



Πανεπιστήμιο Δυτικής Μακεδονίας
Τμήμα Μηχανικών Πληροφορικής & Τηλεπικοινωνιών

Διπλωματική Εργασία

Σχεδιασμός και υλοποίηση της υποδομής μιας ευφυούς οικολογικής οικίας

Design and implementation of the infrastructure of an
intelligent ecological house

Αναστάσιος Λούμπας
Επιβλέπων Καθηγητής: Δρ. Μηνάς Δασυγένης
Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής
Υπολογιστών

Νοέμβριος 2017, Κοζάνη

Δήλωση Πνευματικών Δικαιωμάτων

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ. 3 του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο: “Σχεδιασμός και υλοποίηση της υποδομής μιας ευφυούς οικολογικής οικίας” καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Μηνά Δασυγένη, αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και μόνο.

Copyright (C) Αναστάσιος Λούμπας & Μηνάς Δασυγένης, 2017, Κοζάνη

Υπογραφή Φοιτητή

Ευχαριστίες

Θα ήθελα να ευχαριστήσω όλα τα άτομα τα οποία βρίσκονταν κοντά μου και με βοήθησαν στο να εκπληρώσω ένα μεγάλο αλλά και συνάμα δύσκολο στόχο τον οποίο είχα θέσει πριν από αρκετό καιρό. Προπαντός βέβαια, τίποτα από όλα αυτά δεν θα ήταν δυνατό χωρίς την στήριξη της οικογένειάς μου η οποία δεν δίσταζε να μου προσφέρει ότι χρειαζόμουν ώστε να καταφέρω να επιτύχω το στόχο μου. Το επόμενο πολύ σημαντικό κομμάτι το οφείλω στους καθηγητές μου οι οποίοι φρόντιζαν να με καθοδηγούν προς την επιτυχία. Κλείνοντας, θέλω να ευχαριστήσω τους φίλους μου, οι οποίοι είτε άμεσα είτε έμμεσα συνείσφεραν στον αγώνα αυτό.

Περίληψη

Σε μια σύγχρονη εποχή, όπως η σημερινή, η ανάγκη για ενσωμάτωση της τεχνολογίας στις λειτουργίες ενός σπιτιού είναι μεγάλη και επιθυμητή. Ως απόρροια αυτού, οι άνθρωποι θα κερδίζουν χρόνο από την καθημερινότητά τους, διότι θα έχουν την δυνατότητα να ελέγχουν το σπίτι τους καθώς και να ενημερώνονται για την κατάσταση του από απόσταση. Η αυτοματοποίηση που προσφέρει η είσοδος της τεχνολογίας στη ζωή των ανθρώπων εξυπηρετεί, συνάμα, την εξοικονόμηση κάθε τύπου ενέργειας. Προσπάθειες υλοποίησης ενός τέτοιου σπιτιού έχουν γίνει ήδη. Αδιαμφισβήτητα η παρούσα διπλωματική εργασία αποτελεί μία από αυτές. Η βασική επιδίωξή της είναι να λύσει το πρόβλημά των προηγούμενων, το οποίο αφορά τον ενοποιημένο έλεγχο μέσω μιας κεντρικής δομής.

Την κατανόηση όσων αναλύονται στο κείμενο της διπλωματικής εργασίας ενισχύει η παρουσίαση των εννοιών, των τεχνολογιών και των εργαλείων λογισμικού που πραγματοποιείται. Για την επίτευξη του στόχου της διπλωματικής εργασίας, ήταν απαραίτητη η υλοποίηση μιας κεντρικής διαχειριστικής εφαρμογής καθώς και μίας συσκευής για τη στελέχωση της λειτουργικότητας ολόκληρου του συστήματος.

Η κατασκευή της συσκευής του συστήματος βασίστηκε στην αρχιτεκτονική Arduino. Αποτελεί έναν 'έξυπνο' θερμοστάτη, ο οποίος προσαρμόζεται στις ανάγκες του περιβάλλοντος του και στις απαιτήσεις του χρήστη. Από την άλλη, η εφαρμογή αποτελείται από μία ιστοσελίδα και τον MQTT broker. Μέσω αυτής, πραγματοποιείται ο από απόσταση έλεγχος και η ενημέρωση σχετικά με την κατάστασή του συστήματος. Ο χρήστης ειδοποιείται, τέλος, μέσω email, όταν μερικές από τις συσκευές του βρίσκονται εκτός λειτουργίας για ένα ορισμένο χρονικό διάστημα.

Φυσικά, όλα τα παραπάνω δεν αποτελούν μόνο θεωρητικά σχέδια, αλλά έχουν υλοποιηθεί αποδεικνύοντας ότι η βασική επιδίωξη είναι πραγματικά επιτεύξιμη.

Λέξεις κλειδιά: IoT, Ευφυές, MQTT, Οικολογικό, Προγραμματισμός διαδικτύου, Arduino, Ruby, Rails, MVC

Abstract

In a modern era, like today, the need to integrate technology into the operations of a home is attractive. As a result, people will gain time from their lives because they will be able to control their home and be informed of their status remotely. The automation offered by the entry of technology into the lives of people also helps to save all types of energy. Of course, efforts to solve this problem already exists. Undoubtedly this thesis is one of them. Its main objective is to solve the problem of the former ones, which concerns the unified control through a central structure.

The understanding of what is analyzed in the thesis is enhanced by the presentation of concepts, technologies and software tools. In order to achieve the goal of this thesis, it was necessary to implement a central management application as well as a device for staffing the functionality of the whole system.

The construction of the system device was based on the Arduino architecture. It is a 'smart' thermostat that adapts to the needs of its environment and its user requirements. On the other hand, the application consists of a website and the MQTT broker. Through the application, users are given the ability to control their various devices and to get feedback for their current status. Finally, user is informed by email when some of his devices are off for a certain period of time.

Of course, the previous statements is not only a theoretical plan, but they have been implemented proving that the main objective of this thesis is truly achievable.

Keywords: IoT, Smart, MQTT, Ecological, Web Development, Arduino, Ruby, Rails, MVC

Περιεχόμενα

1	Εισαγωγή	12
1.1	Ανάλυση Διπλωματικής Εργασίας	12
1.2	Συσχετιζόμενη Εργασία	13
1.3	Σύνοψη της Διπλωματικής Εργασίας	14
2	Θεωρητικό υπόβαθρο	16
2.1	Προγραμματισμός διαδικτύου	16
2.1.1	HTML	17
2.1.2	CSS	18
2.1.3	JavaScript	19
2.1.4	MySQL	20
2.1.5	Ruby	20
2.2	Η αρχιτεκτονική Model-View-Controller	21
2.2.1	Model	21
2.2.2	View	22
2.2.3	Controller	22
2.3	Βιβλιοθήκες, frameworks, πρωτόκολλα και τεχνικές	22
2.3.1	CSS προεπεξεργαστές	23
2.3.2	Bootstrap	23
2.3.3	AJAX	23
2.3.4	Ruby on Rails	24
2.3.5	MQTT	25
2.4	Αρχιτεκτονική Arduino	25
2.4.1	Είδη Arduino	26
2.4.2	Arduino Shields	26
2.5	Πλατφόρμες Ανάπτυξης Λογισμικού	27

2.5.1	RubyMine	28
2.5.2	Atom	28
2.6	Λογισμικά σχεδίασης & επεξεργασίας εικόνας	28
2.6.1	Fritzing	28
2.6.2	GIMP	29
3	Δομικά μέρη συστήματος	31
3.1	Απαιτήσεις συστήματος	31
3.2	Ανάλυση βάσης δεδομένων	33
3.2.1	Δομή του πίνακα “users”	34
3.2.2	Δομή της ομάδας πινάκων “devices”	36
3.2.3	Δομή του πίνακα “device_infos”	38
3.2.4	Δομή του πίνακα “actions”	39
3.2.5	Δομή της ομάδας πινάκων “schedules”	40
3.2.6	Δομή της ομάδας πινάκων “quick_buttons”	42
3.2.7	Δομή της ομάδας πινάκων “smart_thermostats”	43
3.3	Διάγραμμα οντοτήτων συσχετίσεων	45
3.4	Πρωτόκολλο επικοινωνίας	46
4	Υλικό μέρος συστήματος	48
4.1	Arduino Uno	49
4.2	Ethernet Shield	52
4.3	Ψηφιακός αισθητήρας θερμοκρασίας & υγρασίας	53
4.4	Διακόπτες	54
4.5	Shift Register	56
4.6	LEDs	57
4.7	Τελικό Κύκλωμα	58
4.8	Υπολογισμός κόστους εξαρτημάτων	59
5	Λογισμικό μέρος συστήματος	62
5.1	Ανάλυση λειτουργιών ιστοσελίδας	62
5.1.1	Αρχική σελίδα	62
5.1.2	Λειτουργίες χρηστών	63
5.1.3	Λειτουργίες συσκευών	64

5.1.4	Λειτουργίες χρονοπρογραμματισμού	68
5.1.5	Λειτουργίες κουμπιών άμεσης εκτέλεσης ενεργειών	71
5.2	Ανάλυση λειτουργιών παρασκηνίου συστήματος	73
5.2.1	Λειτουργία έξυπνου θερμοστάτη	73
5.2.2	Cron Jobs & Rake Tasks	75
5.3	Ανάλυση λογισμικού Arduino	76
5.3.1	Ανάλυση λειτουργίας συσκευής	76
5.3.2	Παρουσίαση βιβλιοθηκών και συναρτήσεων	77
5.4	Μετρικές κώδικα	84
5.4.1	Rails	84
5.4.2	Arduino	84
6	Επίλογος	86
6.1	Σύνοψη και συμπεράσματα	86
6.2	Προβλήματα τα οποία προέκυψαν και η αντιμετώπιση τους	87
6.3	Μελλοντικές επεκτάσεις	88
	Βιβλιογραφία	90
A'	Οδηγίες ρύθμισης του συστήματος	93
A'.1	Εγκατάσταση ιστοσελίδας	93
A'.1.1	Προαπαιτούμενα	93
A'.1.2	Αρχικές ρυθμίσεις	94
A'.1.3	Ρύθμισή της βάσης δεδομένων	94
A'.1.4	Ρύθμιση του Mailer	95
A'.1.5	Ρύθμισή του Cron	95
A'.1.6	Δοκιμή της εφαρμογής σε development περιβάλλον	96
A'.1.7	Μεταφορά της εφαρμογής σε production περιβάλλον μέσω Apache	97
A'.1.8	Βοηθητικές οδηγίες για εγκατάσταση σε FreeBSD	97
A'.2	Ρύθμιση συσκευής Arduino	98
A'.2.1	Atom/PlatformIO	99
A'.2.2	Arduino IDE	99

Κατάλογος Εικόνων

2.1	Παράδειγμα σύνταξης CSS	19
2.2	Σχεδιάγραμμα λειτουργίας της MVC αρχιτεκτονικής	21
2.3	Πλακέτα Arduino UNO	26
2.4	Πλακέτα Arduino LilyPad	26
2.5	Πλακέτα Arduino Nano	27
2.6	Πλακέτα Arduino Mega	27
2.7	Arduino Ethernet Shield	27
2.8	Το περιβάλλον σχεδίασης του Fritzing	29
2.9	Το περιβάλλον χρήσης του GIMP	30
3.1	Διάγραμμα οντοτήτων συσχετίσεων	46
3.2	Draft πρωτοκόλλου επικοινωνίας	47
4.1	Πλακέτα Arduino Uno R3	50
4.2	Ethernet Shield R3	52
4.3	Αισθητήρας Θερμοκρασίας & Υγρασίας RHT03 (DHT22)	53
4.4	Σύνδεση αισθητήρα DHT22 με το Arduino Uno	54
4.5	Διακόπτης	55
4.6	Κύκλωμα διακόπτη με pull-up αντίσταση	55
4.7	Shift Register SN74HC595N	56
4.8	Τα pins του Shift Register SN74HC595N	56
4.9	Διάγραμμα χρονισμού του Shift Register SN74HC595N	58
4.10	LEDs	59
4.11	Κύκλωμα LED	59
4.12	Κύκλωμα σε breadboard	59
4.13	Ηλεκτρονικό σχέδιο κυκλώματος	60
4.14	Η Τελική Συσκευή	61

5.1	Αρχική σελίδα χωρίς σύνδεση	63
5.2	Αρχική σελίδα εφόσον έχει πραγματοποιηθεί σύνδεση	63
5.3	Σελίδα εισόδου χρήστη	64
5.4	Σύνοψη συσκευών χρήστη	64
5.5	Δημιουργία νέας συσκευής	65
5.6	Επεξεργασία συσκευής - Ρυθμίσεις έξυπνου θερμοστάτη	66
5.7	Αναλυτικές πληροφορίες συσκευής - Ονομασία, κατάσταση και χαρακτηριστικά	67
5.8	Αναλυτικές πληροφορίες συσκευής - Πληροφορίες συσκευής, προγράμματα, quick buttons	68
5.9	Σύνοψη χρονοπρογραμματισμών χρήστη	69
5.10	Δημιουργία νέου προγράμματος	70
5.11	Επικαλυπτόμενα προγράμματα	71
5.12	Σύνοψη quick buttons χρήστη	71
5.13	Δημιουργία νέου quick button	72
5.14	Αναλυτικές πληροφορίες ενός Quick Button	73
5.15	Αναλυτικές πληροφορίες συσκευής - Καμπύλες λειτουργίας	74

Κατάλογος πινάκων

2.1	Παρουσίαση βασικών στοιχείων HTML	18
3.1	Πίνακας “users”	34
3.2	Πίνακας “devices”	36
3.3	Πίνακας “device_attributes”	38
3.4	Πίνακας “device_infos”	39
3.5	Πίνακας “actions”	39
3.6	Πίνακας “schedules”	40
3.7	Πίνακας “schedule_events”	41
3.8	Πίνακας “schedule_event_actions”	41
3.9	Πίνακας “quick_buttons”	42
3.10	Πίνακας “quick_button_actions”	43
3.11	Πίνακας “smart_thermostats”	43
3.12	Πίνακας “smart_thermostats_training_set_samples”	44
3.13	Πίνακας “smart_thermostats_computed_datasets”	45
4.1	Περιγραφή λειτουργίας φωτεινών ενδείξεων του Ethernet Shield . . .	53
4.2	Κοστολόγιο υλικών	60
5.1	Μετρικές κώδικα της ιστοσελίδας του συστήματος	84
5.2	Μετρικές κώδικα της συσκευής Arduino του συστήματος	85

Κατάλογος απεικονίσεων

2.1	Παράδειγμα SQL ερωτήματος	20
A'.1	Παράδειγμα δημιουργίας νέας βάσης μέσω εντολής MySQL	94
A'.2	Δημιουργία χρήστη και παράθεση δικαιωμάτων σε βάση	94
A'.3	Δοκιμή αποστολής email	95
A'.4	Ρύθμιση του Cron για την εκτέλεση της εργασίας η οποία αναλαμβάνει την αποστολή ειδοποιήσεων στους χρήστες των μη συνδεδεμένων συσκευών	96
A'.5	Ρύθμιση του Cron για την εκτέλεση της εργασίας με την οποία λαμβάνονται δείγματα για τις απαραίτητες τιμές καθενός 'έξυπνου θερμοστάτη'	96
A'.6	Ρύθμιση του Cron για την εκτέλεση της εργασίας η οποία επαναυπολογίζει τον χρόνο έναρξης προγράμματος ενός 'έξυπνου θερμοστάτη'	96
A'.7	Ρύθμιση του Cron για την εκτέλεση της εργασίας η οποία αναλύει το σύνολο δειγμάτων ενός 'έξυπνου θερμοστάτη'	96

Κεφάλαιο 1

Εισαγωγή

Στον, εν εξελίξει, 21^ο αιώνα οι υποχρεώσεις (επαγγελματικές και μη) των ανθρώπων, αδιαμφισβήτητα, ολοένα και πληθαίνουν. Επομένως, αυτό σημαίνει ότι οι τελευταίοι απουσιάζουν πολλές ώρες από το σπίτι χωρίς να προλαβαίνουν να τις διεκπεραιώνουν όλες. Ο τρέχων αιώνας έχει στιγματιστεί, επίσης, από την ευρεία χρήση του διαδικτύου και την διασύνδεση μεμονωμένων συσκευών σε αυτό. Με αυτά τα δεδομένα, η διασύνδεση των μεμονωμένων συσκευών, η οποία σαφώς επιτυγχάνεται με την χρήση του διαδικτύου, δύναται να διευκολύνει την ζωή των ανθρώπων αφού μπορεί να συμβάλλει στην αυτοματοποίηση των υποχρεώσεων.

Ως επακόλουθο, η προαναφερθείσα αυτοματοποίηση μπορεί να επιτευχθεί μέσω της διασύνδεσης οικιακών συσκευών με κάποια εφαρμογή. Η εφαρμογή αυτή ενορχηστρώνει την διασύνδεση των συσκευών και προσφέρει μια διεπαφή για την αλληλεπίδραση με τον χρήστη. Όσον αφορά τις συσκευές που συνδέονται στο διαδίκτυο, τα τελευταία χρόνια έχει δημιουργηθεί ο εξής όρος για την περιγραφή τους: “Διαδίκτυο των πραγμάτων” (Internet of Things ή IoT). Ο συγκεκριμένος όρος δηλαδή χαρακτηρίζει τις συσκευές που έχουν την δυνατότητα να συνδέονται στο διαδίκτυο. Άρα αυτό σημαίνει ότι, ο οποιοσδήποτε μπορεί να ελέγχει τις συσκευές απομακρυσμένα και παράλληλα να λαμβάνει τις πληροφορίες που η συσκευή αποστέλλει ως ανάδραση (feedback). Τέλος, η κεντρική εφαρμογή που αναλαμβάνει την διασύνδεση όλων των μερών μεταξύ τους, ονομάζεται κεντρικός κόμβος (hub).

1.1 Ανάλυση Διπλωματικής Εργασίας

Συνδυάζοντας τα σύγχρονα τεχνολογικά μέσα που έχουν ήδη καταγραφεί, προέκυψε η ιδέα για τον σχεδιασμό και την κατασκευή μιας υποδομής ενός ευφυούς

οικολογικού σπιτιού.

Το παραπάνω εγχείρημά είναι, στην ουσία, ένα σύστημα, το οποίο απαρτίζεται από τις συσκευές και την εφαρμογή, και κατ' επέκταση το ίδιο το αντικείμενο της παρούσας διπλωματικής εργασίας. Η βασικότερη λειτουργία του συστήματος είναι η ενεργοποίηση και η απενεργοποίηση συσκευών. Αυτό επιτυγχάνεται με την άμεση παρουσία του χρήστη και το πάτημα ενός κουμπιού, ή με την ύστερη ενεργοποίηση του μέσω της λειτουργίας του χρονοπρογραμματισμού.

Παρόμοιες υλοποιήσεις υφίστανται ήδη στην αγορά, οι οποίες ωστόσο παρουσιάζουν ένα καίριο μειονέκτημα. Τα συστήματα που υπάρχουν ήδη δεν έχουν την δυνατότητα της συνεργασίας με αλλά συστήματα πάρα μόνο με συστήματα της ίδιας εταιρίας ή είναι συσκευές που λειτουργούν μεμονωμένα. Αυτό συμβαίνει γιατί, κάθε εταιρία (υλοποίηση) εξυπηρετεί το δικό της προσωπικό συμφέρον με αποτέλεσμα να μην υπάρχει μια ομαλή και διαφανή επικοινωνία μεταξύ όλων αυτών των συστημάτων. Τελευταία γίνονται προσπάθειες αναίρεσης αυτού του περιορισμού. Η ουσιαστικότερη όλων όμως αποτελεί η παρούσα διπλωματική εργασία. Αυτός είναι και ο λόγος για τον οποίο το συγκεκριμένο σύστημα ξεχωρίζει από όλα τα υπόλοιπα.

Μέσα από τις πρωταρχικές λειτουργίες του συστήματος απορρέουν και ορισμένες δευτερεύουσες. Μερικές από αυτές είναι η εξοικονόμηση στην κατανάλωση νερού και ηλεκτρικής ενέργειας, μιας και ο όρος "οικολογία" (βλ. θέμα) από μόνος του αναφέρεται στην μη κατάχρηση των πηγών του φυσικού περιβάλλοντος. Εξοικονομείται, επίσης, χρόνος εφόσον η άμεση παρουσία του ανθρώπου/χρήστη δεν αποτελεί απαραίτητη προϋπόθεση για την διεκπεραίωση των ενεργειών του συστήματος. Το σύστημα μπορεί ταυτόχρονα να αυτοελέγχεται για πιθανές βλάβες με τον έλεγχο των υπαρχόντων αισθητήρων του.

1.2 Συσχετιζόμενη Εργασία

Όπως προαναφέρθηκε στην ανάλυση της διπλωματικής εργασίας, το παρόν έργο δεν είναι το μοναδικό που υφίσταται. Η ύπαρξή του πηγάζει από την ανάγκη να καλυφθούν τα κενά των προηγούμενων υλοποιήσεων επεκτείνοντας τα ή προσθέτοντας νέα χαρακτηριστικά.

Μια υλοποίηση αυτού του είδους αποτελεί η διπλωματική εργασία 'ippower' [40], η οποία είχε ως σκοπό τον απομακρυσμένο έλεγχο συσκευών ηλεκτρικής

ισχύος. Στην περίπτωση αυτή, το σύστημα ήταν περιορισμένο στην ενσωμάτωση ενός μόνο τύπου συσκευών συστήματος. Δηλαδή, σε συσκευές συστήματος οι οποίες επιτηρούν τη ροή ή μη της ηλεκτρικής ισχύος σε μία συσκευή κατανάλωσης, μέσω διαδικτυακά ελεγχόμενων ρελέ. Ωστόσο, προέκυψε η ανάγκη της ενσωμάτωσης και άλλων τύπων συσκευών στο σύστημα απομακρυσμένου ελέγχου. Έτσι, η παρούσα διπλωματική εργασία υλοποιήθηκε με στόχο την άρση του περιορισμού αυτού και επέτρεψε την εύκολη ενσωμάτωση οποιασδήποτε συσκευής στο σύστημα.

Παρόμοια προσπάθεια αποτελεί η εμπορική συσκευή Nest Thermostat [27]. Το Nest είναι μια ‘έξυπνη’ συσκευή θερμοστάτη η οποία αντικαθιστά τον παραδοσιακό κοινό θερμοστάτη. Ο τελευταίος, τίθεται απλά σε μία θερμοκρασία και παραμένει σε αυτή. Ενώ, ο Nest προσφέρει τη δυνατότητα δημιουργίας προγράμματος για τη ρύθμιση της θερμοκρασίας. Πληροφορίες, βέβαια, για το Nest και το περιβάλλον του μπορεί κάποιος να αντλήσει μόνο μέσω της εφαρμογής του. Η συσκευή ‘έξυπνου θερμοστάτη’ που υλοποιήθηκε, όμως, επιλύει το πρόβλημα της σύνδεσης με μία μεμονωμένη εφαρμογή για κάθε συσκευή. Πλέον, η εφαρμογή του συστήματος είναι ευέλικτη και μπορούν να συνδεθούν σε αυτό πολλαπλού τύπου συσκευές. Επιπλέον, το νέο σύστημα μπορεί να λαμβάνει αλλά και να αναλύει δεδομένα σχετικά με την απόδοση του συστήματος θέρμανσης. Βάσει των δεδομένων αυτών, μάλιστα, το σύστημα είναι σε θέση να υπολογίσει αν η πρωτύτρη εκκίνηση, σε σχέση με τον προγραμματισμένο χρόνο έναρξης, του συστήματος θέρμανσης είναι απαραίτητη. Εάν αυτό ισχύει, μπορεί ακόμα να καθορίσει πόσο νωρίτερα πρέπει να ξεκινήσει. Στην ουσία, η ενέργεια αυτή επιτελείται με στόχο την επίτευξη της επιθυμητής θερμοκρασίας την χρονική στιγμή που έχει καθορίσει ο χρήστης.

1.3 Σύνοψη της Διπλωματικής Εργασίας

Η περιγραφή του σχεδιασμού και της κατασκευής ενός έξυπνου σπιτιού αποτελεί την κεντρική ιδέα του κείμενου αυτού. Το στελεχώνουν, επίσης, οι τεχνολογίες που αξιοποιήθηκαν και σε υλικό και σε λογισμικό επίπεδο. Τελευταίο, αλλά σημαντικό ωστόσο, είναι το κομμάτι στο οποίο παρατίθενται οι μελλοντικές επεκτάσεις για την βελτίωση του υπάρχοντος συστήματος.

Το τρέχων κεφάλαιο αποτελεί την εισαγωγή στο θέμα το οποίο πραγματεύεται η διπλωματική εργασία καθώς και τον λόγο για τον οποίο προέκυψε.

Στο επόμενο κεφάλαιο γίνεται πλήρη καταγραφή και ανάλυση των τεχνολογιών που χρησιμοποιήθηκαν κατά την διάρκεια της κατασκευής του συστήματος, προκειμένου να προσαρμοστεί ο αναγνώστης στο γνωστικό αντικείμενο. Χρήσιμη κρίνεται και η αποσαφήνιση θεωρητικών εννοιών και ορολογιών που ακολουθεί.

Στο τρίτο κεφάλαιο ξεκινάει η ουσιαστική περιγραφή της εργασίας. Σε αυτό καταγράφονται οι απαιτήσεις που πρέπει να ικανοποιεί το σύστημα και, παράλληλα, γίνεται μια ανάλυση αυτών. Στη συνέχεια παρουσιάζεται εκτενώς η βάση δεδομένων που προέκυψε, βάσει των απαιτήσεων. Τέλος, ορίζεται το πρωτόκολλο επικοινωνίας των διάφορων συσκευών με την εφαρμογή.

Στο τέταρτο κεφάλαιο σημειώνονται διεξοδικά οι πληροφορίες για τα τεχνικά χαρακτηριστικά κάθε υλικού εξαρτήματος, που χρησιμοποιήθηκε, καθώς και αναπαραστάσεις αυτών. Στο τέλος του κεφαλαίου αναλύεται η ροή της υλοποίησης της συσκευής και η τελική μορφή της.

Στο πέμπτο αναπτύσσεται η δημιουργία του λογισμικού μέρους του ιστοχώρου και της συσκευής του συστήματος. Ειδικότερα, παρουσιάζονται διεξοδικά οι λειτουργίες της ιστοσελίδας, η λειτουργία του 'έξυπνου θερμοστάτη' όπως επίσης και η ανάπτυξη του λογισμικού για τον μικρο-ελεγκτή.

Το τελευταίο κεφάλαιο αποτελεί τον επίλογο του σώματος του κείμενου. Σε αυτό συνοψίζονται επιγραμματικά οι λειτουργίες και τα πλεονεκτήματα του συστήματος, τα προβλήματα που στάθηκαν προσωρινά εμπόδια στην περάτωση του και τέλος οι πιθανές βελτιώσεις.

Κεφάλαιο 2

Θεωρητικό υπόβαθρο

Στόχος αυτού του κεφαλαίου είναι η διασαφήνιση εννοιών, ορολογιών, εργαλείων και πρωτοκόλλων, στα οποία βασίστηκε η παρούσα διπλωματική εργασία. Τα κυριότερα σημεία είναι η ανάλυση του προγραμματισμού διαδικτύου, η παρουσίαση της αρχιτεκτονικής του Arduino, η περιγραφή των πλατφορμών ανάπτυξης λογισμικού και η παρουσίαση λογισμικών σχεδίασης ηλεκτρονικών κυκλωμάτων και επεξεργασίας εικόνων.

2.1 Προγραμματισμός διαδικτύου

Ο προγραμματισμός διαδικτύου (Web Development) [37] είναι ένας όρος ο οποίος την περίοδο συγγραφής της διπλωματικής εργασίας χρησιμοποιούνταν κατά κόρον και φαίνεται ότι θα χρησιμοποιηθεί ακόμα περισσότερο. Ο όρος αυτός δεν είναι κάποιο μεμονωμένο εργαλείο ή κάποιο πρόγραμμα λογισμικού αλλά συμπεριλαμβάνει πολλαπλές τεχνολογίες και εργαλεία τα οποία πρέπει να συνυπάρξουν μεταξύ τους ώστε να επιτευχθεί το επιθυμητό αποτέλεσμα. Το τελευταίο αναφέρεται στις ενέργειες που πρέπει να ακολουθηθούν για την δημιουργία μιας ιστοσελίδας (web site).

Ο προγραμματισμός διαδικτύου μπορεί να διαιρεθεί στις εξής τρεις κατηγορίες: στον υπολογιστή εξυπηρετητή (web server), στον υπολογιστή πελάτη (web client) και τέλος στο σύστημα διαχείρισης βάσεων δεδομένων (Database Management System ή DBMS).

Ο όρος web server [38] αναφέρεται σε εκείνον τον υπολογιστή ο οποίος είναι υπεύθυνος για την διεκπεραίωση των αιτημάτων που του θέτουν οι web clients. Σε αυτόν τον υπολογιστή εκτελούνται γλώσσες προγραμματισμού εξυπηρετητή (server-

side programming) για την παραγωγή HTML εγγράφων. Είναι επίσης υπεύθυνος και για την αποστολή μερικών ακόμα εγγράφων, πέραν των HTML, τα οποία θα περιγράφουν στην συνέχεια.

Η έννοια web client αναφέρεται στον υπολογιστή του χρήστη στον οποίο συνήθως εκτελείται ένα πρόγραμμα περιήγησης στο διαδίκτυο (web browser). Ο περιηγητής δημιουργεί και αποστέλλει αιτήματα προς τον web server και στην συνέχεια παραλαμβάνει τις απαντήσεις τις οποίες εν τέλει εμφανίζει στον χρήστη. Μία ακόμα αρμοδιότητα του browser είναι να εκτελεί γλώσσες προγραμματισμού πελάτη (client-side programming).

Το σύστημα βάσης δεδομένων [12] θεωρείται μια ξεχωριστή κατηγορία παρόλο που μπορεί να εκτελείται και στον υπολογιστή εξυπηρετητή. Ναι μεν είναι αναπόσπαστο κομμάτι του προγραμματισμού διαδικτύου αλλά είναι δε ένα σύστημα γενικού σκοπού, το οποίο χρησιμοποιείται σε πολλές άλλες εφαρμογές οι οποίες δεν είναι απαραίτητα ιστοσελίδες. Τα DBMS είναι υπεύθυνα για την αποθήκευση και την ανάκτηση τις δυναμικής πληροφορίας που προέρχεται από τους χρήστες και από συσκευές καταγραφής δεδομένων. Χαρακτηριστικό παράδειγμα αποτελεί μια συσκευή θερμομέτρου που, ανά τακτά χρονικά διαστήματα, στέλνει στον server την τρέχουσα θερμοκρασία του χώρου στον οποίο βρίσκεται.

2.1.1 HTML

Η HTML (HyperText Markup Language ή Γλώσσα Σήμανσης Κειμένου) [19] είναι η βασική γλώσσα σήμανσης στο διαδίκτυο για την δημιουργία μιας ιστοσελίδας. Η HTML δεν θεωρείται γλώσσα προγραμματισμού αλλά είναι ένα έγγραφο με οδηγίες προς τους browsers για το πως να εμφανίσουν το περιεχόμενο στους χρήστες (markup) [22]. Όπως είναι φανερό η HTML “εκτελείται” στον υπολογιστή πελάτη μέσω του browser. Τα βασικά δομικά συστατικά της HTML είναι τα HTML στοιχεία (HTML elements) τα οποία οριοθετούνται από ετικέτες (tags).

Τα tags είναι λέξεις ή γράμματα που περικλείονται από γωνιώδεις αγκύλες και υπάρχουν ανά ζεύγη. Δηλαδή, υπάρχει ένα tag που σηματοδοτεί την έναρξη και ένα που σηματοδοτεί την λήξη με μερικές εξαιρέσεις. Το tag λήξης περιέχει μία πλάγια γραμμή ‘/’ για να διαχωρίζετε από το tag έναρξης. Ένα παράδειγμα tag είναι το εξής: <html>...</html>. Τα tags μέσα τους περικλείουν κείμενο ή και άλλα

tag (εμφωλευμένα tag). Μερικά βασικά tags παρατίθενται στην εικόνα 2.1.

HTML Στοιχείο	Περιγραφή
<html>...</html>	Είναι το βασικό HTML tag, το οποίο πρέπει να περι- κλείει ολόκληρο των HTML κώδικα
<!DOCTYPE>	Αποτελεί οδηγία προς τον browser σχετικά με την έκ- δοση της HTML που χρησιμοποιούμε
<head>...</head>	Ορίζει πληροφορίες σχετικά με την γλώσσα και την κω- δικοποίηση του εγγράφου και γενικότερα εδώ υπάρ- χουν τα μεταδεδομένα του
<title>...</title>	Τροποποιεί τον τίτλο της ιστοσελίδας
<body>...</body>	Εδώ βρίσκεται το κυρίως κείμενο το οποίο ο browser θα επεξεργαστεί και θα δείξει στον χρήστη
...	Η ετικέτα αυτή ομαδοποιεί τα στοιχεία που βρίσκονται μέσα σε αυτή και μπορούν να επεξεργαστούν συνολικά
<form>...</form>	Χρησιμοποιείται για να σηματοδοτήσει την χρήση της φόρμας για την συλλογή δεδομένων
<input>...</input>	Αποτελεί υποστοιχείο της φόρμας, σε αυτό ο χρήστης μπορεί να εισάγει κείμενο
<button>...</button>	Με την χρήση του δημιουργείται ένα κουμπί μέσα στην ιστοσελίδα το οποίο χρησιμοποιείται για την εκκίνηση ενεργειών
<a>...	Μετατρέπει το κείμενο που περιλαμβάνει σε υπερσύνδε- σμο

Πίνακας 2.1: Παρουσίαση βασικών στοιχείων HTML

Η HTML δημιουργεί τη βασική δομή μιας ιστοσελίδας με αποτέλεσμα να υπάρ-
χουν και άλλες τεχνολογίες που την συμπληρώνουν διαμορφώνοντάς μια πιο ευπα-
ρουσίαστη ιστοσελίδα, όπως η CSS. Από την άλλη η Javascript βελτιώνει την λει-
τουργικότητα της.

2.1.2 CSS

Η CSS (Cascading Style Sheets ή Διαδοχικά Φύλλα Στυλ) [10] χρησιμοποιείται
για να περιγράψει τον τρόπο με τον οποίο το HTML έγγραφο θα εμφανιστεί στην
οθόνη του χρήστη, στην εκτύπωση, σε οθόνες διαφορετικού μεγέθους αλλά και σε
διάφορες άλλες περιπτώσεις. Η CSS δημιουργήθηκε για να διαχωριστούν οι εντολές
τις εμφάνισης από τις εντολές του περιεχομένου μιας ιστοσελίδας. Η CSS βοηθάει
στο να δημιουργηθεί μια ομορφότερη ιστοσελίδα, οπτικά. “Εκτελείται”, επίσης, από
τον browser του χρήστη.

Το συντακτικό της είναι σχετικά απλό. Αποτελείται από μία λέξη ή ένα γράμμα

τα οποία υποδηλώνουν ένα συγκεκριμένο στοιχείο της HTML και αυτό ονομάζεται επιλογήας (css selector). Έπειτα, ακολουθούν μία ή περισσότερες εντολές που περιλαμβάνονται μέσα σε αγκύλες τύπου αγκίστρου (curly braces). Κάθε εντολή βρίσκεται στην δική της γραμμή όπου και τερματίζεται με το ελληνικό ερωτηματικό. Τέλος, η κάθε εντολή αποτελείται από δύο μέρη, την ιδιότητα, η οποία δείχνει ποια ιδιότητα του HTML στοιχείου θα τροποποιηθεί και την τιμή, που θα λάβει η τρέχουσα ιδιότητα. Η ιδιότητα διαχωρίζεται από την τιμή με την άνω-κάτω τελεία. Στην εικόνα 2.1 απεικονίζεται ένα παράδειγμα σύνταξης CSS.



Εικόνα 2.1: Παράδειγμα σύνταξης CSS

2.1.3 JavaScript

Η Javascript (JS) [20] είναι μια γλώσσα προγραμματισμού σεναρίου (scripting language) η οποία εκτελείται από τον browser, στον υπολογιστή πελάτη δηλαδή. Η JS αποτελεί υλοποίηση του προτύπου ECMAScript και δεν είναι η καθαυτή γλώσσα. Ως scripting γλώσσα απαιτεί την ύπαρξη κάποιας μηχανής εκτέλεσης, κάτι το οποίο αναλαμβάνει να το κάνει ο browser. Πλέον όλοι οι σύγχρονοι browsers μπορούν να εκτελούν την JS χωρίς την ανάγκη κάποιου προσθέτου.

Η JS χρησιμοποιείται για να βελτιωθεί η εμπειρία χρήσης. Μπορεί να ανανεώνει δυναμικά τα κομμάτια μιας ιστοσελίδας χωρίς να χρειάζεται η πλήρης ανανέωση της, γεγονός που απαιτεί περισσότερο χρόνο και είναι εμφανές στον χρήστη. Επίσης, η JS μπορεί να ζητήσει δεδομένα στο παρασκήνιο από τον web server χωρίς να διαγράψει το περιεχόμενο που κατά πάσα πιθανότητα ο χρήστης διαβάζει και μόλις αυτό είναι διαθέσιμο το εισάγει δυναμικά στην τρέχουσα ιστοσελίδα. Ουσιαστικά η JS έχει την δυνατότητα να αφαιρεί και να προσθέτει HTML στοιχεία, CSS κανόνες,

και να τροποποιεί συγκεκριμένες ιδιότητες των HTML στοιχείων μεταβάλλοντας τις τιμές τους.

Τέλος, αξίζει να σημειωθεί ότι, ενώ η JavaScript ήταν καθαρά μια γλώσσα προγραμματισμού που εκτελούνταν αυστηρά στον client, πλέον έχει αρχίσει να χρησιμοποιείται και στην πλευρά των υπολογιστών εξυπηρετητών.

2.1.4 MySQL

Η MySQL [26] είναι ένα πρόγραμμα λογισμικού το οποίο συγκαταλέγεται στην κατηγορία των σχεσιακών συστημάτων διαχείρισης βάσεων δεδομένων (RDBMS). Μια βάση δεδομένων αποτελεί τον πυρήνα κάθε συστήματος, μιας και εκεί αποθηκεύεται η σημαντική πληροφορία. Η βάση δεδομένων είναι αναπόσπαστο κομμάτι μιας εφαρμογής ιστοσελίδας. Παρόλ' αυτά συνιστά ξεχωριστή κατηγορία γιατί είναι πολύπλοκη, και επειδή σχετίζεται με άλλες εφαρμογές εκτός του προγραμματισμού διαδικτύου.

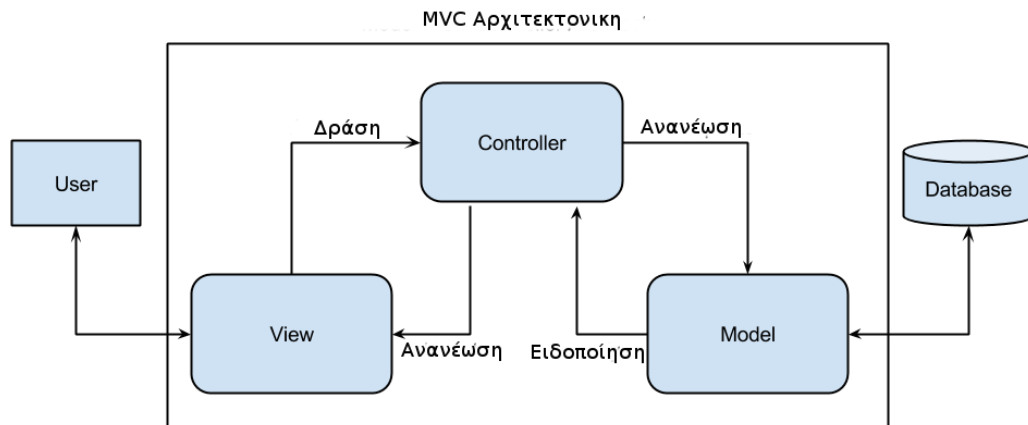
Ένα DBMS είναι υπεύθυνο για την αποδοτική αποθήκευση των δεδομένων σε κάποιο μέσο μόνιμης αποθήκευσης με σκοπό την όσο το δυνατόν πιο γρήγορη ανάκτηση τους όταν αυτή ζητηθεί. Η επικοινωνία με μια βάση δεδομένων επιτυγχάνεται μέσω ενδιάμεσων προγραμμάτων, τους οδηγούς (drivers). Οι τελευταίοι λαμβάνουν μια συγκεκριμένη μορφή εντολών, τις επεξεργάζονται κατάλληλα και τις μεταφράζουν σε γλώσσα κατανοητή από την βάση δεδομένων. Στην συνέχεια τις αποστέλλουν και έπειτα αναμένουν την πληροφορία την οποία μεταφέρουν στον αιτούντα. Οι εντολές αυτές ανήκουν στην γλώσσα SQL (Structured Query Language). Πάραδειγμα τέτοιας εντολής βρίσκεται στην απεικόνιση 2.1.

SELECT email , name **FROM** users ;

Απεικόνιση 2.1: Παράδειγμα SQL ερωτήματος

2.1.5 Ruby

Η Ruby [31] είναι μία αντικειμενοστραφής γλώσσα προγραμματισμού, γενικού σκοπού. Είναι, μια διερμηνευόμενη γλώσσα προγραμματισμού, πράγμα που σημαίνει ότι απαιτείται η ύπαρξη του κατάλληλου διερμηνευτή. Η Ruby ανήκει, και στην κατηγορία, εκείνων των γλωσσών προγραμματισμού που χρησιμοποιούνται από τους web servers για την παραγωγή δυναμικού HTML κώδικα.



Εικόνα 2.2: Σχεδιάγραμμα λειτουργίας της MVC αρχιτεκτονικής

Στο οικοσύστημα της Ruby έχουν αναπτυχθεί αρκετά κομμάτια κώδικα, βιβλιοθήκες δηλαδή, τα οποία είναι διαθέσιμα σε όσους θέλουν να τα χρησιμοποιήσουν. Στον κόσμο της ruby η κάθε βιβλιοθήκη ή αλλιώς πακέτο ονομάζεται 'gem' και ο διαχειριστής των πακέτων 'RubyGems' [33]. Ο διαχειριστής των πακέτων είναι υπεύθυνος για την αυτοματοποιημένη λήψη των gems τοπικά στο σύστημα μας.

2.2 Η αρχιτεκτονική Model-View-Controller

Η Model-View-Controller (Μοντέλο-Ελεγκτής-Εμφάνιση ή MVC) [24] αρχιτεκτονική είναι ένα μοντέλο ανάπτυξης λογισμικού το οποίο χρησιμοποιείται ευρέως. Το μοντέλο ορίζει τον διαχωρισμό του λογισμικού σε τρία πλήρως διασυνδεδεμένα μέρη, τα οποία όμως μπορούν να υλοποιηθούν παράλληλα και από διαφορετικές ομάδες προγραμματιστών. Το κάθε μέρος είναι κατασκευασμένο με τέτοιο τρόπο ώστε να χειρίζεται ένα συγκεκριμένο κομμάτι της εφαρμογής. Στην εικόνα 2.2 παρουσιάζετε η αλληλεπίδραση μεταξύ των μερών σε ένα σύστημα βασισμένο στην MVC.

2.2.1 Model

Το μοντέλο (Model) αναλαμβάνει την παροχή και αποθήκευση των δεδομένων της εφαρμογής με έναν απλοποιημένο τρόπο. Προσπαθεί να κρύψει την πολυπλοκότητα της επικοινωνίας με τα διάφορα συστήματα αποθήκευσης και ανάκτησης δεδομένων, όπως είναι μια βάση δεδομένων. Επίσης, το μοντέλο ελέγχει αν τα δεδομένα είναι έγκυρα βάσει κανόνων που έχουν οριστεί και σε αυτό περιγράφονται οι συσχετίσεις

με άλλα μοντέλα.

2.2.2 View

Στην εμφάνιση (View) βρίσκεται η λογική για την παρουσίαση των δεδομένων στον χρήστη. Σε αυτό το κομμάτι βρίσκεται ο κώδικας που παράγει την HTML η οποία θα αποσταλεί στον browser.

2.2.3 Controller

Όσον αφορά τον ελεγκτή, (Controller) αυτός είναι που αναλαμβάνει να διασυνδέσει όλα τα μέρη μεταξύ τους. Στον controller τοποθετείται κώδικας ο οποίος ζητάει δεδομένα από τα μοντέλα και τα δίνει στα views για εμφάνιση, όταν αυτά είναι έτοιμα. Οι controllers αποτελούνται από ενέργειες (actions), οι οποίες είναι συναρτήσεις/μέθοδοι της εκάστοτε γλώσσας προγραμματισμού. Τέλος, ο χρήστης αλληλεπιδρά, δηλαδή ζητάει τα δεδομένα που επιθυμεί, μέσω των actions.

2.3 Βιβλιοθήκες, frameworks, πρωτόκολλα και τεχνικές

Οι βιβλιοθήκες λογισμικού είναι κομμάτια κώδικα τα οποία κάποιος έγραψε και τα διέθεσε στους υπόλοιπους. Παράδειγμα μιας πολύ απλής βιβλιοθήκης θα μπορούσε να είναι μια συνάρτηση για τον υπολογισμό του εκθέτη. Κάτι πιο σύνθετο θα μπορούσε να είναι η εισαγωγή χαρακτήρων από τον χρήστη στο πρόγραμμα μέχρι κάτι που μπορεί να είναι ακόμα πιο σύνθετο και χρειάζεται περισσότερο χρόνο για την συγγραφή του. Ο κώδικας των βιβλιοθηκών χρησιμοποιείται για την ταχύτερη ανάπτυξη λογισμικού μιας και δεν χρειάζεται η επανασυγγραφή τμημάτων κώδικα που έχουν ξαναγραφεί και είναι ίδια.

Με την έννοια τεχνικές ορίζουμε την συνεργασία τεχνολογιών προγραμματισμού για την επίτευξη κάποιου στόχου. Μια τέτοια τεχνική η οποία χρησιμοποιήθηκε στην παρούσα διπλωματική είναι η AJAX.

Από την άλλη, ο όρος πρωτόκολλο αναφέρεται σε ένα σύνολο κανόνων οι οποίοι έχουν προτυποποιηθεί από κάποια επίσημη αρχή και χρησιμοποιούνται ως κοινός γνώμονας για την επιτυχή επικοινωνία διαφορετικών συστημάτων, οντοτήτων, κ.τ.λ.

Τέλος, η έννοια Framework αναφέρεται σε μια οικογένεια εργαλείων, βιβλιοθηκών και τεχνικών που χρησιμοποιούνται ταυτόχρονα για την δημιουργία μιας

εφαρμογής. Ένα framework, μπορεί να ορίζει βασικούς κανόνες για τον τρόπο προγραμματισμού, όπως προγραμματισμός με την τεχνική MVC. Ακόμα, μπορεί να περιέχει κώδικα για τετριμμένα θέματα τα οποία είναι κοινά και επαναλαμβάνονται σχεδόν σε κάθε εφαρμογή.

2.3.1 CSS προεπεξεργαστές

CSS προ-επεξεργαστής (CSS preprocessor) είναι το εργαλείο το οποίο βοηθά στην συγγραφή κώδικα CSS. Ένας CSS preprocessor χρησιμοποιεί εμφωλευμένες εντολές με αποτέλεσμα να αποφεύγει τον επαναλαμβανόμενο κώδικα. Επίσης, επιτρέπει την χρήση μεταβλητών και συναρτήσεων. Όλα αυτά έχουν ως αποτέλεσμα ο κώδικας CSS να είναι πιο καθαρός και ευανάγνωστος. Στην συνέχεια ο κώδικας αυτός περνάει από ένα κατάλληλο εργαλείο μετατροπής, για την μετατροπή του σε κοινό CSS κώδικα τον οποίο οι browsers αναγνωρίζουν. Παράδειγμα τέτοιων εργαλείων είναι το SASS και το LESS.

2.3.2 Bootstrap

Το bootstrap είναι ένα framework HTML και CSS με επεκτάσεις JavaScript, η οποία διευθετεί πολλά θέματα εμφάνισης για φόρμες, κουμπιά και άλλα. Το bootstrap περιέχει σύνολο κανόνων CSS για διάφορα HTML στοιχεία προσδίδοντας τους μια καλύτερη εμφάνιση. Επίσης περιέχει βιβλιοθήκες javascript για την δυναμική αλλαγή της εμφάνισης των ιστοσελίδων. Το bootstrap είναι μία πολύ καλή επιλογή για μια καινούρια εφαρμογή διαδικτύου διότι μπορεί να δώσει μια ωραία εμφάνιση στην ιστοσελίδα χωρίς να χρειάζεται η συγγραφή καθόλου CSS.

2.3.3 AJAX

Η τεχνική AJAX (Asynchronous Javascript and XML) είναι ένα σύνολο τεχνολογιών διαδικτύου, που χρησιμοποιούνται από τον browser, για την δημιουργία μιας ασύγχρονης επικοινωνίας μεταξύ του browser και του server. Με την χρήση της ajax οι εφαρμογές μπορούν να στέλνουν και να λαμβάνουν δεδομένα από έναν server ασύγχρονα στο παρασκήνιο χωρίς να επεμβαίνουν στην λειτουργία και στην εμφάνιση της υπάρχουσας σελίδας. Η τεχνική αυτή δίνει την δυνατότητα της δυναμικής ανανέωσης μεμονωμένων κομματιών της ιστοσελίδας χωρίς να είναι απαραίτητο

να ανανεωθεί ολόκληρη η ιστοσελίδα. Τα δεδομένα ανταλλάσσονται κατά βάση σε μορφή JSON, την οποία η Javascript υποστηρίζει εγγενώς.

2.3.4 Ruby on Rails

Η Ruby on Rails ή απλά Rails [32, 18] είναι ένα server-side framework το οποίο έχει γραφεί σε ruby και χρησιμοποιείται στον προγραμματισμό διαδικτύου για την δημιουργία δυναμικών ιστοσελίδων. Η rails βασίζεται στην MVC αρχιτεκτονική για την οργάνωση του κώδικα. Επίσης, δίνει μεγάλη έμφαση σε δύο προγραμματιστικές αρχές, στην “Συμβατικότητα κατά την διαμόρφωση” (Convention over Configuration ή CoC) και στην “Μη επαναληψιμότητας στοιχείων” (Don’t Repeat Yourself ή DRY).

Η CoC σημαίνει πως το framework έχει φροντίσει να πάρει αποφάσεις σχετικά με βασικές και κοινότυπες ρυθμίσεις που κάθε προγραμματιστής καλείται να λάβει στην αρχή ενός project. Ένα τέτοιο παράδειγμα είναι η απόφαση για την ονοματολογία των πινάκων και των στηλών σε μια βάση δεδομένων.

Η αρχή του DRY, έχει ως σκοπό την μείωση των επαναλαμβανόμενων στοιχείων της εφαρμογής, με αποτέλεσμα την ευκολότερη κατανόηση του κώδικα και τον συντομότερο έλεγχο. Έτσι, σε περίπτωση που απαιτηθεί μια αλλαγή στην εφαρμογή αυτή θα χρειαστεί να γίνει σε ένα μόνο σημείο. Αυτό σημαίνει πως θα υπάρξουν λιγότερα προβλήματα και ο χρόνος ελέγχου για την επιτυχή αλλαγή θα είναι σημαντικά μικρότερος.

Η rails παρέχει εργαλεία για την αυτοματοποιημένη δημιουργία κώδικα μοντέλων, εμφανίσεων και ελεγκτών. Σκοπός τους είναι η γρήγορη εκκίνηση του προγραμματιστή, μιας και παρέχουν λειτουργικό κώδικα με τον οποίο μπορούν να εισάγουν νέα δεδομένα, να τα ανανεώσουν ή ακόμα και να τα διαγράψουν. Τα αποτελέσματα του αυτοματοποιημένου κώδικα φαίνονται άμεσα στον browser.

Παρέχεται, ακόμα, η δυνατότητα για την δημιουργία της βάσης δεδομένων μέσω οδηγιών, που βρίσκονται μέσα σε αρχείο και εκτελούνται από το κατάλληλο εργαλείο που παρέχει η rails. Η διαδικασία αυτή είναι γνωστή ως ‘database migration’.

Ένα ακόμα σημαντικό χαρακτηριστικό, είναι τα εργαλεία τα οποία παρέχονται με σκοπό τον αυτοματοποιημένο έλεγχο του κώδικα από διάφορες οπτικές γωνίες.

2.3.5 MQTT

Το MQTT (Message Queue Telemetry Transport) [25] είναι ένα πρωτόκολλο επικοινωνίας βασισμένο στο μοντέλο δημοσιοποίησης / εγγραφής (publish / subscribe pattern [30]), το οποίο είναι πολύ “ελαφρύ” και αποδοτικό. Ο όρος “ελαφρύ” αναφέρεται στο γεγονός ότι το μέγεθος των πακέτων και η πληροφορία που αποστέλλεται αποτελούν τα απολύτως απαραίτητα και τα όσο το δυνατόν λιγότερα. Σχεδιάστηκε για την επικοινωνία μεταξύ συσκευών σε απομακρυσμένες περιοχές που πιθανώς να μην έχουν καλή συνδεσιμότητα με το διαδίκτυο και που απαιτούν ελαχιστοποίηση της κατανάλωσης ενέργειας. Παρ’ όλα αυτά, έχει σχεδιαστεί για να αντέχει σε περιπτώσεις χαμηλής ποιότητας σύνδεσης και σε μεγάλες καθυστερήσεις του δικτύου. Τέλος, η αποδοτική του λειτουργία το καθιστά ιδιαίτερα ελκυστικό για χρήση σε συσκευές μικρού μεγέθους με περιορισμένες δυνατότητες. Χαρακτηριστικό παράδειγμα αποτελούν οι μικρο-ελεγκτές. Στα πλαίσια της διπλωματικής εργασίας το πρωτόκολλο αυτό χρησιμοποιήθηκε για την επίτευξη της αποστολής δεδομένων από την εφαρμογή προς τις συσκευές.

2.4 Αρχιτεκτονική Arduino

Το Arduino [3] είναι μια υπολογιστική πλατφόρμα ανοιχτού υλικού (open-source hardware) [28] η οποία είναι μικρή σε μέγεθος και βασίζεται στην τεχνολογία των μικρο-ελεγκτών (microcontrollers) [23]. Διαθέτει αναλογικές και ψηφιακές εισόδους/εξόδους (analog and digital input/output) για την λήψη δεδομένων από το πραγματικό περιβάλλον (sensors) και αντιδρά μέσω των συσκευών ανάδρασης (actuators). Η αρχιτεκτονική-σχεδιασμός του Arduino είναι ανοιχτού κώδικα, κάτι που σημαίνει ότι ο καθένας μπορεί να έχει πρόσβαση στα σχέδια του Arduino ώστε να το ανακατασκευάσει. Το Arduino χρησιμοποιεί κατά βάση του μικρο-ελεγκτές ATMEΛ, με κάποια καινούρια μοντέλα να χρησιμοποιούν και συνδυασμό επεξεργαστών ARM.

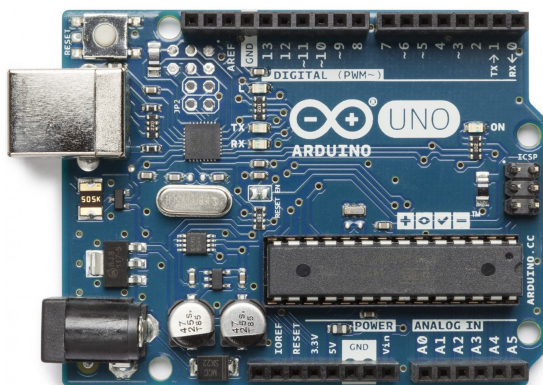
Επίσης, ο όρος του Arduino περιλαμβάνει και το λογισμικό κομμάτι, δηλαδή το περιβάλλον ανάπτυξης κώδικα για το Arduino το οποίο ονομάζεται Arduino IDE. Το Arduino ide είναι και αυτό opensource. Η γλώσσα προγραμματισμού του Arduino είναι η C και η C++.

Το Arduino διατίθεται σε αρκετές παραλλαγές. Η καθεμία, ωστόσο, έχει τις δικές

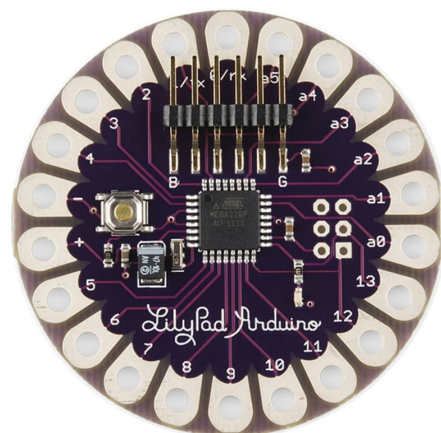
της ιδιαιτερότητες και διαφοροποιήσεις από τις άλλες. Οι διαφορές εμφανίζονται στις επιδόσεις του κάθε συστήματος, στο πλήθος των εισόδων/εξόδων που διαθέτουν και σε χαρακτηριστικά που κάποιες έχουν και κάποιες άλλες όχι. Τέλος, σε κάποια συστήματα Arduino μπορούμε να συνδέσουμε πλακέτες οι οποίες επεκτείνουν τις δυνατότητες του εκάστοτε συστήματος Arduino, γνωστά και ως Shields.

2.4.1 Είδη Arduino

Ξεκινώντας από τις μικρότερες σε μέγεθος και δυνατότητες εκδόσεις Arduino, υπάρχουν τα Arduino Nano εικόνα 2.5 και LilyPad εικόνα 2.4. Ακολουθεί το Arduino Uno, εικόνα 2.3, το οποίο είναι η δημοφιλέστερη και η πιο συχνά χρησιμοποιούμενη έκδοση του Arduino σε όλα τα project. Το Arduino Ethernet πρόκειται για μια κλασική συσκευή Arduino Uno. Σε αυτό έχει ενσωματωθεί το Ethernet Shield με αποτέλεσμα να συνυπάρχουν σε μία κοινή μητρική πλακέτα. Έπειτα υπάρχει το Arduino Mega εικόνα 2.6 το οποίο είναι στην ουσία ένα Arduino Uno στο οποίο έχουν αυξηθεί οι επιδόσεις, οι μνήμες και οι εισοδοί/εξοδοί. Τέλος υπάρχουν εκδόσεις, όπως το Arduino Yun, οι οποίες εκτός του βασικού atmel μικρο-ελεγκτή είναι εξοπλισμένες και με άλλους επεξεργαστές, όπως τους ARM που αναφέρθηκαν προηγουμένως. Τέτοιες εκδόσεις μπορούν να τρέξουν ακόμα και το λειτουργικό σύστημα linux.



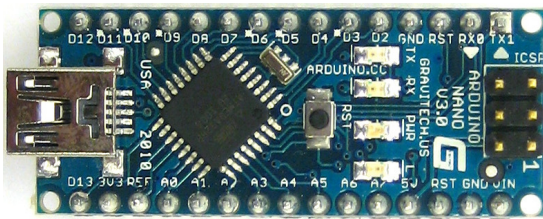
Εικόνα 2.3: Πλακέτα Arduino UNO



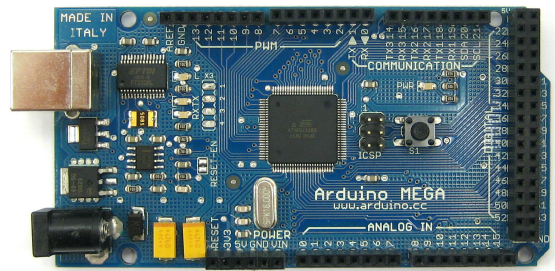
Εικόνα 2.4: Πλακέτα Arduino LilyPad

2.4.2 Arduino Shields

Τα Arduino Shields είναι πλακέτες οι οποίες περιέχουν υλικό που επεκτείνει τις δυνατότητες του βασικού συστήματος. Ένα δημοφιλές παράδειγμα επέκτασης είναι

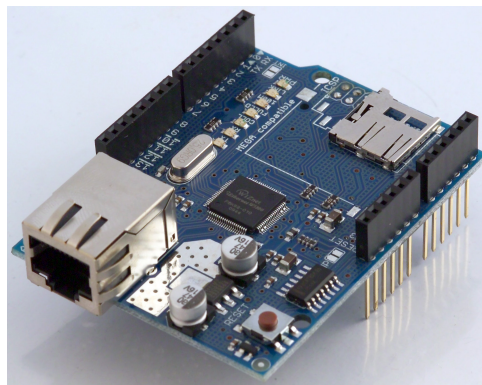


Εικόνα 2.5: Πλακέτα Arduino Nano



Εικόνα 2.6: Πλακέτα Arduino Mega

η προσθήκη μια πλακέτας Ethernet Shield, εικόνα 2.7, η οποία δίνει την δυνατότητα στο Arduino να αποκτήσει πρόσβαση στο διαδίκτυο.



Εικόνα 2.7: Arduino Ethernet Shield

2.5 Πλατφόρμες Ανάπτυξης Λογισμικού

Στα πλαίσια της υλοποίησης της παρούσας διπλωματικής εργασίας ήταν απαραίτητη και η ανάπτυξη λογισμικού. Χρησιμοποιήθηκαν, επομένως, εργαλεία λογισμικού που διευκόλυναν την συγγραφή αυτού.

Μια βασική ενότητα λογισμικών ανάπτυξης είναι το ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment ή IDE). Το IDE είναι πρόγραμμα λογισμικού. Διαθετεί στον προγραμματιστή ένα περιβάλλον στο οποίο παρέχονται συγκεντρωμένες δυνατότητες οι οποίες είναι απαραίτητες για την συγγραφή λογισμικού. Τα βασικά μέρη ενός IDE είναι ο επεξεργαστής πηγαίου κώδικα, εργαλεία για την αυτοματοποιημένη μεταγλώττιση και αποσφαλματωτές. Τα περισσότερα σύγχρονα IDE διαθέτουν επίσης μηχανισμούς για την αυτόματη συμπλήρωση εντολών κατά την συγγραφή κώδικα, μηχανισμούς για την εύρεση λανθασμένης σύνταξης, περιηγητές κλάσεων και αντικειμένων ακόμα και σύστημα διαχείρισης εκδόσεων (Version Control System ή VCS).

2.5.1 RubyMine

Το RubyMine είναι ένα σύγχρονο IDE το οποίο περιέχει τα περισσότερα από τα χαρακτηριστικά που προαναφέρθηκαν. Το RubyMine εξειδικεύεται σε συστήματα τα οποία χρησιμοποιούν τη ruby, ως γλώσσα προγραμματισμού, και τον προγραμματισμό διαδικτύου.

Το RubyMine δεν είναι ένα λογισμικό δωρεάν και ανοιχτού κώδικα. Ωστόσο, υπάρχουν εκπτώσεις σε ορισμένες κατηγορίες και παρέχεται δωρεάν σε μαθητές, φοιτητές, καθηγητές και ομάδες που κατασκευάζουν λογισμικά ανοιχτού κώδικα.

Τέλος, να τονιστεί πως το RubyMine είναι το μοναδικό IDE που προσφέρει την καλύτερη υποστήριξη για συγγραφή κώδικα σε rails. Αυτός είναι και ο λόγος για τον οποίο επιλέχτηκε.

2.5.2 Atom

Το Atom είναι ένας σύγχρονος και ανοιχτού λογισμικού κειμενογράφος καθώς και επεξεργαστής πηγαίου κώδικα. Οι δυνατότητές του και τα χαρακτηριστικά του μπορούν να εμπλουτίζονται μέσω προσθέτων. Το atom, για παράδειγμα, μπορεί να λειτουργήσει και ως IDE με την χρήση του κατάλληλου προσθέτου. Ένα τέτοιο πρόσθετο είναι το platformio το οποίο μετατρέπει τον Atom σε ένα ολοκληρωμένο περιβάλλον ανάπτυξης για προγραμματισμό σε arduino.

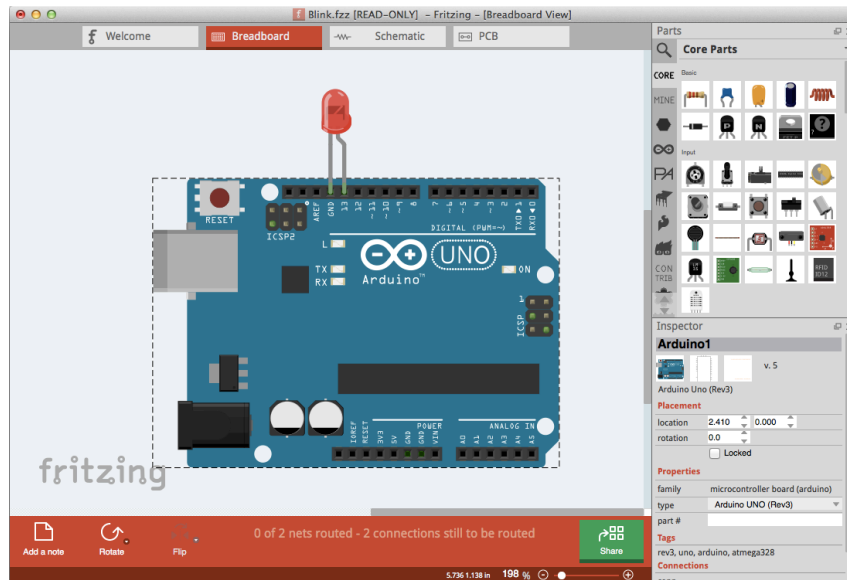
2.6 Λογισμικά σχεδίασης & επεξεργασίας εικόνας

Τα λογισμικά αυτά βοηθούν στην δημιουργία σχηματικών διαγραμμάτων και επεξεργασία των εικόνων.

2.6.1 Fritzing

Το Fritzing [16] είναι ένα πρόγραμμα λογισμικού. Σε αυτό μπορούν να μεταφερθούν οι πρωτότυπες κατασκευές του arduino από breadboard και οποιαδήποτε άλλη ηλεκτρονική κατασκευή. Τα προαναφερθέντα σχέδια χρησιμοποιούνται είτε για τον διαμοιρασμό με άλλους σχεδιαστές είτε και για την εξαγωγή ηλεκτρονικών σχεδιαγραμμάτων. Ακόμα, μπορούν να χρησιμοποιηθούν και για την δημιουργία PCB πλακέτας για την αποστολή σε βιομηχανία δημιουργίας πλακετών. Στην εικόνα 2.8

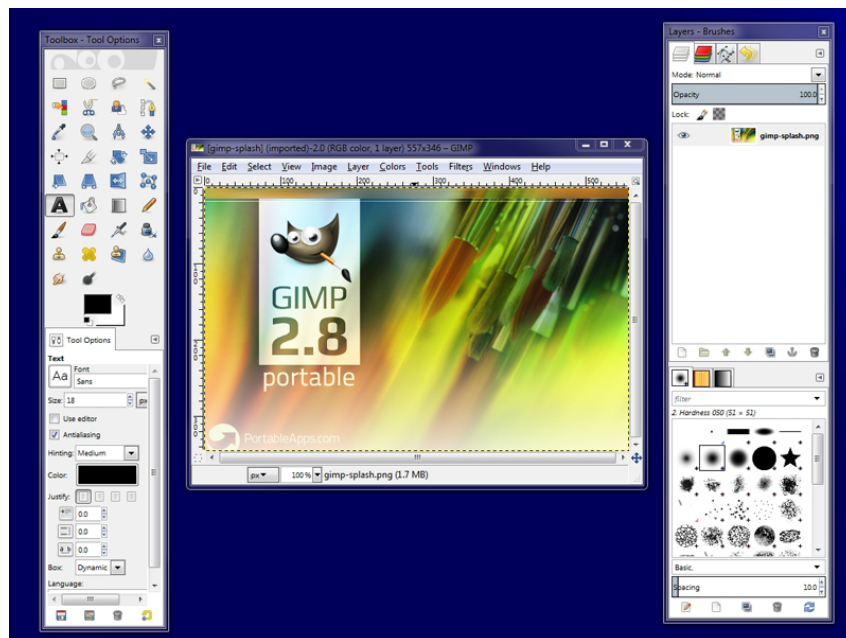
φαίνεται το περιβάλλον σχεδίασης του fritzing. Το Fritzing χρησιμοποιήθηκε για τον σχεδιασμό της πρωτότυπης κατασκευής του arduino και έπειτα για την εξαγωγή του ηλεκτρονικού σχεδίου.



Εικόνα 2.8: Το περιβάλλον σχεδίασης του Fritzing

2.6.2 GIMP

Το GIMP [17] είναι και αυτό ένα πρόγραμμα λογισμικού, το οποίο προσφέρει στον χρήστη την δυνατότητα να επεξεργαστεί ένα αρχείο εικόνας με διάφορους τρόπους. Για παράδειγμα, μπορεί να αφαιρεθεί ένα κομμάτι της εικόνας, να προστεθεί κάποιο κείμενο ακόμα και να διορθωθούν οι χρωματισμοί και η φωτεινότητα της εικόνας. Στην εικόνα 2.9 φαίνεται το περιβάλλον χρήσης του GIMP. Το GIMP χρησιμοποιήθηκε για την επεξεργασία των εικόνων που ενσωματώθηκαν στο παρόν κείμενο.



Εικόνα 2.9: Το περιβάλλον χρήσης του GIMP

Κεφάλαιο 3

Δομικά μέρη συστήματος

Για την ολοκλήρωση της διπλωματικής εργασίας, ήταν απαραίτητη η ανάπτυξη μιας υλικής κατασκευής, η οποία αποτελεί το υλικό μέρος του συστήματος. Το υλικό κομμάτι βέβαια δεν μπορούσε να ολοκληρώσει από μόνο του την εργασία. Έτσι, αναπτύχθηκε και το κατάλληλο λογισμικό μέρος του συστήματος. Το κομμάτι που αφορά το λογισμικό εκτείνεται τόσο στην δημιουργία της εφαρμογής, δηλαδή της κεντρικής διαχειριστικής πλατφόρμας του συστήματος η οποία θα υλοποιηθεί μέσω μιας ιστοσελίδας, όσο και στον προγραμματισμό του Arduino, για την επιτυχή λειτουργία του υλικού μέρους.

Βέβαια, η ανάπτυξη όλων αυτών των μερών δεν υλοποιήθηκε αυθαίρετα αλλά βασίστηκε πάνω σε κάποιες απαιτήσεις. Για το λόγο αυτό, στο παρόν κεφάλαιο θα γίνει η παράθεση των απαιτήσεων του συστήματος που τέθηκαν εξ' αρχής. Θα ακολουθήσει η αναλυτική περιγραφή του σχεδιασμού της βάσης καθώς και του πρωτοκόλλου επικοινωνίας του συστήματος.

3.1 Απαιτήσεις συστήματος

Για την επιτυχή ολοκλήρωση κάθε έργου, ένας πολύ σημαντικός παράγοντας είναι ο σωστός καθορισμός των απαιτήσεων από την αρχική περιγραφή του, μιας και αυτές καθορίζουν την πορεία του έργου.

Συγκεκριμένα, για το παρόν έργο, στόχος είναι η δημιουργία ενός συστήματος για τον έλεγχο των διαφόρων συσκευών μιας οικίας. Αυτό θα μπορεί να γίνεται είτε απομακρυσμένα είτε τοπικά αλλά κυρίως αυτοματοποιημένα, με απώτερο σκοπό την ελαχιστοποίηση κατανάλωσης ενέργειας. Παρακάτω παρατίθενται αναλυτικά οι απαιτήσεις που εξήχθησαν συνοδευόμενες από επεξήγηση.

-
- **Το σύστημα πρέπει να παρέχει μηχανισμό διαχείρισης χρηστών.** Για την χρήση του συστήματος απαραίτητη προϋπόθεση αποτελεί η δημιουργία λογαριασμού χρήστη σε αυτό. Επίσης, θα πρέπει να υπάρχει η δυνατότητα της ανάκτησης κωδικού σε περίπτωση απώλειας του από τον χρήστη.
 - **Το σύστημα θα πρέπει να διαχειρίζεται την καταχώρηση συσκευών.** Το βασικό κομμάτι αυτού του συστήματος είναι οι συσκευές του. Έτσι, πρέπει να υπάρχει ο κατάλληλος μηχανισμός για την επιτυχή σύζευξη του συστήματος με τις συσκευές. Επίσης, μέσα από αυτή τη σύνδεση θα πρέπει να επιτυγχάνεται ο έλεγχος της συσκευής, δηλαδή η αποστολή εντολών από την εφαρμογή προς τις συσκευές. Τέλος, θα πρέπει οι συσκευές να στέλνουν δεδομένα πίσω στην εφαρμογή σχετικά με την τρέχουσα κατάσταση τους, δηλαδή η εφαρμογή θα πρέπει να μπορεί να λάβει δεδομένα ανάδρασης.
 - **Το σύστημα θα πρέπει να δίνει την δυνατότητα στους χρήστες για την δημιουργία προγράμματος.** Τα προγράμματα ρυθμίζονται από τον χρήστη για την εκτέλεση κάποιας ενέργειας στο μέλλον ή με επαναλαμβανόμενο τρόπο. Το πρόγραμμα θα επενεργεί είτε σε μεμονωμένη συσκευή είτε σε ένα πλήθος συσκευών ταυτόχρονα.
 - **Ύπαρξη δυνατότητας για εκτέλεση συγκεκριμένων ενεργειών άμεσα για κάποιο συγκεκριμένο χρονικό διάστημα.** Θα πρέπει το σύστημα να διαθέτει ένα μηχανισμό για την προ-ρύθμιση συγκεκριμένων ενεργειών που θα ενεργοποιούνται με το πάτημα ενός κουμπιού, το οποίο βρίσκεται πάνω στην συσκευή.
 - **Ειδοποίηση χρηστών για την κατάσταση λειτουργίας των συσκευών.** Το σύστημα είναι απαραίτητο να διαθέτει έναν μηχανισμό με τον οποίο θα ελέγχει αν μια συσκευή βρίσκεται σε λειτουργία ή όχι. Σε περίπτωση μη λειτουργίας για ένα ορισμένο χρονικό διάστημα, ο ιδιοκτήτης της συσκευής θα πρέπει να ειδοποιείται για την απώλεια επικοινωνίας με την συσκευή.
 - **Ξεκάθαρο και απλό περιβάλλον χρήσης.** Το περιβάλλον χρήσης του συστήματος, όπως φαίνεται, θα πρέπει να είναι απλό στη χρήση του. Αυτό σημαίνει πως θα πρέπει να μην απαιτούνται εξειδικευμένες γνώσεις και δεξιότητες. Η

απλότητα του να έχει ως αποτέλεσμα όχι μόνο την ευκολία στην χρήση αλλά και να καθοδηγεί τον χρήστη στην εύρεση των λειτουργιών που αναζητεί.

- **Οι συσκευές και η εφαρμογή του συστήματος πρέπει να έχουν πρόσβαση στο διαδίκτυο.** Η επικοινωνία και η μεταφορά δεδομένων μεταξύ συσκευών και εφαρμογής του συστήματος θα πρέπει να μπορεί να γίνει μέσω του διαδικτύου, δηλαδή, με την χρήση του TCP/IP μοντέλου επικοινωνίας. Έτσι συσκευές και εφαρμογή θα πρέπει να έχουν τις κατάλληλες διεπαφές οι οποίες με κάποιο τρόπο θα τους παρέχουν πρόσβασή στο διαδίκτυο.
- **Κρίνεται αναγκαία η εφαρμογή της ασφάλειας σε κάθε πτυχή του συστήματος.** Αυτό σημαίνει πως η επικοινωνία μεταξύ της εφαρμογής και των συσκευών θα πρέπει να υποστηρίζει κάποια μορφή ασφάλειας. Ένα ακόμα σημείο στο οποίο πρέπει να δοθεί έμφαση είναι η επικοινωνία μεταξύ του χρήστη και της εφαρμογής, δηλαδή κατά την πλοήγηση του χρήστη στη ιστοσελίδα.
- **Επεκτάσιμο και συντηρήσιμο.** Οι έννοιες αυτές ορίζουν πως το σύστημα θα πρέπει να κατασκευαστεί, είτε σε επίπεδο κώδικα είτε σε επίπεδο υλικής κατασκευής, με τρόπο εύκολα κατανοητό από κάποιον τρίτο. Στόχος αυτού είναι ο κάθε εμπλεκόμενος με το σύστημα να μπορεί εύκολα να κατανοεί τη ροή των γεγονότων και να επεμβαίνει όπου χρειάζεται. Τέλος πρέπει να τονιστεί πως μία από τις σημαντικές αρχές για την δημιουργία επεκτάσιμου και συντηρήσιμου κώδικα είναι η αποφυγή της επανάληψης κομματιών κώδικα.

3.2 Ανάλυση βάσης δεδομένων

Η βάση δεδομένων κάθε συστήματος αποτελεί τον πυρήνα του, αυτό συμβαίνει και στο παρόν έργο. Βάσει των απαιτήσεων που ορίστηκαν στην ενότητα 3.1 στην σελίδα 31, προέκυψε το σχήμα της βάσης δεδομένων στο οποίο στηρίζεται η υλοποίηση του συστήματος. Παρακάτω παρατίθεται η δομή των πινάκων που χρησιμοποιούνται από αυτό μαζί με μια αναλυτική περιγραφή για κάθε στήλη, κάτω από κάθε πίνακα, για να αποσαφηνιστεί ο λόγος ύπαρξής της.

Στην ανάλυση που θα ακολουθήσει υπάρχουν δεκατρείς πίνακες. Μερικοί από αυτούς αποτελούν τους κεντρικούς πίνακες της βάσης. Αυτοί είναι οι εξής: ο πίνακας διατήρησης χρηστών, ο πίνακας συσκευών, ο πίνακας χρονο-προγραμματισμών,

ο πίνακας κουμπιών άμεσης εκτέλεσης ενεργειών και οι πίνακες της λειτουργίας ‘έξυπνου θερμοστάτη’. Οι υπόλοιποι είναι συμπληρωματικοί πίνακες όλων των πα-
ραπάνω και διατηρούν περαιτέρω πληροφορίες για λογαριασμό των βασικών τους.

Σε κάθε πίνακα η πρώτη στήλη ονομάζεται id, χρησιμοποιείται ως το πρωτεύων κλειδί, έχει τύπο int(10), είναι μη-προσημασμένη (unsigned) και αυξάνεται αυτό-
ματα (auto increment). Για λόγους μη επαναληψιμότητας θα παραληφθεί η παρου-
σίαση της πρώτης στήλης από τις παρακάτω περιγραφές και όπου αυτή η τακτική
παρεκκλίνει θα τονίζεται.

3.2.1 Δομή του πίνακα “users”

Στον πίνακα 3.1 αποτυπώνεται η δομή του πίνακα “users” στον οποίο καταχω-
ρούνται στοιχεία σχετικά με τους χρήστες που έχουν εγγραφεί στην ιστοσελίδα του
συστήματος.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
email	varchar(255)	Όχι	-
encrypted_password	varchar(255)	Όχι	-
reset_password_token	varchar(255)	Ναι	-
reset_password_send_at	datetime	Ναι	-
remember_created_at	datetime	Ναι	-
sign_in_count	int(11)	Όχι	Default 0
current_sign_in_at	datetime	Ναι	-
last_sign_in_at	datetime	Ναι	-
current_sign_in_ip	varchar(255)	Ναι	-
last_sign_in_ip	varchar(255)	Ναι	-
confirmation_token	varchar(255)	Ναι	-
confirmed_at	datetime	Ναι	-
confirmation_send_at	datetime	Ναι	-
unconfirmed_email	varchar(255)	Ναι	-
failed_attempts	int(11)	Όχι	Default 0
unlock_token	varchar(255)	Ναι	-
locked_at	datetime	Ναι	-

Πίνακας 3.1: Πίνακας “users”

Ανάλυση των στηλών του πίνακα “users”:

- **email:** Το email του χρήστη το οποίο χρησιμοποιείται και ως το ‘όνομα χρήστη (username)’ για την είσοδο του στο σύστημα.

-
- **encrypted_password:** Ο κωδικός πρόσβασης του χρήστη σε κρυπτογραφημένη μορφή.
 - **reset_password_token:** Τυχαίο αλφαριθμητικό το οποίο χρησιμοποιείται για την επαναφορά του λογαριασμού του χρήστη σε περίπτωση απώλειας κωδικού πρόσβασης.
 - **reset_password_send_at:** Καταγραφή της χρονικής στιγμής της αποστολής του token επαναφοράς κωδικού.
 - **remember_created_at:** Σε περίπτωση που ο χρήστης επιθυμεί το σύστημα να τον θυμάται, δηλαδή έχει επιλέξει την επιλογή 'remember me', τότε καταγράφεται η χρονική στιγμή του γεγονότος αυτού.
 - **sign_in_count:** Καταγράφεται το πλήθος των εισόδων του χρήστη στο σύστημα από την ημέρα εγγραφής του.
 - **current_sign_in_at:** Χρονική στιγμή της τρέχουσας εισόδου του χρήστη.
 - **last_sign_in_at:** Χρονική στιγμή της τελευταίας εισόδου του.
 - **current_sign_in_ip:** Καταγραφή της ip διεύθυνσης από την οποία ο χρήστης εισήλθε στο σύστημα την τρέχουσα φορά.
 - **last_sign_in_ip:** Η ip διεύθυνση της τελευταίας εισόδου του.
 - **confirmation_token:** Τυχαίο αλφαριθμητικό το οποίο παράγεται κατά την εγγραφή του χρήστη στο σύστημα και αποστέλλεται στο email που δηλώθηκε. Σκοπός αυτού είναι η επιβεβαίωσή της ορθότητας και της κατοχής των στοιχείων που ο χρήστης εισήγαγε.
 - **confirmed_at:** Χρονική στιγμή της επιβεβαίωσης του λογαριασμού του χρήστη.
 - **confirmation_send_at:** Χρονική στιγμή της αποστολής του email για την επιβεβαίωση του λογαριασμού του χρήστη.
 - **unconfirmed_email:** Χρησιμοποιείται στην περίπτωση που ο χρήστης προσπαθήσει να αλλάξει το email του. Το καινούριο email, καταγράφεται εδώ και παραμένει μέχρι την επιβεβαίωσή του, όπου πλέον μεταφέρεται στην στήλη email.

- **failed_attempts:** Καταγραφή, των αποτυχημένων προσπαθειών της εισόδου του χρήστη.
- **unlock_token:** Τυχαιο αλφαριθμητικό, που παράγεται όταν ο χρήστης υπερβεί τον ορισμένο αριθμό αποτυχημένων προσπαθειών εισόδου. Σε αυτή την περίπτωση ο λογαριασμός του χρήστη κλειδώνεται και το token αποστέλλεται στο δηλωθέν email του.
- **locked_at:** Καταγράφεται η χρονική στιγμή που κλειδώθηκε ο λογαριασμός του χρήστη.

3.2.2 Δομή της ομάδας πινάκων “devices”

Στον πίνακα 3.2 φαίνεται η δομή του πίνακα “devices”. Ο πίνακας αυτός αποτελεί το βασικό κομμάτι του συστήματος μιας και σε αυτόν αποθηκεύονται οι βασικές πληροφορίες των συσκευών κάθε χρήστη. Από την άλλη ο πίνακας 3.3 παρουσιάζει την δομή του πίνακα “device_attributes” ο οποίος συμπληρώνει τον πίνακα “devices”. Σε αυτόν αποθηκεύονται πληροφορίες σχετικά με τα χαρακτηριστικά κάθε συσκευής.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
uid	bigint(20)	Όχι	UNSIGNED
user_id	int(10)	Όχι	UNSIGNED
type_c_id	tinyint(3)	Ναι	UNSIGNED
number_of_schedules	tinyint(3)	Ναι	UNSIGNED
name	varchar(25)	Όχι	-
location	varchar(50)	Όχι	-
long_offline_time_notification...	tinyint(3)	Όχι	UNSIGNED
description	text	Ναι	-
access_token	varchar(255)	Ναι	-
last_contact_at	datetime	Ναι	-

Πίνακας 3.2: Πίνακας “devices”

Ανάλυση των στηλών του πίνακα “devices”:

- **uid:** Σε αυτόν τον πίνακα δεν χρησιμοποιείται η στήλη ‘id’, διότι, υπήρξε η ανάγκη για χρήση ενός αριθμού μεγαλύτερου μήκους, ο οποίος παράγεται με τυχαίο τρόπο και υποδεικνύει μια συσκευή μονοσήμαντα μέσα στο σύστημα.

-
- **user_id**: Περιέχει τον αναγνωριστικό αριθμό του χρήστη στον οποίο η τρέχουσα συσκευή ανήκει. Οι τιμές αυτής της στήλης πρέπει να αντικατοπτρίζουν εγγραφές του πίνακα “users”.
 - **type_c_id**: Υποδεικνύει τον τύπο της συσκευής.
 - **number_of_schedules**: Για συσκευές τύπου ‘έξυπνου θερμοστάτη’ αποθηκεύει το πλήθος των προγραμμάτων που μπορεί να διατηρήσει τοπικά η συσκευή.
 - **name**: Διατηρεί ένα αλφαριθμητικό το οποίο χρησιμοποιείται ως το φιλικό όνομα της συσκευής μέσα στο σύστημα.
 - **location**: Καταγράφεται το σημείο στο οποίο η συσκευή έχει τοποθετηθεί.
 - **long_offline_time_notification_status**: Καταγράφεται η κατάσταση παράδοσης ειδοποίησης στον χρήστη της συσκευής, σε περίπτωση που η συσκευή δεν επικοινωνήσει με το σύστημα για περισσότερο από ένα προκαθορισμένο χρονικό διάστημα.
 - **description**: Προαιρετικό πεδίο, για μια πιο εκτενή περιγραφή σχετικά με την συσκευή.
 - **access_token**: Αποτελεί στήλη που διαχειρίζεται το σύστημα. Σε αυτήν αποθηκεύεται το τυχαίο token που δημιουργείται κατά την καταχώρηση νέας συσκευής και χρησιμοποιείται από την συσκευή για την πρόσβαση της εφαρμογής μέσω του API.
 - **last_contact_at**: Αποθηκεύεται η χρονική στιγμή της τελευταίας επικοινωνίας με την συσκευή.

Ανάλυση των στηλών του πίνακα “device_attributes”:

- **device_uid**: Αποθηκεύεται, ο αναγνωριστικός αριθμός της συσκευής στην οποία ανήκει το τρέχων χαρακτηριστικό.
- **index_on_device**: Καταδεικνύει την θέση στην οποία πρέπει να τοποθετηθεί η πληροφορία αυτού του χαρακτηριστικού, στη δομή δεδομένων που διατηρεί τοπικά η συσκευή.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
device_uid	bigint(20)	Όχι	UNSIGNED
index_on_device	tinyint(3)	Όχι	UNSIGNED
name	varchar(20)	Όχι	-
primitive_type_c_id	tinyint(3)	Όχι	UNSIGNED
direction_c_id	tinyint(3)	Όχι	UNSIGNED
unit	varchar(5)	Ναι	-
min_value	decimal(10,3)	Ναι	-
max_value	decimal(10,3)	Ναι	-
set_value	decimal(10,3)	Ναι	-
current_value	decimal(10,3)	Ναι	-

Πίνακας 3.3: Πίνακας “device_attributes”

- **name:** Ορίζει ένα φιλικό όνομα για το χαρακτηριστικό.
- **primitive_type_c_id:** Ορίζει τον πρωτογενή τύπο δεδομένων του χαρακτηριστικού, αν είναι, δηλαδή, ακέραιος, πραγματικός, κ.τ.λ.
- **direction_c_id:** Σε αυτή την στήλη, ορίζεται ο τρόπος ροής της πληροφορίας/δεδομένων. Δηλαδή, αν το συγκεκριμένο χαρακτηριστικό δίνει κάποια τιμή στην συσκευή, αν λαμβάνει ως ανάδραση μια τιμή από την συσκευή ή τέλος αν κάνει και τα δύο.
- **unit:** Σε αυτήν την προαιρετική στήλη ορίζεται το σύμβολο της μονάδας μέτρησης του χαρακτηριστικού.
- **min_value:** Η ελάχιστη τιμή που μπορεί να λάβει το χαρακτηριστικό.
- **max_value:** Η μέγιστη τιμή του χαρακτηριστικού.
- **set_value:** Η τιμή την οποία θέλει ο χρήστης να λάβει το χαρακτηριστικό.
- **current_value:** Η τρέχουσα τιμή του χαρακτηριστικού.

3.2.3 Δομή του πίνακα “device_infos”

Στον πίνακα “device_infos” του οποίου η δομή βρίσκεται στον πίνακα 3.4 διατηρούνται στοιχεία σχετικά με κάποιες βασικές πληροφορίες για την συσκευή. Τέτοιες είναι ο σειριακός αριθμός της, η έκδοση του firmware, η τρέχουσα IP που έλαβε η συσκευή από το DHCP.

Ανάλυση των στηλών του πίνακα “device_infos”:

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
device_uid	bigint(20)	Όχι	UNSIGNED
stat_name	varchar(255)	Όχι	-
value	varchar(255)	Όχι	-

Πίνακας 3.4: Πίνακας “device_infos”

- **device_uid:** Αποθηκεύεται, ο αναγνωριστικός αριθμός της συσκευής στην οποία ανήκει η τρέχουσα πληροφορία.
- **stat_name:** Αναγνωριστικό όνομα το οποίο έστειλε η συσκευή για την τρέχουσα πληροφορία.
- **value:** Η τιμή της πληροφορίας.

3.2.4 Δομή του πίνακα “actions”

Ο πίνακας 3.5 αποτελεί έναν κοινό πίνακα στον οποίο βασίζονται άλλες οντότητες του συστήματος, οι οποίες θα περιγράφουν στη συνέχεια. Ο πίνακας αυτός χρησιμοποιείται για την αποθήκευση των ενεργειών που πρέπει να εκτελέσει μια συσκευή, δηλαδή, διατηρεί τις τιμές που πρέπει να λάβουν τα χαρακτηριστικά μιας συσκευής.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
device_attribute_id	int(10)	Όχι	UNSIGNED
device_attribute_start_value	decimal(10,3)	Όχι	-
device_attribute_end_value	decimal(10,3)	Όχι	-

Πίνακας 3.5: Πίνακας “actions”

Ανάλυση των στηλών του πίνακα “actions”:

- **device_attribute_id:** Εδώ αναγράφεται ο αναγνωριστικός αριθμός του χαρακτηριστικού της συσκευής για το οποίο προορίζονται οι κάτωθι ρυθμίσεις.
- **device_attribute_start_value:** Προσδιορίζει την τιμή που πρέπει να λάβει το χαρακτηριστικό κατά την έναρξη μιας ενέργειας.
- **device_attribute_end_value:** Ενώ, αυτή η στήλη ορίζει την τιμή κατά την λήξη της ενέργειας.

3.2.5 Δομή της ομάδας πινάκων “schedules”

Στον πίνακα 3.6 εμφανίζεται η δομή του πίνακα “schedules”. Εδώ αποθηκεύονται στοιχεία για τον χρονο-προγραμματισμό των συσκευών που θέτει ο χρήστης, δηλαδή, οι ενέργειες που θέλει να εκτελέσουν κάποια χρονική στιγμή στο μέλλον οι συσκευές του. Στην συνέχεια παρουσιάζονται δύο βοηθητικοί πίνακες, οι οποίοι διατηρούν λεπτομέρειες για χάρη του πρώτου. Ο πίνακας “schedule_events” του οποίου η δομή βρίσκετε στον πίνακα 3.7, διατηρεί την σχέση μεταξύ του προγράμματος και σε ποια ή ποιες συσκευές αναφέρεται το συγκεκριμένο πρόγραμμα. Τέλος, ο πίνακας “schedule_event_actions” με την δομή του στον πίνακα 3.8, διατηρεί στοιχεία για τις ενέργειες που πρέπει να εκτελέσει η κάθε συσκευή του “schedule_events”.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
user_id	int(10)	Όχι	UNSIGNED
title	varchar(255)	Όχι	-
description	text	Ναι	-
start_datetime	datetime	Όχι	-
end_datetime	datetime	Όχι	-
priority	tinyint(3)	Ναι	UNSIGNED
is_recurrent	tinyint(1)	Όχι	UNSIGNED
recurrence_frequency	tinyint(3)	Ναι	UNSIGNED
recurrence_unit	tinyint(3)	Ναι	UNSIGNED

Πίνακας 3.6: Πίνακας “schedules”

Ανάλυση των στηλών του πίνακα “schedules”:

- **user_id**: Αναγνωριστικός αριθμός του χρήστη στον οποίο ανήκει το πρόγραμμα.
- **title**: Περιγραφή για το συγκεκριμένο πρόγραμμα.
- **description**: Προαιρετικά, καταγράφονται με μεγαλύτερη ακρίβεια οι ενέργειες αυτού του προγράμματος.
- **start_datetime**: Χρονική στιγμή κατά την οποία το πρόγραμμα πρέπει να ξεκινήσει να εκτελείται.
- **end_datetime**: Χρονική στιγμή κατά την οποία το πρόγραμμα πρέπει να σταματήσει να εκτελείται.

- **priority:** Σε περίπτωση που υπάρχουν αλληλοεπικαλυπτόμενα προγράμματα ορίζεται η προτεραιότητα τους, με τον μικρότερο αριθμό να σημαίνει υψηλότερη προτεραιότητα.
- **is_recurrent:** Καθορίζει αν το πρόγραμμα θα επαναλαμβάνεται.
- **recurrence_frequency:** Αν οριστεί ότι το πρόγραμμα πρέπει να επαναλαμβάνεται, εδώ ορίζεται η συχνότητα επανάληψης.
- **recurrence_unit:** Αν οριστεί ότι το πρόγραμμα πρέπει να επαναλαμβάνεται, εδώ ορίζεται η μονάδα επανάληψης (π.χ. ανά ημέρα, ανά μήνα).

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
schedule_id	int(10)	Όχι	UNSIGNED
device_uid	bigint(20)	Όχι	UNSIGNED

Πίνακας 3.7: Πίνακας “schedule_events”

Ανάλυση των στηλών του πίνακα “schedule_events”:

- **schedule_id:** Αναγνωριστικός αριθμός του προγράμματος στο οποίο ανήκει η εγγραφή.
- **device_uid:** Αναγνωριστικός αριθμός της συσκευής για την οποία έχει οριστεί πρόγραμμα.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
schedule_event_id	int(10)	Όχι	UNSIGNED
action_id	int(10)	Όχι	UNSIGNED

Πίνακας 3.8: Πίνακας “schedule_event_actions”

Ανάλυση των στηλών του πίνακα “schedule_event_actions”:

- **schedule_event_id:** Αναγνωριστικός αριθμός του βοηθητικού πίνακα “schedule_events” για σύνδεση της συσκευής με τις ενέργειες της.
- **action_id:** Αναγνωριστικός αριθμός, ο οποίος αναφέρεται στον πίνακα “actions”, για την υπόδειξη της ενέργειας με την οποία αυτή η εγγραφή συνδέεται.

3.2.6 Δομή της ομάδας πινάκων “quick_buttons”

Ο πίνακας 3.9 αποτυπώνει την δομή του πίνακα ‘quick_buttons’, ο οποίος διατηρεί δεδομένα για τα κουμπιά άμεσης εκτέλεσης ενεργειών. Ο πίνακας 3.10, είναι βοηθητικός του προαναφερθέν και χρησιμοποιείται για την σύνδεση του με τον “actions”.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
user_id	int(10)	Όχι	UNSIGNED
device_uid	bigint(20)	Όχι	UNSIGNED
index_on_device	tinyint(3)	Όχι	UNSIGNED
title	varchar(255)	Όχι	-
description	text	Ναι	-
duration	int(10)	Όχι	UNSIGNED

Πίνακας 3.9: Πίνακας “quick_buttons”

Ανάλυση των στηλών του πίνακα “quick_buttons”:

- **user_id**: Αναγνωριστικό του χρήστη στον οποίο ανήκει το quick button.
- **device_uid**: Αποθηκεύεται, το αναγνωριστικό της συσκευής στην οποία ανήκει το τρέχων quick button.
- **index_on_device**: Καταδεικνύει την θέση στην οποία πρέπει να τοποθετηθεί η πληροφορία αυτού του κουμπιού άμεσης εκτέλεσης ενεργειών, στη δομή δεδομένων που διατηρεί τοπικά η συσκευή.
- **title**: Περιγράφει τις ενέργειες που εκτελεί το quick button.
- **description**: Προαιρετικά, περιγράφεται με μεγαλύτερη σαφήνεια η κατάσταση στην οποία θα περιέλθει η συσκευή κατά την ενεργοποίησή αυτού του quick button.
- **duration**: Η χρονική διάρκεια για την οποία θα έχουν ισχύ οι ενέργειες του quick button.

Ανάλυση των στηλών του πίνακα “quick_button_actions”:

- **quick_button_id**: Αναγνωριστικός αριθμός του πίνακα “quick_buttons” για τον οποίο ορίζεται η ενέργεια.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
quick_button_id	int(10)	Όχι	UNSIGNED
action_id	int(10)	Όχι	UNSIGNED

Πίνακας 3.10: Πίνακας “quick_button_actions”

- **action_id**: Αναγνωριστικός αριθμός, ο οποίος αναφέρεται στον πίνακα “actions”, για την υπόδειξη της ενέργειας με την οποία αυτή η εγγραφή συνδέεται.

3.2.7 Δομή της ομάδας πινάκων “smart_thermostats”

Αυτή η ομάδα πινάκων διατηρεί τις απαραίτητες πληροφορίες για τις συσκευές που ορίζονται ως ‘έξυπνοι θερμοστάτες’. Τα στοιχεία αυτά αξιοποιούνται από το σύστημα ώστε η λειτουργία του να είναι πιο αποδοτική. Αρχικά ο πίνακας “smart_thermostats”, του οποίου η δομή βρίσκεται στον πίνακα 3.11, περιέχει τις συσχετίσεις που χρειάζεται μια τέτοια συσκευή για την λήψη των αναγκαίων δεδομένων, δηλαδή, εσωτερική, εξωτερική θερμοκρασία και κατάσταση λειτουργίας της πηγής ενέργειας. Οι επόμενοι δύο πίνακες, που είναι βοηθητικοί άλλα απαραίτητοι για την ‘έξυπνη’ λειτουργία, διατηρούν δεδομένα/μετρήσεις σχετικά με την κατάσταση της συσκευής, άρα του χώρου στον οποίο βρίσκεται. Ειδικότερα, ο πίνακας “smart_thermostats_training_set_samples” με την δομή του στον πίνακα 3.12 αποθηκεύει τις μετρήσεις, ενώ στον πίνακα “smart_thermostats_computed_datasets”, του οποίου η δομή βρίσκεται στον πίνακα 3.13, αποθηκεύονται οι τιμές που προκύπτουν έπειτα από την ανάλυση του προηγούμενου πίνακα.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
smart_device_uid	bigint(20)	Όχι	UNSIGNED
smart_device_attribute_type_c_id	tinyint(3)	Όχι	UNSIGNED
source_device_uid	bigint(20)	Όχι	UNSIGNED
source_device_attribute_id	int(10)	Όχι	UNSIGNED

Πίνακας 3.11: Πίνακας “smart_thermostats”

Ανάλυση των στηλών του πίνακα “smart_thermostats”:

- **smart_device_uid**: Αναγνωριστικό συσκευής η οποία έχει οριστεί ως ‘έξυπνος θερμοστάτης’.
- **smart_device_attribute_type_c_id**: Ορίζεται ο τύπος πληροφορίας του ‘έξυπνου θερμοστάτη’ για την οποία γίνεται ρύθμιση με την τρέχουσα εγγραφή.

- **source_device_uid:** Αναγνωριστικό συσκευής από την οποία ο ‘έξυπνος θερμοστάτης’ αντλεί την συγκεκριμένη πληροφορία.
- **source_device_attribute_id:** Αναγνωριστικό του χαρακτηριστικού της συσκευής από το οποίο θα αντλείται η πληροφορία για τον ‘έξυπνο θερμοστάτη’.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
device_uid	bigint(20)	Όχι	UNSIGNED
sample_datetime	datetime	Όχι	-
energy_source_status	tinyint(3)	Όχι	UNSIGNED
outside_temperature	int(11)	Όχι	-
inside_temperature	decimal(10,3)	Όχι	-
set_temperature	decimal(10,3)	Όχι	-

Πίνακας 3.12: Πίνακας “smart_thermostats_training_set_samples”

Ανάλυση των στηλών του πίνακα “smart_thermostats_training_set_samples”:

- **device_uid:** Αποθηκεύεται, το αναγνωριστικό της συσκευής του ‘έξυπνου θερμοστάτη’ για τον οποίο λήφθηκε δείγμα.
- **sample_datetime:** Χρονική στιγμή λήψης του δείγματος.
- **energy_source_status:** Η κατάσταση στην οποία βρισκόταν η πηγή ενέργειας κατά την χρονική στιγμή της λήψης του δείγματος.
- **outside_temperature:** Η εξωτερική θερμοκρασία που επικρατούσε, όταν πάρθηκε το δείγμα, κβαντισμένη στον κοντινότερο ακέραιο.
- **inside_temperature:** Η εσωτερική θερμοκρασία του χώρου, στρογγυλεμένη στο πρώτο δεκαδικό ψηφίο.
- **set_temperature:** Η θερμοκρασία την οποία επιθυμούσε ο χρήστης να επιτύχει το σύστημα.

Ανάλυση των στηλών του πίνακα “smart_thermostats_computed_datasets”:

- **device_uid:** Αποθηκεύεται, το αναγνωριστικό της συσκευής ‘έξυπνου θερμοστάτη’ στον οποίο αναφέρεται η ανάλυση.
- **outside_temperature:** Η εξωτερική θερμοκρασία στην οποία βασίζεται η ανάλυση αυτής της εγγραφής.

Όνομα στήλης	Τύπος	Κενό	Έξτρα χαρακτηριστικά
device_uid	bigint(20)	Όχι	UNSIGNED
outside_temperature	int(11)	Όχι	-
inside_temperature	decimal(10,3)	Όχι	-
timeline	int(10)	Όχι	UNSIGNED

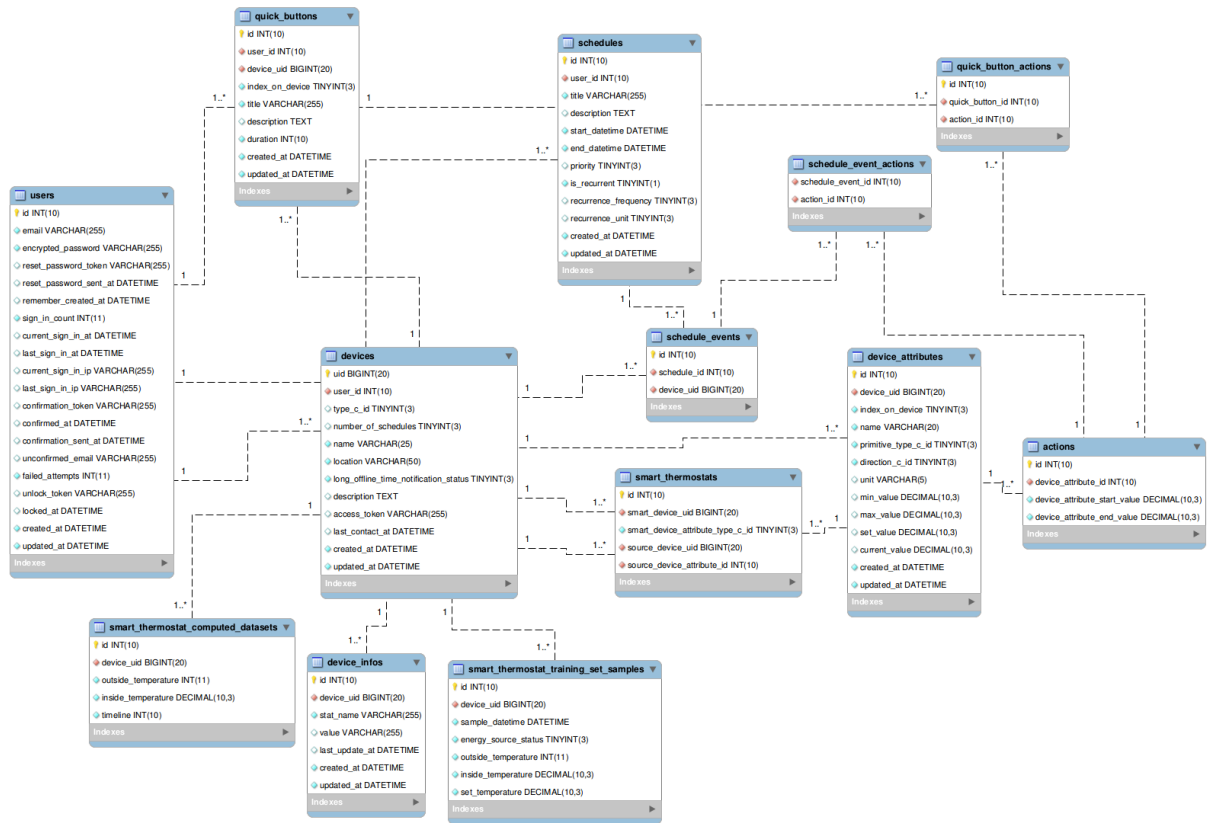
Πίνακας 3.13: Πίνακας “smart_thermostats_computed_datasets”

- **inside_temperature:** Η εσωτερική θερμοκρασία η οποία επιτεύχθει.
- **timeline:** Χρονική στιγμή κατά την οποία η εσωτερική θερμοκρασία επιτεύχθει για μία συγκεκριμένη εξωτερική θερμοκρασία.

3.3 Διάγραμμα οντοτήτων συσχετίσεων

Σε μία σχεσιακή βάση δεδομένων, ορίζονται συσχετίσεις μεταξύ των διαφόρων πινάκων/οντοτήτων της βάσης. Κάθε σχέση ορίζεται μεταξύ δύο στηλών. Σκοπός των συσχετίσεων είναι η οπτικοποίηση της σχέσης της μίας οντότητας με την άλλη, αλλά κυρίως η εφαρμογή περιορισμών. Εφαρμογή περιορισμών σημαίνει πως οι συσχετισμένες στήλες δεν μπορούν να πάρουν οποιεσδήποτε τιμές. Στην μία από τις δυο στήλες υπάρχουν τα κύρια δεδομένα, δηλαδή, ο πίνακας στον οποίο οι άλλοι αναφέρονται (referenced table), όπως για παράδειγμα στον πίνακα “users” η στήλη ‘id’. Ενώ, στην άλλη στήλη αποθηκεύονται υποχρεωτικά μόνο τιμές που υπάρχουν ήδη στην στήλη ‘id’ του πίνακα “users”, δηλαδή ο πίνακας ο οποίος αναφέρει κάποιον άλλον (referencing table).

Στην εικόνα 3.1, εμφανίζονται όλες οι συσχετίσεις που υπάρχουν μεταξύ των οντοτήτων του συστήματος. Έτσι, πλέον, γίνεται κατανοητό ότι οι συσκευές είναι το επίκεντρο ολόκληρου του συστήματος, οι οποίες βέβαια ανήκουν σε κάποιο χρήστη. Άρα, είναι φανερό πως υπάρχει μια συσχέτιση μεταξύ του πίνακα “devices” και του “users”. Ο πρώτος, από τους εναπομείναντες πίνακες, αφορά την διαχείριση των χαρακτηριστικών των συσκευών και έτσι βλέπουμε την συσχέτιση μεταξύ του “device_attributes” και του “devices”. Έπειτα, υπάρχει και ο πίνακας “schedules”, ο οποίος σχετίζεται με κάποιον χρήστη. Μέσω του “schedule_events”, ο πίνακας “schedules” σχετίζεται με κάποια ή κάποιες συσκευές και τελικά μέσω του “schedule_event_actions” με τις ενέργειες (“actions”) της εκάστοτε συσκευής. Από την άλλη, υφίσταται ο πίνακας “quick_buttons” ο οποίος ανήκει και σε κάποια



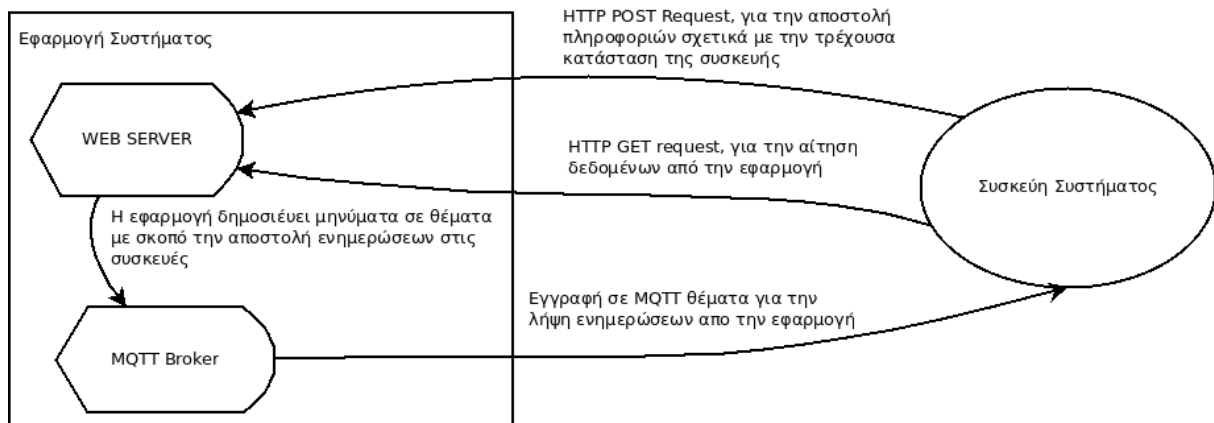
Εικόνα 3.1: Διάγραμμα οντοτήτων συσχετίσεων

συσκευή και σε κάποιον χρήστη, ενώ μέσω του “quick_button_actions” συνδέεται με τον “actions” για να υποδείξει τις ενέργειες της συσκευής. Ο πίνακας “device_infos” ο οποίος συνδέεται με τον πίνακα “devices” διατηρεί γενικές πληροφορίες σχετικά με δεδομένα τα οποία αποστέλλει η συσκευή, για παράδειγμα τον σειριακό της αριθμό. Τέλος, υπάρχουν και οι πίνακες για τη λειτουργία του έξυπνου θερμοστάτη, όπου ο καθένας τους σχετίζεται με τον πίνακα συσκευών.

3.4 Πρωτόκολλο επικοινωνίας

Για την επικοινωνία μέσα στο σύστημα, μεταξύ των συσκευών και της εφαρμογής σχεδιάστηκε μια μορφή επικοινωνίας η οποία συνδυάζει τα παραδοσιακά HTTP requests και HTTP responses και ένα σχετικά νέο πρωτόκολλο, το MQTT. Στο σκαρίφημα της εικόνας 3.2, ορίζεται σχηματικά η μορφή αυτής της επικοινωνίας η οποία καθορίζει το πρωτόκολλο επικοινωνίας του συστήματος της παρούσας διπλωματικής εργασίας.

Στο σκαρίφημα, απεικονίζεται ο υπολογιστής εξυπηρετητής του συστήματος,



Εικόνα 3.2: Draft πρωτοκόλλου επικοινωνίας

στον οποίο εκτελείται ο web server για την λειτουργία της ιστοσελίδας. Σε αυτόν επίσης, εκτελείται και ο MQTT broker. Αυτό το κομμάτι αποτελεί την εφαρμογή του συστήματος. Ακόμα, στο σύστημα περιέχονται και οι συσκευές, για την απλούστευση του σκαριφήματος βέβαια, απεικονίστηκε μόνο μια συσκευή.

Το πρωτόκολλο ορίζει πως οι συσκευές πρέπει να κάνουν εγγραφή σε ένα θέμα (topic) του MQTT broker ώστε η εφαρμογή να μπορεί να στέλνει άμεσα τα νέα δεδομένα μέσω αυτού στην συσκευή. Το όνομα αυτού του θέματος θα πρέπει να είναι ο σειριακός αριθμός της εκάστοτε συσκευής.

Από την άλλη, οι συσκευές θα πρέπει να αποστέλλουν HTTP GET request όταν χρειάζονται κάποια πληροφορία και η εφαρμογή θα αποκριθεί μέσω του MQTT καναλιού.

Τέλος για την ενημέρωση της εφαρμογής σχετικά με την κατάσταση τους, οι συσκευές θα πρέπει να στείλουν κατάλληλα HTTP POST requests.

Κεφάλαιο 4

Υλικό μέρος συστήματος

Σε αυτό το κεφάλαιο θα γίνει η παρουσίαση του υλικού μέρος του συστήματος, το οποίο είναι βασισμένο στον μικρο-ελεγκτή Arduino Uno. Στον τελευταίο συνδέθηκαν όλες οι περιφερειακές συσκευές που απαιτούνται για τη δημιουργία της υλικής υποδομής μιας ευφυούς οικολογικής οικίας.

Συγκεκριμένα στο Arduino είναι συνδεδεμένες οι κάτωθι συσκευές και τα ηλεκτρονικά στοιχεία:

- Ethernet Shield
- Ψηφιακός αισθητήρας Θερμοκρασίας & υγρασίας
- Διακόπτες
- Shift Register
- LEDs

Οι συσκευές που συνδέονται στο Arduino μπορούν να χωριστούν σε συσκευές εισόδου, οι οποίες τροφοδοτούν το Arduino με δεδομένα, σε συσκευές εξόδου, οι οποίες παρέχουν πληροφορίες στον χρήστη, καθώς και σε συσκευές διεπαφής, οι οποίες αναλαμβάνουν να γεφυρώσουν την επικοινωνία μεταξύ του υλικού και του λογισμικού μέρους του συστήματος.

Το Ethernet Shield αποτελεί συσκευή διεπαφής και αναλαμβάνει να εκτελέσει την αμφίδρομη επικοινωνία μεταξύ της εφαρμογής του συστήματος και της συσκευής Arduino. Λαμβάνει τα δεδομένα που εισάγει ο χρήστης μέσω της ιστοσελίδας και τα μεταφέρει στο Arduino. Αντίστοιχα μπορεί να συλλέγει τα δεδομένα

που εισάγει ο χρήστης στη συσκευή και να τα μεταφέρει στην ιστοσελίδα. Ο αισθητήρας θερμοκρασίας & υγρασίας καθώς και οι διακόπτες αποτελούν συσκευές εισόδου και τροφοδοτούν με δεδομένα το Arduino, ενώ τα LEDs και ο Shift Register έχουν το ρόλο συσκευών εξόδου και πληροφορούν το χρήστη για την κατάσταση του συστήματος.

Το Arduino με τη σειρά του, συλλέγει τα δεδομένα που του παρέχουν οι συσκευές εισόδου και τα στέλνει στην εφαρμογή μέσω του Ethernet Shield. Αφού λάβει απάντηση από την εφαρμογή, δρα αντίστοιχα πάνω στις συσκευές εξόδου, ώστε να πραγματοποιήσει τις απαιτούμενες ενέργειες.

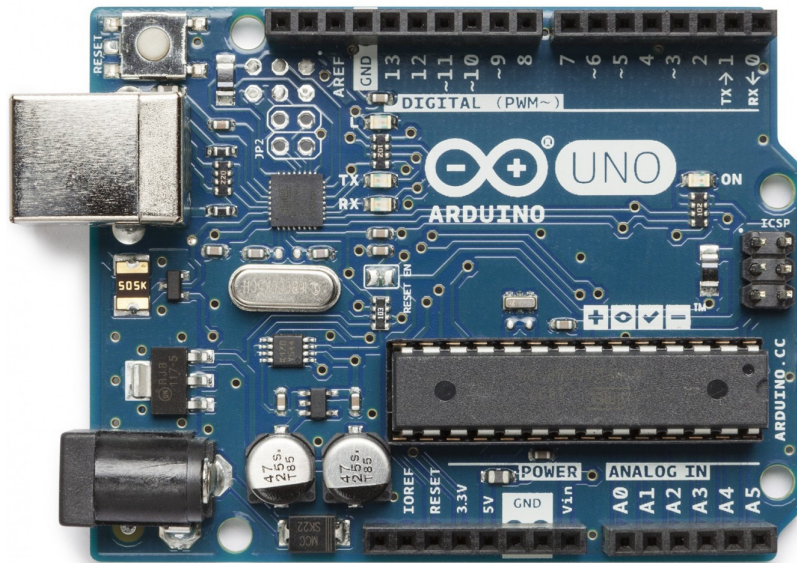
Στις παραγράφους που ακολουθούν θα γίνει παρουσίαση των τεχνικών χαρακτηριστικών όλων των συσκευών και ηλεκτρονικών στοιχείων που χρησιμοποιήθηκαν στην παρούσα διπλωματική εργασία. Θα δοθούν αναλυτικές πληροφορίες για τη λειτουργία του καθενός, καθώς και για τον τρόπο με τον οποίο συνδέονται στο Arduino.

4.1 Arduino Uno

Το Arduino Uno [5] είναι μια επιμέρους συσκευή της αρχιτεκτονικής Arduino. Βασίζεται στον μικρο-ελεγκτή ATmega328 και έχει τάση λειτουργίας 5 Volts. Διαθέτει 14 ψηφιακές εισόδους/εξόδους εκ των οποίων οι 6 παρέχουν PWM έξοδο, 6 αναλογικές εισόδους και ταχύτητα ρολογιού 16 MHz. Παράδειγμα αυτού παρουσιάζεται στην εικόνα 4.1.

Το Arduino Uno μπορεί να τροφοδοτείται είτε μέσω της σύνδεσης USB, είτε μέσω ενός προσαρμογέα AC σε DC από 7 έως 12V, είτε από μπαταρία αντίστοιχης τάσης. Ο προσαρμογέας θα πρέπει να καταλήγει σε βύσμα διαμέτρου 2,1mm με θετικό κέντρο. Η πηγή ενέργειας επιλέγεται αυτόματα και δεν απαιτούνται επιπλέον ενέργειες από το χρήστη.

Όσον αφορά τη μνήμη, ο μικρο-ελεγκτής ATmega328, έχει τρεις κατηγορίες μνήμης. Τη μνήμη Flash, στην οποία αποθηκεύονται τα Arduino sketches, τη μνήμη SRAM (static random access memory) η οποία χρησιμοποιείται από τις μεταβλητές κατά την εκτέλεση του προγράμματος, και τη μνήμη EEPROM, η οποία χρησιμοποιείται από τους προγραμματιστές για την αποθήκευση δεδομένων. Αναλυτικότερα:



Εικόνα 4.1: Πλακέτα Arduino Uno R3

- 2KB μνήμης SRAM: Η μνήμη που μπορούν να χρησιμοποιήσουν τα προγράμματα για να αποθηκεύουν μεταβλητές, πίνακες κ.λ.π. Η SRAM χάνει τα δεδομένα της όταν διακοπεί η παροχή ρεύματος στο Arduino.
- 1KB μνήμης EEPROM: Μπορεί να χρησιμοποιηθεί για εγγραφή ή ανάγνωση δεδομένων από τα προγράμματα. Σε αντίθεση με την SRAM, δε χάνει τα περιεχόμενά της με απώλεια ρεύματος.
- 32KB μνήμης Flash: Τα 2 KB χρησιμοποιούνται για τον bootloader. Τα υπόλοιπα 30KB είναι διαθέσιμα για την αποθήκευση των μεταγλωττισμένων Arduino sketches. Η μνήμη Flash δε χάνει τα περιεχόμενά της με απώλεια τροφοδοσίας.

Στην εικόνα 4.1 διακρίνονται ορισμένοι ακροδέκτες πάνω στην πλακέτα. Από αυτούς, κάποιοι αποτελούν διαθέσιμες εισόδους και εξόδους, ενώ κάποιοι άλλοι επιτελούν διαφορετικές εργασίες.

Αρχικά, οι ακροδέκτες που διακρίνονται κάτω αριστερά στην εικόνα 4.1 αποτελούν pins τροφοδοσίας για το Arduino. Οι υπόλοιποι ακροδέκτες αποτελούν εισόδους και εξόδους για το χρήστη. Συγκεκριμένα:

- VIN: Ο ακροδέκτης αυτός χρησιμοποιείται για την παροχή τάσης στην πλακέτα όταν έχει επιλεγθεί μια εξωτερική πηγή. Είναι δυνατό είτε να εισάγουμε

τροφοδοσία στον ακροδέκτη αυτό, είτε να τον χρησιμοποιούμε για να έχουμε πρόσβαση στην τροφοδοσία αν συνδεθεί το βύσμα 2.1mm.

- 5V: Παρέχει σταθεροποιημένη έξοδο στα 5V.
- 3V3: Παρέχει σταθεροποιημένη έξοδο στα 3,3V.
- GND: Γείωση.
- IOREF: Η τάση αναφοράς με την οποία λειτουργεί ο μικρο-ελεγκτής. Χρησιμοποιείται από τα shields ώστε να επιλεγεί η κατάλληλη πηγή ενέργειας για αυτά.

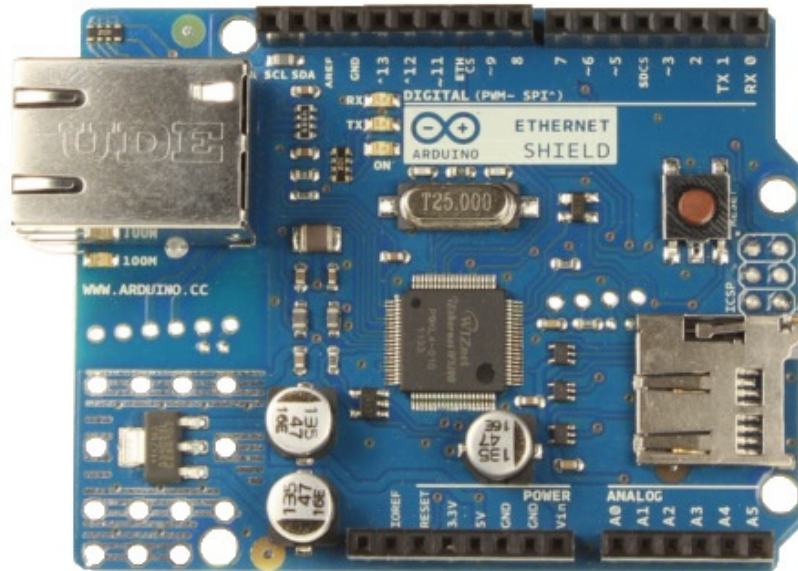
Τα pins από 0 έως 13 μπορούν να χρησιμοποιηθούν ως ψηφιακές εισοδοι/έξοδοι. Το pin AREF χρησιμοποιείται για να παρέχεται τάση αναφοράς στα αναλογικά pins που θα αναλυθούν στη συνέχεια. Εκτός από ψηφιακές εισοδοι/έξοδοι, ορισμένα pins επιτελούν και άλλες εργασίες. Αναλυτικότερα:

- Pins 0 και 1: λειτουργούν ως RX και TX της σειριακής θύρας, εάν αυτή ενεργοποιηθεί στο πρόγραμμα. Αυτό πρακτικά σημαίνει ότι οι δύο ακροδέκτες δεν μπορούν να χρησιμοποιηθούν πλέον ως ψηφιακές εισοδοι/έξοδοι γιατί μεταφέρουν σειριακά δεδομένα.
- Pins 2 και 3: λειτουργούν και ως εξωτερικά interrupt (interrupt 0 και 1 αντίστοιχα). Ρυθμίζονται μέσα από το πρόγραμμα, ώστε να λειτουργούν αποκλειστικά ως ψηφιακές εισοδοι. Όταν στην είσοδο υπάρξει αλλαγή τάσης, η κανονική ροή του προγράμματος σταματάει άμεσα και εκτελείται μια συνάρτηση που έχει ορίσει ο χρήστης.
- Pins 3, 5, 6, 9, 10 και 11: μπορούν να λειτουργήσουν και ως ψευδό-αναλογικές έξοδοι χρησιμοποιώντας PWM (Pulse Width Modulation). Διακρίνονται με σύμμανση “~” στην εικόνα 4.1.

Τέλος, στην εικόνα 4.1, διακρίνεται μια σειρά από 6 ακροδέκτες κάτω δεξιά, οι οποίοι αποτελούν τις αναλογικές εισόδους του Arduino Uno. Η τάση αναφοράς για τις εισόδους αυτές μπορεί να ρυθμιστεί τροφοδοτώντας εξωτερικά με τάση τον ακροδέκτη AREF, που αναφέρθηκε παραπάνω.

4.2 Ethernet Shield

Το Ethernet Shield [2], που φαίνεται στην εικόνα 4.2, προσδίδει στο Arduino τη δυνατότητα να συνδέεται στο internet καθώς και να διαβάζει και να αποθηκεύει δεδομένα σε κάρτα micro-SD. Συνδέεται στις υποδοχές που διαθέτει το Arduino καλύπτοντας το εξ' ολοκλήρου και τροφοδοτείται με ρεύμα από το Arduino. Επικοινωνεί με το τελευταίο χρησιμοποιώντας το SPI bus στους ακροδέκτες 11, 12 και 13. Μέσω του ακροδέκτη 10 επιλέγεται η χρήση του Ethernet Controller, ενώ μέσω του 4 επιλέγεται ο micro-SD card reader. Συνεπώς, οι 5 ακροδέκτες που μόλις αναφέρθηκαν δεν μπορούν να χρησιμοποιούνται σαν είσοδοι/έξοδοι για το Arduino.



Εικόνα 4.2: Ethernet Shield R3

Στην παρούσα διπλωματική χρησιμοποιήθηκε η τρίτη έκδοση του shield (R3), η οποία βασίζεται στο Ethernet chip Wiznet W5100 [36]. Υποστηρίζει τα πρωτόκολλα TCP και UDP και μπορεί να έχει 4 ταυτόχρονες socket connections, ενώ για τον προγραμματισμό του χρησιμοποιεί τη βιβλιοθήκη Ethernet.

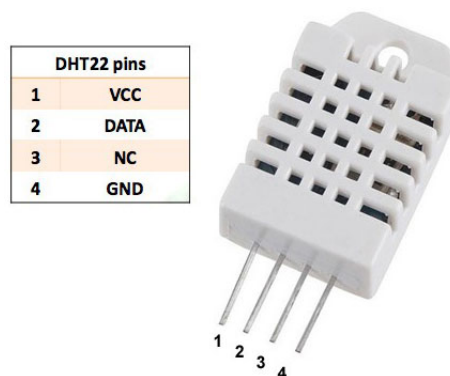
Το Ethernet shield διαθέτει μια σειρά από LEDs για να ενημερώνει το χρήστη σχετικά με τις λειτουργίες του. Αυτές παρουσιάζονται στον πίνακα 4.1

Ονομασία LED	Περιγραφή
PWR	Επιβεβαιώνει την ύπαρξη τροφοδοσίας
LINK	Επιβεβαιώνει την παρουσία δικτύωσης. Αναβοσβήνει κατά την αποστολή ή λήψη δεδομένων.
FULLD	Επιβεβαιώνει την υποστήριξη αμφίδρομης επικοινωνίας εντός του δικτύου.
100M	Επιβεβαιώνει την ύπαρξη σύνδεσης 100 Mb/s.
RX	Αναβοσβήνει κατά τη λήψη δεδομένων.
TX	Αναβοσβήνει κατά την αποστολή δεδομένων.
COLL	Αναβοσβήνει κατά τη διαπίστωση σύγκρουσης (collision) δεδομένων.

Πίνακας 4.1: Περιγραφή λειτουργίας φωτεινών ενδείξεων του Ethernet Shield

4.3 Ψηφιακός αισθητήρας θερμοκρασίας & υγρασίας

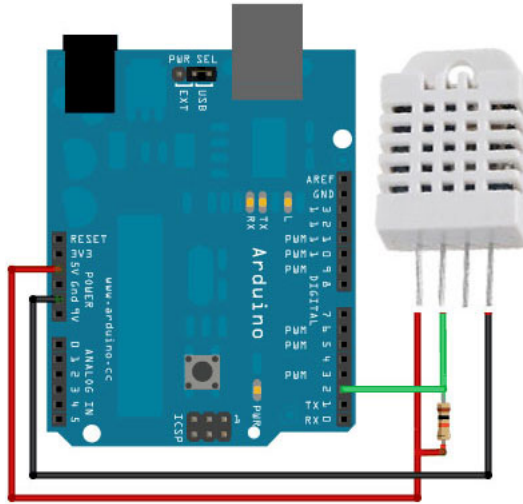
Ο αισθητήρας DHT22 (ή εναλλακτικά RHT03) είναι ένας ψηφιακός αισθητήρας θερμοκρασίας και υγρασίας χαμηλού κόστους [13]. Έχει 4 pins σύνδεσης, εκ των οποίων το πρώτο συνδέεται στον θετικό ακροδέκτη της πηγής, το δεύτερο σε έναν ακροδέκτη ψηφιακής εισόδου για τη μεταφορά δεδομένων στο Arduino και το τέταρτο συνδέεται στη γείωση. Το τρίτο pin δε χρησιμοποιείται. Στην εικόνα 4.3 οπτικοποιείται ένα παράδειγμα τέτοιου αισθητήρα μαζί με το υπόμνημα για την σημασία των ακροδεκτών του.



Εικόνα 4.3: Αισθητήρας Θερμοκρασίας & Υγρασίας RHT03 (DHT22)

Σύμφωνα με τον κατασκευαστή, θα πρέπει να τοποθετηθεί μια αντίσταση pull-up μεταξύ του δεύτερου pin και της τροφοδοσίας έτσι ώστε να εξασφαλίζεται η ορθή μετάδοση των δεδομένων. Στην παρούσα διπλωματική, επειδή ο αισθητήρας βρίσκεται σε πολύ μικρή απόσταση από το Arduino, χρησιμοποιήθηκε μια τυπική αντίσταση pull-up της τάξης των 10kOhm. Σε περίπτωση τοποθέτησης του αισθη-

τήρα σε μεγαλύτερη απόσταση (έως 100 μέτρα) θα ήταν σκόπιμο να χρησιμοποιηθεί πιο ισχυρή pull-up αντίσταση (1kOhm σύμφωνα με το datasheet). Παράδειγμα συνδεσμολογίας για την χρήση με Arduino υπάρχει στην εικόνα 4.4.



Εικόνα 4.4: Σύνδεση αισθητήρα DHT22 με το Arduino Uno

Ο αισθητήρας διαθέτει τα παρακάτω τεχνικά χαρακτηριστικά:

- 3.3-6 V Τάση εισόδου
- 1-1.5 mA κατανάλωση ρεύματος σε λειτουργία
- 40-50 μ A κατανάλωση ρεύματος σε αναμονή
- Μέτρηση υγρασίας από 0 ως 100% RH
- Μέτρηση θερμοκρασίας από -40 ως +80 βαθμούς Κελσίου
- $\pm 2\%$ RH απόκλιση για μετρήσεις υγρασίας
- ± 0.5 βαθμοί Κελσίου απόκλιση για μετρήσεις θερμοκρασίας

4.4 Διακόπτες

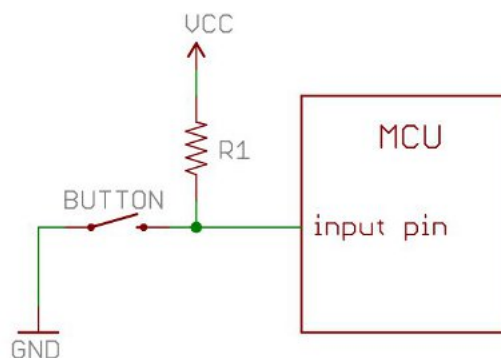
Με τη χρήση του συγκεκριμένου διακόπτη, της εικόνας 4.5, ο χρήστης δύναται να ενεργοποιήσει συγκεκριμένες λειτουργίες της συσκευής. Κάθε διακόπτης συνδέεται σε έναν ακροδέκτη εισόδου και τη γείωση του Arduino. Στην παρούσα διπλωματική οι διακόπτες συνδέθηκαν στους ακροδέκτες αναλογικής εισόδου, οι οποίοι χρησιμοποιήθηκαν όμως ως ψηφιακοί. Αυτό συνέβη διότι δεν υπήρξε η ανάγκη για τη

μέτρηση της τάσης στον ακροδέκτη αλλά για τις δυο διακριτές καταστάσεις στις οποίες αυτός μπορεί να βρεθεί από τη χρήση του διακόπτη.



Εικόνα 4.5: Διακόπτης

Αξίζει στο σημείο αυτό να σημειωθεί ότι η εσωτερική pull-up αντίσταση του Arduino Uno χρησιμοποιείται σε κάθε ακροδέκτη. Έτσι, όταν ο διακόπτης είναι ανοικτός τότε η είσοδος διαρρέεται από ρεύμα και λαμβάνεται η λογική τιμή '1'. Ενώ, όταν είναι κλειστός τότε γειώνεται και λαμβάνονται η λογική τιμή '0'. Στην εικόνα 4.6 περιγράφεται η διαδικασία συνδεσμολογίας μιας pull-up αντίστασης.



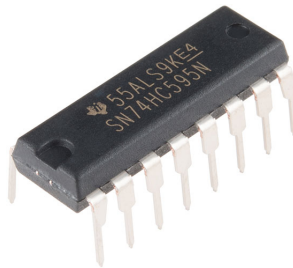
Εικόνα 4.6: Κύκλωμα διακόπτη με pull-up αντίσταση

Συνολικά χρησιμοποιήθηκαν τέσσερις διακόπτες, καθένας εκ των οποίων επιτελεί μία συγκεκριμένη λειτουργία στο σύστημα. Ο πρώτος διακόπτης, ο οποίος διακρίνεται κάτω αριστερά στο εξωτερικό μέρος της συσκευής (εικόνα 4.14), ενεργοποιεί ή απενεργοποιεί τον έλεγχο της εξόδου του κυκλώματος. Με ενεργοποιημένη την λειτουργία ελέγχου της εξόδου, η συσκευή έχει την δυνατότητα να δρα πάνω στην έξοδο ανάλογα με τις εντολές που λαμβάνονται από την ιστοσελίδα. Εάν απενεργοποιηθεί, η συσκευή συνεχίζει να λαμβάνει και να στέλνει δεδομένα από και προς την ιστοσελίδα, όμως δεν επιδρά πάνω στην έξοδο σε καμία περίπτωση. Οι υπόλοιποι τρεις διακόπτες εκτελούν ενέργειες οι οποίες έχουν προκαθοριστεί από το χρήστη στην ιστοσελίδα. Στο εξής αυτοί οι τρεις διακόπτες θα αναφέρονται ως

quick buttons.

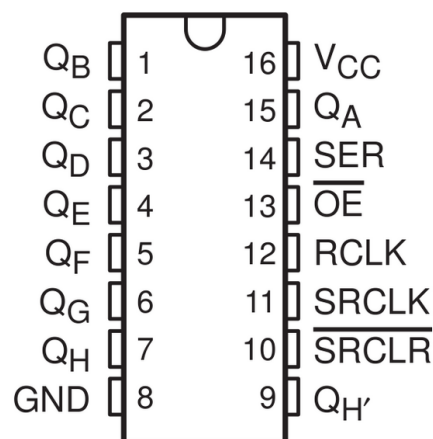
4.5 Shift Register

Ο shift register (καταχωρητής ολίσθησης) SN74HC595N είναι μια ηλεκτρονική διάταξη σειριακής εισόδου - παράλληλης εξόδου (SIPO) των 8 bits [34]. Επιτρέπει τον έλεγχο οκτώ συνολικά συσκευών που δεσμεύουν μόνο τρεις ακροδέκτες στο Arduino. Στη συγκεκριμένη διπλωματική εργασία χρησιμοποιείται για τον έλεγχο των LEDs χωρίς να δεσμεύονται επιπρόσθετες θέσεις, και κυρίως για λόγους αύξησης της επεκτασιμότητας της τελικής συσκευής.



Εικόνα 4.7: Shift Register SN74HC595N

Στην εικόνα 4.8 παρουσιάζονται τα pins του shift register και ακολουθεί η επεξήγηση τους.



Εικόνα 4.8: Τα pins του Shift Register SN74HC595N

- Vcc: Συνδέεται στον θετικό ακροδέκτη της πηγής.
- QA - QH: Οι έξοδοι του shift register.
- SER (Serial): Η (σειριακή) είσοδος για τα δεδομένα.

- SRCLK (Serial Clock): Σε κάθε άνοδο του παλμού του ρολογιού SRCLK εισέρχεται ένα νέο bit στον shift register και τα bits που υπάρχουν ήδη στον shift register μετατοπίζονται κατά μία θέση.
- RCLK (Register Clock): Σε κάθε άνοδο του παλμού του ρολογιού RCLK οι έξοδοι παίρνουν τις αντίστοιχες τιμές που έχουν καταχωρηθεί. Εάν είναι συνδεδεμένο με το ρολόι SRCLK το SRCLK είναι πάντα έναν παλμό μπροστά.
- SRCLR (Serial Clear): Αδειάζει τον καταχωρητή όταν γειώνεται. Στην παρούσα διπλωματική είναι συνδεδεμένο μόνιμα στο Vcc.
- OE (Output Enable): Ενεργοποιεί την έξοδο όταν γειώνεται και την απενεργοποιεί αντίστοιχα όταν συνδέεται στο Vcc. Στην παρούσα διπλωματική είναι συνδεδεμένο μόνιμως στη γείωση.
- QH': Σειριακή έξοδος δεδομένων για σύνδεση με κάποιον επιπλέον shift register. Δε χρησιμοποιείται στην παρούσα διπλωματική εργασία.

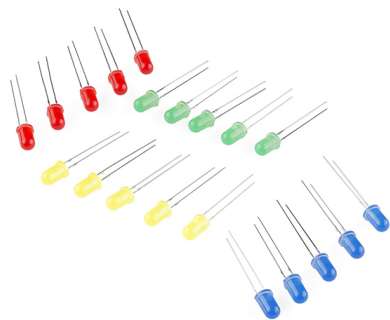
Στην εικόνα 4.9 παρουσιάζεται ένα ενδεικτικό διάγραμμα χρονισμού για τον shift register SN74HC595N για την καλύτερη κατανόηση των παραπάνω επεξηγήσεων. Το 3-state mode που αναφέρεται στο διάγραμμα σημαίνει ότι οι έξοδοι παίρνουν την τιμή Hi-Z, πρακτικά δηλαδή συμπεριφέρονται ως ανοικτό κύκλωμα.

4.6 LEDs

LED (Light Emitting Diode ή Δίοδος Εκπομπής Φωτός), αποκαλείται ένας ημιαγωγός ο οποίος εκπέμπει φωτεινή ακτινοβολία στενού φάσματος όταν του παρέχεται μία ηλεκτρική τάση κατά τη φορά ορθής πόλωσης (forward-biased) [39].

Στη συσκευή που περιγράφεται, τα LEDs χρησιμοποιούνται για να πληροφορούν το χρήστη σχετικά με την κατάσταση της. Παράδειγμα τέτοιων led υπάρχει στην εικόνα 4.10. Κάθε LED συνδέεται στην άνοδο του με μία από τις εξόδους του shift register, ενώ η κάθοδος του συνδέεται στη γείωση σε σειρά με μια αντίσταση, όπως παρουσιάζεται στην εικόνα 4.11.

Για την κατασκευή της συσκευής του συστήματος χρησιμοποιήθηκαν συνολικά έξι LEDs (βλέπε εικόνα 4.14). Το κάτω αριστερά της συσκευής ενημερώνει τον χρήστη σχετικά με το αν είναι ενεργοποιημένος ή όχι ο έλεγχος της εξόδου. Το

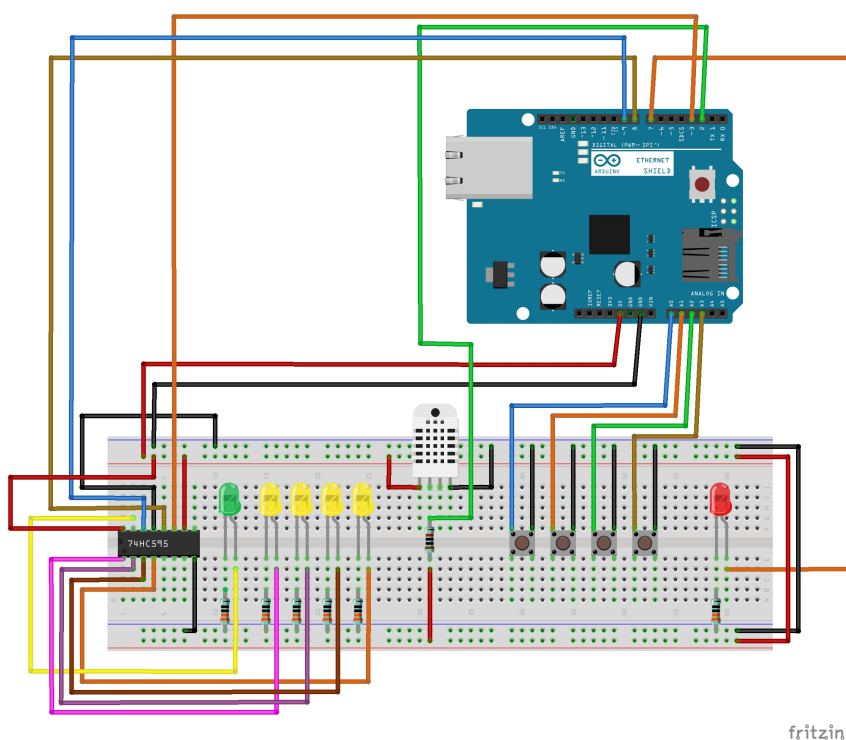


Εικόνα 4.10: LEDs



Εικόνα 4.11: Κύκλωμα LED

κού σχεδίου, του σχεδιαγράμματος της υλοποίησης του πρωτότυπου σε breadboard καθώς και φωτογραφία της τελικής συσκευής που υλοποιήθηκε.

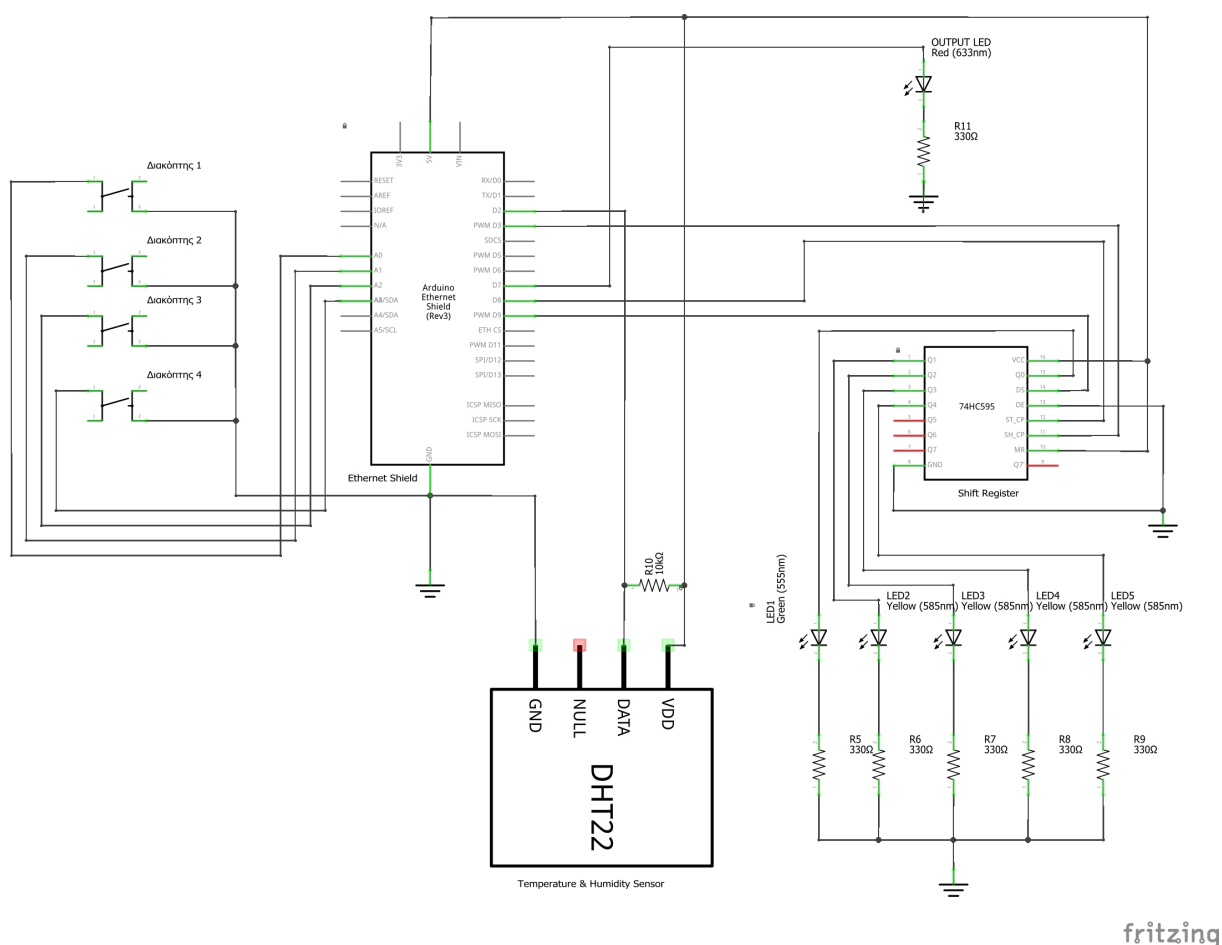


fritzing

Εικόνα 4.12: Κύκλωμα σε breadboard

4.8 Υπολογισμός κόστους εξαρτημάτων

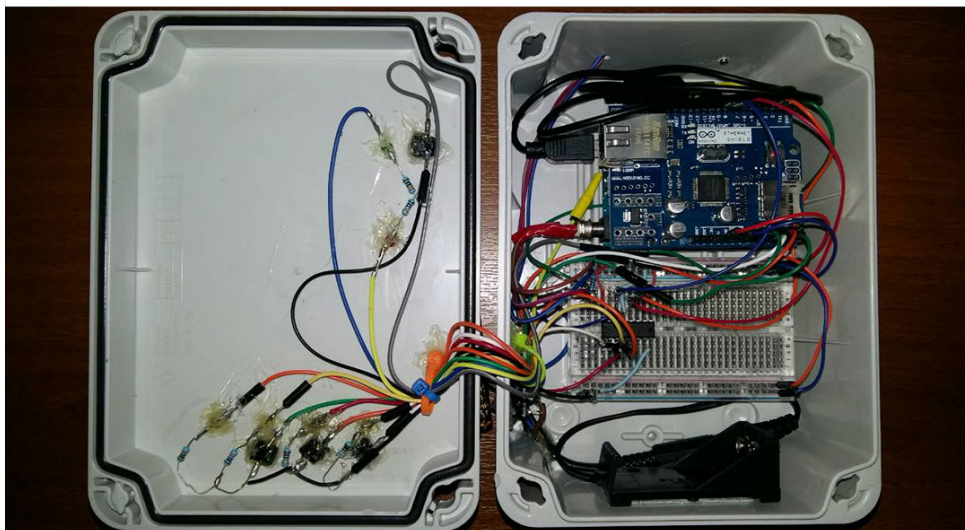
Στην σύντομη αυτή ενότητα, γίνεται μια καταγραφή των συσκευών και των ηλεκτρονικών στοιχείων που χρησιμοποιήθηκαν για την κατασκευή της συσκευής και παρατίθενται στοιχεία για την τιμή και την ποσότητα τους στον πίνακα 4.2.



Εικόνα 4.13: Ηλεκτρονικό σχέδιο κυκλώματος

Όνομα	Ποσότητα	Τιμή μονάδας (€)
Arduino Uno Rev3	1	20.00
Ethernet Shield Rev3	1	32.00
Αισθητήρας Θερμοκρασίας - Υγρασίας DHT22	1	7.80
74HC595 Shift Register	1	0.50
Momentary Push Button Switch	4	0.50
Red Led 5mm	1	0.25
Green Led 5mm	1	0.25
Yellow Led 5mm	4	0.25
Resistor 10K Ohm 1/6th Watt PTH	1	0.01
Resistor 330 Ohm 1/6th Watt PTH	6	0.01
Συνολικό Κόστος: €63.87		

Πίνακας 4.2: Κοστολόγιο υλικών



Εικόνα 4.14: Η Τελική Συσκευή

Κεφάλαιο 5

Λογισμικό μέρος συστήματος

Όπως έχει ήδη αναφερθεί στην σύνοψη κεφαλαίων, το κεντρικό θέμα αυτού του κεφαλαίου αποτελεί το λογισμικό μέρος του συστήματος, με το οποίο και ολοκληρώνεται. Αναλύονται οι λειτουργίες του ιστοχώρου, ο τρόπος προγραμματισμού της συσκευής Arduino και οι ενέργειες παρασκηνίου.

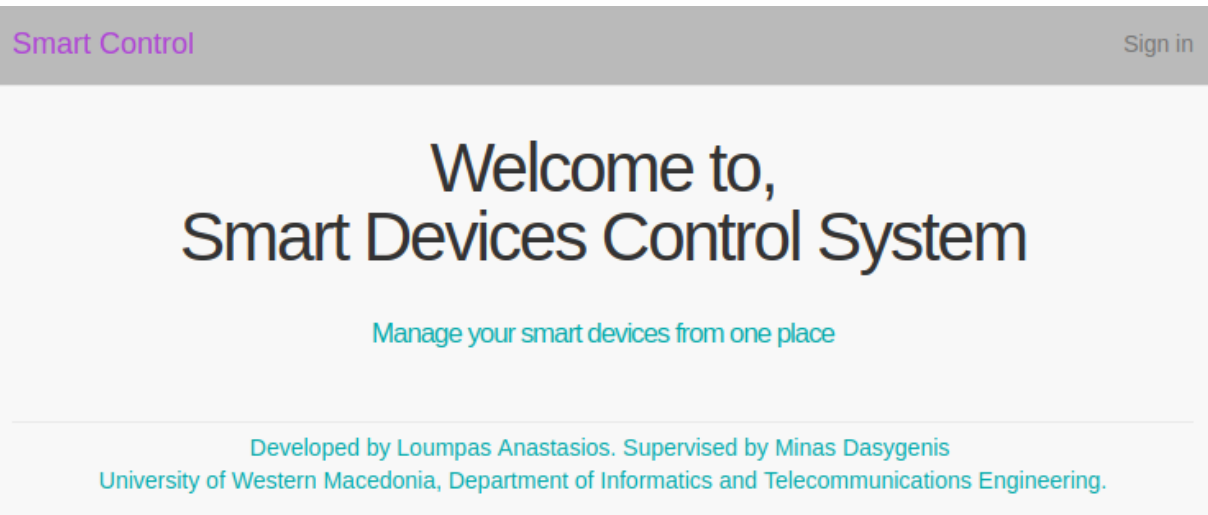
5.1 Ανάλυση λειτουργιών ιστοσελίδας

Το κυριότερο είναι ότι παρουσιάζονται εκτενώς οι λειτουργίες που υποστηρίζονται από την ιστοσελίδα του συστήματος, στο σύνολο τους. Παράλληλα επεξηγούνται όλες οι εμφανίσεις οι οποίες πρόκειται να συναντήσει ο χρήστης καθώς και ο τρόπος με τον οποίο εισάγονται τα δεδομένα στις φόρμες.

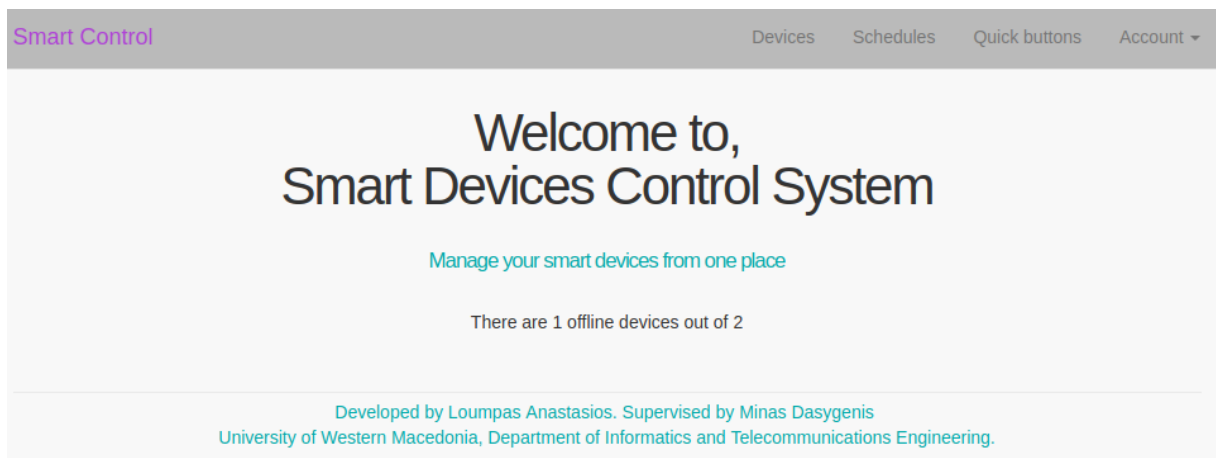
5.1.1 Αρχική σελίδα

Η αρχική είναι η πρώτη σελίδα της εφαρμογής την οποία αντικρίζει κάθε χρήστης που επιθυμεί να αποκτήσει πρόσβαση σε αυτή. Στην περίπτωση που δεν έχει πραγματοποιήσει σύνδεση, εμφανίζονται γενικά μηνύματα σχετικά με το σύστημα, εικόνα 5.1. Εάν έχει συνδεθεί τότε, υπάρχουν επιπρόσθετες πληροφορίες σχετικά με την κατάσταση των συσκευών του, εικόνα 5.2.

Είναι απαραίτητο να τονιστεί πως, η μπάρα της πρώτης γραμμής ονομάζεται μπάρα μενού ή απλά μενού. Εκεί, βρίσκονται οι σύνδεσμοι για την πλοήγηση του χρήστη στην εφαρμογή. Η μπάρα αυτή εμφανίζεται σε κάθε σελίδα της εφαρμογής. Στην αριστερή γωνία της βρίσκεται το όνομα της εφαρμογής, το οποίο πατώντας το ο χρήστης μεταφέρεται πάντα στην αρχική σελίδα.



Εικόνα 5.1: Αρχική σελίδα χωρίς σύνδεση



Εικόνα 5.2: Αρχική σελίδα εφόσον έχει πραγματοποιηθεί σύνδεση

5.1.2 Λειτουργίες χρηστών

Όπως έχει, ήδη, ειπωθεί για να γίνει χρήση της εφαρμογής θα πρέπει ο χρήστης να έχει κάνει είσοδο στο σύστημα. Αυτό το πετυχαίνει πατώντας στον σύνδεσμο 'Sign In' στο δεξί μέρος του μενού, ώστε να μεταφερθεί στην σελίδα εισόδου, που φαίνεται στην εικόνα 5.3.

Ύστερα από επιτυχή σύνδεση, το δεξί μέρος του μενού αλλάζει μορφή, όπως φάνηκε στην εικόνα 5.2. Έτσι, την θέση του συνδέσμου εισόδου έρχονται να πάρουν οι σύνδεσμοι για την πλοήγηση του χρήστη στα υπόλοιπα μέρη της εφαρμογής.

Την πρώτη φορά που ο χρήστης θα θελήσει να αξιοποιήσει την εφαρμογή θα πρέπει να κάνει εγγραφή στο σύστημα πατώντας τον σύνδεσμο 'Sign Up', ο οποίος βρίσκεται στο κάτω μέρος της σελίδας εισόδου του χρήστη. Από εδώ μπορεί, επί-

Πανεπιστήμιο Δυτικής Μακεδονίας
Πολυτεχνική Σχολή
Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Email

Password

☐ Remember me

[Sign up](#)
[Forgot your password?](#)
[Didn't receive confirmation instructions?](#)
[Didn't receive unlock instructions?](#)

Εικόνα 5.3: Σελίδα εισόδου χρήστη

σης, να ζητήσει την υπενθύμιση του κωδικού του και την επαναποστολή οδηγιών ενεργοποίησης και ξεκλειδώματος του λογαριασμού του.

5.1.3 Λειτουργίες συσκευών

Πατώντας τον σύνδεσμο ‘Devices’, από το μενού, επιτυγχάνεται η μετάβαση στις λειτουργίες των συσκευών και συγκεκριμένα στην αρχική οθόνη αυτών. Στην εικόνα 5.4 υπάρχει η σύνοψη αυτή.

Devices List				
Status	Name	Current Values	Location	Actions
ONLINE	Thermostat	ON/OFF: OFF Temperature: 22.3 *C Humidity: 41.1 %	Lounge	@ / x
OFFLINE	Weather Station	Temperature: 11.0 *C	Outside of lounge window	@ / x
New Device				

Εικόνα 5.4: Σύνοψη συσκευών χρήστη

Η συγκεκριμένη εμφάνιση παρουσιάζει πληροφορίες σχετικά με τις καταχωρημένες συσκευές. Αρχικά, στην πρώτη στήλη εμφανίζεται η κατάσταση στην οποία βρίσκεται η κάθε συσκευή, δηλαδή αν είναι σε επικοινωνία με την εφαρμογή ή όχι. Έπειτα, στην στήλη ‘current values’ υπάρχουν συνοπτικά πληροφορίες για τις τρέχουσες τιμές που έχουν τα χαρακτηριστικά της συσκευής. Η στήλη actions, δίνει την δυνατότητα στον χρήστη για την μετάβαση στις αναλυτικές πληροφορίες της συσκευής, στην κατάσταση τροποποίησης ή στην αφαίρεση της συσκευής από το

σύστημα. Τέλος, ο σύνδεσμος ‘New Device’ οδηγεί στην σελίδα εισαγωγής νέας συσκευής στο σύστημα. Πιο συγκεκριμένα:

- **New Device:** Μέσω αυτής της εμφάνισης, η οποία παρουσιάζεται στην εικόνα 5.5, ο χρήστης μπορεί να προσθέσει νέες συσκευές στο σύστημα. Αρχικά, ο χρήστης δίνει ένα όνομα για την συσκευή και περιγράφει την τοποθεσία στην οποία η συσκευή θα βρίσκεται. Έπειτα, με το πεδίο ‘Type’ ορίζει τον τύπο της συσκευής, για παράδειγμα αν είναι ‘έξυπνος θερμοστάτης’. Το πεδίο του τύπου έχει προκαθορισμένες τιμές, αλλά προς το παρόν η μοναδική επιλογή είναι αυτή του ‘έξυπνου θερμοστάτη’. Στην περίπτωση που η συσκευή είναι ‘έξυπνος θερμοστάτης’ θα πρέπει να ορίσει τον αριθμό των προγραμμάτων που μπορεί να αποθηκεύει τοπικά η συσκευή. Στο πεδίο ‘Description’ μπορεί να εισαγάγει μια αναλυτικότερη, αλλά προαιρετική περιγραφή, για την συσκευή.

Create Device								
Basic Info								
Name	Location	Type	No. of shcedules					
		- Select type -						
Description								
Device Attributes								
Add Device Attribute								
Name	Direction Type	Primitive Type	Index on device	Min value	Max value	Set value	Unit	Action
	Bi-directional	Integer						X
Create Device								

Εικόνα 5.5: Δημιουργία νέας συσκευής

Μετά την προσθήκη των βασικών στοιχείων, ακολουθούν οι ρυθμίσεις για τα χαρακτηριστικά της συσκευής (‘Device Attributes’). Πατώντας το κουμπί ‘Add Device Attribute’ εμφανίζεται μία ακόμα γραμμή για την ρύθμιση ενός επιπλέον χαρακτηριστικού για την συσκευή. Για κάθε χαρακτηριστικό πρέπει να οριστεί κάποιο αναγνωριστικό όνομα. Η στήλη ‘Direction’ ορίζει την ροή της πληροφορίας για το τρέχων χαρακτηριστικό, δηλαδή αν το συγκεκριμένο χαρακτηριστικό δέχεται μόνο τιμές για εντολές προς την συσκευή, αν επιστρέφει

μόνο τιμές από την συσκευή ή αν κάνει και τα δύο ταυτόχρονα. Έπεται η ρύθμιση για τον πρωτογενή τύπο δεδομένων που χρησιμοποιεί το χαρακτηριστικό. Η στήλη 'Index on device' δημιουργεί την αντιστοίχιση μεταξύ του χαρακτηριστικού και της θέσης στην οποία πρέπει αυτό να αποθηκευτεί στις τοπικές δομές που διατηρεί η συσκευή. Στην συνέχεια ορίζεται η ελάχιστη, η μέγιστη και η επιθυμητή τιμή για το συγκεκριμένο χαρακτηριστικό. Στην στήλη 'Unit' προαιρετικά καθορίζεται ένα σύμβολο για το τρέχον χαρακτηριστικό. Τέλος, από την στήλη 'action' και με το κουμπί 'X' γίνεται η αφαίρεση του χαρακτηριστικού.

- **Edit Device:** Η εμφάνιση επεξεργασίας της συσκευής είναι ίδια με της δημιουργίας, απλώς τα πεδία είναι προσυμπληρωμένα με τα στοιχεία που εισήχθησαν κατά την δημιουργία νέας συσκευής. Σε αυτήν την σελίδα μπορεί ο χρήστης να τροποποιήσει υπάρχοντα πεδία, να δώσει τιμές σε κενά προαιρετικά πεδία ή και να μεταβάλλει τα στοιχεία και το πλήθος των χαρακτηριστικών.

Αν η συσκευή έχει οριστεί ως 'Smart Thermostat', στον τύπο της συσκευής κατά την δημιουργία της, τότε θα εμφανιστούν μερικά ακόμα πεδία, τα οποία υπάρχουν στην εικόνα 5.6. Αυτά χρησιμοποιούνται με στόχο την σύνδεση των χαρακτηριστικών των συσκευών για την λήψη των δεδομένων που απαιτούνται, ώστε να λειτουργήσει η διαδικασία του 'έξυπνου θερμοστάτη'. Η ανάλυση και ο τρόπος λειτουργίας γίνεται στην ενότητα 5.2.1.

Smart thermostat with device attributes associations

Outside Temperature	Weather Station	Temperature
Inside Temperature		Temperature
Set Temperature		Temperature
Working Status		ON/OFF

Εικόνα 5.6: Επεξεργασία συσκευής - Ρυθμίσεις έξυπνου θερμοστάτη

- **Αναλυτικές πληροφορίες συσκευής:**

Στη τελευταία εμφάνιση παρουσιάζονται αναλυτικά οι πληροφορίες για την συσκευή που έχει επιλεγθεί. Αρχικά, στην εικόνα 5.7, εμφανίζονται οι πληροφο-

Detailed view for the device

Thermostat @ Lounge

Analytical description for this device

Status

OFFLINE (last contact 12 minutes ago)

Attributes settings & status

Content of this section auto-updates every 10 seconds

Attribute	Primitive Type	Min value	Max value	Set value	Current value
ON/OFF	Bool	OFF	ON	23.0	ON
Temperature	Float	-25.0	120.0	23.0 °C	21.7 °C
Humidity	Float	-	-	-	49.3 %

Εικόνα 5.7: Αναλυτικές πληροφορίες συσκευής - Ονομασία, κατάσταση και χαρακτηριστικά

ρίες που αφορούν το όνομα, την τοποθεσία της συσκευής και την προαιρετική περιγραφή, αν έχει οριστεί. Ακολουθεί η ένδειξη για την κατάσταση σύνδεσης της συσκευής με την εφαρμογή και οι αναλυτικές πληροφορίες σχετικά με τα χαρακτηριστικά της. Από την εμφάνιση της εικόνας που περιγράφεται, γίνεται και η ρύθμιση για τις επιθυμητές τιμές των χαρακτηριστικών της συσκευής. Αυτό επιτυγχάνεται μέσω της στήλης ‘Set Value’ και αφού γίνει ‘κλικ’ πάνω στην τιμή που χρειάζεται μεταβολή. Με το ‘κλικ’ αυτό, εμφανίζεται το μικρό αναδυόμενο παράθυρο (tooltip) που φαίνεται στην εικόνα. Εκεί, ο χρήστης πληκτρολογεί την τιμή που θέλει να θέσει για το τρέχων χαρακτηριστικό και ή θα πατήσει το κουμπί αποδοχής ή το κουμπί απόρριψης για την ματαίωση της αλλαγής.

Στην συνέχεια, εικόνα 5.8, ο χρήστης ενημερώνεται για τον αναγνωριστικό αριθμό της συσκευής στο σύστημα. Ακόμα, πληροφορείται και για access token που πρέπει να μεταφερθεί στην συσκευή για την απόκτηση πρόσβασης στο API του συστήματος. Στο επόμενο πλαίσιο εμφανίζονται πληροφορίες οι οποίες προήλθαν από την συσκευή και αφορούν τον σειριακό αριθμό της, την έκδοση του firmware και τέλος την τρέχουσα IP διεύθυνση της. Στα δύο εναπομείναντα πλαίσια παρουσιάζονται πληροφορίες σχετικά με τα προγράμματα που έχουν

System info for device

UID: 505932008267218

Access token: hJ8bWVNGcRKfXZvQ9BFm9LF5

Information about device

Name	Value	Last update at
sn	505932008267218	10/10/2017 10:25
fw	0.0.1	10/10/2017 10:25
ip	10.168.10.115	10/10/2017 10:25
t	sc	04/10/2017 19:16

Schedules

Title	Start datetime	End datetime	Repeat ...
Every 2 weeks	18/10/2017 04:00	18/10/2017 09:00	Every 2 Weeks
Today only	10/10/2017 11:00	10/10/2017 12:00	One Time
Every 3 days	04/10/2017 08:00	04/10/2017 20:00	Every 3 Days

Quick buttons

Title	Description	Duration (s)
No1	This is a test QB	5

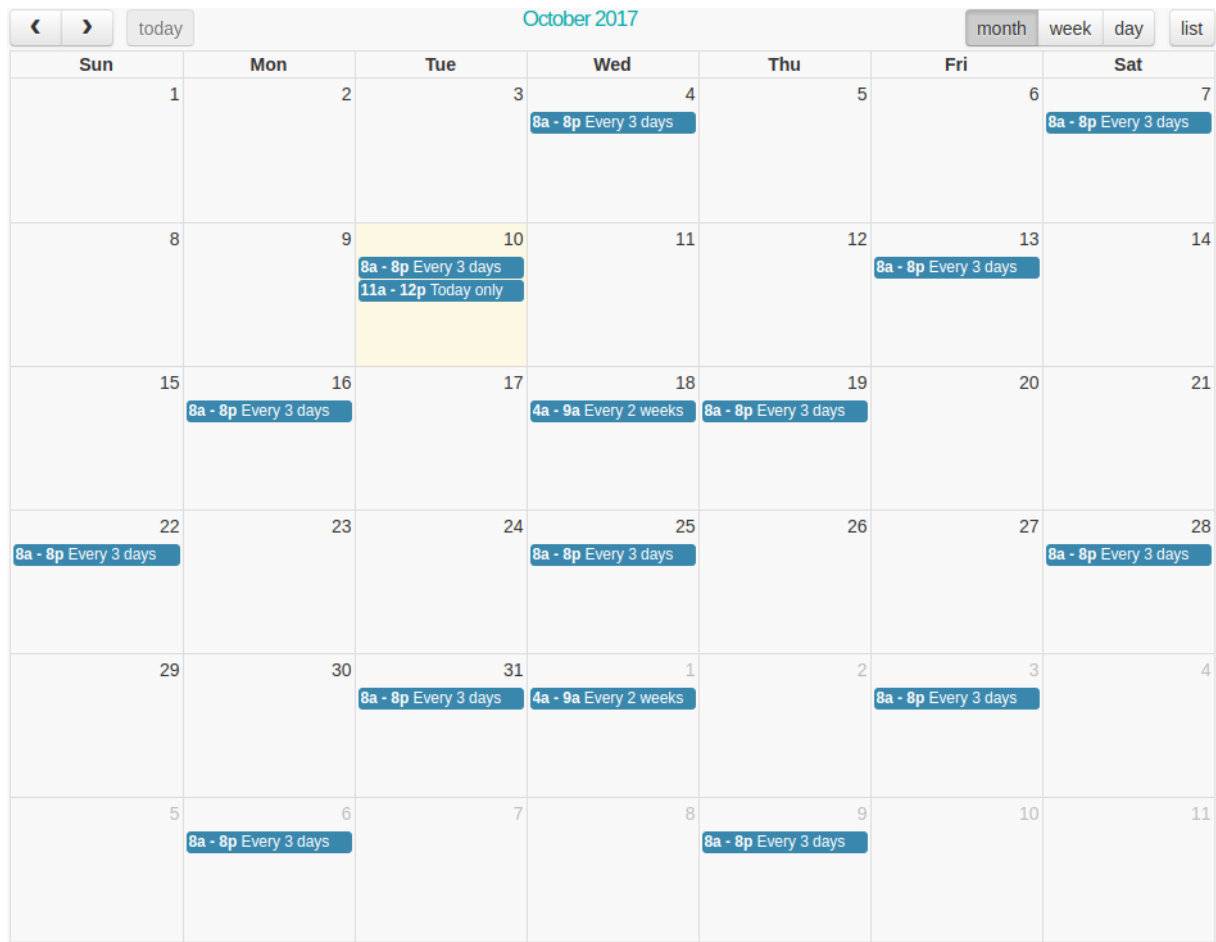
Εικόνα 5.8: Αναλυτικές πληροφορίες συσκευής - Πληροφορίες συσκευής, προγράμματα, quick buttons

οριστεί για την συσκευή αυτή, μαζί με κάποιες λεπτομέρειες που αφορούν τον χρόνο εκτέλεσης τους και το αν είναι επαναλαμβανόμενα ή μη. Παρόμοιες πληροφορίες εμφανίζονται και για τα quick buttons της τρέχουσας συσκευής.

5.1.4 Λειτουργίες χρονοπρογραμματισμού

Στην εικόνα 5.9 αναπαρίσταται η σύνοψη του χρονοπρογραμματισμού η οποία παρέχει στον χρήστη μια συγκεντρωτική εικόνα για τα προγράμματα του μήνα. Μπορεί να μεταβεί σε εμφάνιση εβδομάδας, μέρας και λίστας προγραμμάτων από τα κουμπιά πάνω δεξιά. Επίσης από τα βελάκια, πάνω αριστερά, μπορεί να μεταβεί σε επόμενα ή προηγούμενα χρονικά διαστήματα για την εκάστοτε επιλογή εμφάνισης. Κάνοντας ‘κλικ’ πάνω σε κάποια μέρα εμφανίζεται η οθόνη εισαγωγής νέου προγράμματος. Αν το ‘κλικ’ γίνει πάνω σε υπάρχον πρόγραμμα τότε, εμφανίζεται η οθόνη τροποποίησης του τρέχων προγράμματος.

Οι λειτουργίες του χρονοπρογραμματισμού διαχειρίζονται, όπως ειπώθηκε προη-



Εικόνα 5.9: Σύνοψη χρονοπρογραμματισμών χρήστη

γυμνώντας, κάνοντας ‘κλικ’ είτε σε κενό χώρο είτε σε κάποιο προϋπάρχον πρόγραμμα. Με την κίνηση αυτή, δηλαδή το ‘κλικ’, ανοίγει ένα αναδυόμενο παράθυρο (modal). Η εμφάνιση αυτού υπάρχει στην εικόνα 5.10. Το παράθυρο είναι κενό στην περίπτωση εισαγωγής νέου προγράμματος και προσυμπληρωμένο εάν έχει επιλεγεί κάποιο πρόγραμμα. Τέλος, μέσω του κουμπιού του κάδου ανακύκλωσης, στο κάτω δεξιά μέρος του παραθύρου, το πρόγραμμα διαγράφεται.

Μέσω του παραπάνω παραθύρου, ορίζεται ένας σύντομος τίτλος για το πρόγραμμα και μια προαιρετική περιγραφή. Το πεδίο ‘Priority’ χρησιμοποιείται στην περίπτωση που δυο ή περισσότερα μη επαναλαμβανόμενα προγράμματα υπερκαλύπτονται. Σε αυτή την περίπτωση θα πρέπει να οριστούν προτεραιότητές για να αποφασιστεί ποιο πρόγραμμα θα εκτελείται κάθε χρονική στιγμή. Ο μικρότερος αριθμός υποδηλώνει τη μεγαλύτερη προτεραιότητα. Στη συνέχεια ορίζεται η χρονική στιγμή έναρξης του προγράμματος και αντίστοιχα της λήξης. Επιλέγοντας ή απο-επιλέγοντας το πλαίσιο ελέγχου (checkbox) του ‘Repeat’, εμφανίζονται και εξα-

Schedule Settings
×

Title

Description

Priority

Start datetime

End datetime

☐ Repeat ...

Frequency

Repeat Unit

Thermostat

+ Add device to schedule

Thermostat

Device attribute

Device attribute start value

Device attribute end value

ON/OFF

Temperature

×

 Remove event

Close

Save changes

Εικόνα 5.10: Δημιουργία νέου προγράμματος

φανίζονται οι ρυθμίσεις σχετικά με την επαναληψιμότητα του προγράμματος. Το πεδίο ‘Frequency’ ορίζει κάθε πόσες χρονικές μονάδες θα εκτελείται το πρόγραμμα ενώ, το ‘Repeat Unit’ ορίζει την χρονική μονάδα επανάληψης. Το τελευταίο κομμάτι του παραθύρου αυτού αξιοποιείται για την τοποθέτηση των συσκευών στις οποίες αναφέρεται το συγκεκριμένο πρόγραμμα. Αρχικά, μέσω του πτυσσόμενου πεδίου (select) γίνεται αναζήτηση στις συσκευές του συστήματος. Αφού έχει επιλεγθεί μια συσκευή, ο χρήστης πρέπει να πατήσει το κουμπί ‘Add device to schedule’, ώστε να προστεθεί η συσκευή στην λίστα. Έτσι, πλέον, μπορεί να ρυθμίσει τις ενέργειες που θα εκτελέσει η συσκευή κατά την έναρξη αλλά και κατά την λήξη του προγράμματος.

Σε περίπτωση επικαλυπτόμενων προγραμμάτων, κατά την προσπάθεια αποθήκευσης εμφανίζεται μία ακόμα ενότητα, όπως της εικόνας 5.11. Έτσι ο χρήστης πληροφορείται πως υπάρχει “σύγκρουση” μεταξύ δύο ή περισσότερων προγραμμάτων και η εφαρμογή του ζητάει να θέσει τις προτεραιότητες. Αρχικά, πρέπει να ορίσει

Overlapping Schedules

Name	Start datetime	End datetime	Priority
Daily Event1	01/10/2017 11:00	01/10/2017 17:00	
<button>Update Priorities</button>			

Εικόνα 5.11: Επικαλυπτόμενα προγράμματα

τις προτεραιότητες για τα προϋπάρχοντα προγράμματα μέσω της εικόνας 5.11 και αφού τις θέσει να πατήσει το κουμπί 'Update priorities' για να γίνει η αποθήκευση των προτεραιοτήτων. Τέλος, μέσω της φόρμας του τρέχοντος προγράμματος και του πεδίου 'Priority' ορίζει την προτεραιότητα για το πρόγραμμα.

5.1.5 Λειτουργίες κουμπιών άμεσης εκτέλεσης ενεργειών

Πατώντας τον σύνδεσμο 'Quick Buttons', από το μενού, επιτυγχάνεται η μετάβαση στις λειτουργίες των κουμπιών άμεσης εκτέλεσης ενεργειών και συγκεκριμένα στην αρχική οθόνη αυτών. Στην εικόνα 5.12 εμφανίζεται η σύνοψη αυτή.

Quick Buttons List					
Title	Description	Duration	Index	Device Name	Actions
No1	This is a test QB	5	0	Thermostat	  
New Quick Button					

Εικόνα 5.12: Σύνοψη quick buttons χρήστη

Οι λειτουργίες των Quick Buttons παρουσιάζουν ομοιότητες με τις λειτουργίες των συσκευών. Πιο συγκεκριμένα, στη σύνοψη, η στήλη 'actions' παρέχει ακριβώς τις ίδιες ενέργειες. Οι υπόλοιπες στήλες διαμορφώνονται ως εξής: πρώτα εμφανίζεται ο τίτλος του quick button, ενώ μετά υπάρχει η περιγραφή του. Εν συνεχεία υπάρχει η στήλη με την διάρκεια του quick button, η οποία εκφράζει την χρονική διάρκεια για την οποία οι ενέργειες του quick button θα πρέπει να έχουν ισχύ. Η ιδιότητα 'Index on device' παρουσιάζει την ίδια ακριβώς λειτουργία με αυτή των συσκευών, δηλαδή την αντιστοίχιση με τις τοπικές δομές της συσκευής. Τέλος, εμφανίζεται το όνομα της συσκευής για την οποία έχει οριστεί το quick button. Στο κάτω μέρος της σύνοψης βρίσκεται ο σύνδεσμος για την μεταφορά στην εμφάνιση ρύθμισης νέου

quick button.

- **New Quick Button:** Από την παρουσίαζει της εμφάνισης στην εικόνα 5.13, γίνεται η ρύθμιση ενός νέου quick button για μία συγκεκριμένη συσκευή. Αρχικά, ορίζεται ένας συνοπτικός τίτλος ο οποίος περιγράφει την λειτουργικότητα του quick button, ενώ στο πεδίο ‘description’, προαιρετικά, μπορεί να εισαχθεί μια αναλυτικότερη περιγραφή. Αμέσως μετά ορίζεται το ‘index on device’. Στο πεδίο ‘Duration’, όπως προειπώθηκε, ορίζεται η χρονική διάρκειά για την οποία οι ρυθμίσεις του quick button θα πρέπει να έχουν ισχύ. Τελευταία είναι η περιοχή που αφορά τη ρύθμιση των ενεργειών του quick button για την συσκευή που έχει επιλεγεί.

The form is titled "New Quick Button". It contains the following fields and controls:

- Title:** A text input field.
- Description:** A text input field with a dropdown menu below it showing "Thermostat" and a close button "x".
- Index on device:** A text input field.
- Duration (s):** A text input field.
- Device attribute:** A dropdown menu showing "ON/OFF".
- Device attribute start value:** A text input field.
- Device attribute end value:** A text input field.
- Temperature:** A text input field.
- Create Quick button:** A button at the bottom left.

Εικόνα 5.13: Δημιουργία νέου quick button

- **Edit Quick Button:** Στην εμφάνιση της επεξεργασίας, που είναι ίδια με αυτή της εισαγωγής, τα πεδία είναι προσυμπληρωμένα με αυτά που εισήγαγε ο χρήστης κατά την δημιουργία νέου quick button. Όπως και στην επεξεργασία της συσκευής, έτσι και εδώ, ο χρήστης μπορεί να μεταβάλει υπάρχοντα πεδία, να συμπληρώσει τα άδεια αλλά και να μεταβάλει την συσκευή στην οποία αναφέρεται το quick button, όπως και τις ενέργειες της συσκευής.
- **Αναλυτικές πληροφορίες Quick Button:** Αυτή είναι η εμφάνιση με την οποία ο χρήστης λαμβάνει μια αναλυτική πληροφόρηση σχετικά με τις ρυθμίσεις του quick button.

Detailed view for the Quick Button		
No1 @ Thermostat		
This is a test QB		
Settings		
Index on Device: 0	Duration: 5	
Attribute	Device attribute start value	Device attribute end value
ON/OFF	ON	ON
Temperature	29.0 °C	20.0 °C

Εικόνα 5.14: Αναλυτικές πληροφορίες ενός Quick Button

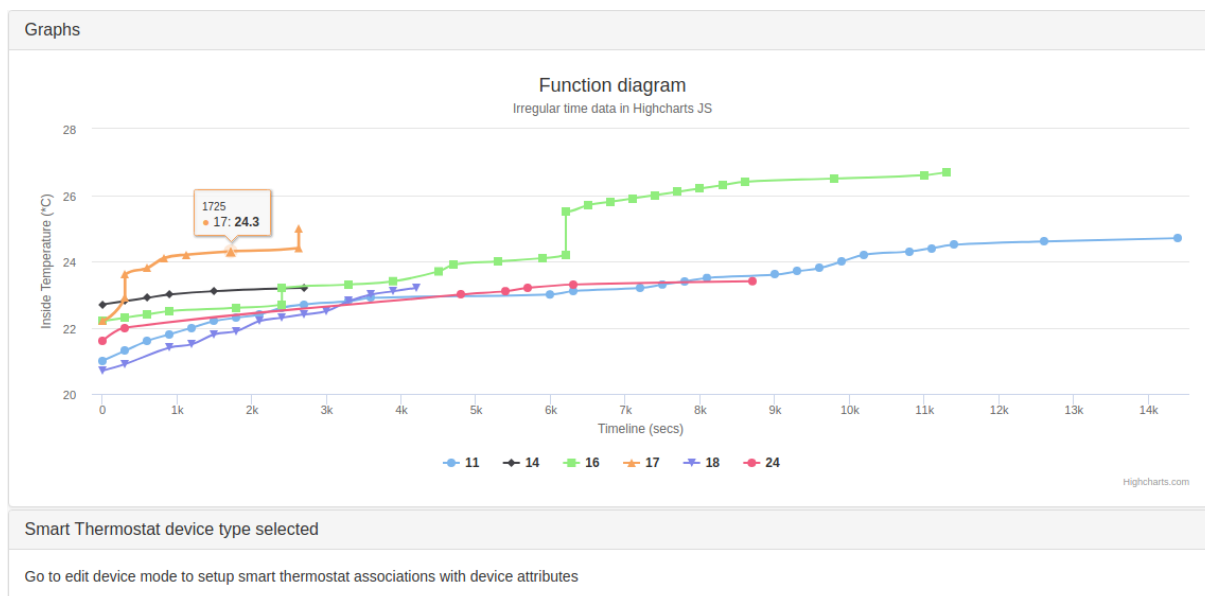
5.2 Ανάλυση λειτουργιών παρασκηνίου συστήματος

Παρακάτω εξετάζονται οι εργασίες οι οποίες επιτελούνται από το σύστημα στο παρασκήνιο. Οι εργασίες αυτές είναι απαραίτητες για την κάλυψη ορισμένων απαιτήσεων, οι οποίες έχουν την ανάγκη για συνεχή επιτήρηση του χώρου και ανανέωση των δεδομένων ανά τακτά χρονικά διάστημα και όχι μόνο κατά το διάστημα που ο χρήστης περιηγείται στην ιστοσελίδα του συστήματος.

5.2.1 Λειτουργία έξυπνου θερμοστάτη

Στην ενότητα 5.1.3, κατά την ανάλυση των λειτουργιών της συσκευής, αναφέρεται ότι στην περίπτωση που ο τύπος της συσκευής έχει οριστεί ως ‘έξυπνος θερμοστάτης’ προκειμένου να ενεργοποιηθεί η λειτουργία αυτή είναι απαραίτητες ορισμένες ρυθμίσεις. Οι εν λόγω ρυθμίσεις φαίνονται στην εικόνα 5.6. Πρόσβαση αποκτά κάποιος σε αυτές μόνο μέσω της ενέργειας της επεξεργασίας της συσκευής μετά την αρχική της αποθήκευση. Η ενεργοποίηση της λειτουργίας ‘έξυπνου θερμοστάτη’ αλλάζει ορισμένα κομμάτια στην εμφάνιση της ιστοσελίδας στις αναλυτικές πληροφορίες της συσκευής και τέλος ενεργοποιείται ένας αριθμός από ενέργειες παρασκηνίου.

Αρχικά, όσον αφορά τις ρυθμίσεις, απαιτείται η πληροφορία σχετικά με την εξωτερική θερμοκρασία και ορίζεται μέσω της γραμμής ‘Outside Temperature’. Για την ρύθμιση αυτή υπάρχει και η δυνατότητά λήψης του δεδομένου από διαφορετική συσκευή, μιας και η τρέχουσα δεν είναι απαραίτητο να διαθέτει δυο αισθητήρες θερ-



Εικόνα 5.15: Αναλυτικές πληροφορίες συσκευής - Καμπύλες λειτουργίας

μοκρασίας. Η επόμενη ρύθμιση, στη γραμμή ‘Inside Temperature’, χρησιμοποιείται για τη λήψη της εσωτερικής θερμοκρασίας και προϋποθέτει την ύπαρξη ανάλογου αισθητήρα στην τρέχουσα συσκευή. Ενώ, με την γραμμή ‘Set Temperature’ γίνεται η λήψη της πληροφορίας σχετικά με την επιθυμητή θερμοκρασία που έχει ο ορίσει ο χρήστης. Τέλος, από την γραμμή ‘Working Status’ λαμβάνεται η πληροφορία σχετικά με την κατάσταση λειτουργίας ή μη της πηγής ενέργειας.

Οι αλλαγές στην εμφάνιση της ιστοσελίδας φαίνονται στην εικόνα 5.15. Το διάγραμμα αυτό προστίθεται στις αναλυτικές πληροφορίες της συσκευής. Πρώτα, στο κάτω μέρος της φωτογραφίας εμφανίζεται το ενημερωτικό μήνυμα, προς τον χρήστη, που τον ειδοποιεί πως για την τροποποίηση των ρυθμίσεων της λειτουργίας ‘έξυπνου θερμοστάτη’, πρέπει να μεταβεί στην κατάσταση τροποποίησης της συσκευής. Στο διάγραμμα απεικονίζεται, επίσης, ο τρόπος απόκρισης του χώρου στην προσφορά θερμότητας σε σχέση με την εξωτερική θερμοκρασία κατά την πάροδο του χρόνου και για τον χώρο στον οποίο βρίσκεται εγκατεστημένη η συσκευή.

Τέλος, σχετικά με τις εργασίες παρασκήνιου, το σύστημα αναλαμβάνει να επιτελέσει ένα πλήθος ενεργειών ώστε να φέρει εις πέρας τη λειτουργία του ‘έξυπνου θερμοστάτη’. Οι ενέργειες είναι απαραίτητες για τη δημιουργία συνόλων δεδομένων που χρειάζεται η λειτουργία ‘έξυπνου θερμοστάτη’. Μέσα από αυτά τα σύνολα δεδομένων δημιουργείται και το διάγραμμα που αναφέρθηκε. Η παρουσίαση των εργασιών που αναφέρθηκαν πραγματοποιείται στην ενότητα 5.2.2.

5.2.2 Cron Jobs & Rake Tasks

Για την εκτέλεση εργασιών στο παρασκήνιο της εφαρμογής χρησιμοποιήθηκε η τεχνολογία Cron Jobs. Το cron είναι ένα λογισμικό το οποίο εκτελείται στον υπολογιστή εξυπηρετητή της εφαρμογής του συστήματος. Το cron αποτελεί, επίσης, ένα σύστημα το οποίο αναλαμβάνει την επαναλαμβανόμενη εκτέλεση εργασιών βάσει ενός αρχείου ρυθμίσεων.

Οι εργασίες του συστήματος δημιουργήθηκαν με την βοήθεια του υποσυστήματος rake, το οποίο είναι ένα κομμάτι του Framework Rails, που χρησιμοποιήθηκε για την κατασκευή της εφαρμογής του συστήματος.

Το σύστημα απαιτεί τη συνεχή εκτέλεση τριών εργασιών για την επιτυχή λειτουργία του 'έξυπνου θερμοστάτη' και μόνο μία για την ειδοποίηση χρηστών σε περίπτωση απώλειας σύνδεσης με τη συσκευή.

Κατά την πρώτη εργασία λαμβάνονται τα δείγματα θερμοκρασίας καθώς και της κατάστασης λειτουργίας, ανά τακτά και σχετικά σύντομα χρονικά διαστήματα, τα οποία αποθηκεύονται στη βάση δεδομένων του συστήματος. Η λήψη των δειγμάτων γίνεται ξεχωριστά για κάθε συσκευή 'έξυπνου θερμοστάτη' και βάσει, φυσικά, των ρυθμίσεων που έχουν οριστεί στην κατάσταση τροποποίησης της συσκευής, μέσω της ενότητας ρυθμίσεων που παρουσιάζεται στην εικόνα 5.6. Στη δεύτερη, αναλύονται τα δείγματα που καταχωρήθηκαν κατά την εκτέλεση της πρώτης εργασίας. Κατασκευάζεται η δομή της εξέλιξης της θερμοκρασίας σε μορφή χρονοδιαγράμματος σε σχέση με την εξωτερική. Οι ενέργειες αυτές είναι απαραίτητες για την κατασκευή του διαγράμματος της εικόνας 5.15. Ο έλεγχος των προγραμμάτων των συσκευών που θέτουν οι χρήστες πραγματοποιείται κατά την διάρκεια εκτέλεσης της τρίτης εργασίας. Ελέγχεται ουσιαστικά εάν χρειάζεται η πρωτύτερη εκκίνηση ενός προγράμματος που έχει θέσει ο χρήστης την χρονική στιγμή εκείνη, προκειμένου το σύστημα να επιτύχει την επιθυμητή αυτή θερμοκρασία.

Στην τελευταία εργασία επιτελείται μια ξεχωριστή λειτουργία η οποία δεν σχετίζεται με τις τρεις προηγούμενες εργασίες. Ο στόχος είναι η ενημέρωση των χρηστών των εκάστοτε συσκευών σε περίπτωση που αυτές έχουν διακόψει την επικοινωνία με την εφαρμογή για κάποιο προκαθορισμένο χρονικό διάστημα.

5.3 Ανάλυση λογισμικού Arduino

Στο τελευταίο κομμάτι της διπλωματικής αυτής εργασίας, πλην του επιλόγου, γίνεται αναφορά στον προγραμματισμό του μικρο-ελεγκτή Arduino Uno. Σε πρώτο στάδιο, θα πραγματοποιηθεί η ανάλυση της λειτουργίας της συσκευής ώστε να γίνει πιο εύκολη η κατανόηση του κώδικα που αναπτύχθηκε. Στη συνέχεια θα παρουσιαστούν οι βιβλιοθήκες, καθώς και οι συναρτήσεις οι οποίες χρησιμοποιήθηκαν κατά την ανάπτυξη του προγράμματος του Arduino.

5.3.1 Ανάλυση λειτουργίας συσκευής

Κατά την εκκίνηση της συσκευής αρχικοποιούνται τρεις Ethernet clients, ένας TCP για τα POST και GET ερωτήματα προς την εφαρμογή, ένας UDP για τη διασύνδεση με τον NTP server και ένας για τον MQTT client. Κατόπιν, εκκινείται η σειριακή θύρα και εμφανίζει κατάλληλο μήνυμα εκκίνησης της συσκευής στο χρήστη. Ακολουθεί η ρύθμιση των ακροδεκτών του Arduino για να τεθούν στην κατάλληλη κατάσταση και αρχικοποιείται η κάρτα δικτύου Ethernet. Έπειτα, αρχικοποιείται και η διασύνδεση με τον NTP server και πραγματοποιείται η σύνδεση με τον MQTT broker. Εφόσον έχει πραγματοποιηθεί η ρύθμιση της δικτύωσης της συσκευής, αποστέλλεται POST ερώτημα από τη συσκευή στην εφαρμογή για την ενημέρωση της τελευταίας σχετικά με βασικά στοιχεία της πρώτης. Στο ερώτημα αυτό περιλαμβάνεται η έκδοση, ο σειριακός αριθμός και η τρέχουσα IP διεύθυνση της συσκευής. Έπεται, το ερώτημα τύπου GET από την συσκευή προς την εφαρμογή για την αποστολή δεδομένων μέσω του MQTT καναλιού, σχετικά με τα προγράμματα, τα quick buttons και τις τρέχουσες ρυθμίσεις των χαρακτηριστικών. Τέλος, γίνεται η ενεργοποίηση του watchdog, η αρχικοποίηση των led και η εκτύπωση στη σειριακή θύρα του σειριακού αριθμού και της έκδοσης Firmware της συσκευής. Πλέον, η συσκευή έχει αρχικοποιηθεί και είναι έτοιμη να ξεκινήσει την κανονική της λειτουργία.

Στο στάδιο κανονικής λειτουργίας της συσκευής πραγματοποιείται έλεγχος για το αν έχουν πατηθεί κάποια από τα κουμπιά της συσκευής και ενημερώνεται η κατάσταση των led. Ακόμα, η εφαρμογή πληροφορείται σχετικά με την κατάσταση της συσκευής. Πιο αναλυτικά, η συσκευή παρακολουθεί αν έχει πατηθεί το κουμπί ενεργοποίησης/απενεργοποίησης του έλεγχου της εξόδου καθώς και αν έχει πατη-

θεί κάποιο από τα quick buttons ώστε να δράσει ανάλογα. Στη συνέχεια, γίνεται ο έλεγχος για το εάν κάποιο ενεργοποιημένο quick button πρέπει να σταματήσει να λειτουργεί, επειδή η χρονική διάρκεια ισχύς του έχει λήξει. Έπειτα, ελέγχεται αν κάποιο πρόγραμμα πρέπει να ενεργοποιηθεί ή να απενεργοποιηθεί. Αξίζει να σημειωθεί ότι, το πρόγραμμα δίνει προτεραιότητα στις λειτουργίες των quick buttons παρακάμπτοντας το πρόγραμμα εάν αυτά ενεργοποιηθούν ταυτόχρονα.

Μετά την ολοκλήρωση των παραπάνω ελέγχων, η συσκευή στέλνει εντολή διατήρησης της σύνδεσης με τον MQTT broker. Ο Ethernet client διαβάζει την HTTP απάντηση από τα HTTP POST και GET ερωτήματα που έχει κάνει η συσκευή τυπώνοντας το μήνυμα της απάντησης σε περίπτωση σφάλματος στη σειριακή θύρα. Κατόπιν, αν έχει τελειώσει η HTTP επικοινωνία τότε τερματίζεται η τρέχουσα TCP σύνδεση.

Στη συνέχεια εκτελείται η συνάρτηση η οποία διαβάζει τις τιμές του αισθητήρα και ανανεώνει κατάλληλα τις τιμές των χαρακτηριστικών στις τοπικές δομές που διατηρεί. Ακόμα, ενημερώνει κατάλληλα την έξοδο της συσκευής σχετικά με το αν πρέπει να ξεκινήσει να λειτουργεί ή όχι και ενημερώνονται τα κατάλληλα led.

Υστερα από όλα αυτά, η συσκευή ενημερώνει την εφαρμογή σχετικά με την κατάσταση της. Αυτό δε συμβαίνει, όμως, σε κάθε εκτέλεση του κεντρικού βρόχου επανάληψης της κανονικής λειτουργίας της συσκευής, αλλά σε πιο αραιά χρονικά διαστήματα. Η ενημέρωση επιτυγχάνεται με τη χρήση χρονομέτρου, το οποίο μόλις λήξει η συσκευή αποστέλλει απευθείας το κατάλληλο HTTP POST ερώτημα προς την εφαρμογή και έπειτα επαναρυθμίζει το χρονόμετρο. Τέλος, εκτελείται η εντολή για την διατήρηση της σύνδεσης με το διαδίκτυο, η εντολή για την λήψη των MQTT μηνυμάτων και μηδενίζεται το χρονόμετρο του watchdog.

5.3.2 Παρουσίαση βιβλιοθηκών και συναρτήσεων

Για τον προγραμματισμό της συσκευής χρησιμοποιήθηκαν οι παρακάτω εννέα βιβλιοθήκες:

- Arduino.h
- ArduinoJson.h
- avr/pgmspace.h

-
- `avr/wdt.h`
 - `dht.h`
 - `Ethernet.h`
 - `EthernetUdp.h`
 - `PubSubClient.h`
 - `SPI.h`
 - `TimeLib.h`

Arduino.h [6]

Η βιβλιοθήκη `Arduino.h` [4] παρέχει τις βασικές λειτουργίες για τον προγραμματισμό μιας πλακέτας `Arduino`. Ενδεικτικά, αναφέρονται παρακάτω ορισμένες από τις συναρτήσεις που χρησιμοποιήθηκαν στην παρούσα διπλωματική εργασία.

- `pinMode(pin, mode)`: Ρυθμίζει τον καθορισμένο ακροδέκτη ώστε να συμπεριφέρεται είτε ως είσοδος είτε ως έξοδος. Όπου `pin` τοποθετείται ο αριθμός του ακροδέκτη που πρόκειται να ρυθμιστεί. Ενώ, όπου `mode` τοποθετείται μία από τις σταθερές `INPUT`, `OUTPUT`, ή `INPUT_PULLUP` ανάλογα με το αν ο προγραμματιστής επιθυμεί ο ακροδέκτης του να είναι είσοδος, έξοδος, ή είσοδος με ενεργοποιημένη την εσωτερική `pullup` αντίσταση του `Arduino`. Αξίζει να σημειωθεί ότι, η παράμετρος `INPUT` απενεργοποιεί ρητά τις εσωτερικές `pullup` αντιστάσεις.
- `digitalWrite(pin, value)`: Δίνει μια τιμή `HIGH` ή `LOW` σε έναν ψηφιακό ακροδέκτη. Όπου `pin` τοποθετείται ο αριθμός του ακροδέκτη και όπου `value` μια εκ των τιμών `HIGH` ή `LOW`.

Εάν ο ακροδέκτης έχει ρυθμιστεί ως `OUTPUT` με χρήση της `pinMode()`, τότε η τάση του θα ρυθμιστεί στην αντίστοιχη τιμή των 5V για `HIGH` και 0V (`ground`) για `LOW`.

Αν, από την άλλη, ο ακροδέκτης έχει οριστεί ως `INPUT`, το `digitalWrite()` θα ενεργοποιήσει ή θα απενεργοποιήσει την εσωτερική `pullup` αντίσταση στον ακροδέκτη εισόδου. Συνιστάται να ρυθμιστεί με την `pinMode()` ως `INPUT_PULLUP`

ο συγκεκριμένος ακροδέκτης, για να ενεργοποιηθεί η εσωτερική pull-up αντίσταση.

Προσοχή χρειάζεται να δοθεί στο ότι εάν δεν έχει οριστεί το `pinMode()` σε OUTPUT για τον ακροδέκτη και έχει συνδεθεί κάποιο ηλεκτρονικό στοιχείο σε αυτόν τον ακροδέκτη, κατά την κλήση της ψηφιακής εγγραφής (HIGH), τότε αυτό μπορεί να υπολειτουργεί. Χωρίς τη ρητή ρύθμιση του `pinMode()`, το `digitalWrite()` θα ενεργοποιήσει την εσωτερική αντίσταση pull-up, η οποία περιορίζει το ρεύμα προς το συνδεδεμένο ηλεκτρονικό στοιχείο.

- `digitalRead(pin)`: Λαμβάνει την τιμή ενός ψηφιακού ακροδέκτη, HIGH ή LOW. Όπου `pin` τοποθετείται ο αριθμός του ακροδέκτη από τον οποίο επιθυμεί κάποιος να διαβάσει τις τιμές του.
- `shiftOut(dataPin, clockPin, bitOrder, value)`: Μεταθέτει ένα byte δεδομένων από ένα bit τη φορά. Ξεκινάει από το πιο σημαντικό (MSBFIRST) ή από το λιγότερο σημαντικό (LSBFIRST) ψηφίο. Κάθε bit γράφεται με τη σειρά του στον ακροδέκτη δεδομένων ανά έναν παλμό ρολογιού.

Όπου `dataPin` ο ακροδέκτης εξόδου σειριακών δεδομένων, `clockPin` ο παλμός ρολογιού, `bitOrder` η σειρά με την οποία θα διαβάσει τα bits (MSBFIRST ή LSBFIRST) και `value` τα δεδομένα που θα εξαχθούν (byte).

ArduinoJson.h [7]

Η βιβλιοθήκη `ArduinoJson` χρησιμοποιείται για την μετατροπή των JSON δεδομένων, που λαμβάνονται από την εφαρμογή του συστήματος, σε πίνακες C++, προκειμένου να χρησιμοποιηθούν από το πρόγραμμα του Arduino.

Στη συγκεκριμένη διπλωματική χρησιμοποιήθηκαν οι παρακάτω συναρτήσεις, κλάσεις και μεταβλητές:

- `StaticJsonBuffer<200> jsonBuffer`: Προ-καταλαμβάνει ένα ποσό μνήμης στην SRAM του Arduino, για τη δημιουργία του JSON πίνακα. Ο αριθμός '200' υποδηλώνει το μέγεθος της κατειλημμένης μνήμης σε bytes. Για πολύπλοκα JSON αντικείμενα, θα πρέπει να αυξηθεί αντίστοιχα το μέγεθος αυτό.
- `JsonObject& root = jsonBuffer.parseObject(json)`: Η συνάρτηση `parseObject()` κατανέμει τα δεδομένα από το json string σε ένα `JsonObject` δια της αναφο-

ράς. Αυτό σημαίνει ότι τα δεδομένα δεν αντιγράφονται από το `jsonBuffer` στο `JsonObject`, αλλά το `JsonObject` περιέχει αναφορές προς το `jsonBuffer` σχετικά με το που μπορεί να βρει το κάθε δεδομένο. Όπου `JsonObject` είναι η ρίζα του JSON δέντρου.

avr/pgmspace.h [8]

Οι συναρτήσεις της συγκεκριμένης βιβλιοθήκης παρέχουν διεπαφές για την πρόσβαση στα δεδομένα που είναι αποθηκευμένα στη μνήμη flash του Arduino UNO. Για να χρησιμοποιηθούν αυτές οι συναρτήσεις, η συσκευή πρέπει να υποστηρίζει τις οδηγίες LPM [21] ή ELPM [14]. Οι συναρτήσεις που χρησιμοποιήθηκαν είναι οι παρακάτω:

- `strlen_P(const char *, size_t)`: Επιστρέφει το μήκος ενός string μη μεταβλητού μεγέθους. Η συνάρτηση `strlen_P()` είναι παρόμοια με την `strlen()`, με τη διαφορά ότι το όρισμα της δεν είναι το string καθ' αυτό, αλλά ένας δείκτης στη μνήμη flash όπου είναι αποθηκευμένο το string.
- `pgm_read_byte_near(address_short)`: Διαβάζει ένα byte από τη μνήμη flash της συσκευής και δέχεται σαν όρισμα μία 16-bit διεύθυνση.

avr/wdt.h [9]

Η συγκεκριμένη βιβλιοθήκη χρησιμοποιήθηκε για την υλοποίηση του watchdog του μηχανισμού, ο οποίος επανεκκινεί αυτόματα τη συσκευή εάν αυτή κολλήσει κατά την εκτέλεση του προγράμματος και τεθεί προσωρινά εκτός φυσιολογικής λειτουργίας. Οι συναρτήσεις και οι εντολές της βιβλιοθήκης που χρησιμοποιήθηκαν ακολουθούν παρακάτω:

- `wdt_enable(time)`: Ενεργοποιεί το χρονόμετρο του watchdog, και το προγραμματίζει να λήξει μετά από μια ορισμένη χρονική στιγμή, όπου και θα πραγματοποιηθεί η επανεκκίνηση της συσκευής. Όπου `time` είναι μια σταθερά η οποία έχει οριστεί στη βιβλιοθήκη του watchdog. Στην παρούσα διπλωματική εργασία επιλέχθηκαν το 8s. Η αντίστοιχη σταθερά για αυτό το χρονικό διάστημα είναι η `WDTO_8S`.
- `wdt_reset()`: Επανεκκινεί το χρονόμετρο του watchdog. Αποτυχία εκτέλεσης της συνάρτησης αυτής πριν τη λήξη του χρονομέτρου σημαίνει επανεκκίνηση της συσκευής.

dht.h [11]

Η βιβλιοθήκη `dht.h` χρησιμοποιείται για την ανάγνωση των δεδομένων της θερμοκρασίας και της υγρασίας από τον αισθητήρα DHT22. Επιστρέφει τις τιμές αυτών μέσω των ορισμάτων `dht22.temperature` και `dht22.humidity`, αντίστοιχα.

- `dht22.read22(pin)`: Διαβάζει τις τιμές που καταγράφει ο αισθητήρας DHT22. Όπου `pin` τοποιοτείται ο αριθμός του ακροδέκτη στον οποίο έχει συνδεθεί το data `pin` του αισθητήρα DHT22.

Ethernet.h & EthernetUdp.h [15]

Πρόκειται για την βιβλιοθήκη η οποία πραγματώνει τη σύνδεση με το διαδίκτυο, μέσω του Ethernet shield. Μπορεί να χρησιμοποιηθεί είτε ως server, είτε ως client. Η βιβλιοθήκη υποστηρίζει έως και τέσσερις ταυτόχρονες συνδέσεις (εισερχόμενες ή και εξερχόμενες). Ακολουθεί περιγραφή των εντολών και των συναρτήσεων της βιβλιοθήκης οι οποίες χρησιμοποιήθηκαν στην παρούσα διπλωματική εργασία.

- `Ethernet.begin(mac, ip, dns, gateway, subnet)`: Αρχικοποιεί τη βιβλιοθήκη Ethernet και τις ρυθμίσεις του δικτύου. Η βιβλιοθήκη Ethernet υποστηρίζει την δυνατότητα απόκτησης διεύθυνσης IP μέσω DHCP με την χρήση της κλήσης `Ethernet.begin(mac)`. Αξίζει να σημειωθεί ότι, η χρήση του DHCP αυξάνει το μέγεθος του προγράμματος σημαντικά.
- `Ethernet.maintain()`: Επιτρέπει την ανανέωση των μισθώσεων από το διακομιστή DHCP. Όταν εκχωρείται μια διεύθυνση IP μέσω DHCP, οι συσκευές Ethernet λαμβάνουν μίσθωση διεύθυνσης IP για ένα χρονικό διάστημα. Με την `Ethernet.maintain()`, είναι δυνατό να ζητηθεί η ανανέωση της από τον DHCP. Ανάλογα με τη διαμόρφωση του DHCP, ενδέχεται να ληφθεί η ίδια διεύθυνση, νέα διεύθυνση ή καμία.
- `Client()`: Είναι η βασική κλάση για όλες τις Ethernet client συναρτήσεις. Δεν καλείται απευθείας, αλλά περιέχεται στην κλήση συνάρτησης που αναφέρονται σε αυτή.
- `client.connect(IP, port)` ή `client.connect(URL, port)`: Πραγματοποιεί σύνδεση με έναν διακομιστή στην διεύθυνση IP ή στην διεύθυνση URL και τη θύρα που

καθορίζεται. Η τιμή επιστροφής δείχνει την επιτυχία ή την αποτυχία της σύνδεσης.

- `client.stop()`: Αποσύνδεση από το διακομιστή.
- `client.print(data_to_print)`: Παρέχει δεδομένα στο διακομιστή με τον οποίο έχει συνδεθεί.
- `client.connected()`: Ελέγχει εάν έχει επιτευχθεί η σύνδεση με το διακομιστή και επιστρέφει τιμές `true` ή `false`.
- `client.find(data_to_find)`: Αναζητεί τα δεδομένα εισόδου (`data_to_find`) στην επιστροφή του διακομιστή.
- `EthernetUDP.begin(localPort)`: Αρχικοποιεί μια UDP σύνδεση στην τοπική `localPort`.
- `UDP.beginPacket(remoteIP, remotePort)`: Ξεκινά μια σύνδεση με τον απομακρυσμένο διακομιστή στην διεύθυνση και την πόρτα που ορίζονται μέσω των παραμέτρων με σκοπό την αποστολή δεδομένων UDP.
- `UDP.endPacket()`: Καλείται μετά την εγγραφή των δεδομένων UDP στην απομακρυσμένη σύνδεση.
- `UDP.parsePacket()`: Ελέγχει την παρουσία ενός πακέτου UDP και αναφέρει το μέγεθος του. Η `parsePacket()` πρέπει να καλείται πριν από την ανάγνωση του buffer με `UDP.read()`.
- `UDP.read(packetBuffer, MaxSize)`: Αναγιγνώσκει δεδομένα UDP από το καθορισμένο buffer. Αν δε δοθούν παράμετροι, θα επιστραφεί ο επόμενος χαρακτήρας στο buffer.
- `UDP.write(message)` ή `UDP.write(buffer, size)`: Γράφει δεδομένα UDP στην απομακρυσμένη σύνδεση. Πρέπει να καλείται μεταξύ `beginPacket()` και `endPacket()`. Η `beginPacket()` αρχικοποιεί το πακέτο δεδομένων, αλλά αυτό δεν αποστέλλεται μέχρι να κληθεί η `endPacket()`.

PubSubClient.h [1]

Η βιβλιοθήκη αυτή, παρέχει ένα πρόγραμμα-πελάτη για τη δημοσίευση μηνυμάτων ή εγγραφή σε θέματα για τη λήψη μηνυμάτων, με ένα διακομιστή που υποστηρίζει MQTT. Παρακάτω παρατίθενται οι συναρτήσεις που χρησιμοποιήθηκαν στην παρούσα διπλωματική εργασία.

- `PubSubClient(server, port, callback, EthernetClient)`: Ο Constructor για την αρχικοποίηση της βιβλιοθήκης. Όπου `callback` η συνάρτηση που καλείται για τη λήψη δεδομένων από τον broker.
- `mqttClient.connect(clientID)`: Συνδέει τον συγκεκριμένο `mqttClient` στον MQTT broker με το αναγνωριστικό `clientID`.
- `mqttClient.loop()`: Η συνάρτηση αυτή, θα πρέπει να καλείται τακτικά στο βρόχο επανάληψης κανονικής λειτουργίας της συσκευής, προκειμένου να επιτρέπεται στον `mqttClient` να επεξεργάζεται τα εισερχόμενα μηνύματα και να διατηρήσει τη σύνδεσή του με τον broker.
- `mqttClient.subscribe(topic)`: Με την κλήση της συνάρτησης, εγγράφεται στο συγκεκριμένο `topic`.

SPI.h [35]

Η συγκεκριμένη βιβλιοθήκη επιτρέπει την επικοινωνία με άλλες SPI συσκευές, με το Arduino να λειτουργεί ως την κύρια συσκευή. Σειριακή περιφερειακή διεπαφή (SPI) είναι ένα σύγχρονο πρωτόκολλο δεδομένων που χρησιμοποιείται από μικροελεγκτές για την επικοινωνία με μία ή περισσότερες περιφερειακές συσκευές σε μικρές αποστάσεις. Μπορεί επίσης, να χρησιμοποιηθεί για την επικοινωνία μεταξύ δύο μικροεπεξεργαστών. Στη συσκευή που κατασκευάστηκε, χρησιμοποιείται για σύνδεση του Arduino Uno με το Ethernet Shield.

TimeLib.h [29]

Η `TimeLib.h` είναι μια βιβλιοθήκη που παρέχει λειτουργικότητα μέτρησης χρόνου στο Arduino. Χρησιμοποιήθηκαν οι κάτωθι συναρτήσεις:

- `setSyncProvider(getTimeFunction)`: Ορίζεται ο πάροχος από τον οποίο θα γίνεται η λήψη της ώρας για τον συγχρονισμό του τοπικού ρολογιού του Arduino. Στη συγκεκριμένη διπλωματική επιλέχθηκε ο συγχρονισμός μέσω NTP Server.

-
- `now()`: Η συνάρτηση `now()` επιστρέφει την τρέχουσα ώρα που διατηρεί τοπικά το Arduino.

5.4 Μετρικές κώδικα

Στο κεφάλαιο αυτό αναλύθηκε το λογισμικό που αναπτύχθηκε στα πλαίσια του συστήματος τόσο για την ιστοσελίδα όσο και για την συσκευή Arduino. Κλείνοντας το, θα παρουσιαστούν οι μετρικές κώδικα των δύο αυτών λογισμικών σε επίπεδο γραμμών.

5.4.1 Rails

Οι μετρικές για την ιστοσελίδα, ομαδοποιήθηκαν ανά ενότητα, μιας και η Rails είναι ένα MVC framework.

Ομάδα αρχείων	Κώδικας	Κλάσεις	Μέθοδοι
Controllers	942	6	31
Helpers	197	0	7
Models	180	16	3
Javascripts	848	0	87
Libraries(Rake tasks)	200	0	0

Πίνακας 5.1: Μετρικές κώδικα της ιστοσελίδας του συστήματος

Να σημειωθεί ότι, στις μετρικές αυτές δεν συμπεριλαμβάνονται στοιχεία σχετικά με τον πυρήνα της Rails και των βιβλιοθηκών της Javascript και του CSS.

5.4.2 Arduino

Οι μετρικές για το λογισμικό του Arduino παρουσιάζονται ανά αρχείο.

Οι μετρικές αφορούν μόνο των κώδικα που αναπτύχθηκε στα πλαίσια της διπλωματικής εργασίας και όχι των εξωτερικών βιβλιοθηκών που χρησιμοποιήθηκαν.

Όνομα αρχείου	Κώδικας	Σχόλια	Σύνολο
src/DataStructures.h	31	0	37
src/MqttCallbacks.cpp	13	2	17
src/MqttCallbacks.h	8	0	11
src/ProccessCallbacks.cpp	30	3	36
src/ProccessCallbacks.h	6	0	10
src/QuickButtons.cpp	46	0	51
src/QuickButtons.h	10	0	15
src/Requests.cpp	169	17	221
src/Requests.h	15	0	21
src/Schedule.cpp	51	3	64
src/Schedule.h	9	0	14
src/System.cpp	179	60	254
src/System.h	58	20	82
src/home-automation-arduino-client.ino	115	90	229

Πίνακας 5.2: Μετρικές κώδικα της συσκευής Arduino του συστήματος

Κεφάλαιο 6

Επίλογος

Το σύστημα το οποίο αναπτύχθηκε στη συγκεκριμένη διπλωματική εργασία εξυπηρετεί τον αυτοματοποιημένο και απομακρυσμένο έλεγχο ορισμένων λειτουργιών ενός σπιτιού. Απώτερος σκοπός είναι να αυτοματοποιηθούν όσο το δυνατόν περισσότερες λειτουργίες. Συνδυάζοντας τη χρήση συσκευών υλικού για τη λήψη δεδομένων, μέσω κατάλληλων αισθητήρων, και ενός λογισμικού για τη διαχείρισή τους καθώς και τη λήψη αποφάσεων, κατέστη εφικτός ο σχεδιασμός και η υλοποίηση του έργου αυτού. Η σύσταση αυτού του συστήματος μπορεί, βέβαια, να επεκταθεί με την ενσωμάτωση ολοένα και περισσότερων συσκευών και να αυτοματοποιηθεί ακόμα περισσότερο, έτσι ώστε να επιτευχθεί η μέγιστη εξοικονόμηση κάθε τύπου ενέργειας και χρόνου. Όπως ήταν λογικό ωστόσο, προέκυψαν ορισμένα προβλήματα κατά τη φάση του σχεδιασμού και της υλοποίησης τα οποία όμως ξεπεράστηκαν.

6.1 Σύνοψη και συμπεράσματα

Συνοψίζοντας, καταλαβαίνει κανείς ότι το έργο αυτό αποτελεί μια καινοτόμα ιδέα, η οποία συνδυάζει διαφορετικές τεχνολογίες για τον έλεγχο και την αυτοματοποίηση ενός σπιτιού. Επιπροσθέτως, συνδυάζει ομαλά την επικοινωνία των διαφόρων συσκευών με την εφαρμογή του συστήματος. Αποτέλεσμα αυτού, είναι η ανάγκη μίας κεντριοποιημένης διαχειριστικής εφαρμογής για όλο το σύστημα και όχι πολλαπλών όπου η κάθε μία θα μπορεί να επικοινωνεί με μια συσκευή μόνο. Χάρη στον αυτοέλεγχο και την αυτενέργεια του συστήματος, ο όρος σπίτι μετονομάζεται σε 'έξυπνο σπίτι'. Το λογισμικό, λοιπόν, σκέφτεται και αποφασίζει αυτόνομα, για τις ενέργειες οι οποίες πρέπει να εκτελεστούν ή όχι σε μία δεδομένη χρονική στιγμή. Κατ' επέκταση, ο όρος αυτός θεωρείται δόκιμος και για τις συσκευές του

σπιτιού, διότι πλέον αυτές δεν είναι άπλα μηχανικά στοιχεία τα οποία κάποιος άνθρωπος ενεργοποιεί και απενεργοποιεί χειροκίνητα. Πλέον, υπάρχει η δυνατότητα οι συσκευές να ‘μιλούν’ με άλλες οντότητες με τις οποίες ανταλλάσσουν πληροφορίες, σχετικά με την τρέχουσα κατάσταση τους και τις ενέργειες στις οποίες πρέπει να προβούν. Χαρακτηριστικό παράδειγμα μίας τέτοιας συσκευής, αποτελεί ο ‘έξυπνος θερμοστάτης’, που υλοποιήθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας, προς απόδειξη της λειτουργικότητας του όλου εγχειρήματος. Η επιλογή της αρχιτεκτονικής του Arduino για την ανάπτυξη του ‘έξυπνου θερμοστάτη’ έγινε με σκοπό την ταχύτερη δυνατή προτυποποίηση μίας υλικής κατασκευής για την απόδειξη ότι ο αρχικός σκοπός είναι εφικτός. Βέβαια, η τάση της διασύνδεσης των οικιακών συσκευών και μικροσυσκευών στο διαδίκτυο είναι αυτή που δημιουργήσε και έναν ακόμα όρο για τις ‘έξυπνες’ συσκευές, αυτόν του IoT.

Έτσι, συμπεραίνεται ότι η παρούσα διπλωματική εργασία αποτελεί ένα ισχυρό παράδειγμα το οποίο επιβεβαιώνει ότι η διασύνδεση των συσκευών και μικροσυσκευών μια οικίας με το διαδίκτυο αλλά και ο έλεγχος μέσω αυτού είναι απολύτως πραγματοποιήσιμη και ιδιαίτερα ανερχόμενη. Η διπλωματική αυτή εισάγει έναν πολύ σημαντικό κρίκο για το επόμενο βήμα στην εποχή των ‘έξυπνων’ σπιτιών, με τον ενοποιημένο πλέον έλεγχο ολοκλήρου του συστήματος. Η χρήση IoT αυξάνεται ακατάπαυστα και θα συνεχίσει να το κάνει σε μεγάλο βαθμό στο μέλλον, με αποτέλεσμα να προσφέρεται η δυνατότητα για ακόμα μεγαλύτερη αυτοματοποίηση και χρήση ενοποιημένων συστημάτων. Έτσι, θα μπορούν ολοένα και περισσότερες οικίες να αξιοποιούν τα καινοτόμα αυτά συστήματα.

6.2 Προβλήματα τα οποία προέκυψαν και η αντιμετώπιση τους

Σε κάθε καινοτόμο έργο, η πιθανότητα να ανακύψει κάποιο πρόβλημα είτε κατά τη φάση του σχεδιασμού είτε κατά τη διάρκεια της υλοποίησης του είναι μεγάλη, όπως συνέβη και με την παρούσα διπλωματική εργασία. Το πρόβλημα που εμφανίστηκε κατά τη φάση του σχεδίου ήταν ο αναξιόπιστος και μη αποδοτικός τρόπος ενημέρωσης των συσκευών από την πλευρά της εφαρμογής για νέα δεδομένα με άμεσο τρόπο.

Ο παραδοσιακός μηχανισμός, που υφίσταται για τη ροή των δεδομένων από μια κεντρική εφαρμογή προς του κόμβους, αποτελεί η επαναλαμβανόμενη ερώτηση από

την μεριά των κόμβων προς την εφαρμογή για την ύπαρξη νέων δεδομένων. Η μέθοδος αυτή, όμως, συνιστά μια κακή πρακτική, διότι απαιτεί αρκετούς πόρους και από τη μεριά του κόμβου/συσκευής αλλά και από την πλευρά της εφαρμογής. Η αυξημένη αυτή χρήση των πόρων έχει ως αποτέλεσμα τη μεγάλη κατανάλωση ενέργειας. Επομένως, ο παραδοσιακός μηχανισμός, είναι πλέον εμφανές ότι, δε συνάδει με τις ανάγκες μιας μικρο-συσκευής, σαν αυτή του θερμοστάτη. Τέτοιου είδους συσκευές, απαιτούν την όσο το δυνατόν πιο χαμηλή κατανάλωση ενέργειας, ειδικά εάν η τροφοδότηση τους γίνεται από μπαταρία. Οι συσκευές αυτές, μάλιστα, διαθέτουν και περιορισμένους πόρους γι' αυτό η χρήση τους πρέπει να γίνεται με φειδώ.

Εφόσον, ερευνήθηκε η χρήση κάποιου εναλλακτικού τρόπου επικοινωνίας, με λιγότερες απαιτήσεις και καλύτερα αποτελέσματα, τελικά εντοπίστηκε και κατόπιν αξιοποιήθηκε το πρωτόκολλο MQTT. Αυτό έχει κατασκευαστεί ώστε να αντιμετωπίσουν οι αδυναμίες του παραδοσιακού μηχανισμού. Προσφέρει, δηλαδή ένα κανάλι επικοινωνίας από την εφαρμογή προς μία συσκευή, το οποίο απαιτεί ελάχιστους πόρους, είναι άμεσο και κατά συνέπεια έχει ελάχιστη κατανάλωση ενέργειας.

6.3 Μελλοντικές επεκτάσεις

Ως το βασικό πλεονέκτημα του συστήματος μπορεί να θεωρηθεί η δυνατότητα βελτίωσης του. Αυτό επιτυγχάνεται με την προσθήκη, νέων χαρακτηριστικών στην εφαρμογή του συστήματος καθώς και με τη δημιουργία νέων συσκευών υλικού για τον έλεγχο και την αυτοματοποίηση μεγαλύτερου εύρους λειτουργιών. Παρακάτω παρατίθενται ενδεικτικά μερικές από τις επεκτάσεις που μπορούν να απογειώσουν το σύστημα.

- **Διατήρηση των προς εκτέλεση ενεργειών σε τοπικές δομές της συσκευής.** Αυτό θα έχει ως απόρροια την γνώση των ενεργειών που πρέπει να εκτελέσει μια συσκευή κατά την εκκίνηση της. Εκκίνηση, η οποία μπορεί να προέλθει είτε από επανεκκίνηση λόγω λήξης του watchdog χρονομέτρου είτε από ενεργοποίηση ύστερα από διακοπή ρεύματος. Ακόμα, θα έχει γνώση των ενεργειών, στην περίπτωση που δεν μπορεί να απόκτηση για οποιονδήποτε λόγο επικοινωνία με την εφαρμογή.
- **Πλήρης MQTT επικοινωνία μεταξύ συσκευών και εφαρμογής για απο-**

στολή δεδομένων από την συσκευή προς την εφαρμογή χωρίς την χρήση HTTP. Η τωρινή υλοποίηση του πρωτοκόλλου επικοινωνίας κάνει χρήση του MQTT μόνο για την περίπτωση αποστολής δεδομένων από την εφαρμογή προς τις συσκευές. Οι συσκευές από την μεριά τους στέλνουν τα δεδομένα τους μέσω HTTP. Η πλήρης χρήση του MQTT και στην επικοινωνία από την συσκευή προς την εφαρμογή, θα βοηθούσε στην περαιτέρω μείωση των απαραίτητων πόρων και κατανάλωσης ενεργείας για την λειτουργία μιας συσκευής.

- **Χρήση του πρωτοκόλλου SSL για την επικοινωνία με τις συσκευές.** Η συσκευή που υλοποιήθηκε για την παρούσα διπλωματική εργασία, έγινε με την χρήση του Arduino Uno. Στην προαναφερθείσα πλατφόρμα, λόγω των περιορισμένων πόρων που διαθέτει δεν ήταν δυνατή η επικοινωνία μέσω κρυπτογραφημένου καναλιού. Έτσι μια σημαντική αναβάθμιση στην ασφάλεια του συστήματος θα ήταν η χρήση κάποιας υλικής πλατφόρμας με περισσότερες δυνατότητες ώστε να μπορεί να ενσωματωθεί το πρωτόκολλο SSL.
- **Ασύρματη δικτύωση.** Μια μελλοντική επέκταση του συστήματος και ειδικότερα νέων συσκευών θα ήταν η χρήση της ασύρματης δικτύωσης. Η ασύρματη δικτύωση μπορεί να χρησιμοποιηθεί σε συσκευές οι οποίες δεν αποτελούν κρίσιμο παράγοντα του συστήματος. Ο όρος κρίσιμος, αναφέρεται σε συσκευές οι οποίες δεν κατέχουν ρόλους ασφαλείας στο σύστημα. Παράδειγμα κρίσιμων συσκευών, είναι ένας αισθητήρας συναγερμού ή μια καταγραφική κάμερα ασφαλείας. Σε μη κρίσιμες συσκευές θα ωφελούσε στην ευελιξία της τοποθέτησης σε σημεία στα οποία δεν υπάρχει ήδη ενσύρματη δικτύωση ή δεν είναι εφικτή με κάποιο τρόπο.

Βιβλιογραφία

- [1] *Arduino Client for MQTT*. Oct. 2017. URL: <https://pubsubclient.knolleary.net/>.
- [2] *Arduino Ethernet Shield V1*. Sept. 2017. URL: <https://www.arduino.cc/en/Main/ArduinoEthernetShieldV1>.
- [3] *Arduino Official Page*. Aug. 2017. URL: <http://www.arduino.org/learning/getting-started/what-is-arduino>.
- [4] *Arduino Reference*. Oct. 2017. URL: <https://www.arduino.cc/en/Reference/HomePage>.
- [5] *Arduino UNO*. Sept. 2017. URL: <https://store.arduino.cc/arduino-uno-rev3>.
- [6] *Arduino.h library*. Oct. 2017. URL: <https://github.com/arduino/Arduino/blob/master/hardware/arduino/avr/cores/arduino/Arduino.h>.
- [7] *ArduinoJson, A JSON library for IoT*. Oct. 2017. URL: <https://bblanchon.github.io/ArduinoJson/>.
- [8] *avr/pgmspace.h: Program Space Utilities*. Oct. 2017. URL: http://www.nongnu.org/avr-libc/user-manual/group__avr__pgmspace.html.
- [9] *avr/wdt.h: Watchdog timer handling*. Oct. 2017. URL: http://www.nongnu.org/avr-libc/user-manual/group__avr__watchdog.html.
- [10] *Cascading Style Sheets*. Aug. 2017. URL: <https://www.w3.org/standards/webdesign/htmlcss>.
- [11] *Class for DHTxx sensors*. Oct. 2017. URL: <http://playground.arduino.cc/Main/DHTLib>.
- [12] *Database Management System*. Aug. 2017. URL: <https://en.wikipedia.org/wiki/Database>.

-
- [13] *Digital relative humidity & temperature sensor DHT22*. Sept. 2017. URL: <https://cdn.sparkfun.com/datasheets/Sensors/Weather/RHT03.pdf>.
 - [14] *ELPM - Extended Load Program Memory*. Oct. 2017. URL: http://www.atmel.com/webdoc/avrassembler/avrassembler.wb_ELPM.html.
 - [15] *Ethernet/Ethernet 2 library*. Oct. 2017. URL: <https://www.arduino.cc/en/Reference/Ethernet>.
 - [16] *Fritzing*. Sept. 2017. URL: <http://fritzing.org>.
 - [17] *GIMP*. Sept. 2017. URL: <https://www.gimp.org/>.
 - [18] Michael Hartl. *Rails Tutorial*. 4th ed. Oct. 2017. URL: <https://www.railstutorial.org/>.
 - [19] *HyperText Markup Language*. Aug. 2017. URL: https://www.w3schools.com/html/html_intro.asp.
 - [20] *JavaScript*. Aug. 2017. URL: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript.
 - [21] *LPM - Load Program Memory*. Oct. 2017. URL: http://www.atmel.com/webdoc/avrassembler/avrassembler.wb_LPM.html.
 - [22] *Markup Language*. Aug. 2017. URL: https://en.wikipedia.org/wiki/Markup_language.
 - [23] *Micro-Controller*. Aug. 2017. URL: <https://en.wikipedia.org/wiki/Microcontroller>.
 - [24] *Model-View-Controller*. Aug. 2017. URL: <https://en.wikipedia.org/wiki/Model-view-controller>.
 - [25] *MQTT*. Oct. 2017. URL: <http://mqtt.org/>.
 - [26] *MySQL*. Aug. 2017. URL: <https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html>.
 - [27] *Nest Smart Thermostat*. Oct. 2017. URL: <https://nest.com/thermostats/nest-learning-thermostat/overview/>.
 - [28] *Open-Source*. Aug. 2017. URL: <https://opensource.com/resources/what-open-source>.

-
- [29] *PaulStoffregen/Time: Time library for Arduino*. Oct. 2017. URL: <https://github.com/PaulStoffregen/Time>.
- [30] *Publish/Subscribe messaging pattern*. Oct. 2017. URL: https://en.wikipedia.org/wiki/Publish-subscribe_pattern.
- [31] *Ruby Official WebPage*. Aug. 2017. URL: <https://www.ruby-lang.org/en/>.
- [32] *Ruby on Rails*. Sept. 2017. URL: <http://rubyonrails.org/doctrine>.
- [33] *RubyGems*. Aug. 2017. URL: <https://rubygems.org/pages/download>.
- [34] *SNx4HC595 8-Bit Shift Registers With 3-State Output Registers*. Sept. 2017. URL: <http://www.ti.com/lit/ds/symlink/sn74hc595.pdf>.
- [35] *SPI library*. Oct. 2017. URL: <https://www.arduino.cc/en/Reference/SPI>.
- [36] *W5100 Datasheet*. Sept. 2017. URL: https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf.
- [37] *Web Development*. Aug. 2017. URL: https://en.wikipedia.org/wiki/Web_development.
- [38] *Web Server*. Aug. 2017. URL: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server.
- [39] *Δίοδος εκπομπής φωτός*. Sept. 2017. URL: https://en.wikipedia.org/wiki/Light-emitting_diode.
- [40] Χρήστος Κατσάνος. “Σχεδιασμός και υλοποίηση απομακρυσμένου συστήματος ελέγχου ηλεκτρικής ισχύος”. Feb. 2015.

Παράρτημα Α΄

Οδηγίες ρύθμισης του συστήματος

Για την επιτυχή εκτέλεση του λογισμικού της εφαρμογής προαπαιτούνται ορισμένα βήματα. Χρήσιμη κρίνεται, επίσης, και η παράθεση των οδηγιών προγραμματισμού της συσκευής Arduino. Στη συνέχεια του παραρτήματος βρίσκονται οι προαναφερθείσες πληροφορίες.

Α΄.1 Εγκατάσταση ιστοσελίδας

Αρχικά καταγράφονται οι οδηγίες και τα βήματα που απαιτούνται για να τεθεί σε λειτουργία η εφαρμογή του συστήματος. Δηλαδή, ο οδηγός αυτός θα αναφερθεί στα προαπαιτούμενα λογισμικά που η εφαρμογή χρειάζεται να εγκατασταθούν πριν να είναι έτοιμη για να λειτουργήσει.

Α΄.1.1 Προαπαιτούμενα

1. Ruby v2.3.1 ή μεγαλύτερη
2. RubyGems διαχειριστής πακέτων
3. Rails v5.1 ή μεγαλύτερη
4. NodeJS v8.0 ή μεγαλύτερη
5. MySQL βάση δεδομένων
6. Mosquitto Broker (MQTT)
7. Bundler

-
8. Yarn διαχειριστής πακέτων για τις εξαρτήσεις του frontend (<https://yarnpkg.com/en/docs/install>)
 9. Apache Web server με το πρόσθετο phusion passenger εγκατεστημένο και ενεργοποιημένο. (<https://www.phusionpassenger.com/library/install/standalone/install/oss/>)

A.1.2 Αρχικές ρυθμίσεις

1. Αρχικά, πρέπει να εγκατασταθούν στο σύστημα τα προαπαιτούμενα πακέτα
2. Στη συνέχεια πρέπει να αντιγραφούν τα αρχεία της εφαρμογής στον φάκελο που ο apache χρησιμοποιεί για τον διαμοιρασμό ιστοσελίδων
3. Έπειτα πρέπει να αντιγραφεί το αρχείο 'config/secrets.yml.example' στο 'config/secrets.yml' και να συμπληρωθούν κατάλληλα τα στοιχεία που βρίσκονται μέσα σε αυτό
4. Τρέξτε την εντολή 'bundle install --path vendor/bundle' για την εγκατάσταση των πακέτων της rails
5. Τρέξτε την εντολή 'yarn install' για την εγκατάσταση των εξαρτήσεων της Javascript που είναι απαραίτητες για την λειτουργία της ιστοσελίδας

A.1.3 Ρύθμισή της βάσης δεδομένων

1. Αρχικά πρέπει να δημιουργηθεί μία νέα κενή βάση δεδομένων στη MySQL

```
CREATE DATABASE home_auto_prod;
```

Απεικόνιση Α.1: Παράδειγμα δημιουργίας νέας βάσης μέσω εντολής MySQL

2. Στην συνέχεια πρέπει να δημιουργηθεί ένας χρήστης και να του γίνει παραχώρηση δικαιωμάτων στην βάση που δημιουργήθηκε σε προηγούμενο βήμα

```
CREATE USER 'home_auto_user'@'localhost' IDENTIFIED BY  
'mypass'; # Change 'mypass' accordingly  
GRANT ALL PRIVILEGES ON home_auto_prod.* TO  
'home_auto_user'@'localhost';
```

FLUSH PRIVILEGES;

Απεικόνιση Α'.2: Δημιουργία χρήστη και παράθεση δικαιωμάτων σε βάση

3. Ενημερώσει του αρχείου 'secrets.yml' με τα στοιχεία της βάσης
4. Εκτέλεση της εντολής 'rails db:migrate' για το χτίσιμο των πινάκων και των συσχετίσεων της βάσης

Α'.1.4 Ρύθμιση του Mailer

Για την επιτυχή λειτουργία του συστήματος είναι απαραίτητη η ρύθμιση του email υποσυστήματος της Rails. Αυτό επιτυγχάνεται, δίνοντας πρόσβαση στην εφαρμογή σε έναν λογαριασμό email, συμπληρώνοντας τα στοιχεία του στην ενότητα 'mail' στο αρχείο 'secrets.yml'

1. Απαραίτητη προϋπόθεση για την λειτουργία του υποσυστήματος email είναι η ρύθμιση του 'domain' και της διεύθυνσης αποστολέα 'from'. Σε περίπτωση μη συμπλήρωσης των στοιχείων αυτών το υποσύστημα των email αποτυγχάνει να λειτουργήσει
2. Η σωστή ρύθμιση του Mailer μπορεί να επιβεβαιωθεί τρέχοντας στη κονσόλα της rails (rails console σε ένα τερματικό) την εντολή που εμφανίζεται στην απεικόνιση Α'.3

```
ActionMailer::Base.mail(  
  from: "test@example.com",  
  to: "valid.recipient@domain.com",  
  subject: "Test", body: "Test"  
) .deliver
```

Απεικόνιση Α'.3: Δοκιμή αποστολής email

Α'.1.5 Ρύθμισή του Cron

Στην ενότητα 5.2.2 παρουσιάστηκε η ανάγκη χρήσης του συστήματος Cron για την κάλυψη ορισμένων απαιτήσεων. Παρακάτω επεξηγούνται οι τροποποιήσεις που πρέπει να γίνουν στο αρχείο ρυθμίσεων του cron.

```
*/5 * * * * cd /var/www/html/home-auto && /home/vagrant/.rvm/
wrappers/ruby-2.4.1/rake RAILS_ENV=<environment> devices:
send_offline_notifications --silent
```

Απεικόνιση Α'4: Ρύθμιση του Cron για την εκτέλεση της εργασίας η οποία αναλαμβάνει την αποστολή ειδοποιήσεων στους χρήστες των μη συνδεδεμένων συσκευών

```
*/5 * * * * cd /var/www/html/home-auto && /home/vagrant/.rvm/
wrappers/ruby-2.4.1/rake RAILS_ENV=<environment>
smart_thermostat:training_set_sample --silent
```

Απεικόνιση Α'5: Ρύθμιση του Cron για την εκτέλεση της εργασίας με την οποία λαμβάνονται δείγματα για τις απαραίτητες τιμές καθενός 'έξυπνου θερμοστάτη'

```
*/30 * * * * cd /var/www/html/home-auto && /home/vagrant/.rvm/
wrappers/ruby-2.4.1/rake RAILS_ENV=<environment>
smart_thermostat:recalculate_smart_thermostat_start_datetime
--silent
```

Απεικόνιση Α'6: Ρύθμιση του Cron για την εκτέλεση της εργασίας η οποία επαναυπολογίζει τον χρόνο έναρξης προγράμματος ενός 'έξυπνου θερμοστάτη'

```
0 */6 * * * cd /var/www/html/home-auto && /home/vagrant/.rvm/
wrappers/ruby-2.4.1/rake RAILS_ENV=<environment>
smart_thermostat:analyze_training_set --silent
```

Απεικόνιση Α'7: Ρύθμιση του Cron για την εκτέλεση της εργασίας η οποία αναλύει το σύνολο δειγμάτων ενός 'έξυπνου θερμοστάτη'

Α'1.6 Δοκιμή της εφαρμογής σε development περιβάλλον

Η εφαρμογή πλέον, έχει προετοιμαστεί και εγκατασταθεί πλήρως και είναι έτοιμη για την δοκιμαστική της λειτουργία σε περιβάλλον ανάπτυξης (development). Αυτό επιτυγχάνεται με την χρήση της εντολής 'rails server' στο τερματικό του εξυπηρετητή της εφαρμογής. Στην περίπτωση που όλοι οι παράμετροι έχουν τεθεί σωστά, θα εμφανιστεί κατάλληλο μήνυμα που θα ενημερώνει τον χρήστη για την IP διεύθυνση και την πόρτα του εξυπηρετητή στα οποία η εφαρμογή εκτελείται. Σε αντίθετη

περίπτωση θα εμφανιστούν μηνύματα λάθους που θα καθοδηγούν τον χρήστη στην επίλυση του προβλήματος.

A.1.7 Μεταφορά της εφαρμογής σε production περιβάλλον μέσω Apache

Για την λειτουργία της εφαρμογής rails σε production περιβάλλον δια μέσω του apache είναι αναγκαία η τροποποίηση μερικών παραμέτρων ακόμα.

1. Στο αρχείο ρυθμίσεων του apache θα πρέπει ως document root να οριστεί ο υποφάκελος 'public' της εφαρμογής rails
2. Πρέπει να ρυθμιστεί η οδηγία (directve) 'RailsEnv production' οι οποία υποδεικνύει στον apache ότι ζητείται η εκτέλεση της εφαρμογής σε production περιβάλλον.
3. Προσοχή πρέπει να δοθεί στην ενημέρωση της ενότητας 'production' στο αρχείο 'secrets.yml'
4. Τρέξτε την εντολή `RAILS_ENV=production rails db:migrate` για το χτίσιμο της production βάσης δεδομένων
5. Τρέξτε την εντολή `RAILS_ENV=production bundle exec rake assets:precompile` ώστε να προετοιμαστούν τα assets της εφαρμογής για εκτέλεση σε περιβάλλον production
6. Για να αναγνωρίσει ο apache τα νέα αρχεία κατά την λειτουργία σε production περιβάλλον απαιτείται η επανεκκίνηση του έπειτα από οποιαδήποτε αλλαγή στα αρχεία της εφαρμογής

A.1.8 Βοηθητικές οδηγίες για εγκατάσταση σε FreeBSD

Στην ενότητα αυτή γίνεται η παράθεση των πακέτων του FreeBSD τα οποία πρέπει να εγκατασταθούν και των εντολών που πρέπει να εκτελεστούν για την εκπλήρωση των προαπαιτούμενων της εφαρμογής.

1. `sudo pkg install apache24`
2. `sudo sysrc apache24_enable=yes`

-
3. `sudo service apache2 start`
 4. `sudo pkg install mysql57-server`
 5. `sudo sysrc mysql_enable=yes`
 6. `sudo service mysql-server start`
 7. `sudo mysql_secure_installation` (give the start command a lot time before running this one)
 8. `sudo pkg install ruby devel/ruby-gems`
 9. `sudo pkg install node mosquitto yarn gmake`
 10. `sudo sysrc mosquitto_enable=yes`
 11. `sudo service mosquitto start`
 12. `sudo gem install rails passenger --no-ri --no-doc`
 13. `sudo passenger-install-apache2-module` (select 'ruby' and 'node.js')

Η σωστή εκτέλεση των παραπάνω εντολών εκπληρώνει τα προαπαιτούμενα της εφαρμογής. Στη συνέχεια θα πρέπει να ακολουθηθούν τα βήματα που αναφέρονται στις αρχικές ρυθμίσεις της εφαρμογής στην ενότητα Α'.1.2.

Α'.2 Ρύθμιση συσκευής Arduino

Η ανάπτυξη του λογισμικού του Arduino έγινε στον κειμενογράφο Atom, χρησιμοποιώντας το πρόσθετο PlatformIO το οποίο τον μετατρέπει σε ένα πλήρες IDE για προγραμματισμό σε περιβάλλον Arduino. Βέβαια, κάποιος μπορεί να χρησιμοποιήσει και το Arduino IDE, χωρίς κανένα πρόβλημα αρκεί όμως να εγκαταστήσει τις προαπαιτούμενες βιβλιοθήκες.

Αφού έχει αποκτήσει κάποιος τον πηγαίο κώδικα του λογισμικού του Arduino, πρέπει αρχικά να αντιγράψει το αρχείο 'src/DeviceSecrets.h.example' στο 'src/DeviceSecrets.h' και να συμπληρώσει κατάλληλα τις πληροφορίες που χρειάζονται. Παρακάτω, παραθέτονται οι οδηγίες για την διαδικασία μεταγλώττισης και ανεβάσματος του εκτελέσιμου αρχείου στην συσκευή Arduino με την χρήση του Atom και του Arduino IDE.

A'.2.1 Atom/PlatformIO

Για την επεξεργασία των πηγαίων αρχείων με την χρήση του Atom, πρέπει να γίνει η εισαγωγή του φακέλου που περιέχει τους πηγαίους κώδικες του Arduino, στον Atom ως νέο project. Στην συνέχεια πατώντας το κουμπί της μεταγλώττισης ο Atom αναλαμβάνει να μεταφορτώσει όλες τις εξαρτήσεις που απαιτούνται βάσει του αρχείου 'platformio.ini'.

A'.2.2 Arduino IDE

Για την επεξεργασία στο επίσημο περιβάλλον ανάπτυξης του Arduino, απαιτείται η χειροκίνητη αναζήτηση και μεταφόρτωση των απαραίτητων βιβλιοθηκών, είτε μέσω του εργαλείου διαχειρίσεις βιβλιοθηκών (Library manager) είτε μέσω της λήψης απευθείας απο το github για βιβλιοθήκες που δεν βρίσκονται στον library manager. Στη συνέχεια γίνεται κανονικά η διαδικασία μεταγλώττισης και ανεβάσματος.