ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

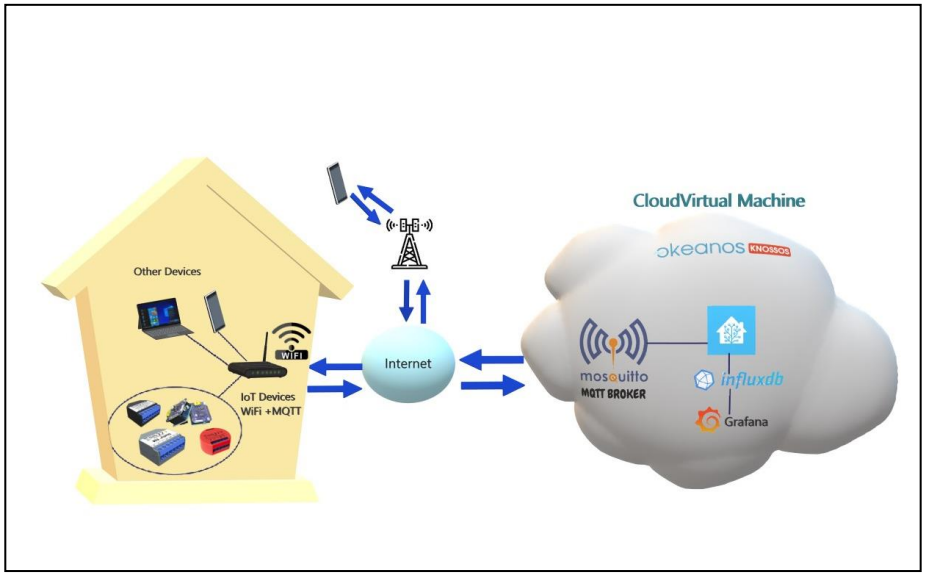
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ & ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ

Πρόγραμμα Μεταπτυχιακών Σπουδών
«ΔΙΑΔΙΚΤΥΟ ΤΩΝ ΠΡΑΓΜΑΤΩΝ ΚΑΙ ΕΥΦΥΗ ΠΕΡΙΒΑΛΛΟΝΤΑ»

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Ευφυές δίκτυο διαχείρισης ενέργειας και μέτρησης ενεργειακής κατανάλωσης οικίας με χρήση τεχνολογιών IoT»



Μεταπτυχιακή Φοιτήτρια: Γεωργία Πηλιχού, ΑΜ msciot20007
Επιβλέπων: Γρηγόριος Κουλούρας, Αναπληρωτής Καθηγητής

ΑΙΓΑΛΕΩ, ΙΟΥΝΙΟΣ 2022

UNIVERSITY OF WEST ATTICA

FACULTY OF ENGINEERING

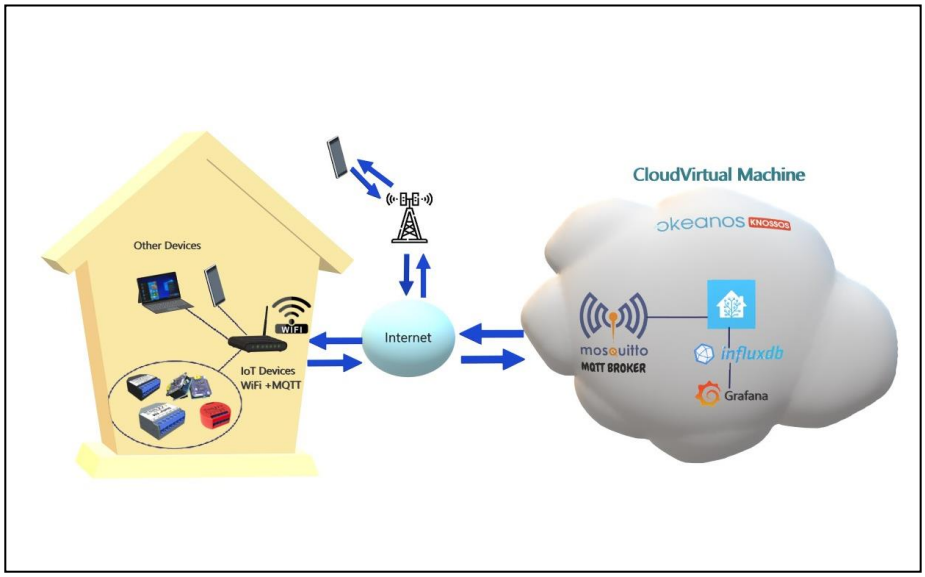
DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

Master of Science in

"INTERNET of THINGS AND INTELLIGENT ENVIRONMENTS"

MSc Thesis

«Smart Energy management and Energy consumption measurement network in a household, using IoT Technologies»



Postgraduate Student: Georgia Pilichou, Registration Number: msciot20007

MSc Thesis Supervisor: Grigorios Koulouras, Associate Professor

ATHENS-EGALEO, JUNE 2022

Η Διπλωματική Εργασία έγινε αποδεκτή και βαθμολογήθηκε από την εξής τριμελή επιτροπή:

Γρηγόριος Κουλούρας, Αναπληρωτής Καθηγητής	Ιωάννης Φαμέλης, Καθηγητής	Αλέξανδρος Αλεξανδρίδης, Καθηγητής
Grigorios Koulouras Digitally signed by Grigorios Koulouras Date: 2022.06.08 17:59:43 +03'00'	Digitally signed by Ioannis Famelis Date: 2022.06.08 21:12:37 +03'00'	Digitally signed by Alexandros Alexandridis Date: 2022.06.08 20:45:15 +03'00'

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Η κάτωθι υπογεγραμμένη **Γεωργία Πηλιχού** του **Ιωάννη**, με αριθμό μητρώου **msciot20007** μεταπτυχιακή φοιτήτρια του Πανεπιστημίου Δυτικής Αττικής της Σχολής ΜΗΧΑΝΙΚΩΝ του Τμήματος ΗΛΕΚΤΡΟΛΟΓΩΝ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ, στο **ΠΜΣ «ΔΙΑΔΙΚΤΥΟ ΤΩΝ ΠΡΑΓΜΑΤΩΝ ΚΑΙ ΕΥΦΥΗ ΠΕΡΙΒΑΛΛΟΝΤΑ»**

δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.
Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του διπλώματός μου.»

Η Δηλούσα

Georgia Pilichou
Digitally signed by Georgia Pilichou
Date: 2022.06.08 17:16:54 +03'00'

Γεωργία Πηλιχού

Ευχαριστίες

Για την εκπόνηση αυτής της διπλωματικής εργασίας, θα ήθελα αρχικά να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Γρηγόριο Κουλούρα. Τον ευχαριστώ θερμά, για τον χρόνο που διέθετε κάθε φορά που χρειαζόμουν την καθοδήγηση του, καθώς και για τις πολύτιμες επιστημονικές γνώσεις που μου προσέφερε καθ' όλη την διάρκεια των σπουδών μου. Επίσης, θα ήθελα να ευχαριστήσω την οικογένεια μου, τους φίλους μου και τον σύντροφο μου, που με τον τρόπο τους με βοήθησαν να μείνω προσηλωμένη στους στόχους μου, με ενθάρρυναν σε στιγμές αδυναμίας και με υποστήριξαν σε όλα τα στάδια της πορείας μου. Τέλος, θα ήθελα να ευχαριστήσω τους συνάδελφους μου στην Ελληνική Αεροπορική Βιομηχανία, που έδειξαν μεγάλο ενδιαφέρον για την συμμετοχή μου στο παρόν πρόγραμμα μεταπτυχιακών σπουδών και συνέβαλαν με την υποστήριξη τους ο κάθε ένας με το δικό του τρόπο.

Περίληψη

Στην παρούσα εργασία σχεδιάστηκε μία εφαρμογή σε οικιακό περιβάλλον, η οποία περιλαμβάνει συσκευές μέτρησης και μετάδοσης μετρήσεων ενέργειας αλλά και των περιβαλλοντικών συνθηκών εντός ενός οικιακού δικτύου με ασύρματη μέθοδο. Μέσω των μετρήσεων αυτών, δόθηκε η δυνατότητα εντοπισμού αυξημένης κατανάλωσης ενέργειας, των αιτιών που την προκαλούν και του συμπεράσματος ότι πρέπει να βελτιωθεί ο τρόπος διαχείρισης των ενεργοβόρων ηλεκτρικών συσκευών. Αρχικά, θα μελετηθούν βασικές έννοιες που αφορούν την ενέργεια , τους τρόπους καταμέτρησης της, το σύστημα ηλεκτροδότησης, διαχείρισης και κοστολόγησης της ενέργειας στην Ελλάδα. Στη συνέχεια, θα εξεταστεί το Διαδίκτυο των πραγμάτων, εστιάζοντας στο πώς συνδυάζει υλικό και λογισμικό, με την απομακρυσμένη διαχείριση συσκευών και αισθητήρων. Αφού δοθεί, σαφής επεξήγηση των εννοιών, θα γίνει πλήρης ανάλυση του υλικού και του λογισμικού που χρησιμοποιήθηκε στην υλοποίηση της εφαρμογής. Η εφαρμογή, περιλαμβάνει ασύρματους μικροελεγκτές WiFi, με δυνατότητες καταμέτρησης ενέργειας, διαχείρισης λειτουργίας των διαφόρων συσκευών, καθώς και αισθητήρες περιβάλλοντος. Για να γίνει, πιο αποδοτική η παρακολούθηση και ο έλεγχος αυτών των μικροελεγκτών, εντάχθηκαν σε μία ενιαία πλατφόρμα η οποία έχει αναπτυχθεί σε υπηρεσία εικονικής μηχανής νέφους (cloud virtual machine). Η πλατφόρμα αυτή, έχει δυνατότητες παρουσίασης, τόσο των δεδομένων ενέργειας/περιβάλλοντος σε πραγματικό χρόνο, όσο και έλεγχο λειτουργίας ή μη των συσκευών που έχουμε ορίσει. Τα δεδομένα αυτά, συλλέχθηκαν σε μία βάση δεδομένων εγκατεστημένη στο εικονικό μηχάνημα νέφους και η απεικόνιση τους έγινε με χρήση λογισμικού απεικόνισης των δεδομένων. Εν κατακλείδι, θα αποδειχθεί πως συνέβαλε η χρήση διάφορων εργαλείων και τεχνολογιών IoT, στο να επιτευχθεί πιο αποδοτική χρήση των οικιακών συσκευών και η έξυπνη εξοικονόμηση ηλεκτρικής ενέργειας.

Λέξεις – κλειδιά

Διαδίκτυο των πραγμάτων, διαχείριση ενέργειας, μέτρηση κατανάλωσης ενέργειας, συλλογή δεδομένων, ασύρματο δίκτυο, αισθητήρες, οπτικοποίηση μετρήσεων αισθητήρων, νέφος, βάση δεδομένων, εικονική μηχανή

Abstract

In the present thesis, an application was designed within a house, which includes measuring devices that also transmit energy consumption and environmental measurements via a wireless network. Through these measurements, it was made possible to track energy overconsumption and the reasons for it, while conclusions were made regarding the need for upgrade and optimization in the managing of devices with high energy consumption. Firstly, basic concepts will be studied related to energy, ways of measuring the energy consumption, the power distribution system and pricing of energy consumption in Greece. Next, the concept of "Internet of Things" will be examined, focusing on how it combines hardware, software with remote controlling of devices or sensors. Once these concepts are detailed, a full analysis of the hardware and software, that was used in this project, will be presented. The project involves microcontrollers using Wi-Fi for wireless communication, with energy consumption measurement capabilities as well as capabilities for remote management of devices and sensors. In order to monitor and manage the microcontrollers in a more effective manner, these were embedded into a single platform that has been developed on a cloud virtual machine. The platform presents energy and environmental data in real time and can control the operation of the defined devices. The data has been collected and stored in a database, which has already been installed at the cloud virtual machine, while visualization was made using a data visualization platform. In the end, the use of IoT technologies combined with different tools will be proved as a way to improve home devices and make them smart and energy efficient.

Keywords

Internet of Things, energy management, energy consumption metering, data collection, wireless network, sensors, sensor measurements visualization, cloud, database, virtual machine.

Περιεχόμενα

Κατάλογος Πινάκων

Κατάλογος Εικόνων

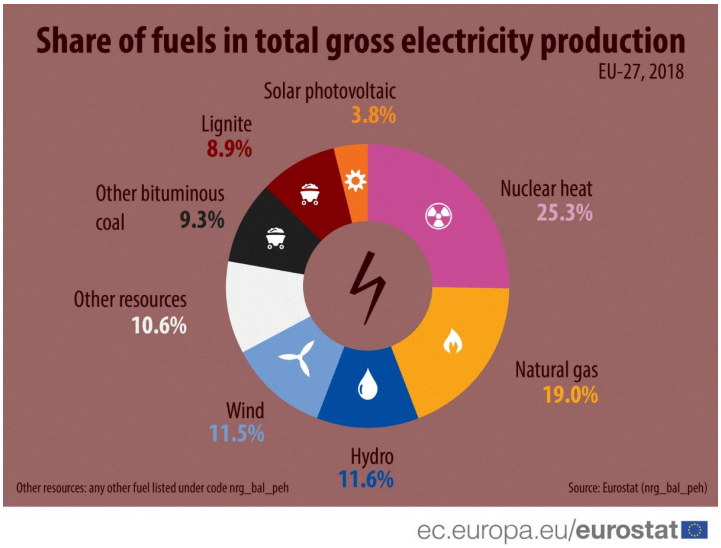
Αλφαβητικό Ευρετήριο

ADC: Analog to Digital Converter
AI: Artificial Intelligence
AES: Advanced encryption standard
BLE: Bluetooth Low Energy
CoAP: Constrained Application Protocol
CRM: Customer relationship management
DAC: Digital to Analog Converter
DB: Database
ECDHE: Elliptic-curve-Diffie-Hellman exchange
HA: Home Assistant
HACS: Home Assistant Community Store
HTTP: Hypertext Transfer Protocol
IaaS: Infrastructure as a Service
IDE: Integrated Development Environment
IEEE: Institute of Electrical and Electronics Engineers
IoT: Internet of Things
IP: Internet Protocol
IR: Infrared
LAN: Local Area Network
LED: Light Emitting Diode
LTE: Long term evolution
M2M: Machine to machine
MQTT: Message Queuing Telemetry Transport
NB-IoT: Narrowband -Internet of Things
OS: Operating System
OSI: Open Systems Interconnection
PaaS: Platform as a Service
PCB: Printed Circuit Board
QoS: Quality of Service
REST: REpresentational State Transfer
SaaS: Software as a Service
SQL: Structured Query Language
SSH: Secure SHell
SSID: Service Set IDentifier
SSL: Secure Sockets Layer
TCP: Transmission Control Protocol
TSDB: Time Series DataBase
UART: Universal Asynchronous Receiver/Transmitter
UDP: User Datagram Protocol
USB: Universal Serial Bus
UTC: Universal Coordinated Time
VM: Virtual Machine
WEP: Wired Equivalent Privacy

WiFi: Wireless Fidelity (Wireless Network Protocol)
WLAN: Wireless Local Area Network
WPA: WiFi Protected Access
Α.Ε.: Ανώνυμη εταιρεία
ΔΕΔΔΗΕ: Διαχειριστής Ελληνικού Δικτύου Διανομής Ηλεκτρικής Ενέργειας
ΔΕΗ: Δημόσια Επιχείρηση Ηλεκτρισμού
ΔτΠ: Διαδίκτυο των Πραγμάτων
Ε.Ε.: Ευρωπαϊκή Ένωση

ΕΙΣΑΓΩΓΗ

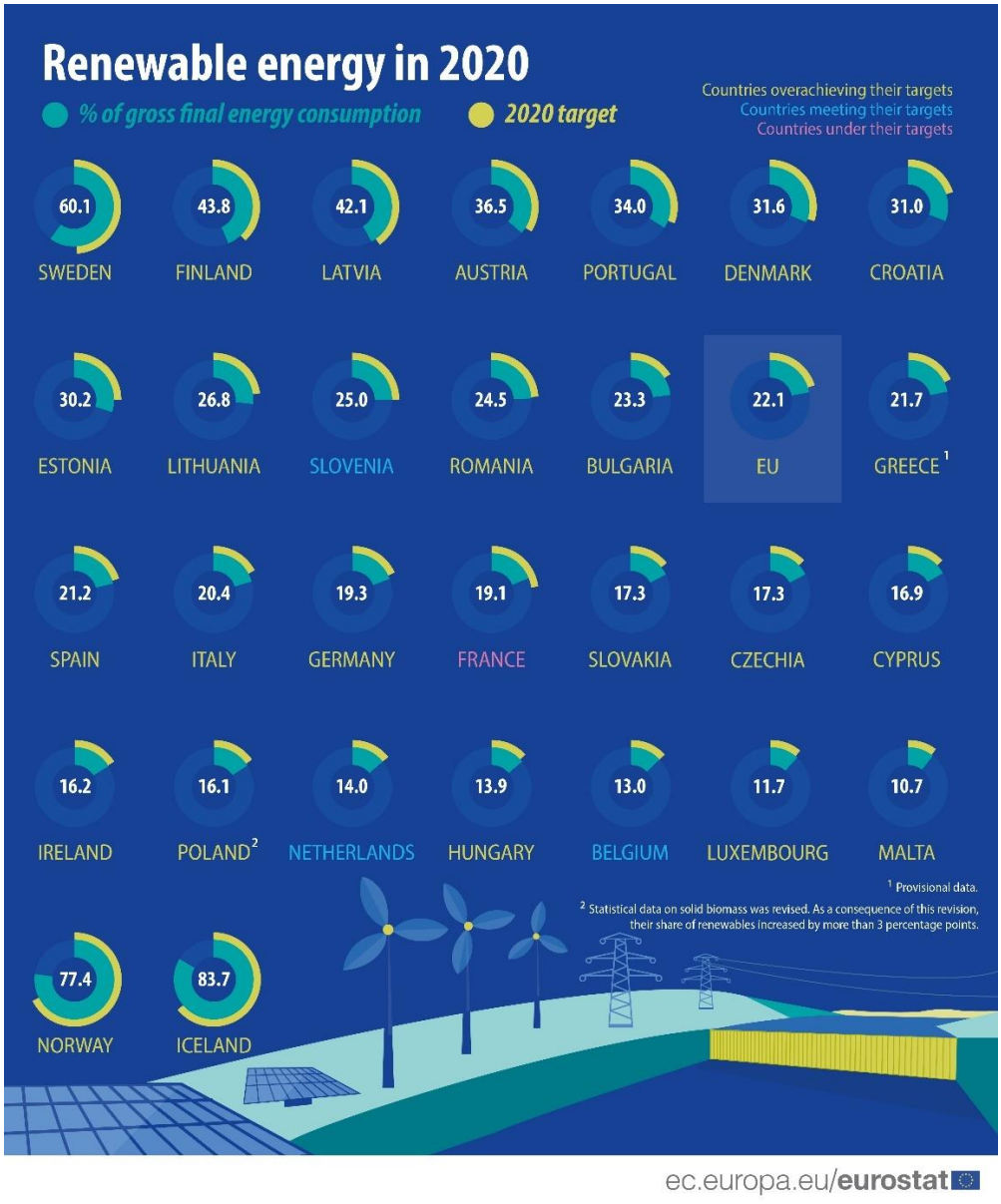
Η ύπαρξη όλων των οντοτήτων δεν θα υφίσταντο αν δεν υπήρχε ενέργεια, καθώς τα πάντα γύρω μας λειτουργούν, κινούνται και αναπνέουν κάνοντας χρήση ενέργειας. Η ενέργεια, όπως είναι ευρέως γνωστό, είναι η παραγωγή έργου και έχει διάφορες μορφές. Οι βασικές της μορφές είναι, η δυναμική και η κινητική ενέργεια. Όταν η ενέργεια είναι αποτέλεσμα κίνησης τότε λέγεται κινητική και μερικά από τα είδη της είναι η ηλεκτρομαγνητική, η θερμική, η αιολική , η ηλεκτρική και η πυρηνική ενέργεια. Οι ζωντανοί οργανισμοί, για παράδειγμα, για να επιβιώσουν χρειάζονται ενέργεια, η τροφή που καταναλώνουν κατά την πέψη κάνει χρήση χημικής ενέργειας που τους προσφέρει τα απαραίτητα συστατικά για την ομαλή λειτουργία του οργανισμού τους. Η μετατροπή της ενέργειας από μία μορφή σε μία άλλη γίνεται είτε εκ φύσεως είτε επιτηδευμένα. Η ηλεκτρική ενέργεια, για παράδειγμα, είναι αποτέλεσμα μετατροπής μίας άλλης πρωτογενούς μορφής ενέργειας. Ορισμένα εργοστάσια παραγωγής ηλεκτρικής ενέργειας χρησιμοποιούν ορυκτά καύσιμα ,όπως το πετρέλαιο, με τα οποία μετατρέπουν την θερμική ενέργεια που παράγεται από την καύση τους σε κινητική και στην συνέχεια με χρήση γεννητριών σε ηλεκτρική. Μετά την κρίση του πετρελαίου του 1970, άρχισαν να χρησιμοποιούνται και άλλου είδους καύσιμα όπως ο άνθρακας, το φυσικό αέριο και η πυρηνική ενέργεια [1]. Οι επιπτώσεις μίας τέτοιας κρίσης είναι κυρίως οικονομικές, έχει μεγάλες συνέπειες στην καθημερινότητα των πολιτών και το βιοτικό τους επίπεδο, αλλά μας υπενθυμίζει πόσο πολύτιμοι είναι οι φυσικοί πόροι του πλανήτη. Το περιβαλλοντικό κόστος της χρήσης ορυκτών πόρων, στην παραγωγή ηλεκτρικής ενέργειας, ενδεχομένως είναι εξαιρετικά σημαντικό. Η καύση υδρογονανθράκων, αυξάνει τις ποσότητες διοξειδίου του άνθρακα και άλλων ρυπογόνων, η ακατάλληλη διαχείριση των αποβλήτων μπορεί να οδηγήσει σε μολύνσεις του υδροφόρου ορίζοντα. Φαινόμενα όπως η υπερθέρμανση του πλανήτη, η κλιματική αλλαγή και η εξάντληση των φυσικών πόρων θα πρέπει να αντιμετωπισθούν [2]. Παρόλα αυτά, η ολοένα και αυξανόμενη παραγωγή και ζήτηση ηλεκτρικών συσκευών οδηγεί σε μεγαλύτερες απαιτήσεις ενέργειας και σε βιομηχανικό αλλά και σε οικιακό περιβάλλον. Στην Ευρωπαϊκή ένωση, το 9% της ηλεκτρικής ενέργειας που παράγεται βασίζεται στον λιγνίτη με την Ελλάδα να βασίζει το 32% της παραγωγής της σε αυτόν.



Εικόνα 0.1 Καύσιμα που χρησιμοποιούνται για την παραγωγή ηλεκτρικής ενέργειας σε χώρες της Ευρωπαϊκής Ένωσης το έτος 2018. [3]

Τα τελευταία χρόνια, γίνονται προσπάθειες μείωσης της χρήσης ορυκτών πόρων ή ρυπογόνων μέσων για την παραγωγή ενέργειας και προωθούνται οι ανανεώσιμες πηγές ενέργειας ως μέσο παραγωγής ηλεκτρικής ενέργειας. Από την εκμετάλλευση των ανανεώσιμών πηγών ενέργειας, μπορεί να παραχθεί ηλεκτρική ενέργεια

αλλά και θερμότητα. Συνεπώς, έχουμε διπλό όφελος καθώς χρησιμοποιούμε δωρεάν και θεωρητικά απεριόριστη ενέργεια ενώ ταυτόχρονα έχουμε σχεδόν μηδενικό περιβαλλοντικό κόστος. Ο όρος «θεωρητικά», αναφέρεται στο άμεσα διαθέσιμο απόθεμα για κατανάλωση αυτών των πηγών. Για παράδειγμα, υπάρχει πιθανότητα να μην υπάρχουν αρκετές ποσότητες κομποστοποιημένης ύλης για καύση ή περιορισμένη ηλιοφάνεια ή η ταχύτητα του ανέμου να μην είναι η κατάλληλη. Στην Ευρωπαϊκή Ένωση, για το έτος 2020, παρατηρήθηκε αύξηση 2% στην ενέργεια που καταναλώθηκε από ανανεώσιμες πηγές ενέργειας, διπλάσιο ποσοστό σε σχέση με το 2004. Η πανδημία από τον COVID-19, συνέβαλε στην μειωμένη κατανάλωση ορυκτών καυσίμων, καθώς μειώθηκαν οι μετακινήσεις και η χρήση μηχανοκίνητων μέσων. Παράλληλα από την Ε.Ε., υπήρχαν σε ισχύ κανονισμοί προώθησης αύξησης του μεριδίου κάθε κράτους μέλους όσων αφορά την χρήση ανανεώσιμων πηγών ενέργειας. Στην Εικόνα 0.2, παρουσιάζονται αναλυτικά το κάθε κράτος μέλος και πως συνέβαλε το κάθε ένα σε αυτό. Είναι σημαντικό, να αναφερθεί ότι ο βασικός στόχος της ευρωπαϊκή κοινότητας είναι, η Ευρώπη να γίνει η πρώτη κλιματικά ουδέτερη ήπειρος μέχρι το 2050. [4].



Εικόνα 0.2 Ενέργεια που καταναλώθηκε από ανανεώσιμες πηγές ενέργειας στην ΕΕ[4]**.**

Η κατανάλωση ηλεκτρικής ενέργειας, δεν αφορά μόνο της εταιρίες παραγωγής ηλεκτρικής ενέργειας αλλά και τις βιομηχανίες και τα οικιακά περιβάλλοντα. Όλοι οφείλουν, να συμβάλουν στην μείωση της κατανάλωσης των φυσικών πόρων του πλανήτη και μπορεί να επιτευχθεί, ξεκινώντας από το οικείο μας περιβάλλον. Η δυσκολία στην πρόσβαση, σε πληροφορίες λειτουργίας ή κατανάλωσης ηλεκτρικής ενέργειας, από τις επιμέρους ηλεκτρικές συσκευές μίας οικίας οδηγούν πολλές φορές σε άσκοπη χρήση τους. Η μόνη πληροφορία που δίνεται στους καταναλωτές, προέρχεται είτε από τον λογαριασμό του παρόχου ηλεκτρικής ενέργειας, είτε από τον μετρητή ηλεκτρικής ενέργειας που έχει εγκαταστήσει ο πάροχος στην εξωτερική κεντρική παροχή. Μία τόσο γενικευμένη και μη ευκόλως προσβάσιμη πληροφορία, δεν δίνει σαφή εικόνα στον καταναλωτή για το πώς να διαχειριστεί την ηλεκτρική ενέργεια που καταναλώνεται, από τις επιμέρους οικιακές ηλεκτρικές συσκευές. Είναι γεγονός, πως για μεγαλύτερη εξοικονόμηση ενέργειας, είναι σημαντικό να γίνουν πρώτα δομικές βελτιώσεις στο κτίρια των οικιών έτσι ώστε να μην υπάρχουν απώλειες ενέργειας. Ωστόσο, το κόστος ενίσχυσης της θερμομόνωσης μίας οικίας, είναι μεγάλο και τα νοικοκυριά δεν έχουν συνήθως την δυνατότητα για τέτοιου είδους επενδύσεις. Η Ελληνική κυβέρνηση, με το πρόγραμμα «Εξοικονομώ» στοχεύει στην ενεργειακή αναβάθμιση έως και 50.000 οικιακών κτιρίων για το 2021 επιδοτώντας του πολίτες που θα συμμετάσχουν σε αυτό. Η επιλογή των νοικοκυριών, γίνεται με οικονομικά και κοινωνικά κριτήρια, καθώς το κόστος είναι αρκετά μεγάλο και προτεραιότητα έχουν τα πιο ευάλωτα νοικοκυριά[5]. Αυτό σημαίνει ότι ενισχύει οικονομικά κάποια νοικοκυριά αλλά όχι όλα, επομένως η εύρεση εναλλακτικής λύσης είναι μονόδρομος. Η αντικατάσταση των ηλεκτρικών συσκευών με νέες, μεγαλύτερου δείκτη ενεργειακής απόδοσης, βελτιώνει επίσης την συνολική ενεργειακή κατανάλωση σε ένα οικιακό περιβάλλον. Αυτές οι μέθοδοι, παρότι αναβαθμίζουν σημαντικά την ενεργειακή απόδοση μίας οικίας για την υλοποίηση τους, απαιτούν ένα υψηλό οικονομικό κόστος και πιθανότατα να αποδώσουν κάποια μη ανακυκλώσιμα απορρίμματα. [6]

Αντικείμενο της διπλωματικής εργασίας

Η διπλωματική εργασία αυτή, έχει ως κεντρικό αντικείμενο τον σχεδιασμό και την υλοποίηση ενός δικτύου συσκευών και αισθητήρων σε οικιακό περιβάλλον. Το παραπάνω δίκτυο θα συμβάλει στον κεντρικό έλεγχο διαφόρων συσκευών εντός της οικίας καθώς και την αυτοματοποίηση κάποιων λειτουργιών τους, έτσι ώστε να μειωθεί η άσκοπη χρήση τους και συνεπώς η συνολική κατανάλωση ενέργειας τους. Η απομακρυσμένη παρακολούθηση και ο έλεγχος των συσκευών αυτού του δικτύου είναι απαραίτητη για την βέλτιστη αξιοποίηση των παραμέτρων κατανάλωσης και αυτό θα το επιτευχθεί κάνοντας χρήση τεχνολογιών IoT (Internet of Things).

Σκοπός και στόχοι

Στο δίκτυο που υλοποιήθηκε, επιδιώχθηκε η καταγραφή και μελέτη των μετρήσεων που ελήφθησαν, καθώς και η επιλογή κατάλληλων μέτρων μείωσης της κατανάλωσης ενέργειας, ανά συσκευή. Η υλοποίηση της εφαρμογής, περιλαμβάνει κατασκευαστικά δύο προκλήσεις, τον έλεγχο των ενεργοβόρων συσκευών, καθώς και την αποστολή δεδομένων λειτουργίας και περιβάλλοντος από διάφορα αισθητήρια. Για την καταμέτρηση της ηλεκτρικής ενέργειας και τον έλεγχο λειτουργίας των περισσότερων συσκευών, έγινε χρήση συσκευών IoT, με δυνατότητα καταμέτρησης της κατανάλωσης ενέργειας, οι οποίες ενσωματώνονται είτε πάνω στην τροφοδοσία των συσκευών είτε πίσω από το κέλυφος των επιτοίχιων ρευματοδοτών που έχει ήδη η οικία. Για τον έλεγχο των περιβαλλοντικών συνθηκών ανά δωμάτιο, αλλά και τον έλεγχο λειτουργίας των κλιματιστικών, χρειάστηκε να υλοποιηθεί εξαρχής μία ασύρματη συσκευή με μικροελεγκτή που έχει δυνατότητες απομακρυσμένης διαχείρισης, ενώ ταυτόχρονα ενσωματώνει τα απαραίτητα αισθητήρια περιβάλλοντος και

τους ενεργοποιητές ελέγχου των κλιματιστικών. Στόχος της παρούσας διπλωματικής, είναι να επιτευχθεί εξοικονόμηση ηλεκτρικής ενέργειας μέσω του ελέγχου λειτουργίας των ηλεκτρικών συσκευών, με χρήση ασύρματης διασύνδεσης και κεντρικοποίησης του ελέγχου όλων των επιμέρους συσκευών και αισθητήρων.

Μεθοδολογία

Όσον αφορά την μεθοδολογία που ακολουθήθηκε στην παρούσα διπλωματική, αρχικά έπρεπε να οριστούν κάποιες βασικές έννοιες κάνοντας την κατάλληλη βιβλιογραφική έρευνα. Αυτές αφορούν την ηλεκτρική ενέργεια, τις τεχνολογίες, τα πρωτόκολλα και τις εφαρμογές που χρησιμοποιήθηκαν κατά την υλοποίηση της εφαρμογής. Στη συνέχεια, περιγράφετε η μορφή που έχει το δίκτυο αισθητήρων που υλοποιήθηκε ξεκινώντας από το ηλεκτρολογικό σχέδιο της οικίας στην οποία έγινε η υλοποίηση. Αφού καταγράφηκε το σύνολο των ηλεκτρικών συσκευών της οικίας, εντοπίστηκαν οι ανάγκες σε έλεγχο και παρακολούθηση της κατανάλωσης ανά συσκευή. Στη συνέχεια, έγινε έρευνα αγοράς που αφορούσε την διαθεσιμότητα συσκευών καταμέτρησης και παρακολούθησης της ηλεκτρικής ενέργειας. Από την οποία προέκυψε ό,τι υπάρχει μεγάλη διαθεσιμότητα και ποικιλία στον τρόπο επικοινωνίας αλλά και παραμετροποίηση της κάθε μίας συσκευής. Η πιο συμφέρουσα επιλογή ήταν η αγορά ενός έτοιμου συστήματος που να μπορεί να εγκατασταθεί εύκολα σε κάθε οικιακό περιβάλλον, να επικοινωνεί με WiFi, να διαθέτει δυνατότητες επικοινωνίας με γνωστά πρωτόκολλα για εφαρμογές IoT (Internet of Things), όπως επίσης να ενσωματώνει δυνατότητες καταμέτρησης και παρακολούθησης της ηλεκτρικής ενέργειας.

Στην οικία που υλοποιήθηκε το κατασκευαστικό μέρος της διπλωματικής, γίνεται χρήση πολλών ηλεκτρικών συσκευών, θέρμανσης και ψύξης και χρειάστηκε για λόγους αυτοματοποίησης να καταγράφονται και οι περιβαλλοντικές συνθήκες ανά δωμάτιο. Έγινε ξανά, έρευνα αγοράς και διαπιστώθηκε ότι οι έτοιμες λύσεις δεν κάλυπταν κάποια από τα προαπαιτούμενα χαρακτηριστικά όπως η χρήση γνωστών πρωτοκόλλων IoT επικοινωνίας, είτε δεν διέθεταν τους απαραίτητους αισθητήρες ή επεκτασιμότητα με περισσότερους αισθητήρες. Η λύση ήταν η υλοποίηση μίας πρωτότυπης υλοποίησης που βασίστηκε σε μικροελεγκτή ESP8266, και θα ενσωματώνει όλες τις προαναφερθείσες δυνατότητες ,όπως επίσης και μεγάλη επεκτασιμότητα.

Στη συνέχεια, έγινε η εγκατάσταση μίας πλατφόρμας κεντρικού ελέγχου για έξυπνα σπίτια, σε μία εικονική μηχανή νέφους,. Αυτή έχει δυνατότητες ενσωμάτωσης, όλων αυτών των συσκευών και διαμορφώθηκε με τέτοιο τρόπο ώστε να είναι φιλική προς τον τελικό χρήστη. Σε αυτή εμφανίζονται αναλυτικά οι συσκευές και οι πληροφορίες που προσφέρονται από το τρέχον λογισμικό τους. Επίσης εμφανίζονται διακόπτες και κουμπιά ελέγχου λειτουργίας και προσαρμόστηκαν οι λειτουργίες των συσκευών ανά περίπτωση με χρήση αυτοματισμών.

Αφού ολοκληρώθηκε ο κεντρικός πίνακας ελέγχου και το σύστημα τέθηκε σε κατάσταση πλήρους λειτουργίας, γίνεται παρουσίαση ενός τρόπου αποθήκευσης των δεδομένων κατανάλωσης και λειτουργίας των συσκευών σε online βάση δεδομένων, όπως επίσης και ο τρόπος οπτικοποίησης αυτών των δεδομένων. Μάλιστα θα γίνει χρήση εργαλείων προβολής διαδραστικών γραφημάτων με υποστήριξη διατήρησης ιστορικού και ανανέωση των δεδομένων σε πραγματικό χρόνο.

Τέλος, γίνεται παράθεση των συμπερασμάτων που προέκυψαν από αυτή την εφαρμογή, τα θέματα ασφαλείας στο σύνολο της υλοποίησης, τις δυσκολίες που προέκυψαν, ενώ θα γίνει αναφορά σε μελλοντικές βελτιώσεις και προοπτικές ενός τέτοιου δικτύου.

Καινοτομία

Τα στοιχεία, που κάνουν την παρακάτω διπλωματική εργασία πρωτότυπη, είναι ταυτόχρονα:

- ✓ Η επίτευξη εξοικονόμησης ενέργειας με χρήση τεχνολογιών IoT.
- ✓ Η μετατροπή των ενεργοβόρων ηλεκτρικών συσκευών σε «έξυπνες».

- ✓ Η προσθήκη κεντρικοποιημένου απομακρυσμένου ελέγχου, αυτοματισμών και διασυνδεσιμότητας.
- ✓ Η ασφαλής πρόσβαση

Και όλα αυτά, με το ελάχιστο δυνατό οικονομικό και περιβαλλοντικό κόστος.

Δομή

- Στο 1ο κεφάλαιο, θα γίνει αναφορά, σε κάποιες βασικές έννοιες απαραίτητες για την πλήρη κατανόηση των όσων θα αναλυθούν στα επόμενα κεφάλαια της διπλωματικής εργασίας.
- Στο 2ο κεφάλαιο, θα αναπτυχθεί πλήρως, το κατασκευαστικό μέρος της υλοποίησης . Θα περιγραφεί η ηλεκτρική εγκατάσταση της οικίας και θα σχεδιαστεί το δίκτυο αισθητήρων και ενεργοποιητών. Στη συνέχεια, θα γίνει παρουσίαση των απαραίτητων υπηρεσιών και η εγκατάσταση τους και τέλος θα περιγραφεί αναλυτικά, ο εξοπλισμός και ο τρόπος παραμετροποίησης του.
- Στο 3ο κεφάλαιο, θα παρουσιαστεί, το γραφικό περιβάλλον του Home Assistant. Θα δοθεί έμφαση στις λειτουργίες που χρησιμοποιήθηκαν στην παρούσα υλοποίηση, θα αναφερθούν κάποιες πολύ χρήσιμες δυνατότητες και θα διασφαλιστεί η ακεραιότητα του προσθέτοντας επιπλέον μέτρο ασφαλείας.
- Στο 4ο κεφάλαιο, θα περιγραφεί η διαδικασία αποθήκευσης των ενδείξεων, που ελήφθησαν από τους αισθητήρες και ο τρόπος οπτικοποίησης τους. Θα γίνει, σύντομη αναφορά στις βάσεις δεδομένων με επικέντρωση στην InfluxDB και τον λόγο για τον οποίο επιλέχθηκε. Στη συνέχεια θα γίνει η οπτικοποίηση των αποθηκευμένων δεδομένων με χρήση της Online πλατφόρμας Grafana. Θα γίνει περιγραφή και των δυνατοτήτων του Grafana και παρουσίαση του γραφικού περιβάλλοντος του.
- Στο τελευταίο κεφάλαιο θα παρουσιαστούν τα αποτελέσματα της ανάλυσης των δεδομένων που ελήφθησαν και θα δοθεί το τελικό συμπέρασμα για την απόδοση της υλοποίησης όσον αφορά την εξοικονόμηση ενέργειας. Επίσης θα γίνει αναφορά στα προβλήματα που προέκυψαν κατά την υλοποίηση καθώς και μελλοντικές βελτιώσεις ή προσθήκες που θα μπορούσαν να βοηθήσουν είτε στην αποδοτικότερη χρήση των συσκευών της οικίας είτε το ίδιο το δίκτυο.

ΚΕΦΑΛΑΙΟ 1: Βασικές Έννοιες

Είναι σημαντικό, να γίνουν σαφείς κάποιες βασικές έννοιες που θα δώσουν τις απαραίτητες γνώσεις στον αναγνώστη για το περιεχόμενο που θα ακολουθήσει. Η τεχνολογία αναπτύσσεται συνεχώς και εντάσσονται στον χώρο πάρα πολλές νέες έννοιες και μέσα. Πολλές φορές, υπάρχει σύγχυση όσον αφορά κάποιους όρους, που χρησιμοποιούνται καθημερινά και γίνονται εσφαλμένες διαπιστώσεις. Το αντικείμενο της παρούσας διπλωματικής εργασίας κάνει αναφορά στην ηλεκτρική ενέργεια, σε τεχνολογίες του Διαδικτύου των Πραγμάτων (ΔτΠ), διαδικτυακές εφαρμογές, πρωτόκολλα επικοινωνίας και μικροελεγκτές. Θα πρέπει να γίνει κατανοητή και σαφής η κάθε έννοια έτσι ώστε να μην υπάρχει περιθώριο αμφισβήτησης του αποτελέσματος.

1.1 Ηλεκτρική Ενέργεια

Η ηλεκτρική ενέργεια είναι, όπως έχει ήδη αναφερθεί μία μορφή ενέργειας που προέρχεται από την μετατροπή μίας άλλης πρωτογενούς ενέργειας. Τα εργοστάσια παραγωγής ενέργειας, με κατάλληλη επεξεργασία της εκάστοτε καύσιμη ύλης, παράγουν ηλεκτρική ενέργεια την οποία μεταφέρουν μέσω του δικτύου διανομής στους καταναλωτές και τα νοικοκυριά τους. Η ηλεκτρική ενέργεια, ως μονάδα μέτρησης, χρησιμοποιεί τα Joule ή W*sec και είναι το συνολικό ποσό της ισχύος που αποδίδεται η καταναλώνεται σε χρονικό διάστημα t (E=P*t). Η ενέργεια, συχνά συγχέεται με την ισχύ, λόγω του ότι ο πάροχοι χρησιμοποιούν την κιλοβατώρα (kWh) ως μονάδα μέτρησης. Η ισχύς, μετριέται σε Watt (ενεργός ισχύς) ή VA (φαινόμενη ισχύς) αλλά δεν προσδιορίζει το ίδιο μέγεθος. Με την ισχύ μετράμε πόσο γρήγορα παράγεται ή καταναλώνεται ενέργεια , ενώ η ηλεκτρική ενέργεια προκύπτει από την κίνηση ηλεκτρικά φορτισμένων σωματιδίων (ηλεκτρόνια) [7]. Η ισχύς σε ένα κύκλωμα με ωμικό φορτίο υπολογίζεται από το γινόμενο της τάσης και του ρεύματος (P=V*I). Η ισχύς σε VA λέγεται και φαινόμενη ισχύς και οφείλεται στην εφαρμογή φορτίων από επαγωγικά (πηνία) ή χωρητικά (πυκνωτές) στοιχεία σε κυκλώματα εναλλασσόμενου ρεύματος. Τέτοιου είδους φορτία, τείνουν να διατηρούν την ενέργεια τους και να την επιστρέφουν πίσω στο δίκτυο λόγω της ημιτονοειδούς εναλλαγής της τάσης και του ρεύματος που επιφέρουν διαφορά φάσης. Η ισχύς σε ένα τέτοιο κύκλωμα δίνεται από τον τύπο P=|V|*|I|*cosφ . Σε ένα οικιακό περιβάλλον, κάποιες από τις συσκευές που συνήθως έχουν επαγωγικά και χωρητικά φορτία είναι το ψυγείο και το κλιματιστικό, συσκευές δηλαδή που διαθέτουν ηλεκτροκινητήρες. Η μονάδα μέτρησης της ενέργειας θα έπρεπε να σύμφωνα με το SI να μετριέται σε Joules, οι λόγοι που έχει κυριαρχήσει η κιλοβατώρα για την μέτρηση της ηλεκτρικής ενέργειας είναι πρακτικοί. Για παράδειγμα:

Αν μία συσκευή έχει αναγραφόμενη στιγμιαία ισχύ 60Watt σημαίνει:

Δεδομένου ότι 1Watt=1Joule/sec έχουμε:

P=60Watt => P=60 Joule/sec

Η αντίστοιχη ενέργεια σε kWh, η οποία χρησιμοποιείται και από τους παρόχους ηλεκτρικής ενέργειας, ορίζει πόση ενέργεια καταναλώνεται σε διάστημα 1 ώρας. Δηλαδή :

*E=P*t=> E=60Watt*1h=>E=60Wh=0.06kWh*

Αντίστοιχα, αν η ισχύς της συσκευής ήταν μεγαλύτερη και θέλαμε να υπολογίσουμε την κατανάλωση ενέργειας θα παρατηρούσαμε ότι η τιμή σε Joule θα είναι πολύ μεγάλη και υπολογιστικά δεν είναι ευκόλως διαχειρίσιμη. Για παράδειγμα:

*Μία συσκευή με στιγμιαία ισχύ P=1kW (1000W ή 1000Joule/sec) θα καταναλώνει σε μία ώρα ενέργεια: E=P*t=> E=1000Joule/sec*3600sec=3.6*10^6 Joule ή πιο απλά*

*E=P*t => E=1000Watt*1h=> E=1000Wh=>E=1kWh* [8]

1.1.1 Παροχή Ηλεκτρικής Ενέργειας

Ο μεγαλύτερος πάροχος ηλεκτρικής ενέργειας στην Ελλάδα είναι η ΔΕΗ Α.Ε. (Δημόσια Επιχείρηση Ηλεκτρισμού) με έτος ίδρυσης το 1950 από το τότε ελληνικό κράτος. Λειτούργησε ως υπηρεσία κοινής ωφέλειας καθιστώντας τη μονοπώλιο εκείνη την εποχή. Ανέπτυξε εθνικό δίκτυο μεταφοράς ενέργειας, κατασκεύασε και εκμεταλλεύτηκε υπάρχουσες εγκαταστάσεις θερμικών και υδροηλεκτρικών εργοστασίων. Το 1999 συστάθηκε η ΡΑΕ (Ρυθμιστική Αρχή Ενέργειας), με βασική αρμοδιότητα την εποπτεία της εγχώριας αγοράς ηλεκτρικής ενέργειας και την εναρμόνιση με τις οδηγίες της Ευρωπαϊκής κοινότητας. Από το 2012 και μετά το μονοπώλιο στην παραγωγή και διανομή ηλεκτρικής ενέργειας έπαψε να υπάρχει, μετά από παρέμβαση της κυβέρνησης και την καθοδήγηση της ΡΑΕ, με γνώμονα την απελευθέρωση της αγοράς ηλεκτρικής ενέργειας και τη αύξηση του ανταγωνισμού [9]. Αυτή τη στιγμή η ΔΕΗ, απασχολεί 16.747 υπαλλήλους και έχει αναπτύξει θυγατρικές εταιρίες που έχουν αναλάβει ρόλους όπως η διαχείριση του δικτύου διανομής (ΔΕΔΔΗΕ Α.Ε.) και η παραγωγή ηλεκτρικής ενέργειας από ανανεώσιμες πηγές ενέργειας (ΔΕΗ ΑΝΑΝΕΩΣΙΜΕΣ Α.Ε.). Σε σχέση με άλλους πάροχους, κατέχει το μεγαλύτερο μερίδιο της αγοράς ενέργειας στα 67,15%. Τα καύσιμα που χρησιμοποιεί για την παραγωγή ηλεκτρικής ενέργειας πέρα των ανανεώσιμων πηγών είναι ο λιγνίτης (48%) και το φυσικό αέριο (20%).

Πίνακας 1.1 Χρήση ανανεώσιμων πηγών ενέργειας από την ΔΕΗ ΑΝΑΝΕΩΣΙΜΕΣ Α.Ε.(Έτος 2020) [10]

Τύπος Ανανεώσιμης Ενέργειας	**Ισχύς**	**Ενεργές Εγκαταστάσεις**
Αιολική ενέργεια	115MW	34
Φωτοβολταϊκά πάρκα	1MW	28
Υδροηλεκτρικοί Σταθμοί	68MW	18
Υβριδικό πάρκο ΝΑΕΡΑΣ (Αιολική & υδροηλεκτρική ενέργεια)	6,85MW	1
Γεωθερμία	Υπό ανάπτυξη	-
Βιομάζα	Υπό ανάπτυξη	-

Το δίκτυο της ΔΕΗ είναι συνολικής ισχύος 12,2GW και διαθέτει τις εξής παροχές ρεύματος:

Πίνακας 1.2 Είδη Παροχών ΔΕΗ

Είδος Παροχής	**Μέγεθος Παροχής**	**Ισχύς Παροχής (P) (kVA)**
Μονοφασική	No.3	5
	No.5	12
Τριφασική	No.1	15
	No.2	25
	No.3	35
	No.4	55
	No.5	85
	No.6	135
	No.7	250

Όλες οι παροχές στην Ελλάδα είναι εναλλασσόμενου ρεύματος με τάση τα 230Volt και συχνότητα 50Hz. Το εναλλασσόμενο ρεύμα, χρησιμοποιείται παγκοσμίως στη διανομή ηλεκτρικής ενέργειας , παρότι ένα μεγάλο πλήθος ηλεκτρικών συσκευών λειτουργούν με συνεχές ρεύμα. Ο λόγος είναι ό,τι μπορεί να μεταφέρεται πιο εύκολα σε μεγαλύτερες αποστάσεις, με πολύ μικρές θερμικές απώλειες αλλά και με λεπτότερους αγωγούς. Είναι γνωστό άλλωστε ό,τι όσο μεγαλύτερη είναι η τάση τόσο μικρότερη θα είναι η ένταση του ρεύματος, άρα και η διατομή του αγωγού.

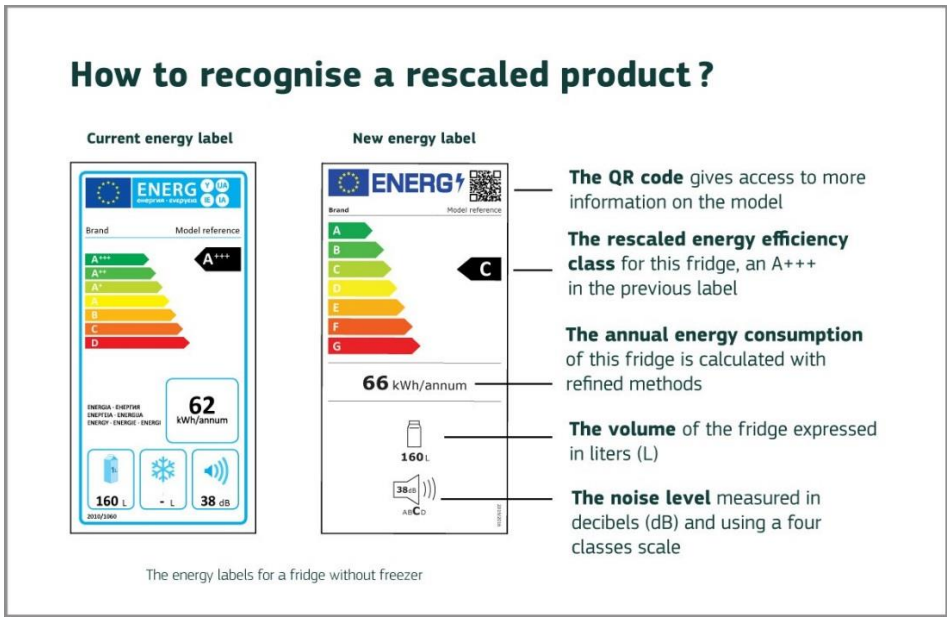
1.1.2 Τρόποι καταμέτρησης και χρέωσης

Ο τρόπος καταμέτρηση της ενέργειας που καταναλώθηκε από την ΔΕΗ, προέρχεται από τις μετρήσεις που λαμβάνει το εκάστοτε προσωπικό από τους μετρητές ηλεκτρικής ενέργειας κάθε 4 μήνες. Οι μετρητές αυτοί είναι ήδη εγκαταστημένοι, σε κάποιο προσβάσιμο για αυτούς σημείο, συνήθως εξωτερικά του κτιρίου. Οι καταναλωτές, λαμβάνουν λογαριασμό «ΕΝΑΝΤΙ» χρέωσης, κάθε 2 μήνες ο οποίος είναι ένας κατ' εκτίμηση λογαριασμός και τους επόμενους 2 μήνες τον «ΕΞΟΦΛΗΤΙΚΟ». Σε αυτούς αναφέρονται αναλυτικά οι μετρήσεις και πιθανές πρόσθετες χρεώσεις που αφορούν δημοτικά τέλη, τέλη συντήρησης του δικτύου κ.α. Υπάρχουν παροχές, που διαθέτουν δύο μετρητές ένα για το «ημερήσιο» και έναν για το «νυχτερινό» τιμολόγιο. Οι μετρήσεις αυτών των μετρητών, φαίνονται αναλυτικά στον λογαριασμό που λαμβάνει ο καταναλωτής. Με αυτούς τους μετρητές, ο καταναλωτής έχει το πλεονέκτημα της οικονομικότερης κατανάλωσης ενέργειας, σε ώρες που έχει ορίσει ο πάροχος. Ταυτόχρονα, ο πάροχος επιτυγχάνει πιο σταθερή χρήση του δικτύου, όλες τις ώρες και ημέρες. Οι ώρες λειτουργίας του νυχτερινού τιμολογίου από την ΔΕΗ είναι:

- Για την χειμερινή περίοδο (1/11-30/4): 02:00-08:00 και 15:00-17:00
- Για την θερινή περίοδο (1/5-31-10): 23:00-07:00

Η χρέωση της κιλοβατώρας στο ημερήσιο τιμολόγιο, γίνεται κλιμακωτά και κυμαίνεται στα 0,11058€/kWh για τις πρώτες 2000kWh και 0,11936€/kWh για πάνω από 2000kWh. Το νυχτερινό τιμολόγιο έχει σταθερή χρέωση στα 0,07897€/kWh.[11]

1.1.3 Τυπικές καταναλώσεις οικιακών συσκευών.



Εικόνα 1.1 Δείγματα παλαιών και νέων ετικετών ενεργειακής απόδοσης[12]

Είναι σημαντικό, να λαμβάνονται υπόψιν οι απαιτήσεις ισχύος των ηλεκτρικών συσκευών που χρησιμοποιούνται. Ένας τρόπος αναγνώρισης των τυπικών καταναλώσεων των ηλεκτρικών συσκευών μίας οικίας είναι να εντοπιστεί το σχετικό αυτοκόλλητο το οποίο αναγράφει συνήθως την μέγιστη απαιτούμενη ισχύ της συσκευής καθώς και τα χαρακτηριστικά λειτουργίας της. Η Ευρωπαϊκή ένωση, στα πλαίσια της περιβαλλοντικής πολιτικής της, έχει θεσπίσει και την ετικέτα ενεργειακής απόδοσης , η οποία μάλιστα ανανεώθηκε από τον Μάρτιο του 2021, καθώς έπρεπε να γίνει αναπροσαρμογή της κλίμακας υπολογισμού της ενεργειακής απόδοσης (Εικόνα 1.1). Ο Πίνακας 1.3, απεικονίζει μερικές από τις ηλεκτρικές συσκευές που χρησιμοποιούνται, σε οικιακό περιβάλλον και τις συνήθεις απαιτήσεις ισχύος τους.

Πίνακας 1.3 Ενδεικτικές τιμές καταναλώσεων οικιακών ηλεκτρικών συσκευών [13]

Συσκευή	Λειτουργία	Ισχύς (Watt)	Ενδεικτική Κατανάλωση ανά περίοδο λειτουργίας (kWh)	Κόστος (€)
Ηλεκτρικό σίδερο	1 ώρα	1000	1	0,15
Τηλεόραση (έγχρωμη)	1 ώρα	41	0,041	0,005
Η/Υ (PC)	1 ώρα	250	0,25	0,035
Πλυντήριο πιάτων μεγάλο	55 °C / γεμάτο (οικονομικό πρόγρ/μα)	3200	0,8	0,10
Πλυντήριο ρούχων	1πλύση (40° C / 5κ. ρούχα)	2800	0,50	0,07
Ψυγείο με κατάψυξη 131 λίτρων	24 ώρες	90	0,50	0,07
Θερμοσίφωνας 80 Λίτρων	50 °C/ 1ώρα/ συντήρηση	4000	2,6	0,33
Φούρνος απλός	1 κέικ (50 λεπτά)	2700	2,25	0,280
Φούρνος μικροκυμάτων	5 λεπτά	800	0,06	0,009
Κλιματιστικό (ψύξη 9000 BTU)	1 ώρα σε χώρο 15 τ.μ.	1000	1	0,14
Ηλεκτρικό θερμαντικό σώμα	1 ώρα	2000	2	0,3
Λαμπτήρας Χαμηλής κατανάλωσης	1 ώρα (ίδιας φωτεινότητας με κοινό 100W)	20	0,02	0,003
Λαμπτήρας (κοινός) πυρακτώσεως	1 ώρα	100	0,10	0,013

Γνωρίζοντας τις παραπάνω ενδεικτικές τιμές, που αφορούν την κατανάλωση κάθε συσκευής μπορεί να γίνει εύκολα αντιληπτό ό,τι οι πιο ενεργοβόρες συσκευές θα πρέπει να περιοριστούν, είτε όσον αφορά τον χρόνο λειτουργίας τους, είτε όσον αφορά την επιλογή τρόπου λειτουργίας. Κάνοντας μία έρευνα εντός του οικιακού

περιβάλλοντος, μπορούν να καταγραφούν κάποιες καθημερινές συνήθειες των ενοίκων που αφορούν την χρήση των ηλεκτρικών συσκευών και να περιοριστούν τυχών σπατάλες ενέργειας. Αυτό όμως, είναι αρκετά δύσκολο αν γίνει χειροκίνητα, καθώς είναι ανθρωπίνως αδύνατο να γίνει παρακολούθηση 24 ώρες το 24ωρο την χρήση όλων των συσκευών. Ο πιο αποδοτικός τρόπος, παρακολούθησης της ενεργειακής κατανάλωσης, είναι με την αυτοματοποίηση της καταγραφής των μετρήσεων και της αυτόματης διατήρησης ιστορικού αυτών των μετρήσεων. Λαμβάνοντας υπόψιν τις μετρήσεις αυτές, το χρονοδιάγραμμα και τις περιβαλλοντικές συνθήκες μπορεί να αποκτηθεί μία πιο σφαιρική εικόνα των ενεργειακών συνηθειών των ενοίκων. Η επιλογή λειτουργιών, «εξοικονόμησης ενέργειας» ή η χρήση ενεργειακά πιο αποδοτικών συσκευών, θα βελτιώσει την συνολική ενεργειακή κατανάλωση, αλλά δεν θα περιορίσει την ακαταλόγιστη χρήση. Το επόμενο στάδιο επομένως, είναι η πρόληψη. Η αποφυγή, δηλαδή, της ενεργειακής σπατάλης λόγω του ανθρώπινου παράγοντα. Η αυτοματοποίηση, της λειτουργίας κάθε συσκευής που φαίνεται να χρησιμοποιείτε ενώ δεν υπάρχει λόγος, είναι ένας τρόπος να περιορίσουμε τυχών ενεργειακή σπατάλη.

1.2 Διαδίκτυο των Πραγμάτων

Ολοένα και πιο συχνά, όταν γίνεται λόγος για έξυπνες συσκευές, απομακρυσμένη διαχείριση, αυτοματισμούς, εμφανίζεται ο όρος «Διαδίκτυο των Πραγμάτων» ή «Internet of Things». Το διαδίκτυο των πραγμάτων θα μπορούσε να εκφραστεί και ως η εξέλιξη της διασυνδεσιμότητας των διαφόρων ηλεκτρικών συσκευών. Το σύνολο αυτών των συσκευών, που διαθέτουν αισθητήρες και ενεργοποιητές, έχουν την δυνατότητα να επικοινωνούν και να αλληλοεπιδρούν μέσα σε ένα δικτυακό περιβάλλον χωρίς να είναι απαραίτητη η ανθρώπινη παρέμβαση. Με το ΔτΠ, γίνεται κατά κάποιο τρόπο σύνδεση του φυσικού κόσμου, με τον ψηφιακό. Οι συσκευές αυτές πολλές φορές αποκαλούνται «έξυπνες», καθώς είναι η εξέλιξη των κοινών συσκευών και διαθέτουν επιπλέον λειτουργίες που τις αυτοματοποιούν ή τους δίνουν δυνατότητες απομακρυσμένης διαχείρισης και παρακολούθησης. Είναι σημαντικό να επισημανθεί, ότι οι συσκευές αυτές δεν είναι απαραίτητα οικιακής χρήσεως. Το διαδίκτυο των πραγμάτων έχει εξαπλωθεί και σε άλλους κλάδους όπως τον ιατρικό κλάδο και την βιομηχανία. [14], [15]. Η ασφάλεια είναι ένας ακόμη σημαντικός παράγοντας σε κλάδους όπως η βιομηχανία και η υγεία. Μία κακόβουλη επίθεση, σε ένα τέτοιο δίκτυο IoT μπορεί να επιφέρει μέχρι και ανθρώπινες απώλειες. Επομένως, τα πρωτόκολλα επικοινωνίας που διαχειρίζονται δίκτυα IoT πρέπει να εξασφαλίζουν την προστασία των συσκευών από εξωτερικές απειλές, να διαθέτουν δικλείδες ασφαλείας σε περίπτωση απώλειας σύνδεσης και να δίνουν αποκλειστικά δικαιώματα μόνο σε επιλεγμένους χρήστες.

Η αρχιτεκτονική ενός δικτύου IoT, σύμφωνα με έρευνα του [16] έχει περιγραφεί είτε με τρία είτε με τέσσερα επίπεδα ο ίδιος επιλέγει τον διαχωρισμό σε έξι επίπεδα η οποία έχει δομή αντίστοιχη με την ιεραρχική δομή όλων των δικτύων που βασίζονται στο πρότυπο OSI . Παρόλα αυτά, η συνοπτική εκδοχή των τριών επιπέδων μπορεί να καλύψει και τα ενδιάμεσα επίπεδα . Τα τρία επίπεδα της αρχιτεκτονικής των δικτύων IoT είναι : της συσκευής (device), της σύνδεσης (connection) και της εφαρμογής (application). Το επίπεδο της συσκευής αποτελεί την βάση της αρχιτεκτονικής και περιλαμβάνει τις συσκευές και τις εφαρμογές που τις απαρτίζουν, μαζί με την συνδεσιμότητα τους σε μικρής εμβέλειας δίκτυα. Το επίπεδο συνδεσιμότητας, αποτελείται από τις πύλες διασύνδεσης (gateways) των συσκευών και τον πυρήνα του δικτύου ο οποίος αναλαμβάνει την επικοινωνία με άλλα δίκτυα όπως το Zigbee,WiFi, WiMAX κ.α.. Το επίπεδο της εφαρμογής, λειτουργεί παράλληλα με τις πύλες σε ένα ενιαίο πλαίσιο διασύνδεσης με τα παραπάνω δίκτυα, επιτρέποντας για παράδειγμα την χρήση υπηρεσιών υπολογιστικής νέφους και μέσω αυτής την διεπαφή για διαχείριση των συσκευών από τον τελικό χρήστη [17].

Με την αύξηση των απαιτήσεων για διασυνδεσιμότητα και επικοινωνία με το διαδίκτυο έχουν αναπτυχθεί υλοποιήσεις μικροελεγκτών, που με κατάλληλο προγραμματισμό μπορούν να μετατρέψουν τις τυπικές συσκευές σε «έξυπνες». Η χρήση αυτών των μικροελεγκτών, δεν περιορίζεται μόνο στον διασύνδεση των συσκευών με το διαδίκτυο, αλλά και την προσθήκη επιπλέον δυνατοτήτων. Οι συσκευές μπορούν, να αποκτήσουν επιπλέον αισθητήρες ή ενεργοποιητές ακόμα και δυνατότητες τεχνητής νοημοσύνης (AI). Στην αγορά υπάρχει πληθώρα έτοιμων «έξυπνων» συσκευών που διαθέτουν τεχνολογίες IoT. Μερικές από αυτές είναι, τα κλιματιστικά με δυνατότητες ασύρματης επικοινωνίας, οι έξυπνοι λαμπτήρες, οι ρομποτικές σκούπες, οι έξυπνοι ρευματοδότες, οι έξυπνοι ασφαλειοδιακόπτες ράγας και οι έξυπνοι μετρητές ενέργειας. Η ενσωμάτωση τεχνολογιών IoT σε συσκευές όπως οι μετρητές ενέργειας, προσφέρουν την δυνατότητες άμεσης και συνεχούς παρακολούθησης αλλά και ελέγχου των επιμέρους συσκευών μίας οικίας . Η δημιουργία ενός δικτύου αισθητήρων, με κεντρικοποιημένο απομακρυσμένο έλεγχο και προσθήκη αυτοματισμών, βελτιστοποιούν στο μέγιστο την εξοικονόμηση ενέργειας. Έμμεσα, συμβάλουν στην εξοικονόμηση ορυκτών πόρων από τις εταιρίες παραγωγής ηλεκτρικής ενέργειας, στην προστασία του περιβάλλοντος λόγω της μειωμένης εκπομπής CO_2 και την οικονομική ευμάρεια των νοικοκυριών.

1.3 Τεχνολογίες Επικοινωνιών

Στην προηγούμενη ενότητα, έγινε αναφορά στα επίπεδα αρχιτεκτονικής του IoT. Στο επίπεδο σύνδεσης έγινε αναφορά στις τεχνολογίες επικοινωνιών μεταξύ των συσκευών και του κεντρικού δικτύου. Στην αγορά, έχουν αρχίσει να κάνουν την εμφάνιση τους συσκευές IoT που έχουν διαφορετικό τρόπο ασύρματης διασύνδεσης πέραν του WiFi. Τα πιο διαδεδομένα για χρήση σε οικιακό περιβάλλον είναι, το Zigbee, το Bluetooth, και το Z-wave, καθώς καλύπτουν τις απαιτήσεις εμβέλειας μίας τυπικής οικίας. Στην παρούσα διπλωματική, επιλέχθηκε η χρήση του WiFi, ως μέσου ασύρματης επικοινωνίας με το κεντρικό δίκτυο της οικίας. Όπως φαίνεται και στον Πίνακας 1.4, το WiFi σε σύγκριση με τις άλλες τεχνολογίες δεν έχει πάρα πολλή μεγάλη εμβέλεια (Zigbee, 10-300m σε οπτική επαφή & 75-100m σε εσωτερικό χώρο), ούτε χαμηλή κατανάλωση ενέργειας (Zigbee, Bluetooth, Z-wave). Παρόλα αυτά, παρέχει υψηλές ταχύτητες, είναι ήδη διαθέσιμο στα περισσότερα νοικοκυριά, δεν απαιτεί επιπρόσθετο κόμβο ή πύλη πρόσβασης για την επικοινωνία μεταξύ άλλων WiFi συσκευών, έχει καλή ασφάλεια πρόσβασης και ενσωματώνεται στην πλειονότητα των IoT συσκευών.

Πίνακας 1.4 Συγκεντρωτικός πίνακας ασύρματων τεχνολογιών IoT.[18]

Ασύρματη Τεχνολογία	**Ρυθμός Μετά/σης**	**Μέγιστο Μήκος Δεδο/νων**	**Εμβέλεια Επικοι/νίας**	**Ασφάλεια**	**Δυνατά σημεία**
LoRaWAN	50kb/s	243 Bytes	~5km Εντός πόλης ~15km-20km Επαρχία	128bit AES	• Χαμηλή κατανάλωση • Μεγάλη εμβέλεια • Χαμηλό κόστος • Ασφαλές • Διαθεσιμότητα
Sigfox	100b/s	12 Bytes	~10km Εντός πόλης ~40km Επαρχία	Καμία κρυπτο/φηση ή Προσα/σιμο ανά περίπτωση	• Μεγάλη εμβέλεια • Χαμηλή κατανάλωση
NB-IoT	200kb/s	1600 Bytes	~1km Εντός πόλης ~10km Επαρχία	Κρυπτο/φηση LTE	• Μεγάλο μέγιστο μήκος μεταδιδόμενης πληροφορίας

Ασύρματη Τεχνολογία	**Ρυθμός Μετά/σης**	**Μέγιστο Μήκος Δεδο/νων**	**Εμβέλεια Επικοι/νίας**	**Ασφάλεια**	**Δυνατά σημεία**
					• Ασφαλές
Wi-Fi	Top 1Gb/s – IEEE 802.11ac	2034 Bytes	1-100m	WPA/WPA2	• Υψηλή ταχύτητα • Προηγμένο/ Ώριμο πρότυπο
Zigbee	250kb/s @2Ghz 40kb/s@15 MHz 20kb/s@86 8MHz	255 Bytes	10-300m σε οπτική επαφή 75-100m σε εσωτερικό χώρο	128bit AES	• Χαμηλό κόστος • Χαμηλής ισχύος • Υποστήριξη πολλών κόμβων(έως 65000 κόμβοι) • Ασφαλές
Bluetooth	2Mb/s 500kb/s (Long Range S=2) 125kb/s (Long Range S=7)	255 Bytes	Έως 200m +200m (BLE)	L1-Καμία ασφάλεια L2-AES 128 L3-AES and Pairing L4-ECDHE	• Εύκολη πρόσβαση και εγκατάσταση • Απλό hardware • Ασφαλές • Χαμηλής κατανάλωσης (BLE)
Z-Wave	100kb/s	64 Bytes	~100m (μπορεί να επηρεαστεί από το πλήθος κόμβων) (έως 4 hops)	Security 2 (S2) (Περι/μβάνει AES-128, ECDHE, secure TLS tunnel)	• Χαμηλής κατανάλωσης • Απλή εγκατάσταση • Ασφαλές • Διασυνδεσιμότητα με συσκευές άλλων κατασκευαστών

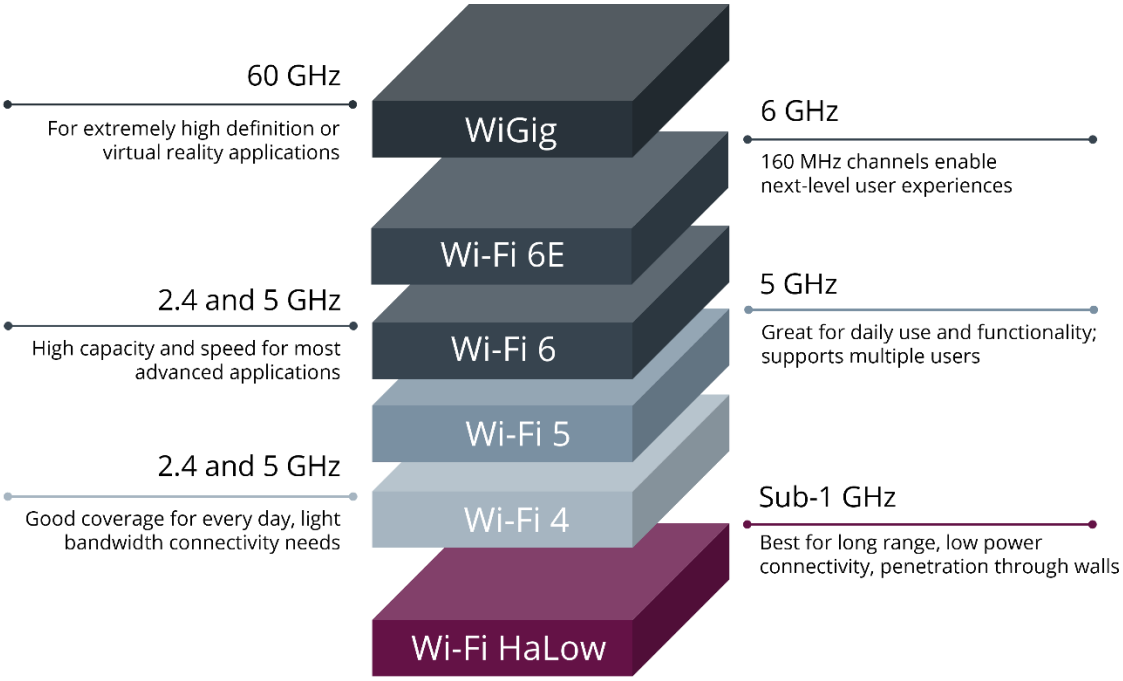
Για να επιτευχθεί όμως η μετάδοση των πληροφοριών, μέσω του WiFi, από τους διάφορους αισθητήρες και ενεργοποιητές έπρεπε να χρησιμοποιηθεί και ένα πρωτόκολλο επικοινωνίας M2M (Machine to Machine) . Το πρωτόκολλο που επιλέχθηκε είναι το MQTT, το οποίο εξασφαλίζει την ταχύτατη και ασφαλή μετάδοση μηνυμάτων από και προς τις διάφορες συσκευές. Ο Πίνακας 1.5, απεικονίζει συγκριτικά, κάποια από τα χαρακτηριστικά των πιο διάσημων IoT πρωτοκόλλων επικοινωνίας.

Πίνακας 1.5 Σύγκριση IoT πρωτοκόλλων M2M[19]

Standard	***MQTT***	***CoAP***	***AMQP***	***HTTP***
Μοντέλο	Publish/Subscribe	Request/Response	Publish/Subscribe	Request/Response
Μέγεθος Πακέτου	Πολύ μικρό	Μικρότερο	Μεγαλύτερο	Μεγάλο
Πρω/κολλο μετάδοσης	TCP	UDP	TCP	UDP
Μορφή κωδικο/σης	Δυαδική	Δυαδική	Δυαδική	Κείμενο

1.3.1 WiFi

Όπως ήδη αναφέρθηκε, στην παρούσα διπλωματική επιλέχθηκε το WiFi ως τεχνολογία ασύρματης επικοινωνίας. Το WiFi, είναι ένα πρότυπο δικτύωσης που χρησιμοποιείται ευρέως στην ασύρματη επικοινωνία των συσκευών, με το τοπικό δίκτυο αλλά και το διαδίκτυο. Αναπτύχθηκε από την WiFi Alliance, από την οποία ορίστηκε και το όνομα του. Ως αρχικό σκοπό ύπαρξης, είχε την δημιουργία ενός πρότυπου τοπικών ασύρματων δικτύων υπολογιστών WLAN (Wireless Local Area Network) και συνεπώς την ασύρματη διασύνδεση των υπολογιστικών συστημάτων, όπου αυτή η σύνδεση με καλώδιο, ήταν δύσκολη ή και αδύνατη. Οι συχνότητες λειτουργίας WiFi, που χρησιμοποιούν οι κοινές δικτυακές συσκευές, που μπορεί να υπάρχουν σε μία οικία, είναι τα 2,4GHz και τα 5GHz. Το φάσμα των 2,4GHz τείνει να χρησιμοποιείτε και από άλλες συσκευές, όπως τα ασύρματα τηλέφωνα σταθερής τηλεφωνίας και ο φούρνος μικροκυμάτων που μπορεί να προκαλέσουν παρεμβολές ή θόρυβο στην μετάδοση δεδομένων. Το φάσμα των 5GHz, αν και δεν χρησιμοποιείται από πολλές συσκευές έχει μικρότερη εμβέλεια και επηρεάζεται πιο εύκολα από φυσικά εμπόδια. Πρόσφατα, η WiFi Alliance ανέπτυξε το WiFi HaLow (802.11ah), ένα πρότυπο που αφορά την ασύρματη δικτύωση IoT συσκευών με σκοπό να καλύψει την ανάγκη για μείωση των παρεμβολών , αύξηση της εμβέλειας σήματος και την μείωση της κατανάλωσης ισχύος , διατηρώντας πάντα υψηλή ασφάλεια επικοινωνίας και εύκολη πρόσβαση χωρίς επιπλέον εξοπλισμό.[20]



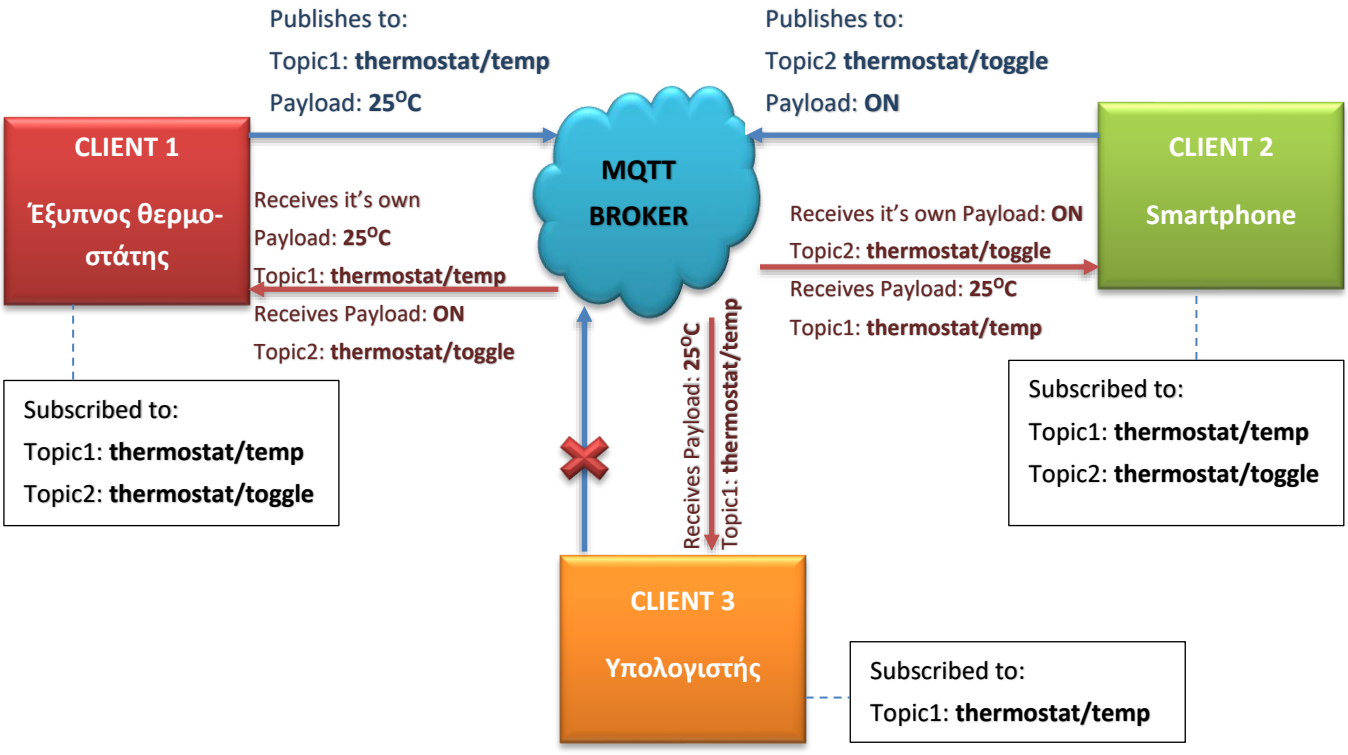
Εικόνα 1.2 Κατηγοριοποίηση προτύπων WiFi ανά συχνότητα λειτουργίας[20]

Το πρότυπο της IEEE 802.11X, ήταν ο πρωταρχικός τρόπος κατηγοριοποίησης των WLAN βάση της ταχύτητας μετάδοσης δεδομένων. Η WiFi Alliance άλλαξε τον τρόπο κατηγοριοποίησης των προτύπων χρησιμοποιώντας την μορφή WiFi και ένα αριθμό βάση του οποίου προσδιορίζεται το φάσμα συχνοτήτων και η ταχύτητα μετάδοσης των δεδομένων. Το πρότυπο για οικιακά δίκτυα με την πιο υψηλή ταχύτητα είναι, κατά την IEEE το 802.11ac ή αντίστοιχα κατά την WiFi Alliance WiFi 5. Χρησιμοποιεί πολλαπλές κεραίες, λειτουργεί στα 5GHz και η ταχύτητα του φθάνει τα 3,46Gbps. Το WiFi 6 (802.11ax) λειτουργεί και στα δύο φάσματα συχνοτητων (2,4GHz και 5GHz) αλλά αφορά κυρίως δίκτυα με πιο μεγάλες απαιτήσεις σε συνδεδεμένες συσκευές όπως γήπεδα και αεροδρόμια προσφέροντας ρυθμούς μετάδοσης έως και 9,6Gbps. Απο πλευράς ασφάλειας,για να αποκτήσει κάποιος πρόσβαση σε ένα δίκτυο WiFi θα πρέπει να γνωρίζει τον κωδικό πρόσβασης του

εκάστοτε δικτύου. Ο κωδικός πρόσβασης ενός δικτύου WiFi κρυπτογραφείτε με βάση κάποιο από τα παρακάτω πρότυπα ταυτοποίησης:WEP, WPA,WPA2, WPA preshared key(WPA-PSK). Τα πρότυπα WPA και WPA2 είναι τα ισχυρά πρότυπα τα οποία αντικατέστησαν το WEP . Το WPA χρησιμοποιεί το προτόκολλο TKIP το οποίο για να υλοποιήσει την τελική κρυπτρογράφηση χρησιμοποιεί μία μέθοδο δημιουργίας κλειδιού ανα πακέτο, αναμειξης με επιπλεον πληροφορίες που αφορούν την συσκευή και δημιουργία ενός τελικού ενιαίου κλειδιού. Το WPA2 λειτουργεί παρόμοια με το WPA μόνο που χρησιμοποιεί ένα πιο ισχυρό πρωτόκολλο το AES. Το WEP έχει μία πιο απλοποιημένη και όχι τόσο ασφαλή μέθοδο κρυπτογράφησης καθώς χρησιμοποιεί ένα κλειδί το οποίο στέλνετε από τον client σε μορφή κειμένου κωδικοποιείται και επιστρέφει ξανά πίσω στον client. Το WPA PSK χρησιμοποιεί μία φράση κλειδί σε περίπτωση που δεν υπάρχει διαθέσιμος τρόπος ταυτοποίησης και μπορεί να είναι το ίδιο αδύναμο με το WEP αν χρησιμοποιηθεί εύκολη φράση-κλειδί.

1.3.2 MQTT

Οι συσκευές IoT, χρειάζεται να αποστέλλουν και να λαμβάνουν ανελλιπώς πληροφορίες από και προς τους ενδιαφερόμενους. Για να γίνει αυτό, έπρεπε να επιλεχθεί ένα πρωτόκολλο επικοινωνίας αρκετά γρήγορο, ελαφρύ, απλό στην χρήση του, με εγγυημένη επικοινωνία με όλες αυτές τις συσκευές καθώς και ένα καλό επίπεδο ασφάλειας. Το MQTT είναι ένα πρωτόκολλο, που σχεδιάστηκε για επικοινωνία M2M (Machine to Machine) με την μέθοδο «Subscribe/Publish». Η αρχιτεκτονική του βασίζεται στο επίπεδο μεταφοράς του πρωτόκολλου TCP/IP και αποτελείται από τον «MQTT Broker» και τους «MQTT clients». Οι «MQTT clients» είναι είτε συσκευές IoT που στέλνουν ή λαμβάνουν μηνύματα προς ή από τον MQTT Broker είτε άλλες συσκευές όπως smartphones ή υπολογιστές. Ο «MQTT Broker» είναι στην ουσία ένας ενδιάμεσος Server ο οποίος διαχειρίζεται τα μηνύματα που λαμβάνει από τους clients προωθώντας τα στα αντίστοιχα «Topics» όπως επίσης, αναλαμβάνει και την διαδικασία του Subscribe/Unsubscribe. Για να μπορέσει ένας client να αποστείλει ή να διαβάσει ένα μήνυμα στον MQTT Broker θα πρέπει να έχει κάνει πρώτα «Subscribe» στο αντίστοιχο Topic.



Εικόνα 1.3 Παράδειγμα αρχιτεκτονικής πρωτόκολλου MQTT

Στο παράδειγμα της Εικόνα 1.3, βλέπουμε μία απλή αρχιτεκτονική MQTT. Υπάρχουν τρείς clients, η συσκευή IoT που είναι ένας έξυπνος θερμοστάτης (Client 1), ένα smartphone (Client 2) και ένας υπολογιστής (Client 3). Η συσκευή IoT, έχει δημιουργήσει δύο topics, στο 1^{ο} (thermostat/temp) αποστέλλει την τρέχουσα θερμοκρασία δωματίου και στο 2^{ο} (thermostat/toggle) περιμένει να λάβει μήνυμα λειτουργίας «ON ή OFF». Μόνο οι εγγεγραμμένοι στα topics μπορούν να δουν τα περιεχόμενα των μηνυμάτων. Ο Client 2 μπορεί να «δει» την τρέχουσα θερμοκρασία και αν έχει σταλεί εντολή λειτουργίας, επίσης μπορεί να στείλει και εντολή λειτουργίας. Ο Client 3 που είναι εγγεγραμμένος, μόνο στο topic της τρέχουσας θερμοκρασίας μπορεί να «δει» την τρέχουσα θερμοκρασία αλλά δεν μπορεί να αποστείλει μήνυμα αλλαγής λειτουργίας του θερμοστάτη.

Στα πλαίσια της διασφάλισης της ποιότητας της αποστολής των μηνυμάτων, το πρωτόκολλο MQTT διαθέτει τρείς επιλογές QoS 0 , QoS 1,QoS 2. Το QoS 0 περιγράφεται ως «το πολύ μία», αφού αποσταλεί το μήνυμα δεν διασφαλίζεται η παραλαβή ή επαναποστολή σε περίπτωση αποτυχίας. Το QoS 1 περιγράφεται ως «τουλάχιστον μία», αφού αποσταλεί το μήνυμα ο client περιμένει ένα μήνυμα επιβεβαίωσης ACK, αν δεν λάβει το ACK ,σε ένα συγκεκριμένο χρονικό διάστημα, τότε το αρχικό μήνυμα ξαναστέλνετε. Με αυτό τον τρόπο διασφαλίζετε ότι το μήνυμα θα φτάσει τουλάχιστον μία φορά αλλά δεν διασφαλίζετε ότι δεν θα επαναληφθεί. Το QoS 2, περιγράφεται ως και διασφαλίζει την αποστολή του μηνύματος ακριβώς μία φορά, χρησιμοποιώντας παρόμοια τεχνική με το QoS 1, προσθέτοντας όμως ένα ακόμη επιβεβαιωτικό μήνυμα που διασφαλίζει την μη επαναληψιμότητα της αποστολής του αρχικού μηνύματος (τετραπλή χειραψία). Η επιλογή του κατάλληλου QoS γίνεται με τα εξής κριτήρια:

- QoS 0: όταν είναι αποδεκτή η περιστασιακή απώλεια κάποιων μηνυμάτων ή όταν έχουμε διασφαλισμένη και σταθερή διασύνδεση μεταξύ client και broker.
- QoS 1: όταν θέλουμε μέγιστη απόδοση με μικρή κατανάλωση πόρων, ή όταν δεν θέλουμε να χάσουμε κανένα μήνυμα ακόμα και αν αυτό σημαίνει ότι μπορεί να το παραλάβουμε ξανά.
- QoS 2: όταν δεν θέλουμε να χάσουμε κανένα μήνυμα αλλά να μην το παραλάβουμε παραπάνω από μία φορά ή όταν θέλουμε να διασφαλιστεί η ολοκληρωμένη αποστολή κάθε μηνύματος όσο και αν αυτό κοστίσει σε χρόνο επεξεργασίας.[21]

Όταν γίνεται η αποστολή ενός μηνύματος, με χρήση του πρωτοκόλλου MQTT, το μήνυμα θα διαβαστεί από τους clients που είναι εκείνη την στιγμή συνδεδεμένοι στον MQTT broker και έχουν κάνει Subscribe στο συγκεκριμένο topic. Οποιοσδήποτε, συνδεθεί σε άλλη χρονική στιγμή στον broker, ακόμα και αν έχει κάνει subscribe σε αυτό το topic, δεν θα μπορέσει να διαβάσει τα προηγούμενα μηνύματα που εστάλησαν στον broker. Για τον λόγο αυτό, δίνεται η δυνατότητα στον Publisher, να ορίσει αν το μήνυμα που αποστέλλει θα είναι «Retained». Με την ενεργοποίηση της επιλογής retain, το τελευταίο μήνυμα που έλαβε ο broker σε αυτό το topic, θα παραμείνει για προβολή από οποιονδήποτε client κάνει subscribe και σύνδεση σε αυτό.

Οι συσκευές που επικοινωνούν με μηνύματα μέσω MQTT, συμβαίνει συχνά να τίθενται εκτός λειτουργίας. Για να μπορεί κάποιος subscriber να γνωρίζει αν η συσκευή είναι συνδεδεμένη και λειτουργεί κανονικά , μπορεί να γίνει χρήση του «Last will message». Το μήνυμα αυτό δήλωσης κατάστασης της συσκευής αποστέλλει στο topic του Last Will ένα μήνυμα που έχει ορίσει ο client της συσκευής αυτής (client A) το οποίο ειδοποιεί όλους τους subscriber που θα συνδεθούν σε αυτό ότι η συσκευή (client A) είναι «ενεργή» ή «ανενεργή». Το μήνυμα «ενεργοποίησης/ απενεργοποίησης» είναι στην ευχέρεια του Client, δεν είναι προκαθορισμένο και μπορεί να λάβει τις ίδιες δυνατότητες QoS και Retain όπως όλα τα μηνύματα MQTT.[22]

Ένας MQTT broker μπορεί να εγκατασταθεί, είτε σε ένα μικρό υπολογιστικό σύστημα όπως το Raspberry Pi, είτε σε κάποιο υπολογιστικό σύστημα νέφους. Στην πρώτη περίπτωση, ο broker επικοινωνεί άμεσα με τις

συσκευές που ανήκουν στο ίδιο τοπικό δίκτυο, ενώ στην δεύτερη περίπτωση επικοινωνεί με τις συσκευές μέσω διαδικτύου, έστω και αν αυτές δεν βρίσκονται στον ίδιο χώρο ή τοπικό δίκτυο. Όταν το MQTT χρησιμοποιείτε μέσω cloud, θα πρέπει οπωσδήποτε να προστεθεί ένα επιπλέον επίπεδο ασφάλειας καθώς είναι πολύ εύκολο κάποιος που γνωρίζει τα topics και την διεύθυνση του broker να επηρεάσει κακόβουλα τα μηνύματα που λαμβάνονται από τους clients. Η ασφάλεια πρόσβασης στον MQTT broker, μπορεί να ενισχυθεί με δύο τρόπους είτε προσθέτοντας όνομα χρήστη και κωδικό πρόσβασης και κρυπτογράφηση είτε με πιστοποίηση της ταυτότητας του Client μέσω του πρωτοκόλλου SSL.

1.4 Υπολογιστική νέφους

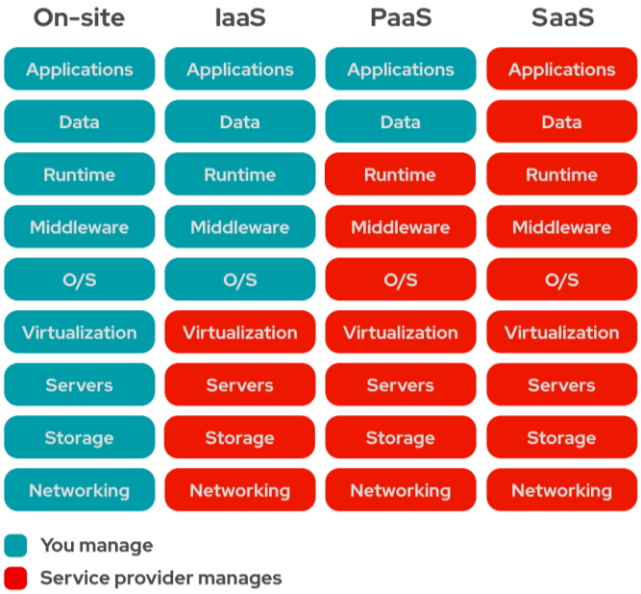
Εικόνα 1.4 Τυπική διασύνδεση συσκευών με χρήση νεφοϋπολογιστικής[1]

Ως νέφος ή Cloud στην πληροφορική, εννοείται μία τεχνολογία η οποία χωροταξικά δεν βρίσκεται στον ίδιο χώρο με τον χρήστη αλλά σε κάποιο υπολογιστικό σύστημα απομακρυσμένα από αυτόν και συνήθως η πρόσβαση σε αυτό γίνεται μέσω του διαδικτύου. Παλαιότερα, μία εταιρεία που ήθελε να κεντρικοποιήσει τις βάσεις δεδομένων της ή να αναπτύξει κάποια εφαρμογή, έπρεπε να επενδύσει σε απόκτηση server. Ένας server σαν υπολογιστικό σύστημα, πρέπει να είναι πάντα ενεργός, να έχει πολύ υψηλές δυνατότητες και επεξεργαστική ισχύ, θα πρέπει συνεχώς να αναβαθμίζετε και να συντηρείτε. Πλέον με την χρήση του Cloud computing, όλες τις παραπάνω εργασίες τις αναλαμβάνει η εταιρεία που της ανήκουν αυτές οι υποδομές. Μειώνεται έτσι, σημαντικά το κόστος για τον τελικό χρήστη και απαιτεί λιγότερη ενασχόληση από την ενοικιάστρια εταιρία. Επίσης, προσφέρει ευελιξία καθώς είναι προσβάσιμη από παντού, διασφάλιση των δεδομένων μας και αποθηκευτικό χώρο. Υπάρχει βέβαια και ένα πολύ σημαντικό ζήτημα ιδιωτικότητας των δεδομένων των χρηστών, καθώς διατηρούνται σε ένα online χώρο δεδομένων και αν προκύψει κάποιο κενό ασφαλείας, μπορεί να βρεθούν σε λάθος «χέρια». Αν και οι εταιρείες, που προσφέρουν υπηρεσίες cloud, εγγυώνται για την ασφάλεια των δεδομένων, η αποθήκευση κρίσιμων πληροφοριών καλό είναι να γίνεται σε κάποιο προσωπικό αρχείο τοπικά.[23]–[25]

Όταν ο χρήστης, κάνει εγγραφή σε μία υπολογιστική πλατφόρμα cloud, καλείτε να επιλέξει τις δυνατότητες και τους διαθέσιμους πόρους του εικονικού μηχανήματος, βάση των απαιτήσεων των υπηρεσιών που θα χρησιμοποιήσει. Μερικοί από τους πιο γνωστούς παρόχους νεφοϋπολογιστικών συστημάτων και υπηρεσιών, είναι η Amazon Web Services, η Microsoft Azure, το Google Cloud Platform και η Oracle. Οι πιο συνηθισμένες υπηρεσίες, που προσφέρονται μέσω cloud είναι η αποθήκευση δεδομένων, η διαχείριση βάσεων δεδομένων και η επεξεργασία και ανάλυση δεδομένων. Οι cloud υπηρεσίες, διαχωρίζονται σε τρία μοντέλα, τα οποία

[1] Πηγή: https://www.pngkey.com/download/u2q8y3w7i1o0r5w7_network-png-cloud-computing/ .

ορίζουν και το επίπεδο διαχείρισης των υποδομών τους από τον χρήστη. Τα τρία μοντέλα υπηρεσιών είναι το IaaS, το PaaS και το SaaS. Στην παρακάτω εικόνα (Εικόνα 1.5), βλέπουμε σε ποια επίπεδα έχει πρόσβαση ο χρήστης και ποια επίπεδα αναλαμβάνει ο πάροχος της υπηρεσίας.



Εικόνα 1.5 Μοντέλα υπηρεσιών νεφοϋπολογιστικής [2]

- **Λογισμικό ως υπηρεσία - Software as a service (SaaS)**

Μία υπηρεσία SaaS, είναι μία ολοκληρωμένη εφαρμογή, που εκτελείται στο Cloud κάποιου άλλου παρόχου και προσφέρεται στον τελικό χρήστη χωρίς να απαιτείται περαιτέρω εγκατάσταση από αυτόν. Κάποια παραδείγματα τέτοιων υπηρεσιών είναι το Gmail , το CRM ,το Google Docs, το Dropbox και άλλου είδους δικτυακές εφαρμογές.[26]

- **Πλατφόρμα ως υπηρεσία - Platform as a service (PaaS)**

Ως PaaS νοείται μία υπηρεσία cloud, που προσφέρει στον χρήστη μία πλατφόρμα πάνω στην οποία μπορεί να αναπτύξει, να διαχειριστεί και να εκτελέσει μία εφαρμογή με απομακρυσμένη μέθοδο. Είναι στην ουσία, η πλατφόρμα πάνω στην οποία εκτελούνται οι εφαρμογές, απαλλαγμένη από την διαχείριση του υλικού συστήματος και του δικτύου. Τέτοιου είδους πλατφόρμες είναι: το Home Assistant, το Node Red, το Google App Engine και το Microsoft Azure IoT.

- **Υποδομή ως υπηρεσία - Infrastructure as a service (IaaS)**

Μία IaaS παρέχει ένα υπολογιστικό σύστημα στο οποίο ο χρήστης, μπορεί να χρησιμοποιήσει τους πόρους του για να αναπτύξει ή να εκτελέσει όποια εφαρμογή θέλει, μέσω εικονοποίησης (virtualization). Μπορεί δηλαδή, να μοιράσει τους διαθέσιμους πόρους του υπολογιστικού συστήματος σε επιμέρους εικονικά συστήματα, το κάθε ένα να έχει διαφορετικό λειτουργικό σύστημα και εφαρμογές. Στην ουσία προσφέρει, την

[2] Πηγή: "IaaS vs PaaS vs SaaS," Red hat , Apr. 2020. https://www.redhat.com/en/topics/cloud-computing/iaas-vs-paas-vs-saas?sc_cid=7013a000002pgRjAAI&gclid=CjwKCAiA9aKQBhBREiwAyGP5lfXisIqefGyH8_mM2G929egL2qRjnP7QE_KEwj0Zeat6SXkbFTTJLhoCtAIQAvD_BwE&gclsrc=aw.ds (accessed Feb. 13, 2022) .

υποδομή για την εκτέλεση των PaaS. Κάποιες πολύ γνωστές εταιρείες που προσφέρουν είδους υπηρεσία είναι η Amazon Web Services, το Google Cloud Platform, και η Microsoft Azure.

ΚΕΦΑΛΑΙΟ 2: Σχεδίαση Εφαρμογής

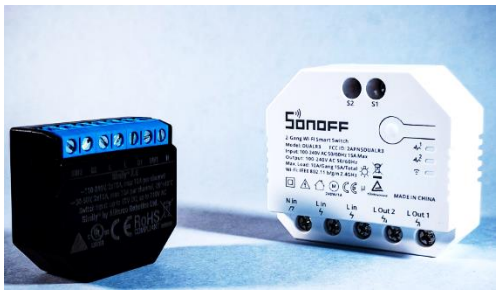
Εικόνα 2.1 Ηλεκτρολογικό Σχέδιο Οικίας

Η υλοποίηση της εφαρμογής, περιλαμβάνει κατασκευαστικά δύο προκλήσεις τον έλεγχο των ενεργοβόρων συσκευών καθώς και την αποστολή δεδομένων λειτουργίας και περιβάλλοντος από διάφορα αισθητήρια. Οι συσκευές που χρησιμοποιήθηκαν, έπρεπε να χρησιμοποιούν τεχνολογία WiFi για την επικοινωνία με τον διαδίκτυο, καθώς είναι διαθέσιμη σχεδόν σε όλα τα νοικοκυριά. Επίσης, η δυνατότητα επικοινωνίας με το πρωτόκολλο επικοινωνίας MQTT ήταν ένα ακόμη κριτήριο επιλογής της κατάλληλης συσκευής. Για την καταμέτρηση της ηλεκτρικής ενέργειας και τον έλεγχο λειτουργίας των περισσότερων συσκευών επιλέχθηκαν οι έτοιμες λύσεις της εταιρίας Shelly, οι οποίες ενσωματώνονται είτε πάνω στην τροφοδοσία των συσκευών είτε πίσω από το κέλυφος των επιτοίχιων ρευματοδοτών της οικίας. Υπάρχουν και άλλες εταιρίες που έχουν αναπτύξει παρόμοιες συσκευές, όπως η Sonoff. Αν και κοστίζουν λιγότερο (SONOFF DUALR3: ~14€, Shelly 2.5 ~20€), υστερούν σε προστασίες υπερφόρτωσης, το λογισμικό τους έχει εμφανίσει κενά ασφαλείας, χρειάζονται να επαναπρογραμματιστούν με άλλο firmware για λειτουργίες επικοινωνίας που στα Shelly είναι ήδη διαθέσιμες και είναι αρκετά ογκώδεις για να εγκατασταθούν χωρίς να επηρεάζουν την αισθητική του χώρου.

Dimension
114x52x33mm
Voltage
100-240V AC 50/60Hz
Wi-Fi
IEEE 802.11 b/g/n
Max. Load
15A/3500W

Dimension
41x36x15mm
Voltage
110-230V ±10%, 50/60Hz
Wi-Fi
IEEE 802.11 b/g/n
Max Load
16A/~3600W

α)



Shelly 2.5	**Sonoff DualR3**
ESP8266	ESP32
110-230V ±10%, 50/60Hz AC	100-240V ±10%, 50/60Hz AC
24 – 60V DC	no
2 Channel 10A per channel	2 Channel 10A per channel (15A Max)
WiFi 802.11 b/g/n	WiFi 802.11 b/g/n
39mm x 36mm x 17 mm	54mm x 49mm x 24 mm

β)

Εικόνα 2.2 Σύγκριση βασικών χαρακτηριστικών α) Sonoff POW R2 - Shelly 1PM και β) Shelly 2.5 -Sonoff R3

Στο παρόν οικιακό περιβάλλον διατίθενται δύο κλιματιστικά, ένα στο σαλόνι και ένα στο υπνοδωμάτιο, τα οποία ελέγχονται με τηλεχειριστήρια εκπομπής υπέρυθρων σημάτων. Οι προδιαγραφές μίας τέτοιας συσκευής ήταν να ενσωματώνει πομπό υπέρυθρων σημάτων, αισθητήρα περιβάλλοντος για έλεγχο των συνθηκών του δωματίου στο οποίο θα είναι εγκατεστημένο, δυνατότητα διασύνδεσης με το WiFi, να είναι συμβατό με το MQTT πρωτόκολλο και να έχει προοπτικές επεκτασιμότητας με επιπλέον αισθητήρες. Αφού έγινε έρευνα αγοράς, προέκυψε ότι δεν υπήρχε κάποια έτοιμη λύση σε προσιτή τιμή, η οποία να προσφέρει όλα τα παραπάνω ταυτόχρονα. Η υλοποίηση μίας πρωτότυπης κατασκευής, που βασίζεται σε ένα μικροελεγκτή ESP8266, με ασύρματη διασύνδεση και που ενσωματώνει όλα τα παραπάνω, ήταν μία αρκετά συμφέρουσα επιλογή. Για την παραπάνω υλοποίηση, απαραίτητο ήταν να γίνει πρώτα η αποκωδικοποίηση των υπέρυθρων σημάτων από τα αρχικά τηλεχειριστήρια. Για τον λόγο αυτό, έγινε και μία δευτερεύουσα βοηθητική υλοποίηση ενός αποκωδικοποιητή υπέρυθρων σημάτων που βασίστηκε σε ένα μικροελεγκτή ESP32 και ένα δέκτη υπέρυθρων σημάτων .

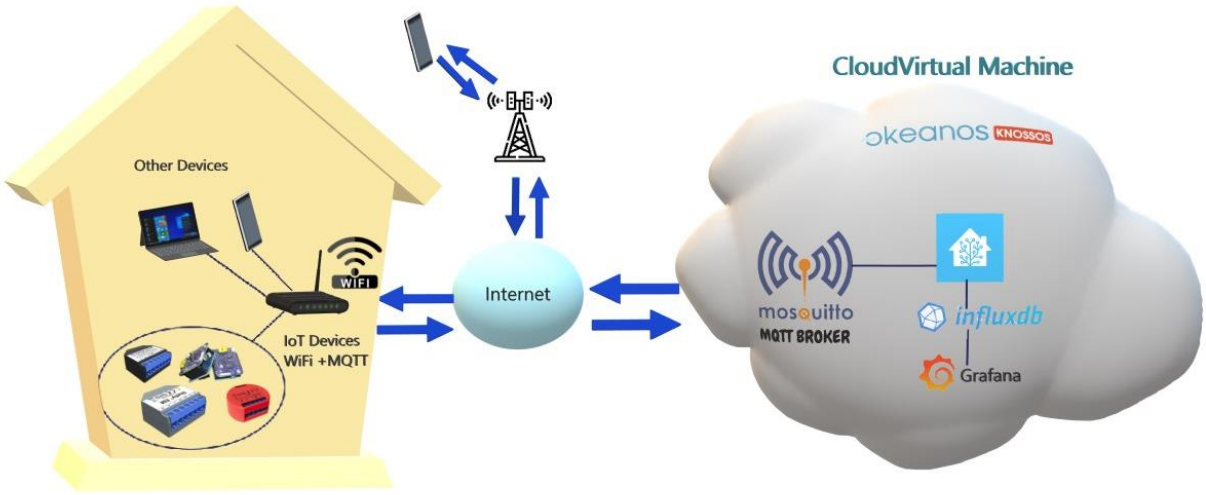
Η χρήση πολλών και διαφορετικών υλοποιήσεων, σε έναν δίκτυο ή η χρήση συσκευών IoT διαφορετικών κατασκευαστών, κάνει δύσκολο τον απομακρυσμένο έλεγχο. Η κάθε υλοποίηση μπορεί να επικοινωνεί με διαφορετικό τρόπο με τον τελικό χρήστη, ακόμα και να χρησιμοποιεί διαφορετικό πρωτόκολλο επικοινωνίας με το διαδίκτυο. Για να επιτευχθεί, η συγκεντρωτική προβολή της κατάστασης λειτουργίας του κάθε συστήματος ή συσκευής, καθώς και η συγκεντρωτική παρακολούθηση της κατανάλωσης ενέργειας, έπρεπε να επιλεχθεί μία πλατφόρμα που θα έχει την δυνατότητα να τα συνδυάζει όλα αυτά, να τα απεικονίζει και να είναι παραμετροποιήσιμη. Σε αυτή την υλοποίηση, χρησιμοποιήθηκε το Home Assistant ως πλατφόρμα ελέγχου και απεικόνισης της κατάστασης λειτουργίας, όλων των επιμέρους συσκευών του δικτύου. Η εγκατάσταση του Home Assistant, όπως και όλων των επιμέρους υπηρεσιών θα γίνει σε μία υπολογιστική πλατφόρμα νέφους,

έτσι ώστε να είναι δυνατή η απομακρυσμένη διαχείριση ενώ το σύστημα είναι ενεργό αδιάλειπτα. Η μετατροπή των κλασσικών ηλεκτρικών συσκευών σε έξυπνες, έγινε τοποθετώντας τις προαναφερθείσες υλοποιήσεις, στις συσκευές της Εικόνα 2.3 που έχουν επιλεγεί με κόκκινο περίγραμμα.

Εικόνα 2.3 Ηλεκτρολογικό σχέδιο - Με επιλεγμένες συσκευές προς έλεγχο.

Πιο συγκεκριμένα:

- Για τα δύο κλιματιστικά (Air Condition) του υπνοδωματίου και του σαλονιού, χρησιμοποιήθηκαν οι μικροελεγκτές WeMos D1 mini pro με ενσωματωμένο πομπό IR και αισθητήρα περιβάλλοντος.
- Ένα επιπλέον WeMos D1 mini pro, χρησιμοποιήθηκε για καταγραφή των περιβαλλοντικών συνθηκών, στο δωμάτιο του γραφείου, μόνο με αισθητήρα θερμοκρασίας.
- Στην κεντρική παροχή ηλεκτρικής ενέργειας για την καταγραφή της συνολικής κατανάλωσης ενέργειας χρησιμοποιήθηκε το Shelly EM με αμπεροτσιμπίδα 50A .
- Για τους θερμοπομπούς και το θερμοσίφωνο χρησιμοποιήθηκαν τα Shelly 1PM . Συγκεκριμένα στο θερμοσίφωνο, προστέθηκε ρελέ ράγας που ελέγχετε από το Shelly 1PM για λόγους ασφαλείας. Οι απαιτήσεις ενέργειας του θερμοσίφωνου είναι στα 3500W και πλησιάζει το όριο λειτουργίας του Shelly, το οποίο μπορεί να οδηγήσει σε αύξηση της θερμοκρασίας και διακοπή λειτουργίας.
- Για τα εξωτερικά φώτα χρησιμοποιήθηκε το Shelly 2.5 το οποίο διαθέτει επιλογή διαχείρισης 2 καναλιών εξόδων.



Εικόνα 2.4 Δίκτυο Διαχείρισης ενέργειας και μέτρησης ενεργειακής κατανάλωσης με τεχνολογίες IoT.

Στη συνέχεια και αφού ολοκληρωθήκαν όλες οι παραπάνω εργασίες και η εγκατάσταση των απαραίτητων εφαρμογών θα γίνει παραμετροποίηση του Home Assistant προσθέτοντας κεντρικό έλεγχο, καταγραφή ιστορικού των λειτουργιών και των καταναλώσεων στην βάση δεδομένων InfluxDB και αυτοματισμούς που αφορούν την λειτουργία των έξυπνων πλέον συσκευών. Θα ακολουθήσει η οπτικοποίηση του ιστορικού μέσω της πλατφόρμας προβολής στατιστικών δεδομένων Grafana η οποία επιτρέπει την κατ' απαίτηση αλλαγή των διαγραμμάτων σε πραγματικό χρόνο. Για να αποδειχθεί η επίτευξη η όχι της εξοικονόμησης ηλεκτρικής ενέργειας αλλά και η επιτυχής λειτουργία του δικτύου και των αυτοματισμών θα ληφθούν δειγματοληπτικά κάποια δεδομένα χρήσης, των συσκευών και θα αναλυθούν και θα συγκριθούν ως προς την αποδοτικότητα τους.

2.1 Cloud Virtual Machine (~Okeanos-Knossos, Cyclades)

Το Okeanos είναι μία υπηρεσία νεφοϋπολογιστικής (IaaS), που προσφέρει την υποδομή για ανάπτυξη «εικονικών μηχανών» και «εικονικών δικτύων». Πρόκειται για μία πλατφόρμα, η οποία δημιουργήθηκε από την GRNET για να καλύπτει ανάγκες υποδομής υπολογιστικών συστημάτων, από μέλη της ακαδημαϊκής κοινότητας (καθηγητές , μαθητές, ερευνητές). Το Cyclades είναι η υπηρεσία του ~Okeanos-knossos που αφορά της υπηρεσίες εικονικών υπολογιστικών συστημάτων και δικτύων. Στα virtual machines που μπορούν να δημιουργηθούν στο Cyclades υπάρχουν οι εξής δυνατότητες:

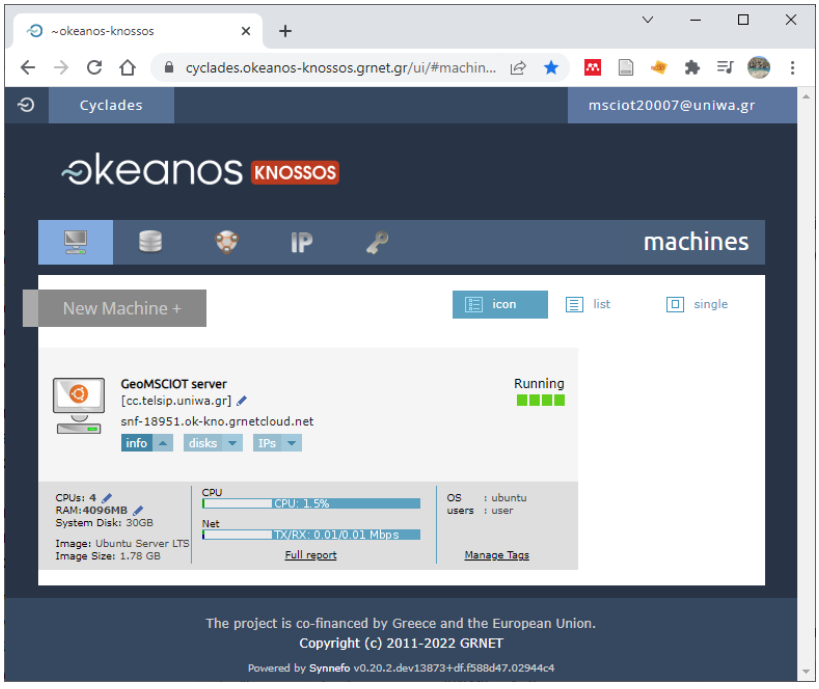
- Επιλογής διάφορων λειτουργικών συστημάτων
- Επιλογής αριθμού πυρήνων της CPU και RAM ανά VM
- Συνεχούς σύνδεση με το διαδίκτυο
- Δημιουργίας ιδιωτικών δικτύων
- Απομακρυσμένης διαχείρισης από οποιοδήποτε σελιδοποιητή (browser)
- Επιλογής τριών ειδών τειχών προστασίας (Firewall)
- Εισαγωγής αρχείων
- RESTful API για εύκολη διαχείριση μέσω γραφικού περιβάλλοντος.

Όσον αφορά την απομακρυσμένη διαχείριση η πλατφόρμα του Cyclades προσφέρει ένα γραφικό περιβάλλον στο οποίο φαίνεται η κατάσταση και τα στοιχεία των εικονικών μηχανημάτων που έχουν δημιουργηθεί. Επίσης, ο χρήστης έχει δυνατότητα αλλαγής των υπολογιστικών δυνατοτήτων σε κάθε VM που διαθέτει καθώς

και να διαχειριστεί την λειτουργία τους (π.χ. reboot, shutdown, resize). Αφού δημιουργηθεί το VM, υπάρχει δυνατότητα επικοινωνίας με το μηχάνημα μέσω του πρωτοκόλλου SSH. Η σύνδεση με SSH, είναι μία κοινή μέθοδος κρυπτογραφημένης διασύνδεσης σε απομακρυσμένα μηχανήματα. Μέσω μίας τερματικής κονσόλας των windows εισάγετε η εντολή : ssh <username>@<IP του Virtual Machine>. Στην υλοποίηση αυτή η εντολή θα δοθεί σε τερματικό των Windows, όπως φαίνεται στην Εικόνα 2.5 και αφού δοθεί και το αντίστοιχο password μας εμφανίζει εισαγωγικές πληροφορίες για το λειτουργικό σύστημα και το δίκτυο του μηχανήματος.

```
C:\Users\geopi>ssh user@83.212.77.18
user@83.212.77.18's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-90-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu 17 Feb 2022 02:04:45 AM EET

  System load:              0.17
  Usage of /:               59.7% of 29.40GB
  Memory usage:             33%
  Swap usage:               0%
  Processes:                206
  Users logged in:          0
  IPv4 address for docker0: 172.17.0.1
  IPv6 address for eth0:    2001:648:2ffe:501:cc00:12ff:fe3d:840a
  IPv4 address for eth1:    83.212.77.18

8 updates can be applied immediately.
8 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


Last login: Thu Feb 17 01:17:44 2022 from 85.73.234.152
user@snf-18951:~$
```

Εικόνα 2.5 Πρόσβαση στο VM από τερματικό με SSH



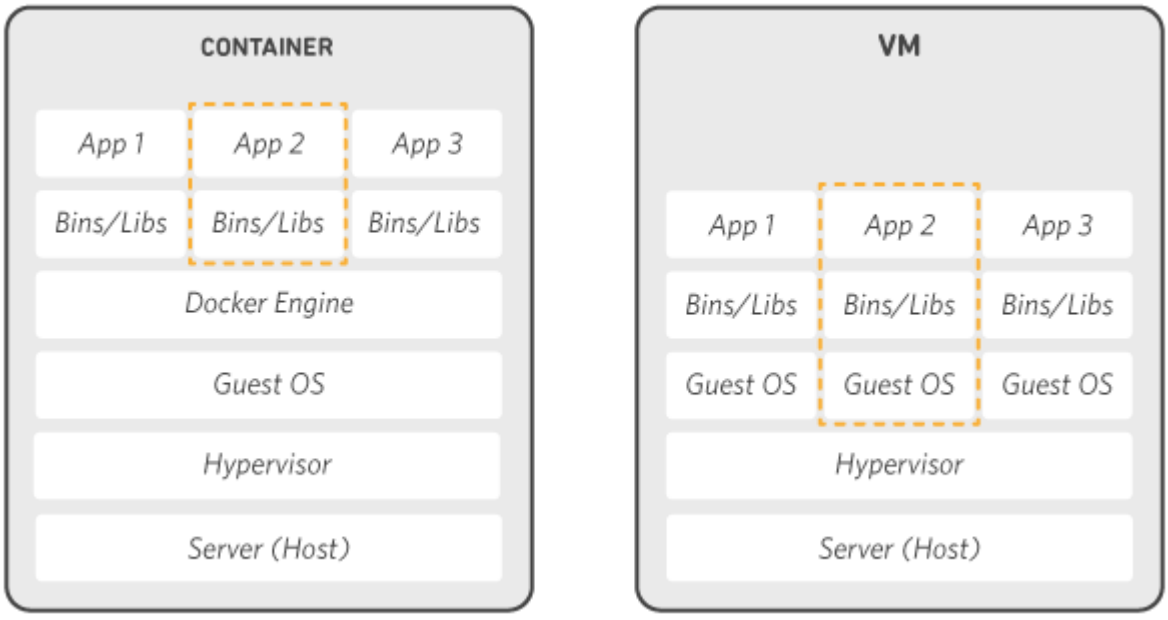
Εικόνα 2.6 Πίνακας ελέγχου Εικονικού μηχανήματος στον ~Okeanos-Knossos

Στην Εικόνα 2.6, φαίνεται ο πίνακας ελέγχου της εικονικής μηχανής που διέθεσε το Πανεπιστήμιο Δυτικής Αττικής, για την υλοποίηση της παρούσας διπλωματικής. Τα χαρακτηριστικά του είναι τα εξής:

- Επεξεργαστής τεσσάρων πυρήνων
- 4Gb μνήμη RAM
- Χωρητικότητα σκληρού δίσκου 30Gb
- Λειτουργικό σύστημα Ubuntu 20.04.3 LTS
- Διεύθυνση δικτύου: **83.212.77.18** ή **snf-18951.ok-kno.grnetcloud.net**
- Δυνατότητα απομακρυσμένης διαχείρισης μέσω πρωτοκόλλου SSH

2.1.1 Εγκατάσταση Docker

Το docker, είναι μία τεχνολογία που χρησιμοποιεί το kernel του Linux και επιτρέπει την εικονοποίηση των εφαρμογών, που βρίσκονται εντός ενός container. Η ανάπτυξη εφαρμογών και η χρήση πολλών διαφορετικών αρχιτεκτονικών και γλωσσών προγραμματισμού έχουν ως αποτέλεσμα την έντονη πολυπλοκότητα και προβλήματα συμβατότητας. Με την χρήση του docker, επιτρέπεται η ταχύτατη δημιουργία, δοκιμή και εκτέλεση μίας οποιαδήποτε εφαρμογής. Τα containers, που αναφέρθηκαν προηγουμένως είναι στην ουσία κάποια πακέτα τα οποία περιλαμβάνουν ό,τι είναι απαραίτητο για την εκτέλεση μίας εφαρμογής [27]. Εγκαθιστώντας το Docker, επιτρέπει στο χρήστη την εγκατάσταση επιμέρους εφαρμογών ανεξάρτητα από το λειτουργικό σύστημα του Host υπολογιστικού συστήματος, εικονοποιώντας το περιβάλλον στο οποίο θα εκτελεστεί η εκάστοτε εφαρμογή. Η διαφορά της αρχιτεκτονικής του Docker με την αρχιτεκτονική ενός Virtual Machine φαίνονται στην Εικόνα 2.7:



Εικόνα 2.7 Αρχιτεκτονική Docker- Αρχιτεκτονική VM[3]

Στην υλοποίηση αυτή, έγινε η εγκατάσταση του Docker στο Virtual Machine του ~Okeanos-knossos καθώς ήταν πολύ χρήσιμη η δυνατότητα εκτέλεσης των εφαρμογής σε δοκιμαστικό περιβάλλον συμβατό με τα περισσότερα λειτουργικά συστήματα. Το Home Assistant, εγκαταστάθηκε με την μέθοδο των Container στο VM του ~Okeanos-knossos. Η μέθοδος αυτή, περιλαμβάνει την λήψη η δημιουργία ενός image αρχείου και την

[3] Πηγή: What is Docker? | AWS.” https://aws.amazon.com/docker/ (accessed Feb. 17, 2022).

εκτέλεση του ως Container. Αν πρόκειται, για εφαρμογή υπό ανάπτυξη μπορεί να «ανεβεί» και στον προσωπικό λογαριασμό του προγραμματιστή στο «Docker Hub».

Ένα docker Container, κατά την δημιουργία του μπορεί να λάβει κάποιες παραμετροποιήσεις, που μπορεί να αφορούν τον τρόπο διασύνδεσης του με το δίκτυο, την δικτυακή πόρτα επικοινωνίας, κάποιο χώρο (volume) στο host μηχάνημα στον οποίο θα αποθηκεύει δεδομένα απαραίτητα για την διατήρηση των ρυθμίσεων της εφαρμογής και ένα όνομα . Για την εγκατάσταση του docker εκτελέστηκαν τα παρακάτω βήματα:

- **Εγκατάσταση Docker με εντολές στο τερματικό του Virtual Machine**

1. Αρχικά, πρέπει δημιουργηθεί ένας λογαριασμός στο site του https://hub.docker.com/ έτσι ώστε να υπάρχει η δυνατότητα λήψης η προώθησης προς το docker hub εφαρμογών υπό ανάπτυξη
2. Πραγματοποιείται σύνδεση στο VM μέσω ssh όπως περιγράφηκε στην ενότητα 2.1
3. Γίνεται η εγκατάσταση τυχών ενημερώσεων για τις υπάρχουσες εφαρμογές του VM

```
$ sudo apt update
```

4. Γίνεται εγκατάσταση των απαιτούμενων αρχείων που επιτρέπουν την εκτέλεση της εντολή apt σε πρωτόκολλο HTTPS

```
$ sudo apt install apt-transport-https ca-certificates curl
software-properties-common

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg |
sudo apt-key add
```

5. Προσθήκη του Docker repository στις πηγές του APT
```
$ sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu focal stable"
```
6. Εγκατάσταση του Docker
```
$ sudo apt install docker-ce
```
7. Μετά την ολοκλήρωση της εγκατάστασης ενεργοποιούμε το daemon ελέγχετε η κατάσταση λειτουργίας του η οποία θα πρέπει να είναι ενεργή (active)
```
$ sudo systemctl status docker
```

Αν η έξοδος είναι όπως στην Εικόνα 2.8 το docker είναι έτοιμο προς χρήση και ενεργό.

Έξοδος στην οθόνη:

```
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2022-02-17 01:12:54 EET; 20h ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 1305 (dockerd)
      Tasks: 78
     Memory: 115.2M
     CGroup: /system.slice/docker.service
             ├─1305 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
             ├─2003 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 9001 -container-ip 172.17.0.2 -container-port 9001
             ├─2040 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 9001 -container-ip 172.17.0.2 -container-port 9001
             ├─2054 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 9000 -container-ip 172.17.0.3 -container-port 9000
             ├─2061 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 9000 -container-ip 172.17.0.3 -container-port 9000
             ├─2128 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 8123 -container-ip 172.17.0.4 -container-port 8123
             ├─2137 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 8123 -container-ip 172.17.0.4 -container-port 8123
             ├─2158 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 1880 -container-ip 172.17.0.5 -container-port 1880
             └─2165 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 1880 -container-ip 172.17.0.5 -container-port 1880
```

Εικόνα 2.8 Κατάσταση λειτουργίας Docker

2.1.2 Εγκατάσταση Portainer

Για την διαχείριση των containers, συνηθίζεται να χρησιμοποιούνται εντολές από το τερματικό. Η χρήση όμως πολλαπλών containers κάνει δύσκολη την διαχείριση τους μέσω εντολών, όχι όμως αδύνατη. Το Portainer είναι ένα PaaS το οποίο προσφέρει στον τελικό χρήστη ένα γραφικό περιβάλλον μέσω browser μέσα από το οποίο μπορεί να διαχειριστεί, να εγκαταστήσει η και να διαγράψει Containers και image files. Η εγκατάσταση του Portainer έγινε με χρήση του docker και αποτελείται από τον Portainer Server και το Portainer Agent. Τα δύο αυτά στοιχεία θα συνεργαστούν κάτω από το ίδιο docker engine. Για την εγκατάσταση και παραμετροποίηση του ακολουθούνται τα εξής βήματα:

1. Γίνεται λήψη του image αρχείου για τον server

Έξοδος στην οθόνη:

```
user@snf-18951:~$ docker pull portainer/portainer-ce:latest
latest: Pulling from portainer/portainer-ce
772227786281: Pull complete
96fd13befc87: Pull complete
b3238bddfe78: Pull complete
Digest: sha256:3ff080a0cd2a45bd0bde046069973b3fe642c3e4d43c5b429dd7b77f0057c7d7
Status: Downloaded newer image for portainer/portainer-ce:latest
docker.io/portainer/portainer-ce:latest
```

2. Στη συνέχεια δημιουργείται και ενεργοποιείτε το αντίστοιχο container

Έξοδος στην οθόνη:

```
user@snf-18951:~$ docker run -d -p 8000:8000 -p 9000:9000 -p 9443:9443     --name=portainer --
restart=always     -v /var/run/docker.sock:/var/run/docker.sock     -v portainer_data:/data
  portainer/portainer-ce:latest
2358e6b856ab1b16d1ca55ab26e1a5d4f5bef27e34c32a3967957d6214ff7fa5
```

3. Γίνεται λήψη του image αρχείου για τον agent

Έξοδος στην οθόνη:

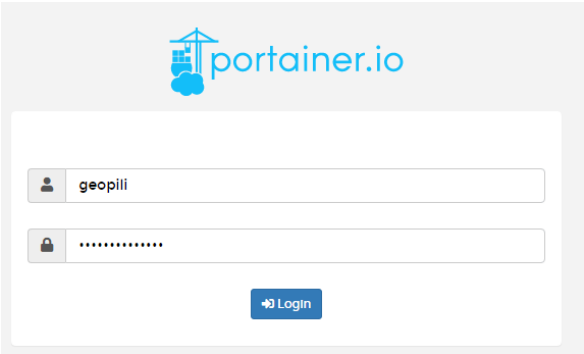
```
user@snf-18951:~$ docker pull portainer/agent:latest
latest: Pulling from portainer/agent
772227786281: Already exists
96fd13befc87: Already exists
77e18c30f701: Pull complete
ec203a08fb24: Pull complete
ebb3f643c085: Pull complete
Digest: sha256:90d8d396d3ac83322dc95b549d2652d69601bcc2e0aee69736529217145bf1e4
Status: Downloaded newer image for portainer/agent:latest
docker.io/portainer/agent:latest
```

4. Δημιουργείται και ενεργοποιείται το αντίστοιχο container

Έξοδος στην οθόνη:

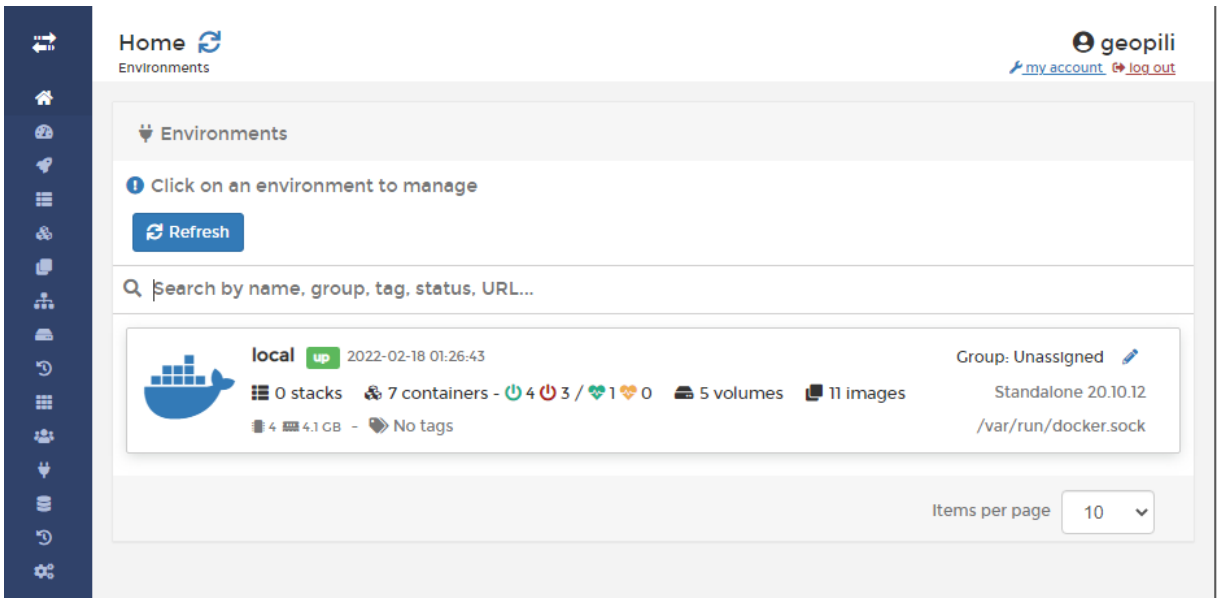
```
user@snf-18951:~$ docker run -d -p 9001:9001 --name portainer_agent --restart=always -v /var/r
un/docker.sock:/var/run/docker.sock -v /var/lib/docker/volumes:/var/lib/docker/volumes portain
er/agent:latest
2038338e581e5e6f88f8b5be2173163ae7d3519bc9b5d58230498dac45f4dbf5
```

Το Portainer είναι πλέον διαθέσιμο στην διεύθυνση: http://83.212.77.18:9000. Το γραφικό του περιβάλλον είναι το παρακάτω:



Εικόνα 2.9 Καρτέλα ταυτοποίησης χρήστη

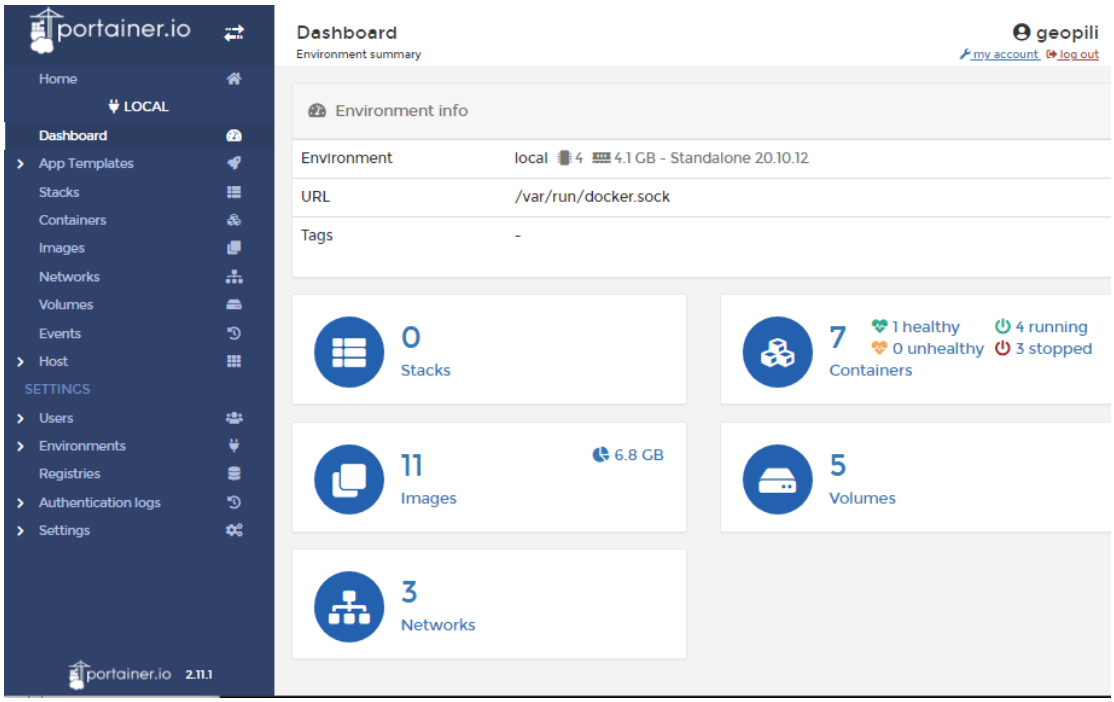
5. Αρχικά, θα ζητηθεί η δημιουργία λογαριασμού για ασφαλή πρόσβαση ή εισαγωγή κάποιου υπάρχοντος λογαριασμού.
6. Εμφανίζονται τα διαθέσιμα docker engines



Εικόνα 2.10 Καρτέλα διαθέσιμων Docker engine

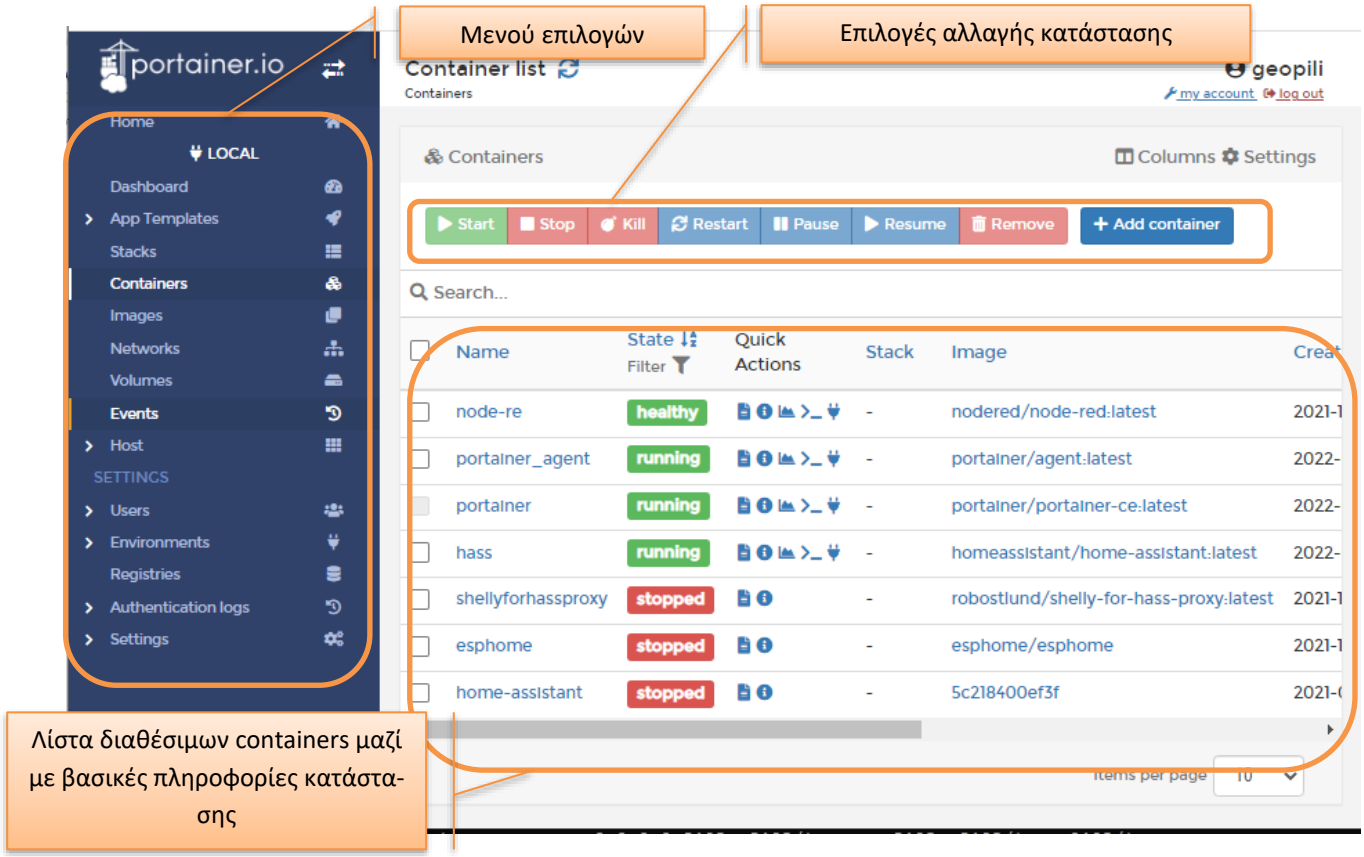
7. Αν επιλεχθεί το υπάρχον , τότε ανοίγει τον «πίνακα ελέγχου» (Dashboard)(Εικόνα 2.11) που περιλαμβάνει όχι μόνο τα διαθέσιμα docker containers αλλά και τα image αρχεία που υπάρχουν στο host

μηχάνημα, τα διαθέσιμα δίκτυα και τα volumes που έχουν δημιουργηθεί από χρήση άλλων containers.



Εικόνα 2.11 Πίνακας ελέγχου (Dashboard) Portainer

8. Επιλέγοντας τα Containers, δίνεται μία πλήρης λίστα με τα ενεργά και μη Containers που υπάρχουν εγκατεστημένα στο host μηχάνημα. Σε αυτό το σημείο, δίνονται επιπλέον λειτουργίες στον χρήστη που έχουν να κάνουν με την διαχείριση των containers.



Εικόνα 2.12 Γραφικό Περιβάλλον Portainer

9. Τα στοιχεία που δίνονται από το Portainer σε αυτό το μενού (Εικόνα 2.12), είναι λίστα των διαθέσιμων containers μαζί με την κατάσταση λειτουργίας (Running (εκτελείται αυτή τη στιγμή), Stopped (Σταματημένο), Healthy (υγιές) και μέσω της γραμμής εργαλείων υπάρχουν επιλογές διαχείρισης των containers (Start, Stop, Kill, Restart κτλ).

2.1.3 Εγκατάσταση Mosquitto MQTT broker

Εικόνα 2.13 Mosquitto MQTT Broker

Ο MQTT broker που επιλέχθηκε είναι, ο ανοιχτού κώδικα Mosquitto broker. Πρόκειται, για ένα δωρεάν server διαχείρισης μηνυμάτων μέσω MQTT και εγκαταστάθηκε στο Virtual Machine από το τερματικό μέσω ssh επικοινωνίας. Με την χρήση αυτού του MQTT broker θα επιτευχθεί η επικοινωνία όλων των συσκευών που θα συμμετέχουν στο δίκτυο της παρούσας υλοποίησης. Για την εγκατάσταση του ακολουθήθηκαν τα παρακάτω βήματα:

1. Εκτελέσθηκε η παρακάτω εντολή για την λήψη του repository και την εγκατάσταση των απαραίτητων πακέτων

```
$ sudo apt install -y mosquitto mosquitto-clients
```

Έξοδος στην οθόνη:

```
Select user@snf-18951: ~
user@snf-18951:~$ sudo apt install -y mosquitto mosquitto-clients
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libdlt2 libev4 libmosquitto1 libwebsockets15
The following NEW packages will be installed:
  libdlt2 libev4 libmosquitto1 libwebsockets15 mosquitto mosquitto-clients
0 upgraded, 6 newly installed, 0 to remove and 153 not upgraded.
Need to get 498 kB of archives.
After this operation, 1,589 kB of additional disk space will be used.
Get:1 http://ftp.cc.uoc.gr/mirrors/linux/ubuntu/packages focal/universe amd64 libdlt2 amd64 2.18.4-0.1 [50.4 kB]
Get:2 http://ftp.cc.uoc.gr/mirrors/linux/ubuntu/packages focal/universe amd64 libmosquitto1 amd64 1.6.9-1 [45.9 kB]
Get:3 http://ftp.cc.uoc.gr/mirrors/linux/ubuntu/packages focal/universe amd64 libev4 amd64 1:4.31-1 [31.2 kB]
Get:4 http://ftp.cc.uoc.gr/mirrors/linux/ubuntu/packages focal/universe amd64 libwebsockets15 amd64 3.2.1-3 [152 kB]
Get:5 http://ftp.cc.uoc.gr/mirrors/linux/ubuntu/packages focal/universe amd64 mosquitto amd64 1.6.9-1 [160 kB]
Get:6 http://ftp.cc.uoc.gr/mirrors/linux/ubuntu/packages focal/universe amd64 mosquitto-clients amd64 1.6.9-1 [58.8 kB]
Fetched 498 kB in 1s (523 kB/s)
Selecting previously unselected package libdlt2:amd64.
(Reading database ... 89919 files and directories currently installed.)
Preparing to unpack .../0-libdlt2_2.18.4-0.1_amd64.deb ...
Unpacking libdlt2:amd64 (2.18.4-0.1) ...
Selecting previously unselected package libmosquitto1:amd64.
Preparing to unpack .../1-libmosquitto1_1.6.9-1_amd64.deb ...
Unpacking libmosquitto1:amd64 (1.6.9-1) ...
Selecting previously unselected package libev4:amd64.
Preparing to unpack .../2-libev4_1%3a4.31-1_amd64.deb ...
Unpacking libev4:amd64 (1:4.31-1) ...
Selecting previously unselected package libwebsockets15:amd64.
Preparing to unpack .../3-libwebsockets15_3.2.1-3_amd64.deb ...
```

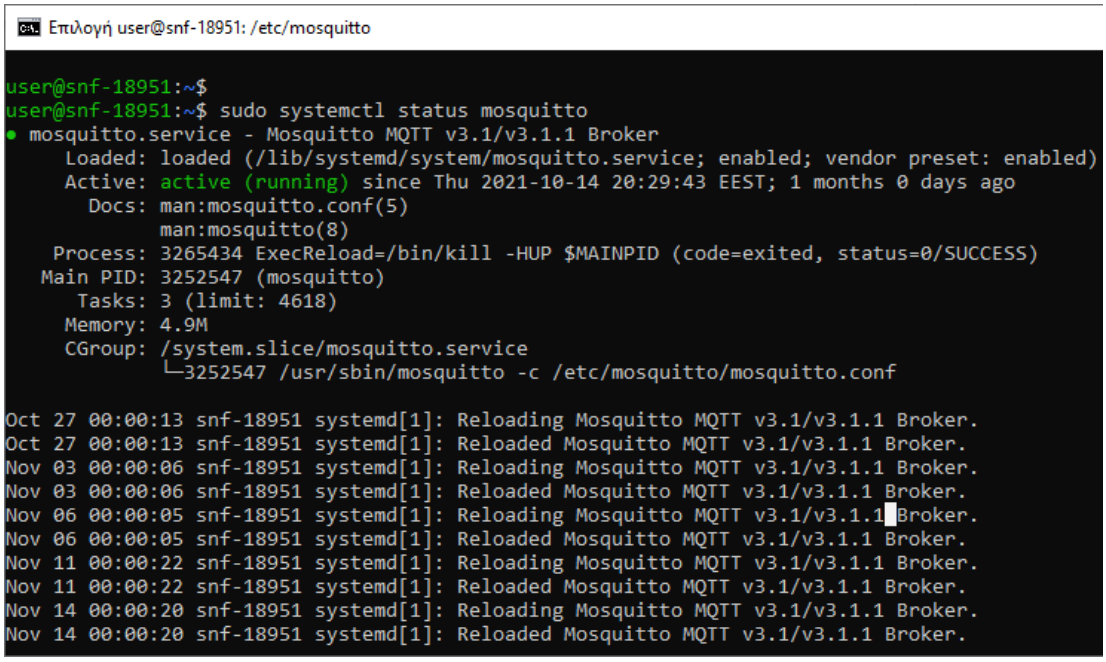
2. Εκτελέσθηκε η παρακάτω εντολή έτσι ώστε το Mosquitto να ενεργοποιείται κάθε φορά που επανεκκινείται το VM.

```
$ sudo systemctl enable mosquitto.service
```

3. Ελέγχθηκε η κατάσταση λειτουργίας του με την εντολή

```
$ sudo systemctl status mosquitto
```

Έξοδος στην οθόνη:



Από προεπιλογή ο MQTT broker δεν διαθέτει κάποιο είδους ασφάλειας πρόσβασης. Με τα παρακάτω βήματα θα γίνει επεξεργασία των ρυθμίσεων του, ώστε να χρειάζεται όνομα χρήστη και κωδικός για να γίνει χρήση του broker. Επίσης, θα προστεθεί ένα ακόμα επίπεδο ασφάλειας κρυπτογραφώντας τον κωδικό πρόσβασης. Τα βήματα επεξεργασίας των ρυθμίσεων είναι:

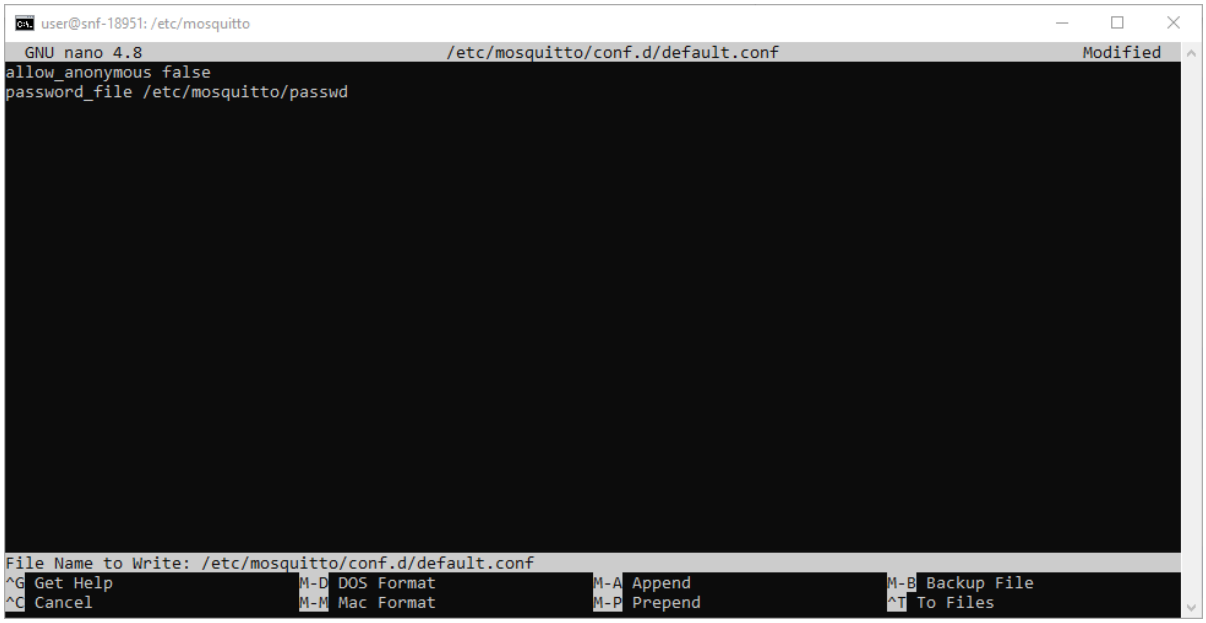
1. Η δημιουργία ενός αρχείου με όνομα default.conf στην διεύθυνση που το mosquitto αποθηκεύει τις ρυθμίσεις του «/etc/mosquitto/conf.d» με την παρακάτω εντολή:
   ```
   $ sudo nano /etc/mosquitto/conf.d/default.conf
   ```

2. Σε αυτό το αρχείο κειμένου εισάγονται τα παρακάτω πεδία ,

```
allow_anonymous false
password_file /etc/mosquitto/passwd
```

έτσι ώστε να μην επιτρέπονται οι ανώνυμες συνδέσεις και να γνωστοποιήσει στο mosquitto ότι το αρχείο που θα περιλαμβάνει τους κωδικούς πρόσβασης θα βρίσκεται σε αυτή την διεύθυνση «/etc/mosquitto/passwd».

Έξοδος στην οθόνη:

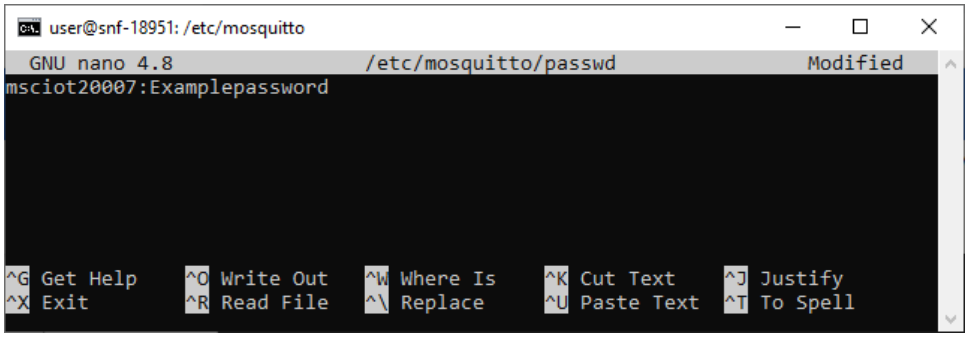


3. Στη συνέχεια, έγινε η αποθήκευση και το κλείσιμο του αρχείου.
4. Ανοίγεται το αρχείο passwd με το nano έτσι ώστε να γίνει επεξεργασία του περιεχόμενού του.

```
$ sudo nano /etc/mosquitto/passwd
```

5. Εισάγονται το όνομα χρήστη και ο κωδικός πρόσβασης που επιθυμεί ο χρήστης και γίνεται αποθήκευση του αρχείου.

Έξοδος στην οθόνη:



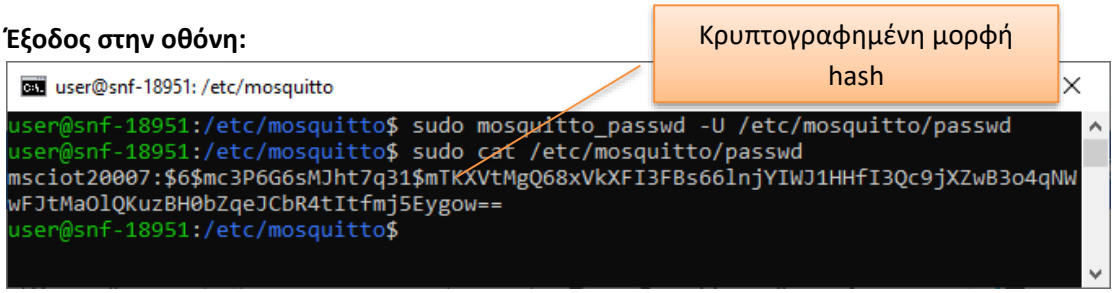
6. Στη συνέχεια, χρησιμοποιείται μία λειτουργία που διαθέτει το mosquitto για κρυπτογράφηση των αρχείων κειμένου που περιέχουν κωδικούς πρόσβασης. Αυτή λειτουργία μετατρέπει το κείμενο του κωδικού πρόσβασης με μία συνάρτηση κατακερματισμού (hash) σε μία μορφή που δεν μπορεί να αποκρυπτογραφηθεί από κανένα άλλο πέραν του mosquitto.

```
$ sudo mosquitto_passwd -U /etc/mosquitto/passwd
```

7. Επιβεβαιώνεται ότι έγινε η κρυπτογράφηση του κωδικού με την εντολή:

```
$ sudo cat /etc/mosquitto/passwd
```

Έξοδος στην οθόνη:



8. Εκτελείται επανεκκίνηση του mosquitto για την φόρτωση των νέων ρυθμίσεων

```
$ sudo systemctl restart mosquitto
```

Πλέον, ο Mosquitto MQTT broker είναι έτοιμος προς χρήση με όνομα χρήστη το **msciot20007**, κρυπτογραφημένο κωδικό πρόσβασης και διεύθυνση πρόσβασης αυτή του host Virtual Machine και την πόρτα 1883. Δηλαδή την: http://83.212.77.18:1883.

2.1.4 Εγκατάσταση Home Assistant (PaaS)

Εικόνα 2.14 Home Assistant

Το Home assistant, είναι μία ανοιχτού κώδικα πλατφόρμα διαχείρισης έξυπνων συσκευών σε οικιακό περιβάλλον. Προσφέρει δυνατότητες, διασύνδεσης IoT συσκευών και επιτρέπει την επικοινωνία μεταξύ τους ακόμα και αν χρησιμοποιούν διαφορετικά πρωτόκολλα επικοινωνίας. Το περιβάλλον και οι λειτουργίες του θα περιγραφθούν στο «ΚΕΦΑΛΑΙΟ 3:Το Home Assistant». Υπάρχουν πολλοί διαθέσιμοι τρόποι εγκατάστασης του Home assistant: α)ως λειτουργικό σύστημα (OS), β) ως Supervised γ) ως Container, , δ) Core. Η 1η μέθοδος (OS) δεσμεύει εξ ’ολοκλήρου το υπολογιστικό σύστημα προφέροντας πλήρης πρόσβαση στις δυνατότητες Supervisor και τα add-ons του home assistant, συνήθως επιλέγεται να εγκατασταθεί σε συστήματα με χαμηλές επιδόσεις. Η 2η μέθοδος ονομάζεται Supervised προσφέρει τις ίδιες δυνατότητες με την 1η μέθοδο αλλά η εγκατάσταση γίνεται σε λειτουργικό σύστημα Linux ως ξεχωριστή εφαρμογή. Η 3η μέθοδος η οποία και επιλέχθηκε εγκαθιστά το Home assistant ως container. Η μέθοδος αυτή θέτει κάποιους περιορισμούς καθώς δεν διαθέτει την επιλογή Supervisor και το μενού Add-ons, αλλά επιτρέπει την εκτέλεση της πλατφόρμας σε λειτουργικά συστήματα που δεν υποστηρίζονται, όπως τα Ubuntu, μέσω εικονοποίησης. Η 4η μέθοδος, αφορά μία χειροκίνητη εγκατάσταση σε ένα εικονικό περιβάλλον Python, είναι στην ουσία μία εφαρμογή python και δεν προσφέρει τις δυνατότητες του Supervisor και τα Add-ons. Όλες οι μέθοδοι εκτός από την core κάνουν χρήση του μηχανισμού του docker οπότε και η εγκατάσταση των επιμέρους addons βασίζεται σε αυτό τον μηχανισμό. Η παρακάτω διαδικασία θα μπορούσε να γίνει και μέσω του γραφικού περιβάλλοντος του

Portainer για ακαδημαϊκούς λόγους όμως, θα παρουσιαστεί η εγκατάσταση του Home Assistant σε περιβάλλον τερματικού[4] των Windows:

1. Εκτελέστηκε η παρακάτω εντολή για την εγκατάσταση του Home Assistant με το docker ως container

$ docker run –init -d –restart=unless-stopped –name=”home-assistant” -e “TZ=Europe/Athens” -v /msciot200007homeassistant:/config -p 8123:8123 homeassistant/home-assistant:latest

Πίνακας 2.1 Επεξήγηση παραμέτρων εντολής εγκατάστασης Home Assistant

Docker run -init	Δημιουργία και εκτέλεση του container με αρχικές ρυθμίσεις
-d	Εκτέλεση του container στο παρασκήνιο με αποτύπωση του ID του
-restart=unless-stopped	Το container θα επανεκιννείται ξανά σε κάθε εκκίνηση του VM εκτός αν του δοθεί εντολή stop
-name=”home-assistant”	Δίνεται ένα όνομα στο container
-e	Παραμετροποίηση των Enviroment μεταβλητών
TZ=Europe/Athens	Ρύθμιση της Ζώνης ώρας στην οποία θα λειτουργεί το Home Assistant
-v	Ορισμός νέου volume
/msciot20007homeassistant:/config	Διεύθυνση του Volume για αποθήκευση των ρυθμίσεων του Home assistant
-p 8123:8123	Ορισμός της πόρτας στην οποία θα είναι προσβάσιμο το Home assistant
homeassistant/home-assistant:latest	Ορισμός του αρχείου image και της έκδοσης που θα ληφθεί και θα εγκατασταθεί

2. Επιβεβαιώθηκε η εγκατάσταση με την εντολή

```
$ docker container ls
```

Έξοδος στην οθόνη:

```
user@snf-18951:~$ docker container ls
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS          PORTS
                                NAMES
259a358b292a   homeassistant/home-assistant:latest   "/init"                  14 seconds ago   Up 11 seconds   0.0.0.
0:8123->8123/tcp, :::8123->8123/tcp   home-assistant
9a7795cadd47   ubuntu                               "/bin/bash"              2 days ago       Up 2 days
                                my-ubuntu2
9daad8a92ba4   ubuntu                               "/bin/bash"              2 days ago       Up 2 days
                                my-ubuntu
3cb3a90b6220   webapp-flask-bgcolor:latest          "python app.py --col…"   13 days ago      Up 13 days      0.0.0.
0:8002->8000/tcp, :::8002->8000/tcp   competent_cohen
7cf0498d167f   webapp-flask-bgcolor:latest          "python app.py"          13 days ago      Up 13 days      0.0.0.
0:8001->8000/tcp, :::8001->8000/tcp   practical_gates
```

3. Ενεργοποιήθηκε το home-assistant container με την παρακάτω εντολή

```
$docker start home-assistant
```

[4] Η πρόσβαση στο περιβάλλον του VM μέσω του τερματικού έγινε κάνοντας χρήστη της εντολής SSH όπως περιγράφηκε και στην ενότητα «2.1 Cloud Virtual Machine (~Okeanos-Knossos, Cyclades)»

Για να επιβεβαιωθεί η σωστή εγκατάσταση εισάγεται η εξής διεύθυνση στον browser: http://83.212.77.18:8123 . Το αποτέλεσμα θα είναι η οθόνη εγγραφής στο Home Assistant. Η παραμετροποίηση του θα περιγραφεί στο «ΚΕΦΑΛΑΙΟ 3:Το Home Assistant».

2.1.5 Εγκατάσταση Βάσης δεδομένων InfluxDB (PaaS)

Εικόνα 2.15 Βάση Δεδομένων InfluxDB

Τα στοιχεία που θα ληφθούν από τους αισθητήρες και τις συσκευές, θα αποστέλλονται στο Home Assistant το οποίο έχει δυνατότητες διατήρησης ιστορικού, αλλά είναι αρκετά περιοριστικές και δεν διαθέτουν μεγάλη ευελιξία. Για τον λόγο αυτό, έπρεπε να επιλεχθεί μία βάση δεδομένων, η οποία θα καταγράφει όλες τις μετρήσεις των αισθητήρων και των συσκευών είτε αυτές αφορούν δεδομένα κατανάλωσης ενέργειας είτε κατάσταση λειτουργίας. Η InfluxDB, είναι από τις πιο διαδεδομένες βάσεις δεδομένων , ανοιχτού κώδικα για χρήση σε IoT εφαρμογές και διαχείριση μεγάλου όγκου δεδομένων, σε πραγματικό χρόνο. Επίσης, μπορεί να εγκατασταθεί εύκολα σε ένα Virtual Machine και συνεργάζεται άψογα με τις περισσότερες πλατφόρμες διαχείρισης IoT εφαρμογών. Στην ενότητα «4.1 Influx DB» θα γίνει πιο λεπτομερής περιγραφή των βάσεων δεδομένων, την διαχείριση μεγάλων δεδομένων και τον χρησιμότητα του κάθε είδους βάσης δεδομένων. Για την εγκατάσταση, της InfluxDB εκτελέστηκαν τα παρακάτω βήματα:

1. Έγινε λήψη και εγκατάσταση του επίσημου κλειδιού για το repository.

```
$ wget -qO- https://repos.influxdata.com/influxdb.key | sudo
apt-key add –
```

2. Έγινε η προσθήκη του repository που απευθύνεται στην έκδοση Linux του VM (Ubuntu 20.04 LTS)

```
$ echo "deb https://repos.influxdata.com/ubuntu focal stable" | sudo
tee /etc/apt/sources.list.d/influxdb.list
```

3. Αφού έγινε η προσθήκη του repository, γίνεται ενημέρωση των διαθέσιμων πακέτων

```
$ sudo apt update
```

4. Τέλος, έγινε η εγκατάσταση των πακέτων της βάσης δεδομένων

```
$ sudo apt install influxdb
```

Έξοδος στην οθόνη:

```
user@snf-18951:~$ sudo apt install influxdb
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-image-4.15.0-139-generic linux-modules-4.15.0-139-generic
  linux-modules-extra-4.15.0-139-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  influxdb
0 upgraded, 1 newly installed, 0 to remove and 4 not upgraded.
Need to get 54.4 MB of archives.
After this operation, 153 MB of additional disk space will be used.
```

5. Στη συνέχεια πρέπει να τεθεί η αυτόματη έναρξη της υπηρεσίας της InfluxDB με την εκκίνηση του συστήματος.

```
$ sudo systemctl unmask influxdb
$ sudo systemctl enable influxdb
```

Η αρχική εγκατάσταση, της InfluxDB δεν προσφέρει κανένα επίπεδο ασφάλειας, για τον λόγο αυτό πρέπει να εισαχθούν στοιχεία ταυτοποίησης χρήστη. Για την επίτευξη αυτού, ακολουθήθηκαν τα παρακάτω βήματα:

1. Έναρξη της InfluxDB

```
$ influx
```

2. Δημιουργία ονόματος χρήστη “admin” και κωδικού χρήστη

```
> CREATE USER admin WITH PASSWORD '«Κωδικόςχρήστη»' WITH ALL
PRIVILEGES
```

3. Έξοδος από την InfluxDB

```
> exit
```

Έξοδος στην οθόνη:

```
user@snf-18951:~$ influx
Connected to http://localhost:8086 version 1.8.10
InfluxDB shell version: 1.8.10
> CREATE USER admin WITH PASSWORD '[redacted]' WITH ALL PRIVILEGES
> exit
```

4. Επεξεργασία αρχείου ρυθμίσεων της InfluxDB, ώστε να χρησιμοποιεί τα στοιχεία ταυτοποίησης που ορίστηκαν παραπάνω.

```
$ sudo nano /etc/influxdb/influxdb.conf
```

5. Ορισμός των παρακάτω μεταβλητών, του τμήματος [HTTP] σε κατάσταση true.

```
auth-enabled = true
pprof-enabled = true
pprof-auth-enabled = true
ping-auth-enabled = true
```

6. Αποθήκευση αλλαγών με CTRL+X και Y
7. Επανεκκίνηση της InfluxDB, χειροκίνητα και η βάση δεδομένων είναι έτοιμη προς χρήση.

```
$ sudo systemctl restart influxdb
```

Από αυτή τη στιγμή, για να έχει ο χρήστη πλήρη πρόσβαση στα δεδομένα της influx θα πρέπει να χρησιμοποιήσει την εντολή εκκίνησης με αυτή την μορφή:

```
$ influx -username admin -password «κωδικός χρήστη»
```

2.1.6 Εγκατάσταση Grafana (SaaS)

Εικόνα 2.16 Grafana

Τα δεδομένα ιστορικού, που αποθηκεύτηκαν στην βάση δεδομένων InfluxDB, δεν είναι επεξεργάσιμα αν δεν μπορούμε να τα επιβλέπουμε ή να επεξεργαζόμαστε τον τρόπο εμφάνισης τους. Για τον λόγο αυτό, ήταν

απαραίτητη η χρήση μίας εφαρμογής, που μπορεί να συνεργάζεται με μία τέτοια βάση δεδομένων και να μετατρέπει αυτά τα δεδομένα σε «εικόνες». Το Grafana, έχει την δυνατότητα λήψης αυτών των δεδομένων σε πραγματικό χρόνο και προσφέρει τα απαραίτητα εργαλεία για την προβολή τους σε όποια μορφή επιλέξει ο χρήστης. Αναλυτική περιγραφή των δυνατοτήτων του, θα γίνει στην ενότητα «4.2 Grafana». Για την εγκατάσταση του ακολουθήθηκαν τα παρακάτω βήματα:

1. Όπως και με την InfluxDB έγινε λήψη του απαραίτητου κλειδιού για το repository

```
$ wget -q -O - https://packages.grafana.com/gpg.key | sudo
apt-key add –
```

2. Έγινε προσθήκη του repository

```
$ echo "deb https://packages.grafana.com/oss/deb stable main"
| sudo tee -a /etc/apt/sources.list.d/grafana.list
```

3. Ενημερώθηκαν τα διαθέσιμων πακέτων

```
$ sudo apt update
```

4. Έγινε η εγκατάσταση του Grafana

```
$ sudo apt install Grafana
```

5. Ενεργοποιήθηκε η υπηρεσία ώστε το Grafana να εκκινείτε αυτόματα με την έναρξη του συστήματος.

```
$ sudo systemctl enable grafana-server
```

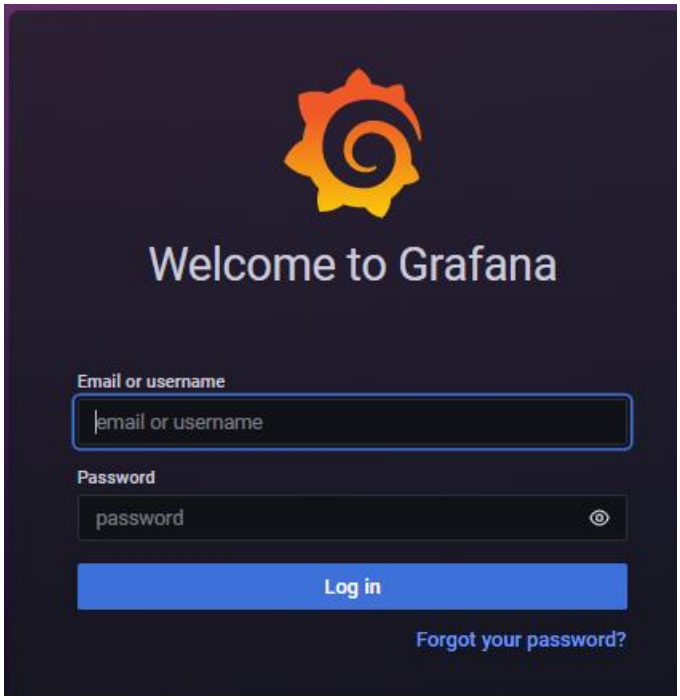
Έξοδος στην οθόνη:

```
user@snf-18951:~$ sudo systemctl enable grafana-server
Synchronizing state of grafana-server.service with SysV service script with /lib/systemd/syst
emd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable grafana-server
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /usr/lib
/systemd/system/grafana-server.service.
```

6. Την πρώτη φορά, η έναρξη γίνεται χειροκίνητα.

```
$ sudo systemctl start grafana-server
```

Για να επιβεβαιωθεί η σωστή εγκατάσταση εισάγεται η εξής διεύθυνση στον browser: http://83.212.77.18:3000. Το αποτέλεσμα θα είναι η οθόνη εγγραφής στο Grafana.



Εικόνα 2.17 Οθόνη εγγραφής Grafana

2.2 Υλοποίηση ελέγχου κλιματιστικών και λήψης περιβαλλοντικών δεδομένων ανά δωμάτιο.

Για τον έλεγχο, των περιβαλλοντικών συνθηκών σε κάθε δωμάτιο, αλλά και τον έλεγχο των κλιματιστικών έπρεπε να γίνει μία υλοποίηση, η οποία θα πρέπει να συνδυάζει αυτές τις ιδιότητες, ενώ ταυτόχρονα να συνδέεται ασύρματα στο δίκτυο με χρήση WiFi και να επικοινωνεί με τον MQTT broker μέσω μηνυμάτων. Η υλοποίηση αυτή, στηρίζεται σε ένα μικροελεγκτή WeMos D1 mini pro, έναν αισθητήρα περιβάλλοντος DHT22 και ένα αισθητήρα εκπομπής υπέρυθρων σημάτων. Για την αποκωδικοποίηση, των υπέρυθρων σημάτων ελέγχου των κλιματιστικών, από τα αυθεντικά τηλεχειριστήρια, υλοποιήσαμε ένα αντίστροφο κύκλωμα, που χρησιμοποιεί ένα μικροελεγκτή ESP32 και ένα δέκτη υπέρυθρων.

2.2.1 Προετοιμασία Arduino IDE

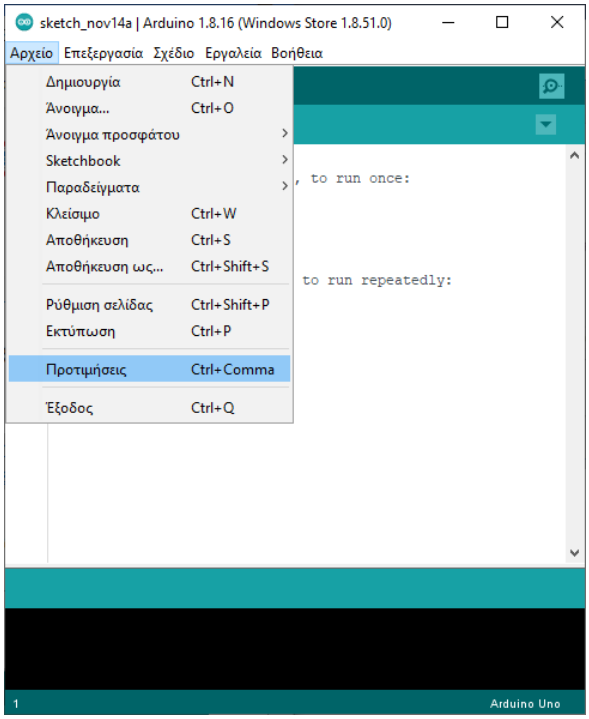
Όσον αφορά, τις υλοποιήσεις αυτές ήταν απαραίτητη η προετοιμασία του Arduino IDE, το οποίο θα χρησιμοποιηθεί για τον προγραμματισμό των μικροελεγκτών. Οι μικροελεγκτές και οι αισθητήρες, που θα χρησιμοποιηθούν δεν εμπεριέχονται στις βασικό πακέτο του Arduino IDE για αυτό τον λόγο πρέπει να γίνει λήψη των απαραίτητων πακέτων και βιβλιοθηκών.

2.2.1.1 Οδηγίες εγκατάστασης πακέτων μικροελεγκτών στο Arduino IDE

Για να αναγνωριστούν οι μικροελεγκτές ESP32 και ESP8266 από το Arduino IDE εκτελέστηκαν τα παρακάτω βήματα:

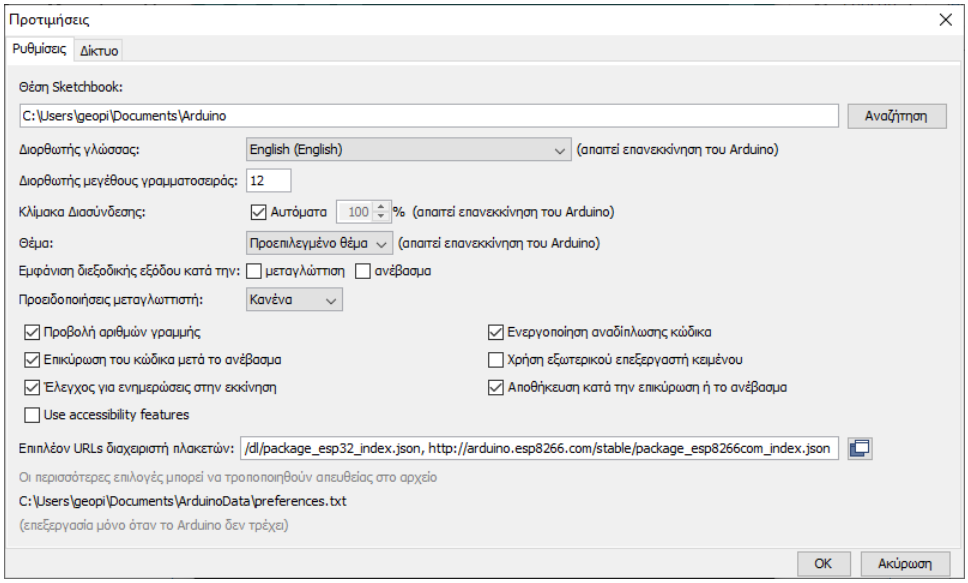
1. Επιλέχθηκε το μενού «Αρχείο» επιλέχθηκε το υπομενού «Προτιμήσεις»

Έξοδος στην οθόνη:



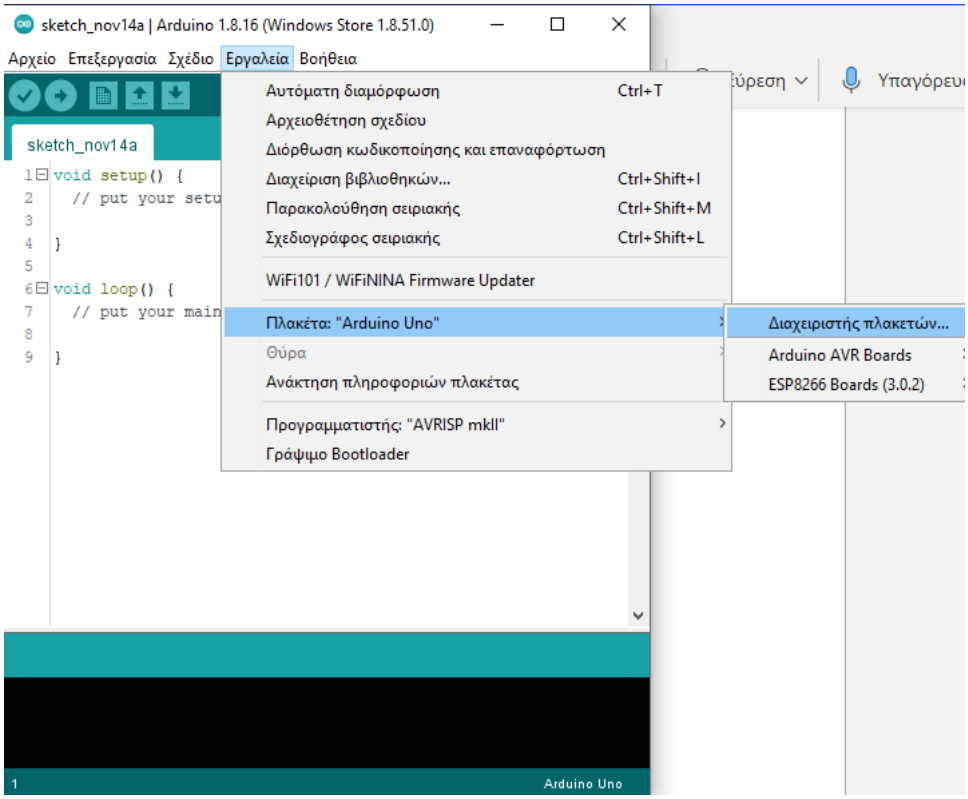
2. Στο νέο παράθυρο, εισήχθησαν οι διευθύνσεις https://dl.espressif.com/dl/package_esp32_index.json, http://arduino.esp8266.com/stable/package_esp8266com_index.json στο πλαίσιο «Επιπλέον URLs διαχειριστή πλακετών» και στην συνέχεια το κουμπί «OK». Με αυτό τον τρόπο το IDE θα εισάγει τις απαραίτητες βιβλιοθήκες που αφορούν την διαχείριση των μικροελεγκτών ESP32 και ESP8266.

Έξοδος στην οθόνη:



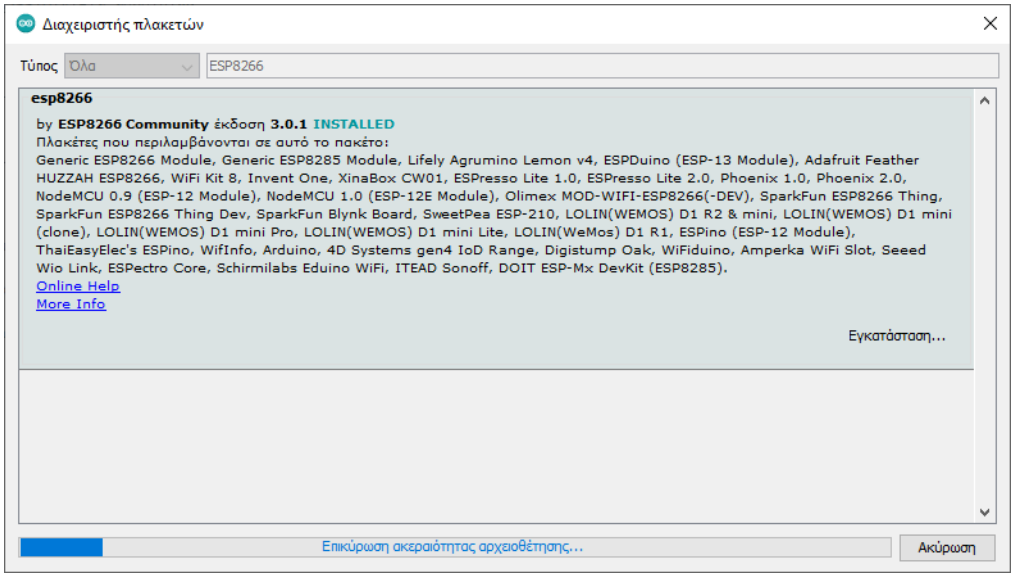
3. Στη συνέχεια, επιλέχθηκε ο «Διαχειριστής πλακετών» από το μενού Εργαλεία>Πλακέτα> Διαχειριστής πλακετών

Έξοδος στην οθόνη:



4. Στο νέο παράθυρο αναζήτησης πληκτρολογήθηκε το «ESP8266» και στην συνέχεια στο πακέτο που αφορά τα ESP8266 επιλέχθηκε το κουμπί «Εγκατάσταση». Το ίδιο έγινε αντίστοιχα και για το «ESP32»

Έξοδος στην οθόνη:

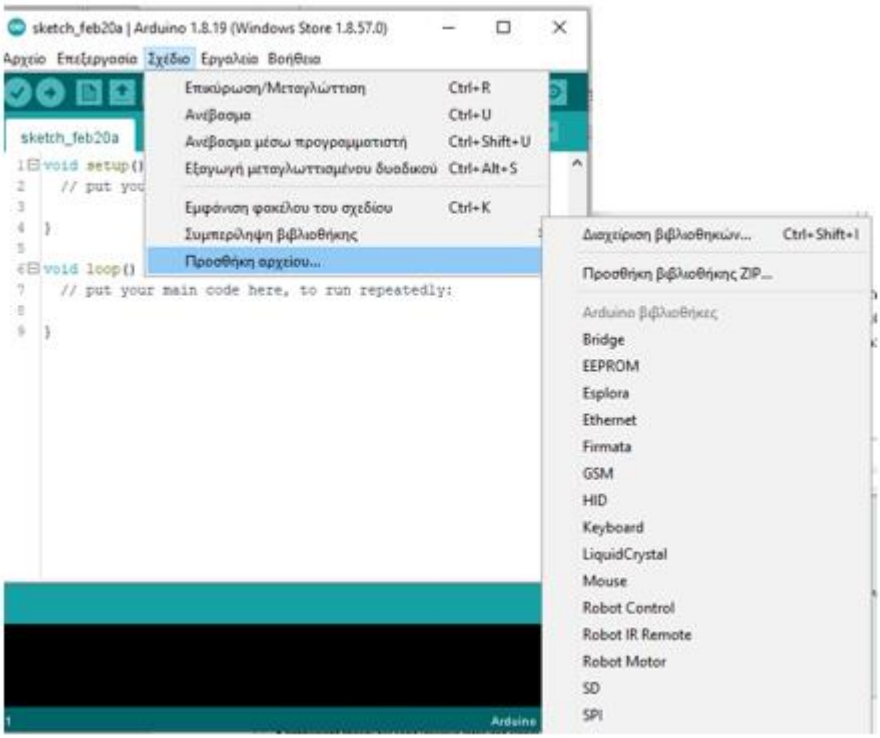


2.2.1.2 Οδηγίες εγκατάστασης βιβλιοθηκών αισθητήρων και ενεργοποιητών στο Arduino IDE

Για την χρήση των διάφορων αισθητήρων η ενεργοποιητών χρειάζεται η λήψη κάποιων βοηθητικών βιβλιοθηκών. Αυτό γίνεται με:

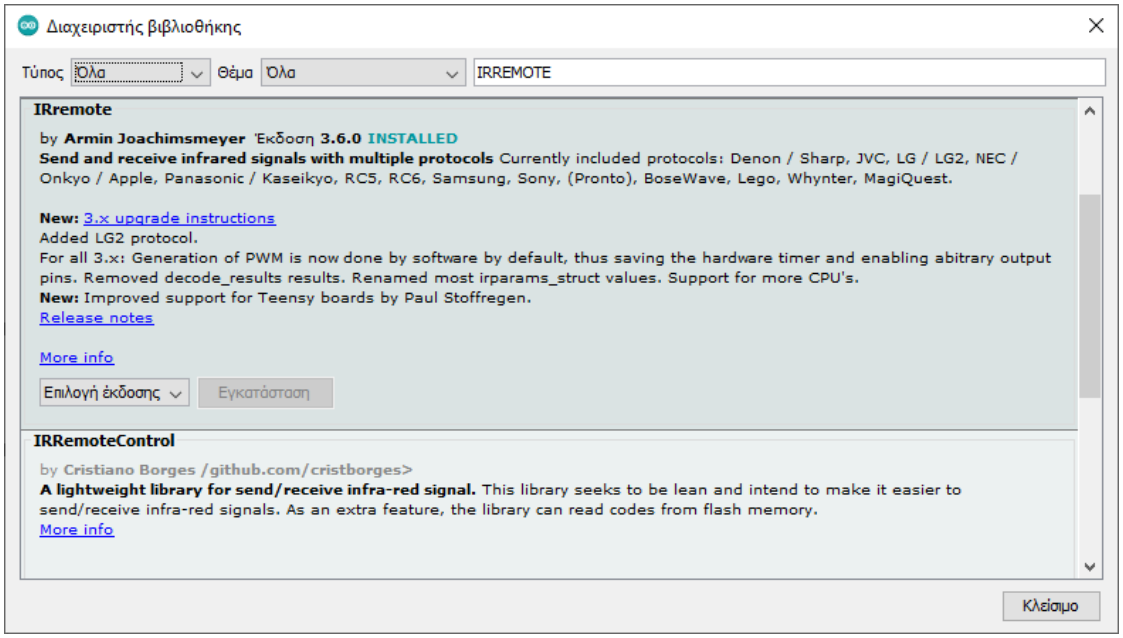
1. Αναζήτηση της επιθυμητής βιβλιοθήκης από το μενού Σχέδιο> Συμπερίληψη Βιβλιοθήκης> Διαχείριση Βιβλιοθηκών

Έξοδος στην οθόνη:



2. Επιλογή της επιθυμητής βιβλιοθήκης και στη συνέχεια επιλέγεται η «Εγκατάσταση»

Έξοδος στην οθόνη:



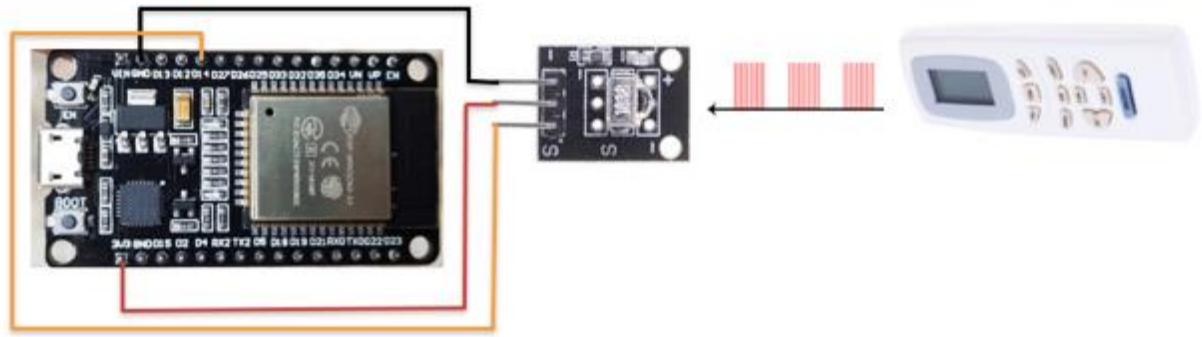
2.2.2 **Αποκωδικοποιητής Υπέρυθρων σημάτων (IR decoder).**[28]

Σε αυτή την ενότητα, θα περιγραφεί η υλοποίηση ενός αποκωδικοποιητή υπέρυθρων σημάτων ή αλλιώς IR decoder. Αυτό το βοηθητικό κύκλωμα, χρησιμοποιήθηκε για την αποκωδικοποίηση των σημάτων, που αποστέλλονται από το κλασσικό τηλεχειριστήριο των κλιματιστικών . Ο τρόπος με τον οποίο έγινε η αποκωδικοποίηση, είναι ο εξής:

1. Το τηλεχειριστήριο του κάθε κλιματιστικού, τοποθετήθηκε στην ίδια ευθεία με τον δέκτη υπέρυθρων σημάτων.
2. Κάθε φορά που επιλέγονταν ένα πλήκτρο λειτουργίας από το τηλεχειριστήριο, ένα διαμορφωμένο σήμα υπέρυθρων αποστέλλονταν στον δέκτη υπέρυθρων που ήταν συνδεδεμένος στο ESP32.
3. Ο δέκτης υπέρυθρων μετέτρεψε το διαμορφωμένο σήμα σε δυαδικό και το προώθησε στον μικροελεγκτή[29]
4. Αυτός, όντας προγραμματισμένος κατάλληλα (Παράρτημα Α. IRrecvDumpV2.ino) και συνδεδεμένος με ένα ηλεκτρονικό υπολογιστή, έκανε την αποκωδικοποίηση του σήματος και στην συνέχεια εμφάνισε στην οθόνη του Serial Monitor του Arduino IDE, την κωδικοποίηση που έλαβε.
5. Οι κωδικοποιήσεις που ελήφθησαν, αντιστοιχούν σε κάποιο συνδυασμό λειτουργιών[5] του κλιματιστικού. Για τον λόγο αυτό καταγράφηκαν σε ένα πρόχειρο αρχείο κειμένου, ώστε να χρησιμοποιηθούν ξανά στην υλοποίηση που θα περιγραφεί στην ενότητα 2.2.3 .

[5] Μία ιδιαιτερότητα που παρουσιάζουν τα χειριστήρια των κλιματιστικών είναι ότι ο χρήστης, μπορεί να επιλέγει για παράδειγμα το κουμπί «ON» αλλά ταυτόχρονα αποστέλλει στο κλιματιστικό πληροφορίες που αφορούν την επιθυμητή θερμοκρασία, την ένταση του ανεμιστήρα, την κατάσταση λειτουργίας ψύξης/ θέρμανσης, και την θέση του κλαπέτου εξαγωγής αέρα.

Το σχηματικό διάγραμμα του κυκλώματος είναι το εξής (Εικόνα 2.18):

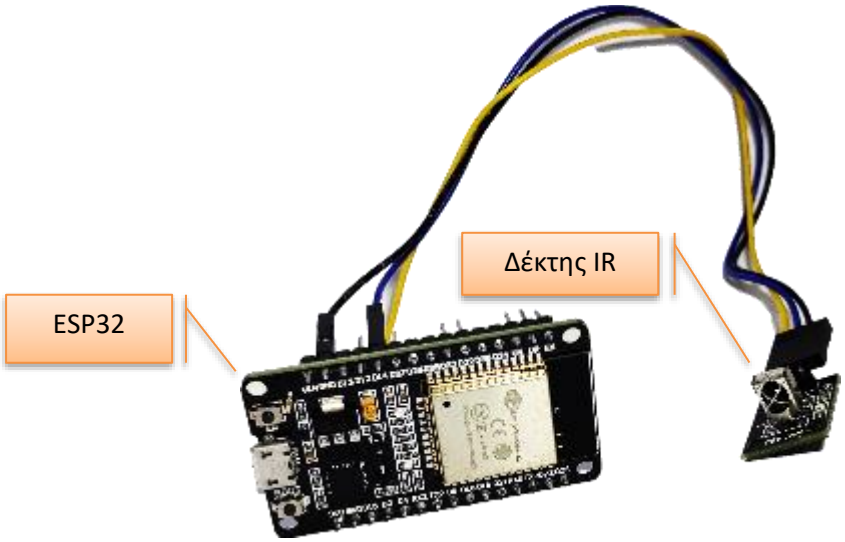


Εικόνα 2.18 Σχηματικό διάγραμμα IR decoder

Τα υλικά που χρησιμοποιήθηκαν ήταν τα εξής (Πίνακας 2.2):

Πίνακας 2.2 Υλικά κατασκευής IR decoder

Υλικά	Τεμάχια	Κόστος
ESP32 Development Board - DEVKIT V1	1	9,90€
Δέκτης Αισθητήρα Υπέρυθρων TL1838	1	1,50€
Καλώδια Jumper(15cm female to female)	3	0,54€



Εικόνα 2.19 Η υλοποιημένη συνδεσμολογία του IR decoder

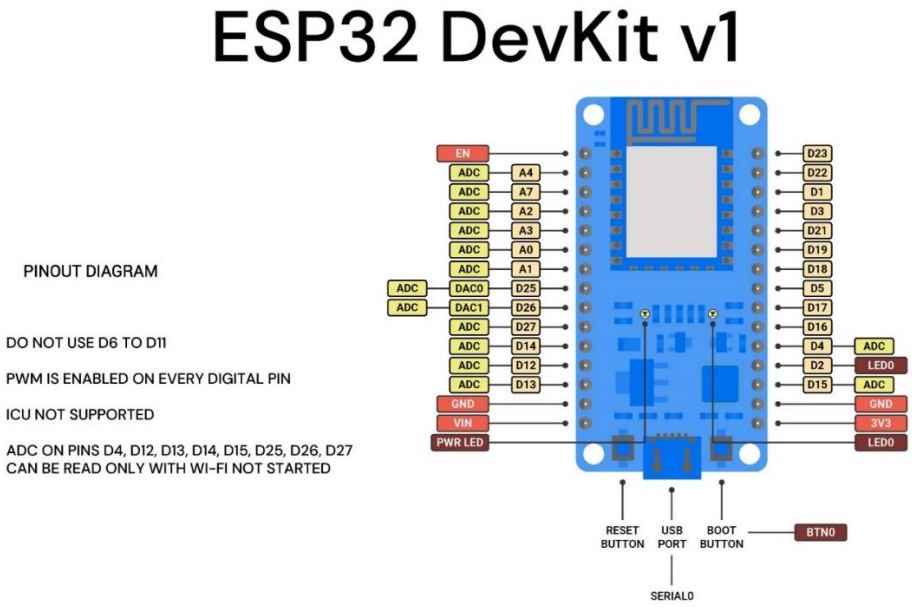
Αφού, υλοποιήθηκε η συνδεσμολογία των εξαρτημάτων (Εικόνα 2.19), έγινε λήψη της βιβλιοθήκης IRremote.h από την οποία χρησιμοποιήθηκε το έτοιμο παράδειγμα με όνομα «IRrecvDumpV2.ino» (βλ. Παράρτημα Α. IRrecvDumpV2.ino). Το πρόγραμμα αυτό, βοήθησε στην συλλογή των κωδικοποιήσεων[6] των δύο τηλεχειριστηρίων των κλιματιστικών, για κάθε λειτουργία.Το κύκλωμα του αποκωδικοποιητή, χρησιμοποιήθηκε για την αποκωδικοποίηση τριών βασικών προρυθμισμένων λειτουργιών, για τα κλιματιστικά. Οι λειτουργίες αυτές είναι:

1. Θέρμανσης (heat): θερμοκρασία 25°C , Ένταση ανεμιστήρα: Αυτόματη, θέση κλαπέτου: χαμηλή
2. Ψύξης (cool): θερμοκρασία 18°C , Ένταση ανεμιστήρα: Αυτόματη, θέση κλαπέτου: υψηλή
3. Απενεργοποίηση Κλιματιστικού (off).

2.2.2.1 Μικροελεγκτής

- **ESP32 Development Board - DEVKIT V1**

Το ESP32 Development Board - DEVKIT V1, είναι μία πλατφόρμα ανάπτυξης βασισμένη στον μικροελεγκτή ESP32. Τα χαρακτηριστικά που την ξεχωρίζουν είναι ότι διαθέτει διπύρηνο επεξεργαστή, διασυνδεσιμότητα WiFi και Bluetooth, πολλά περιφερειακά συστήματα (I2C, UART,SPI κ.α.), ενώ ταυτόχρονα είναι χαμηλής ισχύος. Σε αυτή τη δευτερεύουσα υλοποίηση, δεν θα αξιοποιηθούν όλα αυτά τα χαρακτηριστικά του μικροελεγκτή, καθώς ο λόγος επιλογής του ήταν η διαθεσιμότητα του από προηγούμενες υλοποιήσεις και ότι μπορεί να προγραμματιστεί στο περιβάλλον του Arduino IDE.[30]



Εικόνα 2.20 ESP32 Development Board - DEVKIT V1 Pinout

[6] Οι κωδικοποιήσεις που προέκυψαν από το κάθε τηλεχειριστήριο ανά λειτουργία αναφέρονται στο «Παράρτημα Β. Κωδικοποιήσεις τηλεχειριστηρίων κλιματιστικών»

Πίνακας 2.3 Χαρακτηριστικά λειτουργίας - ESP32 Development Board - DEVKIT V1

Χαρακτηριστικά		**Χαρακτηριστικά (συνέχεια)**	
Μικροελεγκτής	Tensilica 32-bit Single-/Dual-core CPU Xtensa LX6	**SPI**	2
Τάση λειτουργίας	3,3Volts	**I2C**	3
Τάση εισόδου	7-12Volts	**Μνήμη Flash**	4MB
Ψηφιακές Είσοδοι/Εξοδοι	25	**SRAM**	520kB

2.2.2.2 Αισθητήρας

- **Δέκτης Υπέρυθρων - TL1838**

Ο συγκεκριμένος δέκτης είναι ένας αισθητήρας πολύ διαδεδομένος για χρήση σε τηλεχειριστήρια. Τα βασικά του χαρακτηριστικά είναι:

Χαρακτηριστικά	
Τάση λειτουργίας	2,7V-5.5V
Απόσταση λήψης	10-15m
Φέρουσα συχνότητα	38kHz
Γωνία λήψης	35^{O}
Ρεύμα ηρεμίας	0,8-1,5A

2.2.3 Τηλεχειρισμός με υπέρυθρα σήματα και ενσωματωμένο αισθητήρα περιβάλλοντος.

Σε αυτή την ενότητα, θα περιγραφεί η υλοποίηση που αφορά κυρίως τον έλεγχο των κλιματιστικών, που βρίσκονται στο δωμάτιο του «Σαλονιού» και το δωμάτιο του «Υπνοδωματίου». Ο έλεγχος των κλιματιστικών, συνδέεται άμεσα με τις περιβαλλοντικές συνθήκες που επικρατούν σε κάθε δωμάτιο. Αυτό σημαίνει ό,τι ήταν απαραίτητη η παρακολούθηση τους με ένα αισθητήρα περιβάλλοντος. Τα κλιματιστικά της οικίας στην οποία έγινε η υλοποίηση δεν ήταν «έξυπνα», επομένως δεν διέθεταν απομακρυσμένη διαχείριση πέρα του τηλεχειριστηρίου τους το οποίο λειτουργεί με αποστολή υπέρυθρων σημάτων. Στην προηγούμενη ενότητα, περιγράψαμε την υλοποίηση που αποκωδικοποιεί αυτά τα υπέρυθρα σήματα έτσι ώστε να χρησιμοποιήσουμε τα δεδομένα που λάβαμε στην παρούσα υλοποίηση. Συνολικά κατασκευάστηκαν, τρείς τέτοιες υλοποιήσεις η μία εξ ’αυτών, δεν θα χρησιμοποιεί τον πομπό υπέρυθρων σημάτων καθώς θα τοποθετηθεί στο δωμάτιο του γραφείου που δεν διαθέτει κλιματιστικό, παρόλα αυτά θα τον έχει διαθέσιμο για μελλοντική χρήση.

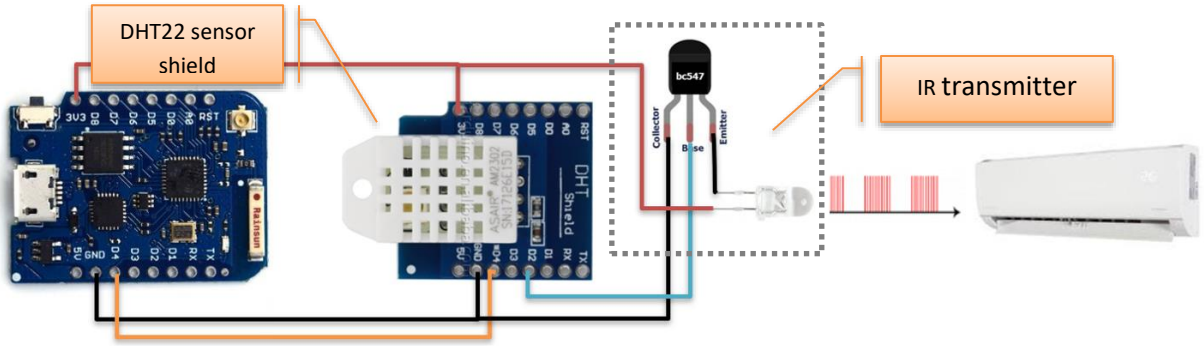
Σκοπός της υλοποίησης αυτής, είναι:

1. Να καταγράφει συνεχώς τις περιβαλλοντικές συνθήκες (θερμοκρασία, υγρασία, αίσθηση θερμοκρασίας)
2. Να αποστέλλει τις περιβαλλοντικές συνθήκες στον MQTT broker μέσω WiFi

Ενώ ταυτόχρονα να μπορεί:

1. Να λαμβάνει μηνύματα από τον MQTT broker που αφορούν την επιλογή λειτουργίας του εκάστοτε κλιματιστικού (αν υπάρχει).

2. Και στη συνέχεια, να αποστέλλει την κωδικοποίηση, που λήφθηκε από τον IR receiver της προηγούμενης ενότητας, με τον πομπό υπέρυθρων σημάτων ή IR transmitter, πίσω στο κλιματιστικό.



Εικόνα 2.21 Σχηματικό διάγραμμα κυκλώματος IR transmitter με αισθητήρα περιβάλλοντος

Τα υλικά που χρησιμοποιήθηκαν ήταν τα εξής **(Πίνακας 2.4):**

Πίνακας 2.4 Υλικά κατασκευής IR transmitter με αισθητήρα περιβάλλοντος.

Υλικά	Τεμάχια	Κόστος
IR LED Transmitter 5MM - 940NM (210MW)	3	0,48 €
WeMos D1 mini pro ESP8266 (V1.0)	3	19,11 €
WeMos D1 mini pro Protoboard Shield	3	1,94 €
Τροφοδοτικό 5V 3A	3	11,61 €
DHT22 AM2302 Temperature and Humidity Sensor Shield Module for WeMos D1 Mini	3	23,22 €
Καλώδιο micro USB ΣΕ USB 2.0 male	3	6,00 €
Τρανζίστορ BC547	3	0,15 €

Εικόνα 2.22 Ο υλοποιημένος IR transmitter με αισθητήρα περιβάλλοντος

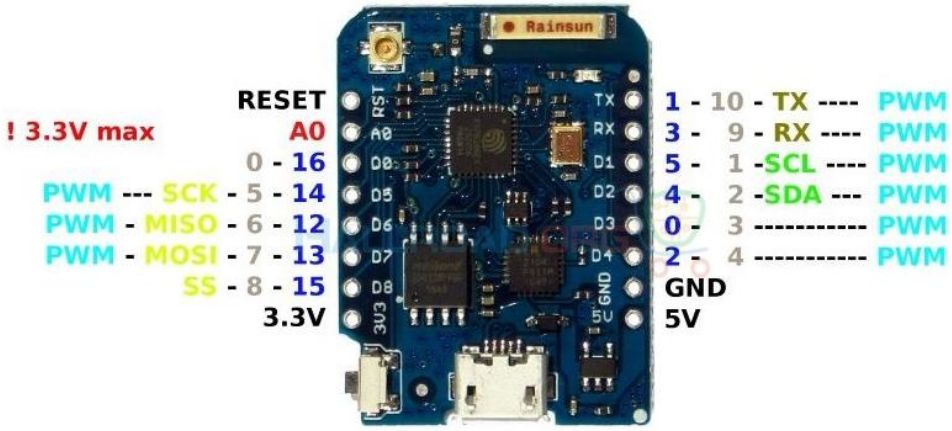
Αφού, υλοποιήθηκε η συνδεσμολογία των εξαρτημάτων (Εικόνα 2.22) έγινε χρήση των βιβλιοθηκών IRremoteESP8266.h, IRsend.h (IR transmitter), DHT.h (αισθητήρας περιβάλλοντος DHT22), ESP8266WiFi.h (WiFi) και PubSubClient.h (MQTT) και δημιουργήθηκε ένας ενιαίος κώδικας που ενσωματώνει όλες τις απαραίτητες λειτουργίες σε μία συσκευή. Επίσης, προστέθηκαν και οι κωδικοποιήσεις, που εξήχθησαν από την υλοποίηση της προηγούμενης ενότητας (2.2.2) για κάθε λειτουργία ανά κλιματιστικό, στα αντίστοιχα μέρη του κώδικα. Μόλις η συσκευή, συνδεθεί με την τροφοδοσία ρεύματος, θα συνδεθεί αυτόματα στο WiFi της οικίας και τον MQTT broker και στη συνέχεια θα αποστέλλει συνεχώς τις μετρήσεις θερμοκρασία υγρασίας και αίσθησης θερμοκρασίας στα αντίστοιχα topics. Η συσκευές που θα ελέγχουν τα κλιματιστικά, θα έχουν κάνει subscribe στα topics λειτουργίας τους και θα αναμένουν μηνύματα MQTT από την πλατφόρμα του Home Assistant. Τα μηνύματα αυτά θα αντιστοιχηθούν στις εντολές λειτουργίας των κλιματιστικών, που περιλαμβάνουν τις κωδικοποιήσεις που προστέθηκαν προηγουμένως.

Οι κώδικες που υλοποιήθηκαν, για κάθε δωμάτιο/συσκευή, περιγράφονται αναλυτικά στο «Παράρτημα Γ. Κώδικες συσκευών ελέγχου κλιματιστικών και περιβαλλοντικών συνθηκών ανά δωμάτιο/συσκευή.».

2.2.3.1 Μικροελεγκτής

- **WeMos D1 mini pro ESP8266 (V1.0)**

Το WeMos D1 mini pro, είναι ένας χαμηλού κόστους, και χαμηλής κατανάλωσης αναπτυξιακός μικροελεγκτής που έχει δυνατότητα ασύρματης επικοινωνίας με WiFi. Η δομή του, βασίζεται στον επεξεργαστή ESP8266EX, της εταιρίας Espressif Systems και διαθέτει, μία Αναλογική θύρα και έντεκα Ψηφιακές θύρες οι οποίες προγραμματίζονται μέσω της θύρας micro-USB, με τον ενσωματωμένο προγραμματιστή CP2104. Ο προγραμματισμός του, μπορεί να επιτευχθεί μέσω του Arduino IDE (που χρησιμοποιήθηκε), με Micro Python ή NodeMCU. Το WeMos πέραν των χαρακτηριστικών λειτουργίας του διαθέτει και πολλών ειδών Shields που μπορούν να προσαρμοστούν σε αυτό χωρίς να χρειάζονται μόνιμες συνδέσεις ή κολλήσεις (soldering).



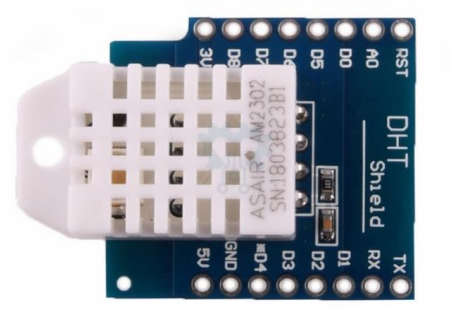
Εικόνα 2.23 WeMos D1 mini pro - Pin Out

Πίνακας 2.5 Χαρακτηριστικά λειτουργίας WeMos D1 mini pro

Χαρακτηριστικά	
Μικροελεγκτής	ESP-8266EX
Τάση λειτουργίας	3,3V
Ταχύτητα χρονισμού	80MHz/160MHz
Ψηφιακές Είσοδοι/Εξοδοι	11
Αναλογική Είσοδος	1 (3.2V max input)
Μνήμη Flash	16Mbytes(128M bit)
Άλλα περιφερειακά	Υποδοχή για εξωτερική κεραία, ενσωματωμένη κεραμική κεραία, CP2104 USB-TO-UART IC, Όλα τα I/O pin έχουν Interrupt /PWM/I2C/ υποστηρίζουν 1-wire εκτός από το pin D0, Όλα τα I/O pins λειτουργούν στα 3,3V.
Διαστάσεις	34.2x 25.6x
Weight:	2.5g

2.2.3.2 Αισθητήρας και Ενεργοποιητής

- **DHT22 Αισθητήρας θερμοκρασίας/Υγρασίας shield board For WeMos D1 Mini**



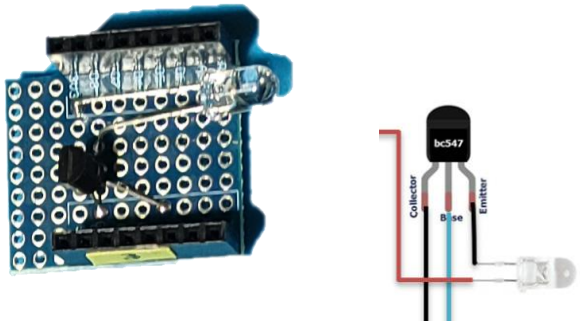
Εικόνα 2.24 Shield Αισθητήρα θερμοκρασίας/υγρασίας DHT22

Για την καταγραφή, της θερμοκρασίας και της υγρασίας επιλέχθηκε ο αισθητήρας DHT22. Ο αισθητήρας αυτός, ήταν προεγκατεστημένος σε έτοιμο PCB (Shield) με κάποια περιφερειακά, ειδικά σχεδιασμένο για να

προσαρμόζεται ακριβώς πάνω στον μικροελεγκτή WeMos D1 mini pro. Πρόκειται για ένα βαθμονομημένο αισθητήρα περιβάλλοντος υψηλής αξιοπιστίας και ακρίβειας, καθώς λειτουργεί στέλνοντας ψηφιακά σήματα στον μικροελεγκτή. Το shield, περιλαμβάνει ένα χωρητικό αισθητήρα υγρασίας το οποίο σημαίνει ότι δεν απαιτείται η άμεση επαφή με το αντικείμενο του οποίου θα μετρήσει την υγρασία. Ενώ περιέχει και ένα υψηλής ακρίβειας αισθητήρα θερμοκρασίας με μεγάλο εύρος θερμοκρασιών από -40 ^{O}C έως 80 ^{O}C. Οι δύο αυτοί αισθητήρες, επικοινωνούν με ένα μικροεπεξεργαστή 8-bit από τον οποίο μεταδίδονται οι μετρήσεις στην έξοδο (επαφή D4 του WeMos).

Χαρακτηριστικά	
Τάση λειτουργίας	3,3-5,5V
Ρεύμα τροφοδοσίας	1,3-2,1mA
Έξοδος σήματος	1 ψηφιακό σήμα μέσω 1-wire
Περίοδος λειτουργίας	~2sec
Αισθητήριο	Πολυμερής Πυκνωτής υγρασίας
Εύρος Υγρασίας	0-100%RH
Εύρος θερμοκρασίας	-40^{O}C – 80^{O}C
Διαστάσεις	34.2x25.6x6mm

- **Ενεργοποιητής- κύκλωμα εκπομπής υπέρυθρων σημάτων**[28]



Εικόνα 2.25 Κύκλωμα Εκπομπής υπέρυθρων σημάτων σε shield προτυποποίησης

Στην Εικόνα 2.25, φαίνεται η υλοποίηση του εκπομπού υπέρυθρων σημάτων, όπως ενσωματώθηκε σε shield προτυποποίησης. Το shield αυτό χρησιμοποιήθηκε για την ταχύτερη διασύνδεση των κυκλωμάτων με το WeMos D1 mini pro. Αποτελείται από, 1 LED IR που εκπέμπει στα 940nm και ένα γενικής χρήσεως τρανζίστορ NPN το BC547. Η συνδεσμολογία τους, επιτρέπει την καλύτερη μετάδοση υπέρυθρων σημάτων από το LED καθώς το τρανζίστορ ενισχύει το περιορισμένο εύρος εκπομπής του. Υπενθυμίζεται, ότι το κύκλωμα αυτό αν και υλοποιήθηκε, δεν θα χρησιμοποιηθεί από την υλοποίηση που τοποθετήθηκε στον χώρο του γραφείου, θα είναι όμως διαθέσιμο για μελλοντική χρήση.

2.3 Υλοποίηση καταμέτρησης ενέργειας και ελέγχου λειτουργίας οικιακών συσκευών[31]

Εικόνα 2.26 Shelly

Εικόνα 2.27 Τα προϊόντα Shelly που χρησιμοποιήθηκαν

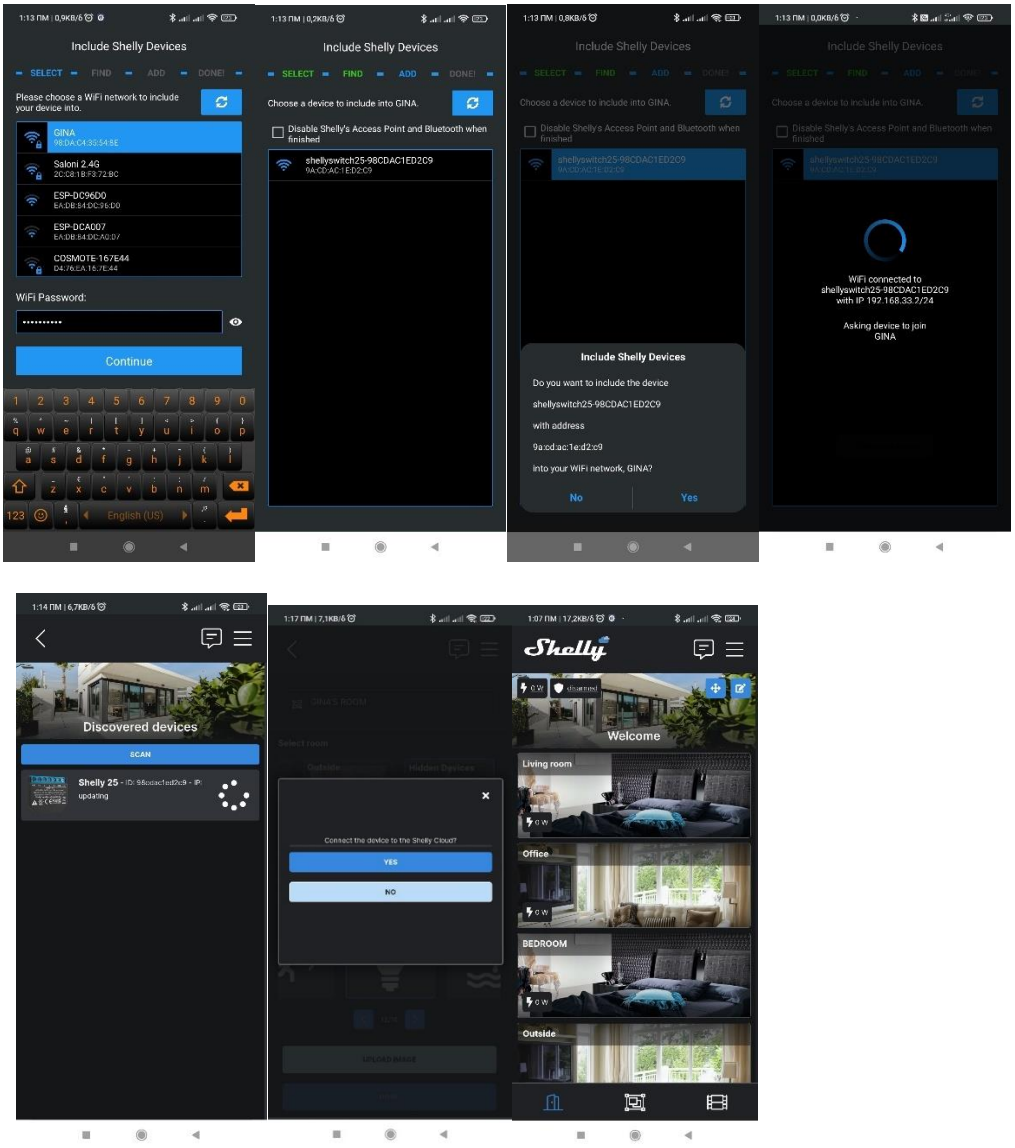
Στην ενότητα αυτή, θα γίνει αναφορά στις συσκευές που πέρα από τον έλεγχο των συσκευών στις οποίες θα συνδεθούν, θα μας προσφέρουν και τις απαραίτητες ενδείξεις κατανάλωσης ενέργειας. Οι συσκευές αυτές, όπως έχει ήδη αναφερθεί, ονομάζονται Shelly και κατασκευάζονται από την Alterco Robotics, που εδρεύει στην Βουλγαρία, ενώ από το 2019 και στις ΗΠΑ. Η εταιρεία ειδικεύεται στην παραγωγή υλοποιήσεων για

«έξυπνα σπίτια» που είναι εύκολα στην χρήση και την εγκατάσταση, μπορούν να δεχθούν αυτοματισμούς και όλα αυτά με προσιτό κόστος. Τα προϊόντα που διαθέτει μπορεί να είναι κάποια «έξυπνα» relay, «έξυπνοι» αισθητήρες η ακόμα και προϊόντα όπως «έξυπνοι» ρευματοδότες ή «έξυπνοι» λαμπτήρες. Τα Shelly διαθέτουν την δική τους cloud εφαρμογή η οποία επιτρέπει τον έλεγχο των συσκευών, δημιουργία προγραμματισμένων λειτουργιών, σκηνές λειτουργίες και καταγραφή της ενεργειακής κατανάλωσης. Ο πιο σημαντικός παράγοντας που κάνει τα Shelly ελκυστικά είναι ότι έχουν την δυνατότητα να επικοινωνούν ασύρματα μέσω WiFi με τα πιο διαδεδομένα πρωτόκολλα IoT όπως το MQTT, το CoAP και το REST . Ταυτόχρονα, συνεργάζονται με πολύ γνωστές εμπορικές πλατφόρμες έξυπνων σπιτιών όπως το Google Home,την Alexa , αλλά και πλατφόρμες ανοιχτού κώδικα όπως το Home Assistant και το Node Red. Τα Shelly, έχουν μεγάλη ευελιξία ακόμη και στο λογισμικό τους καθώς διαθέτουν προσβάσιμα pins για προγραμματισμό αλλά και αποσφαλμάτωση μέσω ενός USB to UART adapter και μίας τροφοδοσίας 3,3V. Η υλοποίηση τους βασίζεται στον γνωστό μικροεπεξεργαστή ESP8266 με 2 MB μνήμη Flash. Ενώ παράλληλα έχουν πολύ ποιοτική και ασφαλή υλοποίηση με προστασίες υπερθέρμανσης και υπερφόρτωσης.

Όπως έχει ήδη αναφερθεί, χρειάζεται να ελεγχθούν οι ενεργοβόρες ηλεκτρικές συσκευές αρχικά, λαμβάνοντας κάποια στατιστικά δεδομένα για την κατανάλωση ενέργειας τους και στην συνέχεια με αυτοματοποίηση του τρόπου λειτουργίας τους. Όλες οι Shelly συσκευές, που χρησιμοποιήθηκαν θα επικοινωνούν με το Home Assistant μέσω MQTT. Από την στιγμή, που θα ενεργοποιηθεί η επικοινωνία με το πρωτόκολλο MQTT, οι συσκευές παύουν να επικοινωνούν με την υπηρεσία cloud της Shelly. Επομένως, χάνεται η απομακρυσμένη διαχείριση τους μέσω του Shelly cloud app. Οι συσκευές, είναι προ ρυθμισμένες να αποστέλλουν συνεχώς στατιστικά δεδομένα και δεδομένα λειτουργίας, σε συγκεκριμένα topics που έχουν οριστεί από την Shelly. Εφόσον, είναι ενεργοποιημένη η επικοινωνία με τον MQTT broker μπορούν να ληφθούν αυτά τα δεδομένα από τον χρήστη ή την πλατφόρμα ελέγχου, ενώ παράλληλα, δίνεται και η δυνατότητα αποστολής μηνυμάτων εντολών στα αντίστοιχα topics ελέγχου λειτουργίας της εκάστοτε συσκευής.

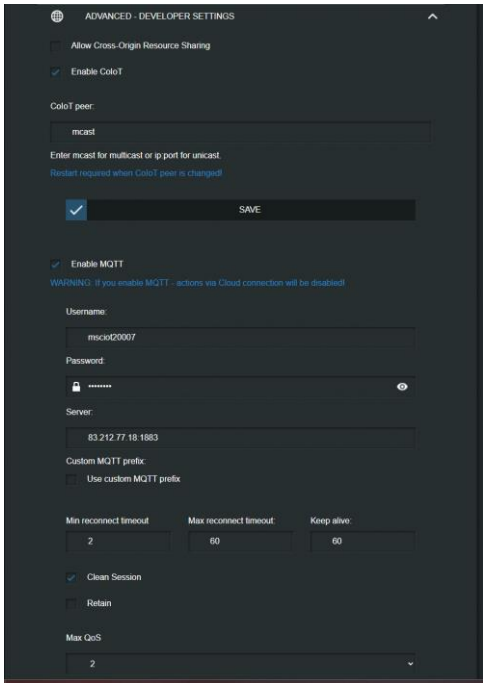
Κατά την πρώτη τους ενεργοποίηση:

1. Πραγματοποιείται η σύνδεση τους στο δίκτυο WiFi της οικίας και την cloud εφαρμογή της Shelly βάση του επίσημου οδηγού εγκατάστασης (Εικόνα 2.28).[32]



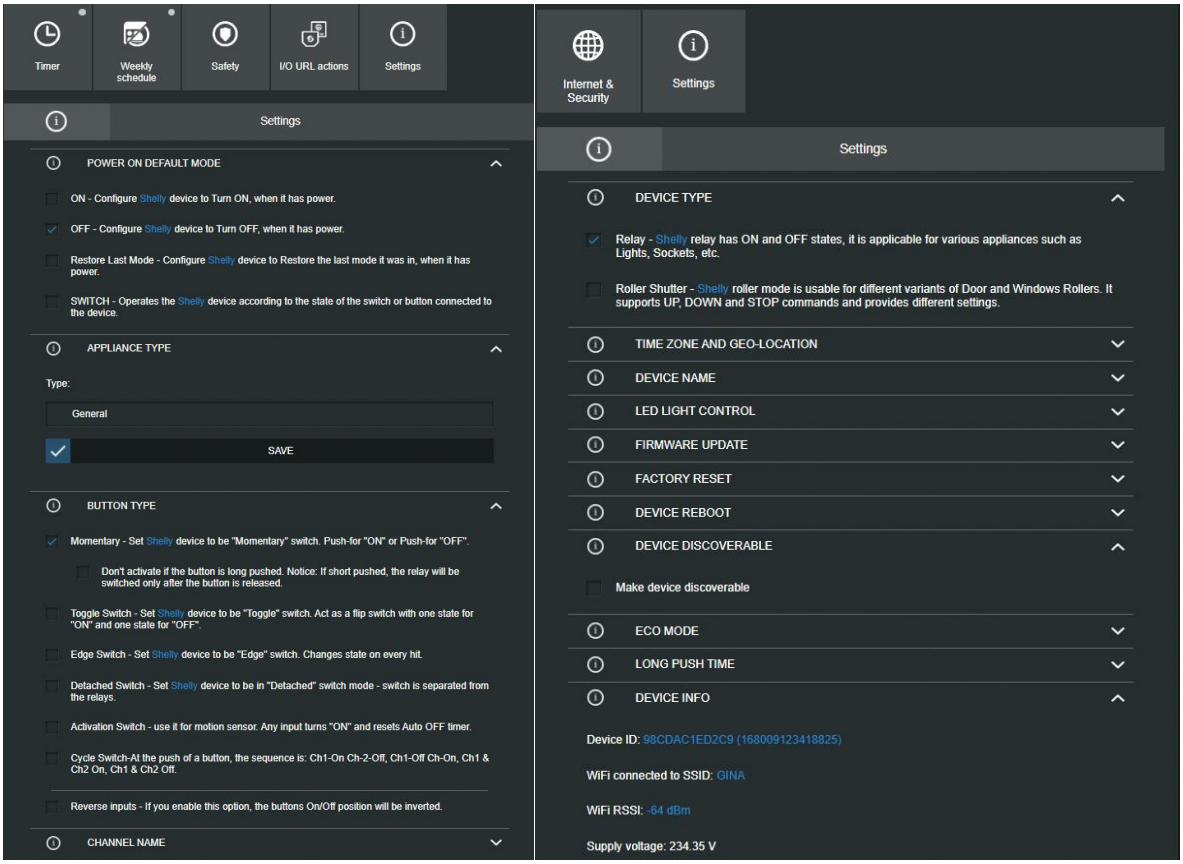
Εικόνα 2.28 Στιγμιότυπα από αρχική ενεργοποίηση και εγκατάσταση ενός Shelly στο Shelly cloud App.[31]

2. Στη συνέχεια, αφού εντοπιστεί η τοπική IP από την εφαρμογή της Shelly, εισάγεται στον browser.
3. Σε αυτή την διεύθυνση, επιλέγεται το μενού «ADVANCED-DEVELOPER SETTINGS» από την καρτέλα «Internet & Security» ώστε να ενεργοποιηθεί η επικοινωνία με τον MQTT broker, να εισαχθούν τα απαραίτητα διαπιστευτήρια και οι επιθυμητές ρυθμίσεις.



Εικόνα 2.29 Ρυθμίσεις MQTT για τα Shelly μέσω Browser

4. Μέσω αυτής της διεύθυνσης, είναι δυνατή και περαιτέρω παραμετροποίηση των Shelly. Οι επιλογές αφορούν την διαχείριση των εξόδων του, το δίκτυο, τον χρονικό προγραμματισμό λειτουργίας και άλλα πρωτόκολλα επικοινωνίας.



Εικόνα 2.30 Επιπλέον επιλογές παραμετροποίησης Shelly

Στις επόμενες ενότητες θα γίνει μία σύντομη περιγραφή των χαρακτηριστικών κάθε συσκευής και που εγκαταστάθηκε η κάθε μία.(Εικόνα 2.31)

Πίνακας 2.6 Σύγκριση Shelly 1PM, Shelly 2.5 και Shelly EM[31]

Προϊόν	**Shelly 1PM**	**Shelly 2.5 - CE + UL**	**Shelly EM + 50A Clamp**
Εικόνα			
Τιμή	15.48€	23.88€	50.28€
Βάρος	0.06kg	0.06kg	0.14kg
Χαρακτηριστικά λειτουργίας			
Τροφοδοσία AC	110-230V ±10%, 50/60Hz		
Μέγιστο φορτίο	16A	10A per channel	max 2A μόνο για όπλιση ρελέ ράγας.
Τροφοδοσία DC	24 – 60V	24 – 60V	Όχι
Έξοδοι	1 Channel	2 Channel	2 Channel
Ειδικές λειτουργίες			
Λειτουργία ρολών	Όχι	Ναι	Όχι
Προστασία θερμοκρασίας συσκευής	Ναι	Ναι	Όχι
Προστασία υπερφόρτωσης	Ναι	Ναι	Όχι
Καταμέτρηση Ισχύος	Ναι		
Άλλα χαρακτηριστικά			
Θερμοκρασία λειτουργίας	0°C to + 40 °C		
Κατανάλωσης συσκευής	< 1 W		
Έξυπνο On/Off	Ναι		
Τοπικός και απομακρυσμένος έλεγχος	Ναι		
Λειτουργία Δύσης/Ανατολής ηλίου	Ναι		
Εβδομαδιαίος προγραμματισμός	Ναι		
Πιστοποίηση UL	Όχι	Ναι	Όχι
Συνδεσιμότητα			
Ασύρματη	Ναι		
Πρωτόκολλο Wi-Fi	802.11 b/g/n		
Wi-Fi RF	2412 - 2484 MHz		
Ισχύς εκπομπής Wi-Fi	1mW		
Εμβέλεια Wi-Fi	έως 50 m σε εξωτερικούς χώρους 30 m σε εσωτερικούς (εξαρτάται από τα δομικά υλικά του κτιρίου)		
Διαστάσεις			
Μέγεθος	41 x 36x 17mm	42x 36 x 14 mm	42x 36 x 14 mm

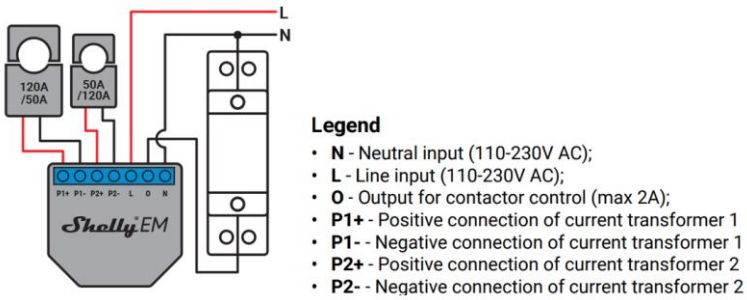
ID συσκευής	Μοντέλο	Τοπική IP	Όνομα συσκευής	Δωμάτιο
C45BBE6C4D01	Shelly EM+ 50A clamp	192.168.1.111	Total power meter	Livingroom
98CDAC2FC1A7	Shelly PM1	192.168.1.112	Heater	Livingroom
D8BFC01A17F0	Shelly PM1	192.168.1.113	Heater	Dinningroom
98CDAC1EDDB0	Shelly PM1	192.168.1.114	Heater	Office
98CDAC1F4E69	Shelly PM1	192.168.1.115	Water heater	Kitchen
D8BFC019B746	Shelly PM1	192.168.1.116	Heater	Bedroom
98CDAC1E50BC	Shelly 2.5	192.168.1.117	Lights	Outdoors

Εικόνα 2.31 Σύνολο εγκατεστημένων Shelly

2.3.1 Shelly EM

Εικόνα 2.32 Shelly EM

Το Shelly EM (energy meter), είναι μία συσκευή η οποία χρησιμοποιήθηκε στη παρούσα υλοποίηση, για την καταμέτρηση της συνολικής κατανάλωσης της οικίας. Έχει δυνατότητα καταμέτρησης δεδομένων, σχετικά με τη ποιότητα ενός δικτύου ηλεκτρικής ενέργειας μίας φάσεως. Αυτό επιτυγχάνεται, καθώς η συσκευή αυτή μπορεί να κάνει χρήση έως και δύο επαγωγικών αμπεροτσιμπίδων, σε συνδυασμό με την καταμέτρηση της τάσης του δικτύου στο οποίο έχει εγκατασταθεί μέσω της εισόδου τροφοδοσίας της. Η χρήση επαγωγικών αμπεροτσιμπίδων, προσφέρει ένα επίπεδο προστασίας στην ίδια την συσκευή αλλά και στο δίκτυο παροχής ηλεκτρικού ρεύματος, καθώς δεν χρειάζεται να γίνει καμία επέμβαση στους αγωγούς της κεντρικής παροχής ηλεκτρικής ενέργειας της οικίας. Επιπλέον, το Shelly EM διαθέτει προστασίες υπερκατανάλωσης και υπερφόρτωσης, καθώς διαθέτει έξοδο για τον έλεγχο ρελέ ισχύος, δίνοντας έτσι την δυνατότητα διαχείρισης κομβικών σημείων μιας ηλεκτρολογικής εγκατάστασης ή συσκευών. Στην παρούσα υλοποίηση, δεν αξιοποιήθηκε αυτή η επιλογή καθώς το Shelly EM λόγο του μικρού του μεγέθους τοποθετήθηκε εντός του κεντρικού ηλεκτρολογικού πίνακα της οικίας σε συνδυασμό με μία αμπεροτσιμπίδα των 50A. [33]

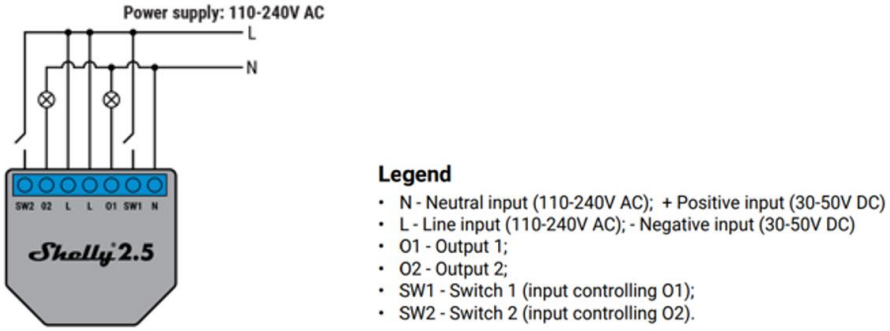


Εικόνα 2.33 Διάγραμμα διασύνδεσης Shelly EM[33]

2.3.2 Shelly 2.5

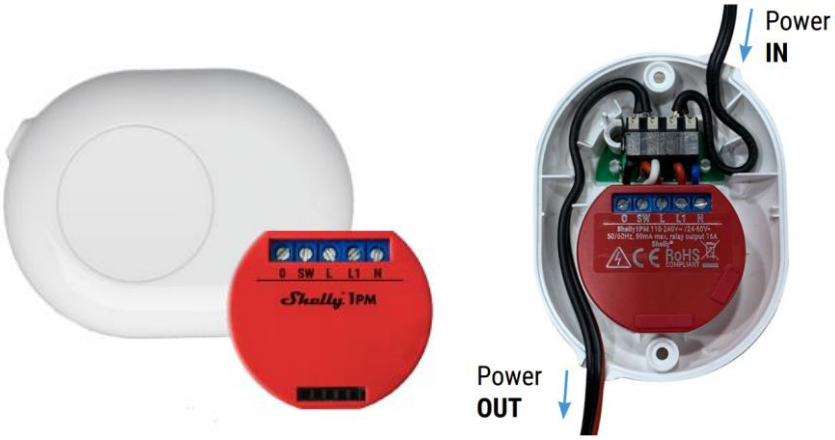
Εικόνα 2.34 Shelly 2.5

Το Shelly 2.5, χρησιμοποιήθηκε στη παρούσα υλοποίηση για τον έλεγχο των εξωτερικών φώτων της οικίας, για αυτό και τοποθετήθηκε πίσω από τον διακόπτη λειτουργίας τους, αξιοποιώντας μόνο το ένα από τα δύο κανάλια εξόδου. Η συσκευή είναι σχεδιασμένη για τοποθέτηση σε τερματικά σημεία ενός μονοφασικού δικτύου, όπως πίσω από πρίζες ή διακόπτες για τον έλεγχο διαφόρων οικιακών συσκευών. Κάθε Shelly 2.5 μπορεί να ελέγξει ξεχωριστά μέχρι δύο φορτία, ενώ ταυτόχρονα για το κάθε φορτίο μπορεί να καταγράψει μετρήσεις κατανάλωσης ηλεκτρικής ενέργειας, ρεύματος και τάσης. Ένα ιδιαίτερο χαρακτηριστικό αυτής της συσκευής είναι ότι μπορεί να τεθεί σε λειτουργία «Roller» που σημαίνει ότι μπορεί να ελέγξει ηλεκτρικά ρολά ή ηλεκτρικά συστήματα για κουρτίνες.[32]



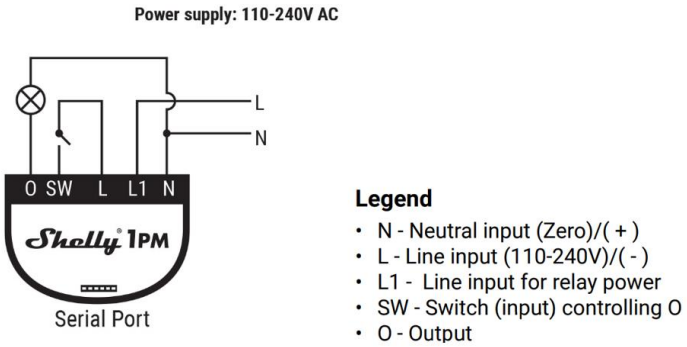
Εικόνα 2.35 Διάγραμμα διασύνδεσης Shelly 2.5 με τροφοδοσία 230V AC[32]

2.3.3 Shelly 1PM



Εικόνα 2.36 Shelly 1PM και Shelly Button

Το Shelly 1PM χρησιμοποιήθηκε στη παρούσα υλοποίηση, για τον έλεγχο των θερμαντικών σωμάτων της οικίας καθώς και για τον έλεγχο του ηλεκτρικού θερμοσίφωνα . Η συσκευή είναι σχεδιασμένη για τοποθέτηση σε τερματικά σημεία ενός μονοφασικού δικτύου. Ένα Shelly 1PM μπορεί να ελέγξει ένα μεγάλο φορτίο και αντίστοιχα να καταγράψει μετρήσεις κατανάλωσης ηλεκτρικής ενέργειας, ρεύματος και τάσης. Παρόλο που το Shelly 1PM έχει τις προδιαγραφές για να συνδεθεί και να ελέγξει απευθείας μέσω των επαφών του τον ηλεκτρικό θερμοσίφωνα, για επιπλέον ασφάλεια, οι έξοδοι του οδηγούν ένα ρελέ ράγας. Ως εκ τούτου το ρελέ ράγας οπλίζεται και αφοπλίζεται με εντολή από το Shelly και αντίστοιχα το ρελέ ελέγχει τον θερμοσίφωνα. Η ηλεκτρολογική διάταξη ελέγχου του ηλεκτρικού θερμοσίφωνα που περιγράφηκε εγκαταστάθηκε σε ξεχωριστό υπό-πίνακα που βρίσκεται σε εξωτερικό βοηθητικό χώρο της οικίας. Τέλος στα θερμαντικά σώματα, το Shelly 1PM τοποθετήθηκε πάνω στο καλώδιο τροφοδοσίας τους κάνοντας χρήση της ειδικής θήκης Shelly Button το οποίο ενσωματώνει μπουτόν, καθώς δεν υπήρχε αρκετός χώρος εντός των κουτιών των ρευματοδοτών.



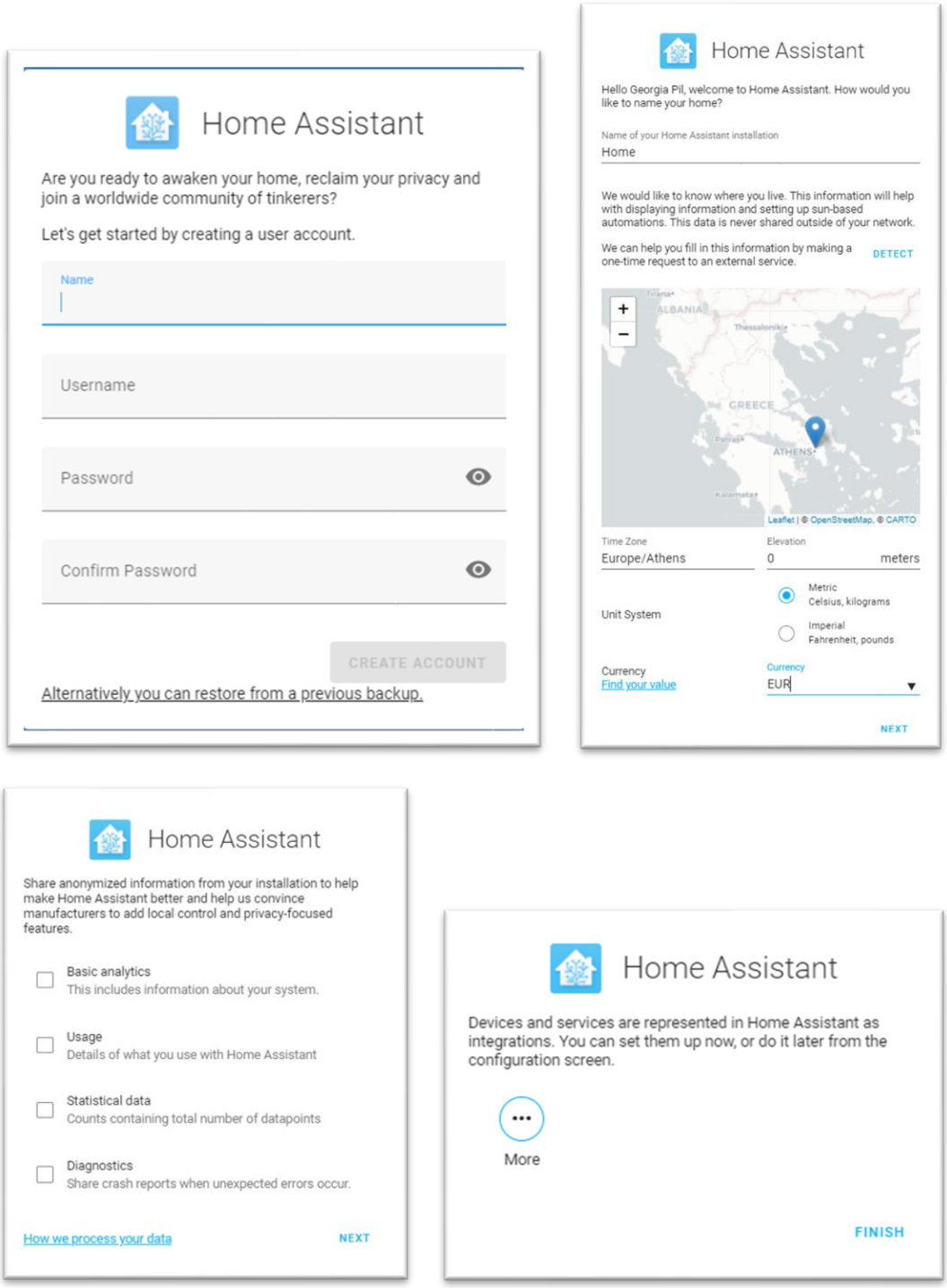
Εικόνα 2.37 Διάγραμμα διασύνδεσης Shelly 1PM

ΚΕΦΑΛΑΙΟ 3: Το Home Assistant

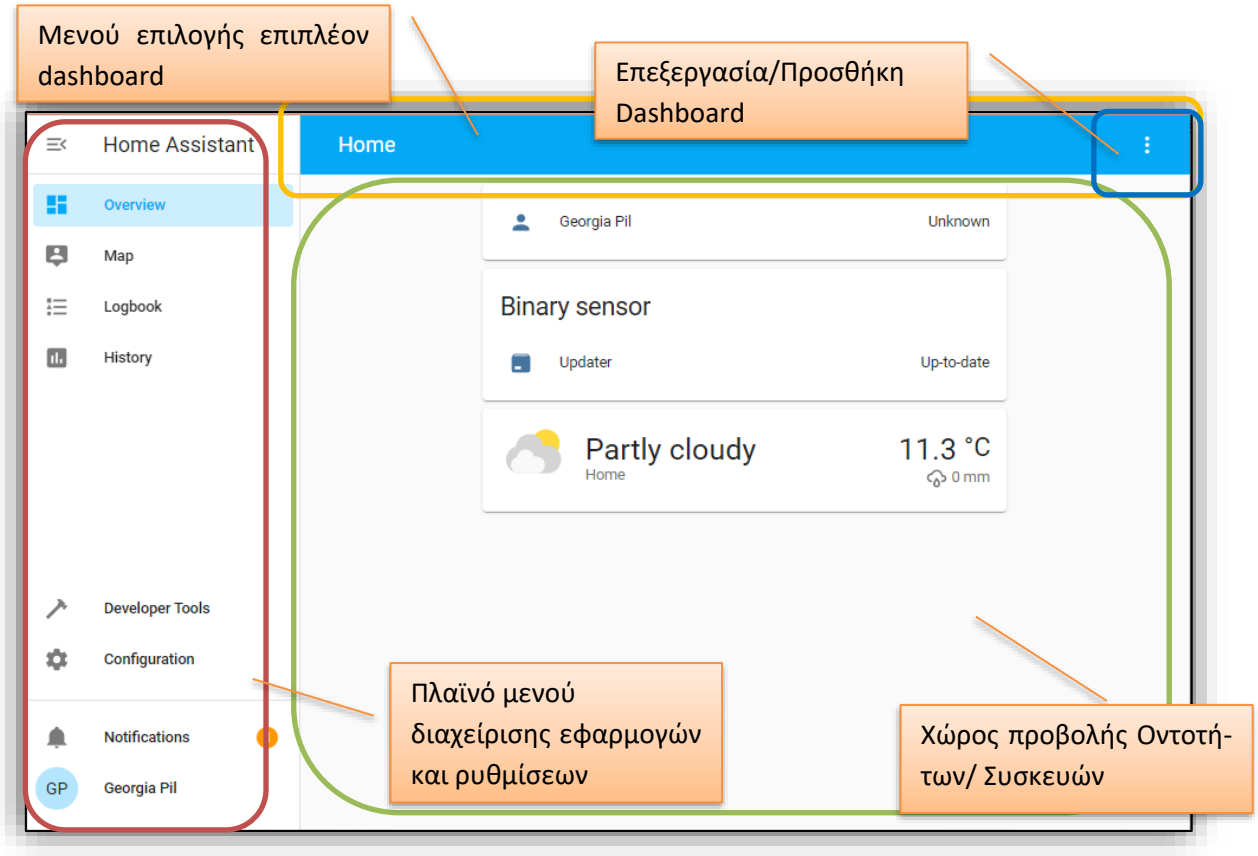
Στο κεφάλαιο αυτό, θα γίνει αναλυτική παρουσίαση της πλατφόρμας (PaaS) ανοιχτού κώδικα Home Assistant. Στην έρευνα που υλοποιήθηκε, για την επιλογή της κατάλληλης πλατφόρμας βρέθηκαν και άλλες εξίσου επιτυχημένες πλατφόρμες στον χώρο των IoT, όπως το Node Red. Το Node red είναι μία εξίσου καλή πλατφόρμα για υλοποίηση κεντρικού πίνακα ελέγχου(dashboard) και αυτοματισμών και δοκιμάστηκε στα πλαίσια της επιλογής της κατάλληλης πλατφόρμας. Παρόλα αυτά δεν επιλέχθηκε καθώς ο τρόπος υλοποίησης των αυτοματισμών γίνεται σε προγραμματιστικό περιβάλλον που χρησιμοποιεί «Κόμβους» και συνδέσεις ροών. Ο τρόπος διασύνδεσης τόσο πολλών συσκευών, απαιτούσε πολλές και περίπλοκες διαμορφώσεις του προγραμματιστικού περιβάλλοντος του, είτε για την προβολή ενδείξεων λειτουργίας είτε για την υλοποίηση αυτοματισμών. Χωρίς αυτό να σημαίνει ότι το Node Red δεν μπορούσε να ανταπεξέλθει, το Home assistant φαίνεται να έχει πιο φιλικό γραφικό περιβάλλον, με πιο εύκολη υλοποίηση αυτοματισμών, ενώ το περιεχόμενο του εμπλουτίζεται συνεχώς με επιπλέον πρόσθετα (integrations) τα οποία ανανεώνονται από μία μεγάλη κοινότητα προγραμματιστών και λάτρεις του χώρου. Το Node red, παρόλα αυτά, μπορεί να προστεθεί σαν βοηθητική πλατφόρμα εντός του Home Assistant και να συνεργαστεί με αυτό σε άλλες εργασίες. Το Home assistant είναι μία πλατφόρμα, που υποστηρίζει μεγάλο πλήθος συσκευών και μπορεί να τις ενσωματώσει όλες μαζί υλοποιώντας ένα ενιαίο δίκτυο επικοινωνίας μεταξύ τους. Συνεργάζεται, με όλα τα γνωστά πρωτόκολλα επικοινωνίας IoT και μπορεί να υλοποιήσει εύκολα, αυτοματισμούς μεταξύ τους.

3.1 Γραφικό Περιβάλλον

Το Home assistant, όπως προαναφέραμε, εγκαταστάθηκε στον Cloud Virtual Machine του ~Okeanos-knossos με χρήση Docker containers (βλ. 2.1.4 Εγκατάσταση Home Assistant (PaaS)) και αυτό διότι το VM έχει εγκατεστημένο το λειτουργικό σύστημα Ubuntu, το οποίο δεν υποστηρίζονται από την Supervised μέθοδο. Η πλατφόρμα είναι προσβάσιμη μέσω της διεύθυνσης http://83.212.77.18:8123/ η οποία καταχωρείται σε έναν web browser. Την πρώτη φορά που ο χρήστης αποκτά πρόσβαση στο Home Assistant, μετά την εγκατάσταση, του ζητείται να δημιουργήσει ένα λογαριασμό πρόσβασης εισάγοντας τα στοιχεία ταυτοποίησης που επιθυμεί (Εικόνα 3.1). Ακολουθώντας τα βήματα μπορεί να καταχωρήσει την τοποθεσία του και κάποιες συσκευές που μπορεί να εντοπίστηκαν στο οικιακό δίκτυο. Στην περίπτωση αυτή δεν εντοπίστηκε κάτι καθώς το Home Assistant δεν βρίσκεται στο ίδιο τοπικό δίκτυο με τις συσκευές αλλά στο Cloud VM. Στη συνέχεια δημιουργείται αυτόματα ο πρώτος πίνακας ελέγχου Dashboard οποίος περιλαμβάνει κάποιες βασικές οντότητες/συσκευές
.



Εικόνα 3.1 Οθόνη δημιουργίας λογαριασμού χρήστη, πρώτων ρυθμίσεων και νέων συσκευών



Εικόνα 3.2 Αρχικός Πίνακας Ελέγχου (Dashboard)

Το γραφικό περιβάλλον του Home Assistant, είναι πολύ φιλικό στον χρήστη. Όπως φαίνεται και στην Εικόνα 3.2, στα αριστερά είναι το κεντρικό μενού από το οποίο ο χρήστης μπορεί να επιλέξει κάποια επιμέρους εφαρμογή, είτε να χρησιμοποιήσει κάποια εργαλεία που αφορούν τους προγραμματιστές (Developers Tools), είτε να παραμετροποιήσει διάφορες ρυθμίσεις. Στο μενού Overview, προβάλλεται ο κεντρικός πίνακας ελέγχου που με την σειρά του αποτελείται από ένα μενού και το κεντρικό παράθυρο στο οποίο μπορούν να εντοπιστούν πολλαπλές οντότητες ή συσκευές. Ως οντότητα (entity), ορίζεται οποιαδήποτε λειτουργία η συσκευή αναγνωρίζεται από το Home assistant και μπορεί να υπάρξει ακόμη και μόνη της χωρίς απαραίτητα να ανήκει στο υποσύνολο οντοτήτων μίας συσκευής (device).

3.1.1 Προετοιμασία του Home Assistant

Η εγκατάσταση του Home assistant ως container, δεν μας έχει διαθέσιμες τις δυνατότητες supervisor και το Add-on. Αυτό σημαίνει πως η εγκατάσταση κάποιων πρόσθετων εφαρμογών είναι πιο περίπλοκη, αλλά όχι αδύνατη και πρέπει να γίνει με χρήση άλλων μεθόδων. Ένα εργαλείο απαραίτητο για την λήψη custom repositories που αφορούν πρόσθετα (integrations) του Home Assistant είναι το HACS (Home Assistant Community Store). Για την εγκατάσταση του εκτελέστηκαν τα παρακάτω βήματα:

1. Αποκτήθηκε πρόσβαση από το τερματικό μέσω SSH (βλ. Ενότητα 2.1)
2. Εκτελέσθηκε η παρακάτω εντολή για να αποκτηθεί πρόσβαση στα αρχεία του container του Home Assistant

```
$ docker exec -it homeassistant bash
```

3. Στην συνέχεια, εκτελέστηκε το script για την εγκατάσταση του HACS

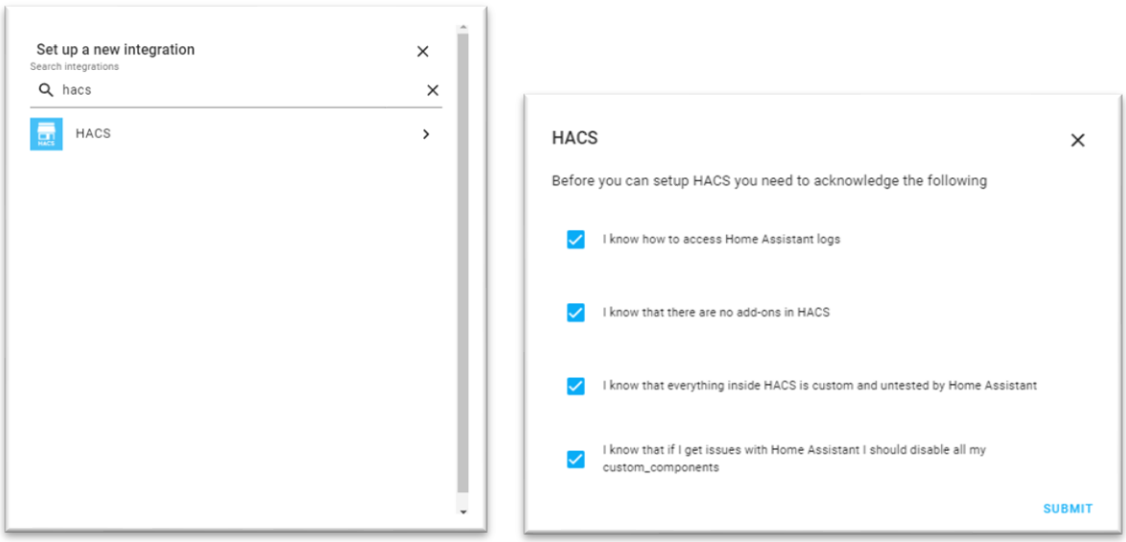
```
$ wget -q -O - https://install.hacs.xyz | bash -
```

```
user@snf-18951:~$ docker exec -it home-assistant bash
bash-5.0# wget -q -O - https://install.hacs.xyz | bash -
INFO: Trying to find the correct directory...
INFO: Found Home Assistant configuration directory at '/config'
INFO: Creating custom_components directory...
INFO: Changing to the custom_components directory...
INFO: Downloading HACS
Connecting to github.com (140.82.121.3:443)
Connecting to github.com (140.82.121.4:443)
Connecting to github-releases.githubusercontent.com (185.199.111.154:443)
saving to 'hacs.zip'
hacs.zip             100% |********************************************************************************| 77586  0:00:00 ETA
'hacs.zip' saved
INFO: Creating HACS directory...
INFO: Unpacking HACS...
INFO: Removing HACS zip file...
INFO: Installation complete.

INFO: Remember to restart Home Assistant before you configure it
bash-5.0# exit
exit
```

4. Αφού ολοκληρώθηκε η εγκατάσταση έγινε έξοδος από το container και επανεκκίνηση του container του home assistant.

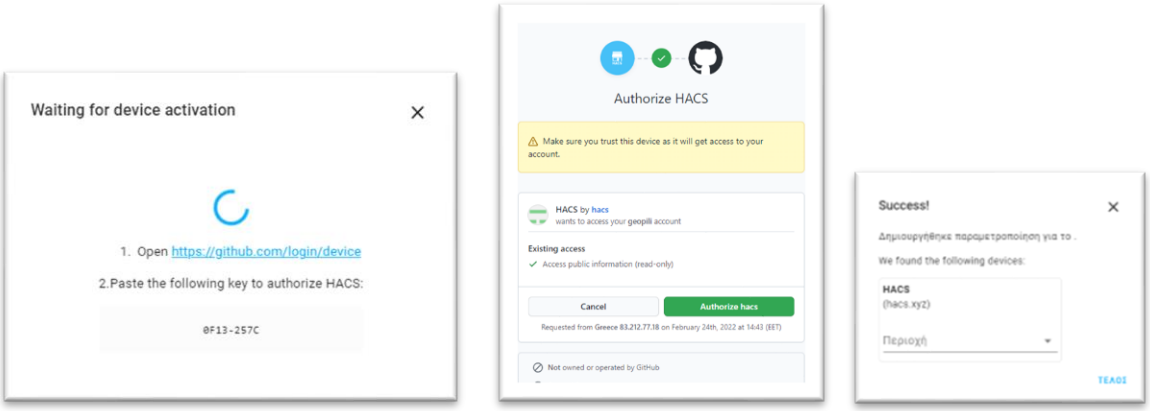
```
exit
$ docker restart home-assistant
```

5. Στη συνέχεια, στο περιβάλλον του home assistant, Από το πλαϊνό μενού επιλέχθηκαν διαδοχικά το Configuration> Devices & Services το κουμπί «Add Integration» και από την λίστα που προέκυψε επιλέχθηκε το «HACS»



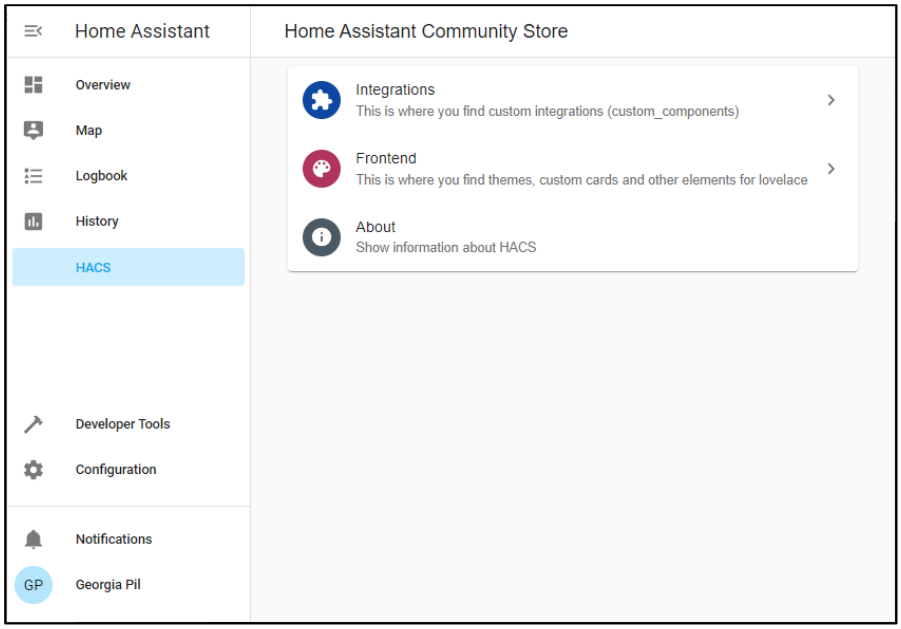
Εικόνα 3.3 Προσθήκη HACS integration

6. Για την πρώτη παραμετροποίηση του HACS, ακολουθώντας τις οδηγίες που θα προβληθούν στην οθόνη, δίνουμε δικαιοδοσία πρόσβασης με ένα ήδη υπάρχων λογαριασμό στο GitHub.



Εικόνα 3.4 Δικαιοδοσία πρόσβασης του «HACS» με το «GitHub»

7. Στο πλαϊνό μενού υπάρχει πλέον το HACS integration και επιλέγοντας το έχουμε πρόσβαση στο περιεχόμενο του.

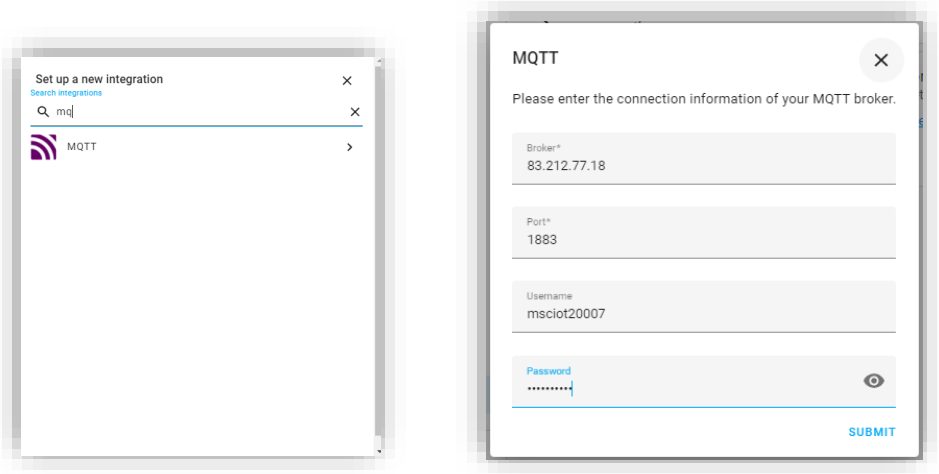


Εικόνα 3.5 Πρώτη έναρξη του HACS

3.1.2 Εισαγωγή «Mosquitto MQTT» integration

Για την επικοινωνία των συσκευών του δικτύου, που υλοποιήθηκε χρησιμοποιήσαμε το πρωτόκολλο επικοινωνίας MQTT. Οι συσκευές έχουν ρυθμιστεί έτσι ώστε να αποστέλλουν και να λαμβάνουν μηνύματα σε συγκεκριμένα topics, μέσω του Mosquitto MQTT Broker που έχει εγκατασταθεί στο Cloud VM. Για να μπορεί το Home Assistant να «διαβάζει» αυτά τα μηνύματα και στη συνέχεια να τα προβάλει στον πίνακα ελέγχου θα πρέπει να εγκατασταθεί το αντίστοιχο πρόσθετο (integration). Για την εγκατάσταση του integration ακολουθήθηκαν τα εξής βήματα:

8. Από το πλαϊνό μενού, επιλέχθηκε το Configuration>Devices & Services και στη συνέχεια το «Add Integration».
9. Από την λίστα των διαθέσιμων integrations, επιλέχθηκε το «MQTT».(Εικόνα 3.6 α))



α) β)

Εικόνα 3.6 α) Επιλογή MQTT integration β) Καταχώρηση στοιχείων πρόσβασης στον MQTT broker

10. Στη συνέχεια, καταχωρήθηκαν τα στοιχεία του MQTT broker (Εικόνα 3.6 β))που δημιουργήθηκαν στην ενότητα «2.1.3 Εγκατάσταση Mosquitto MQTT broker» και ολοκληρώθηκε η διαδικασία προσθήκης.

Εναλλακτικά, υπάρχει και η χειροκίνητη ρύθμιση του MQTT broker:

1. Γίνεται εισαγωγή στο container, από το τερματικό

```
$ docker exec -it home-assistant bash
```

2. και πληκτρολογείται η παρακάτω εντολή, έτσι ώστε να επεξεργαστούμε το αρχείο ρυθμίσεων του Home Assistant:

```
bash-5.1# vi configuration.yaml
```

3. Εκεί, προσθέτουμε τα παρακάτω πεδία και αποθηκεύουμε το αρχείο, πατώντας το πλήκτρο ESC και στην συνέχεια :wq!

configuration.yaml

```
mqtt:
  broker: "83.212.77.18"
  username: msciot20007
  password: !secret mqtt_pass
```

Εικόνα 3.7 Αρχείο configuration.yaml

4. Για επιπρόσθετη ασφάλεια, τον κωδικό πρόσβασης τον καταχωρούμε σε δευτερεύον αρχείο με όνομα secrets.yaml και πληκτρολογούμε την εντολή `bash-5.1# vi secrets.yaml` έτσι ώστε να επεξεργαστούμε το αρχείο secrets.yaml

secrets.yaml

```
# Hiding my passwords from main configuration file
mqtt_pass: [redacted]
```

Εικόνα 3.8 Αρχείο secrets.yaml

3.1.3 Εισαγωγή αισθητήρων περιβάλλοντος και ελεγκτών κλιματιστικών ως οντότητες

Στην ενότητα «2.2.3», υλοποιήθηκε ένας «Τηλεχειρισμός με υπέρυθρα σήματα και ενσωματωμένο αισθητήρα περιβάλλοντος.». Πρόκειται, για μία χειροποίητη υλοποίηση, οπότε δεν υπάρχει διαθέσιμο integration που να μπορεί να εισάγει αυτόματα τους αισθητήρες και τους ενεργοποιητές στο Home Assistant. Χάριν όμως της επικοινωνίας τους με τον MQTT broker, μπορεί να γίνει εκμετάλλευση αυτών των topics και να γίνει προβολή των μηνυμάτων δηλώνοντας το κάθε topic ως αισθητήρα. Για να επιτευχθεί αυτό, έπρεπε να γίνει επεξεργασία στο αρχείο configuration.yaml με τον εξής τρόπο:

1. Αρχικά, αποκτήθηκε πρόσβαση στο αρχείο configuration.yaml, όπως περιγράφηκε στην ενότητα «3.1.1 Προετοιμασία του Home Assistant»
2. Στη συνέχεια , προστέθηκαν οι παρακάτω γραμμές, δηλώνοντας έτσι την οντότητα ενός αισθητήρα που λαμβάνει τις μετρήσεις του από το topic που αναφέρεται.

configuration.yaml

```
sensor:
 - platform: mqtt
    device_class: "temperature"
    name: "Office Temperature"
    state_topic: "Office/temperature"
```

Εδώ δηλώνονται:
το πρωτόκολλο MQTT
Το είδος του αισθητήρα
Ένα όνομα για την οντότητα
Το topic από το οποίο θα ληφθούν οι μετρήσεις

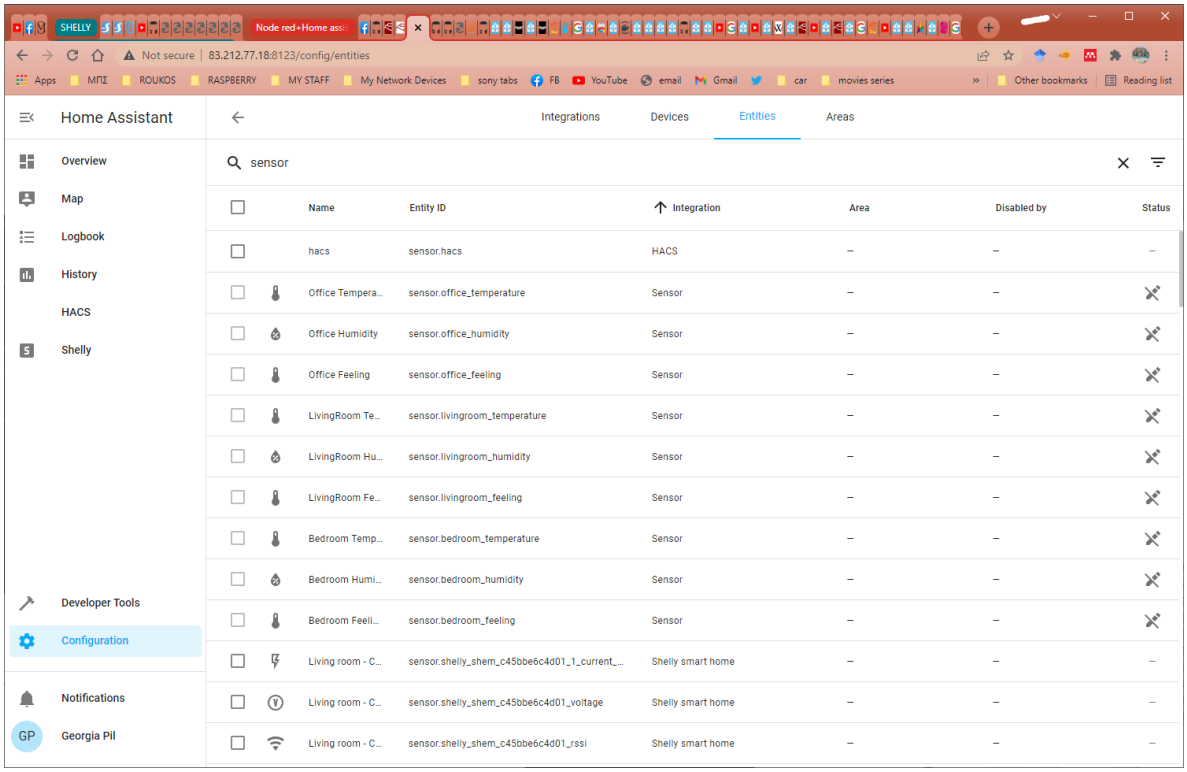
Αντίστοιχα, δηλώθηκαν οι αισθητήρες θερμοκρασίας, υγρασίας και αίσθησης θερμοκρασίας για τις τρείς υλοποιήσεις που εγκαταστάθηκαν στα δωμάτια του σαλονιού, του γραφείου και του υπνοδωματίου.

configuration.yaml

```
#Office sensors---------------------------------------------------------------------
- platform: mqtt
    device_class: "temperature"
    name: "Office Temperature"
    state_topic: "Office/temperature"
    unit_of_measurement: '°C'
  - platform: mqtt
    device_class: "humidity"
    name: "Office Humidity"
    state_topic: "Office/humidity"
    unit_of_measurement: '%'
  - platform: mqtt
    device_class: "temperature"
    name: "Office Feeling"
    state_topic: "Office/heat_index"
    unit_of_measurement: '°C'
```

Εικόνα 3.9 Παράδειγμα: Καταχώρηση αισθητήρων περιβάλλοντος για το δωμάτιο του γραφείου.

3. Αφού καταχωρήθηκαν, όλοι οι αισθητήρες περιβάλλοντος, έγινε επανεκκίνηση του Home Assistant και πηγαίνοντας στο Configuration > Devices & Services > Entities διαπιστώθηκε η εμφάνιση των αισθητήρων.



Ο έλεγχος των κλιματιστικών γίνεται με την αποστολή μηνύματος λειτουργίας σε ένα από τα διαθέσιμα topics του MQTT.

Πίνακας 3.1 Διαθέσιμα topics από αισθητήρες περιβάλλοντος και ελέγχου κλιματιστικών.

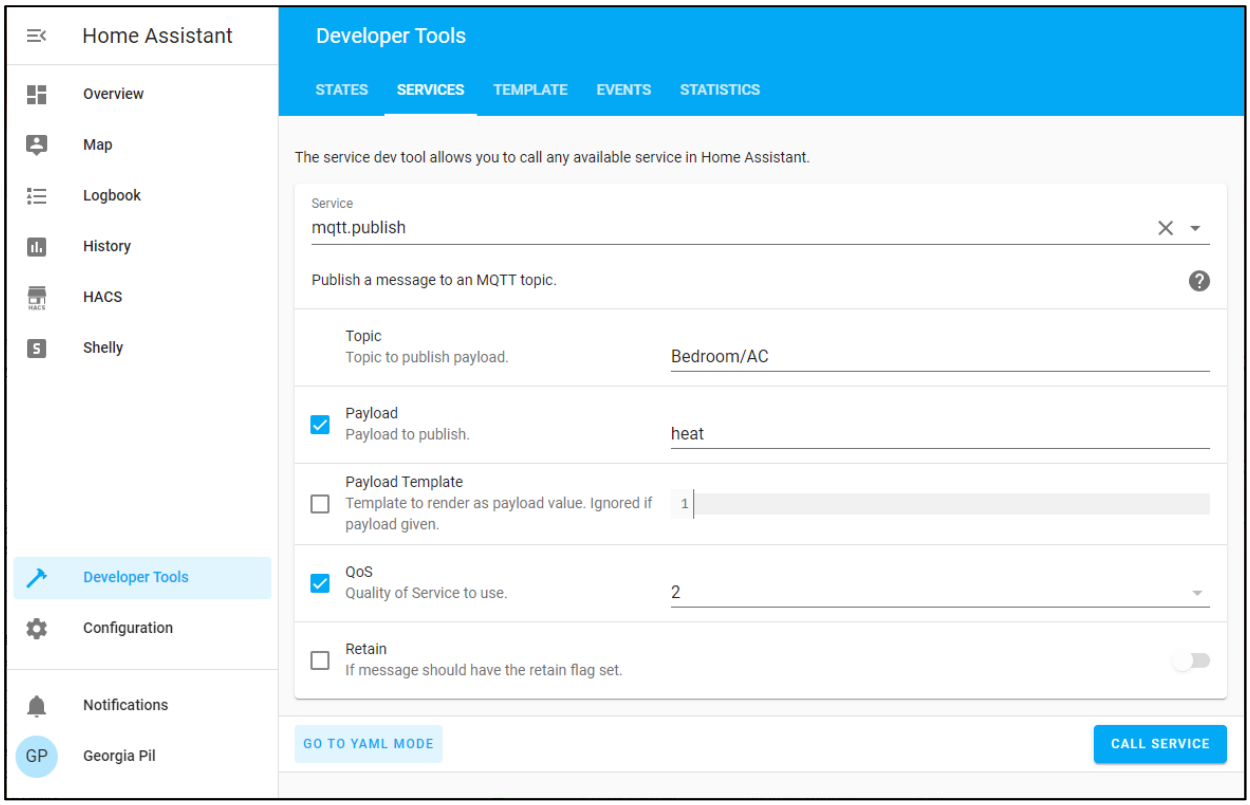
Topics	Λειτουργία/Ένδειξη
Office/temperature	Ένδειξη θερμοκρασίας για το γραφείο
Office/humidity	Ένδειξη υγρασίας για το γραφείο
Office/heat_index	Ένδειξη αίσθησης θερμοκρασίας για το γραφείο
Bedroom/temperature	Ένδειξη θερμοκρασίας για το υπνοδωμάτιο
Bedroom/humidity	Ένδειξη υγρασίας για το υπνοδωμάτιο
Bedroom/heat_index	Ένδειξη αίσθησης θερμοκρασίας για το υπνοδωμάτιο
Livingroom/temperature	Ένδειξη θερμοκρασίας για το σαλόνι
Livingroom/humidity	Ένδειξη υγρασίας για το σαλόνι
Livingroom/heat_index	Ένδειξη αίσθησης θερμοκρασίας για το σαλόνι
Bedroom/AC	Μήνυμα λειτουργίας κλιματιστικού. Επιλογές: heat, cool , off
Livingroom/AC	Μήνυμα λειτουργίας κλιματιστικού. Επιλογές: heat, cool , off

Ο έλεγχος των κλιματιστικών, έγινε αξιοποιώντας και πάλι τα topics ελέγχου των κλιματιστικών. Στο Home Assistant, υπάρχει η δυνατότητα αποστολής μηνυμάτων MQTT, μέσω του «Call service» και θα χρησιμοποιηθεί, η δομή της εντολής για την δημιουργία του κατάλληλου «πλήκτρου». Ένας τρόπος να συνταχθεί η εντολή με ημιαυτόματο τρόπο είναι:

1. να επιλεχθεί το μενού Developer Tools > Services
2. να γίνει αναζήτηση και επιλογή του Service με τίτλο MQTT:Publish.

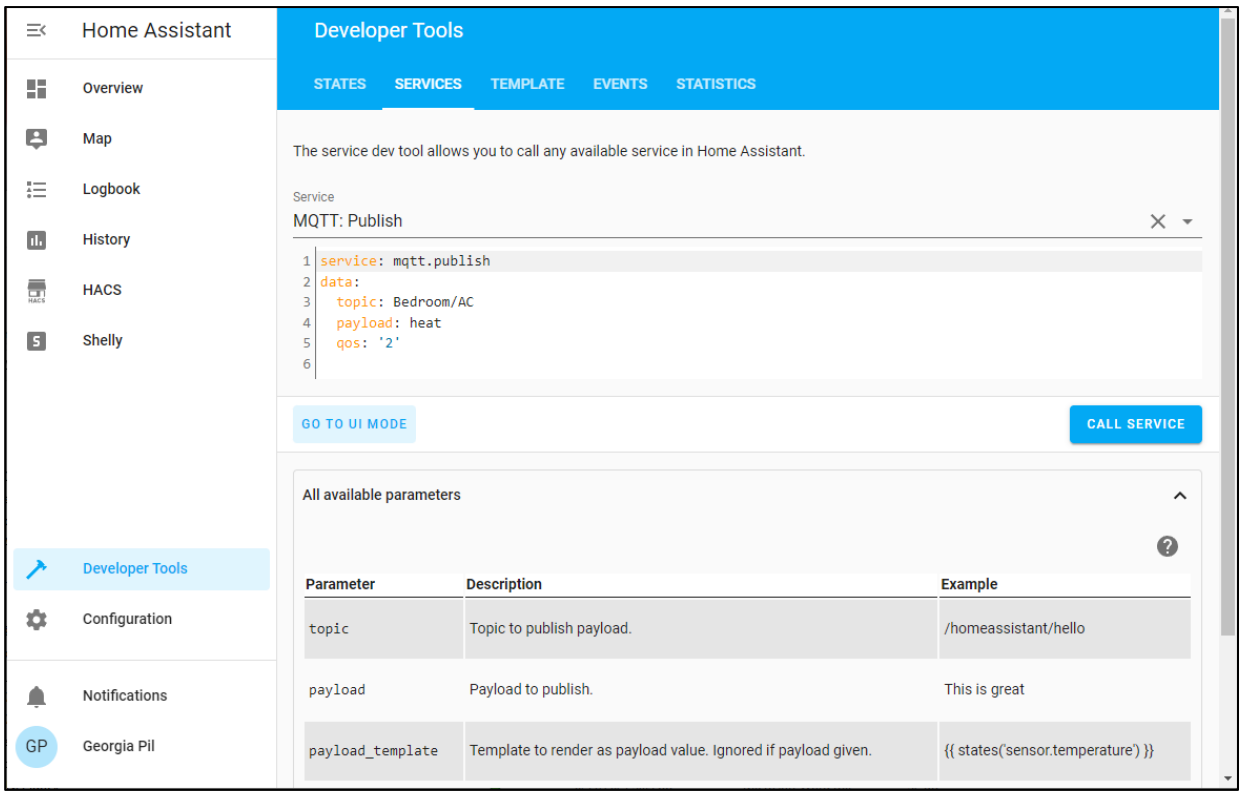
Τότε θα εμφανιστεί στην οθόνη ένα γραφικό περιβάλλον το οποίο περιμένει να του δοθούν τα πεδία του topic, payload,QoS και retain.

3. Αφού καταχωρηθούν τα επιθυμητά πεδία το μήνυμα είναι έτοιμο για αποστολή, απλά πατώντας το πλήκτρο «Call Service».



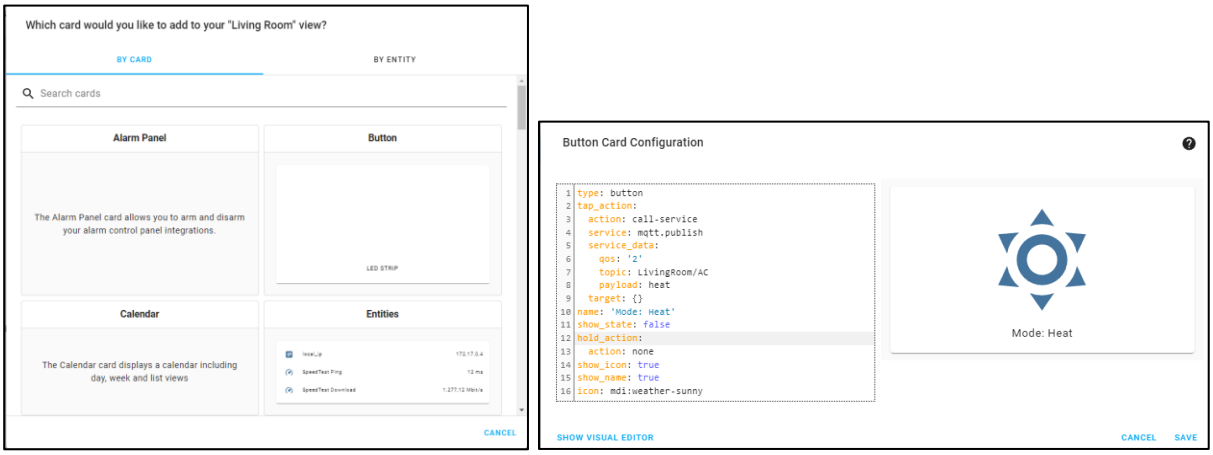
Εικόνα 3.10 Γραφικό περιβάλλον για Call Service: MQTT Publish

4. Για να μετατραπεί το γραφικό περιβάλλον σε την κώδικα YAML επιλέγεται το «GO TO YAML MODE»

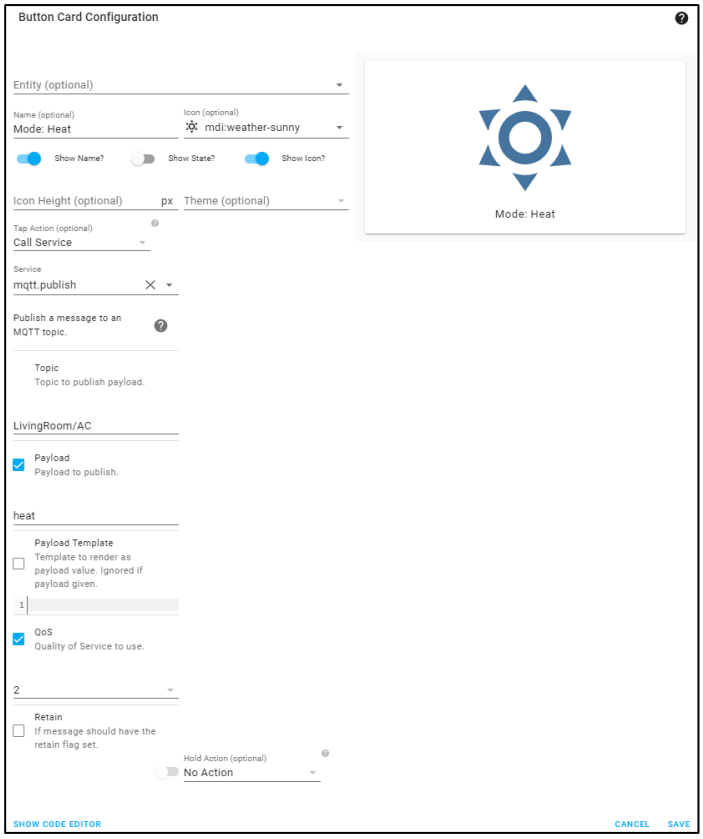


Εικόνα 3.11 Περιβάλλον κώδικα YAML για Call Service: MQTT Publish

5. Ο κώδικας του «Call Service» σε YAML, αν αντιγραφεί σε κάποιο CARD (μέσω του YAML editor) στον κεντρικό πίνακα ελέγχου, θα εμφανίσει μία επιλογή ενέργειας για αυτό το topic. Σε άλλη περίπτωση απλώς χρησιμοποιείται ο γραφικός επεξεργαστής (Visual editor)



Εικόνα 3.12 Παράδειγμα: Δημιουργίας Button Card με το YAML editor



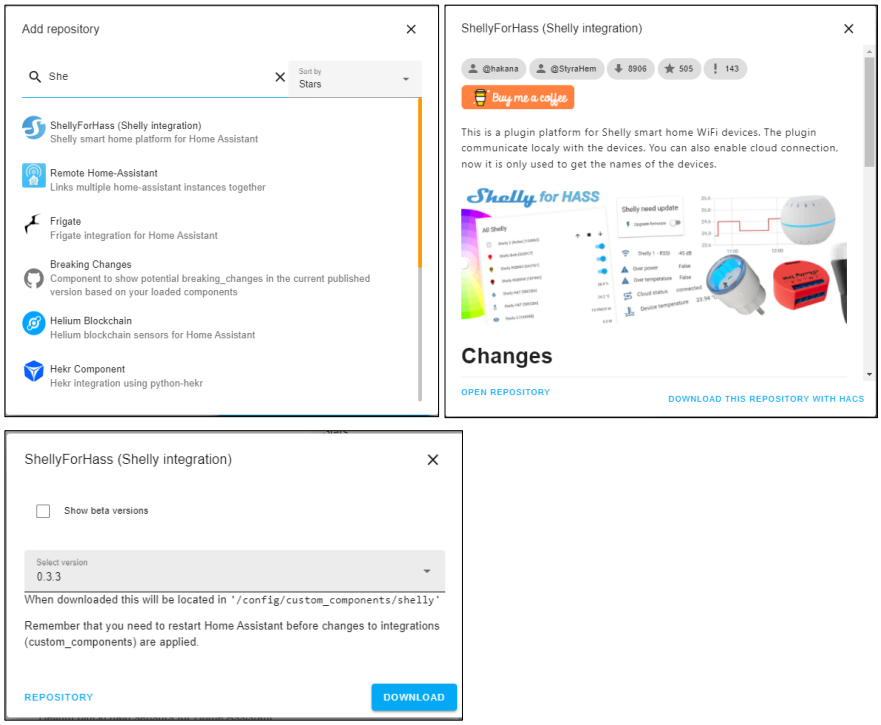
Εικόνα 3.13 Παράδειγμα δημιουργίας Button Card με το Visual editor

3.1.4 Εισαγωγή Shelly συσκευών ως οντότητες

Για να εντοπίσει το Home assistant, τις συσκευές της εταιρείας Shelly θα πρέπει να γίνει προσθήκη του αντίστοιχου integration. Τα Shelly έχουν ρυθμιστεί να επικοινωνούν με το MQTT πρωτόκολλο καθώς δεν βρίσκονται στο ίδιο δίκτυο με το Home assistant, οπότε δεν είναι εφικτή η χρήση του integration που υπάρχει στην επίσημη λίστα. Την λύση έδωσε το integration HACS, το οποίο διαθέτει το αντίστοιχο integration υποστήριξης των Shelly. Για την προσθήκη των Shelly, ακολουθήθηκαν τα εξής βήματα:

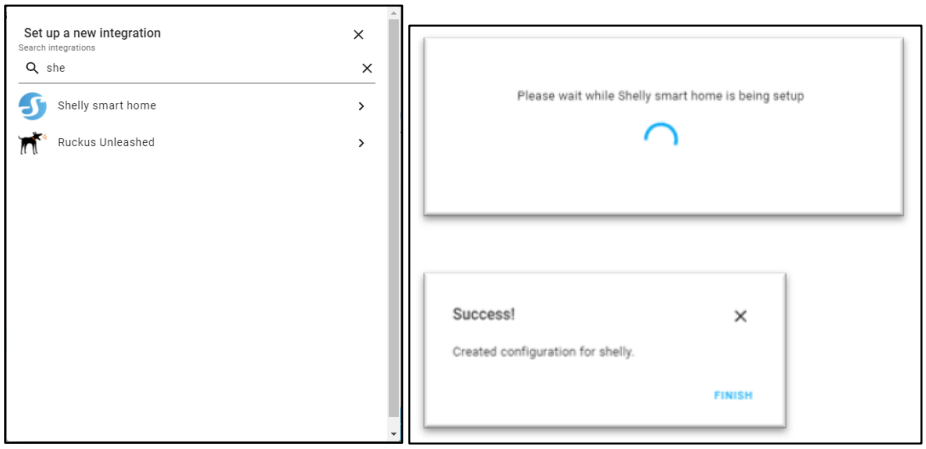
1. Από το πλαϊνό μενού, επιλέχθηκε το HACS > Integrations και στη συνέχεια το «Explore & Download Repositories».
2. Επιλέχθηκε, το «Download this repository with HACS» και στην επόμενη οθόνη επιλέχθηκε η πιο πρόσφατη έκδοση και η επιλογή «Download».
3. Στη συνέχεια έγινε επανεκκίνηση του Home Assistant από το μενού Configuration > Settings >Server Management πατώντας την επιλογή «RESTART»

4. Στην λίστα που προβλήθηκε, αναζητήθηκε το integration «ShellyForHass» και έγινε η λήψη του repository και έγινε εκ νέου επανεκκίνιση του Home assistant.



Εικόνα 3.14 Εγκατάσταση ShellyFoHass repository

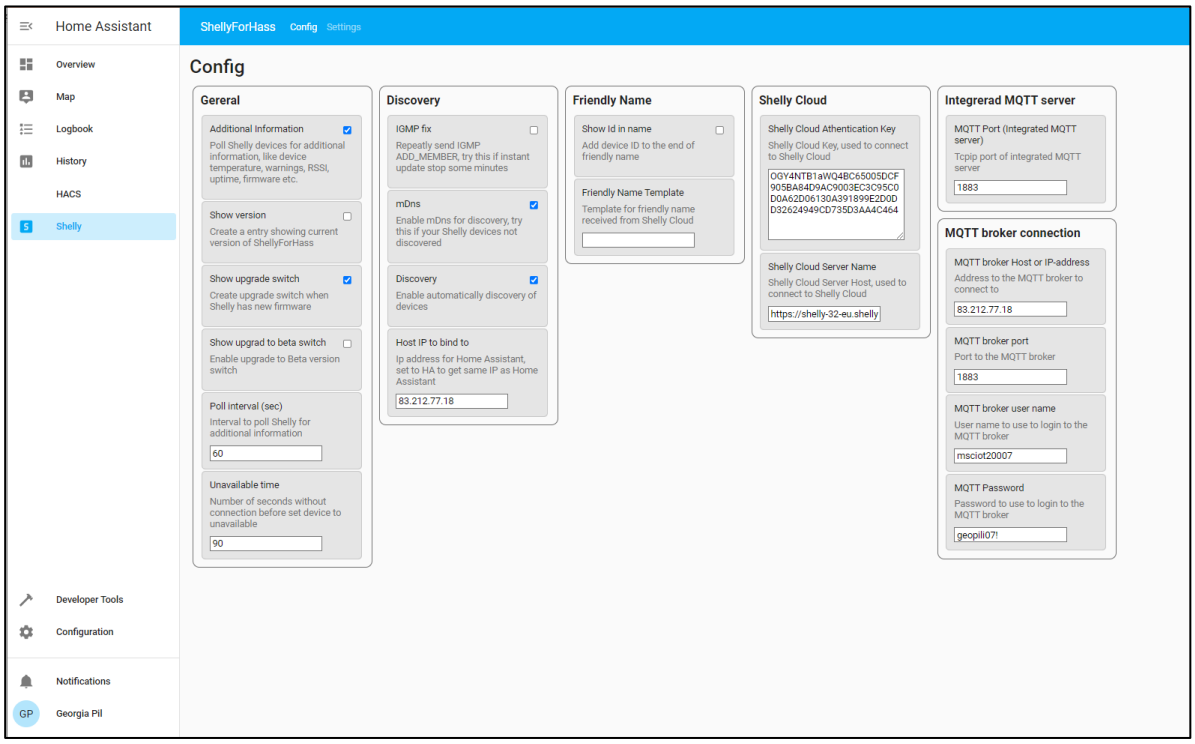
5. Μετά την επανεκκίνηση, από το πλαϊνό μενού, επιλέχθηκε το Configuration>Devices & Services και στη συνέχεια το «Add Integration». Από την λίστα που αναπτύχθηκε αναζητήθηκε και επιλέχθηκε το «Shelly smart home». Ολοκληρώνοντας με αυτό τον τρόπο την προσθήκη του integration για τα Shelly.



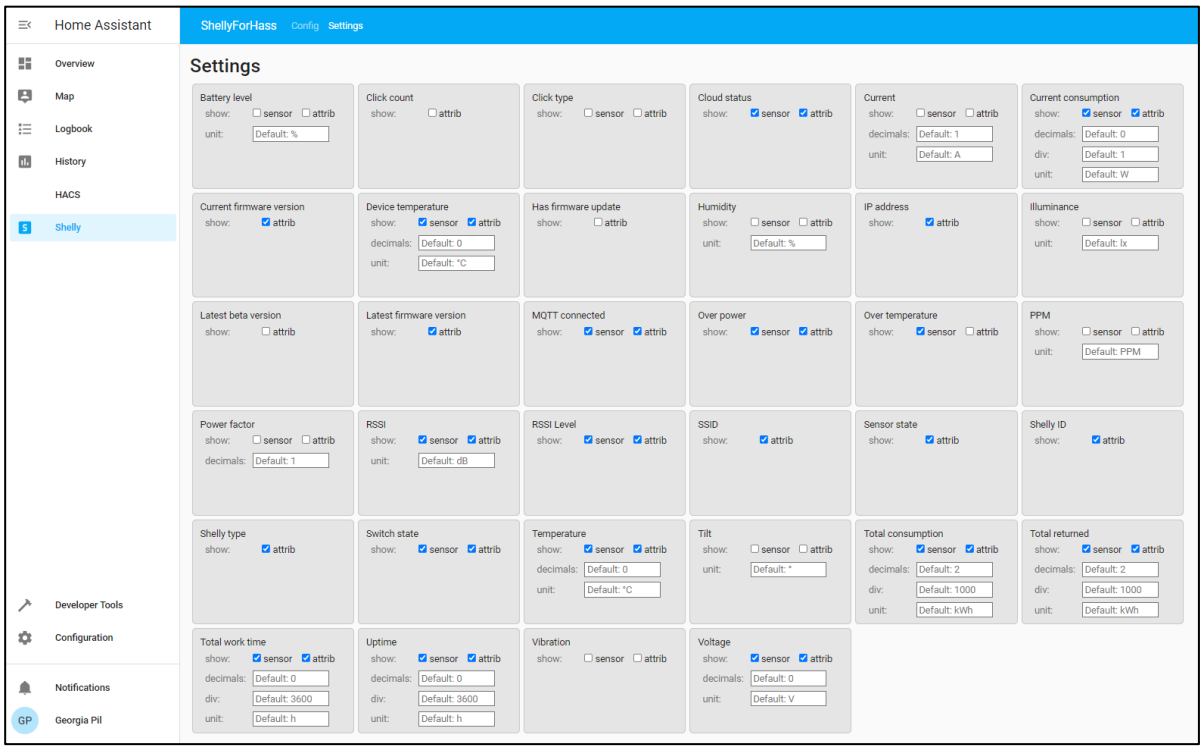
Εικόνα 3.15 Προσθήκη «Shelly smart home» integration

6. Στη συνέχεια, στο πλαϊνό μενού εμφανίζεται η εγγραφή «Shelly» στην οποία θα καταχωρήσουμε τα στοιχεία του MQTT broker και του Cloud (όταν χρησιμοποιείται) (Εικόνα 3.16) που συνεργάζεται με

τα Shelly και θα γίνουν κάποιες επιλογές από το μενού Shelly > Settings που αφορούν τα στοιχεία λειτουργίας και προβολής των συσκευών (Εικόνα 3.17).



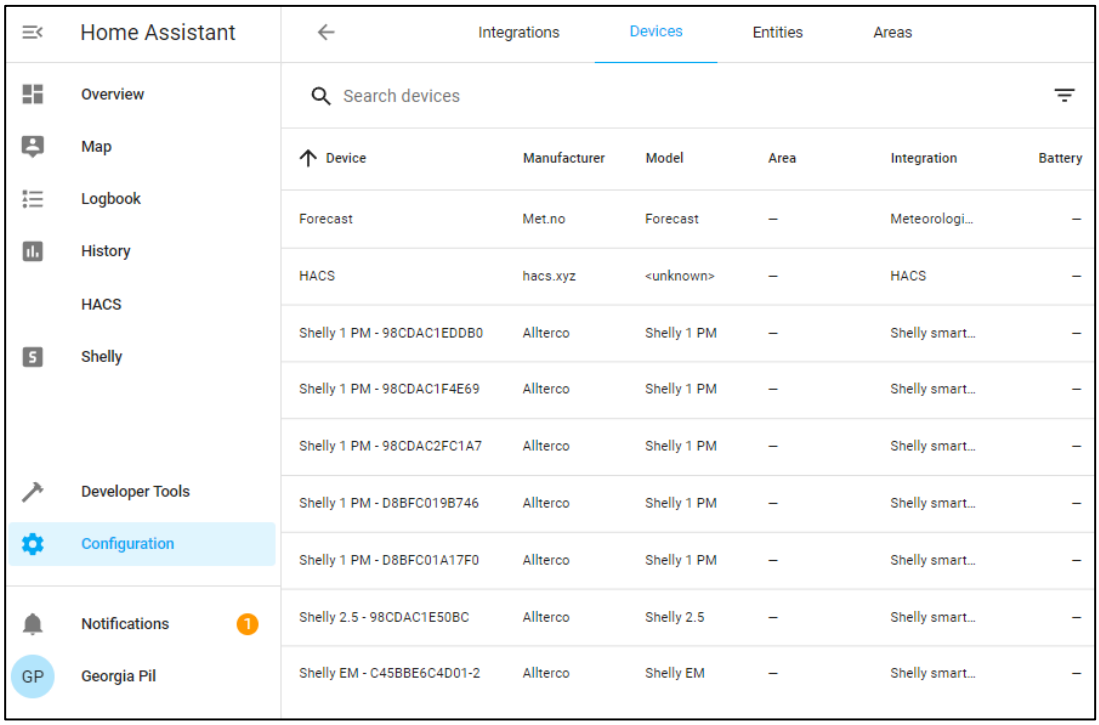
Εικόνα 3.16 Ρυθμίσεις σύνδεσης με το MQTT και το Cloud (όταν είναι ενεργό)



Εικόνα 3.17 Ρυθμίσεις προβολής αισθητήρων και άλλων στοιχείων λειτουργίας των Shelly

7. Αφού έγιναν οι απαραίτητες ρυθμίσεις έγινε επανεκκίνηση του Home Assistant από το μενού Configuration > Settings >Server Management πατώντας την επιλογή «RESTART»

8. Οι συσκευές πλέον εμφανίζονται στο μενού Configuration >Devices & Services > Devices

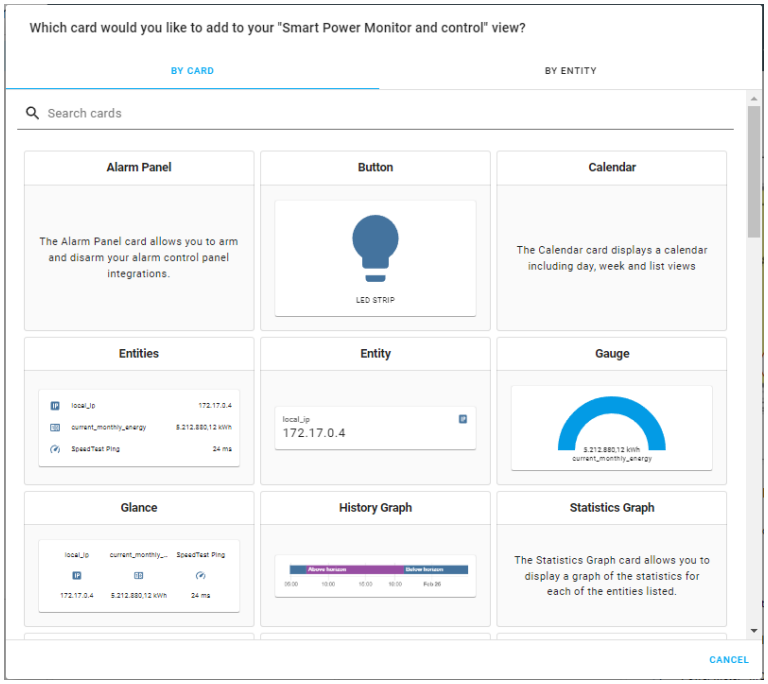


Εικόνα 3.18 Προβολή συσκευών Shelly στο Home Assistant

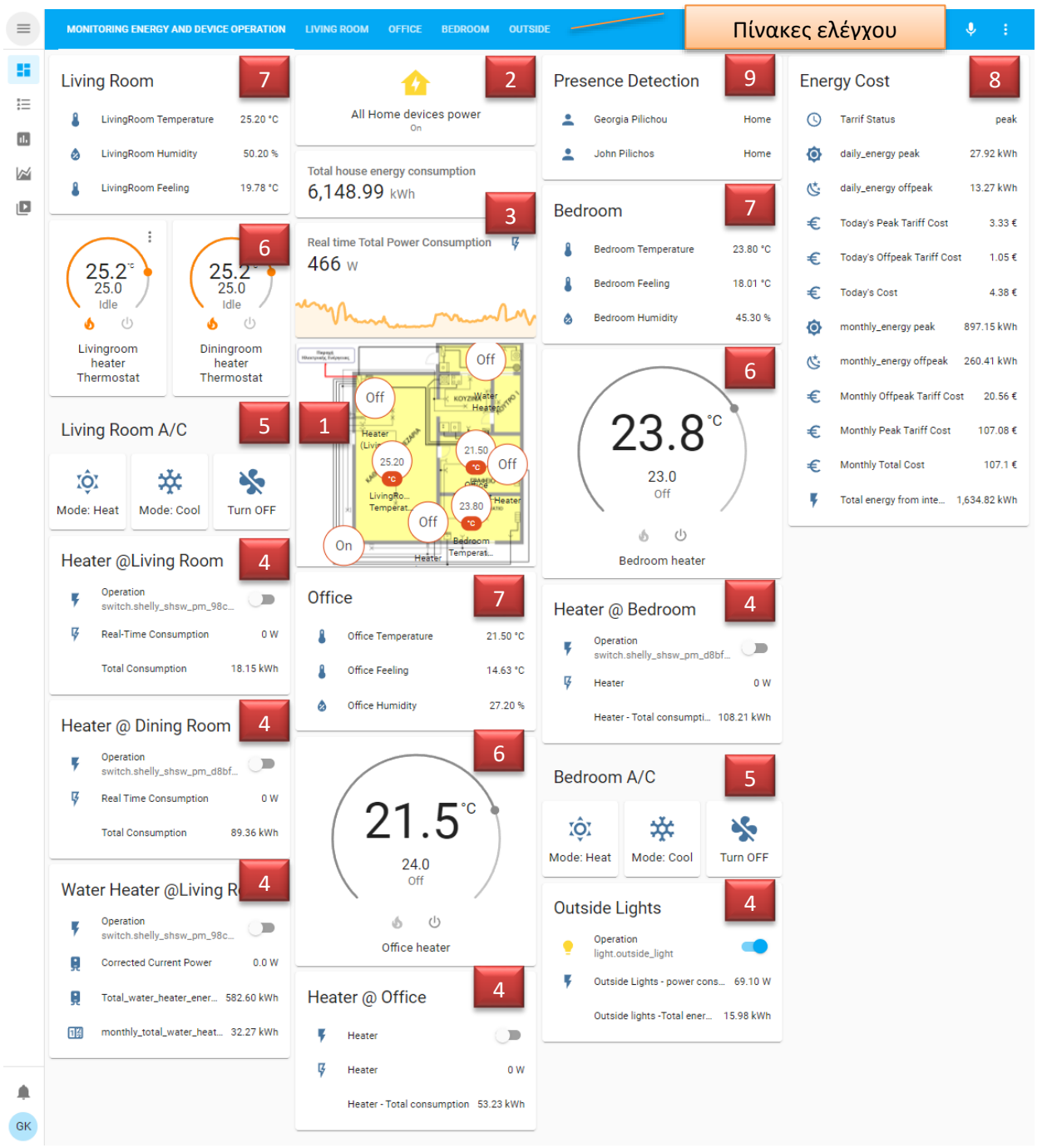
Από αυτό το μενού μπορούν να γίνουν αλλαγές που αφορούν τα ονόματα των συσκευών καθώς και να προβληθούν τα entities που τις απαρτίζουν.

3.1.5 Μορφοποίηση Πίνακα ελέγχου (Dashboard)

Στα πλαίσια της κεντρικοποίησης του ελέγχου των συσκευών, που συμμετέχουν στο δίκτυο IoT της οικίας, έπρεπε να δημιουργηθεί ένας ενιαίος πίνακας ελέγχου. Ο πίνακας αυτός, έπρεπε να δίνει στον χρήστη σαφή εικόνα της κατάστασης λειτουργίας των συσκευών, την κατανάλωση ενέργειας όπου ήταν διαθέσιμη καθώς και επιλογές ελέγχου της λειτουργίας τους. Το Home assistant, διαθέτει πληθώρα επιλογών όσων αφορά την γραφική απεικόνιση των οντοτήτων ,στον κεντρικό πίνακα πατώντας την επιλογή «Add Card», εμφανίζεται η λίστα με τις κάρτες (Εικόνα 3.19) . Κάθε φορά που επιλέγεται μία κάρτα δίνεται η δυνατότητα περαιτέρω επεξεργασίας της μέσω του YAML editor. Επίσης, οι οντότητες μπορούν να μετακινηθούν σε διάφορες θέσεις εντός του πίνακα και μπορούν να υπάρξουν πάνω από μία φορά, καθώς και να εμφανίζονται και σε άλλους πίνακες.



Εικόνα 3.19 Διαθέσιμες επιλογές απεικόνισης οντοτήτων



Εικόνα 3.20 Πίνακας ελέγχου/Dashboard Home Assistant

Στην παρούσα υλοποίηση, δημιουργήθηκαν πέντε πίνακες ελέγχου, ένας για κάθε δωμάτιο με περισσότερες πληροφορίες ανά συσκευή και ένας κεντρικός πίνακας ελέγχου που εμφανίζει το σύνολο των συσκευών μόνο με τις ενδείξεις κατανάλωσης ενέργειας και τους διακόπτες ενεργοποίησης/απενεργοποίησης. Στον κεντρικό πίνακα ελέγχου (Εικόνα 3.20), φαίνεται:

1. η κάτοψη της οικίας, με δείκτες λειτουργίας των συσκευών και τις θερμοκρασίες των δωματίων.
2. Ένα μπουτόν, που λειτουργεί σαν γενικός διακόπτης ταυτόχρονης ενεργοποίησης/απενεργοποίησης όλων των ελεγχόμενων συσκευών
3. η κατανάλωση της οικίας, σε πραγματικό χρόνο και η συνολική κατανάλωση από την έναρξη λειτουργίας του Shelly EM με μορφή αριθμού και ιστορικού γραφήματος.

4. Εμφανίζεται, η κάθε συσκευή ξεχωριστά με διακόπτη λειτουργίας, ένδειξη τρέχουσας κατανάλωσης και συνολικής καταναλωθείσας ενέργειας. Στα θερμαντικά σώματα, έχει προστεθεί και ένδειξη θερμοστάτη για αυτοματοποιημένη λειτουργία βάση θερμοκρασίας
5. Κουμπιά επιλογής λειτουργίας, για τα δύο διαθέσιμα κλιματιστικά
6. Θερμοστάτες ελέγχου λειτουργίας, των θερμαντικών σωμάτων
7. Ενδείξεις, θερμοκρασίας/υγρασίας/αίσθησης θερμοκρασίας ανά δωμάτιο
8. Μία κάρτα ενεργειακού κόστους, που περιλαμβάνει ενδείξεις ημερήσιας κατανάλωσης σε ώρες αιχμής και μη, το κόστος αυτών των καταναλώσεων αλλά και τις αντίστοιχες μηνιαίες ενδείξεις.
9. Και μία κάρτα εντοπισμού θέσης, κάποιων από τους ένοικους της οικίας

Καθώς προθέτονταν τα Shelly, στον πίνακα ελέγχου διαπιστώθηκε ότι το Shelly 2.5 δεν περιελάβανε στις οντότητες του, τον διακόπτη λειτουργίας και την κατανάλωση ενέργειας. Για να επιλυθεί αυτό το ζήτημα, έγινε εκμετάλλευση των MQTT topics, που δημιουργούνται και ενημερώνονται αυτόματα από όλα τα Shelly. Αυτά, προστέθηκαν απλώς, συμπληρώνοντας τα παρακάτω πεδία στο configuration.yaml του Home assistant.

configuration.yaml

```
#outside lights  power management------------------------
  - platform: mqtt
    name: Outside Lights - power consumption
    device_class: power
    unit_of_measurement: 'W'
    state_topic: shellies/shellyswitch25-98CDAC1E50BC/relay/0/power
  - platform: mqtt
    name: Outside Lights - energy consumption since last reboot
    device_class: energy
    unit_of_measurement: 'Wh'
    value_template: "{{ value | float / 60 | round(2) }}"
    state_topic: shellies/shellyswitch25-98CDAC1E50BC/relay/0/energy
------

------
#outside lights operation switch------------------------
light:
  - platform: mqtt
    unique_id: shelly25outside
    name: "Outside light"
    state_topic: "shellies/shellyswitch25-98CDAC1E50BC/relay/0"
    command_topic: "shellies/shellyswitch25-98CDAC1E50BC/relay/0/command"
    payload_on: 'on'
    payload_off: 'off'
    retain: false
    qos: 1
```

Εικόνα 3.21 Δημιουργία οντοτήτων μέτρησης κατανάλωσης ενέργειας και διακόπτη λειτουργίας για τα εξωτερικά φώτα

Επίσης, στο σημείο του κώδικα όπου δηλώνονται οι αισθητήρες, προστέθηκε η παρακάτω εγγραφή που μετατρέπει την ισχύ σε συνολική ενέργεια που καταναλώθηκε, κάνοντας χρήση του integration «Riemann sum integral»

configuration.yaml

```
#Outside lights Total power consumption with Riemann sum integral-integration
  - platform: integration
    source: sensor.outside_lights_power_consumption
    name: outside_lights_total_energy_consumption
    unit_prefix: k
    round: 2
```

Εικόνα 3.22 Δημιουργία μετρητή συνολικής κατανάλωσης ενέργειας με Riemann sum integral - integration

Η κάρτα, που εμφανίζει το ενεργειακό κόστος, «Energy Cost», στηρίζεται στο integration «Utility meter». Για την υλοποίηση του, χρησιμοποιήθηκε ο οδηγός που υπάρχει στην επίσημη σελίδα του Home Assistant και έχει τα εξής βήματα:

1. Αρχικά, αποκτήθηκε πρόσβαση στο αρχείο configuration.yaml όπως περιγράφηκε στην ενότητα «3.1.1 Προετοιμασία του Home Assistant»
2. Στη συνέχεια , προστέθηκαν οι παρακάτω γραμμές από το «Riemann sum integral» integration, δημιουργώντας μία οντότητα ενός αισθητήρα κατανάλωσης ενέργειας που λαμβάνει τις μετρήσεις του από τον αισθητήρα μέτρησης της τρέχουσας ισχύος του Shelly EM .

configuration.yaml (Riemann sum integral

```
sensor:
  - platform: integration
    source: sensor.shelly_shem_c45bbe6c4d01_1_current_consumption
    name: energy_pow
    unit_prefix: k
    round: 2
```

Εικόνα 3.23 Δημιουργία οντότητας υπολογισμού συνολικής κατανάλωσης ενέργειας με Riemann sum integral - integration

3. Αφού, δημιουργήθηκε η οντότητα του αισθητήρα, προστέθηκαν τα παρακάτω πεδία στο configuration.yaml, για να γίνει η ενεργοποίηση του «Utility meter».

```
# UTILITY METER ENTITIES AND SENSORS-------------
--
utility_meter:
  daily_energy:
    source: sensor.energy_pow                    ⇐ Εδώ ορίζεται ο ημερήσιος μετρητής ενέργειας.
    cycle: daily                                 ⇐ Η πηγή μετρήσεων από τον αισθητήρα του προηγούμενου
    tariffs:                                       βήματος.
      - peak                                     ⇐ Επαναφέρει τις μετρήσεις με ημερήσιο κύκλο.
      - offpeak                                  ⇐ Έχει δύο κλίμακες μετρήσεων peak/offpeak (ώρες αιχμής/
  monthly_energy:                                  αιχμής)
    source: sensor.energy_pow
    cycle: monthly                               ⇐ Αντίστοιχα ορίζεται και ο μηνιαίος μετρητής ενέργειας
    tariffs:
      - peak
      - offpeak
  current_monthly_energy:
    source: sensor.energy_pow
    cycle: monthly
```

Εικόνα 3.24 Δημιουργία ημερήσιου και μηνιαίου μετρητή ενέργειας

4. Για υπολογισμό και την εμφάνιση του κόστους της ηλεκτρικής ενέργειας σε Ευρώ, έπρεπε να δημιουργηθεί ένα Template, το οποίο δημιουργεί μία οντότητα που κάνει τον υπολογισμό κόστους με την κατάλληλη μονάδα μέτρησης. Παρακάτω, φαίνεται μέρος του κώδικα που κάνει αυτή την μετατροπή, για τον υπολογισμό του ημερήσιου κόστους σε: ώρες αιχμής ,μη αιχμής και το συνολικό ποσό αυτών των δύο.

```
configuration.yaml
#Daily Costs-----------------------------------------------------------------
  - platform: template
    sensors:
      daily_energy_cost_peak:
        unit_of_measurement: '€'
        value_template: "{{ (states('sensor.daily_energy_peak') | float * 0.11936 ) | round(2)
}}"
        friendly_name: Today's Peak Tariff Cost
  - platform: template
    sensors:
      daily_energy_cost_offpeak:
        unit_of_measurement: '€'
        value_template: "{{ (states('sensor.daily_energy_offpeak') | float * 0.07897 ) |
round(2) }}"
        friendly_name: Today's Offpeak Tariff Cost
  - platform: template
    sensors:
      daily_energy_cost_all:
        value_template: "{{ ((states('sensor.daily_energy_peak') | float *0.11936) +
(states('sensor.daily_energy_offpeak') | float *0.07897 )) | round(2) }}"
        friendly_name: Today's Cost
        unit_of_measurement:  '€'
```

Εικόνα 3.25 Δημιουργία οντοτήτων υπολογισμού κόστους ηλεκτρικής ενέργειας

Αντίστοιχα, υλοποιήθηκε ο υπολογισμός κόστους και για τις μηνιαίες μετρήσεις κατανάλωσης ενέργειας και οι ενδείξεις αυτών των μετρήσεων εντάχθηκαν στην παρακάτω κάρτα.

Εικόνα 3.26 Utility meter

Μία ακόμη ιδιαιτερότητα, εμφανίστηκε και στην απεικόνιση των μετρήσεων κατανάλωσης του ηλεκτρικού θερμοσίφωνα. Το Shelly 1PM όπως αναφέρθηκε και στην ενότητα 2.3.3 δεν ελέγχει απευθείας τον ηλεκτρικό θερμοσίφωνα αλλά μέσω ενός ρελέ ισχύος ράγας. Αυτό συνεπάγεται ότι οι μετρήσεις κατανάλωσης ενέργειας, αφορούν την κατανάλωση του ρελέ και όχι του ίδιου του ηλεκτρικού θερμοσίφωνα. Υπολογίστηκε από τις τιμές που εμφάνιζε το Shelly EM (κεντρικός μετρητής) , ότι ο ηλεκτρικός θερμοσίφωνας έχει ισχύ περίπου 3250W, οπότε αυτή η τιμή προστέθηκε σε ένα template το οποίο διορθώνει την κατανάλωση που δείχνει ο μετρητής τρέχουσας ισχύος κάθε φορά που ενεργοποιείται ο θερμοσίφωνας από ~1W σε 3250W. Παρακάτω, φαίνεται το template που χρησιμοποιήθηκε (Εικόνα 3.27) όπως επίσης και ο μηνιαίος μετρητής συνολικής κατανάλωσης ενέργειας που μετατρέπει τα W σε Wh μέσω του «Riemann sum integral» integration.

configuration.yaml

```
# Manual correction at current energy consumption measurement of Water Heater-------------
  - platform: template
    sensors:
      power_water_heater:
        friendly_name: "Water heater power consumption"
        unit_of_measurement: 'W'
        value_template: >
          {% if is_state('sensor.shelly_shsw_pm_98cdac1f4e69_current_consumption', 0) %}
            {{ states('sensor.shelly_shsw_pm_98cdac1f4e69_current_consumption') | int* 0}}
          {% else %}
            {{ states('sensor.shelly_shsw_pm_98cdac1f4e69_current_consumption') | float*
3250}}
          {% endif %}
```

Εικόνα 3.27 Διόρθωση τιμής κατανάλωσης ενέργειας ηλεκτρικού θερμοσίφωνα

```
configuration.yaml
- platform: integration
    source: sensor.power_water_heater
    name: Total_water_heater_energy_consumption
    unit_prefix: k
    round: 2
------------------------------------------------------------

------------------------------------------------------------
utility_meter:

----

  monthly_total_water_heater_energy_consumption:
    source: sensor.total_water_heater_energy_consumption
    cycle: monthly
```

Εικόνα 3.28 Συνολική κατανάλωσης ενέργειας με Riemann sum integral - integration και μηνιαίος μετρητής συνολικής κατανάλωσης (utility meter)

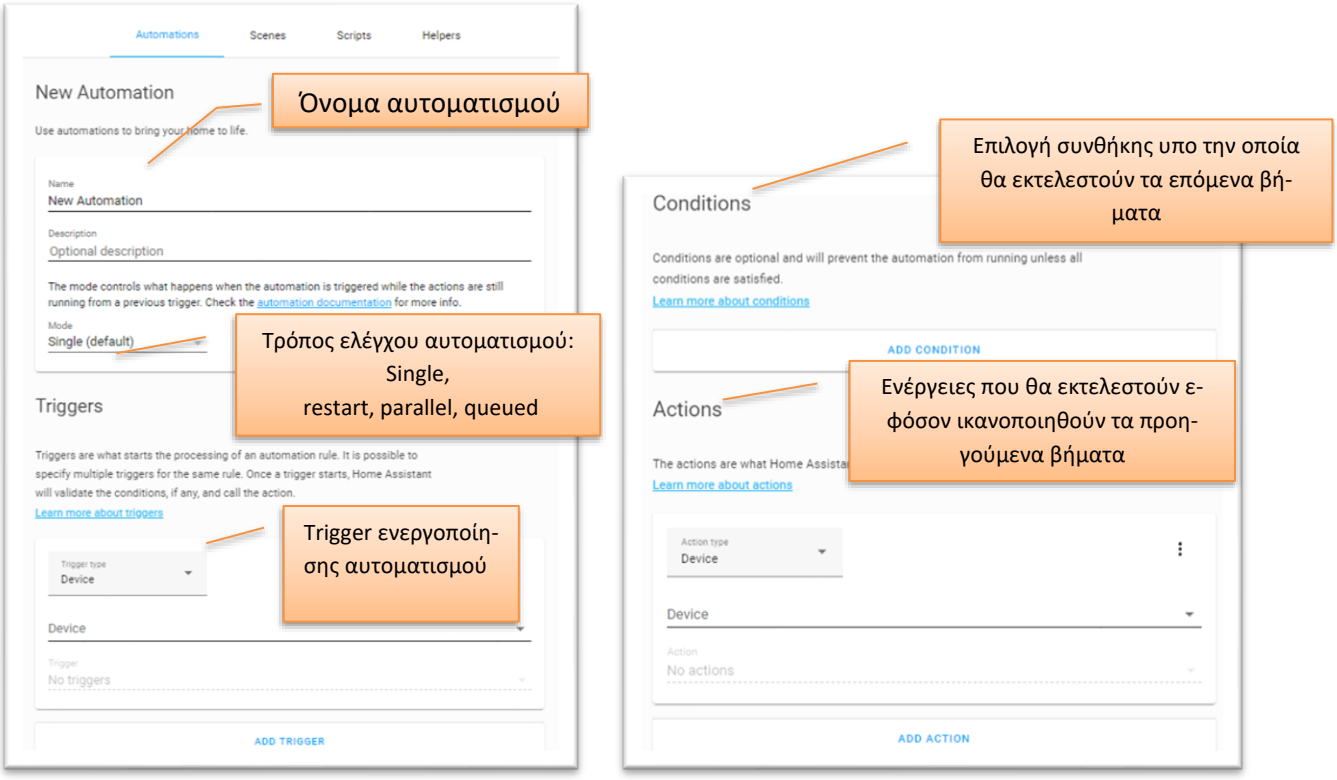
Το μπουτόν διακοπής λειτουργίας όλων των συσκευών τις οικίας (εκτός των εξωτερικών φώτων) δημιουργήθηκε καταχωρώντας στο groups.yaml τα παρακάτω πεδία και στη συνέχεια αφού έγινε επανεκκίνηση του home assistant προστέθηκε η νέα οντότητα από το group στο κεντρικό dashboard.

```
groups.yaml
#creating a group of devices to operate with a single switch
 alldevices:
    name: All devices operation
    entities:
    - switch.shelly_shsw_pm_98cdac1f4e69
    - climate.livingroom_heater
    - climate.diningroom_heater
    - climate.office_heater
    - climate.bedroom_heater
```

Εικόνα 3.29 Μπουτόν ταυτόχρονου ελέγχου λειτουργίας όλων των εσωτερικών συσκευών

3.2 Αυτοματισμοί

Οι αυτοματισμοί, μπορούν να υλοποιηθούν και να επεξεργασθούν, κάνοντας χρήση της γλώσσας προγραμματισμού YAML ή για μεγαλύτερη ευκολία μέσω του εργαλείου επεξεργασίας αυτοματισμών, που επιτρέπει την διαχείριση κανόνων αυτοματισμών. Οι αυτοματισμοί θα βοηθήσουν αρχικά στην αλόγιστη χρήση κάποιων ενεργοβόρων ηλεκτρικών συσκευών και στη συνέχεια θα χρησιμοποιηθούν για να βοηθήσουν στην συνεργασία κάποιων integration, με το συνολικό δίκτυο. Για την υλοποίηση αυτοματισμών, ακολουθείτε η διαδρομή από το πλαϊνό μενού Configuration > Automation & Scenes επιλέγεται το «Add Automation» επιλέγεται η δημιουργία ενός νέου αυτοματισμού και στη συνέχεια εμφανίζεται η παρακάτω οθόνη (Εικόνα 3.30).



Εικόνα 3.30 Οθόνη αυτοματισμών (διαχωρισμένη σε δύο μέρη)

Οι αυτοματισμοί που υλοποιήθηκαν ήταν οι εξής:

- Αυτοματισμός εναλλαγής κατάστασης peak/off-peak, για χρήση στον υπολογισμό του κόστους της καταναλωθείσας ενέργειας από το Utility meter.
- Αυτοματισμός απενεργοποίησης ηλεκτρικού θερμοσίφωνα μετά από 30 λεπτά [7]και ειδοποίηση χρήστη.
- Αυτοματισμός ενεργοποίησης/απενεργοποίησης εξωτερικών φώτων, ώστε να λειτουργούν μόνο το βράδυ.
- Αυτοματισμός απενεργοποίησης όλων των συσκευών, που βρίσκονται στο σαλόνι μετά τις 02:00 τα μεσάνυχτα και μετά από 30 λεπτά απενεργοποιείται και το θερμαντικό σώμα του γραφείου.
- Αυτοματισμός ενεργοποίησης/απενεργοποίησης, των θερμαντικών σωμάτων όλης της οικίας αν η θερμοκρασία φτάσει τα επιθυμητά επίπεδα.

Ο πρώτος αυτοματισμός, που υλοποιήθηκε αφορά την εναλλαγή κατάστασης peak/off-peak του μετρητή ενέργειας, που δημιουργήθηκε στην προηγούμενη ενότητα, του «Utility meter». Έχει ως σκοπό να προσομοιάσει την λειτουργία «ημερήσιου και νυχτερινού» τιμολογίου της ΔΕΗ και εναλλάσσεται τις ίδιες ώρες με αυτές που έχει ορίσει ο πάροχος.

Ο δεύτερος αυτοματισμός, αφορά τον χρονικό περιορισμό χρήσης του θερμοσίφωνου, καθώς είχε παρατηρηθεί ότι έμενε ανοιχτό παρότι μπορεί να είχε ήδη εξυπηρετήσει τους ένοικους της οικίας.

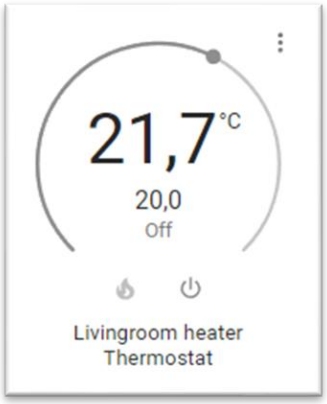
[7] Ο χρόνος λειτουργίας του θερμοσίφωνα τις κρύες ημέρες, αυξήθηκε στα 40 λεπτά καθώς το νερό κρύωνε πιο γρήγορα, όσο παρέμενε ανενεργός και ανανεωνόταν με νέο νερό.

Ο τρίτος και ο τέταρτος αυτοματισμός, είναι απλοί χρονικοί αυτοματισμοί απενεργοποίησης και ενεργοποίησης των συσκευών που περιέχονται σε αυτούς. .

Ο τελευταίος αυτοματισμός, αφορά την χρήση θερμοστάτη για τον έλεγχο της χρήσης των θερμαντικών σωμάτων . Ο θερμοστάτης αυτός, δημιουργήθηκε χάρη στο integration «climate» και συντάχθηκε όπως παρακάτω και για τα τέσσερα διαθέσιμα θερμαντικά σώματα, αλλάζοντας μόνο τα πεδία «heater» και «target_sensor».

configuration.yaml

```yaml
climate:
  - platform: generic_thermostat
    name: Livingroom_heater
    heater: switch.shelly_shsw_pm_98cdac2fc1a7
    target_sensor: sensor.livingroom_temperature
    min_temp: 5
    max_temp: 30
    ac_mode: false
    hot_tolerance: 0.5
    cold_tolerance: 0.1
    min_cycle_duration:
      minutes: 5
    keep_alive:
      minutes: 30
```

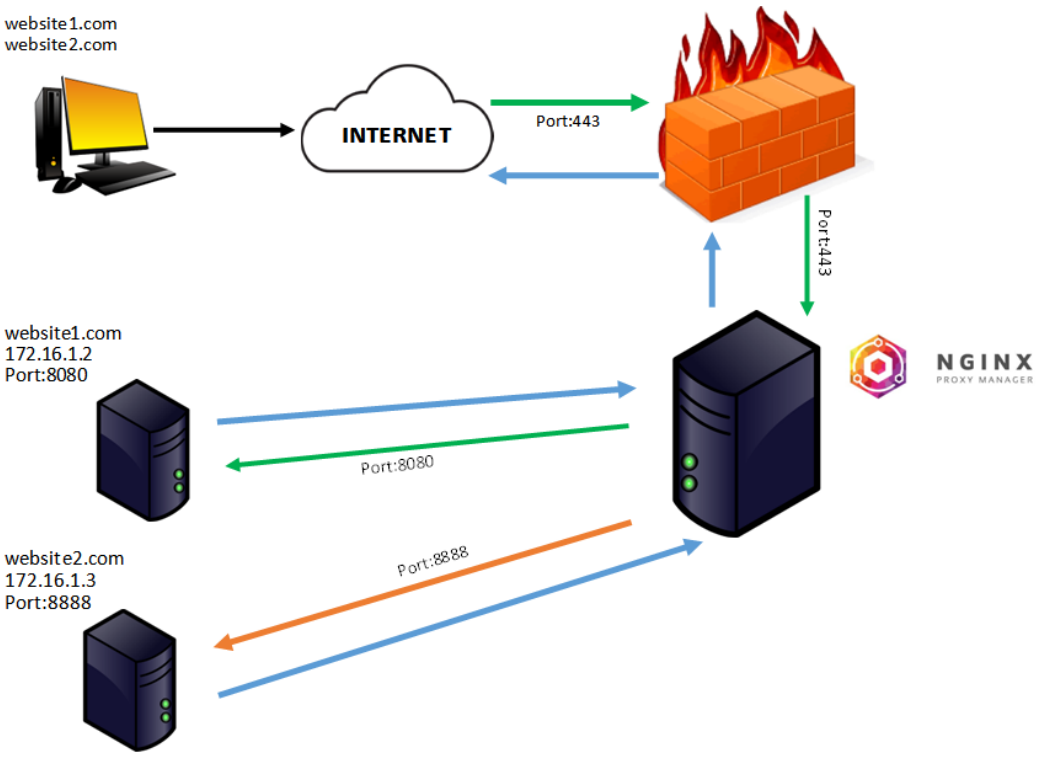


Εικόνα 3.31 Παράδειγμα δημιουργίας οντότητας θερμοστάτη για το θερμαντικό σώμα στο σαλόνι

3.3 Ασφάλεια Πρόσβασης και Ειδοποιήσεις

3.3.1 Nginx proxy manager

Το Home assistant έχει εγκατασταθεί όπως έχει ήδη αναφερθεί σε ένα Cloud Virtual Machine στην πρώτη του εκκίνηση δημιουργήθηκε λογαριασμός χρήστη και η διεύθυνση από την οποία είναι προσβάσιμο είναι η http://83.212.77.18:8123/. Μία διεύθυνση http δεν είναι απόλυτα ασφαλής καθώς δεν έχει κάποιου είδους κρυπτογράφηση SSL ή TLS και ο κωδικός πρόσβασης του χρήστη είναι επιρρεπής σε επιθέσεις. Επίσης το να διαχειριστεί το ίδιο το Home assistant την έκδοση και διαχείριση αυτών των πιστοποιητικών του προσθέτει επιπλέον φόρτο εργασίας στο συνολικό σύστημα. Χρησιμοποιώντας ένα Nginx proxy manager αυτός ο φόρτος εργασίας μεταφέρεται στον manager και το Home assistant ασχολείται μόνο με τις συσκευές του δικτύου του.



Εικόνα 3.32 Διάγραμμα λειτουργίας Nginx proxy manager

Με βάση το παραπάνω διάγραμμα, κάθε φορά που ο χρήστης θέλει να αποκτήσει πρόσβαση στο website1 ή website 2:

1. πληκτρολογεί την διεύθυνση αυτού του site στον browser
2. τότε μέσω της ασφαλούς σύνδεσης από την port: 443 και το firewall κατευθύνεται προς τον Nginx manager
3. εκεί αφού επικυρωθεί η ταυτότητα του χρήστη, η οποία έχει κωδικοποιηθεί και αποθηκευτεί στον manager
4. Συνδέεται με το αρχικό site το οποίο λειτουργούσε σε μη ασφαλείς ports όπως η 8080 και η 8888
5. Και γίνεται η προβολή του site που αιτήθηκε.

Η διαδικασία που περιγράφεται παραπάνω, εκτελείται από το Nginx ο οποίος μπορεί να περιγραφεί και ως reverse proxy server. Ένας απλός proxy server απλώς προωθεί, τα αιτήματα των clients για προβολή των περιεχομένων διάφορων server στο διαδίκτυο. Ένας reverse proxy server, βρίσκεται πριν το firewall ενός ιδιωτικού δικτύου, προσφέροντας ένα επιπλέον επίπεδο ασφάλειας και ελέγχου της δικτυακής κίνησης μεταξύ των servers και των clients. Στην ουσία με ένα reverse proxy :

- μπορεί να γίνει έλεγχος του φορτίου του δικτύου ώστε να διανεμηθεί εξίσου στους clients , αυξάνοντας την επίδοση, την ταχύτητα και την χωρητικότητα του.
- Μπορούν πέραν του ελέγχου των διακινούμενων δεδομένων, να αναλάβει όπως αναφέρθηκε και προηγουμένως την SSL κρυπτογράφηση, αποφορτίζοντας έτσι τους τελικούς servers από αυτή την εργασία.
- Μπορεί να ενισχυθεί η ανωνυμία των εμπλεκομένων, καθώς παρεμβάλλεται στο ενδιάμεσο προστατεύοντας την ταυτότητα τους. Ενώ παράλληλα, προσφέρει δυνατότητα σύνδεσης σε πολλαπλούς server, μέσω κοινής διεύθυνσης URL.[34]

Για να ενισχυθεί η ασφάλεια του Home Assistant υλοποιήθηκαν τα παρακάτω βήματα:

3.3.1.1 Εγκατάσταση Nginx με Docker compose.

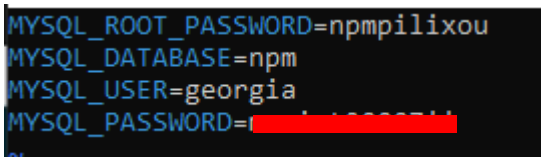
Για να γίνει η εγκατάσταση του Nginx προτιμήθηκε η χρήση των docker containers και συγκεκριμένα η χρήση της μεθόδου *docker compose.*

1. Έγινε λήψη του πακέτου με τα απαραίτητα αρχεία από το gitlab από το παρακάτω link.

```
user@snf-18951:~$ git clone https://gitlab.com/gregkoul/dc-nginx-proxy-manager.git
```

2. Έγινε η εισαγωγή των στοιχείων ταυτοποίησης στο αρχείο παραμετροποίησης του nginx config.json Οθόνη

3. Παραμετροποιήθηκε το αρχείο .env ώστε να περιλαμβάνει τα παρακάτω στοιχεία ταυτοποίησης που αφορούν την βάση δεδομένων MariaDB που χρησιμοποιείτε από το nginx.



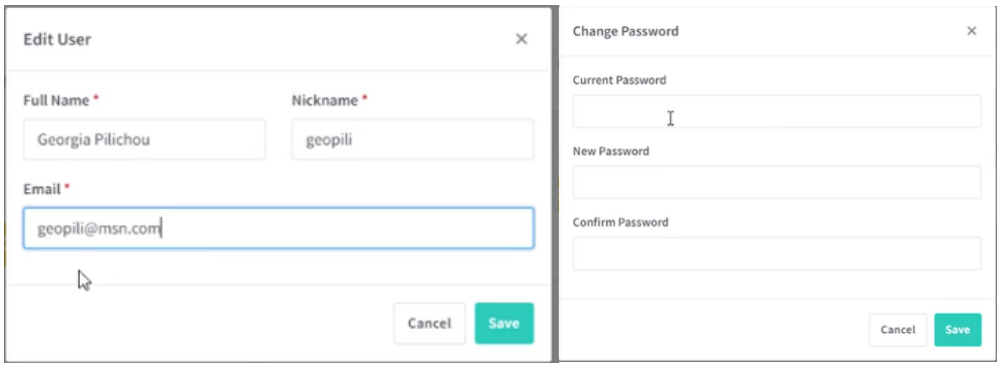
4. Συντάχθηκε το παρακάτω αρχείο docker compose

```
docker-compose.yaml
version: '3'
services:
  app:
    image: 'jc21/nginx-proxy-manager:2.8.0'
    ports:
      - '80:80'
      - '81:81'
      - '443:443'
    volumes:
      - ./config.json:/app/config/production.json
      - ./data:/data
      - ./letsencrypt:/etc/letsencrypt
    restart: unless-stopped
  db:
    image: 'jc21/mariadb-aria:10.4.15'
    env_file: .env
    volumes:
      - ./data/mysql:/var/lib/mysql
    restart: unless-stopped
```

5. Στη συνέχεια εκτελέσθηκε η παρακάτω εντολή, και ολοκληρώθηκε η εγκατάσταση του nginx proxy manager παράλληλα με την βάση δεδομένων MariaDB.

```
docker-compose up -d
```

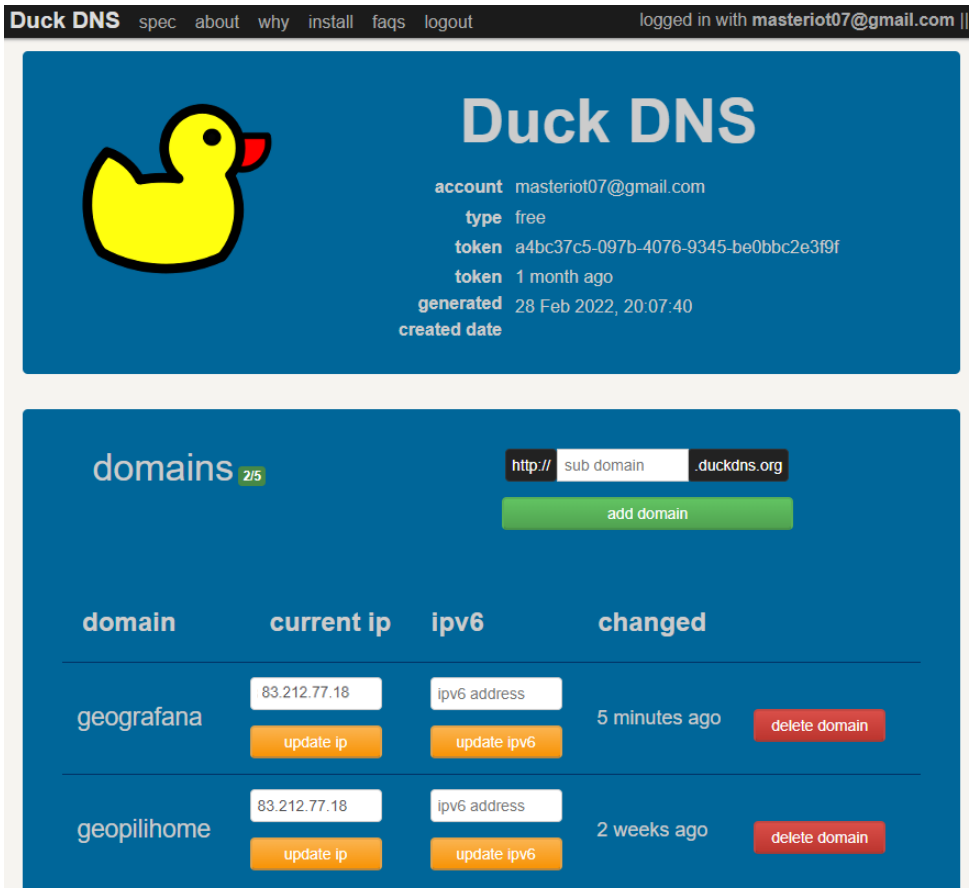
6. Εισάγοντας στον browser την διεύθυνση http://83.212.77.18:81 επιβεβαιώνεται η ορθή λειτουργία του nginx και εισάγονται τα πρώτα στοιχεία ταυτοποίησης του διαχειριστή της πλατφόρμας.



Εικόνα 3.33 Έναρξη λειτουργίας nginx και εισαγωγή στοιχείων ταυτοποίησης

3.3.1.2 Δημιουργία DNS στο DuckDnS

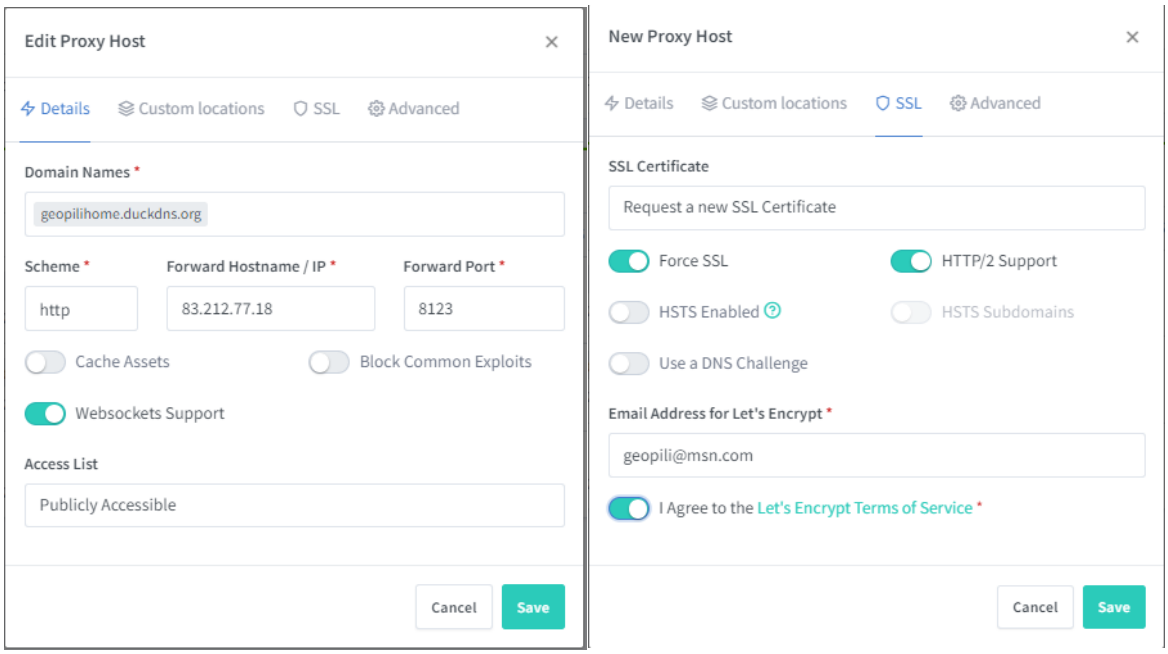
1. Γίνεται η εγγραφή στις υπηρεσίες του DuckDNS το οποίο προσφέρει δωρεάν μέχρι 5 DNS (Domain Name Services)
2. Αφού ολοκληρωθεί η εγγραφή δημιουργούνται δύο domain names τα οποία θα χρησιμοποιηθούν για να υλοποιηθεί reverse proxy για το Grafana και το Home Assistant
3. Οι διευθύνσεις που δημιουργήθηκαν είναι η http://geografana.duckdns.org και η http://geopili-home.duckdns.org



3.3.1.3 Δημιουργία Reverse proxy για το Home Assistant και αντίστοιχα για το Grafana:

1. Στον πίνακα ελέγχου του nginx (http://83.212.77.18:81) δημιουργήθηκε δύο proxy hosts ένας για το Home Assistant και ένας για το Grafana.

2. Επιλέχθηκε από το μενού Hosts η επιλογή Add Proxy Host
3. Στην καρτέλα Details εισήχθησαν το Domain name που δημιουργήθηκε προηγουμένως στο DuckDNS, ενώ στα επόμενα πεδία η αρχική μη ασφαλής διεύθυνση και port του Home Assistant (http://83.212.77.18:8123) καθώς η υποστήριξη των Websockets.



4. Στην καρτέλα SSL επιλέχθηκε να γίνει αίτημα, για νέο πιστοποιητικό SSL με ενεργοποιημένες τις επιλογές «Force SSL» και «HTTP/2 Support».

3.3.2 Ειδοποιήσεις

Οι ειδοποιήσεις που λαμβάνει ο χρήστης από το Home Assistant είναι κυρίως από τους αυτοματισμούς που τίθενται σε ισχύ. Οι ένοικοι με διαθέσιμη την εφαρμογή του Home Assistant στα smartphones τους μπορούν και λαμβάνουν αυτές τις ειδοποιήσεις απευθείας σε αυτά. Για τους υπόλοιπους ενοίκους του σπιτιού οι ειδοποιήσεις λαμβάνονται σε ένα smartphone που δεν χρησιμοποιείται πλέον από κανένα και είναι συνεχώς ενεργό στην σελίδα του home assistant ενώ βρίσκεται σε κεντρικό σημείο της οικίας. Από εκεί υπάρχει και η δυνατότητα χειροκίνητου ελέγχου των διαθέσιμων συσκευών του δικτύου.

ΚΕΦΑΛΑΙΟ 4: Συλλογή Δεδομένων και Οπτικοποίηση

Εικόνα 4.1 InfluxDB και Grafana

Σε μία υλοποίηση του διαδικτύου των πραγμάτων, συνήθως γίνεται χρήση πολλών συσκευών . Η κάθε συσκευή, αποτελείται από αισθητήρες και ενεργοποιητές. Στα πλαίσια της επικοινωνίας, όλων των περιφερειακών που απαρτίζουν τις συσκευές, συγκεντρώνεται ένας μεγάλος όγκος δεδομένων που αφορούν είτε την λειτουργία των συσκευών, είτε μετρήσεις που λαμβάνονται από τα διάφορα αισθητήρια. Αυτό σημαίνει ό,τι μία εφαρμογή IoT έχει ανάγκη από διαχείριση των δεδομένων αυτών με αυτοματοποιημένο τρόπο. Στα πλαίσια της παρούσας υλοποίησης, η IoT εφαρμογή περιορίζεται στα πλαίσια μίας οικίας, αλλά ο όγκος των δεδομένων είναι αρκετά μεγάλος, ώστε απαιτεί μία αυτοματοποιημένη μέθοδο συλλογής και αποθήκευσης των δεδομένων. Οι NoSQL (μη σχεσιακές) βάσεις δεδομένων, συνίστανται για χρήση σε εφαρμογές IoT καθώς διαθέτουν μεγάλη επεκτασιμότητα, χωρητικότητα και ταχύτητα. Λόγω της λήψης νέων δεδομένων σε πραγματικό χρόνο μία NoSQL βάση δεδομένων χρονικής σειράς η αλλιώς «Time series Data Base» είναι η ιδανική λύση. Στην ουσία μία time-series βάση δεδομένων, είναι ένα ημερολόγιο εργασιών στο οποίο διατηρούνται όλες οι πληροφορίες, της κάθε εργασίας μαζί με μία ένδειξη χρόνου. Η χρήση βάσεων δεδομένων, σε συνδυασμό με την απεικόνιση των δεδομένων σε μία πλατφόρμα απεικόνισης βοηθάει στην μακροπρόθεσμη διαχείριση και ανάλυση τους. Η πλατφόρμα απεικόνισης, θα πρέπει να διαθέτει εργαλεία λήψης δεδομένων από βάσεις δεδομένων, όπως επίσης να είναι και ευέλικτη στον τρόπο διαχείρισης και παρουσίασης των δεδομένων αυτών. Το Home Assistant, διαθέτει κάποια εργαλεία διατήρησης ιστορικού και προβολής δεδομένων. Δεν έχει όμως την ευελιξία και την χωρητικότητα να διατηρεί μεγάλο όγκο δεδομένων ενώ τα εργαλεία απεικόνισης των δεδομένων δεν προσφέρουν πολλές παραμετροποιήσεις. Στις επόμενες ενότητες, θα περιγραφούν η βάση δεδομένων InfluxDB και η πλατφόρμα απεικόνισης Grafana καθώς και τα αποτελέσματα των αναλύσεων από τα δεδομένα που συλλέχθηκαν. Επιλέχθηκαν, ώστε να κάνουν λήψη των δεδομένων, που προσφέρουν οι οντότητες που υφίστανται στο Home assistant και να τις προβάλουν σε ένα «Πίνακα ελέγχου/Dashboard», με την μορφή γραφημάτων και στατιστικών δεδομένων.[35]–[37]

4.1 Influx DB

Η InfluxDB, είναι μία από τις πιο διαδεδομένες time series βάσεις δεδομένων ανοιχτού κώδικα χωρίς «σχήμα» που προτείνονται για εφαρμογές IoT (Εικόνα 4.2). Οι Time Series βάσεις δεδομένων είναι ένα είδος βάσεων δεδομένων που δημιουργήθηκαν για να εξυπηρετούν δεδομένα που βασίζονται σε χρονικές σειρές ή χρειάζεται να έχουν χρονική σήμανση. Οι TSDB έχουν βελτιστοποιηθεί ώστε ο χρήστης να μπορεί με τον πιο αποδοτικό τρόπο να δημιουργεί, να απαριθμεί, να ενημερώνει, να καταργεί και να οργανώνει γενικότερα διάφορες χρονικές σειρές δεδομένων [35].

Rank			DBMS	Database Model	Score		
Feb 2022	Jan 2022	Feb 2021			Feb 2022	Jan 2022	Feb 2021
1.	1.	1.	InfluxDB	Time Series, Multi-model	29.34	-0.74	+3.09
2.	2.	2.	Kdb+	Time Series, Multi-model	9.11	+0.34	+1.33
3.	3.	3.	Prometheus	Time Series	6.39	+0.12	+0.63
4.	4.	4.	Graphite	Time Series	5.58	+0.00	+0.96
5.	5.	↑ 6.	TimescaleDB	Time Series, Multi-model	4.37	+0.15	+1.51
6.	6.	↑ 7.	Apache Druid	Multi-model	3.40	-0.04	+0.74
7.	7.	↓ 5.	RRDtool	Time Series	2.40	+0.32	-0.60
8.	8.	8.	OpenTSDB	Time Series	1.83	-0.03	-0.20
9.	9.	↑ 10.	GridDB	Time Series, Multi-model	1.44	+0.07	+0.62
10.	↑ 11.	↑ 11.	DolphinDB	Time Series, Multi-model	1.32	+0.09	+0.51

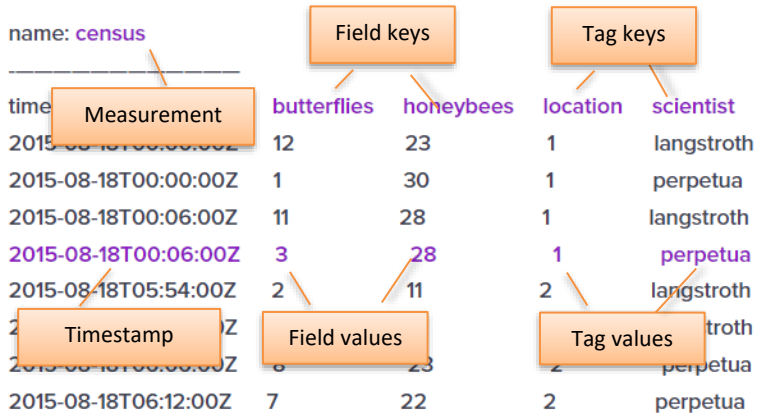
Εικόνα 4.2 Πίνακας βαθμολογίας TSDBMS [8]

Η InfluxDB, είναι προγραμματισμένη σε γλώσσα προγραμματισμού «Go» και το περιβάλλον της χρησιμοποιεί Queries (ερωτήματα) όπως και οι σχεσιακές βάσεις δεδομένων (RDBM). Όλα τα δεδομένα που αποθηκεύονται στην InfluxDB, διαθέτουν μία στήλη στην οποία καταχωρούνται οι χρονικές σημάνσεις (timestamps) με την μορφή RFC3339 UTC. Στην δομή της, περιλαμβάνονται και τα πεδία δεδομένων τα οποία συντάσσονται με την μορφή key-value. Το πεδίο (field) key, είναι κατηγορίας string και σε αυτό αποθηκεύονται τα meta data[9]. Το πεδίο (field) value, μπορεί να είναι κατηγορίας string, float, integer ή Boolean[10] και περιλαμβάνει τα δεδομένα της πηγής. Τα δύο πεδία αυτά, δημιουργούν ένα σετ το οποίο δεν διαθέτει δυνατότητα χρήσης ευρετηρίου για τον λόγο αυτό μπορούν να χρησιμοποιηθούν προαιρετικά τα Tags. Αυτά είναι σετ από key και values, αλλά περιέχουν μόνο metadata και διαθέτουν την δυνατότητα χρήσης ευρετηρίου, κάνοντας πιο γρήγορη την εκτέλεση των Queries. Το στοιχείο «Measurement», το οποίο είναι τύπου string χρησιμοποιείται από την InfluxDB για να υποδείξει τον χώρο/πίνακα που περιλαμβάνει όλα τα στοιχεία που δομούν την βάση δεδομένων μίας πηγής[38].

[8] Πηγή: "DB-Engines Ranking - popularity ranking of time Series DBMS." https://db-engines.com/en/ranking/time+series+dbms (accessed Feb. 26, 2022).

[9] Meta data: είναι τα δεδομένα που δίνουν συγκεντρωτικές πληροφορίες, για τις διάφορες εκδοχές των κυρίως δεδομένων, διευκολύνοντας έτσι την παρακολούθηση και εργασία πάνω σε αυτά.

[10] String: κείμενο, float: αριθμός κινητής υποδιαστολής, integer: ακέραιος αριθμός, Boolean: λογική τιμή



Εικόνα 4.3 Παράδειγμα βάσης δεδομένων[11]

Στην υλοποίηση αυτή, έγινε η εγκατάσταση της InfluxDB στο cloud VM (Ενότητα 2.1.5 Εγκατάσταση Βάσης δεδομένων InfluxDB (PaaS)) που εκτελείται και το Home Assistant έτσι ώστε τα δεδομένα που θα συλλεχθούν να είναι άμεσα προσβάσιμα για περαιτέρω επεξεργασία και προβολή από την πλατφόρμα απεικόνισης Grafana. Για να μπορέσει το home assistant να αποκτήσει πρόσβαση στην InfluxDB έπρεπε:

1. να εισαχθούν τα παρακάτω πεδία στο αρχείο configuration.yaml και στο secrets.yaml του home assistant:

```
configuration.yaml
# influxDB manual initilization
influxdb:
  host: 83.212.77.18
  port: 8086
  username: admin
  password: !secret influx_pass

secret.yaml
# Hiding my passwords from main configuration file
influx_pass: [redacted]
```

2. μέσα από το περιβάλλον του τερματικού του Cloud VM, να δημιουργήσουμε μία βάση δεδομένων με όνομα **home_assistant.**

Έξοδος στην οθόνη:

```
user@snf-18951:~$ influx -username admin -password [redacted]
Connected to http://localhost:8086 version 1.8.10
InfluxDB shell version: 1.8.10
> CREATE DATABASE home-assistant
```

3. Να γίνει επανεκκίνηση του Home assistant και τέλος να επιβεβαιωθεί ότι όντως η InfluxDB λαμβάνει δεδομένα από το home assistant πληκτρολογώντας τις παρακάτω εντολές:

[11] Πηγή: "InfluxDB key concepts | InfluxDB OSS 1.5 Documentation." https://docs.influxdata.com/influxdb/v1.5/concepts/key_concepts (accessed Feb. 26, 2022).

Έξοδος στην οθόνη:

```
> USE home_assistant
Using database home_assistant
> SHOW series
key
---
\ ,domain=sensor,entity_id=shelly_shem_c45bbe6c4d01_1_power_factor
\ ,domain=sensor,entity_id=shelly_shem_c45bbe6c4d01_2_power_factor
%,domain=sensor,entity_id=bedroom_humidity
%,domain=sensor,entity_id=gina_humidity
%,domain=sensor,entity_id=livingroom_humidity
%,domain=sensor,entity_id=office_humidity
%,domain=sensor,entity_id=poco_x3_battery_level
%,domain=sensor,entity_id=pocox3_battery_level
%,domain=sensor,entity_id=redmi_note_4_battery_level
...,domain=sensor,entity_id=daily_energy_cost_all
...,domain=sensor,entity_id=daily_energy_cost_offpeak
...,domain=sensor,entity_id=daily_energy_cost_peak
```

Σειρές δεδομένων που ελήφθησαν από το Home Assistant

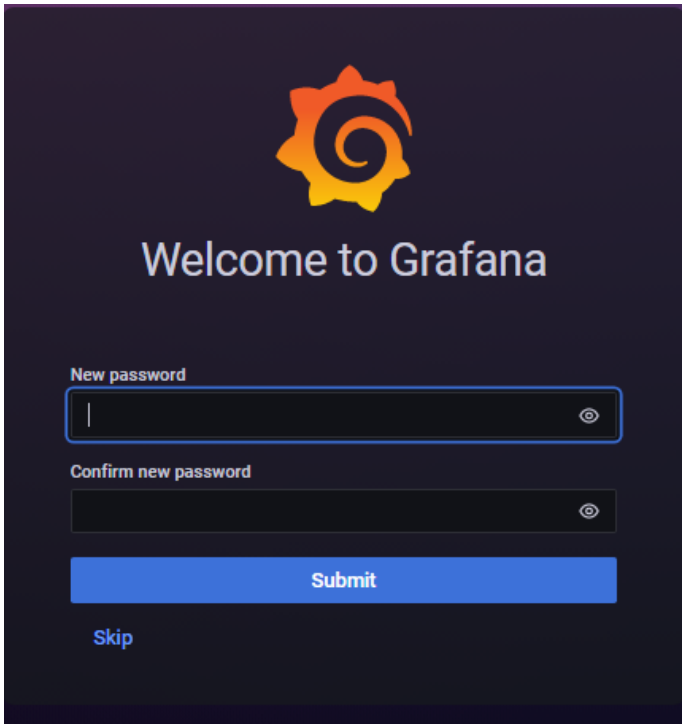
4.2 Grafana

Εφόσον, εγκαταστάθηκε και παραμετροποιήθηκε η βάση δεδομένων InfluxDB, χρειαζόταν μία πλατφόρμα η οποία μπορεί να εκμεταλλεύεται τέτοιου είδους δεδομένα και να τα προβάλει σε φιλική μορφή για τον χρήστη. Με το Grafana, υπάρχει η δυνατότητα ανάπτυξης πολύ προχωρημένων στατιστικών και γραφημάτων με αρκετά μεγάλο ιστορικό παρελθοντικών μετρήσεων. Με την σχεδίαση και ανάπτυξη των κατάλληλων γραφημάτων, γίνεται πολύ πιο αποδοτική και εύκολη η ανάλυση των δεδομένων. Στην υλοποίηση αυτή είναι πολύ σημαντικό, να μπορεί ο χρήστης να αντιλαμβάνεται με μία εικόνα το τι συμβαίνει αυτή τη στιγμή στο δίκτυο που υλοποιήθηκε. Θα πρέπει, να μπορεί να έχει πλήρη εικόνα των χρονικών στιγμών των δεδομένων που απεικονίζονται, καθώς και να μπορεί να επιλέγει το χρονικό διάστημα από το οποίο θα προβάλει τα δεδομένα που έλαβε. Η δυνατότητα, να μπορεί να ανατρέξει σε προηγούμενες μετρήσεις, βάση του πότε ελήφθησαν θα δώσει στον χρήστη την ευχέρεια να συγκρίνει τις διάφορες περιπτώσεις. Μερικές από τις δυνατότητες του Grafana είναι:

- Η δημιουργία πολυδιάστατων γραφημάτων
- Η αναζήτηση μετρήσεων δεδομένων
- Οι πίνακες ελέγχου με δυναμικό περιεχόμενο και μορφή
- Η ημερολογιακή ανάλυση δεδομένων
- Η δημιουργία ειδοποιήσεων βάση οπτικών επιλογών
- Οι σημειώσεις πάνω σε γραφήματα
- Η βελτιωμένη αναζήτηση και οπτικοποίηση δεδομένων
- Η ταυτόχρονη επεξεργασία ανόμοιων δεδομένων από διάφορες πηγές
- Η κατανομή των δεδομένων
- Ότι διαθέτει ενσωματωμένη πλατφόρμα αποστολής Query[39]

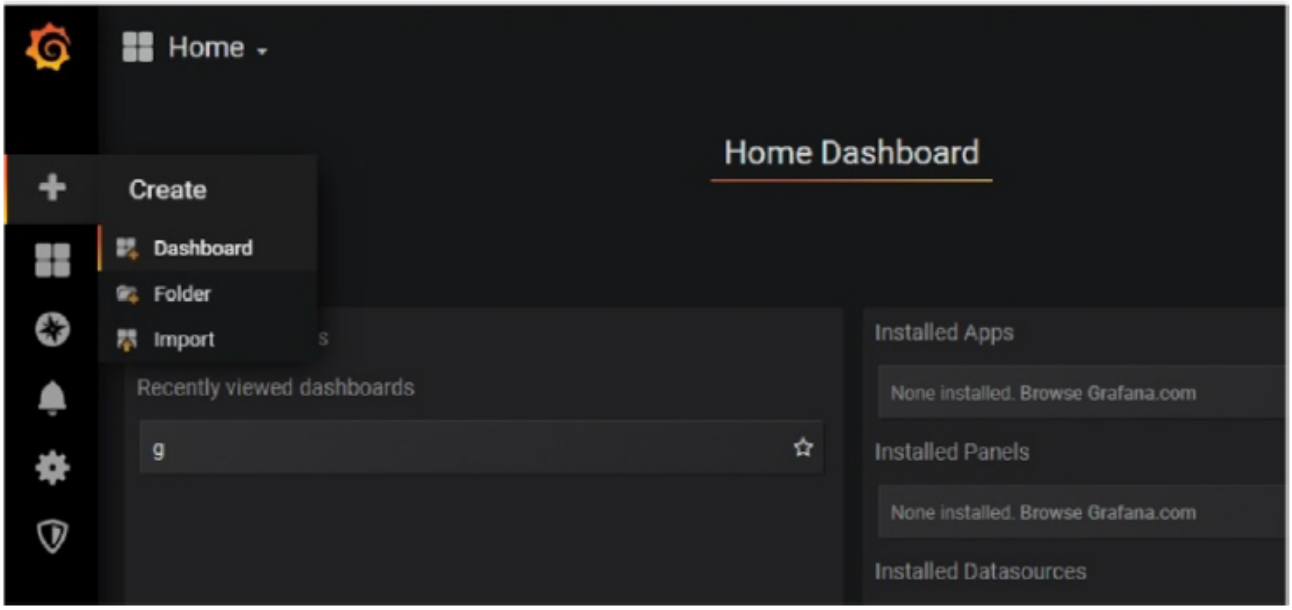
Μετά την πρώτη εγκατάσταση του Grafana που περιγράφηκε στην ενότητα (2.1.6 Εγκατάσταση Grafana (SaaS)) πρέπει να γίνουν οι πρώτες παραμετροποιήσεις έτσι ώστε να ενσωματωθεί στην πλατφόρμα του Home assistant αλλά και για να μπορεί να συνεργαστεί με την βάση δεδομένων που υλοποιήθηκε στην προηγούμενη ενότητα. Τα βήματα που ακολουθήθηκαν είναι τα εξής[40]:

1. Αποκτάται πρόσβαση στην σελίδα του Grafana βάζοντας την παρακάτω διεύθυνση στον Web browser: http://83.212.77.18:3000/
2. Στην πρώτη είσοδο, ζητείται η δημιουργία λογαριασμού και κωδικός πρόσβασης, για τον διαχειριστή του Grafana



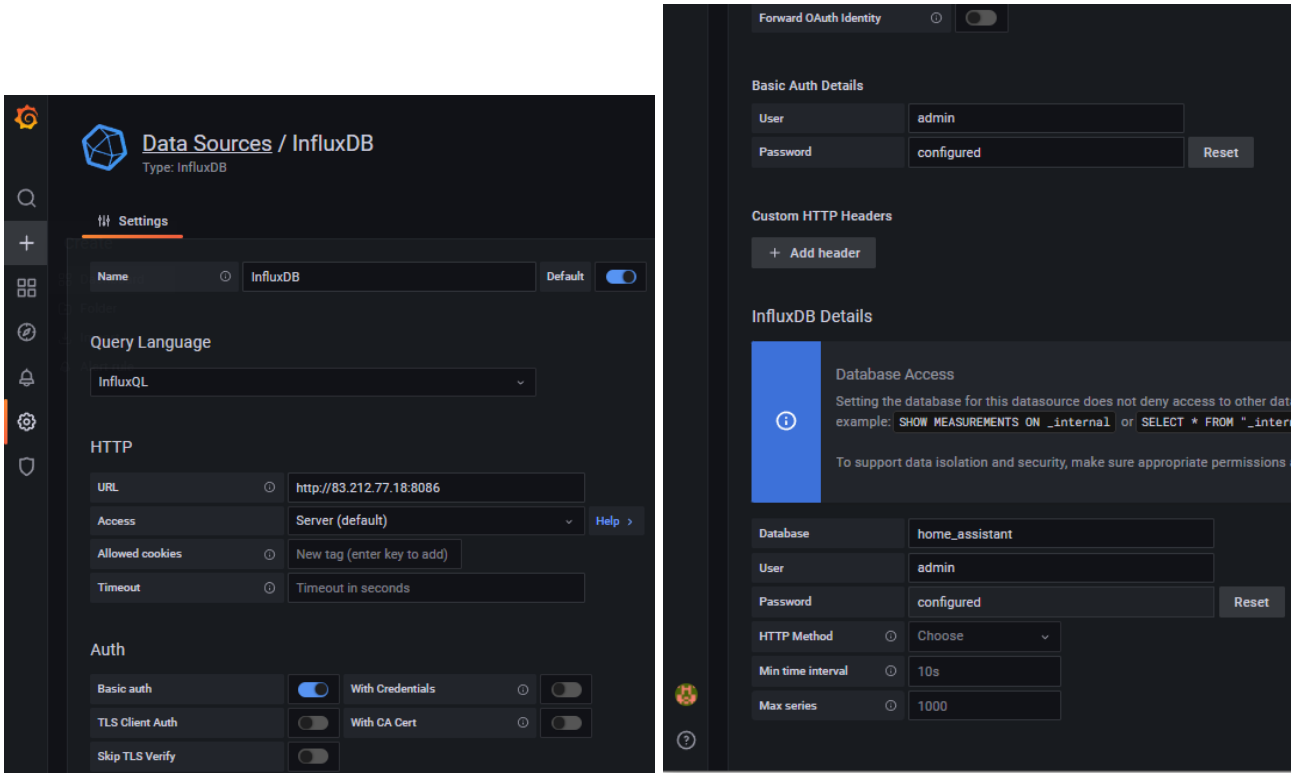
Εικόνα 4.4 Οθόνη δημιουργίας λογαριασμού χρήστη/διαχειριστή Grafana

1. Στη συνέχεια, ανοίγει η αρχική οθόνη του Grafana, στην οποία τοποθετούνται νέα panels και γραφικά βάση των προτιμήσεων του διαχειριστή της πλατφόρμας.



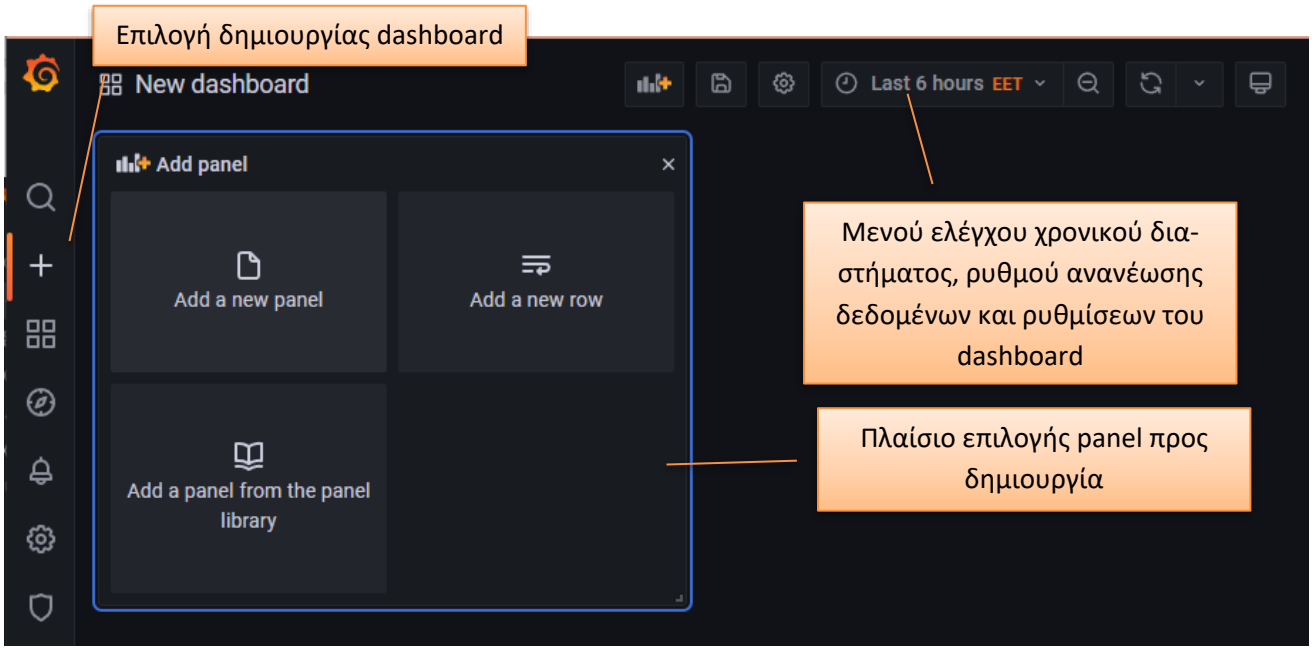
Εικόνα 4.5 Αρχικός κενός πίνακας ελέγχου.

2. Επιλέγοντας από το πλαϊνό μενού, το Configuration >data sources και «add data source», εισάγεται η InfluxDB ως πηγή δεδομένων και ορίζονται τα στοιχεία εισόδου του χρήστη σε αυτήν.



Εικόνα 4.6 Προσθήκη InfluxDB ως πηγή δεδομένων

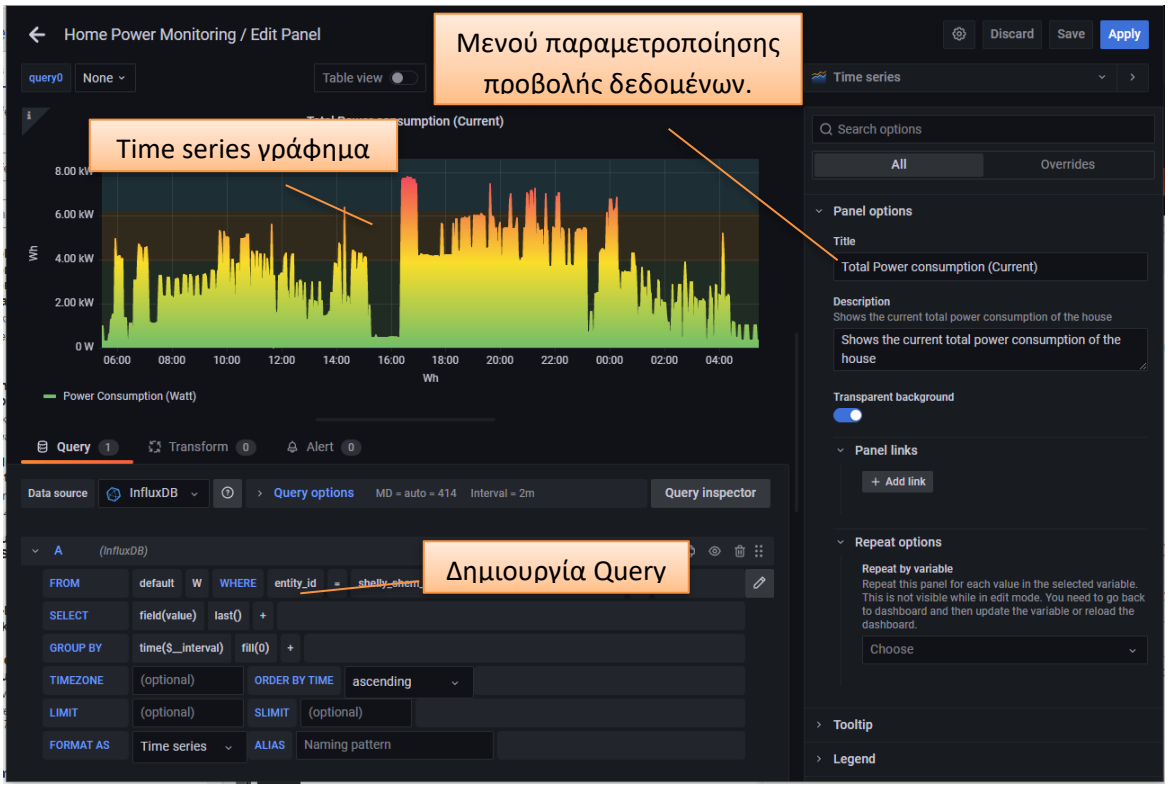
3. Τέλος, επιλέγεται η αποθήκευση των ρυθμίσεων και πλέον το μόνο που μένει είναι, η διαμόρφωση του πίνακα ελέγχου.



Εικόνα 4.7 Οθόνη δημιουργίας πίνακα ελέγχου/Dashboard

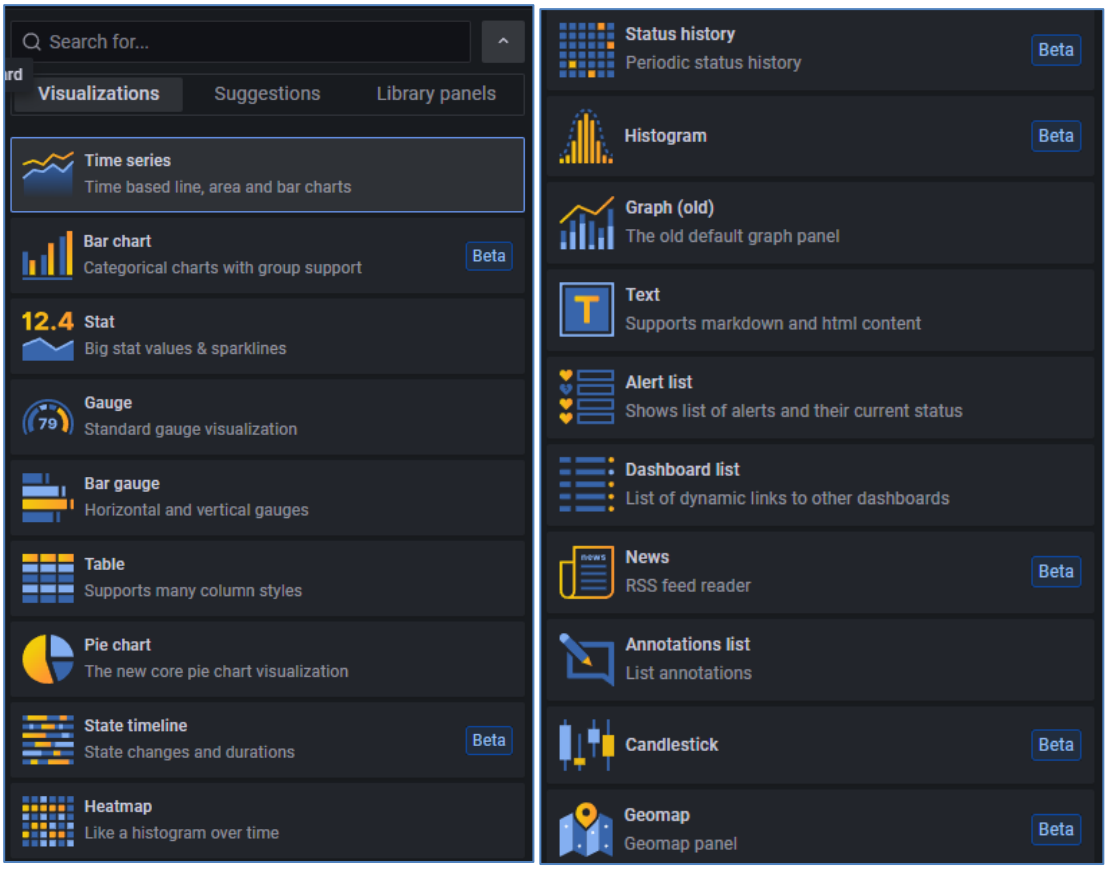
Σε αυτό το σημείο, ο χρήστης μπορεί να προσθέσει νέα panel, να επιλέξει πως θα απεικονίζονται τα δεδομένα που θα προβληθούν, καθώς σε ποιο χρονικό διάστημα αναφέρονται τα προβαλλόμενα δεδομένα (Εικόνα 4.7).

Κατά την δημιουργία ενός panel, ο χρήστης καλείται να εισάγει τα δεδομένα που επιθυμεί, υλοποιώντας ένα query είτε χρησιμοποιώντας το γραφικό περιβάλλον είτε με χρήση εντολών SQL.



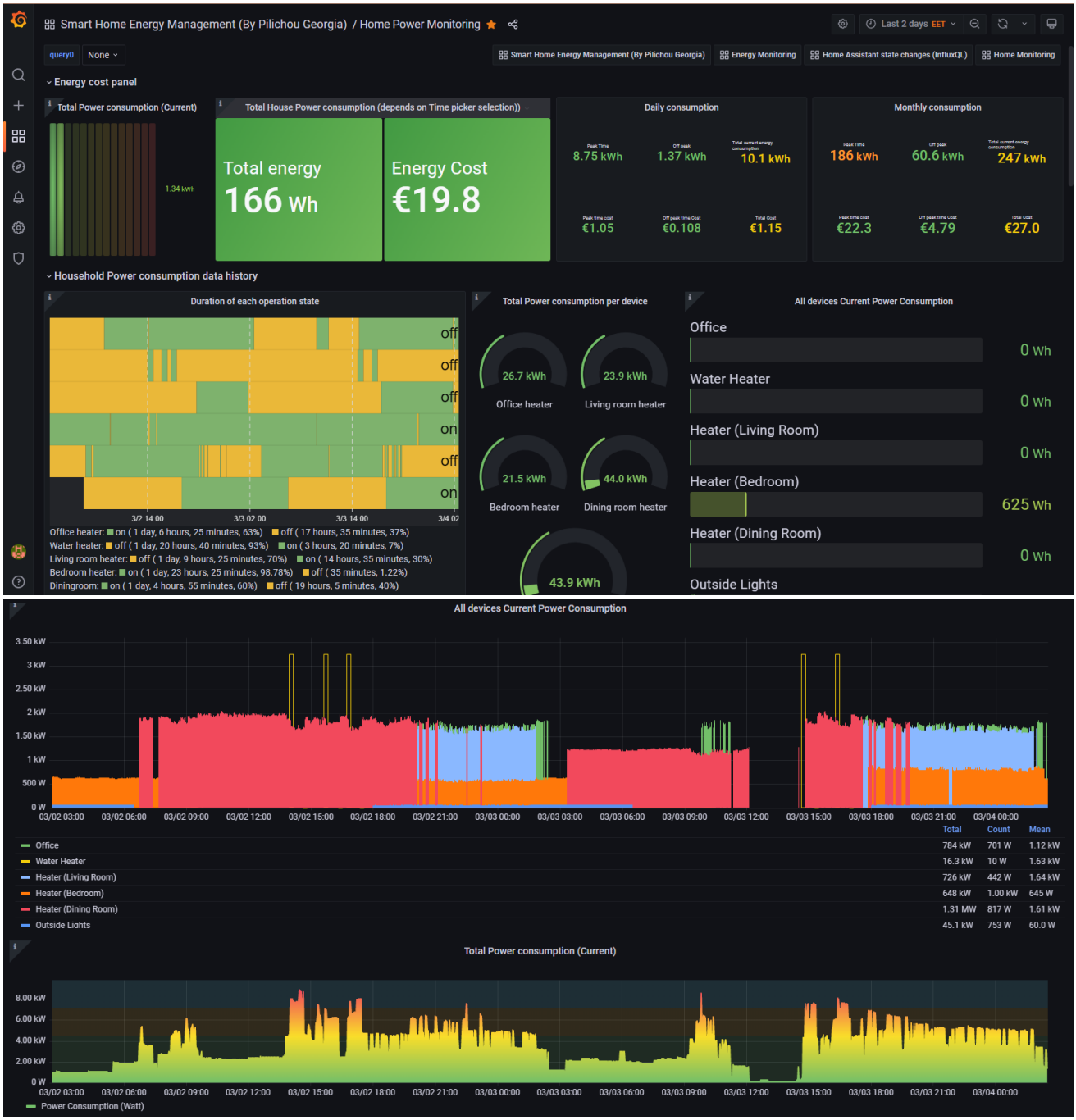
Εικόνα 4.8 Δημιουργία panel με γραφική απεικόνιση (time series) της συνολικής κατανάλωσης ενέργειας.

Επίσης, υπάρχουν πολλοί διαφορετικοί τρόποι να προβληθούν τα δεδομένα, αναλόγως την περίπτωση.

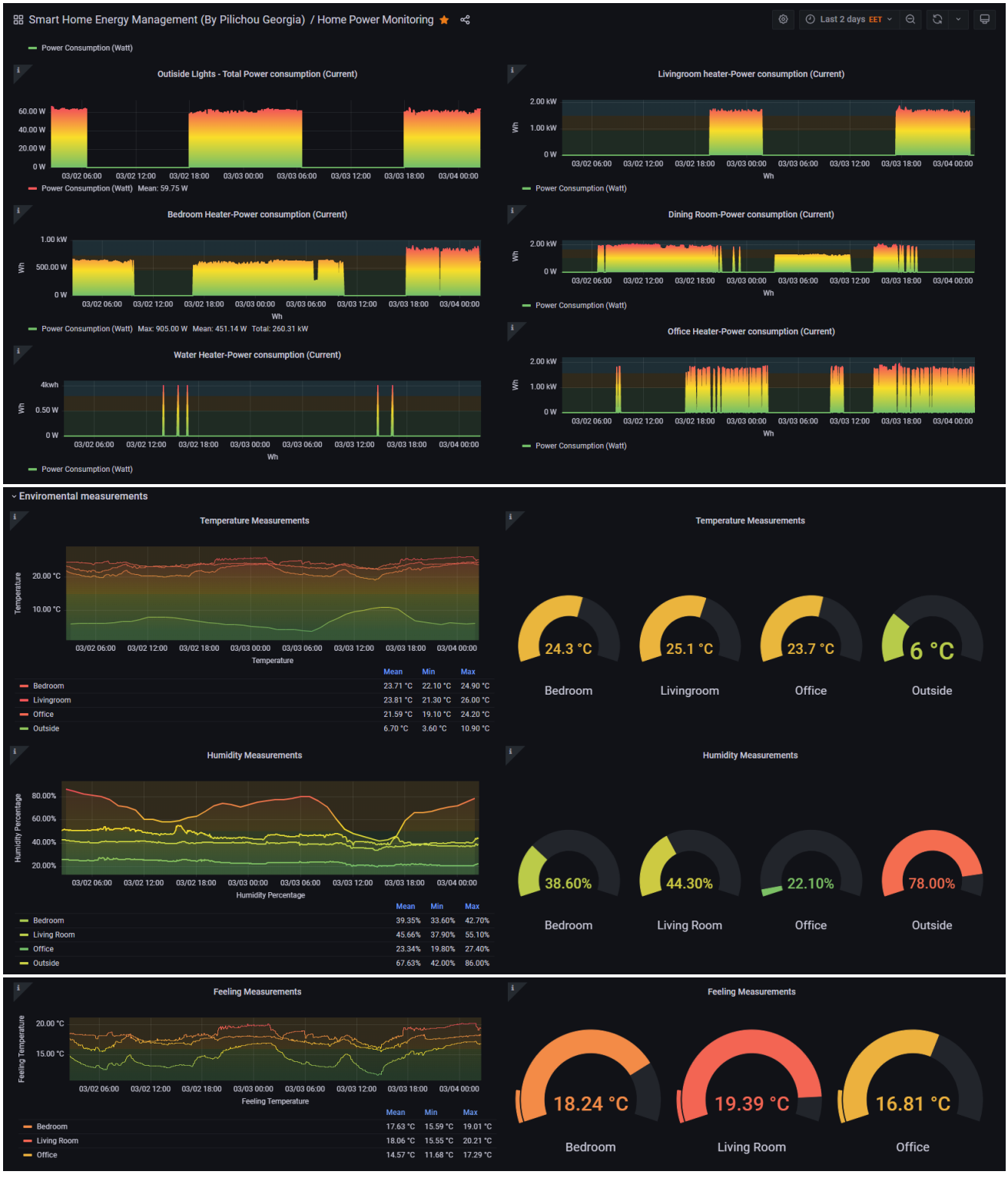


Εικόνα 4.9 Επιλογές γραφημάτων από το δεξί πλαϊνό μενού

Στη συνέχεια έγινε διαμόρφωση του κεντρικού πίνακα ελέγχου και το τελικό αποτέλεσμα ήταν το παρακάτω:



Εικόνα 4.10 Πίνακας ελέγχου Grafana σε (1/2)

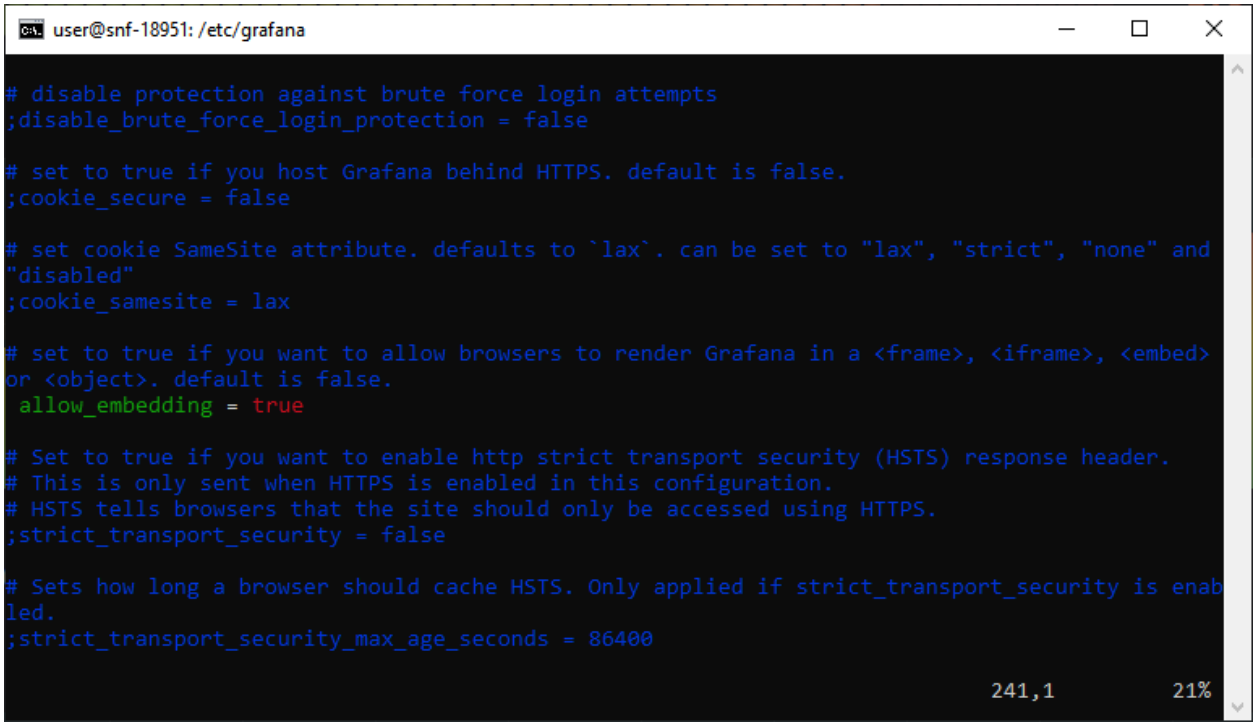


Εικόνα 4.11 Πίνακας ελέγχου Grafana σε (2/2)

Το επόμενο βήμα είναι, η ενσωμάτωση του Grafana στην πλατφόρμα του Home assistant. Για να γίνει αυτό πρέπει:

1. Αρχικά να ενεργοποιηθεί η επιλογή *allow_embeding = true*, κάνοντας επεξεργασία του αρχείου ρυθμίσεων του Grafana όπως φαίνεται στις επόμενες εικόνες,

```
user@snf-18951:/etc/grafana$ sudo vi grafana.ini
```



2. Αφού γίνει επανεκκίνηση του service του Grafana

```
user@snf-18951:/etc/grafana$ sudo service grafana-server restart
[sudo] password for user:
```

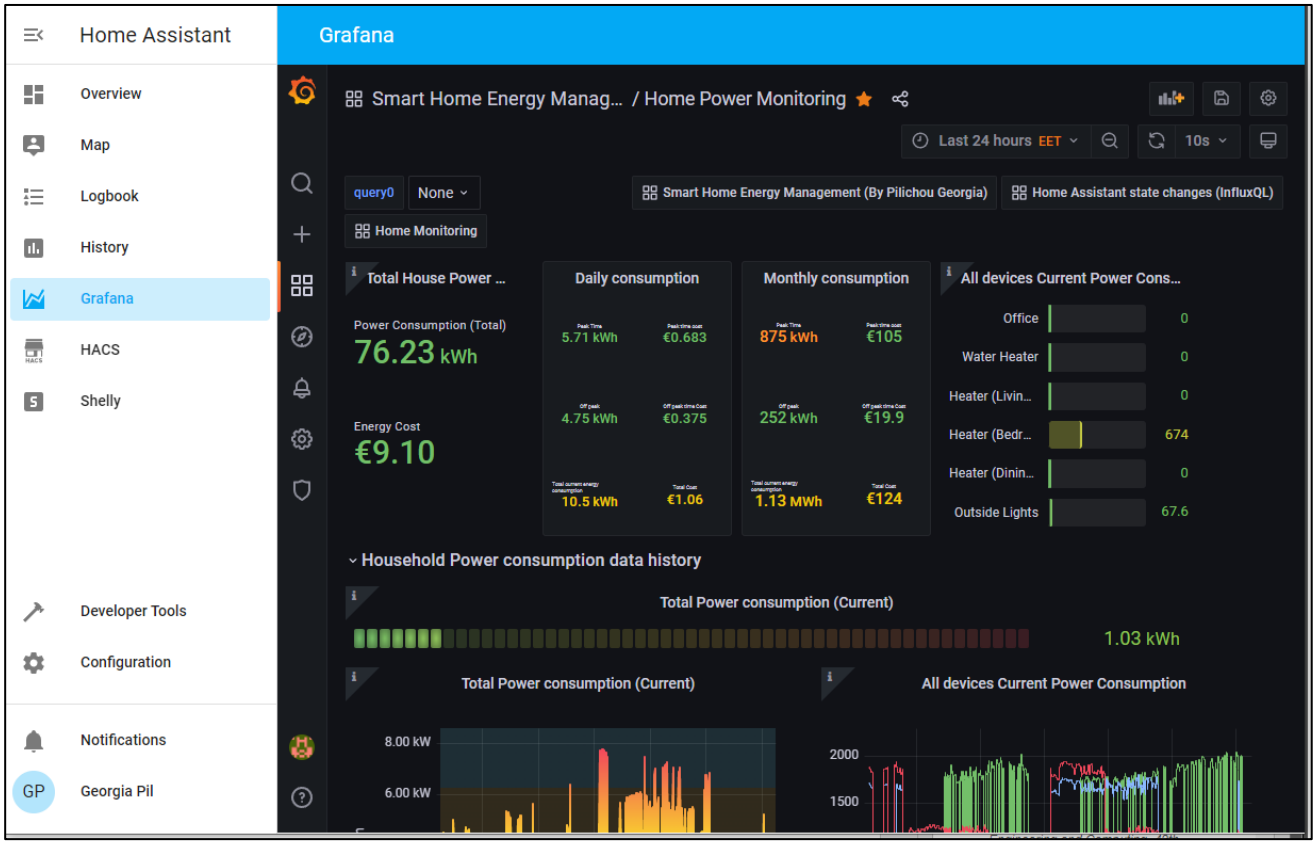
3. Γίνεται προσθήκη των παρακάτω πεδίων στο configuration.yaml του home assistant

```
user@snf-18951:~$ docker exec -it home-assistant bash
bash-5.0# vi configuration.yaml
```

configuration.yaml

```yaml
panel_iframe:
  grafana:
    title: 'Grafana'
    url: 'http://83.212.77.18:3000'
    icon: mdi:chart-areaspline
```

4. Γίνεται επανεκκίνηση του Home assistant
5. Τέλος, διαπιστώνεται η προβολή ενός ακόμη εργαλείου στο πλαϊνό μενού σαν iframe το οποίο μας οδηγεί στην πλατφόρμα του Grafana.



Εικόνα 4.12 Ενσωμάτωση Grafana interface στο Home assistant

4.2.1 Ασφάλεια πρόσβασης

Στην ενότητα 3.3.1.3 περιγράφηκε η διαδικασία μετατροπής μίας ανασφαλούς σύνδεσης http:// σε https:// με χρήση του nginx ως reverse proxy. Η ίδια διαδικασία όπως έχει αναφερθεί εκτέλεστηκε και για το Grafana στην περίπτωση του όμως χρειάστηκαν κάποιες επιπλέον αλλαγές. Το iframe δεν μπορεί να προβάλει ανασφαλείς σελίδες εφόσον το home assistant που το φιλοξενεί είναι σε ασφαλες περιβάλλον. Επομένως, αφού και το Grafana λειτουργεί πλεον με reverse proxy, έγιναν και οι εξής αλλαγές:

1. Παραμετροποιήθηκε το αρχείο Grafana.ini θέτοντας στα παρακάτω πεδία τις αντίστοιχες τιμές που φαίνονται παρακάτω, έτσι ώστε να λειτουργεί σωστά το iframe στο Home Assistant:

```
# set to true if you host Grafana behind HTTPS. default is false.
cookie_secure = true

# set cookie SameSite attribute. defaults to `lax`. can be set to "lax", "s
trict", "none" and "disabled"
cookie_samesite = none

# set to true if you want to allow browsers to render Grafana in a <frame>,
 <iframe>, <embed> or <object>. default is false.
 allow_embedding = true
```

2. Το αρχείο configuration.yaml του Home assistant έπρεπε να διορθωθεί, ώστε να εμφανίζει την α-σφαλή έκδοση της σελίδας του Grafana. Συγκεκριμένα η γραμμή που αφορά το iframe του Grafana

αλλάχθηκε στο πεδίο του URL από το ανασφαλές http://83.212.77.18:3000 σε https://geo-grafana.duckdns.org/

```
configuration.yaml
panel_iframe:
  grafana:
    title: 'Grafana'
    url: 'https://geografana.duckdns.org/'
    icon: mdi:chart-areaspline
```

ΚΕΦΑΛΑΙΟ 5: ΕΠΙΛΟΓΟΣ

Το δίκτυο διαχείρισης ενέργειας και ενεργειακής κατανάλωσης της οικίας με χρήση τεχνολογιών IoT, είναι πλέον ολοκληρωμένο και λειτουργικό. Τα δεδομένα από τις συσκευές, λαμβάνονται από το Home assistant, εκείνο τις προωθεί στην βάση δεδομένων InfluxDB και το Grafana αναλαμβάνει, την γραφική απεικόνιση τους βάση των επιλογών του χρήστη. Κάνοντας τις κατάλληλες παραμετροποιήσεις, στο δυναμικό περιβάλλον του Grafana ο χρήστης μπορεί να αντιληφθεί εύκολα αν κάποια συσκευή χρίζει βελτιστοποίησης ή αν υπάρχουν παράμετροι, που καθιστούν δύσκολη την εξοικονόμηση ενέργειας. Στην οικία αυτή, οι πιο ενεργοβόρες συσκευές είναι τα θερμαντικά σώματα και ο ηλεκτρικός θερμοσίφωνας καθώς όπως φαίνεται και από τα μηνιαία δεδομένα χρήσης τους (Εικόνα 5.1) ενεργοποιούνται καθημερινά και για αρκετά μεγάλο διάστημα.

Εικόνα 5.1 Μηνιαία χρήση συσκευών

Τα εξωτερικά φώτα φαίνεται να είναι μία από τις συσκευές που δεν γίνεται να βελτιστοποιηθούν περεταίρω καθώς η λειτουργία τους είναι απαραίτητη κάθε μέρα, όλο το χρόνο για τις νυχτερινές ώρες. Χρησιμοποιούνται ήδη οικονομικοί λαμπτήρες LED και η συνολική τους ισχύ δεν ξεπερνά τα 70W. Αντίστοιχες πληροφορίες κατανάλωσης μπορούν να ληφθούν και για τις υπόλοιπες συσκευές που διαθέτουν μετρητή κατανάλωσης ενέργειας.

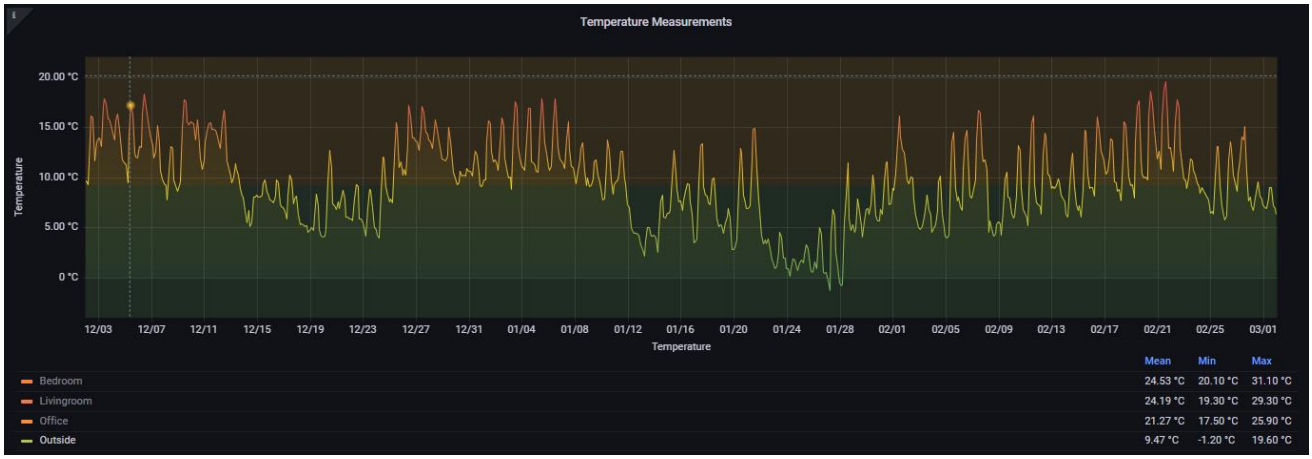
Πίνακας 5.1 Μέσος όρος θερμοκρασιών τριμήνου των δωματίων και εξωτερικού περιβάλλοντος

Θερμοκρασία/Δωμάτιο	Δεκέμβριος	Ιανουάριος	Φεβρουάριος
Υπνοδωμάτιο	25,9^{O}C	24 ^{O}C	23,9 ^{O}C
Σαλόνι	24,0 ^{O}C	24,4 ^{O}C	21,1 ^{O}C
Γραφείο	21,5 ^{O}C	21 ^{O}C	21,1 ^{O}C
Εξωτερική θερμοκρασία	8,8 ^{O}C	7.69 ^{O}C	10,1 ^{O}C

Από τα διαθέσιμα δεδομένα, που προβλήθηκαν μέσω του Grafana, παρατηρήθηκε μία δυσλειτουργία που αφορούσε την ένδειξη «συνολικής καταναλωθείσας ενέργειας από όλα τα Shelly (εκτός του Shelly EM). Ο μετρητής φαίνεται να μηδενίζει σε ανύποπτο χρόνο και αυτό είχε ως αποτέλεσμα να μην είναι δυνατός ο υπολογισμός της συνολικής κατανάλωσης ενέργειας βάση του διατηρηθέντος ιστορικού μετρήσεων (Πίνακας 5.2). Για να επιλυθεί αυτό το ζήτημα, θα πρέπει να αντικατασταθεί αυτός ο μετρητής με ένα εικονικό μετρητή ενέργειας, που θα υλοποιηθεί στο Home assistant κάνοντας χρήση του Riemann sum integral – integration. Η διαδικασία δημιουργίας, ενός τέτοιου εικονικού μετρητή έχει περιγραφεί στην σελίδα 90, όταν έπρεπε να δημιουργηθεί μία οντότητα μετρητή με δυνατότητες εναλλαγής κλίμακας κόστους. Μία άλλη δυσλειτουργία παρατηρήθηκε στο Shelly PM1 που ελέγχει τον ηλεκτρικό θερμοσίφωνα. Η συσκευή πιθανότατα έχει κάποιο εργοστασιακό ελάττωμα καθώς από τις πρώτες μέρες χρήσης φάνηκε να «παγώνει» σε απρόοπτες στιγμές και να χάνει την επικοινωνία και με τον MQTT broker αλλά και από το Shelly cloud. Από το επίσημο forum της εταιρείας δεν φαίνεται να έχει αντιμετωπιστεί αυτό το πρόβλημα παρότι παρουσιάζεται και σε άλλες συσκευές Shelly. Η αντικατάσταση του, με κάποιο άλλο πιθανότατα να επιλύσει αυτό το πρόβλημα.

Πίνακας 5.2 Δείγμα μετρητών συνολικής κατανάλωσης ενέργειας ανά συσκευή

Time	Office heater	Livingroom heater	Bedroom heater	Diningroom heater
12/12/2021 10:00	139 Wh	17.2 Wh	33.1 Wh	79.2 Wh
12/12/2021 11:00	139 Wh	17.2 Wh	33.1 Wh	79.2 Wh
12/12/2021 12:00	0 Wh	0 Wh	0 Wh	0 Wh
12/12/2021 18:00	0 Wh	0 Wh	0 Wh	0.930 Wh
12/12/2021 19:00	0 Wh	0 Wh	0 Wh	2.24 Wh
12/12/2021 20:00	0 Wh	0 Wh	0 Wh	3.49 Wh
12/12/2021 21:00	0 Wh	0 Wh	0 Wh	4.78 Wh
12/12/2021 22:00	0 Wh	0 Wh	0 Wh	6.02 Wh
12/12/2021 23:00	0 Wh	0.180 Wh	0.280 Wh	6.06 Wh
13/12/2021 0:00	0 Wh	1.89 Wh	0.540 Wh	6.06 Wh
13/12/2021 1:00	0 Wh	3.04 Wh	0.920 Wh	6.06 Wh
13/12/2021 2:00	0 Wh	3.04 Wh	1.21 Wh	6.06 Wh



Εικόνα 5.2 Μετρήσεις εξωτερικής θερμοκρασίας τριμήνου (Δεκέμβριος 2021-Φεβρουάριος 2022)

Εκ των μετρήσεων, που ελήφθησαν μέσω του Grafana, παρατηρήθηκε πόσες ώρες λειτουργίας είχε η κάθε συσκευή ανά ημέρα και συγκρίθηκαν οι μετρήσεις με τις θερμοκρασίες που επικρατούσαν στο εξωτερικό περιβάλλον της οικίας. Είναι πολύ σημαντικό να αναφερθεί ότι κατά την διάρκεια του τριμήνου δεν επικρατούσαν πολύ καλές καιρικές συνθήκες ο μέσος όρος του τριμήνου ήταν στους 9,47 ^{O}C με ένα εύρος θερμοκρασιών από -1,2 ^{O}C έως 19,6 ^{O}C (Πίνακας 5.1). Αυτό σημαίνει ότι υπήρχαν αυξημένες απαιτήσεις θέρμανσης στην οικία ιδιαίτερα τις πολύ κρύες ημέρες. Το θερμοσίφωνο καθώς βρίσκεται σε αποθήκη εκτός της οικίας χωρίς άλλη θέρμανση έπρεπε να λειτουργεί περισσότερες ώρες. Το νερό, που ανανέωνε το περιεχόμενο της δεξαμενής του, κρύωνε το υπάρχον περιεχόμενο πιο γρήγορα με αποτέλεσμα το όριο λειτουργίας (χρονικό όριο 30 λεπτών) στον αυτοματισμό να μην επαρκεί. Στο σύνολο της η υλοποίηση παρόλα, κατά τη διάρκεια του τριμήνου δεν παρέμεινε στάσιμη. Οι αυτοματισμοί υπέστησαν βελτιώσεις και αλλαγές ώστε να εξυπηρετούν επαρκώς τις ανάγκες των ενοίκων καθώς ταυτόχρονα γίνονταν ενεργειακά αποδοτικότεροι.

Πίνακας 5.3 Ώρες λειτουργίας συγκριτικά με περιβαλλοντικές συνθήκες και κόστη

Δείγμα : 8 ημερών	**12/12-20/12, Μέση θερμοκρασία 9^{O}C**		**01/12-08/12, Μέση θερμοκρασία 13 ^{O}C**		**21/01 -28/1 , Μέση θερμοκρασία 3,47 ^{O}C**		**15/2-22/2, Μέση θερμοκρασία 12,4 ^{O}C**	
	Ώρες	**Κόστος**	**Ώρες**	**Κόστος**	**Ώρες**	**Κόστος**	**Ώρες**	**Κόστος**
Office heater	96h	11,46 €	46h	5,49 €	100h	11,94 €	39h	4,66 €
Water heater	48h	5,73 €	96h	11,46 €	33h	3,94 €	25h	2,98 €
Living room heater	72h	8,59 €	40h	4,77 €	99h	11,82 €	22h	2,63 €
Bedroom heater	120h	14,32 €	110h	13,13 €	118h	14,08 €	96h	11,46 €
Dining room heater	120h	14,32 €	87h	10,38 €	132h	15,76 €	58h	6,92 €
Outside lights	96h	11,46 €	102h	12,17 €	118h	14,08 €	101h	12,06 €
ΣΥΝΟΛΑ		**65,89 €**		**57,41 €**		**71,62 €**		**40,70 €**

Τα χρονικά διαστήματα 01/12-08/12 και 15/2-22/2, παρατηρήθηκαν περίπου ίδιες καιρικές συνθήκες με τις θερμοκρασίες περιβάλλοντος να είναι κατά μέσο όρο στους 13 ^{O}C. Παρόλα αυτά, λόγω βελτιστοποίησης των αυτοματισμών και του ελέγχου που επιτεύχθηκε, είχαμε μία μείωση στην χρήση των συσκευών που αντιστοιχεί σε ένα ποσοστό 29,11% το οποίο μεταφράζεται σε μία εβδομαδιαία μείωση κόστους στα 16,71€.

Συμπερασματικά, διαπιστώθηκε ότι χρήση ενός δικτύου έξυπνων μετρητών ενέργειας δεν μείωσε απλά το ενεργειακό κόστος των συσκευών της οικίας, αλλά προσέφερε και ένα μέσο παρακολούθησης των επιμέρους καταναλώσεων των συσκευών. Οι χρήστες και ένοικοι των ηλεκτρικών συσκευών, μέσω της απομακρυσμένης διαχείρισης μπορούν να έχουν άμεση και έγκαιρη ενημέρωση του τι συμβαίνει στο σπίτι καθώς και να επέμβουν σε περίπτωση ανάγκης. Το δίκτυο αυτό, μπορεί να υλοποιήθηκε για την βελτίωση της ενεργειακής απόδοσης της οικίας αλλά η χρήση τεχνολογιών του διαδικτύου των πραγμάτων, του δίνει μεγάλη επεκτασιμότητα. Επομένως, μελλοντικά μπορούν να γίνουν πολλές βελτιώσεις και προσθήκες όπως:

- Προσθήκη αισθητήρων κίνησης ή υπερύθρων ή καμερών με οπτική αναγνώριση για εντοπισμό ανθρώπινης παρουσίας
- Ενσωμάτωση κάποιου συστήματος συναγερμού ο οποίος να συνεργάζεται με τους υπόλοιπους αυτοματισμούς.
- Ενσωμάτωση αισθητήρων ασφαλείας όπως πυρανιχνευτές και ανιχνευτές διαρροής νερού.
- Βελτίωση των αυτοματισμών

Βιβλιογραφία – Αναφορές - Διαδικτυακές Πηγές

[1] Lyndon G. and Donev Jason, "Oil crisis of the 1970s," *Energy Education*, Nov. 2016. https://energyeducation.ca/encyclopedia/Oil_crisis_of_the_1970s (accessed Jan. 24, 2022).

[2] G. T. (George T. Miller and Scott. Spoolman, *Living in the environment : concepts, connections, and solutions*. Thomson Brooks/Cole, 2009.

[3] "Electricity produced in the EU: 9% based on lignite - Products Eurostat News - Eurostat," *Eurostat*, Sep. 2020. https://ec.europa.eu/eurostat/web/products-eurostat-news/-/ddn-20200914-1 (accessed Jan. 25, 2022).

[4] "Renewable energy statistics," Jan. 2022. https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Renewable_energy_statistics#Share_of_renewable_energy_more_than_doubled_between_2004_and_2020 (accessed Jan. 25, 2022).

[5] "«Εξοικονομώ 2021», με στόχο την ενεργειακή αναβάθμιση περισσότερων από 50.000 κατοικιών | Ελληνική Κυβέρνηση." https://government.gov.gr/exikonomo-2021-me-stocho-tin-energiaki-anavathmisi-perissoteron-apo-50-000-katikion/ (accessed Jan. 25, 2022).

[6] G. Petrecca, *Energy Conversion and Management Principles and Applications*. Cham: Springer International Publishing, 2014. doi: 10.1007/978-3-319-06560-1.

[7] "Electrical Energy, Power and Charge." https://www.electronics-tutorials.ws/dccircuits/electrical-energy.html (accessed Feb. 10, 2022).

[8] "Electrical Power in AC Circuits and Reactive Power." https://www.electronics-tutorials.ws/accircuits/power-in-ac-circuits.html (accessed Feb. 10, 2022).

[9] "Σχετικά με τη ΡΑΕ - Rae Website." https://www.rae.gr/sxetika-me-ti-rae/ (accessed Feb. 10, 2022).

[10] "ΔΕΗ Ανανεώσιμες - Υπάρχουσες εγκαταστάσεις." https://ppcr.gr/el/projects/current-projects (accessed Feb. 10, 2022).

[11] "Νυχτερινό Ρεύμα με το Οικιακό Νυχτερινό Τιμολόγιο | ΔΕΗ Α.Ε." https://www.dei.gr/el/oikiakoi-pelates/timologia-1-jan-2021/oikiako-timol-me-xronoxrewsi-oik-nuxter-1-aug-2021 (accessed Feb. 10, 2022).

[12] "New EU energy labels applicable from 1 March 2021." https://ec.europa.eu/commission/presscorner/detail/en/ip_21_818 (accessed Feb. 10, 2022).

[13] "Κατανάλωση Οικιακών Συσκευών | ΔΕΗ Α.Ε." https://www.dei.gr/el/i-dei/perivallon/orthologiki-xrisi-energeiassep262013110848617am/katanalwsi-oikiakwn-suskeuwnsep262013110848673am (accessed Feb. 07, 2022).

[14] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013, doi: 10.1016/j.future.2013.01.010.

[15] "What is the Internet of Things, and how does it work?" https://www.ibm.com/blogs/internet-of-things/what-is-the-iot/ (accessed Feb. 10, 2022).

[16] M. U. Farooq, M. Waseem, M. U. Farooq, S. Mazhar, A. Khairi, and T. Kamal, "A Review on Internet of Things (IoT) Spectrum Sensing in Cognitive Radio View project A Review on Internet of Things (IoT)," *Article in International Journal of Computer Applications*, vol. 113, no. 1, 2015, doi: 10.5120/19787-1571.

[17] C. L. Hsu and J. C. C. Lin, "An empirical examination of consumer adoption of Internet of Things services: Network externalities and concern for information privacy perspectives," *Computers in Human Behavior*, vol. 62, pp. 516–527, Sep. 2016, doi: 10.1016/J.CHB.2016.04.023.

[18] D. Santos and J. C. Ferreira, "IoT power monitoring system for smart environments," *Sustainability (Switzerland)*, vol. 11, no. 19, Oct. 2019, doi: 10.3390/su11195355.

[19] X. Liu, T. Zhang, N. Hu, P. Zhang, and Y. Zhang, "The method of Internet of Things access and network communication based on MQTT," *Computer Communications*, vol. 153, pp. 169–176, Mar. 2020, doi: 10.1016/j.comcom.2020.01.044.

[20] "Wi-Fi CERTIFIED HaLow | Wi-Fi Alliance." https://www.wi-fi.org/discover-wi-fi/wi-fi-certified-halow (accessed Feb. 11, 2022).

[21] "Introduction to MQTT QoS (Quality of Service) | EMQ." https://www.emqx.com/en/blog/introduction-to-mqtt-qos (accessed Feb. 12, 2022).

[22] "Learn MQTT Protocol | EMQ." https://www.emqx.com/en/mqtt (accessed Feb. 18, 2022).

[23] Γ. Κουλούρας, "Εισαγωγή στην Νεφουπολογιστική." Αιγάλεω, 2021.

[24] "Τι είναι το Cloud; - Digital Library." https://www.interactive-education.gr/plerophorike/focus-digital-marketing/digital-library/cloud-storage.html (accessed Feb. 13, 2022).

[25] "Υπολογιστικό Νέφος (Cloud Computing) | ΕΠΣΕΤ." http://www.epset.gr/el/content/ypologistiko-nefos-cloud-computing (accessed Feb. 13, 2022).

[26] I. Youcef, "Cloud computing security," Thesis presented for obtaining the degree PhD from UTC, University of Technology of Compiègne, Compiègne, 2019.

[27] "What is Docker? | AWS." https://aws.amazon.com/docker/ (accessed Feb. 17, 2022).

[28] "IRremoteESP8266/examples at master · crankyoldgit/IRremoteESP8266." https://github.com/crankyoldgit/IRremoteESP8266/tree/master/examples (accessed Feb. 22, 2022).

[29] "How to Set Up an IR Remote and Receiver on an Arduino - Circuit Basics." https://www.circuitbasics.com/arduino-ir-remote-receiver-tutorial/ (accessed Feb. 20, 2022).

[30] S. Santos, "ESP32 Development Boards Review and Comparison - Maker Advisor," *MAKER ADVISOR*, May 31, 2021. https://makeradvisor.com/esp32-development-boards-review-comparison/ (accessed Feb. 20, 2022).

[31] "Home - Shelly Cloud." https://shelly.cloud/ (accessed Feb. 21, 2022).

[32] "Shelly 2.5 - Shelly Cloud." https://shelly.cloud/knowledge-base/devices/shelly-25/ (accessed Feb. 22, 2022).

[33] "Shelly EM - Shelly Cloud." https://shelly.cloud/knowledge-base/devices/shelly-em/ (accessed Feb. 23, 2022).

[34] "What is a Reverse Proxy Server? | NGINX." https://www.nginx.com/resources/glossary/reverse-proxy-server/ (accessed Mar. 04, 2022).

[35] S. N. Z. Naqvi, S. Yfantidou, and E. Zimányi, "Time Series Databases and InfluxDB," Bruxelles, Dec. 2017.

[36] E. Musa, G. Delač, M. Šilić, and K. Vladimir, "Comparison of Relational and Time-Series Databases for Real-Time Massive Datasets," Zagreb, 2019.

[37] M. Nasar and M. A. Kausar, "Suitability of influxdb database for iot applications," *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 10, pp. 1850–1857, Aug. 2019, doi: 10.35940/ijitee.J9225.0881019.

[38] "InfluxDB key concepts | InfluxDB OSS 1.5 Documentation." https://docs.influxdata.com/influxdb/v1.5/concepts/key_concepts/ (accessed Feb. 26, 2022).

[39] S. Venkatramulu, M. S. B. Phridviraj, C. Srinivas, and V. C. S. Rao, "Implementation of Grafana as open source visualization and query processing platform for data scientists and researchers," *Materials Today: Proceedings*, Apr. 2021, doi: 10.1016/j.matpr.2021.03.364.

[40] M. Chakraborty and A. P. Kundan, *Monitoring Cloud-Native Applications*. Apress, 2021. doi: 10.1007/978-1-4842-6888-9.

Παράρτημα Α. IRrecvDumpV2.ino

```
/*
 * IRremoteESP8266: IRrecvDumpV2 - dump details of IR codes with IRrecv
 * An IR detector/demodulator must be connected to the input kRecvPin.
 *
 * Copyright 2009 Ken Shirriff, http://arcfn.com
 * Copyright 2017-2019 David Conran
 *
 * Example circuit diagram:
 *  https://github.com/crankyoldgit/IRremoteESP8266/wiki#ir-receiving
 *
 * Changes:
 *   Version 1.2 October, 2020
 *     - Enable easy setting of the decoding tolerance value.
 *   Version 1.0 October, 2019
 *     - Internationalisation (i18n) support.
 *     - Stop displaying the legacy raw timing info.
 *   Version 0.5 June, 2019
 *     - Move A/C description to IRac.cpp.
 *   Version 0.4 July, 2018
 *     - Minor improvements and more A/C unit support.
 *   Version 0.3 November, 2017
 *     - Support for A/C decoding for some protocols.
 *   Version 0.2 April, 2017
 *     - Decode from a copy of the data so we can start capturing faster thus
 *       reduce the likelihood of miscaptures.
 * Based on Ken Shirriff's IrsendDemo Version 0.1 July, 2009,
 */

#include <Arduino.h>
#include <assert.h>
#include <IRrecv.h>
#include <IRremoteESP8266.h>
#include <IRac.h>
#include <IRtext.h>
#include <IRutils.h>

// ==================== start of TUNEABLE PARAMETERS ====================
// An IR detector/demodulator is connected to GPIO pin 14
// e.g. D5 on a NodeMCU board.
// Note: GPIO 16 won't work on the ESP8266 as it does not have interrupts.
const uint16_t kRecvPin =14;

// The Serial connection baud rate.
// i.e. Status message will be sent to the PC at this baud rate.
// Try to avoid slow speeds like 9600, as you will miss messages and
// cause other problems. 115200 (or faster) is recommended.
```

```
// NOTE: Make sure you set your Serial Monitor to the same speed.
const uint32_t kBaudRate = 115200;

// As this program is a special purpose capture/decoder, let us use a larger
// than normal buffer so we can handle Air Conditioner remote codes.
const uint16_t kCaptureBufferSize = 1024;

// kTimeout is the Nr. of milli-Seconds of no-more-data before we consider a
// message ended.
// This parameter is an interesting trade-off. The longer the timeout, the more
// complex a message it can capture. e.g. Some device protocols will send
// multiple message packets in quick succession, like Air Conditioner remotes.
// Air Coniditioner protocols often have a considerable gap (20-40+ms) between
// packets.
// The downside of a large timeout value is a lot of less complex protocols
// send multiple messages when the remote's button is held down. The gap between
// them is often also around 20+ms. This can result in the raw data be 2-3+
// times larger than needed as it has captured 2-3+ messages in a single
// capture. Setting a low timeout value can resolve this.
// So, choosing the best kTimeout value for your use particular case is
// quite nuanced. Good luck and happy hunting.
// NOTE: Don't exceed kMaxTimeoutMs. Typically 130ms.
#if DECODE_AC
// Some A/C units have gaps in their protocols of ~40ms. e.g. Kelvinator
// A value this large may swallow repeats of some protocols
const uint8_t kTimeout = 50;
#else   // DECODE_AC
// Suits most messages, while not swallowing many repeats.
const uint8_t kTimeout = 15;
#endif  // DECODE_AC
// Alternatives:
// const uint8_t kTimeout = 90;
// Suits messages with big gaps like XMP-1 & some aircon units, but can
// accidentally swallow repeated messages in the rawData[] output.
//
// const uint8_t kTimeout = kMaxTimeoutMs;
// This will set it to our currently allowed maximum.
// Values this high are problematic because it is roughly the typical boundary
// where most messages repeat.
// e.g. It will stop decoding a message and start sending it to serial at
//      precisely the time when the next message is likely to be transmitted,
//      and may miss it.

// Set the smallest sized "UNKNOWN" message packets we actually care about.
// This value helps reduce the false-positive detection rate of IR background
// noise as real messages. The chances of background IR noise getting detected
// as a message increases with the length of the kTimeout value. (See above)
// The downside of setting this message too large is you can miss some valid
```

```
// short messages for protocols that this library doesn't yet decode.
//
// Set higher if you get lots of random short UNKNOWN messages when nothing
// should be sending a message.
// Set lower if you are sure your setup is working, but it doesn't see messages
// from your device. (e.g. Other IR remotes work.)
// NOTE: Set this value very high to effectively turn off UNKNOWN detection.
const uint16_t kMinUnknownSize = 12;

// How much percentage lee way do we give to incoming signals in order to match
// it?
// e.g. +/- 25% (default) to an expected value of 500 would mean matching a
//      value between 375 & 625 inclusive.
// Note: Default is 25(%). Going to a value >= 50(%) will cause some protocols
//       to no longer match correctly. In normal situations you probably do not
//       need to adjust this value. Typically that's when the library detects
//       your remote's message some of the time, but not all of the time.
const uint8_t kTolerancePercentage = kTolerance;  // kTolerance is normally 25%

// Legacy (No longer supported!)
//
// Change to `true` if you miss/need the old "Raw Timing[]" display.
#define LEGACY_TIMING_INFO false
// ==================== end of TUNEABLE PARAMETERS ====================

// Use turn on the save buffer feature for more complete capture coverage.
IRrecv irrecv(kRecvPin, kCaptureBufferSize, kTimeout, true);
decode_results results;  // Somewhere to store the results

// This section of code runs only once at start-up.
void setup() {
#if defined(ESP8266)
  Serial.begin(kBaudRate, SERIAL_8N1, SERIAL_TX_ONLY);
#else  // ESP8266
  Serial.begin(kBaudRate, SERIAL_8N1);
#endif  // ESP8266
  while (!Serial)  // Wait for the serial connection to be establised.
    delay(50);
  // Perform a low level sanity checks that the compiler performs bit field
  // packing as we expect and Endianness is as we expect.
  assert(irutils::lowLevelSanityCheck() == 0);

  Serial.printf("\n" D_STR_IRRECVDUMP_STARTUP "\n", kRecvPin);
#if DECODE_HASH
  // Ignore messages with less than minimum on or off pulses.
  irrecv.setUnknownThreshold(kMinUnknownSize);
#endif  // DECODE_HASH
  irrecv.setTolerance(kTolerancePercentage);  // Override the default tolerance.
```

```
  irrecv.enableIRIn();  // Start the receiver
}

// The repeating section of the code
void loop() {
  // Check if the IR code has been received.
  if (irrecv.decode(&results)) {
    // Display a crude timestamp.
    uint32_t now = millis();
    Serial.printf(D_STR_TIMESTAMP " : %06u.%03u\n", now / 1000, now % 1000);
    // Check if we got an IR message that was to big for our capture buffer.
    if (results.overflow)
      Serial.printf(D_WARN_BUFFERFULL "\n", kCaptureBufferSize);
    // Display the library version the message was captured with.
    Serial.println(D_STR_LIBRARY "   : v" _IRREMOTEESP8266_VERSION_ "\n");
    // Display the tolerance percentage if it has been change from the default.
    if (kTolerancePercentage != kTolerance)
      Serial.printf(D_STR_TOLERANCE " : %d%%\n", kTolerancePercentage);
    // Display the basic output of what we found.
    Serial.print(resultToHumanReadableBasic(&results));
    // Display any extra A/C info if we have it.
    String description = IRAcUtils::resultAcToString(&results);
    if (description.length()) Serial.println(D_STR_MESGDESC ": " + description);
    yield();  // Feed the WDT as the text output can take a while to print.
#if LEGACY_TIMING_INFO
    // Output legacy RAW timing info of the result.
    Serial.println(resultToTimingInfo(&results));
    yield();  // Feed the WDT (again)
#endif  // LEGACY_TIMING_INFO
    // Output the results as source code
    Serial.println(resultToSourceCode(&results));
    Serial.println();    // Blank line between entries
    yield();             // Feed the WDT (again)
  }
}
```

Παράρτημα Β. Κωδικοποιήσεις τηλεχειριστηρίων κλιματιστικών

Τηλεχειριστήριο Υπνοδωματίου	
Λειτουργία θέρμανσης	uint16_t rawData[227] = {3394, 1652, 482, 1272, 464, 1268, 462, 442, 460, 442, 464, 440, 460, 1272, 466, 436, 460, 458, 460, 1272, 460, 1272, 464, 440, 460, 1272, 464, 440, 462, 442, 456, 1272, 466, 1282, 464, 440, 462, 1270, 460, 1270, 462, 442, 462, 440, 464, 1266, 464, 440, 460, 458, 458, 1272, 464, 440, 462, 440, 464, 440, 460, 442, 460, 442, 456, 446, 460, 458, 460, 444, 458, 444, 462, 442, 458, 446, 462, 440, 460, 442, 456, 446, 460, 458, 466, 436, 460, 442, 458, 1250, 488, 438, 460, 442, 464, 1268, 460, 444, 460, 458, 466, 1266, 466, 438, 460, 444, 458, 444, 464, 416, 486, 440, 460, 444, 460, 458, 458, 444, 462, 1268, 462, 1270, 460, 444, 458, 444, 462, 442, 460, 442, 462, 456, 458,

	444, 458, 446, 456, 446, 458, 444, 462, 1270, 460, 444, 462, 1270, 464, 454, 462, 442, 460, 442, 458, 444, 462, 442, 458, 444, 460, 442, 462, 440, 458, 458, 462, 442, 460, 444, 458, 444, 460, 442, 458, 444, 464, 438, 458, 444, 460, 458, 460, 444, 462, 440, 460, 442, 456, 446, 458, 446, 460, 442, 460, 436, 466, 458, 462, 440, 458, 444, 456, 446, 462, 442, 460, 442, 452, 450, 460, 442, 460, 458, 458, 444, 460, 444, 460, 442, 462, 440, 458, 1272, 464, 444, 458, 440, 460, 1276, 460}; // MITSUBISHI112
Λειτουργία ψύξης	uint16_t rawData[227] = {3442, 1604, 524, 1208, 524, 1210, 522, 408, 496, 408, 494, 408, 494, 1208, 524, 408, 494, 424, 494, 1210, 522, 1210, 524, 408, 494, 1208, 524, 408, 494, 410, 494, 1210, 522, 1224, 522, 408, 494, 1210, 522, 1208, 524, 408, 494, 408, 494, 1210, 520, 410, 492, 424, 494, 1206, 526, 410, 492, 410, 494, 410, 494, 410, 494, 408, 494, 408, 494, 426, 492, 410, 492, 410, 494, 408, 492, 410, 494, 410, 494, 408, 494, 408, 492, 424, 494, 410, 492, 410, 492, 1210, 522, 410, 494, 408, 494, 1208, 522, 410, 494, 424, 494, 1210, 520, 1210, 522, 410, 494, 408, 494, 408, 494, 410, 494, 408, 494, 424, 494, 1208, 522, 408, 494, 408, 494, 1208, 524, 410, 494, 408, 494, 408, 494, 424, 494, 410, 494, 408, 494, 408, 494, 408, 494, 1206, 526, 408, 494, 1212, 520, 424, 494, 408, 496, 408, 496, 408, 494, 408, 494, 410, 494, 408, 494, 408, 494, 424, 494, 408, 494, 408, 494, 408, 494, 408, 496, 408, 496, 408, 496, 406, 496, 422, 494, 410, 494, 408, 498, 406, 494, 410, 494, 408, 494, 408, 496, 408, 494, 424, 494, 408, 494, 408, 494, 408, 494, 408, 496, 406, 494, 408, 494, 408, 496, 424, 494, 1208, 524, 408, 494, 1210, 522, 408, 494, 1210, 522, 408, 494, 412, 490, 1216, 520}; // MITSUBISHI112
Απενεργο-ποίηση	uint16_t rawData[227] = {3412, 1636, 488, 1242, 464, 1270, 464, 466, 462, 440, 462, 442, 462, 1242, 466, 466, 436, 480, 440, 1264, 468, 1266, 464, 466, 438, 1264, 466, 466, 462, 442, 438, 1264, 466, 1282, 464, 466, 462, 1242, 492, 1242, 486, 444, 460, 440, 462, 1270, 436, 466, 438, 480, 462, 1242, 466, 466, 438, 464, 438, 464, 462, 440, 464, 440, 462, 440, 440, 478, 462, 440, 438, 464, 464, 440, 438, 464, 440, 462, 464, 440, 464, 438, 464, 454, 440, 462, 464, 440, 462, 440, 464, 438, 462, 440, 440, 1262, 492, 440, 440, 480, 440, 1264, 464, 466, 464, 438, 462, 440, 440, 464, 462, 440, 464, 438, 462, 456, 462, 440, 464, 1240, 466, 1266, 466, 464, 464, 438, 466, 436, 440, 462, 464, 454, 440, 462, 464, 438, 464, 438, 440, 462, 440, 1266, 474, 456, 440, 1264, 488, 458, 464, 438, 440, 462, 464, 440, 462, 440, 462, 440, 440, 462, 440, 464, 462, 454, 464, 440, 464, 438, 466, 438, 466, 438, 462, 440, 440, 462, 462, 440, 464, 454, 464, 438, 440, 462, 464, 438, 464, 438, 464, 438, 468, 434, 466, 438, 440, 478, 464, 440, 466, 436, 442, 462, 464, 438, 452, 450, 440, 464, 440, 462, 466, 452, 464, 438, 466, 436, 466, 1238, 494, 1240, 468, 462, 464, 438, 466, 436, 442, 1268, 492}; // MITSUBISHI112
Τηλεχειριστήριο Σαλονιού	
Λειτουργία θέρμανσης	uint16_t rawData[275] = {3312, 1648, 392, 464, 368, 464, 366, 466, 366, 464, 366, 464, 366, 464, 368, 1280, 392, 464, 368, 464, 368, 464, 366, 464, 366, 466, 368, 464, 368, 464, 366, 464, 366, 464, 368, 464, 392, 440, 366, 1282, 414, 440, 368, 1282, 392, 462, 368, 464, 366, 464, 390, 440, 390, 442, 388, 442, 392, 440, 390, 440, 392, 440, 392, 440, 390, 1256, 418, 1254, 416, 1280, 392, 440, 392, 438, 394, 438, 392, 438, 392, 1256, 416, 438, 394, 438, 394, 1254, 444, 410, 394, 438, 394, 436, 394, 438, 394, 1252, 420, 436, 394, 436, 396, 436, 394, 1252, 420, 1250, 424, 434, 396, 1250, 422, 434, 396, 436, 394, 436, 394, 436, 394, 436, 396, 436, 394, 436, 396, 436, 392, 1256, 418, 436, 394, 1254, 420, 1254, 418, 436, 392, 440, 392, 438, 392, 1256, 418, 436, 392, 440, 390, 440, 392,

	438, 392, 438, 392, 440, 392, 438, 394, 438, 390, 440, 392, 438, 392, 440, 392, 438, 390, 442, 392, 1256, 418, 438, 368, 1280, 414, 1256, 414, 440, 368, 464, 366, 464, 366, 466, 366, 470, 362, 464, 366, 466, 366, 464, 368, 464, 366, 466, 364, 466, 366, 464, 366, 466, 366, 464, 364, 466, 366, 464, 366, 466, 364, 468, 364, 466, 366, 1286, 386, 466, 364, 466, 364, 466, 364, 466, 364, 468, 364, 468, 364, 466, 364, 468, 364, 466, 366, 466, 366, 466, 364, 466, 364, 468, 364, 468, 364, 466, 364, 468, 364, 466, 364, 468, 364, 466, 364, 468, 364, 1286, 386, 1284, 388, 1286, 386, 1284, 388, 1284, 388, 1284, 388, 464, 366, 464, 366}; // UNKNOWN 9D288BB4
Λειτουργία ψύξης	uint16_t rawData[275] = {3282, 1676, 388, 468, 382, 448, 360, 470, 382, 450, 362, 470, 362, 470, 360, 1288, 386, 468, 362, 468, 362, 470, 360, 470, 382, 450, 362, 470, 362, 470, 382, 438, 370, 470, 382, 450, 366, 466, 382, 1266, 386, 468, 362, 1286, 386, 468, 362, 470, 362, 470, 360, 470, 362, 470, 362, 468, 364, 468, 362, 470, 382, 448, 362, 470, 382, 1266, 388, 1284, 406, 1266, 386, 470, 362, 468, 362, 468, 362, 468, 362, 1288, 384, 470, 362, 468, 362, 1310, 362, 468, 362, 470, 382, 448, 362, 470, 362, 468, 362, 1288, 384, 468, 360, 470, 362, 1284, 388, 470, 360, 470, 362, 1288, 384, 470, 362, 470, 362, 470, 362, 470, 362, 470, 362, 470, 362, 470, 362, 1288, 384, 470, 362, 1286, 386, 1284, 388, 468, 360, 470, 360, 470, 362, 1288, 384, 470, 360, 470, 362, 470, 362, 470, 360, 470, 362, 470, 362, 470, 362, 470, 362, 470, 360, 470, 362, 468, 362, 468, 362, 470, 360, 1288, 384, 470, 362, 1284, 388, 1284, 388, 474, 356, 470, 362, 468, 362, 470, 362, 470, 362, 468, 362, 470, 362, 470, 362, 468, 364, 468, 362, 470, 362, 468, 362, 470, 360, 470, 362, 468, 362, 470, 362, 470, 362, 1310, 362, 470, 362, 1286, 386, 468, 360, 470, 362, 468, 362, 470, 360, 470, 362, 470, 362, 470, 362, 470, 362, 470, 362, 468, 362, 470, 366, 466, 362, 468, 362, 470, 362, 468, 362, 468, 362, 470, 362, 470, 360, 470, 382, 448, 360, 470, 360, 470, 382, 450, 360, 1288, 384, 1312, 382, 1266, 406, 1264, 386, 468, 366, 466, 362}; // UNKNOWN 56974786
Απενεργο-ποίηση	uint16_t rawData[275] = {3376, 1584, 482, 374, 454, 378, 452, 376, 456, 376, 456, 374, 456, 376, 456, 1190, 482, 374, 454, 376, 456, 374, 456, 376, 456, 376, 456, 376, 454, 376, 454, 376, 456, 376, 454, 376, 456, 376, 456, 1192, 480, 376, 458, 1192, 478, 376, 454, 376, 456, 374, 456, 376, 456, 374, 454, 376, 458, 374, 456, 374, 456, 376, 454, 376, 456, 1192, 480, 1192, 480, 1190, 482, 374, 456, 376, 456, 376, 456, 376, 454, 1192, 480, 376, 456, 1192, 480, 376, 454, 376, 456, 374, 456, 374, 456, 374, 456, 376, 454, 1192, 482, 374, 456, 376, 456, 1192, 480, 376, 456, 376, 456, 1190, 478, 378, 456, 376, 454, 376, 456, 376, 456, 376, 454, 376, 458, 374, 456, 374, 456, 1192, 480, 374, 456, 1192, 482, 1190, 482, 374, 456, 376, 456, 376, 454, 376, 454, 376, 456, 374, 456, 376, 456, 374, 454, 376, 456, 376, 456, 376, 456, 378, 452, 376, 456, 374, 456, 374, 456, 376, 456, 376, 454, 1194, 478, 374, 456, 1192, 480, 1194, 478, 376, 456, 376, 456, 376, 456, 374, 456, 376, 456, 376, 454, 376, 454, 376, 456, 374, 456, 376, 456, 376, 456, 376, 456, 376, 456, 376, 456, 376, 456, 374, 454, 376, 456, 1192, 480, 376, 454, 1192, 480, 376, 454, 376, 454, 376, 456, 376, 456, 376, 456, 374, 456, 374, 456, 376, 454, 376, 456, 376, 454, 376, 454, 378, 456, 376, 454, 376, 456, 376, 454, 376, 454, 378, 454, 376, 454, 376, 456, 376, 454, 376, 454, 1192, 480, 378, 454, 376, 456, 1192, 480, 1194, 478, 1192, 480, 376, 456, 376, 454}; // UNKNOWN F6CFCC5E

Παράρτημα Γ. Κώδικες συσκευών ελέγχου κλιματιστικών και περιβαλλοντικών συνθηκών ανά δωμάτιο/συσκευή.

- **Υπνοδωμάτιο**

```
//DEFINING TEMP/HUMIDITY SENSOR COMPONENTS-------------------------
#include "DHT.h"
#define DHTPIN D4
//NOTE: uncomment the temperature/humidity sensor you will use
//#define DHTTYPE DHT11   // DHT 11
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
DHT dht(DHTPIN, DHTTYPE);
//DEFINING IR SENSOR COMPONENTS-----------------------------------
#include <Arduino.h>
#include <IRremoteESP8266.h>
#include <IRsend.h>
const uint16_t kIrLed = D2;
IRsend irsend(kIrLed);
//RAW DATA PER A/C OPERATION---------------------------------------
//NOTE: Raw data retrieved from IRrecvDumpV2.ino
//AC model: Firstline(mitsubishi protocol)
//Room:Bedroom
//ON COOLING
uint16_t  cool[227]  =  {3442,  1604,   524,  1208,   524,  1210,   522,  408,   496,
408,  494,  408,  494,  1208,  524,  408,  494,  424,  494,  1210,  522,  1210,  524,
408,  494,  1208,  524,  408,  494,  410,  494,  1210,  522,  1224,  522,  408,  494,
1210,  522,  1208,  524,  408,  494,  408,  494,  1210,  520,  410,  492,  424,  494,
1206,  526,  410,  492,  410,  494,  410,  494,  410,  494,  408,  494,  408,  494,
426,  492,  410,  492,  410,  494,  408,  492,  410,  494,  410,  494,  408,  494,
408,  492,  424,  494,  410,  492,  410,  492,  1210,  522,  410,  494,  408,  494,
1208,  522,  410,  494,  424,  494,  1210,  520,  1210,  522,  410,  494,  408,  494,
408,  494,  410,  494,  408,  494,  424,  494,  1208,  522,  408,  494,  408,  494,
1208,  524,  410,  494,  408,  494,  408,  494,  424,  494,  410,  494,  408,  494,
408,  494,  408,  494,  1206,  526,  408,  494,  1212,  520,  424,  494,  408,  496,
408,  496,  408,  494,  408,  494,  410,  494,  408,  494,  408,  494,  424,  494,
408,  494,  408,  494,  408,  494,  408,  496,  408,  496,  408,  496,  406,  496,
422,  494,  410,  494,  408,  498,  406,  494,  410,  494,  408,  494,  408,  496,
408,  494,  424,  494,  408,  494,  408,  494,  408,  494,  408,  496,  406,  494,
408,  494,  408,  496,  424,  494,  1208,  524,  408,  494,  1210,  522,  408,  494,
1210,  522,  408,  494,  412,  490,  1216,  520};  // MITSUBISHI112
//TURN OFF
uint16_t  turnoff[227]  =  {3412,  1636,  488,  1242,  464,  1270,  464,  466,  462,
440,  462,  442,  462,  1242,  466,  466,  436,  480,  440,  1264,  468,  1266,  464,
466,  438,  1264,  466,  466,  462,  442,  438,  1264,  466,  1282,  464,  466,  462,
1242,  492,  1242,  486,  444,  460,  440,  462,  1270,  436,  466,  438,  480,  462,
1242,  466,  466,  438,  464,  438,  464,  462,  440,  464,  440,  462,  440,  440,
478,  462,  440,  438,  464,  464,  440,  438,  464,  440,  462,  464,  440,  464,
438,  464,  454,  440,  462,  464,  440,  462,  440,  464,  438,  462,  440,  440,
```

```
1262,  492, 440,  440, 480,  440, 1264,  464, 466,  464, 438,  462, 440,  440,
464,  462, 440,  464, 438,  462, 456,  462, 440,  464, 1240,  466, 1266,  466,
464,  464, 438,  466, 436,  440, 462,  464, 454,  440, 462,  464, 438,  464,
438,  440, 462,  440, 1266,  474, 456,  440, 1264,  488, 458,  464, 438,  440,
462,  464, 440,  462, 440,  462, 440,  440, 462,  440, 464,  462, 454,  464,
440,  464, 438,  466, 438,  466, 438,  462, 440,  440, 462,  462, 440,  464,
454,  464, 438,  440, 462,  464, 438,  464, 438,  464, 438,  468, 434,  466,
438,  440, 478,  464, 440,  466, 436,  442, 462,  464, 438,  452, 450,  440,
464,  440, 462,  466, 452,  464, 438,  466, 436,  466, 1238,  494, 1240,  468,
462,  464, 438,  466, 436,  442, 1268,  492};  // MITSUBISHI112
//ON HEATING
uint16_t  heat[227]  =  {3394,  1652,  482,  1272,  464,  1268,  462,  442,  460,
442,  464,  440,  460,  1272,  466,  436,  460,  458,  460,  1272,  460,  1272,  464,
440,  460,  1272,  464,  440,  462,  442,  456,  1272,  466,  1282,  464,  440,  462,
1270,  460,  1270,  462,  442,  462,  440,  464,  1266,  464,  440,  460,  458,  458,
1272,  464,  440,  462,  440,  464,  440,  460,  442,  460,  442,  456,  446,  460,
458,  460,  444,  458,  444,  462,  442,  458,  446,  462,  440,  460,  442,  456,
446,  460,  458,  466,  436,  460,  442,  458,  1250,  488,  438,  460,  442,  464,
1268,  460,  444,  460,  458,  466,  1266,  466,  438,  460,  444,  458,  444,  464,
416,  486,  440,  460,  444,  460,  458,  458,  444,  462,  1268,  462,  1270,  460,
444,  458,  444,  462,  442,  460,  442,  462,  456,  458,  444,  458,  446,  456,
446,  458,  444,  462,  1270,  460,  444,  462,  1270,  464,  454,  462,  442,  460,
442,  458,  444,  462,  442,  458,  444,  460,  442,  462,  440,  458,  458,  462,
442,  460,  444,  458,  444,  460,  442,  458,  444,  464,  438,  458,  444,  460,
458,  460,  444,  462,  440,  460,  442,  456,  446,  458,  446,  460,  442,  460,
436,  466,  458,  462,  440,  458,  444,  456,  446,  462,  442,  460,  442,  452,
450,  460,  442,  460,  458,  458,  444,  460,  444,  460,  442,  462,  440,  458,
1272,  464,  444,  458,  440,  460,  1276,  460};  // MITSUBISHI112
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
//WIFI credentials, MQTT broker IP and other Components
const char* ssid = "Saloni 2.4G";
const char* password = "6974253002";
const char* mqtt_server = "83.212.77.18";
WiFiClient espBedroom3;
PubSubClient client(espBedroom3);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE  (50)
char msg[MSG_BUFFER_SIZE];
int value = 0;
void setup() {
  Serial.begin(115200);
  //-----------------------------------------//
  dht.begin();
  //-----------------------------------------//
  irsend.begin();
#if ESP8266
  Serial.begin(115200, SERIAL_8N1, SERIAL_TX_ONLY);
```

```
#else  // ESP8266
  Serial.begin(115200, SERIAL_8N1);
#endif  // ESP8266
  //-----------------------------------------//
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
//-----------------------------------------//
// We start by connecting to a WiFi network
void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
//---------------------END OF VOID *SETUP WIFI*-------------------------------//
//-------------------BEGINING OF VOID *CALLBACK*-----------------------------//
void callback(char* topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String operation;
  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    operation += (char)message[i];
  }
  Serial.println();
  // Feel free to add more if statements to control more GPIOs with MQTT
  // If a message is received on the topic "Bedroom/AC", you check if the message
is either «heat ,cool or off».
  if (String(topic) == "Bedroom/AC") {
    Serial.print("Changing output to ");
    if (operation == "cool") {
      Serial.println("Power: On, Mode: 1 (Cool), Temp: 23C, Fan: 0 (Auto),Swing:
Off");
      irsend.sendRaw(cool, 227, 38);  // Send a raw data capture at 38kHz.
    }
    else if (operation == "heat") {
```

```
        Serial.println("AC Power: On, Mode: 4 (Heat), Temp: 26C, Fan: 0
(Auto), Swing: Off");
      irsend.sendRaw(heat, 227, 38);  // Send a raw data capture at 38kHz.
    }
    else if (operation == "off") {
      Serial.println("AC Power: Off,");
      irsend.sendRaw(turnoff, 227, 38);  // Send a raw data capture at 38kHz.
    }
  }
}
//-------------------END OF VOID *CALLBACK*---------------------------------//
//----------------------BEGINING OF VOID *RECONNECT*-------------------------//
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("espBedroom3", "msciot20007", "geopili07!")) {
      Serial.println("connected");
      // Subscribe to ALL available TOPICS
      client.subscribe("Bedroom/temperature");
      client.subscribe("Bedroom/humidity");
      client.subscribe("Bedroom/heat_index");
      client.subscribe("Bedroom/AC");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
//----------------------END OF VOID *RECONNECT*--------------------------------
//
//-------------------BEGINING OF VOID *LOOP*-------------------------------
//
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  long now = millis();
  if (now - lastMsg > 5000) {
    lastMsg = now;
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t)) {
```

```
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, true);
  //serial print of meausurements for debuging throught serial monitor
  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print("°C ");
  Serial.print(F(" Heat index: "));
  Serial.print(hic);
  Serial.println(F("°C "));
  //Publishing measurements with MQTT
  // Publish a message to "Bedroom/temperature"
  client.publish("Bedroom/humidity", String(h).c_str(), false); // You can acti-
vate the retain flag by setting the third parameter to true
  // Publish a message to "Bedroom/humidity"
  client.publish("Bedroom/temperature", String(t).c_str(), false); // You can ac-
tivate the retain flag by setting the third parameter to true
  // Publish a message to "Bedroom/heat_index"
   client.publish("Bedroom/heat_index", String(hic).c_str(), false); // You can
activate the retain flag by setting the third parameter to true
  }
}
//-------------------END OF VOID *LOOP*-------------------------------------//
```

- **Σαλόνι**

```
//DEFINING TEMP/HUMIDITY SENSOR COMPONENTS-----------------------------------
-
#include "DHT.h"
#define DHTPIN D4
//NOTE: uncomment the temperature/humidity sensor you will use
//define DHTTYPE DHT11   // DHT 11
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
DHT dht(DHTPIN, DHTTYPE);
//DEFINING IR SENSOR COMPONENTS---------------------------------------------
-
#include <Arduino.h>
#include <IRremoteESP8266.h>
#include <IRsend.h>
const uint16_t kIrLed = D2;
IRsend irsend(kIrLed);
//RAW DATA PER A/C OPERATION-------------------------------------------------
//NOTE: Raw data retrieved from IRrecvDumpV2.ino
//AC model: Firstline
//Room:LivingRoom
```

```
//ON COOLING
uint16_t cool[275] = {3282, 1676,  388, 468,  382, 448,  360, 470,  382, 450,  362,
470,  362, 470,  360, 1288,  386, 468,  362, 468,  362, 470,  360, 470,  382,
450,  362, 470,  362, 470,  382, 438,  370, 470,  382, 450,  366, 466,  382,
1266,  386, 468,  362, 1286,  386, 468,  362, 470,  362, 470,  360, 470,  362,
470,  362, 468,  364, 468,  362, 470,  382, 448,  362, 470,  382, 1266,  388,
1284,  406, 1266,  386, 470,  362, 468,  362, 468,  362, 468,  362, 1288,  384,
470,  362, 468,  362, 1310,  362, 468,  362, 470,  382, 448,  362, 470,  362,
468,  362, 1288,  384, 468,  360, 470,  362, 1284,  388, 470,  360, 470,  362,
1288,  384, 470,  362, 470,  362, 470,  362, 470,  362, 470,  362, 470,  362,
470,  362, 470,  362, 1288,  384, 470,  362, 1286,  386, 1284,  388, 468,  360,
470,  360, 470,  362, 1288,  384, 470,  360, 470,  362, 470,  362, 470,  360,
470,  362, 470,  362, 470,  362, 470,  362, 470,  360, 470,  362, 468,  362,
468,  362, 470,  360, 1288,  384, 470,  362, 1284,  388, 1284,  388, 474,  356,
470,  362, 468,  362, 470,  362, 470,  362, 468,  362, 470,  362, 470,  362,
468,  364, 468,  362, 470,  362, 468,  362, 470,  360, 470,  362, 468,  362,
470,  362, 470,  362, 1310,  362, 470,  362, 1286,  386, 468,  360, 470,  362,
468,  362, 470,  360, 470,  362, 470,  362, 470,  362, 470,  362, 470,  362,
468,  362, 470,  366, 466,  362, 468,  362, 470,  362, 468,  362, 468,  362,
470,  362, 470,  360, 470,  382, 448,  360, 470,  360, 470,  382, 450,  360,
1288,  384, 1312,  382, 1266,  406, 1264,  386, 468,  366, 466,  362};  // UNKNOWN
56974786
//TURN OFF
uint16_t  turnoff[275]  =  {3376,  1584,  482,  374,  454,  378,  452,  376,  456,
376,  456,  374,  456,  376,  456,  1190,  482,  374,  454,  376,  456,  374,  456,
376,  456,  376,  456,  376,  454,  376,  454,  376,  456,  376,  454,  376,  456,
376,  456,  1192,  480,  376,  458,  1192,  478,  376,  454,  376,  456,  374,  456,
376,  456,  374,  454,  376,  458,  374,  456,  374,  456,  376,  454,  376,  456,
1192,  480,  1192,  480,  1190,  482,  374,  456,  376,  456,  376,  456,  376,  454,
1192,  480,  376,  456,  1192,  480,  376,  454,  376,  456,  374,  456,  374,  456,
374,  456,  376,  454,  1192,  482,  374,  456,  376,  456,  1192,  480,  376,  456,
376,  456,  1190,  478,  378,  456,  376,  454,  376,  456,  376,  456,  376,  454,
376,  458,  374,  456,  374,  456,  1192,  480,  374,  456,  1192,  482,  1190,  482,
374,  456,  376,  456,  376,  454,  376,  454,  376,  456,  374,  456,  376,  456,
374,  454,  376,  456,  376,  456,  376,  456,  378,  452,  376,  456,  374,  456,
374,  456,  376,  456,  376,  454,  1194,  478,  374,  456,  1192,  480,  1194,  478,
376,  456,  376,  456,  376,  456,  374,  456,  376,  456,  376,  454,  376,  454,
376,  456,  374,  456,  376,  456,  376,  456,  376,  456,  376,  456,  376,  456,
376,  456,  374,  454,  376,  456,  1192,  480,  376,  454,  1192,  480,  376,  454,
376,  454,  376,  456,  376,  456,  376,  456,  374,  456,  374,  456,  376,  454,
376,  456,  376,  454,  376,  454,  378,  456,  376,  454,  376,  456,  376,  454,
376,  454,  378,  454,  376,  454,  376,  456,  376,  454,  376,  454,  1192,  480,
378,  454,  376,  456,  1192,  480,  1194,  478,  1192,  480,  376,  456,  376,  454};  //
UNKNOWN F6CFCC5E
//ON HEATING
uint16_t heat[275] = {3312, 1648,  392, 464,  368, 464,  366, 466,  366, 464,  366,
464,  366, 464,  368, 1280,  392, 464,  368, 464,  368, 464,  366, 464,  366,
466,  368, 464,  368, 464,  366, 464,  366, 464,  368, 464,  392, 440,  366,
```

```
1282,  414, 440,  368, 1282,  392, 462,  368, 464,  366, 464,  390, 440,  390,
442,  388, 442,  392, 440,  390, 440,  392, 440,  392, 440,  390, 1256,  418,
1254,  416, 1280,  392, 440,  392, 438,  394, 438,  392, 438,  392, 1256,  416,
438,  394, 438,  394, 1254,  444, 410,  394, 438,  394, 436,  394, 438,  394,
1252,  420, 436,  394, 436,  396, 436,  394, 1252,  420, 1250,  424, 434,  396,
1250,  422, 434,  396, 436,  394, 436,  394, 436,  394, 436,  396, 436,  394,
436,  396, 436,  392, 1256,  418, 436,  394, 1254,  420, 1254,  418, 436,  392,
440,  392, 438,  392, 1256,  418, 436,  392, 440,  390, 440,  392, 438,  392,
438,  392, 440,  392, 438,  394, 438,  390, 440,  392, 438,  392, 440,  392,
438,  390, 442,  392, 1256,  418, 438,  368, 1280,  414, 1256,  414, 440,  368,
464,  366, 464,  366, 466,  366, 470,  362, 464,  366, 466,  366, 464,  368,
464,  366, 466,  364, 466,  366, 464,  366, 466,  366, 464,  364, 466,  366,
464,  366, 466,  364, 468,  364, 466,  366, 1286,  386, 466,  364, 466,  364,
466,  364, 466,  364, 466,  364, 468,  364, 468,  364, 466,  364, 468,  364,
466,  366, 466,  366, 466,  364, 466,  364, 468,  364, 468,  364, 466,  364,
468,  364, 466,  364, 468,  364, 466,  364, 468,  364, 1286,  386, 1284,  388,
1286,  386, 1284,  388, 1284,  388, 1284,  388, 464,  366, 464,  366};  // UNKNOWN
9D288BB4
//DEFINING MQTT and WiFi COMPONENTS---------------------------------------
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
//WIFI credentials, MQTT broker IP and other Components
const char* ssid = "Saloni 2.4G";
const char* password = "6974253002";
const char* mqtt_server = "83.212.77.18";
WiFiClient espLivingRoom1;
PubSubClient client(espLivingRoom1);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE  (50)
char msg[MSG_BUFFER_SIZE];
int value = 0;
void setup() {
  Serial.begin(115200);
  //-----------------------------------------//
  dht.begin();
  //-----------------------------------------//
  irsend.begin();
#if ESP8266
  Serial.begin(115200, SERIAL_8N1, SERIAL_TX_ONLY);
#else  // ESP8266
  Serial.begin(115200, SERIAL_8N1);
#endif  // ESP8266
  //-----------------------------------------//
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
//-----------------------------------------//
```

```
// Beginning by connecting to a WiFi network
void setup_wifi() {
  delay(10);
    Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
//---------------------END OF VOID *SETUP WIFI*--------------------------------
//
//-------------------BEGINING OF VOID *CALLBACK*-----------------------------
//
void callback(char* topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String operation;
  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    operation += (char)message[i];
  }
  Serial.println();
  // Feel free to add more if statements to control more GPIOs with MQTT
   // If a message is received on the topic "LivingRoom/AC", you check if the
message is either « heat ,cool or «off».
  // Changes the output state according to the message
  if (String(topic) == "LivingRoom/AC") {
    Serial.print("Changing output to ");
    if (operation == "cool") {
      Serial.println("Power: On, Mode: 1 (Cool), Temp: 23C, Fan: 0 (Auto),Swing:
Off");
      irsend.sendRaw(cool, 275, 38);  // Send a raw data capture at 38kHz.
    }
    else if (operation == "heat") {
        Serial.println("AC  Power:  On,  Mode:  4  (Heat),  Temp:  26C,  Fan:  0
(Auto),  Swing: Off");
      irsend.sendRaw(heat, 275, 38);  // Send a raw data capture at 38kHz.
    }
    else if (operation == "off") {
      Serial.println("AC Power: Off,");
```

```
      irsend.sendRaw(turnoff, 275, 38);  // Send a raw data capture at 38kHz.
    }
  }
}
//-------------------END OF VOID *CALLBACK*-------------------------------------
-//
//----------------------BEGINING OF VOID *RECONNECT*----------------------------
-//
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("espLivingRoom1", "msciot20007", "geopili07!")) {
      Serial.println("connected");
      // Subscribe to ALL available TOPICS
      client.subscribe("LivingRoom/temperature");
      client.subscribe("LivingRoom/humidity");
      client.subscribe("LivingRoom/heat_index");
      client.subscribe("LivingRoom/AC");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
//----------------------END OF VOID *RECONNECT*--------------------------------
-//
//-------------------BEGINING OF VOID *LOOP*-------------------------------------
//
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  long now = millis();
  if (now - lastMsg > 5000) {
    lastMsg = now;
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  // Compute heat index in Celsius (isFahreheit = false)
```

```
  float hic = dht.computeHeatIndex(t, h, true);
  //serial print of meausurements for debuging throught serial monitor
  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print("°C ");
  Serial.print(F(" Heat index: "));
  Serial.print(hic);
  Serial.println(F("°C "));
  //Publishing measurements with MQTT
  // Publish a message to "LivingRoom/temperature"
  client.publish("LivingRoom/humidity", String(h).c_str(), false); // You can ac-
tivate the retain flag by setting the third parameter to true
  // Publish a message to "LivingRoom/humidity"
  client.publish("LivingRoom/temperature", String(t).c_str(), false); // You can
activate the retain flag by setting the third parameter to true
  // Publish a message to "LivingRoom/heat_index"
  client.publish("LivingRoom/heat_index", String(hic).c_str(), false); // You can
activate the retain flag by setting the third parameter to true
  }
}
//-------------------END OF VOID *LOOP*-----------------------------------//
```

- **Γραφείο**

```
//DEFINING TEMP/HUMIDITY SENSOR COMPONENTS-------------------------------------
--
#include "DHT.h"
#define DHTPIN D4
//NOTE: uncomment the temperature/humidity sensor you will use
//#define DHTTYPE DHT11   // DHT 11
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
DHT dht(DHTPIN, DHTTYPE);
//DEFINING IR SENSOR COMPONENTS------------------------------------------------
---
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
//WIFI credentials, MQTT broker IP and other Components
const char* ssid = "GINA";
const char* password = "6974253002";
const char* mqtt_server = "83.212.77.18";
WiFiClient espOffice;
PubSubClient client(espOffice);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE  (50)
char msg[MSG_BUFFER_SIZE];
int value = 0;
```

```
void setup() {
  Serial.begin(115200);
  //-------------------------------------------//
  dht.begin();
  //-------------------------------------------//
#if ESP8266
  Serial.begin(115200, SERIAL_8N1, SERIAL_TX_ONLY);
#else  // ESP8266
  Serial.begin(115200, SERIAL_8N1);
#endif  // ESP8266
  //-------------------------------------------//
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
//-------------------------------------------//
// We start by connecting to a WiFi network
void setup_wifi() {
  delay(10);
    Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
//---------------------END OF VOID *SETUP WIFI*---------------------------------
//
//-------------------BEGINING OF VOID *CALLBACK*-----------------------------
//
void callback(char* topic, byte* message, unsigned int length) {
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String operation;
  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    operation += (char)message[i];
  }
  Serial.println();
}
```

```
//-------------------END OF VOID *CALLBACK*-----------------------------------
//
//----------------------BEGINING OF VOID *RECONNECT*---------------------------
//
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Attempt to connect
    if (client.connect("espOffice", "msciot20007", "geopili07!")) {
      Serial.println("connected");
      // Subscribe to ALL available TOPICS
      client.subscribe("Office/temperature");
      client.subscribe("Office/humidity");
      client.subscribe("Office/heat_index");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

//----------------------END OF VOID *RECONNECT*---------------------------------
//
//-------------------BEGINING OF VOID *LOOP*-----------------------------------
//
void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  long now = millis();
  if (now - lastMsg > 5000) {
    lastMsg = now;
  float h = dht.readHumidity();
  float t = dht.readTemperature()-10;
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, true);
  //serial print of meausurements for debuging throught serial monitor
  Serial.print(F("Humidity: "));
  Serial.print(h);
```

```
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print("°C ");
  Serial.print(F(" Heat index: "));
  Serial.print(hic);
  Serial.println(F("°C "));
  //Publishing measurements with MQTT
  // Publish a message to "Office/temperature"
  client.publish("Office/humidity", String(h).c_str(), false); // You can activate
the retain flag by setting the third parameter to true
  // Publish a message to "Office/humidity"
  client.publish("Office/temperature", String(t).c_str(), false); // You can ac-
tivate the retain flag by setting the third parameter to true
  // Publish a message to "espOffice/heat_index"
  client.publish("Office/heat_index", String(hic).c_str(), false); // You can ac-
tivate the retain flag by setting the third parameter to true
  }
}
//-------------------END OF VOID *LOOP*-------------------------------------//
```

Παράρτημα Δ. Home assistant - Configuration.yaml

```
##Configures a default setup of Home Assistant (frontend, api, etc)
default_config:
# Text to speech
tts:
  - platform: google_translate
group: !include groups.yaml
automation: !include automations.yaml
script: !include scripts.yaml
scene: !include scenes.yaml
mqtt:
  broker: "83.212.77.18"
  username: msciot20007
  password: !secret mqtt_pass
panel_iframe:
  grafana:
    title: 'Grafana'
    url: 'http://83.212.77.18:3000'
    icon: mdi:chart-areaspline
# UTILITY METER ENTITIES AND SENSORS---------------
utility_meter:
  daily_energy:
    source: sensor.energy_pow
    cycle: daily
    tariffs:
```

```
      - peak
      - offpeak
  monthly_energy:
    source: sensor.energy_pow
    cycle: monthly
    tariffs:
      - peak
      - offpeak
  current_monthly_energy:
    source: sensor.energy_pow
    cycle: monthly
influxdb:
  host: 83.212.77.18
  port: 8086
  username: admin
  password: !secret influx_pass
sensor:
  - platform: integration
    source: sensor.shelly_shem_c45bbe6c4d01_1_current_consumption
    name: energy_pow
    unit_prefix: k
    round: 2
  - platform: integration
    source: sensor.outside_lights_power_consumption
    name: outside_lights_total_energy_consumption
    unit_prefix: k
    round: 2
  - platform: integration
    source: sensor.power_water_heater
    name: Total_water_heater_energy_consumption
    unit_prefix: k
    round: 2
#Monthly Costs-------------------------------------------------------------------------
  - platform: template
    sensors:
      monthly_energy_cost_peak:
        unit_of_measurement: '€'
        friendly_name: Monthly Peak Tariff Cost
         value_template: "{{ (states('sensor.monthly_energy_peak') | float * 0.11936 ) |
round(2) }}"
  - platform: template
    sensors:
      monthly_energy_cost_offpeak:
        unit_of_measurement: '€'
         value_template: "{{ (states('sensor.monthly_energy_offpeak') | float * 0.07897 )
| round(2) }}"
        friendly_name: Monthly Offpeak Tariff Cost
  - platform: template
```

```
    sensors:
      monthly_energy_cost_all:
         value_template: "{{ ((states('sensor.monthly_energy_peak') | float *0.11936) +
(states('sensor.monthly_energy_offpeak') | float /1000 | float *0.07897 )) | round(2) }}"
        friendly_name: Monthly Total Cost
        unit_of_measurement: '€'
#Daily Costs-----------------------------------------------------------------------
  - platform: template
    sensors:
      daily_energy_cost_peak:
        unit_of_measurement: '€'
         value_template: "{{ (states('sensor.daily_energy_peak') | float * 0.11936 ) |
round(2) }}"
        friendly_name: Today's Peak Tariff Cost
  - platform: template
    sensors:
      daily_energy_cost_offpeak:
        unit_of_measurement: '€'
        value_template: "{{ (states('sensor.daily_energy_offpeak') | float * 0.07897 ) |
round(2) }}"
        friendly_name: Today's Offpeak Tariff Cost
  - platform: template
    sensors:
      daily_energy_cost_all:
         value_template: "{{ ((states('sensor.daily_energy_peak') | float *0.11936) +
(states('sensor.daily_energy_offpeak') | float *0.07897 )) | round(2) }}"
        friendly_name: Today's Cost
        unit_of_measurement:  '€'
# Manual correction at current energy consumption measurement of Water Heater-------
  - platform: template
    sensors:
      power_water_heater:
        friendly_name: "Water heater power consumption"
        unit_of_measurement: 'W'
        value_template: >
          {% if is_state('sensor.shelly_shsw_pm_98cdac1f4e69_current_consumption', 0) %}
           {{ states('sensor.shelly_shsw_pm_98cdac1f4e69_current_consumption') | int* 0}}
          {% else %}
            {{ states('sensor.shelly_shsw_pm_98cdac1f4e69_current_consumption') | float*
3250}}
          {% endif %}
#Office sensors--------------------------------------------------------------------
  - platform: mqtt
    device_class: "temperature"
    name: "Office Temperature"
    state_topic: "Office/temperature"
    unit_of_measurement: '°C'
  - platform: mqtt
```

```
    device_class: "humidity"
    name: "Office Humidity"
    state_topic: "Office/humidity"
    unit_of_measurement: '%'
  - platform: mqtt
    device_class: "temperature"
    name: "Office Feeling"
    state_topic: "Office/heat_index"
    unit_of_measurement: '°C'
#Living Room Sensors-------------------------------
  - platform: mqtt
    device_class: "temperature"
    name: "LivingRoom Temperature"
    state_topic: "LivingRoom/temperature"
    unit_of_measurement: '°C'
  - platform: mqtt
    device_class: "humidity"
    name: "LivingRoom Humidity"
    state_topic: "LivingRoom/humidity"
    unit_of_measurement: '%'
  - platform: mqtt
    device_class: "temperature"
    name: "LivingRoom Feeling"
    state_topic: "LivingRoom/heat_index"
    unit_of_measurement: '°C'
#Bedroom Sensors-----------------------------------
  - platform: mqtt
    device_class: "temperature"
    name: "Bedroom Temperature"
    state_topic: "Bedroom/temperature"
    unit_of_measurement: '°C'
  - platform: mqtt
    device_class: "humidity"
    name: "Bedroom Humidity"
    state_topic: "Bedroom/humidity"
    unit_of_measurement: '%'
  - platform: mqtt
    device_class: "temperature"
    name: "Bedroom Feeling"
    state_topic: "Bedroom/heat_index"
    unit_of_measurement: '°C'
#outside lights  power management------------------------
  - platform: mqtt
    name: Outside Lights - power consumption
    device_class: power
    unit_of_measurement: 'W'
    state_topic: shellies/shellyswitch25-98CDAC1E50BC/relay/0/power
  - platform: mqtt
```

```
    name: Outside Lights - energy consumption since last reboot
    device_class: energy
    unit_of_measurement: 'Wh'
    value_template: "{{ value | float / 60 | round(2) }}"
    state_topic: shellies/shellyswitch25-98CDAC1E50BC/relay/0/energy
#SHELLY----------------------------------------
shelly:
  version: true
  sensors:
    - all
                                                                    cloud_auth_key:
"OGY4NTB1aWQ4BC65005DCF905BA84D9AC9003EC3C95C0D0A62D06130A391899E2D0DD32624949CD735D3AA4
C464"
  cloud_server: https://shelly-32-eu.shelly.cloud
#outside lights  switch entity-----------------------
light:
  - platform: mqtt
    unique_id: shelly25outside
    name: "Outside light"
    state_topic: "shellies/shellyswitch25-98CDAC1E50BC/relay/0"
    command_topic: "shellies/shellyswitch25-98CDAC1E50BC/relay/0/command"
    payload_on: 'on'
    payload_off: 'off'
    retain: false
    qos: 1
```

Παράρτημα Ζ. Home assistant automation.yaml

```
  #Automation of turning off All devices at Livingroom at night
- id: '1636401606273'
  alias: turn off all devices of Living Room
  description: Turns of heaters at 2 o'clock midnight
  trigger:
  - platform: time
    at: 02:00
  condition: []
  action:
  - scene: scene.switch_off_all_living_room_devices
  - service: notify.mobile_app_pocox3
    data:
      message: 'All Heaters at Living room are OFF! '
      title: The time is 2 o'clock time to sleep!
  - service: notify.mobile_app_redmi_note_4
    data:
      message: 'All heaters at Living room are OFF! '
      title: The time is 2 o'clock time to sleep!
  - delay:
      hours: 0
```

```
      minutes: 30
      seconds: 0
      milliseconds: 0
  - type: turn_off
    device_id: 08516b364b6bc743a060ba9bc59570bc
    entity_id: switch.shelly_shsw_pm_98cdac1eddb0
    domain: switch
  - service: notify.mobile_app_redmi_note_4
    data:
      message: Turned Off
      title: Heater at Office
  - service: notify.mobile_app_poco_x3
    data:
      message: Turned Off
      title: Heater at Office
  mode: restart
    #Automation time related of Water Heater
- id: '1640294571613'
  alias: Water Heater Control
  description: ''
  trigger:
  - platform: device
    type: turned_on
    device_id: b13dcffbabb360213d4235ef716800c2
    entity_id: switch.shelly_shsw_pm_98cdac1f4e69
    domain: switch
    for:
      hours: 0
      minutes: 40
      seconds: 0
      milliseconds: 0
  condition:
  - condition: state
    state: 'on'
    for: 00:30:00
    entity_id: switch.shelly_shsw_pm_98cdac1f4e69
  action:
  - type: turn_off
    device_id: b13dcffbabb360213d4235ef716800c2
    entity_id: switch.shelly_shsw_pm_98cdac1f4e69
    domain: switch
  - service: notify.mobile_app_pocox3
    data:
      message: 'Water heater Turned OFF after 30 minutes! '
  - service: notify.mobile_app_redmi_note_4
    data:
      message: 'Water heater Turned OFF after 30 minutes! '
  mode: restart
```

```
  #Automation of Livingroom/Diningroom heaters
- id: '1640295987388'
  alias: Turn on  heat mode-Livingroom Heaters
  description: ''
  trigger:
  - platform: numeric_state
    for: 00:20:00
    entity_id: sensor.livingroom_temperature
    below: '22'
  condition:
  - condition: or
    conditions:
    - condition: device
      type: is_off
      device_id: 50583267889cb5fb7e3cf2076055d829
      entity_id: switch.shelly_shsw_pm_98cdac2fc1a7
      domain: switch
    - condition: device
      type: is_off
      device_id: 7a4f793fcf4fcbdf180732a6d6cbaf2e
      entity_id: switch.shelly_shsw_pm_d8bfc01a17f0
      domain: switch
  - condition: time
    after: '12:00'
  action:
  - service: climate.set_temperature
    data:
      temperature: 25
    target:
      entity_id:
      - climate.diningroom_heater
      - climate.livingroom_heater
  - service: climate.set_hvac_mode
    data:
      hvac_mode: heat
    target:
      entity_id:
      - climate.diningroom_heater
      - climate.livingroom_heater
  - service: notify.mobile_app_poco_x3
    data:
      message: All heaters at Living room will now TURN ON!
      title: Temperature at Living room Low
  - service: notify.mobile_app_redmi_note_4
    data:
      message: All heaters at Living room will now TURN ON!
      title: Temperature at Living room Low
  mode: restart
```

```
  #Automation of Utility meter's tarif modes
- id: '1642460208149'
  alias: Peak/offpeak
  description: ''
  trigger:
  - platform: time
    at: 02:00:00
  - platform: time
    at: 08:00:00
  - platform: time
    at: '15:00:00'
  - platform: time
    at: '17:00:00'
  condition: []
  action:
  - service: utility_meter.next_tariff
    data:
      entity_id: utility_meter.daily_energy
  - service: utility_meter.next_tariff
    data:
      entity_id: utility_meter.monthly_energy
  mode: single
  #Automation of Outside lights day/night operation
- id: '1642464780223'
  alias: on off Outiside Lights
  description: ''
  trigger:
  - platform: time
    at: '18:00:00'
  - platform: time
    at: 06:30
  condition: []
  action:
  - choose:
    - conditions:
      - condition: time
        after: 06:30
        before: '17:35'
      sequence:
      - service: light.turn_off
        target:
          entity_id: light.outside_light
        data: {}
      - service: notify.mobile_app_poco_x3
        data:
          message: are OFF
          title: Outiside Lights
      - service: notify.mobile_app_redmi_note_4
```

```
        data:
          message: are OFF
          title: Outiside Lights
    - conditions:
      - condition: time
        after: '17:34:00'
        before: 07:00
      sequence:
      - service: light.turn_on
        target:
          entity_id: light.outside_light
        data: {}
      - service: notify.mobile_app_poco_x3
        data:
          message: are ON
          title: Outside Lights
      - service: notify.mobile_app_redmi_note_4
        data:
          message: are ON
          title: Outside Lights
    default: []
  mode: restart
  #Automation for Bedroom Heater
- id: '1644475290509'
  alias: Bedroom heater
  description: Auto on/off
  trigger:
  - platform: time
    at: 00:00
  - platform: time
    at: '10:00'
  condition:
  - condition: and
    conditions:
    - condition: device
      device_id: be4bc5eb5f5947af0ff3215d4ec35491
      domain: device_tracker
      entity_id: device_tracker.redmi_note_4
      type: is_home
  action:
  - choose:
    - conditions:
      - condition: time
        after: 00:00
        before: '10:00'
      sequence:
      - service: climate.set_temperature
        data:
```

```
          temperature: 24
        target:
          entity_id: climate.bedroom_heater
      - service: climate.set_hvac_mode
        data:
          hvac_mode: heat
        target:
          entity_id: climate.bedroom_heater
    - conditions:
      - condition: time
        after: '10:01'
        before: '23:59'
      sequence:
      - type: turn_off
        device_id: 7ed93e64dafca9abf19bac8f9caafba9
        entity_id: switch.shelly_shsw_pm_d8bfc019b746
        domain: switch
    - conditions:
      - condition: not
        conditions:
        - condition: zone
          entity_id: device_tracker.redmi_note_4
          zone: zone.home
      sequence:
      - type: turn_off
        device_id: 7ed93e64dafca9abf19bac8f9caafba9
        entity_id: switch.shelly_shsw_pm_d8bfc019b746
        domain: switch
    default: []
  mode: restart
  #Automation for Office heater
- id: '1645846563942'
  alias: Office heater (thermostat mode on)
  description: ''
  trigger:
  - platform: device
    type: turned_on
    device_id: 08516b364b6bc743a060ba9bc59570bc
    entity_id: switch.shelly_shsw_pm_98cdac1eddb0
    domain: switch
  condition: []
  action:
  - service: climate.set_temperature
    data:
      temperature: 24
    target:
      entity_id: climate.office_heater
  - service: climate.set_hvac_mode
```

```
    data:
      hvac_mode: heat
    target:
      entity_id: climate.office_heater
mode: restart
```

Παράρτημα Ε. Home assistant secrets.yaml

```
# Hiding my passwords from main configuration file
ewelink_pass: [redacted]
influx_pass: [redacted]
mqtt_pass: [redacted]
```

Παράρτημα Η. Home assistant- groups.yaml

```
groups.yaml
#creating a group of devices to operate with a single switch
 alldevices:
    name: All devices operation
    entities:
    - switch.shelly_shsw_pm_98cdac1f4e69
    - climate.livingroom_heater
    - climate.diningroom_heater
    - climate.office_heater
    - climate.bedroom_heater
```