



**ΔΙΚΤΥΟ ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ: ΑΝΑΠΤΥΞΗ ΔΙΚΤΥΟΥ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ ΓΙΑ ΠΑΡΑΓΩΓΗ ΚΑΙ ΠΑΡΑΚΟΛΟΥΘΗΣΗ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΔΕΔΟΜΕΝΩΝ**

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΜΑΡΟΥΛΑΚΟΣ-ΣΕΦΕΡΙΑΔΗΣ ΠΑΡΗΣ Α.Μ. :57128

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΣΠΥΡΙΔΩΝ ΜΟΥΡΟΥΤΣΟΣ

ΞΑΝΘΗ 2021

**ΔΙΚΤΥΟ ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ: ΑΝΑΠΤΥΞΗ ΔΙΚΤΥΟΥ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ ΓΙΑ ΠΑΡΑΓΩΓΗ ΚΑΙ ΠΑΡΑΚΟΛΟΥΘΗΣΗ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΔΕΔΟΜΕΝΩΝ**



**ΔΙΚΤΥΟ ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ: ΑΝΑΠΤΥΞΗ ΔΙΚΤΥΟΥ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ ΓΙΑ ΠΑΡΑΓΩΓΗ ΚΑΙ ΠΑΡΑΚΟΛΟΥΘΗΣΗ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΔΕΔΟΜΕΝΩΝ**

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΜΑΡΟΥΛΑΚΟΣ-ΣΕΦΕΡΙΑΔΗΣ ΠΑΡΗΣ Α.Μ. :57128

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΣΠΥΡΙΔΩΝ ΜΟΥΡΟΥΤΣΟΣ

ΞΑΝΘΗ 2021

**ΔΙΚΤΥΟ ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ: ΑΝΑΠΤΥΞΗ ΔΙΚΤΥΟΥ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΣΤΑΘΜΩΝ ΓΙΑ ΠΑΡΑΓΩΓΗ ΚΑΙ ΠΑΡΑΚΟΛΟΥΘΗΣΗ
ΜΕΤΕΩΡΟΛΟΓΙΚΩΝ ΔΕΔΟΜΕΝΩΝ**

Διπλωματική Εργασία/DTh b88 2021

Μαρουλάκος Σεφεριάδης Πάρης (ΑΕΜ: 57128)(parimaro@ee.duth.gr)

Τομέας Ενεργειακών Συστημάτων

Εργαστήριο Μηχανοτρονικής και Αυτοματισμών Η-Μ Συστημάτων

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Πολυτεχνική Σχολή Ξάνθης - Δημοκρίτειο Πανεπιστήμιο Θράκης

Επιβλέπων Καθηγητής: Σ. Γ. Μουρούτσος (sgmour@ee.duth.gr)

Στην οικογένεια μου

Περίληψη

Η παρούσα εργασία έχει σκοπό τη δημιουργία ενός δικτύου μετεωρολογικών σταθμών, το οποίο, θα συλλέγει τα δεδομένα που θα καταγράφουν οι επιμέρους μετεωρολογικοί σταθμοί σε έναν κεντρικό εξυπηρετητή (server), σε κατάλληλα διαμορφωμένη βάση δεδομένων. Βασιζόμενοι στην επιστήμη του IoT, εκμεταλλευόμαστε τις πιο εξελιγμένες εκδόσεις μικροελεγκτών, μικροϋπολογιστών, αισθητηρίων και λογισμικού με στόχο τη δημιουργία ενός αυτόνομου αλλά και διαχωρίσιμου δικτύου απομακρυσμένης τοποθέτησης και παρακολούθησης των μετεωρολογικών σταθμών και των δεδομένων που αυτοί παράγουν. Το δίκτυο που μελετήσαμε και κατασκευάσαμε διαθέτει γραφικό περιβάλλον για την αλληλεπίδραση με το χρήστη και υλοποιήθηκε ως διαδικτυακή ιστοσελίδα. Συγκεκριμένα, η διαδικτυακή ιστοσελίδα, δίνει τη δυνατότητα στο χρήστη να παρακολουθεί σε πραγματικό χρόνο τις μετρήσεις των σταθμών, που είναι συνδεδεμένοι στο δίκτυο. Βασικοί μας στόχοι, ήταν η δυνατότητα εύκολης τοποθέτησης-εγκατάστασης του δικτύου σε οποιοδήποτε περιβάλλον χρειαστεί, η εύκολη διάσπαση του δικτύου σε περίπτωση που κάποιος χρήστης θελήσει να χρησιμοποιήσει ένα συγκεκριμένο μόνο αριθμό από τους μετεωρολογικούς σταθμούς του δικτύου και η συλλογή αξιόπιστων μετεωρολογικών δεδομένων της περιοχής που τοποθετείται το σύστημα. Για την εύκολη διάσπαση του δικτύου καταστήσαμε κάθε σταθμό αυτόνομο, με διαφορετικό πρωτόκολλο αποστολής δεδομένων, ώστε να μπορεί κάθε σταθμός να σταθεί και από μόνος του ως ένα δίκτυο.

Abstract

The present Diploma has the purpose of creating a network of meteorological stations, which will collect data recorded by the stations and send them to a suitable database at a central server. Based on science of IoT, we take advantage of the most evolved editions of microcontrollers, microcomputers, sensors and software with a goal to create an autonomous and separable network of distant placing and monitoring of the stations and the data they produce. The network we designed and constructed has its own graphic interface for user interaction, implemented as a webpage. Specifically, the webpage, gives the user the possibility to monitor in real time the sensors measurements of the stations connected to the network. Basic goals, was the capability of easy implementation of the network to any environment necessary for any user, the easy separation of the network in case a user doesn't want to use all the stations and the collection of reliable weather data at the area that the network is implemented. For the easy separation of the network we made every station autonomous, with a different communication protocol with the server, In order each station to be capable to stand alone as the network.

Πρόλογος

Για να γίνει πιο κατανοητή η περιγραφή καθώς και η συνολική παρουσίαση της εργασίας στα στάδια που εκπονήθηκε, έγινε διαχωρισμός της κυρίως εργασίας σε 3 μέρη:

- 1) Ανάπτυξη και κατασκευή μετεωρολογικών σταθμών, με τους οποίους θα καταγράφονται, θα αποθηκεύονται και θα αποστέλλονται μετεωρολογικά δεδομένα
- 2) Δημιουργία εξυπηρετητή ο οποίος θα συγκεντρώνει τα δεδομένα των σταθμών σε μια βάση δεδομένων
- 3) Δημιουργία και ανάπτυξη εφαρμογών για την αλληλεπίδραση του χρήστη με τα δεδομένα

1) Αρχικά για κάθε σταθμό χρησιμοποιήθηκαν διάφορων ειδών αισθητήρια, όπως για παράδειγμα, αισθητήρια για την μέτρηση της ατμοσφαιρικής πίεσης, της θερμοκρασίας, της υγρασίας, της έντασης του φωτός και άλλα.

Στη συνέχεια, αυτά τα ξεχωριστά για κάθε σταθμό αισθητήρια, συνδέθηκαν σε ένα κατάλληλα προγραμματισμένο να τα υποδεχθεί Arduino (ένα για κάθε σταθμό).

Ο κάθε σταθμός-το κάθε Arduino, αποθηκεύει τα δεδομένα σε μια SD card, στέλνει τα δεδομένα που καταγράφει, είτε μέσω ίντερνετ, είτε μέσω κάρτας SIM και του δικτύου 2g, είτε μέσω του πρωτοκόλλου RF (Κεραιών) στον server.

2) Για τον εξυπηρετητή (server) χρησιμοποιήθηκε ένας μικροπολογιστής τύπου raspberry pi, ο οποίος προγραμματίστηκε κατάλληλα ώστε να δέχεται τα δεδομένα κάθε σταθμού.

Στη συνέχεια, δημιουργήθηκε μια βάση δεδομένων ώστε να αποθηκεύονται καταλλήλως τα δεδομένα που δέχεται ο server.

3) Για την αλληλεπίδραση του χρήστη με τα δεδομένα δημιουργήθηκε ένας ιστότοπος που δίνει διάφορες δυνατότητες στο χρήστη όπως π.χ. μενού διαχείρισης, live δεδομένα ανάλογα με την ώρα-λεπτά, πρόβλεψη καιρού για το μέρος που είναι εγκατεστημένο το δίκτυο κ.λπ.

Αναλυτικότερα, το κείμενο της παρούσας διπλωματικής εργασίας, διαρθρώνεται σε έξι κεφάλαια και δύο παραρτήματα.

Στο 1ο Κεφάλαιο γίνεται μία εισαγωγή στο «μετεωρολογικούς σταθμούς» και στις έννοιες δίκτυα υπολογιστών και δομές δικτύων.

Στο 2ο Κεφάλαιο παρουσιάζονται τα υλικά (hardware) που χρησιμοποιήθηκαν για την υλοποίηση των σταθμών και του εξυπηρετητή, ο ρόλος τους στην ανάπτυξη του δικτύου καθώς και τα γραφικά περιβάλλοντα που αυτά αναπτύχθηκαν.

Στο 3ο Κεφάλαιο παρουσιάζονται τα πρωτόκολλα επικοινωνίας καθώς και το πως αυτά υλοποιήθηκαν για τους σκοπούς της εργασίας.

Στο 4ο Κεφάλαιο παρουσιάζεται το στήσιμο του όλου hardware, το στήσιμο του απαραίτητου software και ο συνδυασμός τους στις ανάγκες της παρούσας εργασίας.

Στο 5ο Κεφάλαιο παρουσιάζεται η υλοποίηση του ιστότοπου.

Στο 6ο Κεφάλαιο παρουσιάζονται τα συμπεράσματα της εργασίας και προτάσεις για την μελλοντική βελτίωση της.

Το Παράρτημα Α περιλαμβάνει τους πηγαίους κώδικες όλων των επιμέρους σταθμών.

Το Παράρτημα Β περιλαμβάνει τους πηγαίους κώδικες των προγραμμάτων του εξυπηρετητή.

Με την περάτωση της διπλωματικής, θα ήθελα να ευχαριστήσω τον διευθυντή του Εργαστηρίου Μηχανοτρονικής και Αυτοματισμών Η-Μ Συστημάτων και επιβλέποντα καθηγητή μου κ. Σπυρίδωνα Γ. Μουρούτσο για την ευκαιρία που μου έδωσε να ασχοληθώ με ένα τόσο ενδιαφέρον και σύγχρονο αντικείμενο. Τον ευχαριστώ ακόμα για την υποστήριξη, τις συμβουλές του, καθώς και τη βοήθεια που μου παρείχε. Επίσης, ένα μεγάλο ευχαριστώ στον υπεύθυνο εργαστηρίου κ. Ηλία Τσαγκαλίδη για τη βοήθεια που μου προσέφερε καθ' όλη τη διάρκεια εκπόνησης της διπλωματικής μου εργασίας.

Πίνακας Περιεχομένων

Περίληψη	9
Abstract.....	11
Πρόλογος	13
Πίνακας Περιεχομένων	15
Κατάλογος Εικόνων.....	19
1. ΚΕΦΑΛΑΙΟ ΠΡΩΤΟ	21
1.1 Μετεωρολογία	21
1.1.1 Μετεωρολογικοί σταθμοί	21
1.2 Δίκτυο υπολογιστών.....	23
1.3 Πλεονεκτήματα χρήσης δικτύου μετεωρολογικών σταθμών	24
1.4 Δομή δικτύου μετεωρολογικών σταθμών	25
2 ΚΕΦΑΛΑΙΟ ΔΕΥΤΕΡΟ	26
2.1 Τι είναι οι μικροελεγκτές Arduino;.....	26
2.1.1 Γιατί να επιλέξω το Arduino;.....	27
2.1.2 Τεχνικά χαρακτηριστικά του μικροελεγκτή Arduino Mega	28
2.1.3 Περιβάλλον ανάπτυξης	30
2.2 Τι είναι το Raspberry pi;	31
2.2.1 Γιατί να επιλέξω το Raspberry pi;.....	32
2.2.2 Περιβάλλον ανάπτυξης	33
2.3 Αισθητήρια	34
2.3.1 Αισθητήριο DHT-22	34
2.3.2 Αισθητήριο LPS22HB	35
2.3.3 TSL2591X-Light Sensor	36
2.3.4 Αισθητήριο UV Detection.....	37
3 ΚΕΦΑΛΑΙΟ ΤΡΙΤΟ	40
3.1 Επικοινωνία & Πρωτόκολλα.....	40
3.1.1 Πρωτόκολλα και επικοινωνία αισθητηρίων με το Arduino	40
3.1.1.1 UART PROTOCOL:	40
3.1.1.2 SPI PROTOCOL:	40
3.1.1.3 I2C PROTOCOL:	41
3.1.2 Πρωτόκολλα και επικοινωνία των Arduino με τον Server	41
3.1.2.1 Wi-Fi.....	41

3.1.2.2	GSM	42
3.1.2.2.1	Αρχιτεκτονική GSM	42
3.1.2.3	RF	43
3.1.2.3.1	Αμφίδρομοι σύνδεσμοι για τηλεχειριστήριο RF:.....	44
3.1.2.3.2	Πρωτόκολλα επικοινωνίας RF:.....	45
3.1.2.3.3	Εφαρμογές επικοινωνίας RF	46
3.2	Υλοποίηση των πρωτοκόλλων στην εργασία.....	47
3.2.1	Ethernet shield	47
3.2.2	GSM MKR 1400.....	48
3.2.3	LoRa-02 SX1278	51
3.3	Βλάβες των Modules και άλλα εμπόδια	52
3.4	Συχνότητα Μετρήσεων.....	54
4	ΚΕΦΑΛΑΙΟ ΤΕΤΑΡΤΟ.....	55
4.1	Υλοποίηση hardware μετεωρολογικών σταθμών & server	55
4.1.1	Σταθμός HMMY	55
4.1.2	Σταθμός ΠΡΟΚΑΤ	57
4.1.3	Σταθμοί Ε.Κ ΑΘΗΝΑ & ΕΣΤΙΩΝ	61
4.1.4	Server.....	64
4.2	Υλοποίηση software server	66
4.2.1	Το λειτουργικό σύστημα	66
4.2.2	Apache web server	68
4.2.3	MySQL & Βάσεις δεδομένων.....	71
4.2.4	PhpMyAdmin	72
4.2.5	PHP	73
4.2.6	Port Forwarding.....	74
4.2.6.1	Static Ip	75
4.2.6.2	DDNS.....	78
4.3	Software μετεωρολογικών σταθμών	79
4.3.1	RTC Module	80
4.3.2	SD Card Module.....	80
4.3.3	Σταθμός HMMY	80
4.3.4	Σταθμός ΠΡΟΚΑΤ	82
4.3.5	Σταθμοί Ε.Κ ΑΘΗΝΑ & ΕΣΤΙΩΝ	83
4.4	Πακέτο δεδομένων.....	83
4.5	Raspberry Pi & προγραμματισμός κεραίας.....	84
4.5.1	Python.....	84

4.5.2	Ανεμιστήρας	86
4.5.3	Χρόνο-προγραμματιστής Cron	86
5	ΚΕΦΑΛΑΙΟ ΠΕΜΠΤΟ	89
5.1	Ιστοσελίδα για παρουσίαση και αλληλεπίδραση με τα δεδομένα	89
5.1.1	HTML & CSS	89
5.1.2	JavaScript & JQuery	90
5.1.3	Σελίδα	91
6	ΚΕΦΑΛΑΙΟ ΕΚΤΟ	96
6.1	Συμπεράσματα	96
6.2	Βελτιώσεις-Προτάσεις.....	96
6.2.1	Μετεωρολογικοί σταθμοί	96
6.2.2	Server.....	97
7	Βιβλιογραφία.....	99
8	ΠΑΡΑΡΤΗΜΑ Α Κώδικες Σταθμών	102
8.1	A.1 ethernetshieldworkingwithrpi.ino	102
8.2	A.2 mkrghsm1400working.ino	106
8.3	A.3 rfwithrtcdatalogger.ino	110
9	ΠΑΡΑΡΤΗΜΑ Β Κώδικες Διακομιστή	116
9.1	B.1 rfp.py	116
9.2	B.2 ran-fan.py	117
9.3	B.3 index.php.....	118
9.4	B.4 hmmy.php	120
9.5	B.5 ekathens.php.....	126
9.6	B.6 esties.php	132
9.7	B.7 prokat.php.....	139
9.8	B.8 zampretti.php	145
9.9	B.9 meteo.php	149
9.10	B.10 connection.php	150
9.11	B.11 data_hmmy.php	151
9.12	B.12 data_prokat.php.....	151
9.13	B.13 data_esties.php	151
9.14	B.14 data_athina.php	152
9.15	B.15 hmmy_db.sql.....	152
9.16	B.16 prokat_db.sql	152
9.17	B.17 athina_db.sql.....	153

9.18	B.18 esties_db.sql.....	153
9.19	B.19 betel_cast.js.....	153
9.20	B.20 main.js	157

Κατάλογος Εικόνων

Εικόνα 1. 1 Τοπολογία σταθμών	25
Εικόνα 2. 1 Arduino UNO	26
Εικόνα 2. 2 Arduino MEGA	26
Εικόνα 2. 3 Περιβάλλον ανάπτυξης Arduino(ide)	31
Εικόνα 2. 4 Raspberry Pi 3 model b+	32
Εικόνα 2. 5 Αισθητήριο DHT22	35
Εικόνα 2. 6 Αισθητήριο LPS22HB.....	36
Εικόνα 2. 7 Αισθητήριο TSL2591X	37
Εικόνα 2. 8 Πίνακας δείκτη UV	38
Εικόνα 2. 9 Αισθητήριο UV-Light	39
Εικόνα 3. 1 Αρχιτεκτονική GSM(e-class Πάτρας ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ ΚΑΙ ΠΡΩΤΟΚΟΛΛΑ ΔΙΚΤΥΩΝ II).....	43
Εικόνα 3. 2 Ethernet shield μπροστά και πίσω	47
Εικόνα 3. 3 Arduino GSM MKR 1400	48
Εικόνα 3. 4 Arduino MKR GSM 1400 schematics.....	50
Εικόνα 3. 5 LoRa-02 SX1278 & Antenna for LoRa-02 SX1278 module	51
Εικόνα 3. 6 SD card module & RTC module with CR2032 lithium	54
Εικόνα 4. 1 Αρχείο .fzz του σταθμού ΗΜΜΥ	55
Εικόνα 4. 2 Σταθμός ΗΜΜΥ.....	57
Εικόνα 4. 3 Αρχείο .fzz του σταθμού ΠΡΟΚΑΤ.....	58
Εικόνα 4. 4 Σταθμός ΠΡΟΚΑΤ	60
Εικόνα 4. 5 Αρχείο .fzz των σταθμών Εστιών και Ε.Κ ΑΘΗΝΑ	61
Εικόνα 4. 6 Σταθμοί ΑΘΗΝΑ, ΕΣΤΙΩΝ	64
Εικόνα 4. 7 Αρχείο .fzz του server.....	64
Εικόνα 4. 8 Ο server	66
Εικόνα 4. 9 Το γραφικό περιβάλλον του Raspbian	68
Εικόνα 4. 10 Η αρχική σελίδα του Apache	70
Εικόνα 4. 11 Τα αρχεία του site.....	71
Εικόνα 4. 12 Γραφικό περιβάλλον PhpMyAdmin	73
Εικόνα 4. 13 Αποτέλεσμα εντολής ifconfig.....	76
Εικόνα 4. 14 Αρχική σελίδα router	77
Εικόνα 4. 15 Ρυθμίσεις Port Forward	78
Εικόνα 4. 16 Παράδειγμα αποτελέσματος εκτέλεσης	81
Εικόνα 4. 17 Byte ip & char serv	81
Εικόνα 4. 18 Port & file	81
Εικόνα 4. 19 GPRS APN	82
Εικόνα 4. 20 Char path & char server & port	82
Εικόνα 4. 21 String ide & logRF.....	83
Εικόνα 4. 22 Μεταβλητές μεγέθους μετρήσεων & θέσεις μετρήσεων	85
Εικόνα 4. 23 If statement.....	85

Εικόνα 4. 24 If statement που ανοιγοκλείνει τον ανεμιστήρα	86
Εικόνα 5. 1 Αρχική σελίδα Index.php	92
Εικόνα 5. 2 SQL Query.....	92
Εικόνα 5. 3 data_esties.php.....	93
Εικόνα 5. 4 Πίνακας σταθμού ΠΡΟΚΑΤ και γράφημα θερμοκρασιών	93
Εικόνα 5. 5 Φόρμα πρόβλεψης καιρού και widget ανέμων	94
Εικόνα 5. 6 Μετεώγραμμο και πληροφορίες.....	95

1. ΚΕΦΑΛΑΙΟ ΠΡΩΤΟ

1.1 Μετεωρολογία

Η μετεωρολογία ανήκει στις θετικές επιστήμες και ασχολείται με τα διάφορα φαινόμενα που συμβαίνουν στην ατμόσφαιρα. Διερευνά τον τρόπο με τον οποίο δημιουργούνται και μεταβάλλονται στο χρόνο.

Ο όρος μετεωρολογία προέρχεται από την αρχαία Ελληνική λέξη "μετέωρον", με την οποία χαρακτηριζόταν κάθε αντικείμενο που αιωρείται στην ατμόσφαιρα, όπως τα σύννεφα, οι σταγόνες βροχής, το χαλάζι κλπ. Για τους αρχαίους Έλληνες φιλόσοφους, η μετεωρολογία συνδεόταν στενά με την επιστήμη της αστρονομίας. Από τις πρώτες παρατηρήσεις που σχετίζονταν με την Μετεωρολογία είναι εκείνες που πραγματοποίησε ο Θαλής, το 640 π.Χ., ο οποίος απέδωσε τη διαδοχή των τεσσάρων εποχών του έτους στην συνεχή των θέσεων του Ήλιου στον ουρανό, και τις διαχώρισε (όπως άλλωστε ισχύει και σήμερα) με βάση την εαρινή και τη φθινοπωρινή ισημερία, καθώς και με το χειμερινό και το θερινό ηλιοστάσιο. Τα «Μετεωρολογικά» του Αριστοτέλη ήταν το πρότυπο διδακτικό βιβλίο της Μετεωρολογίας για πάνω από δύο χιλιάδες χρόνια. Ο Θεόφραστος με το βιβλίο «Σημείων» έγραψε κανόνες για την πρόγνωση του καιρού συνδυάζοντας παράδοση και επιστήμη. Η εξέλιξη της μετεωρολογίας συνεχίστηκε με σχετικά αργό ρυθμό ανά τα χρόνια μέχρι τον 19ό αιώνα που άρχισαν να παράγονται οι πρώτοι χάρτες καιρού και να δημιουργούνται οι πρώτες μετεωρολογικές υπηρεσίες σε εθνικό επίπεδο. Το 1878 ιδρύθηκε ο διεθνής Μετεωρολογικός Οργανισμός, ο οποίος από το 1950 μετονομάστηκε σε «Παγκόσμιο Μετεωρολογικό Οργανισμό» (W.M.O.). Τον 20ό αιώνα αναπτύχθηκε η σύγχρονη μετεωρολογία όπως την ξέρουμε σήμερα, με την πραγματοποίηση καθημερινών μετρήσεων στην ανώτερη ατμόσφαιρα και στην επιφάνεια της γης που μαζί με την ταχεία εξέλιξη των τεχνικών μέσων παρακολούθησης, καταγραφής και πρόβλεψης, συνέβαλαν στην αλματώδη αύξηση των μετεωρολογικών γνώσεων και άρα στην αύξηση της ακρίβειας των μοντέλων πρόβλεψης καιρού.

1.1.1 Μετεωρολογικοί σταθμοί

Για την παρατήρηση και την καταγραφή των καιρικών φαινομένων, έχουν κατασκευαστεί ποικίλα μετεωρολογικά συστήματα, τα οποία κυμαίνονται σε ένα ευρύ φάσμα τιμών και δυνατοτήτων. Τα συστήματα αυτά, απευθύνονται σε ένα ευρύ κοινό, το οποίο μπορεί να αποτελείται από ερασιτέχνες όπως αγρότες, μετεωρολόγους, αλιείς, πιλότους και άλλους μέχρι επαγγελματίες μετεωρολόγους ή εταιρίες με υψηλές απαιτήσεις.

Ανάλογα την πολυπλοκότητα τους, τα συστήματα μπορεί να είναι από απλά, όπως συστήματα απεικόνισης των δεδομένων, έως ιδιαίτερα πολύπλοκα συστήματα, τα οποία περιλαμβάνουν την καταγραφή δεδομένων και φαινομένων και τη

βραχυπρόθεσμη ή μακροπρόθεσμη πρόβλεψη του καιρού μέσω διαφόρων αλγορίθμων.

Μετεωρολογικά δεδομένα

Ένας μετεωρολογικός σταθμός ή ένα μετεωρολογικό σύστημα έχει την δυνατότητα να καταγράφει μια ποικιλία μετεωρολογικών δεδομένων. Τα δεδομένα αφορούν την θερμοκρασία, την υγρασία, την ατμοσφαιρική πίεση, την ένταση τους φωτός, την ηλιακή ακτινοβολία, τα χαρακτηριστικά των ανέμων, την ποιότητα του αέρα μιας περιοχής και πολλά άλλα.

Η γνώση του κλίματος που μας προσφέρουν τα μετεωρολογικά δεδομένα, μπορεί να αποβεί ένα πολύ χρήσιμο εργαλείο τόσο για έναν απλό χρήστη όσο και για ένα επαγγελματία. Ακόμη, μπορεί και σε ορισμένες περιπτώσεις πρέπει, να επηρεάζει τη λήψη οικονομικών και πολιτικών αποφάσεων για τις δραστηριότητες μιας ολόκληρης περιοχής, η οποία βρίσκεται η προβλέπεται ότι θα βρεθεί κάτω από ιδιαίτερες καιρικές συνθήκες (π.χ. ενός μουσώνα). Η έξυπνη διαχείριση των μετεωρολογικών υπηρεσιών είναι σημαντικός παράγοντας στην ανάπτυξη της οικονομίας μιας περιοχής με άστατο κλίμα.

Για τον μέσο απλό χρήστη και τις καθημερινές του ασχολίες, τα φαινόμενα του καιρού ίσως να μην τον επηρεάζουν ιδιαίτερα. Όμως υπάρχουν άνθρωποι οι οποίοι επηρεάζονται άμεσα και σημαντικά από τις μεταβολές των καιρικών συνθηκών. Ακόμη υπάρχουν άνθρωποι που χρειάζονται μετεωρολογικά δεδομένα για ένα πολύ πιο κλειστό χώρο που δεν επηρεάζεται ιδιαίτερα από τις καιρικές συνθήκες της περιοχής του. Για παράδειγμα ένας κάτοχος θερμοκηπίου, έχει ανάγκη να γνωρίζει με ακρίβεια τις συνθήκες του καιρού μέσα στο θερμοκήπιο, για να προγραμματίζει τις εργασίες του καταλλήλως. Ένας αγρότης, με χωράφι έξω από τον αστικό ιστό, μπορεί να χρειάζεται ένα δικό του σύστημα καταγραφής μετεωρολογικών δεδομένων, καθώς τα περισσότερα δελτία καιρού που μεταδίδονται από τα τηλεοπτικά μέσα, αναφέρονται συνήθως στις αστικές περιοχές όπου και διαμένει το μεγαλύτερο ποσοστό του πληθυσμού της χώρας. Στις περιπτώσεις αυτές λοιπόν, την λύση δίνουν τα τοπικά μετεωρολογικά συστήματα, τα οποία συλλέγουν δεδομένα για την περιοχή που είναι εγκατεστημένα και αν κρίνεται απαραίτητο εξάγουν και μια σχετικά ασφαλή πρόγνωση. Συνήθως, τα συστήματα αυτά έχουν υψηλό κόστος, με αποτέλεσμα για ορισμένους χρήστες και σε συνάρτηση πάντα με τα οφέλη τα οποία προκύπτουν από την χρήση τέτοιων σταθμών, να είναι απαγορευτικά. Ένα σύστημα μετεωρολογικών σταθμών για καταγραφή των μετεωρολογικών δεδομένων υλοποιήθηκε σε αυτή την εργασία με την μορφή ενός δικτύου αποτελούμενου από 4 μετεωρολογικούς σταθμούς και ενός εξυπηρετητή.

1.2 Δίκτυο υπολογιστών

Ένα δίκτυο υπολογιστών είναι ένα τηλεπικοινωνιακό σύστημα από αυτόνομους ή και μη, διασυνδεδεμένους υπολογιστές. Οι υπολογιστές ή αλλιώς κόμβοι, θεωρούνται διασυνδεδεμένοι όταν είναι σε θέση να ανταλλάξουν πληροφορίες σε μορφή κειμένου ήχου ή και εικόνας μεταξύ τους και αυτόνομοι όταν δεν είναι δυνατό κάποιος υπολογιστής να ελέγξει τη λειτουργία (π.χ. εκκίνηση ή τερματισμό) κάποιου άλλου. Η επιστημονική μελέτη των δικτύων υπολογιστών γίνεται από τα υπολογιστικά συστήματα, έναν βασικό κλάδο της πληροφορικής. Το θεμελιώδες ηλεκτρονικό υλικό των τηλεπικοινωνιακών συσκευών μελετάται επίσης από την ηλεκτρονική μηχανική.

Σε ένα δίκτυο υπολογιστών μπορούν να διασυνδεθούν μεταξύ τους εκτός από τα παραδοσιακά επιτραπέζια PC και άλλου τύπου συσκευές όπως PDAs (Personal Digital Assistants), κινητά τηλέφωνα, τηλεοράσεις, εκτυπωτές, σαρωτές και άλλα.

Τα δίκτυα φέρουν τους εξής χαρακτηρισμούς, που καθορίζουν και την κατηγορία τους :

- Ανάλογα με το φυσικό μέσο διασύνδεσής τους χαρακτηρίζονται ως «ενσύρματα» ή «ασύρματα».
- Ανάλογα με τον τρόπο πρόσβασης σε αυτά χαρακτηρίζονται ως «δημόσια» ή «ιδιωτικά» δίκτυα.
- Ανάλογα με την γεωγραφική κάλυψη του δικτύου χαρακτηρίζονται ως «τοπικά» (LAN και WLAN), «μητροπολιτικά» (MAN και WMAN), «ευρείας κάλυψης» (WAN και WWAN) και «προσωπικά» (PAN και WPAN).

Οι χαρακτηρισμοί με το πρόσθετο W ανταποκρίνονται στον ασύρματο (Wireless) τρόπο σύνδεσης.

Ανά γεωγραφική κάλυψη:

- Τοπικά: Τα «τοπικά δίκτυα» ή και «LAN» (local area networks) είναι δίκτυα που συνδέουν υπολογιστές σε κοντινές αποστάσεις, π.χ. από υπολογιστές που βρίσκονται σε ένα δωμάτιο μέχρι υπολογιστές που απέχουν μερικά χιλιόμετρα μεταξύ τους. Χρησιμοποιούνται συνήθως για να συνδέουν προσωπικούς υπολογιστές και σταθμούς εργασίας σε γραφεία εταιρειών, εργοστάσια, πανεπιστήμια κ.λπ.
- Μητροπολιτικά: Ένα «μητροπολιτικό δίκτυο» ή και «MAN» (metropolitan area network) είναι μια μεγαλύτερη εκδοχή ενός τοπικού δικτύου καθώς καλύπτει μεγαλύτερες αποστάσεις, π.χ. από μια ομάδα γειτονικών γραφείων μιας εταιρείας έως μια πόλη.

- Ευρείας περιοχής: Τα «δίκτυα ευρείας περιοχής» ή «WAN» (wide area network) καλύπτουν μεγάλες γεωγραφικές περιοχές, π.χ. από σύνδεση μεταξύ διαφορετικών πόλεων μέχρι μιας ολόκληρης ηπείρου και μπορούν να συνδέσουν ακόμη και περισσότερα από ένα τοπικά δίκτυα καθώς και ομάδες τοπικών δικτύων. Τα περισσότερα δίκτυα ευρείας περιοχής χρησιμοποιούν τηλεφωνικά δίκτυα ή τηλεπικοινωνιακούς δορυφόρους.
- Διαδίκτυα: Τα «διαδίκτυα» είναι δίκτυα ευρείας περιοχής τα οποία καλύπτουν γεωγραφικές περιοχές μίας ή περισσότερων ηπείρων διασυνδέοντας επιμέρους δίκτυα. Σε ένα διαδίκτυο μπορεί να συνυπάρχουν διασυνδεδεμένοι υπολογιστές και δίκτυα που χρησιμοποιούν διαφορετικές τεχνολογίες και λειτουργικά συστήματα. Το Διαδίκτυο (Internet) είναι το μεγαλύτερο τέτοιου είδους δίκτυο.

1.3 Πλεονεκτήματα χρήσης δικτύου μετεωρολογικών σταθμών

Ένα βασικό μειονέκτημα των περισσότερων μετεωρολογικών σταθμών είναι ότι απαιτείται η φυσική παρουσία του χρήστη στην τοποθεσία που βρίσκεται εγκατεστημένος ο σταθμός. Το πρόβλημα αυτό αντιμετωπίζεται εν μέρη στα πιο ακριβά μοντέλα τα οποία διαθέτουν ασύρματη μετάδοση των δεδομένων τους (Wi-fi, κεραίες κ.λ.π.).

Οι υπό κατασκευή σταθμοί θα πρέπει να έχουν όσο το δυνατόν μεγαλύτερο εύρος δράσης. Σε αυτήν την περίπτωση η ιδανική λύση που προτείνεται είναι η σύνδεση του σταθμού στο διαδίκτυο, καθώς οι τεχνολογίες του μας επιτρέπουν να εκτελούμε εργασίες οι οποίες μας διευκολύνουν είτε σε επαγγελματικό είτε σε προσωπικό επίπεδο. Με αυτόν τον τρόπο, η προβολή των δεδομένων θα γίνεται μέσω ιστοσελίδας έτσι ώστε με την χρήση ενός προσωπικού υπολογιστή ή ακόμα ενός «έξυπνου» κινητού τηλεφώνου θα επιτρέπεται η πρόσβαση στα δεδομένα του σταθμού από οποιοδήποτε σημείο και να βρίσκεται ο χρήστης, χωρίς επιπλέον κόστος.

Η κατασκευή ενός δικτύου σταθμών μας δίνει την δυνατότητα σύγκρισης περισσότερων δεδομένων και άρα την διεξαγωγή ασφαλέστερων και ακριβέστερων συμπερασμάτων για τα μετεωρολογικά φαινόμενα μιας περιοχής σε σχέση με αυτά ενός μόνου σταθμού.

Το δίκτυο που κατασκευάστηκε για τις ανάγκες της παρούσας εργασία είναι διασπασίμο, επεκτάσιμο, μετακινήσιμο και καλύπτει την περιοχή της πόλης της Ξάνθης. Οι σταθμοί είναι τελείως αυτόνομοι μεταξύ τους και με τον εξυπηρετητή. Ένας χρήστης με τις κατάλληλες (λίγες) οδηγίες μπορεί να πάρει τον αριθμό των

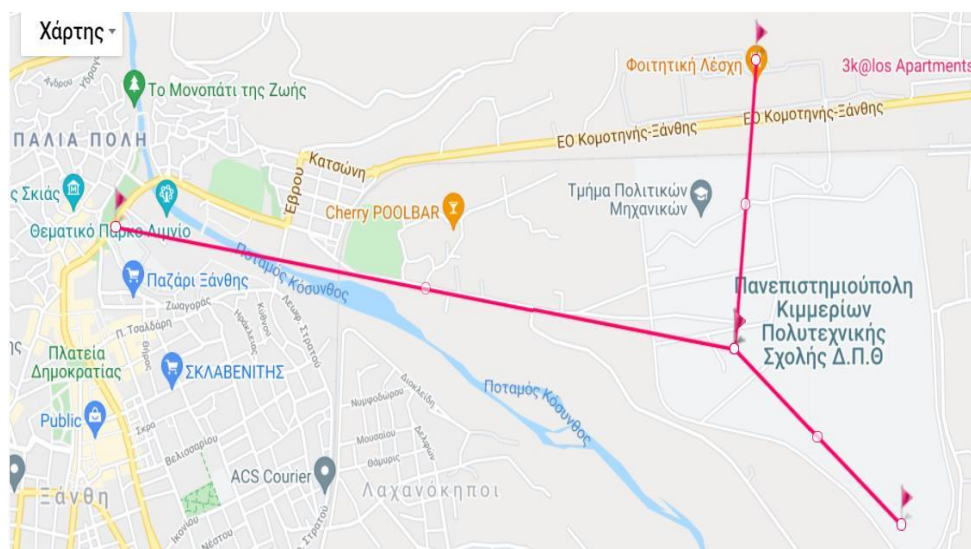
σταθμών που χρειάζεται για να καλύψει μια περιοχή και τον server , να τους τοποθετήσει εκεί που χρειάζεται και το σύστημα να δουλεύει κανονικά.

1.4 Δομή δικτύου μετεωρολογικών σταθμών

Τοποθεσία σταθμών :

- Τμήμα ΗΜΜΥ (<100m)
- Εστίες (730m)
- Ερευνητικό ΑΘΗΝΑ(740m)
- ΠΡΟΚΑΤ(2000m)

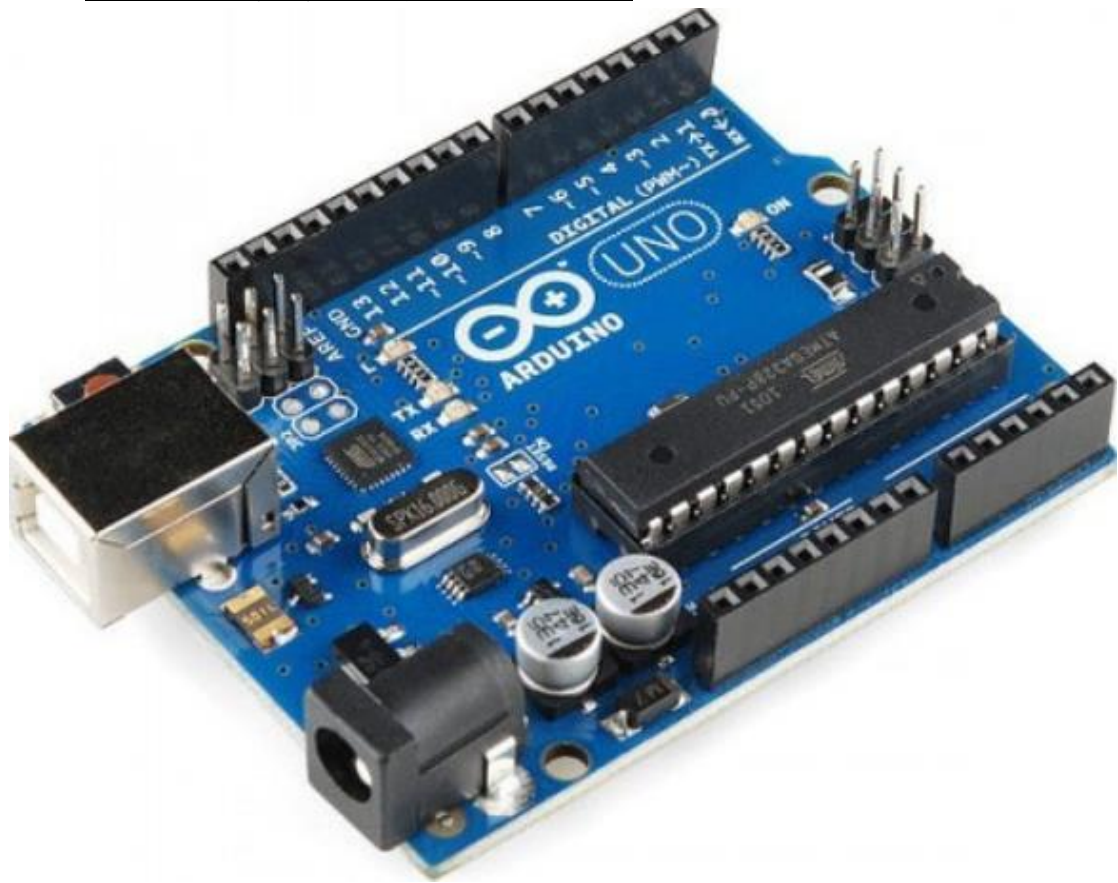
Η επιλογή των θέσεων των σταθμών έγινε με βάση τρεις παράγοντες συνδυαστικούς μεταξύ τους. Πρώτος παράγοντας, ήταν να υπήρχε η δυνατότητα εγκατάστασης του σταθμού σε ασφαλή τοποθεσία για να ελαχιστοποιηθεί το ενδεχόμενο κλοπής ή κάποιας βλάβης από αστάθμητους παράγοντες. Δεύτερος παράγοντας, ήταν να καλύπτει όσο το δυνατόν καλύτερα την γεωγραφική περιοχή για αξιόπιστη αποτύπωση των μετεωρολογικών δεδομένων αυτής. Τρίτος παράγοντας, ήταν οι αποστάσεις των σταθμών από τον εξυπηρετητή να είναι τέτοιες, ώστε να δίνετε η επιλογή υλοποίησης πρωτοκόλλων επικοινωνίας με διαφορετικές δυνατότητες στο εύρος ζώνης. Πρακτικά καταλήξαμε τα πρωτόκολλα επικοινωνίας, με βάση την απόσταση, να μπορούν να υλοποιηθούν σε κάθε σταθμό. Οι αποστάσεις των σταθμών καταγράφονται μέσα στις παρενθέσεις δίπλα από κάθε σταθμό.



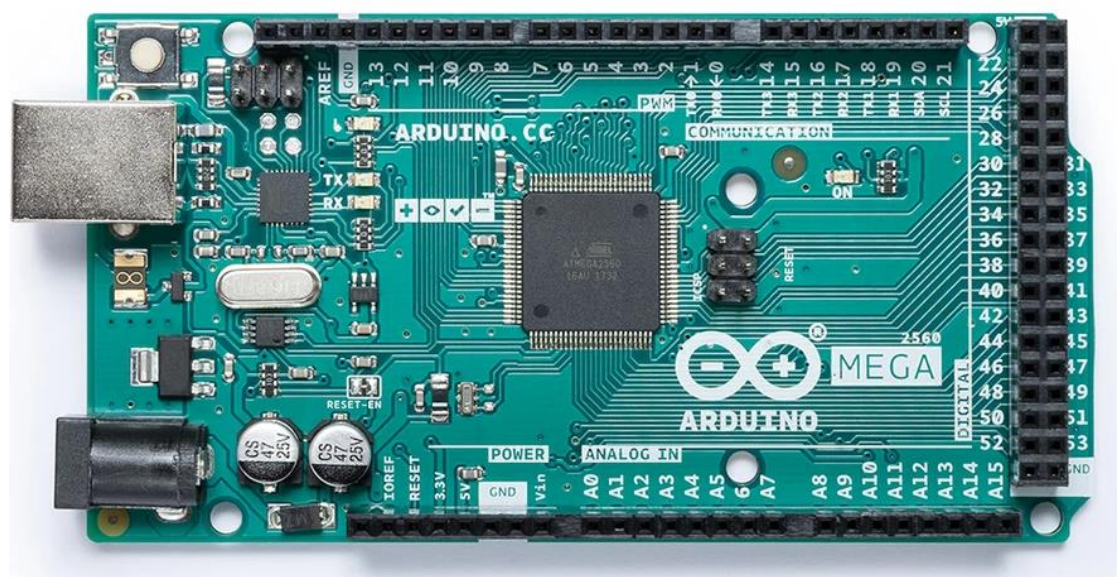
Εικόνα 1. 1 Τοπολογία σταθμών

2 ΚΕΦΑΛΑΙΟ ΔΕΥΤΕΡΟ

2.1 Τι είναι οι μικροελεγκτές Arduino;



Εικόνα 2. 1 Arduino UNO



Εικόνα 2. 2 Arduino MEGA

Το Arduino είναι ένα εργαλείο, μια πλακέτα, για να κατασκευάσουμε ένα υπολογιστικό σύστημα με την έννοια ότι αυτό ελέγχει συσκευές του φυσικού κόσμου, σε αντίθεση με τον κοινό Ηλεκτρονικό Υπολογιστή. Είναι ανοιχτού υλικού και λογισμικού και βασίζεται σε μια αναπτυξιακή πλακέτα που ενσωματώνει επάνω έναν μικροελεγκτή και συνδέεται με τον Η/Υ για να το προγραμματίσουμε μέσα από ένα απλό περιβάλλον ανάπτυξης. Ένα Arduino μπορεί να χρησιμοποιηθεί για να αναπτύξουμε διαδραστικά αντικείμενα, να δεχτούμε εισόδους από πληθώρα αισθητηρίων οργάνων και διακόπτες, αλλά και να ελέγχουμε διάφορα φώτα, κινητήρες και άλλες συσκευές εξόδου του φυσικού κόσμου. Τα Projects του εν λόγω Μικροελεγκτή μπορούν να είναι αυτόνομα (σε επίπεδο hardware) ή να επικοινωνούν με κάποιο software στον Η/Υ του προγραμματιστή (προγράμματα όπως τα Flash, Processing, MaxMSP). Οι πλακέτες μπορούν εύκολα να συναρμολογηθούν ακόμη και από έναν αρχάριο ή να αγοραστούν μονταρισμένες. Το περιβάλλον ανάπτυξης του λογισμικού βασίζεται στην γλώσσα προγραμματισμού Processing και την γλώσσα προγραμματισμού Wiring, οι οποίες είναι ανοιχτού κώδικα (open source) και μπορεί κάποιος να τις "κατεβάσει δωρεάν". Η Γλώσσα προγραμματισμού του Arduino αποτελεί μια εφαρμογή σε software επίπεδο της καλωδίωσης. Εξομοιώνει θα λέγαμε απόλυτα το φυσικό περιβάλλον του μικροελεγκτή.

2.1.1 Γιατί να επιλέξω το Arduino;

Υπάρχει πληθώρα άλλων μικροελεγκτών και αναπτυξιακών στο εμπόριο για να ασχοληθεί όποιος ενδιαφέρεται. Το Basic Stamp της Parallax, το BX-24 της Netmedia, το Handyboard του MIT και πολλά άλλα όμοιας λειτουργικότητας. Όλα αυτά τα εργαλεία που προαναφέραμε, είναι απλά και για τον αρχάριο χρήστη, καθώς "κρύβουν" τις δύσκολες λεπτομέρειες της αρχιτεκτονικής και επιτρέπουν τον άμεσο προγραμματισμό του μικροελεγκτή, προσφέροντας τα πάντα σε ένα και μόνο "πακέτο" έτοιμο για χρήση.

Το Arduino διαφέρει από τους προηγούμενους γιατί απλοποιεί την διαδικασία να δουλεύει κάποιος με μικροελεγκτές. Κάποια πλεονεκτήματα που προσφέρει σε σχέση με άλλους μικροελεγκτές για χρήση από δασκάλους, μαθητές και άλλους hobbίστες είναι τα παρακάτω:

- Φθηνό - Οι πλακέτες του Arduino είναι εξαιρετικά φθηνές σε σχέση με άλλες πλατφόρμες μικροελεγκτών. Ειδικά δε μπορεί με τα σχηματικά που κυκλοφορούν στο Internet να κατασκευάσει κάποιος την φθηνότερη εκδοχή ενός Arduino. Ωστόσο ακόμα και αν προμηθευτεί την έτοιμη (μονταρισμένη πλακέτα) αυτή θα κοστίσει το μέγιστο 50 ευρώ.
- Τρέχει σε διάφορα Λειτουργικά Συστήματα. Οι μηχανικοί λογισμικού, ανέπτυξαν το περιβάλλον προγραμματισμού του Arduino για Windows, Macintosh OSX και για λειτουργικά συστήματα Linux. Τα

περισσότερα συστήματα ανάπτυξης Μικροελεγκτών περιορίζονται στα Windows.

- Απλό, ξεκάθαρο προγραμματιστικό περιβάλλον. Το περιβάλλον προγραμματισμού ενός Arduino ενδείκνυται για αρχάριους, αλλά είναι ταυτόχρονα και ευέλικτο και για πιο προχωρημένους χρήστες.
- Ανοιχτού λογισμικού και λογισμικού που επεκτείνεται και παραμετροποιείται. Το software του Arduino διανέμεται με την μορφή εργαλείων ανοιχτού λογισμικού και είναι διαθέσιμο προς επέκταση για έμπειρους προγραμματιστές. Η γλώσσα προγραμματισμού του μπορεί να επεκταθεί διαμέσου των βιβλιοθηκών της C++ και οι άνθρωποι που θέλουν να ασχοληθούν περισσότερο με τους μικροελεγκτές, μπορούν να μεταβούν από τον Arduino στην AVR C που είναι για προγραμματισμό των Atmel Μικροελεγκτών και η γλώσσα στην οποία βασίστηκε το λογισμικό του Arduino. Ομοίως μπορεί κάποιος να προσθέσει κώδικα της AVR-C στο πρόγραμμα που έχει γράψει για το Arduino του.
- Ανοιχτού Υλικού το οποίο μπορεί να επεκταθεί. Ο Arduino βασίζεται στους μικροελεγκτές της Atmel ATMEGA8 και ATMEGA168. Τα σχηματικά για τα αναπτυξιακά είναι κάτω από την άδεια της Creative Commons, επιτρέποντας σε έμπειρους σχεδιαστές να κατασκευάσουν το δικό τους αναπτυξιακό, εξελίσσοντας το ήδη υπάρχον χωρίς να έχουν νομικά προβλήματα. Η ακόμη καλύτερα, όχι τόσο έμπειροι χρήστες μπορούν να επιδιώξουν την αντιγραφή και κατασκευή της πλακέτας σε ράστερ για να καταλάβουν την λειτουργία ενός Arduino.

2.1.2 Τεχνικά χαρακτηριστικά του μικροελεγκτή Arduino Mega

MICROCONTROLLER	ATmega2560
OPERATING VOLTAGE	5V
INPUT VOLTAGE (RECOMMENDED)	7-12V
INPUT VOLTAGE (LIMIT)	6-20V
DIGITAL I/O PINS	54 (of which 15 provide PWM output)
ANALOG INPUT PINS	16
DC CURRENT PER I/O PIN	20 mA
DC CURRENT FOR 3.3V PIN	50 mA
FLASH MEMORY	256 KB of which 8 KB used by bootloader
SRAM	8 KB

EEPROM	4 KB
CLOCK SPEED	16 MHz
LED_BUILTIN	13
LENGTH	101.52 mm
WIDTH	53.3 mm
WEIGHT	37 g

Μερικά pins, έχουν συγκεκριμένες λειτουργίες:

- Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.
- External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). These pins can be configured to trigger an interrupt on a low level, a rising or falling edge, or a change in level.
- PWM: 2 to 13 and 44 to 46. Provide 8-bit PWM output with the `analogWrite()` function.
- SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). Αυτά τα Pins υποστηρίζουν το πρωτόκολλο επικοινωνίας SPI. Τα SPI pins υπάρχουν επίσης και στο ICSP header.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: 20 (SDA) and 21 (SCL). Support TWI communication using the Wire library. Note that these pins are not in the same location as the TWI pins on the old Duemilanove or Diecimila Arduino boards.

Το Mega 2560 έχει 16 αναλογικές εισόδους, κάθε μια από τις οποίες παρέχει 10 bits ανάλυση (1024 διαφορετικές τιμές). Από προεπιλογή μετράνε από την γείωση (0 volts) μέχρι τα 5 volts, παρόλα αυτά είναι δυνατόν να αλλάξει κάποιος το άνω άκρο χρησιμοποιώντας το AREF pin και την συνάρτηση `analogReference()`.

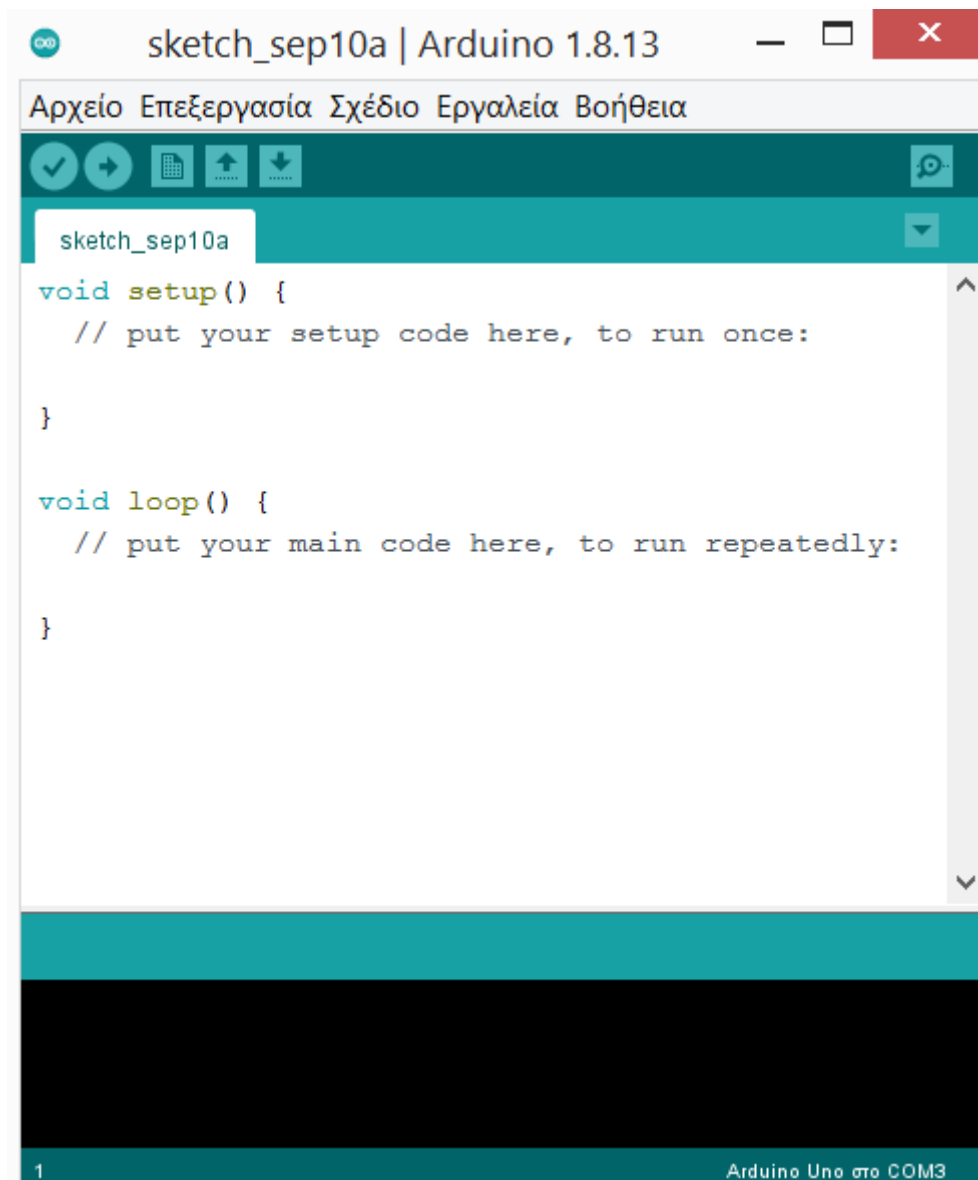
- AREF. Βάση τάσης για όλες τις αναλογικές εισόδους . Χρησιμοποιείται με την `analogReference()`.
- Reset. Όταν θέτουμε αυτό το pin στο LOW ο μικροελεγκτής κάνει reset.

2.1.3 Περιβάλλον ανάπτυξης

Το ολοκληρωμένο περιβάλλον ανάπτυξης του λογισμικού (Integrated Development Environment - IDE) έχει τη βάση του στις γλώσσες προγραμματισμού Processing και Wiring, είναι ανοιχτού κώδικα (open source) και μπορεί κάποιος να το κατεβάσει χωρίς κόστος. Πρόκειται για μια τροποποίηση της γλώσσας προγραμματισμού C/C++. Για τον προγραμματισμό του Arduino χρησιμοποιείται το Arduino IDE που περιέχει το περιβάλλον σύνταξης των εντολών, όπου κάθε νέο αρχείο ονομάζεται αυτόματα "sketch". Επίσης, περιλαμβάνονται κάποιες έτοιμες βιβλιοθήκες, εκ των οποίων μερικές μπορεί να χρειαστούν ενημέρωση(update), ή αλλιώς μπορεί κάποιος να κατεβάσει τις κατάλληλες βιβλιοθήκες που χρειάζεται για το τρέχον πρόγραμμα, εφόσον βρεθεί στο διαδίκτυο. Σε αυτή την περίπτωση είναι σημαντικό να αποθηκευτούν στον ίδιο φάκελο του "sketch" για να μπορεί να τις αναγνωρίσει το πρόγραμμα. Επίσης, έχει compiler (μεταγλωττιστή) για τη μεταγλώττιση του προγράμματος και ένα σειριακό παράθυρο, όπου μπορεί να εμφανίζει την σειριακή επικοινωνία, εφόσον αυτή είναι σε λειτουργία. Για τη σύνταξη του κώδικα υπάρχει μια πρότυπη μορφή :

- η συνάρτηση set up() και
- η συνάρτηση loop().

Πριν τη συνάρτηση set up() δηλώνονται οι μεταβλητές και εισάγονται οι απαραίτητες βιβλιοθήκες που θα χρησιμοποιηθούν μέσα στο πρόγραμμα. Μέσα στην συνάρτηση set up() αρχικοποιούνται οι μεταβλητές και ορίζονται οι ακροδέκτες . Να σημειωθεί, ότι αυτή η συνάρτηση εκτελείται μόνο μία φορά, επομένως ότι δηλωθεί σε αυτή ισχύει για όλο το πρόγραμμα. Αμέσως μετά, ακολουθεί η συνάρτηση loop() η οποία εκτελεί διαδοχικές λειτουργίες που επαναλαμβάνονται μέσα στο πρόγραμμα μέχρι να σταματήσει η τροφοδότηση του, ή να πατηθεί το κουμπί reset που υπάρχει πάνω στην πλακέτα



Εικόνα 2. 3 Περιβάλλον ανάπτυξης Arduino(ide)

2.2 Τι είναι το Raspberry pi;

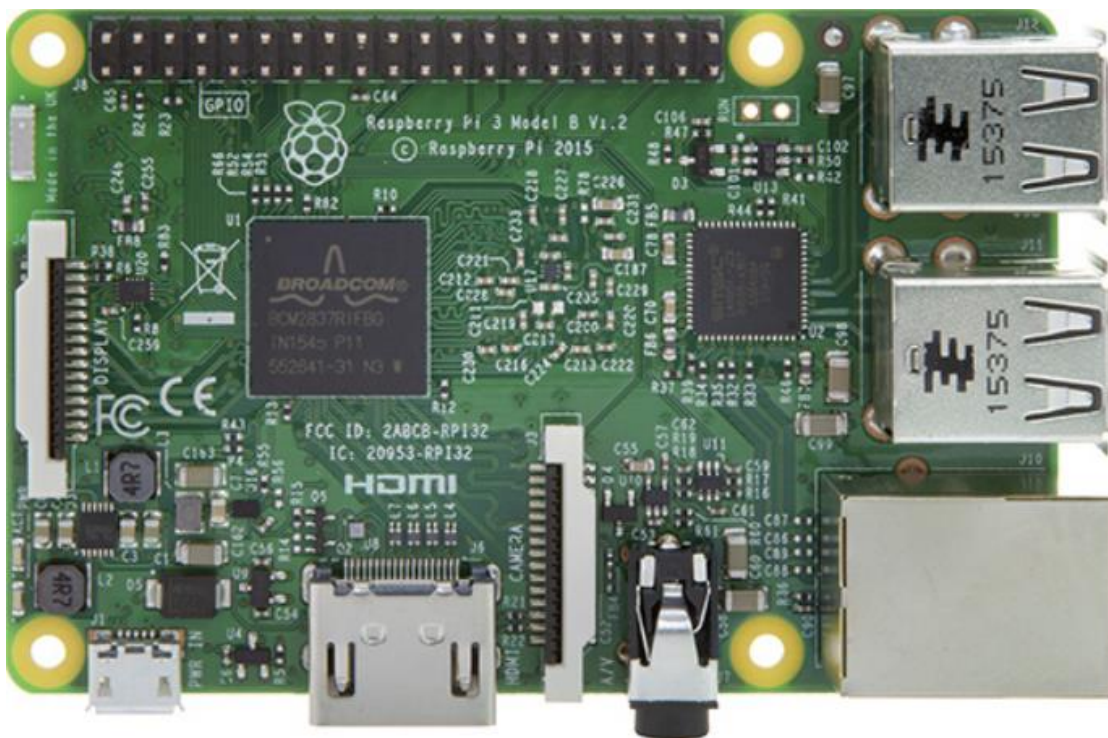
Το Raspberry pi ανήκει στην κατηγορία των “single board computers (SBC)”, δηλαδή μικροί υπολογιστές μονής πλακέτας. Είναι ένας πλήρης υπολογιστής μικρού μεγέθους, ο οποίος μπορεί να χρησιμοποιηθεί σε ηλεκτρονικές εφαρμογές ή ακόμα και για προσωπική χρήση εκτελώντας λειτουργίες ενός κανονικού υπολογιστή.

Τα κύρια χαρακτηριστικά του είναι:

- ο τετραπύρηνος επεξεργαστής Broadcom BCM2837 ARMv8 όπου τρέχει στα 1200MHz
- 40 pins γενικής χρήσης (General Purpose Input/Output – GPIO pins)

- 1GB RAM
- 4 θύρες USB
- μία θύρα HDMI
- μια θύρα Ethernet
- επαφή για σύνδεση με οθόνη και κάμερα χωριστά
- μια ενιαία θύρα για video και ήχο
- υποδοχή για Micro SD card
- τροφοδοσία μέσω Micro USB 5V

Υπάρχουν διάφορα μοντέλα στο εμπόριο, αλλά σε αυτή την εργασία χρησιμοποιήθηκε το Raspberry Pi 3 Model B. Κατά κύριο λόγο, το Raspberry Pi αναπτύχθηκε για εκπαιδευτικούς σκοπούς για να εισαχθεί στα σχολεία η βασική αρχή του προγραμματισμού.



Εικόνα 2. 4 Raspberry Pi 3 model b+

2.2.1 Γιατί να επιλέξω το Raspberry pi;

Υπάρχουν διάφοροι τρόποι να υλοποιήσει κάποιος ένα server. Η επιλογή της υλοποίησης του server της εργασίας έγινε βάσει χώρου, κόστους και ευκολίας επικοινωνίας. Το πρώτο που μελετήθηκε είναι η χρησιμοποίηση ενός πύργου, στην συνέχεια ενός Arduino και τέλος ενός raspberry pi.

- **Πύργος:** Το προφανές πλεονέκτημα ενός πύργου είναι η χωρητικότητα. Στην ουσία λόγω και των δεδομένων του project μπορούμε να έχουμε άπειρα δεδομένα και άπειρο κώδικα. Στα μειονεκτήματα τώρα το κόστος είναι πολύ μεγάλο για ένα υπολογιστή που θα κάνει μόνο αυτή την δουλειά. Ωστόσο θα μπορούσε να χρησιμοποιηθεί ένας υπάρχων υπολογιστής του εργαστηρίου που χρησιμοποιείται και για άλλες δουλειές, αλλά θα έπρεπε να είναι μόνιμα ανοιχτός και συνδεδεμένος στο Internet με ότι σημαίνει αυτό και για την κατανάλωση του ρεύματος. Ένα ακόμη σοβαρό μειονέκτημα είναι στην επικοινωνία. Θα έπρεπε να χρησιμοποιηθεί ο μετεωρολογικός σταθμός του τμήματος HMMY και ως κεντρικός σταθμός που θα στέλνονταν τα δεδομένα από τους δύο σταθμούς που χρησιμοποιούν κεραίες. Από εκεί μέσω Wi-fi τα δεδομένα θα στέλνονταν στον πύργο-server. Η εναλλακτική του να συνδέαμε την κεραία στο πύργο απαιτούσε γνώσεις και δουλειά πολύ υψηλού επιπέδου.
- **Arduino:** Τα βασικά πλεονεκτήματα της χρήσης ενός Arduino ως server είναι η ευκολία επικοινωνίας με τους μετεωρολογικούς σταθμούς (τα άλλα Arduino) καθώς και το σχετικά χαμηλό κόστος υλοποίησης και λειτουργίας. Παρόλα αυτά υπάρχει ένα πολύ σοβαρό μειονέκτημα. Από άποψη χωρητικότητας για τα δεδομένα και τον κώδικα θα έχουμε πρόβλημα. Το Arduino μην ξεχνάμε δεν έχει της δυνατότητες ενός υπολογιστή και δεν θα πρέπει να χρησιμοποιείται ως τέτοιος στα Project, παρά μόνο για κάποιες πολύ ειδικές περιπτώσεις.
- **Raspberry Pi:** Είναι η ιδανική περίπτωση για server μικρομεσαίου μεγέθους. Είναι στην ουσία ένας υπολογιστής πολύ χαμηλού κόστους. Έχει ως ένα βαθμό το πλεονέκτημα που έχει ένας υπολογιστής στη χωρητικότητα χωρίς να έχει το κόστος και τα μειονεκτήματα που έχει ένας κανονικός σταθερός υπολογιστής. Το κόστος του όσον αφορά στην αγορά και στην κατανάλωση ρεύματος μας επιτρέπει να αφιερώσουμε ένα Pi για να τρέχει ως server 24 ώρες το 24ώρο. Ακόμη το να συνδεθεί και να προγραμματιστεί η κεραία ως δέκτης στην πλακέτα είναι σχετικά εύκολη δουλειά, επιτρέποντας μας να μην επιβαρύνουμε με δουλειές το Arduino σταθμό των HMMY όπως απαιτεί η υλοποίηση με πύργο.

2.2.2 Περιβάλλον ανάπτυξης

Στο Raspberry Pi πρέπει να εγκατασταθεί το λειτουργικό σύστημα Raspbian, το οποίο βασίζεται σε Debian Linux. Αυτό γίνεται εύκολα με την εγκατάσταση των αρχείων του φακέλου NOOBS, το οποίο υπάρχει διαθέσιμο στο διαδίκτυο, στην κάρτα Micro SD.

Το Raspbian διαθέτει εργαλεία για :

- προβολή εικόνων,
- διαχείριση αρχείων και φακέλων,
- σημειωματάριο,

- την αντίστοιχη γραμμή εντολών (command prompt - cmd)
- και άλλες βασικές λειτουργίες ενός κανονικού υπολογιστή.

Μπορεί να γίνει περιήγηση στο Internet από δύο διαφορετικά προγράμματα περιήγησης (browsers). Τέλος, μας δίνει τη δυνατότητα προγραμματισμού με μια γλώσσα υψηλού επιπέδου, την Python, Integrated development and learning environment - IDLE σε δυο εκδόσεις, αλλά και με τη Scratch, η οποία χρησιμοποιείται περισσότερο για εκπαιδευτικούς σκοπούς, καθώς και τη Squeak, η οποία είναι η πλατφόρμα που τρέχει η Scratch.

2.3 Αισθητήρια

Αισθητήριο είναι μια συσκευή η οποία καταμετρά ένα φυσικό μέγεθος και το μετατρέπει σε μετρήσιμη έξοδο. Πιο συγκεκριμένα, τα αισθητήρια που χρησιμοποιήθηκαν στις παρούσες κατασκευές είναι:

- το DHT22-11 για τη μέτρηση της θερμοκρασίας και της υγρασίας
- το LPS22HB για τη μέτρηση της πίεσης και της θερμοκρασίας
- Το TSL2591X για τη μέτρηση της φωτεινότητας
- το UV Detection Sensor για τη μέτρηση της ηλιακής ακτινοβολίας

Τα αισθητήρια αυτά συνδέονται στις ψηφιακές και αναλογικές εισόδους του Arduino, το οποίο κάνοντας χρήση των κατάλληλων βιβλιοθηκών, διαβάζει τις τιμές τάσης στις εισόδους του και στην συνέχεια μετατρέπει τις εισερχόμενες τάσεις σε τιμές τις οποίες μπορούμε να αξιοποιήσουμε. Όλα τα αισθητήρια έχουν κάποια τεχνικά χαρακτηριστικά, γνωστά και ως datasheets, που δημοσιεύονται από την κατασκευαστική εταιρεία τους. Κάποια από αυτά θα αναλυθούν συνοπτικά παρακάτω για την κατανόηση του τρόπου λειτουργίας του κάθε αισθητήριου.

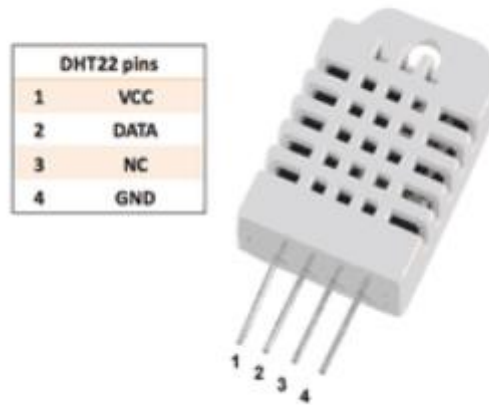
2.3.1 Αισθητήριο DHT-22

Το DHT-22 είναι ένα αισθητήριο που χρησιμοποιείτε για την εύρεση της σχετικής υγρασίας και θερμοκρασίας στον χώρο. Το DHT-22 είναι ένα βασικό, χαμηλού κόστους, αισθητήριο για την εύρεση της υγρασίας και της θερμοκρασίας στο χώρο. Στο εσωτερικό του κρύβει ένα αισθητήριο υγρασίας και ένα θερμίστορ (μεταβλητή αντίσταση που η τιμή της αλλάζει σε σχέση με τη θερμοκρασία) «διαβάζοντας» έτσι τον αέρα που το περιβάλλει.

Οι συνδέσεις είναι απλές, το πρώτο pin από αριστερά στα 3-5V, το δεύτερο pin (data) σε μια ψηφιακή είσοδο και το pin δεξιά στη γείωση.

Τεχνικές πληροφορίες:

- Πηγή : 3-5V
- Μέγιστο ρεύμα: 2.5mA
- Υγρασία: 0-100%, ακρίβεια 2-5%
- Εύρος μέτρησης και ακρίβεια: -40 to 80°C, ακρίβεια $\pm 0.5^{\circ}\text{C}$



Εικόνα 2. 5 Αισθητήριο DHT22

2.3.2 Αισθητήριο LPS22HB

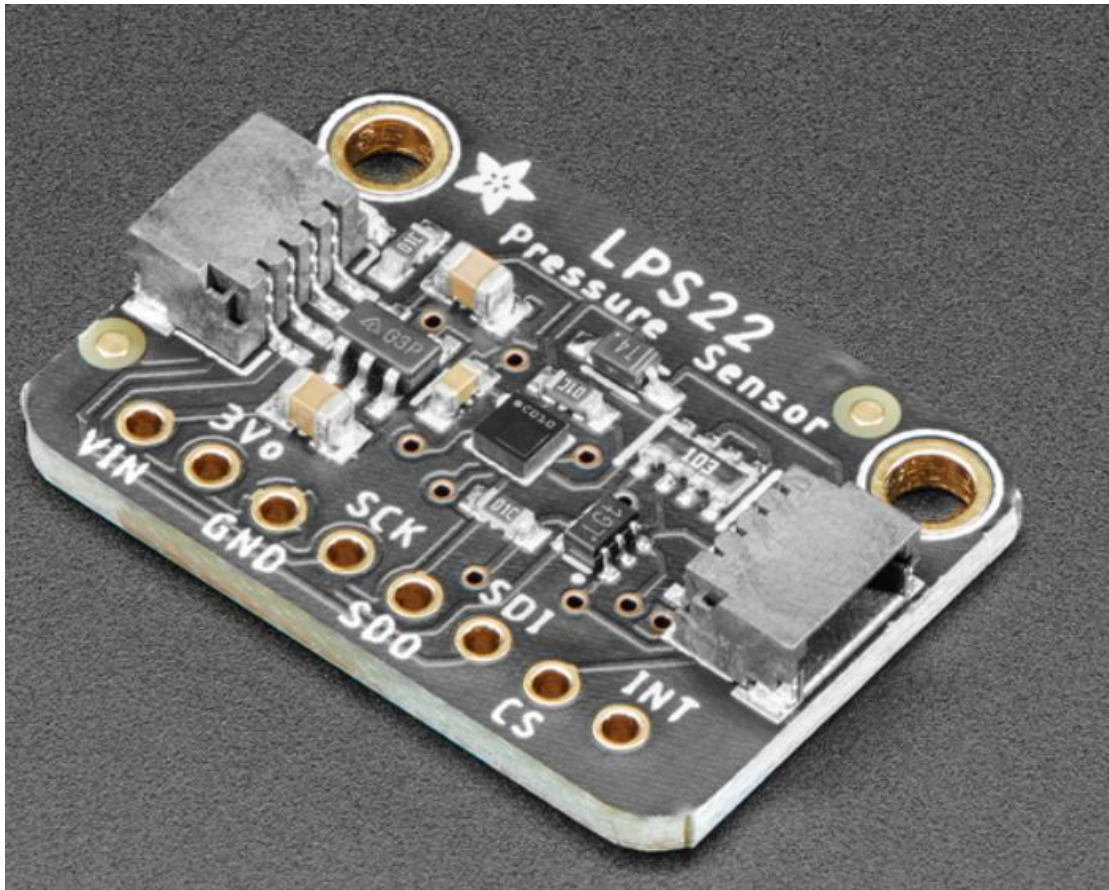
Το LPS22HB είναι ένα αισθητήριο που χρησιμοποιείτε για την εύρεση της πίεσης του αέρα και της θερμοκρασίας στον χώρο. Το LPS22 είναι, σύμφωνα με την εταιρία που τον κατασκευάζει(ST), το μικρότερο αισθητήριο στον κόσμο, με διαστάσεις μόνο 2x2mm. Χρησιμοποιείται σε μια ποικιλία εφαρμογών που έχουν ανάγκη να χρησιμοποιούν την ατμοσφαιρική πίεση. Έχει εύρος μετρήσεων από 260-1260hPa, με 24-bit δεδομένα μετρήσεων πίεσης, με μέγιστη συχνότητα 75Hz, οπότε είμαστε σίγουροι πως όποια στιγμή θέλουμε μπορούμε να έχουμε μια ενημερωμένη και σχετικά ακριβή μέτρηση. Το αισθητήριο μετράει την πίεση με μια απόκλιση της τάξης του $\pm 0,1$ hPa μετά την παραμετροποίηση του(± 1 hPa πριν την παραμετροποίηση)

Τεχνικές πληροφορίες:

- 260 to 1260 hPa absolute pressure range
- High overpressure capability: 20x full-scale
- Embedded temperature compensation
- 24-bit pressure data output
- 16-bit temperature data output
- ODR from 1 Hz to 75 Hz
- SPI and I²C interfaces
- Embedded FIFO
- Interrupt functions: Data Ready, FIFO flags, pressure thresholds
- High shock survivability: 22,000 g
- Small and thin package

Breakout Specs:

- Default I2C address 0x5D, can be set to 0x5C with jumper
- 3-5V power and logic compatible for I2C or SPI usage



Εικόνα 2. 6 Αισθητήριο LPS22HB

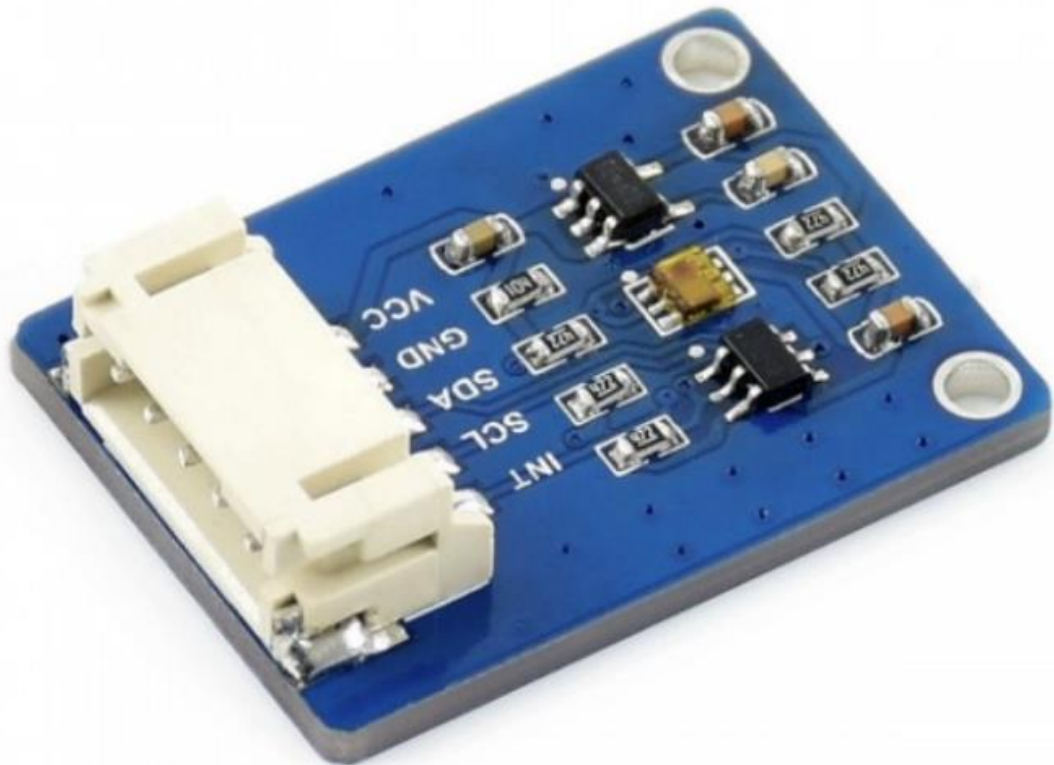
2.3.3 TSL2591X-Light Sensor

Το TSL2591X είναι ένα αισθητήριο που χρησιμοποιείτε για την εύρεση της φωτεινότητας του χώρου στον οποίο βρίσκεται. Το TSL2591X είναι ένα υψηλής ευαισθησίας ψηφιακό αισθητήριο, ικανό για μετρήσεις από μηδέν μέχρι και 88000 lux. Χρησιμοποιεί το πρωτόκολλο I2C για επικοινωνία με τις πλακέτες. Είναι χαμηλής κατανάλωσης και χρησιμοποιείται για ποικιλία καταστάσεων φωτεινότητας. Βασικό του πλεονέκτημα είναι η λογαριθμική κλίμακα που χρησιμοποιεί για τα επίπεδα φωτός.

Το TSL2591X είναι λογαριθμικό πάνω σε μια μεγάλη δυναμική κλίμακα από 0 έως 88.000 Lux, οπότε και έχει μεγάλη ευαισθησία στις σκοτεινές περιοχές και δύσκολα «μαξάρει» στις περιοχές με πολύ φως, σε αντίθεση με τους απλούς αισθητήρες που μετράνε την φωτεινότητα (π.χ. μια φωτοαντίσταση). Μπορεί να χρησιμοποιηθεί λοιπόν, τόσο σε εξωτερικούς όσο και σε εσωτερικούς χώρους, χωρίς να χρειαστεί πολλές αλλαγές στον κώδικα για παραμετροποίηση.

Τεχνικές πληροφορίες:

- Τάση από 2,3-6V
- Για board 68kΩ αντίσταση για max 3V αναλογική έξοδο
- 0,2 γραμμάρια
- 0.4" x 0.5" x 0.06" (10mm x 13mm x 1.5mm)
- 0.1" (2.54mm) mounting hole














Εικόνα 2. 7 Αισθητήριο TSL2591X

2.3.4 Αισθητήριο UV Detection

Το UV detection είναι ένα αισθητήριο που χρησιμοποιείτε για την εύρεση του δείκτη ηλιακής ακτινοβολίας του χώρου στον οποίο βρίσκεται. Έχει σχεδιαστεί για εφαρμογές που απαιτούν μεγάλη αξιοπιστία και μεγάλη ακρίβεια της μέτρησης του δείκτη UV.

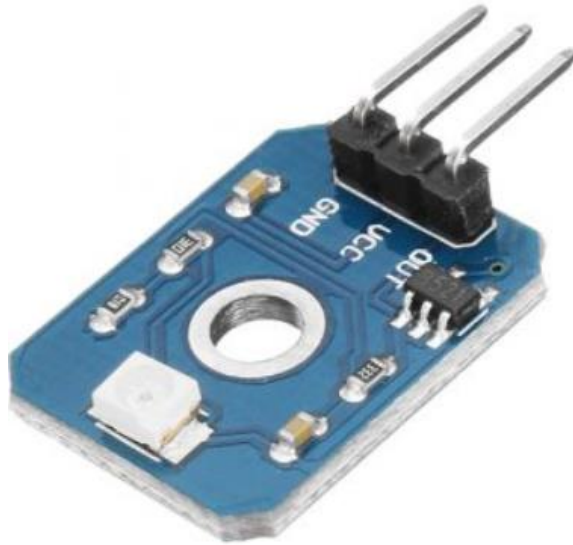
UV index chart:

UV Index	0					
Vout(mV)	<50	227	318	408	503	606
UV Index						
Vout(mV)	696	795	881	976	1079	1170+

Εικόνα 2. 8 Πίνακας δείκτη UV

Τεχνικές πληροφορίες:

- Τάση λειτουργίας από 3,3-5V
- Τάση εξόδου 0-1V
- Ακρίβεια μέτρησης $\pm 1\text{UV}$
- Χρόνος απόκρισης μικρότερος των 0,5s(συχνότητα >2Hz)
- Response wavelength: 200nm-370nm
- Detect UV wavelength: 200-370nm
- Μέγεθος 19.80x15mm



Εικόνα 2. 9 Αισθητήριο UV-Light

3 ΚΕΦΑΛΑΙΟ ΤΡΙΤΟ

3.1 Επικοινωνία & Πρωτόκολλα

Για καλύτερη κατανόηση του δικτύου πρέπει να αναλυθούν τόσο ο τρόπος επικοινωνίας των αισθητηρίων με την πλακέτα, όσο και ο τρόπος επικοινωνίας των σταθμών-πλακέτων με τον κεντρικό server. Για όλες τις επικοινωνίες χρησιμοποιήθηκαν υπάρχοντα πρωτόκολλα τα οποία «τροποποιήθηκαν» καταλλήλως ώστε να καλύπτονται οι ανάγκες του κάθε υποσυστήματος.

3.1.1 Πρωτόκολλα και επικοινωνία αισθητηρίων με το Arduino

3.1.1.1 UART PROTOCOL:

Το UART είναι συντομογραφία για το Universal Asynchronous Reception Transmission . Αυτό το πρωτόκολλο επιτρέπει στο Arduino να επικοινωνεί με σειριακές συσκευές. Αναπτύχθηκε από τον Gordon Bell στην Digital Equipment Corporation το 1960. Το σύστημα επικοινωνεί μέσω των ψηφιακών Pins 0 και 1 με τον υπολογιστή, μέσω της θύρας USB. Τα δεδομένα μεταφέρονται ως μια σειρά από bits στην μια γραμμή που είναι για μεταφορά δεδομένων (TX). Η άλλη γραμμή είναι για την υποδοχή των δεδομένων (RX). Μια συσκευή για να μπορεί να χρησιμοποιήσει αυτό το πρωτόκολλο πρέπει να έχει το κατάλληλο hardware (π.χ. το Arduino). Οι συσκευές που επικοινωνούν με αυτό τον τρόπο λαμβάνουν και στέλνουν επιπλέον πληροφορίες που αφορούν το τέλος και την αρχή των δεδομένων για να καταλαβαίνουν πότε ένα μήνυμα έχει σταλθεί σωστά. Το πρωτόκολλο είναι ασύγχρονο καθώς δεν βασίζεται σε ρολόι για τις επικοινωνίες του. Ο βασικός σκοπός του είναι η αποστολή και η λήψη σειριακών δεδομένων.

3.1.1.2 SPI PROTOCOL:

Το Serial Protocol Interface η αλλιώς SPI είναι ένα σύνηθες πρωτόκολλο επικοινωνίας, που χρησιμοποιείται για two way επικοινωνία μεταξύ δυο ή και παραπάνω συσκευών. Αναπτύχθηκε από την Motorola το 1980. Αντίθετα με τα ασύγχρονα πρωτόκολλα, το SPI δεν είναι συμμετρικό αλλά σύγχρονο, δηλαδή χρησιμοποιεί το σήμα του ρολογιού. Το μόνο που απαιτείται είναι η συχνότητα του ρολογιού να είναι χαμηλότερη από τις μέγιστες συχνότητες των συσκευών που χρησιμοποιούνται. Υπάρχουν 4 διαφορετικά μοντέλα που όλα έχουν να κάνουν με το σήμα του ρολογιού. Τα 4 αυτά μοντέλα προκύπτουν από 2 παραμέτρους, τις CPOL και CPHA. Το CPOL η αλλιώς clock polarity και δείχνει την καθορισμένη τιμή(υψηλή/χαμηλή) του ρολογιού όταν δεν μεταφέρονται δεδομένα. Το CPHA η αλλιώς clock phase καθορίζει σε ποια φάση του ρολογιού θα κάνουμε δειγματοληψία (στην άνοδο/ στην κάθοδο). Το SPI χρησιμοποιείται για την αποστολή δεδομένων από έναν μικροεπεξεργαστή όπως το Arduino σε μικρότερα περιφερειακά όπως καταχωρητές, αισθητήρια και SD κάρτες. Ο μικροεπεξεργαστής στην ουσία είναι ο αφέντης και τα περιφερειακά οι σκλάβοι. Οι σκλάβοι θα έχουν όλοι τρία ίδια κοινά καλώδια(CLK,MISO,MOSI) με τον αφέντη και διαφορετικό chip select μεταξύ τους. Οπότε ο αριθμός των μέγιστων περιφερειακών

σκλάβων που μπορούμε να έχουμε είναι ανάλογος των πόσων Pins έχουμε στην διάθεση μας για CS.

3.1.1.3 I2C PROTOCOL:

Το I2C πρωτόκολλο είναι ένας συνδυασμός του SPI και των UARTs. Το I2C η αλλιώς interintegrated circuit, αναπτύχθηκε από την Philips στις αρχές της δεκαετίας του 80. Χρησιμοποιώντας το I2C μπορούμε να συνδέσουμε πολλαπλές συσκευές ως σκλάβους σε μια μόνο συσκευή αφέντη η ακόμα και πολλές συσκευές αφέντες σε μία η πολλές συσκευές σκλάβους. Π.χ. χρησιμοποιούμε το πρωτόκολλο I2C όταν θέλουμε παραπάνω από ένα μικροελεγκτή να στέλνουν δεδομένα στην ίδια SD κάρτα ή να στέλνουν κείμενα σε μια LCD οθόνη. Το πρωτόκολλο χρησιμοποιεί μόνο 2 καλώδια για τη μεταφορά δεδομένων μεταξύ συσκευών. Με αυτά, τα δεδομένα μεταφέρονται ως μηνύματα. Τα μηνύματα χωρίζονται σε πλαίσια δεδομένων. Κάθε μήνυμα έχει μια διεύθυνση πλαισίου, που περιέχει μια δυαδική διεύθυνση του σκλάβου και μια η παραπάνω, για τα πλαίσια που περιέχουν τα δεδομένα που θέλουμε να στείλουμε. Οι καταστάσεις όπως αρχή, τέλος, τα γράψε και διάβασε bits, τα ACK και NACK bits επίσης περιέχονται σε κάθε πλαίσιο δεδομένων του μηνύματος. Το I2C είναι ένα πολύ γνωστό και ευρέως χρησιμοποιούμενο πρωτόκολλο. Χρησιμοποιείται κυρίως για επικοινωνία μεγάλων αποστάσεων, σε αντίθεση με το SPI που χρησιμοποιείται κυρίως για επικοινωνίες μικρών αποστάσεων.

3.1.2 Πρωτόκολλα και επικοινωνία των Arduino με τον Server

3.1.2.1 Wi-Fi

Το πρωτόκολλο επικοινωνίας Wi-Fi επιτρέπει την ασύρματη σύνδεση συμβατών συσκευών. Αυτή τη στιγμή είναι τα πιο δημοφιλή πρωτόκολλα ασύρματης δικτύωσης. Ένα δίκτυο Wi-Fi αποτελείται από δύο και περισσότερες συσκευές. Η δημιουργία δικτύου έως και τριών υπολογιστών μπορεί να πραγματοποιηθεί μεταξύ τους (peer-to-peer) χωρίς να χρειάζεται επιπλέον εξοπλισμός, αρκεί να υποστηρίζουν Wi-Fi.

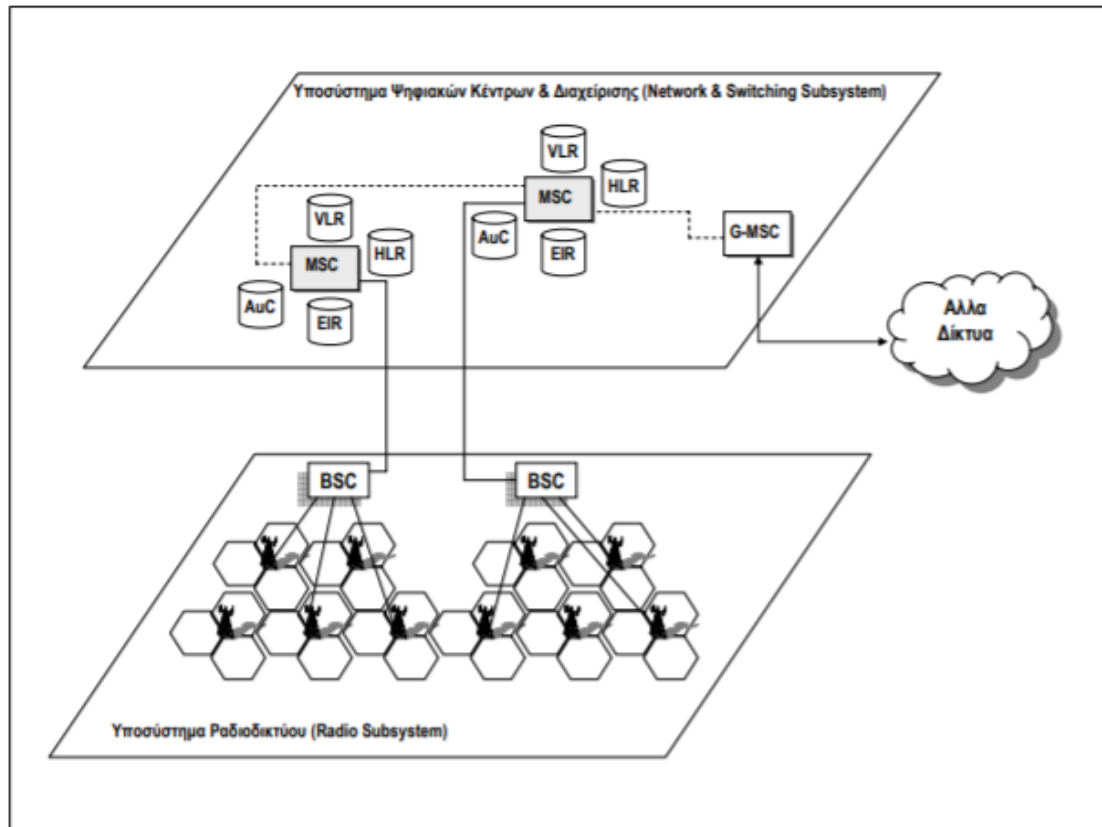
Με αυτό τον τρόπο θα μπορούν να ανταλλάσσουν μεταξύ τους δεδομένα και να μοιράζονται μια σύνδεση στο ίντερνετ. Από τρεις υπολογιστές ή συμβατές συσκευές Wi-Fi και πάνω χρειαζόμαστε επιπλέον εξοπλισμό όπως Gateway και Access Point οι οποίες ελέγχουν το δίκτυο και μοιράζουν σε αυτό τα δεδομένα. Αρκετοί υπολογιστές και περισσότερο φορητοί, υποστηρίζουν πρωτόκολλα Wi-Fi. Όσοι δεν ενσωματώνουν τέτοια χαρακτηριστικά μπορούν με κάρτες , αντάπτορες αλλά και εξωτερικές συσκευές να τα προσθέσουν στον υπολογιστή τους, επιτραπέζιο ή φορητό και PDA. Οι συσκευές Access Point και Gateway είναι η καρδιά του ασύρματου δικτύου. Στέλνουν και δέχονται ασύρματα σήματα προς και από τους ηλεκτρονικούς υπολογιστές, τα PDA αλλά και τα περιφερειακά όπως εκτυπωτές, επιτρέποντας να μοιράζονται σύνδεση στο ίντερνετ με άλλους χρήστες. Μέσω πρωτοκόλλου 802.11b

ένας μόνο χρήστης, θεωρητικά μπορεί να στείλει και να δεχθεί σήμα με ταχύτητα 11Mbps. Αν 10 χρήστες χρησιμοποιούν ταυτόχρονα το δίκτυο, θεωρητικά τους αναλογεί από 1Mbps. Αυτό όμως δεν είναι απόλυτο και έχει σχέση με το πόσοι χρήστες το χρησιμοποιούν ταυτόχρονα και τι εργασίες εκτελούν μέσω αυτού. Με τις αναβαθμίσεις των πρωτοκόλλων τα τελευταία χρόνια βλέπουμε όλο και μεγαλύτερες ταχύτητες στη λειτουργία του ίντερνετ.

3.1.2.2 GSM

Η ονομασία GSM προήλθε από το Global System for Mobile communications (GSM). Το GSM έχει προδιαγραφεί από τον Ευρωπαϊκό Οργανισμό Τηλεπικοινωνιακών Προτύπων και είναι ένα αποδεκτό πρότυπο παγκόσμιας εμβέλειας το οποίο υποστηρίζει αξιόπιστες ψηφιακές ραδιοεπικοινωνίες. Οι συχνότητες λειτουργίας, ανήκουν στις περιοχές των 900 MHz και 1800 MHz του ραδιοφάσματος συχνοτήτων. Στις Ηνωμένες Πολιτείες της Αμερικής, το πρότυπο αυτό λειτουργεί στις συχνότητες 800 MHz και 1900 MHz. Το σύστημα αυτό στηρίζεται στην κυψελωτή ιδέα (cellular) όπου η γεωγραφική περιοχή παροχής κινητών υπηρεσιών χωρίζεται σε ζώνες ή κυψέλες (cells) και με την φιλοσοφία της επαναχρησιμοποίησης των ιδίων συχνοτήτων μπορεί να εξυπηρετηθεί μεγάλος αριθμός κινητών συνδρομητών διασφαλίζοντας υψηλές απαιτήσεις ποιότητας παροχής υπηρεσιών (Quality of service – QoS). Στο πλαίσιο της υλοποίησης του GSM, κατά το χρονικό διάστημα από το 1982 έως το 1985 υπήρχε έντονος προβληματισμός από τις αντίστοιχες ομάδες εργασίες του ETSI για το αν το GSM θα έμενε στην αναλογική τεχνολογία ή θα υλοποιούνταν με ψηφιακή τεχνολογία. Μετά από πειραματικές μετρήσεις πεδίου πάνω σε πιλοτικά συστήματα αποφασίσθηκε το GSM να υιοθετήσει την ψηφιακή τεχνολογία. Επίσης, το επόμενο στάδιο ήταν ο προβληματισμός χρήσης στενοζωνικής (narrowband) ή ευρυζωνικής (broadband) λύσης. Τον Μάιο του 1987, αποφασίσθηκε η υιοθέτηση στενοζωνικής TDMA τεχνικής. Με τον τρόπο αυτό, το ψηφιακό ραδιοσύστημα θα μπορούσε να εξυπηρετήσει περισσότερους συνδρομητές συγκρινόμενο με ένα σύστημα το οποίο υιοθετεί FDMA τεχνική, διασφαλίζοντας κατ'αυτόν τον τρόπο τις απαιτήσεις για οικονομία του φάσματος συχνοτήτων χωρίς από την άλλη μεριά να επηρεάζεται η ποιότητα των παρεχόμενων υπηρεσιών. Εκτός από το GSM έχουν κατασκευασθεί από εταιρείες και άλλα κυψελωτά ραδιοδίκτυα τα οποία δεν υποστηρίζουν τις προδιαγραφές του ETSI. Στην περίπτωση αυτή και στο πλαίσιο της διασύνδεσης αυτών με την τεχνολογία GSM απαιτείται η χρήση ειδικών διεπαφών για τη διασφάλιση των ανομορφιών συμβατότητας κατά την μετάβαση της πληροφορίας από τη μια τεχνολογία στην άλλη.

3.1.2.2.1 Αρχιτεκτονική GSM



Εικόνα 3. 1 Αρχιτεκτονική GSM(e-class Πάτρας ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ ΚΑΙ ΠΡΩΤΟΚΟΛΛΑ ΔΙΚΤΥΩΝ II)

Όπου σύμφωνα με την εικόνα 3.1:

- Υποσύστημα Δικτύου Μεταγωγής (Network & Switching Subsystem – NSS) και Υποσύστημα Υποστήριξης Λειτουργίας (Operation Support Subsystem -OSS): Τα υποσυστήματα αυτά περιλαμβάνουν όλο το υλικό και λογισμικό προκειμένου να υποστηρίζονται όλες τις διαδικασίες μεταγωγής και ενσωματώνει και τις διαδικασίες διαχείρισης του δικτύου
- Υποσύστημα Ραδιοδικτύου (Radio Subsystem - RSS): Το υποσύστημα αυτό περιλαμβάνει όλο τον αναγκαίο ραδιο-εξοπλισμό προκειμένου να υποστηρίζονται όλες τις διαδικασίες οι οποίες λαμβάνουν χώρα στο ραδιοδίκτυο.

3.1.2.3 RF

RF η αλλιώς ραδιοσυχνότητα είναι μια ταλάντωση που μεταδίδεται με την ταχύτητα του φωτός, ένα ηλεκτρομαγνητικό κύμα στην περιοχή των χαμηλότερων συχνοτήτων ή των μεγαλύτερων μηκών κύματος του ηλεκτρομαγνητικού φάσματος (100000-10m). Αναφέρεται στις συχνότητες που πέφτουν στο ηλεκτρομαγνητικό φάσμα που σχετίζεται με τη διάδοση των ραδιοκυμάτων. Το ρεύμα RF δημιουργεί ηλεκτρομαγνητικά πεδία όταν εφαρμόζεται σε μια κεραία που μεταδίδει το

εφαρμοζόμενο σήμα μέσω του χώρου. Οι επικοινωνίες με βάση το ηλεκτρομαγνητικό κύμα έχουν χρησιμοποιηθεί για πολλές δεκαετίες, ειδικά για ασύρματες φωνητικές επικοινωνίες και επικοινωνίες δεδομένων. Η συχνότητα του σήματος RF είναι αντιστρόφως ανάλογη προς το μήκος κύματος του πεδίου. Ο ρυθμός ταλάντωσης για τις ραδιοσυχνότητες κυμαίνεται μεταξύ 30 KHz και 300 GHz.

Τα κύματα RF που έχουν ρυθμιστεί ώστε να περιέχουν πληροφορίες ονομάζονται σήματα ραδιοσυχνοτήτων. Αυτά τα σήματα ραδιοσυχνοτήτων έχουν ορισμένες συμπεριφορές που μπορούν να προβλεφθούν και να ανιχνευθούν και μπορούν να διασυνδεθούν με άλλα σήματα. Οι κεραίες πρέπει να χρησιμοποιούνται για τη λήψη των ραδιοφωνικών σημάτων. Αυτές οι κεραίες θα πάρουν περισσότερο αριθμό ραδιοφωνικών σημάτων κάθε φορά. Χρησιμοποιώντας ραδιοφωνικούς δέκτες μπορούν να ληφθούν συγκεκριμένες συχνότητες. Υπάρχουν διαθέσιμες κάποιες ελεύθερες ζώνες, οι οποίες χρησιμοποιούνται για τηλεχειρισμό εφαρμογών. Αυτές ονομάζονται επίσης ISM (βιομηχανικές, επιστημονικές και ιατρικές) ζώνες. Η πιο ελκυστική ζώνη συχνοτήτων είναι τα 434 MHz.

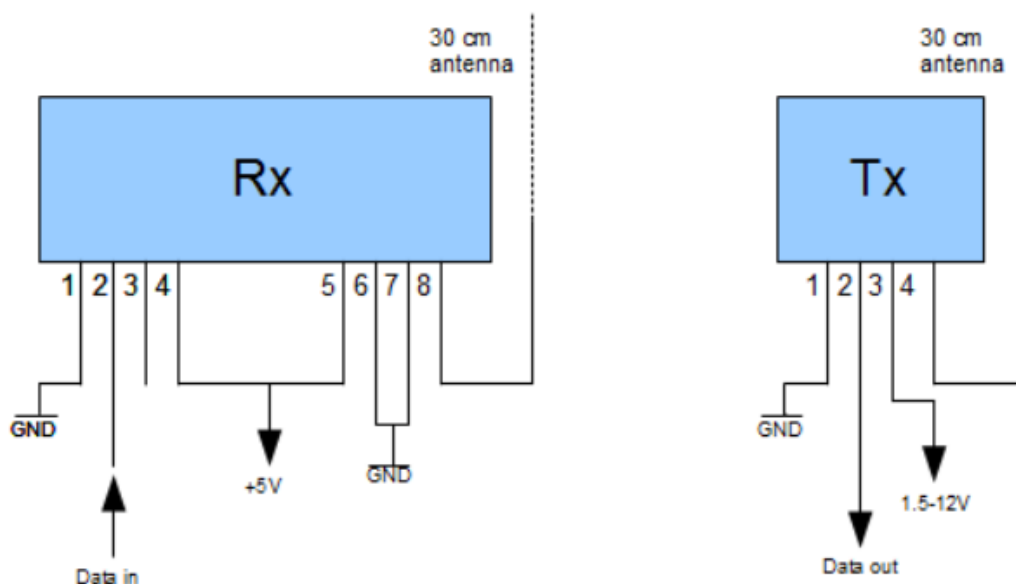
Τα δεδομένα ωφέλιμου φορτίου πρέπει να διαμορφώνονται στον φορέα RF. Η διαμόρφωση μετατόπισης πλάτους (ASK) και η διαμόρφωσή μετατόπισης συχνότητας (FSK) είναι δημοφιλείς για το σκοπό αυτό. Για λόγους κατανάλωσης ενέργειας, το ASK εφαρμόζεται ως επί το πλείστον ως κλειδί ON-OFF (OOK). Η πρόκληση είναι να βρεθεί ένας σχεδιασμός ή μια ιδέα της κεραίας που αντιπροσωπεύει έναν τέλει συμβιβασμό μεταξύ κόστους και απόδοσης. Ένας σαφής σχεδιασμός RF είναι απαραίτητος για την τήρηση των κανονισμών.

3.1.2.3.1 Αμφίδρομοι σύνδεσμοι για τηλεχειριστήριο RF:

Τα τηλεχειριστήρια υψηλής απόστασης μπορούν να χρησιμοποιηθούν με βάση αμφίδρομες συνδέσεις RF. Εκτός από τη σύνδεση για το τηλεχειριστήριο με την ελεγχόμενη συσκευή υπάρχει ένας πρόσθετος σύνδεσμος προς τα πίσω από τη συσκευή στον ελεγκτή. Αυτός ο προς τα πίσω σύνδεσμος μπορεί να χρησιμοποιηθεί για την εξασφάλιση της αντοχής του απομακρυσμένου συνδέσμου χρησιμοποιώντας πρωτόκολλα χειραψίας και παροχή ανατροφοδότησης στο χρήστη. Οι αμφίδρομοι σύνδεσμοι RF υλοποιούνται χρησιμοποιώντας ICs πομποδέκτη RF που περιλαμβάνουν έναν δέκτη ραδιοσυχνοτήτων και ένα πομπό ραδιοσυχνοτήτων που μοιράζονται ένα μόνο PLL και μία ενιαία κεραία.

3.1.2.3.2 Πρωτόκολλα επικοινωνίας RF:

Για τη βελτιωμένη ευρωστία της σύνδεσης RF, οι τιμές κυκλικού ελέγχου πλεονασμού (CRC) συχνά παράγονται και μεταδίδονται ως μέρος του πλαισίου. Ο δέκτης μπορεί να αναγνωρίσει με σαφήνεια τυχόν σφάλματα δυαδικών ψηφίων επανυπολογίζοντας τις τιμές CRC του ληφθέντος πλαισίου δεδομένων και να συγκριθεί με αυτό που δημιουργήθηκε πριν από τη μετάδοση. Το επίπεδο φόρτισης της μπαταρίας του πομπού μπορεί να σηματοδοτείται με πλήρες πεδίο δεδομένων 4-bit ή 8-bit που αντιπροσωπεύει τη μετρηθείσα τάση μπαταρίας. Τα συστήματα επιτρέπουν μία επικοινωνία μεταξύ δύο κόμβων, συγκεκριμένα τη μετάδοση και τη λήψη.



Εικόνα 10. Rx&Tx schematics

Οι μονάδες RF έχουν χρησιμοποιηθεί σε συνδυασμό με ένα σύνολο τεσσάρων καναλιών IC και αποκωδικοποιητή. HT-12E και HT-12D ή HT-640 και HT-648 είναι οι πιο συχνά χρησιμοποιούμενοι κωδικοποιητές και αποκωδικοποιητές αντίστοιχα στην επικοινωνία RF. Ο κωδικοποιητής χρησιμοποιείται για την κωδικοποίηση δεδομένων μετάδοσης ενώ η λήψη αποκωδικοποιείται από αποκωδικοποιητή. Ο κωδικοποιητής θα χρησιμοποιηθεί για τη σειριακή μετάδοση των δεδομένων αντί για την αποστολή

παράλληλων. Αυτά τα σήματα μεταδίδονται σειριακά μέσω RF στο σημείο λήψης. Ο αποκωδικοποιητής χρησιμοποιείται για την αποκωδικοποίηση των σειριακών δεδομένων στο δέκτη και καλύπτει ως παράλληλα δεδομένα.

3.1.2.3.3 Εφαρμογές επικοινωνίας RF

Η επικοινωνία RF χρησιμοποιείται κυρίως για ασύρματα δεδομένα, εφαρμογές φωνητικής μεταφοράς και εφαρμογές οικιακού αυτοματισμού, εφαρμογές τηλεχειρισμού και σε εφαρμογές με γνώμονα τη βιομηχανία.

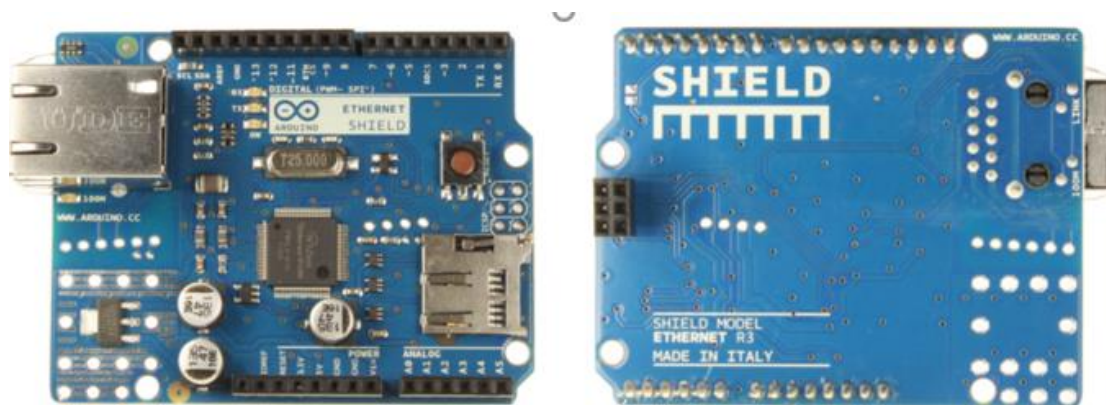
Για παράδειγμα, στις εφαρμογές οικιακού αυτοματισμού μπορούμε να χρησιμοποιήσουμε ελεγχόμενους διακόπτες RF αντί για συμβατικούς διακόπτες. Για το σκοπό αυτό, ένα τηλεχειριστήριο RF μπορεί να χρησιμοποιηθεί για τον έλεγχο των φώτων και άλλων συσκευών χωρίς να μετακινηθούν σε άλλα μέρη. Αυτή η εφαρμογή είναι ως επί το πλείστον χρήσιμη για άτομα με ειδικές ανάγκες. Σε βιομηχανικά προσανατολισμένες εφαρμογές για έλεγχο ρομπότ και οχημάτων μπορεί να χρησιμοποιηθεί επικοινωνία RF. Τα οχήματα ρομπότ που χρησιμοποιούνται γενικά σε επικίνδυνες επιχειρήσεις, δεν μπορούν να εκτελεστούν από ανθρώπους. Για το σκοπό αυτό απαιτείται μια μονάδα μετάδοσης για τον έλεγχο της κίνησης των οχημάτων ρομπότ.

Λόγω πολλών αιτιών η μετάδοση μέσω RF είναι καλύτερη από την υπέρυθρη ακτινοβολία. Πρώτον, το σήμα μέσω RF μπορεί να ταξιδέψει σε μεγαλύτερες αποστάσεις καθιστώντας το κατάλληλο για εφαρμογές μεγαλύτερης εμβέλειας. Το IR λειτουργεί ως επί το πλείστον σε κατάσταση ορατότητας, αλλά τα σήματα ραδιοσυχνοτήτων μπορούν να ταξιδεύουν ακόμη και όταν υπάρχει εμπόδιο μεταξύ του πομπού και του δέκτη. Η μετάδοση RF έχει υψηλή αξιοπιστία από τις απομακρυσμένες επικοινωνίες υπέρυθρων. Οι επικοινωνίες ραδιοσυχνοτήτων χρησιμοποιούν συγκεκριμένη συχνότητα, αλλά το IR δεν θα χρησιμοποιεί συγκεκριμένο εύρος και θα επηρεάζεται από άλλες πηγές εκπομπής IR.

3.2 Υλοποίηση των πρωτοκόλλων στην εργασία

Για την επικοινωνία των πλακετών των σταθμών με τον server στην εργασία, θα μπορούσε να χρησιμοποιηθεί μόνο ένα πρωτόκολλο επικοινωνίας. Όμως αυτό, ενώ θα ήταν πιο εύκολο πρακτικά, θα άφηνε το όλο σύστημα ευάλωτο στις εκάστοτε βλάβες που θα προκύπταν στο μέσο της επικοινωνίας που θα είχε χρησιμοποιηθεί. Για παράδειγμα αν χρησιμοποιούσαμε μόνο Wi-fi και αυτό έπεφτε σε κάποιες από τις περιοχές που θα είχαν εγκατασταθεί πλακέτες, θα έπαυε η επικοινωνία με τον εξυπηρετητή και θα χάναμε τελείως τα δεδομένα των πλακετών αυτών. Αντίστοιχα σε ένα δίκτυο που θα χρησιμοποιούσαμε μόνο κεραίες θα μπορούσε, λόγω αστάθμητων παραγόντων, να εισάγεται κάποιος θόρυβος στις συχνότητες που στέλνουν οι κεραίες και έτσι η επικοινωνία των σταθμών με τον server να σταματήσει, έχοντας ως αποτέλεσμα να χαθούν τα δεδομένα. Χρησιμοποιώντας τους 3 διαφορετικούς τρόπους επικοινωνίας (Wi-Fi, GSM, RF), που αναφέραμε και πιο πάνω, η πιθανότητα να σταματήσει η επικοινωνία του εξυπηρετητή με πάνω από μία πλακέτα τη φορά μικραίνει αισθητά (θα είχε ελαχιστοποιηθεί αν χρησιμοποιούσαμε τέσσερεις διαφορετικούς τρόπους, έναν για κάθε πλακέτα) και έτσι μειώνεται και η πιθανότητα απώλειας δεδομένων. Επίσης, χρησιμοποιώντας τέτοια ποικιλία, δίνεται η δυνατότητα σε κάποιον χρήστη, που για κάποιο λόγο είναι πιο εξοικειωμένος μόνο με έναν ή με κάποιους από αυτούς, να χρησιμοποιήσει μόνο το μέρος ή τα μέρη του δικτύου που τον βολεύει.

3.2.1 Ethernet shield



Εικόνα 3. 2 Ethernet shield μπροστά και πίσω

Η Πλακέτα Ethernet συνδέει το Arduino με το Internet μέσα σε λίγα λεπτά. Αρκεί να συνδεθεί πάνω στην πλακέτα του Arduino και να συνδεθεί στο δίκτυο με ένα καλώδιο RJ45. Η πλακέτα δουλεύει στα 5V και υποστηρίζει ταχύτητες έως 100 Mb/s. Επικοινωνεί με το Arduino μέσω SPI.

Έχει ενσωματωμένο το W5100 τσιπάκι με εσωτερικό buffer μεγέθους 16K. Το τσιπάκι δίνει μια IP δικτύου για συνδέσεις TCP και UDP. Για να γράψουμε προγράμματα που

χρησιμοποιούν συνδέσεις στο Internet χρησιμοποιούμε τη βιβλιοθήκη ethernet (Ethernet Library).

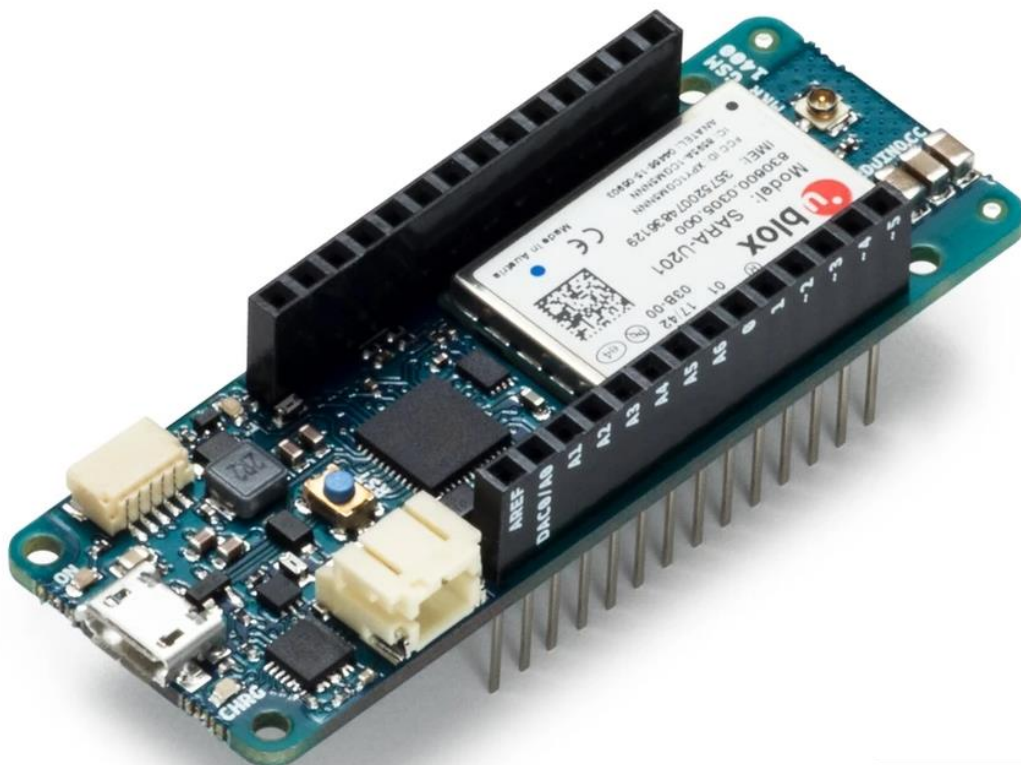
Πάνω στην πλακέτα Ethernet υπάρχει θήκη για κάρτα SD που μπορεί να χρησιμοποιηθεί για αποθήκευση αρχείων. Είναι προσβάσιμη μέσω κώδικα με τη βιβλιοθήκη SD (SD Library).

Το Arduino επικοινωνεί με την πλακέτα μέσω SPI με το ICSP header. Για το Uno χρησιμοποιεί τα Pins 11 (MOSI), 12 (MISO), 13 (SCK) και για το Mega τα pins 51 (MOSI), 50 (MISO), 52 (SCK).

Για να δουλέψουμε με τη βιβλιοθήκη Ethernet, πρέπει το Slave-Select pin να είναι ενεργό. Αν χρησιμοποιούμε και τις δυο βιβλιοθήκες πρέπει κάθε φορά να ενεργοποιούμε το αντίστοιχο slave-select(ss) pin . Για τη βιβλιοθήκη Ethernet το SS είναι το pin 10. Για τη βιβλιοθήκη SD είναι το Pin 4.

3.2.2 GSM MKR 1400

Για την υλοποίηση με GSM χρησιμοποιήθηκε ο μικροελεγκτής Arduino MKR GSM 1400, αντί για ένα Arduino uno ή mega με ένα GSM module (π.χ. το sim900L).



Εικόνα 3. 3 Arduino GSM MKR 1400

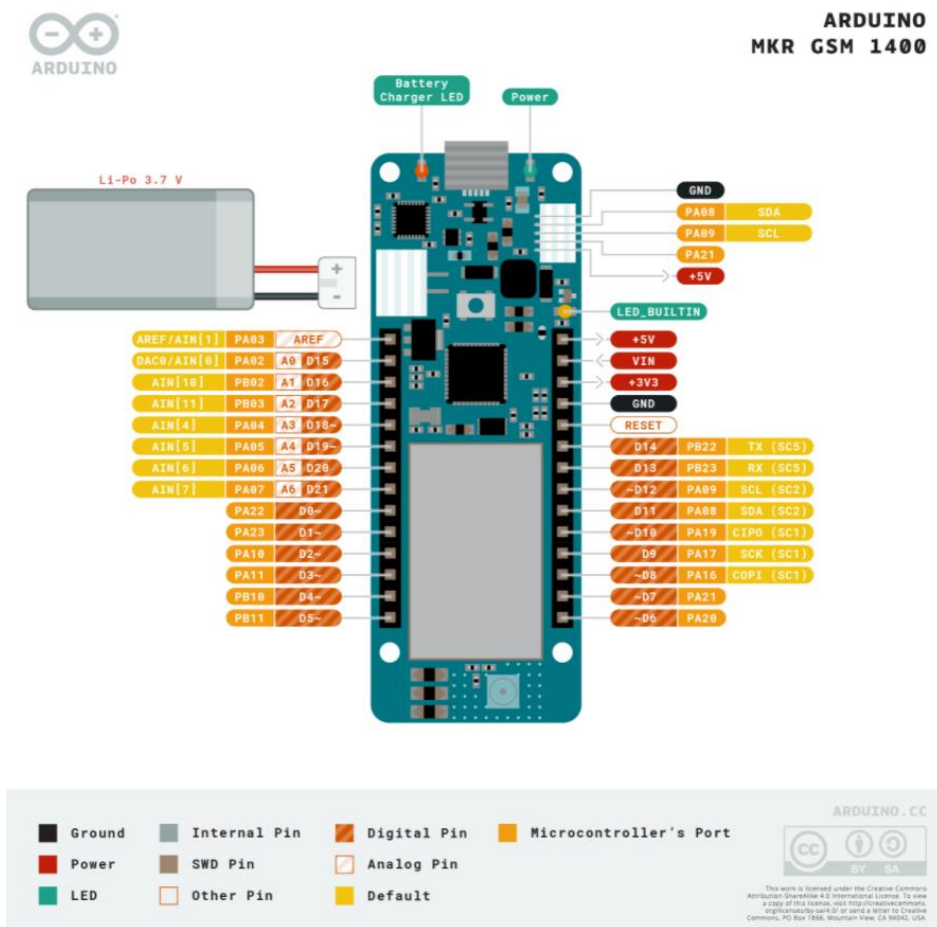
Το Arduino MKR GSM 1400 χρησιμοποιεί το δίκτυο κινητής ως μέσο επικοινωνίας. Το GSM/3G δίκτυο είναι αυτό που καλύπτει κατά μεγαλύτερο ποσοστό την επιφάνεια του πλανήτη, κάνοντας το, το πιο ελκυστικό μέσο επικοινωνίας, καθώς μπορεί να χρησιμοποιηθεί εκεί που όλα τα υπόλοιπα δεν μπορούν.

Ο κύριος επεξεργαστής της πλακέτας είναι ο χαμηλής κατανάλωσης Arm® Cortex®-M0 32-bit SAMD21. Η GSM/3g σύνδεση πραγματοποιείται με το Module SARA-U201 ένα τσιπ χαμηλής κατανάλωσης που δρα σε διαφορετικές μπάντες του εύρους του κινητού δικτύου (GSM 850 MHz, E-GSM 1900 MHz, DCS 1800 MHz, PCS 1900 MHz). Η επικοινωνία είναι ασφαλής και κρυπτογραφείται με το Microchip® ECC508 crypto chip. Για να δουλέψει χρειαζόμαστε επίσης μια πηγή τάσης, μια κεραία και μια κάρτα sim. Τα τεχνικά του χαρακτηριστικά είναι τα εξής :

MICROCONTROLLER	SAMD21 Cortex®-M0+ 32bit low power ARM MCU
RADIO MODULE	u-blox SARA-U201
SECURE ELEMENT	ATECC508
BOARD POWER SUPPLY (USB/VIN)	5V
SUPPORTED BATTERY	Li-Po Single Cell, 3.7V, 2500mAh Minimum
CIRCUIT OPERATING VOLTAGE	3.3V
DIGITAL I/O PINS	8
PWM PINS	13 (0 .. 8, 10, 12, 18 / A3, 19 / A4)
UART	1
SPI	1
I2C	1
ANALOG INPUT PINS	7 (ADC 8/10/12 bit)
ANALOG OUTPUT PINS	1 (DAC 10 bit)
EXTERNAL INTERRUPTS	8 (0, 1, 4, 5, 6, 7, 8, 16 / A1, 17 / A2)
DC CURRENT PER I/O PIN	7 mA
FLASH MEMORY	256 KB (internal)
SRAM	32 KB
EEPROM	no

CLOCK SPEED	32.768 kHz (RTC), 48 MHz
LED_BUILTIN	6
FULL-SPEED USB DEVICE AND EMBEDDED HOST	
ANTENNA GAIN	2dB (bundled antenna at the Arduino Store)
CARRIER FREQUENCY	GSM 850 MHz, E-GSM 1900 MHz, DCS 1800 MHz, PCS
WORKING REGION	Global
SIM CARD	MicroSIM (not included with the board)
LENGTH	67.64 mm
WIDTH	25 mm
WEIGHT	32 gr.

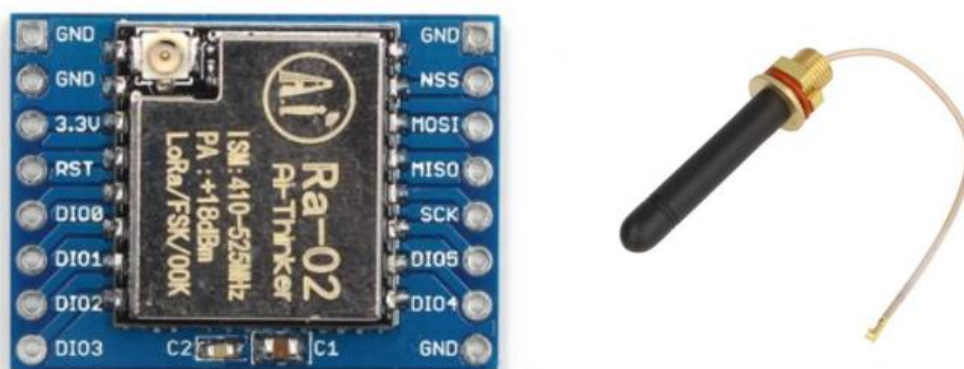
Διάγραμμα:



Εικόνα 3. 4 Arduino MKR GSM 1400 schematics

3.2.3 LoRa-02 SX1278

Για την επικοινωνία με RF χρησιμοποιήθηκαν 3 RF link kits . Το κάθε Kit περιέχει από ένα LoRa Module - SX1278 και μια κεραία. Τα δύο τοποθετήθηκαν στις πλακέτες Mega και το τρίτο στο raspberry pi -server. Τα LoRa των Mega όπως θα δούμε αναλυτικότερα παρακάτω προγραμματίστηκαν ως πομποί και το LoRa του Pi ως δέκτης.



Εικόνα 3. 5 LoRa-02 SX1278 & Antenna for LoRa-02 SX1278 module

Το LoRa-02 είναι ένα Module για ασύρματη επικοινωνία βασισμένο στο SEMTECH's SX1278. Έχει εύρος επικοινωνίας έως και 10 χιλιόμετρα χωρίς εμπόδια. Έχει πολύ ισχυρή ικανότητα anti-jamming και χρησιμοποιεί την συνάρτηση wake up για την κατανάλωση. Χρησιμοποιείται για μεγάλου εύρους-ανοιχτού φάσματος επικοινωνία με ελάχιστη κατανάλωση ρεύματος. Έχει υψηλή ευαισθησία στα -148dBm με Power output από + 20dBm. Σε σχέση με άλλα πιο παραδοσιακά modules έχει προφανή πλεονεκτήματα στο αντί-μπλοκάρισμα και στις επιλογές, λύνοντας το πρόβλημα των παραδοσιακών designs που δεν παίρνουν υπόψιν τους την απόσταση, τις παρεμβολές και την κατανάλωση ταυτόχρονα αλλά μεμονωμένα κάποια από αυτά τα χαρακτηριστικά.

Χαρακτηριστικά:

- Μέγιστη απόσταση: 15KM
- Ευαισθησία: down to -148dBm
- Programmable bit rates: up to 300kbps
- RSSI dynamic range: 127dB
- Wireless frequency: 433MHz
- Working voltage: 1.8 - 3.7v
- Working temperature: -40 - 80°C
- Dimensions: 28x20.3x5mm

Δυνατότητες:

- LoRa[™] Spread Spectrum modulation technology
- Constant RF power output at + 20dBm-100mW voltage change
- Half-duplex SPI communication
- Supports FSK, GFSK, MSK, GMSK, LoRa[™] and OOK modulation modes
- Automatic RF signal detection, CAD mode and very high speed AFC
- Packet engine with CRC up to 256 bytes
- Small footprint dual-row stamp-hole patch package
- Shielded housing
- Spring Antenna

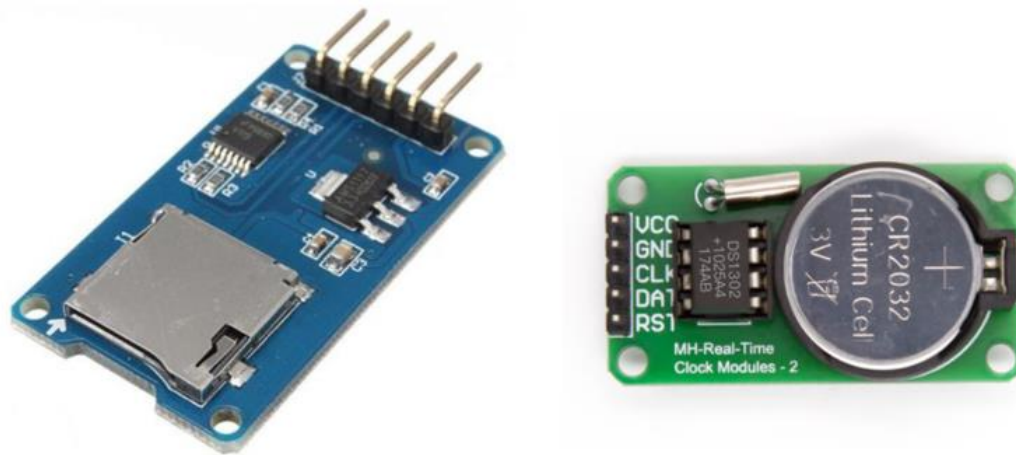
3.3 Βλάβες των Modules και άλλα εμπόδια

Τι γίνεται αν κάποιο ή κάποια από τα Modules που επιλέχθηκαν για την επικοινωνία των σταθμών με τον server για κάποιο λόγο σταματήσουν να λειτουργούν; Τι θα γίνει αν για κάποιο λόγο, ενώ λειτουργούν πέσει το Wi-Fi ή το κινητό δίκτυο της περιοχής στην οποία βρίσκονται;

Αν τα δεδομένα απλά μετρούνται και στέλνονται από το σταθμό στο server τότε θα τα χάνουμε για ένα χρονικό διάστημα ίσο με το χρόνο αποκατάστασης της βλάβης. Οι απώλειες για μικρά χρονικά διαστήματα, θεωρητικά(ανάλογα την εφαρμογή και τις ανάγκες του χρήστη) μπορεί και να μην είναι σημαντικές, όμως από ένα χρονικό σημείο και μετά, είναι απλά απαγορευτικές για τις περισσότερες εφαρμογές, καθώς χάνεται ένας ολόενas και μεγαλύτερος όγκος μετεωρολογικών δεδομένων.

Την μερική λύση σε αυτό το πρόβλημα, έρχονται να την δώσουν δύο άλλα modules. Το SD card module σε συνδυασμό με το real time clock module. Συνδέοντας 3 SD card modules σε κάθε σταθμό(η πλακέτα ethernet έχει ενσωματωμένη θήκη για SD οπότε δεν χρειάζεται) μαζί με μία κάρτα SD (σύνολο 4), υπάρχει πλέον ο χώρος τοπικά για την αποθήκευση των δεδομένων. Ακόμα και αν συμβεί κάποια βλάβη τα δεδομένα θα υπάρχουν τοπικά στο σταθμό, ώστε αν χρειάζεται κάποιος χρήστης πρόσβαση σε αυτά, το μόνο που έχει να κάνει, είναι να πάρει την SD και να την συνδέσει με κάποιον αντάπτορα σε έναν υπολογιστή. Στην εργασία επιλέξαμε να γίνεται αποθήκευση όλων των μετρήσεων στις κάρτες SD (όχι μόνο αυτών που δεν φτάνουν στον server), καθώς μπορεί να χρειαστούν δεδομένα από μια σχετικά μεγάλη χρονική περίοδο πίσω, τα οποία μπορεί είτε να μην υπάρχουν πλέον στην βάση δεδομένων είτε να είναι πιο δύσκολη η πρόσβαση σε αυτά (π.χ. άμα είναι corrupted). Αυτό έχει ως αποτέλεσμα οι SD κάρτες να γεμίζουν τον αποθηκευτικό τους χώρο πιο γρήγορα και άρα να θέλουν πιο συχνά αλλαγή. Ωστόσο αυτή η λύση είναι πιο αποδοτική όσον αφορά το back up των δεδομένων αν η βάση έχει βλάβες η διαγραφούν τιμές και όχι ιδιαίτερα πιο δαπανηρή, καθώς 4 κάρτες των 8GB θα θέλουν αλλαγή κάθε 15 χρόνια, από την στιγμή που μια μέτρηση πιάνει περίπου 80 byte, οι σταθμοί παίρνουν και αποθηκεύουν μέτρηση κάθε ένα λεπτό και υποθέτοντας ότι οι σταθμοί θα δουλεύουν 24 ώρες το 24ωρο συνεχόμενα κάθε μέρα. Δηλαδή 20 ευρώ ανά 15 χρόνια με τις τρέχουσες τιμές για κάρτες των 8GB (που κατά πάσα πιθανότητα θα έχουν μειωθεί αισθητά στο βάθος της δεκαπενταετίας).

Το rtc module στην ουσία είναι ένα ρολόι το οποίο είναι απαραίτητο για την σωστή αποθήκευση των δεδομένων, ώστε να μπορούμε να ξεχωρίσουμε πότε μετρήθηκαν οι τιμές των αισθητηρίων. Έτσι μπορούμε να βρούμε τις μετρήσεις που θέλουμε αν γνωρίζουμε τις ώρες και την χρονική διάρκεια της βλάβης. Τα Arduino , χωρίς να είναι συνδεδεμένα σε κάποιο υπολογιστή ή σε κάποιο δίκτυο, δεν μπορούν να μετρήσουν την ώρα και να κρατήσουν την ώρα από μόνα τους χωρίς κάποιο rtc module. Στο κάθε ρολόι πρέπει επίσης να συνδεθεί μια μπαταρία λιθίου τύπου CR2032, ώστε να μπορεί να κρατάει την ώρα ακόμα και όταν το Arduino «κοιμάται», δηλαδή δεν καταναλώνει ρεύμα.



Εικόνα 3. 6 SD card module & RTC module with CR2032 lithium

3.4 Συχνότητα Μετρήσεων

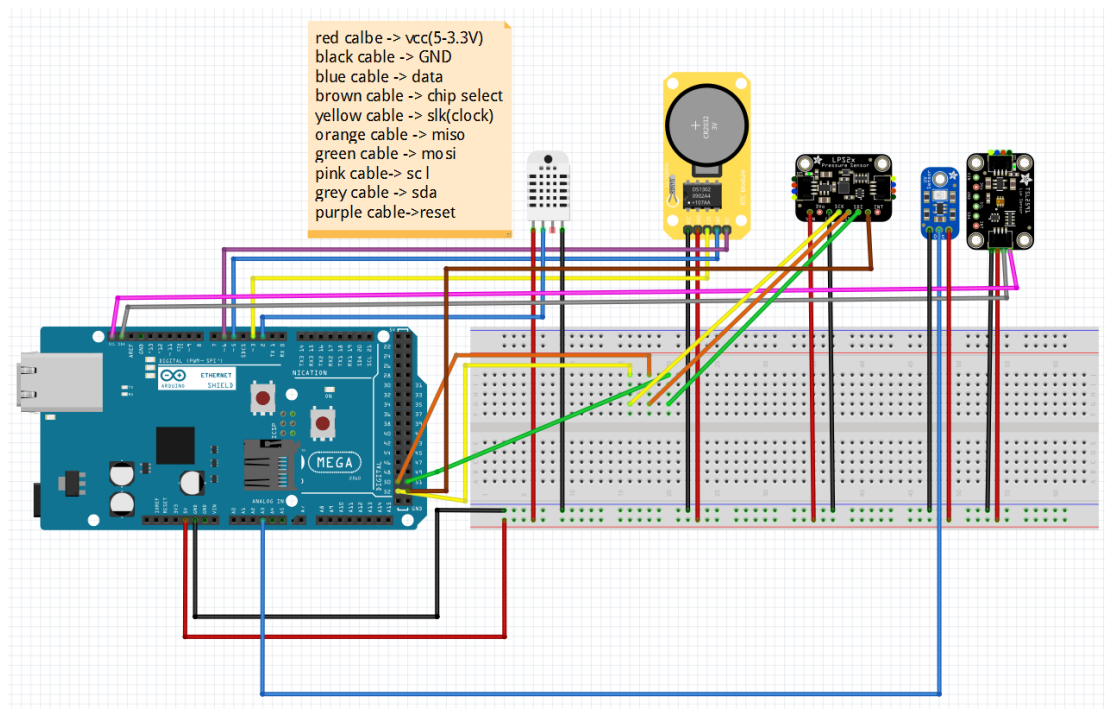
Η συχνότητα που οι μετρήσεις στέλνονται στον εξυπηρετητή συνδέεται άμεσα με μια ενδεχόμενη βλάβη των Module επικοινωνίας των σταθμών. Δεν πρέπει να ξεχνάμε ότι τα Modules (και των επικοινωνιών και τα υπόλοιπα), όπως και τα αισθητήρια, είναι Hardware και ότι το Hardware πάντα έχει ημερομηνία λήξης, η οποία εξαρτάται κυρίως από την ποιότητα του αλλά και από την συχνότητα της χρήσης του. Έτσι έχοντας υπόψη την ποιότητα του Hardware που χρησιμοποιήθηκε, το ενδεχόμενο βλάβης λόγω συχνής χρήσης, την μέγιστη και ελάχιστη συχνότητα λειτουργίας των Modules και των αισθητηρίων αλλά και την απαραίτητη συχνότητα μετρήσεων που χρειάζεται κάθε μετεωρολογικό μέγεθος για βρίσκεται αξιόπιστα η συμπεριφορά του καταλήξαμε τα δεδομένα να στέλνονται κάθε ένα λεπτό στον εξυπηρετητή.

4 ΚΕΦΑΛΑΙΟ ΤΕΤΑΡΤΟ

4.1 Υλοποίηση hardware μετεωρολογικών σταθμών & server

Για την υλοποίηση των σταθμών αρχικά θα πρέπει να ανοίξει το αρχείο .fzz , του κάθε σταθμού στο πρόγραμμα Fritzing, σε έναν υπολογιστή. Με οδηγό τα σχέδια, θα γίνουν οι συνδέσεις κάθε πλακέτας με άλλες πλακέτες(shields), με τα αισθητήρια και με όλα τα υπόλοιπα απαραίτητα modules. Για να μην μπερδευτεί κανείς με τον αριθμό των καλωδίων και των χρωμάτων, υπάρχει σε κάθε αρχείο, οδηγός για κάθε χρώμα καλωδίου. Ακόμη αρκεί να πατήσει κάποιος πάνω σε κάποιο καλώδιο για να δει ποια pins συνδέει.

4.1.1 Σταθμός HMMY



Εικόνα 4. 1 Αρχείο .fzz του σταθμού HMMY

- Αρχικά συνδέουμε το Arduino Mega με το ethernet shield

Η σύνδεση είναι αρκετά εύκολη, αρκεί να προσέξουμε την αντιστοιχία των pins ώστε το καθένα να είναι στην σωστή θέση καθώς και να κουμπώσει το ISCP header. Η επικοινωνία του Mega με την πλακέτα ethernet γίνεται με SPI μέσω του ISCP.

- Συνδέουμε ένα ένα τα αισθητήρια όπως φαίνεται στο σχέδιο της εικόνας 4.1

DHT 22:

- Το Pin + με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin – με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin Out με ένα καλώδιο στο digital Pin 2 του Mega

LPS22HB:

- Το Pin Vin με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin SCK με ένα καλώδιο στη θέση του digital Pin 52 στο Breadboard (clock)
- Το Pin SDO με ένα καλώδιο στη θέση του digital Pin 50 στο Breadboard (MISO)
- Το Pin SDI με ένα καλώδιο στη θέση του digital Pin 51 στο Breadboard (MOSI)
- Το Pin CS με ένα καλώδιο στο digital Pin 53 στο (Chip select) του Mega

Η επικοινωνία του LPS22HB με το Mega γίνεται μέσω SPI (τα Pins που χρησιμοποιεί φαίνονται στις παρενθέσεις).

TSL2591X:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin SDA με ένα καλώδιο στη θέση του Pin SDA1 του Mega
- Το Pin SCL με ένα καλώδιο στη θέση του Pin SCL1 του Mega

Η επικοινωνία του TSL2591X με το Mega γίνεται μέσω I2C(τα Pins είναι τα SDA1 και SCL1).

UV Detection:

- Το Pin VCC με ένα καλώδιο στην τροφοδοσία των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στη γείωση στο Breadboard
- Το Pin OUT με ένα καλώδιο στο analog Pin A3 του Mega

- Συνδέουμε τα υπόλοιπα Modules στο Mega όπως φαίνεται στο σχήμα 4.1 και πιο αναλυτικά:

RTC module:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard

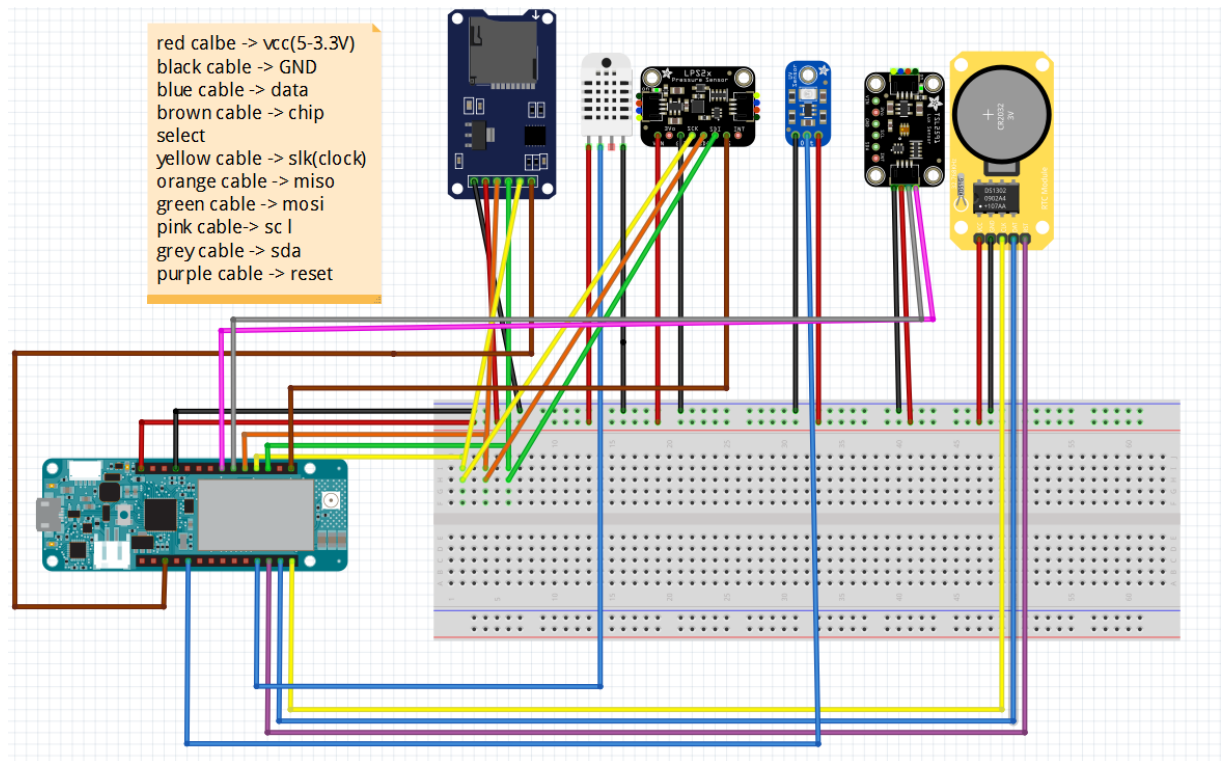
- Το Pin CLK με ένα καλώδιο στο digital Pin 3 του Mega
- Το Pin DAT με ένα καλώδιο στο digital Pin 5 του Mega
- Το Pin RST με ένα καλώδιο στο digital Pin 6 του Mega

Ο σταθμός που θα τοποθετηθεί στο κτήριο Β' του κτηριακού συγκροτήματος του τμήματος ΗΜΜΥ είναι έτοιμος από άποψη hardware.



Εικόνα 4. 2 Σταθμός ΗΜΜΥ

4.1.2 Σταθμός ΠΡΟΚΑΤ



Εικόνα 4. 3 Αρχείο .fzz του σταθμού ΠΡΟΚΑΤ

- Αρχικά συνδέουμε την κεραία στην υποδοχή της στο Arduino MKR GSM 1400
- Συνδέουμε τα αισθητήρια όπως φαίνεται στο παραπάνω σχέδιο της εικόνας 4.3

DHT 22:

- Το Pin + με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin – με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin Out με ένα καλώδιο στο digital Pin 2 του MKR

LPS22HB:

- Το Pin Vin με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin SCK με ένα καλώδιο στη θέση του digital Pin 9 στο Breadboard (clock)
- Το Pin SDO με ένα καλώδιο στη θέση του digital Pin 10 στο Breadboard (MISO)

- Το Pin SDI με ένα καλώδιο στη θέση του digital Pin 8 στο Breadboard (MOSI)
- Το Pin CS με ένα καλώδιο στο digital Pin 6 στο (Chip select) του MKR

Η επικοινωνία του LPS22HB με το MKR γίνεται μέσω SPI (τα Pins φαίνονται που χρησιμοποιεί φαίνονται στις παρενθέσεις).

TSL2591X:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin SDA με ένα καλώδιο στο digital Pin 11 του MKR
- Το Pin SCL με ένα καλώδιο στο digital Pin 12 του MKR

Η επικοινωνία του TSL2591X με το MKR γίνεται μέσω I2C(τα Pins pin 11 και το Pin 12).

UV Detection:

- Το Pin VCC με ένα καλώδιο στην τροφοδοσία των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin OUT με ένα καλώδιο στο analog Pin A3 του MKR

- Συνδέουμε τα υπόλοιπα Modules στο MKR όπως φαίνεται στο σχήμα 4.3 και πιο αναλυτικά:

RTC module:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin CLK με ένα καλώδιο στο digital Pin 5 του MKR
- Το Pin DAT με ένα καλώδιο στο digital Pin 4 του MKR
- Το Pin RST με ένα καλώδιο στο digital Pin 3 του MKR

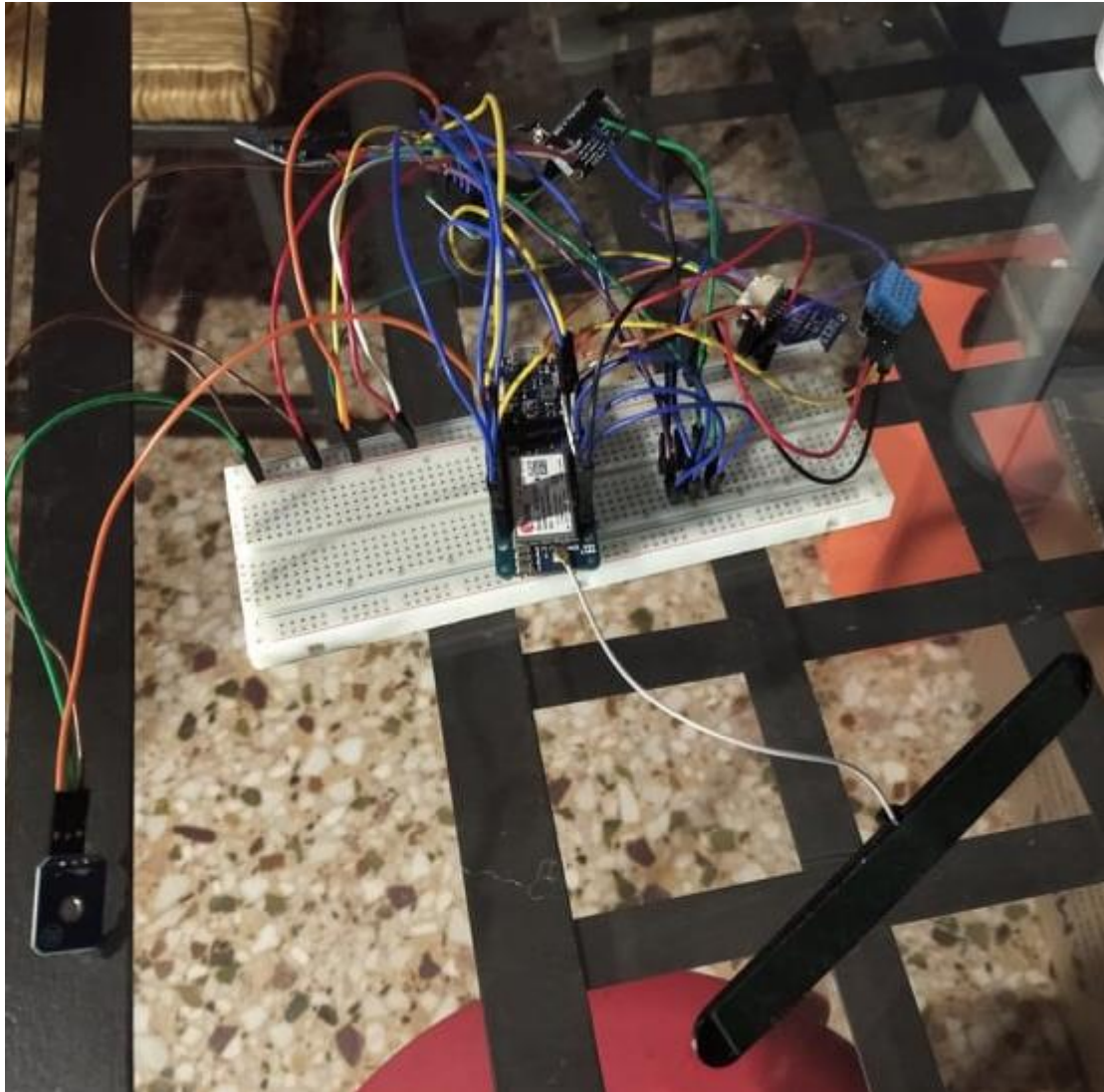
SD card Module:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin SCK με ένα καλώδιο στη θέση του digital Pin 9 στο Breadboard (clock)
- Το Pin MISO με ένα καλώδιο στη θέση του digital Pin 10 (MISO)στο Breadboard
- Το Pin MOSI με ένα καλώδιο στη θέση του digital Pin 8 (MOSI) στο Breadboard

- Το Pin CS με ένα καλώδιο στο digital Pin 16(είναι το ίδιο με το analog Pin A1) στο MKR (Chip select)

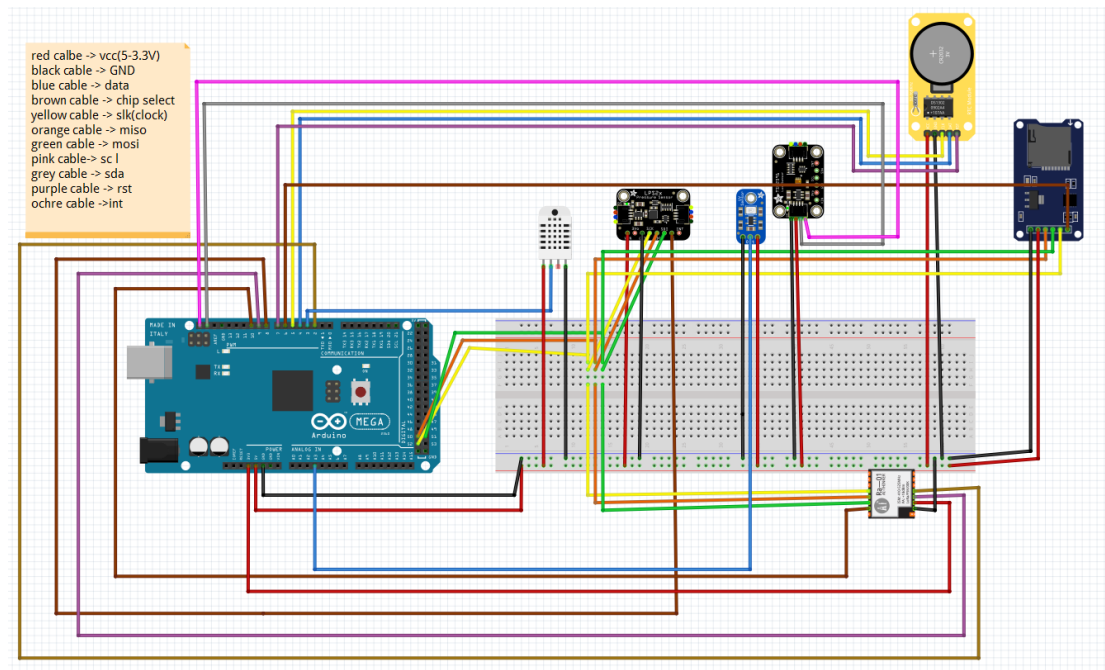
Η επικοινωνία του SD card module με το Mega γίνεται μέσω SPI (τα Pins που χρησιμοποιεί φαίνονται στις παρενθέσεις).

Ο σταθμός που θα τοποθετηθεί στο κτηριακό συγκρότημα των ΠΡΟΚΑΤ, και ειδικά στο κτήριο των εργαστηρίων του τμήματος ΗΜΜΥ είναι έτοιμος από άποψη hardware.



Εικόνα 4. 4 Σταθμός ΠΡΟΚΑΤ

4.1.3 Σταθμοί Ε.Κ ΑΘΗΝΑ & ΕΣΤΙΩΝ



Εικόνα 4. 5 Αρχείο .fzz των σταθμών Εστιών και Ε.Κ ΑΘΗΝΑ

- Όλα τα βήματα πρέπει να γίνουν δύο φορές καθώς οι δύο σταθμοί δεν έχουν διαφορά στο hardware τους
- Συνδέουμε ένα ένα τα αισθητήρια όπως φαίνεται στο παραπάνω σχέδιο της εικόνας 4.5

DHT 22:

- Το Pin + με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin – με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin Out με ένα καλώδιο στο digital Pin 3 του Mega

LPS22HB:

- Το Pin Vin με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin SCK με ένα καλώδιο στη θέση του digital Pin 52 στο Breadboard (clock)
- Το Pin SDO με ένα καλώδιο στη θέση του digital Pin 50 στο Breadboard (MISO)
- Το Pin SDI με ένα καλώδιο στη θέση του digital Pin 51 στο Breadboard (MOSI)
- Το Pin CS με ένα καλώδιο στο digital Pin 8 στο (Chip select) του Mega

Η επικοινωνία του LPS22HB με το Mega γίνεται μέσω SPI (τα Pins που χρησιμοποιεί φαίνονται στις παρενθέσεις).

TSL2591X:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin SDA με ένα καλώδιο στη θέση SDA1 του Mega
- Το Pin SCL με ένα καλώδιο στη θέση SCL1 του Mega

Η επικοινωνία του TSL2591X με το Mega γίνεται μέσω I2C(τα Pins είναι τα SDA1 και SCL1).

UV Detection:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin OUT με ένα καλώδιο στο analog Pin A3 του Mega

- Συνδέουμε τα υπόλοιπα Modules στο Mega όπως φαίνεται στο σχήμα 4.5 και πιο αναλυτικά:

RTC module:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard
- Το Pin CLK με ένα καλώδιο στο digital Pin 5 του Mega
- Το Pin DAT με ένα καλώδιο στο digital Pin 4 του Mega
- Το Pin RST με ένα καλώδιο στο digital Pin 7 του Mega

SD card Module:

- Το Pin VCC με ένα καλώδιο στο Pin της τροφοδοσίας των 5 volt στο Breadboard
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard

- Το Pin SCK με ένα καλώδιο στην θέση του digital Pin 52 στο Breadboard (clock)
- Το Pin MISO με ένα καλώδιο στην θέση του digital Pin 50 στο Breadboard (MISO)
- Το Pin MOSI με ένα καλώδιο στην θέση του digital Pin 51 στο Breadboard (MOSI)
- Το Pin CS με ένα καλώδιο στο digital Pin 6 (Chip select) του Mega

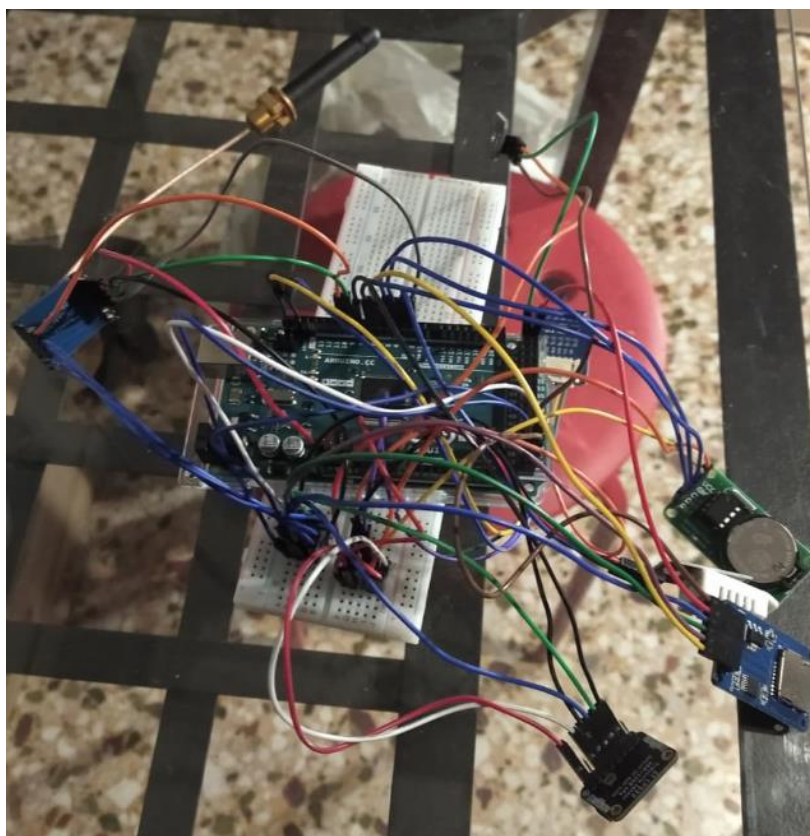
Η επικοινωνία του SD card module με το Mega γίνεται μέσω SPI (τα Pins που χρησιμοποιεί φαίνονται στις παρενθέσεις) .

LoRa-02 SX1278 module:

- Συνδέουμε την κεραία στην υποδοχή της στο Module
- Το Pin 3.3V με ένα καλώδιο στο Pin της τροφοδοσίας των 3.3 volt του Mega
- Το Pin GND με ένα καλώδιο στο Pin της γείωσης στο Breadboard του Mega
- Το Pin SCK με ένα καλώδιο στην θέση του digital Pin 52 στο Breadboard (clock)
- Το Pin MISO με ένα καλώδιο στην θέση του digital Pin 50 στο Breadboard
- Το Pin MOSI με ένα καλώδιο στην θέση του digital Pin 51 στο Breadboard
- Το Pin NSS(CS) με ένα καλώδιο στο digital Pin 10 (Chip select) του Mega
- Το Pin DIO0(INT) με ένα καλώδιο στο digital Pin 2 του Mega
- Το Pin RST με ένα καλώδιο στο digital Pin 9 του Mega

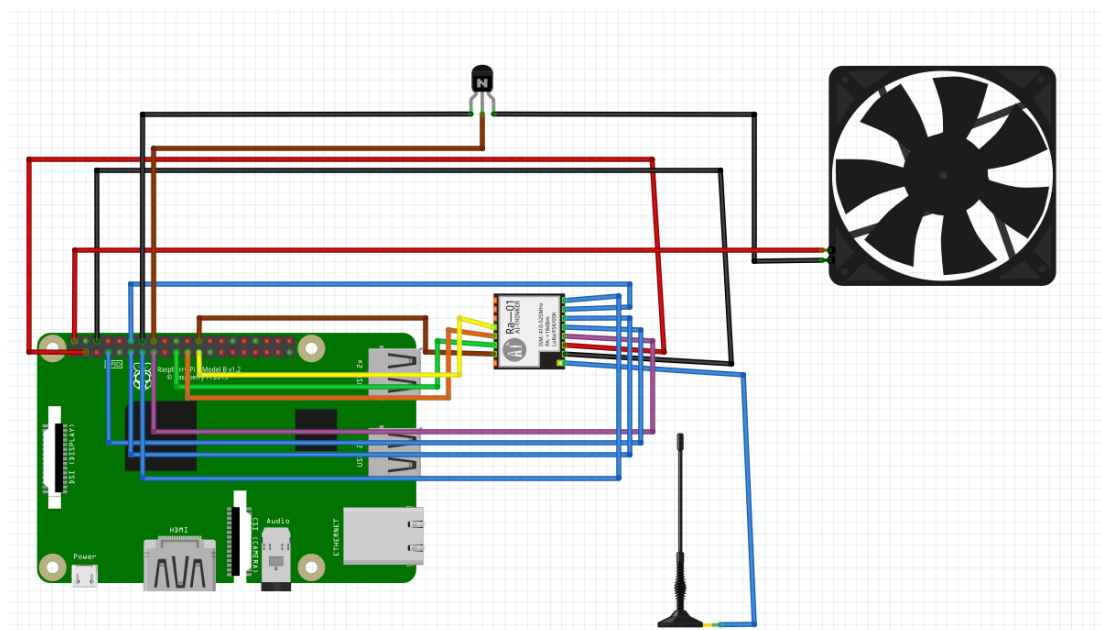
Η επικοινωνία του LoRa-02 SX1278 module με το Mega γίνεται μέσω SPI.

Οι σταθμοί που θα τοποθετηθούν στο κτήριο των εστιών και στο κτήριο του Ε.Κ ΑΘΗΝΑ είναι έτοιμοι από άποψη hardware.



Εικόνα 4. 6 Σταθμοί ΑΘΗΝΑ, ΕΣΤΙΩΝ

4.1.4 Server



Εικόνα 4. 7 Αρχείο .fzz του server

- Συνδέουμε τα Modules στο Raspberry pi όπως φαίνεται στο σχήμα της εικόνας 4.7

LoRa-02 SX1278 module:

- Συνδέουμε την κεραία στην υποδοχή της στο LoRa-02 SX1278 Module
- Το Pin 3.3V με ένα καλώδιο στην τροφοδοσία των 3.3 volt του Pi
- Το Pin GND με ένα καλώδιο σε μία από τις γειώσεις του Pi
- Το Pin MOSI με ένα καλώδιο στο GPIO 10 του Pi
- Το Pin MISO με ένα καλώδιο στο GPIO 9 του Pi
- Το Pin SCK με ένα καλώδιο στο GPIO 11 του Pi
- Το Pin NSS(CS) με ένα καλώδιο στο GPIO 8 του Pi
- Το Pin DIO0 με ένα καλώδιο στο GPIO 4 του Pi
- Το Pin DIO1 με ένα καλώδιο στο GPIO 17 του Pi
- Το Pin DIO2 με ένα καλώδιο στο GPIO 18 του Pi
- Το Pin DIO3 με ένα καλώδιο στο GPIO 27 του Pi
- Το Pin RST με ένα καλώδιο στο GPIO 22 του Pi

Fan module*:

- Συνδέουμε τη γείωση του ανεμιστήρα στον collector του transistor
- Συνδέουμε την τροφοδοσία του ανεμιστήρα σε μία από τις θέσεις των 5 volt του Pi
- Συνδέουμε μια από τις γειώσεις του Pi στον emitter του transistor
- Συνδέουμε τη βάση του transistor στο GPIO 23 του Pi

Ο server που θα τοποθετηθεί στο κτήριο Β των ηλεκτρολόγων είναι έτοιμος από άποψη hardware

* Ο ανεμιστήρας κρίθηκε απαραίτητος να τοποθετηθεί πάνω από τον επεξεργαστή του Raspberry pi, καθώς όσο τρέχει το πρόγραμμα που υποδέχεται τα δεδομένα από την κεραία και τα προγράμματα του server, η θερμοκρασία του επεξεργαστή ανεβαίνει σε μη επιθυμητά επίπεδα.



Εικόνα 4. 8 O server

4.2 Υλοποίηση software server

Σε αυτή την ενότητα θα αναλυθεί πρώτα το software που υλοποιεί τον εξυπηρετητή, καθώς εκεί στέλνονται τα δεδομένα των μετρήσεων που παίρνουν οι σταθμοί. Στην συνέχεια, θα αναλυθεί το software που υλοποιεί τους σταθμούς για να ολοκληρωθεί το σύστημα.

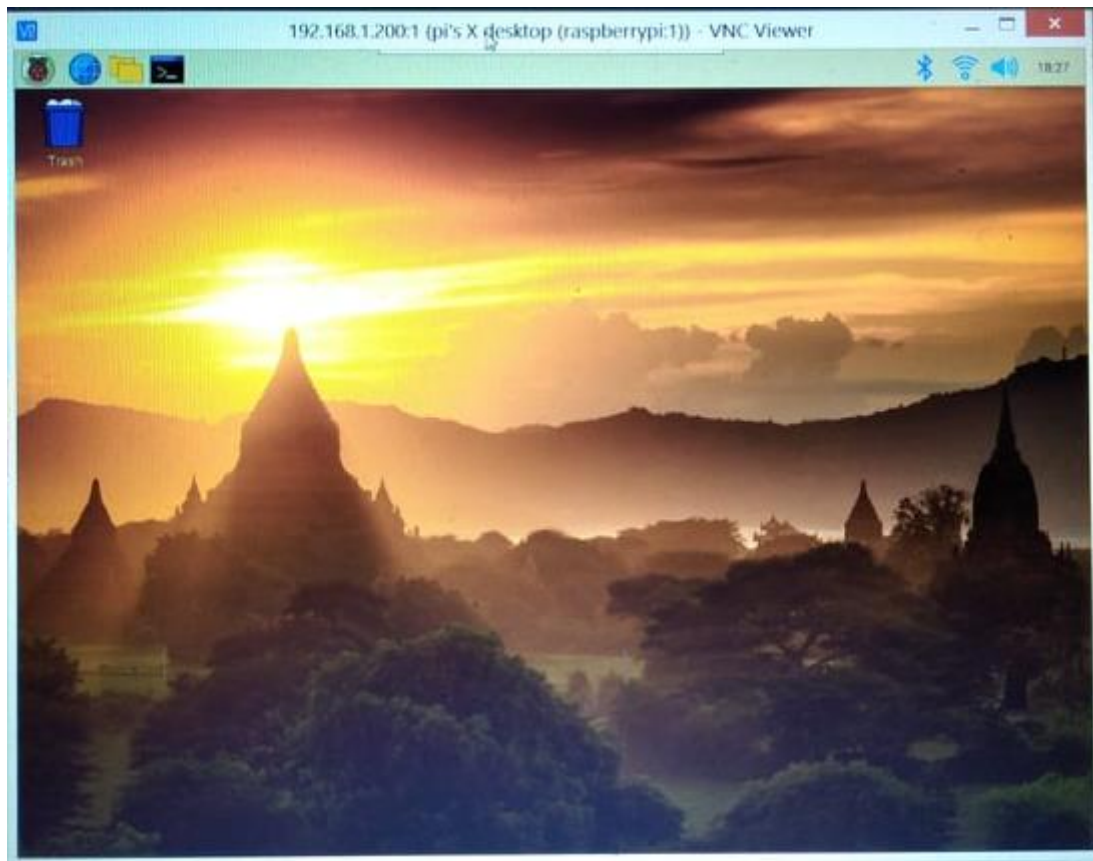
4.2.1 Το λειτουργικό σύστημα

Λειτουργικό σύστημα (Operating System ή OS) ονομάζεται στην επιστήμη της πληροφορικής το λογισμικό του υπολογιστή που είναι υπεύθυνο για τη διαχείριση και το συντονισμό των εργασιών, καθώς και την κατανομή των διαθέσιμων πόρων. Το λειτουργικό σύστημα έχει ως αποστολή τη διαχείριση της μνήμης, του επεξεργαστή, των μονάδων αποθήκευσης, της κάρτας γραφικών και των λοιπών στοιχείων που απαρτίζουν έναν υπολογιστή και την παροχή τελικά, προς τα προγράμματα χρήστη, ενός καλύτερου και πιο σαφούς μοντέλου του υπολογιστή.

Στον εξυπηρετητή(server) της εργασίας, εγκαταστάθηκε το λειτουργικό σύστημα Raspbian ,το οποίο είναι μια παραλλαγή του Debian (έκδοση Linux), τροποποιημένο κατάλληλα για το υλικό (hardware) του Raspberry Pi. Προσφέρεται δωρεάν και διαθέτει πάνω από 35000 πακέτα λογισμικού ομαδοποιημένα σε κατάλληλη μορφή για εύκολη εγκατάσταση στο Raspberry Pi.

Η διαδικασία της εγκατάστασης του Raspbian στο Raspberry Pi είναι η εξής:

- Κατεβάζουμε σε έναν υπολογιστή την τελευταία έκδοση Raspbian (αρχείο img) από το κεντρικό site.
- Συνδέουμε μια SD κάρτα, επαρκούς χωρητικότητας(τουλάχιστον 8Gb), σε έναν adaptor και στη συνέχεια στον υπολογιστή.
- Ανοίγουμε το πρόγραμμα win32 Disk imager και κάνουμε write το αρχείο στην κάρτα.
- Αφού τελειώσουμε τοποθετούμε την κάρτα SD στο Raspberry Pi και συνδέουμε με ένα καλώδιο ethernet το Pi με ένα router.
- Συνδέουμε την τροφοδοσία στο Raspberry Pi το οποίο θα εκκινήσει και θα κάνει αυτόματα τις κατάλληλες ρυθμίσεις για την προετοιμασία του λειτουργικού.
- Κατεβάζουμε το Putty, αν δεν το έχουμε ήδη και το ανοίγουμε.
- Ανοίγουμε έναν οποιοδήποτε πρόγραμμα περιήγησης στο Internet και πληκτρολογούμε την IP του router που έχουμε συνδέσει το Pi(η IP Μπορεί να βρεθεί πάνω στο router).
- Στις ρυθμίσεις του router πάμε στα Lan ports και βρίσκουμε την IP του raspberry pi.
- Βάζουμε την IP που βρήκαμε στο putty και κάνουμε μια SSH σύνδεση.
- Γράφουμε pi για Username και raspberry για κωδικό.
- Έχουμε συνδεθεί επιτυχώς με το Pi.
- Για γραφικό περιβάλλον κατεβάζουμε την εφαρμογή VNC VIEWER στον υπολογιστή.
- Αφού την κατεβάσουμε πληκτρολογούμε στο terminal που έχει ανοίξει το Putty την εντολή: **sudo apt-get install tightvncserver**
- Όταν τελειώσει με την εγκατάσταση θα μας ζητήσει να βάλουμε ένα κωδικό.
- Βάζουμε τον κωδικό(σημαντικό να τον θυμόμαστε μετά) και γράφουμε την εντολή **tightvncserver** και κρατάμε τον αριθμό της οθόνης που θα μας δώσει(desktop number).
- Ανοίγουμε την εφαρμογή VNC VIEWER από τα windows βάζουμε την IP του Pi άνω κάτω τελεία και τον αριθμό της οθόνης(π.χ. 192.168.0.105:1).
- Βάζουμε τον κωδικό που μας ζητήθηκε να δώσουμε πριν πατάμε Ignore and continue και θα μας ανοίξει το γραφικό περιβάλλον του Raspbian στο VNC Viewer, έτσι μπορούμε να χρησιμοποιήσουμε το Pi με την οθόνη το πληκτρολόγιο και το ποντίκι του υπολογιστή μας χωρίς να χρειαστούμε κάτι έξτρα.



Εικόνα 4. 9 Το γραφικό περιβάλλον του Raspbian

4.2.2 Apache web server

Ο εξυπηρετητής ή διακομιστής, στην πιο απλή του μορφή είναι ένας ηλεκτρονικός υπολογιστής που τρέχει κατάλληλο λογισμικό ώστε να εξυπηρετεί τους χρήστες που συνδέονται με αυτόν για κάποιο σκοπό. Ο ρόλος ενός Web Server είναι βασικά να δέχεται τα αιτήματα των χρηστών (clients) και να στέλνει απαντήσεις στα αιτήματα αυτά. Ένας Web Server παίρνει μια διεύθυνση URL από το χρήστη μέσω ενός προγράμματος περιήγησης (Opera, Firefox, Chrome, Internet Explorer κ.α.) και στην πιο γενική περίπτωση:

- Είτε τη μεταφράζει σε ένα όνομα αρχείου και στέλνει το αρχείο που ζητήθηκε πίσω στο χρήστη μέσω του διαδικτύου
- Είτε τη μεταφράζει σε ένα όνομα προγράμματος, το οποίο εκτελεί, και στη συνέχεια στέλνει το αποτέλεσμα του εν λόγω προγράμματος πίσω, μέσω του διαδικτύου, στον αιτούντα.

Αν για οποιοδήποτε λόγο ο Web Server δεν είναι σε θέση να απαντήσει στην αίτηση, επιστρέφει ένα μήνυμα σφάλματος

Για την εγκατάσταση του server της εργασίας επιλέχθηκε το πρόγραμμα Apache web server. Το Apache Web Server είναι ένα λογισμικό που τρέχει στο παρασκήνιο κάτω από ένα λειτουργικό σύστημα το οποίο υποστηρίζει multitasking και παρέχει υπηρεσίες σε άλλες εφαρμογές όπως είναι τα προγράμματα περιήγησης (web browsers) που αναφέρθηκαν παραπάνω. Φροντίζει για την επικοινωνία στο δίκτυο που εξυπηρετεί χρησιμοποιώντας το πρωτόκολλο TCP/IP, το οποίο επιτρέπει στις συσκευές, με διευθύνσεις IP, να επικοινωνούν μεταξύ τους εντός του ίδιου δικτύου.

Επιλέχθηκε για τους εξής λόγους:

- Είναι ελεύθερο λογισμικό (open source): ο πηγαίος κώδικας και η βιβλιογραφία του είναι διαθέσιμα σε οποιονδήποτε, γεγονός που καθιστά εύκολη την κατανόηση της λειτουργίας του.
- Υπερκαλύπτει όλες τις ανάγκες του συστήματός μας: Το Apache μπορεί το ίδιο εύκολα να χρησιμοποιηθεί για μία ιστοσελίδα που διαθέτει μία-δύο σελίδες περιεχομένου όσο και να χρησιμοποιηθεί για ιστοσελίδες με τεράστιο αριθμό σελίδων και περιεχομένου, εξυπηρετώντας ταυτόχρονα εκατομμύρια χρήστες κάθε μήνα.
- Διατίθεται δωρεάν για όλους

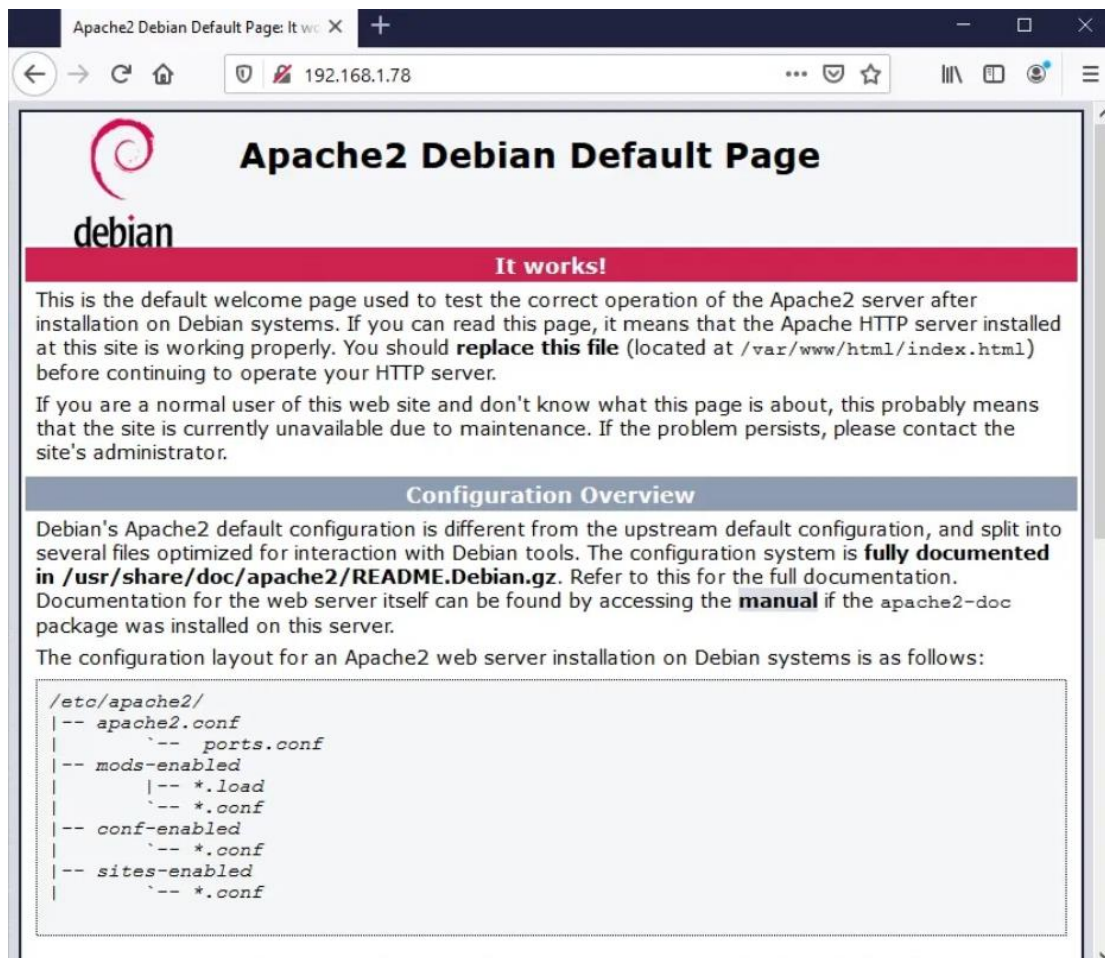
Πριν ξεκινήσουμε την εγκατάσταση του Apache web server πρέπει να τρέξουμε δύο εντολές:

- `sudo apt update -y`
- `sudo apt upgrade -y`

Αφού εγκατασταθούν τα απαραίτητα πακέτα το σύστημα είναι έτοιμο για την εγκατάσταση του Apache. Χρειάζεται μόνο μια εντολή :

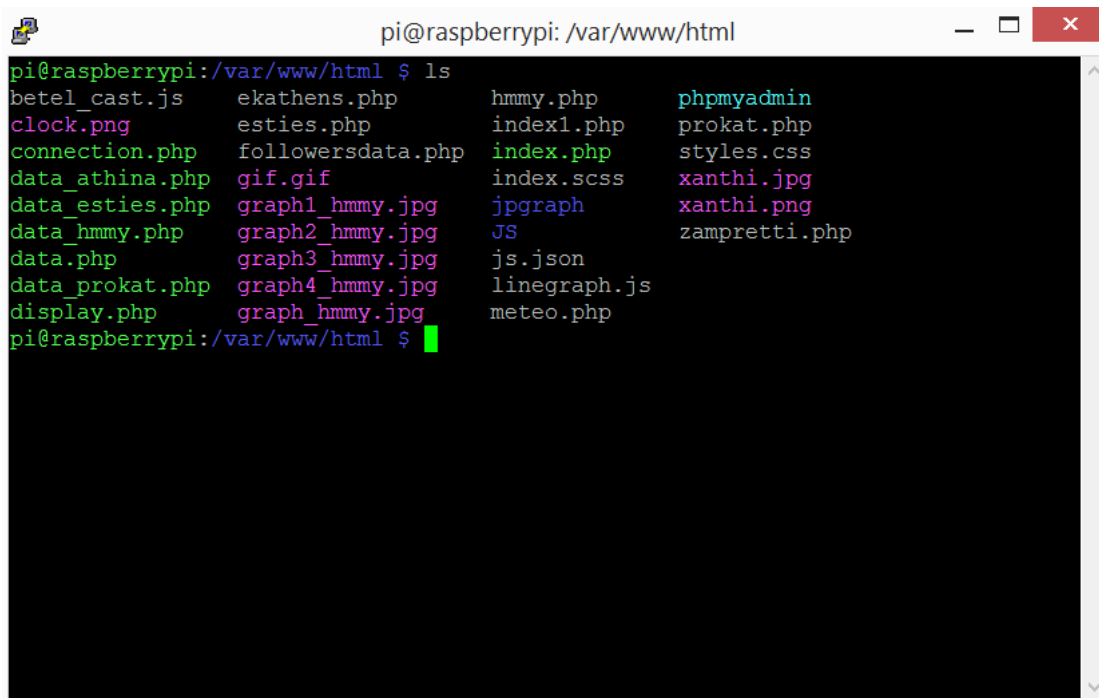
- `sudo apt install apache2 -y`

Για να σιγουρευτούμε πως εγκαταστάθηκε σωστά ανοίγουμε τον browser του Pi και πληκτρολογούμε την Ιp του Pi. Αν όλα έχουν πάει καλά θα πρέπει να εμφανιστεί η αρχική σελίδα του Apache



Εικόνα 4. 10 Η αρχική σελίδα του Apache

Ο Server που εγκαταστάθηκε στο σύστημα, περιέχει όλα τα αρχεία κώδικα, τις εικόνες και τα λοιπά στοιχεία που συντελούν στη δημιουργία του site, στον φάκελο που βρίσκεται στη διαδρομή: /var/www/html του Raspbian όπως φαίνεται και παρακάτω



```
pi@raspberrypi: /var/www/html
pi@raspberrypi:/var/www/html $ ls
betel_cast.js      ekathens.php      hmmy.php           phpmysadmin
clock.png          esties.php         index1.php         prokat.php
connection.php     followersdata.php index.php           styles.css
data_athina.php    gif.gif           index.scss         xanthi.jpg
data_esties.php    graph1_hmmy.jpg   jpggraph          xanthi.png
data_hmmy.php      graph2_hmmy.jpg   JS                zampretti.php
data.php           graph3_hmmy.jpg   js.json
data_prokat.php    graph4_hmmy.jpg   linegraph.js
display.php        graph_hmmy.jpg    meteo.php
```

Εικόνα 4. 11 Τα αρχεία του site

4.2.3 MySQL & Βάσεις δεδομένων

Μια Βάση δεδομένων είναι μία συλλογή δεδομένων οργανωμένη σε πίνακες που παρέχει ταυτόχρονα ένα μηχανισμό για ανάγνωση, εγγραφή, τροποποίηση ή και πιο πολύπλοκες διαδικασίες πάνω στα δεδομένα. Ο σκοπός μιας βάσης δεδομένων είναι η οργανωμένη αποθήκευση πληροφορίας και η δυνατότητα εξαγωγής της πληροφορίας αυτής, ιδίως σε πιο οργανωμένη μορφή, σύμφωνα με ερωτήματα (queries) που τίθενται στη βάση δεδομένων. Τα δεδομένα είναι δυνατόν να αναδιοργανώνονται με διάφορους τρόπους, σε νοητούς πίνακες, χωρίς να είναι απαραίτητη η αναδιοργάνωση των φυσικών πινάκων που αποθηκεύονται.

Για τις ανάγκες της εργασίας ήταν απαραίτητη μια βάση δεδομένων με τουλάχιστον 4 διαφορετικούς πίνακες, ώστε να αποθηκεύονται τα δεδομένα που στέλνουν οι σταθμοί και στη συνέχεια να παρουσιάζονται στον ιστότοπο.

Το MySQL είναι ένα open source σύστημα διαχείρισης βάσεων δεδομένων βασισμένο στο πρότυπο της SQL, που είναι η δημοφιλέστερη γλώσσα που χρησιμοποιείται για την περιγραφή και τον σχεδιασμό των βάσεων αυτών. Ένας MySQL Server αποτελείται από ένα σύνολο βάσεων δεδομένων (databases). Κάθε βάση δεδομένων είναι ένα σύνολο πινάκων (tables). Όπως είπαμε και πιο πάνω αφού εγκαταστήσουμε το MySQL και το phpMyAdmin, θα δημιουργήσουμε μια βάση δεδομένων για τις ανάγκες της εργασίας.

Για να εγκαταστήσουμε την MySQL αρκεί να τρέξουμε την παρακάτω εντολή:

- `sudo apt install mariadb-server`

Αφού εγκατασταθεί θα πρέπει να θέσουμε ένα κωδικό για τον root χρήστη για λόγους ασφαλείας. Τρέχουμε την εντολή, πατάμε ναι σε ότι μας ρωτήσει (Y) και θέτουμε κωδικό για τον χρήστη root:

- `sudo mysql_secure_installation`

Για να μπούμε στο MySQL server τρέχουμε την εντολή και βάζουμε τον κωδικό που θέσαμε πριν λίγο :

- `sudo mysql -u root -p`

Εάν θέλουμε μπορούμε να φτιάξουμε ένα νέο χρήστη τρέχοντας την παρακάτω εντολή:

- `CREATE USER 'username'@'localhost' IDENTIFIED BY 'password';`

Όπου στο username βάζουμε το όνομα χρήστη που θέλουμε(π.χ. paris) και στο password βάζουμε τον κωδικό που θέλουμε(π.χ. parisparis) .

Στην συνέχεια τρέχουμε την εντολή :

- `grant all privileges on *.* to 'username'@'localhost';`

Για να έχουμε τα δικαιώματα να διαχειριζόμαστε τις βάσεις δεδομένων. Τέλος τρέχουμε την εντολή:

- `flush privileges;`

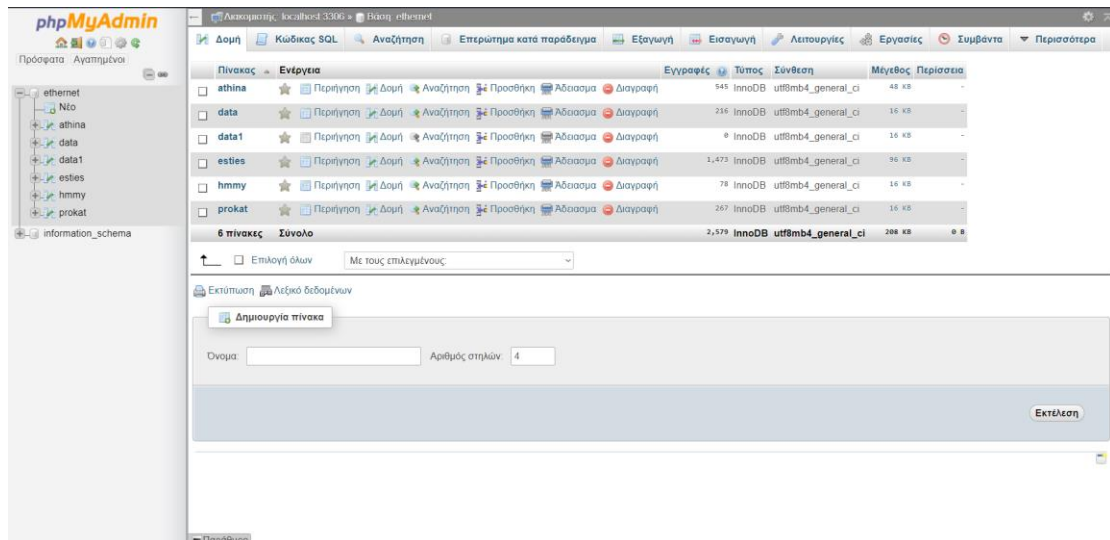
Για να ανανεωθούν τα δικαιώματα.

Κανονικά τώρα μπορούμε να φτιάξουμε την βάση με τους πίνακες που θέλουμε αλλά θα εγκαταστήσουμε πρώτα το PhpMyAdmin για πιο εύκολη χρήση και περιήγηση.

4.2.4 PhpMyAdmin

Το phpMyAdmin είναι ένα open source εργαλείο, αναπτυγμένο σε γλώσσα PHP. Έχει σχεδιαστεί για να διαχειρίζεται το σύστημα MySQL με τη χρήση ενός προγράμματος περιήγησης. Οι λόγοι που θα εγκαταστήσουμε το phpMyAdmin αντί να δουλέψουμε με το terminal είναι οι παρακάτω:

- Διαθέτει γραφικό περιβάλλον φιλικό προς το χρήστη.
- Υποστηρίζει τις περισσότερες λειτουργίες του MySQL όπως ενδεικτικά είναι η δημιουργία, η τροποποίηση ή η διαγραφή βάσεων, πινάκων, πεδίων ή γραμμών



Εικόνα 4. 12 Γραφικό περιβάλλον PhpMyAdmin

Για να το εγκαταστήσουμε αρκεί να τρέξουμε την εντολή:

- `sudo apt install phpmyadmin -y`

Αφού τρέξει η εντολή και εγκατασταθεί μπορούμε να έχουμε πρόσβαση ανοίγοντας ένα πρόγραμμα περιήγησης και πληκτρολογώντας την Ιp του Pi/phpmyadmin(π.χ. <http://192.168.1.200/phpmyadmin>)

Βάζουμε τα στοιχεία του νέου χρήστη που φτιάξαμε νωρίτερα και μπαίνουμε στην σελίδα.

Για να φτιάξουμε τη βάση και τους πίνακες που χρειαζόμαστε πηγαίνουμε στο PhpMyAdmin, στην ενότητα Κώδικας SQL, κάνουμε copy paste τον κώδικα των αρχείων SQL που υπάρχουν στο παράρτημα Β και πατάμε εκτέλεση. Μετά την εκτέλεση η μορφή της κεντρικής σελίδας του PhpMyadmin θα πρέπει να μοιάζει με την εικόνα πιο πάνω. Αφού τελειώσουμε τα παραπάνω βήματα η βάση δεδομένων και οι πίνακες είναι έτοιμοι προς χρήση.

4.2.5 PHP

Η PHP είναι μια server-side scripting γλώσσα που είναι σχεδιασμένη για τον προγραμματισμό τόσο σελίδων και εφαρμογών του διαδικτύου όσο και εφαρμογών γενικής χρήσης (general purpose). Οι βασικοί λόγοι που χρησιμοποιείται η PHP σε σύστημα είναι:

- Μπορεί εύκολα να επικοινωνήσει με τη βάση δεδομένων

- Επικοινωνεί εύκολα με τη σειριακή θύρα για αποστολή και λήψη δεδομένων
- Διαθέτει κατάλληλες συναρτήσεις που βοηθούν στον έλεγχο της ορθότητας των δεδομένων που εισάγονται από το χρήστη πριν αυτά καταχωρηθούν στο σύστημα (validation)
- Χρησιμοποιείται στην κατασκευή δυναμικών ιστοσελίδων ή web εφαρμογών

Για να εγκατασταθεί η PHP στο Raspberry Pi γράφουμε στο terminal τις εντολές:

- `sudo apt install php -y` για την php και
- `sudo apt php-mysql -y` για τις βιβλιοθήκες που επιτρέπουν την πρόσβαση στη βάση MySQL

Μετά και την τελευταία εγκατάσταση θα πρέπει να μεταφέρουμε όλα τα απαραίτητα αρχεία στο `/var/www/html` και θα έχουμε επιτυχώς υλοποιήσει τον server τοπικά στο δίκτυο μας, έχοντας την δυνατότητα πρόσβασης όποτε θέλουμε, όχι όμως από οπουδήποτε. Εξωτερικά από το τοπικό δίκτυο, ο διακομιστής δεν είναι προσβάσιμος. Άρα, αν ο σταθμός με την πλακέτα ethernet δεν είναι συνδεδεμένος στο ίδιο δίκτυο δεν θα μπορεί να στείλει τα δεδομένα. Το δίκτυο κινητής είναι εξωτερικό δίκτυο άρα ο σταθμός με την GSM υλοποίηση δεν θα μπορεί να στείλει τα δεδομένα.

Στην παρούσα κατάσταση μόνο οι σταθμοί με τις κεραίες μπορούν να στείλουν τις μετρήσεις τους στον server, καθώς το Pi τις διαβάζει τοπικά μέσω τις κεραίας του και τις ανεβάζει τοπικά στο δίκτυο.

Συνεπώς, το δίκτυο πρέπει να γίνει προσβάσιμο και από εξωτερικά δίκτυα για την πλήρη λειτουργικότητα και χρηστικότητα του. Για να γίνει αυτό θα χρησιμοποιήσουμε μια τεχνική που λέγεται Port forwarding. Έτσι και ο σταθμός με το gsm θα μπορεί να στείλει τα δεδομένα του και ο σταθμός με την πλακέτα ethernet θα μπορεί να τοποθετηθεί όχι μόνο στο ίδιο τοπικό δίκτυο που θα είναι συνδεδεμένος ο διακομιστής, αλλά οπουδήποτε υπάρχει η δυνατότητα ενσύρματης σύνδεσης στο Internet .

4.2.6 Port Forwarding

Το Port Forwarding είναι η διαδικασία που πρέπει να γίνει, για να είναι ένας υπολογιστής που βρίσκεται πίσω από έναν δρομολογητή (Router), προσβάσιμος από

άλλους υπολογιστές στο διαδίκτυο. Η διαδικασία αυτή γίνεται σε όλους τους δρομολογητές με τον ίδιο τρόπο σε 3 απλά βήματα:

1. Ρυθμίζουμε την εφαρμογή που θέλουμε, να «ακούει» σε συγκεκριμένη πόρτα (στο παράδειγμά μας είναι η 8081).
2. Ρυθμίζουμε το Router να επιτρέπει την κίνηση στη συγκεκριμένη πόρτα και να την δρομολογεί στην συσκευή που θέλουμε, δηλώνοντας external και internal port (στην περίπτωση απλών δικτύων και εφαρμογών που εξυπηρετούμε, αυτές συμπίπτουν πάντα. Στο παράδειγμά μας, θα είναι internal port: 8081, external port: 8081 για την ιδιωτική / εσωτερική IP 192.168.1.200 του διακομιστή-RPi, στο οποίο θέλουμε να δρομολογείται η κίνηση.
3. Κάνουμε reboot το δρομολογητή και τη συσκευή (Raspberry pi).

Τα παραπάνω βήματα είναι η γενική μεθοδολογία του Port forwarding. Μετά από ένα σημείο τα βήματα αλλάζουν ανάλογα τον πάροχο ίντερνετ κάθε χρήστη.

4.2.6.1 Static Ip

Πριν όμως ξεκινήσει η διαδικασία του Port Forwarding πρέπει να δώσουμε μια static Ip στο Pi. Αυτό χρειάζεται, γιατί κάθε συσκευή που συνδέεται στον router, είτε ασύρματα είτε ενσύρματα, παίρνει αυτόματα μια δυναμική Ip. Αυτό σημαίνει ότι η Ip θα αλλάζει ανά κάποιο χρονικό διάστημα (συνήθως μερικών ημερών). Κάτι τέτοιο, πέρα από εμπόδιο στο Port Forwarding αποτελεί και εμπόδιο στα προγράμματα των Arduino καθώς θα έπρεπε να αλλάζουμε και εκεί τον κώδικα, κάθε φορά που ο router θα αλλάζει την Ip.

Για να δώσουμε μια static Ip στο Pi ανοίγουμε το terminal και δίνουμε την εντολή:

- `sudo nano /etc/dhcpd.conf`

Αυτή η εντολή θα ανοίξει ένα αρχείο κειμένου με τις ρυθμίσεις δικτύου του Pi. Εκεί θα προσθέσουμε/αλλάξουμε τις ρυθμίσεις του δικτύου όπου χρειάζεται.. Στο τέλος του αρχείου προσθέτουμε τα εξής:

```
interface eth0
metric 300
static ip_address=192.168.0.200/24
static routers=192.168.1.1
static domain_name_servers=192.168.1.1
```

```
interface wlan0
metric 200
static ip_address=192.168.0.201/24
```

```
static routers=192.168.1.1
static domain_name_servers=192.168.1.1
```

Προσοχή η Ip που θα δώσουμε, πρέπει να είναι στο εύρος των διαθέσιμων Ip που μπορεί να δώσει ο router και ταυτόχρονα όχι κοντά στο εύρος των δυναμικών Ip που δίνει το router στις υπόλοιπες συσκευές, ώστε το router να μην μπερδέψει το Pi με κάποια άλλη συσκευή. Αποθηκεύουμε και κλείνουμε το αρχείο. Για να δούμε εάν έγιναν σωστά οι αλλαγές θα πρέπει αρχικά να κάνουμε ένα reboot το raspberry Pi. Στην συνέχεια αφού ανοίξει πρέπει να τρέξουμε την εντολή :

- ifconfig

Αν έχουν πάει όλα καλά θα πρέπει να βλέπουμε την καινούρια static Ip.

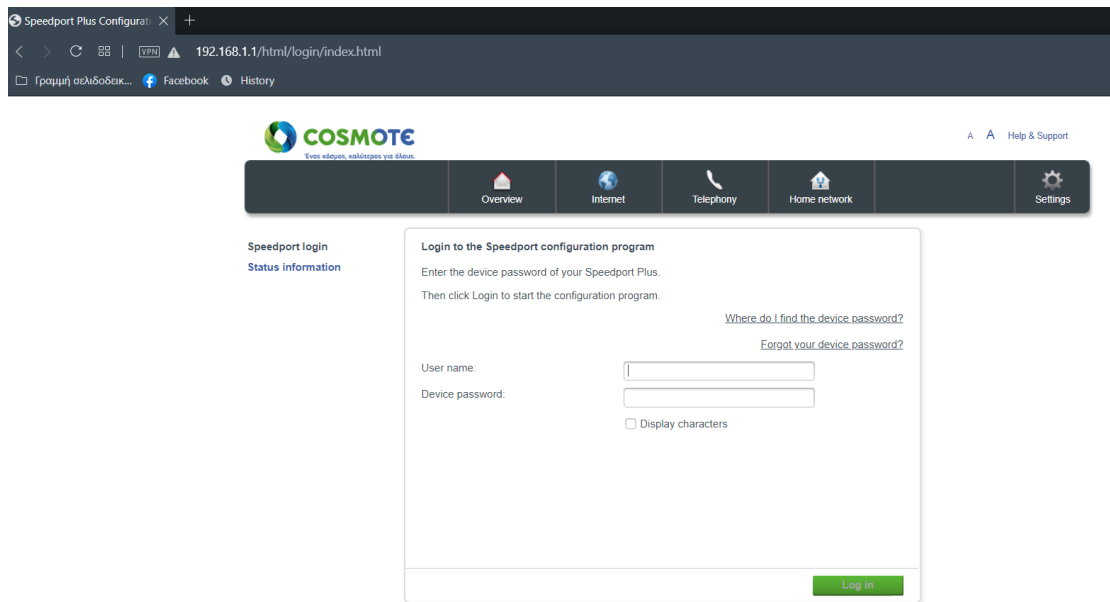
```
pi@raspberrypi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.201 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::ee4d:e276:a172:9bc1 prefixlen 64 scopeid 0x20<link>
    inet6 2a02:587:2010:39ca:7e22:de28:4403:3f33 prefixlen 64 scopeid 0x0<
global>
    ether b8:27:eb:35:3a:fa txqueuelen 1000 (Ethernet)
    RX packets 1960 bytes 610320 (596.0 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1547 bytes 167364 (163.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2160 bytes 242531 (236.8 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2160 bytes 242531 (236.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.200 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 2a02:587:2010:39ca:3416:9f3a:cddf:bfd9 prefixlen 64 scopeid 0x0<
global>
    inet6 fe80::976d:78b1:dfd8:220f prefixlen 64 scopeid 0x20<link>
    ether b8:27:eb:60:6f:af txqueuelen 1000 (Ethernet)
    RX packets 13497 bytes 1060128 (1.0 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 13489 bytes 5609063 (5.3 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Εικόνα 4. 13 Αποτέλεσμα εντολής ifconfig

Πλέον είμαστε έτοιμοι να ξεκινήσουμε το Port Forward. Ανοίγουμε στον υπολογιστή μας ένα δρομολογητή και πληκτρολογούμε την Ip του router. Στην προκειμένη περίπτωση χρησιμοποιήθηκε ο router της Cosmote που είναι η σπιτική σύνδεση μου.



Εικόνα 4. 14 Αρχική σελίδα router

Βάζουμε τα στοιχεία που αναγράφονται πάνω το router και συνδεόμαστε. Στην συνέχεια πάμε στην ενότητα internet->Features-> Port Forwarding. Εδώ θα καταχωρήσουμε τα στοιχεία που χρειάζεται για το Port Forward.

- Κάνουμε κλικ στο enable
- Βάζουμε ένα οποιοδήποτε όνομα
- Protocol διαλέγουμε Both
- WAN Connection διαλέγουμε Internet VDSL
- WAN Start Port 8081
- WAN End Port 8081
- LAN Host Ip Address 192.168.1.200(Pi static Ip)
- LAN Host Start Port 8081
- LAN Host End Port 8081
- Κάνουμε Save

Enable	<input checked="" type="checkbox"/>
Name	<input type="text" value="httpweb"/>
Protocol	<input type="text" value="Both"/>
WAN Host Start IP Address	<input type="text"/> . <input type="text"/> . <input type="text"/> . <input type="text"/>
WAN Host End IP Address	<input type="text"/> . <input type="text"/> . <input type="text"/> . <input type="text"/>
WAN Connection	<input type="text" value="Internet_VDSL"/>
WAN Start Port	<input type="text" value="8081"/> (1-65535)
WAN End Port	<input type="text" value="8081"/> (1-65535)
Enable MAC Mapping	<input type="checkbox"/>
LAN Host IP Address	<input type="text" value="192"/> . <input type="text" value="168"/> . <input type="text" value="1"/> . <input type="text" value="200"/>
LAN Host Start Port	<input type="text" value="8081"/> (1-65535)
LAN Host End Port	<input type="text" value="8081"/> (1-65535)

Εικόνα 4. 15 Ρυθμίσεις Port Forward

Αφού τελειώσουμε όμως θα χρειαστούμε και μία dynamic DNS υπηρεσία.

4.2.6.2 DDNS

DDNS, dynamic DNS, ή πιο συγκεκριμένα dynamic domain name system, είναι μια υπηρεσία η οποία διατηρεί και «μεταφράζει» τα domain names σε δυναμικές διευθύνσεις IP. Μέσω μιας τέτοιας υπηρεσίας, μπορεί κάποιος να έχει πρόσβαση στον υπολογιστή ή στα αρχεία του που βρίσκονται στο σπίτι του, από οπουδήποτε στον κόσμο.

Η υπηρεσία DDNS εξυπηρετεί έναν παρόμοιο σκοπό με την υπηρεσία DNS. Η DDNS επιτρέπει σε όποιον έχει στην κατοχή του ftp servers, ή άλλου είδους υπηρεσίες, να έχει πρόσβαση σε αυτές τις υπηρεσίες «καλώντας» ένα συγκεκριμένο όνομα (domain name) αντί για την διεύθυνση IP, η οποία αλλάζει ανά τακτά χρονικά διαστήματα.

Στην ουσία μέσω του DDNS θα πάρουμε ένα στατικό url η αλλιώς domain name το οποίο θα ακούει στην IPv4 διεύθυνση δικτύου του υπολογιστή μας. Κάθε φορά που θα γίνεται μια αλλαγή στην Ip η υπηρεσία θα βάζει το στατικό url αυτόματα να κοιτάει στην καινούρια διεύθυνση χωρίς να χρειάζεται εμείς να κάνουμε κάτι.

Μια τέτοια υπηρεσία προσφέρει δωρεάν το site noip.com. Αφού δημιουργήσουμε και επαληθεύσουμε τον λογαριασμό μας δωρεάν, πάμε στην ενότητα MyAccount-> Dynamic DNS-> Hostnames και πατάμε create hostname. Στις ρυθμίσεις:

- Διαλέγουμε ένα όνομα για το Url μας (weatherdatapms για την εργασία)
- Διαλέγουμε ένα από τα ελεύθερα domains, όποιο θέλουμε (ddns.net για την εργασία)
- Βάζουμε την IPv4 διεύθυνση(Μπορούμε να την βρούμε με ένα απλό google search)
- Πατάμε create hostname και είμαστε έτοιμοι

Το μόνο που έχουμε να θυμόμαστε είναι κάθε 30 μέρες να μπαίνουμε στο λογαριασμό μας και να επιβεβαιώνουμε το hostname για να μένει ενεργό. Όταν περάσουν οι 30 μέρες θα έρθει Mail ειδοποίηση στο λογαριασμό που έχουμε βάλει και θα έχουμε 7 μέρες να κάνουμε verify το hostname. Μετά τις 7 μέρες αν δεν έχουμε κάνει verify το hostname θα έχει αφαιρεθεί από το DNS (δεν θα δουλεύει το link) ωστόσο θα μπορούμε ακόμα να κάνουμε confirm το Hostname για 7 ακόμα μέρες. Στο πέρας των 7 ημερών το Hostname θα μπει σε «redemption phase» και μετά από 14 ημέρες θα είναι ελεύθερο να το χρησιμοποιήσει ένας άλλος χρήστης.

Το τελευταίο που έχουμε να κάνουμε τώρα είναι να καταχωρήσουμε την υπηρεσία στο router. Οπότε πάμε πάλι στην σελίδα του router κάνουμε Log in , πάμε στην ενότητα internet->Features-> DDNS και κάνουμε τα εξής:

- Click στην επιλογή Enable dynamic DNS
- Ανοίγουμε τα access data
- Provider βάζουμε το noip.com
- Domain name το url.domain (weatherdatapms.ddns.net για την εργασία)
- Username και Password τα στοιχεία μας για log-in στο noip.com

Τώρα αρκεί να πληκτρολογήσουμε σε έναν περιηγητή, από οποιοδήποτε δίκτυο, το url.domain:port/index.php όπου url το url που επιλέξαμε προ ολίγου, domain αυτό που επιλέξαμε προ ολίγου, port την θύρα που διαλέξαμε στο Port forward του router και Index.php οποιοδήποτε αρχείο έχουμε βάλει στον κατάλογο /var/www/html του διακομιστή μας.

Για παράδειγμα το <http://weatherdatapms.ddns.net:8081/index.php> θα μας πάει στην αρχική σελίδα του website που δημιουργήθηκε για την εργασία.

4.3 Software μετεωρολογικών σταθμών

Αρχικά πριν εξετάσουμε τις ιδιαιτερότητες κάθε σταθμού, κάποιες ρυθμίσεις θα πρέπει να γίνουν για τα modules όλων των σταθμών.

4.3.1 RTC Module

Για το ρολόι κάθε σταθμού αρχικά θα πρέπει να τρέξουμε το πρόγραμμα που θα αρχικοποιήσει την ώρα και θα την αποθηκεύσει στην μπαταρία. Το πρόγραμμα λέγεται DS1302_Simple(δες παράρτημα Α) και μπορούμε να το ανοίξουμε είτε τοπικά, είτε μέσω των παραδειγμάτων της βιβλιοθήκης RTC by mankuna αφού την κατεβάσουμε. Θα χρειαστεί να το τρέξουμε για κάθε ρολόι από μια φορά. Στην ουσία αυτό που κάνει το πρόγραμμα είναι να βάζει το κάθε ρολόι, να ξεκινήσει να μετράει σύμφωνα με την ώρα που θα πάρει τοπικά από τον υπολογιστή.

4.3.2 SD Card Module

Εδώ δεν χρειάζεται να κάνουμε αλλαγές στο Module ή στον κώδικα αλλά να προετοιμάσουμε τις ίδιες τις SD κάρτες. Συνδέουμε μία κάρτα κάθε φορά στον adaptor και μετά στον υπολογιστή. Ανοίγουμε την κάρτα στον υπολογιστή και δημιουργούμε κάθε φορά ένα απλό αρχείο κειμένου με τα εξής ονόματα:

1. LOGETHERNET (στην πρώτη SD)
2. LOGCN (στην δεύτερη SD)
3. LOGRF1 (στην τρίτη SD)
4. LOGRF2 (στην τέταρτη SD)

Μόλις δημιουργήσουμε το κάθε αρχείο το ανοίγουμε και στην πρώτη σειρά γράφουμε τα εξής:

```
DATE | TIME | TEMPERATURE | HUMIDITY | LUX | PRESSURE | LTEMPERATURE  
| UV
```

Κάνουμε αποθήκευση και κλείνουμε. Οι κάρτες τώρα είναι έτοιμες να μπούνε στο κάθε module τους.

4.3.3 Σταθμός HMMY

Για να μπορέσει ο σταθμός να στείλει τα δεδομένα στο διακομιστή αρχικά πρέπει να συνδέσουμε την πλακέτα ethernet με ένα καλώδιο ethernet στο router. Στην συνέχεια συνδέουμε το Arduino Mega στον υπολογιστή πηγαίνουμε στην ενότητα Files->Examples->Ethernet->DHCPAdressPrinter και κατεβάζουμε το πρόγραμμα στο Mega. Αφού ξεκινήσει να τρέχει ανοίγουμε την σειριακή οθόνη που μας δείχνει την Ip της πλακέτας ethernet.



Εικόνα 4. 16 Παράδειγμα αποτελέσματος εκτέλεσης

Στην συνέχεια βάζουμε την Ip που κρατήσαμε, στο πρόγραμμα ethernet_shield_working_with_rpi (παράρτημα Α), στην μεταβλητή byte ip καθώς και το όνομα του διακομιστή μας που φτιάξαμε νωρίτερα στην μεταβλητή char serv.

```
ThreeWire myWire(5,3,6); // IO, SCLK, CE
RtcDS1302<ThreeWire> Rtc(myWire);
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
byte ip[] = {192, 168, 137, 13 }; //Enter the IP of ethernet shield 192.168.1.5
char serv[] = "weatherdatapms.ddns.net"; // the ip address for pi
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
```

Εικόνα 4. 17 Byte ip &char serv

Ένα ακόμη σημείο που πρέπει να προσέξουμε στο πρόγραμμα του σταθμού HMMY είναι η θύρα που θα του ζητήσουμε να συνδεθεί, το όνομα του αρχείου που θα καλέσουμε καθώς και το όνομα του αρχείου της SD που θα αποθηκεύσουμε τα δεδομένα.

```
if (client.connect(serv, 8081)) { //Connecting at the IP address and port we saved before
  Serial.println("connected");
  client.print("GET /data_hmmy.php?"); //Connecting and Sending ues to database
```

Εικόνα 4. 18 Port & file

Η θύρα πρέπει να είναι η 8081 που διαλέξαμε για το Port Forward, το όνομα του αρχείου είναι το data_hmmy.php που όταν κληθεί θα βάλει τα δεδομένα στο σωστό πίνακα της βάσης και τέλος το όνομα του αρχείου που θα αποθηκεύσουμε τα δεδομένα στην sd είναι logethernet.txt

Τώρα μπορούμε να κατεβάσουμε το πρόγραμμα ethernet_shield_working_with_rpi κανονικά στο hardware που έχουμε στήσει για τον σταθμό HMMY.

4.3.4 Σταθμός ΠΡΟΚΑΤ

Για να μπορέσει ο σταθμός να στείλει τα δεδομένα στο διακομιστή αρχικά πρέπει να συνδέσουμε την κάρτα SIM στην ειδική υποδοχή κάτω από την πλακέτα. Στον κώδικα για την SIM πρέπει να θυμόμαστε το Pin που έχουμε βάλει(αν έχουμε βάλει αλλιώς το αφήνουμε κενό) καθώς και να βρούμε το APN του παρόχου της SIM.

```
#define SECRET_PINNUMBER      ""
#define SECRET_GPRS_APN      "internet" // replace your GPRS APN
#define SECRET_GPRS_LOGIN    ""         // replace with your GPRS login
#define SECRET_GPRS_PASSWORD ""         // replace with your GPRS password
```

Εικόνα 4. 19 GPRS APN

Ένα ακόμη σημείο που πρέπει να προσέξουμε στο πρόγραμμα του σταθμού ΠΡΟΚΑΤ, όπως και στον σταθμό HMMY είναι η θύρα που θα του ζητήσουμε να συνδεθεί, το όνομα του αρχείου που θα καλέσουμε καθώς και το όνομα του αρχείου της SD που θα αποθηκεύσουμε τα δεδομένα.

```
char server[] = "weatherdatapms.ddns.net";
char path[] = "/data_prokat.php?";
int port = 8081; // port 80 is the default for HTTP
```

Εικόνα 4. 20 Char path & char server &port

Η θύρα πρέπει να είναι η 8081 που διαλέξαμε για το Port Forward, το όνομα του αρχείου είναι το data_prokat.php που όταν καλεστεί θα βάλει τα δεδομένα στο σωστό πίνακα της βάσης και τέλος το όνομα του αρχείου που θα αποθηκεύσουμε τα δεδομένα στην SD είναι logCN.txt

Τώρα μπορούμε να κατεβάσουμε το πρόγραμμα mkrghsm1400working κανονικά στο hardware που έχουμε στήσει για τον σταθμό ΠΡΟΚΑΤ.

4.3.5 Σταθμοί Ε.Κ ΑΘΗΝΑ & ΕΣΤΙΩΝ

Σε αυτούς τους δύο σταθμούς πρέπει να δώσουμε ιδιαίτερη **προσοχή** καθώς θα κατεβάσουμε το `rfwithrtcdatalogger` (Παράρτημα Α) κάθε φορά με δύο αλλαγές μόνο:

1. Το όνομα της μεταβλητής `ide`. Για τον ένα σταθμό είναι 1.1 και για τον άλλο 1.2
2. Το όνομα του αρχείου που θα αποθηκεύσουμε τα δεδομένα στην SD. Για τον πρώτο είναι `logrf1.txt` και για τον δεύτερο `logrf2.txt`

```
String ide = "1.2", String ide = "1.1";  
dataLog = SD.open("LogRF1.txt", FILE_WRITE);  
dataLog = SD.open("LogRF2.txt", FILE_WRITE);
```

Εικόνα 4. 21 String ide & logRF

4.4 Πακέτο δεδομένων

Ένα κομμάτι που αξίζει να τονιστεί σε αυτούς τους δύο σταθμούς είναι το πως οι μετρήσεις στέλνονται στον server. Αρχικά, συγκεντρώνουμε όλες τις μετρήσεις και τις βάζουμε σε ένα πίνακα χαρακτήρων, δημιουργώντας έτσι ένα πακέτο δεδομένων, που στέλνει κάθε φορά η κεραία. Η κεραία του δέκτη στην συνέχεια παραλαμβάνει το πακέτο και το σπάει σε κομμάτια γνωρίζοντας εκ των προτέρων την θέση της κάθε μέτρησης. Το πρόβλημα στην αρχή ήταν δύο.

Το πρώτο ήταν πως θα ξεχωρίζει η κεραία του διακομιστή από ποιόν σταθμό παρέλαβε το πακέτο. Το πρόβλημα αυτό λύθηκε βάζοντας την σταθερά `ide` που αναφέρεται πιο πάνω, διαφορετική για κάθε σταθμό στην αρχή του πίνακα του πακέτου.

Το δεύτερο πρόβλημα ήταν ότι οι μετρήσεις δεν είχαν σταθερό μέγεθος, δηλαδή, δεν πιάνανε τον ίδιο αριθμό θέσεων στον πίνακα του πακέτου κάθε φορά. Μια μέτρηση της φωτεινότητας για παράδειγμα, μπορεί την μια χρονική στιγμή να ήταν 20 lux και να έπιανε δύο θέσεις στον πίνακα του πακέτου και στην επόμενη να ήταν 132 lux και να έπιανε τρεις θέσεις στον πίνακα του πακέτου. Έτσι στον διακομιστή, αν είχαμε καθορίσει 2 θέσεις για την φωτεινότητα στην επόμενη μέτρηση αντί για 132 θα είχαμε 13 lux ή αντίστροφα, αν είχαμε καθορίσει 3 θέσεις στην πρώτη μέτρηση θα είχαμε τριψήφιο αριθμό lux με το τρίτο ψηφίο να είναι το πρώτο της επόμενης μέτρησης π.χ. της πίεσης που θα το έχανε. Το πρόβλημα αυτό λύθηκε κρατώντας σε διαφορετικές μεταβλητές τα μεγέθη των μετρήσεων που στέλνουμε κάθε φορά. Βάλαμε τις μεταβλητές αυτές στις πρώτες θέσεις του πίνακα χαρακτήρων αμέσως

μετά την σταθερά `ide`. Έτσι στην κεραία του διακομιστή αρχικά ξεχωρίζουμε το `ide` (άρα από ποιο σταθμό ήρθαν τα δεδομένα) και μετά το μέγεθος των μετρήσεων. Με αυτό τον τρόπο έγιναν δυναμικές οι θέσεις για την κάθε μέτρηση και δεν καταστρεφόταν όλο το πακέτο δεδομένων αν άλλαζε μέγεθος κάποια από αυτές σε επόμενες μέτρησεις.

4.5 Raspberry Pi & προγραμματισμός κεραίας

Για να γράψουμε το πρόγραμμα υποδοχής των πακέτων της κεραίας χρησιμοποιήσαμε τη γλώσσα προγραμματισμού Python.

4.5.1 Python

Η Python θεωρείται ίσως η απλούστερη γλώσσα με την οποία θα μπορούσε κανείς να μαθαίνει προγραμματισμό, αλλά δεν υστερεί καθόλου σε δυνατότητες. Έχει γίνει ιδιαίτερα δημοφιλής στον ακαδημαϊκό και επιστημονικό χώρο για τους εξής λόγους:

- Αν και είναι γλώσσα υψηλού επιπέδου, διαθέτει πολύ απλή σύνταξη. Έτσι, επιτρέπει στον προγραμματιστή να ασχοληθεί με την επίλυση του προβλήματος κι όχι με τις ιδιαιτερότητες της.
- Παρόλο που φτάνει σε πολύ υψηλό επίπεδο δυνατοτήτων (π.χ. ενσωματώνει συναρτησιακό και αντικειμενοστραφή προγραμματισμό), η απλή της σύνταξη κρατά «κρυμμένες» αυτές τις δυνατότητες από τους αρχάριους προγραμματιστές και δεν τους αναγκάζει να προγραμματίσουν με κάποιο συγκεκριμένο μοντέλο.
- Μπορεί κανείς να την χρησιμοποιήσει διαδραστικά γράφοντας μία εντολή με τον ίδιο τρόπο που γράφουμε σε μία αριθμομηχανή και να πάρει το αποτέλεσμα.
- Μπορεί να συνδυαστεί με άλλες δημοφιλείς γλώσσες όπως, C, C++ και Java.
- Αποτελεί ελεύθερο λογισμικό και υπάρχουν εκδόσεις για κάθε λειτουργικό σύστημα (ακόμη και για κινητά τηλέφωνα). Στο Διαδίκτυο μπορεί κανείς να βρει επίσης πληθώρα βιβλιοθηκών και υποστηρικτικού λογισμικού για την Python που έχουν κατασκευαστεί από διάφορους προγραμματιστές και διατίθενται ελεύθερα.

Η Python και οι απαραίτητες βιβλιοθήκες που χρειάστηκαν στην υλοποίηση του συστήματος μας είναι προεγκατεστημένα στο λειτουργικό σύστημα Raspbian.

Από το πρόγραμμα `rfrg.py` (Παράρτημα Β) αξίζει να προσέξουμε τις μεταβλητές που μπαίνουν τα μεγέθη των μετρήσεων και που καθορίζουν εν τέλει σε ποιες θέσεις βρίσκονται στο πίνακα του πακέτου.

```

ide= (data[2:5])
st=int(data[5])
sh=int(data[6])
sl=int(data[7])
slp=int(data[8])
slt=int(data[9])
su=int(data[10])
temp = float ((data[11:(11+st)]))
humidity = float ((data[(11+st):(11+st+sh)]))
Lux =float ((data[(11+st+sh):(11+st+sh+sl)]))
Lpress=float ((data[(11+st+sh+sl):(11+st+sh+sl+slp)]))
Ltemp=float ((data[(11+st+sh+sl+slp):(11+st+sh+sl+slp+slt)]))
Uv=int ((data[(11+st+sh+sl+slp+slt):(11+st+sh+sl+slp+slt+su)]))

```

Εικόνα 4. 22 Μεταβλητές μεγέθους μετρήσεων & θέσεις μετρήσεων

Τέλος την If statement που διαχωρίζει ανάλογα με το Ide σε ποιον πίνακα της βάσης δεδομένων θα στείλει τις μετρήσεις.

```

if (ide == "1.1"):
    userdata ={"temperature" : temp,"humidity": humidity,"Lux":Lux,"lps_temperature":Ltemp,"lps_pressure":Lpress,"UV-Light":Uv }
    resp = requests.post('http://localhost/data_esties.php',params=userdata)
if (ide == "1.2"):
    userdata ={"temperature" : temp,"humidity": humidity, "Lux": Lux,"lps_temperature":Ltemp,"lps_pressure":Lpress,"UV-Light":Uv }
    resp = requests.post('http://localhost/data_athina.php',params=userdata)

```

Εικόνα 4. 23 If statement

Για να τρέξουμε το πρόγραμμα σε πρώτο στάδιο, ώστε η κεραία του διακομιστή να δέχεται τα πακέτα των δύο σταθμών αρχικά πρέπει να κατευθυνθούμε στο directory που βρίσκεται το πρόγραμμα. Για την εργασία ανοίγουμε το terminal και εκτελούμε την εντολή :

- Cd /home/pi/Downloads

Αν θέλουμε να δούμε τα περιεχόμενα του φακέλου πατάμε Ls και βλέπουμε ότι το αρχείο που θέλουμε είναι πράγματι εκεί. Για να τρέξουμε το πρόγραμμα εκτελούμε την εντολή:

- Python rfp.py

Αυτή η εντολή βάζει το πρόγραμμα να τρέχει. Αν θέλουμε να σταματήσουμε το πρόγραμμα πατάμε οποιαδήποτε στιγμή ctrl+c(keyboard interrupt). Όσο τρέχει το πρόγραμμα η κεραία είναι σε θέση να λαμβάνει πακέτα, να τα ξεχωρίζει όπως αναλύσαμε προηγουμένως και να στέλνει τα δεδομένα στον εκάστοτε πίνακα της βάσης δεδομένων.

4.5.2 Ανεμιστήρας

Όσο έτρεχε το rfry.py, όπως αναφέρθηκε και πιο πάνω, διαπιστώθηκε σημαντική αύξηση της θερμοκρασίας του επεξεργαστή του διακομιστή. Με το σκεπτικό ότι το πρόγραμμα για την κεραία θα χρειάζεται να τρέχει ,αν όχι συνέχεια, τότε για μεγάλα χρονικά διαστήματα, η αύξηση της θερμοκρασίας έπρεπε να αντιμετωπιστεί. Το πρόβλημα λύθηκε με ένα απλό πρόγραμμα και με την σύνδεση ενός τρανζίστορ με ένα ανεμιστήρα στην πλακέτα του Raspberry Pi. Το πρόγραμμα λέγεται ran-fan.py (Παράρτημα Β) και στην ουσία ελέγχει την θερμοκρασία του επεξεργαστή και εάν αυτή ξεπεράσει τους 39 βαθμούς τότε το πρόγραμμα ενεργοποιεί τον ανεμιστήρα. Ο ανεμιστήρας σταματάει όταν η θερμοκρασία φτάσει στο επιθυμητό επίπεδο (κάτω από 39).

```
def getTEMP():  
    CPU_temp = float(getCPUtemperature())  
    if CPU_temp>maxTMP:  
        fanON()  
    else:  
        fanOFF()  
    return()
```

Εικόνα 4. 24 If statement που ανοιγοκλείνει τον ανεμιστήρα

Το πρόγραμμα τρέχει με την απλή εντολή:

- Python ran-fan.py

Όμως δεν θέλουμε το πρόγραμμα κάθε φορά να το τρέχει ο χρήστης χειροκίνητα. Θέλουμε να τρέχει το πρόγραμμα κάθε φορά που κάνει Boot το Raspberry Pi. Για να το πετύχουμε αυτό θα χρησιμοποιήσουμε τον Crontab.

4.5.3 Χρόνο-προγραμματιστής Cron

Το βοηθητικό πρόγραμμα λογισμικού Cron είναι ο χρόνοπρογραμματιστής διεργασιών (task scheduler) του Raspbian και γενικά των λειτουργικών συστημάτων βασισμένων σε Unix. Ο Cron χρησιμοποιείται, δηλαδή, από τους προγραμματιστές για να δώσουν εντολή στο σύστημα να εκτελέσει αυτόματα μία διεργασία σε

συγκεκριμένη ώρα και ημερομηνία. Οι εντολές δίνονται με τη μορφή εντολών συστήματος (system commands) ή σεναρίων κελύφους (shell scripts). Ακολουθούν μερικά παραδείγματα διεργασιών που μπορούμε να προγραμματίσουμε με χρήση του Cron:

- Τερματισμός του συστήματος αυτόματα σε συγκεκριμένη ώρα και ημέρα.
- Λήψη email αυτόματα ανά τακτά χρονικά διαστήματα εφόσον υπάρχει σύνδεση στο διαδίκτυο.
- Προγραμματισμός αυτόματου backup αρχείων κ.α. Ο Cron αντλεί τις πληροφορίες που χρειάζεται από ένα αρχείο Crontab. Ουσιαστικά πρόκειται για ένα αρχείο παραμετροποίησης (configuration file) που περιέχει όλες τις παραμέτρους που χρειάζεται ο Cron για να χρόνο-προγραμματίσει μία διεργασία.

Για να βάλουμε το πρόγραμμα να τρέχει κάθε φορά που κάνει boot το Raspberry pi πρέπει να κάνουμε τα εξής βήματα:

- Ανοίγουμε το terminal και πάμε με την εντολή cd(change directory) στην τοποθεσία του αρχείου.
- Εκεί τρέχουμε την εντολή sudo nano launcher.sh .

Στο αρχείο που θα ανοίξει στον text editor γράφουμε τα εξής:

```
#!/bin/sh
# launcher.sh
# navigate to home directory, then to this directory, then execute python script, then
back home

cd /
cd home/pi/
sudo python ran-fan.py
cd /
```

- Κάνουμε αποθήκευση και κλείνουμε το αρχείο launcher

Για να κάνουμε το αρχείο Launcher εκτελέσιμο τρέχουμε την εντολή:

- chmod 755 launcher.sh

Αν θέλουμε να κάνουμε τεστ ότι όντως έγινε μπορούμε με την εντολή:

- sh launcher.sh

Για να χρησιμοποιήσουμε τον Cron πρέπει πρώτα να φτιάξουμε ένα αρχείο για να καταχωρούνται τα οποιαδήποτε σφάλματα(error logs), ώστε να ξέρει που να πηγαίνει ο Cron σε τέτοιες περιπτώσεις. Τα βήματα είναι τα εξής:

- Cd στο home directory
- Mkdir logs

Για να χρησιμοποιήσουμε τον Cron τρέχουμε την παρακάτω εντολή :

- `sudo crontab -e`

Στο αρχείο που θα ανοίξει προσθέτουμε την γραμμή:

```
@reboot sh /home/pi/ran-fan/launcher.sh >/home/pi/logs/cronlog 2>&1
```

Αποθηκεύουμε, κλείνουμε το αρχείο και κάνουμε reboot το Raspberry Pi. Αν όλα έχουν πάει καλά και εάν έχουμε συνδέσει σωστά τον ανεμιστήρα με το τρανζίστορ και την πλακέτα ο ανεμιστήρας θα πρέπει να ξεκινήσει να δουλεύει μετά από κάποιο χρονικό διάστημα(`crutemp>39 C`). Επαναλαμβάνουμε την ίδια διαδικασία για το πρόγραμμα `rfry.py` ώστε να μην χρειάζεται να το ξεκινάμε κάθε φορά χειροκίνητα όταν κάνουμε reboot στον διακομιστή.

5 ΚΕΦΑΛΑΙΟ ΠΕΜΠΤΟ

5.1 Ιστοσελίδα για παρουσίαση και αλληλεπίδραση με τα δεδομένα

Αφού ολοκληρώσαμε την εγκατάσταση όλου του δικτύου για την εργασία το μόνο που έμεινε είναι η δημιουργία ενός ιστότοπου για αλληλεπίδραση με τον χρήστη και παρουσίαση των δεδομένων. Τι εργαλεία χρησιμοποιήσαμε όμως για να φτιάξουμε το site;

5.1.1 HTML & CSS

Στον παγκόσμιο ιστό η γλώσσα HTML (Hypertext Markup Language) χρησιμοποιείται ως γλώσσα κωδικοποίησης τους περιεχομένου των ιστοσελίδων. Η γλώσσα αυτή είναι υποσύνολο της γλώσσας SGML (Standard Generalized Markup Language). Η έκδοση 4 της HTML αποτελεί πρότυπο ISO (ISO/IEC 15445). Η έκδοση HTML 5 έχει λάβει τη μορφή ενός υπό διαμόρφωση προτύπου, το οποίο ήδη υποστηρίζεται από τους σύγχρονους φυλλομετρητές (Explorer, Firefox, Chrome κ.λπ.). Ο οργανισμός που ορίζει τα πρότυπα του διαδικτύου είναι ο w3c (world-wide web consortium). Στην πρώτη έκδοση της HTML υπήρχε η δυνατότητα απεικόνισης στατικής και κινούμενης εικόνας καθώς και η δυνατότητα αναπαραγωγής ήχου. Στη δεύτερη έκδοση προστέθηκε η δυνατότητα εισαγωγής στοιχείων από το χρήστη μέσω φορμών. Στην τελευταία έκδοση έχει προστεθεί η δυνατότητα απεικόνισης πινάκων, μαθηματικών συμβόλων κ.α. Ένα έγγραφο HTML αποτελείται από κείμενο και ετικέτες (tags). Οι ετικέτες είναι οι εντολές οι οποίες καθορίζουν την εμφάνιση του κειμένου, τη δόμηση του σε ενότητες, ενώ εισάγουν και υπερσυνδέσμους. Οι περισσότερες ετικέτες εσωκλείονται μεταξύ των χαρακτήρων "<" και ">", ώστε να είναι δυνατή η διαφοροποίηση τους από το κείμενο. Υπάρχουν απλές και διπλές ετικέτες. Ένα παράδειγμα απλής ετικέτας είναι η "
" η οποία δηλώνει αλλαγή γραμμής στο κείμενο. Στην περίπτωση διπλών ετικετών, αυτές υποδεικνύουν την αρχή και το τέλος του κειμένου που θα επηρεάσουν, για παράδειγμα hello .

Πιο κάτω αναφέρονται ενδεικτικά μερικές από τις βασικές ετικέτες της HTML με σύντομη επεξήγηση τους:.

- <html></html>: Κάθε ιστοσελίδα πρέπει να αρχίζει και να τελειώνει με αυτήν την ετικέτα αφού χρησιμοποιείται ως κωδικός αναγνώρισης του κειμένου ως HTML από το πρόγραμμα περιήγησης.

- `<head></head>` : Η ετικέτα αυτή ορίζει την επικεφαλίδα και παρέχει πληροφορίες για το έγγραφο που ακολουθεί.
- `<title></title>` : Ο τίτλος της ιστοσελίδας.
- `<body></body>`: Μέσα σε αυτή τη διπλή ετικέτα περιέχονται όλα τα συστατικά στοιχεία της σελίδας, μορφοποιημένο κείμενο, εικόνες, ήχοι κ.λπ.
- `<h1></h1>`: Δηλώνει την επικεφαλίδα πρώτου επιπέδου
- `<p> </p>`: Ετικέτα αλλαγής παραγράφου

Η HTML σήμερα χρησιμοποιεί σε μεγάλο βαθμό για τη μορφοποίηση περιεχομένου την τεχνολογία φύλλων μορφοποίησης CSS (Cascading Style Sheets) τα οποία επιτρέπουν το διαχωρισμό και την επαναχρησιμοποίηση εντολών μορφοποίησης, κάτι αντίστοιχο με τα στυλ μορφοποίησης κειμένου στους επεξεργαστές κειμένου. Η CSS είναι μια γλώσσα προγραμματισμού προορισμένη να αναπτύσσει στιλιστικά μια ιστοσελίδα, δηλαδή, να διαμορφώνει τα χαρακτηριστικά της όπως χρώματα, στοίχιση, γραμματοσειρές. Για μια όμορφη και καλοσχεδιασμένη ιστοσελίδα η χρήση της CSS κρίνεται ως απαραίτητη.

Η σύνδεση του κώδικα σε CSS με την HTML γίνεται είτε μέσα στο ίδιο το αρχείο HTML (όπως το αρχείο της σελίδας της εργασίας) όπου και τον τοποθετούμε μέσα στη διπλή ετικέτα `<style></style>` είτε τον γράφουμε σε ξεχωριστό αρχείο (.css) την ύπαρξη του οποίου δηλώνουμε κατάλληλα στην επικεφαλίδα (`<head>`) της HTML.

5.1.2 JavaScript & JQuery

Η JavaScript είναι μια scripting γλώσσα η οποία έχει σχεδιαστεί και χρησιμοποιείται για να εισάγουμε την έννοια της διαδραστικότητας στις σελίδες HTML. Είναι μια διερμηνευόμενη (interpreted) γλώσσα, δηλαδή το script εκτελείται χωρίς να έχει περάσει από την διαδικασία της μεταγλώττισης (compiling). Ο διερμηνέας εκτελεί απευθείας το πρόγραμμα, μεταφράζοντας κάθε εντολή σε μία σειρά υπορουτίνων που είναι ήδη μεταγλωττισμένες στη γλώσσα μηχανής. Εκτελείται στο περιβάλλον του browser και επιτρέπει μεγάλη διαδραστικότητα μεταξύ του χρήστη και της ιστοσελίδας. Ο συνδυασμός της JavaScript και της ασύγχρονης εκτέλεσης εντολών HTTP (AJAX: Asynchronous JavaScript and XML) επιτρέπουν την ανανέωση του περιεχομένου της ιστοσελίδας χωρίς τη μετάβαση σε νέα ιστοσελίδα. Σήμερα εμφανίζεται όλο και περισσότερο το φαινόμενο ο χρήστης να παραμένει στην ίδια ιστοσελίδα με την οποία αλληλοεπιδρά και ως συνέπεια των ενεργειών του το περιεχόμενο της ανανεώνεται με κλήσεις του πρωτοκόλλου HTTP. Τέτοια παραδείγματα είναι διαδικτυακές εφαρμογές (Web apps) όπως το google maps, το Facebook, διαδικτυακά παιχνίδια κ.λπ. Μερικές από τις βασικότερες δυνατότητες που μας δίνει η JavaScript:

- Μπορούμε να εκτελέσουμε κάποια ενέργεια όταν συμβαίνει ένα γεγονός, για παράδειγμα όταν ο χρήστης κάνει κλικ σε ένα HTML στοιχείο, να εκτελείται κάποιο script και λαμβάνουμε τα αντίστοιχα αποτελέσματα.
- Η JavaScript μπορεί να διαβάσει και να αλλάξει τα περιεχόμενα ενός HTML στοιχείου.
- Μπορούμε να την χρησιμοποιήσουμε για να επικυρώσουμε δεδομένα μιας φόρμας (validate) πριν υποβληθούν στον Server. Χάρη σε αυτή τη δυνατότητα και αφού η JavaScript τρέχει στο περιβάλλον του χρήστη (client) απαλλάσσουμε το Server από επιπλέον υπολογιστικό έργο για να κάνει αυτός τον έλεγχο.
- Μπορούμε να δημιουργήσουμε cookies (πακέτα δεδομένων που χρησιμοποιεί ο browser για να θυμάται πράγματα και να κάνει την περιήγηση μας πιο εύκολη).

Η jQuery είναι μια open source βιβλιοθήκη της JavaScript, συμβατή με όλους τους browsers που κυκλοφορούν. Απλοποιεί την εκμάθηση και την χρήση της γλώσσας JavaScript και χρησιμοποιείται στην δημιουργία ιστοσελίδων και Web εφαρμογών. Ενδεικτικά, με την χρήση της μπορούμε:

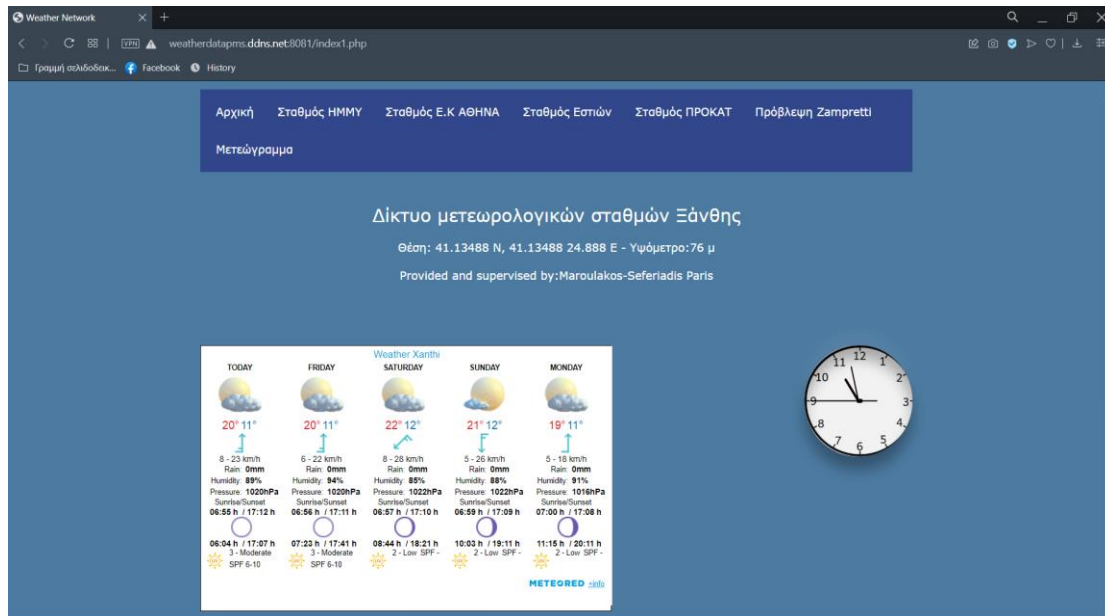
- Να προσθέσουμε εύκολα κίνηση (animation)
- Να αυξήσουμε την διαδραστικότητα του χρήστη (user interaction)
- Να αλλάξουμε το περιεχόμενο της σελίδας μέσω AJAX χωρίς ο χρήστης να πρέπει να μεταφερθεί σε νέα σελίδα, κυριολεκτικά σε μία γραμμή κώδικα κάτι που με τη χρήση απλής JavaScript θα ήταν χρονοβόρο
- Να δημιουργήσουμε διάφορα εφέ και πολλά περισσότερα. Η σύνδεση του κώδικα σε JavaScript/jQuery με την HTML γίνεται είτε μέσα στο ίδιο το αρχείο HTML όπου και τον τοποθετούμε μέσα στη διπλή ετικέτα `<script></script>` είτε τον γράφουμε σε ξεχωριστό αρχείο (.js) την ύπαρξη του οποίου δηλώνουμε κατάλληλα στην επικεφαλίδα `<head>` της HTML.

5.1.3 Σελίδα

Όλα τα απαραίτητα αρχεία για να τρέχει η web σελίδα είναι στο directory `/var/www/html` του Raspberry pi όπως έχουμε πει και πιο πάνω.

- Το αρχείο `Index.php` είναι ο κώδικας της αρχικής σελίδας (εικόνα 5.1) που περιέχει το μενού. Έχει ένα widget που δείχνει μετεωρολογικά δεδομένα για την Ξάνθη και την πρόβλεψη του καιρού για τις επόμενες 4 μέρες, για σύγκριση με τα δεδομένα των σταθμών και με την πρόβλεψη μέσω του αλγορίθμου του Zampretti. Ακόμη με την βοήθεια του `main.js` και του `style.css`

έχει ένα αναλογικό ρολόι που δείχνει την ώρα καθώς και μια κάμερα που δείχνει στην κεντρική πλατεία της πόλης.



Εικόνα 5. 1 Αρχική σελίδα Index.php

- Το αρχείο connection.php είναι απαραίτητο για τις σελίδες των σταθμών του ιστότοπου, καθώς μέσω αυτού συνδέεται στην βάση δεδομένων για να αντλήσει τα δεδομένα των μετρήσεων. Χωρίς αυτό δεν θα μπορούσαν να εκτελεστούν τα sql queries(Εικόνα 5.2) που βάζουν τα δεδομένα στους πίνακες, δεν θα μπορούσαμε να δούμε τις μετρήσεις και δεν θα μπορούσαμε να αναπτύξουμε τα γραφήματα στις σελίδες των σταθμών.

```
<?php
include('connection.php');
$result = mysqli_query($con, 'SELECT * FROM hmy ORDER BY id DESC LIMIT 5');
// Process every record
$row = true;

while($row = mysqli_fetch_array($result))
{
    if ($row)
    {
        $css_class=' class="table_cells_odd"';
    }
    else
    {
        $css_class=' class="table_cells_even"';
    }
    $row = !$row;
    echo "<tr align=center style= 'background:white;'>";
    echo "<td '." . $row['event'] . "</td>";
    echo "<td '." . $row['DHT_temperature'] . "</td>";
    echo "<td '." . $row['DHT_humidity'] . "</td>";
    echo "<td '." . $row['Brightness'] . "</td>";
    echo "<td '." . $row['LPS_pressure'] . "</td>";
    echo "<td '." . $row['UV_Light'] . "</td>";
    echo "</tr>";
}

// Close the connection
mysqli_close($con);
?>
```

Εικόνα 5. 2 SQL Query

- Τα αρχεία data_hmmy.php, data_prokat.php, που υπάρχουν στο ίδιο directory, είναι τα αρχεία που χρησιμοποιούν οι σταθμοί-arduino για να στείλουν τις μετρήσεις τους στους πίνακες της βάσης δεδομένων. Τα αρχεία data_esties.php, data_athina.php είναι τα αρχεία που καλεί το πρόγραμμα rfrpy.py όταν διαβάζει δεδομένα από την κεραία του Raspberry pi.

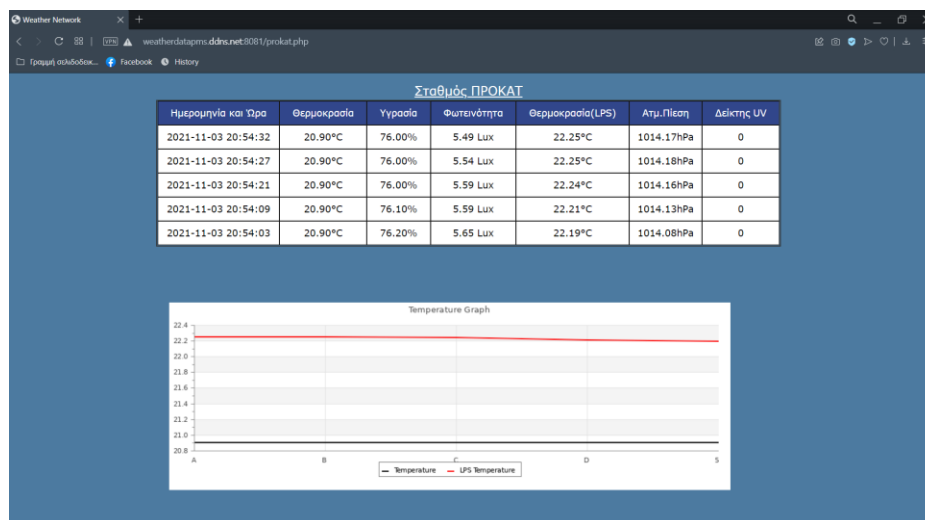
```

1 //php
2 include ('connection.php');
3 $sql_insert = "INSERT INTO esties (DHT_temperature, DHT_humidity, Brightness, LPS_pressure, LPS_humidity, UV_Light) VALUES ('".$_GET["temperature"]."','".$_GET["humidity"]."','".$_GET["brightness"]."','".$_GET["lps_pressure"]."','".$_GET["lps_humidity"]."','".$_GET["uv_light"]."')";
4 if(mysqli_query($con,$sql_insert))
5 {
6     echo "Done";
7     mysqli_close($con);
8 }
9 else
10 {
11     echo "error is ".mysqli_error($con);
12 }
13 >>

```

Εικόνα 5. 3 data_esties.php

- Τα αρχεία hmmy.php, esties.php, ekathens.php, prokat.php περιέχουν τον κώδικα για τις σελίδες των σταθμών και καλούνται από το μενού της κάθε σελίδας. Η κάθε σελίδα έχει τους πίνακες με τις τελευταίες μετρήσεις των σταθμών, καθώς και γραφήματα για την παρουσίασης των τιμών και της συμπεριφοράς κάθε μετεωρολογικού μεγέθους. Οι σελίδες ανανεώνονται αυτόματα κάθε δυο λεπτά για να περνάνε στους πίνακες οι τελευταίες μετρήσεις κάθε σταθμού και να φαίνεται καλύτερα η τάση των διάφορων μετεωρολογικών μεγεθών στο κάθε γράφημα. Επίσης περιέχουν και αυτές μενού για την ευκολότερη περιήγηση του χρήστη στις διάφορες σελίδες.



Εικόνα 5. 4 Πίνακας σταθμού ΠΡΟΚΑΤ και γράφημα θερμοκρασιών

- Το αρχείο Zampretti.php μαζί με το betel_cast.js (αρχείο Javascript) περιέχει τον κώδικα για την σελίδα της πρόβλεψης και καλείται και αυτό από το μενού της κάθε σελίδας. Η σελίδα έχει μια φόρμα συμπλήρωσης συνθηκών του καιρού η οποία σε συνδυασμό με το αρχείο της javascript μπορεί να δώσει μια πρόβλεψη του καιρού για τις επόμενες δώδεκα ώρες χρησιμοποιώντας και δεδομένα που συλλέγονται από τους σταθμούς που έχουμε υλοποιήσει (ατμοσφαιρική πίεση). Στην σελίδα παρέχονται αναλυτικές οδηγίες για την σωστή συμπλήρωση της φόρμας καθώς και μενού για την ευκολότερη περιήγηση του χρήστη στις διάφορες σελίδες. Ακόμη περιέχονται 2 widgets με πληροφορίες για τις συνθήκες και την συμπεριφορά του ανέμου στην περιοχή της Ξάνθης για ευκολότερη συμπλήρωση της φόρμας.

Πρόβλεψη με αλγόριθμο Zampretti

Δεδομένα

Περιβάλλον:	Ημερομηνία: <input type="text"/>	Εύρος Πίεσης: <input type="text"/>	Χαμηλή: <input type="text"/>	Υψηλή: <input type="text"/>
Τωρινές Μεταβλητές:	Συνθήκες Αέρα: <input type="text"/>	Month: <input type="text"/>		
Πίεση: (Σχετική / Sea Level Adjusted)	hPa: <input type="text"/>	Τάση: <input type="text"/>		
<input type="button" value="Πρόβλεψη"/>				

Πρόβλεψη 12 ωρών: Η πρόβλεψη θα εμφανιστεί εδώ

Τα καλύτερα αποτελέσματα τα έχουμε όταν κάνουμε την πρόβλεψη στις 9 το πρωί

Οδηγίες για συμπλήρωση της φόρμας πρόβλεψης:
 Αρχικά συμπληρώνουμε στις τωρινές μεταβλητές τις συνθήκες αέρα και το μήνα που βλέπουμε από την κάρτα κάτω
 Στην συνέχεια συμπληρώνουμε την τιμή της πίεσης από την πιο πρόσφατη τιμή από αυτές που βλέπουμε στους πίνακες των σταθμών.
 Για την τάση και το βαρομετρικό χαμηλό και υψηλό ηγαιόμαστε στο site pentell.meteo.gr/stations/xanthi/ και παίρνουμε τις σχετικές τιμές.
 Η πρόβλεψη δίνει τα καλύτερα αποτελέσματα αν γίνει στις 9 το πρωί(τοπική ώρα)

Xanthi WINDFINDER

Full statistic > Full forecast > Superforecast > Forecast map >

Whole year: \swarrow 327° (NNW) / 10 km/h / 18°C

Month	Jan	Feb	Mar	Apr	May	Jun	Jul
Wind direction (daytime Ø)	\swarrow	\swarrow	\swarrow	\swarrow	\rightarrow	\swarrow	\swarrow
Wind speed	NW	NNW	NNW	NW	W	NNW	NW

Εικόνα 5. 5 Φόρμα πρόβλεψης καιρού και widget ανέμων

- Το αρχείο Meteo.php περιέχει τον κώδικα για την σελίδα του μετεωγράμματος και καλείται από το μενού της κάθε σελίδας. Η σελίδα παρουσιάζει το μετεώγραμμο 5 ημερών της Ξάνθης, καθώς και αναλυτικές οδηγίες για το πως διαβάζεται. Το μετεώγραμμο ανανεώνεται κάθε 12 ώρες για να δείχνει τις τελευταίες μετρήσεις.

6 ΚΕΦΑΛΑΙΟ ΕΚΤΟ

6.1 Συμπεράσματα

Από τις δοκιμές του συστήματος που πραγματοποιήθηκαν στο πλαίσιο της διπλωματικής εργασίας, προέκυψε ότι:

- Επιτεύχθηκε ο πρώτος βασικός στόχος που αφορούσε τη δυνατότητα εγκατάστασης στην μεγαλύτερη δυνατή ποικιλία περιβαλλόντων που ενδεχόμενα θα απαιτήσει ένας χρήστης κυρίως λόγω της ποικιλίας πρωτοκόλλων επικοινωνίας με τον εξυπηρετητή .
- Επιτεύχθηκε ο δεύτερος βασικός στόχος που αφορούσε το ευελιξία του δικτύου όσον αφορά την μετατροπή σε δίκτυο που να χρησιμοποιεί λιγότερους σταθμούς.
- Η ασύρματη επικοινωνία μεταξύ των κόμβων του συστήματος είναι αρκετά αξιόπιστη.
- Επιτεύχθηκε και ο τρίτος βασικός στόχος που αφορούσε την συλλογή αξιόπιστων μετεωρολογικών δεδομένων με την χρησιμοποίηση και κατάλληλη παραμετροποίηση υψηλής ακρίβειας αισθητήρων.
- Τέλος επιτεύχθηκε και ο στόχος της δημιουργίας ενός ιστότοπου φιλικού στον χρήστη, που παρουσιάζει αναλυτικά τα δεδομένα που συλλέγονται από τους σταθμούς και ως ένα βαθμό δίνει την δυνατότητα αξιοποίησης τους (πρόβλεψη Zampretti).

Η ασύρματη επικοινωνία των σταθμών του δικτύου είναι αρκετά αξιόπιστη κυρίως για τους σταθμούς που χρησιμοποιούν το Wi-fi και το GSM και η χρήση των SD καρτών τοπικά, φροντίζει για μην χαθούν τα πακέτα των μετρήσεων αν και εφόσον για κάποιο λόγο αυτά δεν φτάσουν στον προορισμό τους.. Ταυτόχρονα το κόστος κρατήθηκε όσο γινόταν σε χαμηλά. Φυσικά το κόστος κατασκευής θα μειωθεί πολύ περισσότερο σε περίπτωση μαζικής παραγωγής του δικτύου.

Αξίζει να τονιστεί ότι το δίκτυο έχει υλοποιηθεί ώστε να προσαρμόζεται όσο το δυνατόν δυναμικά ανάλογα με την εγκατάσταση, δηλαδή να χρειάζεται την ελάχιστη δυνατή επέμβαση στους πηγαίους κώδικες.

6.2 Βελτιώσεις-Προτάσεις

Το κομμάτι των προτάσεων για βελτιώσεις είναι αρκετά ευρύ, καθώς η εργασία αποτελείται από πολλές και διαφορετικές μεταξύ τους ενότητες και υποενότητες.

6.2.1 Μετεωρολογικοί σταθμοί

Αρχικά, η πιο προφανής βελτίωση που μπορεί να γίνει στους σταθμούς είναι η πρόσθεση και άλλων αισθητηρίων που μετράνε μετεωρολογικά δεδομένα. Για

παράδειγμα θα μπορούσαν να προστεθούν αισθητήρια που να μετράνε την ταχύτητα και διεύθυνση του αέρα, την πυκνότητα της βροχής, τα επίπεδα διαφόρων αερίων στην ατμόσφαιρα και άλλοι. Ακόμη θα μπορούσαν να προστεθούν αισθητήρια που μετράνε τα ίδια δεδομένα με ίδια η μεγαλύτερη ακρίβεια για σύγκριση μεταξύ των μετρήσεων και καλύτερη αξιολόγηση της συμπεριφοράς των μεγεθών.

Δεύτερον, θα μπορούσε να προστεθεί κώδικας για λιγότερη κατανάλωση ενέργειας από τους σταθμούς. Δηλαδή στο χρονικό διάστημα μεταξύ των μετρήσεων θα μπορούσαμε να βάλουμε τα Arduino σε sleep mode. Ωστόσο για να γίνει κάτι τέτοιο θα πρέπει το χρονικό διάστημα μεταξύ των μετρήσεων να αυξηθεί ώστε η εξοικονόμηση ενέργειας να έχει νόημα, καθώς στην παρούσα υλοποίηση το διάστημα είναι αρκετά μικρό.

Τρίτον, θα μπορούσε να τροποποιηθεί ο κώδικας ώστε στο αρχείο της SD να καταγράφονται μόνο οι μετρήσεις που χάνονται κατά την αποστολή στον εξυπηρετητή και όταν το αρχείο αυτό γεμίζει, όταν μπαίνει νέα μέτρηση να διαγράφεται η παλαιότερη του αρχείου και ούτω καθεξής για να μην χρειάζεται ποτέ αλλαγή η SD. Αυτή η «βελτίωση» προτείνεται για υλοποιήσεις που στοχεύουν χρήστες που δεν χρειάζονται ιδιαίτερα λεπτομερή καταγραφή των μετρήσεων τοπικά.

Τέταρτον, θα μπορούσε να τροποποιηθεί ο κώδικας στην υλοποίηση των πρωτόκολλων επικοινωνίας με τον εξυπηρετητή, ώστε αυτή να γίνει αμφίδρομη, κάτι που από την μια θα αφαιρούσε την αυτονομία των σταθμών, αλλά από την άλλη θα έδινε την δυνατότητα σε ένα χρήστη, να ελέγχει σε πολύ μεγαλύτερο βαθμό τις συνθήκες και τον τρόπο λειτουργίας τους, χωρίς απαραίτητα να απαιτεί την παρουσία του τοπικά στον σταθμό για κάποια αλλαγή .

Τέλος, θα μπορούσαμε να προσθέσουμε ένα αριθμό από άλλους μετεωρολογικούς σταθμούς με διαφορετικά πρωτόκολλα επικοινωνίας για να μειωθεί ακόμα παραπάνω η πιθανότητα να χάσουμε την επικοινωνία με πάνω από μια πλακέτα την φορά. Η πρόταση αυτή δίνει επίσης μεγαλύτερη ποικιλία επιλογών στο χρήστη όσον αφορά πόσους και ποιους σταθμούς θέλει να χρησιμοποιήσει και την δυνατότητα συλλογής περισσότερων μετεωρολογικών δεδομένων μιας περιοχής.

6.2.2 Server

Αρχικά θα μπορούσε η ιστοσελίδα να βελτιωθεί σημαντικά και να γίνει πιο φιλική και διαδραστική με τον χρήστη, προσθέτοντας chat ερωτήσεων , Faq, δυνατότητα sign up & sign in και πολλά άλλα.

Ακόμη αν συνδυάσουμε την πρόταση για προσθήκη αισθητηρίων που μετράνε και άλλα μεγέθη στους σταθμούς, θα μπορούσε να τροποποιηθεί το υπάρχον script, που κάνει την πρόβλεψη του καιρού, με τέτοιο τρόπο ώστε η συμπλήρωση της φόρμας να βασίζεται αποκλειστικά σε δικά μας δεδομένα.

Τέλος, σε συνδυασμό με τη τροποποίηση της υλοποίησης των πρωτοκόλλων επικοινωνίας των σταθμών με τον server, θα μπορούσε να προστεθεί μια σελίδα ακόμα στο ιστότοπο, η οποία θα υλοποιηθεί με τέτοιο τρόπο, που να δίνει σε ένα χρήστη την δυνατότητα ελέγχου και τροποποίησης των σταθμών μέσω αυτής.

7 Βιβλιογραφία

1. Καλαϊτζάκης, Κ. και Κουτρούλης, Ε. ΗΛΕΚΤΡΙΚΕΣ ΜΕΤΡΗΣΕΙΣ ΚΑΙ ΑΙΣΘΗΤΗΡΕΣ. σ.λ. : ΚΛΕΙΔΑΡΙΘΜΟΣ, 2010.
2. Larry L. Peterson - Bruce S. Davie- Επιμέλεια Β. Τσαουσίδης Δίκτυα υπολογιστών - Μια προσέγγιση από τη σκοπιά των συστημάτων 4^η ΑΜΕΡΙΚΑΝΙΚΗ ΕΚΔΟΣΗ ΚΛΕΙΔΑΡΙΘΜΟΣ
3. Ways to transfer data from arduino
<https://forum.arduino.cc/index.php?topic=245101.0>
4. Black, Bruce A., και συν. ΕΙΣΑΓΩΓΗ ΣΤΑ ΑΣΥΡΜΑΤΑ ΣΥΣΤΗΜΑΤΑ. σ.λ. : Μ.ΓΚΙΟΥΡΔΑΣ, 2010.
5. SPI & I2C <https://articles.saleae.com/logic-analyzers/spi-vs-i2c-protocol-differences-and-things-to-consider>
6. element14. Compute Module Dev Kit Comparison Chart. element14.
[Ηλεκτρονικό] [Παραπομπή: 9 10 2015.]
<http://www.element14.com/community/docs/DOC-68090/1/raspberrypi-2-b-a-a-compute-module-dev-kit-comparison-chart>.
7. Tanenbaum, Andrew S. Σύγχρονα λειτουργικά συστήματα. σ.λ. : ΚΛΕΙΔΑΡΙΘΜΟΣ, 2009.
8. Raspbian. Raspbian. [Ηλεκτρονικό] <https://www.raspbian.org/>.
9. —. Raspbian setup guide. [Ηλεκτρονικό]
<https://www.raspberrypi.org/help/noobs-setup/>.
10. —. Downloads. [Ηλεκτρονικό] <https://www.raspberrypi.org/downloads/>
11. Garcia-Molina, Hector, Ullman, Jeffrey D. και Widom, Jenifer. Συστήματα βάσεων δεδομένων. σ.λ. : ΠΑΝΕΠΙΣΤΗΜΙΑΚΕΣ ΕΚΔΟΣΕΙΣ ΚΡΗΤΗΣ.
12. MySQL. MySQL The world's most popular open source database. [Ηλεκτρονικό]
[Παραπομπή: 12 10 2015.] <https://www.mysql.com/>. 102
13. phpMyAdmin. phpMyAdmin Bringing MySQL to the web. [Ηλεκτρονικό]
[Παραπομπή: 12 10 2015.] <http://www.phpmyadmin.net/>.
14. The World Wide Web Consortium. The World Wide Web Consortium.
[Ηλεκτρονικό] <http://www.w3.org/>.
15. Αβούρης, Νικόλαος, και συν. Εισαγωγή στους υπολογιστές με τη γλώσσα Python. σ.λ. : Εκδόσεις Πανεπιστημίου Πατρών, 2012.
16. JavaScript. JavaScript Web site. [Ηλεκτρονικό] [Παραπομπή: 12 10 2015.]
<https://www.javascript.com/>.

17. jQuery. jQuery Web site. [Ηλεκτρονικό] [Παραπομπή: 12 10 2015.]
<https://jquery.com/>.
18. Arduino. Home page. [Online] [Cited: 09 27, 2012.] <http://www.arduino.cc/>.
19. Arduino tutorials <https://www.ardumotive.com/tutorials.html>
20. «Η αξιοποίηση των αισθητήρων του Arduino στις εργαστηριακές και ερευνητικές δραστηριότητες» Πάλλας Αναστάσιος, Ορφανάκης Στυλιανός https://4syn-thess2016.ekped.gr/wp-content/uploads/2016/04/vol4_139-212-223.pdf
21. ΑΣΥΡΜΑΤΑ ΔΙΚΤΥΑ ΑΙΣΘΗΤΗΡΩΝ ΓΟΥΣΗΣ ΓΕΩΡΓΙΟΣ
<https://apothetirio.lib.uoi.gr/xmlui/bitstream/handle/123456789/4771/ΑΣΥΡΜΑΤΑ%20ΔΙΚΤΥΑ%20ΑΙΣΘΗΤΗΡΩΝ-ΓΙΩΡΓΟΣ%20ΓΟΥΣΗΣ.pdf?sequence=1>
22. Καλοβρέκτης, Κωσταντίνος και Κατέβας, Νικόλαος. Αισθητήρες Μέτρησης και Ελέγχου. s.l. : ΤΖΙΟΛΑ, 2014.
23. Α. Γαστεράτος, Σ. Μουρούτσος, Ι. Ανδρεάδης. Τεχνολογία Μετρήσεων - Αισθητήρια. s.l. : Γκιούρδας Εκδοτική, 2008.
24. Raspberry Pi Web Server <https://www.toptal.com/raspberry-pi/how-to-turn-your-raspberry-pi-into-a-development-server>
25. Μουρούτσος, Σ. Γ. και Μάλιαρης, Γ. ΤΕΧΝΙΚΟ ΣΧΕΔΙΟ, Μηχανολογικό, Ηλεκτρολογικό, Σχέδιο βιομηχανικών αυτοματισμών. 2014.
26. Raspberry Pi Web Server with Apache
<https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md>
27. Ethernet shield to send weather data
https://create.arduino.cc/projecthub/muhammad-aqib/logging-data-to-database-using-arduino-ethernet-shield-3e9a0e?ref=tag&ref_id=weather&offset=38
28. Arduino data transmit through GSM
<https://www.allaboutcircuits.com/projects/using-a-sim900a-to-send-sensor-data-to-a-website/>
29. Arduino data transmit through LoRa-02 SX1278
<https://iotdesignpro.com/projects/wireless-communication-between-arduino-and-raspberry-pi-using-lora-module-sx1278>
30. Python to database <http://educ8s.tv/raspberry-pi-online-weather-log/>
31. Sensors code and wiring <https://learn.adafruit.com/>
32. Arduino & Raspberry pi communication with LoRa
<https://iotdesignpro.com/projects/wireless-communication-between-arduino-and-raspberry-pi-using-lora-module-sx1278>

8 ΠΑΡΑΡΤΗΜΑ Α Κώδικες Σταθμών

8.1 A.1 ethernetshieldworkingwithrpi.ino

```
#include <SPI.h>
#include <Ethernet.h>
#include "DHT.h"
#include "Adafruit_TSL2591.h"
#include <Wire.h>
#include <Adafruit_LPS2X.h>
#include <Adafruit_Sensor.h>
#include <SD.h> // include Arduino SD library
#include <ThreeWire.h>
#include <RtcDS1302.h>

#define DHTPIN 2 // Digital pin connected to the DHT sensor
#define DHTTYPE DHT22 // DHT 22
#define LPS_CS 53
#define LPS_SCK 52
#define LPS_MISO 50
#define LPS_MOSI 51
#define countof(a) (sizeof(a) / sizeof(a[0]))

ThreeWire myWire(5,3,6); // IO, SCLK, CE
RtcDS1302<ThreeWire> Rtc(myWire);
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
byte ip[] = {192, 168, 137, 13 }; //Enter the IP of ethernet shield
192.168.1.5
char serv[] = "weatherdatapms.ddns.net"; // the ip adress for pi
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
DHT dht(DHTPIN, DHTTYPE);
File dataLog;
EthernetClient client;
Adafruit_LPS22 lps;
sensors_event_t lps_temp;
sensors_event_t lps_press;
float sensorVoltage;
int sensorValue;

void setup() {
  pinMode(53, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(A3, INPUT);
  digitalWrite(10, HIGH); // disable ethernet cs
  digitalWrite(4, HIGH);
  digitalWrite(53, LOW); // enable lps cs
  lps.begin_SPI(LPS_CS);
  digitalWrite(53, HIGH); // disable lps cs
  digitalWrite(4, LOW);
  if ( !SD.begin(4) )
    Serial.println("failed!"); // initialization error
  digitalWrite(4, HIGH);
  dht.begin();
  tsl.begin();
```

```

    tsl.setGain(TSL2591_GAIN_MED);
    tsl.setTiming(TSL2591_INTEGRATIONTIME_300MS);
    Serial.begin(9600); //setting the baud rate at 9600
    Rtc.Begin();
    if (Rtc.GetIsWriteProtected())
    {
        Serial.println("RTC was write protected, enabling writing
now");
        Rtc.SetIsWriteProtected(false);
    }

    if (!Rtc.GetIsRunning())
    {
        Serial.println("RTC was not actively running, starting now");
        Rtc.SetIsRunning(true);
    }
    RtcDateTime now = Rtc.GetDateTime();
    digitalWrite(10, LOW); // enable ethernet
    Ethernet.begin(mac, ip);
}

void loop() {

    float h = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();
    uint32_t lum = tsl.getFullLuminosity();
    uint16_t ir, full;
    ir = lum >> 16;
    full = lum & 0xFFFF;
    float L = tsl.calculateLux(full, ir);
    digitalWrite(10, HIGH); // disable ethernet cs

    digitalWrite(53, LOW); // enable lps cs
    lps.getEvent(&lps_press, &lps_temp); // get lps pressure & lps temp
    float temperature = lps_temp.temperature;
    float pressure = lps_press.pressure;
    digitalWrite(53, HIGH); // disable lps cs
    sensorValue = analogRead(A3);
    sensorVoltage = (5.0/1023.0)*(sensorValue)*(1000);
    int uv=volts_to_uv(sensorVoltage);
    Serial.print(F(" Humidity: "));
    Serial.print(h);
    Serial.print(F("% Temperature: "));
    Serial.print(t);
    Serial.print(F("C "));

    // Serial.print(F("F Heat index: "));
    // Serial.print(hic);
    // Serial.print(F("C "));

    Serial.print("Lux = ");
    Serial.println(L);
    Serial.print("Pressure = ");
    Serial.println(pressure);
    Serial.print("Ltemp = ");
    Serial.println(temperature);
    Serial.print("sensorVoltage = ");
    Serial.println(sensorVoltage);
    Serial.print("UV-Index = ");
    Serial.println(uv);
}

```

```

RtcDateTime now = Rtc.GetDateTime();
char datestring[22];
snprintf_P(datestring,
            countof(datestring),
            PSTR("%02u/%02u/%04u    %02u:%02u:%02u"),
            now.Month(),
            now.Day(),
            now.Year(),
            now.Hour(),
            now.Minute(),
            now.Second());
Serial.println(datestring);
if (!now.IsValid())
{
    // Common Causes:
    //    1) the battery on the device is low or even missing and
the power line was disconnected
    Serial.println("RTC lost confidence in the DateTime!");
}

digitalWrite(4, LOW); // ENABLE SD

//open Log.txt file with write permission

dataLog = SD.open("LogEthernet.txt", FILE_WRITE);
dataLog.print(datestring);
dataLog.print("    ");
dataLog.print(t);
dataLog.print(" °C    |    ");
dataLog.print(h);
dataLog.print(" | ");
dataLog.print(L);
dataLog.print(" | ");
dataLog.print(pressure);
dataLog.print(" °C    |    ");
dataLog.println(temperature);
dataLog.close();
delay(100); // wait 100ms
digitalWrite(4, HIGH); // DISABLE SD

digitalWrite(10, LOW); // enable ethernet
if (client.connect(serv, 8081)) { //Connecting at the IP address
and port we saved before
    Serial.println("connected");
    client.print("GET /data_hmmy.php?"); //Connecting and Sending ues
to database
    client.print("&temperature=");
    client.print(t);
    client.print("&humidity=");
    client.print(h);
    client.print("&Lux=");
    client.print(L);
    client.print("&lps_temperature=");
    client.print(temperature);
    client.print("&lps_pressure=");
    client.print(pressure);
    client.print("&UV-Light=");
    client.println(uv);
    client.stop(); //Closing the connection
}

```



```

    else {
        // if you didn't get a connection to the server:
        Serial.println("connection failed");
    }
    // digitalWrite(10, HIGH); // enable ethernet
    delay(5000);
}

int volts_to_uv(float sensorVoltage)
{
    int uv;
    if (sensorVoltage<50.0){
        uv=0;
    }
    else if (sensorVoltage<=227.0){
        uv=1;
    }
    else if (sensorVoltage<=318.0){
        uv=2;
    }
    else if (sensorVoltage<=408.0){
        uv=3;
    }
    else if (sensorVoltage<=503.0){
        uv=4;
    }
    else if (sensorVoltage<=606.0){
        uv=5;
    }
    else if (sensorVoltage<=696.0){
        uv=6;
    }
    else if (sensorVoltage<=795.0){
        uv=7;
    }
    else if (sensorVoltage<=881.0){
        uv=8;
    }
    else if (sensorVoltage<=976.0){
        uv=9;
    }
    else if (sensorVoltage<=1079.0){
        uv=10;
    }
    else {
        uv=11;
    }
    return uv;
}

```

8.2 A.2 mkrghsm1400working.ino

```
#include <MKRGSM.h>
#include <DHT.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include "Adafruit_TSL2591.h"
#include <Adafruit_LPS2X.h>
#include <SD.h> // include Arduino SD library
#include <ThreeWire.h>
#include <RtcDS1302.h>

#define DHTPIN 2 // Digital pin connected to the DHT sensor
#define DHTTYPE DHT11 // DHT 11
#define SECRET_PINNUMBER ""
#define SECRET_GPRS_APN "internet" // replace your GPRS APN
#define SECRET_GPRS_LOGIN "" // replace with your GPRS login
#define SECRET_GPRS_PASSWORD "" // replace with your GPRS password
#define countof(a) (sizeof(a) / sizeof(a[0]))
// For SPI mode, we need a CS pin
#define LPS_CS 6

Adafruit_LPS22 lps;
sensors_event_t lps_temp;
sensors_event_t lps_press;
ThreeWire myWire(4,5,3); // IO, SCLK, CE
RtcDS1302<ThreeWire> mkrRtc(myWire);
float sensorVoltage;
int sensorValue;

// Please enter your sensitive data in the Secret tab or
// arduino_secrets.h
// PIN Number
const char PINNUMBER[] = SECRET_PINNUMBER;
// APN data
const char GPRS_APN[] = SECRET_GPRS_APN;
const char GPRS_LOGIN[] = SECRET_GPRS_LOGIN;
const char GPRS_PASSWORD[] = SECRET_GPRS_PASSWORD;

// initialize the library instance
GSM gsmAccess; // GSM access: include a 'true' parameter for
// debug enabled

GPRS gprs; // GPRS access

GSMClient client; // Client service for TCP connection
DHT dht(DHTPIN, DHTTYPE);
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
// timeout
const unsigned long __TIMEOUT__ = 10 * 1000;
char server[] = "weatherdatapms.ddns.net";
char path[] = "/data_prokat.php?";
int port = 8081; // port 80 is the default for HTTP
File dataLog;

void setup() {
// initialize serial communications and wait for port to open:
Serial.begin(9600);
```

```

// while (!Serial) {
//   ; // wait for serial port to connect. Needed for Leonardo only
// }

Serial.println("starting,..");
// connection state
bool connected = false;

// Start GSM shield
// If your SIM has PIN, pass it as a parameter of begin() in quotes
while (!connected) {
  if ((gsmAccess.begin(PINNUMBER) == GSM_READY) &&
      (gprs.attachGPRS(GPRS_APN, GPRS_LOGIN, GPRS_PASSWORD) ==
GPRS_READY)) {
    connected = true;
  } else {
    Serial.println("Not connected");
    delay(1000);
  }
}

Serial.println("Connected to GPRS network");
pinMode(16, OUTPUT);
pinMode(6, OUTPUT);
pinMode(A3, INPUT);
digitalWrite(16, HIGH); // disable sd cs
digitalWrite(6, HIGH); // disable lps cs
if (tsl.begin())
{
  Serial.println(F("Found a TSL2591 sensor"));
}
else
{
  Serial.println(F("No sensor found ... check your wiring?"));
  while (1);
}
tsl.setGain(TSL2591_GAIN_MED);
tsl.setTiming(TSL2591_INTEGRATIONTIME_300MS);
dht.begin();
digitalWrite(6, LOW); // disable lps cs
lps.begin_SPI(LPS_CS);
digitalWrite(6, HIGH); // disable lps cs
digitalWrite(16, LOW);
if ( !SD.begin(16) )
  Serial.println("failed!"); // initialization error
digitalWrite(16, HIGH);
mkrRtc.Begin();
if (mkrRtc.GetIsWriteProtected())
{
  Serial.println("RTC was write protected, enabling writing
now");
  mkrRtc.SetIsWriteProtected(false);
}

if (!mkrRtc.GetIsRunning())
{
  Serial.println("RTC was not actively running, starting now");
  mkrRtc.SetIsRunning(true);
}

RtcDateTime now = mkrRtc.GetDateTime();

```

```

}

void loop() {
    delay (2000);
    float humidity = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float temperature = dht.readTemperature();
    // Compute heat index in Celsius (isFahreheit = false)
    float heat_index = dht.computeHeatIndex(temperature,humidity,
false);
    digitalWrite(6, LOW); // disable lps cs
    lps.getEvent(&lps_press, &lps_temp); // get lps pressure & lps temp
    float ltemperature = lps_temp.temperature;
    float lpressure = lps_press.pressure;
    digitalWrite(6, HIGH); // disable lps cs
    uint32_t lum = tsl.getFullLuminosity();
    uint16_t ir, full;
    ir = lum >> 16;
    full = lum & 0xFFFF;
    float L= tsl.calculateLux(full, ir) ;
    sensorValue = analogRead(A3);
    sensorVoltage = (5.0/1023.0)*(sensorValue)*(1000);
    int uv=volts_to_uv(sensorVoltage);
    Serial.print(F(" Humidity: "));
    Serial.print(humidity);
    Serial.print(F("% Temperature: "));
    Serial.print(temperature);
    Serial.print(F("C "));
    Serial.print(F(" Heat index: "));
    Serial.print(heat_index);
    Serial.print(F("C "));
    Serial.print("Lux = ");
    Serial.println(L);
    Serial.print("Pressure = ");
    Serial.println(lpressure);
    Serial.print("Ltemp = ");
    Serial.println(ltemperature);
    Serial.print("UV-Index = ");
    Serial.println(uv);

    // initialize http service
    // client.print("AT+HTTPIPINIT");
    // delay(2000);
    //
    //
    // // set http param value // SOS CHECK THE VARIABLES SYNTAX
    // GRAMMAR AND DELAY
    //
    client.print("AT+HTTTPARA=\"URL\", \"http://localhost/ethernet_example
/data.php?");
    if (client.connect(server, port)) {
        Serial.println("connected");
        client.print("GET ");
        client.print(path);
        client.print("temperature=");
        client.print(temperature);
        client.print("&humidity=");
        client.print(humidity);
        client.print("&Lux=");
        client.print(L);
        client.print("&lps temperature=");

```

```

        client.print(ltemperature);
        client.print("&lps_pressure=");
        client.print(lpressure);
        client.print("&UV-Light=");
        client.println(uv);
    } else {
        // if you didn't get a connection to the server:
        Serial.println("connection failed");
    }

    RtcDateTime now = mkrRtc.GetDateTime();
    char datestring[22];
    snprintf_P(datestring,
                countof(datestring),
                PSTR("%02u/%02u/%04u    %02u:%02u:%02u"),
                now.Month(),
                now.Day(),
                now.Year(),
                now.Hour(),
                now.Minute(),
                now.Second());
    Serial.println(datestring);
    if (!now.IsValid())
    {
        // Common Causes:
        //    1) the battery on the device is low or even missing and
the power line was disconnected
        Serial.println("RTC lost confidence in the DateTime!");
    }

    digitalWrite(16, LOW); // ENABLE SD
    // open Log.txt file with write permission
    dataLog = SD.open("LogCN.txt", FILE_WRITE);
    dataLog.print(datestring);
    dataLog.print("    ");
    dataLog.print(temperature);
    dataLog.print(" °C    |    ");
    dataLog.print(humidity);
    dataLog.print("    |    ");
    dataLog.print(L);
    dataLog.print("    |    ");
    dataLog.print(lpressure);
    dataLog.print("    |    ");
    dataLog.println(ltemperature);
    dataLog.close();

    delay(100); // wait 100ms
    digitalWrite(16, HIGH); // DISABLE SD

    delay(2000);
}

int volts_to_uv(float sensorVoltage)
{
    int uv;
    if (sensorVoltage<50.0){
        uv=0;
    }
}

```

```

else if (sensorVoltage<=227.0){
    uv=1;
}
else if (sensorVoltage<=318.0){
    uv=2;
}
else if (sensorVoltage<=408.0){
    uv=3;
}
else if (sensorVoltage<=503.0){
    uv=4;
}
else if (sensorVoltage<=606.0){
    uv=5;
}
else if (sensorVoltage<=696.0){
    uv=6;
}
else if (sensorVoltage<=795.0){
    uv=7;
}
else if (sensorVoltage<=881.0){
    uv=8;
}
else if (sensorVoltage<=976.0){
    uv=9;
}
else if (sensorVoltage<=1079.0){
    uv=10;
}
else {
    uv=11;
}
return uv;
}

```

8.3 A.3 rfwithrtcdatalogger.ino

```

#include <SPI.h>
#include <RadioHead.h>
#include "RH_RF95.h"
#include "DHT.h"
#include <Wire.h>
#include <Adafruit_LPS2X.h>
#include <Adafruit_Sensor.h>
#include "Adafruit_TSL2591.h"
#include <SD.h> // include Arduino SD library
#include <ThreeWire.h>
#include <RtcDS1302.h>

#define LPS_CS 8
#define RFM95_CS 10 //CS if Lora connected to pin 10
#define RFM95_RST 9 //RST of Lora connected to pin 9

```

```

#define RFM95_INT 2 //INT of Lora connected to pin 2
#define RF95_FREQ 434.0 // Change to 434.0 or other frequency, must
match RX's freq!
#define countof(a) (sizeof(a) / sizeof(a[0]))

ThreeWire myWire(4,5,7); // IO, SCLK, CE
RtcDS1302<ThreeWire> Rtc(myWire);
RH_RF95 rf95(RFM95_CS, RFM95_INT);
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
DHT dht(3,DHT22);
Adafruit_LPS22 lps;
sensors_event_t lps_temp;
sensors_event_t lps_press;
String ide = "1.2";
File dataLog;
float sensorVoltage;
int sensorValue;
void setup()
{
  //   Serial.print("compiled: ");
  //   Serial.print(__DATE__);
  //   Serial.println(__TIME__);

  Rtc.Begin();
  //   Only the first time just to set the time
  //   RtcDateTime compiled = RtcDateTime(__DATE__, __TIME__);
  //   printDateTime(compiled);
  //   Serial.println();

  //   if (!Rtc.IsDateTimeValid())
  //   {
  //       // Common Causes:
  //       //   1) first time you ran and the device wasn't running
  yet
  //       //   2) the battery on the device is low or even missing
  //
  //       Serial.println("RTC lost confidence in the DateTime!");
  //       Rtc.SetDateTime(compiled);
  //   }

  if (Rtc.GetIsWriteProtected())
  {
    Serial.println("RTC was write protected, enabling writing
now");
    Rtc.SetIsWriteProtected(false);
  }

  if (!Rtc.GetIsRunning())
  {
    Serial.println("RTC was not actively running, starting now");
    Rtc.SetIsRunning(true);
  }

  RtcDateTime now = Rtc.GetDateTime();
  pinMode(8, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(A3, INPUT);
  digitalWrite(10, HIGH); // disable ethernet cs
  digitalWrite(6, HIGH);
  digitalWrite(8, LOW); // enable lps cs

```

```

lps.begin_SPI(LPS_CS);
digitalWrite(8, HIGH); // disable lps cs
Serial.begin(9600);
dht.begin();
tsl.begin();
// if (tsl.begin())
// {
//   Serial.println(F("Found a TSL2591 sensor"));
// }
// else
// {
//   Serial.println(F("No sensor found ... check your wiring?"));
//   while (1);
// }
tsl.setGain(TSL2591_GAIN_MED);
tsl.setTiming(TSL2591_INTEGRATIONTIME_300MS);
// manual reset
digitalWrite(10, LOW); // enable rf
digitalWrite(RFM95_RST, LOW);
pinMode(RFM95_RST, OUTPUT);

delayMicroseconds(100);
pinMode(RFM95_RST, INPUT);
delay(20);

while (!rf95.init()) {
  Serial.println("LoRas radio init failed");
  delay(1000);
}
//rf95.init();
//
//rf95.setFrequency(RF95_FREQ);
//Set the default frequency 434.0MHz
if (!rf95.setFrequency(RF95_FREQ)) {
  Serial.println("setFrequency failed");
  while (1);
}
digitalWrite(10, HIGH); // disable rf
digitalWrite(6, LOW);
if (!SD.begin(6))
  Serial.println("failed!"); // initialization error
digitalWrite(6, HIGH);
}

void loop()
{
  String data= "";
  Serial.print(F(" data: "));
  Serial.println(data);
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);
  digitalWrite(8, LOW); // enable lps cs
  lps.getEvent(&lps_press, &lps_temp); // get lps pressure & lps temp
  float ltemperature = lps_temp.temperature;
  float lpressure = lps_press.pressure;
  digitalWrite(8, HIGH); // disable lps cs

```



```

uint32_t lum = tsl.getFullLuminosity();
uint16_t ir, full;
ir = lum >> 16;
full = lum & 0xFFFF;
float L= tsl.calculateLux(full, ir);
sensorValue = analogRead(A3);
sensorVoltage = (5.0/1023.0)*(sensorValue)*(1000);
int uv=volts_to_uv(sensorVoltage);
//Serial.print(F("Lux: ")); Serial.println(L);
String humidity = String(h,2); //int to String
String sh= String(humidity.length());
String temperature = String(t,2);
String st= String(temperature.length());
String heatid= String(hic,2);
String Lux=String (L,2);
String sl= String(Lux.length());
String lps_press=String(lpressure,2);
String slp= String(lps_press.length());
String lps_temp=String(ltemperature,2);
String slt= String(lps_temp.length());
String uvi=String(uv,2);
String su= String(uvi.length());
Serial.print(F(" ID: "));
Serial.println(ide);
Serial.print(F(" Humidity: "));
Serial.print(h);
Serial.print(F(" Sh: "));
Serial.println(sh);
Serial.print(F("% Temperature: "));
Serial.print(t);
Serial.print(F(" St: "));
Serial.println(st);
Serial.print(F("C "));
Serial.print(F("F Heat index: "));
Serial.print(hic);
Serial.print(F("F Lux: "));
Serial.println(Lux);
Serial.print(F(" Sl: "));
Serial.println(sl);
Serial.print("Pressure = ");
Serial.print(lpressure);
Serial.print(F(" Slp: "));
Serial.println(slp);
Serial.print("Ltemp = ");
Serial.print(ltemperature);
Serial.print(F(" Slt: "));
Serial.println(slt);
Serial.print("UV-Index = ");
Serial.print(uv);
Serial.print(F(" Su: "));
Serial.println(su);

data =ide;
data.concat(st);
data.concat(sh);
data.concat(sl);
data.concat(slp);
data.concat(slt);
data.concat(su);
data.concat(temperature);
data.concat(humidity);

```

```

data.concat(Lux);
data.concat(lps_press);
data.concat(lps_temp);
data.concat(uvi);
Serial.println(data);
int dat= data.length();
Serial.println(dat);
char d[dat+1];
data.toCharArray(d,dat+1); //String to char array
data= "";
Serial.println((d));
Serial.println(strlen(d));
Serial.println(F("Sending to rf95_server"));
digitalWrite(10, LOW);
rf95.send((uint8_t *)d, strlen(d));
rf95.waitPacketSent();
digitalWrite(10, HIGH);
RtcDateTime now = Rtc.GetDateTime();
char datestring[22];
snprintf_P(datestring,
            countof(datestring),
            PSTR("%02u/%02u/%04u    %02u:%02u:%02u"),
            now.Month(),
            now.Day(),
            now.Year(),
            now.Hour(),
            now.Minute(),
            now.Second());
Serial.println(datestring);
if (!now.IsValid())
{
    // Common Causes:
    //    1) the battery on the device is low or even missing and
the power line was disconnected
    Serial.println("RTC lost confidence in the DateTime!");
}

digitalWrite(6, LOW); // ENABLE SD
// open Log.txt file with write permission
dataLog = SD.open("LogRF.txt", FILE_WRITE);
dataLog.print(datestring);
dataLog.print("    ");
dataLog.print(t);
dataLog.print(" °C    |    ");
dataLog.print(h);
dataLog.print(" | ");
dataLog.print(L);
dataLog.print(" | ");
dataLog.print(lpressure);
dataLog.print(" | ");
dataLog.println(ltemperature);
dataLog.close();

delay(100); // wait 100ms
digitalWrite(6, HIGH); // DISABLE SD

delay(4000);
}

int volts_to_uv(float sensorVoltage)
{

```

```
int uv;
if (sensorVoltage<50.0){
    uv=0;
}
else if (sensorVoltage<=227.0){
    uv=1;
}
else if (sensorVoltage<=318.0){
    uv=2;
}
else if (sensorVoltage<=408.0){
    uv=3;
}
else if (sensorVoltage<=503.0){
    uv=4;
}
else if (sensorVoltage<=606.0){
    uv=5;
}
else if (sensorVoltage<=696.0){
    uv=6;
}
else if (sensorVoltage<=795.0){
    uv=7;
}
else if (sensorVoltage<=881.0){
    uv=8;
}
else if (sensorVoltage<=976.0){
    uv=9;
}
else if (sensorVoltage<=1079.0){
    uv=10;
}
else {
    uv=11;
}
return uv;
}
```

9 ΠΑΡΑΡΤΗΜΑ Β Κώδικες Διακομιστή

9.1 B.1 rfp.py

```
from time import sleep
from SX127x.LoRa import *
from SX127x.board_config import BOARD
import urllib
import requests

BOARD.setup()
class LoRaRcvCont(LoRa):
    def __init__(self, verbose=False):
        super(LoRaRcvCont, self).__init__(verbose)
        self.set_mode(MODE.SLEEP)
        self.set_dio_mapping([0] * 6)
    def start(self):
        self.reset_ptr_rx()
        self.set_mode(MODE.RXCONT)
        while True:
            sleep(.5)
            rssi_value = self.get_rssi_value()
            status = self.get_modem_status()
            sys.stdout.flush()
    def on_rx_done(self):
        print ("\nReceived: ")
        self.clear_irq_flags(RxDone=1)
        payload = self.read_payload(nocheck=True)
        print (bytes(payload).decode("utf-8", 'ignore'))
        data = bytes(payload).decode("utf-8", 'ignore')
        print(data)
        ide= (data[2:5])
        st=int(data[5])
        sh=int(data[6])
        sl=int(data[7])
        slp=int(data[8])
        slt=int(data[9])
        su=int(data[10])
        temp = float ((data[11:(11+st)]))
        humidity = float((data[(11+st):(11+st+sh)]))
        Lux =float ((data[(11+st+sh):(11+st+sh+sl)]))
        Lpress=float((data[(11+st+sh+sl):(11+st+sh+sl+slp)]))
        Ltemp=float((data[(11+st+sh+sl+slp):(11+st+sh+sl+slp+slt)]))

        Uv=int((data[(11+st+sh+sl+slp+slt):(11+st+sh+sl+slp+slt+su)]))
        print("ID:")
        print(ide)
        print("st:")
        print(st)
        print("sh:")
        print(sh)
        print("sl:")
```

```

        print(slp)
        print("slp:")
        print(slp)
        print("slt:")
        print(slt)
        print("su:")
        print(su)
        print("Temperature:")
        print(temp)
        print("Humidity:")
        print(humidity)
        print("Lux:")
        print(Lux)
        print("LPressure:")
        print(Lpress)
        print("LTemperature:")
        print(Ltemp)
        print("UV-Index:")
        print(Uv)
        if (ide == "1.1"):
            userdata={"temperature" : temp,"humidity":
humidity,"Lux":Lux,"lps_temperature":Ltemp,"lps_pressure":Lpress,"UV-
Light":Uv }
            resp =
requests.post('http://localhost/data_esties.php',params=userdata)
            if (ide == "1.2"):
                userdata={"temperature" : temp,"humidity": humidity,
"Lux": Lux,"lps_temperature":Ltemp,"lps_pressure":Lpress,"UV-
Light":Uv }
                resp =
requests.post('http://localhost/data_athina.php',params=userdata)
                self.set_mode(MODE.SLEEP)
                self.reset_ptr_rx()
                self.set_mode(MODE.RXCONT)
lora = LoRaRcvCont(verbose=False)
lora.set_mode(MODE.STDBY)
# Medium Range Defaults after init are 434.0MHz, Bw = 125 kHz, Cr =
4/5, Sf = 128chips/symbol, CRC on 13 dBm
lora.set_pa_config(pa_select=1)
try:
    lora.start()
except KeyboardInterrupt:
    sys.stdout.flush()
    print("")
    sys.stderr.write("KeyboardInterrupt\n")
finally:
    sys.stdout.flush()
    print("")
    lora.set_mode(MODE.SLEEP)
    BOARD.teardown()

```

9.2 B.2 ran-fan.py

```

#!/usr/bin/env python3
# Author: Edoardo Paolo Scalafiotti <edoardo849@gmail.com>
import os
from time import sleep
import signal

```

```

import sys
import RPi.GPIO as GPIO
pin = 23 # The pin ID, edit here to change it
maxTMP = 39 # The maximum temperature in Celsius after which we trigger the fan
def setup():
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(pin, GPIO.OUT)
    GPIO.setwarnings(False)
    return()
def getCPUtemperature():
    res = os.popen('vcgencmd measure_temp').readline()
    temp =(res.replace("temp=", "").replace("'C\n", ""))
    # print("temp is {}".format(temp)) #Uncomment here for testing
    return temp
def fanON():
    setPin(True)
    return()
def fanOFF():
    setPin(False)
    return()
def getTEMP():
    CPU_temp = float(getCPUtemperature())
    if CPU_temp>maxTMP:
        fanON()
    else:
        fanOFF()
    return()
def setPin(mode): # A little redundant function but useful if you want to add logging
    GPIO.output(pin, mode)
    return()
try:
    setup()
    while True:
        getTEMP()
        sleep(5) # Read the temperature every 5 sec, increase or decrease this limit if you want
except KeyboardInterrupt: # trap a CTRL+C keyboard interrupt
    GPIO.cleanup() # resets all GPIO ports used by this program

```

9.3 B.3 index.php

```

<!DOCTYPE html>
<?php
$url=$_SERVER['REQUEST_URI'];
?>
<html lang="en">

<head>
    <link rel="stylesheet" href="styles.css">
</head>

<title>Weather Network</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"

```

```

href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Montserrat">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

<body>

<!-- Navbar -->
<ul>
  <li><a href="index1.php">Αρχική</a></li>
  <li><a href="hmmmy.php">Σταθμός HMMY </a></li>
  <li><a href="ekathens.php">Σταθμός Ε.Κ ΑΘΗΝΑ</a></li>
  <li><a href="esties.php">Σταθμός Εστιών</a></li>
  <li><a href="prokat.php">Σταθμός ΠΡΟΚΑΤ</a></li>
  <li><a href="zampretti.php">Πρόβλεψη Zampretti</a></li>
  <li><a href="meteo.php">Μετεωρογράμμο</a></li>
</ul>

<!-- Header -->
<header class="header1" >Δίκτυο μετεωρολογικών σταθμών Εάνθης
</header>
<p class = "subheader1"> Θέση: 41.13488 N, 41.13488 24.888 E -
Υψόμετρο:76 μ </p>
<p class = "subheader2"> Provided and supervised by:Maroulakos-
Seferiadis Paris </p>
<br> <br> <br>

<div class = "wrapper1">
  <!--<div>
    <iframe scrolling="no" frameborder="0" allowtransparency="true"
src="https://gr.k24.net/widgets/weather_w3.aspx?p=48" class=
"centerwid" ></iframe>
    <a target="blank" style="color: #999999; width: 300px; display:
block; text-align: center; font: 10px/10px Arial,san-serif; text-
decoration: none;" href="https://gr.k24.net"> πρόγνωση καιρού από το
k24.net</a>
  </div-->
  <div id="cont_d1e05f0f1222b704bcb309e0b5a569c3"><script
type="text/javascript" async
src="https://www.theweather.com/wid_loader/d1e05f0f1222b704bcb309e0b5
a569c3"></script></div>
  <p class = "subheader4"> Πρόβλεψη Καιρού </p>
  <p class = "subheader4"> Πρόβλεψη Καιρού </p>

  <div class="clock">
    <div class="hour">
      <div class="hr" id="hr">
      </div>
    </div>
    <div class="min">
      <div class="mn" id="mn">
      </div>
    </div>
    <div class="sec">
      <div class="sc" id="sc">
      </div>
    </div>
  </div>

```

```

        </div>
    </div>

    <script src="JS/main.js"></script>
</div>

<div id="SG"> </div>

<br><br><br><br>

</div>

<div id="TG"> </div>

<div id=wrapper-bg>
    <div class="w3-container w3-center">
        <p class="subheader3">Live Camera Εόvθη, Clock Tower</p>
        <h1 class="w3-margin w3-xlarge"></h1>
        </div>
    </div>
<script>
    // Used to toggle the menu on small screens when clicking on the
    menu button
    function myFunction() {
        var x = document.getElementById("navDemo");
        if (x.className.indexOf("w3-show") == -1) {
            x.className += " w3-show";
        } else {
            x.className = x.className.replace(" w3-show", "");
        }
    }
</script>

</script>
</body>
</html>

```

9.4 B.4 hmmy.php

```

<!DOCTYPE html>
<?php
$url=$_SERVER['REQUEST_URI'];
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph.php');
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph_line.php');

include('connection.php');

$result = mysqli_query($con,'SELECT DHT_temperature FROM hmmy ORDER
BY id DESC LIMIT 5');
// Process every record
$odddrow = true;

```



```

$count = 0;
while($row = mysqli_fetch_array($result)){

    $datay1[$count] = $row['DHT_temperature'];
    $count = $count + 1;
}

$result = mysqli_query($con,'SELECT LPS_temperature FROM hmmy ORDER
BY id DESC LIMIT 5');
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay4[$count] = $row['LPS_temperature'];
    $count = $count + 1;
}

// Setup the graph
$graph = new Graph(900,300);
$graph->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph->SetTheme($theme_class);
$graph->img->SetAntiAliasing(false);
$graph->title->Set('Temperature Graph');
$graph->SetBox(false);

$graph->SetMargin(40,20,36,63);

$graph->img->SetAntiAliasing();

$graph->yaxis->HideZeroLabel();
$graph->yaxis->HideLine(false);
$graph->yaxis->HideTicks(false,false);

$graph->xgrid->Show();
$graph->xgrid->SetLineStyle("solid");
$graph->xaxis->SetTickLabels(array('A','B','C','D'));
$graph->xgrid->SetColor('#E3E3E3');

// Create the first line
$p1 = new LinePlot($datay1);
$graph->Add($p1);
$p1->SetColor("#000000");
$p1->SetLegend('Temperature');

$p4 = new LinePlot($datay4);
$graph->Add($p4);
$p4->SetColor("#FF0000");
$p4->SetLegend('LPS Temperature');

$graph->legend->SetFrameWeight(1);

// Output line
$graph->Stroke('graph_hmmy.jpg');

$result = mysqli_query($con,'SELECT DHT_humidity FROM hmmy ORDER BY
id DESC LIMIT 5');
// Process every record

```

```

$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay2[$count] = $row['DHT_humidity'];
    $count = $count + 1;
}

// Setup the graph
$graph1 = new Graph(900,300);
$graph1->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph1->SetTheme($theme_class);
$graph1->img->SetAntiAliasing(false);
$graph1->title->Set('Humidity Graph');
$graph1->SetBox(false);

$graph1->SetMargin(40,20,36,63);

$graph1->img->SetAntiAliasing();

$graph1->yaxis->HideZeroLabel();
$graph1->yaxis->HideLine(false);
$graph1->yaxis->HideTicks(false,false);

$graph1->xgrid->Show();
$graph1->xgrid->SetLineStyle("solid");
$graph1->xaxis->SetTickLabels(array('A','B','C','D'));
$graph1->xgrid->SetColor('#E3E3E3');

// Create the first line
$p2 = new LinePlot($datay2);
$graph1->Add($p2);
$p2->SetColor("##000000");
$p2->SetLegend('Humidity');

$graph1->legend->SetFrameWeight(1);

// Output line
$graph1->Stroke('graph1_hmmy.jpg');

$result = mysqli_query($con,"SELECT Brightness FROM hmmy ORDER BY id
DESC LIMIT 5");
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay3[$count] = $row['Brightness'];
    $count = $count + 1;
}

// Setup the graph
$graph2 = new Graph(900,300);
$graph2->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph2->SetTheme($theme_class);
$graph2->img->SetAntiAliasing(false);

```

```

$graph2->title->Set('Brightness Graph');
$graph2->SetBox(false);

$graph2->SetMargin(40,20,36,63);

$graph2->img->SetAntiAliasing();

$graph2->yaxis->HideZeroLabel();
$graph2->yaxis->HideLine(false);
$graph2->yaxis->HideTicks(false,false);

$graph2->xgrid->Show();
$graph2->xgrid->SetLineStyle("solid");
$graph2->xaxis->SetTickLabels(array('A','B','C','D'));
$graph2->xgrid->SetColor('#E3E3E3');

// Create the first line
$p3 = new LinePlot($datay3);
$graph2->Add($p3);
$p3->SetColor("#000000");
$p3->SetLegend('Brightness');

$graph2->legend->SetFrameWeight(1);

// Output line
$graph2->Stroke('graph2_hmmy.jpg');

$result = mysqli_query($con,'SELECT LPS_pressure FROM hmmy ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay5[$count] = $row['LPS_pressure'];
    $count = $count + 1;
}

// Setup the graph
$graph3 = new Graph(900,300);
$graph3->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph3->SetTheme($theme_class);
$graph3->img->SetAntiAliasing(false);
$graph3->title->Set('Pressure Graph');
$graph3->SetBox(false);

$graph3->SetMargin(40,20,36,63);

$graph3->img->SetAntiAliasing();

$graph3->yaxis->HideZeroLabel();
$graph3->yaxis->HideLine(false);
$graph3->yaxis->HideTicks(false,false);

$graph3->xgrid->Show();
$graph3->xgrid->SetLineStyle("solid");
$graph3->xaxis->SetTickLabels(array('A','B','C','D'));
$graph3->xgrid->SetColor('#E3E3E3');

```

```

// Create the first line
$p5 = new LinePlot($datay5);
$graph3->Add($p5);
$p5->SetColor("##000000");
$p5->SetLegend('LPS_pressure');

$graph3->legend->SetFrameWeight(1);

// Output line
$graph3->Stroke('graph3_hmmy.jpg');

$result = mysqli_query($con,'SELECT UV_Light FROM hmmy ORDER BY id
DESC LIMIT 5');
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay6[$count] = $row['UV_Light'];
    $count = $count + 1;
}

// Setup the graph
$graph4 = new Graph(900,300);
$graph4->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph4->SetTheme($theme_class);
$graph4->img->SetAntiAliasing(false);
$graph4->title->Set('UV-Index Graph');
$graph4->SetBox(false);

$graph4->SetMargin(40,20,36,63);

$graph4->img->SetAntiAliasing();

$graph4->yaxis->HideZeroLabel();
$graph4->yaxis->HideLine(false);
$graph4->yaxis->HideTicks(false,false);

$graph4->xgrid->Show();
$graph4->xgrid->SetLineStyle("solid");
$graph4->xaxis->SetTickLabels(array('A','B','C','D'));
$graph4->xgrid->SetColor('#E3E3E3');

// Create the first line
$p6 = new LinePlot($datay6);
$graph4->Add($p6);
$p6->SetColor("##000000");
$p6->SetLegend('UV_Light');

$graph4->legend->SetFrameWeight(1);

// Output line
$graph4->Stroke('graph4_hmmy.jpg');
?>
<html lang="en">

<head>
    <link rel="stylesheet" href="styles.css">

```

```

</head>

<title>Weather Network</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Montserrat">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

<body>

<!-- Navbar -->
<ul>
  <li><a href="index1.php">Αρχική</a></li>
  <li><a href="hmmy.php">Σταθμός HMMY </a></li>
  <li><a href="ekathens.php">Σταθμός Ε.Κ ΑΘΗΝΑ</a></li>
  <li><a href="esties.php">Σταθμός Εστιών</a></li>
  <li><a href="prokat.php">Σταθμός ΠΡΟΚΑΤ</a></li>
  <li><a href="zampretti.php">Πρόβλεψη Zampretti</a></li>
  <li><a href="meteo.php">Μετεωρόγραμμα</a></li>
</ul>

<!-- Header -->
<header class="header1" >Δίκτυο μετεωρολογικών σταθμών Εάνθης
</header>
<p class = "subheader1"> Θέση: 41.13488 N, 41.13488 24.888 E -
Υψόμετρο:76 μ </p>
<p class = "subheader2"> Provided and supervised by:Maroulakos-
Seferiadis Paris </p>
<br> <br> <br>

<div><p class = "header1">Οι τιμές των πινάκων θα ανανεωθούν σε <span
id="cnt" style="color:red;">120</span> δευτερόλεπτα</p></div>
<script>
  var counter = 120;

  // The countdown method.
  window.setInterval(function () {
    counter--;
    if (counter >= 0) {
      var span;
      span = document.getElementById("cnt");
      span.innerHTML = counter;
    }
    if (counter === 0) {
      clearInterval(counter);
    }

  }, 1000);

```

9.5 B.5 ekathens.php

```
<!DOCTYPE html>
<?php
$url=$_SERVER['REQUEST_URI'];
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph.php');
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph_line.php');

include('connection.php');

$result = mysqli_query($con,'SELECT DHT_temperature FROM athina ORDER
BY id DESC LIMIT 5');
// Process every record
$odddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){

    $datay1[$count] = $row['DHT_temperature'];
    $count = $count + 1;
}

$result = mysqli_query($con,'SELECT LPS_temperature FROM athina ORDER
BY id DESC LIMIT 5');
// Process every record
$odddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay4[$count] = $row['LPS_temperature'];
    $count = $count + 1;
}

// Setup the graph
$graph = new Graph(900,300);
$graph->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph->SetTheme($theme_class);
$graph->img->SetAntiAliasing(false);
$graph->title->Set('Temperature Graph');
$graph->SetBox(false);

$graph->SetMargin(40,20,36,63);

$graph->img->SetAntiAliasing();

$graph->yaxis->HideZeroLabel();
$graph->yaxis->HideLine(false);
$graph->yaxis->HideTicks(false,false);

$graph->xgrid->Show();
$graph->xgrid->SetLineStyle("solid");
$graph->xaxis->SetTickLabels(array('A','B','C','D'));
$graph->xgrid->SetColor('#E3E3E3');

// Create the first line
$p1 = new LinePlot($datay1);
$graph->Add($p1);
$p1->SetColor("#000000");
$p1->SetLegend('Temperature');
```

```

$p4 = new LinePlot($datay4);
$graph->Add($p4);
$p4->SetColor("#FF0000");
$p4->SetLegend('LPS Temperature');

$graph->legend->SetFrameWeight(1);

// Output line
$graph->Stroke('graph_hmmy.jpg');

$result = mysqli_query($con,'SELECT DHT_humidity FROM athina ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay2[$count] = $row['DHT_humidity'];
    $count = $count + 1;
}

// Setup the graph
$graph1 = new Graph(900,300);
$graph1->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph1->SetTheme($theme_class);
$graph1->img->SetAntiAliasing(false);
$graph1->title->Set('Humidity Graph');
$graph1->SetBox(false);

$graph1->SetMargin(40,20,36,63);

$graph1->img->SetAntiAliasing();

$graph1->yaxis->HideZeroLabel();
$graph1->yaxis->HideLine(false);
$graph1->yaxis->HideTicks(false,false);

$graph1->xgrid->Show();
$graph1->xgrid->SetLineStyle("solid");
$graph1->xaxis->SetTickLabels(array('A','B','C','D'));
$graph1->xgrid->SetColor('#E3E3E3');

// Create the first line
$p2 = new LinePlot($datay2);
$graph1->Add($p2);
$p2->SetColor("##000000");
$p2->SetLegend('Humidity');

$graph1->legend->SetFrameWeight(1);

// Output line
$graph1->Stroke('graph1_hmmy.jpg');

$result = mysqli_query($con,'SELECT Brightness FROM athina ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrow = true;

```

```

$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay3[$count] = $row['Brightness'];
    $count = $count + 1;
}

// Setup the graph
$graph2 = new Graph(900,300);
$graph2->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph2->SetTheme($theme_class);
$graph2->img->SetAntiAliasing(false);
$graph2->title->Set('Brightness Graph');
$graph2->SetBox(false);

$graph2->SetMargin(40,20,36,63);

$graph2->img->SetAntiAliasing();

$graph2->yaxis->HideZeroLabel();
$graph2->yaxis->HideLine(false);
$graph2->yaxis->HideTicks(false,false);

$graph2->xgrid->Show();
$graph2->xgrid->SetLineStyle("solid");
$graph2->xaxis->SetTickLabels(array('A','B','C','D'));
$graph2->xgrid->SetColor('#E3E3E3');

// Create the first line
$p3 = new LinePlot($datay3);
$graph2->Add($p3);
$p3->SetColor("#000000");
$p3->SetLegend('Brightness');

$graph2->legend->SetFrameWeight(1);

// Output line
$graph2->Stroke('graph2_hmmy.jpg');

$result = mysqli_query($con,'SELECT LPS_pressure FROM athina ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay5[$count] = $row['LPS_pressure'];
    $count = $count + 1;
}

// Setup the graph
$graph3 = new Graph(900,300);
$graph3->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph3->SetTheme($theme_class);
$graph3->img->SetAntiAliasing(false);
$graph3->title->Set('Pressure Graph');

```



```

$graph3->SetBox(false);

$graph3->SetMargin(40,20,36,63);

$graph3->img->SetAntiAliasing();

$graph3->yaxis->HideZeroLabel();
$graph3->yaxis->HideLine(false);
$graph3->yaxis->HideTicks(false,false);

$graph3->xgrid->Show();
$graph3->xgrid->SetLineStyle("solid");
$graph3->xaxis->SetTickLabels(array('A','B','C','D'));
$graph3->xgrid->SetColor('#E3E3E3');

// Create the first line
$p5 = new LinePlot($datay5);
$graph3->Add($p5);
$p5->SetColor("##000000");
$p5->SetLegend('LPS_pressure');

$graph3->legend->SetFrameWeight(1);

// Output line
$graph3->Stroke('graph3_hmmy.jpg');

$result = mysqli_query($con,'SELECT UV_Light FROM athina ORDER BY id
DESC LIMIT 5');
// Process every record
$oaddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay6[$count] = $row['UV_Light'];
    $count = $count + 1;
}

// Setup the graph
$graph4 = new Graph(900,300);
$graph4->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph4->SetTheme($theme_class);
$graph4->img->SetAntiAliasing(false);
$graph4->title->Set('UV-Index Graph');
$graph4->SetBox(false);

$graph4->SetMargin(40,20,36,63);

$graph4->img->SetAntiAliasing();

$graph4->yaxis->HideZeroLabel();
$graph4->yaxis->HideLine(false);
$graph4->yaxis->HideTicks(false,false);

$graph4->xgrid->Show();
$graph4->xgrid->SetLineStyle("solid");
$graph4->xaxis->SetTickLabels(array('A','B','C','D'));
$graph4->xgrid->SetColor('#E3E3E3');

// Create the first line

```

```

$p6 = new LinePlot($datay6);
$graph4->Add($p6);
$p6->SetColor("##000000");
$p6->SetLegend('UV_Light');

$graph4->legend->SetFrameWeight(1);

// Output line
$graph4->Stroke('graph4_hmmy.jpg');
?>
<html lang="en">

<head>
  <link rel="stylesheet" href="styles.css">
</head>

<title>Weather Network</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Montserrat">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

<body>

<!-- Navbar -->
<ul>
  <li><a href="index1.php">Αρχική</a></li>
  <li><a href="hmmy.php">Σταθμός HMMY </a></li>
  <li><a href="ekathens.php">Σταθμός Ε.Κ ΑΘΗΝΑ</a></li>
  <li><a href="esties.php">Σταθμός Εστιών</a></li>
  <li><a href="prokat.php">Σταθμός ΠΡΟΚΑΤ</a></li>
  <li><a href="zampretti.php">Πρόβλεψη Zampretti</a></li>
  <li><a href="meteo.php">Μετεωρόγραμμα</a></li>
</ul>

<!-- Header -->
<header class="header1" >Δίκτυο μετεωρολογικών σταθμών Εάνθης
</header>
<p class = "subheader1"> Θέση: 41.13488 N, 41.13488 24.888 E -
Υψόμετρο:76 μ </p>
<p class = "subheader2"> Provided and supervised by:Maroulakos-
Seferiadis Paris </p>
<br> <br> <br>

<div><p class = "header1">Οι τιμές των πινάκων θα ανανεωθούν σε <span
id="cnt" style="color:red;">120</span> δευτερόλεπτα</p></div>
<script>
  var counter = 120;

  // The countdown method.
  window.setInterval(function () {
    counter--;
    if (counter >= 0) {

```

```

        var span;
        span = document.getElementById("cnt");
        span.innerHTML = counter;
    }
    if (counter === 0) {
        clearInterval(counter);
    }

}, 1000);

window.setInterval('refresh()', 30000);

// Refresh or reload page.
function refresh() {
    window .location.reload();
}
</script>

<table border="5" cellspacing="0" cellpadding="7" class ="center">
    <caption> Σταθμός Ε.Κ ΑΘΗΝΑ </caption>
    <tr>
        <td class="table_titles">Ημερομηνία και Ωρα</td>
        <td class="table_titles">Θερμοκρασία</td>
        <td class="table_titles">Υγρασία</td>
        <td class="table_titles">Φωτεινότητα</td>
        <td class="table_titles">Θερμοκρασία (LPS)</td>
        <td class="table_titles">Ατμ.Πίεση</td>
        <td class="table_titles">Δείκτης UV</td>
    </tr>
    <?php
        include('connection.php');
        $result = mysqli_query($con, 'SELECT * FROM athina ORDER BY id
DESC LIMIT 5');
        // Process every record
        $oddrow = true;

        while($row = mysqli_fetch_array($result))
        {
            if ($oddrow)
            {
                $css_class=' class="table_cells_odd"';
            }
            else
            {
                $css_class=' class="table_cells_even"';
            }
            $oddrow = !$oddrow;
            echo "<tr align=center style= 'background:white;'>";
            echo "<td '.$css_class.'>" . $row['event'] . "</td>";
            echo "<td '.$css_class.'>" . $row['DHT_temperature'] .
            "°C</td>" ;
            echo "<td '.$css_class.'>" . $row['DHT_humidity'] .
            "%</td>";
            echo "<td '.$css_class.'>" . $row['Brightness'] . "
Lux</td>";
            echo "<td '.$css_class.'>" . $row['LPS_temperature'] .
            "°C</td>" ;
            echo "<td '.$css_class.'>" . $row['LPS_Pressure'] .
            "hPa</td>";
            echo "<td '.$css_class.'>" . $row['UV_Light'] . "</td>";
            echo "</tr>";

```

```

    }

    // Close the connection
    mysqli_close($con);
    ?>
</table>
<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>
<br> <br> <br> <br>

<div class = "wrapper1">
    
</div>
<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

```

9.6 B.6 esties.php

```

<!DOCTYPE html>
<?php
$url=$_SERVER['REQUEST_URI'];
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph.php');
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph_line.php');

include('connection.php');

$result = mysqli_query($con,'SELECT DHT_temperature FROM esties ORDER
BY id DESC LIMIT 5');
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){

    $datay1[$count] = $row['DHT_temperature'];
    $count = $count + 1;
}

$result = mysqli_query($con,'SELECT LPS_temperature FROM esties ORDER
BY id DESC LIMIT 5');
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay4[$count] = $row['LPS_temperature'];
    $count = $count + 1;
}

```

```

}

// Setup the graph
$graph = new Graph(900,300);
$graph->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph->SetTheme($theme_class);
$graph->img->SetAntiAliasing(false);
$graph->title->Set('Temperature Graph');
$graph->SetBox(false);

$graph->SetMargin(40,20,36,63);

$graph->img->SetAntiAliasing();

$graph->yaxis->HideZeroLabel();
$graph->yaxis->HideLine(false);
$graph->yaxis->HideTicks(false,false);

$graph->xgrid->Show();
$graph->xgrid->SetLineStyle("solid");
$graph->xaxis->SetTickLabels(array('A','B','C','D'));
$graph->xgrid->SetColor('#E3E3E3');

// Create the first line
$p1 = new LinePlot($datay1);
$graph->Add($p1);
$p1->SetColor("#000000");
$p1->SetLegend('Temperature');

$p4 = new LinePlot($datay4);
$graph->Add($p4);
$p4->SetColor("#FF0000");
$p4->SetLegend('LPS Temperature');

$graph->legend->SetFrameWeight(1);

// Output line
$graph->Stroke('graph_hmmy.jpg');

$result = mysqli_query($con,'SELECT DHT_humidity FROM esties ORDER BY
id DESC LIMIT 5');
// Process every record
$odddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay2[$count] = $row['DHT_humidity'];
    $count = $count + 1;
}

// Setup the graph
$graph1 = new Graph(900,300);
$graph1->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph1->SetTheme($theme_class);
$graph1->img->SetAntiAliasing(false);
$graph1->title->Set('Humidity Graph');

```

```

$graph1->SetBox(false);

$graph1->SetMargin(40,20,36,63);

$graph1->img->SetAntiAliasing();

$graph1->yaxis->HideZeroLabel();
$graph1->yaxis->HideLine(false);
$graph1->yaxis->HideTicks(false,false);

$graph1->xgrid->Show();
$graph1->xgrid->SetLineStyle("solid");
$graph1->xaxis->SetTickLabels(array('A','B','C','D'));
$graph1->xgrid->SetColor('#E3E3E3');

// Create the first line
$p2 = new LinePlot($datay2);
$graph1->Add($p2);
$p2->SetColor("##000000");
$p2->SetLegend('Humidity');

$graph1->legend->SetFrameWeight(1);

// Output line
$graph1->Stroke('graph1_hmmy.jpg');

$result = mysqli_query($con,'SELECT Brightness FROM esties ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay3[$count] = $row['Brightness'];
    $count = $count + 1;
}

// Setup the graph
$graph2 = new Graph(900,300);
$graph2->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph2->SetTheme($theme_class);
$graph2->img->SetAntiAliasing(false);
$graph2->title->Set('Brightness Graph');
$graph2->SetBox(false);

$graph2->SetMargin(40,20,36,63);

$graph2->img->SetAntiAliasing();

$graph2->yaxis->HideZeroLabel();
$graph2->yaxis->HideLine(false);
$graph2->yaxis->HideTicks(false,false);

$graph2->xgrid->Show();
$graph2->xgrid->SetLineStyle("solid");
$graph2->xaxis->SetTickLabels(array('A','B','C','D'));
$graph2->xgrid->SetColor('#E3E3E3');

```

```

// Create the first line
$p3 = new LinePlot($datay3);
$graph2->Add($p3);
$p3->SetColor("#000000");
$p3->SetLegend('Brightness');

$graph2->legend->SetFrameWeight(1);

// Output line
$graph2->Stroke('graph2_hmmy.jpg');

$result = mysqli_query($con, 'SELECT LPS_pressure FROM esties ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay5[$count] = $row['LPS_pressure'];
    $count = $count + 1;
}

// Setup the graph
$graph3 = new Graph(900,300);
$graph3->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph3->SetTheme($theme_class);
$graph3->img->SetAntiAliasing(false);
$graph3->title->Set('Pressure Graph');
$graph3->SetBox(false);

$graph3->SetMargin(40,20,36,63);

$graph3->img->SetAntiAliasing();

$graph3->yaxis->HideZeroLabel();
$graph3->yaxis->HideLine(false);
$graph3->yaxis->HideTicks(false,false);

$graph3->xgrid->Show();
$graph3->xgrid->SetLineStyle("solid");
$graph3->xaxis->SetTickLabels(array('A','B','C','D'));
$graph3->xgrid->SetColor('#E3E3E3');

// Create the first line
$p5 = new LinePlot($datay5);
$graph3->Add($p5);
$p5->SetColor("#000000");
$p5->SetLegend('LPS_pressure');

$graph3->legend->SetFrameWeight(1);

// Output line
$graph3->Stroke('graph3_hmmy.jpg');

$result = mysqli_query($con, 'SELECT UV_Light FROM esties ORDER BY id
DESC LIMIT 5');
// Process every record
$oddrow = true;

```

```

$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay6[$count] = $row['UV_Light'];
    $count = $count + 1;
}

// Setup the graph
$graph4 = new Graph(900,300);
$graph4->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph4->SetTheme($theme_class);
$graph4->img->SetAntiAliasing(false);
$graph4->title->Set('UV-Index Graph');
$graph4->SetBox(false);

$graph4->SetMargin(40,20,36,63);

$graph4->img->SetAntiAliasing();

$graph4->yaxis->HideZeroLabel();
$graph4->yaxis->HideLine(false);
$graph4->yaxis->HideTicks(false,false);

$graph4->xgrid->Show();
$graph4->xgrid->SetLineStyle("solid");
$graph4->xaxis->SetTickLabels(array('A','B','C','D'));
$graph4->xgrid->SetColor('#E3E3E3');

// Create the first line
$p6 = new LinePlot($datay6);
$graph4->Add($p6);
$p6->SetColor("000000");
$p6->SetLegend('UV_Light');

$graph4->legend->SetFrameWeight(1);

// Output line
$graph4->Stroke('graph4_hmmy.jpg');
?>
<html lang="en">

<head>
    <link rel="stylesheet" href="styles.css">
</head>

<title>Weather Network</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Montserrat">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

<body>

```



```

<!-- Navbar -->
<ul>
  <li><a href="index1.php">Αρχική</a></li>
  <li><a href="hmmmy.php">Σταθμός HMMY </a></li>
  <li><a href="ekathens.php">Σταθμός Ε.Κ ΑΘΗΝΑ</a></li>
  <li><a href="esties.php">Σταθμός Εστιών</a></li>
  <li><a href="prokat.php">Σταθμός ΠΡΟΚΑΤ</a></li>
  <li><a href="zampretti.php">Πρόβλεψη Zampretti</a></li>
  <li><a href="meteo.php">Μετεωγράμμα</a></li>
</ul>

<!-- Header -->
<header class="header1" >Δίκτυο μετεωρολογικών σταθμών Εάνθης
</header>
<p class = "subheader1"> Θέση: 41.13488 N, 41.13488 24.888 E -
Υψόμετρο:76 μ </p>
<p class = "subheader2"> Provided and supervised by:Maroulakos-
Seferiadis Paris </p>
<br> <br> <br>

<div><p class = "header1">Οι τιμές των πινάκων θα ανανεωθούν σε <span
id="cnt" style="color:red;">120</span> δευτερόλεπτα</p></div>
<script>
  var counter = 120;

  // The countdown method.
  window.setInterval(function () {
    counter--;
    if (counter >= 0) {
      var span;
      span = document.getElementById("cnt");
      span.innerHTML = counter;
    }
    if (counter === 0) {
      clearInterval(counter);
    }

  }, 1000);

  window.setInterval('refresh()', 30000);

  // Refresh or reload page.
  function refresh() {
    window .location.reload();
  }
</script>

<table border="5" cellspacing="0" cellpadding="7" class ="center">
  <caption> Σταθμός Εστιών </caption>
  <tr>
    <td class="table_titles">Ημερομηνία και Ώρα</td>
    <td class="table_titles">Θερμοκρασία</td>
    <td class="table_titles">Υγρασία</td>
    <td class="table_titles">Φωτεινότητα</td>
    <td class="table_titles">Θερμοκρασία (LPS)</td>
    <td class="table_titles">Ατμ.Πίεση</td>
    <td class="table_titles">Δείκτης UV</td>
  </tr>
<?php

```

```

        include('connection.php');
        $result = mysqli_query($con,'SELECT * FROM esties ORDER BY id
DESC LIMIT 5');
        // Process every record
        $oddrow = true;

        while($row = mysqli_fetch_array($result))
        {
            if ($oddrow)
            {
                $css_class=' class="table_cells_odd"';
            }
            else
            {
                $css_class=' class="table_cells_even"';
            }
            $oddrow = !$oddrow;
            echo "<tr align=center style= 'background:white;'>";
            echo "<td '.$css_class.'>" . $row['event'] . "</td>";
            echo "<td '.$css_class.'>" . $row['DHT_temperature'] .
            "°C</td>" ;
            echo "<td '.$css_class.'>" . $row['DHT_humidity'] .
            "%</td>";
            echo "<td '.$css_class.'>" . $row['Brightness'] . "
Lux</td>";
            echo "<td '.$css_class.'>" . $row['LPS_temperature'] .
            "°C</td>" ;
            echo "<td '.$css_class.'>" . $row['LPS_pressure'] .
            "hPa</td>";
            echo "<td '.$css_class.'>" . $row['UV_Light'] . "</td>";
            echo "</tr>";
        }

        // Close the connection
        mysqli_close($con);
    ?>
</table>
<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>
<br> <br> <br> <br>

<div class = "wrapper1">
    
</div>
<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

```

9.7 B.7 prokat.php

```
<!DOCTYPE html>
<?php
$url=$_SERVER['REQUEST_URI'];
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph.php');
require_once ('jpgraph/jpgraph-4.3.5/src/jpgraph_line.php');

include('connection.php');

$result = mysqli_query($con,'SELECT DHT_temperature FROM prokat ORDER
BY id DESC LIMIT 5');
// Process every record
$odddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){

    $datay1[$count] = $row['DHT_temperature'];
    $count = $count + 1;
}

$result = mysqli_query($con,'SELECT LPS_temperature FROM prokat ORDER
BY id DESC LIMIT 5');
// Process every record
$odddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay4[$count] = $row['LPS_temperature'];
    $count = $count + 1;
}

// Setup the graph
$graph = new Graph(900,300);
$graph->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph->SetTheme($theme_class);
$graph->img->SetAntiAliasing(false);
$graph->title->Set('Temperature Graph');
$graph->SetBox(false);

$graph->SetMargin(40,20,36,63);

$graph->img->SetAntiAliasing();

$graph->yaxis->HideZeroLabel();
$graph->yaxis->HideLine(false);
$graph->yaxis->HideTicks(false,false);

$graph->xgrid->Show();
$graph->xgrid->SetLineStyle("solid");
$graph->xaxis->SetTickLabels(array('A','B','C','D'));
$graph->xgrid->SetColor('#E3E3E3');

// Create the first line
$p1 = new LinePlot($datay1);
$graph->Add($p1);
$p1->SetColor("#000000");
$p1->SetLegend('Temperature');
```

```

$p4 = new LinePlot($datay4);
$graph->Add($p4);
$p4->SetColor("#FF0000");
$p4->SetLegend('LPS Temperature');

$graph->legend->SetFrameWeight(1);

// Output line
$graph->Stroke('graph_hmmy.jpg');

$result = mysqli_query($con,'SELECT DHT_humidity FROM prokat ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay2[$count] = $row['DHT_humidity'];
    $count = $count + 1;
}

// Setup the graph
$graph1 = new Graph(900,300);
$graph1->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph1->SetTheme($theme_class);
$graph1->img->SetAntiAliasing(false);
$graph1->title->Set('Humidity Graph');
$graph1->SetBox(false);

$graph1->SetMargin(40,20,36,63);

$graph1->img->SetAntiAliasing();

$graph1->yaxis->HideZeroLabel();
$graph1->yaxis->HideLine(false);
$graph1->yaxis->HideTicks(false,false);

$graph1->xgrid->Show();
$graph1->xgrid->SetLineStyle("solid");
$graph1->xaxis->SetTickLabels(array('A','B','C','D'));
$graph1->xgrid->SetColor('#E3E3E3');

// Create the first line
$p2 = new LinePlot($datay2);
$graph1->Add($p2);
$p2->SetColor("##000000");
$p2->SetLegend('Humidity');

$graph1->legend->SetFrameWeight(1);

// Output line
$graph1->Stroke('graph1_hmmy.jpg');

$result = mysqli_query($con,'SELECT Brightness FROM prokat ORDER BY
id DESC LIMIT 5');
// Process every record
$oddrow = true;

```

```

$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay3[$count] = $row['Brightness'];
    $count = $count + 1;
}

// Setup the graph
$graph2 = new Graph(900,300);
$graph2->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph2->SetTheme($theme_class);
$graph2->img->SetAntiAliasing(false);
$graph2->title->Set('Brightness Graph');
$graph2->SetBox(false);

$graph2->SetMargin(40,20,36,63);

$graph2->img->SetAntiAliasing();

$graph2->yaxis->HideZeroLabel();
$graph2->yaxis->HideLine(false);
$graph2->yaxis->HideTicks(false,false);

$graph2->xgrid->Show();
$graph2->xgrid->SetLineStyle("solid");
$graph2->xaxis->SetTickLabels(array('A','B','C','D'));
$graph2->xgrid->SetColor('#E3E3E3');

// Create the first line
$p3 = new LinePlot($datay3);
$graph2->Add($p3);
$p3->SetColor("#000000");
$p3->SetLegend('Brightness');

$graph2->legend->SetFrameWeight(1);

// Output line
$graph2->Stroke('graph2_hmmy.jpg');

$result = mysqli_query($con,'SELECT LPS_pressure FROM prokat ORDER BY
id DESC LIMIT 5');
// Process every record
$ooddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay5[$count] = $row['LPS_pressure'];
    $count = $count + 1;
}

// Setup the graph
$graph3 = new Graph(900,300);
$graph3->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph3->SetTheme($theme_class);
$graph3->img->SetAntiAliasing(false);
$graph3->title->Set('Pressure Graph');

```

```

$graph3->SetBox(false);

$graph3->SetMargin(40,20,36,63);

$graph3->img->SetAntiAliasing();

$graph3->yaxis->HideZeroLabel();
$graph3->yaxis->HideLine(false);
$graph3->yaxis->HideTicks(false,false);

$graph3->xgrid->Show();
$graph3->xgrid->SetLineStyle("solid");
$graph3->xaxis->SetTickLabels(array('A','B','C','D'));
$graph3->xgrid->SetColor('#E3E3E3');

// Create the first line
$p5 = new LinePlot($datay5);
$graph3->Add($p5);
$p5->SetColor("##000000");
$p5->SetLegend('LPS_pressure');

$graph3->legend->SetFrameWeight(1);

// Output line
$graph3->Stroke('graph3_hmmy.jpg');

$result = mysqli_query($con,'SELECT UV_Light FROM esties ORDER BY id
DESC LIMIT 5');
// Process every record
$oaddrow = true;
$count = 0;
while($row = mysqli_fetch_array($result)){
    $datay6[$count] = $row['UV_Light'];
    $count = $count + 1;
}

// Setup the graph
$graph4 = new Graph(900,300);
$graph4->SetScale("textlin");

$theme_class=new UniversalTheme;

$graph4->SetTheme($theme_class);
$graph4->img->SetAntiAliasing(false);
$graph4->title->Set('UV-Index Graph');
$graph4->SetBox(false);

$graph4->SetMargin(40,20,36,63);

$graph4->img->SetAntiAliasing();

$graph4->yaxis->HideZeroLabel();
$graph4->yaxis->HideLine(false);
$graph4->yaxis->HideTicks(false,false);

$graph4->xgrid->Show();
$graph4->xgrid->SetLineStyle("solid");
$graph4->xaxis->SetTickLabels(array('A','B','C','D'));
$graph4->xgrid->SetColor('#E3E3E3');

// Create the first line

```

```

$p6 = new LinePlot($datay6);
$graph4->Add($p6);
$p6->SetColor("##000000");
$p6->SetLegend('UV_Light');

$graph4->legend->SetFrameWeight(1);

// Output line
$graph4->Stroke('graph4_hmmy.jpg');
?>
<html lang="en">

<head>
  <link rel="stylesheet" href="styles.css">
</head>

<title>Weather Network</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Montserrat">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

<body>

<!-- Navbar -->
<ul>
  <li><a href="index1.php">Αρχική</a></li>
  <li><a href="hmmy.php">Σταθμός HMMY </a></li>
  <li><a href="ekathens.php">Σταθμός Ε.Κ ΑΘΗΝΑ</a></li>
  <li><a href="esties.php">Σταθμός Εστιών</a></li>
  <li><a href="prokat.php">Σταθμός ΠΡΟΚΑΤ</a></li>
  <li><a href="zampretti.php">Πρόβλεψη Zampretti</a></li>
  <li><a href="meteo.php">Μετεωρόγραμμα</a></li>
</ul>

<!-- Header -->
<header class="header1" >Δίκτυο μετεωρολογικών σταθμών Εάνθης
</header>
<p class = "subheader1"> Θέση: 41.13488 N, 41.13488 24.888 E -
Υψόμετρο:76 μ </p>
<p class = "subheader2"> Provided and supervised by:Maroulakos-
Seferiadis Paris </p>
<br> <br> <br>

<div><p class = "header1">Οι τιμές των πινάκων θα ανανεωθούν σε <span
id="cnt" style="color:red;">120</span> δευτερόλεπτα</p></div>
<script>
  var counter = 120;

  // The countdown method.
  window.setInterval(function () {
    counter--;
    if (counter >= 0) {

```

```

        var span;
        span = document.getElementById("cnt");
        span.innerHTML = counter;
    }
    if (counter === 0) {
        clearInterval(counter);
    }

}, 1000);

window.setInterval('refresh()', 30000);

// Refresh or reload page.
function refresh() {
    window .location.reload();
}
</script>

<table border="5" cellspacing="0" cellpadding="7" class ="center">
    <caption> Σταθμός ΠΡΟΚΑΤ </caption>
    <tr>
        <td class="table_titles">Ημερομηνία και Ωρα</td>
        <td class="table_titles">Θερμοκρασία</td>
        <td class="table_titles">Υγρασία</td>
        <td class="table_titles">Φωτεινότητα</td>
        <td class="table_titles">Θερμοκρασία (LPS)</td>
        <td class="table_titles">Ατμ.Πίεση</td>
        <td class="table_titles">Δείκτης UV</td>
    </tr>
    <?php
        include('connection.php');
        $result = mysqli_query($con, 'SELECT * FROM prokat ORDER BY id
DESC LIMIT 5');
        // Process every record
        $oddrow = true;

        while($row = mysqli_fetch_array($result))
        {
            if ($oddrow)
            {
                $css_class=' class="table_cells_odd"';
            }
            else
            {
                $css_class=' class="table_cells_even"';
            }
            $oddrow = !$oddrow;
            echo "<tr align=center style= 'background:white;'>";
            echo "<td '.$css_class.'>" . $row['event'] . "</td>";
            echo "<td '.$css_class.'>" . $row['DHT_temperature'] .
            "<td>" . "</td>";
            echo "<td '.$css_class.'>" . $row['DHT_humidity'] .
            "%</td>";
            echo "<td '.$css_class.'>" . $row['Brightness'] . "
            Lux</td>";
            echo "<td '.$css_class.'>" . $row['LPS_temperature'] .
            "<td>" . "</td>";
            echo "<td '.$css_class.'>" . $row['LPS_pressure'] .
            "hPa</td>";
            echo "<td '.$css_class.'>" . $row['UV_Light'] . "</td>";
            echo "</tr>";

```



```

    }

    // Close the connection
    mysqli_close($con);
    ?>
</table>
<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>
<br> <br> <br> <br>

<div class = "wrapper1">
    
</div>
<br> <br> <br> <br>
<div class = "wrapper1">
    
</div>

```

9.8 B.8 zampretti.php

```

<!DOCTYPE html>
<?php
$url=$_SERVER['REQUEST_URI'];
?>
<html lang="en">

<head>
    <link rel="stylesheet" href="styles.css">
</head>

<title>Weather Network</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Montserrat">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

<body>

<!-- Navbar -->
<ul>
    <li><a href="index1.php">Αρχική</a></li>

```



```

        <option value = "ENE">ENE
        <option value = "E">E
        <option value = "ESE">ESE
        <option value = "SE">SE
        <option value = "SSE">SSE
        <option value = "S">S
        <option value = "SSW">SSW
        <option value = "SW">SW
        <option value = "WSW">WSW
        <option value = "W">W
        <option value = "WNW">WNW
        <option value = "NW">NW
        <option value = "NNW">NNW
    </select>
    &nbsp;&nbsp;&nbsp;Month: <select name="month" size="1">
        <option value = 1>Ιανουάριος
        <option value = 2>Φεβρουάριος
        <option value = 3>Μάρτιος
        <option value = 4>Απρίλιος
        <option value = 5>Μάιος
        <option value = 6>Ιούνιος
        <option value = 7>Ιούλιος
        <option value = 8>Αύγουστος
        <option value = 9>Σεπτέμβριος
        <option value = 10>Οκτώβριος
        <option value = 11>Νοέμβριος
        <option value = 12>Δεκέμβριος
    </select>
</td>
</tr>
<tr bgcolor="#EFEFEF">
    <td width="30%" bgcolor="#EFEFEF">
        <font size="2" face="Verdana, Arial, Helvetica, sans-
        serif">Πίεση: <br />(Σχετική / <b>S</b>ea <b>L</b>evel
        <b>A</b>djusted)</font>
    </td>
    <td width="50%">
        <font size="2" face="Verdana, Arial, Helvetica, sans-
        serif"> hPa:</font> <input type="text" name="baro" size="4"> <font
        size="2" face="Verdana, Arial, Helvetica, sans-serif">&nbsp;&nbsp;&nbsp;
        Τάση:</font>
        <select name="trend" size="1">
            <option selected value = 0>Σταθερή
            <option value = 1>Αυξανόμενη
            <option value = 2>Μειωτική
        </select>
    </td>
</tr>
<tr bgcolor="#EFEFEF">
    <td width="30%" bgcolor="#EFEFEF">&nbsp;&nbsp;&nbsp;</td>
    <td width="50%">
        <input type="submit" value="Πρόβλεψη" onClick="return
        babblecrap()" >
    </td>
</tr>
</table>
</form>

<table border="0" cellspacing="1" cellpadding="5"
align="center" width="50%">
    <tr bgcolor="#2E59AA">

```

```

        <td >
            <font size="4" face="Verdana, Arial, Helvetica,
sans-serif"><b>Πρόβλεψη 12 ωρών:</b> <span id="forecast">Η πρόβλεψη
θα εμφανιστεί εδώ</span></font></font>
        </td>
    </tr>
    <tr bgcolor="#EFFFFFF">
        <td width="30%" bgcolor="#EFFFFFF"><font size="2"
face="Verdana, Arial, Helvetica, sans-serif">Τα καλύτερα αποτελέσματα
τα έχουμε όταν κάνουμε την πρόβλεψη στις 9 το πρωί</font>
        </td>
    </tr>
</table>

</td>
</tr>
</table> <!-- END form container -->

    <p class= "subheader1">Οδηγίες για συμπλήρωση της φόρμας
πρόβλεψης:<br>
    Αρχικά συμπληρώνουμε στις τωρινές μεταβλητές τις συνθήκες
αέρα και το μήνα που βλέπουμε από την κάρτα κάτω <br>
    Στην συνέχεια συμπληρώνουμε την τιμή της πίεσης απο την πιο
πρόσφατη τιμή απο αυτές που βλέπουμε στους πίνακες των σταθμών.<br>
    Για την τάση και το βαρομετρικό χαμηλό και υψηλό πηγαίνουμε
στο site penteli.meteo.gr/stations/xanthi/ και παίρνουμε τις σχετικές
τιμές. <br>
    Η πρόβλεψη δίνει τα καλύτερα αποτελέσματα αν γίνει στις 9 το
πρωί (τοπική ώρα)
    </p>

    </font>
    </td>
    </tr>
</table>

</td></td></table> <!-- END blurb container -->

<div class="wrapper">
    <script
src="https://www.windfinder.com/widget/statistics/js/xanthi?unit_wind
=kmh&unit_temperature=c"></script><noscript><a rel="nofollow"
href="https://www.windfinder.com/statistics/xanthi?utm_source=statist
ics&utm_medium=web&utm_campaign=homepageweather&utm_content=noscript-
statistics">Wind forecast for Xanthi</a> provided by <a
rel="nofollow"
href="https://www.windfinder.com?utm_source=statistics&utm_medium=web
&utm_campaign=homepageweather&utm_content=noscript-
logo">windfinder.com</a></noscript>
</div>
<br><br>
<div class="wrapper">
    <script
src="https://www.windfinder.com/widget/forecast/js/xanthi?unit_wave=m
&unit_rain=mm&unit_temperature=c&unit_wind=kts&unit_pressure=hPa&days
=4&show_day=1&show_waves=0"></script><noscript><a rel="nofollow"
href="https://www.windfinder.com/forecast/xanthi?utm_source=forecast&
utm_medium=web&utm_campaign=homepageweather&utm_content=noscript-
forecast">Wind forecast for Xanthi</a> provided by <a rel="nofollow"
href="https://www.windfinder.com?utm_source=forecast&utm_medium=web&u

```

```

tm_campaign=homepageweather&utm_content=noscript-
logo">windfinder.com</a></noscript>
</div>
<p class="subheader1">Ο ιστότοπος αυτός δημιουργήθηκε για την
παρουσίαση των μετεωρολογικών δεδομένων, που καταγράφονται στους 4
μετεωρολογικούς σταθμούς.</p>

<script>
// Used to toggle the menu on small screens when clicking on the menu
button
function myFunction() {
  var x = document.getElementById("navDemo");
  if (x.className.indexOf("w3-show") == -1) {
    x.className += " w3-show";
  } else {
    x.className = x.className.replace(" w3-show", "");
  }
}

```

9.9 B.9 meteo.php

```

<!DOCTYPE html>
<?php
$url=$_SERVER['REQUEST_URI'];
?>
<html lang="en">

<head>
  <link rel="stylesheet" href="styles.css">
</head>

<title>Weather Network</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lato">
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Montserrat">
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/font-awesome@4.7.0/css/font-awesome.min.css">

<body>

<!-- Navbar -->
<ul>
  <li><a href="index1.php">Αρχική</a></li>
  <li><a href="hmmmy.php">Σταθμός HMMY </a></li>
  <li><a href="ekathens.php">Σταθμός Ε.Κ ΑΘΗΝΑ</a></li>
  <li><a href="esties.php">Σταθμός Εστιών</a></li>
  <li><a href="prokat.php">Σταθμός ΠΡΟΚΑΤ</a></li>
  <li><a href="zampretti.php">Πρόβλεψη Zampretti</a></li>
  <li><a href="meteo.php">Μετεωρόγραμμα</a></li>
</ul>

<!-- Header -->
<header class="header1" >Δίκτυο μετεωρολογικών σταθμών Εάνθης

```

```

</header>
<p class = "subheader1"> Θέση: 41.13488 N, 41.13488 24.888 E -
Υψόμετρο:76 μ </p>
<p class = "subheader2"> Provided and supervised by:Maroulakos-
Seferiadis Paris </p>
<br> <br> <br>
<header class="header1" > Πως διαβάζεται το μετεώγραμμα</header>
<p class = "subheader1"> Η κάθετη κίτρινη περιοχή είναι το φως της
ημέρας και η λευκή η περιοχή είναι η περίοδος της νύχτας
Υπάρχουν τρία διαφορετικά εικονογράμματα:<br>
Θερμοκρασία<br>
Σύννεφα-Βροχόπτωση<br>
Άνεμος<br>
Στο πρώτο διάγραμμα είναι η καμπύλη της θερμοκρασίας με ωριαία βήματα
και με χρωματική απεικόνιση, καθώς και το εικονόγραμμα των καιρικών
συνθηκών, σε βήματα των 6 ωρών. Το αστερί συμβολίζει το χιόνι και το
θαυμαστικό τη βροχή (Όταν αυτά υπάρχουν).<br>
Στο δεύτερο διάγραμμα τα σύννεφα εμφανίζονται με διάφορες χρωματικές
πυκνότητες (αποχρώσεις του γκρι) και σε υψόμετρο ως και 14
χιλιόμετρα.
Ο ολικός υετός (μπλε) και οι όμβροι (γαλάζιο) σε mm, παρουσιάζονται
σαν ένα ραβδόγραμμα στο κάτω μέρος.<br>
Το τρίτο γράφημα εμφανίζει τα βέλη κατεύθυνσης του ανέμου (κάθε
βέλος αντιστοιχεί σε δύο ώρες), τη μέση ταχύτητα του ανέμου(μπλε) και
την αιολική ριπή (πράσινο). <br><br><br>
Πηγή πληροφοριών Vyron.meteonvyronas.gr</p>
<div class="meteogramplace">
<p class = "meteogram"> Μετεώγραμμα 5ημερών για την Εάνθη-Πρόγνωση-
Ανανέωση κάθε 12 ώρες (9:12 UTC,21:12 UTC) </p>
<a
href="https://www.meteoblue.com/el/weather/forecast/multimodel/41.13N
24.89E81" target="_blank" title="meteoblue weather prediction for
Vyronas">

</a>
</div>

</body>
</html>

```

9.10 B.10 connection.php

```

<?php
$username = "paris";
$pass = "paris1997";
$host = "localhost:3306";
$db_name = "ethernet";
$con = mysqli_connect ($host, $username, $pass);

```

```
$db = mysqli_select_db ( $con, $db_name );
?>
```

9.11 B.11 data_hmmy.php

```
<?php
include ('connection.php');
$sql_insert = "INSERT INTO hmmy (DHT_temperature, DHT_humidity,
Brightness, LPS_temperature, LPS_pressure, UV_Light) VALUES
('".$_GET["temperature"]."', '".$_GET["humidity"]."',
'".$_GET["Lux"]."', '".$_GET["lps_temperature"]."',
'".$_GET["lps_pressure"]."', '".$_GET["UV-Light"]."' );";
if(mysqli_query($con,$sql_insert))
{
echo "Done";
mysqli_close($con);
}
else
{
echo "error is ".mysqli_error($con );
}
?>
```

9.12 B.12 data_prokat.php

```
<?php
include ('connection.php');
$sql_insert = "INSERT INTO prokat (DHT_temperature, DHT_humidity,
Brightness, LPS_temperature, LPS_pressure, UV_Light) VALUES
('".$_GET["temperature"]."', '".$_GET["humidity"]."',
'".$_GET["Lux"]."', '".$_GET["lps_temperature"]."',
'".$_GET["lps_pressure"]."', '".$_GET["UV-Light"]."' );";
if(mysqli_query($con,$sql_insert))
{
echo "Done";
mysqli_close($con);
}
else
{
echo "error is ".mysqli_error($con );
}
?>
```

9.13 B.13 data_esties.php

```
<?php
include ('connection.php');
$sql_insert = "INSERT INTO esties (DHT_temperature, DHT_humidity,
Brightness, LPS_temperature, LPS_pressure, UV_Light) VALUES
('".$_GET["temperature"]."', '".$_GET["humidity"]."',
'".$_GET["Lux"]."', '".$_GET["lps_temperature"]."',
'".$_GET["lps_pressure"]."', '".$_GET["UV-Light"]."' );";
```

```

if(mysqli_query($con,$sql_insert))
{
echo "Done";
mysqli_close($con);
}
else
{
echo "error is ".mysqli_error($con );
}
?>

```

9.14 B.14 data_athina.php

```

<?php
include ('connection.php');
$sql_insert = "INSERT INTO athina (DHT_temperature, DHT_humidity,
Brightness, LPS_temperature, LPS_pressure, UV_Light) VALUES
('".$_GET["temperature"]."', '".$_GET["humidity"]."',
'".$_GET["Lux"]."', '".$_GET["lps_temperature"]."',
'".$_GET["lps_pressure"]."', '".$_GET["UV-Light"]."' );";
if(mysqli_query($con,$sql_insert))
{
echo "Done";
mysqli_close($con);
}
else
{
echo "error is ".mysqli_error($con );
}
?>

```

9.15 B.15 hmmy_db.sql

```

CREATE TABLE IF NOT EXISTS ethernet.hmmy(
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    event TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    DHT_temperature VARCHAR(6) NOT NULL,
    DHT_humidity VARCHAR(6) NOT NULL,
    Brightness VARCHAR(8) NOT NULL,
    LPS_temperature VARCHAR(10) NOT NULL,
    LPS_pressure VARCHAR(10) NOT NULL,
    UV_Light VARCHAR(3) NOT NULL
)

```

9.16 B.16 prokat_db.sql

```

CREATE TABLE IF NOT EXISTS ethernet.prokat(
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    event TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    DHT_temperature VARCHAR(6) NOT NULL,

```



```

    DHT_humidity VARCHAR(6) NOT NULL,
    Brightness VARCHAR(8) NOT NULL,
    LPS_temperature VARCHAR(10) NOT NULL,
    LPS_pressure VARCHAR(10) NOT NULL,
    UV_Light VARCHAR(3) NOT NULL
)

```

9.17 B.17 athina db.sql

```

CREATE TABLE IF NOT EXISTS ethernet.athina(
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    event TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    DHT_temperature VARCHAR(6) NOT NULL,
    DHT_humidity VARCHAR(6) NOT NULL,
    Brightness VARCHAR(8) NOT NULL,
    LPS_temperature VARCHAR(10) NOT NULL,
    LPS_Pressure VARCHAR(10) NOT NULL,
    UV_Light VARCHAR(3) NOT NULL
)

```

9.18 B.18 esties db.sql

```

CREATE TABLE IF NOT EXISTS ethernet.esties(
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    event TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    DHT_temperature VARCHAR(6) NOT NULL,
    DHT_humidity VARCHAR(6) NOT NULL,
    Brightness VARCHAR(8) NOT NULL,
    LPS_temperature VARCHAR(10) NOT NULL,
    LPS_pressure VARCHAR(10) NOT NULL,
    UV_Light VARCHAR(3) NOT NULL
)

```

9.19 B.19 betel cast.js

```

// beteljuice.com - near enough Zambretti Algorhithm
// June 2008 - v1.0
// tweak added so decision # can be output

/* Negretti and Zambras 'slide rule' is supposed to be better than
90% accurate
for a local forecast upto 12 hrs, it is most accurate in the
temperate zones and about 09:00 hrs local solar time.
I hope I have been able to 'tweak it' a little better ;- )
This code is free to use and redistribute as long as NO CHARGE is
EVER made for its use or output
*/

// ---- 'environment' variables -----
var z_where = 1; // Northern = 1 or Southern = 2 hemisphere
var z_baro_top = 1050; // upper limits of your local 'weather window'
(1050.0 hPa for UK)
var z_baro_bottom = 950; // lower limits of your local 'weather
window' (950.0 hPa for UK)

```

```

// usage:  forecast = betel_cast( z_hpa, z_month, z_wind, z_trend [,
z_where] [, z_baro_top] [, z_baro_bottom])[0];

// z_hpa is Sea Level Adjusted (Relative) barometer in hPa or mB
// z_month is current month as a number between 1 to 12
// z_wind is English windrose cardinal eg. N, NNW, NW etc.
// NB. if calm a 'nonsense' value should be sent as z_wind
(direction) eg. 1 or calm !
// z_trend is barometer trend: 0 = no change, 1= rise, 2 = fall
// z_where - OPTIONAL for posting with form
// z_baro_top - OPTIONAL for posting with form
// z_baro_bottom - OPTIONAL for posting with form
// [0] a short forecast text is returned
// [1] zambretti severity number (0 - 25) is returned ie.
betel_cast() returns a two deep array

var z_forecast = new Array("Settled fine", "Fine weather", "Becoming
fine", "Fine, becoming less settled", "Fine, possible showers",
"Fairly fine, improving", "Fairly fine, possible showers early",
"Fairly fine, showery later", "Showery early, improving",
"Changeable, mending", "Fairly fine, showers likely", "Rather
unsettled clearing later", "Unsettled, probably improving", "Showery,
bright intervals", "Showery, becoming less settled", "Changeable,
some rain", "Unsettled, short fine intervals", "Unsettled, rain
later", "Unsettled, some rain", "Mostly very unsettled", "Occasional
rain, worsening", "Rain at times, very unsettled", "Rain at frequent
intervals", "Rain, very unsettled", "Stormy, may improve", "Stormy,
much rain");

// equivalents of Zambretti 'dial window' letters A - Z
var rise_options  = new
Array(25,25,25,24,24,19,16,12,11,9,8,6,5,2,1,1,0,0,0,0,0,0) ;
var steady_options = new
Array(25,25,25,25,25,25,23,23,22,18,15,13,10,4,1,1,0,0,0,0,0,0) ;
var fall_options = new
Array(25,25,25,25,25,25,25,25,23,23,21,20,17,14,7,3,1,1,1,0,0,0) ;

var z_test = new Array();

// ---- MAIN FUNCTION -----
---
function betel_cast( z_hpa, z_month, z_wind, z_trend, z_hemisphere,
z_upper, z_lower) {
    if(z_hemisphere) z_where = z_hemisphere;    // used by input form
    if(z_upper) z_baro_top = z_upper;    // used by input form
    if(z_lower) z_baro_bottom = z_lower;    // used by input form
    z_range = z_baro_top - z_baro_bottom;
    z_constant = (z_range / 22).toFixed(3);

    z_season = (z_month >= 4 && z_month <= 9) ;    // true if 'Summer'
    if (z_where == 1) {    // North hemisphere
        if (z_wind == "N") {
            z_hpa += 6 / 100 * z_range ;
        } else if (z_wind == "NNE") {
            z_hpa += 5 / 100 * z_range ;
        } else if (z_wind == "NE") {
            z_hpa += 4 ;
        } else if (z_wind == "ENE") {
            z_hpa += 2 / 100 * z_range ;
        }
    }
}

```

```

    } else if (z_wind == "E") {
        z_hpa -= 0.5 / 100 * z_range ;
    } else if (z_wind == "ESE") {
//      z_hpa -= 3 ;
        z_hpa -= 2 / 100 * z_range ;
    } else if (z_wind == "SE") {
        z_hpa -= 5 / 100 * z_range ;
    } else if (z_wind == "SSE") {
        z_hpa -= 8.5 / 100 * z_range ;
    } else if (z_wind == "S") {
//      z_hpa -= 11 ;
        z_hpa -= 12 / 100 * z_range ;
    } else if (z_wind == "SSW") {
        z_hpa -= 10 / 100 * z_range ; //
    } else if (z_wind == "SW") {
        z_hpa -= 6 / 100 * z_range ;
    } else if (z_wind == "WSW") {
        z_hpa -= 4.5 / 100 * z_range ; //
    } else if (z_wind == "W") {
        z_hpa -= 3 / 100 * z_range ;
    } else if (z_wind == "WNW") {
        z_hpa -= 0.5 / 100 * z_range ;
    } else if (z_wind == "NW") {
        z_hpa += 1.5 / 100 * z_range ;
    } else if (z_wind == "NNW") {
        z_hpa += 3 / 100 * z_range ;
    }
    if (z_season == 1) { // if Summer
        if (z_trend == 1) { // rising
            z_hpa += 7 / 100 * z_range ;
        } else if (z_trend == 2) { // falling
            z_hpa -= 7 / 100 * z_range ;
        }
    }
} else { // must be South hemisphere
    if (z_wind == "S") {
        z_hpa += 6 / 100 * z_range ;
    } else if (z_wind == "SSW") {
        z_hpa += 5 / 100 * z_range ;
    } else if (z_wind == "SW") {
//      z_hpa += 4 ;
        z_hpa += 5 / 100 * z_range ;
    } else if (z_wind == "WSW") {
        z_hpa += 2 / 100 * z_range ;
    } else if (z_wind == "W") {
        z_hpa -= 0.5 / 100 * z_range ;
    } else if (z_wind == "WNW") {
//      z_hpa -= 3 ;
        z_hpa -= 2 / 100 * z_range ;
    } else if (z_wind == "NW") {
        z_hpa -= 5 / 100 * z_range ;
    } else if (z_wind == "NNW") {
        z_hpa -= 8.5 / 100 * z_range ;
    } else if (z_wind == "N") {
//      z_hpa -= 11 ;
        z_hpa -= 12 / 100 * z_range ;
    } else if (z_wind == "NNE") {
        z_hpa -= 10 / 100 * z_range ; //
    } else if (z_wind == "NE") {
        z_hpa -= 6 / 100 * z_range ;
    } else if (z_wind == "ENE") {

```

```

        z_hpa -= 4.5 / 100 * z_range ;    //
    } else if (z_wind == "E") {
        z_hpa -= 3 / 100 * z_range ;
    } else if (z_wind == "ESE") {
        z_hpa -= 0.5 / 100 * z_range ;
    } else if (z_wind == "SE") {
        z_hpa += 1.5 / 100 * z_range ;
    } else if (z_wind == "SSE") {
        z_hpa += 3 / 100 * z_range ;
    }
    if (z_season == 0) {    // if Winter
        if (z_trend == 1) { // rising
            z_hpa += 7 / 100 * z_range;
        } else if (z_trend == 2) { // falling
            z_hpa -= 7 / 100 * z_range;
        }
    }
} // END North / South

if(z_hpa == z_baro_top) z_hpa = z_baro_top - 1;
z_option = Math.floor((z_hpa - z_baro_bottom) / z_constant);
z_output = "";
if(z_option < 0) {
    z_option = 0;
    z_output = "Exceptional Weather, ";
}
if(z_option > 21) {
    z_option = 21;
    z_output = "Exceptional Weather, ";
}

    if (z_trend == 1) {    // rising
        z_output += z_forecast[rise_options[z_option]] ;
    } else if (z_trend == 2) {    // falling
        z_output += z_forecast[fall_options[z_option]] ;
    } else {    // must be 'steady'
        z_output += z_forecast[steady_options[z_option]] ;
    }
z_test[1] = z_output;
// return z_output ;
z_test[0] = z_output ;
return z_test ;
} // END function

function babblecrap() {
    baro = document.bert.baro.value *1;
    month = document.bert.month.value;
    wind = document.bert.wind.value;
    trend = document.bert.trend.value;
    where = document.bert.hemisphere.value;
    high = document.bert.range_high.value *1;
    low = document.bert.range_low.value *1;

    johnny = document.getElementById('forecast');
    johnny.innerHTML= betel_cast(baro, month, wind, trend, where,
high, low)[0] ;
} // END function babblecrap()

```

9.20 B.20 main.js

```
// For toggle button;

function toggleClass()
{
    const body = document.querySelector('body');
    body.classList.toggle('light');
    body.style.transition = `0.3s linear`;
}

// for time;
const deg = 6;
// 360 / (12 * 5);

const hr = document.querySelector('#hr');
const mn = document.querySelector('#mn');
const sc = document.querySelector('#sc');

setInterval(() => {

    let day = new Date();
    let hh = day.getHours() * 30;
    let mm = day.getMinutes() * deg;
    let ss = day.getSeconds() * deg;
    let msec = day.getMilliseconds();

    // VERY IMPORTANT STEP:

    hr.style.transform = `rotateZ(${(hh) + (mm / 12)}deg)`;
    mn.style.transform = `rotateZ(${mm}deg)`;
    sc.style.transform = `rotateZ(${ss}deg)`;

    // gives the smooth transitioning effect, but there's a bug here!
    // sc.style.transition = `1s`;

});
```