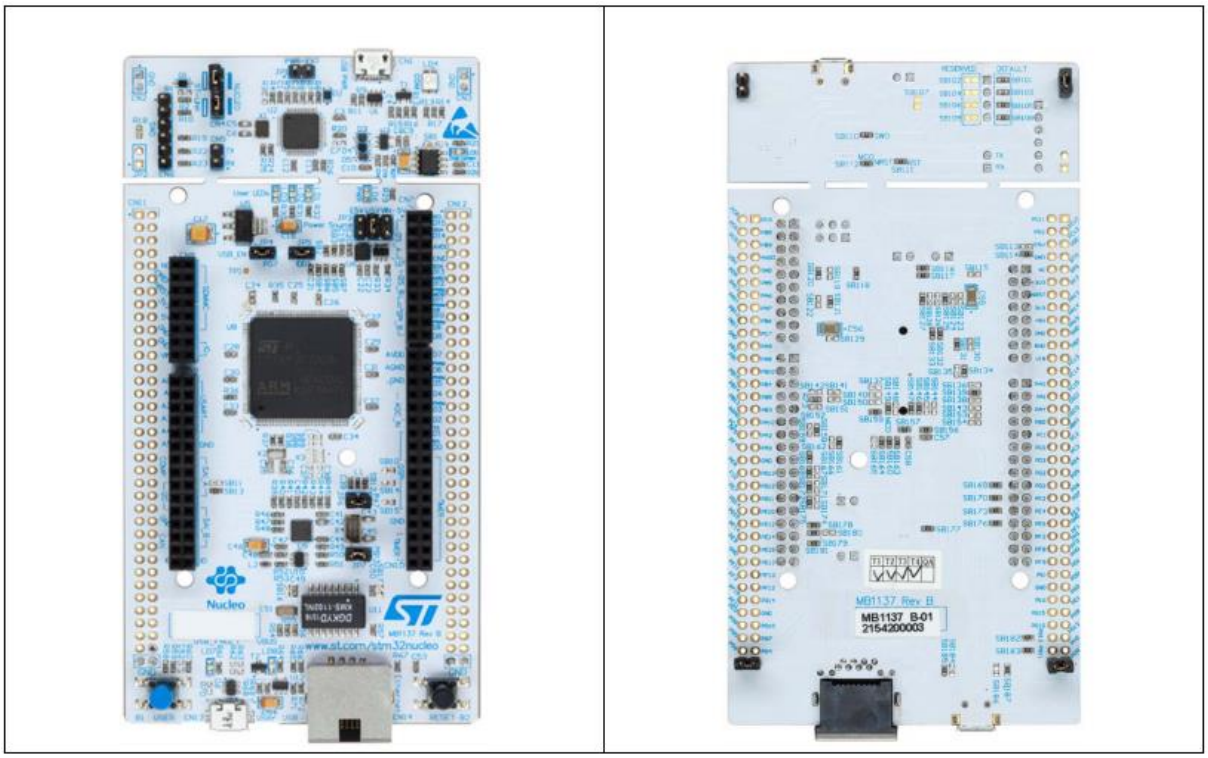
Ο STM32H743ZI αποτελεί έναν μικροελεγκτή και ανήκει στο project Nucleo της εταιρίας ST. Στόχος της είναι η δημιουργία προσιτών σε κόστος board ώστε να καθιστούν στον χρήστη εύκολη, γρήγορη και με πολλές δυνατότητες να υλοποιήσει τις νέες ιδέες του. Το συγκεκριμένο board έχει ενσωματωμένο τον μικροελεγκτή STM32H743ZI της Arm ο οποίος περιλαμβάνει τον Cortex-M7 επεξεργαστή με FPU και με μέγιστη συχνότητα λειτουργίας του τα 480MHz. Ο Cortex-M7 αναπτύχθηκε ειδικά έτσι ώστε να εξυπηρετήσει εφαρμογές με ενσωματωμένα συστήματα, τα οποία απαιτούν χαρακτηριστικά όπως υψηλή απόδοση, χαμηλό κόστος λειτουργίας και χαμηλή κατανάλωση. Μία από τις δυνατότητες που παρέχει το συγκεκριμένο board όσον αφορά την διασυνδεσιμότητα με άλλους μικροελεγκτές είναι ότι διαθέτει ST Zio connector για σύνδεση με arduino (εικόνα 15) **[31]**.



Εικόνα 15: ST Zio connector (www.st.com)

Τα βασικά χαρακτηριστικά της πλακέτας είναι **[32]**:

Διαθέτει όπως προαναφέρθηκε ενσωματωμένο τον μικροελεγκτή STM32H743ZI **[33]** της Arm ο οποίος περιλαμβάνει τον Cortex-M7 επεξεργαστή με FPU. Η μέγιστη συχνότητα λειτουργίας του είναι 480MHz. Επίσης διαθέτει ενσωματωμένο ένα μέρος το οποίο λέγεται ST-LINK/V2-1και αποτελεί ένα εργαλείο προγραμματισμού και εντοπισμού σφαλμάτων (Debugging) στον κώδικα που εκτελείται, δίνοντας πρόσβαση με αυτόν τον τρόπο στον προγραμματιστή σε τιμές καταχωριστών, μεταβλητών και μια σειρά παρόμοιων δυνατοτήτων, διευκολύνοντας τον εντοπισμό σφαλμάτων στον κώδικα. Το ST-LINK/v2-1 υποστηρίζει μόνο SWD (Serial Wire Debug) και μικροελεγκτές που ανήκουν στην κατηγορία STM32. Άλλα χαρακτηριστικά του board είναι ότι χρησιμοποιεί 3 Leds, 2 reset push-buttons και θύρες σύνδεσης USB με Micro-AB, SWD, Ethernet RJ45, ST Zio επαφές για σύνδεση με arduino. Στην παρακάτω εικόνα φαίνεται η ολοκληρωμένη πλακέτα Nucleo.



Εικόνα 16: Nucleo STM32H743ZI (www.st.com)

Επίσης ο μικροελεγκτής ενσωματώνει:

- 2MB μνήμης Flash, 1MB μνήμης SRAM
- Δώδεκα 16-bit χρονιστές (timers) γενικού σκοπού
- Δύο 32-bit χρονιστές (timers) γενικού σκοπού
- Τρεις 12-bit ADCs (Analog to Digital Converters)
- Δύο 12-bit DACs (Digital to Analog Converters)

και αποτελείται από τις ακόλουθες προηγμένες διεπαφές επικοινωνίας:

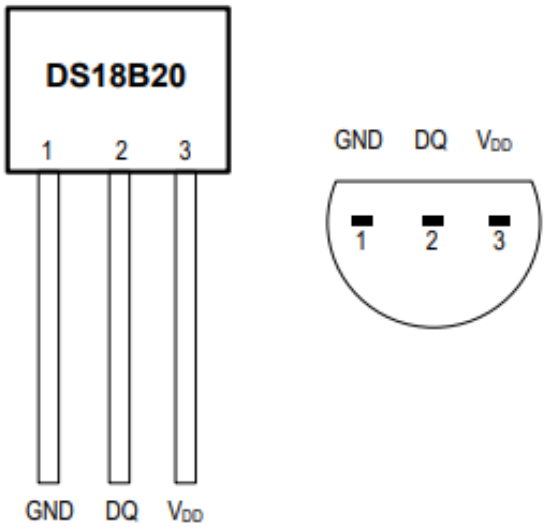
- Μέχρι τέσσερα I2C (Inter-Integrated Circuit)
- Μέχρι τέσσερα USART και τέσσερα UART
- Μέχρι έξι SPI (Serial Peripheral Interface)
- Δύο SAI (Serial Audio Interface)
- Δύο CAN (Controller Area Network)
- Ένα USB OTG (On The Go) full speed και ένα USB OTG με δυνατότητα full speed
- Διεπαφή κάμερας
- Ethernet συμβατό με IEEE-802.3-2002
- VDD από 1.62V μέχρι 3.6V
- High-resolution timer (2.1ns)
- Τρία I2S
- Ένα USB OTG Full Speed and High Speed
- Ένα USB OTG Full Speed
- Τέσσερα SPDIF_Rx
- Ένα HDMI_CEC
- Ένα Dual Mode Quad SPI
- Μέχρι 114 GPIO με δυνατότητα εξωτερικής διακοπής (interrupt)
- True Random Number Generator (RNG)
- 16-κανάλια DMA
- Ελεγκτή LCD-TFT με ανάλυση XGA

4.4 Έλεγχος της θερμοκρασίας του νερού

Ένας πολύ σημαντικός παράγοντας της ομαλής διαβίωσης στο εσωτερικό του δοχείου του ενυδρείου αποτελεί η θερμοκρασία, παίζοντας σημαντικό ρόλο τόσο στη ρύθμιση όσο και στην ανάπτυξη όλων των οργανισμών (ψάρια, υδρόβια φυτά). Επηρεάζει την ποσότητα του οξυγόνου που περιέχεται στο νερό αλλά και επιδρά στην θερμοκρασία του σώματος των ψαριών καθώς εξαρτώνται άμεσα από την θερμοκρασία του νερού που τα περιβάλει. Ουσιαστικά είναι ποικιλόθερμοι οργανισμοί, που μπορούν να επιβιώσουν μόνο σε συγκεκριμένο εύρος θερμοκρασιών και ορισμένα είδη είναι περισσότερο ευαίσθητα στις διακυμάνσεις της θερμοκρασίας. Μικρές μεταβολές της θερμοκρασίας έχουν ελάχιστες συνέπειες για τους περισσότερους οργανισμούς παρόλα αυτά όμως η απότομη πτώση της θερμοκρασίας στο εσωτερικό του δοχείου του ενυδρείου ευνοεί την ανάπτυξη ασθενειών. Ειδικότερα τα ψάρια έχουν την ικανότητα να αντέξουν σχετικά μεγάλες μεταβολές της θερμοκρασίας αλλά για μικρό χρονικό διάστημα, καθώς ο μεταβολισμός τους διαταράσσεται και αυτό έχει ως αποτέλεσμα τον θάνατό τους.

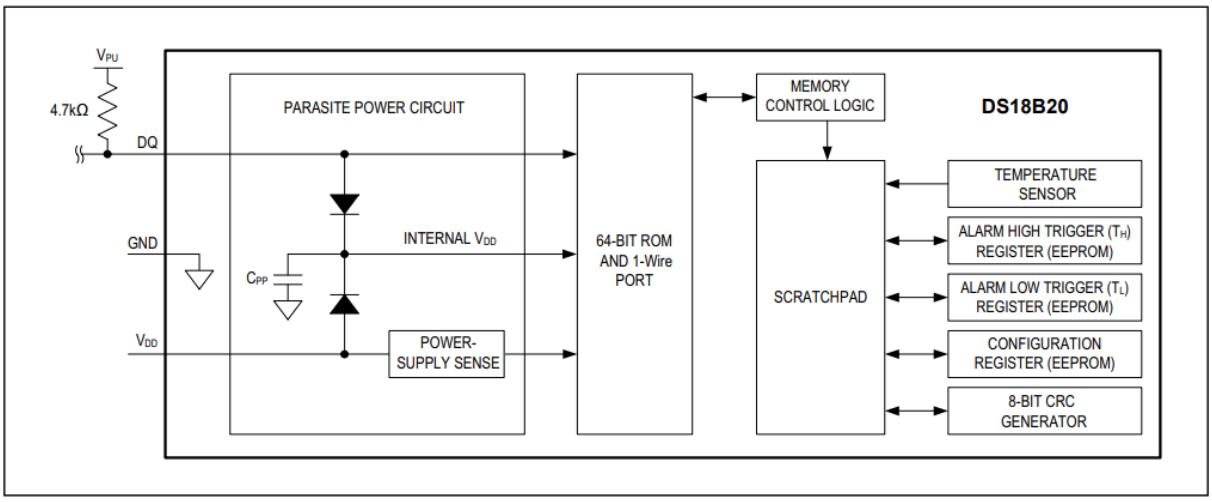
4.4.1 Αισθητήρας θερμοκρασίας

Έχοντας κατά νου την βαρύτητα της συγκεκριμένης παραμέτρου, για την μέτρηση της θερμοκρασίας του νερού στο εσωτερικό του ενυδρείου χρησιμοποιήθηκε το αισθητήριο DS18B20.



Εικόνα 17: Pin του αισθητήρα DS18B20

Το συγκεκριμένο αισθητήριο παρέχει 9-bit έως 12-bit μετρήσεις στην κλίμακα Κελσίου και επικοινωνεί μέσω του 1-Wire διαύλου, ο οποίος απαιτεί μία μόνο γραμμή, με ένα κεντρικό μικροελεγκτή. Το εύρος τιμών που καλύπτει είναι από -55 ^{o}C έως +125^{o}C με ακρίβεια ±0.5 ^{o}C στο διάστημα -10 ^{o}C έως +85 ^{o}C **[34]**.



Σχήμα 4.2: Διάγραμμα του αισθητήρα DS18B20

Τα βασικά χαρακτηριστικά του αισθητήρα DS18B20 είναι:

Πίνακας 1: Χαρακτηριστικά του DS18B20

Τάση τροφοδοσίας	3V-5.5V
Εύρος λειτουργίας	-55°C- +125 °C
Απόκλιση στο εύρος -10°C- +85°C	±5°C
Μέση απόκλιση	±2°C
Ρεύμα λειτουργίας	1mA
Τύπος αισθητήρα	Ψηφιακός

Το DS18B20 διαθέτει ψηφιακό αισθητήρα μέτρησης θερμοκρασίας η ανάλυση του οποίου ρυθμίζεται από τον χρήση σε 9, 10, 11 και προκαθορισμένη τα12-bit με βήμα που αντιστοιχεί σε μεταβολές 0.5 °C, 025 °C, 0.125 °C, 0.0625 °C. Όταν τροφοδοτηθεί, ο αισθητήρας βρίσκεται στην κατάσταση αναμονής, ενώ για να ξεκινήσει μια μέτρηση θερμοκρασίας ώστε να γίνει η μετατροπή της σε ψηφιακή τιμή πρέπει να σταλθεί μία εντολή. Όταν ολοκληρωθεί η μετατροπή τα δεδομένα αποθηκεύονται σε ένα καταχωρητή 2byte και το DS18B20 στην κατάσταση αναμονής.

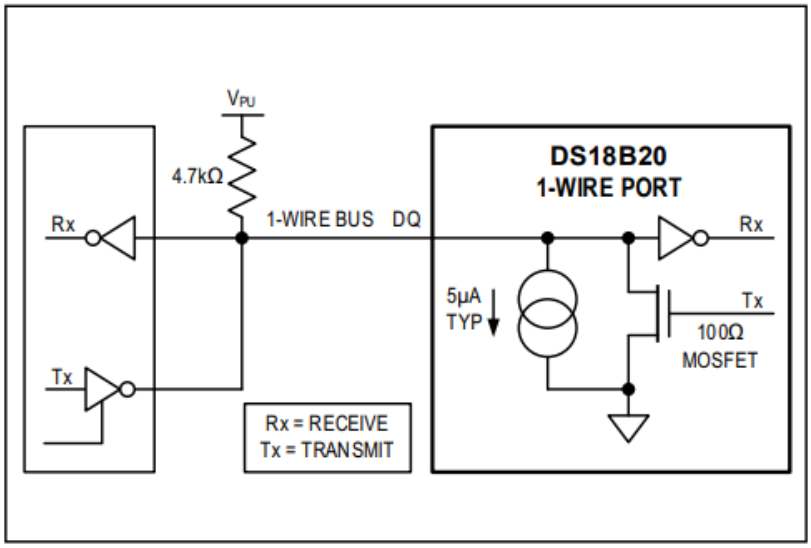
	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LS BYTE	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	BIT 15	**BIT 14**	**BIT 13**	**BIT 12**	**BIT 11**	**BIT 10**	**BIT 9**	**BIT 8**
MS BYTE	S	S	S	S	S	2^6	2^5	2^4

S = SIGN

Εικόνα 18: Καταχωρητής θερμοκρασίας

Τα sign bits (s) αντιπροσωπεύουν αν η θερμοκρασία είναι θετική ή αρνητική. Όταν s=0 αναφέρεται σε θετικούς αριθμούς ενώ όταν s=1 σε αρνητικούς. Αν το DS18B20 έχει ρυθμιστεί για 12-bit ανάλυση τότε όλα τα bit στον καταχωρητή θερμοκρασίας θα περιέχουν έγκυρα δεδομένα. Επίσης το σύστημα επικοινωνίας του αισθητήρα ονομάζεται 1-Wire επειδή χρειάζεται μία γραμμή για τον έλεγχό του και

την αποστολή των δεδομένων του. Η αποστολή των δεδομένων γίνεται μέσω μιας θύρας 3-state η open-drain, η οποία απαιτεί μια εξωτερική pull-up αντίσταση 4.7kΩ και επιτρέπει την συσκευή να αποδεσμεύσει την γραμμή όταν δεν αποστέλλει δεδομένα για να είναι διαθέσιμη προς χρήση από κάποια άλλη. Όπως φαίνεται στην εικόνα η θύρα 1-Wire είναι open drain.



Σχήμα 4.3: Τρόπος διασύνδεσης του DS18B20

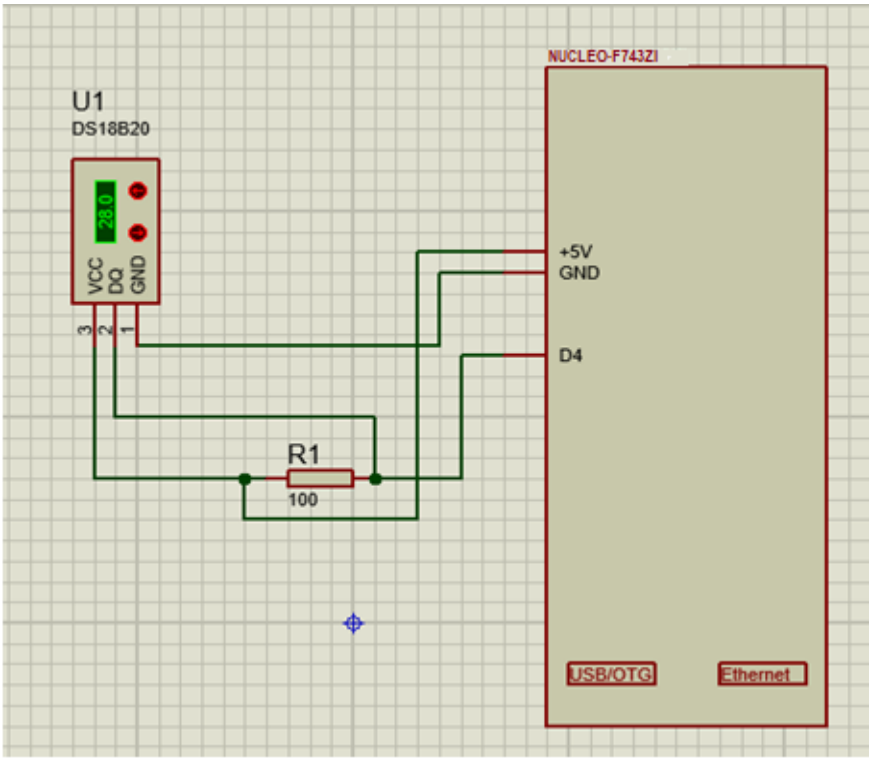
4.4.2 Συνδεσμολογία του αισθητήρα θερμοκρασίας στην κατασκευή

Το αισθητήριο έχει τρία pins, το κόκκινο καλώδιο αντιστοιχεί στην τροφοδοσία, το μαύρο στην γείωση και το κίτρινο στην αποστολή των μετρήσεων. Όπως φαίνεται στην παρακάτω εικόνα.

Εικόνα 19: Αισθητήρας DS18B20

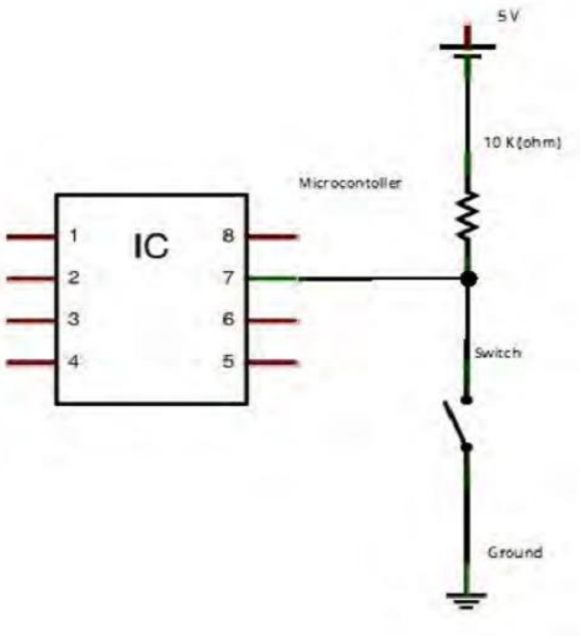
Η τροφοδοσία πραγματοποιείται από τον μικροελεγκτή στο pin που δίνει 5 volt όπως επίσης και η γείωση στο αντίστοιχο pin του μικροελεγκτή, ενώ το κίτρινο καλώδιο είναι συνδεδεμένο στο ψηφιακό

pin 4. Τέλος έχει τοποθετηθεί μία pull up αντίσταση 100Ω ανάμεσα στην τροφοδοσία και το pin data **[35]**, σχήμα 4.4.



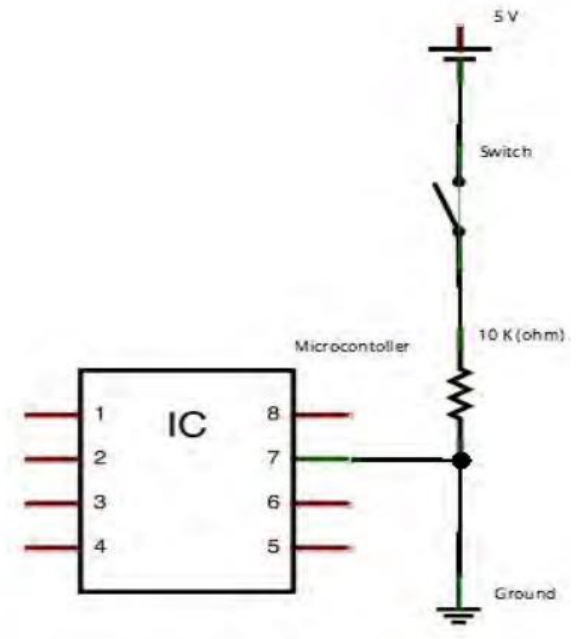
Σχήμα 4.4: Συνδεσμολογία του αισθητήρα DS18B20 (Proteus Design Suite)

Οι αντιστάσεις pull up χρησιμοποιούνται σε λογικές εισόδους ώστε να διατηρούν το σωστό επίπεδο τάσης, ακόμα και όταν αφαιρεθεί από το κύκλωμα ο αισθητήρας. Λέγεται pull up γιατί «τραβάνε» (pull) την τάση σε ένα υψηλό (up) προκαθορισμένο επίπεδο. Εάν κάποιο άλλο στοιχείο του κυκλώματος θέσει την τάση σε διαφορετικό επίπεδο, η pull up αντίσταση δεν θα επηρεάσει το κύκλωμα. Η βασική λειτουργία της pull up αντίστασης είναι να περιορίζει το κύκλωμα από το να περάσει ένα υψηλό ρεύμα που μπορεί να προκαλέσει βλάβη και για αυτό χρησιμοποιείται ευρύτατα. Οι περισσότεροι μικροελεγκτής διαθέτουν ενσωματωμένες αντιστάσεις pull up, αυτό σημαίνει πως μπορεί να συνδεθεί απευθείας κάποιος αισθητήρας αλλά θα πρέπει παρόλα αυτά για την ασφάλεια του κυκλώματος να συνδεθεί μία τέτοιου είδος αντίσταση. Όπως φαίνεται και στο σχήμα 4.5 εάν η αντίσταση είναι συνδεδεμένη στο θετικό άκρο +5V αντιπροσωπεύει μία pull up αντίσταση.



Σχήμα 4.5: Pull up αντίσταση

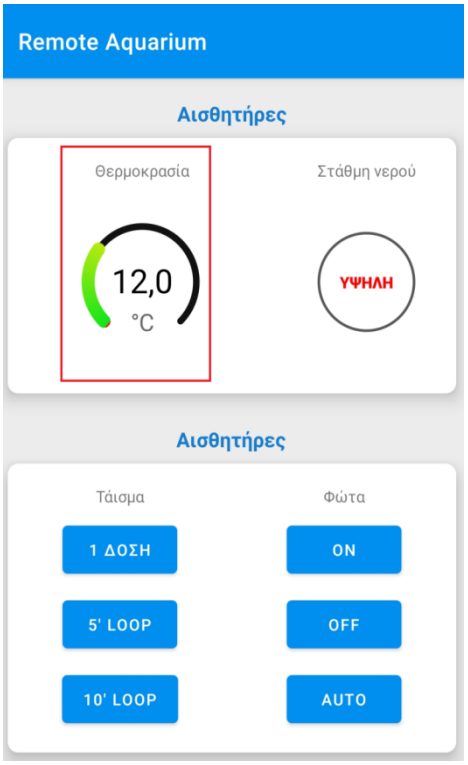
Σαν εναλλακτική λύση στη χρήση αντιστάσεων pull up, είναι η χρήση αντιστάσεων pull down .Όπως και οι pull up αντιστάσεις έτσι και η χρήση αντιστάσεων pull down, χρησιμοποιούνται για να περιορίσουν το ρεύμα του κυκλώματος , ανάμεσα στο Vcc και το GND. Για αυτό το λόγο μπορεί να ενσωματωθεί στο κύκλωμα μία αντίσταση 10K ή 472KΩ. Η ακριβής τιμή δεν έχει ιδιαίτερη σημασία από τη στιγμή που είναι αρκετά μεγάλη ώστε να περιορίζει το διαρρέον ρεύμα στις εισόδους του μικροελεγκτή. Αν είναι συνδεδεμένη στο άκρο GND τότε αντιπροσωπεύει μία pull down αντίσταση όπως φαίνεται και στο παρακάτω σχήμα **[36]**.



Σχήμα 4.6: Pull down αντίσταση

4.4.3 Εφαρμογή android και αισθητήρας θερμοκρασίας

Η ολοκληρωμένη εφαρμογή android στο συγκεκριμένο σκέλος της εργασίας χρησιμοποιείται ώστε να δώσει την δυνατότητα ελέγχου και ενημέρωσης της θερμοκρασίας του νερού στο δοχείο του ενυδρείου. Σύμφωνα με τα γραφικό περιβάλλον της εφαρμογής και με την εικόνα στο επάνω μέρος στην αριστερή πλευρά εμφανίζεται η κατάσταση της θερμοκρασίας του νερού (κόκκινο περίγραμμα, εικόνα 20) ανάλογα τις τιμές που λαμβάνει ο αισθητήρας θερμοκρασίας. Η εφαρμογή ενημερώνει τον χρήστη αναγράφοντας την θερμοκρασία σε βαθμούς κελσίου (^{o}C) σε μορφή δείκτη. Με αυτή την δυνατότητα ο κάτοχος του συστήματος ενημερώνεται ώστε ανάλογα με την θερμοκρασία και εφόσον χρειάζεται να προχωρήσει σε ενέργειες που απαιτούνται για την επιβίωση των οργανισμών του ενυδρείου.



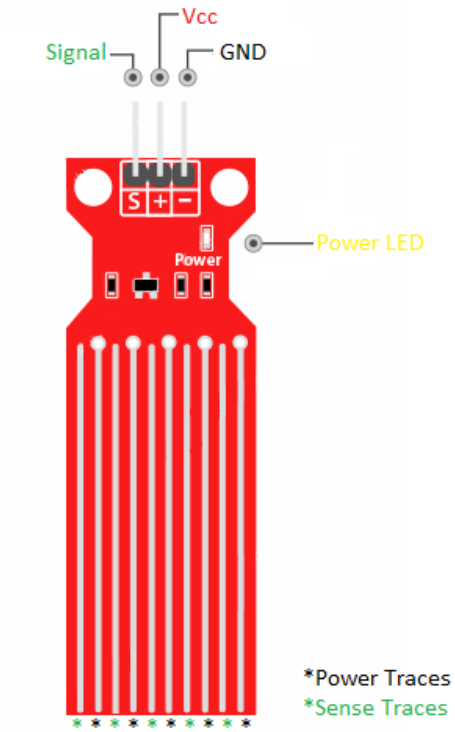
Εικόνα 20: Γραφικό περιβάλλον του app για την θερμοκρασία

4.5 Έλεγχος της στάθμης του νερού

Ο έλεγχος της στάθμης του νερού στο δοχείο του ενυδρείου έχει ιδιαίτερη βαρύτητα για την ζωή των οργανισμών στο εσωτερικό του. Τα ψάρια απαιτούν συγκεκριμένες ποσότητες οξυγόνου το οποίο αντλούν από το νερό. Επίσης ο αριθμός των ψαριών σε ένα ενυδρείο υπολογίζεται ανάλογα με το μέγεθος του ενυδρείου και την ποσότητα του νερού που μπορεί να χωρέσει σε αυτό, για τον λόγο αυτό είναι υψίστης σημασίας να γνωρίζουμε την στάθμη του νερού. Ένας άλλος παράγοντας που θα πρέπει να αναφερθεί είναι ότι στις ζεστές καλοκαιρινές μέρες και όχι απαραίτητα μόνο τότε, αρκετό νερό από το ενυδρείο εξατμίζεται, ιδίως σε ανοιχτά ενυδρεία χωρίς κάλυμμα. Η έλλειψη αυτή του νερού είναι απειλητική για την ζωή των οργανισμών στο εσωτερικό του δοχείου του ενυδρείου. Το βασικότερο επομένως είναι η διασφάλιση της επάρκειας σε οξυγόνο και αυτό μπορεί επιτευχθεί με τον έλεγχο της στάθμης του νερού.

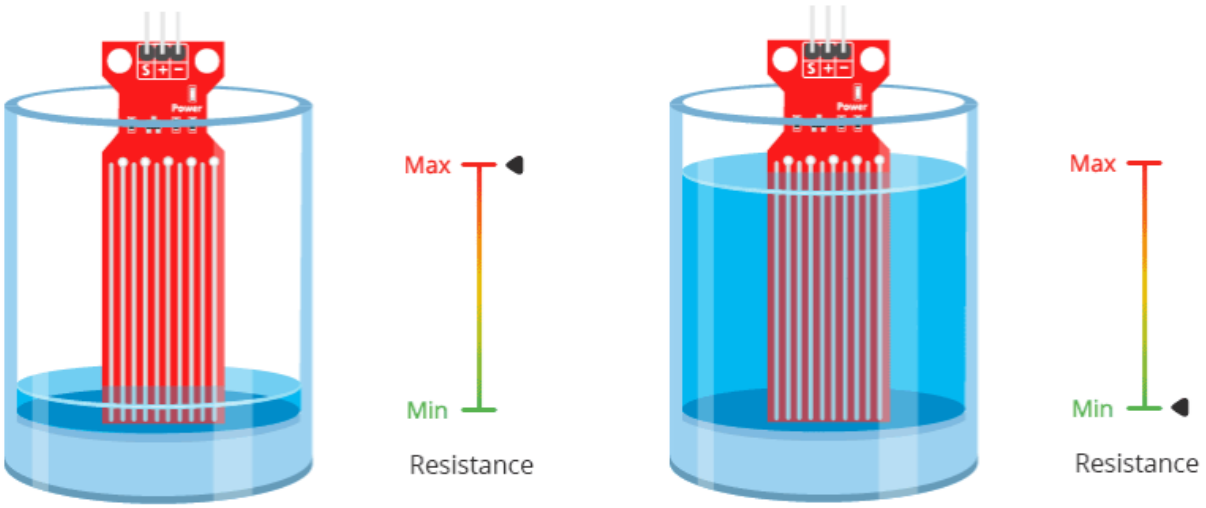
4.5.1 Αισθητήρας στάθμης

Όσον αφορά την λειτουργία του αισθητήρα στην κατασκευή, η μέτρηση της στάθμης του νερού στο εσωτερικό του ενυδρείου πραγματοποιήθηκε από την τοποθέτηση του αισθητήρα water level sensor (εικόνα 21). Ο συγκεκριμένος αισθητήρας αποτελείται από μια σειρά δέκα εκτεθειμένων ιχνών χαλκού, πέντε από τα οποία είναι ίχνη ισχύος (πράσινα αστεράκια, εικόνα 21) και πέντε ίχνη αίσθησης (μαύρα αστεράκια, εικόνα 21). Δεν υπάρχει σύνδεση μεταξύ των ιχνών αλλά γεφυρώνονται όταν βυθίζεται ο αισθητήρας στο νερό. Το led στο πάνω μέρος ενεργοποιείται όταν το αισθητήριο τροφοδοτείται.



Εικόνα 21: Αισθητήρας στάθμης νερού

Η λειτουργία του συγκεκριμένου αισθητήρα στάθμης νερού είναι πολύ απλή καθώς η σειρά των εκτεθειμένων παράλληλων αγωγών λειτουργεί ως μεταβλητή αντίσταση της οποίας η τιμή της ποικίλλει ανάλογα με τη στάθμη του νερού, ουσιαστικά η αλλαγή στην τιμή της αντίστασης αντιστοιχεί στην απόσταση από την κορυφή του αισθητήρα έως την επιφάνεια του νερού. Η αντίσταση είναι αντιστρόφως ανάλογη με το ύψος του νερού, με άλλα λόγια όταν το αισθητήριο βυθίζεται μέχρι πάνω, τόσο καλύτερη αγωγιμότητα υπάρχει και θα έχει ως αποτέλεσμα χαμηλότερη αντίσταση (εικόνα 22 δεξιά). Αντίθετα όταν το αισθητήριο βυθίζεται λιγότερο στο νερό δεν υπάρχει καλή αγωγιμότητα άρα θα έχει υψηλότερη αντίσταση (εικόνα 22 αριστερά). Επίσης ο αισθητήρας παράγει μια τάση εξόδου σύμφωνα με την αντίσταση και με αυτό τον τρόπο προσδιορίζεται η στάθμη του νερού **[37]**.



Εικόνα 22: Στάθμη νερού και αντίσταση

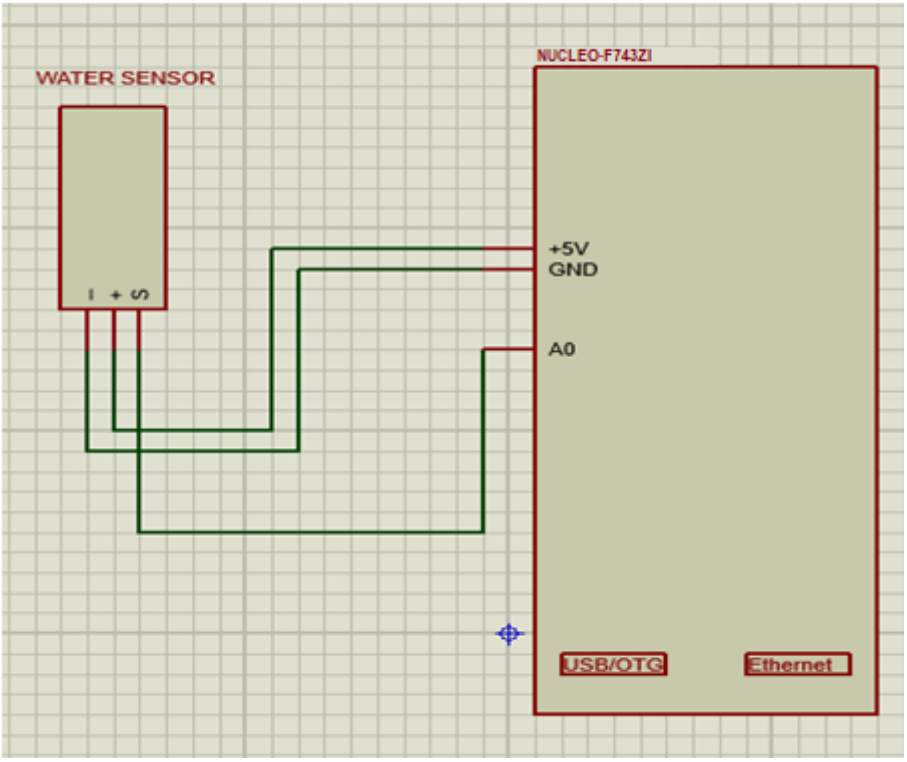
Τα βασικά χαρακτηριστικά του αισθητήρα αναγράφονται στον παρακάτω πίνακα.

Πίνακας 2: Χαρακτηριστικά του water level sensor

Τάση τροφοδοσίας	3V-5V
Θερμοκρασία	10^{o}C-30^{o}C
Υγρασία	10%-90%
Ρεύμα λειτουργίας	>20mA
Τύπος αισθητήρα	Αναλογικός

4.5.2 Συνδεσμολογία του αισθητήρα στάθμης νερού στην κατασκευή

Ο water sensor διαθέτει τρία pins, τα οποία συμβολίζονται με «+», «-» και «s». Η τροφοδοσία του συγκεκριμένου αισθητήρα πραγματοποιείται από τον μικροελεγκτή από το pin 5 volt όπως επίσης και η γείωση στο αντίστοιχο pin του μικροελεγκτή, ενώ το pin «s» είναι συνδεδεμένο στο αναλογικό pin A_O και παίρνει τιμές από 0 έως 1024, όπως φαίνεται και στο σχήμα 4.7. Για τις ανάγκες της συγκεκριμένης κατασκευής και λειτουργίας έχουν οριστεί τρία επίπεδα τιμών ώστε το σύστημα να αντιλαμβάνεται την αλλαγή κατάστασης του νερού. Για παράδειγμα έχουν οριστεί τα στάδια τιμών 0-300 (κατάσταση 1), 301-900 (κατάσταση 2) και 901-1024 (κατάσταση 3). Έστω λοιπόν ότι η στάθμη του νερού βρίσκεται σε ένα συγκεκριμένο σημείο ώστε το σύστημα να λαμβάνει τιμές μέσα στο διάστημα 301-900, δηλαδή στην κατάσταση 2, αν για οποιονδήποτε λόγο η στάθμη του νερού στο δοχείο μειωθεί ή αυξηθεί σε τέτοιο σημείο ώστε οι τιμές που λαμβάνει ο αισθητήρας να βρίσκονται είτε στο διάστημα 0-301 (κατάσταση 1), είτε στο διάστημα 901-1024 (κατάσταση 3) και όχι στο διάστημα 301-900 (κατάσταση 2) όπου βρισκόταν, αυτό έχει ως αποτέλεσμα το σύστημα να αντιληφθεί την αλλαγή μιας κατάστασης της στάθμης του νερού ώστε να εκτελέσει μία συγκεκριμένη ενέργεια. Στο παρακάτω σχήμα αναπαριστάται η συνδεσμολογία του αισθητήρα στάθμης.

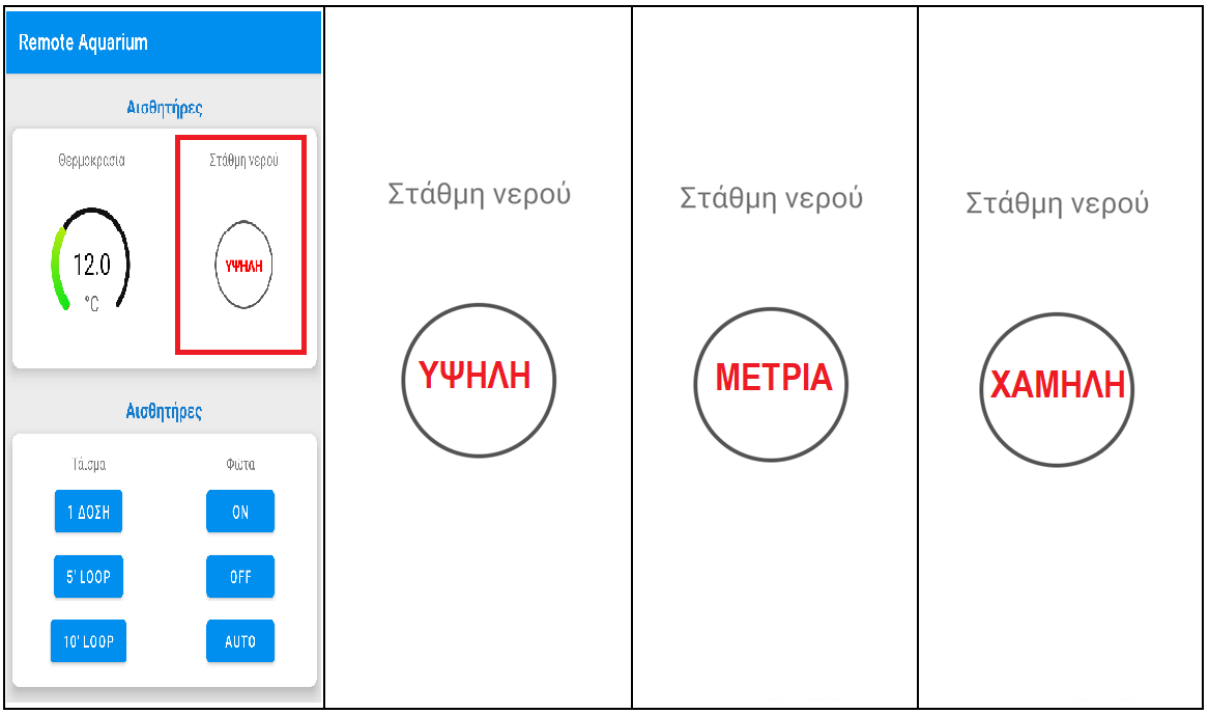


Σχήμα 4.7: Συνδεσμολογία του αισθητήρα στάθμης νερού (Proteus Design Suite)

4.5.3 Εφαρμογή android και αισθητήρας στάθμης νερού

Όπως προαναφέρθηκε και στην προηγούμενη ενότητα για την λειτουργία του αισθητήρα έχουν οριστεί τρία επίπεδα τιμών τα οποία για να χρησιμοποιηθούν μέσα στην εφαρμογή android έχουν οριστεί με ονομασία «υψηλή» (κατάσταση 3), «μέτρια» (κατάσταση 2) και «χαμηλή» (κατάσταση 1). Το επίπεδο «υψηλή» αντιπροσωπεύει στο γεμάτο με νερό δοχείο του ενυδρείο, το επίπεδο «μέτρια» στο γεγονός ότι χρειάζεται να συμπληρωθεί ορισμένη ποσότητα νερού και το «χαμηλή» ότι το νερό που βρίσκεται στο ενυδρείο είναι στα πολύ χαμηλά επίπεδα και χρειάζεται άμεσα συμπλήρωση αρκετής ποσότητας νερού με την προϋπόθεση ότι στο εσωτερικό του δοχείο υπάρχουν οργανισμοί όπως ψάρια ή φυτά τα οποία μπορεί να κινδυνεύουν από την έλλειψη νερού.

Επομένως η ολοκληρωμένη εφαρμογή android στο συγκεκριμένο σκέλος της εργασίας χρησιμοποιείται ώστε να δώσει την δυνατότητα ελέγχου και ενημέρωσης της στάθμης του νερού στο δοχείο του ενυδρείου. Σύμφωνα με τα γραφικό περιβάλλον της εφαρμογής και με την εικόνα 23 στο επάνω μέρος στην δεξιά πλευρά εμφανίζεται η κατάσταση της στάθμης του νερού (κόκκινο περίγραμμα, εικόνα 23) ανάλογα τις τιμές που λαμβάνει ο αισθητήρας στάθμης. Η εφαρμογή ενημερώνει τον χρήστη αναγράφοντας την ένδειξη «ΥΨΗΛΗ» όταν η στάθμη του νερού βρίσκεται στην κατάσταση 3 και ο αισθητήρας λαμβάνει τιμές μέσα στο διάστημα τιμών που έχει οριστεί και αναφερθεί προηγουμένως. Με άλλα λόγια αντιπροσωπεύει την κατάσταση όπου ο αισθητήρας έχει βυθιστεί μέχρι πάνω. Επίσης αναγράφοντας την ένδειξη «ΜΕΤΡΙΑ» όταν η στάθμη του νερού βρίσκεται στην κατάσταση 2 δηλαδή αντιπροσωπεύει την κατάσταση όπου ο αισθητήρας έχει βυθιστεί μέχρι την μέση ενώ η ένδειξη χαμηλή «ΧΑΜΗΛΗ» όταν η στάθμη του νερού βρίσκεται στην κατάσταση 1 και δεν υπάρχει επαφή του αισθητήρα με το νερό στο εσωτερικό του δοχείου του ενυδρείου.



Εικόνα 23: Γραφικό περιβάλλον του app για την στάθμη του νερού

4.6 Έλεγχος της φωτεινότητας

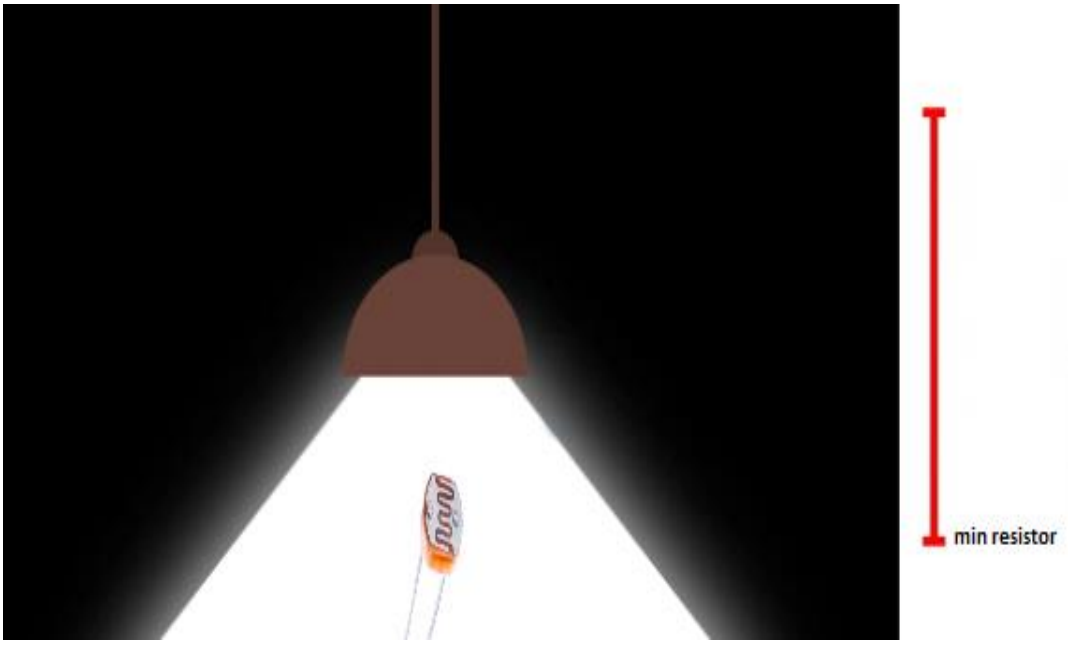
Η χρήση φως στο ενυδρείο για την κατάλληλη ένταση φωτός στο ενυδρείο που είναι απαραίτητη για τους οργανισμούς που βρίσκονται μέσα σε αυτό δεν χρησιμοποιείται μόνο για αισθητικούς σκοπούς. Αντίθετα έχει ένα πολύ σημαντικό ρόλο, καθώς με τον έλεγχο της διάρκειας λειτουργίας της λάμπας και του φως που εκπέμπεται, τα όρια της θερμοκρασίας που παράγει η λάμπα κυμαίνονται στα επιθυμητά πλαίσια ώστε να μην υπάρξει εξασθένιση της ποσότητας του νερού στο ενυδρείου από την θερμοκρασία που αναπτύσσεται. Ένας άλλος σημαντικός ρόλος είναι αυτός του ρυθμιστή στο βιολογικό ρολόι των οργανισμών που ζουν στο εσωτερικό του δοχείου του ενυδρείου και ειδικότερα για τα υδρόβια φυτά. Τα φυτά χρειάζονται το φως για να φωτοσυνθέσουν, καθώς λαμβάνουν όλη την απαιτούμενη για τις ζωτικές τους λειτουργίες, ενέργεια. Η χρονική διάρκεια που η λάμπα εκπέμπει φως αποτελεί πολύ σημαντικό παράγοντα καθώς όλα τα είδη δεν επηρεάζονται εξίσου, ούτε έχουν τις ίδιες ανάγκες για φως. Επομένως η χρήση ενός αισθητήρα φωτός ώστε να υπάρχει φως ανάλογα με την ένταση της φωτεινότητας του χώρου στον οποίο βρίσκεται τοποθετημένο το ενυδρείο και την επιθυμητή χρονική διάρκεια που απαιτείται επιτρέπει την αυτοματοποίηση της συγκεκριμένης λειτουργίας και μας απαλλάσσει από την καθημερινή ρουτίνα του άνοιξε κλείσε του διακόπτη.

4.6.1 Αισθητήρας φωτεινότητας

Συνεπώς η αίσθηση την οποία θα προσθέσουμε στην συγκεκριμένη κατασκευή είναι αυτή της λήψης της τιμής του επιπέδου φωτεινότητας στον χώρο που είναι εγκατεστημένο το σύστημα και αυτό θα πραγματοποιείται από τον αισθητήρα φωτός. Ο συγκεκριμένος αισθητήρας έχει τρία pins, το V_{CC} από το οποίο τροφοδοτείται, το GND (γείωση) και το Data **[38]**.

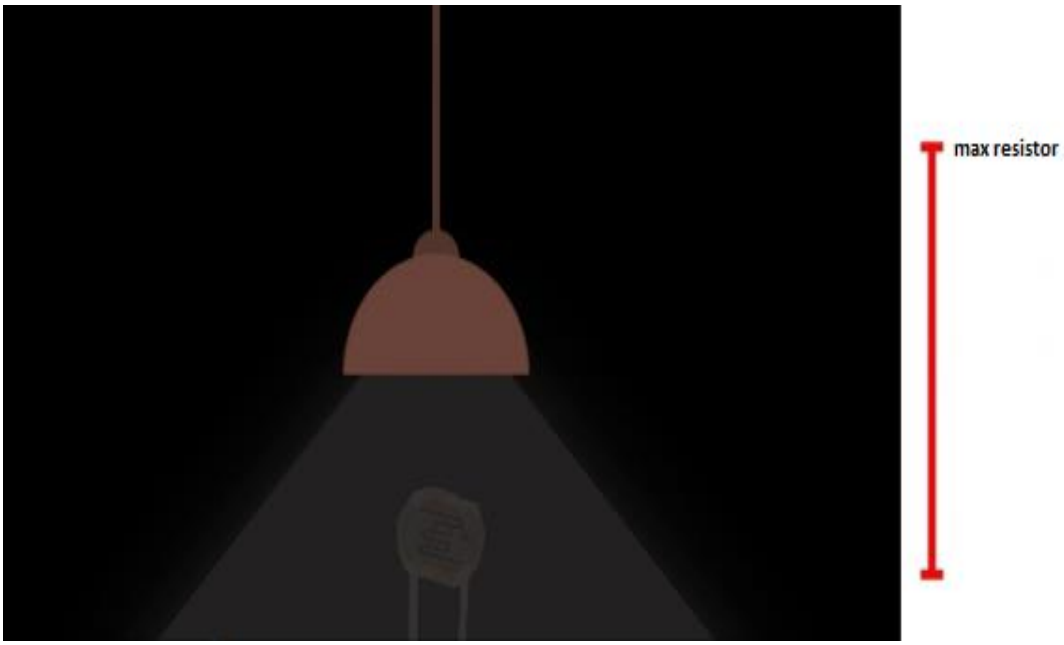
Εικόνα 24: Photoresistor

Ουσιαστικά πρόκειται για μία αντίσταση μεταβλητής τιμής όπου η τιμής της επηρεάζεται από την φωτεινότητα του χώρου. Η photoresistor ή LDR (light depended resistor) όταν σε αυτήν προσπίπτουν ακτίνες φωτός η τιμής της αντίστασης κυμαίνεται σε μικρότερες τιμές όπως φαίνεται και στην παρακάτω εικόνα:



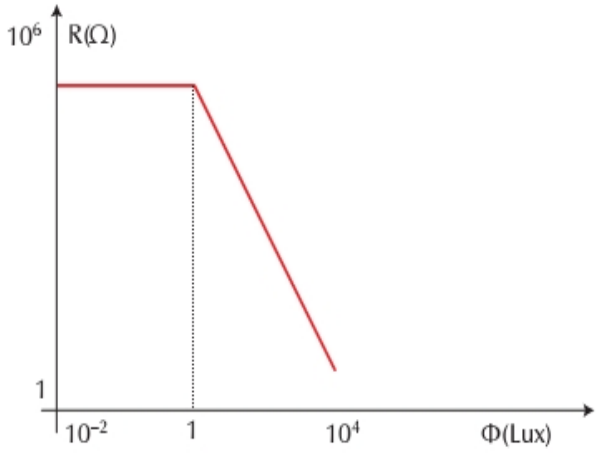
Εικόνα 25: Αισθητήρας και αντίσταση

Αντίθετα όταν ο χώρος στον οποίο βρίσκεται είναι σκοτεινός η τιμή της αντίστασης κυμαίνεται σε μεγαλύτερες τιμές (εικόνα 26).



Εικόνα 26: Αισθητήρας και αντίσταση

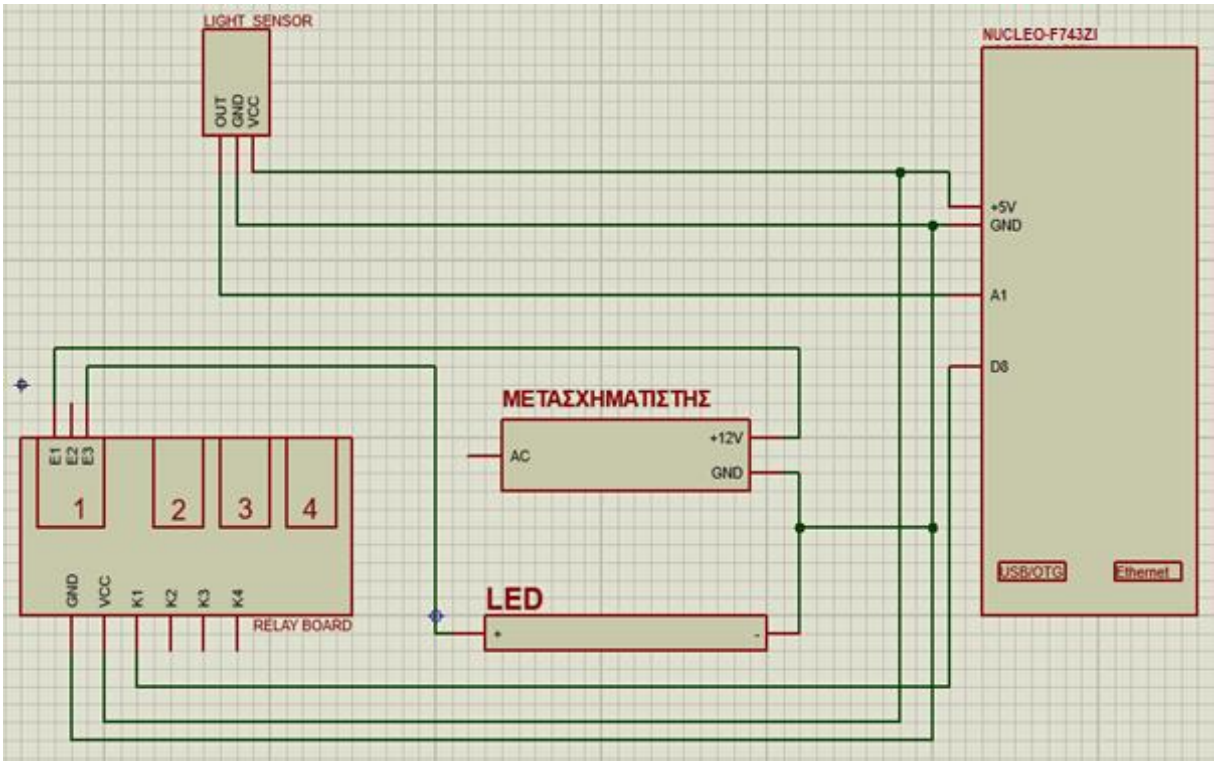
Με άλλα λόγια καθώς το φως διεισδύει στον ανιχνευτεί η τιμή της αντίστασης μεταβάλλεται από 1ΜΩ όταν ο ανιχνευτής βρίσκεται στο σκοτάδι έως μερικές εκατοντάδες Ω όταν πέφτει φως πάνω στον ανιχνευτεί. Τα προσπίπτοντα φωτόνια τα οποία δεσμεύει ο ημιαγωγός, αποδεσμεύουν τα δεσμευμένα ηλεκτρόνια και τα αποσπούν από τα άτομα που ανήκαν καθιστώντας τα ελευθέρα και με αυτόν τον τρόπο υπάρχει μείωση της αντίστασης του φωτοαγώγιμου υλικού. Δηλαδή όσο περισσότερο φως εκτίθεται πάνω στην φωτοαντίσταση τόσο μικρότερη είναι η αντίστασή της. Το παρακάτω διάγραμμα αναπαριστά την μεταβολή της αντίστασης στην ένταση της τιμής του φως όταν προσπίπτει στην επιφάνεια της φωτοαντίστασης. Η αρχική τιμή της αντίστασης είναι αρκετά μεγάλη των ΜΩ και μειώνεται με την ένταση του φωτός που προσπίπτει στην επιφάνεια **[39].**



Σχήμα 4.8: Μεταβολή της αντίστασης συγκριτικά με την ένταση φωτεινής ακτινοβολίας

4.6.2 Συνδεσμολογία του αισθητήρα φωτεινότητας στην κατασκευή

Ο αισθητήρας φωτεινότητας όπως φαίνεται και σχήμα 4.9, τροφοδοτείται από τον μικροελεγκτή από το pin 5 volt καθώς και η γείωση στο αντίστοιχο pin του μικροελεγκτή, ενώ το pin data είναι συνδεδεμένο στο αναλογικό pin A_1 του μικροελεγκτή. Στην λειτουργία αυτή έχει χρησιμοποιηθεί επίσης ένας μετασχηματιστής ώστε να μετατρέπει την τάση από 240V AC σε 12V DC, μία led ταινία και ένας διακόπτης ρελέ. Το θετικό άκρο (12V) του μετασχηματιστή συνδέεται στην επαφή E_1 του ρελέ και στο αρνητικό άκρο της led ταινίας ενώ η επαφή E_3 του ρελέ με το θετικό άκρο της led ταινίας ώστε να «κλείσει» το κύκλωμα μόλις δοθεί το σήμα. Η πλακέτα ρελέ με τα τέσσερα κανάλια τροφοδοτείται με 5V και γειώνετε από τον μικροελεγκτή. Επίσης μία από τις τέσσερις επαφές που αντιστοιχεί στο πρώτο ρελέ είναι συνδεδεμένη με το ψηφιακό pin D_8 του μικροελεγκτή ώστε να δέχεται σήμα με σκοπό να «κλείσει» το κύκλωμα.



Σχήμα 4.9: Συνδεσμολογία του αισθητήρα Photoresistor (Proteus Design Suite)

Ο ηλεκτρονόμος ή ρελέ είναι ένας ηλεκτρικός διακόπτης που ανοιγοκλείνει ένα ηλεκτρικό κύκλωμα ελεγχόμενο από ένα άλλο ηλεκτρικό κύκλωμα και εφευρέθηκε από τον Τζόζεφ Χένρυ το 1835 **[40]**. Χρησιμοποιείται για την κατασκευή ηλεκτρονικών κυκλωμάτων και με την βοήθεια αισθητήρων εξυπηρετούν τον αυτοματισμό του κυκλώματος χωρίς την ανθρώπινη παρέμβαση. Επαφές ενός ηλεκτρονόμου αποτελούν οι «normally open», «normally closed» και η «change over».

Για την παρούσα κατασκευή ως ρελέ έχει χρησιμοποιηθεί η πλακέτα της εικόνα 27. Η συγκεκριμένη πλακέτα διαθέτει ενσωματωμένα τέσσερα κανάλια για την κατασκευή όμως χρησιμοποιήθηκε ένα κανάλι έτσι ώστε να ενεργοποιείται ή να απενεργοποιείται η led ταινία.

Εικόνα 27: Πλακέτα ρελέ με 4κανάλια

Τα βασικά χαρακτηριστικά της πλακέτας ρελέ είναι:

- τέσσερα κανάλια τα οποία μπορούν να ελεγχθούν άμεσα από μικροελεγκτές
- μπορεί να ελέγχει διάφορες συσκευές αλλά και συσκευές με μεγάλο ρεύμα
- μέγιστη έξοδος είναι AC 250V 10A και DC 30V 10A
- μπορεί να συνδεθεί άμεσα με μικροελεγκτές
- διαθέτει κόκκινο ενδεικτικό λαμπάκι λειτουργίας για την ασφαλή χρήση
- μέγεθος πλακέτας περίπου 7,1x5,3x1,7 εκατοστά

Επίσης ως ενεργοποιητή στην συγκεκριμένη λειτουργία έχει χρησιμοποιηθεί μία led ταινία του εμπορίου η οποία λειτουργεί με 12Volt όπως φαίνεται στην παρακάτω εικόνα.

Εικόνα 28: Led ταινία

Και για την μετατροπή της τάσης από 240Volt AC σε 12Volt DC χρησιμοποιήθηκε ο μετασχηματιστής ATM-L32B09W12V της εταιρίας atman:

Εικόνα 29: Μετασχηματιστής

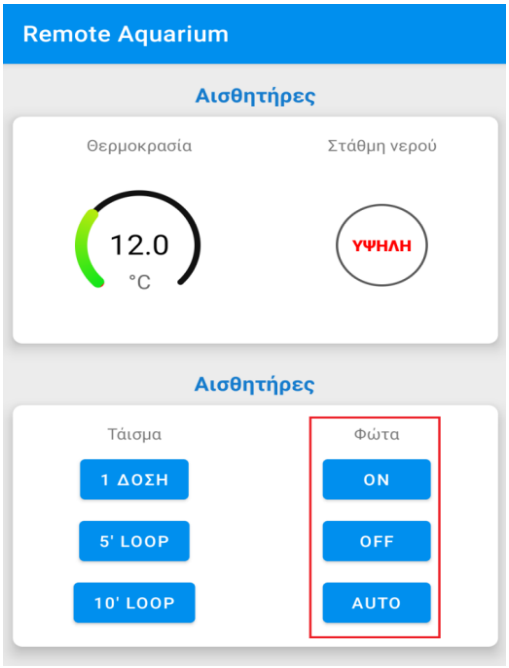
Τα χαρακτηριστικά του οποίου είναι:

Input: 220-240V-50/60Hz

Output: DC 12V Max.:750mA Max.:9W

4.6.3 Εφαρμογή android και αισθητήρας φωτεινότητας

Η ολοκληρωμένη εφαρμογή android στο συγκεκριμένο σκέλος της εργασίας χρησιμοποιείται ώστε να δώσει την δυνατότητα ελέγχου, ενεργοποίησης/απενεργοποίησης και αλληλεπίδρασης μεταξύ της εφαρμογής (app) και της κατασκευής, αυτόματα ή μεταξύ της κατασκευής και του χρήστη μέσω της εφαρμογής (app), χειροκίνητα. Σύμφωνα με τα γραφικό περιβάλλον της εφαρμογής και με την εικόνα 30 στο κάτω μέρος στην δεξιά πλευρά της εφαρμογής (κόκκινο περίγραμμα), αναγράφονται οι επιλογές της συγκεκριμένης λειτουργίας.



Εικόνα 30: Γραφικό περιβάλλον του app για το φως

Στο κομμάτι αυτό του χειρισμού υπάρχει η χειροκίνητη ενεργοποίηση ή απενεργοποίηση της led ταινίας με την επιλογή «ON» ή «OFF» αντίστοιχα, αλλά και η επιλογή «AUTO» στην οποία το φως ενεργοποιείται ανάλογα με την ένταση της φωτεινότητας του χώρου με την βοήθεια της φωτοαντίστασης που υπάρχει ενσωματωμένη πάνω στον αισθητήρα. Όταν ο χρήστης επιθυμεί να ενεργοποιήσει ή να απενεργοποιήσει χειροκίνητα το φως τότε μέσα από το app επιλέγει «ON» ή «OFF» αντίστοιχα, ενώ όταν ο χρήστης επιθυμεί να ενεργοποιείται ή να απενεργοποιείται αυτόματα το φως στο ενυδρείο τότε πάλι μέσα από το app επιλέγει «AUTO» και τότε με βάση την παραπάνω λειτουργία του αισθητήρα το led ενεργοποιείται όταν είναι απαραίτητο.

4.7 Η λειτουργία της ταΐστρας

Μία από τις βασικότερες ενέργειες του κατόχου του ενυδρείου αποτελεί η συστηματική ρίψη τροφής μέσα στον χώρο του ενυδρείου, με άλλα λόγια μέσα στο δοχείο που φιλοξενούνται οι ζωντανοί οργανισμοί (ψάρια), καθώς τα ψάρια δεν έχουν την δυνατότητα να αναζητήσουν και να βρουν τροφή. Για τον λόγο αυτό η ικανοποίηση των διατροφικών αναγκών των ψαριών εξαρτάται και είναι υποχρέωση του κατόχου και για τον σκοπό αυτό υπάρχουν διάφορες συσκευές στην αγορά αποκαλούμενες, ηλεκτρονικές ταΐστρες για την περιοδική προσθήκη συγκεκριμένης ποσότητας τροφής στο δοχείο του ενυδρείου. Παρόλο όμως που η ύπαρξη τέτοιον βοηθητικών συσκευών κάνει πιο εύκολη την συντήρηση του ενυδρείου, δεν αλλάζει το γεγονός ότι υπάρχουν περιορισμοί στον συγκεκριμένο αυτοματισμό.

Επομένως μέσω της παρούσας κατασκευής υλοποιείται η περιοδική και προγραμματισμένη απελευθέρωση τροφής στο ενυδρείο, χειροκίνητα ή αυτόματα μέσω εφαρμογής android με κεντρικό σύστημα ελέγχου τον μικροελεγκτή STM32H743ZI.

4.7.1 Η ταΐστρα

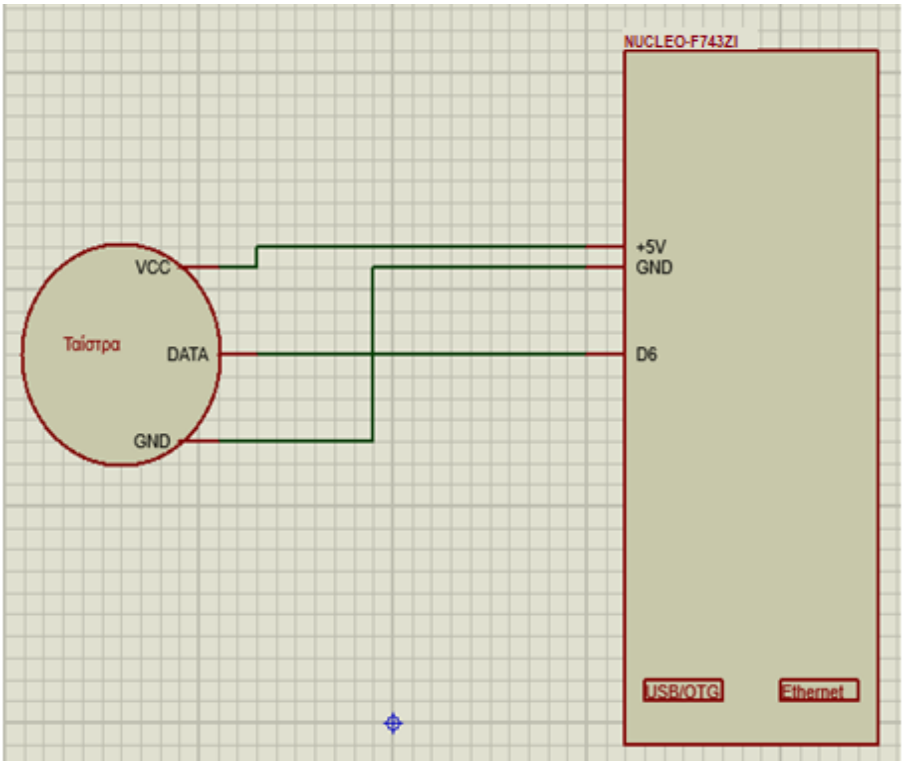
Στην συγκεκριμένη λειτουργία χρησιμοποιήθηκε μία ταΐστρα του εμπορίου που είναι κατασκευασμένη από πλαστικό και διατίθεται στην αγορά ως αυτόματη συσκευή με τη χρήση μπαταριών, διαθέτει αυτόματο άλεσμα και διανομή της ξηρής τροφής στους προκαθορισμένους χρόνους (ανά 12 ώρες) ώστε να αποτραπεί η ανάπτυξη υγρασίας στην τροφή. Διαθέτει επίσης χειροκίνητη επιλογή, διανέμοντας την τροφή με το πάτημα ενός button. Έχει διαστάσεις 10.5cm x 10cm x 6cm και βάρος 0.249Kg.

Εικόνα 31: Ταΐστρα

Για τις ανάγκες όμως της παρούσας κατασκευής ώστε να υλοποιηθεί σύμφωνα με τις απαιτήσεις που είχαμε θέσει εκ των προτέρων στην συγκεκριμένη κατασκευή χρειάστηκε η τροποποίησή της. Αρχικά αφαιρέθηκε ο διακόπτης και προστέθηκε ένα καλώδιο ώστε να συνδεθεί στον μικροελεγκτή ως pin data (pin 6).

4.7.2 Συνδεσμολογία της ταΐστρας στην κατασκευή

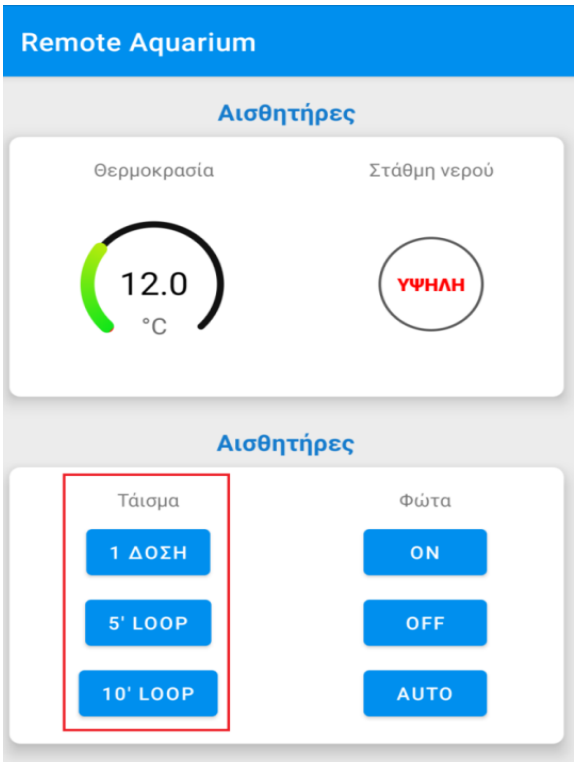
Έπειτα από την τροποποίηση της ταΐστρας η συσκευή διαθέτει τρία pins. Το πρώτο pin είναι η τροφοδοσία η οποία πραγματοποιείται πάνω από τον μικροελεγκτή στο pin με τα 5 volt αλλά και η γείωση στο αντίστοιχο pin του μικροελεγκτή και το οποίο αποτελεί το δεύτερο pin της ταΐστρας. Το τρίτο pin της ταΐστρας συνδέεται στο ψηφιακό pin 6 του μικροελεγκτή. Στο παρακάτω σχήμα εμφανίζεται η συνδεσμολογία της ταΐστρας.



Σχήμα 4.10: Συνδεσμολογία της ταΐστρας (Proteus Design Suite)

4.7.3 Εφαρμογή android και λειτουργία ταΐσματος

Η λειτουργία για το τάισμα πραγματοποιείται από την εφαρμογή android χρησιμοποιώντας ένα smart phone ο χρήστης έχει την δυνατότητα να επιλέξει ανάμεσα σε τρεις επιλογές. Στην κάτω πλευρά της εφαρμογής εμφανίζονται οι επιλογές του χρήστη για της διαδικασία του ταΐσματος (κόκκινο περίγραμμα, εικόνα 32). Ο χρήσης έχει την δυνατότητα να επιλέξει χειροκίνητα το τάισμα, με την πρώτη επιλογή «1 ΔΟΣΗ» και η ταΐστρα θα ενεργοποιηθεί για μία φορά και να σταματήσει, ενώ οι άλλες δύο επιλογές υπάρχουν ώστε το η ενεργοποίηση της ταΐστρας να γίνεται για λόγους προσομοίωσης κάθε 5 λεπτά, επιλογή «5’ LOO» ή κάθε 10 λεπτά, επιλογή «10’ LOOP» από την στιγμή που πατηθεί η αντίστοιχη επιλογή.



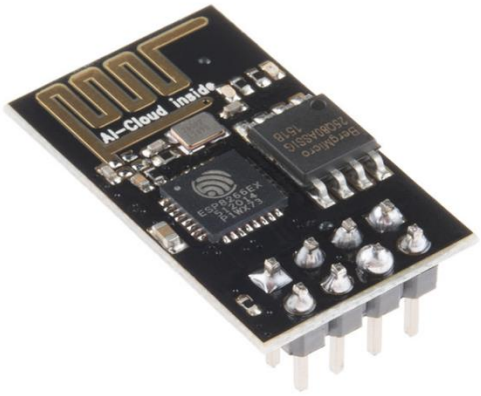
Εικόνα 32: Γραφικό περιβάλλον του app για το τάισμα

4.8 Σύνδεση μέσω Wife

Για την σύνδεση του συστήματος στο internet παρ' όλο που ο μικροελεγκτής STM32H743ZI διαθέτει την δυνατότητα ενσύρματης σύνδεσης μέσω Ethernet, έχει χρησιμοποιηθεί το WiFi module ESP-01. Η επιλογή αυτή έγινε καθώς κατά την προσπάθεια και στην δοκιμή να υλοποιηθεί η σύνδεση στο διαδίκτυο παρουσιάζονταν αρκετά προβλήματα. Έτσι η λύση ήταν να χρησιμοποιηθεί το ESP-01 το οποίο παρέχει σύνδεση στο διαδίκτυο μέσω της ενσωματωμένης κεραίας WiFi. Είναι ικανό να δρομολογήσει τα δεδομένα διαμέσου του δικτύου και υποστηρίζει ασύρματη σύνδεση στο διαδίκτυο.

Το ESP8266 αποτελεί ιδανική λύση καθώς θεωρείται ένα χαμηλού κόστους και αυτόνομο SOC με ενσωματωμένη στοίβα πρωτοκόλλων TCP/IP δίνοντας σε οποιονδήποτε μικροελεγκτή πρόσβαση στο WiFi. Είναι κατασκευασμένο από την Κινεζική εταιρεία Espressif Systems η οποία εδρεύει στην Σαγκάη. Το συγκεκριμένο module αποτελεί ιδανική επιλογή για εκπαιδευτικές κατασκευές αλλά και για project με εμπορική χρήση. Το δημοφιλέστερο μοντέλο του ESP8266 είναι το ESP-01 το οποίο κατασκευάστηκε από την Ai-Thinker. Έχει μικρές διαστάσεις 14.3mm x 24.8mm και λίγα pins **[41]**.

4.8.1 Ο ESP-01

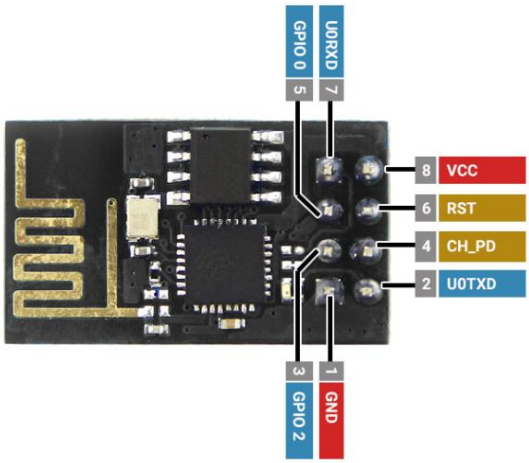


Εικόνα 33: Esp-01

Τα pin out του ESP-01 είναι:

- GND, Γείωση
- TX, Μετάδοση δεδομένων
- GPIO 2, General-purpose input/output
- CH_PD, Chip power-down
- GPIO 0, General-purpose input/output
- RST, Reset
- RX, Λήψη δεδομένων
- VCC, Τροφοδοσία 3.3V

Στην παρακάτω εικόνα εμφανίζονται ακριβώς τι αντιπροσωπεύει το κάθε pin.



Εικόνα 34: Τα pins του Esp-01

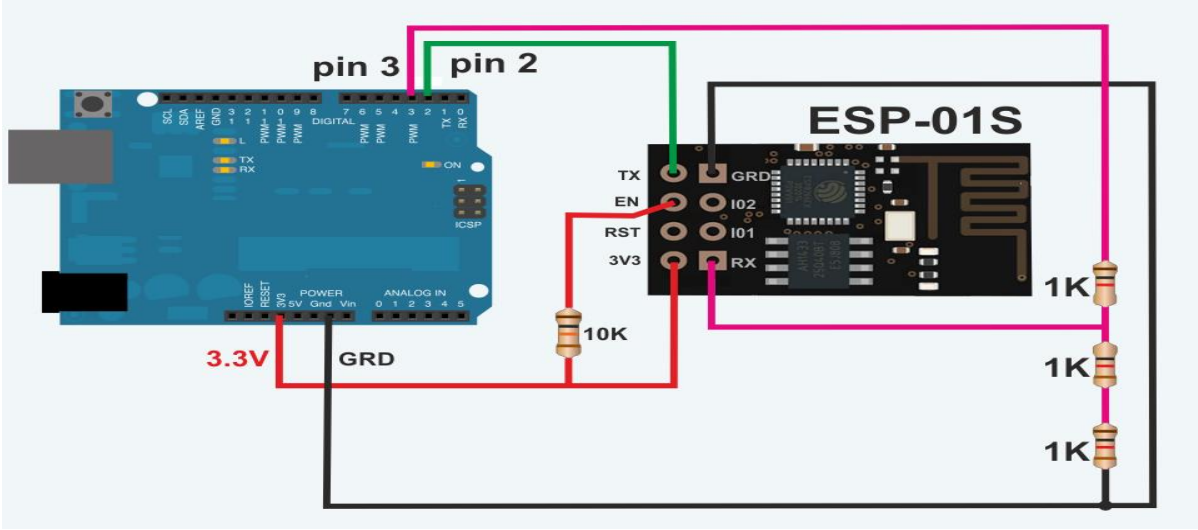
Τεχνικά χαρακτηριστικά του ESP-01 είναι:

- Ενσωματωμένη στοίβα πρωτοκόλλου TCP / IP
- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Υποστηρίζει ποικιλία κεραιών
- Ενσωματωμένα PLL , ρυθμιστές και διαχειριστές ισχύος
- CPU χαμηλής ισχύος 32-bit
- Κατανάλωση ισχύος αναμονής < 1.0 mW

- Τάση λειτουργίας 3.0-3.6 V

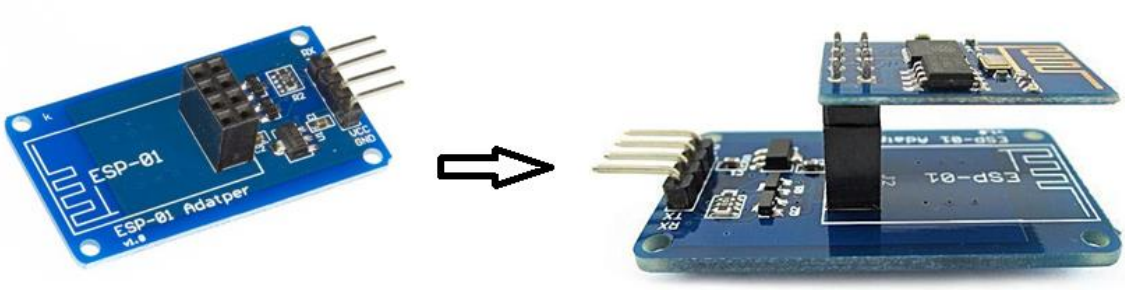
4.8.2 Συνδεσμολογία του ESP-01 στην κατασκευή

Στην κατασκευή η σύνδεση στο WiFi πραγματοποιείται μέσω του ESP01 όπως προαναφέρθηκε. Ο ESP01 τροφοδοτείται και γειώνεται πάνω από τον μικροελεγκτή STM32H743ZI με 3.3 volt και δεν δέχεται 5 volt. Για τον λόγο αυτό το pin Rx το οποίο λαμβάνει δεδομένα από τον μικροελεγκτή STM32H743ZI τοποθετήθηκε στο pin 3 του STM32H743ZI με έναν διαιρέτη τάσης ώστε να μειωθεί η τάση, ενώ το Tx για την μετάδοση στο pin 2 του μικροελεγκτή **[42]**, σχήμα 4.11.

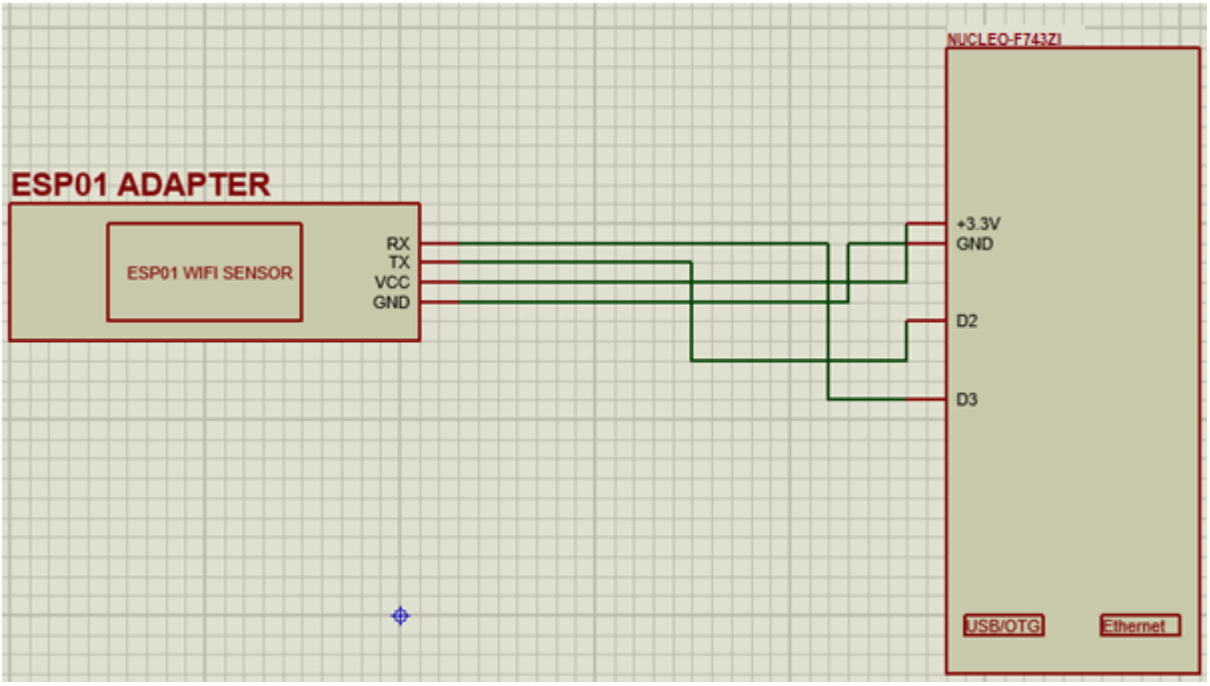


Σχήμα 4.11: Σύνδεση του ESP-01 με τον μικροελεγκτή

Για εξοικονόμηση χώρου στην κατασκευή χρησιμοποιήθηκε ο παρακάτω ESP-01 Adapter, σχήμα 4.12, ο οποίος έχει ακριβώς την ίδια λειτουργία και δίνει την δυνατότητα να τοποθετείται το ESP01 από την πάνω πλευρά στις ειδικά διαμορφωμένες υποδοχές (εικόνα 35). Ο ESP-01 Adapter έχει τάση λειτουργίας 4.5-5.5V (με ενσωματωμένο 3.3V LDO Regulator), ρεύμα λειτουργίας 0-240mA και interface logic voltage 3.5/5V (On-board level shift circuit) **[43]**.



Εικόνα 35: Esp-01 Adapter



Σχήμα 4.12: Σύνδεση του adapter ESP-01 με τον μικροελεγκτή (Proteus Design Suite)

4.9 Άλλα υλικά

Δοχείο ενυδρείου

Το πιο βασικό στοιχείο ενός ολοκληρωμένου ενυδρείου είναι φυσικά το δοχείο του (αυτό που εννοείται συνήθως με τον όρο ενυδρείο), περιεχόμενο του οποίου είναι οι ζωντανοί οργανισμοί, το νερό, τα συστήματα υποστήριξης και η διακόσμηση. Η επιλογή του κατάλληλου δοχείου του ενυδρείου είναι εκείνο που θα καλύπτει τις ιδιαίτερες ανάγκες των ψαριών τόσο ανάλογα με το είδος τους όσο και με τον αριθμό τους. Το υλικό του δοχείου που χρησιμοποιήθηκε στην παρούσα κατασκευή αποτελείται από γυαλί. Το γυάλινο δοχείο συναρμολογείται κατάλληλα, ενώνοντας τις κομμένες γυάλινες επιφάνειες με συγκολλητικό υλικό τη σιλικόνη. Το πλάτος του δοχείου είναι 16cm, το μήκος 30cm και το ύψος 24cm.

Εικόνα 36: Δοχείο ενυδρείου

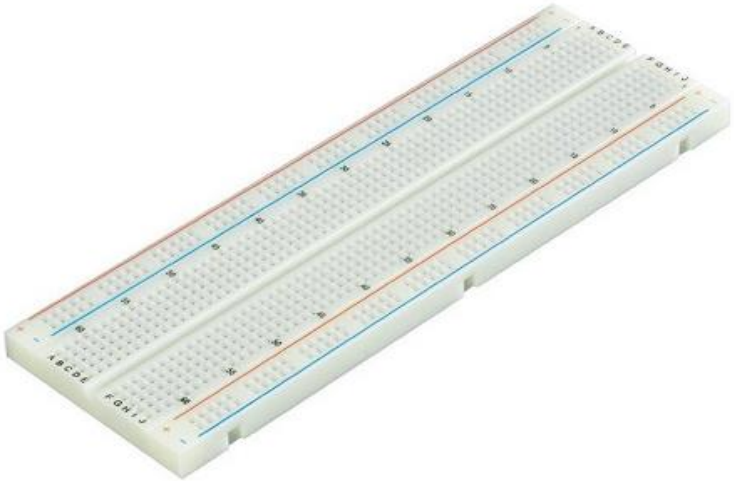
Καλώδια σύνδεσης

Για να είναι δυνατή η ένωση μεταξύ των επαφών στα κυκλώματα τα οποία χρησιμοποιήθηκαν στην κατασκευή απαραίτητη προϋπόθεση ήταν η ύπαρξη καλωδίων. Με την χρήση των καλωδίων ήταν εφικτό να συνδεθούν τα αισθητήρια με τον μικροελεγκτή και γενικότερα να πραγματοποιηθεί η ένωση όπου ήταν απαραίτητο. Τα καλώδια είναι τύπου θηλυκού-αρσενικού, με μήκος 40cm.

Εικόνα 37: Καλώδια σύνδεσης

Ράστερ

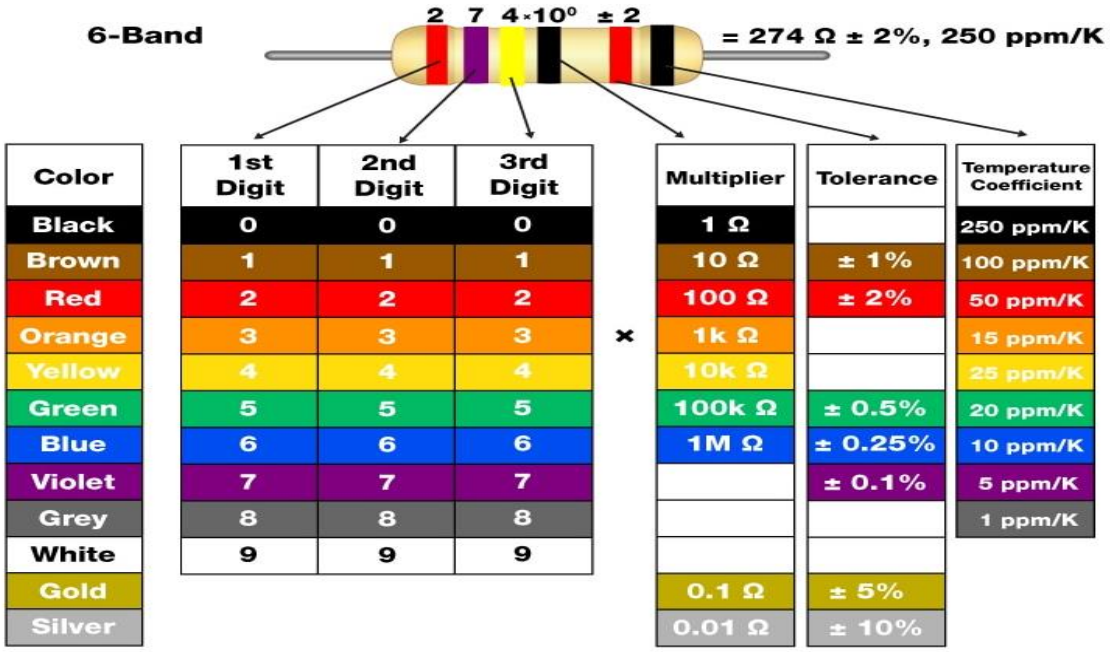
Το ράστερ είναι μία βάση κατασκευής για την παραγωγή πρωτότυπων ηλεκτρονικών κυκλωμάτων. Με το ράστερ δεν απαιτείται συγκόλληση και έτσι καθίσταται επαναχρησιμοποιήσιμο. Αυτό επίσης το καθιστά εύκολο στη χρήση για τη δημιουργία προσωρινών πρωτότυπων κυκλωμάτων, συνήθως χρησιμοποιείται για πειραματική σχεδίαση κυκλωμάτων ή για εκπαιδευτικούς λόγους. Για αυτό τον λόγο είναι εξαιρετικά δημοφιλείς ς



Εικόνα 38: Ράστερ

Αντιστάσεις

Σύμφωνα και με τον όρο resistors, οι αντιστάσεις αντιστέκονται στη ροή ηλεκτρισμού και όσο υψηλότερη είναι η τιμή της αντίστασης, τόσο περισσότερο αντιστέκεται και τόσο λιγότερο ρεύμα θα ρέει μέσα από αυτό. Τα χρώματα σε κάθε αντίσταση αντικατοπτρίζουν την τιμή της.

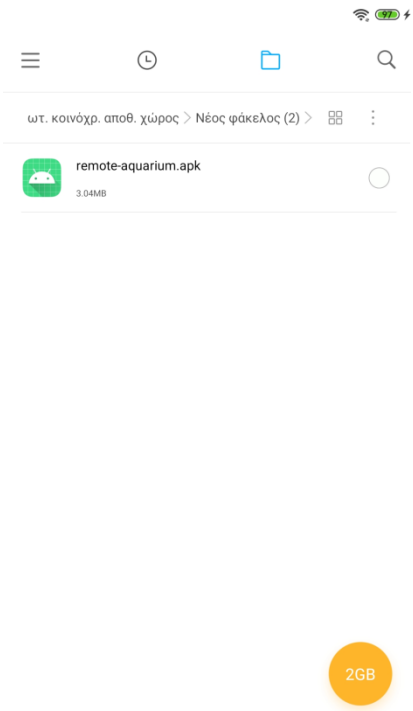


Εικόνα 39: Συμβολισμός χρωμάτων στις αντιστάσεις

4.10 Εγκατάσταση της εφαρμογής android σε συσκευή κινητής τηλεφωνίας

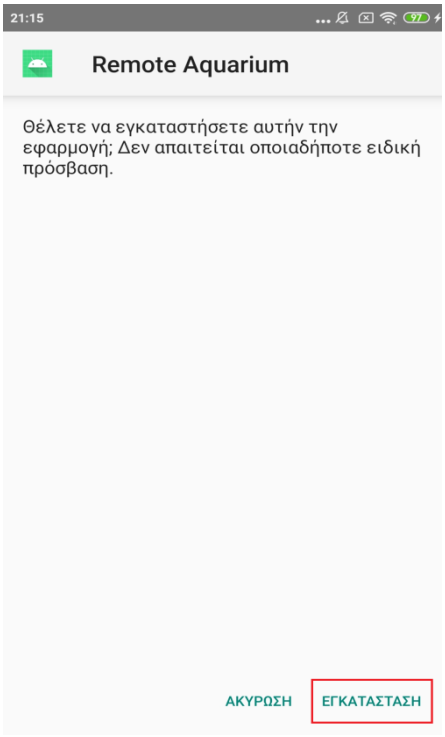
Η ολοκληρωμένη εφαρμογή android η οποία υλοποιήθηκε για τις ανάγκες της παρούσας Δ.Ε. απαιτεί την ύπαρξη μιας συσκευής κινητής τηλεφωνίας με λειτουργικό android και πιο συγκεκριμένα απαιτεί η συσκευή smart phone να διαθέτει μεταγενέστερη έκδοση λειτουργικού από android 7.0 και μετά.

Έπειτα για την εγκατάσταση της εφαρμογής στην συσκευή χρειάζεται η λήψη του αρχείου .apk στον αποθηκευτικό χώρο της συσκευής (εικόνα 41). Το apk ουσιαστικά είναι ένα συμπιεσμένο αρχείο όπως για παράδειγμα ένα αρχείο zip, που περιέχει μέσα όλα τα απαραίτητα αρχεία και πληροφορίες για να εγκατασταθεί μία εφαρμογή σε ένα smart phone με λειτουργικό android **[44]**.



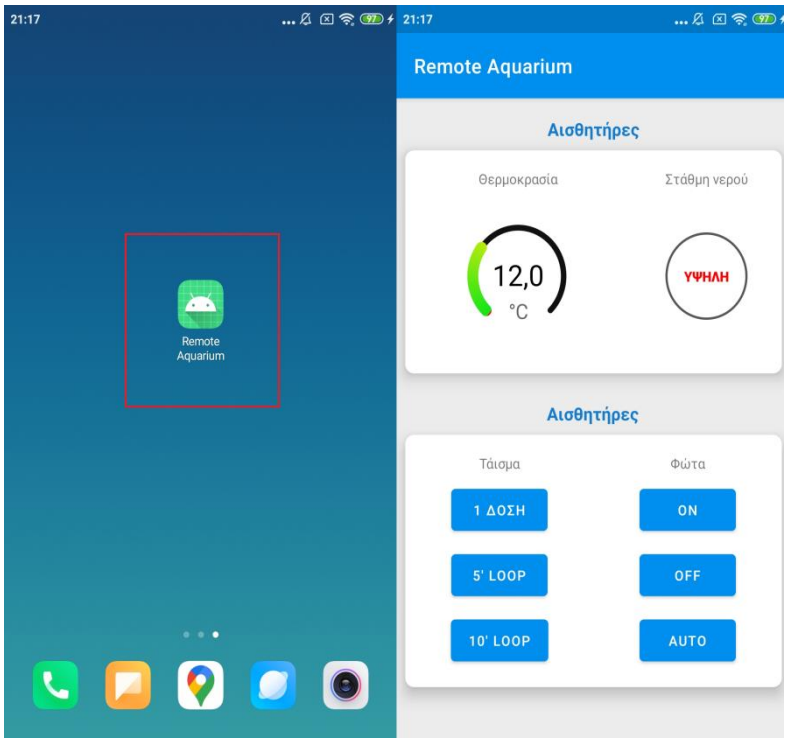
Εικόνα 40: Αποθηκευτικό χώρος του smart phone

Κατά την αποθήκευση του αρχείου .apk, επιλέγοντας το αρχείο αυτό από την συσκευή εμφανίζεται μία ειδοποίηση για εγκατάσταση της εφαρμογής στην συσκευή. Όπως φαίνεται και στην παρακάτω εικόνα με την επιλογή «ΕΓΚΑΤΑΣΤΑΣΗ» το κινητό τηλέφωνο θα εγκαταστήσει την εφαρμογή ώστε να μπορεί να χρησιμοποιηθεί.



Εικόνα 41: Εγκατάσταση .apk αρχείου

Όταν ολοκληρωθεί η εγκατάσταση της εφαρμογής θα εμφανιστεί το εικονίδιο του app στην επιφάνεια εργασίας του smart phone. Τέλος η εφαρμογή είναι έτοιμη προς χρήση με το γραφικό περιβάλλον να αποτελείται από τις αντίστοιχες ενδείξεις και επιλογές προς τον χρήστη.



Εικόνα 42: Εικονίδιο και γραφικό περιβάλλον της εφαρμογής

4.11 Επίλογος

Η επιλογή των κατάλληλων μικροελεγκτών, αισθητήρων και ηλεκτρονικών εργαλείων απαιτεί αρχικά να οριστούν οι προδιαγραφές όσον αφορά την λειτουργία και τους στόχους τους οποίους επιθυμούμε να πετύχουμε μέσω του συστήματος. Η επιλογή στην αυτοματοποίηση συγκεκριμένων λειτουργιών ενός συστήματος έχει σαν αποτέλεσμα την έρευνα των κατάλληλων αισθητήρων και η επιλογή των κατάλληλων αισθητήρων έχει σαν αποτέλεσμα την επιλογή των κατάλληλων ενεργοποιητών. Με αυτή την διαδικασία επιτυγχάνεται η δημιουργία μιας έξυπνης κατασκευής η οποία δεν απαιτεί την ανθρώπινη παρουσία.

Κεφάλαιο 5ο: Ανάπτυξη λογισμικού

5.1 Εισαγωγή

Όπως είναι φυσικό κάθε σύστημα το οποίο εκτελεί μία ή περισσότερες ενέργειες και αποτελείται από ένα κεντρικό σύστημα ελέγχου όπως έναν μικροελεγκτή θα πρέπει να προγραμματιστεί ώστε να εκτελέσει την επιθυμητή ενέργεια ή τις επιθυμητές ενέργειες. Έτσι για τον προγραμματισμό του μικροελεγκτή απαιτείται κάποιο ολοκληρωμένο προγραμματιστικό περιβάλλον στο οποίο με την κατάλληλη χρήση εντολών και γλώσσα προγραμματισμού να αναπτυχθεί ένα λογισμικό που θα είναι ικανό να εκτελέσει προκαθορισμένες εντολές. Σκοπός επομένως του κεφαλαίου αυτού είναι η παρουσίαση και η αναφορά στα ηλεκτρονικά εργαλεία που είναι απαραίτητα να χρησιμοποιηθούν για την ανάπτυξη και σύνταξη του firmware τόσο του μικροελεγκτή όσο και της ολοκληρωμένης

εφαρμογής android. Επίσης αναλύεται η δομή των αλγορίθμων με μορφή διαγραμμάτων έτσι ώστε ο αναγνώστης της παρούσας Δ.Ε. να κατανοήσει καλύτερα την δομή του λογισμικού καθώς ο πλήρης κώδικας βρίσκεται στα παραρτήματα στο τέλος της εργασίας.

5.2 Android Studio

Στα πλαίσια υλοποίησης της συγκεκριμένης Δ.Ε. και συγκεκριμένα για την υλοποίηση της εφαρμογής android χρησιμοποιήθηκε το Andoid Studio **[45]** το οποίο αποτελεί από τον Δεκέμβριο του 2014 ένα ολοκληρωμένο προγραμματιστικό περιβάλλον (IDE-Integrated Development Environment) για ανάπτυξη εφαρμογών στο λειτουργικό android. Με την άδεια Apache License 2.0 το Android Studio είναι διαθέσιμο ελεύθερα.

Εικόνα 43: Λογότυπο του Android Studio

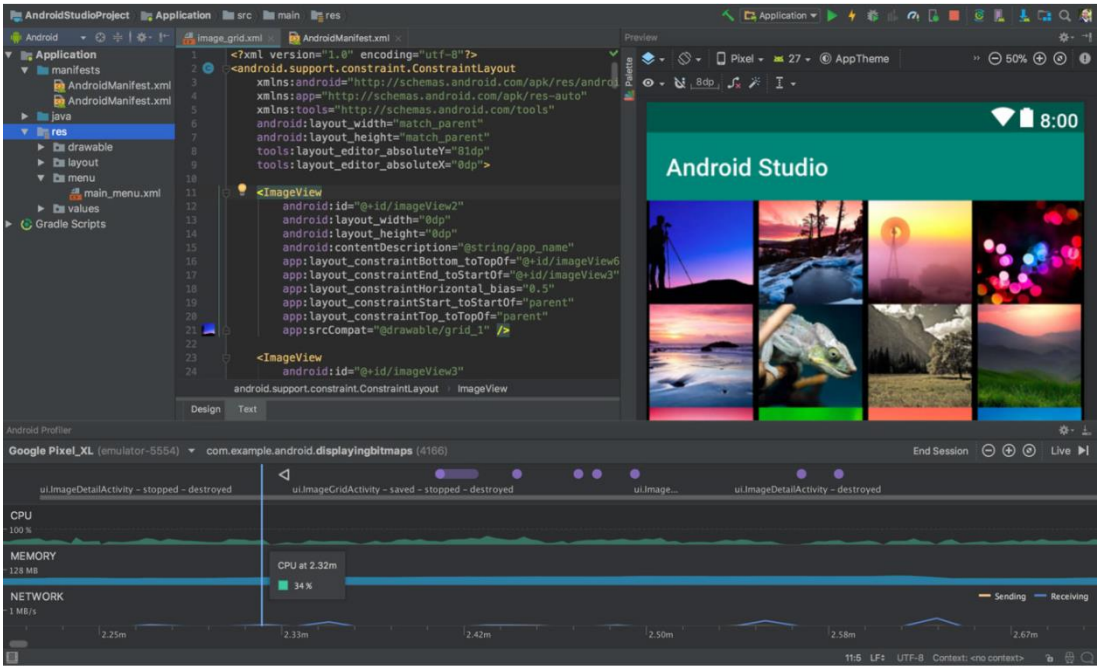
Η πρώτη πρώιμη έκδοση του Android Studio για προεπισκόπηση ήταν η έκδοση 0.1 και ανακοινώθηκε τον Μάιο του 2013 στο συνέδριο Google I/O από την Google Product Manager, Katherine Chou, έπειτα συνεχίστηκε με την δοκιμαστική έκδοση 0.8 τον Ιούνιο του 2014 και η πρώτη σταθερή έκδοση εμφανίστηκε το Δεκέμβριο του 2014 (έκδοση 1.0). Βασισμένο στο λογισμικό της JetBrains' IntelliJ IDEA, το Android Studio είναι σχεδιασμένο αποκλειστικά για προγραμματισμό android και είναι διαθέσιμο για windows, macOS και linux αντικαθιστώντας τα Eclipse Android Development Tools (ADT) ως το κύριο IDE της Google για ανάπτυξη εφαρμογών android.

Όπως και ο προκάτοχός του (Eclipse ADT) το Android Studio λειτουργεί με τον ίδιο τρόπο ωστόσο προσφέρει περισσότερες δυνατότητες στον χρήστη, καθιστώντας με αυτό τον τρόπο την δημιουργία μιας εφαρμογής κινητού τηλεφώνου ευκολότερη και γρηγορότερη. Η ίδια η Google έδωσε βαρύτητα στην ανάπτυξη και τη βελτίωση του Android Studio μετά από ένα σημείο, προτείνοντάς το ως το καταλληλότερο IDE για ανάπτυξη εφαρμογών σε android.

Βασικά χαρακτηριστικά του Android Studio είναι **[46]**:

- Ζωντανή εμφάνιση της διάταξης, γίνεται αναπαράσταση με οπτική διάταξη του κώδικα, δηλαδή πραγματοποιείται προσομοίωση της εφαρμογής όπως ακριβώς θα εκτελούνταν η εφαρμογή εάν ήταν εγκατεστημένη σε μία συσκευή.
- Ένα ευέλικτο Cradle-based σύστημα κατασκευής.
- Ένα γρήγορο και με πολλές λειτουργίες εξομοιωτή (emulator).
- Γρήγορες διορθώσεις στον κώδικα της εφαρμογής.
- Ειδικό lint εργαλείο το οποίο ελέγχει τον κώδικα για σφάλματα, παρατυπίες, ασυμβατότητα εκδόσεων λογισμικού.
- Όταν γίνονται αλλαγές στον κώδικα το Android Studio επιτρέπει την άμεση εκτέλεσή του χωρίς να δημιουργεί καινούριο εκτελέσιμο apk αρχείο (αρχείο .apk είναι τα εκτελέσιμα αρχεία εγκατάστασης εφαρμογών android).
- Διάφορα templates-πρότυπα σελίδων, όπως σελίδες χαρτών, σελίδες εισόδου (log in).
- Ενσωματωμένη υποστήριξη για την πλατφόρμα Cloud της Google.

Γραφικό περιβάλλον του Android Studio παρατίθεται στην παρακάτω εικόνα:



Εικόνα 44: Γραφικό περιβάλλον του Android Studio

5.3 Microsoft Visual Studio

Για την ανάπτυξη του λογισμικού στον μικροελεγκτή χρησιμοποιήθηκε το Microsoft Visual Studio **[47]** με το πρόσθετο PlatformIO. Το Microsoft Visual Studio αποτελεί ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) από την εταιρεία Microsoft. Χρησιμοποιείται για την ανάπτυξη προγραμμάτων ηλεκτρονικών υπολογιστών για τα Microsoft Windows, καθώς και για ιστοσελίδες, διαδικτυακές εφαρμογές και υπηρεσίες web. Χρησιμοποιεί πλατφόρμες ανάπτυξης λογισμικού της Microsoft, όπως Windows API, Windows Forms, Windows Presentation Foundation, Windows Store και το Microsoft Silverlight.

Εικόνα 45: Λογότυπο του Visual Studio

Το Visual Studio περιλαμβάνει έναν επεξεργαστή κώδικα που υποστηρίζει το IntelliSense, ένα component αυτόματης συμπλήρωσης κώδικα, καθώς επίσης και code refactoring. Το ενσωματωμένο πρόγραμμα εντοπισμού σφαλμάτων λειτουργεί ως ένα πρόγραμμα εντοπισμού σφαλμάτων πηγαίου κώδικα (source-level debugger) και ως ένα πρόγραμμα εντοπισμού σφαλμάτων μηχανής (machine-level debugger). Κάποια άλλα ενσωματωμένα εργαλεία που έχει αποτελούν η σχεδίαση φόρμας για τη δημιουργία εφαρμογών GUI, η σχεδίαση ιστοσελίδων, η σχεδίαση κλάσεων και η σχεδίαση σχήματος βάσης δεδομένων. Δέχεται plug-ins που ενισχύουν τη λειτουργικότητα σχεδόν σε κάθε επίπεδο, προσθέτοντας υποστήριξη για συστήματα πηγαίου κώδικα (source-control systems) όπως η Subversion, προσθήκη νέων toolsets όπως editors και visual designers για domain-specific γλώσσες προγραμματισμού.

Επίσης το Visual Studio υποστηρίζει διάφορες γλώσσες προγραμματισμού και παρέχει code editor και debugger, σχεδόν σε οποιαδήποτε γλώσσα προγραμματισμού. Ενσωματωμένες γλώσσες αποτελούν C, C++ και C++/CLI μέσω της Visual C++, VB.NET μέσω της Visual Basic.NET, C# μέσω Visual C #, και F#. Υποστήριξη για άλλες γλώσσες, όπως η M, η Python και η Ruby, μεταξύ άλλων, είναι διαθέσιμες μέσω των υπηρεσιών εγκατάστασης γλωσσών που παρέχεται, για κάθε μία ξεχωριστά. Υποστηρίζει επίσης XML/XSLT, HTML/XHTML, JavaScript και CSS. Η Microsoft παρέχει εκδόσεις του Visual Studio χωρίς κόστος.

Το platformIO IDE είναι ένα πρόσθετο περιβάλλον ανάπτυξης λογισμικού στο Visual Studio. Μπορεί να ληφθεί πλέον μέσα από το Visual Studio και αποτελεί ένα ευχάριστο προγραμματιστικό περιβάλλον με διαφορετικά χρώματα και ικανοποιητική στοίχιση. Διαθέτει ενσωματωμένο τερματικό με το εργαλείο PlatformIO CLI αλλά και ισχυρό σύστημα παρακολούθησης σειριακών θυρών. Επίσης διαθέτει σύστημα δημιουργίας πολλαπλών πλατφορμών χωρίς εξωτερικές εξαρτήσεις από το λογισμικό λειτουργικού του συστήματος: 800 πίνακες, 35 πλατφόρμες ανάπτυξης, 20 πλαίσια. Ακόμα διαθέτει εντοπισμό σφαλμάτων, απομακρυσμένη ανάπτυξη, δοκιμή μονάδας, για C/C++ έξυπνη ολοκλήρωση κώδικα, διαχείριση βιβλιοθήκης για τις εκατοντάδες δημοφιλείς βιβλιοθήκες και υποστήριξη θεμάτων με σκούρα και ανοιχτά χρώματα **[48][49]**.

Εικόνα 46: Λογότυπο του PlatformIO

Το γραφικό περιβάλλον του PlatformIO είναι:

Εικόνα 47: Γραφικό περιβάλλον του PlatformIO

5.4 Ανάπτυξη λογισμικού στον μικροελεγκτή ST32H743ZI (κυρίως πρόγραμμα)

Ο μικροελεγκτής STM32H743ZI μπορεί να προγραμματιστεί ώστε να εκτελέσει τις ενέργειες που απαιτούνται για τις ανάγκες του συγκεκριμένου συστήματος σε γλώσσα προγραμματισμού C. Κατά την έναρξη του firmware για την εκτέλεση των εντολών που πραγματοποιούνται στον μικροελεγκτή η

πρώτη ενέργεια που γίνεται είναι η δήλωση των απαραίτητων βιβλιοθηκών (σημείο 1, σχήμα 5.1). Συγκεκριμένα στον αλγόριθμο που υλοποιήθηκε δηλώνονται πέντε βιβλιοθήκες οι οποίες είναι:

- **«OneWire.H»**: Η συγκεκριμένη βιβλιοθήκη δίνει την δυνατότητα σε έναν αισθητήρα να λειτουργήσει με ένα μόνο καλώδιο, δηλώνοντας απλά τον ακροδέκτη του μικροελεγκτή που θα δεσμευτεί για τον αισθητήρα αυτόν. Το σήμα με τις πληροφορίες των μετρήσεων μπορεί να σταλθεί από και προς τον αισθητήρα μέσω μιας διεπαφής με χρήση ενός καλωδίου, έτσι ώστε μόνο ένα καλώδιο πληροφορίας, η γείωση και η τροφοδοσία του αισθητήρα να χρειάζεται να είναι συνδεδεμένα στον κεντρικό μικροελεγκτή. Αυτό αποτελεί έναν εύκολο και σίγουρο τρόπο σύνδεσης του αισθητήρα με τον μικροελεγκτή **[50]**.
- **«DallasTemperature.h»**: Η βιβλιοθήκη αυτή υποστηρίζει ορισμένους αισθητήρες θερμοκρασίας, όπως τον DS18S20, DS1822, DS1820 και τον MAX31820 αλλά και τον DS18B20 που έχει χρησιμοποιηθεί και αναφερθεί σε προηγούμενο κεφάλαιο στο συγκεκριμένο σύστημα για την Δ.Ε **[51]**.
- **«SoftwateSerial.h»**: Η βιβλιοθήκη «SoftwareSerial.h» έχει αναπτυχθεί για να επιτρέπει σειριακή επικοινωνία σε άλλα ψηφιακά pins. Επίσης δίνει τη δυνατότητα να διαχειρίζεται πολλαπλές σειριακές συνδέσεις μέχρι 115200bps **[52]**.
- **«TinyGsmClient.h»**: Η «TinyGsmClient.h» είναι μία βιβλιοθήκη για GPRS modules και υποστηρίζει μονάδες GSM, LTE και Wifi με διεπαφές AT **[53]**.
- **«PubSubClient.h»**: Η οποία αποτελεί μία βιβλιοθήκη client για απλή δημοσίευση/εγγραφή μηνυμάτων με ένα διακομιστή που υποστηρίζει το πρωτόκολλο MQTT **[54]**.

Έπειτα στο σημείο 2 από το διάγραμμα ροής (σχήμα 5.1) δηλώνονται τα pins που χρησιμοποιήθηκαν και δεσμεύτηκαν από τον μικροελεγκτή STM32 για την σύνδεση των αισθητήρων και του ESP8266 και ορίζονται οι μεταβλητές. Στην συνέχεια πραγματοποιείται η αρχικοποίηση του συστήματος, των μεταβλητών και των αισθητηρίων (σημείο 3, σχήμα 5.1). Με την έννοια αρχικοποίηση αναφερόμαστε στην αρχική κατάσταση την οποία θα βρίσκεται για παράδειγμα ο αισθητήρας στάθμης που χρησιμοποιείται στην κατασκευή κατά την έναρξη της εκτέλεσης του κώδικα.

Στα σημεία 4 και 5 του διαγράμματος (σχήμα 5.1) πραγματοποιείται η σύνδεση και ο έλεγχος για την σύνδεση με το modem. Τον ρόλο αυτό στην συγκεκριμένη περίπτωση στην κατασκευή τον έχει ο ESP8266. Στην περίπτωση όπου υπάρχει σύνδεση με το modem γίνεται η δοκιμή του SSID και του password για την σύνδεση με το WiFi (σημείο 6, σχήμα 5.1), διαφορετικά ο αλγόριθμος επιστρέφει στην διαδικασία του ελέγχου σύνδεσης με το modem ώσπου να συνδεθεί, σημείο 5 (σχήμα 5.1). Η διαδικασία αυτή επαναλαμβάνεται μέχρι να υπάρξει η σύνδεση με το modem με κάποια χρονική καθυστέρηση λίγων δευτερολέπτων.

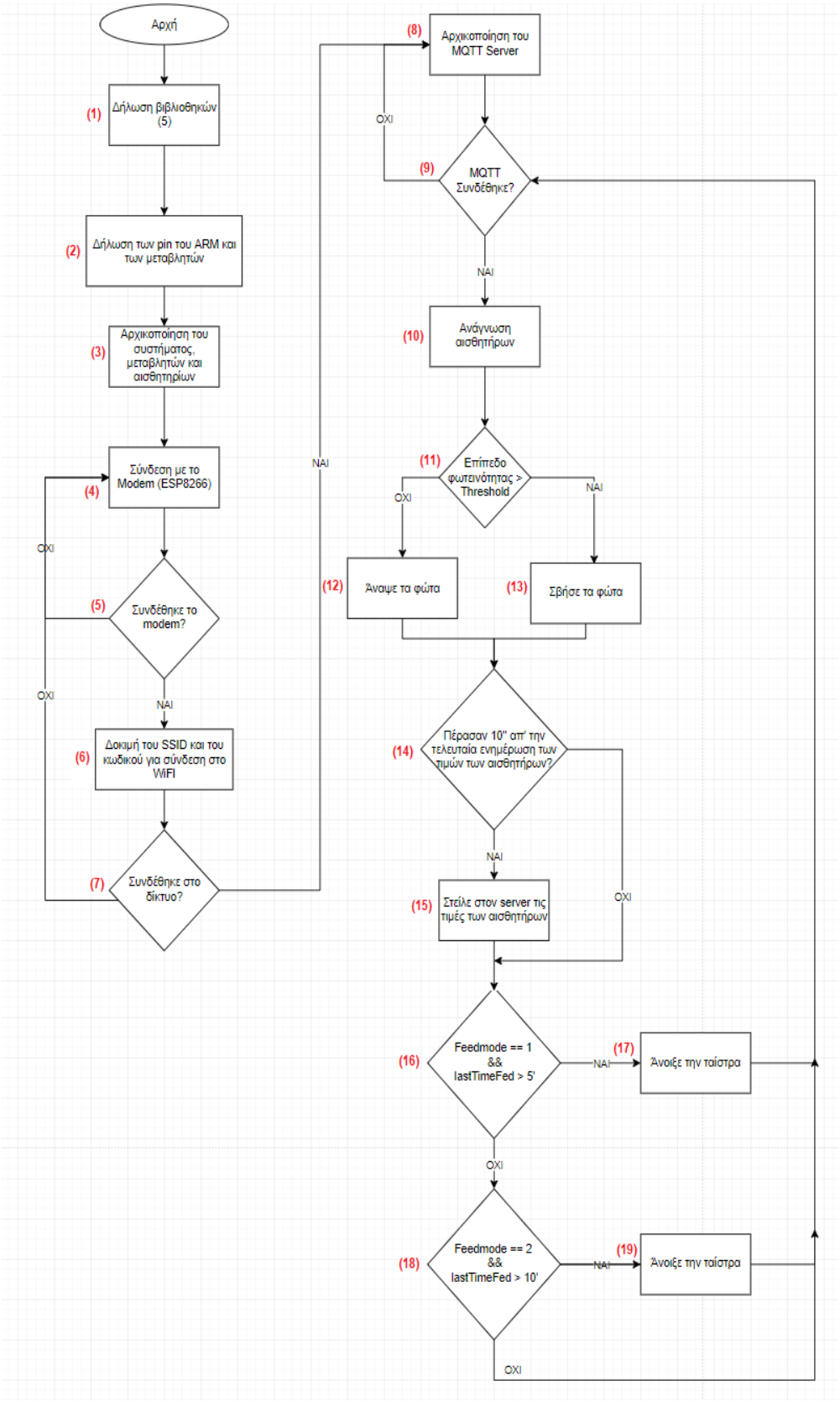
Στην πορεία στο σημείο 7 (σχήμα 5.1) πραγματοποιείται ακόμα ένας έλεγχος ώστε να εξακριβωθεί η σύνδεση στο δίκτυο. Αν η σύνδεση στο δίκτυο πραγματοποιηθεί με επιτυχία τότε στο σημείο 8 του διαγράμματος (σχήμα 5.1) γίνεται η αρχικοποίηση του MQTT Server(δηλώνεται η διεύθυνση και η πόρτα του server) διαφορετικά επιστρέφει πάλι στην διαδικασία ελέγχου σύνδεσης με το modem, σημείο 5 (σχήμα 5.1). Στο κομμάτι αυτό γίνεται έλεγχος για την επιτυχή σύνδεση με τον server, σημείο 9 (σχήμα 5.1). Στην περίπτωση της επιτυχής σύνδεσης με τον server πραγματοποιείται η ανάγνωση των αισθητήρων της στάθμης του νερού και του αισθητήρα φωτός, σημείο 10 (σχήμα 5.1), λαμβάνεται με άλλα λόγια η τιμή που έχει κάθε αισθητήρας την στιγμή της σύνδεσης με τον server, διαφορετικά γίνεται προσπάθεια επανασύνδεσης με τον MQTT Server, σημείο 9 του διαγράμματος ροής (σχήμα 5.1).

Συνεχίζοντας πραγματοποιείται έλεγχος του επιπέδου φωτεινότητας συγκριτικά με την μεταβλητή «Threshold» που έχει οριστεί με τιμή 550, σημείο 11 (σχήμα 5.1). Εάν το επίπεδο φωτεινότητας είναι μεγαλύτερο από την μεταβλητή «Threshold» τότε δίνεται η εντολή σβήσε την led ταινία, σημείο 13

(σχήμα 5.1) διαφορετικά αν είναι μικρότερο το επίπεδο φωτεινότητας άναψε την led ταινία, σημείο 12 (σχήμα 5.1).

Έπειτα γίνεται έλεγχος εάν έχουν περάσει δέκα δευτερόλεπτα από την τελευταία ενημέρωση των τιμών των αισθητήρων, σημείο 14 (σχήμα 5.1) και αυτός ο έλεγχος γίνεται ώστε να υπάρχει μια «υγιή» χρονική καθυστέρηση για την αποστολή των νέων τιμών της θερμοκρασίας και της στάθμης νερού στον server. Στην περίπτωση όπου έχουν περάσει τα δέκα δευτερόλεπτα γίνεται η αποστολή των τιμών των αισθητήρων στον server, σημείο 15 (σχήμα 5.1), αλλιώς παραλείπεται το βήμα αυτό και ο αλγόριθμος πάει στο επόμενο βήμα.

Στο επόμενο βήμα το πρόγραμμα που έχει δημιουργηθεί ελέγχει εάν η μεταβλητή «Feedmode» είναι ίσο με ένα και εάν η μεταβλητή «lastTimeFed» είναι μεγαλύτερη από τα πέντε λεπτά, σημείο 16 του διαγράμματος (σχήμα 5.1). Η μεταβλητή «Feedmode» ίσο με ένα αντιπροσωπεύει την κατάσταση όπου η διαδικασία του ταΐσματος γίνεται κάθε πέντε λεπτά ενώ ίσο με δύο την διαδικασία που το τάισμα γίνεται κάθε δέκα λεπτά. Επομένως στην περίπτωση που η «Feedmode»=1 και έχουν περάσει τα πέντε λεπτά τότε ενεργοποιείται η ταΐστρα, σημείο 17 (σχήμα 5.1) και το πρόγραμμα ολοκληρώνεται επιστρέφοντας στην διαδικασία ελέγχου σύνδεσης με τον MQTT Server, σημείο 9 (σχήμα 5.1). Στην περίπτωση που η «Feedmode»=2 και η «lastTimeFed» είναι μεγαλύτερη από τα δέκα λεπτά, σημείο 18 του διαγράμματος (σχήμα 5.1) δηλαδή έχουν περάσει δέκα λεπτά από την τελευταία φορά που ενεργοποιήθηκε η ταΐστρα τότε ενεργοποιείται η ταΐστρα και πάλι το πρόγραμμα ολοκληρώνεται.



Σχήμα 5.1: Διάγραμμα ροής του λογισμικού του μικροελεγκτή (app.diagrams.net)

5.4.1 Interrupt στο κυρίως πρόγραμμα του μικροελεγκτή ST32H743ZI

Στο πρόγραμμα που έχει δημιουργηθεί «τρέχει» ένα interrupt το οποίο αναμένει να λάβει κάποιο μήνυμα από τον server. Μόλις λάβει ένα νέο μήνυμα από τον server σε οποιοδήποτε σημείο βρίσκεται το κυρίως πρόγραμμα διακόπτεται ώστε να εκτελεστεί το interrupt το οποίο θα αναλυθεί παρακάτω με το κατάλληλο διάγραμμα ροής. Έπειτα κατά την ολοκλήρωση του interrupt το πρόγραμμα συνεχίζει την εκτέλεση του κύριου προγράμματος ακριβώς από το σημείο το οποίο είχε σταματήσει ώστε να εκτελεστεί το interrupt. Αυτή η διαδικασία πραγματοποιείται σε οποιοδήποτε σημείο του κυρίως προγράμματος και στην συνέχεια εκτελείται από το σημείο διακοπής μέχρι την ολοκλήρωσή του. Όπως φαίνεται και στην παρακάτω εικόνα αλλά και όπως προαναφέρθηκε παραπάνω αυτός είναι ο τρόπος με τον οποίο λειτουργεί ένα interrupt σε ένα κυρίως πρόγραμμα **[55]**.

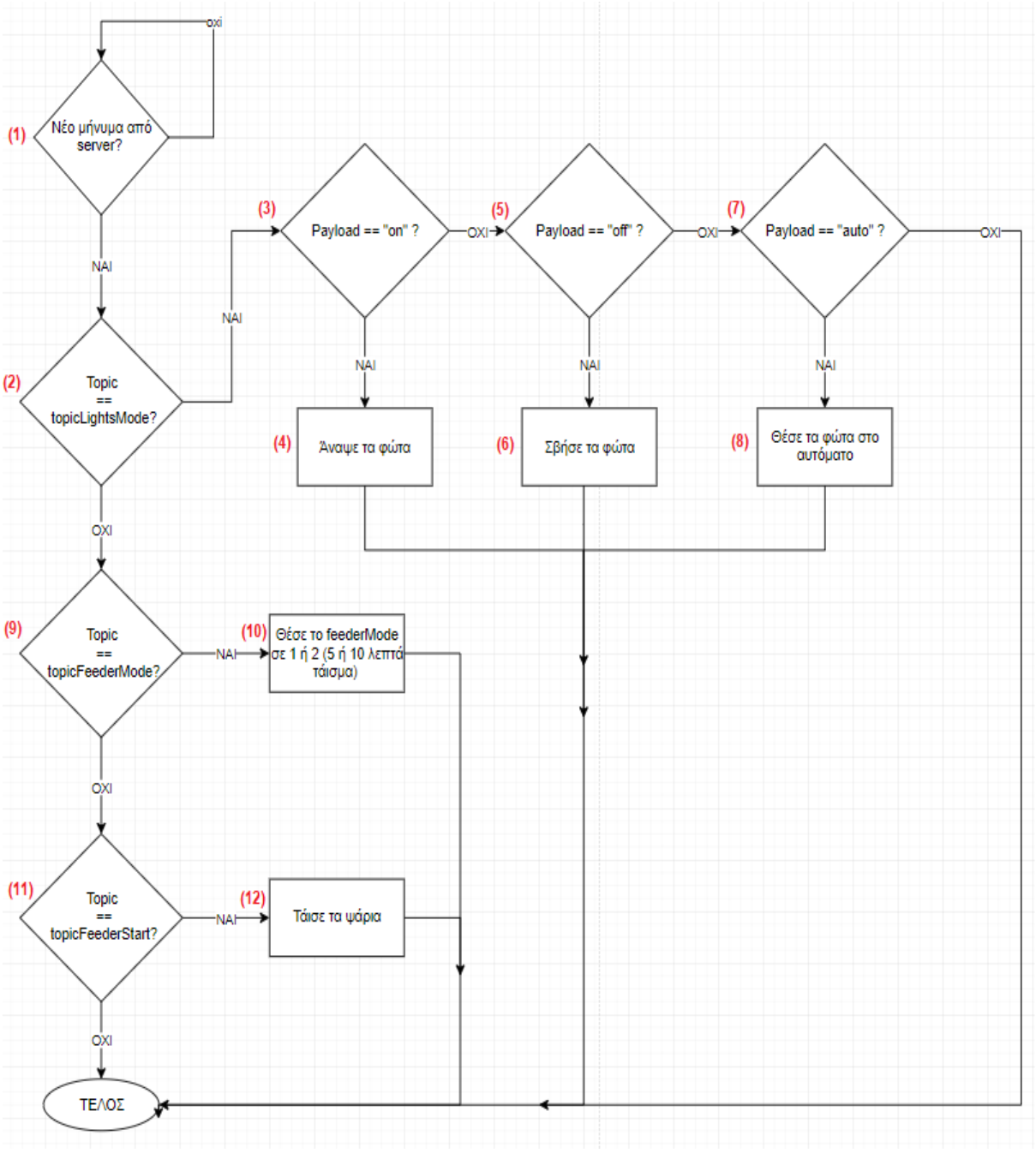
Εικόνα 48: Interrupt (Διακοπή)

Επομένως μόλις ληφθεί ένα νέο μήνυμα, στο σημείο 1 του διαγράμματος (σχήμα 5.2), από τον server πραγματοποιείται ο έλεγχος στα topic. Στην περίπτωση που το topic είναι ίσο με το «topicLightsMode», σημείο 2 (σχήμα 5.2), το οποίο αντιπροσωπεύει την κατάσταση της ενεργοποίησης της led ταινίας (αυτόματη ή χειροκίνητη ενεργοποίηση) τότε ελέγχει το περιεχόμενο του payload. Στην περίπτωση που το payload είναι ίσο με «on» τότε ενεργοποιείται η led ταινία, σημείο 3 και 4 του διαγράμματος (σχήμα 5.2), εάν είναι ίσο με «off» τότε απενεργοποιείται η led ταινία, σημεία 5 και 6 του διαγράμματος (σχήμα 5.2), εάν το payload είναι ίσο με «auto» θέτει στην αυτόματη λειτουργία ενεργοποίησης ή απενεργοποίησης της led ταινίας, σημείο 7 και 8 του διαγράμματος (σχήμα 5.2). Έπειτα ολοκληρώνεται η διαδικασία και επιστρέφει αναμένοντας νέο μήνυμα από τον server, σημείο 1 (σχήμα 5.2) και το κυρίως πρόγραμμα το οποίο αναλύθηκε με το προηγούμενο διάγραμμα ροής συνεχίζει να εκτελείται από το σημείο το οποίο είχε διακοπεί.

Στην περίπτωση που το topic είναι ίσο με το «topicFeederMode», σημείο 9 (σχήμα 5.2) το οποίο αντιπροσωπεύει την κατάσταση ενεργοποίησης της ταΐστρας (κάθε πέντε ή δέκα λεπτά ενεργοποίησης) τότε θέτει την μεταβλητή «feederMode» στην κατάσταση ένα ή δύο, δηλαδή στην κατάσταση όπου το τάισμα γίνεται κάθε πέντε λεπτά ή κάθε δέκα λεπτά, σημείο 10 (σχήμα 5.2). Κατά

την ολοκλήρωση αυτής της διαδικασίας ολοκληρώνεται πάλι το interrupt και το πρόγραμμα επιστρέφει αναμένοντας πάλι ένα νέο μήνυμα από τον server, σημείο 1 (σχήμα 5.2) και το κυρίως πρόγραμμα εκτελείται ακριβώς από το σημείο εκείνο που είχε διακοπεί για να εκτελέσει το συγκεκριμένο interrupt.

Τέλος στην περίπτωση που το topic είναι ίσο με το «topicFeederStart», σημείο 11 (σχήμα 5.2), το οποίο αντιπροσωπεύει την ενεργοποίηση της ταΐστρας απευθείας, τότε πραγματοποιείται η ενεργοποίηση της ταΐστρας, σημείο 12 (σχήμα 5.2), χωρίς να προηγηθεί κάποια άλλη ενέργεια. Μόλις η ταΐστρα ενεργοποιηθεί και εκτελεστεί η εντολή αυτή τότε και πάλι ολοκληρώνεται το interrupt και το πρόγραμμα επιστρέφει αναμένοντας ένα νέο μήνυμα από τον server, σημείο 1 (σχήμα 5.2) και το κυρίως πρόγραμμα εκτελείται από το σημείο που είχε διακοπεί.

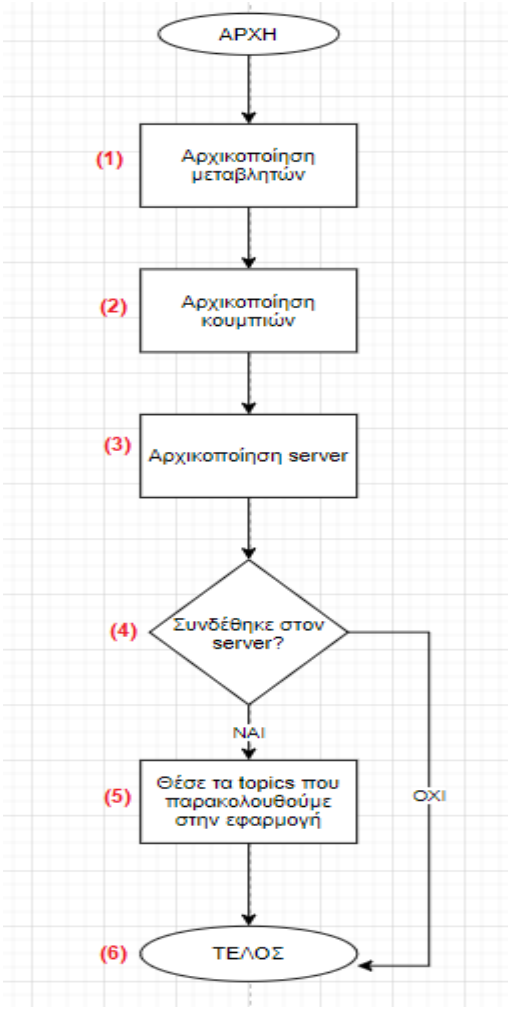


Σχήμα 5.2: Διάγραμμα ροής του λογισμικού του μικροελεγκτή (interrupt) (app.diagrams.net)

5.5 Ανάπτυξη εφαρμογής android (κυρίως πρόγραμμα)

Η ανάπτυξη λογισμικού για την εφαρμογή android πραγματοποιήθηκε σε γλώσσα προγραμματισμού Java. Κατά την έναρξη του κώδικα για την εκτέλεση των εντολών που πραγματοποιούνται στο app η πρώτη ενέργεια που γίνεται είναι η αρχικοποίηση των μεταβλητών, σημείο 1του διαγράμματος ροής (σχήμα 5.3). Έπειτα γίνεται η αρχικοποίηση των κουμπιών που χρησιμοποιήθηκαν στο app, σημείο 2 (σχήμα 5.3). Όπως αναφέρθηκε και αναλύθηκε το app το οποίο έχει δημιουργηθεί έχει επιλογές-κουμπιά προς τον χρήστη για τον έλεγχο στο κομμάτι της led ταινίας όπως για παράδειγμα την επιλογή «ON», «OFF» για χειροκίνητη ενεργοποίηση και απενεργοποίηση αλλά και το κουμπί «AUTO» για αυτόματη ενεργοποίηση/απενεργοποίηση της led ταινίας. Αντίστοιχα και στο κομμάτι για την ενεργοποίηση της ταΐστρας υπάρχουν τα κουμπιά ως επιλογή για «1 ΔΟΣΗ», «5΄ LOOP» και «10΄ LOOP».

Στην συνέχεια γίνεται η αρχικοποίηση του MQTT Server, σημείο 3 (σχήμα 5.3), δηλώνεται δηλαδή η διεύθυνση και η πόρτα του server και μετά γίνεται ο έλεγχος για σύνδεση στον server. Εάν η σύνδεση στον server είναι επιτυχής τότε το πρόγραμμα θέτει τα topics που παρακολουθούνται στην εφαρμογή, σημείο 5 (σχήμα 5.3) διαφορετικά το κυρίως πρόγραμμα ολοκληρώνεται, σημείο 6 (σχήμα 5.3).

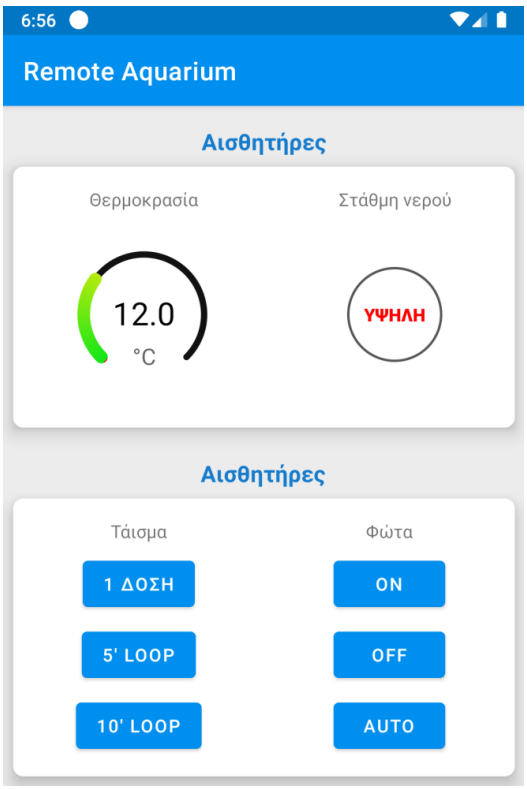


Σχήμα 5.3: Διάγραμμα ροής του λογισμικού του (app.diagrams.net)

5.5.1 Πρώτο interrupt στο κυρίως πρόγραμμα της εφαρμογής android

Στο πρόγραμμα για την εφαρμογή android έχουν δημιουργηθεί δύο interrupt. Στο πρώτο interrupt το οποίο αναμένει να λάβει κάποιο μήνυμα από τον server, σημείο 1του διαγράμματος ροής (σχήμα 5.4). Μόλις λάβει ένα νέο μήνυμα από τον server σε οποιοδήποτε σημείο βρίσκεται το κυρίως πρόγραμμα διακόπτεται ώστε να εκτελεστεί το interrupt το οποίο θα αναλυθεί παρακάτω με το κατάλληλο διάγραμμα ροής. Έπειτα κατά την ολοκλήρωση του interrupt το πρόγραμμα συνεχίζει την εκτέλεση του κυρίως προγράμματος ακριβώς από το σημείο το οποίο είχε σταματήσει ώστε να εκτελεστεί το interrupt. Αυτή η διαδικασία πραγματοποιείται σε οποιοδήποτε σημείο του κύριου προγράμματος και στην συνέχεια εκτελείται από το σημείο διακοπής μέχρι την ολοκλήρωσή του.

Επομένως μόλις ληφθεί ένα νέο μήνυμα από τον server τότε αν το topic είναι ίσο με «arm/temperature», σημείο 2 (σχήμα 5.4), το οποίο αντιπροσωπεύει την θερμοκρασία που έχει λάβει το αισθητήριο που χρησιμοποιείται για την μέτρηση της θερμοκρασίας στο ενυδρείο, εμφανίζεται στην οθόνη του χρήση στην επάνω αριστερή πλευρά η τιμή της θερμοκρασίας σε βαθμούς κελσίου, σημείο 3 (σχήμα 5.4). Όπως φαίνεται και στην εικόνα από το γραφικό περιβάλλον της εφαρμογής.

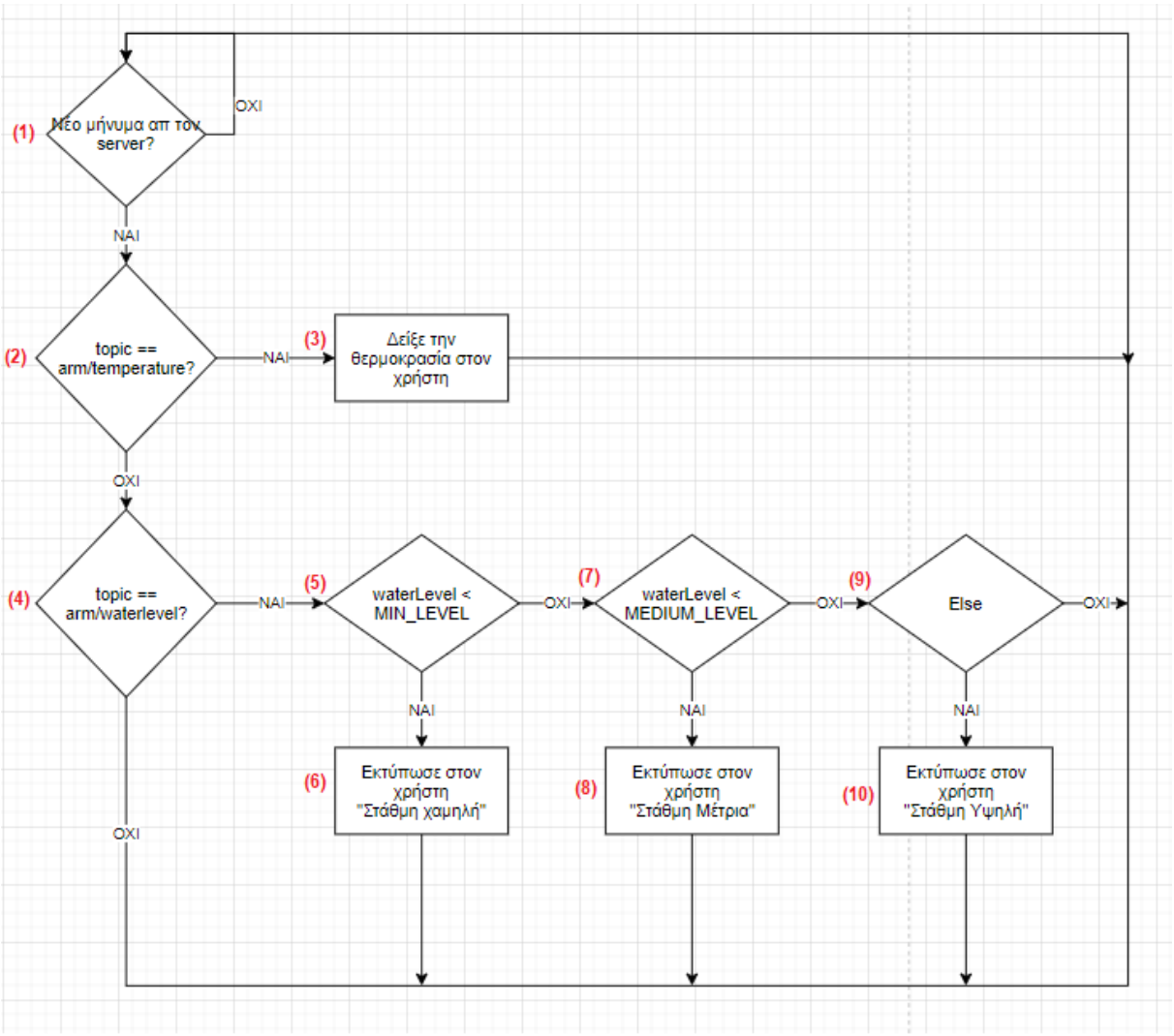


Εικόνα 49: Γραφικό περιβάλλον εφαρμογής

Έπειτα το πρόγραμμα επιστρέφει στο σημείο 1 (σχήμα 5.4) και αναμένει ένα νέο μήνυμα από τον server και το κυρίως πρόγραμμα εκτελείται από το σημείο όπου διακόπηκε για να εκτελέσει το interrupt.

Διαφορετικά εάν το topic είναι ίσο με «arm/waterlevel», σημείο 4 (σχήμα 5.4) το οποίο αντιπροσωπεύει την μέτρηση που λαμβάνει από το αισθητήριο στάθμης νερού τότε ελέγχεται αν το επίπεδο του νερού είναι μικρότερο από την μεταβλητή «MIN_LEVEL» στην οποία έχει δοθεί για παράδειγμα η τιμή 300, εάν ισχύει τότε εμφανίζει στον χρήστη στην πάνω δεξιά γωνία της οθόνης την ένδειξη «Στάθμη Χαμηλή» (εικόνα 50), σημείο 5 και 6 του διαγράμματος ροής (σχήμα 5.4). Ακόμα εάν το επίπεδο νερού είναι μικρότερο από την μεταβλητή «Medium_LEVEL» στην οποία έχει δοθεί

για παράδειγμα η τιμή 600, εάν ισχύει τότε στην περίπτωση αυτή εμφανίζεται στον χρήστη η ένδειξη «Στάθμη Μέτρια», σημείο 7 κα 8 του διαγράμματος ροής (σχήμα 5.4). Διαφορετικά εμφανίζει την ένδειξη «Στάθμη Υψηλή», σημείο 9 και 10 του διαγράμματος ροής (σχήμα 5.4). Κατά την ολοκλήρωση αυτού του ελέγχου των τριών σταδίων το πρόγραμμα επιστρέφει και αναμένει ένα νέο μήνυμα από τον server, σημείο 1 (σχήμα 5.4) και το κυρίως πρόγραμμα εκτελείται από το σημείο όπου διακόπηκε για να εκτελέσει το interrupt.

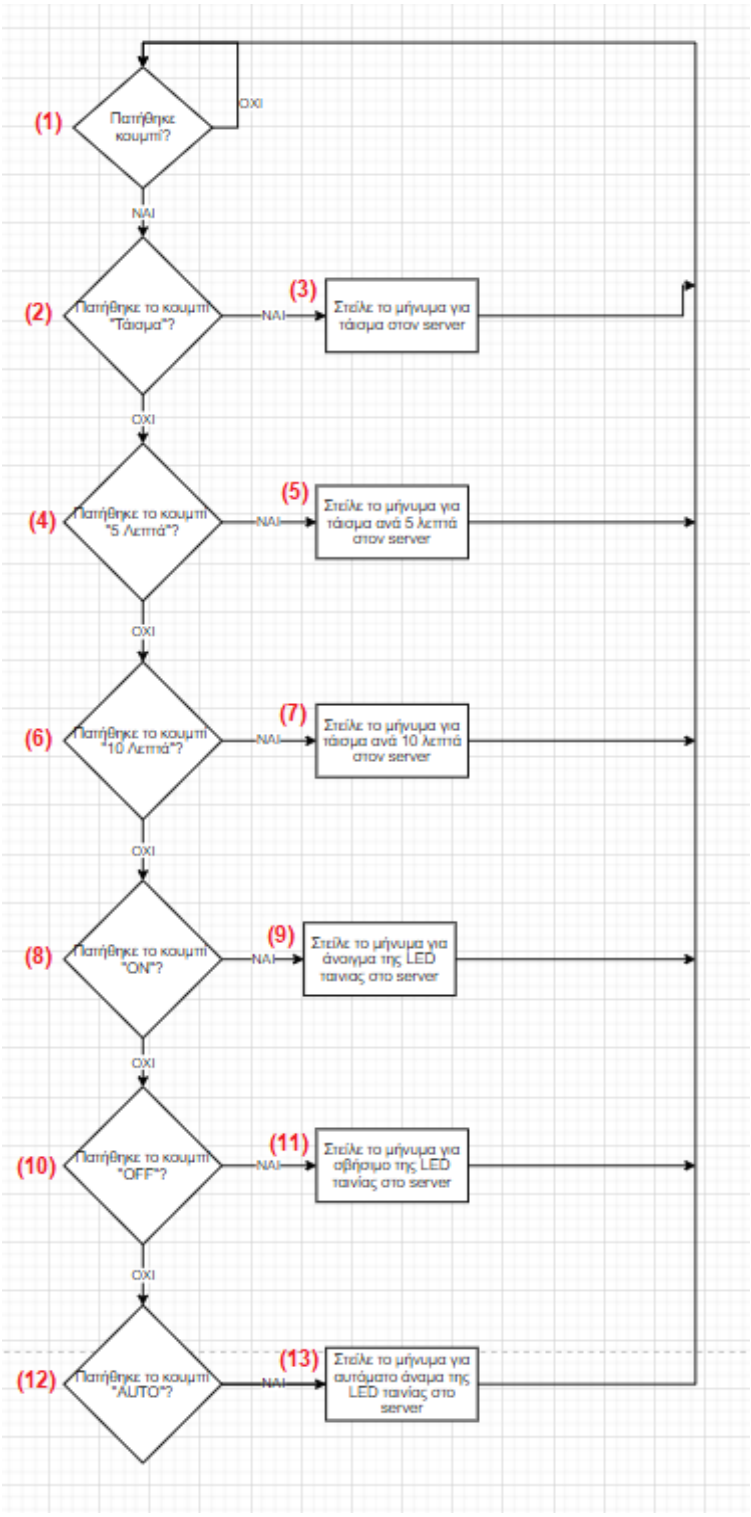


Σχήμα 5.4: Διάγραμμα ροής του λογισμικού του app (1ο interrupt) (app.diagrams.net)

5.5.2 Δεύτερο interrupt στο κυρίως πρόγραμμα της εφαρμογής android

Το δεύτερο interrupt αναμένει στην περίπτωση που ο χρήστης επιλέξει κάποιο από τα κουμπιά της εφαρμογής. Ουσιαστικά γίνεται έλεγχος αν έχει πατηθεί κάποιο κουμπί, σημείο 1 (σχήμα 5.5). Στην περίπτωση όπου έχει πατηθεί το κουμπί για τάισμα τότε γίνεται η αποστολή του μηνύματος για ενεργοποίηση της ταΐστρας στον server, σημείο 2 και 3 του διαγράμματος (σχήμα 5.5), έτσι με την διαδικασία αυτή ολοκληρώνεται το συγκεκριμένο interrupt. Στην περίπτωση που έχει πατηθεί το κουμπί «5 λεπτά» τότε γίνεται η αποστολή του μηνύματος για ενεργοποίηση της ταΐστρας κάθε πέντε λεπτά, στον server, σημείο 4 και 5 του διαγράμματος ροής (σχήμα 5.5). Εάν έχει επιλεγεί το κουμπί «10 λεπτά» τότε αποστέλλεται μήνυμα για ενεργοποίηση της ταΐστρας ανά δέκα λεπτά, στον server, σημείο 6 και 7 του διαγράμματος ροής (σχήμα 5.5). Ακόμα εάν έχει πατηθεί το κουμπί «ON» τότε το

μήνυμα αφορά την ενεργοποίηση της Led ταινίας και αποστέλλεται το μήνυμα για άνοιγμα της Led ταινίας, στον server, σημείο 8 και 9 (σχήμα 5.5) ενώ το κουμπί «OFF» για απενεργοποίηση της Led ταινίας και αποστέλλεται το μήνυμα για απενεργοποίηση της Led ταινίας, στον server, σημείο 10 και 11 του διαγράμματος ροής (σχήμα 5.5). Τέλος εάν επιλεγεί το κουμπί «AUTO» τότε γίνεται η αποστολή του μηνύματος για αυτόματη ενεργοποίηση/απενεργοποίηση της Led ταινίας στον server, σημείο 12 και 13 (σχήμα 5.5). Έπειτα από την εκτέλεση το interrupt ολοκληρώνεται.



Σχήμα 5.5: Διάγραμμα ροής του λογισμικού του app (2ο interrupt) (app.diagrams.net)

5.6 Επίλογος

Με την χρήση του Android Studio όσο και με την χρήση του Visual Studio, δύο πολύ χρήσιμων ηλεκτρονικών εργαλείων IDE, κατέστη δυνατόν η υλοποίηση των firmware για την εκτέλεση όλων των εντολών που ήταν απαραίτητες ώστε το συγκεκριμένο σύστημα να εκτελεί συγκεκριμένες ενέργειες με σκοπό την δημιουργία του έξυπνου ενυδρείου. Παρατίθενται ακόμα διαγράμματα ροής χρησιμοποιώντας το app.diagrams.net έτσι ώστε να γίνουν περισσότερο κατανοητοί οι αλγόριθμοι προς τον αναγνώστη της παρούσας Δ.Ε.

Κεφάλαιο 6ο: Συμπεράσματα και προτάσεις βελτίωσης

6.1 Συμπεράσματα

Στην παρούσα Δ.Ε σχεδιάστηκε και υλοποιήθηκε ένα αυτόματο σύστημα ελέγχου ενυδρείου. Το οποίο αποτελείται από τον μικροελεγκτή STM32H743ZI, τους αισθητήρες, τους ενεργοποιητές, τον server και την εφαρμογή android. Το τελικό σύστημα αξιοποιεί τις δυνατότητες της σύγχρονης τεχνολογίας τόσο σε επίπεδο software όσο και σε επίπεδο hardware.

Σε επίπεδο hardware χρειάζεται να οριστούν οι προδιαγραφές όσον αφορά την λειτουργία και τους στόχους για το τελικό αποτέλεσμα. Η επιλογή συγκεκριμένων λειτουργιών ενός συστήματος ώστε να αυτοματοποιηθούν έχει ως αποτέλεσμα την έρευνα των κατάλληλων αισθητήρων και η επιλογή των κατάλληλων αισθητήρων έχει ως αποτέλεσμα την επιλογή των κατάλληλων ενεργοποιητών.

Από την άλλη σε επίπεδο software, το android αποτελεί ένα open source λειτουργικό, το ολοκληρωμένο προγραμματιστικό περιβάλλον (IDE) παρέχεται δωρεάν και οι εφαρμογές υλοποιούνται σε γλώσσα προγραμματισμού Java που χρησιμοποιείται ευρέως και αυτά έχουν ως αποτέλεσμα η ανάπτυξη λογισμικού με τις πληθώρα βιβλιοθήκες που υπάρχουν να διευκολύνουν την υλοποίηση και εκτέλεση μιας εφαρμογής. Όπως επίσης το platformIO IDE είναι ένα πρόσθετο περιβάλλον ανάπτυξης λογισμικού στο Visual Studio. Μπορεί να ληφθεί μέσα από το Visual Studio δωρεάν και αποτελεί ένα προγραμματιστικό περιβάλλον με το οποίο μπορεί να «φορτωθεί» ο κώδικας στον μικροελεγκτή STM32H743ZI.

Τέλος σε κάθε κατασκευή όπως είναι φυσικό μπορούν να δημιουργηθούν προβλήματα. Ένα από τα σημαντικά προβλήματα που δημιουργήθηκε και στην παρούσα κατασκευή ήταν η σύνδεση στο WiFi μέσω του ESP-01. Αρκετές φορές και ενώ η σύνδεση στο διαδίκτυο ήταν καλή ο ESP-01 δεν μπορούσε να συνδεθεί, αυτό είχε ως αποτέλεσμα η επικοινωνία της κατασκευής με τον server και την εφαρμογή android να μην πραγματοποιείται. Αυτό που πολύ πιθανόν να επηρέαζε την λειτουργία ήταν κάποιες από τις βιβλιοθήκες του μικροελεγκτή STM32H743ZI και όχι ο ESP-01 καθώς αποτελεί ένα νέο υλικό της συγκεκριμένης εταιρίας και δεν υπάρχουν αρκετές βιβλιοθήκες.

6.2 Βελτίωση του συστήματος

Η ενασχόληση με το συγκεκριμένο αντικείμενο της Δ.Ε. έχει αρκετές δυνατότητες ως προς την επέκταση και την βελτίωσή του. Υπάρχει πληθώρα αισθητήρων και ενεργοποιητών που θα μπορούσαν να συμπληρωθούν ώστε το σύστημα να βελτιωθεί ακόμα περισσότερο. Ορισμένες από τις λειτουργίες που θα μπορούσαν να προστεθούν αποτελούν και θα αναλυθούν παρακάτω.

Με την χρήση του αισθητήρα στάθμης νερού το σύστημα στην παρούσα κατασκευή ενημερώνει τον χρήστη για το επίπεδο της στάθμης του νερού. Στη συγκεκριμένη λειτουργία θα μπορούσε ως

επέκταση το σύστημα να παρέχει την δυνατότητα και την επιλογή στον χρήστη να επιλέξει μέσω της εφαρμογής android να ενεργοποιείται η αντλία νερού ώστε να συμπληρωθεί ορισμένη ποσότητα νερού εάν το επίπεδο της στάθμης είναι σε χαμηλά επίπεδα. Αυτή η λειτουργία απαιτεί την χρήση ουσιαστικά ενεργοποιητή, δηλαδή την χρήση της αντλίας νερού.

Μία άλλη δυνατότητα που θα μπορούσε να δίνεται στον χρήστη είναι η αλλαγή νερού. Μέσω της εφαρμογής android ο χρήστης θα μπορούσε να επιλέξει την επιλογή «αλλαγή νερού» στο δοχείο του ενυδρείου. Αυτή η λειτουργία απαιτεί την χρήση δύο αντλιών, η μία αντλία θα λειτουργούσε ως αντλία συμπλήρωσης νερού στο προκαθορισμένο επίπεδο στάθμης, ενώ η δεύτερη αντλία ως αντλία αφαίρεσης νερού.

Ακόμα ένας αισθητήρας που θα μπορούσε να χρησιμοποιηθεί είναι ο αισθητήρας ph του νερού. Όπως ο αισθητήρας θερμοκρασίας στην παρούσα κατασκευή ενημερώνει τον χρήστη μέσω της εφαρμογής android έτσι και με τον αισθητήρα ph και με το κατάλληλο γραφικό διάγραμμα μέσω της εφαρμογής android ο χρήστης θα μπορούσε να ενημερώνεται αντίστοιχα. Στην συγκεκριμένη λειτουργία ως ενεργοποιητή θα μπορούσε να χρησιμοποιηθεί μία συσκευή του εμπορίου η λειτουργία του οποίου είναι να φιλτράρει το νερό που βρίσκεται στο εσωτερικό δοχείο του ενυδρείου. Με αυτή την λειτουργία το φιλτράρισμα θα μπορούσε να πραγματοποιείται όταν τα επίπεδα του ph του νερού είναι σε μη επιθυμητά επίπεδα για την συντήρηση των οργανισμών στο ενυδρείο. Βέβαια θα μπορούσε να συνδεθεί και με το προηγούμενο υποσύστημα το οποίο αναφέρθηκε στην προηγούμενη παράγραφο ώστε να ενεργοποιούνται οι αντλίες για αλλαγή νερού εάν το φιλτράρισμα του νερού δεν φέρει τα επιθυμητά αποτελέσματα.

Όλες αυτές οι λειτουργίες που αναφέρθηκαν προηγουμένως χρησιμοποιούν διάφορα αισθητήρια και ορισμένους ενεργοποιητές οι οποίου ουσιαστικά συλλέγουν πληροφορίες. Οι πληροφορίες αυτές θα μπορούσαν να έχουν σημαντικό αντίκτυπο για περεταίρω μελέτη και βελτίωση του συστήματος. Στην παρούσα Δ.Ε. τα δεδομένα που συλλέγονται από τα αισθητήρα δεν χρησιμοποιούνται μακροπρόθεσμα, με άλλα λόγια δεν αποθηκεύονται σε κάποιο cloud ώστε να χρησιμοποιηθούν. Αυτό που θα μπορούσε να γίνει και να έχει ιδιαίτερη σημασία για περαιτέρω μελέτη είναι η αποθήκευση των δεδομένων αυτών χρησιμοποιώντας την βάση δεδομένων Firebase σε πραγματικό χρόνο (realtime). Επομένως ας υποθέσουμε ότι ο αισθητήρας ph του νερού συλλέγει δεδομένα. Ο server «στέλνει» τα δεδομένα αυτά στην βάση δεδομένων Firebase ώστε να αποθηκευτούν. Έπειτα από κάποιο χρονικό διάστημα και αφού ο αριθμός των δεδομένων είναι ικανοποιητικός θα μπορούσε να πραγματοποιηθεί μία μελέτη για την ποιότητα του νερού. Έτσι με αυτόν τον τρόπο ο server θα μπορούσε να συλλέξει τα δεδομένα από την βάση δεδομένων και η ανάλυση και επεξεργασία των δεδομένων αυτών θα μπορούσε να πραγματοποιηθεί στον server.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] K. Ashton, "That 'Internet of Things' Thing In the real world, things matter more than ideas," 2002. [Online]. Available: http://www.itrco.jp/libraries/RFIDjournal-That%20Internet%20of%20Things%20Thing.pdf.

[2] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of Things (IoT): A Literature Review," *Journal of Computer and Communications*, vol. 03, pp. 164-173, May 2015. [Online]. Available: https://www.scirp.org/pdf/JCC_2015052516013923.pdf.

[3] H. Tschofenig, D. Thaler, and D. Mcpherson, "Internet Architecture Board (IAB)," pp. 4-9, Mar. 2015. [Online]. Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc7452.txt.pdf.

[4] H. Tschofenig, D. Thaler, and D. Mcpherson, "Internet Architecture Board (IAB)," pp. 1, Mar. 2015. [Online]. Available: https://www.rfc-editor.org/rfc/pdfrfc/rfc7452.txt.pdf.

[5] International Telecommunication Union (ITU), "Overview of the Internet of things," Jun. 2015. [Online]. Available: https://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11559&lang=en.

[6] "Internet of Things | IEEE Communications Society," *www.comsoc.org*. [Online]. Available: https://www.comsoc.org/publications/magazines/ieee-communications-magazine/cfp/internet-things.

[7] Wikipedia Contributors, "Internet of things," Wikipedia, Mar. 2019. [Online]. Available: https://en.wikipedia.org/wiki/Internet_of_Things.

[8] P. Sethi and S. R. Sarangi, "Internet of Things: Architectures, Protocols, and Applications," *Journal of Electrical and Computer Engineering*, vol. 2017, pp. 1–25, Jan. 2017, [Online]. Available: https://www.hindawi.com/journals/jece/2017/9324035/.

[9] *Cloudfront.net*, 2017. [Online]. Available: https://d197for5662m48.cloudfront.net/documents/publicationstatus/37889/preprint_pdf/340edf4833a5f1abcd84452af75c2426.pdf.

[10] "MQTT Protocol Tutorial: Technical description with practical Mosquitto example," *SwA*, Oct. 25, 2016. [Online]. Available: https://www.survivingwithandroid.com/mqtt-protocol-tutorial/.

[11] T. H. Team, "Quality of Service 0,1 & 2 - MQTT Essentials: Part 6," *www.hivemq.com*. [Online]. Available: https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/.

[12] The HiveMQ Team, "MQTT Publish, Subscribe & Unsubscribe - MQTT Essentials: Part 4," *Hivemq.com*, 2015. [Online]. Available: https://www.hivemq.com/blog/mqtt-essentials-part-4-mqtt-publish-subscribe-unsubscribe/.

[13] A. Lohachab and Karambir, "ECC based inter-device authentication and authorization scheme using MQTT for IoT networks," *Journal of Information Security and Applications*, vol. 46, pp. 1–12, Jun. 2019, [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S2214212618306513.

[14] P. Mell and T. Grance, "Special Publication 800-145 The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology," Sep. 2011. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf.

[15] J. Sluijs, P. Larouche, and W. Sauter, "Cloud Computing in the EU Policy Sphere Interoperability, Vertical Integration and the Internal Market,." Accessed: Jul. 29, 2021. [Online]. Available: https://www.jipitec.eu/issues/jipitec-3-1-2012/3320/sluijs.pdf.

[16] "The Top 5 Cloud Computing Trends to Watch in 2020," *OffsiteNOC*, Sep. 25, 2020. [Online]. Available: https://www.offsitenoc.com/the-top-5-cloud-computing-trends-to-watch-in-2020/.

[17] P. Mell and T. Grance, "The NIST Definition of Cloud Computing,." Accessed: Jul. 29, 2021. [Online]. Available: https://www.nist.gov/system/files/documents/itl/cloud/cloud-def-v15.pdf

[18] P. Mell and T. Grance, “Special Publication 800-145 The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology,” Sep. 2011. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf.
[19] D. Rountree and I. Castrillo, “Chapter 1 - Introduction to the Cloud,” *ScienceDirect*, Jan. 01, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780124059320000013.
[20] Administrator, “Difference between IaaS, PaaS, and SaaS. Examples of Cloud Service Models | LitsLink Blog,” *LITSLINK*, Aug. 15, 2019. [Online]. Available: https://litslink.com/blog/iaas-paas-saas.
[21] T. Dillon, C. Wu, and E. Chang, “Cloud Computing: Issues and Challenges,” *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 2010, [Online]. Available: https://ieeexplore.ieee.org/document/5474674.
[22] K. Singh, “What is the difference between Public, Private and Hybrid Cloud?,” *Medium*, Jan. 16, 2020. [Online]. Available: https://karansinghreen.medium.com/what-is-the-difference-between-public-private-and-hybrid-cloud-a41bba631479.
[23] M. Armbrust et al., “Above the Clouds: A Berkeley View of Cloud Computing,” Feb. 2009. [Online]. Available: https://www2.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf.
[24] “What is a Private Cloud - Definition | Microsoft Azure,” *Microsoft.com*, 2019. https://azure.microsoft.com/en-us/overview/what-is-a-private-cloud/.
[25] A. Marinos and G. Briscoe, “Community Cloud Computing,” *Lecture Notes in Computer Science*, pp. 472–484, 2009, [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-10665-1_43.
[26] L. Yan, C. Rong, and G. Zhao, “Strengthen Cloud Computing Security with Federal Identity Management Using Hierarchical Identity-Based Cryptography,” *Lecture Notes in Computer Science*, pp. 167–177, 2009, [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-642-10665-1_15.
[27] Amazon, “Amazon Web Services (AWS) - Cloud Computing Services,” *Amazon Web Services*, Inc., 2018. https://aws.amazon.com/.
[28] “What is AWS,” *Amazon Web Services*, Inc., 2018. https://aws.amazon.com/what-is-aws/?nc1=f_cc.
[29] K. Mani, “[John Rittinghouse, James Ransome] Cloud Computing (Bookos org),” *www.academia.edu*, Accessed: Jul. 29, 2021. [Online]. Available: https://www.academia.edu/4052556/_John_Rittinghouse_James_Ransome_Cloud_Computing_Bookos_org.
[30] “Amazon EC2 FAQs - Amazon Web Services,” *Amazon Web Services*, Inc., 2019. https://aws.amazon.com/ec2/faqs/.
[31] “User manual STM32 Nucleo-144 boards (MB1137),” 2020. Accessed: Aug. 02, 2021. [Online]. Available: https://www.st.com/resource/en/user_manual/dm00244518-stm32-nucleo144-boards-mb1137-stmicroelectronics.pdf.
[32] “STM32H743ZI - High-performance and DSP with DP-FPU, Arm Cortex-M7 MCU with 2MBytes of Flash memory, 1MB RAM, 480 MHz CPU, Art Accelerator, L1 cache, external memory interface, large set of peripherals - STMicroelectronics,” www.st.com. (accessed Aug. 02, 2021). [Online]. Available: https://www.st.com/content/st_com/en/products/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus/stm32-high-performance-mcus/stm32h7-series/stm32h743-753/stm32h743zi.html#overview&secondary=st_description_sec-nav-tab.
[33] “This is information on a product in full production,” 2021. Accessed: Aug. 02, 2021. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32h743zi.pdf.
[34] “DS18B20 pdf, DS18B20 description, DS18B20 datasheets, DS18B20 ALLDATASHEET,” [Online]. Available: https://pdf1.alldatasheet.com/datasheet-pdf/view/433921/MAXIM/DS18B20.html.

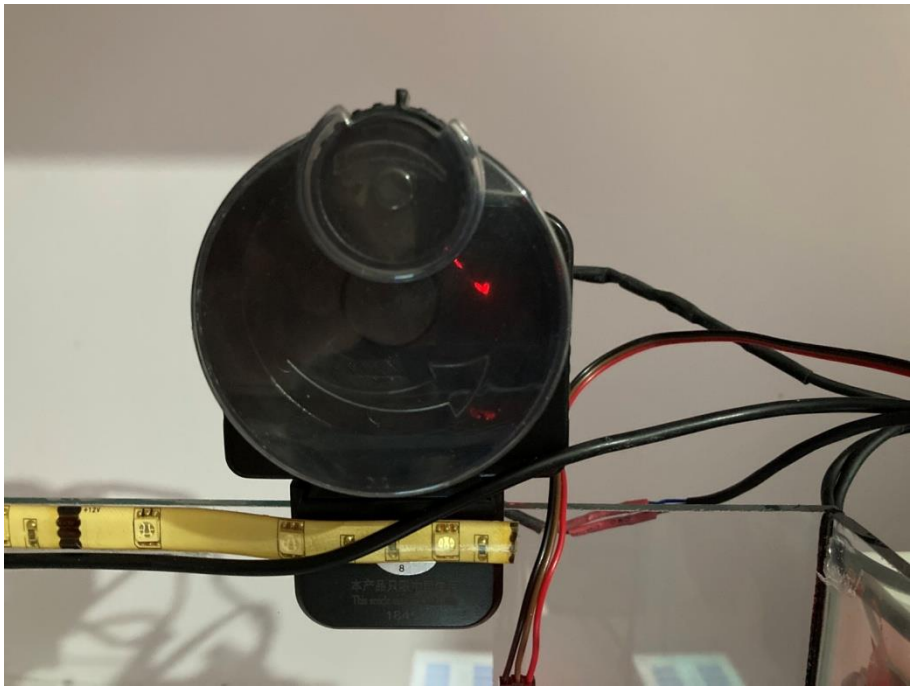
[35] "Interfacing DS18B20 1-Wire Digital Temperature Sensor with Arduino," Last Minute Engineers, Dec. 29, 2018. [Online]. Available: https://lastminuteengineers.com/ds18b20-arduino-tutorial/.
[36] "Pull-up Resistors - learn.sparkfun.com," *Sparkfun.com*, 2019. https://learn.sparkfun.com/tutorials/pull-up-resistors/all.
[37] "In-Depth: How Water Level Sensor Works and Interface it with Arduino," *Last Minute Engineers*, Nov. 29, 2019. https://lastminuteengineers.com/water-level-sensor-arduino-tutorial/.
[38] E. Stuff, "Interfacing LDR Module with Arduino," *EProjectbox*, Aug. 12, 2019. [Online]. Available: https://www.eprojectbox.com/interfacing-ldr-module-with-arduino/.
[39] "kgiannaras - Φωτοαντίσταση," *users.sch.gr*. [Online]. Available: http://users.sch.gr/kgiannaras/genika-ilektronika/fotoantistasi.html.
[40] Συνεισφέροντες στα εγχειρήματα Wikimedia, "Ηλεκτρονόμος," *Wikipedia.org*, Apr. 04, 2006. [Online]. Available: https://el.wikipedia.org/wiki/%CE%97%CE%BB%CE%B5%CE%BA%CF%84%CF%81%CE%BF%CE%BD%CF%8C%CE%BC%CE%BF%CF%82.
[41] "ESP-01 pdf, ESP-01 description, ESP-01 datasheets, ESP-01 ALLDATASHEET," *pdf1.alldatasheet.com*. https://pdf1.alldatasheet.com/datasheet-pdf/view/1179098/ETC2/ESP-01.html (accessed Aug. 02, 2021).
[42] "Disclaimer and Copyright Notice," [Online]. Available: https://www.microchip.ua/wireless/esp01.pdf
[43] "ESP-01 Adapter Module 3.3-5V," *grobotronics.com*. https://grobotronics.com/esp-01-adapter-module-3.3-5v.html (accessed Aug. 02, 2021).
[44] "Application Fundamentals | Android Developers," *Android Developers*, 2019. https://developer.android.com/guide/components/fundamentals.
[45] "Download Android Studio and SDK tools," *Android Developers*, 2019. https://developer.android.com/studio.
[46] "Android Studio: An IDE built for Android," *Android Developers Blog*. https://android-developers.googleblog.com/2013/05/android-studio-ide-built-for-android.html.
[47] "Downloads | IDE, Code, & Team Foundation Server | Visual Studio," *Visual Studio*, 2019. https://visualstudio.microsoft.com/downloads/.
[48] PlatformIO, "PlatformIO is a professional collaborative platform for embedded development," *PlatformIO*. https://platformio.org/.
[49] "PlatformIO IDE — PlatformIO latest documentation," *docs.platformio.org*. https://docs.platformio.org/en/latest/integration/ide/pioide.html#pioide (accessed Aug. 02, 2021).
[50] "OneWire - Arduino Reference," *www.arduino.cc*. https://www.arduino.cc/reference/en/libraries/onewire/ (accessed Aug. 02, 2021).
[51] "DallasTemperature - Arduino Reference," *www.arduino.cc*. https://www.arduino.cc/reference/en/libraries/dallastemperature/.
[52] "Arduino - SoftwareSerial," *Arduino.cc*, 2019. https://www.arduino.cc/en/Reference/softwareSerial.
[53] "TinhGSM", *PlatformIO.org*, 2021. *https://platformio.org/lib/show/1287/TinyGSM/examples?file=HttpClient.ino*.
[54] "PubSubClient - Arduino Reference," *www.arduino.cc*. https://www.arduino.cc/reference/en/libraries/pubsubclient/ (accessed Aug. 02, 2021).
[55] "Arduino Reference," *www.arduino.cc*. https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/.

ΠΑΡΑΡΤΗΜΑ Α Εικόνες από την κατασκευή

Ο αισθητήρας για τον έλεγχο της λειτουργίας του φως είναι τοποθετημένος πάνω στο κουτί της κατασκευής ώστε η φωτοανίσταση να μην εμποδίζεται.

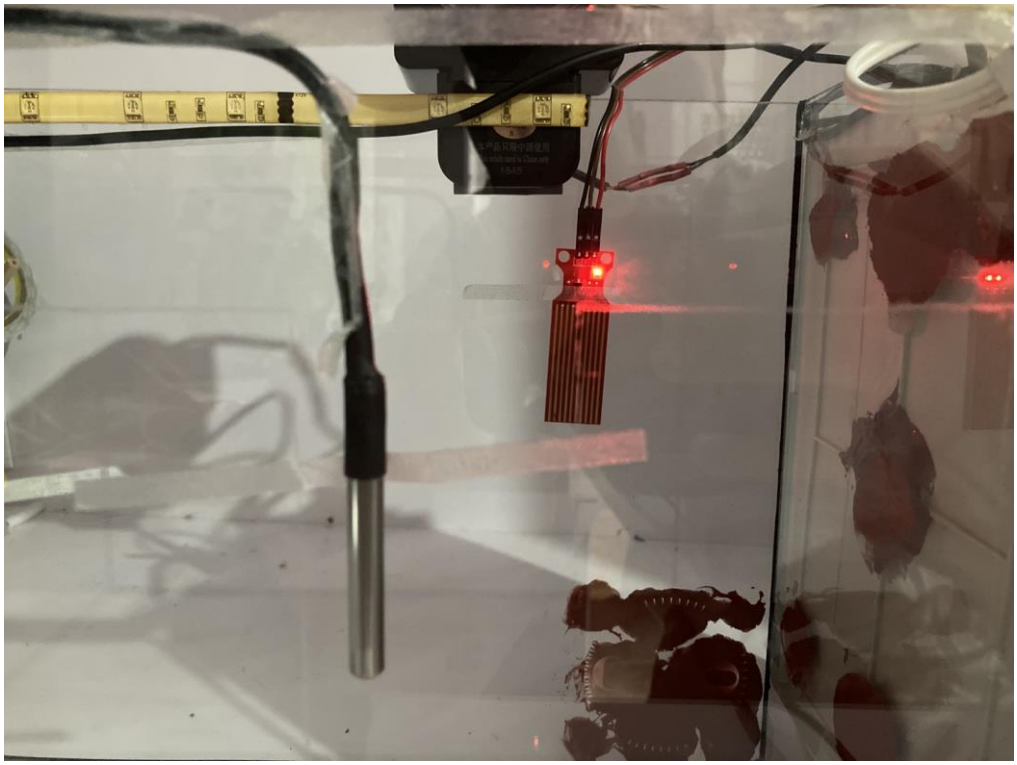
Εικόνα 50: Αισθητήρας φωτεινότητας (φωτοαντίσταση)

Ο μηχανισμός ταΐστρας έχει τοποθετηθεί στην ειδική εγκοπή της γυάλας.



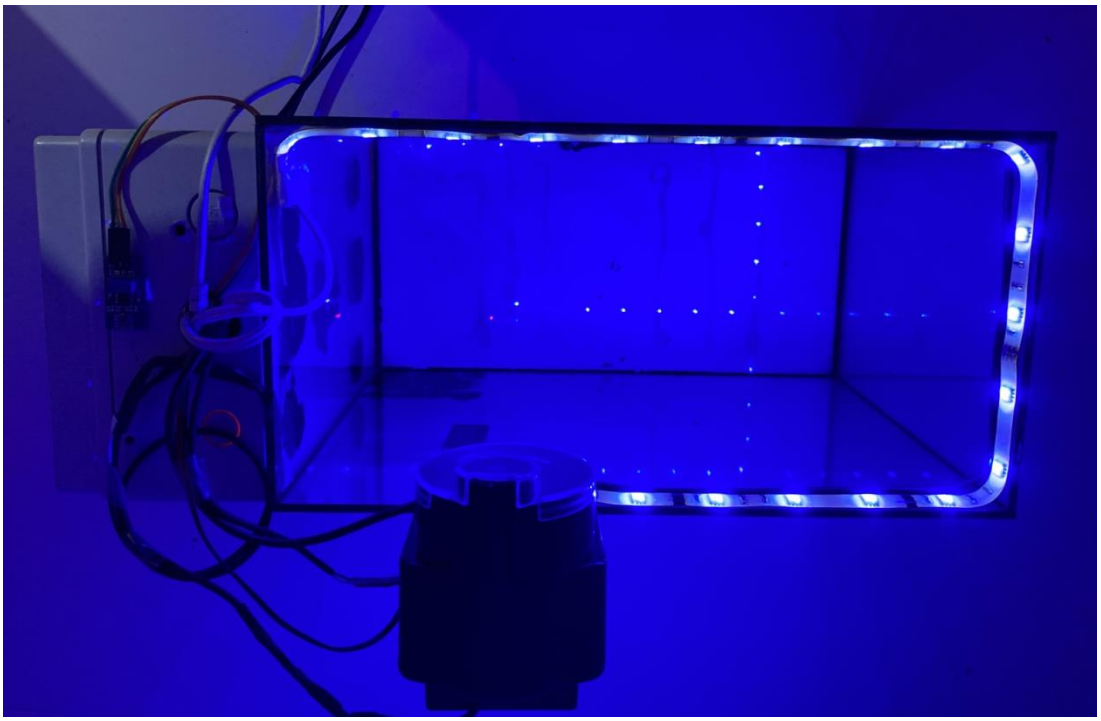
Εικόνα 51: Μηχανισμός ταΐστρας

Ο αισθητήρας θερμοκρασίας έχει τοποθετηθεί στην μία πλευρά της γυάλας όπως επίσης και ο αισθητήρας στάθμης νερού ώστε να έρχονται σε επαφή με το νερό.



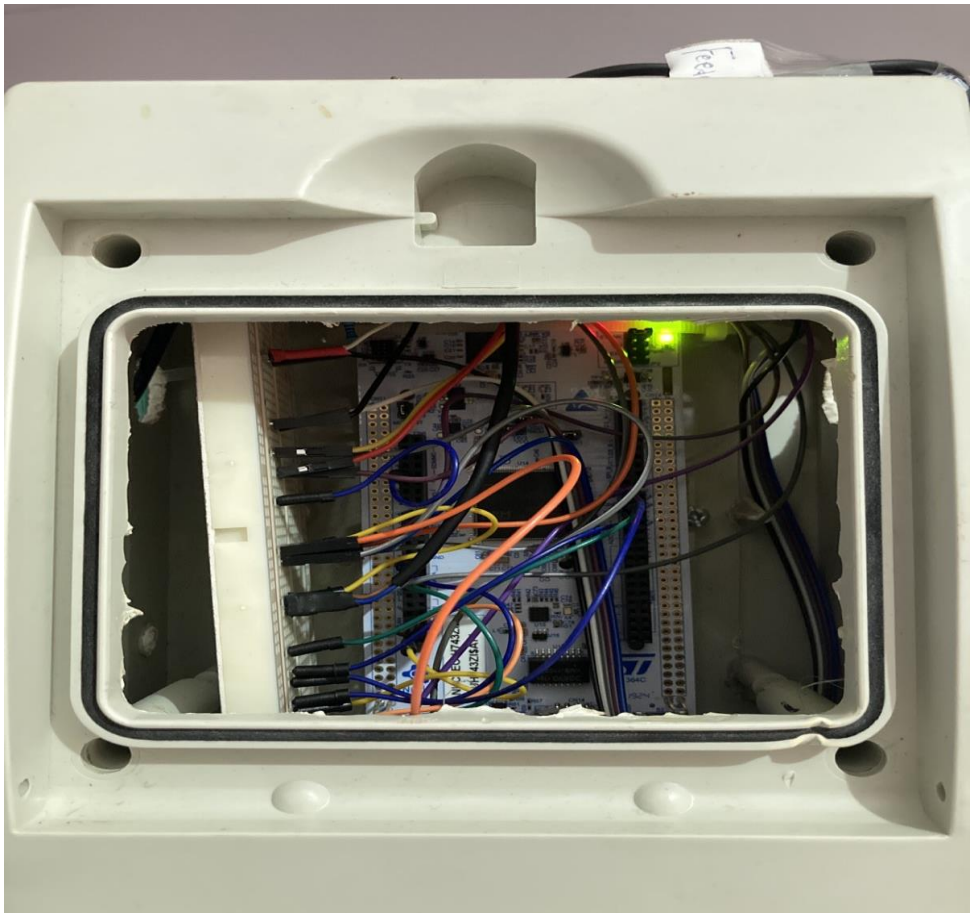
Εικόνα 52: Αισθητήρας θερμοκρασίας και στάθμης νερού

Η led ταινία σε κατάσταση λειτουργίας με χαμηλό φωτισμό στον χώρο που έχει τοποθετηθεί η κατασκευή.



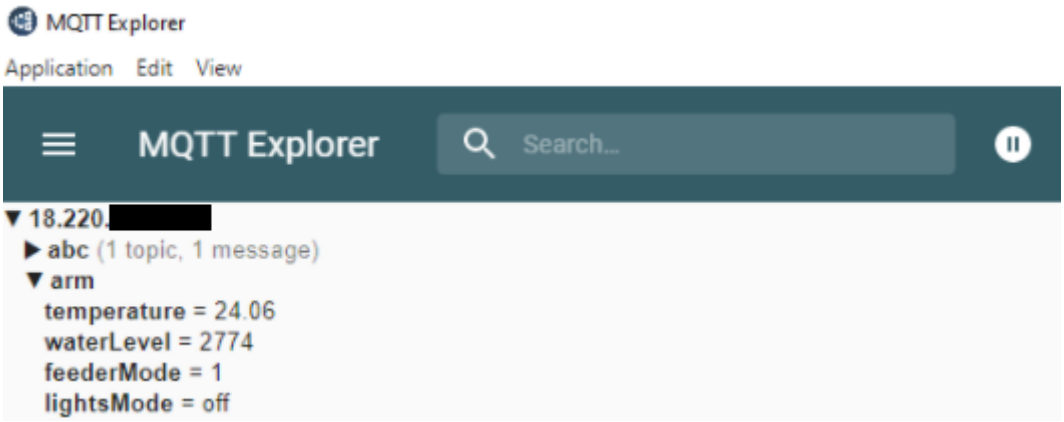
Εικόνα 53: Led

Το κουτί στο οποίο έχει τοποθετηθεί όλο το κύκλωμα συμπεριλαμβανομένου του μικροελεγκτή, του Esp-01 και του ράστερ.



Εικόνα 54: Κουτί κατασκευής

Με την χρήση του MQTT Explorer είναι δυνατή η εμφάνιση των μηνυμάτων που δημιουργούνται στον MQTT server από τις συσκευές που είναι συνδεδεμένες στον server, όπως ο αισθητήρας θερμοκρασίας, στάθμης νερού, μηχανισμός ταΐστρας και αισθητήρας φωτεινότητας.



Εικόνα 55: Μηνύματα στον server

ΠΑΡΑΡΤΗΜΑ Β Κώδικας λογισμικού εφαρμογής Android

```
1.  import androidx.appcompat.app.AppCompatActivity;
2.  import android.graphics.Color;
3.  import android.os.Bundle;
4.  import android.util.Log;
5.  import android.view.View;
6.  import android.widget.Button;
7.  import com.uin.remoteaquarium.views.ColorArcProgressBar;
8.  import org.eclipse.paho.android.service.MqttAndroidClient;
9.  import org.eclipse.paho.client.mqttv3.IMqttActionListener;
10. import org.eclipse.paho.client.mqttv3.IMqttDeliveryToken;
11. import org.eclipse.paho.client.mqttv3.IMqttMessageListener;
12. import org.eclipse.paho.client.mqttv3.IMqttToken;
13. import org.eclipse.paho.client.mqttv3.MqttCallbackExtended;
14. import org.eclipse.paho.client.mqttv3.MqttClient;
15. import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
16. import org.eclipse.paho.client.mqttv3.MqttException;
17. import org.eclipse.paho.client.mqttv3.MqttMessage;
18. public class MainActivity extends AppCompatActivity implements View.OnClickListener {
19.     private static final String TAG = "MainActivity";
20.     private ColorArcProgressBar mProgressBar;
21.     private MqttAndroidClient client;
22.     private Button mBtnFeeder;
23.     private Button mBtnFeederDelay5;
24.     private Button mBtnFeederDelay10;
25.     private Button mBtnLightsOn;
26.     private Button mBtnLightsOff;
27.     private Button mBtnLightsAuto;
28.     private IMqttToken token;
29.     private Button mBtnWaterLevel;
30.     @Override
31.     protected void onCreate(Bundle savedInstanceState) {
32.       super.onCreate(savedInstanceState);
33.       setContentView(R.layout.activity_main);
34.       mProgressBar = findViewById(R.id.temp_sensor);
35.       mBtnFeeder = findViewById(R.id.btn_feeder);
36.       mBtnFeederDelay5 = findViewById(R.id.btn_feeder_delay5);
37.       mBtnFeederDelay10 = findViewById(R.id.btn_feeder_delay10);
38.       mBtnLightsOn = findViewById(R.id.btn_lights_on);
39.       mBtnLightsOff = findViewById(R.id.btn_lights_off);
40.       mBtnLightsAuto = findViewById(R.id.btn_lights_auto);
41.       mBtnWaterLevel = findViewById(R.id.water_level_indicator);
42.       mBtnFeeder.setOnClickListener(this);
43.       mBtnFeederDelay5.setOnClickListener(this);
44.       mBtnFeederDelay10.setOnClickListener(this);
45.       mBtnLightsOn.setOnClickListener(this);
46.       mBtnLightsOff.setOnClickListener(this);
47.       mBtnLightsAuto.setOnClickListener(this);
48.       mProgressBar.setMaxValues(40);
49.       mProgressBar.setUnit("°C");
50.       mProgressBar.setCurrentValues(10, 0);
51.       final String clientId = MqttClient.generateClientId();
52.        client = new MqttAndroidClient(this, "tcp://XX.XXX.XXX.XXX:1883", clientId);
53.       client.setCallback(new MqttCallbackExtended() {
54.        @Override
55.        public void connectComplete(boolean reconnect, String serverURI) {
56.         Log.d(TAG, "connectComplete: ");
57.         if(reconnect){
58.          subscribeToTopics();
59.          }
60.         }

61.       @Override
62.       public void connectionLost(Throwable cause) {
63.         Log.d(TAG, "connectionLost: ");
64.       }
65.       @Override
66.        public void messageArrived(String topic, MqttMessage message) throws Exception{
67.        Log.d(TAG, "messageArrived: " + topic + " " + message.toString());
68.        if(topic.equals("arm/temperature")){
69.         mProgressBar.setCurrentValues(Float.valueOf(message.toString()), 0);
70.        }else if(topic.equals("arm/waterLevel")){
71.         int waterLevel = Integer.valueOf(message.toString());
72.         if(waterLevel < 341){
```

```
73.           mBtnWaterLevel.setTextColor(Color.GREEN);
74.           mBtnWaterLevel.setText("ΧΑΜΗΛΗ");
75.         }else if(waterLevel < 681){
76.           mBtnWaterLevel.setTextColor(Color.YELLOW);
77.           mBtnWaterLevel.setText("МЕТРІА");
78.         }else{
79.           mBtnWaterLevel.setText("ΥΨΗΛΗ");
80.           mBtnWaterLevel.setTextColor(Color.RED);
81.          }
82.         }
83.        }
84.        @Override
85.        public void deliveryComplete(IMqttDeliveryToken token) {
86.         Log.d(TAG, "deliveryComplete: ");
87.        }
88.       });
89.       MqttConnectOptions connOpts = new MqttConnectOptions();
90.       connOpts.setAutomaticReconnect(true);
91.       connOpts.setCleanSession(true);
92.   //        connOpts.setKeepAliveInterval(300);
93.       try {
94.         token = client.connect(connOpts);
95.       } catch (MqttException e) {
96.         e.printStackTrace();
97.       }
98.       token.setActionCallback(new IMqttActionListener() {
99.         @Override
100.                                    public        void        onSuccess(IMqttToken
    asyncActionToken) {

101.                                   subscribeToTopics();
102.                                   }
103.                                   @Override
104.                                   public        void        onFailure(IMqttToken
    asyncActionToken, Throwable exception) {

105.                                   }
106.                                   });
107.                                   }
108.                                   private void subscribeToTopics() {
109.                                   try {
110.                                   //            client.subscribe
111.                                   client.subscribe("arm/temperature",  2,  null,
    new IMqttActionListener() {
112.                                   @Override
113.                                   public        void        onSuccess(IMqttToken
    asyncActionToken) {
114.                                   //                                    Log.d(TAG,
    "messageArrived: " + message.toString());
115.                                   //
    mProgressBar.setCurrentValues(Float.valueOf(message.toString()), 0);
116.                                   }
117.                                   @Override
118.                                   public        void        onFailure(IMqttToken
    asyncActionToken, Throwable exception) {

119.                                   }
120.                                   });
121.                                   client.subscribe("arm/waterLevel",  2,  null,
    new IMqttActionListener() {
122.                                   @Override
123.                                   public        void        onSuccess(IMqttToken
    asyncActionToken) {

124.                                   }
125.                                   @Override
126.                                   public        void        onFailure(IMqttToken
    asyncActionToken, Throwable exception) {

127.                                   }
128.                                   });
129.                                   } catch (MqttException e) {
130.                                   e.printStackTrace();
131.                                   }
132.                                   }
133.                                   private String mqttPubMessage = "";
134.                                   private String mqttTopic = "";
```

```
                                        @Override
                                        public void onClick(View v) {
                                        if(v.getId() == R.id.btn_feeder){
                                        mqttTopic = "arm/feederStart";
                                        mqttPubMessage = "";
                                        }else             if(v.getId()             ==
    R.id.btn_feeder_delay5){
                                        mqttTopic = "arm/feederMode";
                                        mqttPubMessage = "1";
                                        }else             if(v.getId()             ==
    R.id.btn_feeder_delay10){
                                        mqttTopic = "arm/feederMode";
                                        mqttPubMessage = "2";
                                        }else if(v.getId() == R.id.btn_lights_on){
                                        mqttTopic = "arm/lightsMode";
                                        mqttPubMessage = "on";
                                        }else if(v.getId() == R.id.btn_lights_off){
                                        mqttTopic = "arm/lightsMode";
                                        mqttPubMessage = "off";
                                        }else if(v.getId() == R.id.btn_lights_auto){
                                        mqttTopic = "arm/lightsMode";
                                        mqttPubMessage = "auto";
                                        }
                                        try {
                                        Log.d(TAG,    "onSuccess:    Sending    mqtt
    message");
                                        MqttMessage message = new MqttMessage();
                                        message.setQos(1);
                                        message.setRetained(true);
                                        message.setPayload(mqttPubMessage.getBytes());
                                        client.publish(mqttTopic, message);
                                        } catch (MqttException e) {
                                        e.printStackTrace();
                                        }
                                        }
                                        }
```

ΠΑΡΑΡΤΗΜΑ C Κώδικας λογισμικού του μικροελεγκτή ST32H743ZI

```
#include <Arduino.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <SoftwareSerial.h>
#define TINY_GSM_MODEM_ESP8266
#include <TinyGsmClient.h>
#include <PubSubClient.h>
#define ONE_WIRE_BUS 4
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
#define SerialMon Serial
SoftwareSerial SerialAT(2, 3); // RX, TX
// Define the serial console for debug prints, if needed
#define TINY_GSM_DEBUG SerialMon
// Range to attempt to autobaud
#define GSM_AUTOBAUD_MIN 9600
#define GSM_AUTOBAUD_MAX 115200
// Your WiFi connection credentials, if applicable
const char wifiSSID[] = "XXXXXX";
const char wifiPass[] = "xxxxxx";
// MQTT details
const char* broker = "xx.xxx.xxx.xxx";
const char* topicInit = "arm/init";
const char* topicTemperature = "arm/temperature";
const char* topicWaterLevel = "arm/waterLevel";
const char* topicLightLevel = "arm/lightLevel";
const char* topicFeederStart = "arm/feederStart";
const char* topicFeederMode = "arm/feederMode";
const char* topicLightsMode = "arm/lightsMode";
TinyGsm modem(SerialAT);
TinyGsmClient client(modem);
PubSubClient mqtt(client);
// Sensor pins
```

```
#define waterSensorPower 7
#define waterSensorPin A0
#define lightSensorPin A1
#define LED_PIN 8
#define FEEDER_PIN 6
uint32_t lastBroadcast = 0;
uint32_t lastTimeFed = 0;
int waterLevel = 0;
int feedMode = 0; // 0 = manual, 1 = delay 5mins, 2 = delay 10mins
bool lightsAuto = true;
#define LIGHTS_THRESHOLD 550
int readSensor() {
  digitalWrite(waterSensorPower, HIGH);
  delay(10);
  waterLevel = analogRead(waterSensorPin);
  digitalWrite(waterSensorPower, LOW);
  return waterLevel;
}
void feedFish(){
  SerialMon.println("Feeding fish - Feeder on");
  digitalWrite(FEEDER_PIN, HIGH);
  delay(50);
  digitalWrite(FEEDER_PIN, LOW);
}
void mqttCallback(char* topic, byte* payload, unsigned int len) {
  SerialMon.print("Message arrived [");
  SerialMon.print(topic);
  SerialMon.print("]: ");
  SerialMon.write(payload, len);
  SerialMon.println();
  String payloadString = "";
  for (int i = 0; i < len; i++) {
    payloadString += (char)payload[i];
  }
  if (String(topic) == topicLightsMode) {
    if(payloadString == "on"){
      lightsAuto = false;
      digitalWrite(LED_PIN, HIGH);
      SerialMon.println("MANUAL - LIGHTS ON");
    }else if(payloadString == "off"){
      lightsAuto = false;
      digitalWrite(LED_PIN, LOW);
      SerialMon.println("MANUAL - LIGHTS OFF");
    }else if(payloadString == "auto"){
      lightsAuto = true;
      SerialMon.println("MANUAL - LIGHTS AUTO");
    }
  }else if (String(topic) == topicFeederMode) {
    feedMode = payloadString.toInt();
  }else if (String(topic) == topicFeederStart) {
    feedFish();
  }
}
boolean mqttConnect() {
  SerialMon.print("Connecting to ");
  SerialMon.print(broker);
  // Connect to MQTT Broker
  boolean status = mqtt.connect("AquariumTest");
  // Or, if you want to authenticate MQTT:
  //boolean status = mqtt.connect("GsmClientName", "mqtt_user", "mqtt_pass");
  if (status == false) {
    SerialMon.println(" fail");
    return false;
  }
  SerialMon.println(" success");
  mqtt.publish(topicInit, "ARM Aquarium System started");
                                          mqtt.subscribe(topicFeederStart);
                                          mqtt.subscribe(topicFeederMode);
                                          mqtt.subscribe(topicLightsMode);
                                          return mqtt.connected();
                                        }
                                        void setup() {
                                          SerialMon.begin(115200);
                                          delay(10);
                                          sensors.begin();
                                          pinMode(waterSensorPower, OUTPUT);
                                          pinMode(FEEDER_PIN, OUTPUT);
```

```
111.                                          pinMode(LED_PIN, OUTPUT);
112.                                          digitalWrite(waterSensorPower, LOW);
113.                                          SerialMon.println("Wait...");
114.                                          // Set GSM module baud rate
115.                                          TinyGsmAutoBaud(SerialAT,  GSM_AUTOBAUD_MIN,
    GSM_AUTOBAUD_MAX);
116.                                          // SerialAT.begin(115200);
117.                                          delay(5000);
118.                                          // Restart takes quite some time
119.                                        //  To  skip  it,  call  init()  instead  of
    restart()
120.                                          SerialMon.println("Initializing modem...");
121.                                          // modem.restart();
122.                                           modem.init();
123.                                          SerialMon.print(F("Setting
    SSID/password..."));
124.                                          if          (!modem.networkConnect(wifiSSID,
    wifiPass)) {
125.                                            SerialMon.println(" fail");
126.                                            delay(10000);
127.                                            return;
128.                                          }
129.                                          SerialMon.println(" success");
130.
131.                                          SerialMon.print("Waiting for network...");
132.                                          if (!modem.waitForNetwork()) {
133.                                            SerialMon.println(" fail");
134.                                            delay(10000);
135.                                            return;
136.                                          }
137.                                          SerialMon.println(" success");
138.
139.                                          if (modem.isNetworkConnected()) {
140.                                            SerialMon.println("Network connected");
141.                                          }
142.                                          mqtt.setServer(broker, 1883);
143.                                          mqtt.setCallback(mqttCallback);
144.                                        }
145.                                        void loop() {
146.                                          if (!mqtt.connected()) {
147.                                            SerialMon.println("=== MQTT NOT CONNECTED
    ===");
148.                                            SerialMon.print("Is network connected?");
149.
    SerialMon.println(modem.isNetworkConnected());
150.                                            mqttConnect();
151.                                            delay(50);
152.                                            return;
153.                                          }
154.                                          mqtt.loop();
155.
156.                                          int level = readSensor();
157.                                          sensors.requestTemperatures();
158.                                          int             lightIntensity             =
    analogRead(lightSensorPin);
159.                                          SerialMon.print("Light Intensity: ");
160.                                          SerialMon.println(lightIntensity);
161.
162.                                          if(lightsAuto    &&    (lightIntensity    <
    LIGHTS_THRESHOLD)){
163.                                            SerialMon.println("AUTO - LIGHTS ON");
164.                                            digitalWrite(LED_PIN, HIGH);
165.                                          }else  if(lightsAuto  &&  (lightIntensity  >=
    LIGHTS_THRESHOLD)){
166.                                            SerialMon.println("AUTO - LIGHTS OFF");
167.                                            digitalWrite(LED_PIN, LOW);
168.                                          }
169.                                          if (millis() - lastBroadcast > 10000L) {
170.                                              lastBroadcast = millis();
171.                                              SerialMon.println("Broadcast data");
172.                                              SerialMon.print("Water level: ");
173.                                              SerialMon.println(level);
174.                                              Serial.print("Celsius temperature: ");
175.
    Serial.println(sensors.getTempCByIndex(0));
176.
```

```
177.                                                 mqtt.publish(topicTemperature,
    String(sensors.getTempCByIndex(0)).c_str());
178.                                                 mqtt.publish(topicWaterLevel,
    String(level).c_str());
179.                                             }
180.
181.                                             if  ((feedMode  ==  1)  &&  ((millis()  -
    lastTimeFed) > 300000L)) {
182.                                                 lastTimeFed = millis();
183.                                                 SerialMon.println();
184.                                                 SerialMon.println("---FEED FISH---5");
185.                                                 SerialMon.println();
186.                                                 feedFish();
187.                                             }
188.
189.                                             if  ((feedMode  ==  2)  &&  ((millis()  -
    lastTimeFed) > 600000L)) {
190.                                                 lastTimeFed = millis();
191.                                                 SerialMon.println();
192.                                                 SerialMon.println("---FEED FISH---10");
193.                                                 SerialMon.println();
194.                                                 feedFish();
195.                                             }
196.                                         }
```