ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΗΛΕΚΤΡΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΩΝ ΔΙΑΤΑΞΕΩΝ & ΣΥΣΤΗΜΑΤΩΝ ΑΠΟΦΑΣΕΩΝ

ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΓΙΑ ΠΑΡΑΚΟΛΟΥΘΗΣΗ ΣΥΜΠΕΡΙΦΟΡΑΣ ΚΑΤΑΝΑΛΩΣΗΣ ΕΝΕΡΓΕΙΑΣ ΟΙΚΙΑΚΩΝ ΧΡΗΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δήμητρα Καρούτσου

ΕΠΙΒΛΕΠΩΝ : **Μαρινάκης Ευάγγελος**

Επίκουρος Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2024

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΗΛΕΚΤΡΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΩΝ ΔΙΑΤΑΞΕΩΝ & ΣΥΣΤΗΜΑΤΩΝ ΑΠΟΦΑΣΕΩΝ

ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΓΙΑ ΠΑΡΑΚΟΛΟΥΘΗΣΗ ΣΥΜΠΕΡΙΦΟΡΑΣ ΚΑΤΑΝΑΛΩΣΗΣ ΕΝΕΡΓΕΙΑΣ ΟΙΚΙΑΚΩΝ ΧΡΗΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δήμητρα Καρούτσου

ΕΠΙΒΛΕΠΩΝ : **Μαρινάκης Ευάγγελος**

Επίκουρος Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 15η Ιουλίου 2024.

.....	.....	.....
Βαγγέλης Μαρινάκης	Ψαρράς Ιωάννης	Ασκούνης Δημήτριος
Επίκουρος Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2024

.....

Καρούτσου Δήμητρα

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π

ΠΕΡΙΛΗΨΗ

Η διαχείριση των δεδομένων ενεργειακής κατανάλωσης είναι κρίσιμη για την αποτελεσματική παρακολούθηση και βελτιστοποίηση της χρήσης ενέργειας σε οικιακό επίπεδο. Μέσω της συλλογής και της ανάλυσης αυτών των δεδομένων, είναι δυνατόν να προσδιοριστούν τα μοτίβα κατανάλωσης και να εντοπιστούν στρατηγικές για την εξοικονόμηση της ενέργειας.

Η αποτελεσματική διαχείριση της ενέργειας μπορεί να συμβάλει στη μείωση δαπανών διαβίωσης, στη βελτιστοποίηση της ποιότητας ζωής και στην περιβαλλοντική βιωσιμότητα, ενέργειες οι οποίες θα μπορούσαν να χαρακτηριστούν ως αναγκαίες ,αν αναλογιστούμε τις επιπτώσεις που έχει η εκτεταμένη χρήση της ηλεκτρικής ενέργειας τόσο στην κλιματική αλλαγή του πλανήτη όσο και στη ζωή των καταναλωτών.

Η τεχνολογική εξέλιξη στον τομέα των έξυπνων σπιτιών και συσκευών έχει ως αποτέλεσμα τη βαθμιαία αύξηση της παραγωγής δεδομένων που σχετίζονται με την ενεργειακή κατανάλωση και τις συσκευές στα νοικοκυριά. Αυτή η πληθώρα δεδομένων παρέχει σημαντικές ευκαιρίες για την ανάλυση, τον σχεδιασμό και τη βελτιστοποίηση της διαχείρισης ενέργειας.

Η παρούσα διπλωματική εργασία επικεντρώνεται στο σχεδιασμό ενός συστήματος διαχείρισης βάσεων δεδομένων η οποία θα συγκεντρώνει, θα αποθηκεύει και θα επεξεργάζεται με αποδοτικό τρόπο δεδομένα που σχετίζονται με το Διαδίκτυο των Πραγμάτων.

Συγκεκριμένα, προτείνεται ένα σύστημα το οποίο περιλαμβάνει μια PostgreSQL βάση δεδομένων, στην οποία αποθηκεύονται πληροφορίες και χαρακτηριστικά που αφορούν τις διάφορες οντότητες που εμπλέκονται στη δομή ενός έξυπνου σπιτιού, όπως οι ιδιοκτήτες – κάτοικοι των σπιτιών, τα ίδια τα σπίτια, καθώς και οι έξυπνες συσκευές του δικτύου. Επιπλέον, το σύστημα διαθέτει μια βάση δεδομένων χρονοσειρών Timescale στην οποία συλλέγονται και αποθηκεύονται οι μετρήσεις των συσκευών ,ενώ ταυτόχρονα επικοινωνεί με την PostgreSQL βάση. Οι κάτοικοι των σπιτιών ή οι ιδιοκτήτες έχουν πρόσβαση στα δεδομένα μέσω μιας εφαρμογής διεπαφής χρήστη, η οποία εξασφαλίζει την ιδιωτικότητα των δεδομένων και την ανάκτησή τους μόνο από εξουσιοδοτημένους χρήστες.

Λέξεις Κλειδιά : έξυπνα σπίτια, Διαδίκτυο των Πραγμάτων, διαχείριση δεδομένων ενεργειακής κατανάλωσης , PostgreSQL βάση δεδομένων, βάση δεδομένων χρονοσειρών

ABSTRACT

Managing energy consumption data is critical for effectively monitoring and optimizing energy use at the household level. By collecting and analyzing this data, it is possible to identify consumption patterns and develop strategies for energy savings. Effective energy management can contribute to reducing living expenses, optimizing quality of life, and promoting environmental sustainability, actions which could be deemed necessary considering the impacts of extensive electricity usage on both climate change and consumer life.

Technological advancements in the field of smart homes and devices have led to a gradual increase in the production of data related to energy consumption and household appliances. This abundance of data provides significant opportunities for the analysis, planning, and optimization of energy management.

This thesis focuses on the design of a database management system that efficiently gathers, stores, and processes data related to the Internet of Things. Specifically, it proposes a system that includes a PostgreSQL database to store information and characteristics related to various entities involved in the structure of a smart home, such as homeowners and residents, the homes themselves, and the smart devices within the network. Additionally, the system features a time-series database that collects and stores device measurements while simultaneously communicating with the PostgreSQL database. Homeowners or residents can access the data through a user interface application, which ensures data privacy and retrieval only by authorized users.

Keywords: smart homes, Internet of Things, energy consumption data management, PostgreSQL database, time-series database

Στην οικογένειά μου

ΕΥΧΑΡΙΣΤΙΕΣ

Με μεγάλη χαρά και συγκίνηση φτάνω στο τέλος της ακαδημαϊκής μου διαδρομής στη σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου , η οποία ολοκληρώνεται με την εκπόνηση αυτής της διπλωματικής εργασίας. Θα ήθελα , λοιπόν, να εκφράσω τις θερμές μου ευχαριστίες σε όσους συνέβαλαν σε αυτήν την επιτυχία.

Αρχικά, θα ήθελα να ευχαριστήσω τον κύριο Ευάγγελο Μαρινάκη για την ευκαιρία που μου έδωσε να εκπονήσω τη διπλωματική μου εργασία στο Εργαστήριο Συστημάτων Αποφάσεων και Διοίκησης. Ένα μεγάλο ευχαριστώ στον Νικόλα , στον Φίλιππο και στον Νίκο για την άψογη καθοδήγηση , την άμεση συνεργασία τους και την εμπιστοσύνη που μου έδειξαν .

Επίσης, θα ήθελα να εκφράσω την ευγνωμοσύνη μου στην οικογένειά μου και στους αγαπημένους μου ανθρώπους . Ευχαριστώ από καρδιάς τους γονείς μου, Νίκο και Γιάννα, για την υποστήριξη και την αγάπη τους σε κάθε μου βήμα. Σε αυτούς οφείλω τον άνθρωπο που είμαι σήμερα! Την αδερφή μου, Μαρία, που βρίσκεται πάντα στο πλευρό μου. Τον Ηλία, που πιστεύει πάντα σε εμένα και με ενθαρρύνει να προσπαθώ για το καλύτερο. Σας αγαπώ και σας ευχαριστώ!

Τέλος, ευχαριστώ τους συμφοιτητές μου , το παρεάκι μου, τον Αλέξανδρο, τον Βαγγέλη, τον Βασίλη, τον Θοδωρή και τον Λευτέρη, για τη συνεργασία μας , που έκανε αυτή τη διαδρομή ακόμα πιο πολύτιμη και ευχάριστη , και για τις αξέχαστες στιγμές που μοιραστήκαμε εντός και εκτός σχολής. Παιδιά, έχετε σφραγίσει τα φοιτητικά μου χρόνια!

Χωρίς τη συμβολή αυτών των υπέροχων ανθρώπων, το ταξίδι αυτό δεν θα ήταν το ίδιο μαγικό και δημιουργικό.

Σας ευχαριστώ όλους !

Αθήνα, Ιούλιος 2024

Δήμητρα Καρούτσου

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

ΚΕΦΑΛΑΙΟ 1

1.ΕΙΣΑΓΩΓΗ

1.1 ΥΠΟΒΑΘΡΟ ΚΑΙ ΚΙΝΗΤΡΟ

1.1.1 ΕΠΙΣΚΟΠΗΣΗ ΤΩΝ ΤΕΧΝΟΛΟΓΙΩΝ «ΔΙΑΔΙΚΤΥΟ ΤΩΝ ΠΡΑΓΜΑΤΩΝ (INTERNET OF THINGS-IOT)» ΚΑΙ «ΕΞΥΠΝΑ ΣΠΙΤΙΑ (SMART HOMES)»

1.1.1.1. ΟΡΙΣΜΟΣ ΕΞΥΠΝΩΝ ΣΠΙΤΙΩΝ

Με τον όρο «έξυπνο σπίτι» αναφερόμαστε σε μία κατοικία εξοπλισμένη με υπολογιστική και πληροφοριακή τεχνολογία , η οποία χρησιμοποιεί συσκευές συνδεδεμένες στο διαδίκτυο, προκειμένου να επιτύχει τον απομακρυσμένο έλεγχο και τη διαχείρισή τους [1] . Ο έλεγχος των συσκευών πραγματοποιείται συνήθως με τη βοήθεια κατάλληλης εφαρμογής τόσο μέσω smartphones όσο και μέσω κάποιων άλλων συσκευών δικτύου. Τα καινοτόμα έξυπνα σπίτια παρέχουν άνεση, οικονομία και τη δημιουργία ιδανικών συνθηκών διαβίωσης στους διαμένοντες , ενώ ταυτόχρονα τους παρέχουν τη δυνατότητα να ενημερώνονται για την ενεργειακή κατανάλωσή τους [1] .

1.1.1.2. ΠΩΣ ΛΕΙΤΟΥΡΓΟΥΝ

Ένα έξυπνο σπίτι δεν είναι απλώς μια συλλογή από απομονωμένες έξυπνες συσκευές που λειτουργούν η κάθε μία ξεχωριστά, αλλά αντίθετα αυτές συνεργάζονται μεταξύ τους , συλλέγοντας και ανταλλάσσοντας δεδομένα, προκειμένου να επιτευχθεί ,τελικά, ένα απομακρυσμένα ελεγχόμενο δίκτυο [2].

Οι συσκευές ενός έξυπνου σπιτιού συνδέονται σε ένα ενιαίο δίκτυο το οποίο τους επιτρέπει να επικοινωνούν τόσο μεταξύ τους όσο και με ένα κεντρικό σημείο (πχ smartphone, tablet,laptop). Η σύνδεση αυτή είναι δυνατόν να πραγματοποιηθεί μέσω Wi-Fi, Bluetooth ή με άλλα πρωτόκολλα επικοινωνίας, όπως το Zigbee. Το σύστημα εγκαθίσταται στη συσκευή δικτύου που έχει επιλέξει ο χρήστης, επιτρέποντας του να ανακτήσει πληροφορίες και μετρήσεις που χρειάζεται, να ρυθμίσει ορισμένες τιμές, όπως πχ τη θερμοκρασία του σπιτιού, καθώς και να δημιουργήσει χρονοπρογράμματα [3] .

Τα σύγχρονα έξυπνα σπίτια είναι συνήθως εξοπλισμένα με μεγάλο αριθμό δικτυωμένων αισθητήρων και επεξεργαστών που συνεργάζονται μεταξύ τους, αλλά και με τα υπόλοιπα μέρη του συστήματος, συλλέγοντας δεδομένα και εξάγοντας συμπεράσματα σχετικά με την κατάσταση του σπιτιού, καθώς και τις δραστηριότητες και τις συμπεριφορές των κατοίκων του [4]. Για παράδειγμα, ένα έξυπνο σπίτι έχει τη δυνατότητα να ενημερώσει τους κατοίκους του σε περίπτωση κάποιας διαρροής ή ανίχνευσης ύποπτης δραστηριότητας , όπως κάποια διάρρηξη , προκειμένου να αντιμετωπιστούν εγκαίρως πιθανές ζημιές. Επίσης, έχουν, τη δυνατότητα να εξάγουν στατιστικά των χρηστών σχετικά με την ενεργειακή τους κατανάλωση ανά τακτά χρονικά διαστήματα , προκειμένου να τους βοηθήσουν στην εξοικονόμηση δαπανών στους λογαριασμούς τους, αλλά και να τους ενημερώσουν σε περίπτωση υπερκατανάλωσης.

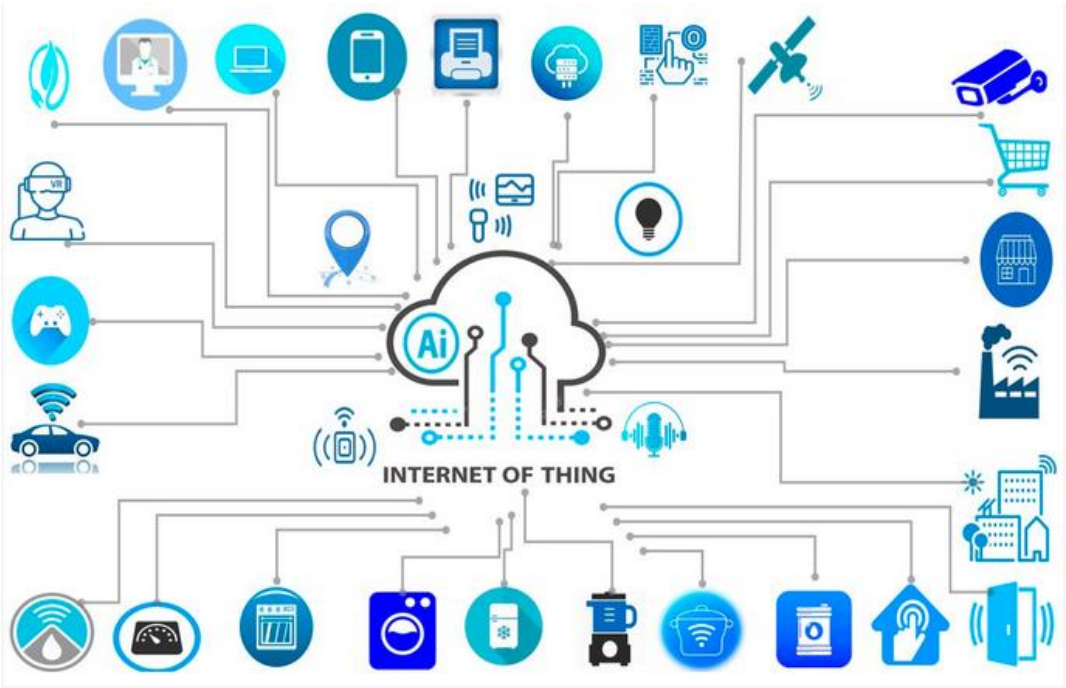
Ένα ακόμη χαρακτηριστικό του έξυπνου σπιτιού, είναι η ικανότητά του να παρακολουθεί τις καθημερινές δραστηριότητες και την ασφάλεια των κατοίκων και να ανιχνεύει αλλαγές στις καθημερινές τους ρουτίνες [4] . Για παράδειγμα, οι έξυπνοι θερμοστάτες «απομνημονεύουν» τις προτιμήσεις των χρηστών σχετικά με τη θέρμανση και την ψύξη του χώρου και προσαρμόζουν ανάλογα τη θερμοκρασία των εσωτερικών χώρων.

Αυτά αποτελούν μερικά μόνο παραδείγματα συστημάτων αυτοματισμού που διαθέτουν τα έξυπνα σπίτια. Παρακάτω παρατίθενται αναλυτικά τα πλεονεκτήματά τους σε κάθε τομέα της ζωής των χρηστών.

1.1.1.3. ΟΡΙΣΜΟΣ IOT

Το σύστημα το οποίο αναφέρθηκε παραπάνω και στο οποίο στηρίζεται η λειτουργία ενός έξυπνου σπιτιού ονομάζεται Διαδίκτυο των Πραγμάτων (Internet of Things – IoT). Ένα τέτοιο σύστημα αποτελείται από ένα διασυνδεδεμένο δίκτυο συσκευών, αισθητήρων , αντικειμένων και λογισμικών που συλλέγουν δεδομένα από το περιβάλλον τους, επικοινωνούν απροβλημάτιστα μεταξύ τους και ανταλλάσσουν δεδομένα μέσω του Διαδικτύου και του Cloud [5] . Όταν τα δεδομένα φτάσουν στο Cloud, το εκάστοτε λογισμικό τα επεξεργάζεται και αποφασίζει τι ενέργεια πρέπει να κάνει, όπως το να στείλει μια ειδοποίηση ή να αλλάξει κάποια ρύθμιση χωρίς την παρέμβαση του χρήστη [6] .

Το IoT αποτελεί μια προχωρημένη τάση που συνδέει άτομα με συσκευές ,αλλά και συσκευές μεταξύ τους , οποτεδήποτε και οπουδήποτε. Περιλαμβάνει μια πληθώρα διασυνδεδεμένων συσκευών, από καθημερινά αντικείμενα όπως τηλεοράσεις και ψυγεία μέχρι προηγμένους αισθητήρες, όπως απεικονίζεται χαρακτηριστικά στο παρακάτω σχήμα [5] .



Εικόνα 1.1 : Το IoT συνδέεται με διάφορες συσκευές [5]

Ένα τυπικό σύστημα IoT αποτελείται από τα παρακάτω 3 στοιχεία [6] :

1. **Έξυπνες Συσκευές :** Δηλαδή συσκευές που τους έχει δοθεί μια υπολογιστική ικανότητα, έχοντας έτσι τη δυνατότητα να συλλέγουν δεδομένα από το περιβάλλον τους και να σχηματίζουν μοτίβα δεδομένων, με σκοπό να τα ανταλλάξουν ή να τα επεξεργαστούν μέσω του Internet
2. **Εφαρμογές IoT :** Είναι μια συλλογή από συσκευές και λογισμικά που λαμβάνουν και στέλνουν δεδομένα. Χρησιμοποιούν μηχανική μάθηση και τεχνητή νοημοσύνη, ώστε να αναλύσουν κατάλληλα τα δεδομένα αυτά και να λάβουν τις σωστές αποφάσεις, προκειμένου να ανταποκριθούν έξυπνα στα δεδομένα που δέχονται.

3. **User Interface :** Η επικοινωνία και η ανταλλαγή πληροφοριών μεταξύ των συσκευών και των χρηστών γίνεται μέσω μιας διεπαφής χρήστη (User Interface - UI) χρησιμοποιώντας το πρωτόκολλο Διαδικτύου (IP).

Ο στόχος πίσω από το Διαδίκτυο των πραγμάτων είναι να έχουμε συσκευές που αναφέρουν αυτόματα σε πραγματικό χρόνο, βελτιώνοντας την αποδοτικότητα και φέρνοντας σημαντικές πληροφορίες στην επιφάνεια πιο γρήγορα από ένα σύστημα που εξαρτάται από ανθρώπινη παρέμβαση [7] . Η δυνατότητα που προσφέρει για απομακρυσμένη πρόσβαση και έλεγχο, έχουν οδηγήσει στην ευρεία εφαρμογή του σε διάφορους τομείς, συμπεριλαμβανομένου στρατιωτικές επιχειρήσεις, προόδους στις λύσεις έξυπνης υγείας, έξυπνα σπίτια, πόλεις του μέλλοντος, ακριβή γεωργία και περιβαλλοντική παρακολούθηση [5] .

Ωστόσο, η πλήρης αξιοποίηση του IoT εξαρτάται από την αποτελεσματική διαχείριση των τεράστιων όγκων δεδομένων που παράγονται από αυτό, τη διασφάλιση της διαλειτουργικότητας μεταξύ διαφόρων συσκευών και πλατφορμών, την εξαγωγή πολύτιμων πληροφοριών μέσω ανάλυσης δεδομένων και την προστασία της ιδιωτικότητας και της ασφάλειας των δεδομένων [5] .

Τα βασικά χαρακτηριστικά του δικτύου περιλαμβάνουν [5] :

1. **Συνδεσιμότητα:** Επιτρέπει στις συσκευές να επικοινωνούν τόσο μεταξύ τους όσο και με κεντρικά συστήματα. Αυτή η διασύνδεση αποτελεί τη βάση για την ανταλλαγή δεδομένων σε πραγματικό χρόνο και την έξυπνη λήψη αποφάσεων [8,9] .

2. **Ανίχνευση και Αντίληψη:** Οι συσκευές του IoT είναι εξοπλισμένες με αισθητήρες που ανιχνεύουν και αντιλαμβάνονται το περιβάλλον. Αυτοί οι αισθητήρες μπορούν να ανιχνεύουν διάφορες παραμέτρους, όπως θερμοκρασία, υγρασία, κίνηση και άλλες, επιτρέποντας στα συστήματα να συλλέγουν πολύτιμα δεδομένα [9,10].

3. **Ανάλυση Δεδομένων και Νοημοσύνη:** Τα δεδομένα που συλλέγονται από τις συσκευές απαιτούν προηγμένες τεχνικές ανάλυσης δεδομένων και τεχνητής νοημοσύνης. Το IoT αξιοποιεί αυτήν τη νοημοσύνη για να ανιχνεύσει πρότυπα και να βελτιστοποιήσει διαδικασίες, διευκολύνοντας έτσι τη λήψη αποφάσεων [11,12].

4. **Αυτοματισμός και Έλεγχος:** Επιτρέπει τον αυτοματισμό δίνοντας τη δυνατότητα στις συσκευές να εκτελούν προκαθορισμένες εργασίες χωρίς ανθρώπινη παρέμβαση. Αυτός ο αυτοματισμός προάγει την αποτελεσματικότητα, μειώνει τα ανθρώπινα λάθη και βελτιώνει την παραγωγικότητα [10,12].

5. **Κλιμακωσιμότητα και Ευελιξία:** Η ευελιξία του IoT επιτρέπει την άνετη επέκταση και ένταξη νέων συσκευών και τεχνολογιών. Η κλιμακωσιμότητα των συστημάτων διασφαλίζει ότι μπορούν να προσαρμοστούν και να φιλοξενήσουν διάφορες χρήσεις [13].

6. **Διαλειτουργικότητα:** Για την επιτυχή λειτουργία του IoT, η διαλειτουργικότητα μεταξύ διαφορετικών συσκευών και πλατφορμών είναι ζωτικής σημασίας. Οι προδιαγραφές και τα πρωτόκολλα του IoT διευκολύνουν την ομαλή επικοινωνία και συνεργασία μεταξύ διαφορετικών ανομοιόμορφων συστημάτων [14,15,16].

7. **Ασφάλεια και Απόρρητο:** Με την έκταση της ανταλλαγής δεδομένων, η εγγύηση της τήρησης των μέτρων ασφαλείας και απορρήτου είναι ζωτικής σημασίας. Τα συστήματα IoT πρέπει να εφαρμόζουν κρυπτογράφηση, πιστοποίηση και άλλους μηχανισμούς για να προστατεύσουν τα ευαίσθητα δεδομένα και το απόρρητο των χρηστών [17,16].

8. **Απόκριση σε Πραγματικό Χρόνο:** Επιτρέπει άμεσες ενέργειες και αντιδράσεις. Τα συστήματα IoT μπορούν να αντιδρούν σε μεταβαλλόμενες συνθήκες, καθιστώντας τα ιδανικά για εφαρμογές που απαιτούν γρήγορη λήψη αποφάσεων και χρόνους ανταπόκρισης [18].

9. **Αποδοτικότητα Ενέργειας:** Οι συσκευές σχεδιάζονται με γνώμονα την ενεργειακή αποδοτικότητα, εξασφαλίζοντας μακροχρόνια διάρκεια μπαταρίας και μειωμένη κατανάλωση ενέργειας. Αυτό το χαρακτηριστικό είναι ιδιαίτερα ζωτικό για εφαρμογές IoT που βασίζονται σε συσκευές που τροφοδοτούνται από μπαταρίες, αλλά και για την εξοικονόμηση της ενέργειας των χρηστών [19].

10. **Καθολική Πρόσβαση:** Το IoT ξεπερνά τις παραδοσιακές συσκευές υπολογιστών και προσφέρει καθολική πρόσβαση σε δεδομένα και υπηρεσίες. Οι χρήστες μπορούν να αλληλεπιδρούν με τα συστήματα μέσω smartphone, φορετών συσκευών και άλλων συνδεδεμένων συσκευών από οπουδήποτε και οποτεδήποτε [18].

1.1.1.4. ΒΑΣΙΚΗ ΕΦΑΡΜΟΓΗ ΚΑΙ ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΤΟΥ ΙΟΤ ΣΤΑ ΈΞΥΠΝΑ ΣΠΙΤΙΑ

1. **Αυτοματοποίηση Εργασιών:** Οι διάφορες συσκευές μπορούν να ελέγχονται απομακρυσμένα ή να ρυθμιστούν να λειτουργούν με βάση κάποιο πρόγραμμα [5].Για παράδειγμα, οι κάτοικοι του σπιτιού έχουν τη δυνατότητα να σβήνουν το θερμοσίφωνο ακόμα και αφού έχουν φύγει από το σπίτι τους, σε περίπτωση που το έχουν ξεχάσει, ενώ μπορούν και να φτιάξουν προγράμματα με την ενεργοποίηση των φώτων για συγκεκριμένες ώρες κάθε μέρα. Ορισμένες συσκευές επιτρέπουν και τις φωνητικές εντολές για τον έλεγχό τους, αυξάνοντας την αποδοτικότητά τους [5], ενώ υπάρχουν και φωνητικοί βοηθοί ,όπως η Google Assistant, η Alexa και η Siri που μπορούν να βοηθήσουν στον έλεγχο των συσκευών, να απαντούν σε ερωτήσεις και να ορίζουν υπενθυμίσεις με φωνητικές εντολές [20].

2. **Ασφάλεια Σπιτιού:** Ανιχνευτές κίνησης , βίντεο κουδούνια, έξυπνες κλειδαριές και κάμερες παρέχουν παρακολούθηση του σπιτιού σε πραγματικό χρόνο, επιτρέποντας στους ιδιοκτήτες να ελέγχουν την περιουσία τους, όποτε το επιθυμούν, και να λαμβάνουν ειδοποιήσεις για πιθανές παραβιάσεις της ασφάλειάς τους [5] , ενημερώνοντάς τους για ανεπιθύμητους επισκέπτες και πιθανούς κινδύνους [20].

3. **Ενεργειακή Απόδοση :** Πολλές συσκευές μπορούν να προγραμματιστούν να λειτουργούν με μεγαλύτερη αποδοτικότητα ή να μάθουν τις προτιμήσεις του χρήστη με τον χρόνο, μειώνοντας με αυτόν τον τρόπο την κατανάλωση ενέργειας και βελτιώνοντας τη χρήση της [5] .

4. **Συντήρηση Σπιτιού :** Οι αισθητήρες IoT μπορούν να παρακολουθούν την κατάσταση των συστημάτων του σπιτιού, όπως η θέρμανση και τα υδραυλικά, και να στέλνουν ειδοποιήσεις στους χρήστες όταν απαιτείται συντήρηση, αποτρέποντας δαπανηρές βλάβες [5] .

1.1.2 Η ΣΗΜΑΣΙΑ ΤΗΣ ΔΙΑΧΕΙΡΙΣΗΣ ΔΕΔΟΜΕΝΩΝ ΚΑΤΑΝΑΛΩΣΗΣ ΕΝΕΡΓΕΙΑΣ (ENERGY CONSUMPTION DATA)

Η μη αποτελεσματική διαχείριση ενέργειας σε παλαιά κτίρια, σε συνδυασμό με την ολοένα και αυξανόμενη κατασκευαστική δραστηριότητα , θα προκαλέσει την εκτίναξη της κατανάλωσης ενέργειας στο άμεσο μέλλον [21].

Μπροστά στον κίνδυνο της έλλειψης ενέργειας και της κλιματικής αλλαγής, που προκύπτουν από αυτή την υπερκατανάλωση, απαιτείται η εφαρμογή πιο έξυπνων στρατηγικών για την προσαρμογή

και μείωση της κατανάλωσης ενέργειας, καθώς και την εύρεση εναλλακτικών και βιώσιμων πηγών ενέργειας [21].

Η αποτελεσματική διαχείριση της ενεργειακής κατανάλωσης, τόσο στα έξυπνα σπίτια που μελετάμε σε αυτή τη διπλωματική εργασία, όσο και σε διάφορα άλλα περιβαλλοντικά πλαίσια , αποτελεί δημοφιλές ερευνητικό θέμα και επιτακτική ανάγκη για λόγους που αφορούν την παγκόσμια ζήτηση ηλεκτρικής ενέργειας , καθώς και την υπερθέρμανση του πλανήτη, αφού οι ανθρώπινες δραστηριότητες έχουν αρνητικό αντίκτυπο στο περιβάλλον. Η συνεχής παρακολούθηση της ενέργειας έχει ως στόχο την καλύτερη κατανόηση της κατανάλωσης, της παραγωγής και της μεταφοράς της [21] , χωρίς βέβαια να επηρεάζεται η άνεση των καταναλωτών.

Για το σκοπό αυτό, τα έξυπνα σπίτια ενσωματώνουν ψηφιακές συσκευές ανίχνευσης και επικοινωνίας, οι οποίες επιτρέπουν τη συνεχή παρακολούθηση της κατανάλωσης, τον έξυπνο έλεγχο των συσκευών και την επικοινωνία με τον πάροχο ενέργειας και το δίκτυο [22] .Τα έξυπνα σπίτια υποστηρίζουν την βέλτιστη διαχείριση των πόρων και των υποδομών του δικτύου και συμβάλλουν στην ενεργειακή αποδοτικότητα . Τα κτίρια ευθύνονται για περίπου το 40% της παγκόσμιας κατανάλωσης ενέργειας, ποσοστό που αναμένεται να αυξηθεί σημαντικά τα επόμενα χρόνια [22] . Συνεπώς, οι βελτιώσεις στην αποδοτικότητα, όταν εφαρμόζονται σε παγκόσμια κλίμακα, μπορούν να μειώσουν την κατανάλωση ενέργειας, την σπατάλη, το κόστος και τους περιβαλλοντικούς κινδύνους που σχετίζονται. Παράλληλα, η αποδοτικότερη διαχείριση της ενέργειας μειώνει και τη συνολική ζήτηση για ενέργεια και συνεπώς δεν θα χρειάζεται να γίνεται μελλοντικά μεγάλη επένδυση στην παραγωγή, τη μεταφορά και τη διανομή τις ενέργειας, καθώς οι ανάγκες θα είναι μειωμένες [22].

Από την πλευρά των καταναλωτών, η σωστή αξιοποίηση των δεδομένων ενέργειας που προέρχονται από τις συσκευές του σπιτιού βοηθά τους χρήστες να βελτιστοποιήσουν την κατανάλωση ενέργειας για να μειώσουν το κόστος. Η έξυπνη διαχείριση των δεδομένων ενεργειακής κατανάλωσης βοηθάει στην ακριβή μέτρηση της ποσότητας της ενέργειας που καταναλώνεται και καταγράφει τον ακριβή χρόνο που συμβαίνει αυτή η κατανάλωση, έτσι ώστε οι λογαριασμοί να μην είναι απλώς μια συνολική αξία κατανάλωσης ενέργειας , αλλά επιτρέποντας την πραγματική απεικόνιση της καταναλωμένης ενέργειας, να αποτελούν μια απεικόνιση της ενεργειακής συμπεριφοράς των χρηστών , την οποία η ίδιοι θα είναι σε θέση να κατανοήσουν. Αυτό έχει το πλεονέκτημα της μείωσης της ενεργειακής σπατάλης και ενθαρρύνει τους χρήστες να τροποποιήσουν τις μη αποδοτικές συνήθειες. Η διαχείριση αυτών των δεδομένων είναι σημαντική, καθώς στο μέλλον, θα βελτιώσει σημαντικά τη φάση λήψης αποφάσεων και θα επηρεάσει θετικά τη συμπεριφορά ενέργειας των τελικών χρηστών και διαχειριστών (ενεργειακή συνείδηση) [21] .

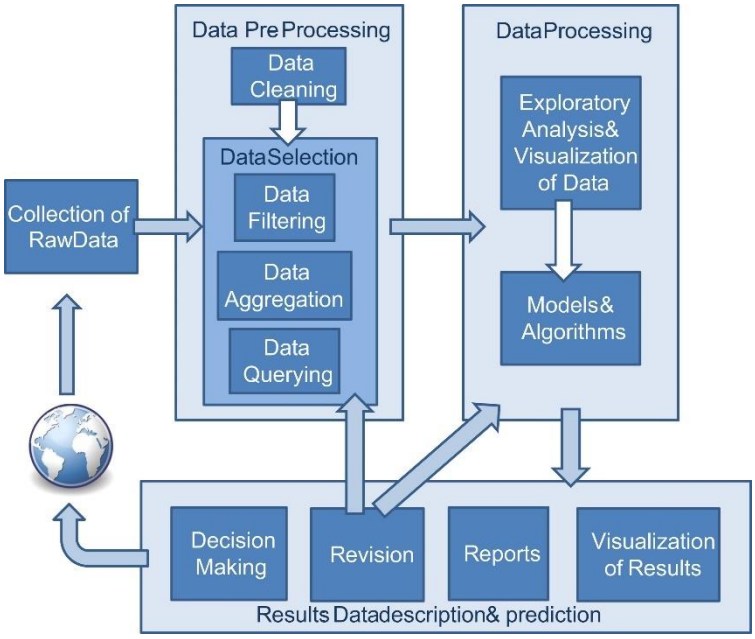
Σε αυτό το πλαίσιο, τα συστήματα διαχείρισης ενέργειας σπιτιού (HEMS) χρησιμοποιούνται κυρίως για την παρακολούθηση της κατανάλωσης σε πραγματικό χρόνο και για τον προγραμματισμό της λειτουργίας των συσκευών, ώστε να ελαχιστοποιηθεί ο λογαριασμός ενέργειας ή να ικανοποιηθούν άλλα συγκεκριμένα κριτήρια. Η προδιαγραφή των Συστημάτων Διαχείρισης Κατανάλωσης Ενέργειας στοχεύει στον εντοπισμό των απαραίτητων χαρακτηριστικών εξοικονόμησης ενέργειας που είναι τόσο οικονομικά αποδοτικά όσο και βολικά για τους οικιακούς χρήστες [22] . Οι κύριοι στόχοι του είναι να διασφαλίζεται ότι οι συσκευές χρησιμοποιούνται μόνο όταν είναι απαραίτητο με βελτιστοποίηση της χρήσης, αποθήκευσης και παραγωγής ενέργειας εύκολα και με περιορισμένο κόστος και μειωμένο περιβαλλοντικό αντίκτυπο [22]. Παράλληλα, χρειάζεται να προσφέρει μια εξαιρετική εμπειρία πελάτη και να προωθεί την διαλειτουργικότητα και την ασφάλεια.

Γίνεται κατανοητό λοιπόν, από τα παραπάνω, πως η ανάγκη σωστής διαχείρισης των δεδομένων κατανάλωσης ενέργειας είναι επιτακτική τόσο για περιβαλλοντικούς λόγους όσο και για το όφελος των ίδιων των καταναλωτών. Ο εντοπισμός των μοτίβων κατανάλωσης εξοπλισμού και χρηστών αναμφίβολα θα εξοικονομήσει χρήματα, θα βελτιώσει την άνεση και θα μειώσει τις εκπομπές ρύπων.

Για το σκοπό αυτό , αναπτύσσονται λύσεις βασισμένες στην Εξόρυξη Δεδομένων (Data Mining), η οποία προσανατολίζεται στην αυτόματη ανακάλυψη γνώσης από δεδομένα, και στην Επιστήμη Δεδομένων (Data Science), η οποία περιλαμβάνει ένα ευρύ φάσμα τεχνικών και πιο σύνθετα σύνολα δεδομένων [21].

Η Επιστήμη των Δεδομένων, ειδικότερα, διαδραματίζει ένα κρίσιμο ρόλο αναπτύσσοντας συστήματα και αλγόριθμους για την ανακάλυψη γνώσης, τον εντοπισμό προτύπων και την παραγωγή χρήσιμων προβλέψεων από σύνολα δεδομένων. Το αποτέλεσμα συχνά περιλαμβάνει την πρόβλεψη νέων τιμών και την οπτικοποίηση των αποτελεσμάτων. Συνδυάζει μαθηματική και στατιστική ανάλυση με προηγμένα εργαλεία για να εξάγει συμπεράσματα και πιθανές λύσεις στην αποδοτικότητα και τη διαχείριση ενέργειας [21] .

Όπως φαίνεται στο παρακάτω σχήμα [21] , η διαδικασία περιλαμβάνει τη συλλογή των ακατέργαστων δεδομένων (Collection of RawData) , τον καθαρισμό τους (Data Cleaning) και το φιλτράρισμα (Data Filtering) ή τη διατύπωση ερωτημάτων (Data Querying) προκειμένου να αφαιρεθούν οι ανεπιθύμητες πληροφορίες. Σε αυτό το στάδιο, μπορεί επίσης να ενσωματωθούν και να συγχωνευθούν επιπλέον πηγές πληροφοριών (Data Aggregation) . Μόλις τα δεδομένα είναι έτοιμα αποφασίζεται ποιες μέθοδοι ή αλγόριθμοι είναι πιο αποτελεσματικοί για την επεξεργασία των εκάστοτε δεδομένων και την απόκτηση της επιθυμητής γνώσης (Data Processing) . Τελικά, οδηγούμαστε σε ένα σύνολο αποτελεσμάτων που περιλαμβάνει οπτικοποιήσεις (Visualization), αναφορές (Reports) , καθώς και αναθεωρήσεις (Revision) στα προηγούμενα στάδια , τα οποία χρησιμεύουν στη λήψη της απόφασης ,την πρόβλεψη αποτελεσμάτων και τη βελτιστοποίηση της διαδικασίας με αναδρομές.



Εικόνα 1.2 : Διαδικασία Επιστήμης Δεδομένων [21]

1.2 ΔΙΑΤΥΠΩΣΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

1.2.1 ΠΡΟΚΛΗΣΕΙΣ ΣΤΗΝ ΟΡΓΑΝΩΣΗ ΚΑΙ ΣΤΗ ΔΙΑΧΕΙΡΙΣΗ ΔΕΔΟΜΕΝΩΝ ΑΠΟ ΣΥΣΚΕΥΕΣ ΕΞΥΠΝΩΝ ΣΠΙΤΙΩΝ

Τα έξυπνα σπίτια αποτελούν μία από τις κύριες τάσεις στην ανάπτυξη του Διαδικτύου των Πραγμάτων (IoT). Η διαχείριση τέτοιων συστημάτων, ωστόσο , παρά τα πληθώρα οφέλη που προσφέρει, δημιουργεί μια σειρά από προκλήσεις, τόσο στην οργάνωση όσο και στη διαχείριση

των δεδομένων που συλλέγονται από τις συσκευές. Παρακάτω παρουσιάζονται μερικές βασικές απαιτήσεις που είναι αναγκαίο να ληφθούν υπόψη κατά το σχεδιασμό τέτοιων συστημάτων, προκειμένου να υπάρξει το επιθυμητό αποτέλεσμα [5]:

1. **Ασφάλεια και ιδιωτικότητα:**Τα δεδομένα που δημιουργούν και ανταλλάσσουν μεταξύ τους οι συσκευές του IoT αποτελούν προσωπικές πληροφορίες, και έτσι η διασφάλιση της ιδιωτικότητας των χρηστών μέσα από τη συνεχή ροή δεδομένων αποτελεί επιτακτική ανάγκη. Η παρουσία των συσκευών παντού και η συμμετοχή τους σε πληθώρα δικτύων συνδεδεμένα στο Internet τις κάνει ευάλωτες σε παραβιάσεις που θέτουν σε κίνδυνο αυτά τα ευαίσθητα δεδομένα [23]. Η διασφάλιση ότι τα δεδομένα μεταδίδονται, αποθηκεύονται και επεξεργάζονται με κρυπτογραφημένο τρόπο και ανώνυμα παρουσιάζει μια πρόκληση. Για το σκοπό αυτό είναι ανάγκη να συσπειρωθούν οι πάροχοι των υπηρεσιών cloud και οι φορείς ασφαλείας, για ενίσχυση της παροχής συσκευών και την εποπτεία λειτουργίας, δημιουργώντας προστατευμένα συστήματα IoT από κακόβουλες επιθέσεις. Η απειλή των επιθέσεων θα συνεχίσει να υπάρχει , ωστόσο, η ασφάλεια των δεδομένων χρειάζεται να προσανατολιστεί στην αποτροπή απωλειών δεδομένων, τη διασφάλιση της εμπιστευτικότητας και την αποφυγή δημιουργίας εσφαλμένων δεδομένων [24].

2. **Εμπιστευτικότητα και Αυθεντικότητα:** Η διατήρηση της εμπιστευτικότητας σχετίζεται με την ασφάλεια και την ιδιωτικότητα, με την επιβολή αυστηρών ελέγχων στην πρόσβαση στα δεδομένα και την κρυπτογράφηση ευαίσθητων πληροφοριών με τη βοήθεια αλγορίθμων και κλειδιών. Η διαχείριση των κρυπτογραφικών κλειδιών εξασφαλίζει τη συμμόρφωση με καθιερωμένα πρότυπα και κανονισμούς. Η ανεπαρκής διαχείριση κλειδιών και η αποτυχία διατήρησης της εμπιστευτικότητας μπορούν να οδηγήσουν σε μη εξουσιοδοτημένη αποκάλυψη δεδομένων, υπονομεύοντας την εμπιστοσύνη και ανοίγοντας τον δρόμο για την κυβερνοκατασκοπεία, ενώ μπορεί να προκαλέσουν διαρροές δεδομένων και κλοπή πνευματικής ιδιοκτησίας . Η αυθεντικότητα από την άλλη, αφορά την επαλήθευση της ταυτότητας των χρηστών . Θεμελιώνει την εμπιστοσύνη και εξασφαλίζει ότι οι ανταλλαγές δεδομένων πραγματοποιούνται μόνο μεταξύ ατόμων που έχουν δικαίωμα. Η αποτυχία να διασφαλιστεί η αυθεντικότητα ανοίγει την πόρτα για πλαστοπροσωπία και μη εξουσιοδοτημένη πρόσβαση. Η απουσία αυθεντικότητας διευκολύνει τις επιθέσεις, θέτοντας σε κίνδυνο ευαίσθητα δεδομένα και κλονίζοντας την εμπιστοσύνη προς το σύστημα .

3. **Ακεραιότητα:** Η ακεραιότητα των δεδομένων αποτελεί θεμέλιο για την αξιόπιστη λήψης των αποφάσεων , για αυτό χρειάζεται να διασφαλίζεται ότι τα δεδομένα παραμένουν συνεχώς αναλλοίωτα και ενημερωμένα. Σε αντίθετη περίπτωση, μπορεί να προκληθεί παραπληροφόρηση που ενδέχεται να προκαλέσει δυσλειτουργίες συστήματος και λανθασμένες αποφάσεις [25], οδηγώντας σε ενέργειες που κλονίζουν την εμπιστοσύνη των ενδιαφερόμενων στο σύστημα.

4. **Διαθεσιμότητα:** Τα εξουσιοδοτημένα άτομα είναι αναγκαίο να έχουν συνεχώς απρόσκοπτη πρόσβαση στα δεδομένα για τη λήψη των αποφάσεων. Οι καθυστερήσεις ή οι διακοπές στη διαθεσιμότητα των δεδομένων μπορούν να παραλύσουν κρίσιμες διαδικασίες, να εμποδίσουν την ενημερωμένη λήψη αποφάσεων και να καθυστερήσουν την έγκαιρη εκτέλεση ενεργειών . Η μειωμένη διαθεσιμότητα μπορεί να διακόψει τις λειτουργίες, προκαλώντας οικονομικές απώλειες και δυσαρέσκεια των πελατών.

5. **Διαλειτουργικότητα :** Αφορά την προσαρμογή και τη συνεργασία μεταξύ συστημάτων. Αυτή η ικανότητα ενθαρρύνει τη δημιουργία δικτύων και τη μεταφορά δεδομένων μεταξύ προγραμμάτων [26,27]. Δεδομένης της πολυπλοκότητας των συστημάτων IoT με ετερογενείς συσκευές, η διαλειτουργικότητα παραμένει ένα κρίσιμο χαρακτηριστικό, περιλαμβάνοντας την επικοινωνία, την εκτέλεση προγραμμάτων και τη μεταφορά δεδομένων μεταξύ ανόμοιων μονάδων [28].

6. **Διαχείριση Δεδομένων:** Οι ροές πληροφορίας που παράγονται από το δίκτυο των συσκευών είναι ασταμάτητες. Αν παραμείνει ανεξέλεγκτη, αυτή η άνοδος των δεδομένων μπορεί να οδηγήσει σε μια σειρά προβλημάτων, συμπεριλαμβανομένης της λειτουργικής ανεπάρκειας και της καθυστέρησης των δικτύων που υπερφορτώνονται από τα δεδομένα. Αυτό πιθανώς να εμποδίσει την ομαλή λειτουργία εφαρμογών που στηρίζονται στην έγκαιρη ανάκτηση στιγμιαίων ενδείξεων δεδομένων. Για τις εφαρμογές σε πραγματικό χρόνο, από τα αυτόνομα οχήματα που λαμβάνουν αποφάσεις σε κλάσματα του δευτερολέπτου έως τα συστήματα απομακρυσμένης παρακολούθησης υγείας που ανιχνεύουν κρίσιμες ανωμαλίες, η επίτευξη βέλτιστης ανταποκριτικότητας είναι υψίστης σημασίας . Ωστόσο, τα ίδια τα δεδομένα που τροφοδοτούν αυτές τις εφαρμογές μπορούν να μετατραπούν σε εμπόδιο αν δεν αντιμετωπιστούν σωστά. Όταν τα δίκτυα πνίγονται από τον καταιγισμό των δεδομένων, η καθυστέρηση μεταξύ της δημιουργίας των δεδομένων και της λήψης και επεξεργασίας αυξάνεται, ενώ η ουσία της άμεσης ανάλυσης καταρρέει .

7. **Επεκτασιμότητα :** Καθώς ο αριθμός των διασυνδεδεμένων συσκευών αυξάνεται εκθετικά, τα δίκτυα που τα διαχειρίζονται πρέπει να σχεδιαστούν με προνοητικότητα που να προβλέπει αυτήν την ανάπτυξη. Είναι υψίστης σημασίας τα δίκτυα να είναι αρχιτεκτονικά σχεδιασμένα ώστε να μπορούν να προσαρμόζονται ομαλά στην προσθήκη νέων συσκευών, χωρίς να προκαλείται συμφόρηση. Αυτό επιτάσσει την προτίμηση ευέλικτων δομών που μπορούν να επεκταθούν ή να συρρικνωθούν ανάλογα με τις μεταβαλλόμενες απαιτήσεις των δεδομένων [29,30]. Η απαίτηση για επεκτασιμότητα δεν περιορίζεται μόνο στην αρχιτεκτονική των δικτύων, αλλά είναι αναγκαία και στην υποστήριξη της επεξεργασίας δεδομένων, όπως αναφέρθηκε και παραπάνω στη διαχείριση των δεδομένων. Η έκρηξη δεδομένων οδηγεί σε αυξήσεις της υπολογιστικής ζήτησης, η ικανοποίηση της οποίας απαιτεί μια υποδομή που να αντικατοπτρίζει την ευελιξία των ίδιων των δικτύων, όπως είναι για παράδειγμα το το cloud computing, με την ικανότητά του να παρέχει πόρους κατ' απαίτηση. Εξίσου σημαντικό είναι και το edge computing, το οποίο φέρνει την υπολογιστική επεξεργασία πιο κοντά στις πηγές δεδομένων, μειώνοντας την καθυστέρηση και αποφορτίζοντας την πίεση στους κεντρικούς κόμβους επεξεργασίας [31,32,33,34,35].

8. **Πολυπλοκότητα Δικτύου και Ευρυζωνικότητας :** Η συνεχής ροή δεδομένων αυξάνει τις απαιτήσεις του δικτύου και δημιουργεί προκλήσεις συμφόρησης . Ο τεράστιος όγκος δεδομένων που προέρχονται από τις συσκευές IoT, όλες διεκδικώντας περιορισμένους δικτυακούς πόρους, μπορεί να οδηγήσει σε σημεία συμφόρησης, προκαλώντας καθυστερήσεις στη μετάδοση δεδομένων και αυξημένη καθυστέρηση. Αυτό οδηγεί στην επιτακτική ανάγκη για εξερεύνηση τεχνολογιών που μπορούν να ανακουφίσουν την πίεση στα δίκτυα. Οι πρόοδοι στο 5G , λόγου χάρη, υπόσχονται υψηλότερες ταχύτητες μετάδοσης δεδομένων και μειωμένη καθυστέρηση, προσφέροντας πιθανώς την απαραίτητη εύρος ζώνης για να υποστηρίξουν την ροή δεδομένων του IoT.

9. **Κατανάλωση Ενέργειας:** Η αποτελεσματική διαχείριση των πηγών ενέργειας έχει ως όφελος επεκτεταμένες χρονικές περιόδους λειτουργίας και μειωμένα έξοδα. Για την αντιμετώπιση αυτής της πρόκλησης απαιτείται ένας συνδυασμός σχεδιασμού υλικού, βελτιστοποίησης λογισμικού και στρατηγικών διαχείρισης ενέργειας. Οι κατασκευαστές συσκευών πρέπει να δίνουν προτεραιότητα στην ανάπτυξη χαμηλής κατανάλωσης συστατικών και αισθητήρων, εκμεταλλευόμενοι τεχνολογίες που επιτρέπουν στις συσκευές να λειτουργούν αποτελεσματικά ακόμα και σε περιβάλλοντα με περιορισμένους πόρους [36,37]. Επιπλέον, πρέπει να εφαρμοστούν προηγμένοι αλγόριθμοι και τεχνικές διαχείρισης ενέργειας στο επίπεδο λογισμικού, επιτρέποντας στις συσκευές να προσαρμόζουν δυναμικά την κατανάλωση ενέργειας τους βάσει παραγόντων

περιβάλλοντος, απαιτήσεων χρήστη και προτύπων κίνησης δεδομένων [38]. Η ενσωμάτωση ανανεώσιμων πηγών ενέργειας, όπως η ηλιακή ή η κινητική ενέργεια, ενισχύει περαιτέρω τη δυνατότητα για βιώσιμες υλοποιήσεις IoT. Η επίλυση αυτού του προβλήματος συμβάλλει, μεταξύ άλλων, σε ένα πιο βιώσιμο και συνδεδεμένο μέλλον.

1.3 ΣΤΟΧΟΙ

Στόχος της παρούσας διπλωματικής εργασίας είναι ο σχεδιασμός και η βελτιστοποίηση ενός προτύπου συστήματος αναπαράστασης δεδομένων που προέρχονται από συσκευές έξυπνων σπιτιών για τη μέτρηση και την παρακολούθηση της συμπεριφοράς κατανάλωσης ενέργειας των χρηστών, προκειμένου να επιτευχθεί αποδοτικότερη ενεργειακή διαχείριση.

Οι λειτουργίες που θέλουμε να επιτελούνται είναι η οργάνωση των δεδομένων, δηλαδή το πώς θα αποθηκευτούν, καθώς και η διαχείρισή τους, δηλαδή το πώς θα αξιοποιηθούν και θα αναλυθούν έτσι ώστε να μπορούν να εξαχθούν ορισμένα συμπεράσματα.

Για το σκοπό αυτό χρειαζόμαστε ένα σύστημα που θα διαχειρίζεται χρονοσειρές δεδομένων , τα οποία αποτελούν τις μετρήσεις που προέρχονται από κάθε συσκευή και σχετίζονται με την ενεργειακή κατανάλωση των χρηστών. Η ανάλυση αυτών των δεδομένων έχει κρίσιμη σημασία , για λόγους βιωσιμότητας, μείωσης της ενεργειακής σπατάλης και οικονομίας, όπως αναφέρθηκε εκτενώς στην παραπάνω ενότητα 1.1.2.

Το σύστημά αυτό πρέπει να συνδέεται με μία βάση δεδομένων στην οποία αποθηκεύονται πληροφορίες και χαρακτηριστικά που αφορούν τις διάφορες οντότητες που εμπλέκονται στη δομή ενός έξυπνου σπιτιού. Κύριο επίκεντρο αποτελούν οι ιδιοκτήτες – κάτοικοι των σπιτιών, τα ίδια τα σπίτια, καθώς και οι έξυπνες συσκευές του δικτύου, οι οποίες υπακούν στη δομή του Διαδικτύου των Πραγμάτων και λειτουργούν όπως αναφέρθηκε σε προηγούμενη ενότητα.

Εν συνεχεία, οι χρήστες θα μπορούν να επωφεληθούν από μια εφαρμογή μέσω της οποίας, μετά την εγγραφή τους, θα έχουν πρόσβαση σε πληροφορίες σχετικά με τα συστήματά τους, με βασικό στόχο να ενημερώνονται για την ενεργειακή τους κατανάλωση ανά τακτά χρονικά διαστήματα που εκείνοι θα επιλέξουν (ημερήσια, εβδομαδιαία, μηνιαία). Θα μπορούν επίσης να ανακτούν πληροφορίες που σχετίζονται με τις οντότητες με τις οποίες έχουν συνδεθεί οι ίδιοι στο σύστημα, όπως για παράδειγμα με τα δωμάτια του σπιτιού τους, για τα οποία θα μπορούν να ζητήσουν να τους εμφανιστούν όλες οι πληροφορίες που είναι αποθηκευμένες, ή για τα χαρακτηριστικά κάποιας συγκεκριμένης συσκευής. Μπορούν, ακόμη , να ενημερώνονται για την ημερήσια, εβδομαδιαία ή μηνιαία κατανάλωσή τους ή και για πιθανά προβλήματα και κινδύνους, να βλέπουν στατιστικά του σπιτιού τους, να έχουν πρόσβαση στις προγραμματισμένες ενέργειες

Για να διασφαλίσουμε την ιδιωτικότητα των δεδομένων αυτών, χρειάζεται οι ζητούμενες πληροφορίες να επιστρέφονται μόνο στους χρήστες που είναι εξουσιοδοτημένοι να έχουν πρόσβαση σε αυτές. Για το σκοπό αυτό, κατά τη πραγματοποίηση οποιουδήποτε αιτήματος πάνω στη βάση μας χρειάζεται να μεσολαβεί ένα ενδιάμεσο σύστημα το οποίο θα λαμβάνει τα στοιχεία του χρήστη που πραγματοποιεί το αίτημα και θα επιβεβαιώνει ή θα απορρίπτει την πρόσβασή του στα αντίστοιχα δεδομένα που ζητάει. Το σύστημα αυτό είναι γνωστό ως API Gateway και λειτουργεί ως μεσολαβητής ανάμεσα στον χρήστη και στη βάση μας , διαχειρίζοντας και δρομολογώντας τα αιτήματα που αφορούν το backend μας, έτσι ώστε να εξασφαλιστεί η προστασία των δεδομένων και η μη διαρροή τους σε τρίτους.

1.4 ΕΥΡΟΣ ΤΗΣ ΔΙΑΤΡΙΒΗΣ

Στην παρούσα διπλωματική θα επικεντρωθούμε στο σχεδιασμό της βάσης δεδομένων στην οποία αποθηκεύονται οι διάφορες οντότητες του συστήματος μαζί με τα χαρακτηριστικά τους, ενώ θα αναφερθούμε και θα αναλύσουμε και τη βάση χρονοσειρών ,η οποία περιλαμβάνει μετρήσεις που προέρχονται από τις συσκευές μας, χωρίς ωστόσο να επεκταθούμε στο σχεδιασμό της.

Θα γίνει μια επισκόπηση των τεχνολογιών που χρησιμοποιήθηκαν για τη δημιουργία ενός τέτοιου συστήματος, καθώς και ο λόγος που επιλέχθηκε κάθε μία από αυτές.

Στη συνέχεια , θα παρουσιαστεί αναλυτικά το σχήμα της κύριας βάσης μας , με διαγράμματα και λεπτομερείς αναφορές για κάθε χαρακτηριστικό, καθώς και μέθοδοι που χρησιμοποιούνται για την ανάκτηση δεδομένων από τη βάση μέσω ερωτημάτων.

Τέλος, ένα ολόκληρο κεφάλαιο θα αφιερωθεί στο testing του συστήματος και σε συχνά use cases που πρέπει αυτό να ικανοποιεί.

Το σύστημα API Gateway για θέματα ιδιωτικότητας και αυθεντικοποίησης των χρηστών θα παρουσιαστεί σύντομα στην αντίστοιχη ενότητα, αλλά δεν θα γίνει αναλυτική περιγραφή των συστατικών του , καθώς δεν καλύπτονται στα πλαίσια αυτής της διπλωματικής .

ΚΕΦΑΛΑΙΟ 2

2.ΑΝΑΣΚΟΠΗΣΗ ΤΗΣ ΒΙΒΛΙΟΓΡΑΦΙΑΣ

2.1. ΔΙΑΔΙΚΤΥΟ ΤΩΝ ΠΡΑΓΜΑΤΩΝ (INTERNET OF THINGS-IOT) ΚΑΙ ΈΞΥΠΝΑ ΣΠΙΤΙΑ (SMART HOMES)

2.1.1. ΤΡΕΧΟΥΣΕΣ ΤΑΣΕΙΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ

Στον τομέα της διαχείρισης ενέργειας κτιρίων, η Επιστήμη Δεδομένων χρησιμοποιείται πλέον για την αντιμετώπιση προβλημάτων όπως η πρόβλεψη της ζήτησης ενέργειας για την προσαρμογή της παραγωγής και της διανομής, η ανάλυση των λειτουργιών, της κατάστασης και των βλαβών των κτιρίων για τη βελτιστοποίηση του κόστους λειτουργίας και συντήρησης και η ανίχνευση προτύπων κατανάλωσης ενέργειας για τη δημιουργία εξατομικευμένων εμπορικών προσφορών και την ανίχνευση απάτης. Αυτό απαιτεί τη συλλογή δεδομένων που αφορούν τη λειτουργία των κτιρίων και τη συμπεριφορά των χρηστών. Τα δεδομένα αυτά πρέπει επίσης να ερμηνεύονται για την εφαρμογή προσαρμοσμένων πολιτικών διαχείρισης ενέργειας. Οι πληροφορίες που συλλέγονται μπορεί να προέρχονται από πολύ ετερογενείς πηγές, όπως αισθητήρες που βρίσκονται στον εξοπλισμό και στο άμεσο περιβάλλον και εξωτερικές παραμέτρους , όπως λόγου χάρη ο καιρός και το κόστος ενέργειας.

Η Επιστήμη των Δεδομένων αξιοποιείται στις τρέχουσες τάσεις που εφαρμόζονται στα έξυπνα σπίτια και στο Διαδίκτυο των Πραγμάτων, με γνώμονα τι απαιτήσεις που παρουσιάστηκαν παραπάνω και την προτεραιότητα που δίνεται σε κάθε μία από αυτές. Αναλυτικότερα, οι τάσεις που επικρατούν στη διαχείριση αυτών των προκλήσεων και στη επίτευξη των ζητούμενων αποτελεσμάτων αφορούν τεχνολογίες οι οποίες αξιοποιούνται κατάλληλα με σκοπό να πετύχουν βέλτιστη διαχείριση ενέργειας και βιωσιμότητα. Με τον όρο «βιωσιμότητα» αναφερόμαστε στην ικανότητα κάλυψης των ενεργειακών αναγκών του παρόντος, χωρίς ωστόσο να θέτουμε σε κίνδυνο την κάλυψη των αναγκών του μέλλοντος. Η Επιστήμη των Δεδομένων μπορεί να συμβάλει σε αυτό με τον εντοπισμό προτύπων κατανάλωσης για τη βελτιστοποίηση της χρήσης ενέργειας, την πρόληψη βλαβών του εξοπλισμού για την αποφυγή σπατάλης, την αυτόματη προσαρμογή των συσκευών με βάση τις ανάγκες κ.ά. Η διαχείριση της ενέργειας ,επίσης, εκτός από τα έξυπνα σπίτια αφορά επίσης τόσο συστήματα ανανεώσιμων πηγών ενέργειας, όπως φωτοβολταϊκά, αιολικά πάρκα κλπ , όσο και ενεργειακές κοινότητες ,οι οποίες έχουν να κάνουν με την αυτοπαραγωγή ενέργειας των πολιτών από αυτές τις ανανεώσιμες πηγές .

Παρακάτω παρουσιάζονται οι τρέχουσες τάσεις και τεχνολογίες που εφαρμόζονται στα έξυπνα σπίτια μαζί με μερικά ενδεικτικά παραδείγματα για το καθένα [39]:

1. **Πρόβλεψη του ενεργειακού φορτίου και της ενεργειακής ζήτησης :** Η ενεργειακή ζήτηση, ή το ενεργειακό φορτίο, αναφέρεται στο ποσό ενέργειας που απαιτείται από ένα κτίριο κατά τη διάρκεια ενός συγκεκριμένου χρονικού διαστήματος, ή για ένα χρονικό σημείο, προκειμένου αυτό να εκτελεί πλήρως και αποτελεσματικά όλες τις απαραίτητες λειτουργίες του.

   Παραδείγματα τεχνολογιών που αξιοποιούνται για αυτό τον σκοπό είναι τα συστήματα HVAC, τα οποία επικεντρώνονται στα θερμικά φορτία και αναφέρονται στην ποσότητα θερμότητας και ψύξης που πρέπει να προστεθεί ή να αφαιρεθεί από το κτίριο για να διατηρηθεί η άνεση των ενοίκων του. Τα θερμικά φορτία μπορούν να ταξινομηθούν ως εσωτερικά φορτία, όταν η μεταφορά/επίδραση θερμότητας παράγεται από στοιχεία (π.χ. φωτισμός, εξοπλισμός, ή άνθρωποι) εντός του κτιρίου κατά τη λειτουργία του, ή ως εξωτερικά φορτία όταν η πηγή της επίδρασης οφείλεται σε εξωτερικούς (γενικά περιβαλλοντικούς) παράγοντες, όπως ο ήλιος, ο

αέρας και η υγρασία. Στο παράδειγμα του Kusiak και των συνεργατών του [40] η πρόβλεψη των θερμικών φορτίων (θέρμανσης και ψύξης) του κτιρίου γίνεται με τη βοήθεια νευρωνικών δικτύων, δημιουργώντας ένα μοντέλο για την επιλογή των παραμέτρων που απαιτούνται για μια μη γραμμική αντιστοίχιση των κλιματικών μετρήσεων.

Στο ίδιο πλαίσιο, μια διαφορετική εφαρμογή αποτελεί η ανάλυση των φορτίων ηλεκτρικής ενέργειας με μεθόδους ομαδοποίησης. Αυτό είναι σημαντικό, καθώς ,όταν τα επίπεδα παραγωγής και προσφοράς δεν είναι σε θέση να ικανοποιήσουν τη ζήτηση, μπορεί να προκληθούν διακοπές ρεύματος και μείωση φορτίου, τα οποία δυσαρεστούν τους πελάτες. Ο Prahastono και οι συνεργάτες του [41] παρουσίασαν μια επισκόπηση των κοινών μεθόδων ομαδοποίησης (π.χ. ιεραρχική, k-means, fuzzy k-means, follow the leader και fuzzy relation) και συνέκριναν την αποτελεσματικότητά τους στην ταξινόμηση των πελατών και τη δημιουργία προφίλ ηλεκτρικών φορτίων.

Ο Li [42] ανέπτυξε έναν αλγόριθμο δέντρου αποφάσεων για την ανάλυση της επίδρασης εξωτερικών παραγόντων κατά τις χρονικές περιόδους στις οποίες η παροχή ηλεκτρικής ενέργειας αναμένεται να φτάσει σε σημαντικά υψηλότερα επίπεδα από τον μέσο όρο. Στόχος αυτής της έρευνας ήταν να μάθει διαφορετικά μοντέλα για τα ενεργειακά φορτία κτιρίων σε διαφορετικά κλίματα μέσω δύο αλγορίθμων παλινδρόμησης και να χρησιμοποιήσει τις πληροφορίες από τα δέντρα αποφάσεων για να προβλέψει το μέγιστο αναμενόμενο φορτίο για την επόμενη ημέρα. Αντίθετα, ο Yang και οι συνεργάτες του [43] χρησιμοποίησαν τη μεθοδολογία ταξινόμησης C4.5 για να αναλύσουν έναν συνδυασμό εσωτερικών και εξωτερικών περιβαλλοντικών συνθηκών και να καθορίσουν την επίδραση των εξωτερικών συνθηκών στην εσωτερική άνεση των χρηστών του κτιρίου.

Είναι επομένως εξαιρετικά σημαντικό να αναπτυχθούν διαδικασίες που μπορούν να προβλέπουν την κορύφωση της ζήτησης για οποιαδήποτε δεδομένη ημέρα και να μοντελοποιούν το σύντομο και το μεσοπρόθεσμο φορτίο ενέργειας, τόσο θερμικής και ηλεκτρικής, όπως αναφέρθηκαν παραπάνω, όσο και άλλων μορφών ενέργειας.

2. **Βελτιστοποίηση της λειτουργίας των κτιρίων :** Τα δεδομένα που συλλέγονται από τα συστήματα λειτουργίας κτιρίων περιλαμβάνουν πληροφορίες σχετικά με τη θερμοκρασία, την υγρασία, τη ροή, την πίεση, την ισχύ, τα σήματα ελέγχου, την κατάσταση του εξοπλισμού κ.λπ. Αυτά, στη συνέχεια , μπορούν να αναλυθούν και να εκμεταλλευτούν για να εξαχθούν κανόνες λειτουργίας που υποστηρίζουν τη διαχείριση του κτιρίου και τη βελτίωση συγκεκριμένων τμημάτων ή μεταβλητών του. Οι τεχνικές ομαδοποίησης έχουν επίσης εφαρμοστεί εδώ, για να αναλύσουν τα σύνολα δεδομένων που παράγονται από τα συστήματα αυτοματισμού κτιρίων.

Τα μοντέλα ταξινόμησης είναι αποτελεσματικά εργαλεία που μπορούν να χρησιμοποιηθούν για να προβλέψουν την άνεση των χρηστών του κτιρίου υπό διαφορετικές περιβαλλοντικές συνθήκες [43]. Ο May-Ostendorp και οι συνεργάτες του [44] εφάρμοσαν διάφορες τεχνικές ταξινόμησης για να εξάγουν κανόνες από αποτελέσματα offline προγνωστικού ελέγχου μοντέλου και να επιτύχουν σχεδόν βέλτιστες στρατηγικές εποπτικού ελέγχου για ένα κτίριο μικτής λειτουργίας κατά την περίοδο ψύξης.

Στην αναγνώριση σημαντικών κανόνων για τη λειτουργία των κτιρίων μπορούν επίσης να βοηθήσουν οι κανόνες συσχέτισης (ARs). Αυτή η τεχνική μπορεί να επεξεργαστεί δεδομένα που σχετίζονται με τα κτίρια και να εξάγει κρυφές συσχετίσεις που δεν είναι τόσο εμφανείς για έμπειρους στη διαχείριση ενέργειας. Οι Yu και συνεργάτες του [45] χρησιμοποίησαν κανόνες συσχέτισης για να προσδιορίσουν συσχετίσεις στα δεδομένα λειτουργίας κτιρίων, ενώ οι Xiao και συνεργάτες [46], χρησιμοποίησαν κανόνες συσχέτισης για να προσδιορίσουν τις σχέσεις μεταξύ της κατανάλωσης ενέργειας των κύριων συστατικών σε κάθε ομάδα.

Άλλες μελέτες επικεντρώθηκαν στη βελτίωση συγκεκριμένων τμημάτων ή μεταβλητών του κτιρίου. Για παράδειγμα, οι Kusiak και συνεργάτες του [47] ερεύνησαν τις σχέσεις μεταξύ των

ρυθμίσεων ελέγχου της μονάδας χειρισμού αέρα και της κατανάλωσης ενέργειας του συστήματος HVAC. Σε αυτήν την περίπτωση, χρησιμοποιήθηκε ένας αλγόριθμος που βασίζεται στον συνδυασμό νευρωνικών δικτύων για τη μοντελοποίηση της μη γραμμικής σχέσης μεταξύ της κατανάλωσης ενέργειας, των ρυθμίσεων ελέγχου (θερμοκρασία εισόδου αέρα και στατική πίεση εισόδου αέρα) και ενός συνόλου μη ελέγξιμων παραμέτρων.

Τέλος, για την ανάλυση των συνόλων δεδομένων που παράγονται από συστήματα αυτοματισμού κτιρίων έχουν εφαρμοστεί , επίσης, τεχνικές συσταδοποίησης. Οι Xaio και Fan [46] χρησιμοποίησαν ανάλυση συστάδων για να αναγνωρίσουν μοτίβα ημερήσιας κατανάλωσης ενέργειας, ενώ οι Morbitzer και συνεργάτες του [48] εφάρμοσαν συσταδοποίηση για να αναλύσουν τα αποτελέσματα προσομοίωσης για προβλέψεις επίδοσης και να εξάγουν κανόνες πρόβλεψης λειτουργίας.

3. **Ανάλυση και αναβάθμιση υποδομών :** Η μελέτη και η ανάλυση πτυχών που σχετίζονται με την ενεργειακή απόδοση και τη βιώσιμη ανάπτυξη αποτελούν, πλέον, προτεραιότητες για τους σχεδιαστές και τους ιδιοκτήτες κτιρίων, ανεξαρτήτως εάν το κτίριο είναι νεοκατασκευασμένο ή σε ανακαίνιση.

   Στο πλαίσιο αυτό εξετάζονται οι σχέσεις μεταξύ των φορτίων ενέργειας, της πραγματικής κατανάλωσης και των διαφόρων στοιχείων του κτιρίου , όπως τοίχοι, παράθυρα, πόρτες, φωτισμός, θέρμανση, ψύξη και εξαερισμός. Όταν πρέπει να επιβεβαιωθεί η κατάσταση του εξοπλισμού και να ανιχνευθούν πιθανά προβλήματα, οι τεχνικές επιστήμης δεδομένων αποτελούν ένα ισχυρό εργαλείο για την εξαγωγή χρήσιμων προτύπων και συσχετίσεων μεταξύ αυτών των στοιχείων.

   Οι τεχνικές ομαδοποίησης και ταξινόμησης είναι εξαιρετικά αποτελεσματικές στην ανάλυση των συσχετίσεων μεταξύ της υποδομής κτιρίων και της απόδοσής τους. Οι Morbitzer και συνεργάτες [48] εφάρμοσαν αλγόριθμους συσταδοποίησης για την επεξεργασία δεδομένων παρακολούθησης κτιρίων και την ανακάλυψη μη-προφανών παραγόντων απώλειας ενέργειας στις υποδομές των κτιρίων. Από την άλλη, οι Ahmed και συνεργάτες [49] χρησιμοποίησαν μοντέλα ταξινόμησης για να εκτιμήσουν δείκτες συμπεριφοράς του κτιρίου, όπως άνεση και χρήση δωματίων. Αυτοί κατέληξαν ότι η προσέγγισή τους παρήγαγε καλύτερα αποτελέσματα σε σύγκριση με τα παραδοσιακά αναλυτικά εργαλεία. Ωστόσο, και οι δύο εφάρμοσαν επίσης τεχνικές ταξινόμησης και παλινδρόμησης σε συνδυασμό με μεθόδους εσωτερικού φωτισμού κτιρίων για να υποστηρίξουν τη λήψη αποφάσεων και την βελτιστοποίηση του σχεδιασμού κτιρίων [50].

   Ένα ακόμη χρήσιμο εργαλείο αποτελούν οι τεχνικές ανάλυσης ακολουθιών, όπως για παράδειγμα είναι η εξόρυξη μοτίβων. Αξιοποιώντας αυτά , οι Shao και συνεργάτες [51] εξήγαγαν σχέσεις για να συγκρίνουν καλύτερα τις τάσεις κατανάλωσης σε κατοικίες και εμπορικά κτίρια με διαφορετική ηλεκτρική υποδομή.

   Επιπλέον, η Επιστήμη Δεδομένων μπορεί να αποτελέσει κρίσιμο εργαλείο για τους σχεδιαστές κτιρίων στη διαδικασία λήψης αποφάσεων, εντοπίζοντας πολύπλοκες αλληλοσυσχετίσεις και πρότυπα. Μέσω της ανάλυσης μεγάλων συνόλων δεδομένων, οι σχεδιαστές μπορούν να ανακαλύψουν σημαντικές τάσεις και παράγοντες που επηρεάζουν τις ενεργειακές επιδόσεις των κτιρίων, υποστηρίζοντας έτσι τον σχεδιασμό κτιρίων με χαμηλές εκπομπές και αυξημένη ενεργειακή αποδοτικότητα. Επιπλέον, με την αξιοποίηση τεχνικών μηχανικής μάθησης και προγνωστικών μοντέλων, μπορούν να ληφθούν υπόψη παράγοντες όπως οι κλιματολογικές συνθήκες, τα υλικά κατασκευής και οι ενεργειακές ανάγκες, για τη δημιουργία βιώσιμων και φιλικών προς το περιβάλλον κτιρίων.

4. **Επιβεβαίωση της λειτουργικής κατάστασης - Ανίχνευση και πρόληψη βλαβών:** Μέσω της συνεχούς παρακολούθησης του κτιρίου με προηγμένα συστήματα, είναι δυνατόν να ανιχνευθεί πότε συμβαίνει μια βλάβη ή μια αποτυχία . Η ανάλυση αυτών των δεδομένων

επιτρέπει την κατανόηση του τρόπου με τον οποίο η βλάβη επηρεάζει τον υπόλοιπο εξοπλισμό και το δίκτυο του κτιρίου.

Σημαντικό είναι επίσης να μπορούν να προβλεφθούν τέτοιες βλάβες Σε αυτό το πλαίσιο, η αναγνώριση προτύπων και οι τεχνικές παλινδρόμησης είναι πολύ χρήσιμες ,καθώς μπορούν να βοηθήσουν στην υλοποίηση μέτρων για την αύξηση της ανθεκτικότητας του κτιρίου και στην πρόληψη δαπανηρών περιστατικών.

Ο Capozzoli και οι συνεργάτες του [52] περιέγραψαν μια προσέγγιση για την αυτόματη ανίχνευση βλαβών στον εξοπλισμό ενέργειας κτιρίου. Η μεθοδολογία τους βασίζεται στην ανάλυση καταγεγραμμένων δεδομένων ενεργητικής ηλεκτρικής ισχύος για τον φωτισμό και τη συνολική ενεργητική ηλεκτρική ισχύ χρησιμοποιώντας νευρωνικά δίκτυα και ανιχνευτές ανωμαλιών.

Οι Sedano και συνεργάτες του [53] παρουσίασαν μια παρόμοια πρόταση για την ανίχνευση αποτυχιών θερμομονωτικών υλικών σε κτίρια, επίσης βασιζόμενοι σε νευρωνικά δίκτυα. Η αναγνώριση των συνεπειών των βλαβών σε κρίσιμες υποδομές έχει εξεταστεί με τη χρήση τεχνικών ανάλυσης γράφων [54].

Με τη χρήση τεχνικών ανάλυσης συσχέτισης, μπορούν να εντοπιστούν οι βασικές αιτίες των βλαβών και να εφαρμοστούν προληπτικά μέτρα. Επιπλέον, η πρόβλεψη πιθανών μελλοντικών βλαβών μέσω μοντέλων μηχανικής μάθησης μπορεί να συμβάλει στη διατήρηση της απρόσκοπτης λειτουργίας του κτιρίου και στη μείωση των απρογραμμάτιστων διακοπών και προβλημάτων.

Υπάρχουν , ωστόσο, και βλάβες λόγω εξωτερικών φαινομένων και απρόβλεπτων γεγονότων, όπως ενδεχόμενες προθέσεις και τυχαίες αποτυχίες, οι οποίες δεν μπορούν πάντα να προβλεφθούν βασιζόμενες σε προηγούμενα δεδομένα. Σε αυτές τις περιπτώσεις, η διενέργεια ενός διαγνωστικού ελέγχου μπορεί να αποκαλύψει πρότυπα που αυξάνουν την επίπτωση των βλαβών.

Οι Meléndez και συνεργάτες του [55] χρησιμοποίησαν τεχνικές αναγνώρισης προτύπων για την αναγνώριση ακολουθιών ηλεκτρικών γεγονότων σε υποσταθμούς που συσχετίζονται με σημαντικές αποτυχίες. Αυτή η διαδικασία ανάλυσης μπορεί να ενισχυθεί αν τα δεδομένα αισθητήρων εμπλουτιστούν με πληροφορίες περιβάλλοντος, όπως σημασιολογικά δεδομένα κτιρίων [56] και βάσεις δεδομένων διασύνδεσης [57]. Σε αυτήν την περίπτωση, οι τεχνικές σύνθεσης δεδομένων παρέχουν υποστήριξη για την ευθυγράμμιση των δεδομένων, την ανίχνευση γεγονότων και την αξιολόγηση της κατάστασης.

5. **Ανάλυση του οικονομικού και εμπορικού αντίκτυπου της κατανάλωσης ενέργειας των χρηστών :** Πολλές εταιρείες , κυρίως πάροχοι υπηρεσιών διανομής ενέργειας, αναζητούν τη βοήθεια της Επιστήμης Δεδομένων προκειμένου να ανακαλύψουν και να κατανοήσουν πώς και πότε οι πελάτες τους χρησιμοποιούν ενέργεια. Οι τεχνικές που παραδοσιακά χρησιμοποιούνται για αυτό τον σκοπό είναι η ταξινόμηση, η ομαδοποίηση και η ανάλυση προτύπων.

Ένα από τα πρώτα έργα σε αυτήν την κατεύθυνση ήταν αυτό του Chicco και συνεργατών [58], οι οποίοι ομαδοποίησαν τους πελάτες σε κατηγορίες, βασισμένες στη συμπεριφορά τους ως προς την κατανάλωση της ηλεκτρικής ενέργειας. Για τη σύγκριση των αποτελεσμάτων χρησιμοποιήθηκαν αυτοοργανωμένοι χάρτες και η ταξινόμηση που προέκυψε από αυτό ήταν ένα πρώτο βήμα προς την καθορισμό των επιλογών διαφοροποίησης τιμολογίων.

Ο αλγόριθμος ISPC (Incremental Summarization and Pattern Characterization) χρησιμοποιήθηκε από τους De Silva και συνεργάτες [59] για να δομήσει τα δεδομένα συνεχούς ροής σε μια αποθήκη δεδομένων, προκειμένου να διευκολυνθεί η γρήγορη προσωρινή

περίληψη. Αυτή η μελέτη επέτρεψε τη δημιουργία συνεχών περιλήψεων, που συγχωνεύουν περιοδικά τα αναγνωρισμένα πρότυπα, επιτρέποντας έτσι την ανάλυση των δεδομένων και την πρόβλεψή τους. Έχουν δημιουργηθεί επίσης μοντέλα για να παρέχουν ενδιαφέρουσες πληροφορίες σχετικά με την ενεργειακή σπατάλη λόγω φωτισμού, δίνοντας τη δυνατότητα στις εταιρείες να προτείνουν λύσεις για τη μείωση της ενεργειακής κατανάλωσης και την προώθηση πιο βιώσιμων πρακτικών.

Έχουν δημιουργηθεί επίσης μοντέλα σχετικά με την ενεργειακή σπατάλη λόγω του φωτισμού. Στη μελέτη του φωτισμού, οι Motta-Cabrera και Zareipour [60] χρησιμοποίησαν κανόνες συσχέτισης για να ανακαλύψουν τις υπάρχουσες σχέσεις μεταξύ χρόνου, κατοικίας και ενεργειακής σπατάλης που σχετίζεται με τον φωτισμό στις αίθουσες διδασκαλίας. Οι Santamouris και συνεργάτες [61] πρότειναν μια μέθοδο για την αξιολόγηση και την ταξινόμηση της κατανάλωσης ενέργειας και της αποδοτικότητας των σχολικών κτιρίων σε σύγκριση με άλλα παρόμοια κτίρια. Βρήκαν ότι οι τεχνικές ασαφούς ομαδοποίησης θα μπορούσαν να χρησιμοποιηθούν για να προστατεύσουν ένα πιο ανθεκτικό σύνολο τάξεων, αποφεύγοντας έτσι προβλήματα που πηγάζουν από μη ισορροπημένη ταξινόμηση.

6. **Ανίχνευση και πρόληψη απάτης στην ενέργεια :** Μερικές φορές, η κατανάλωση ενέργειας και οι υπηρεσίες δεν χρεώνονται κατάλληλα λόγω αποτυχιών στον εξοπλισμό μέτρησης. Αυτές οι αποτυχίες μπορεί να είναι είτε ατύχημα είτε προϊόν από απάτη.

   Ο León και οι συνεργάτες του [62] πρότειναν ένα πλαίσιο για την ανίχνευση, μέσω ανάλυσης εξόρυξης δεδομένων, μη τεχνικών απωλειών (NTLs) και την ανάκτηση ηλεκτρικής ενέργειας , η οποία χανόταν λόγω ανωμαλιών ή κάποιας απάτης. Το προγνωστικό εργαλείο ανάλυσής τους, το οποίο περιλάμβανε και μια μέθοδο κατηγοριοποίησης δέντρου δυαδικής αναζήτησης, χρησιμοποιήθηκε για την εντοπισμό συσχετισμών στα δεδομένα. Αυτοί οι ίδιοι συγγραφείς [63] πρότειναν ένα σύστημα για την ανίχνευση των NTL, το οποίο περιλάμβανε ανάλυση και ταξινόμηση όλων των διαθέσιμων χρήσιμων πληροφοριών για κάθε πελάτη από ένα ολοκληρωμένο σύστημα εμπειρογνωμόνων. Σε αυτό εντάσσονται τεχνικές εξόρυξης κειμένου για την ανάλυση των σχολίων των επιθεωρητών και την εξαγωγή πρόσθετων πληροφοριών για τον πελάτη, εξόρυξη δεδομένων για την εκτιμώμενη κατανάλωση του πελάτη, καθώς και κανόνες συσχετισμού για την εξαγωγή επιπρόσθετων πληροφοριών πελατών από την ηλεκτρική εταιρεία.

   Η ανίχνευση ανωμαλιών είναι η πιο δημοφιλής τεχνική και έχει χρησιμοποιηθεί συχνά για την ανακάλυψη μη επιθυμητών συμπεριφορών των χρηστών, όπως η απάτη, η εισβολή ή η προεπιλογή λογαριασμού.

   Ο Cabral και οι συνεργάτες του [64] πρότειναν μια μεθοδολογία για τον εντοπισμό απάτης για καταναλωτές ηλεκτρικής ενέργειας , η οποία βασίζεται σε ένα μη επιβλεπόμενο τεχνητό νευρωνικό δίκτυο που ονομάζεται SOM (Χάρτες Αυτό-Οργάνωσης), το οποίο επιτρέπει την ταυτοποίηση του προφίλ κατανάλωσης που έχει καταγραφεί ιστορικά για έναν καταναλωτή, και τη σύγκρισή του με την παρούσα συμπεριφορά, υποδεικνύοντας πιθανές απάτες. Τα νευρωνικά δίκτυα έχουν συνδυαστεί για τον εντοπισμό της ανώμαλης συμπεριφοράς των πελατών και σε άλλες εργασίες ,όπως αυτή του Sforna [65].

7. **Ασφάλεια στη διαχείριση δεδομένων**: Αυτή η ενότητα περιλαμβάνει τόσο την έρευνα όσο και την τεχνολογία για τη δικτύωση και την ασφάλεια των δεδομένων και την ιδιωτικότητα κατά τη χρήση των υπηρεσιών Διαδικτύου ή της οικιακής υποδομής [66].

2.1.2. ΠΑΡΑΔΕΙΓΜΑΤΑ ΕΦΑΡΜΟΓΩΝ

Το ενδιαφέρον αυτής της διπλωματικής επικεντρώνεται στις τεχνικές που αναφέρθηκαν παραπάνω και αφορούν τη διαχείριση των δεδομένων ενέργειας με σκοπό την ανάλυσή τους και την εξαγωγή συμπερασμάτων και προβλέψεων. Μεταξύ των κύριων προόδων για την κατανάλωση ενέργειας μπορούμε να αναφέρουμε τα παρακάτω παραδείγματα:

1. Δημιουργία μοντέλου πρόβλεψης κατανάλωσης ενέργειας με τη μέθοδο Ανάλυσης Επέκτασης Νευρώνων για ερμηνεύσιμες Χρονολογικές Σειρές (N-BEATS) εκπαιδευμένο σε ένα μεγάλο σύνολο δεδομένων κατανάλωσης ενέργειας από 169 πελάτες στο [67].

   Λόγω έλλειψης δεδομένων για πολλούς πελάτες, τα μοντέλα που ήδη υπήρχαν δεν μπορούσαν να προβλέψουν με ακρίβεια την κατανάλωση ενέργειας όταν εφαρμόζονταν σε δεδομένα από πολλούς πελάτες ,ενώ εξέταζαν μόνο τη μελλοντική κατανάλωση ενέργειας χωρίς να λαμβάνουν υπόψη τη συμπεριφορά του πελάτη.

   Το προτεινόμενο μοντέλο βελτιώνει την ακρίβεια πρόβλεψης σε μεγάλα σύνολα δεδομένων που περιέχουν προφίλ κατανάλωσης ενέργειας πολλαπλών πελατών και έχει επιδείξει καλύτερη απόδοση στις προβλέψεις καθημερινής, εβδομαδιαίας και μηνιαίας κατανάλωσης ενέργειας των 43 πελατών που χρησιμοποιήθηκαν σαν δεδομένα δοκιμής.

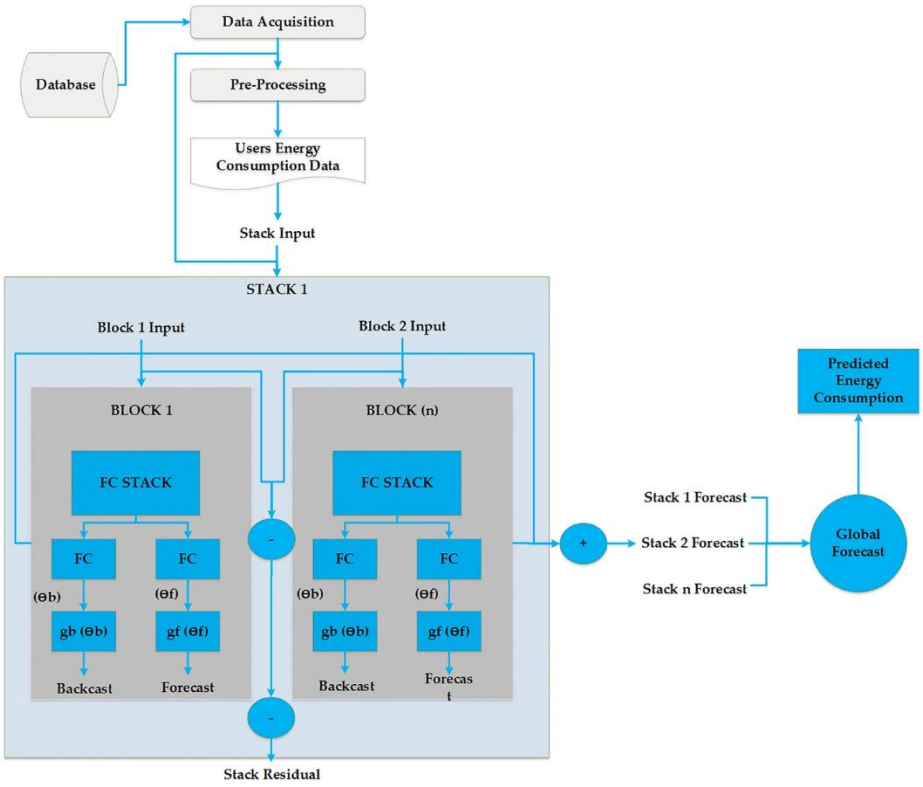
   Το μοντέλο που αναπτύχθηκε είναι βασισμένο στο N-BEATS και λαμβάνει υπόψη τη συμπεριφορά των πελατών για την πρόβλεψη κατανάλωσης ενέργειας για μέρες, εβδομάδες και μήνες εκ των προτέρων , προκειμένου να επιτευχθεί αποτελεσματικότερη διαχείριση της ζήτησης.

   Για να το πετύχει αυτό, χρησιμοποιεί δεδομένα που αντικατοπτρίζουν τη συμπεριφορά κατανάλωσης από πολλούς πελάτες , σε αντίθεση με τις παραδοσιακές μεθόδους που ασχολούνταν με έναν μικρό μόνο αριθμό χρηστών. Η εξέταση των δεδομένων από πολλούς πελάτες καθιστά το μοντέλο μοναδικό και αξιόπιστο για το έξυπνο δίκτυο. Επίσης, πραγματοποιεί ανάλυση χρονοσειρών της εισόδου και διατηρεί τη συμπεριφορά των χρονοσειρών ως μέρος της διαδικασίας εκπαίδευσής του.

   Σκοπός είναι να εξαλειφθεί το πρόβλημα της υπερεκπαίδευσης που προκαλείται από αλλαγές στο πρότυπο δεδομένων με την πάροδο του χρόνου λόγω ολοένα και μεγαλύτερων μεταβολών στο πλήθος και την ποιότητα των δεδομένων.

   Για την αξιολόγηση του προτεινόμενου μοντέλου N-BEATS χρησιμοποιείται μια ποικιλία προηγμένων αλγορίθμων βαθιάς μάθησης, συμπεριλαμβανομένων των LSTMs, interpretable LSTMs, GRUs, interpretable GRUs και TCNs.

   Η προτεινόμενη μεθοδολογία για την πρόβλεψη της μελλοντικής κατανάλωσης ενέργειας στοχεύει στη βελτίωση της ακρίβειας του ερμηνεύσιμου αλγορίθμου N-BEATS. Το μοντέλο παρουσιάζεται στο παρακάτω σχήμα :



Εικόνα 2.1 : N-BEATS ερμηνεύσιμο μοντέλο νευρωνικής ανάλυσης επέκτασης βάσης για χρονοσειρές [67]

2. Ένα ευρωπαϊκό έργο το οποίο χρηματοδοτείται από το πρόγραμμα Horizon 2020 της ΕΕ και στοχεύει στην αποδοτική διαχείριση της ενέργειας από τους τελικούς χρήστες παρουσιάζεται στο [68]. Το έργο InterConnect εργάζεται για την επίλυση αυτών των ζητημάτων σημασιολογικής διαλειτουργικότητας για να υποστηρίξει την Οδηγία Ενεργειακής Απόδοσης της ΕΕ, η οποία θέτει έναν δεσμευτικό στόχο στα κράτη μέλη να μειώσουν την τελική κατανάλωση ενέργειας κατά τουλάχιστον 11,7% μέχρι το 2030. Το έργο δημιουργεί μεγάλης κλίμακας πιλοτικά προγράμματα που θα χρησιμοποιούν στόχους Δεικτών Βασικής Απόδοσης (KPI) για να μετρήσουν αν η ΕΕ βρίσκεται σε καλό δρόμο για την επίτευξη του στόχου. Τα πιλοτικά προγράμματα θα επιδιώξουν να μεγιστοποιήσουν την ενσωμάτωση των κτιρίων με μικροδίκτυα, την ηλεκτροκίνηση και τις υπηρεσίες έξυπνων πόλεων και στη συνέχεια να μετρήσουν τις βελτιώσεις στην ενεργειακή απόδοση που προκύπτουν.

   Το πρόβλημα που έπρεπε να αντιμετωπιστεί ήταν διαλειτουργικότητα των δεδομένων και η ασυμβατότητα ανάμεσα σε συστήματα και συσκευές σε διαφορετικές χώρες της ΕΕ , η οποία εμπόδιζε την αποτελεσματική μεταξύ τους επικοινωνία.

   Βασικό παράδειγμα αποτελεί η αγορά ενός έξυπνου θερμοστάτη ,με δυνατότητα σύνδεσης στο Wi-Fi, από κάποιον καταναλωτή ,που όμως καταλήγει σε αδυναμία εγκατάστασής του , καθώς ο θερμοστάτης δεν μπορεί επικοινωνήσει με τον λέβητα αερίου του. Εν συνεχεία, ο λέβητας αερίου και η μονάδα ηλεκτρικής ενέργειας μπορεί να μην είναι σε θέση να επικοινωνούν με τον διαχειριστή του δικτύου και την υπηρεσία κοινής ωφέλειας σε πραγματικό χρόνο για να τους παρέχουν δεδομένα σχετικά με την κατανάλωση.

   Παρόμοια εμπόδια που μπορεί να προκύψουν είναι η αδυναμία επιστροφής στους χρήστες και στις κυβερνήσεις των δεδομένων που παράγονται από το δίκτυο ,η ασυμβατότητα των δεδομένων στις κυβερνήσεις , αλλά και η ασυμβατότητα έξυπνων συσκευών που αγοράζονται από ένα κράτος μέλος της ΕΕ με το σύστημα δεδομένων ενός άλλου κράτους.

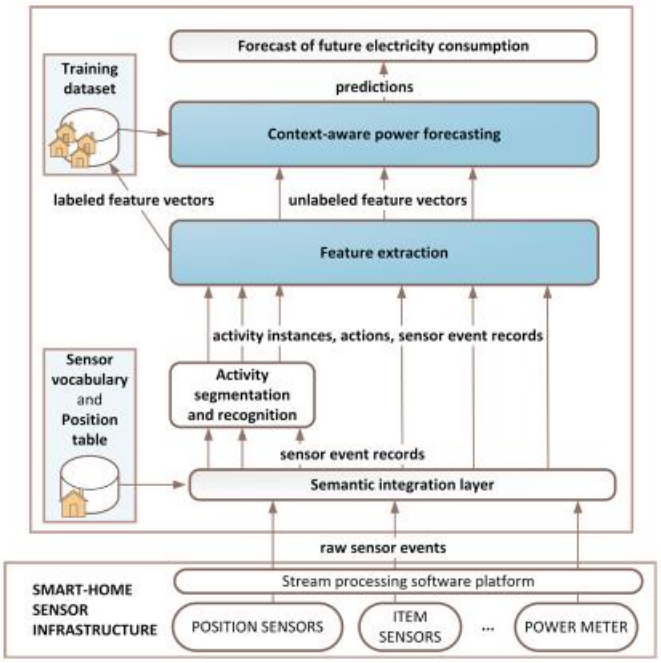
Το έργο ανέπτυξε επίσης τρεις εφαρμογές για κινητά: τη Flexi App στην Ιταλία, τη Pilot App στην Ελλάδα και την The Wattch.r στην Πορτογαλία. Αυτές οι εφαρμογές επιτρέπουν στους καταναλωτές να παρακολουθούν την κατανάλωση ενέργειάς τους, να κατανοούν πώς οι επιλογές τους επηρεάζουν το δίκτυο, να εντοπίζουν ευκαιρίες για μείωση της κατανάλωσης ενέργειας και εξοικονόμηση χρημάτων και να υποστηρίζουν τη σταθερότητα του δικτύου μέσω έξυπνης διαχείρισης ενέργειας.

Οι τεχνολογίες που αναπτύχθηκαν από το InterConnect αποτέλεσαν τη βάση για νέα έργα που περιλαμβάνουν λύσεις που αφορούν τη βελτίωση της διαλειτουργικότητα μεταξύ των συσκευών έξυπνου σπιτιού και των συστημάτων κτιρίων , τη βελτιστοποίηση της ευελιξίας από την πλευρά της ζήτησης για να επιτευχθεί μεγαλύτερη σταθερότητα δικτύου, την ανάπτυξη καινοτόμων λύσεων για ηλεκτρικά οχήματα και υποδομές φόρτισης , καθώς και τη δημιουργία έξυπνων αγορών ενέργειας και μηχανισμών κινήτρων.

Τα επόμενα έργα που πρόκειται να αναπτυχθούν έχουν ως στόχο να φτάσουμε σε ένα μέλλον όπου το σπίτι δεν είναι απλώς ένας τόπος διαμονής αλλά ένα ενεργό συμμετέχον στο ενεργειακό δίκτυο, όπου οι καταναλωτές έχουν τη δυνατότητα να κάνουν ενημερωμένες επιλογές για την κατανάλωση ενέργειας τους και όπου οι επιχειρήσεις μπορούν να αναπτύξουν καινοτόμες λύσεις που ευνοούν τόσο το περιβάλλον όσο και το οικονομικό τους αποτέλεσμα.

3. Ένα σύστημα πρόβλεψης κατανάλωσης ενέργειας μέσω προσέγγισης βαθιάς μάθησης προτάθηκε στο [69] . Όπως έχει αναφερθεί και παραπάνω, η πρόβλεψη της μελλοντικής κατανάλωσης ενέργειας σε κατοικίες είναι σημαντική για τη βελτιστοποίηση του δικτύου ηλεκτρικής ενέργειας, την υποστήριξη των κατοίκων στις καθημερινές τους δραστηριότητες και την εξοικονόμηση ενέργειας. Σε αυτή την εφαρμογή λαμβάνονται υπόψη όχι μόνο τα δεδομένα ιστορικής κατανάλωσης, όπως στις περισσότερες μεθόδους που έχουν προταθεί, αλλά και τα δεδομένα περιβάλλοντος όπως οι ενέργειες και οι δραστηριότητες των κατοίκων, η χρήση οικιακών συσκευών, η αλληλεπίδραση με έπιπλα ,πόρτες και περιβαλλοντικά δεδομένα.

Στην παρακάτω εικόνα παρουσιάζεται μια επισκόπηση του συστήματος :



Εικόνα 2.2 : Επισκόπηση συστήματος [69]

Το έξυπνο σπίτι είναι εξοπλισμένο με διάφορους αισθητήρες για τη συλλογή δεδομένων σχετικά με τις δραστηριότητες των κατοίκων και τις περιβαλλοντικές συνθήκες, όπως είναι οι αισθητήρες θέσης για την ανίχνευση κινήσεων στο σπίτι, καθώς και αισθητήρες για την ανίχνευση αλληλεπιδράσεων με συσκευές και αντικείμενα. Περιλαμβάνεται ,επίσης, ένας μετρητής ισχύος που περιοδικά ανιχνεύει την στιγμιαία κατανάλωση ηλεκτρικής ενέργειας στο διαμέρισμα.

Όλα τα ακατέργαστα δεδομένα που παράγονται από αυτούς τους αισθητήρες επικοινωνούν με ένα επίπεδο λογισμικού "σημασιολογικής ενσωμάτωσης", το οποίο ,όπως φαίνεται, είναι υπεύθυνο για την ανάθεση σημασιολογίας στα δεδομένα σύμφωνα με ένα λεξιλόγιο αισθητήρων και αποθηκεύει σε έναν πίνακα θέσεων τη σχετική θέση κάθε αισθητήρα στο σπίτι, για να προσδιορίσει την τοποθεσία των κατοίκων με βάση τους ενεργοποιημένους αισθητήρες. Τα δεδομένα που προκύπτουν από αυτό ονομάζονται "εγγραφές συμβάντων αισθητήρων".

Στη συνέχεια, η ροή αυτών των εγγραφών επεξεργάζεται και πραγματοποιείται η τμηματοποίηση και η αναγνώριση των δραστηριοτήτων που συμβαίνουν στο σπίτι, καθώς και των ενεργειών που αποτελούν αυτές τις δραστηριότητες. Η διαδικασία αυτή περιλαμβάνει αλγόριθμους για την τμηματοποίηση των δραστηριοτήτων και την αναγνώριση τους ως προς την έναρξη και το τέλος τους.

Έπειτα, οι αναγνωρισμένες δραστηριότητες, ενέργειες και εγγραφές συμβάντων αισθητήρων, συμπεριλαμβανομένων των μετρήσεων του μετρητή ισχύος, αποστέλλονται για εξαγωγή στατιστικών χαρακτηριστικών, χρησιμοποιώντας ένα παράθυρο ολίσθησης σταθερού μεγέθους. Για κάθε παράθυρο υπολογίζεται ένα διάνυσμα χαρακτηριστικών.

Τα διανύσματα χαρακτηριστικών αποστέλλονται για πρόβλεψη κατανάλωσης ενεργειακής ευαισθησίας, όπου γίνεται ο υπολογισμός για την πρόβλεψη παροχής της μελλοντικής κατανάλωσης ηλεκτρικής ενέργειας στο έξυπνο σπίτι. Για το σκοπό αυτό, χρησιμοποιείται ένας επιβλεπόμενος αλγόριθμος μηχανικής μάθησης που έχει εκπαιδευτεί σε ένα ανώνυμο σύνολο δεδομένων που συλλέχθηκαν από διαφορετικά έξυπνα σπίτια.

4. Ένα ενοποιημένο σύστημα που καλύπτει όλες τις πτυχές της διαχείρισης της ενέργειας, της ευημερίας και της υγείας σε μια μόνο λύση αναπτύχθηκε στο [70].

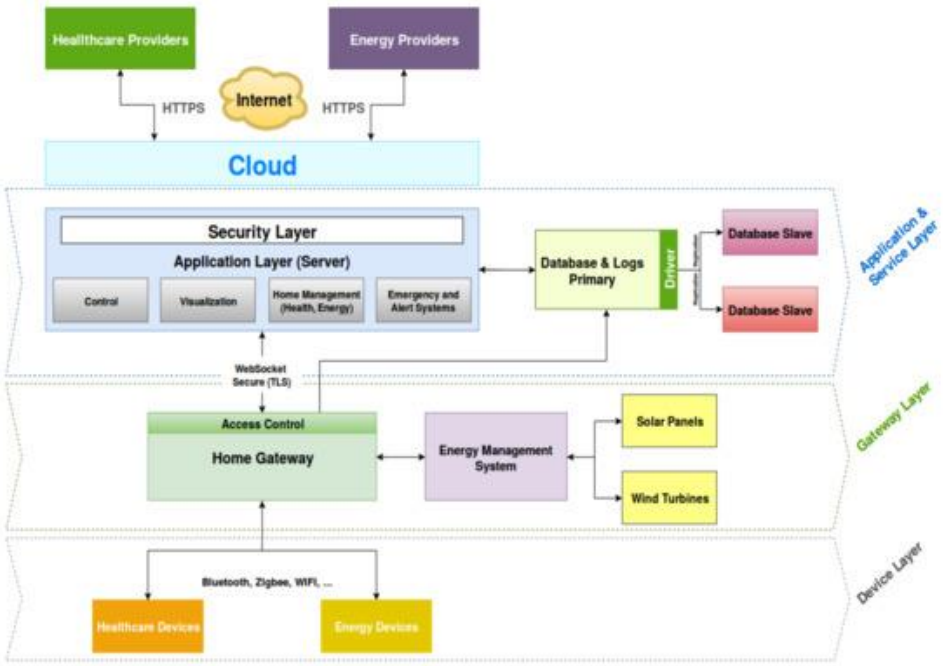
   Προτείνεται μια υλοποίηση για τα έξυπνα σπίτι επόμενης γενιάς, όπου ετερογενή δεδομένα, που προέρχονται από πολλαπλούς αισθητήρες και συσκευές σπιτιού ,όπως έξυπνο ψυγείο, έξυπνη τηλεόραση, κλπ ,πρέπει να διαχειρίζονται και να οπτικοποιούνται. Στοχεύει να θέσει τα θεμέλια για τη διαχείριση κάθε τύπου πληροφορίας που προκύπτει από τις αλληλεπιδράσεις με το έξυπνο σπίτι, οι οποίες μπορεί να περιλαμβάνουν τεχνητή νοημοσύνη και μηχανική μάθηση.

   Το προτεινόμενο σύστημα ασχολείται τόσο με το κομμάτι της ηλεκτρονικής υγείας, όσο και με το κομμάτι της ενέργειας.

   Σχετικά με το μέρος της ενέργειας, το οποίο μας ενδιαφέρει, ο στόχος είναι η συλλογή της κατανάλωσης ενέργειας των χρηστών μέσα στο έξυπνο σπίτι, η οποία μπορεί να προέρχεται από διάφορες πηγές (θέρμανση, νερό, αέριο ή ηλεκτρικό ρεύμα), και η δυνατότητα χρήσης προηγμένων αλγορίθμων για την πρόβλεψη και διαχείριση της τοπικής κατανάλωσης και παραγωγής ενέργειας. Αυτή η προσέγγιση συνδυάζει δεδομένα που συλλέγονται από έξυπνους μετρητές, λειτουργικές πληροφορίες από έξυπνες ενεργειακές συσκευές ,όπως η κατάσταση των έξυπνων πριζών, τις προτιμήσεις των χρηστών και εξωτερικά σήματα δικτύου όπως οι τιμές της ενέργειας. Χρησιμοποιώντας ένα σύστημα διαχείρισης ενέργειας στο σπίτι που δέχεται τέτοιες παραμέτρους εισόδου, η λειτουργία των συσκευών και ειδών στο σπίτι

μπορεί να ελέγχεται βέλτιστα σύμφωνα με διαφορετικούς στόχους ,όπως είναι η ελαχιστοποίηση του κόστους ενέργειας κα η μέγιστη άνεση του χρήστη.

Η αρχιτεκτονική του μοντέλου που προτείνεται αποτελείται από 4 επίπεδα και παρουσιάζεται στο παρακάτω σχήμα :



Εικόνα 2.3 : Αρχιτεκτονική έξυπνου σπιτιού επόμενης γενιάς [70]

- **Επίπεδο Συσκευών:** αποτελείται από όλες τις IoT συσκευές και τους αισθητήρες στο έξυπνο σπίτι. Σε αυτό το έργο, οι συσκευές χωρίζονται στις ιατρικές IoT συσκευές και τις ενεργειακές συσκευές . Στη δεύτερη κατηγορία ανήκουν και οι έξυπνοι μετρητές, οι οποίοι χρησιμοποιούνται για τη συλλογή της κατανάλωσης ενέργειας, όπως οι έξυπνες πρίζες, θερμότητας και νερού. Οι ανανεώσιμες πηγές ενέργειας είναι επίσης εξοπλισμένες με αισθητήρες που μετρούν το ρεύμα, την τάση, την ισχύ .
- **Επίπεδο Πύλης:** παρέχει τη δυνατότητα επικοινωνίας με τις διάφορες έξυπνες συσκευές και λήψης των δεδομένων από αυτές χρησιμοποιώντας διάφορα πρωτόκολλα επικοινωνίας , όπως Bluetooth, WIFI και ZigBee. Η πύλη αλληλοεπιδρά επίσης με βάσεις δεδομένων για την αποθήκευση δεδομένων που προέρχονται από όλους τους αισθητήρες, σφάλματα, ειδοποιήσεις και λειτουργικά αρχεία καταγραφής. Από αυτό το επίπεδο, δημιουργείται μια ασφαλής υποδοχή web (web socket) μεταξύ του διακομιστή εφαρμογών και της πύλης του σπιτιού για την ασφαλή ανταλλαγή δεδομένων και εντολών. Και τα δύο στοιχεία βρίσκονται σε ένα ιδιωτικό τοπικό δίκτυο του έξυπνου σπιτιού και συνδέονται σε ασφαλές WiFi.
- **Επίπεδο Εφαρμογών και Υπηρεσιών:** περιλαμβάνει τις κύριες υπηρεσίες του έξυπνου σπιτιού ,μαζί με τη δυνατότητα αλληλεπίδρασης με εξωτερικούς παράγοντες, και τον έλεγχο των έξυπνων συσκευών. Μέσω μιας διεπαφής οπτικοποίησης, παρέχει τη δυνατότητα ελέγχου και διαχείρισής όλων των δεδομένων που συλλέγονται από τους διάφορους αισθητήρες του έξυπνου σπιτιού. Ένας διακομιστής Node.js που παρέχει ένα Web API είναι αναπτυγμένος. Αυτός ο διακομιστής μπορεί είτε να εξάγει δεδομένα σε πραγματικό χρόνο απευθείας από την πύλη είτε να κάνει ερωτήματα σε μια βάση δεδομένων για ιστορικά αρχεία. Επιπλέον, για τη λήψη ειδοποιήσεων και μηνυμάτων σφάλματος από συγκεκριμένους κρίσιμους αισθητήρες αναπτύχθηκε μια απλή εφαρμογή για Android, η οποία χρησιμοποιεί Bluetooth Low Energy (BLE) για να επικοινωνεί με την πύλη και μπορεί είτε να

διαβάζει, να γράφει ή να εγγράφεται σε κάποιον αισθητήρα. Για το κομμάτι της ενέργειας, τέλος, χρησιμοποιείται το Σύστημα Διαχείρισης Ενέργειας (EMS) για την ανάλυση των μετρούμενων δεδομένων .

- **Επίπεδο Ασφαλείας :** προστίθεται πάνω από το επίπεδο εφαρμογής, έτσι ώστε οι χρήστες που έχουν πρόσβαση σε πόρους και δεδομένα του συστήματος να είναι εξουσιοδοτημένοι. Επιπλέον, όλες οι ανταλλαγές δεδομένων είναι κρυπτογραφημένες χρησιμοποιώντας AES128 για να παρέχουν εμπιστευτικότητα και ακεραιότητα. Με αυτό τον τρόπο εξασφαλίζεται η ιδιωτικότητα και η προστασία των προσωπικών δεδομένων των χρηστών .

Όλες αυτές οι υπηρεσίες μπορούν να αναπτυχθούν στο Cloud.

Οι πάροχοι τρίτων μερών προστίθενται προκειμένου να συνδέσουν το έξυπνο σπίτι με διάφορους φορείς και παρόχους υπηρεσιών. Μπορεί επίσης να θεωρηθεί ως μια απόπειρα για την οικοδόμηση ενός δικτύου έξυπνης πόλης, όπου κάθε έξυπνο σπίτι συνδέεται με τις διάφορες έξυπνες εγκαταστάσεις της πόλης. Για παράδειγμα, μπορεί να δημιουργηθεί μια υπηρεσία για την πρόγνωση καιρού, που χρησιμοποιείται από το Σύστημα Διαχείρισης Ενέργειας (EMS) για την εκτίμηση του ημερήσιου προφίλ κατανάλωσης.

2.2. ΠΑΡΑΚΟΛΟΥΘΗΣΗ ΚΑΤΑΝΑΛΩΣΗΣ ΕΝΕΡΓΕΙΑΣ (ENERGY CONSUMPTION MONITORING)

Σε αυτή την ενότητα παρουσιάζονται τα υπάρχοντα συστήματα που υπάρχουν για αυτό το σκοπό , μαζί με τυχόν περιορισμούς που μπορεί να έχουν.

Τα εργαλεία Μεγάλων Δεδομένων που εφαρμόζονται σε ενεργειακά προβλήματα μπορούν να επιφέρουν εξειδικευμένες λύσεις για την παρακολούθηση της ενεργειακής κατανάλωσης και για αυτό το λόγο υπερτερούν άλλων, πιο παραδοσιακών εργαλείων. Τα αποτελέσματα που προκύπτουν μπορούν να οδηγήσουν σε τεράστια οικονομικά οφέλη, λόγω αποδοτικότερης κατανάλωσης. Εκτός από τα Μεγάλα Δεδομένα, άλλες τεχνολογίες που αναμένεται να έχουν σημαντικό αντίκτυπο στην Ενεργειακή Απόδοση και Διαχείριση περιλαμβάνουν την έξυπνη μέτρηση, το Διαδίκτυο των Πραγμάτων (IoT) και το υπολογιστικό νέφος. Αυτές οι τεχνολογίες, ωστόσο, δημιουργούν ζητήματα που δεν πρέπει να αγνοήσουμε, όπως είναι η ιδιωτικότητα, η αβεβαιότητα και η ανακρίβεια [71] :

1. **Έξυπνη μέτρηση :** Αφορά τη συνεχή παρακολούθηση της κατανάλωσης ενέργειας με στόχο την καλύτερη κατανόηση των σταδίων κατανάλωσης, παραγωγής και μεταφοράς ενέργειας. Με την παρακολούθηση σε πραγματικό χρόνο, οι διαχειριστές ενέργειας και μπορούν να αναλύουν λεπτομερώς τα δεδομένα που συλλέγονται και να εντοπίζουν τάσεις και μοτίβα στη χρήση της ενέργειας, βοηθώντας στην αποδοτικότερη και οικονομικότερη κατανάλωσή της .

   Αυτό έχει τεράστια κοινωνικά και τεχνικά οφέλη. Σε κοινωνικό επίπεδο, η κατανόηση της κατανάλωσης ενέργειας μπορεί να συμβάλει στη μείωση των ενεργειακών δαπανών για τους καταναλωτές, προωθώντας παράλληλα πιο βιώσιμες πρακτικές. Οι καταναλωτές ενημερώνονται καλύτερα για τη χρήση της ενέργειας και μπορούν να προσαρμόσουν τη συμπεριφορά τους για να εξοικονομήσουν ενέργεια και χρήματα.

   Σε τεχνικό επίπεδο, η συνεχής παρακολούθηση επιτρέπει τον εντοπισμό προβλημάτων ή ανωμαλιών στο δίκτυο ενέργειας, όπως βλάβες στον εξοπλισμό ή απώλειες, βοηθώντας στην ταχύτερη αποκατάσταση και στη βελτίωση της αξιοπιστίας του δικτύου.
   Ωστόσο, παρά τα οφέλη, οι έξυπνοι μετρητές δημιουργούν ζητήματα ασφάλειας και ιδιωτικότητας, καθώς οι χρήστες είναι υποχρεωμένοι να μοιράζονται δεδομένα χρήσης [72] και από τις πληροφορίες της μέτρησης μπορεί να αποκαλυφθούν ιδιωτικές λεπτομέρειες ενός σπιτιού, όπως ο αριθμός των κατοίκων, οι καθημερινές τους ρουτίνες, οι ηλεκτρονικές τους συσκευές, κ.λπ. Λύση σε αυτό αποτελεί η ανωνυμοποίηση των δεδομένων [73] , ώστε να μην

ταυτοποιούνται οι μεμονωμένοι χρήστες κατά τη διάρκεια της διαδικασίας ανάλυσης. Οι χρήστες πρέπει ,ακόμη, να έχουν τη δυνατότητα να ρυθμίζουν το τι μοιράζονται με τις εταιρείες κοινής ωφέλειας , όπως η συχνότητα συλλογής δεδομένων, το είδος των δεδομένων που αποστέλλονται, καθώς και το επίπεδο λεπτομέρειας για αυτά.

Μια άλλη πρόκληση αποτελεί η κατανόηση του συνόλου των δεδομένων που είναι χρήσιμο για την απάντηση των ερωτημάτων που διευκολύνουν τη λήψη αποφάσεων, προκειμένου να διευκολυνθεί η απάντηση των ερωτημάτων.

2. **Big Data :** Η τεράστια ποσότητα δεδομένων που παράγονται από αισθητήρες , τα οποία είναι πλέον γνωστά ως Μεγάλα Δεδομένα (Big Data), μπορεί να αξιοποιηθεί για την αύξηση της ενεργειακής απόδοσης. Ωστόσο, αυτή η ποσότητα των δεδομένων είναι δύσκολο να διαχειριστεί από τις παραδοσιακές προσεγγίσεις ανάλυσης δεδομένων, και για αυτό το σκοπό απαιτούνται νέες υποδομές και εργαλεία. Η ετερογένεια των δεδομένων και η έλλειψη δομής τους κάνει επιτακτική την ανάγκη για συστήματα τα οποία καλούνται να φέρουν εις πέρας τη διαχείριση τεράστιων ποσοτήτων ετερογενών δεδομένων σε πραγματικό χρόνο, καθώς και να βρουν τρόπους αντιμετώπισης της σχετιζόμενης αβεβαιότητας.

   Τα Μεγάλα Δεδομένα αφορούν δεδομένα που είναι πολύ μεγάλα και πολύπλοκα για να αντιμετωπιστούν από παραδοσιακές βάσεις δεδομένων και περιγράφονται καλύτερα από τρία Vs [74]:
   - Όγκος (volume) : Αφορά την τεράστια ποσότητα των δεδομένων που παράγονται από τους αισθητήρες και τις λοιπές συσκευές και καλείται να διαχειριστεί το σύστημα.
   - Ποικιλία (variety) : Αφορά την ποικιλομορφία των δεδομένων, καθώς προέρχονται από διαφορετικές πηγές και μπορεί να είναι σε διαφορετικές μορφές , όπως δομημένα, μη δομημένα, πολυμέσα, κείμενα, κ.λπ.
   - Ταχύτητα (velocity): Η ταχύτητα με την οποία αυτά τα δεδομένα συλλέγονται και αποθηκεύονται ,αλλά και πόσο γρήγορα μπορούν οι χρήστες να αποκτήσουν πρόσβαση σε αυτά.

   Εταιρίες όπως οι Siemens και Teradata έχουν αρχίσει να συνεργάζονται σε λύσεις επιστήμης δεδομένων για τον έλεγχο και την παρακολούθηση της ενεργειακής τους υποδομής [75], ενώ άλλες νεοφυείς επιχειρήσεις ,όπως η Nest Labs προσέθετε αισθητήρες, υπολογιστική και τεχνολογία επικοινωνιών για να κάνει καθημερινά αντικείμενα πιο χρήσιμα.

   Ωστόσο, και σε αυτό το κομμάτι, υπάρχουν ανησυχίες σχετικά με την ιδιωτικότητα και τη χρήση που μπορεί να κάνουν οι εταιρείες στα δεδομένα που αντικατοπτρίζουν τη συμπεριφορά των χρηστών στην κατανάλωση ενέργειας.

3. **IoT :** Το Διαδίκτυο των πραγμάτων , όπως αναλύθηκε εκτενώς στο προηγούμενο κεφάλαιο , αποτελεί την τρέχουσα τάση σχετικά με τη διαχείριση των δεδομένων ενέργειας. Στον τομέα της ενεργειακής απόδοσης και διαχείρισης, οι έξυπνοι μετρητές και οι συσκευές συνδέονται στο Διαδίκτυο και είναι εξοπλισμένοι με νοημοσύνη, παράγοντας τεράστια ποσότητα δεδομένων.

   Με τη χρήση προηγμένων αλγορίθμων και μηχανικής μάθησης επιτρέπεται η συλλογή και ανάλυση δεδομένων σε πραγματικό χρόνο, βελτιώνοντας την ακρίβεια των μοντέλων διαχείρισης ενέργειας, καθώς και η ακριβής παρακολούθηση των ενεργειακών ροών, βασισμένη σε δεδομένα πραγματικού χρόνου. Με αυτό τον τρόπο επιτυγχάνεται η πρόβλεψη και η εύκολη διόρθωση , σε σχεδόν πραγματικό χρόνο ,της διανομή της ενέργειας ,αποφεύγοντας τα σφάλματα και προλαμβάνοντας την υπερφόρτωση.

   Οι συνδεδεμένα συσκευές μπορούν επίσης να εκτελούν εργασίες διαχείρισης ισχύος με μεγαλύτερη ακρίβεια και ταχύτερους χρόνους αντίδρασης από τα χειροκίνητα ή εξαρτημένα

από τον ανθρώπινο παράγοντα συστήματα, εξοικονομώντας έτσι ενέργεια, δίνοντας προτεραιότητα στη χρήση και θέτοντας πολιτικές για την αντίδραση σε διακοπές ρεύματος.

Το κύριο μειονέκτημα του Διαδικτύου των Πραγμάτων είναι , όπως αναλύθηκε και παραπάνω, ότι προκαλεί νέες ανησυχίες σχετικά με την απορρήτου των δεδομένων και την ασφάλειά τους από τρίτους.

4. **Cloud Computing :** Περιλαμβάνει τη συγκέντρωση των πόρων υπολογισμού, με στόχο τη μείωση των δαπανών υπολογισμού και την αύξηση της ευελιξίας και της αξιοπιστίας των συστημάτων. Αυτή η τεχνολογία επιτρέπει την αποθήκευση, επεξεργασία και ανάλυση μεγάλων ποσοτήτων δεδομένων σε απομακρυσμένα κέντρα δεδομένων, τα οποία μπορεί να ανήκουν είτε σε ενεργειακές εταιρείες είτε σε τρίτα μέρη. Η χρήση του cloud computing συμβάλλει στην εξοικονόμηση πόρων, καθώς οι εταιρείες μπορούν να πληρώνουν μόνο για τη χρήση των υπηρεσιών που χρειάζονται, χωρίς να απαιτείται η αγορά και συντήρηση δικού τους εξοπλισμού. Το cloud computing παρέχει επίσης τη δυνατότητα κλιμάκωσης των πόρων υπολογισμού ανάλογα με τις ανάγκες, επιτρέποντας στις εταιρείες να ανταποκρίνονται σε περιόδους μεγάλης ζήτησης χωρίς προβλήματα.

   Η περίπτωση της ιδιοκτησίας από τρίτους προσθέτει έναν ακόμα εμπλεκόμενο στη διαδικασία, γεγονός που προκαλεί ανησυχίες ιδιωτικότητας τόσο για τους χρήστες όσο και για την ενεργειακή εταιρεία, καθώς όλο και περισσότεροι έχουν πρόσβαση σε προσωπικά δεδομένα. Συνεπώς, δημιουργείται ο κίνδυνος της έκθεσης των δεδομένων σε μη εξουσιοδοτημένους χρήστες , αλλά και της χρήσης τους για άλλους σκοπούς ,χωρίς τη συγκατάθεση των ιδιοκτητών.

   Για την αντιμετώπιση αυτών των προκλήσεων, απαιτούνται ισχυρά μέτρα ασφαλείας και πολιτικές προστασίας δεδομένων, με τη διασφάλιση από τις εταιρίες ότι τα δεδομένα κρυπτογραφούνται κατά τη μεταφορά και την αποθήκευση, ενώ η πρόσβαση σε αυτά επιτρέπεται μόνο σε εξουσιοδοτημένα άτομα .

2.3. ΣΥΣΤΗΜΑΤΑ ΔΙΑΧΕΙΡΙΣΗΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ (DATABASE MENAGEMENT SYSTEMS-DBMS)

Παρακάτω παρουσιάζεται μια επισκόπηση των διαφορετικών επιλογών διαχείρισης βάσεων δεδομένων (database management systems-DBMS).

Όπως έχει γίνει μέχρι τώρα κατανοητό, το Internet of Things (IoT) είναι ένα δικτυακό παράδειγμα στο οποίο φυσικά, ψηφιακά και εικονικά αντικείμενα είναι εξοπλισμένα με λειτουργίες αναγνώρισης, ανίχνευσης, δικτύωσης και επεξεργασίας, ώστε να επικοινωνούν μεταξύ τους και με άλλες συσκευές και υπηρεσίες στο Διαδίκτυο για να εκτελούν τις απαιτούμενες από τους χρήστες εργασίες.

Η τεχνολογία IoT βασίζεται στη συλλογή ενός τεράστιου όγκου δεδομένων από διάφορες πηγές, όπως αισθητήρες, συσκευές παρακολούθησης και έξυπνα τηλέφωνα. Αυτού του είδους τα δεδομένα εκπέμπονται μέσω του δικτύου στην κατάλληλη πλατφόρμα νέφους για να διαχειριστούν από τις αντίστοιχες υπηρεσίες. Οι διακομιστές και τα κέντρα δεδομένων είναι υπεύθυνα για τη χειρισμό και την αποθήκευση όλων των δεδομένων που λαμβάνονται. Στη συνέχεια, τα δεδομένα αυτά επεξεργάζονται από κατάλληλη γλώσσα προγραμματισμού για την εκτέλεση των λειτουργιών χειρισμού. Οι γλώσσες προγραμματισμού στην πλευρά του διακομιστή είναι υπεύθυνες για πολλές εργασίες, όπως η επεξεργασία των συλλεγμένων δεδομένων, η αλληλεπίδραση με τους διακομιστές, η αλληλεπίδραση με τις βάσεις δεδομένων και οι λειτουργίες επεξεργασίας πάνω στις βάσεις δεδομένων [76] . Τελικά, αποθηκεύονται σε βάσεις δεδομένων, όπου στη συνέχεια επεξεργάζονται και αναλύονται με βάση τις ανάγκες και τους σκοπούς που απαιτούνται κάθε φορά.

Το λογισμικό που είναι υπεύθυνο για την αποθήκευση και διαχείριση των βάσεων δεδομένων ονομάζεται Σύστημα Διαχείρισης Βάσεων Δεδομένων (DBMS) .

Σε αυτό το πλαίσιο, θα μπορούσαμε να πούμε πως το IoT περιλαμβάνει τρία κύρια στοιχεία [77] :

1. τη **συλλογή δεδομένων** , η οποία στοχεύει στην αναγνώριση και την ανακάλυψη των συσκευών και των συστημάτων από τα οποία λαμβάνονται τα δεδομένα που αποθηκεύονται στο IoT. Αυτή η διαδικασία περιλαμβάνει τη χαρτογράφηση αισθητήρων και συσκευών που καταγράφουν διάφορες παραμέτρους, όπως θερμοκρασία, υγρασία, κατανάλωση ενέργειας, κίνηση και άλλα, προκειμένου να γίνει κατανοητό η δομή των δεδομένων που στέλνονται από τις διάφορες συσκευές. Η αποτελεσματική συλλογή δεδομένων είναι κρίσιμη για την ανάλυση και τη λήψη αποφάσεων, καθώς είναι η βάση για την ανάπτυξη έξυπνων εφαρμογών και υπηρεσιών.
2. το **σχεδιασμό συστήματος διαχείρισης δεδομένων** , το οποίο περιλαμβάνει το σχεδιασμό της αρχιτεκτονικής του συστήματος διαχείρισης δεδομένων και τον τρόπο αποθήκευσης και αρχειοθέτησης των δεδομένων σε αυτό. Αυτό το βήμα περιλαμβάνει την επιλογή της κατάλληλης τεχνολογίας βάσης δεδομένων, την ανάπτυξη των απαιτούμενων υποδομών και την εφαρμογή πρωτοκόλλων για την ασφάλεια και την προστασία των δεδομένων. Ο σχεδιασμός πρέπει να λαμβάνει υπόψη την κλίμακα των δεδομένων, την ανάγκη για γρήγορη πρόσβαση και ανάλυση, καθώς και την ευκολία συντήρησης και επεκτασιμότητας του συστήματος. Επιπλέον, πρέπει να διασφαλίζει την ελαχιστοποίηση των κινδύνων από απώλεια δεδομένων και να επιτρέπει την εύκολη ανάκτηση και διαχείριση των δεδομένων.
3. την **επεξεργασία των δεδομένων** , η οποία ασχολείται με την πραγματική πρόσβαση των χρηστών στους αποθηκευτικούς χώρους δεδομένων. Αυτή η φάση περιλαμβάνει την ανάλυση των δεδομένων που έχουν συλλεχθεί και αποθηκευτεί, χρησιμοποιώντας διάφορα εργαλεία και τεχνικές ανάλυσης, όπως η μηχανική μάθηση, η ανάλυση μεγάλων δεδομένων και η εξόρυξη δεδομένων. Οι χρήστες πρέπει να έχουν τη δυνατότητα να προσπελάζουν τα δεδομένα με ασφαλή και αποτελεσματικό τρόπο, ώστε να μπορούν να εξάγουν χρήσιμες πληροφορίες και να λαμβάνουν τεκμηριωμένες αποφάσεις. Η επεξεργασία των δεδομένων περιλαμβάνει επίσης τη δημιουργία αναφορών και την οπτικοποίηση των αποτελεσμάτων, διασφαλίζοντας την προστασία των προσωπικών δεδομένων και την τήρηση των κανονισμών ιδιωτικότητας και ασφάλειας.

Λόγω της ποικιλίας, της ετερογένειας και του μεγάλου όγκου δεδομένων που παράγονται από τις συσκευές του IoT, η χρήση παραδοσιακών συστημάτων διαχείρισης βάσεων δεδομένων δεν είναι γενικά κατάλληλη για τους παρακάτω λόγους :

1. **Συλλογή Δεδομένων - Ετερογένεια Πηγών:** Στα παραδοσιακά συστήματα βάσεων δεδομένων, η μάζα των δεδομένων συλλέγεται από πεπερασμένες και προκαθορισμένες πηγές και στη συνέχεια αποθηκεύεται σύμφωνα με αυστηρούς κανόνες σχέσεων. Αντίθετα, στο IoT, υπάρχει ένας μεγάλος και αυξανόμενος αριθμός πηγών δεδομένων με αισθητήρες, ενσωματωμένα συστήματα κ.λπ.

   Τα δεδομένα έχουν τεράστιο όγκο και είναι ποικίλας φύσης, καθώς οι μετρήσεις προκύπτουν σε πραγματικό χρόνο από εκατομμύρια γεωγραφικά διασκορπισμένες συσκευές που περιοδικά στέλνουν παρατηρήσεις σχετικά με συγκεκριμένα φαινόμενα ή αναφέρουν την εμφάνιση βλαβών ή ανωμαλιών . Οι περιοδικές παρατηρήσεις είναι οι πιο απαιτητικές ως προς την επικοινωνία και την αποθήκευση λόγω της συνεχούς φύσης τους, ενώ λαμβάνεται υπόψη και ο περιορισμός του χρόνου , καθώς ορισμένα γεγονότα απαιτούν επείγουσα ανάγκη αντίδρασης [77].

2. **Αποθήκευση και Ανάλυση Δεδομένων :** Τα παραδοσιακά συστήματα διαχείρισης δεδομένων χειρίζονται την αποθήκευση, την ανάκτηση και την ενημέρωση βασικών στοιχείων δεδομένων, αρχείων και φακέλων. Στο πλαίσιο του IoT, τα συστήματα διαχείρισης δεδομένων πρέπει να λαμβάνουν υπόψη τα διαδικτυακά δεδομένα παρέχοντας παράλληλα δυνατότητες αποθήκευσης, καταγραφής και ελέγχου για διαδικτυακή ανάλυση. Σε αντίθεση με τις περιστασιακές ερωτήσεις και ενημερώσεις που αποστέλλονται σε παραδοσιακά συστήματα

διαχείρισης βάσεων δεδομένων (DBMS), η ροή των δεδομένων στο IoT είναι συνεχόμενη από πλήθος αντικειμένων σε αποθήκες δεδομένων και οι ερωτήσεις είναι πιο συχνές και με περισσότερες απαιτήσεις.

Η χρήση της τεχνολογίας IoT δημιουργεί έναν μεγάλο όγκο διαφορετικών δεδομένων όπως κείμενα, αριθμούς, ήχους, βίντεο και εικόνες. Αυτοί οι τύποι δεδομένων πρέπει να μεταφέρονται, επεξεργάζονται και αποθηκεύονται σε έναν διακομιστή νέφους. Επίσης, πρέπει να είναι δυνατή η ανάκτηση πληροφοριών και η ενημέρωσή τους μετά από αντίστοιχο αίτημα του χρήστη. Η αποθήκευση και διαχείριση ενός μεγάλου όγκου δεδομένων IoT με αποδοτικό τρόπο είναι μία από τις μεγάλες προκλήσεις. Για την αντιμετώπιση αυτής της πρόκλησης απαιτείται ένα ευέλικτο Σύστημα Διαχείρισης Βάσεων Δεδομένων (DBMS) [78].

3. **Σχήμα Βάσης Δεδομένων :** Στο Διαδίκτυο των Πραγμάτων το αυστηρό σχήμα της βάσης δεδομένων που υπάρχει στα παραδοσιακά συστήματα αποθήκευσης πρέπει να αντικατασταθεί με μια πιο ευέλικτη δομή αποθήκευσης , η οποία αφενός καλύπτει και μη δομημένα δεδομένα και αφετέρου προσαρμόζεται στους διαφορετικούς τύπους δεδομένων και στα πολύπλοκα ερωτήματα ανάκτησής τους. Τα σύγχρονα DBMS πρέπει να μπορούν να διαχειριστούν δεδομένα που δεν είναι δομημένα, όπως εικόνες, βίντεο και αρχεία καταγραφής, που συχνά προέρχονται από IoT συσκευές.

4. **Μεταδεδομένα (metadata) :** Στις τεχνολογίες IoT υπάρχουν μεταδεδομένα που περιγράφουν τις συσκευές, εκτός από τα ίδια τα δεδομένα που δημιουργούνται από αυτές. Η αναγνώριση του αντικειμένου με βάση το όνομα που του έχει δοθεί, η τοποθεσία που βρίσκεται μέσα στο σπίτι, οι υπηρεσίες που υποστηρίζει είναι παραδείγματα τέτοιων δεδομένων. Εκτός από αυτά, υπάρχουν και δεδομένα που χαρακτηρίζουν και τις υπόλοιπες οντότητες που εμπλέκονται στο σύστημα ,όπως πληροφορίες για το σπίτι, για τα δωμάτια, για τις κατηγορίες χρηστών. Τα δεδομένα αυτά χρειάζονται επίσης μια βάση , η οποία μπορεί να έχει και σταθερό σχήμα , εφόσον γνωρίζουμε εκ των προτέρων τα χαρακτηριστικά και τη δομή τους, η οποία θα επικοινωνεί με το σύστημα στο οποίο αποθηκεύονται οι μετρήσεις. Επομένως, η επικοινωνία, η αποθήκευση και η επεξεργασία είναι καθοριστικοί παράγοντες για τον σχεδιασμό ενός συστήματος διαχείρισης δεδομένων για το IoT [77] .

Όπως έγινε κατανοητό από τα παραπάνω, τα DBMS για το IoT αντιμετωπίζουν την πρόκληση της αποθήκευσης και του χειρισμού ενός τεράστιου όγκου διαφορετικών , ως προς τη δομή και τα χαρακτηριστικά τους, δεδομένων .

Τα DBMS μπορούν να κατηγοριοποιηθούν σε δύο βασικούς τύπους: τα Συστήματα Διαχείρισης Βάσεων Δεδομένων Σχεσιακού Μοντέλου (RDBMS) και τα Συστήματα Διαχείρισης Βάσεων Δεδομένων Μη Σχεσιακού Μοντέλου (NRDBMS).

1. **RDBMS :** Οι Σχεσιακές DBMS όπως η MySQL, η Oracle και η PostgreSQL είναι γνωστές ως βάσεις δεδομένων SQL και είναι οι πιο κοινές και ευρέως χρησιμοποιούμενες βάσεις δεδομένων.

   Χρησιμοποιούν τη γλώσσα Δομημένων Ερωτημάτων (SQL) ως γλώσσα προγραμματισμού των εφαρμογών τους [78].

   Τα σχεσιακά συστήματα αφορούν δεδομένα που ακολουθούν την ίδια δομή και τα οποία αποθηκεύονται σε πίνακες (σχέσεις), σύμφωνα με ένα προκαθορισμένο σχήμα. Η διαδικασία σχεδιασμού περιλαμβάνει φάσεις όπως η κανονικοποίηση και η υλοποίηση για να ικανοποιήσει τους κατάλληλους περιορισμούς για μια συγκεκριμένη εφαρμογή. Συνεπώς, οποιαδήποτε αλλαγή στη δομή των δεδομένων θα διαταράξει το σύστημα , το οποίο είναι δύσκολο να προσαρμοστεί και να ρυθμιστεί εκ νέου, καθώς πρέπει να αλλάξει από την αρχή το σχήμα. Η χρήση δικτύων IoT οδήγησε στη δημιουργία μεγάλου όγκου ανομοιόμορφων

δεδομένων και οι παραδοσιακές SQL DBMS δεν σχεδιάστηκαν για να λύσουν αυτά τα προβλήματα.

Ένας άλλος χαρακτηριστικό της τεχνολογίας των σχεσιακών DBMS αφορά την ακεραιότητα των δεδομένων . Λαμβάνοντας υπόψη ότι το DBMS αναμένεται να εξυπηρετεί πολλούς χρήστες παράλληλα σε πραγματικό χρόνο, αντιλαμβανόμαστε πως η ταυτόχρονη ενημέρωση και ανάκτηση των ίδιων δεδομένων από διαφορετικά άτομα μπορεί να δημιουργήσει προβλήματα αξιοπιστίας και αποδοτικότητας. Η ταυτόχρονη επεξεργασία δεδομένων πραγματοποιείται κυρίως με πρωτόκολλα βασισμένα σε κλειδώματα, κατά τα οποία μόνο ένας χρήστης κάθε φορά μπορεί να έχει πρόσβαση σε ένα σύνολο δεδομένων, ενώ για μη ολοκληρωμένες διεργασίες εφαρμόζεται η λειτουργία της αναίρεσης [79].

2. **NRDBMS :** Οι Μη Σχεσιακές DBMS ,όπως η MongoDB , η Cassandra και η Hbase , είναι γνωστές ως NoSQL βάσεις δεδομένων, καθώς έχουν τη δυνατότητα να αποθηκεύουν και να επεξεργάζονται αταξινόμητα ή ανομοιογενή δεδομένα.

   Για το σκοπό αυτό το σχήμα τους δεν είναι προκαθορισμένο, αλλά δυναμικό , προκειμένου να αλλάζει και να προσαρμόζεται, έτσι ώστε να εξυπηρετεί όσο το δυνατόν περισσότερες δομές δεδομένων, καθώς αυτά είναι ανομοιόμορφα. Όλοι οι τύποι δεδομένων αποθηκεύονται και διαχειρίζονται με πολλούς τρόπους, όπως είναι η οριοθέτηση από στήλες, τα ζεύγη κλειδί-τιμή, οι γράφοι και τα έγγραφα.

   Είναι ιδανικές για συστήματα IoT , στα οποία τα δεδομένα που παράγονται είναι πολύ αταξινόμητα και ομοιογενή και χρειάζονται ευέλικτες τεχνικές επεξεργασίας σε σύγκριση με αυτές που χρησιμοποιούνται στα συστήματα σχεσιακών βάσεων δεδομένων. Σήμερα, οι μεγαλύτερες εταιρείες στον τομέα του διαδικτύου όπως η Google, η Amazon, η Facebook, η Twitter κ.λπ. χρησιμοποιούν λύσεις βασισμένες σε NoSQL, καθώς παρέχουν τη δυνατότητα της αλλαγής και της προσαρμογής όλων των εσωτερικών συστημάτων, από την αποθήκευση και επεξεργασία δεδομένων έως την ακεραιότητα του συστήματος, βάσει κινήτρων και των συνθηκών κάθε φορά [79] .

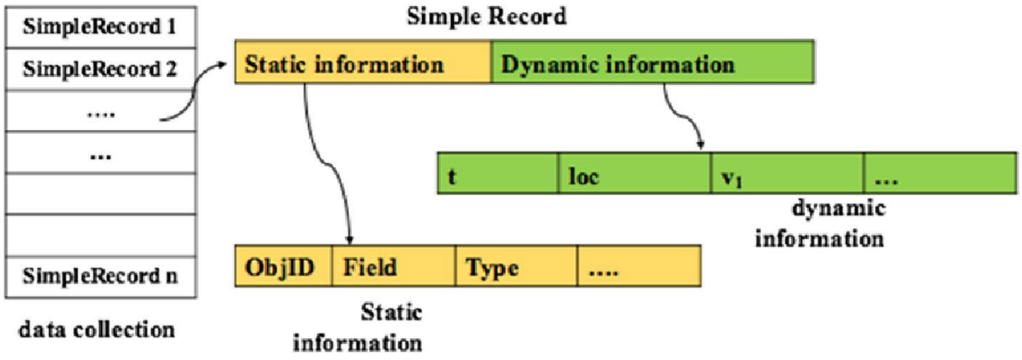
   Η MongoDB είναι μια από τις πιο γνωστές βάσεις δεδομένων μη σχεσιακής μορφής . Είναι προσανατολισμένη σε έγγραφα και μπορεί να χρησιμοποιηθεί για τη διάθεση και αποθήκευση μεγάλων δυαδικών αρχείων όπως βίντεο και εικόνες. Αποθηκεύει τα δεδομένα ως έγγραφα σε μια δυαδική αναπαράσταση που ονομάζεται αντικείμενα BSON (Binary JSON), τα οποία είναι δυαδικά κωδικοποιημένα αντικείμενα JSON. Σχετικές πληροφορίες αποθηκεύονται μαζί για γρήγορη πρόσβαση στο ερώτημα μέσω της γλώσσας ερωτημάτων του MongoDB. Τα έγγραφα μπορούν να οργανωθούν σε "συλλογές" και τα πεδία μπορεί να διαφέρουν από έγγραφο σε έγγραφο, ενώ δεν χρειάζεται να δηλωθεί η δομή τους στο σύστημα , αφού αυτά είναι αυτοπεριγραφικά. Αν χρειαστεί να προστεθεί ένα νέο πεδίο σε ένα έγγραφο, τότε το πεδίο μπορεί να δημιουργηθεί χωρίς να επηρεάσει όλα τα άλλα έγγραφα στη συλλογή, χωρίς να ενημερωθεί ένα κεντρικό κατάλογο συστήματος και χωρίς να ληφθεί το σύστημα εκτός λειτουργίας. Έτσι, έχει καλύτερη απόδοση από άλλες βάσεις δεδομένων όσον αφορά τη χρήση πόρων και τη μακροπρόθεσμη αποθήκευση για την εργασία με μεγάλα ποσά δεδομένων. Επιπλέον, η MongoDB υποστηρίζει το δεικτοδότηση πάνω σε ενσωματωμένα αντικείμενα και πίνακες. Αλληλοεπιδρά αποτελεσματικά με αποθήκευση μνήμης, πολύπλοκα δεδομένα και δυναμικά ερωτήματα σε σύγκριση με άλλες NoSQL βάσεις, παρέχοντας ευελιξία για να λειτουργεί σε περίπτωση επέκτασης του υλικού.

   Ο Li και οι συνεργάτες του προτείνουν ένα κεντρικό σύστημα διαχείρισης αποθήκευσης δεδομένων για μαζικά και ετερογενή δεδομένα του IoT στο [80,81]. Αυτή η λύση χρησιμοποιεί μια αρχιτεκτονική που ονομάζεται IOTMDB και βασίζεται σε NoSQL, καθώς τα συστήματα NoSQL εφαρμόζονται επιτυχώς για την επίλυση των προβλημάτων αποθήκευσης και διαχείρισης των μαζικών δεδομένων του IoT. Τα δεδομένα που παρέχονται από εφαρμογές συλλέγονται και αποστέλλονται στο σύστημα IOTMDB. Σε αυτήν την αρχιτεκτονική, τα δεδομένα λαμβάνονται από έναν κόμβο λήψης δεδομένων και αποθηκεύονται σε έναν κόμβο .Επιπλέον, προτείνουν στρατηγικές αποθήκευσης για την έκφραση και οργάνωση των

δεδομένων του IoT ομοιόμορφα. Στην προσέγγισή τους, τα δεδομένα των διαφόρων αντικειμένων εφαρμογής αναπαρίστανται στο σύστημα IOTMDB στη μορφή μιας συλλογής SampleRecords. Το SampleRecord είναι η βασική μονάδα των δεδομένων που αποθηκεύονται στο IOTMDB και αποτελείται από ένα σύνολο SampleElement. Το SampleElement είναι η μικρότερη μονάδα του IOTMDB και ορίζεται ως εξής :

<Sample Element>=<key,value> [80]

Η παρακάτω εικόνα δείχνει την έκφραση δεδομένων στο IOTMDB και παρουσιάζει ότι υπάρχουν στατικά και δυναμικά μέρη πληροφορίας. Η στατική πληροφορία είναι η αμετάβλητη πληροφορία σχετικά με το αντικείμενο, όπως το αναγνωριστικό αντικειμένου (object ID), το πεδίο στο οποίο ανήκει το αντικείμενο, κ.λπ. Η δυναμική πληροφορία είναι οι τιμές δειγματοληψίας που περιγράφουν την αλλαγή κατάστασης του αντικειμένου, όπως ο χρόνος δειγματοληψίας, η τοποθεσία, η ταχύτητα κλπ.



Εικόνα 2.4 : Έκφραση δεδομένων στο IOTMDB [80]

Συνεπώς, τα μη σχεσιακά Συστήματα Διαχείρισης Βάσεων Δεδομένων προτείνονται για να επιλύσουν τους περιορισμούς των παραδοσιακών RDBMS στο κομμάτι των IoT, καθώς αναμένεται να είναι εύκολα κατανεμημένα με υψηλή επεκτασιμότητα και διαθεσιμότητα [78].

Ένας τύπος βάσεων δεδομένων , οποίος έχει γίνει ιδιαίτερα δημοφιλής στο Διαδίκτυο των Πραγμάτων είναι οι **Βάσεις Δεδομένων Χρονοσειρών (TSDBs)** , όπως οι InfluxDB και TimescaleDB , και διαθέτει κύριες αρχιτεκτονικές ιδιότητες που την κάνουν πολύ διαφορετική από άλλες βάσεις δεδομένων.

Είναι Σχεσιακές Βάσεις Δεδομένων, καθώς αποθηκεύουν δομημένα δεδομένα, αλλά με πιο εύκολο indexing στο χρόνο. Αυτό σημαίνει ότι:

1. Επιτρέπουν την εύκολη και άμεση πρόσβαση σε δεδομένα που αφορούν συγκεκριμένες χρονικές στιγμές ή περιόδους, αφού κάθε εγγραφή έχει μια χρονική σφραγίδα (timestamp).
2. Επιτρέπουν γρήγορη αναζήτηση και πρόσβαση σε χρονικά διαστήματα με τη χρήση βελτιστοποιημένων δομών δεδομένων, όπως B-trees , καθώς είναι σχεδιασμένες για να αποθηκεύουν και να ανακτούν δεδομένα με βάση τη χρονική τους σφραγίδα.
3. Επιτρέπει στους χρήστες να κάνουν γρήγορα και αποτελεσματικά πολύπλοκες αναλύσεις σε μεγάλες ποσότητες δεδομένων , καθώς προσφέρουν ενσωματωμένες συναρτήσεις ,όπως όπως moving averages και aggregations.
4. Γρήγορη ανάκτηση δεδομένων , αφού χρησιμοποιούν τεχνικές συμπίεσης δεδομένων και αποθήκευσης που είναι ιδιαίτερα αποδοτικές για δεδομένα χρονοσειρών.

Το indexing αυτό είναι ιδιαίτερα χρήσιμο, καθώς συνήθως τίθενται ερωτήματα σχετικά με τα δεδομένα με την πάροδο του χρόνου, όπως για παράδειγμα για αυτά που έχουν συγκεντρωθεί για

ένα συγκεκριμένο χρονικό διάστημα. Αυτή είναι και η βασική διαφορά των δεδομένων χρονοσειρών από τα κανονικά δεδομένα.

Αυτός ο τύπος βάσεων είναι σχεδιασμένος για τη διαχείριση δεδομένων χρονοσειρών ή δεδομένων με χρονική σήμανση και έχει τη δυνατότητα να διαχειρίζεται μετρήσεις, γεγονότα ή καταγραφές που αλλάζουν, παρακολουθούνται και συγκεντρώνονται με την πάροδο του χρόνου. Συγκεκριμένα, διαθέτει ιδιότητες που περιλαμβάνουν την αποθήκευση και συμπίεση δεδομένων που έχουν χρονοσήμανση, τη διαχείριση της διάρκειας ζωής των δεδομένων, τη συνοπτική ανάλυση των δεδομένων, την ικανότητά τους να επεξεργάζονται μεγάλους όγκους δεδομένων που εξαρτώνται από χρονοσειρές [82].

Οι TSDBs έγιναν ακόμη πιο δημοφιλείς με την ανάπτυξη του Διαδικτύου των Πραγμάτων, επειδή μπορούν να χρησιμοποιηθούν για την αποθήκευση των μετρήσεων αισθητήρων και των δεδομένων που προέρχονται από τις διάφορες συσκευές του σπιτιού. Αυτό εξυπηρετεί ενέργειες όπως η εξαγωγή στατιστικών και συμπερασμάτων με βάση τη χρονική στιγμή που έγιναν οι μετρήσεις, όπως πχ η εβδομαδιαία κατανάλωση ενέργειας, ενώ βοηθάει και στην οπτικοποίησή της χρονικής εξέλιξής τους [83].

Μια άλλη τεχνική που αξιοποιείται για τη διαχείριση των δεδομένων του IoT είναι το **Σύστημα Κατανεμημένων Αρχείων του Hadoop (HDFS)** , το οποίο είναι το κύριο σύστημα αποθήκευσης που χρησιμοποιείται από τις εφαρμογές Hadoop. καθώς παρέχει ένα μέσο για τη διαχείριση και την υποστήριξη μεγάλων δεδομένων. Το HDFS είναι ένα υποσύστημα αρχείων του Hadoop και αναφέρεται σε ένα κατανεμημένο σύστημα αρχείων με υψηλή ανοχή σε σφάλματα που υποστηρίζει το σχεδιασμό ενός αλγόριθμου βελτιστοποίησης πρόσβασης και αποθήκευσης δεδομένων στο cloud computing [84].

Χρησιμοποιεί έναν κύριο NameNode , ο οποίος γνωρίζει τα πάντα για τα δεδομένα και ελέγχει την πρόσβαση στους κόμβους, και πολλούς άλλους DataNodes οι οποίοι μπορούν να συνεργαστούν μεταξύ τους μεταφέροντας δεδομένα και είναι σε συνεχή επικοινωνία με τον NameNode .Τα δεδομένα μου λαμβάνει το σύστημα διασπώνται σε μπλοκ και κατανέμονται στους DataNodes για αποθήκευση. Για να μειωθούν οι πιθανότητες απώλειας δεδομένων, το σύστημα αρχείων χρησιμοποιεί αντιγραφή δεδομένων για να εξασφαλίσει ότι το μπλοκ αποθηκεύεται πολλαπλές φορές .Αυτό αποτελεί ένα σύστημα αντιγράφου ασφαλείας σε περίπτωση απώλειας δεδομένων. Τα μπλοκ μπορούν επίσης να αναπαραχθούν σε διάφορους κόμβους, γεγονός που επιτρέπει την αποδοτική παράλληλη επεξεργασία. Όταν ένας DataNode δεν λειτουργεί σωστά, αφαιρείται και ο NameNode μπορεί αυτόματα να επανακατανείμει τις εργασίες του σε έναν άλλο κόμβο και αυτό δεν οδηγεί σε καμία απώλεια πληροφορίας , αφού τα δεδομένα είναι αναπαραγμένα σε διάφορους DataNodes [85] .

Το HDFS είναι υψηλής ευαισθησίας και μπορεί να ανιχνεύει γρήγορα τυχόν βλάβες. Μπορεί επίσης να προσαρμοστεί όπου απαιτείται και να προστεθούν ή να αφαιρεθούν κόμβοι ανάλογα με τις ανάγκες και την χωρητικότητα του διακομιστή , κάτι που είναι χρήσιμο όταν υπάρχει μεγάλη έξαρση δεδομένων. Αυτό είναι ιδιαίτερα χρήσιμο για τις εφαρμογές IoT , οι οποίες χαρακτηρίζονται από μεγάλους όγκους δεδομένων [85] .

Συμπερασματικά, επειδή τα δεδομένα που προκύπτουν από τις συσκευές IoT έχουν μεγάλο όγκο και αυξάνονται με γρήγορους ρυθμούς ,οι λύσεις αποθήκευσης δεδομένων πρέπει όχι μόνο να είναι σε θέση να αποθηκεύουν αποτελεσματικά τα μαζικά δεδομένα ,αλλά και να υποστηρίζουν την επεκτασιμότητα. Επιπλέον, τα δεδομένα που συλλέγονται προέρχονται από διαφορετικές πηγές και μπορεί να έχουν ή όχι συγκεκριμένη δομή.

Ο βασικός στόχος ενός μοντέλου διαχείρισης βάσεων δεδομένων IoT είναι η συνδυασμένη χρήση πολλαπλών βάσεων δεδομένων, τόσο για την αντιστοίχιση αντικειμένων του πραγματικού κόσμου σε οντότητες στις βάσεις δεδομένων , όσο και για την αποθήκευση των μετρήσεων που προκύπτουν από τις διάφορες συσκευές.

ΚΕΦΑΛΑΙΟ 3

3.ΜΕΘΟΔΟΛΟΓΙΑ

3.1. ΑΠΑΙΤΗΣΕΙΣ ΚΑΙ ΑΝΑΛΥΣΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

3.1.1. ΛΕΙΤΟΥΡΓΙΚΕΣ ΚΑΙ ΜΗ ΛΕΙΤΟΥΡΓΙΚΕΣ ΑΠΑΙΤΗΣΕΙΣ

Κατά τον σχεδιασμό του συστήματός μας είναι αναγκαίο να ληφθούν υπόψη δύο ειδών απαιτήσεις: οι λειτουργικές και οι μη λειτουργικές.

Οι πρώτες αφορούν το πώς πρέπει να συμπεριφέρεται το σύστημα , προκειμένου να ανταποκρίνεται στις ανάγκες και στις προσδοκίες του χρήστη. Μπορούν να θεωρηθούν χαρακτηριστικά που εντοπίζει και ο ίδιος κατά την αλληλεπίδρασή του με το σύστημά μας. Σύμφωνα με αυτές, πρέπει να φροντίσουμε έτσι ώστε ο κάθε χρήστης να έχει πρόσβαση σε ένα app ,το οποίο συνδέεται με τη βάση μας , και μέσω μια διεπαφής χρήστη (REST API) να μπορεί να εκτελέσει τις εξής ενέργειες:

- Εγγραφή του στο σύστημα και προσθήκη των στοιχείων του
- Login χρησιμοποιώντας το όνομά του και τον κωδικό πρόσβασής του, καθώς και logout
- Update τόσο των στοιχείων του όσο και των στοιχείων των οντοτήτων που σχετίζονται με αυτόν
- Delete για όποια πληροφορία δεν χρειάζεται
- Πρόσβαση σε πληροφορίες που αφορούν τα σπίτια του, τις συσκευές του κλπ.
- Προσθήκη νέου σπιτιού ,συσκευής κλπ
- Ενημέρωση για τις μετρήσεις ανά διαστήματα και για την ενεργειακή του κατανάλωση.

Από την άλλη , οι μη λειτουργικές απαιτήσεις είναι περιορισμοί που επιβάλλονται στο σύστημα και καθορίζουν τα ποιοτικά του χαρακτηριστικά.

1. **Ασφάλεια των δεδομένων και ιδιωτικότητα :** Η συλλογή και η μετάδοση ευαίσθητων δεδομένων μεταξύ των συσκευών του σπιτιού δημιουργεί κινδύνους σχετικά με την προστασία τους. Είναι αναγκαία , λοιπόν, η ύπαρξη αξιόπιστων μέτρων ασφαλείας για την φύλαξη των προσωπικών πληροφοριών από άτομα και υπηρεσίες που δεν είναι εξουσιοδοτημένα να έχουν πρόσβαση σε αυτά [86].

2. **Διαλειτουργικότητα:** Οι συσκευές και τα συστήματα IoT πρέπει να είναι συμβατά και τυποποιημένα για να διασφαλιστεί η άψογη ανταλλαγή δεδομένων και η ολοκλήρωσή τους. Για την επίτευξη αυτού του σκοπού χρησιμοποιούνται συνήθως κοινά πρωτόκολλα και πρότυπα επικοινωνίας [86].

3. **Επεκτασιμότητα:** οι νέες δυνατότητες, συσκευές και λογισμικό πρέπει να είναι εύκολο να προστεθούν και να συνεργαστούν με όλες τις υπάρχουσες [87].

4. **Ετερογένεια:** διαφοροποίηση των πιθανών εφαρμογών και συστημάτων που πρέπει να κατασκευαστούν για να εξυπηρετούν ανθρώπους με διάφορες ανάγκες, να υποστηρίζουν ποικιλία δομών δεδομένων και να φιλοξενούν διαφορετικές λειτουργίες [87].

5. **Ευφυΐα:** όλες οι συσκευές και το λογισμικό πρέπει να υποστηρίζουν τους οικιακούς χρήστες προτείνοντας αναγκαίες ενέργειες και ακόμη και προβλέποντας τις ανάγκες των ιδιοκτητών [87].

6. **Ενεργειακή Αποδοτικότητα:** οι εφαρμογές του έξυπνου σπιτιού και πρέπει να είναι επωφελείς για την εξοικονόμηση ενέργειας και συνεπώς για την υποστήριξη του περιβάλλοντος [87].

Παρατηρώντας τις παραπάνω απαιτήσεις καταλαβαίνουμε πως η τεχνολογία που εφευρέθηκε για το έξυπνο σπίτι πρέπει να έχει τον άνθρωπο στο επίκεντρο, υλοποιώντας τις ανάγκες των χρηστών μέσω εφαρμοσμένου λογισμικού και συσκευών.

3.2. ΤΕΧΝΟΛΟΓΙΕΣ

3.2.1. ΕΡΓΑΛΕΙΑ,ΠΛΑΤΦΟΡΜΕΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΕΣ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ

Για την αποθήκευση των πληροφοριών και των χαρακτηριστικών κάθε οντότητας έχουμε σχεδιάσει και υλοποιήσει μια PostgreSQL βάση, καθώς γνωρίζουμε από πριν τη δομή των δεδομένων και άρα μπορούμε να χρησιμοποιήσουμε μια σχεσιακή βάση για την αποθήκευσή τους.

Η βάση αυτή έχει στηθεί στο Docker.

Όπως αναφέρθηκε και παραπάνω, η τεχνολογία PostgreSQL ,ως σχεσιακή βάση, δεν είναι κατάλληλη για τα δεδομένα που προέρχονται από τις συσκευές IoT , λόγω της ανάγκης των δεδομένων για χρονική σήμανση. Για αυτόν τον σκοπό έχει σχεδιαστεί μια timeseries database, και συγκεκριμένα μια Timescale βάση, της οποίας ,όμως, η υλοποίηση δεν είναι στα πλαίσια αυτής της διπλωματικής. Περισσότερες πληροφορίες για αυτό το είδος βάσεων έχουν αναφερθεί παραπάνω και θα αναλυθούν και στο επόμενο κεφάλαιο.

Για το σχεδιασμό των Διαγραμμάτων Οντοτήτων Συσχετίσεων χρησιμοποιήσαμε εργαλεία όπως το Lucidchart και το draw.io.

Για τη δημιουργία των Tables και των Queries στη βάση έχουμε χρησιμοποιήσει την GORM, η οποία είναι μια δημοφιλής βιβλιοθήκη της γλώσσας Go και αναλύεται στο Κεφάλαιο 4.

Τέλος, η δημιουργία endpoints , μέσω ενός API, πραγματοποιήθηκε με το Gin Framework της Go, ενώ οι δοκιμές έγιναν με το εργαλείο Postman, η χρήση του οποίο αναλύεται στο Κεφάλαιο 5.

3.2.1.1 ΣΧΕΣΙΑΚΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Η PostgreSQL ανήκει στις σχεσιακές βάσεις δεδομένων.

Οι σχεσιακές βάσεις δεδομένων βασίζονται στο σχεσιακό μοντέλο, το οποίο είναι ένας κατανοητός και απλός τρόπος αναπαράστασης των δεδομένων σε πίνακες. Σε μια σχεσιακή βάση, κάθε γραμμή στον πίνακα είναι μια εγγραφή με ένα μοναδικό αναγνωριστικό που ονομάζεται κλειδί. Οι στήλες του πίνακα περιέχουν χαρακτηριστικά των δεδομένων, και κάθε εγγραφή συνήθως έχει μια τιμή για κάθε χαρακτηριστικό [88].

Το σχεσιακό μοντέλο δεδομένων παρέχει έναν τυποποιημένο τρόπο αναπαράστασης και ανάκτησης δεδομένων και κύρια δύναμή του αποτελεί η χρήση πινάκων, οι οποίοι είναι ένας κατανοητός, αποδοτικός και ευέλικτος τρόπος αποθήκευσης και πρόσβασης σε δομημένες πληροφορίες.

Ένα ακόμα βασικό στοιχείο του σχεσιακού μοντέλου είναι η χρήση της γλώσσας δομημένων ερωτημάτων (SQL) για την εγγραφή και την αναζήτηση δεδομένων σε μια βάση. Η SQL χρησιμοποιείται ευρέως ως η γλώσσα για ερωτήματα βάσεων δεδομένων, καθώς είναι βασισμένη στη σχεσιακή άλγεβρα και διευκολύνει τη βελτίωση της απόδοσης όλων των ερωτημάτων .

Οι κρίσιμες ιδιότητες που χαρακτηρίζουν τις σχεσιακές βάσεις δεδομένων αναφέρονται ως ACID και είναι η ατομικότητα, η συνέπεια, η απομόνωση και η ανθεκτικότητα [88] .

1. Η ατομικότητα ορίζει όλα τα στοιχεία που συνθέτουν μια πλήρη αλλαγή στα δεδομένα της βάσης
2. Η συνέπεια ορίζει τους κανόνες για τη διατήρηση των σημείων δεδομένων σε σωστή κατάσταση μετά από μια αλλαγή
3. Η απομόνωση κρατά την αλλαγή αόρατη στους άλλους μέχρι να δεσμευτεί, για να αποφευχθεί σύγχυση.
4. Η ανθεκτικότητα διασφαλίζει ότι οι αλλαγές δεδομένων γίνονται μόνιμες μόλις η ενέργεια ολοκληρωθεί.

Η χρήση της Σχεσιακής Βάσης Δεδομένων για το σκοπό μας, εφόσον γνωρίζουμε τη δομή των δεδομένων μας ,επιλέχθηκε λόγω των παρακάτω πλεονεκτημάτων [88] :

1. **Συνέπεια Δεδομένων :** Το σχεσιακό μοντέλο είναι το καλύτερο στη διατήρηση της συνέπειας των δεδομένων μεταξύ εφαρμογών και αντιγράφων ,τα οποία ονομάζονται instances, της βάσης δεδομένων.Το χαρακτηριστικό αυτό γίνεται αντιληπτό αν εξετάσουμε την περίπτωση των συναλλαγών. Για παράδειγμα, όταν ένας πελάτης καταθέτει χρήματα σε ένα ATM και στη συνέχεια ελέγχει το υπόλοιπο του λογαριασμού του σε ένα κινητό τηλέφωνο, ο πελάτης αναμένει να δει αυτή την κατάθεση να αντικατοπτρίζεται αμέσως στο ενημερωμένο υπόλοιπο λογαριασμού. Οι σχεσιακές βάσεις δεδομένων διασφαλίζουν ότι πολλαπλά instances μιας βάσης δεδομένων έχουν τα ίδια δεδομένα συνεχώς.

   Είναι δύσκολο για άλλους τύπους βάσεων δεδομένων να διατηρήσουν αυτό το επίπεδο συνέπειας με μεγάλο πλήθος δεδομένων. Ορισμένες βάσεις, όπως οι NoSQL, μπορούν να παρέχουν μόνο τελική συνέπεια, δηλαδή όταν η βάση δεδομένων επεκτείνεται ή όταν πολλαπλοί χρήστες έχουν πρόσβαση στα ίδια δεδομένα ταυτόχρονα, τα δεδομένα χρειάζονται κάποιο χρόνο για να συγχρονιστούν.

2. **Ατομικότητα :** Η ατομικότητα συμβάλλει στη διατήρηση της ακρίβειας των δεδομένων στη βάση και τη διασφάλιση ότι αυτά συμμορφώνονται με τους κανονισμούς και τις πολιτικές τις οποίες εξυπηρετούν, καθιστώντας μια αλλαγή στη βάση δεδομένων μόνιμη. Στο παράδειγμα των επιχειρήσεων, έστω ότι υπάρχει μια βάση δεδομένων αποθεμάτων που παρακολουθεί τρία εξαρτήματα που χρησιμοποιούνται πάντα μαζί. Όταν ένα εξάρτημα αποσύρεται από το απόθεμα, τα άλλα δύο πρέπει επίσης να αποσυρθούν. Αν ένα από τα τρία εξαρτήματα δεν είναι διαθέσιμο, πρέπει και τα υπόλοιπα εξαρτήματα να αποσυρθούν, καθώς όλα τα τρία εξαρτήματα πρέπει να είναι διαθέσιμα πριν η βάση δεδομένων κάνει οποιαδήποτε δέσμευση. Μια σχεσιακή βάση δεδομένων δεν θα δεσμεύσει για ένα εξάρτημα μέχρι να ξέρει ότι μπορεί να δεσμεύσει για όλα τα τρία.

3. **Πρόσβαση στα Δεδομένα :** Η πρόσβαση στα δεδομένα περιλαμβάνει πολλές ενέργειες που επαναλαμβάνονται . Για παράδειγμα, ένα απλό ερώτημα για την ανάκτηση πληροφοριών από έναν πίνακα δεδομένων μπορεί να χρειαστεί να επαναληφθεί εκατοντάδες ή χιλιάδες φορές για να παραχθεί το επιθυμητό αποτέλεσμα. Αυτές οι λειτουργίες πρόσβασης στα δεδομένα απαιτούν κάποιο είδος κώδικα για την πρόσβαση στη βάση δεδομένων. Για να αποφευχθεί η δημιουργία νέου κώδικα για αυτές τις λειτουργίες σε κάθε νέα εφαρμογή ,οι σχεσιακές βάσεις δεδομένων επιτρέπουν αποθηκευμένες διαδικασίες, οι οποίες είναι μπλοκ κώδικα που μπορούν να προσεγγιστούν με μια απλή κλήση από την εφαρμογή. Για παράδειγμα, μια μοναδική αποθηκευμένη διαδικασία μπορεί να παρέχει συνεπή ετικέτα εγγραφών για χρήστες πολλαπλών εφαρμογών. Οι αποθηκευμένες διαδικασίες μπορούν επίσης να βοηθήσουν τους προγραμματιστές να διασφαλίσουν ότι ορισμένες λειτουργίες δεδομένων στην εφαρμογή υλοποιούνται με συγκεκριμένο τρόπο.

4. **Ταυτόχρονη αλλαγή Δεδομένων :** Όταν πολλοί χρήστες ή εφαρμογές επιχειρούν να αλλάξουν τα ίδια δεδομένα ταυτόχρονα μπορεί να προκύψουν συγκρούσεις σε μια βάση δεδομένων. Τεχνικές κλειδώματος και ταυτόχρονης προσπέλασης μειώνουν την πιθανότητα συγκρούσεων, διατηρώντας την ακεραιότητα των δεδομένων.

   Το κλείδωμα εμποδίζει άλλους χρήστες και εφαρμογές από την πρόσβαση στα δεδομένα ενώ αυτά ενημερώνονται. Σε ορισμένες βάσεις δεδομένων, το κλείδωμα εφαρμόζεται σε ολόκληρο τον πίνακα, κάτι που μπορεί να επηρεάσει αρνητικά την απόδοση της εφαρμογής. Άλλες βάσεις δεδομένων, όπως οι σχεσιακές βάσεις δεδομένων της Oracle, εφαρμόζουν κλειδώματα σε επίπεδο εγγραφής, αφήνοντας τις άλλες εγγραφές εντός του πίνακα διαθέσιμες, βοηθώντας έτσι στη διασφάλιση καλύτερης απόδοσης της εφαρμογής.

   Η ταυτόχρονη προσπέλαση διαχειρίζεται τη δραστηριότητα όταν πολλοί χρήστες ή εφαρμογές εκτελούν ερωτήματα ταυτόχρονα στην ίδια βάση δεδομένων. Αυτή η δυνατότητα παρέχει τη σωστή πρόσβαση σε χρήστες και εφαρμογές .

3.2.1.1.1. POSTGRESQL

Η PostgreSQL είναι ένα αντικειμενοστρεφές σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων (ORDBMS) ανοικτού κώδικα γνωστό για την αξιοπιστία του, την ακεραιότητα των δεδομένων και τις εκτεταμένες δυνατότητές του. Είναι προσαρμόσιμο και επεκτάσιμο, επιτρέποντας στους χρήστες να ορίσουν τους δικούς τους τύπους δεδομένων, συναρτήσεις και προσαρμοσμένους τελεστές [89,90], καθώς και μα προσθέτουν νέες λειτουργίες αλλά και άλλα χαρακτηριστικά .Μπορεί να χειριστεί πολύπλοκες ερωτήσεις, triggers και προβολές, ενώ είναι συμβατό και με μια ποικιλία προγραμματιστικών γλωσσών. Η ισχυρή συμμόρφωσή του με τα πρότυπα SQL, σε συνδυασμό με την υποστήριξη των ACID (Ατομικότητα, Συνέπεια, Απομόνωση, Ανθεκτικότητα), το καθιστούν ιδανική επιλογή για προγραμματιστές και επιχειρήσεις που αναζητούν μια κλιμακούμενη, αποδοτική και ασφαλή βάση δεδομένων.

Τα χαρακτηριστικά μιας PostgreSQL Βάσης είναι τα εξής [89] :

- Γρήγορη και αποτελεσματική αποθήκευση και επεξεργασία δεδομένων γρήγορα με τη διαχείριση μεγάλων συνόλων δεδομένων χωρίς να τίθεται σε κίνδυνο η ταχύτητα.
- Κατάλληλη για πολύπλοκες εφαρμογές και λύσεις big data, χάρη στην ικανότητά της να αποθηκεύει και να διαχειρίζεται πολύπλοκες ερωτήσεις.
- Ευστάθεια, αξιοπιστία και συνεπής απόδοσή .
- Παροχή υποστήριξης, πόρων για αντιμετώπιση προβλημάτων και συνεχών βελτιώσεων από τη δυναμική κοινότητα ανοικτού κώδικα της PostgreSQL.
- Οριζόντια κλιμακούμενη δυνατότητα μέσω τεχνικών όπως η διαίρεση πινάκων και η streaming αντιγραφή, εξασφαλίζοντας την αποτελεσματική διαχείριση μεγάλων συνόλων δεδομένων.
- Βελτιστοποιημένο για απόδοση, επιτρέποντας γρήγορη και αποτελεσματική ανάκτηση και επεξεργασία δεδομένων.

Τα πλεονεκτήματα της PostgreSQL, τα οποία μας οδήγησαν στην επιλογή της για το έργο μας, σε σχέση με την απλή SQL βάση είναι τα εξής [89,90]:

1. **Ευελιξία και Επεκτασιμότητα :** επιτρέπει στους χρήστες να προσαρμόσουν τη βάση δεδομένων σύμφωνα με τις απαιτήσεις τους. Υποστηρίζει μια ευρεία γκάμα σύνθετων τύπων δεδομένων, συμπεριλαμβανομένων της κληρονομικότητας πινάκων , XML , καθώς και προσαρμοσμένων τύπων που δημιουργούν οι χρήστες και άλλα προηγμένα χαρακτηριστικά. Επίσης, υποστηρίζει γλώσσα δεδομένων JSON , καθιστώντας την μια καλή επιλογή για εφαρμογές που απαιτούν δομημένα και ημιδομημένα δεδομένα , και προσφέρει πολλαπλές προγραμματιστικές γλώσσες για τη σύνταξη αποθηκευμένων

διαδικασιών.Επιπλέον , διαθέτει μεγαλύτερη ποικιλία ενσωματωμένων συναρτήσεων σε σύγκριση με τον SQL Server [89]. Τα παραπάνω την καθιστούν πιο ευέλικτη από τον SQL Server και επιτρέπει στον χρήστη να επεκτείνει τις δυνατότητες της βάσης δεδομένων με προσαρμοσμένες συναρτήσεις, τελεστές και νέες γλώσσες, εξασφαλίζοντας ότι η PostgreSQL μπορεί να εξελιχθεί με τις ανάγκες ενός έργου [90].

2. **Προηγμένη βελτιστοποίηση ερωτημάτων :** οι εκτεταμένες δυνατότητες βελτιστοποίησης ερωτημάτων της PostgreSQL επιτρέπουν την αποδοτική εκτέλεση σύνθετων ερωτημάτων. Περιλαμβάνει δυνατότητες όπως ανιχνεύσεις μόνο με δείκτες, ανιχνεύσεις σωρών bitmap και γενετική βελτιστοποίηση ερωτημάτων για τη μείωση των χρόνων εκτέλεσης των ερωτημάτων. Αυτό καθιστά την PostgreSQL ιδανική για εφαρμογές ανάλυσης δεδομένων και επιχειρηματικής ευφυΐας που απαιτούν γρήγορη και ακριβή ανάκτηση δεδομένων. Περιλαμβάνει, επίσης ,υποστήριξη για συναρτήσεις παραθύρων, κοινές εκφράσεις πινάκων (CTEs) και αναδρομικά ερωτήματα, τομείς στους οποίους η MySQL μπορεί να προσφέρει περιορισμένη λειτουργικότητα [90].

3. **Απόδοση και Κλιμάκωση :** Η PostgreSQL είναι γνωστή για την απόδοσή της και μπορεί να διαχειριστεί μεγάλες ποσότητες δεδομένων και ταυτόχρονων συναλλαγών , σε αντίθεση με τον SQL Server που προσφέρει καλύτερη απόδοση για ορισμένες όμως εφαρμογές λόγω της πιο παραδοσιακής του δομής [89].Η PostgreSQL χρησιμοποιεί πολύπλοκες τεχνικές βελτιστοποίησης για να εξασφαλίσει αποτελεσματική αποθήκευση και ανάκτηση δεδομένων, γεγονός που την καθιστά κατάλληλη για περιβάλλοντα με υψηλές απαιτήσεις. Η αρχιτεκτονική της προσφέρει οριζόντια κλιμακούμενη δυνατότητα, διαίρεση και αντιγραφή, επιτρέποντας στις εφαρμογές να κλιμακώνονται αυτόματα ανάλογα με την αύξηση των όγκων δεδομένων και της κίνησης των χρηστών [90].

4. **Συστήματα Λειτουργίας :** Η PostgreSQL έχει σχεδιαστεί για να λειτουργεί σε πολλά λειτουργικά συστήματα, συμπεριλαμβανομένων των Linux, macOS και Windows, ενώ ο SQL Server έχει σχεδιαστεί κυρίως για να λειτουργεί σε Windows, αλλά μπορεί επίσης να εκτελεστεί σε Linux [89].

5. **Κόστος :** Η PostgreSQL είναι δωρεάν στη χρήση, ενώ ο SQL Server απαιτεί άδεια και μπορεί να αποδειχθεί ακριβός, ιδιαίτερα για μεγαλύτερους οργανισμούς [89].

6. **Αξιόπιστη υποστήριξη συναλλαγών :** Η αξιόπιστη υποστήριξη συναλλαγών είναι θεμελιώδης για την PostgreSQL, εξασφαλίζοντας ακεραιότητα και συνέπεια των δεδομένων μέσω πλήρους συμμόρφωσης με το ACID και πολλαπλών επιπέδων απομόνωσης συναλλαγών. Αυτό το πλαίσιο προστατεύει από ανωμαλίες δεδομένων και εγγυάται αξιόπιστη επεξεργασία συναλλαγών, ακόμα και σε πολύπλοκα, πολυχρηστικά περιβάλλοντα. Η PostgreSQL προσφέρει μια ευρύτερη γκάμα επιπέδων απομόνωσης συναλλαγών, σε σχέση με τον SQL Server, συμπεριλαμβανομένου του πολύ αυστηρού επιπέδου Serializable, παρέχοντας περισσότερες επιλογές για τον έλεγχο της σύγχρονης επεξεργασίας και της ακεραιότητας των δεδομένων [90].

3.2.1.2. DOCKER

Η δημιουργία και η εγκατάσταση της PostgreSQL βάση που σχεδιάσαμε για τις ανάγκες του έργου μας έγινε μέσα σε έναν Docker Container.

Το Docker είναι μια ανοιχτή πλατφόρμα για την ανάπτυξη, την αποστολή και την εκτέλεση εφαρμογών. Επιτρέπει τον διαχωρισμό των εφαρμογών από την υποδομή, ώστε να είναι εύκολη και γρήγορη η παράδοση του λογισμικού . Με το Docker, η υποδομή διαχειρίζεται με τον ίδιο τρόπο που διαχειρίζονται και οι εφαρμογές . Οι μεθοδολογίες του για αποστολή, δοκιμή και ανάπτυξη

κώδικα, μειώνουν σημαντικά την καθυστέρηση μεταξύ της συγγραφής κώδικα και της εκτέλεσής του [91].

Το Docker παρέχει τη δυνατότητα συσκευασίας και εκτέλεσης μιας εφαρμογής σε ένα χαλαρά απομονωμένο περιβάλλον που ονομάζεται container. Η απομόνωση και η ασφάλεια επιτρέπουν την εκτέλεση πολλών containers ταυτόχρονα σε έναν δεδομένο κεντρικό υπολογιστή. Τα containers είναι ελαφριά και περιέχουν όλα όσα χρειάζονται για την εκτέλεση της εφαρμογής, ώστε να μην χρειάζεται ό,τι είναι εγκατεστημένο στον κεντρικό υπολογιστή. Είναι δυνατός ο διαμοιρασμός container κατά τη διάρκεια μιας εργασίας , έτσι ώστε όλοι όσοι έχουν πρόσβαση σε αυτόν να έχουν το ίδιο περιβάλλον που λειτουργεί με τον ίδιο τρόπο, χωρίς να χρειάζεται να γίνουν επιπλέον εγκαταστάσεις.

Τα βασικά συστατικά του Docker είναι οι εικόνες (images) και οι containers [91] :

- **Images:**

  Μια εικόνα (image) είναι ένα πρότυπο με οδηγίες για τη δημιουργία ενός container Docker, το οποίο είναι μόνο αναγνώσιμο (read-only). Συχνά, μια εικόνα βασίζεται σε μια άλλη εικόνα, με κάποιες πρόσθετες προσαρμογές. Για παράδειγμα, μπορεί να δημιουργηθεί μια εικόνα που βασίζεται στην εικόνα του Ubuntu, αλλά εγκαθιστά τον διακομιστή ιστού Apache και την εφαρμογή που θέλουμε, καθώς και τις λεπτομέρειες διαμόρφωσης που απαιτούνται για την εκτέλεση της εφαρμογής.

  Παρέχεται η δυνατότητα ο χρήστης να δημιουργήσει τις δικές του εικόνες ή να χρησιμοποιήσει εικόνες που έχουν δημιουργηθεί από άλλους και έχουν δημοσιευτεί σε ένα αποθετήριο (registry). Για τη δημιουργία νέας εικόνας, δημιουργείτε ένα αρχείο Dockerfile με μια απλή σύνταξη για τον ορισμό των βημάτων που απαιτούνται για τη δημιουργία και την εκτέλεση της εικόνας. Κάθε εντολή σε ένα Dockerfile δημιουργεί ένα επίπεδο (layer) στην εικόνα. Όταν αλλάζει το Dockerfile και ξαναχτίζεται η εικόνα, αναδομούνται μόνο εκείνα τα επίπεδα που έχουν αλλάξει. Αυτό καθιστά τις εικόνες τόσο ελαφριές, μικρές και γρήγορες σε σύγκριση με άλλες τεχνολογίες.

- **Containers :**

  Ένας container είναι μια εκτελέσιμη εικόνα. Με τη χρήση του API ή το CLI του Docker δίνεται η δυνατότητα για δημιουργία, έναρξη, παύση, μετακίνηση ή διαγραφή ενός container. Ένας container μπορεί να συνδεθεί σε μία ή περισσότερες δικτυακές διεπαφές, να γίνεται αποθήκευση σε αυτόν, ή και να δημιουργηθεί μια νέα εικόνα βασισμένη στην τρέχουσα κατάστασή του.

  Από προεπιλογή, ένας container είναι σχετικά καλά απομονωμένος από άλλους containers και από τον κεντρικό υπολογιστή του.

  Ορίζεται από την εικόνα του καθώς και από οποιεσδήποτε επιλογές διαμόρφωσης παρέχονται όταν δημιουργείται ή όταν εκκινεί. Όταν ένας container αφαιρεθεί, οποιεσδήποτε αλλαγές στην κατάστασή του που δεν αποθηκεύονται σε μόνιμη αποθήκευση εξαφανίζονται.

Παρακάτω παρουσιάζονται τα πλεονεκτήματα του Docker που μας οδήγησαν στο να στήσουμε τη βάση μας σε ένα Docker Container [91]:

1. Τα containers του Docker μπορούν να εκτελούνται σε τοπικό φορητό υπολογιστή, σε φυσικές ή εικονικές μηχανές σε ένα κέντρο δεδομένων, σε παρόχους cloud ή σε ένα μείγμα

περιβαλλόντων, χωρίς να απαιτείται εκ νέου εγκατάσταση εφαρμογών, καθώς παρέχεται έτοιμη η υποδομή.
2. Η φορητότητα και η ελαφριά φύση του Docker καθιστούν εύκολη τη δυναμική διαχείριση φορτίων εργασίας, την κλιμάκωση ή την αποσύνθεση εφαρμογών και υπηρεσιών σύμφωνα με τις ανάγκες, σχεδόν σε πραγματικό χρόνο.
3. Το Docker είναι ελαφρύ και γρήγορο. Παρέχει μια οικονομικά αποδοτική εναλλακτική λύση στις εικονικές μηχανές που βασίζονται σε υπερεπόπτη (hypervisor), ώστε να είναι δυνατή η χρήση περισσότερων από των ικανοτήτων του διακομιστή για την επίτευξη των στόχων. Το Docker είναι ιδανικό για περιβάλλοντα υψηλής πυκνότητας και για μικρές και μεσαίες εγκαταστάσεις όπου χρειάζεται να γίνουν περισσότερα με λιγότερους πόρους.
4. Η χρήση μιας τοπικής βάσης δεδομένων σε container προσφέρει ευελιξία και ευκολία στην εγκατάσταση, επιτρέποντάς την πιστή αντιγραφή του περιβάλλοντος παραγωγής χωρίς τον υπερβολικό φόρτο των παραδοσιακών εγκαταστάσεων βάσεων δεδομένων. Το Docker απλοποιεί αυτήν τη διαδικασία, επιτρέποντάς την ανάπτυξη, τη διαχείριση και την κλιμάκωση βάσεων δεδομένων σε απομονωμένους containers με λίγες εντολές.

3.2.1.3. LUCIDCHART ΚΑΙ DRAW.IO

Για τη σχεδίαση του ERD μοντέλου της βάσης μας χρησιμοποιήσαμε 2 εργαλεία, τα οποία αποτελούν δύο από τα πιο δημοφιλή εργαλεία αναπαράστασης διαγράμματος. Σχεδιάσαμε δύο διαφορετικές μορφές ERD , που όμως απεικονίζουν το ίδιο μοντέλο. Για αυτό το λόγο χρησιμοποιήσαμε το lucidchart για το 1ο , όπου η οντότητες απεικονίζονται σαν πίνακες και τα χαρακτηριστικά τους σαν πεδία, και το draw.io για το 2ο όπου οι οντότητες είναι με τη μορφή ορθογωνίων ,ενώ ονοματίζονται και οι σχέσεις μεταξύ των οντοτήτων. Προσφέρουν εύχρηστο δημιουργό ERD με πληθώρα προτύπων και σχήματα, επιτρέποντας στους χρήστες να σχεδιάζουν και να διαχειρίζονται τα ERD με ευκολία.

Τα ERD που δημιουργήσαμε παρουσιάζονται αναλυτικά στην επόμενη ενότητα.

3.2.1.4. GORM

Η GORM αποτελεί μια Object-Relational Mapping (ORM) βιβλιοθήκη της γλώσσας Go. Τη χρησιμοποιήσαμε για τους μηχανισμούς ερωτήσεων και τη διαχείριση των δεδομένων , καθώς επιτρέπει τη μετατροπή της μορφής των δεδομένων από τη γλώσσα Go στη μορφή της σχεσιακής βάσης δεδομένων μας και αντίστροφα [93]. Χρησιμοποιείται για την αλληλεπίδραση με βάσεις δεδομένων με έναν πιο αντικειμενοστραφή τρόπο, επιτρέποντας στους προγραμματιστές να εργάζονται με δεδομένα ως αντικείμενα, αντί να γράφουν απευθείας SQL ερωτήματα. Με τη GORM μπορούμε να δημιουργήσουμε ένα σχήμα βάσης δεδομένων από την αρχή, να διατηρήσουμε ενημερωμένο το σχήμα, να αντιστοιχίσουμε πίνακες σε δομές μοντέλων και να εκτελέσουμε εύκολα λειτουργίες Δημιουργίας (Create), Ανάγνωσης (Read), Ενημέρωσης (Update) και Διαγραφής (Delete) στους πίνακες της βάσης δεδομένων [93].

Μερικά βασικά χαρακτηριστικά της παρουσιάζονται παρακάτω:

1. **Απλή σύνταξη**: Η GORM προσφέρει μια εύχρηστη και κατανοητή σύνταξη που διευκολύνει τη δημιουργία, ανάγνωση, ενημέρωση και διαγραφή (CRUD) εγγραφών στη βάση δεδομένων.

2. **Αυτόματος Χειρισμός Μοντέλων**: Μπορεί να δημιουργήσει αυτόματα πίνακες, ευρετήρια, και περιορισμούς βάσει των δομών που ορίζουμε στα Go μοντέλα μας.

3. **Σχέσεις**: Υποστηρίζει τη διαχείριση σχέσεων μεταξύ των μοντέλων, όπως one-to-one, one-to-many και many-to-many.

4. **Migrator**: Προσφέρει λειτουργίες για την εκτέλεση μεταναστεύσεων βάσεων δεδομένων, επιτρέποντας την εύκολη αλλαγή της δομής της βάσης δεδομένων μας.

5. **Επεκτασιμότητα**: Η GORM είναι επεκτάσιμη και μπορεί να προσαρμοστεί με plugins ή προσαρμοσμένους τύπους.

6. **Πλήρης υποστήριξη SQL**: Επιτρέπει την εκτέλεση SQL ερωτημάτων όταν χρειάζεται, προσφέροντας ευελιξία .

7. **Query Builder**: Διαθέτει έναν ισχυρό query builder για τη δημιουργία πολύπλοκων ερωτημάτων με ευανάγνωστο και συντηρήσιμο κώδικα.

3.2.1.5. GIN FRAMEWORK

Για τη δημιουργία των endpoints του API μας , με τα οποία επιτυγχάνεται η επικοινωνία του χρήστη με τη βάση μας χρησιμοποιήσαμε το Gin Framework της γλώσσας Go. Είναι γνωστό για την απλότητά και την υψηλή απόδοσή του, τα οποία το καθιστούν μια δημοφιλή επιλογή για ευκολότερη δημιουργία εφαρμογών και API στο web με τη χρήση της γλώσσας Go.

Παρέχει τα εξής βασικά χαρακτηριστικά :

1. **Δρομολόγηση**: Παρέχει έναν ισχυρό δρομολογητή με υποστήριξη παραμέτρων και προτύπων.

2. **Ενδιάμεσο**: Η υποστήριξη ενδιάμεσου επιτρέπει στους προγραμματιστές να εισάγουν κώδικα που τρέχει πριν ή μετά από τις αιτήσεις, ενδυναμώνοντας λειτουργίες όπως το logging, η ταυτοποίηση, ο περιορισμός ρυθμού, κ.λπ.

3. **Ανάλυση JSON**: Το Gin παρέχει ενσωματωμένη υποστήριξη για την ανάλυση αιτημάτων JSON και την απόδοση απαντήσεων JSON.

4. **Επικύρωση**: Το πλαίσιο περιλαμβάνει έναν μηχανισμό επικύρωσης για την επικύρωση δεδομένων αιτήσεων που εισέρχονται.

5. **Χειρισμός σφαλμάτων**: Το Gin παρέχει μηχανισμούς για τη χειρισμό σφαλμάτων με ευγενικό τρόπο, συμπεριλαμβανομένης της ανάκτησης των πανικών και της προσαρμοσμένης χειριστικής αντιμετώπισης σφαλμάτων.

6. **Παροχή στατικών αρχείων**: Μπορεί να παρέχει στατικά αρχεία και πόρους αποτελεσματικά.

Η επιλογή του Gin έναντι άλλων πλαισίων εργασίας για τη γλώσσα Go οφείλεται στο ευρύ φάσμα λειτουργιών που προσφέρει, το οποίο προαναφέρθηκε, στη σαφή και ευέλικτη σύνταξή του που καθιστά εύκολη την κατασκευή API και web εφαρμογών, καθώς και για την για υψηλή απόδοση και κλιμακωσιμότητα, που το καθιστούν ιδανικό για εφαρμογές που απαιτούν γρήγορη απόκριση και χειρισμό μεγάλων φορτίων.

3.2.1.6. POSTMAN

Για τον έλεγχο της λειτουργίας και της εγκυρότητας του συστήματός μας θα χρησιμοποιήσουμε το εργαλείο Postman. Το Postman είναι μια πλατφόρμα ανάπτυξης και δοκιμής API (Διεπαφής προγραμματισμού εφαρμογών). Χρησιμοποιείται ευρέως από προγραμματιστές για να απλοποιήσουν τη διαδικασία δοκιμής APIs, παρέχοντας ένα φιλικό προς τον χρήστη περιβάλλον για την εκτέλεση αιτημάτων, την προβολή απαντήσεων και τον εντοπισμό σφαλμάτων.

Το Postman παρέχει τις εξής δυνατότητες [97]:

1. **Σχεδίαση API :** Με τη διεπαφή χρήστη του μας παρέχει τη δυνατότητα να δημιουργήσουμε API endpoints, να στείλουμε αιτήματα προς τον server της εφαρμογής και να ορίσουμε τις παραμέτρους , τα αιτήματα και τις αποκρίσεις, καθώς και τις ρυθμίσεις αυθεντικοποίησης. Ο οπτικός επεξεργαστής επιτρέπει επίσης την τεκμηρίωση του API με σαφείς και συνοπτικές περιγραφές, κάνοντάς το εύκολο για άλλους προγραμματιστές να το κατανοήσουν και να το χρησιμοποιήσουν.

2. **API Debugging :** Το Postman προσφέρει σενάρια πριν και μετά τα αιτήματά μας που μας επιτρέπουν να τα προσαρμόσουμε και να αυτοματοποιήσουμε ορισμένες εργασίες. Τα σενάρια πριν το αίτημα μπορούν να χρησιμοποιηθούν για τον ορισμό δυναμικών μεταβλητών, κεφαλίδων και παραμέτρων, ενώ τα σενάρια μετά το αίτημα μπορούν να χρησιμοποιηθούν για την εκτέλεση ενεργειών βάσει της απάντησης του API. Με αυτά τα σενάρια, μπορούμε να εντοπίσουμε και να διορθώσουμε γρήγορα τυχόν προβλήματα.

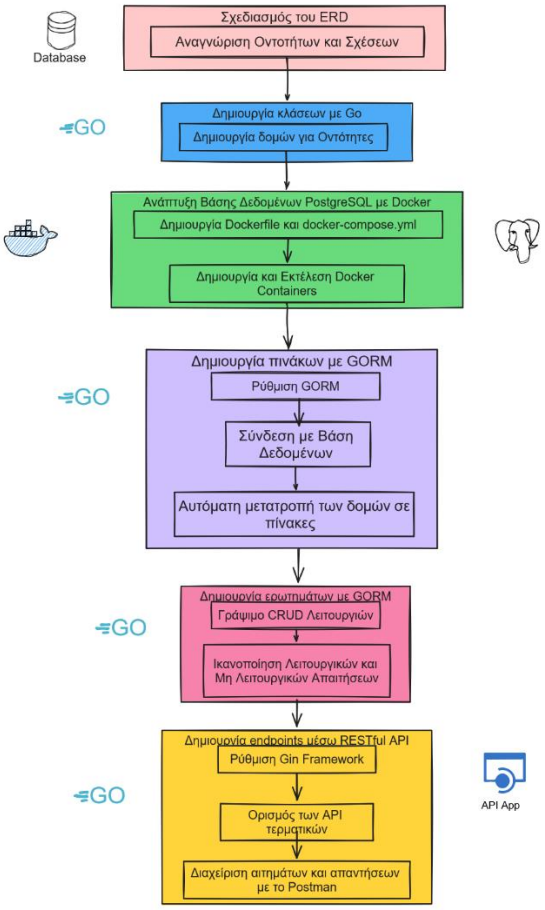
3. **API Testing :** Το Postman παρέχει ένα ολοκληρωμένο σύνολο χαρακτηριστικών που μας βοηθάει να δοκιμάσουμε το API μας και να διασφαλίσουμε ότι πληροί τα απαιτούμενα πρότυπα που χρειαζόμαστε. Με την αυτοματοποιημένη δοκιμή, μπορούμε εύκολα να ρυθμίσουμε ροές εργασίας συνεχούς ενσωμάτωσης και συνεχούς παράδοσης (CI/CD), επιτρέποντάς μας να δοκιμάζουμε το API σας σε κάθε στάδιο της ανάπτυξης. Παρέχει επίσης μια σειρά εργαλείων δοκιμών, συμπεριλαμβανομένων των unit tests, integration tests και load tests, προκειμένου να διασφαλίσουμε ότι το API μας αποδίδει όπως αναμένεται υπό διάφορες συνθήκες. Με αυτά τα εργαλεία, μπορούμε να διασφαλίσουμε ότι το API είναι αξιόπιστο, ασφαλές και επεκτάσιμο.

Οι μέθοδοι που εφαρμόσαμε για να δοκιμάσουμε το σύστημά μας , καθώς και ο τρόπος με τον οποίο τις χρησιμοποιήσαμε το Postman αναλύονται εκτενώς στο Κεφάλαιο 5.

3.3. ΣΧΕΔΙΟ ΥΛΟΠΟΙΗΣΗΣ

3.3.1. ΒΗΜΑ ΠΡΟΣ ΒΗΜΑ ΠΡΟΣΕΓΓΙΣΗ ΓΙΑ ΤΗΝ ΑΝΑΠΤΥΞΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Τα βήματα που ακολουθήσαμε για την ανάπτυξη του συστήματός μας παρουσιάζονται με τη σειρά στο παρακάτω διάγραμμα :



Εικόνα 3.1: Διάγραμμα ανάπτυξης του συστήματος

1. **Σχεδιασμός του ERD :** Για να το πετύχουμε αυτό χρειάστηκε αρχικά να αναγνωρίσουμε τις οντότητες και τις μεταξύ τους σχέσεις που θελουμε να περιλαμβάνει η βάση δεδομένων μας. Στη συνέχεια, με τη βοήθεια εργαλείων όπως το Lucidchart και το draw.io δημιουργήσαμε το διάγραμμα οντοτήτων-σχέσεων για να απεικονίσουμε αυτές τις σχέσεις και να οργανώσουμε τη δομή της βάσης δεδομένων σας.

2. **Δημιουργία χρησιμοποιώντας Go :** Γράψαμε τις δομές (structs) για κάθε οντότητα που ορίσαμε παραπάνω, περιλαμβάνοντας τις ιδιότητες και τα χαρακτηριστικά τους. Επίσης, ορίσαμε μέσα στις δομές και τις σχέσεις μεταξύ των οντοτήτων, όπου και εάν υπάρχουν.

3. **Ανάπτυξη της Βάσης Δεδομένων PostgreSQL χρησιμοποιώντας Docker :** Για αρχή δημιουργήσαμε ένα Dockerfile και ένα αρχείο docker-compose.yml για να ορίσουμε τις ρυθμίσεις του περιβάλλοντος της βάσης δεδομένων μας. Στη συνέχεια, δημιουργήσαμε και να εκτελέσαμε containers στο Docker ,για να φιλοξενήσουν τη βάση δεδομένων PostgreSQL.

   Παρακάτω παρουσιάζονται τα βήματα που έγιναν , καθώς και η χρησιμότητα καθενός από αυτά, για την εγκατάσταση και τον σχεδιασμό της PostgreSQL βάσης μας στον Docker container :

- Για το project μας θα χρησιμοποιήσουμε την γλώσσα προγραμματισμού Go, οπότε για αρχή εγκαθιστούμε στο σύστημά μας τα απαραίτητα πακέτα που χρειάζονται τόσο για αυτή όσο και για την PostgreSQL βάση μας.

- Δημιουργούμε τοπικά έναν φάκελο για το project μας και μέσα σε αυτόν, φτιάχνουμε ένα αρχείο με το όνομα docker-compose.yml , στο οποίο περιλαμβάνονται τα credentials και οι πληροφορίες για να συνδεθούμε στη βάση μας, όπως το όνομα της βάσης, το όνομα του χρήστη για να συνδεθεί κάποιος, ο κωδικός πρόσβασης , το image στο οποίο θα δημιουργηθεί η βάση , καθώς και η αντιστοίχιση του τοπικού μας port με το port στο Docker. Μέσα σε αυτό το φάκελο περιλαμβάνεται και το αρχείο main.go με το οποίο γίνεται η σύνδεση στη βάση και η δημιουργία των πινάκων.

- Ανοίγουμε το Docker Desktop για να εκκινήσουμε το Docker engine.

- Ανοίγουμε ένα παράθυρο στη γραμμή εντολών ,κατευθυνόμαστε στον φάκελο του project μας και τρέχουμε τις παρακάτω εντολές. Οι εντολές που παρουσιάζονται στο βήμα 4 μπορούν να συγκεντρωθούν σε ένα αρχείο run_script , το οποίο θα εκτελείται κατευθείαν με την εντολή **./run_script.sh** , έτσι ώστε να μη χρειάζεται να εκτελούμε μία-μία τις εντολές του βήματος 4 :
  - **sudo docker-compose up --build -d** -> Για να δημιουργηθεί ο PostgreSQL container και να γίνει διαθέσιμος στο port που έχουμε ορίσει στο docker-compose.yml. Εμείς έχουμε ορίσει το port 5432 του μηχανήματός μας να αντιστοιχίζεται στο port 5432 στον container. Η εντολή αυτή γίνεται την 1η φορά και χτίζει κάθε φορά από την αρχή την εικόνα που έχει οριστεί για τις υπηρεσίες στο yml αρχείο και εκκινεί τον container. Για να μην ξαναδημιουργούμε κάθε φορά τον container μπορούμε να τρέχουμε την εντολή sudo docker-compose up -d , η οποία απλώς τον εκκινεί

Εικόνα 3.2 : Δημιουργία και εκκίνηση του Docker container

  - **sudo docker exec -it postgresqlgooptional-postgres-1 sh** -> Για να αλληλεπιδράσουμε με τον container που εκκινήσαμε παραπάνω. Το postgresqlgooptional-postgres-1 είναι το όνομα του container μας και δημιουργείται αυτόματα από το όνομα του project μας, το όνομα της υπηρεσίας που έχουμε ορίσει στο docker-compose file και έναν σειριακό αριθμό.

    Για να βρούμε το όνομα του container εάν δεν το γνωρίζουμε μπορούμε να εκτελέσουμε την εντολή sudo docker ps , η οποία μας επιστρέφει μια λίστα με όλους τους containers και τις πληροφορίες τους για να βρούμε αυτόν που μας ενδιαφέρει.

Εικόνα 3.3 : Λίστα των Docker containers

  - **psql -U admin -d admin** -> Για να συνδεθούμε και να αποκτήσουμε πρόσβαση στην PostgreSQL βάση μας. Το πρώτο admin που είναι δεξιά από το U αντιπροσωπεύει το όνομα του χρήστη με το οποίο έχει δημιουργηθεί η βάση, ενώ το δεύτερο admin αντιπροσωπεύει το όνομα

της βάσης μας , και τα δύο ορισμένα στο docker-compose αρχείο. Μετά από αυτό είμαστε μέσα στη βάση και μπορούμε να πραγματοποιήσουμε ό,τι ενέργειες χρειαζόμαστε σε γλώσσα SQL, όπως δημιουργία πινάκων, εισαγωγή δεδομένων, δημιουργία τύπων και ερωτήματα για την ανάκτηση δεδομένων που μας ενδιαφέρουν.

```
# psql -U admin -d admin
psql (16.2 (Debian 16.2-1.pgdg120+2))
Type "help" for help.

admin=#
```

Εικόνα 3.4 : Πρόσβαση στην PostgreSQL βάση

- Ανοίγουμε ένα νέο παράθυρο στη γραμμή εντολών, κατευθυνόμαστε στον φάκελο του project μας και τρέχουμε το main αρχείο , στο οποίο περιλαμβάνεται η σύνδεση σε αυτή τη βάση, η δημιουργία των πινάκων, η δημιουργία των endpoints κλπ, με την εντολή go run main.go, με την οποία γίνεται διαθέσιμος ο http server για να δεχτεί αιτήματα στη θύρα 1414.

```
Initializing database connection...
Database connection succesfull 😎
```

Εικόνα 3.5 : Σύνδεση στην PostgreSQL βάση

```
[GIN-debug] Listening and serving HTTP on :1414
```

Εικόνα 3.6: Εκκίνηση του HTTP server στο port 1414

4. **Δημιουργία πινάκων χρησιμοποιώντας GORM :** Χρησιμοποιήσαμε τη λειτουργία αυτοματοποίησης του GORM (Automigrate) για να δημιουργήσουμε τους πίνακες στη βάση δεδομένων από τις δομές (structs) που ορίσαμε στο βήμα 2.

5. **Δημιουργία ερωτημάτων χρησιμοποιώντας GORM :** Δημιουργήσαμε ερωτήματα με βάση τις λειτουργίες CRUD (Create, Read, Update, Delete) χρησιμοποιώντας τη GORM για να αλληλοεπιδράσουμε με τους πίνακες της βάσης δεδομένων μας. Τα ερωτήματα αυτά πρέπει να ικανοποιούν τις λειτουργικές και μη λειτουργικές απαιτήσεις που θέλουμε να ικανοποιεί το σύστημά μας, έτσι όπως αναλύθηκαν στην υποενότητα 3.1.1. , όπως για παράδειγμα η ασφάλεια των δεδομένων, η εγγραφή νέου χρήστη, η ανάκτηση πληροφοριών για μια συσκευή κλπ.

6. **Δημιουργία endpoints μέσω RESTful API :** Το τελευταίο βήμα ήταν η ρύθμιση του Gin Framework και ο ορισμός των τερματικών σημείων endpoints με routes που σχετίζονται για τον κάθε πίνακα οντοτήτων , όπως το /house/getall που επιστρέφει όλα τα σπίτια που είναι αποθηκευμένα στη βάση μας. Στη συνέχεια, μέσω του Postman δοκιμάσαμε τα URLs αυτών των endpoints και επιβεβαιώσαμε ότι λειτουργούν σωστά και μας επιστρέφουν τα επιθυμητά αποτελέσματα.

ΚΕΦΑΛΑΙΟ 4

4.ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ

4.1. ΣΧΕΔΙΑΣΗ ΣΧΗΜΑΤΟΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

4.1.1. ΔΙΑΓΡΑΜΜΑ ΟΝΤΟΤΗΤΩΝ-ΣΥΣΧΕΤΙΣΕΩΝ ΚΑΙ ΛΕΠΤΟΜΕΡΕΙΕΣ ΣΧΗΜΑΤΟΣ

4.1.1.1. ΟΡΙΣΜΟΣ ΔΙΑΓΡΑΜΜΑΤΟΣ ΟΝΤΟΤΗΤΩΝ-ΣΥΣΧΕΤΙΣΕΩΝ (ERD)

Το διάγραμμα οντοτήτων-συσχετίσεων είναι ένας τύπος ροής που απεικονίζει πώς οντότητες , όπως άνθρωποι, σπίτια , αντικείμενα, σχετίζονται μεταξύ τους εντός ενός συστήματος. Αποτελούν οπτικοποίηση των σχεσιακών βάσεων δεδομένων , βοηθώντας έτσι στο σχεδιασμό και στην αποσφαλμάτωσή τους. Χρησιμοποιούν ορισμένο σύνολο συμβόλων, όπως ορθογώνια, ρόμβους , οβάλ , προκειμένου να απεικονίσουν τις οντότητες, τη συνδεσιμότητά τους , τις σχέσεις και τα γνωρίσματά τους.

4.1.1.2. ΧΡΗΣΙΜΟΤΗΤΑ

Ένα ERD μοντελοποιεί τη βάση και αποτελεί ένα αρχικό βήμα για τον προσδιορισμό των απαιτήσεων ενός έργου, των λογικών κανόνων και των τεχνολογιών που θα εφαρμοστούν. Χρησιμεύει , ακόμη, στην ανάλυση της βάσης και ,κατά συνέπεια, στον ευκολότερο εντοπισμό και στην επίλυση προβλημάτων που μπορεί να προκύψουν στην πορεία τόσο στη λογική όσο και στην εγκατάστασή της. Καθώς περιλαμβάνουν οντότητες, ενέργειες και μεταξύ τους αλληλεπιδράσεις , μπορούν να αποκαλύψουν με μεγαλύτερη ευκολία πληροφορίες, έχοντας έτσι καθοριστικό ρόλο στην ανάλυση δεδομένων και σε έρευνες που αφορούν αυτά [92].

4.1.1.3. ΒΑΣΙΚΑ ΣΥΣΤΑΤΙΚΑ

1. **Οντότητες (Entities) :** Αντιπροσωπεύουν στοιχεία δεδομένων μέσα σε μια βάση , τα οποία μπορεί να είναι ζωντανά ή μη ζωντανά , πραγματικά η αφηρημένα, όπως για παράδειγμα άνθρωποι , τοποθεσίες, αντικείμενα, γεγονότα. Στο ERD συνήθως απεικονίζονται με ορθογώνια , με το όνομα της οντότητας στο πάνω μέρος.

2. **Χαρακτηριστικά (Attributes) :** Είναι τα γνωρίσματα και οι ιδιότητες που μπορεί να έχει μια οντότητα. Για παράδειγμα, η οντότητα του ανθρώπου μπορεί να έχει σαν χαρακτηριστικά το όνομά του, το τηλέφωνό του και την ημερομηνία γέννησής του. Συνήθως απεικονίζονται με ένα οβάλ συνδεδεμένο με την οντότητα, ή σαν γραμμές μέσα στον πίνακά της. Δύο βασικά χαρακτηριστικά των οντοτήτων, τα οποία τα δηλώνουμε και ξεχωριστά , είναι το πρωτεύον κλειδί (primary key-PK) και το ξένο κλειδί (foreign key-FK). Το PK περιέχεται υποχρεωτικά σε κάθε οντότητα και αποτελεί το μοναδικό αναγνωριστικό κάθε εγγραφής του πίνακά της.Το FK σε έναν πίνακα αναφέρεται στο πρωτεύον κλειδί ενός άλλου πίνακα με τον οποίο αυτός συνδέεται, μεταφέροντας τα χαρακτηριστικά μιας εγγραφής του 1^{ου} στον 2^{ο}.

3. **Σχέσεις (relationships) :** Υποδηλώνουν πώς διάφορες οντότητες αλληλοεπιδρούν μεταξύ τους. Για παράδειγμα, σε μια βάση δεδομένων για μια υπηρεσία ενοικίασης αυτοκινήτων , η οντότητα «πελάτης» θα δημιουργήσει μια σχέση με την οντότητα «όχημα». Η σχέση μεταξύ δύο πινάκων μπορεί να είναι υποχρεωτική (δηλώνεται με μια γραμμή κάθετη στη γραμμή της σχέσης) ή προαιρετική (δηλώνεται με έναν κενό κύκλο). Επίσης, υποδεικνύεται και ο αριθμός των φορών που μια οντότητα μπορεί να συσχετιστεί με μια άλλη οντότητα. Το νούμερο αυτό ονομάζεται καρδιναλιότητα (cardinality).

Υπάρχουν 3 είδη καρδιναλιότητας :

- One-to-one : Σχέση 1 προς 1. Ένα στιγμιότυπο μιας οντότητας μπορεί να συσχετιστεί μόνο με ένα στιγμιότυπο από μια άλλη οντότητα. Για παράδειγμα, ένας πελάτης μπορεί να νοικιάσει μόνο 1 όχημα τη φορά και ένα όχημα μπορεί να νοικιαστεί μόνο από έναν πελάτη σε μια συγκεκριμένη χρονική περίοδο.

- One-to-many : Σχέση 1 προς πολλά. Ένα στιγμιότυπο μιας οντότητας μπορεί να συσχετιστεί με πολλά στιγμιότυπα από μια άλλη οντότητα. Για παράδειγμα, ένας πελάτης μπορεί να νοικιάσει πολλά αυτοκίνητα , αλλά ένα όχημα μπορεί να νοικιαστεί μόνο από έναν πελάτη σε μια συγκεκριμένη χρονική περίοδο.

- Many-to-many : Σχέση πολλά προς πολλά. Πολλά στιγμιότυπα μιας οντότητας μπορούν να συσχετιστούν με πολλά στιγμιότυπα από μια άλλη οντότητα. Για παράδειγμα, ένας πελάτης μπορεί να νοικιάσει πολλά αυτοκίνητα και ένα όχημα μπορεί να νοικιαστεί από πολλούς πελάτες σε μια συγκεκριμένη χρονική περίοδο.

4.1.1.4. ΤΟ ERD ΤΗΣ ΒΑΣΗΣ ΜΑΣ

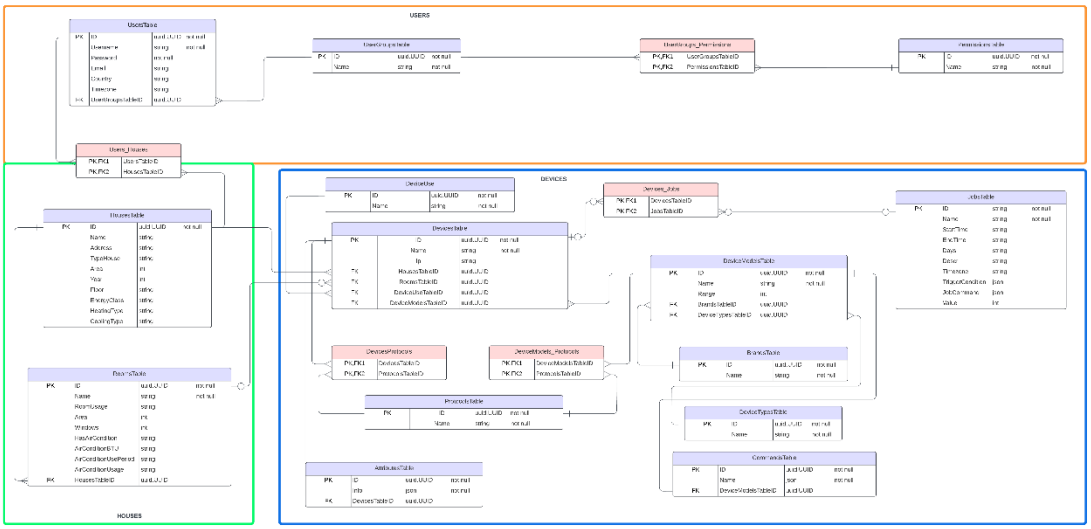
Το πρώτο βήμα για τη δημιουργία μιας νέας βάσης δεδομένων είναι το μοντέλο οντοτήτων-συσχετίσεων(ERD), το οποίο έχει καθορισμένη δομή και παρέχει το εννοιολογικό σχήμα για τη σχεδίαση της επιθυμητής βάσης δεδομένων.

Η επιλογή και ορισμός των οντοτήτων είναι κρίσιμος για τη σωστή λειτουργία του συστήματος, ενώ οι σχέσεις αποτελούν το κλειδί για την κατανόηση της δυναμικής μεταξύ τους.

Το ERD βοηθάει στην κατανόηση των αλληλεπιδράσεων μεταξύ διαφόρων στοιχείων ενός συστήματος, για αυτό και αποτελεί απαραίτητη προϋπόθεση για τον σχεδιασμό και την υλοποίηση της βάσης δεδομένων.

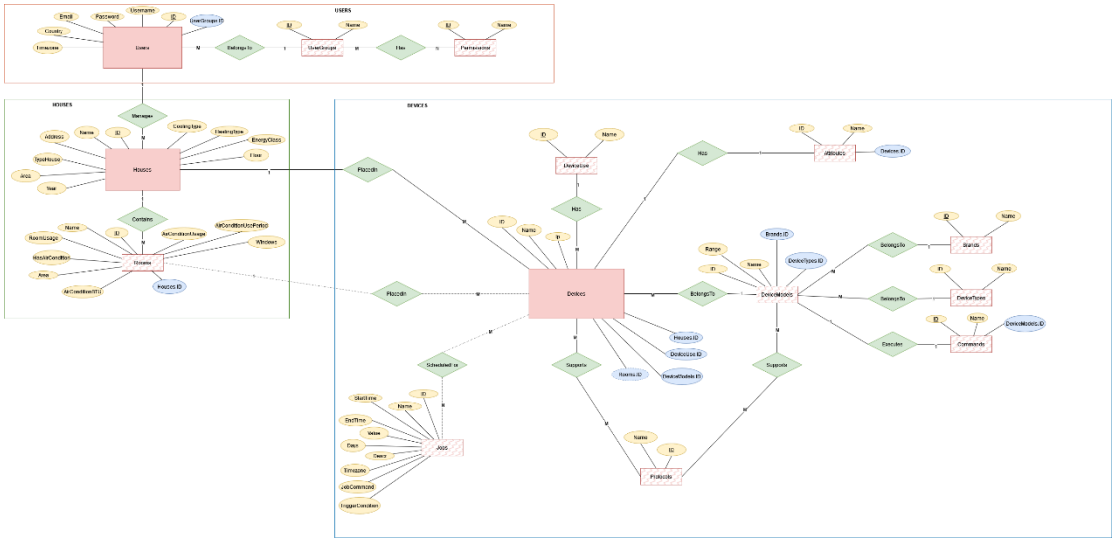
Παρακάτω παρουσιάζεται σε δύο διαφορετικές μορφές το διάγραμμα οντοτήτων-συσχετίσεων που μοντελοποιεί τη βάση μας.

Στην πρώτη (Information Engineering Notation) οι οντότητες απεικονίζονται με μορφή πινάκων και τα χαρακτηριστικά τους προστίθενται μέσα στον πίνακα σε γραμμές, ξεχωρίζοντας δίπλα το primary key (που είναι και υπογραμμισμένο) και τα foreign keys . Οι 14 βασικές οντότητες απεικονίζονται με μωβ χρώμα, ενώ οι υπόλοιπες 5 που αφορούν ενδιαμέσους πίνακες λόγω των σχέσεων many-to-many (θα εξηγηθούν παρακάτω) είναι με πορτοκαλί χρώμα. Επίσης, σημειώνονται οι υποχρεωτικές και προαιρετικές σχέσεις μεταξύ των οντοτήτων, καθώς και το cardinality τους. Όπως γίνεται εμφανές και από το διάγραμμα, οι οντότητές μας μπορούν να χωριστούν σε 3 βασικές κατηγορίες : (i)αυτές που έχουν να κάνουν με τους χρήστες (πορτοκαλί) και έχουν ως επίκεντρο τον πίνακα των users, (ii)αυτές που έχουν να κάνουν με τα σπίτια (πράσινο) και έχουν ως επίκεντρο τον πίνακα των houses, (iii) και αυτές που έχουν να κάνουν με τις συσκευές (μπλε) και έχουν σαν επίκεντρο τον πίνακα των devices.



Εικόνα 4.1 : ERD (Information Engineering Notation)

Στη δεύτερη (Chen Notation) οι οντότητες απεικονίζονται με ορθογώνια, τα χαρακτηριστικά τους με κίτρινα οβάλ (με υπογράμμιση για το primary key) και τα foreign keys με γαλάζια οβάλ. Όσο αφορά τι σχέσεις, δηλώνεται επίσης το cardinality τους και η υποχρεωτικότητά τους (οι διακεκομμένες γραμμές εκφράζουν προαιρετικές σχέσεις, ενώ όλες οι υπόλοιπες υποχρεωτικές) . Η διαφορά με το προηγούμενο διάγραμμα είναι η προσθήκη των πράξεων (actions) στις σχέσεις, οι οποίες διευκολύνουν την ανάγνωση και την κατανόηση του διαγράμματος, χωρίς ωστόσο να είναι απαραίτητες. Και εγώ είναι εμφανείς οι 3 κατηγορίες οντοτήτων: users (πορτοκαλί) , houses(πράσινο),devices(μπλε). Οικ 3 βασικές οντότητες των κατηγοριών είναι με συμπαγές χρώμα, ενώ οι υπόλοιπες είναι με ζιγκ-ζαγκ γραμμές. Παρατηρούμε ότι η κατηγορία των devices έχει τη μορφή αστέρα (star schema), δηλαδή υπάρχει ένας κεντρικός πίνακας γεγονότων (fact table) , στην περίπτωσή μας ο πίνακας των devices, που συνδέεται με πολλούς πίνακες διαστάσεων (dimension tables). Η μορφή με την οποία είναι δομημένη η κατηγορία των devices μας δείχνει πως το σύστημά μας μπορεί να υποστηρίξει οποιοδήποτε τύπο συσκευής και οποιοδήποτε εξοπλισμό έχει να κάνει με το έξυπνο σπίτι.



Εικόνα 4.2 : ERD (Chen Notation)

Το ERD της βάσης μας , όπως φαίνεται και παραπάνω, αποτελείται από τα παρακάτω 14 βασικά entities :

1. UsersTable
2. UserGroupsTable
3. PermissionsTable
4. HousesTable
5. RoomsTable
6. DeviceUseTable
7. BrandsTable
8. DeviceTypesTable
9. DeviceModelsTable
10. DevicesTable
11. AttributesTable
12. ProtocolsTable
13. CommandsTable
14. JobsTable

Entities και Attributes

Παρακάτω περιγράφονται αναλυτικά οι 14 οντότητες και τα attributes τους μαζί με σχόλια και επεξηγήσεις:

1. **UsersTable :** Οντότητα στην οποία αποθηκεύονται οι πληροφορίες για κάθε χρήστη.

Users Table			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή χρήστη.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Username	Συμβολοσειρά (string)	Το όνομα, το οποίο ορίζεται από κάθε χρήστη κατά την εγγραφή του στην εφαρμογή και απαιτείται για κάθε μελλοντική είσοδό του σε αυτό.	Δεν μπορεί να πάρει κενές τιμές.
Password	Συμβολοσειρά (string)	Ο κωδικός , ο οποίος ορίζεται από κάθε χρήστη κατά την εγγραφή τους στην εφαρμογή και απαιτείται για κάθε μελλοντική είσοδό του σε αυτό.	Όπως θα δούμε και παρακάτω , ο κωδικός αποθηκεύεται στη βάση μας κρυπτογραφημένος, έτσι ώστε να μην μπορεί κανένας να έχει πρόσβαση σε αυτόν. Δεν μπορεί να πάρει κενές τιμές.

Email	Συμβολοσειρά (string)	Η ηλεκτρονική διεύθυνση του χρήστη.	
Country	Συμβολοσειρά (string)	Η χώρα στην οποία ζει ο χρήστης.	
Timezone	Συμβολοσειρά (string)	Η ζώνη ώρας της χώρας στην οποία ζει ο χρήστης .	
UserGroupsTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των users με την οντότητα των user groups.	

Πίνακας 4.1 : Οντότητα Users

2. **UserGroupsTable :** Οντότητα στην οποία αποθηκεύονται οι κατηγορίες στις οποίες ανήκουν οι χρήστες, με βάση τις πληροφορίες που μπορούν έχουν πρόσβαση, όπως για παράδειγμα admins ή members.

UsersGroupsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή κατηγορίας χρηστών.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα της κάθε κατηγορίας.	Δεν μπορεί να πάρει κενές τιμές.

Πίνακας 4.2 : Οντότητα UserGroups

3. **PermissionsTable :** Οντότητα στην οποία αποθηκεύονται οι άδειες που μπορούν έχουν οι χρήστες σε σχέση με τις συσκευές του σπιτιού , όπως για παράδειγμα να τους επιτρέπεται να έχουν πρόσβαση στις κάμερες ασφαλείας ή να ανοιγοκλείνουν κάποιους διακόπτες. Οι άδειες των χρηστών εξαρτώνται από την κατηγορία στην οποία ανήκουν.

PermissionsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή άδειας.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.

Name	Συμβολοσειρά (string)	Το όνομα της κάθε κατηγορίας.	Δεν μπορεί να πάρει κενές τιμές.

Πίνακας 4.3 : Οντότητα Permissions

4. **HousesTable :** Οντότητα στην οποία αποθηκεύονται τα χαρακτηριστικά κάθε σπιτιού.

HousesTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή σπιτού.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα του σπιτιού, έτσι όπως το δίνει ο χρήστης.	Δεν μπορεί να πάρει κενές τιμές.
Address	Συμβολοσειρά (string)	Η διεύθυνση του σπιτιού.	
TypeHouse	Συμβολοσειρά (string)	Το είδος του σπιτιού.	
Area	Ακέραιος αριθμός (int)	Τα τετραγωνικά μέτρα του σπιτιού.	
Year	Ακέραιος αριθμός (int)	Το έτος κατασκευής του σπιτιού.	
Floor	Συμβολοσειρά (string)	Σε ποιόν όροφο είναι το σπίτι.	
EnergyClass	Συμβολοσειρά (string)	Η ενεργειακή κατηγορία του σπιτιού.	
HeatingType	Συμβολοσειρά (string)	Ο τρόπος θέρμανσης του σπιτιού.	
CoolingType	Συμβολοσειρά (string)	Ο τρόπος ψύξης του σπιτιού.	

Πίνακας 4.4 : Οντότητα Houses

5. **RoomsTable :** Οντότητα στην οποία αποθηκεύονται τα χαρακτηριστικά κάθε δωματίου.

RoomsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή δωματίου.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα του δωματίου, έτσι όπως το δίνει ο χρήστης.	Δεν μπορεί να πάρει κενές τιμές.

RoomUsage	Συμβολοσειρά (string)	Η χρήση του δωματίου.	
Area	Ακέραιος αριθμός (int)	Τα τετραγωνικά μέτρα του δωματίου.	
Windows	Ακέραιος αριθμός (int)	Το πλήθος των παραθύρων που διαθέτει το δωμάτιο.	
HasAirCondition	Συμβολοσειρά (string)	Τιμή που δείχνει το δωμάτιο διαθέτει κλιματιστικό.	
AirConditionBTU	Συμβολοσειρά (string)	Πόσα btu είναι το κλιματιστικό του δωματίου , εάν υπάρχει.	
AirConditionUsePeriod	Συμβολοσειρά (string)	Το χρονικό διάστημα που χρησιμοποιείται το κλιματιστικό του δωματίου, εάν υπάρχει.	
AirConditionUsage	Συμβολοσειρά (string)	Η χρήση του κλιματιστικού του δωματίου, εάν υπάρχει.	
HousesTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των rooms με την οντότητα των houses .	

Πίνακας 4.5 : Οντότητα Rooms

6. **DeviceUseTable:** Οντότητα στην οποία αποθηκεύονται οι πληροφορίες για τις χρήσεις που μπορούν να έχουν οι συσκευές.

DeviceUseTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή χρήσης συσκευής.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα της χρήσης της συσκευής.	Δεν μπορεί να πάρει κενές τιμές.

Πίνακας 4.6 : Οντότητα DeviceUse

7. **BrandsTable:** Οντότητα στην οποία αποθηκεύονται οι πληροφορίες για τη μάρκα κάθε μοντέλου συσκευής.

BrandsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή μάρκας συσκευής.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα της μάρκας της συσκευής.	Δεν μπορεί να πάρει κενές τιμές.

Πίνακας 4.7 : Οντότητα Brands

8. **DeviceTypesTable:** Οντότητα στην οποία αποθηκεύονται οι πληροφορίες για το είδος της κάθε συσκευής.

DeviceTypesTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή τύπου συσκευής.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα της χρήσης της συσκευής.	Δεν μπορεί να πάρει κενές τιμές. Υπάρχουν μόνο 3 είδη έξυπνων συσκευών με τα οποία απασχολούμαστε, οι αισθητήρες (Sensors) ,οι ελεγκτές (Controllers) και αυτές που είναι και τα δύο (Both). Οπότε αυτό το πεδίο μπορεί να πάρει μόνο τις 3 αυτές τιμές.

Πίνακας 4.8 : Οντότητα DeviceTypes

9. **DeviceModelsTable:** Οντότητα στην οποία αποθηκεύονται τα χαρακτηριστικά κάθε μοντέλου συσκευής.

DeviceModelsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή μοντέλου συσκευής.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.

Name	Συμβολοσειρά (string)	Το όνομα του μοντέλου της συσκευής.	Δεν μπορεί να πάρει κενές τιμές.
Range	Ακέραιος αριθμός (int)	Που μπορούν να κυμανθούν οι τιμές κάθε μοντέλου.	
BrandsTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των device models με την οντότητα των brands.	
DeviceTypesTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των device models με την οντότητα των device types.	

Πίνακας 4.9 : Οντότητα DeviceModels

10. **DevicesTable :** Οντότητα στην οποία αποθηκεύονται τα βασικά στατικά χαρακτηριστικά για κάθε συσκευή . Διαφέρει από την παραπάνω οντότητα, καθώς στο DeviceModelsTable περιλαμβάνονται χαρακτηριστικά που έχουν όλες οι συσκευές ενός συγκεκριμένου μοντέλου, ενώ εδώ προστίθενται σε αυτά και σταθερά χαρακτηριστικά που αφορούν τη συγκεκριμένη συσκευή, όπως το όνομά της και η ip της.

DevicesTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή συσκευής.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα της συσκευής, έτσι όπως το δίνει ο χρήστης.	Δεν μπορεί να πάρει κενές τιμές.
Ip	Συμβολοσειρά (string)	Το Πρωτόκολλό Διαδικτύου που υποστηρίζει η συσκευή για την επικοινωνία της.	
HousesTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των devices με την οντότητα των houses.	Η σύνδεση με τον πίνακα των houses συμπληρωματικά με των πίνακα των rooms καλύπτει και την περίπτωση των συσκευών , όπως θερμοστάτες, που ανήκουν σε όλο το σπίτι και όχι σε κάποιο συγκεκριμένο δωμάτιο.
RoomsTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique	Foreign key. Συνδέει την οντότητα των devices με την οντότητα των rooms.	

	indentifier - uuid.UUID)		
DeviceUseTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των devices με την οντότητα των device use.	
DeviceModelsTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των devices με την οντότητα των device models.	

Πίνακας 4.10 : Οντότητα Devices

11. **AttributesTable:** Οντότητα στην οποία αποθηκεύονται τα δυναμικά χαρακτηριστικά κάθε συσκευής, τα οποία διαφέρουν από συσκευή σε συσκευή, και δεν αφορούν κοινά χαρακτηριστικά συσκευών που ανήκουν στο ίδιο μοντέλο.

AttributesTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή χαρακτηριστικών.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Info	Json format	Τα δυναμικά χαρακτηριστικά της συσκευής	Δεν μπορεί να πάρει κενές τιμές. Είναι ευέλικτο και δίνει τη δυνατότητα να αποθηκευτούν χαρακτηριστικά που δεν τα διαθέτουν όλες οι συσκευές. Για παράδειγμα, για μία συσκευή μπορεί να παίρνει τιμές {"topic":"topic","period":30,"is_main_switch":false}, ενώ για μια άλλη {"mac":"00:1A:2B:3C:4D:5E","port":"12345"}
DevicesTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των attributes με την οντότητα των devices .	

Πίνακας 4.11 : Οντότητα Attributes

12. **ProtocolsTable:** Οντότητα στην οποία αποθηκεύονται οι πληροφορίες για τα πρωτόκολλα επικοινωνίας που μπορούν να υποστηρίξουν τα μοντέλα συσκευών ή συγκεκριμένες συσκευές.

ProtocolsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή πρωτοκόλλου επικοινωνίας.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα του πρωτοκόλλου επικοινωνίας.	Δεν μπορεί να πάρει κενές τιμές.

Πίνακας 4.12 : Οντότητα Protocols

13. **CommandsTable:** Οντότητα στην οποία αποθηκεύονται όλες οι εντολές τις οποίες μπορεί να εκτελέσει κάθε συσκευή.

CommandsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή εντολών.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Json format	Οι εντολές που μπορεί να εκτελέσει μια συσκευή.	Δεν μπορεί να πάρει κενές τιμές. Δίνει ευελιξία στις τιμές που μπορούν να αποθηκευτούν, για παράδειγμα {"commands":["on","off"]}
DeviceModelsTableID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Foreign key. Συνδέει την οντότητα των commands με την οντότητα των device models .	

Πίνακας 4.13 : Οντότητα Commands

14. **JobsTable:** Οντότητα στην οποία αποθηκεύονται όλες οι εργασίες που είναι προγραμματισμένες και οι οποίες σχετίζονται με κάποια ή κάποιες συσκευές. Σε μια εργασία μπορεί να εμπλέκονται πάνω από 1 συσκευές, συσκευές-αισθητήρες που συλλέγουν δεδομένα, συσκευές-ελεγκτές που εκτελούν μια εργασία, ή και τα δύο.

JobsTable			
Χαρακτηριστικό (Attribute)	**Μορφή Δεδομένων (Data Type)**	**Τι είναι**	**Σχόλια**
ID	Καθολικά μοναδικό αναγνωριστικό (universally unique indentifier - uuid.UUID)	Είναι το primary key της οντότητας. Αναγνωριστικό και μοναδικό για κάθε νέα εγγραφή προγραμμάτων.	Παίρνει μια τυχαία τιμή , με τη βοήθεια γεννήτριας τιμών, για κάθε νέα εγγραφή και δεν μπορεί να πάρει κενές τιμές.
Name	Συμβολοσειρά (string)	Το όνομα της προγραμματισμένης εργασίας πχ lights on sunset.	Δεν μπορεί να πάρει κενές τιμές.
StartTime	Συμβολοσειρά (string)	Η ώρα έναρξης της εργασίας .	
EndTime	Συμβολοσειρά (string)	Η ώρα λήξης της εργασίας .	
Days	Συμβολοσειρά (string)	Ποιες μέρες θα εκτελείται η εργασία .	πχ Κάθε μέρα ή μόνο Σαββατοκύριακα.
Descr	Συμβολοσειρά (string)	Σύντομη περιγραφή της εργασίας.	
Timezone	Συμβολοσειρά (string)	Η ζώνη ώρας η οποία ισχύει στην περιοχή που βρίσκονται οι συσκευές που σχετίζονται με αυτή την εργασία.	
TriggerCondition	Json format	Η συνθήκη που πυροδοτεί την έναρξη της εργασίας.	Επεξήγηση στο σχόλιο παρακάτω
JobCommand	Json format	Η εντολή που θα εκτελεστεί τις ώρες ή μόλις πραγματοποιηθεί η συνθήκη που έχει οριστεί από την εργασία.	
Value	Ακέραιος αριθμός (int)	Η τιμή που θα πάρει η συσκευή που έχει οριστεί.	Μπορεί να είναι και null , καθώς δεν χρειάζονται όλες οι εντολές συσκευών να οριστεί τιμή . Για παράδειγμα το απλό άνοιγμα των φώτων μπορεί να αφορά μόνο εντολές on/off, οπότε το πεδίο value να είναι κενό, ενώ ο ορισμός της θερμοκρασίας ενός κλιματιστικού χρειάζεται συγκεκριμένη τιμή.

Πίνακας 4.14 : Οντότητα Jobs

Σχόλιο : Ο πίνακας αυτός περιλαμβάνει 2 βασικές περιπτώσεις προγραμματισμού εργασιών.

Η πρώτη αφορά την περίπτωση που μια εργασία ορίζεται να εκτελείται για συγκεκριμένες ώρες σε συγκεκριμένες μέρες, όπως για παράδειγμα το άνοιγμα των φώτων κατά τις ώρες του ηλιοβασιλέματος. Σε αυτή την περίπτωση θα έχουν τιμές τα πεδία **StartTime, EndTime , Days , JobCommand** (πχ οι τιμές αντίστοιχα μπορεί να είναι 6,9,everyday, on), ενώ το πεδίο **TriggerCondition** θα είναι κενό, καθώς ο προγραμματισμός της εργασίας εξαρτάται από συγκεκριμένες ώρες και όχι από άλλες συνθήκες. Όλα τα υπόλοιπα πεδία

έχουν κανονικά τις τιμές τους, ενώ το **Value** τυχαίνει να μην ορίζεται εδώ, όπως αναφέρθηκε και στο αντίστοιχο σχόλιο.

Η δεύτερη περίπτωση αφορά τον προγραμματισμό μια εργασίας όταν η τιμή ενός αισθητήρα φτάσει ένα συγκεκριμένο κατώφλι, όπως για παράδειγμα το άνοιγμα των φώτων όταν ένα συγκεκριμένος αισθητήρας ανιχνεύσει κάποια τιμή φωτεινότητας που του έχουμε ορίσει εμείς. Τότε τα πεδία **StartTime, EndTime** θα είναι κενά, το **TriggerCondition** θα ορίζει ποιος αισθητήρας είναι αυτός και ποιο είναι το κατώφλι του (για αυτό έχει μορφή json, για να υπάρχει ευελιξία στις πληροφορίες που αποθηκεύονται) .Το **JobCommand** θα είναι το ίδιο, το **Value** θα είναι κενό , όλα τα υπόλοιπα πεδία έχουν κανονικά τις τιμές τους, ενώ το **Days** μπορεί πάλι να παίρνει τιμές πχ everyday που θα δηλώνουν ότι ο έλεγχος των τιμών του αισθητήρα θα γίνεται κάθε μέρα, στο συγκεκριμένο παράδειγμα.

Relationships

Παρακάτω παρουσιάζονται τα 3 ειδών σχέσεων που συναντάμε στα διαγράμματα Οντοτήτων-Συσχετίσεων.

1. **One – to- many:** Κάθε εγγραφή μιας οντότητας μπορεί να συνδεθεί με πολλές εγγραφές μιας άλλης οντότητας, αλλά η κάθε εγγραφή της 2ης οντότητας μπορεί να συνδεθεί μόνο με 1 από την 1η οντότητα.
2. **Many-to-many:** Κάθε εγγραφή της μίας οντότητας μπορεί να συνδεθεί με πολλές εγγραφές της άλλης οντότητας και αντίστροφα. Για αυτή τη σύνδεση δημιουργείται ένας ενδιάμεσος πίνακας που συνδέεται με τα 2 tables και περιλαμβάνει τα 2 foreign keys των ids των 2 αυτών πινάκων.
3. **One-to-one:** Κάθε εγγραφή της μίας οντότητας μπορεί να συνδεθεί με 1 μοναδική εγγραφή της άλλης οντότητας και αντίστροφα.

Στο σχήμα της βάσης μας οι σχέσεις που έχουν οριστεί παρουσιάζονται ανά κατηγορίες παρακάτω :

1. **One- to- many :**
   - UserGroupsTable-UsersTable: Κάθε χρήστης μπορεί να ανήκει σε μία μόνο κατηγορία , η οποία ορίζει τις άδειες που έχει για κάθε συσκευή. Κάθε κατηγορία μπορεί να περιλαμβάνει περισσότερους από 1 χρήστες . Για παράδειγμα, οι γονείς μιας οικογένειας μπορεί να θέλουμε να έχουν διαφορετική πρόσβαση στις συσκευές του σπιτιού από ότι τα ανήλικα παιδιά.

   - HousesTable-DevicesTable : Κάθε συσκευή μπορεί να ανήκει σε 1 μόνο σπίτι , αλλά κάθε σπίτι περιλαμβάνει πολλές συσκευές.

   - HousesTable-RoomsTable: Κάθε σπίτι περιλαμβάνει πολλά δωμάτια, αλλά κάθε δωμάτιο μπορεί να ανήκει σε 1 μόνο σπίτι.

   - RoomsTable-DevicesTable: Κάθε συσκευή μπορεί να ανήκει σε 1 μόνο δωμάτιο , αλλά κάθε δωμάτιο μπορεί να περιλαμβάνει πολλές συσκευές. Η σχέση αυτή είναι

προαιρετική (optional) , καθώς μια συσκευή μπορεί να μη συνδέεται με κανένα δωμάτιο, αλλά να αποτελεί συσκευή γενικού για ένα σπίτι, οπότε να συνδέεται μόνο με αυτό (πχ ένας θερμοσίφωνας).

- DeviceUseTable-DevicesTable: Κάθε συσκευή έχει μια συγκεκριμένη χρήση, αλλά η ίδια χρήση μπορεί να επιτυγχάνεται από πολλές διαφορετικές συσκευές.

- DeviceModelsTable-DevicesTable: Κάθε συσκευή ανήκει σε 1 μόνο μοντέλο , ενώ 1 μοντέλο μπορεί να περιλαμβάνει πολλές συσκευές.

- BrandsTable-DeviceModelsTable: Κάθε μοντέλο ανήκει σε ένα συγκεκριμένο brand μιας εταιρίας, αλλά ένα brand μπορεί να περιλαμβάνει περισσότερα από 1 μοντέλα.

- DeviceTypesTable-DeviceModelsTable: Κάθε μοντέλο συσκευής είναι 1 συγκεκριμένου τύπου (αισθητήρας, ελεγκτής ή και τα δύο) , αλλά κάθε τύπος μπορεί να περιλαμβάνει πάνω από 1 μοντέλα.

2. Many-to-many:

- UserGroupsTable-PermissionsTable: Κάθε κατηγορία χρηστών (πχ admin,member) περιλαμβάνει πολλές άδειες ως προς τη διαχείριση των συσκευών (πχ Switch On/Off Plug) , και αντίστοιχα πολλές άδειες μπορούν να ανήκουν σε παραπάνω από 1 κατηγορίες. Δημιουργείται ένας ενδιάμεσος πίνακας *permission_usergroup* ο οποίος περιλαμβάνει το id του *permission_table_id* και το *user_groups_table_id*.

- UsersTable-HousesTable : Κάθε χρήστης μπορεί να έχει στην ιδιοκτησία του (/να νοικιάζει/να διαμένει) σε περισσότερα από 1 σπίτια στων οποίων τα δεδομένα θέλει να έχει πρόσβαση. Αντίστοιχα, σε ένα σπίτι μπορούν να εμπλέκονται περισσότεροι από 1 χρήστες. Δημιουργείται ένας ενδιάμεσος πίνακας *user_houses* ο οποίος περιλαμβάνει το id του *houses_table_id* και το *users_table_id*.

- DeviceModelsTable-ProtocolsTable: Κάθε μοντέλο συσκευής μπορεί να υποστηρίζει περισσότερα από 1 πρωτόκολλα επικοινωνίας , και αντίστροφα, κάποιο πρωτόκολλο επικοινωνίας μπορεί να υποστηρίζεται από διαφορετικά μοντέλα συσκευών. Δημιουργείται ένας ενδιάμεσος πίνακας *protocol_devicemodel* ο οποίος περιλαμβάνει το id του *protocols_table_id* και το *device_models_table_id*.

- DevicesTable-ProtocolsTable: Κάθε συσκευή μπορεί να υποστηρίζει περισσότερα από 1 πρωτόκολλα επικοινωνίας , και αντίστροφα, κάποιο πρωτόκολλο επικοινωνίας μπορεί να υποστηρίζεται από διαφορετικές συσκευές. Δημιουργείται ένας ενδιάμεσος πίνακας *protocol_device* ο οποίος περιλαμβάνει το id του *protocols_table_id* και το *devices_table_id*.

- DevicesTable-JobsTable: Κάθε συσκευή μπορεί να εμπλέκεται σε παραπάνω από 1 προγραμματισμένες εργασίες και αντίστροφα μια προγραμματισμένη εργασία μπορεί να απαιτεί πάνω από 1 συσκευές. Δημιουργείται ένας ενδιάμεσος πίνακας *device_job* ο οποίος περιλαμβάνει το id του *jobs_table_id* και το *devices_table_id*.

Τα 5 παραπάνω entities που αναφέρθηκαν στην προηγούμενη ενότητα (για αυτό και έχουμε συνολικά 19 πίνακες στη βάση μας και όχι 14) αφορούν ακριβώς αυτές τις 5 many-to-many σχέσεις, 1 πίνακας για την καθεμία.

3. **One-to-one:**
   - DevicesTable-AttributesTable: Κάθε συσκευή, και μόνο αυτή, έχει ένα συγκεκριμένο συνδυασμό χαρακτηριστικών , οπότε κάθε εγγραφή συσκευής αντιστοιχίζεται σε 1 μόνο εγγραφή χαρακτηριστικών και αντίστροφα, κάθε εγγραφή χαρακτηριστικών μπορεί να αντιστοιχίζεται σε 1 μόνο συσκευή.

   - DeviceModelsTable-CommandsTable: Κάθε μοντέλο συσκευής, και μόνο αυτό, έχει ένα συγκεκριμένο συνδυασμό εντολών , οπότε κάθε εγγραφή μοντέλου συσκευής αντιστοιχίζεται σε 1 μόνο εγγραφή εντολών και αντίστροφα, κάθε εγγραφή εντολών μπορεί να αντιστοιχίζεται σε 1 μόνο μοντέλο συσκευής.

Η PostgreSQL βάση που παρουσιάζεται χρησιμοποιείται για την αποθήκευση χαρακτηριστικών και πληροφοριών που αφορούν τις συσκευές, τα σπίτια, τα δωμάτια κλπ όπως αναλύονται παραπάνω. Τα δεδομένα αυτά κάποια είναι περασμένα από πριν, όπως τα Brands των διαθέσιμων συσκευών, οι κατηγορίες χρηστών που υπάρχουν, τα είδη των συσκευών κλπ και οι χρήστες μπορούν να επιλέξουν τα επιθυμητά κατά την εισαγωγή μιας νέας συσκευής ή ενός νέου χρήστη. Άλλα δεδομένα, όπως το όνομα του σπιτιού ή του δωματίου και όλα τα χαρακτηριστικά τους, οι πληροφορίες χρήστη, το όνομα μιας συσκευής κλπ εισάγονται αποκλειστικά από τους χρήστες.

4.2. ΜΕΘΟΔΟΙ ΑΝΑΚΤΗΣΗΣ ΚΑΙ ΔΙΑΧΕΙΡΙΣΗΣ ΔΕΔΟΜΕΝΩΝ

4.2.1. SQL

Ο ευκολότερος και πιο συνηθισμένος τρόπος αλληλεπίδρασης του χρήστη με τη βάση είναι η χρήση απλών SQL ερωτημάτων. Αφού συνδεθούμε στη βάση μας με τον τρόπο που εξηγήθηκε παραπάνω, μπορούμε να κάνουμε μερικές δοκιμές σχετικά με τη διαχείριση των δεδομένων μας.

Η δημιουργία των πινάκων, για παράδειγμα του πίνακα των χρηστών, με τη βοήθεια της SQL μπορεί να γίνει ως εξής :

```
admin=# CREATE TABLE users_tables (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    username VARCHAR NOT NULL,
    password VARCHAR NOT NULL,
    email VARCHAR,
    country VARCHAR,
    timezone VARCHAR,
    project VARCHAR,
    user_groups_table_id UUID REFERENCES user_groups(id) ON UPDATE CASCADE ON DELETE SET NULL
);
```

Εικόνα 4.3 : Δημιουργία οντότητας users με SQL

```
admin=# CREATE TABLE houses_tables (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    name VARCHAR NOT NULL,
    address VARCHAR,
    typehouse VARCHAR,
    area INT,
    year INT,
    floor VARCHAR,
    energyclass VARCHAR,
    heatingtype VARCHAR,
    coolingtype VARCHAR
);
```

Εικόνα 4.4 : Δημιουργία οντότητας houses με SQL

```
admin=# CREATE TABLE user_houses (
    user_id UUID REFERENCES users(id) ON DELETE CASCADE,
    house_id UUID REFERENCES houses(id) ON DELETE CASCADE,
    PRIMARY KEY (user_id, house_id)
);
```

Εικόνα 4.5 : Δημιουργία ενδιάμεσου πίνακα user_houses με SQL

```
admin=# CREATE TABLE user_groups_tables (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    name VARCHAR NOT NULL
);
```

Εικόνα 4.6 : Δημιουργία οντότητας user_groups με SQL

Αντίστοιχα μπορούν να δημιουργηθούν και οι υπόλοιποι πίνακες των οντοτήτων που χρειαζόμαστε.

Οι δοκιμές που μπορούμε να κάνουμε στα δεδομένα μας παρουσιάζονται παρακάτω:

- **Εισαγωγή νέας εγγραφής user group**. Το “INSERT 0 1” μας δείχνει ότι έχει εισαχθεί στη βάση επιτυχώς :

```
admin=# INSERT INTO user_groups_tables (
    ID,
    Name
) VALUES (
    uuid_generate_v4(),
    'admin');
INSERT 0 1
```

Εικόνα 4.7 : Νέα εγγραφή user_groups με SQL

- **Εισαγωγή νέας εγγραφής χρήστη με το FK του user group.** Το “INSERT 0 1” μας δείχνει ότι έχει εισαχθεί στη βάση επιτυχώς :

```
admin=# INSERT INTO users_tables (
    ID,
    Username,
    Password,
    Email,
    Country,
    Timezone,
    Project,
    user_groups_table_id
) VALUES (
    uuid_generate_v4(),
    'mary',
    'securepassword123',
    'mary@example.com',
    'Greece',
    'EET',
    'EHS',
    '584d4cef-96fc-40cf-8f8c-8673cd18457a'
);
INSERT 0 1
```

Εικόνα 4.8 : Νέα εγγραφή user με SQL

- **Ανάκτηση όλων των χρηστών που είναι αποθηκευμένοι στη βάση μας:**

```
admin=# SELECT * FROM users_tables;
```

Εικόνα 4.9 : Εντολή ανάκτησης όλων των χρηστών με SQL

```
                  id                  | username |     password      |        email        | country | timezone | project |          user_groups_
table_id
--------------------------------------+----------+-------------------+---------------------+---------+----------+---------+----------------------
------------------
 89099e8a-5547-467c-b66b-4031c6d208c5 | dimitra  | securepassword123 | dimitra@example.com | Greece  | EET      | EHS     |
 28c3e1ae-b325-4cfd-9b0f-d03d5612fbc9 | mary     | securepassword123 | mary@example.com    | Greece  | EET      | EHS     | 584d4cef-96fc-40cf-8
f8c-8673cd18457a
(2 rows)
```

Εικόνα 4.10 : Αποτελέσματα ανάκτησης όλων των χρηστών με SQL

- **Ανάκτηση των πληροφοριών για συγκεκριμένο χρήστη :**

```
admin=# SELECT *  FROM users_tables
WHERE ID = '89099e8a-5547-467c-b66b-4031c6d208c5';
```

Εικόνα 4.11 : Εντολή ανάκτησης των πληροφοριών συγκεκριμένου χρήστη με SQL

```
                  id                  | username |     password      |        email        | country | timezone | project | user_groups_table_id
--------------------------------------+----------+-------------------+---------------------+---------+----------+---------+----------------------
-
 89099e8a-5547-467c-b66b-4031c6d208c5 | dimitra  | securepassword123 | dimitra@example.com | Greece  | EET      | EHS     |
(1 row)
```

Εικόνα 4.12 : Αποτελέσματα ανάκτησης των πληροφοριών συγκεκριμένου χρήστη με SQL

- **Διαγραφή συγκεκριμένου χρήστη :**

```
admin=# DELETE FROM users_tables
WHERE ID = '89099e8a-5547-467c-b66b-4031c6d208c5';
DELETE 1
```

Εικόνα 4.13 : Εντολή διαγραφής συγκεκριμένου χρήστη με SQL

Το “DELETE 1” σημαίνει ότι ο χρήστης έχει διαγραφεί επιτυχώς, και αν ζητήσουμε να μας επιστραφούν όλοι οι χρήστες που είναι αποθηκευμένοι τώρα στη βάση, με την εντολή SELECT * FROM users_tables; θα επιβεβαιώσουμε ότι πράγματι αυτός ο χρήστης με το συγκεκριμένο id έχει διαγραφεί , εκτελώντας την εντολή SELECT * FROM users_tables και λαμβάνοντας το αποτέλεσμα:

Εικόνα 4.14 : Αποτέλεσμα διαγραφής του χρήστη με SQL

- **Ενημέρωση των στοιχείων συγκεκριμένου χρήστη :**

Αλλάζουμε το email του παραπάνω χρήστη από mary@example.com σε mary@gmail.com

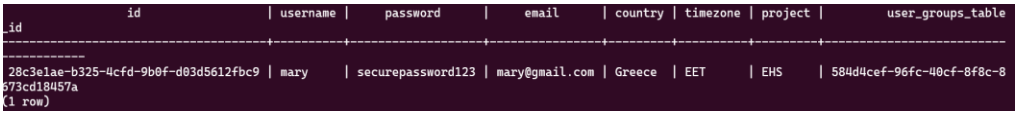
```
admin=# UPDATE users_tables
SET
    Email = 'mary@gmail.com'
WHERE
    ID = '28c3e1ae-b325-4cfd-9b0f-d03d5612fbc9';
UPDATE 1
```

Εικόνα 4.15 : Εντολή ενημέρωσης των στοιχείων συγκεκριμένου χρήστη με SQL

Το “UPDATE 1” σημαίνει ότι έχει γίνει η ενημέρωση και αν ζητήσουμε να μας επιστραφούν οι πληροφορίες του συγκεκριμένου χρήστη μπορούμε αν το επιβεβαιώσουμε :

```
admin=# SELECT *  FROM users_tables
WHERE ID = '28c3e1ae-b325-4cfd-9b0f-d03d5612fbc9';
```

Εικόνα 4.16 : Εντολή ανάκτησης των πληροφοριών του συγκεκριμένου χρήστη με SQL



Εικόνα 4.17 : Αποτελέσματα ανάκτησης των πληροφοριών του συγκεκριμένου χρήστη με SQL

- **Δημιουργία νέας εγγραφής σπιτιού :**

```
admin=# INSERT INTO houses_tables (
    ID,
    Name,
    Address,
    Type_House,
    Area,
    Year,
    Floor,
    Energy_Class,
    Heating_Type,
    Cooling_Type
) VALUES (
    uuid_generate_v4(),
    'Example  House',
    '1234 Street',
    'Detached',
    90,
    2022,
    '2',
    'A',
    'Central',
    'None'
);
INSERT 0 1
```

Εικόνα 4.18 : Νέα εγγραφή σπιτιού με SQL

```
admin=# SELECT * FROM houses_tables;
```

Εικόνα 4.19 : Εντολή ανάκτησης όλων των σπιτιών με SQL

```
                  id                  |      name      |   address   | type_house | area | year | floor | energy_class | heating_type | cooling_
type
--------------------------------------+----------------+-------------+------------+------+------+-------+--------------+--------------+---------
-----
 cc37ca72-6a99-416c-8df7-b72083a8182e | Example  House | 1234 Street | Detached   |   90 | 2022 | 2     | A            | Central      | None
(1 row)
```

Εικόνα 4.20 : Αποτελέσματα ανάκτησης όλων των σπιτιών με SQL

- **Σύνδεση ενός χρήστη με ένα σπίτι, μέσω του ενδιάμεσου πίνακα της many-to-many σχέσης :**

Για να συνδέσουμε τον χρήστη με id =28c3e1ae-b325-4cfd-9b0f-d03d5612fbc9 με το σπίτι με id=cc37ca72-6a99-416c-8df7-b72083a8182e , θα πραγματοποιήσουμε μια νέα εγγραφή στον ενδιάμεσο πίνακα user_houses που εκφράζει τη μεταξύ τους many-to-many σχέση.

```
admin=# INSERT INTO user_houses (
    users_table_id,houses_table_id
) VALUES (
   '28c3e1ae-b325-4cfd-9b0f-d03d5612fbc9','cc37ca72-6a99-416c-8df7-b72083a8182e'
);
INSERT 0 1
```

Εικόνα 4.21 : Εντολή σύνδεσης χρήστη-σπίτι με SQL

Και με την εντολή «SELECT * FROM user_houses;» παρατηρούμε ότι όντως έχει γίνει η σύνδεση

```
admin=# SELECT * FROM user_houses;
           houses_table_id            |            users_table_id
--------------------------------------+--------------------------------------
 cc37ca72-6a99-416c-8df7-b72083a8182e | 28c3e1ae-b325-4cfd-9b0f-d03d5612fbc9
(1 row)
```

Εικόνα 4.22 : Αποτέλεσμα ανάκτησης όλων των συνδέσεων χρήστη-σπίτι με SQL

Όπως γίνεται κατανοητό από τα παραπάνω, η δημιουργία όλων των πινάκων με SQL πρέπει να γίνει χειροκίνητα και να δημιουργηθούν τα ξένα κλειδιά και οι ενδιάμεσοι πίνακες έτσι ώστε να τεθούν σε λειτουργία οι σχέσεις μεταξύ των πινάκων. Για παράδειγμα, η σύνδεση της many-to-many σχέσης μεταξύ της οντότητας των χρηστών και της οντότητας των σπιτιών πρέπει να γίνει με τη χειροκίνητη δημιουργία του ενδιάμεσου πίνακα user_houses.

4.2.2.GORM

Η GORM διαθέτει τύπο δεδομένων struct για τη δημιουργία μοντέλων που αντιπροσωπεύουν συγκεκριμένους πίνακες βάσεων δεδομένων. Έτσι, οι πίνακες χρηστών και σπιτιών , για παράδειγμα, που ορίστηκαν παραπάνω με τη γλώσσα SQL , μπορούν τώρα να οριστούν ως εξής :

```
type UsersTable struct {
    ID                uuid.UUID       `gorm:"type:uuid;default:uuid_generate_v4();primaryKey"`
    Username          string          `json:"username" gorm:"not null"`
    Password          string          `json:"password" gorm:"not null"`
    Email             string          `json:"email"`
    Country           string          `json:"country"`
    Timezone          string          `json:"timezone"`
    UserGroupsTableID *uuid.UUID      `json:"user_groups_table_id" gorm:"type:uuid;constraint:OnUpdate:CASCADE,OnDelete:SET NULL;"`
    UserGroup         UserGroupsTable `json:"user_group" gorm:"foreignKey:UserGroupsTableID"`
    Houses            []*HousesTable  `gorm:"many2many:user_houses;"`
}

type HousesTable struct {
    ID          uuid.UUID      `gorm:"type:uuid;default:uuid_generate_v4();primaryKey"`
    Name        string         `json:"name" gorm:"not null"`
    Address     string         `json:"address"`
    TypeHouse   string         `json:"typehouse"`
    Area        int            `json:"area"`
    Year        int            `json:"year"`
    Floor       string         `json:"floor"`
    EnergyClass string         `json:"energyclass"`
    HeatingType string         `json:"heatingtype"`
    CoolingType string         `json:"coolingtype"`
    Users       []*UsersTable  `gorm:"many2many:user_houses;"`
    Devices     []DevicesTable `json:"devices"`
    Rooms       []RoomsTable   `json:"rooms"`
}
```

Εικόνα 4.23 : Ορισμός πινάκων (structs) users και houses με GORM

Με αυτό τον τρόπο, ο ενδιάμεσος πίνακας user_houses που αντιπροσωπεύει την many-to-many σχέση μεταξύ τους δημιουργείται αυτόματα στη βάση, σύμφωνα με τα χαρακτηριστικά που έχουμε ορίσει, χωρίς να χρειάζεται να κάνουμε κάτι εμείς χειροκίνητα.

Ένα άλλο πλεονέκτημα είναι η δυνατότητα εκτέλεσης «μεταναστεύσεων» για τη διάδοση αλλαγών που γίνονται στη βάση δεδομένων, όπως η δημιουργία ενός νέου σχήματος, η προσθήκη νέων πινάκων ή η ενημέρωση της δομής των υπαρχόντων πινάκων, παρέχοντας με αυτό τον τρόπο ευέλικτο τρόπο για τη δημιουργία της βάσης δεδομένων. Απλώς δηλώνουμε τις δομές μοντέλων που αντιπροσωπεύουν τους πίνακες , όπως κάναμε παραπάνω για τους χρήστες και τα σπίτια, και στη συνέχεια χρησιμοποιούμε τη μέθοδο AutoMigrate() της GORM για τη δημιουργία του σχήματος.

```
    //creating this tables to the database
    Db.AutoMigrate(&UsersTable{},
        &HousesTable{},
        &RoomsTable{},
        &DevicesTable{},
        &PermissionsTable{},
        &DeviceTypesTable{},
        &BrandsTable{},
        &JobsTable{},
        &DeviceUseTable{},
        &UserGroupsTable{},
        &AttributesTable{},
        &DeviceModelsTable{},
        &ProtocolsTable{},
        &CommandsTable{})

}
```

Εικόνα 4.24 : Δημιουργία πινάκων οντοτήτων με GORM

Σε περίπτωση που χρειαστεί να αλλάξουμε κάτι σε κάποιον πίνακα, απλώς τον διαγράφουμε και ξανακάνουμε AutoMigrate() τον νέο πίνακα με τις αλλαγές και το σχήμα ενημερώνεται κατευθείαν.

Ένα άλλο κύριο χαρακτηριστικό του GORM είναι η διεπαφή του CRUD (Create, Read, Update, Delete), η οποία επιτρέπει την εκτέλεση λειτουργιών Δημιουργίας (Create), Ανάγνωσης (Read), Ενημέρωσης (Update) και Διαγραφής (Delete) με ελάχιστες γνώσεις SQL.

Για παράδειγμα, η δημιουργία μιας νέας εγγραφής χρήστη μπορεί να γίνει κατευθείαν ως εξής :

```
newUser := UsersTable{
        Username:  "dimitra",
        Password:  "secretpassword",
        Email:     "dimitra@example.com",
        Country:   "Greece",
        Timezone:  "EET",
        Project:   "EHS",

    }

    // Create the record in the database
    result := db.Create(&newUser)
    if result.Error != nil {
        fmt.Println("Error inserting user:", result.Error)
        return
    }
```

Εικόνα 4.25: : Δημιουργία εγγραφής νέου χρήστη με GORM

χωρίς να χρειάζονται οι δηλώσεις SQL που είναι πιο περίπλοκες και απαιτούν δέσιμο των παραμέτρων.
Επίσης, με τη μέθοδο Create() μπορούμε να ορίσουμε έτσι ώστε ο κωδικός πρόσβασης κάθε νέου χρήστη που εγγράφεται να αποθηκεύεται στη βάση μας κωδικοποιημένος, προκειμένου να μην υπάρχει η δυνατότητα από κανέναν τρίτο να τον υποκλέψει.

```go
func (u *UsersTable) BeforeCreate(tx *gorm.DB) (err error) {

    //Generating a hashed password using bcrypt hashing algorithm
    tempPassword, err := bcrypt.GenerateFromPassword([]byte(u.Password), 14)
    //Converting the hashed byte slice back to a string and assigns it to the Password
    u.Password = string(tempPassword)

    return
}
```

Εικόνα 4.26 : Κωδικοποίηση password κατά τη δημιουργία νέου χρήστη

Ακόμη, μετά την εισαγωγή εγγραφών στους πίνακες, μπορούμε να χρησιμοποιήσουμε τη μέθοδο Where() της GORM για να καθορίσουμε κριτήρια φιλτραρίσματος και να τη συνδυάσουμε με τη μέθοδο First() για την ανάκτηση της πρώτης εγγραφής που αντιστοιχίζει τα κριτήρια ή τη μέθοδο Find() για να ανακτήσουμε όλες τις αντίστοιχες εγγραφές.

```go
//For a specific house
var house HousesTable
if err := db.Where("name = ?", "example_house").First(&house).Error; err != nil {
    fmt.Println("Error fetching house:", err)
    return
}

//For all houses
var houses []HousesTable
if err := db.Where("typehouse = ?", "villa").Find(&houses).Error; err != nil {
    fmt.Println("Error fetching houses:", err)
    return
}
```

Εικόνα 4.27 : Φιλτράρισμα σπιτιών με συγκεκριμένα κριτήρια με GORM

Για την επικοινωνία του χρήστη με το σύστημά της GORM χρησιμοποιούμε τα endpoints. Τα endpoints χρησιμεύουν ως διεπαφές σε ένα API (Application Programming Interface – Διασύνδεση Προγραμματισμού Εφαρμογών) που επιτρέπουν την ανταλλαγή δεδομένων ανάμεσα στον χρήστη και στο σύστημα και είναι σχεδιασμένα με τη βοήθεια του Gin Framework. Τα endpoints λειτουργούν ως συγκεκριμένες διευθύνσεις URL που εκτελούν προκαθορισμένες λειτουργίες, όπως η ανάκτηση, η προσθήκη, η ενημέρωση και η διαγραφή δεδομένων. Τα δεδομένα τα οποία καταχωρούν οι χρήστες στην εφαρμογή αποστέλλονται σε έναν server και στέλνονται ξανά πίσω σε εκείνους με τις απαντήσεις και τα αποτελέσματα που επιθυμούν.

Παρακάτω παρουσιάζονται όλα τα δυνατά routes που αναγνωρίζει ο server μας και που μπορούν να εκτελεστούν για κάθε πίνακα της PostgreSQL βάσης μας :

1. **UsersTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/user/create”	POST	Δημιουργεί νέο χρήστη. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.

“/user/delete/:id”	DELETE	Διαγράφει τον χρήστη που έχει το id που δίνεται.
“/user/update/:id”	PUT	Ενημερώνει τα στοιχεία του χρήστη που έχει το id που δίνεται.
“/user/get/:userId”	GET	Επιστρέφει όλες τις πληροφορίες του χρήστη που έχει το id που δίνεται
“/user/getall”	GET	Επιστρέφει όλους τους χρήστες που είναι εγγεγραμμένοι στη βάση.
“/user/getusergroup/:userId”	GET	Επιστρέφει το user group στο οποίο ανήκει ο χρήστης που έχει το id που δίνεται.
“/user/getallhouses/:userID”	GET	Επιστρέφει όλα τα σπίτια με τα οποία είναι συνδεδεμένος ο χρήστης που έχει το id που δίνεται.
“/user/getallrooms/:userId”	GET	Επιστρέφει όλα τα δωμάτιά με τα οποία είναι συνδεδεμένος ο χρήστης που έχει το id που δίνεται.
“/user/getallhousesandrooms/:userID”	GET	Επιστρέφει όλα τα σπίτια με τα δωμάτιά τους με τα οποία είναι συνδεδεμένος ο χρήστης που έχει το id που δίνεται.

Πίνακας 4.15 : Routes στην οντότητα Users

2. **UserGroupsTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/usergroup/create”	POST	Δημιουργεί νέα κατηγορία χρηστών. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/usergroup/delete/:id”	DELETE	Διαγράφει την κατηγορία χρηστών που έχει το id που δίνεται.
“/usergroup/update/:id”	PUT	Ενημερώνει τα στοιχεία της κατηγορίας χρηστών που έχει το id που δίνεται.
“/usergroup/get/:usergroupId”	GET	Επιστρέφει όλες τις πληροφορίες της κατηγορίας

		χρηστών που έχει το id που δίνεται.
“/usergroup/getall”	GET	Επιστρέφει όλες τις κατηγορίες χρηστών που είναι αποθηκευμένες στη βάση.
“/usergroup/getallusers/:usergroupId”	GET	Επιστρέφει όλους τους users που ανήκουν στο user group που έχει id που δίνεται.
“/usergroup/addpermission”	POST	Προσθέτει μια άδεια σε μια κατηγορία χρηστών/Κάνει μια νέα σύνδεση κατηγορίας χρηστών-άδειας. Τα id της κατηγορίας χρηστών και της άδειας που πρέπει να συνδεθούν δηλώνονται στις παραμέτρους.
“/usergroup/getallpermissions/:usergroupId”	GET	Επιστρέφει όλες τις άδειες με τις οποίες είναι συνδεδεμένη (στην οποία ανήκουν) η κατηγορία χρηστών που έχει το id που δίνεται.

Πίνακας 4.16 : Routes στην οντότητα UserGroups

3. **PermissionsTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/permission/create”	POST	Δημιουργεί νέα άδεια. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/permission/delete/:id”	DELETE	Διαγράφει την άδεια που έχει το id που δίνεται.
“/permission/update/:id”	PUT	Ενημερώνει τα στοιχεία της άδειας που έχει το id που δίνεται.
“/permission/get/:permissionId”	GET	Επιστρέφει όλες τις πληροφορίες της άδειας που έχει το id που δίνεται

“/permission/getall”	GET	Επιστρέφει όλες τις άδειες που είναι αποθηκευμένες στη βάση.
“/permission/getallusergroups/:permissionID”	GET	Επιστρέφει όλα τα user groups στα οποίο ανήκει η άδεια που έχει το id που δίνεται.

Πίνακας 4.17 : Routes στην οντότητα Permissions

4. **HousesTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/house/create”	POST	Δημιουργεί νέο σπίτι. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/house/delete/:id”	DELETE	Διαγράφει το σπίτι που έχει το id που δίνεται.
“/house/update/:id”	PUT	Ενημερώνει τα στοιχεία του σπιτιού που έχει το id που δίνεται.
“/house/get/:houseId”	GET	Επιστρέφει όλες τις πληροφορίες του σπιτιού που έχει το id που δίνεται
“/house/getall”	GET	Επιστρέφει όλα τα σπίτια που είναι αποθηκευμένα στη βάση.
“/house/getallrooms/:houseId”	GET	Επιστρέφει όλα τα δωμάτια τα οποία βρίσκονται στο σπίτι που έχει το id που δίνεται.
“/house/getalldevices/:houseId”	GET	Επιστρέφει όλες τις συσκευές που ανήκουν στο σπίτι που έχει το id που δίνεται.
“/house/getalldevicesandrooms/:houseId”	GET	Επιστρέφει όλα τα δωμάτια μαζί με τις συσκευές τους , καθώς και τα συσκευές που είναι στο σπίτι αλλά δεν ανήκουν σε κάποιο δωμάτιο, τα οποία βρίσκονται στο σπίτι που έχει το id που δίνεται.
“/house/adduser”	POST	Προσθέτει νέο χρήστη σε κάποιο σπίτι . Κάνει μια νέα σύνδεση χρήστη-σπιτιού. Τα id του χρήστη και του σπιτιού που πρέπει να συνδεθούν

		δηλώνονται στις παραμέτρους.
“/house/getallusers/:houseId”	GET	Επιστρέφει όλους τους χρήστες που ανήκουν στο σπίτιπου έχει το id που δίνεται.

Πίνακας 4. 18 : Routes στην οντότητα Houses

5. **RoomsTable**

Endpoint	Μέθοδος	Τι κάνει
“/room/create”	POST	Δημιουργεί νέο δωμάτιο. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/room/delete/:id”	DELETE	Διαγράφει το δωμάτιο που έχει το id που δίνεται.
“/room/update/:id”	PUT	Ενημερώνει τα στοιχεία του δωματίου που έχει το id που δίνεται.
“/room/get/:roomId”	GET	Επιστρέφει όλες της πληροφορίες του δωματίου που έχει το id που δίνεται
“/room/getall”	GET	Επιστρέφει όλα τα δωμάτια που είναι αποθηκευμένα στη βάση.
“/room/getalldevices/:roomId”	GET	Επιστρέφει όλες τις συσκευές που βρίσκονται στο δωμάτιο που έχει το id που δίνεται.
“/room/gethouse/:roomId”	GET	Επιστρέφει το σπίτι στο οποίο ανήκει το δωμάτιο που έχει το id που δίνεται.

Πίνακας 4.19 : Routes στην οντότητα Rooms

6. **DeviceUseTable**

Endpoint	Μέθοδος	Τι κάνει
“/deviceuse/create”	POST	Δημιουργεί νέα χρήση συσκευής. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/deviceuse/delete/:id”	DELETE	Διαγράφει τη χρήση συσκευής που έχει το id που δίνεται.

“/deviceuse/update/:id”	PUT	Ενημερώνει τα στοιχεία της χρήσης συσκευής που έχει το id που δίνεται.
“/deviceuse/get/:deviceuseId”	GET	Επιστρέφει όλες της πληροφορίες για τη χρήση συσκευής που έχει το id που δίνεται
“/deviceuse/getall”	GET	Επιστρέφει όλες τις χρήσεις συσκευής που είναι αποθηκευμένες στη βάση.
“/deviceuse/getalldevices/:deviceuseId”	GET	Επιστρέφει όλες τις συσκευές που έχω χρήση τη χρήση συσκευής με το id που δίνεται.

Πίνακας 4.20 : Routes στην οντότητα DeviceUse

7. **BrandsTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/brand/create”	POST	Δημιουργεί νέα μάρκα συσκευής. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/brand/delete/:id”	DELETE	Διαγράφει τη μάρκα που έχει το id που δίνεται.
“/brand/update/:id”	PUT	Ενημερώνει τα στοιχεία της μάρκας που έχει το id που δίνεται.
“/brand/get/:brandId”	GET	Επιστρέφει όλες της πληροφορίες της μάρκας που έχει το id που δίνεται
“/brand/getall”	GET	Επιστρέφει όλες τις μάρκες που είναι αποθηκευμένες στη βάση.
“/brand/getalldevicemodels/:brandId”	GET	Επιστρέφει όλα τα μοντέλα συσκευών τα οποία ανήκουν στη μάρκα που έχει το id που δίνεται.

Πίνακας 4.21: Routes στην οντότητα Brands

8. **DeviceTypesTable**

Endpoint	**Μέθοδος**	**Τι κάνει**

“/devicetype/create”	POST	Δημιουργεί νέο τύπο συσκευής. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/devicetype/delete/:id”	DELETE	Διαγράφει τον τύπο συσκευής που έχει το id που δίνεται.
“/devicetype/update/:id”	PUT	Ενημερώνει τα στοιχεία του τύπου συσκευής που έχει το id που δίνεται.
“/devicetype/get/:devicetypeId”	GET	Επιστρέφει όλες της πληροφορίες του τύπου συσκευής που έχει το id που δίνεται
“/devicetype/getall”	GET	Επιστρέφει όλους τους τύπους συσκευών που είναι αποθηκευμένοι στη βάση.
“/devicetype/getalldevicemodels/:devicetypeId”	GET	Επιστρέφει όλα τα μοντέλα συσκευών τα οποία είναι του τύπου που έχει το id που δίνεται.

Πίνακας 4.22 : Routes στην οντότητα DeviceTypes

9. **DeviceModelsTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/devicemodel/create”	POST	Δημιουργεί νέο μοντέλο συσκευής. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/devicemodel/delete/:id”	DELETE	Διαγράφει το μοντέλο συσκευής που έχει το id που δίνεται.
“/devicemodel/update/:id”	PUT	Ενημερώνει τα στοιχεία του μοντέλου συσκευής που έχει το id που δίνεται.
“/devicemodel/get/:devicemodelId”	GET	Επιστρέφει όλες της πληροφορίες για το μοντέλο συσκευής που έχει το id που δίνεται
“/devicemodel/getall”	GET	Επιστρέφει όλα τα μοντέλα συσκευής που είναι αποθηκευμένα στη βάση.

“/devicemodel/getalldevices/:devicemodelId”	GET	Επιστρέφει όλες τις συσκευές οι οποίες ανήκουν στο μοντέλο που έχει το id που δίνεται.
“/devicemodel/getcommands/:devicemodelId”	GET	Επιστρέφει όλες τις εντολές που υποστηρίζει το μοντέλο συσκευής που έχει το id που δίνεται.
“/devicemodel/getdevicetype/:devicemodelId”	GET	Επιστρέφει τον τύπο της συσκευής που είναι το μοντέλο που έχει το id που δίνεται.
“/devicemodel/getbrand/:devicemodelId”	GET	Επιστρέφει τη μάρκα στην οποία ανήκει το μοντέλο συσκευής που έχει το id που δίνεται.
“/devicemodel/addprotocol”	POST	Επιστρέφει όλα τα user groups στα οποίο ανήκει η άδεια που έχει το id που δίνεται/Κάνει μια νέα σύνδεση μοντέλου συσκευής-πρωτοκόλλου. Τα id του μοντέλου συσκευής και του πρωτοκόλλου που πρέπει να συνδεθούν δηλώνονται στις παραμέτρους.
“/devicemodel/getallprotocols/:devicemodelId”	GET	Επιστρέφει όλα τα πρωτόκολλα επικοινωνίας που υποστηρίζει το μονέτλο συσκευής που έχει το id που δίνεται.

Πίνακας 4.23 : Routes στην οντότητα DeviceModels

10. **DevicesTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/device/create”	POST	Δημιουργεί νέα συσκευή. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/device/delete/:id”	DELETE	Διαγράφει τη συσκευή που έχει το id που δίνεται.
“/device/update/:id”	PUT	Ενημερώνει τα στοιχεία της συσκευής που έχει το id που δίνεται.
“/device/get/:deviceId”	GET	Επιστρέφει όλες της πληροφορίες της συσκευής που έχει το id που δίνεται

“/device/getall”	GET	Επιστρέφει όλες τις συσκευές που είναι αποθηκευμένες στη βάση.
“/device/getattributes/:deviceId”	GET	Επιστρέφει όλα τα δυναμικά χαρακτηριστικά της συσκευής που έχει το id που δίνεται.
“/device/gethouse/:deviceId”	GET	Επιστρέφει το σπίτι στα οποίο ανήκει η συσκευή που έχει το id που δίνεται.
“/device/getroom/:deviceId”	GET	Επιστρέφει το δωμάτιο στα οποίο ανήκει η συσκευή που έχει το id που δίνεται. Μπορεί να επιστρέψει και κενό, εάν μια συσκευή είναι συνδεδεμένη με ολόκληρο τ σπίτι και όχι με συγκεκριμένο δωμάτιο.
“/device/getdeviceuse/:deviceId”	GET	Επιστρέφει τη χρήση της συσκευής που έχει το id που δίνεται.
“/device/addprotocol”	POST	Προσθέτει νέο πρωτόκολλο επικοινωνίας που μπορεί να υποστηρίξει η συγκεκριμένη συσκευή/ Κάνει μια νέα σύνδεση συσκευής-πρωτοκόλλου. Τα id της συσκευής και του πρωτοκόλλου που πρέπει να συνδεθούν δηλώνονται στις παραμέτρους.
“/device/getallprotocols/:deviceId”	GET	Επιστρέφει όλα τα πρωτόκολλα που μπορεί ν υποστηρίξει η συσκευή που έχει το id που δίνεται.
“/device/getalljobs/:deviceId”	GET	Επιστρέφει όλες τις προγραμματισμένες εργασίες στις οποίες συμμετέχει η συσκευή που έχει το id που δίνεται.

Πίνακας 4.24 : Routes στην οντότητα Devices

11. **AttributesTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/attribute/create”	POST	Δημιουργεί νέο χαρακτηριστικό για συσκευές. Οι πληροφορίες που πρέπει να

		καταχωρηθούν δηλώνονται στις παραμέτρους.
“/attribute/delete/:id”	DELETE	Διαγράφει το χαρακτηριστικό που έχει το id που δίνεται.
“/attribute/update/:id”	PUT	Ενημερώνει τα στοιχεία του χαρακτηριστικού που έχει το id που δίνεται.
“/attribute/get/:attributeId”	GET	Επιστρέφει όλες της πληροφορίες για το χαρακτηριστικό που έχει το id που δίνεται
“/attribute/getall”	GET	Επιστρέφει όλα τα χαρακτηριστικά που είναι αποθηκευμένα στη βάση.
“/attribute/getdevice/:attributeId”	GET	Επιστρέφει τη συσκευή η οποία διαθέτει το χαρακτηριστικό που έχει το id που δίνεται.

Πίνακας 4.25 : Routes στην οντότητα Attributes

12. **ProtocolsTable**

Endpoint	**Μέθοδος**	**Τι κάνει**
“/protocol/create”	POST	Δημιουργεί νέο πρωτόκολλο επικοινωνίας. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/protocol/delete/:id”	DELETE	Διαγράφει το πρωτόκολλο που έχει το id που δίνεται.
“/protocol/update/:id”	PUT	Ενημερώνει τα στοιχεία του πρωτοκόλλου που έχει το id που δίνεται.
“/protocol/get/:protocolId”	GET	Επιστρέφει όλες της πληροφορίες για το πρωτόκολλο που έχει το id που δίνεται
“/protocol/getall”	GET	Επιστρέφει όλα τα πρωτόκολλα που είναι αποθηκευμένα στη βάση.
“/protocol/getalldevices/:protocolId”	GET	Επιστρέφει όλες τις συσκευές που υποστηρίζουν το πρωτόκολλο που έχει το id που δίνεται.
“/protocol/getalldevicemodels/:protocolId”	GET	Επιστρέφει όλα τα μοντέλα συσκευών που υποστηρίζουν το

		πρωτόκολλο που έχει το id που δίνεται.

Πίνακας 4.26 : Routes στην οντότητα Protocols

13. **CommandsTable**

Endpoint	Μέθοδος	Τι κάνει
“/command/create”	POST	Δημιουργεί νέο σύνολο εντολών. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/command/delete/:id”	DELETE	Διαγράφει το σύνολο εντολών που έχει το id που δίνεται.
“/command/update/:id”	PUT	Ενημερώνει τα στοιχεία του συνόλου εντολών που έχει το id που δίνεται.
“/command/get/:commandId”	GET	Επιστρέφει όλες της πληροφορίες για το σύνολο εντολών που έχει το id που δίνεται
“/command/getall”	GET	Επιστρέφει όλα τα σύνολα εντολών που είναι αποθηκευμένα στη βάση.
“/command/getdevicemodel/:commandId”	GET	Επιστρέφει το μοντέλο συσκευών στο οποίο ανήκει το σύνολο εντολών που έχει το id που δίνεται.

Πίνακας 4.27 : Routes στην οντότητα Commands

14. **JobsTable**

Endpoint	Μέθοδος	Τι κάνει
“/job/create”	POST	Δημιουργεί νέα προγραμματισμένη εργασία. Οι πληροφορίες που πρέπει να καταχωρηθούν δηλώνονται στις παραμέτρους.
“/job/delete/:id”	DELETE	Διαγράφει την εργασία που έχει το id που δίνεται.
“/job/update/:id”	PUT	Ενημερώνει τα στοιχεία της εργασίας που έχει το id που δίνεται.

“/job/get/:jobId”	GET	Επιστρέφει όλες της πληροφορίες της εργασίας που έχει το id που δίνεται.
“/job/getall”	GET	Επιστρέφει όλες τις εργασίες που είναι αποθηκευμένες στη βάση.
“/job/adddevice”	POST	Προσθέτει νέα συσκευή που θα συμμετέχει σε μια συγκεκριμένη προγραμματισμένη εργασία. Κάνει μια νέα σύνδεση συσκευής-προγραμματισμένης εργασίας. Τα id συσκευής και της προγραμματισμένης εργασίας που πρέπει να συνδεθούν δηλώνονται στις παραμέτρους.
“/job/getalldevices/:jobId”	GET	Επιστρέφει όλες τις συσκευές που συμμετέχουν στην εργασία που έχει το id που δίνεται.

Πίνακας 4.28 : Routes στην οντότητα Jobs

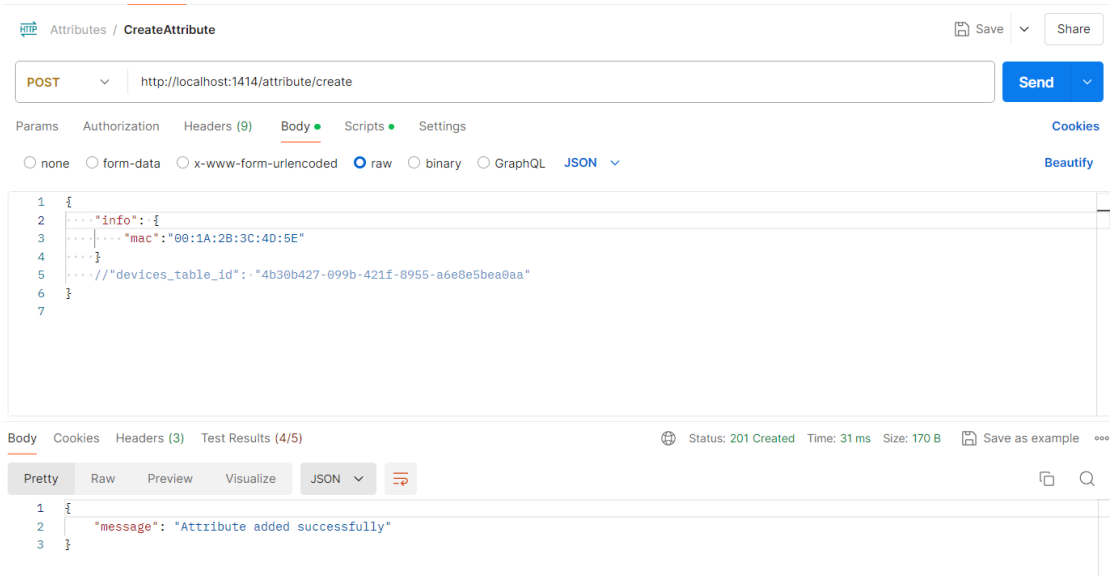
Τα routes στον κώδικα, για παράδειγμα για τα UserGroups ορίζονται όπως φαίνεται στην εικόνα παρακάτω:

```
//UserGroups
r.POST("/usergroup/create", func(c *gin.Context) {
    usergroups.CreateUserGroup(c)
})

r.DELETE("/usergroup/delete/:id", func(c *gin.Context) {
    usergroups.DeleteUserGroup(c)
})

r.PUT("/usergroup/update/:id", func(c *gin.Context) {
    usergroups.UpdateUserGroup(c)
})

r.GET("/usergroup/get/:usergroupId", func(c *gin.Context) {
    usergroups.GetAUserGroup(c)
})

r.GET("/usergroup/getall", func(c *gin.Context) {
    usergroups.GetAllUserGroups(c)
})

r.GET("/usergroup/getallusers/:usergroupId", func(c *gin.Context) {
    usergroups.GetAllUsersForUserGroup(c)
})

r.POST("/usergroup/addpermission", func(c *gin.Context) {
    usergroups.AddPermissionUserGroup(c)
})

r.GET("/usergroup/getallpermissions/:usergroupId", func(c *gin.Context) {
    usergroups.GetAllPermissionsForUserGroup(c)
})
```

Εικόνα 4.28 : Routes για τον πίνακα των UserGroups

Και παρόμοια και για τις υπόλοιπες οντότητες.

Παράδειγμα εφαρμογής αποτελεί η δημιουργία ενός νέου δυναμικού χαρακτηριστικού για τις συσκευές, η οποία μπορεί να πραγματοποιηθεί και να δοκιμαστεί με τη βοήθεια του Postman,όπως θα αναλυθεί εκτενώς και στην επόμενη ενότητα, με τη μέθοδο POST όπως φαίνεται παρακάτω :



Εικόνα 4.29 : Δημιουργία νέας εγγραφής χαρακτηριστικού με Postman

Η παραπάνω ενέργεια στη γραμμή εντολών εμφανίζεται ως εξής :

```
[207.206ms] [rows:1] INSERT INTO "attributes_tables" ("info","devices_table_id") VALUES ('map[]',NULL) RETURNI
NG "id"
[GIN] 2024/07/02 - 12:01:27 | 201 |  226.856113ms |             ::1 | POST     "/attribute/create"
```

Εικόνα 4.30 : Συσχέτιση GORM endpoints με SQL ερωτήματα

Είναι ξεκάθαρο ότι πως πίσω από την GORM , με την οποία δουλέψαμε, «τρέχει» η SQL, όπως αναφέρθηκε και παραπάνω. Με αυτό τον τρόπο, τα πιο απλά queries , όπως η δημιουργία νέας εγγραφής ή η ανάκτηση πληροφοριών, πραγματοποιούνται γρηγορότερα και ευκολότερα με τη βιβλιοθήκη GORM και με το Gin Framework της με το οποίο στήσαμε το API μας, παρά με την εκτέλεση κλασσικών SQL ερωτημάτων.

4.3. ΑΣΦΑΛΕΙΑ ΚΑΙ ΙΔΙΩΤΙΚΟΤΗΤΑ

Η διατήρηση της ασφάλειας και της εμπιστευτικότητας των προσωπικών δεδομένων των χρηστών από παράνομη πρόσβαση και αποκάλυψη από εξωτερικούς διακομιστές και μη εξουσιοδοτημένα πρόσωπα αποτελεί βασικό κριτήριο για τη δημιουργία της βάσης μας.

Το 1ο βήμα που εφαρμόσαμε για αυτό το σκοπό είναι οι ιδιότητες της κρυπτογράφησης στα ευαίσθητα προσωπικά δεδομένα κατά την αποθήκευσή τους στη βάση, ενώ τα μη ευαίσθητα δεδομένα διατηρούνται ακρυπτογράφητα. Το κύριο δεδομένο το οποίο δεν θέλουμε να διαρρεύσει σε κανέναν μη εξουσιοδοτημένο χρήστη, ακόμα και στους διαχειριστές της βάσης, είναι ο κωδικός πρόσβασης που χρησιμοποιεί ο εκάστοτε εγγεγραμμένος χρήστης για να συνδεθεί

στην εφαρμογή μας. Για το σκοπό αυτό, έχουμε μεριμνήσει με κατάλληλη συνάρτηση, η οποία φαίνεται στον κώδικα παρακάτω, έτσι ώστε να καλείται αυτόματα από τη GORM πριν δημιουργηθεί μια νέα εγγραφή στον πίνακα βάσης δεδομένων UsersTable, να κρυπτογραφείται ο κωδικός τους (password) , με τη βοήθεια του αλγόριθμου κρυπτογράφησης bcrypt. Ο αλγόριθμος αυτός μετατρέπει τον κωδικό πρόσβασης από συμβολοσειρά σε ένα slice byte . Μετά τη δημιουργία του κρυπτογραφημένου κωδικού, μετατρέπει το slice byte του κρυπτογραφημένου κωδικού (tempPassword) πίσω σε μια συμβολοσειρά και τον αναθέτει στο πεδίο Password του struct UsersTable. Αυτό αντικαθιστά αποτελεσματικά τον αρχικό κωδικό πρόσβασης με την κρυπτογραφημένη του έκδοση πριν από την αποθήκευσή του στη βάση δεδομένων.

```go
func (u *UsersTable) BeforeCreate(tx *gorm.DB) (err error) {

    //Generating a hashed password using bcrypt hashing algorithm
    tempPassword, err := bcrypt.GenerateFromPassword([]byte(u.Password), 14)
    //Converting the hashed byte slice back to a string and assigns it to the Password
    u.Password = string(tempPassword)

    return
}
```

Εικόνα 4.31 : Κωδικοποίηση password κατά τη δημιουργία νέου χρήστη

Η εμπιστευτικότητα των δεδομένων διασφαλίζεται επίσης με τη χρήση ενός API Gateway. Το σύστημα αυτό δεν καλύπτεται σε αυτή τη διπλωματική, αποτελεί, ωστόσο, προέκταση της βάσης δεδομένων μας και συνδέεται με αυτήν, συνεπώς αξίζει να αναφερθεί.

Ένα API Gateway δέχεται αιτήματα API από έναν πελάτη, τα επεξεργάζεται βάσει καθορισμένων πολιτικών, τα κατευθύνει στις κατάλληλες υπηρεσίες και συνδυάζει τις απαντήσεις για μια απλοποιημένη εμπειρία χρήστη. Συνήθως, διαχειρίζεται ένα αίτημα καλώντας πολλαπλές μικροϋπηρεσίες και συγκεντρώνοντας τα αποτελέσματα [94].

Μία πύλη API είναι ένα κρίσιμο στοιχείο στο οικοσύστημα των API, το οποίο λειτουργεί ως κεντρικό σημείο εισόδου για αιτήματα API. Λαμβάνει εισερχόμενα αιτήματα, τα δρομολογεί στις κατάλληλες υπηρεσίες υποδομής, και παρέχει διάφορες λειτουργίες όπως ισορροπία φορτίου, μετασχηματισμό αιτημάτων και απαντήσεων, προσωρινή αποθήκευση, και εφαρμογή ασφάλειας.

Μία πύλη API παίζει καθοριστικό ρόλο στην ενίσχυση της αξιοπιστίας και της απόδοσης των API. Διαχειρίζεται τη ροή των αιτημάτων μεταξύ πελατών και υπηρεσιών υποδομής, εξασφαλίζοντας απρόσκοπτη επικοινωνία και βελτιστοποιώντας τη χρήση των πόρων. Όταν ένας πελάτης ξεκινά ένα αίτημα, η πύλη API είναι το αρχικό σημείο επαφής, αναλαμβάνοντας τη δρομολόγηση των αιτημάτων, την ισορροπία φορτίου και ακόμη και τους μετασχηματισμούς αιτημάτων και απαντήσεων. Αυτή η λειτουργία δρομολόγησης κατευθύνει τα εισερχόμενα αιτήματα στον κατάλληλο διακομιστή υποδομής. Ταυτόχρονα, η ισορροπία φορτίου εξασφαλίζει ότι τα αιτήματα κατανέμονται ομοιόμορφα μεταξύ πολλών διακομιστών ή υποδομών, βελτιώνοντας την απόδοση και την ανοχή σε σφάλματα.

Οι πύλες API παρέχουν ,επίσης, ένα ασφαλές επίπεδο με την επιβολή αυθεντικοποίησης και εξουσιοδότησης, προστατεύοντας έναντι επιθέσεων και απειλών και επιτρέποντας την κρυπτογράφηση. Έχουν ,ακόμη, άλλους ρόλους στη διαχείριση της κίνησης, καθώς μπορούν να τροποποιήσουν παραμέτρους αιτημάτων ή φορτία και να μετασχηματίσουν απαντήσεις όπως χρειάζεται, προσαρμόζοντας τις μορφές δεδομένων μεταξύ πελατών και API.

Η διαχείριση API περιλαμβάνει διάφορες λειτουργίες, όπως ο σχεδιασμός, η δημιουργία, η παρακολούθηση, η ασφάλεια, τα αναλυτικά δεδομένα και η δέσμευση των προγραμματιστών. Η πύλη API είναι αναπόσπαστο μέρος αυτών των λειτουργιών, εξασφαλίζοντας ότι τα API είναι ασφαλή, αποδοτικά και προσβάσιμα. Τελικά, η πύλη API χρησιμοποιείται για την ενίσχυση της ασφάλειας και της ισορροπίας φορτίου, διευκολύνοντας την κλιμάκωση και απλοποιώντας τις αλληλεπιδράσεις των πελατών με τα backend API, καθιστώντας την ένα αναντικατάστατο στοιχείο στη σύγχρονη αρχιτεκτονική λογισμικού.

Από τα παραπάνω γίνεται κατανοητό πως η αξιοποίηση μιας πύλης API είναι κρίσιμη για τη βελτιστοποίηση της επικοινωνίας μεταξύ της εφαρμογής μας και των υπηρεσιών υποδομής backend, προκειμένου να βελτιωθεί η ορατότητα, η ακρίβεια και οι εμπειρίες των πελατών. Η εφαρμογή μας μπορεί να αξιοποιήσει μια πύλη API για να διευκολύνει [95]:

1. **Δρομολόγηση Αιτημάτων:** Η πύλη API δρομολογεί τα εισερχόμενα αιτήματα από την εφαρμογή στις κατάλληλες υπηρεσίες υποδομής, όπως πληροφορίες για τα σπίτια που έχει ένας χρήστης , προσθήκη νέου χρήστη σε ένα σπίτι κλπ
2. **Ισορροπία Φορτίου:** Κατανέμει την εισερχόμενη κίνηση σε πολλούς διακομιστές ή υποδομές για να εξασφαλίσει υψηλή διαθεσιμότητα και βελτιωμένη απόδοση, ακόμη και κατά τη διάρκεια περιόδων αιχμής.
3. **Επιβολή Ασφάλειας:** Η πύλη API επιβάλλει αυστηρές πολιτικές ασφαλείας, συμπεριλαμβανομένων της αυθεντικοποίησης και της εξουσιοδότησης, για την προστασία των δεδομένων των πελατών από μη εξουσιοδοτημένη πρόσβαση.
4. **Διαχείριση Κίνησης:** Μπορεί να τροποποιήσει παραμέτρους αιτημάτων, να χειριστεί τον μετασχηματισμό αιτημάτων και απαντήσεων και να διαχειριστεί τον περιορισμό ρυθμού για να εξασφαλίσει μια ομαλή και άμεση εμπειρία χρήστη.
5. **Προσωρινή Αποθήκευση:** Η πύλη μπορεί να αποθηκεύει στην προσωρινή μνήμη δεδομένα που χρησιμοποιούνται συχνά, μειώνοντας το φορτίο στους πίσω κατακερματιστές και επιταχύνοντας τους χρόνους απόκρισης.

Συνοπτικά, η πύλη API εξασφαλίζει ότι οι συναλλαγές των πελατών είναι ασφαλείς, άμεσες και υψηλά διαθέσιμες, ταυτόχρονα βελτιώνοντας τη χρήση των πόρων στην υποκείμενη υποδομή.

Συγκεκριμένα, στο κομμάτι της ασφάλειας και της ιδιωτικότητας που εξετάζουμε σε αυτό το κεφάλαιο, κάθε φορά που κάποιος χρήστης προσπαθεί να συνδεθεί στην εφαρμογή, γίνεται έλεγχος στα credentials του προκειμένου να διαπιστωθεί αν ο user αυτός έχει πρόσβαση στην εφαρμογή.

Το σύστημα λαμβάνει το username και το password που εισάγει ο χρήστης κατά το login του, και ψάχνει την αντιστοιχία στο Keycloak, στο οποίο είναι αποθηκευμένα τα username και τανpasswords όλων των χρηστών.

Το Keycloak είναι ένα εργαλείο διαχείρισης ταυτότητας και πρόσβασης ανοικτού κώδικα. Ως εργαλείο διαχείρισης ταυτότητας και πρόσβασης, διευκολύνει τη διαδικασία πιστοποίησης για εφαρμογές και υπηρεσίες πληροφορικής. Ο σκοπός ενός εργαλείου είναι να εξασφαλίσει ότι οι σωστοί άνθρωποι έχουν κατάλληλη πρόσβαση στους πόρους [96] .

Στη συνέχεια το Keycloak ,εφόσον επιβεβαιωθεί ότι ο χρήστης έχει δικαίωμα να έχει πρόσβαση, επιστρέφει ένα Token (ένα αντικείμενο το οποίο αντιπροσωπεύει το δικαίωμα να εκτελείται κάποια λειτουργία) αναγνωριστικό, έτσι ώστε να μην αποκαλυφθεί ο κωδικός του χρήστη, και με αυτό ο χρήστης έχει πρόσβαση σε όλα τα routes της εφαρμογής που έχουμε δημιουργήσει.

Η πύλη API δεν αναλύεται στην παρούσα διπλωματική, ωστόσο αναφέρεται ως συμπλήρωμα για τους προβληματισμούς ασφάλειας και ιδιωτικότητας.

4.4. ΣΥΛΛΟΓΗ ΚΑΙ ΑΠΟΘΗΚΕΥΣΗ ΔΕΔΟΜΕΝΩΝ ΑΠΟ ΤΙΣ ΣΥΣΚΕΥΕΣ

4.4.1. ΒΑΣΗ ΧΡΟΝΟΣΕΙΡΩΝ (TIMESERIES DATABASE)

Τα δεδομένα που προκύπτουν από τις συσκευές των έξυπνων σπιτιών αφορούν μετρήσεις από αισθητήρες και έξυπνες πρίζες, μαζί με κάποια χρονική σήμανση. Τα δεδομένα αυτά συγκεντρώνονται και καταγράφονται με την πάροδο του χρόνου, προκειμένου να εξαχθούν συμπεράσματα και να αναλυθούν σε χρονοσειρές. Για το σκοπό αυτό, όπως αναφέρθηκε και παραπάνω, είναι κατάλληλη μια timeseries database. Συγκεκριμένα , έχει χρησιμοποιηθεί μια Timescale βάση. Η υλοποίησή της δεν καλύπτεται σε αυτή τη διπλωματική, ωστόσο σε αυτό το κεφάλαιο παρουσιάζεται η λειτουργία της .

Στην παρακάτω εικόνα φαίνεται το διάγραμμα που απεικονίζει τη διαδικασία διαχείρισης των δεδομένων χρονοσειρών, μαζί με επεξήγηση για κάθε βήμα :

Εικόνα 4.32 : Διαχείριση δεδομένων χρονοσειρών [100]

1. **Αποτελεσματική συλλογή και αποθήκευση δεδομένων :** Τα δεδομένα συλλέγονται από διάφορες πηγές του IoT, όπως αισθητήρες ,έξυπνες πρίζες και έξυπνες συσκευές. Οι μετρήσεις αυτές περιλαμβάνουν πεδία όπως η θερμοκρασία, η υγρασία, η τάση, η κατανάλωση ενέργειας και, φυσικά ,η ακριβής χρονική σήμανση που έγινε η μέτρηση. Τα πεδία που μας δίνει η κάθε συσκευή τα γνωρίζουμε από πριν και φροντίζουμε να σχεδιάσουμε τη βάση χρονοσειρών μας με αυτά τα κατάλληλα

attributes σε κάθε οντότητα. Η βάση που επιλέγουμε υποστηρίζει διαχείριση και ανάκτηση μεγάλων ποσοτήτων δεδομένων.

2. **Συχνή αναζήτηση πρόσφατων ακατέργαστων δεδομένων :** Τα πρόσφατα ακατέργαστα δεδομένα αναζητούνται συχνά είτε σε πραγματικό χρόνο είτε σε τακτά χρονικά διαστήματα, όπως για παράδειγμα οι μετρήσεις που προέκυψαν από έναν αισθητήρα κατά τη διάρκεια μιας ημέρας. Αυτά στη συνέχεια αναλύονται για την ανίχνευση μοτίβων, τον εντοπισμό ανωμαλιών και τη λήψη αποφάσεων. Για παράδειγμα, μπορούν να χρησιμοποιηθούν για την εξαγωγή συμπερασμάτων σχετικά με την ενεργειακή κατανάλωση ανά ημέρα, εβδομάδα ή μήνα ή για εντοπισμό υπερκατανάλωσης σε μια χρονική περίοδο, προκειμένου να ληφθούν διορθωτικά μέτρα και αποφάσεις για την αποτελεσματικότερη διαχείριση της ενέργειας. Αυτό βοηθά στην παρακολούθηση των τρεχουσών συνθηκών και στη λήψη έγκαιρων αποφάσεων με βάση τα πιο πρόσφατα δεδομένα.

3. **Δημιουργία συγκεντρωτικών ιστορικών αναφορών:** Η συγκέντρωση των δεδομένων ανά μεγάλες χρονικές περιόδους συμβάλλει στη δημιουργία ιστορικών αναφορών. Αυτές οι αναφορές αποτελούν μια σύνοψη των δεδομένων και περιλαμβάνουν τη συνολική περίληψή τους, με μέσους όρους, μέγιστες και ελάχιστες τιμές κλπ. ανά ώρα, ημέρα ή μήνα. Έτσι, μειώνεται ο όγκος των δεδομένων που πρέπει να επεξεργαστούν για ιστορική ανάλυση, διατηρώντας παράλληλα τις βασικές πληροφορίες. Αυτό βοηθά στην κατανόηση των της συμπεριφοράς των δεδομένων χωρίς να απαιτείται η επεξεργασία όλου του όγκου τους.

4. **Αρχειοθέτηση παλαιότερων ακατέργαστων δεδομένων:** Τα παλαιότερα ακατέργαστα δεδομένα τα οποία δεν χρησιμοποιούνται συχνά και τα οποία έχουν αναλυθεί μεταφέρονται σε φθηνότερες πιο αργές λύσεις αποθήκευσης , προκειμένου να «ανοίξει» χώρος στη βάση μας για νέα δεδομένα .Τα δεδομένα αυτά κατηγοριοποιούνται , πχ ανά χρονικά διαστήματα, έτσι ώστε να είναι ευκολότερη η πρόσβαση σε αυτά στο μέλλον , αν χρειαστεί Αυτό βοηθά στη διαχείριση του κόστους αποθήκευσης, διατηρώντας , όμως, παράλληλα τα δεδομένα για ενδεχόμενη μελλοντική ανάλυση.

5. **Διαγραφή ακατέργαστων δεδομένων μετά από προκαθορισμένο χρονικό διάστημα:** Τα ακατέργαστα δεδομένα που θεωρούνται περιττά και δεν χρησιμοποιούνται διαγράφονται μετά από κάποιο χρονικό διάστημα , το οποίο καθορίζεται ανάλογα με την αξία των δεδομένων και τις ανάγκες αποθήκευσης. Αυτό βοηθάει στην εκκαθάριση του αποθηκευτικού χρόνου και στην αποφυγή προβλημάτων υπερφόρτωσης του συστήματος με παλιά δεδομένα, διατηρώντας το αποδοτικό και μειώνοντας το κόστος αποθήκευσης.

4.4.1.1. ΑΡΧΙΤΕΚΤΟΝΙΚΗ

Αυτός ο κύκλος επαναλαμβάνεται, διασφαλίζοντας την αποδοτική διαχείριση των δεδομένων χρονοσειρών, καλύπτοντας την ανάγκη για πρόσβαση σε δεδομένα τόσο σε πραγματικό χρόνο όσο και για ιστορική ανάλυση μακροπρόθεσμα, ενώ βελτιστοποιεί τους πόρους αποθήκευσης και επεξεργασίας.

Η Timescale βάση μας είναι , όπως έχει αναφερθεί, μια σχεσιακή βάση δεδομένων, οπότε αποθηκεύει δομημένα δεδομένα και έχει συγκεκριμένο σχήμα. Τα δεδομένα που έχουμε προέρχονται τόσο από μονοφασικούς όσο και από τριφασικούς μετρητές κατανάλωσης ηλεκτρικής ενέργειας, για αυτό το σχήμα μας διαθέτει έναν πίνακα (οντότητα) για την κάθε περίπτωση. Ο μονοφασικός πίνακας περιλαμβάνει στήλες τα χαρακτηριστικά όπως είναι η τάση του ηλεκτρικού ρεύματος , το ρεύμα, η ενεργό ισχύς , η φαινόμενη ισχύς , ο συντελεστής ισχύος, η συχνότητα του ηλεκτρικού ρεύματος κ.ά. Ο τριφασικός πίνακας περιλαμβάνει τα ίδια χαρακτηριστικά αλλά για κάθε φάση.

Και οι δύο πίνακες περιλαμβάνουν μια στήλη για το μοναδικό χαρακτηριστικό id του μετρητή από τον οποίο προέρχονται τα δεδομένα, καθώς και μια χρονοσήμανση timestamp για τη χρονική στιγμή της καταγραφής των δεδομένων. Με αυτό το id της συσκευής γίνεται η σύνδεση με τη PostgreSQL βάση μας που αναλύθηκε παραπάνω και συγκεκριμένα με τον πίνακα Devices και το μοναδικό αναγνωριστικό των εγγραφών του, έτσι ώστε να καθορίζεται σε ποια συσκευή ανήκει η κάθε μέτρηση.

Μερικά επιπλέον χαρακτηριστικά της βάσης χρονοσειρών μας παρουσιάζονται παρακάτω :

1. **Συμπίεση στηλών (columnar compression) :** Η βάση χρονοσειρών μας συγκεντρώνει πολύ μεγάλους όγκους δεδομένων και θέλουμε να είναι βέλτιστη και όσο το δυνατόν πιο αποδοτική. Για το λόγο αυτό έχουμε εφαρμόσει τόσο μέθοδο αποθήκευσης προσανατολισμένη σε γραμμές όσο και μέθοδο συμπίεσης στηλών, καθώς ανάλογα τις ανάγκες μας μας εξυπηρετεί καλύτερα πότε η μία και πότε η άλλη. Οι παραδοσιακές σχεσιακές βάσεις δεδομένων είναι ,συνήθως, προσανατολισμένες σε γραμμές, δηλαδή τα δεδομένα οργανώνονται κατά γραμμή και κάθε γραμμή να περιέχει όλα τα δεδομένα για μια συγκεκριμένη εγγραφή. Αυτή η προσέγγιση είναι κατάλληλη για εφαρμογές όπου λειτουργίες εγγραφής, ενημέρωσης και διαγραφής είναι συχνές και επικεντρώνονται σε μεμονωμένες γραμμές, όπως για παράδειγμα η ανάκτηση όλων των πληροφοριών για μια συγκεκριμένη μέτρηση που έγινε μια χρονική στιγμή και από κάποια συσκευή. Η εγγραφή αυτή είναι μια μόνο γραμμή στη βάση που ικανοποιεί αυτές τις προϋποθέσεις. Για να γίνει πιο επεκτάσιμη η βάση των χρονοσειρών μας , έχουμε χρησιμοποιήσει και κάποιον μηχανισμό συμπίεσης στηλών, ο οποίος επιτρέπει να μειώσουμε το μέγεθος των μεγάλων πινάκων έως και 10-20 φορές [99]. Οι βάσεις δεδομένων που είναι προσανατολισμένες σε στήλες (column-oriented) οργανώνουν τα δεδομένα κατά στήλη και είναι ιδανικές για εφαρμογές όπου οι λειτουργίες ανάγνωσης επικεντρώνονται σε πολλές γραμμές των ίδιων στηλών. Κάθε στήλη αποθηκεύει όλα τα δεδομένα για μια συγκεκριμένη ιδιότητα σε πολλές εγγραφές. Αυτή η τακτική είναι ιδανική για το σύστημά μας , όπου τα ερωτήματα συχνά περιλαμβάνουν αθροίσεις μετρήσεων και λειτουργίες σε πολλές εγγραφές με βάση κάποιο χαρακτηριστικό τους. Η αποθήκευση όλων των εγγραφών ανά χαρακτηριστικό βοηθάει στο να μη χρειάζεται να ανακτηθεί όλο το σύνολο δεδομένων για να γίνει κάποιος υπολογισμός αλλά να ανακτηθούν μόνο οι συγκεκριμένες στήλες που μας ενδιαφέρουν. Για παράδειγμα, με τη μέθοδο των γραμμών έχουμε έναν πίνακα στον οποίο αποθηκεύονται οι μετρήσεις και σαν στήλες έχουν τη χρονοσήμανση, το id της συσκευή και την τάση που μετριέται. Αυτός ο πίνακας θα έχει τόσες γραμμές όσα και τα δεδομένα μας, δηλαδή εκατομμύρια. Η ολόκληρη γραμμή για συγκεκριμένο id συσκευής και timestamp αποθηκεύεται συνεχόμενα, κάνοντας την ανάκτηση γρήγορη. Ως αποτέλεσμα, ερωτήματα του τύπου «Φέρε μου όλες τις πληροφορίες για τον μετρητή Χ σε ένα συγκεκριμένο δωμάτιο μέσα στη μέρα» θα είναι γρήγορα , καθώς απαιτούν να επιστραφούν ολόκληρες γραμμές .Αντιθέτως, με τη μέθοδο στηλών θα αποθηκευτούν

όλα τα ids μαζί, όλες οι τάσεις μαζί, όλες οι θερμοκρασίες μαζί και ούτω καθεξής, έτσι ώστε τα δεδομένα κάθε στήλης να αποθηκεύονται συνεχόμενα στο δίσκο. Αυτή η αρχιτεκτονική βάσης δεδομένων ευνοεί τα ερωτήματα του τύπου "Υπολόγισε το μέσο όρο της τάσης για μια συγκεκριμένη συσκευή τον τελευταίο μήνα για να ελέγξουμε ανωμαλίες.", καθώς θα επικεντρωθούμε σε μια μόνο στήλη (τάση) αλλά θα χρειαστεί να ανακτήσουμε αυτήν την πληροφορία από έναν μεγάλο αριθμό γραμμών που αντιστοιχούν στο χρονικό διάστημα που μας ενδιαφέρει.

2. **Προβολές (views) :** Οι προβολές είναι εικονικοί πίνακες που δημιουργούνται από σύνθετα ερωτήματα , τα οποία μπορούν να αποθηκευτούν και να χρησιμοποιηθούν ξανά κατευθείαν, καλώντας τα απλώς με το όνομά τους. Τα πλεονεκτήματα που προσφέρουν αφορούν τις παρακάτω λειτουργίες :

    - Ενοποίηση δεδομένων από πολλούς πίνακες : Μπορούν να συνδυάσουν δεδομένα από διαφορετικούς πίνακες, δημιουργώντας έναν συνολικό ενοποιημένο πίνακα. Στο σύστημά μας είναι αναγκαία η ένωση του πίνακα των συσκευών της PostgreSQL βάσης με τον πίνακα των μετρήσεων της Timescale βάσης με το αναγνωριστικό id της συσκευής, προκειμένου να καθοριστεί σε ποια συσκευή ανήκει η κάθε μέτρηση.

    - Απλοποίηση Πολύπλοκων Ερωτημάτων : Αντί να γράφουμε ένα πολύπλοκο ερώτημα κάθε φορά, δημιουργούμε ένα view και απλώς το καλούμε όποτε το χρειαζόμαστε. Για παράδειγμα , δημιουργηθούμε views για να συνοψίσουν τα δεδομένα μας και να υπολογίσουν τον μέσο όρο, το ελάχιστο και το μέγιστο για συγκεκριμένα χρονικά διαστήματα.

    - Διαχείριση και Φιλτράρισμα Δεδομένων : Βοηθούν στην κατηγοριοποίηση και την οργάνωση των δεδομένων, κάνοντας την αναζήτηση και την ανάλυση πιο αποδοτική. Φιλτράρουμε τα δεδομένα με βάση συγκεκριμένα κριτήρια, όπως το χρονικό διάστημα, και δημιουργούμε κάποιο view που θα περιέχει ,για παράδειγμα, τα δεδομένα όλων των μετρητών για την τελευταία εβδομάδα ή τις μετρήσεις μιας συγκεκριμένης μόνο συσκευής για μία μέρα.

    - Ευκολία Συντήρησης : Μπορούμε να κάνουμε αλλαγές στο σχήμα της βάσης δεδομένων μας χωρίς να επηρεαστούν τα ερωτήματα που βασίζονται σε αυτές. Για παράδειγμα, αν αλλάξουμε το όνομα μιας στήλης στον πρωτότυπο πίνακα, μπορούμε απλώς να ενημερώσουμε το view αντί να αλλάξουμε όλα τα ερωτήματα στις εφαρμογές μας.

    - Βελτιστοποίηση Απόδοσης : Βελτιστοποιούν τη διαδικασία ερωτημάτων, όταν πρόκειται για πολύπλοκα και επαναλαμβανόμενα ερωτήματα, επιτρέποντας την επανειλημμένη χρήση των αποτελεσμάτων, χωρίς να χρειάζεται να υπολογιστούν ξανά αυτά που ήδη έχουν υπολογιστεί , αλλά να προστίθενται απλώς τα νέα δεδομένα. Με αυτό τον τρόπο μειώνεται ο υπολογιστικός φόρτος.

4.4.1.2.ΜΕΘΟΔΟΙ ΑΝΑΚΤΗΣΗΣ ΔΕΔΟΜΕΝΩΝ

Όπως και στην κύρια βάση μας, ο χρήστης μπορεί να επικοινωνήσει με τη βάση χρονοσειρών μέσω κάποιου API.

Τα routes σε αυτή την περίπτωση είναι κυρίως Get και Delete , αφού τα δεδομένα προέρχονται αποκλειστικά από τις μετρήσεις των συσκευών και αποθηκεύονται αυτόματα στη βάση, οπότε ο χρήστης δε σχετίζεται με τις λειτουργίες Create και Update. Ό,τι έχει να κάνει με τις ίδιες τις συσκευές και τις πληροφορίες τους διαχειρίζεται από την PostgreSQL βάση.

Μερικά παραδείγματα είναι :

- **GET /device/26f531d5-3509-407e-aaa1-b2714d06d672/data**

  Επιστρέφει όλες τι μετρήσεις της συσκευής με το id που δίνεται

- **GET /device/ 26f531d5-3509-407e-aaa1-b2714d06d672/data?start_time=2024-07-06 T00:00:00Z&end_time=2024-07-06 T23:59:59Z**

  Επιστρέφει όλες τι μετρήσεις της συσκευής με το id που δίνεται αλλά για συγκεκριμένη χρονική περίοδο , για παράδειγμα εδώ όλες τις μετρήσεις της συσκευής για την ημέρα 6/7/2024.

- **GET /device/ 26f531d5-3509-407e-aaa1-b2714d06d672/data?start_time=2023-01-01 T00:00:00Z&end_time=2023-12-31 T23:59:59Z &metrics=avg,max,min**

  Με παρόμοιο τρόπο μπορούμε να εξάγουμε στατιστικά για μια συσκευή μέσα σε μια χρονική περίοδο, καθορίζοντας επίσης τις μετρικές που μας ενδιαφέρουν, όπως ο μέσος όρος, η μέγιστη και η ελάχιστη τιμή.

- **DELETE /device/ 26f531d5-3509-407e-aaa1-b2714d06d672/data?start_time=2023-01-01 T00:00:00Z&end_time=2023-12-31 T23:59:59Z**

  Διαγράφει όλες τι μετρήσεις της συσκευής με το id που δίνεται ,για συγκεκριμένη χρονική περίοδο . Αυτό μας χρησιμεύει καθώς λόγω μεγάλου όγκου δεδομένων θέλουμε να διαγράφονται τα παλαιότερα για να μην πιάνουν αποθηκευτικό χώρο και καθυστερούν τις διαδικασίες. Για παράδειγμα εδώ θέλουμε να διαγραφούν όλες οι μετρήσεις της συσκευής για το 2023.

- **DELETE /device/data?start_time=2023-01-01 T00:00:00Z&end_time=2023-12-31 T23:59:59Z**

  Αντίστοιχα μπορούμε να διαγράψουμε όλες τις μετρήσεις για όλες τις συσκευές για την ίδια χρονική περίοδο.

- **DELETE /device/ 26f531d5-3509-407e-aaa1-b2714d06d672/**

  Ή όλες τις μετρήσεις μιας συσκευής γενικά.

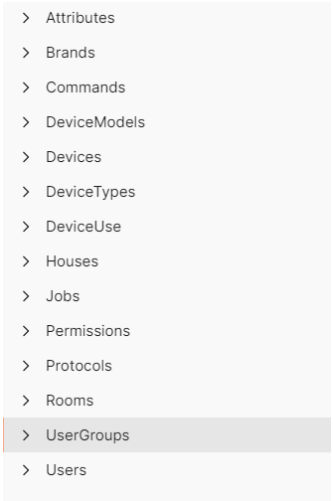
ΚΕΦΑΛΑΙΟ 5

5.ΔΟΚΙΜΗ ΚΑΙ ΑΞΙΟΛΟΓΗΣΗ

5.1. ΜΕΘΟΔΟΛΟΓΙΑ ΔΟΚΙΜΩΝ

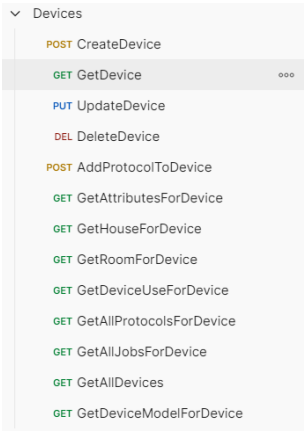
5.1.1. ΠΡΟΣΕΓΓΙΣΕΙΣ ΓΙΑ ΤΟΝ ΕΛΕΓΧΟ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

Για να πραγματοποιήσουμε το testing του συστήματός μας και να επιβεβαιώσουμε ότι αλληλοεπιδρά αποτελεσματικά με τον χρήστη και επιστρέφει τα σωστά αποτελέσματα σε κάθε περίπτωση, θα δοκιμάσουμε τα routes που ορίσαμε στο προηγούμενο κεφάλαιο. Για να το πετύχουμε αυτό φτιάχνουμε στο Postman 14 συλλογές (collections) , μία για κάθε κύρια οντότητα της βάσης μας, και σε κάθε μία από αυτές εισάγουμε όλα τα πιθανά requests που μπορεί να ζητήσει κάποιος χρήστης και αντιστοιχούν στα routes που ορίσαμε στον κώδικα.



Εικόνα 5.1: Συλλογές (collections) στο Postman

Για παράδειγμα η συλλογή των Devices ορίζεται ως εξής :



Εικόνα 5.2 : Endpoints για η συλλογή των συσκευών στο Postman

Θα εκτελέσουμε αυτά τα endpoints αρχικά εξετάζοντας ορισμένα use cases.

Στο ERD που έχει παρουσιαστεί παραπάνω θεωρούμε όλες τις σχέσεις μεταξύ των πινάκων , εκτός από δύο (Rooms-Devices και Devices-Jobs) υποχρεωτικές. Για λόγους ευκολίας και συντομίας ωστόσο στις δοκιμές μας θα τις θεωρήσουμε όλες προαιρετικές, έτσι ώστε να μπορούμε να εισάγουμε νέα εγγραφή σε κάποιον πίνακα πχ Users χωρίς να χρειάζεται προηγουμένως να έχουμε εισάγει εγγραφή στον πίνακα των User Groups σαν FK. Η υποχρεωτικότητα των σχέσεων και η αδυναμία εισαγωγής κάποιας εγγραφής χωρίς την ύπαρξη ξένου κλειδιού έχει δοκιμαστεί με επιτυχία σε προηγούμενους ελέγχους .

Θα δοκιμάσουμε τις διεπαφές CRUD (Create, Read, Update, Delete). Στην ενότητα 5.2.1. παρουσιάζονται ορισμένα ενδεικτικά παραδείγματα

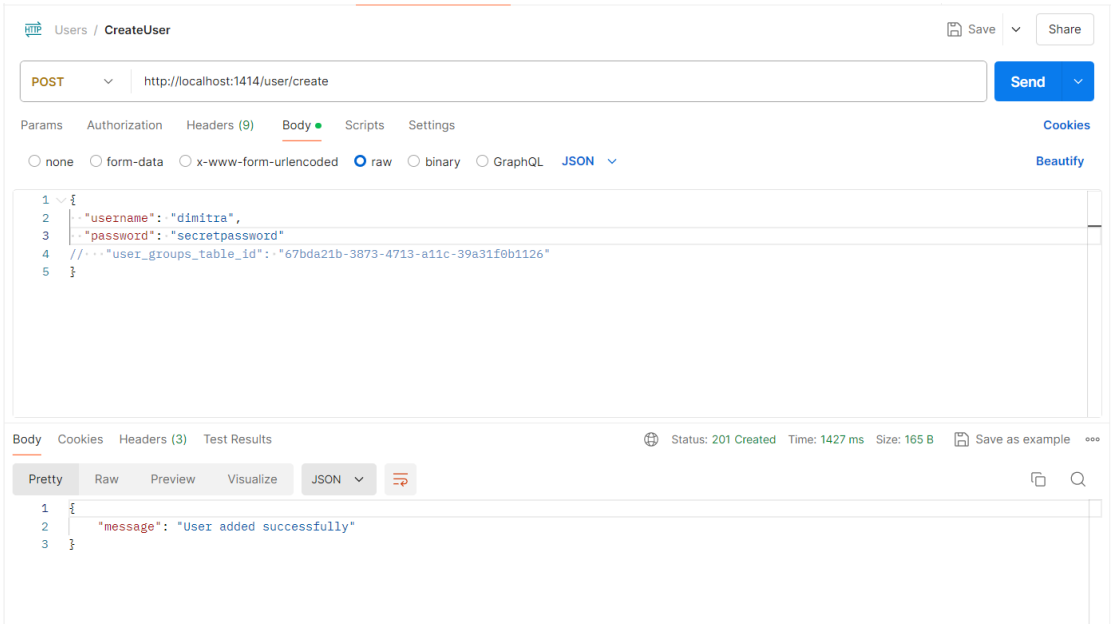
5.2. ΔΟΚΙΜΑΣΤΙΚΕΣ ΠΕΡΙΠΤΩΣΕΙΣ ΚΑΙ ΣΕΝΑΡΙΑ

5.2.1. ΣΥΓΚΕΚΡΙΜΕΝΕΣ ΠΕΡΙΠΤΩΣΕΙΣ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΓΙΑ ΤΟΝ ΕΛΕΓΧΟ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

5.2.1.1. CREATE (POST)

- **Create User :** Δημιουργία νέου χρήστη.

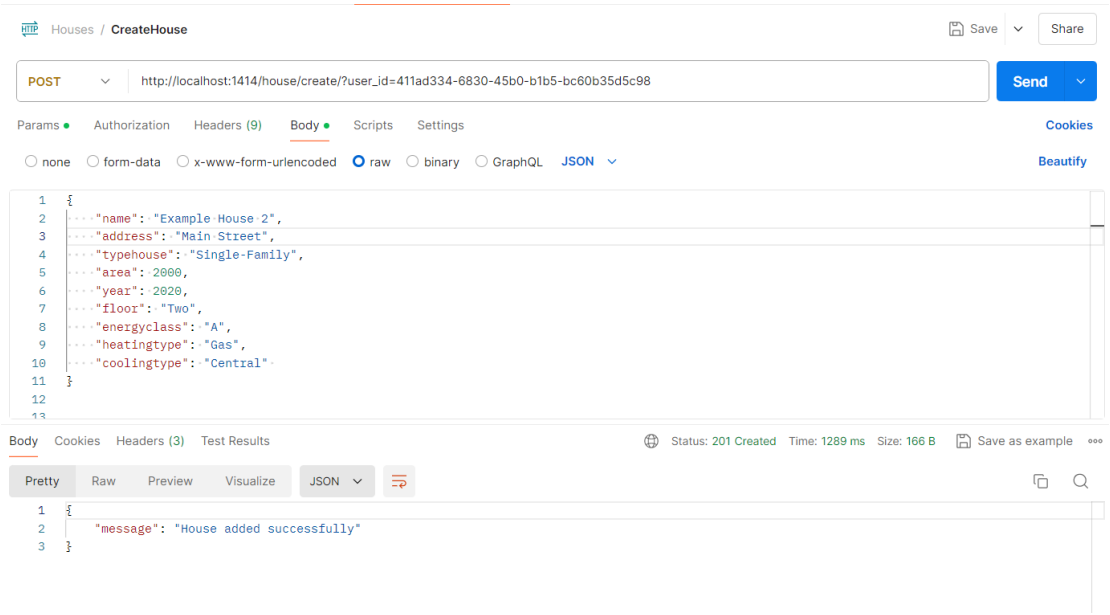
  Για αρχή εισάγουμε σαν πληροφορίες χρήστη μόνο το username και το password, τα οποία είναι υποχρεωτικά και τα έχουμε ορίσει να είναι not null.
  Χτυπάμε το URL για το αντίστοιχο endpoint για τη μέθοδο POST και βάζουμε τις κατάλληλες πληροφορίες σε μορφή JSON, το key είναι το όνομα του πεδίου στο schema που έχουμε ορίσει και το value είναι η τιμή που θέλουμε να του δώσουμε , οπότε μας εμφανίζει το μήνυμα ότι η εισαγωγή ήταν επιτυχής (περιέχεται και σε σχόλιο το πεδίο με το id από κάποια εγγραφή του user groups , σαν FK, σε περίπτωση που θέλουμε να συνδέσουμε τον χρήστη με αυτήν) :



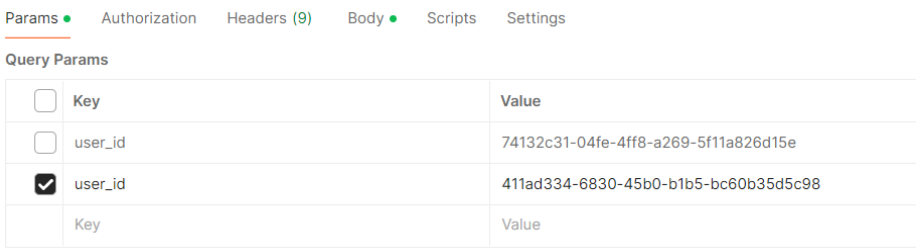
Εικόνα 5.3 : Δημιουργία νέου χρήστη με Postman

Αν προσπαθήσω να εισάγω χρήστη με το ίδιο username, μου επιστρέφει το μήνυμα “User already exists” και δεν με αφήνει, γιατί θέλουμε κάθε username να είναι μοναδικό .

- **Create House :** Δημιουργία νέου σπιτιού.



Εικόνα 5.4 : Δημιουργία νέου σπιτιού με Postman



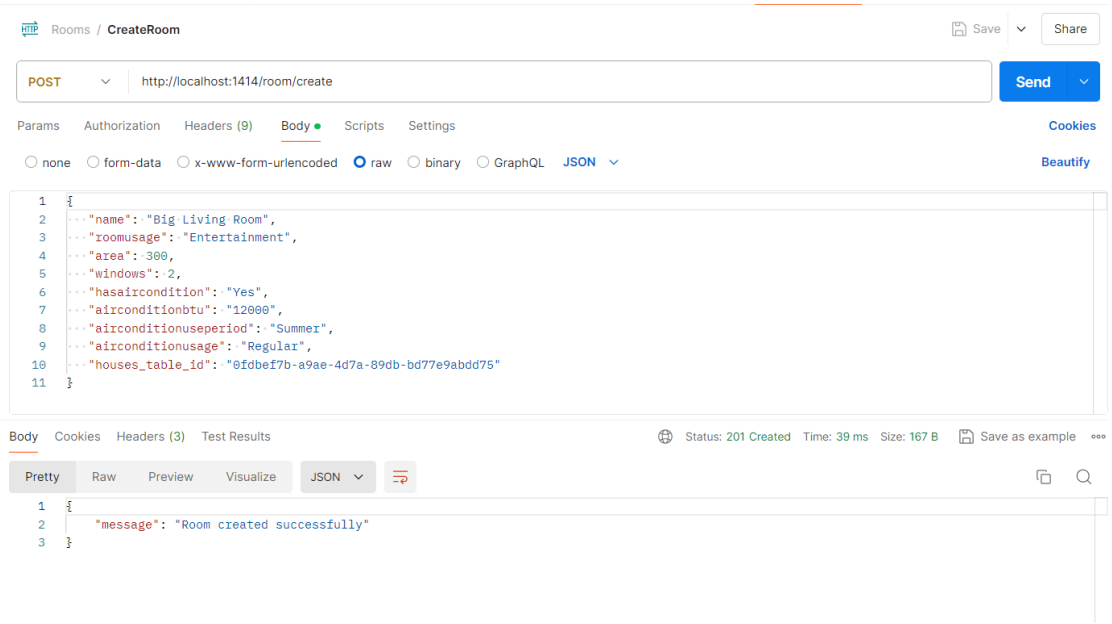
Εικόνα 5.5 : Σύνδεση χρήστη με το νέο σπίτι με Postman

Το user id καθορίζεται σαν κλειδί στις παραμέτρους του house,λόγω της σχέσης many-to-many που τα συνδέει. Αν το id του χρήστη που πάω να εισάγω δεν υπάρχει τότε μου επιστρέφει “User not found”.
Με τον καθορισμό αυτή τη σχέση many-to-many δημιουργείται αυτόματα και μια νέα εγγραφή στον ενδιάμεσο πίνακα user_houses:

```
admin=# SELECT * FROM user_houses;
           houses_table_id            |           users_table_id
--------------------------------------+--------------------------------------
 0fdbef7b-a9ae-4d7a-89db-bd77e9abdd75 | 411ad334-6830-45b0-b1b5-bc60b35d5c98
```

Εικόνα 5.6 : Επιβεβαίωση σύνδεσης χρήστη-σπιτιού στον ενδιάμεσο πίνακα user_houses στη γραμμή εντολών

- **Create Room :** Δημιουργία νέου δωματίου.



Εικόνα 5.7 : Δημιουργία νέου δωματίου με Postman

Με τον ίδιο τρόπο μπορούμε να εισάγουμε εγγραφές και στους υπόλοιπους 11 βασικούς πίνακες (όχι στους 5 ενδιάμεσους) .Παρατηρούμε ότι για να εισαχθεί μια εγγραφή είναι υποχρεωτικό να ορίσουμε μόνο το name της (στον user και το password), δηλαδή αυτά που έχουμε ορίσει στο σχήμα ως not null. Τα FK, όπως αναφέρθηκε και παραπάνω, τα έχουμε κάνει όλα προαιρετικά για να μας διευκολύνει στις δοκιμές.

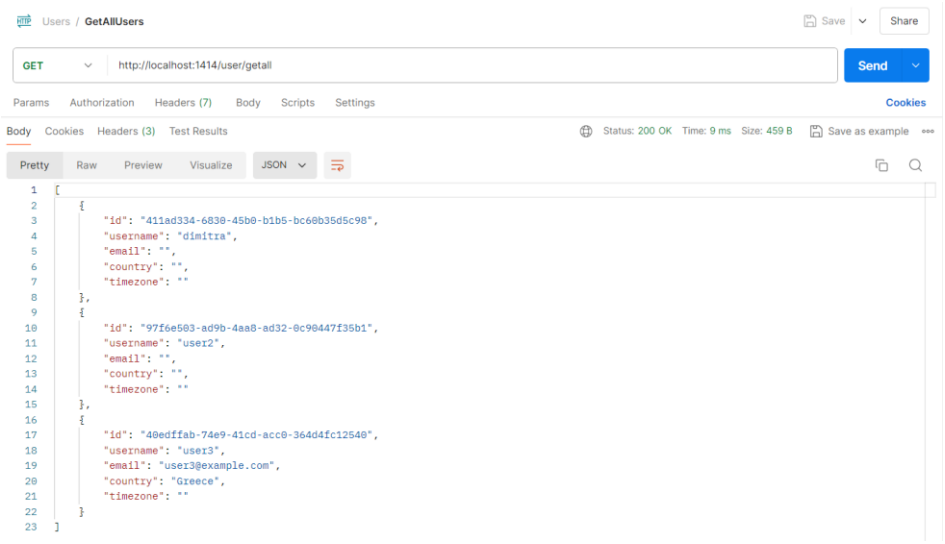
Σε περίπτωση που δοκιμάσουμε να εισάγουμε μια εγγραφή με κάποιο FK που δεν υπάρχει τότε η βάση θα μας επιστρέψει το αντίστοιχο μήνυμα λάθους, όπως και στην περίπτωση που εισάγουμε μη έγκυρο json (πχ κάποιο λάθος πεδίο).

Επίσης, σε one-to-one σχέσεις, όπως πχ τα attributes με τα devices, έχουμε επιβεβαιώσει ότι δεν μπορεί να εισαχθεί εγγραφή σε κάποιον πίνακα των attributes που να έχει FK το ίδιο id στα devices που έχει ήδη κάποια άλλη εγγραφή του πίνακα attributes .

Αν είχαμε υλοποιήσει όλες τις σχέσεις υποχρεωτικές, όπως φαίνεται στο ERD, τότε για να εισάγουμε μια νέα εγγραφή θα έπρεπε εκτός από τα πεδία not null να έχουμε υποχρεωτικά και id στα FK. Έτσι, έπρεπε να εισάγονται πρώτα οι εγγραφές των πινάκων που χρησιμοποιούνται ως FK σε άλλους πίνακες , πριν εισαχθούν εγγραφές σε αυτούς τους πίνακες, γεγονός που απαιτεί συγκεκριμένη σειρά πρώτων εγγραφών και περιπλέκει τα πράγματα στο testing.

5.2.1.2. READ (GET)

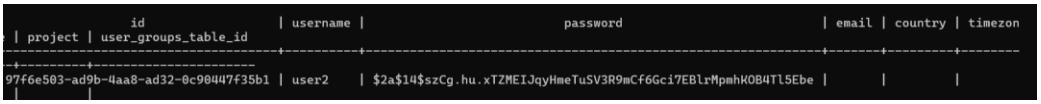
- **Get All Users :** Μπορούμε να ζητήσουμε να μας επιστραφούν όλοι οι χρήστες που είναι αποθηκευμένοι στη βάση μας .



Εικόνα 5.8 : Ανάκτηση όλων των χρηστών με Postman

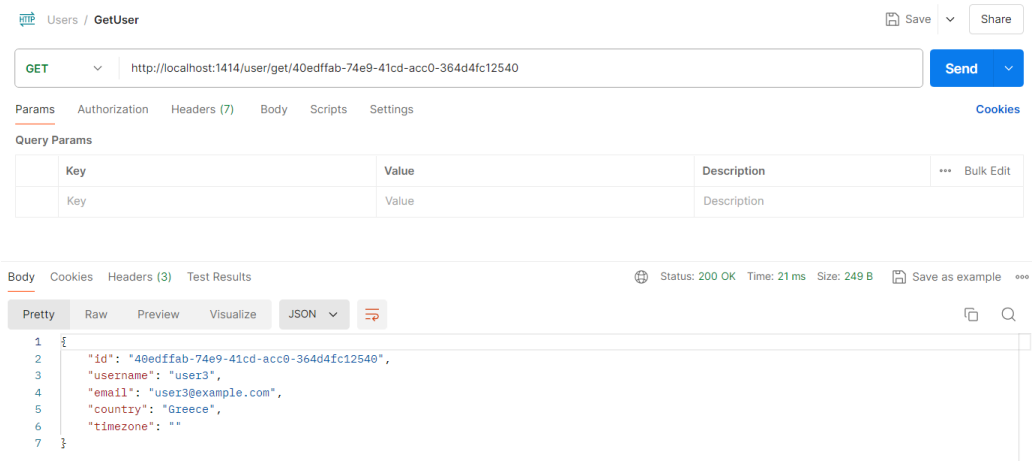
Σε όποιο πεδίο δεν έχει δοθεί τιμή κατά την εγγραφή παρατηρούμε ότι υπάρχει κενό.

Εκτελώντας στη γραμμή εντολών την εντολή «SELECT * FROM users_tables;» διαπιστώνουμε ότι , για παράδειγμα, στον χρήστη 2 ο κωδικός που δώσαμε κατά την εγγραφή αποθηκεύεται κωδικοποιημένος. Αυτό εξασφαλίζει την προστασία των προσωπικών δεδομένων των χρηστών ,καθώς κανένας ,ούτε οι διαχειριστές, δεν έχει πρόσβαση σε αυτά.



Εικόνα 5.9 : Επιβεβαίωση κωδικοποίησης του password από τη γραμμή εντολών

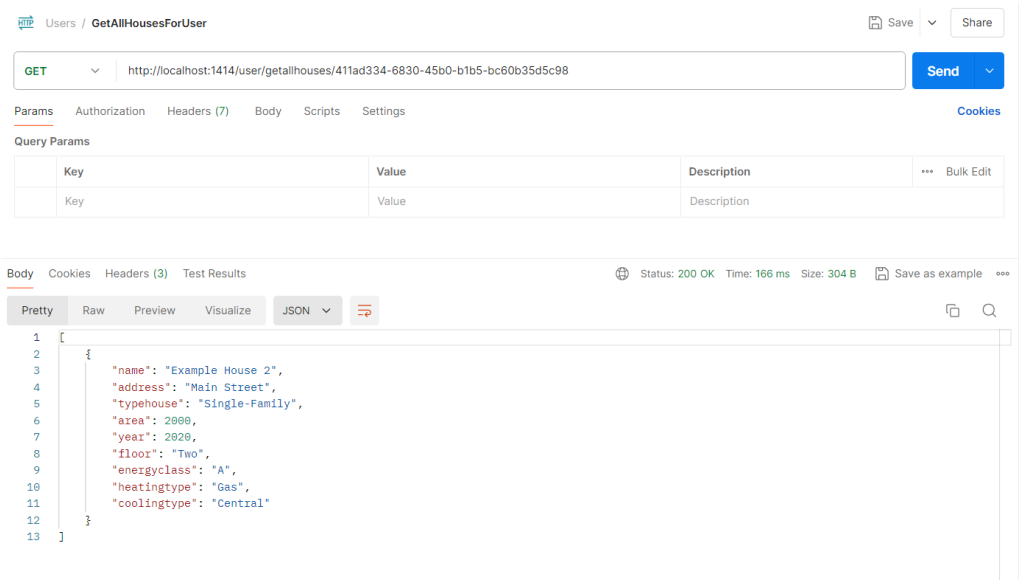
- **Get User :** Ζητάμε να μας επιστραφούν πληροφορίες για έναν συγκεκριμένο χρήστη .



Εικόνα 5.10 : Ανάκτηση των πληροφοριών ενός συγκεκριμένου χρήστη με Postman

Σε περίπτωση που δεν υπάρχει ο χρήστης με το συγκεκριμένο id επιστρέφει error “record not found” .

- **Get All Houses For User :** Ζητάμε να μας επιστρέψει τα σπίτι με τα οποία είναι συνδεδεμένος κάποιος χρήστης.

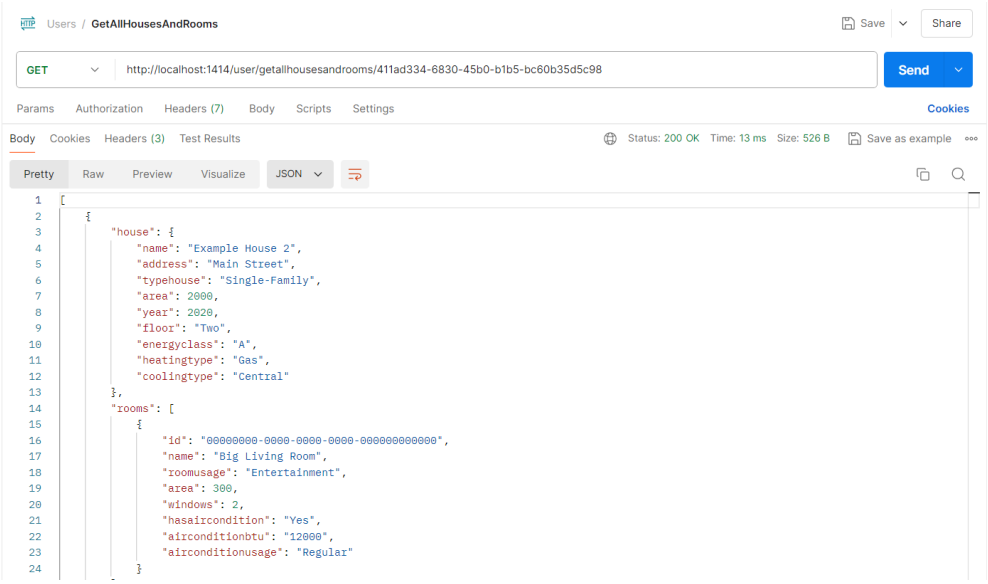


Εικόνα 5.11 : Ανάκτηση όλων των σπιτιών που συνδέονται με ένα συγκεκριμένο χρήστη με Postman

Αν ο χρήστης δεν είναι συνδεδεμένος με κανένα σπίτι τότε επιστρέφει null.

Παρατηρούμε ότι εδώ που ζητούνται να επιστραφούν πληροφορίες για κάποια εγγραφή άλλου πίνακα δεν επιστρέφεται το id της 2ης.

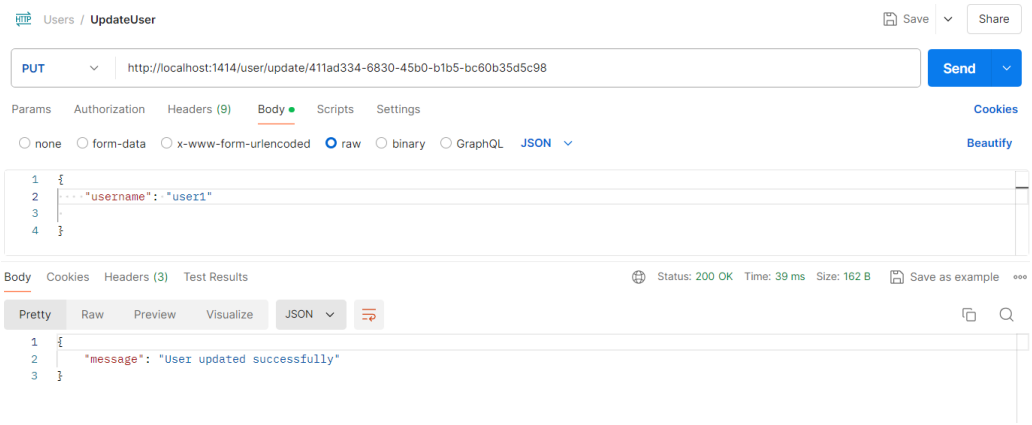
- **Get All Houses And Rooms For User :** Ζητάμε να μας επιστρέψει τα σπίτια και τα δωμάτια με τα οποία συνδέεται ο χρήστης.



Εικόνα 5.12 : Ανάκτηση όλων των σπιτιών και των δωματίων που συνδέονται με ένα συγκεκριμένο χρήστη με Postman

5.2.1.3. UPDATE

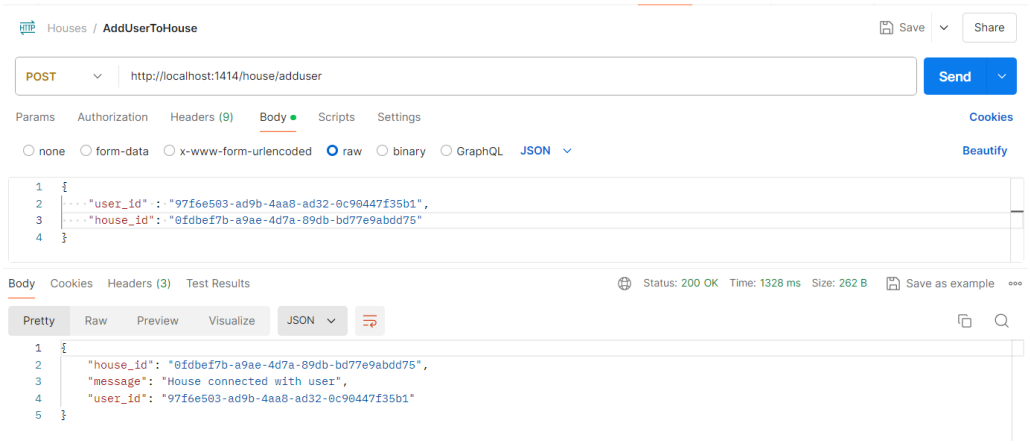
- **Update User :** Ενημέρωση κάποιου/κάποιων στοιχείων συγκεκριμένου χρήστη.



Εικόνα 5.13 : Ενημέρωση των στοιχείων ενός χρήστη με Postman

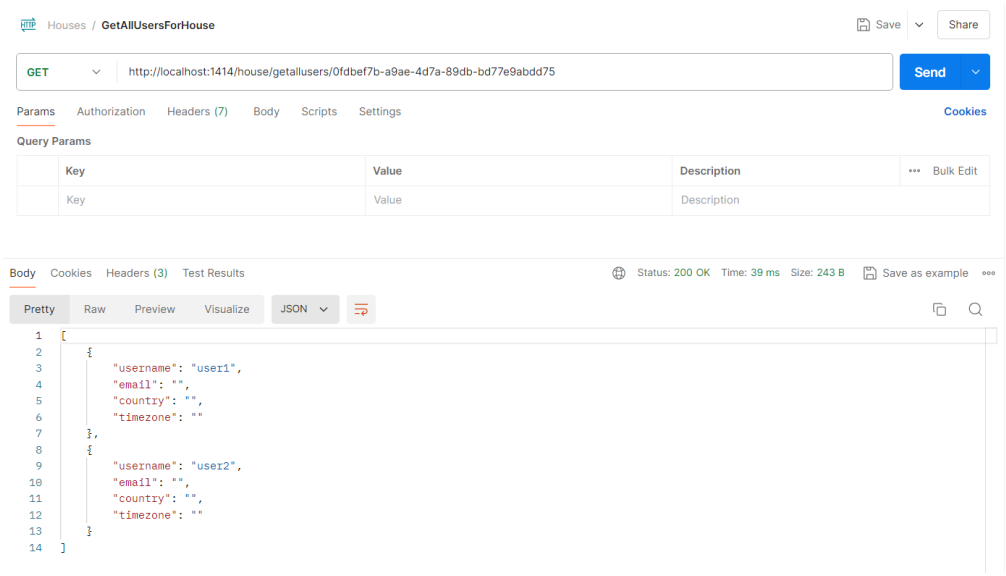
Αν ζητήσουμε να μας επιστραφούν οι πληροφορίες του συγκεκριμένου χρήστη θα παρατηρήσουμε ότι όντως έχουν ενημερωθεί τα στοιχεία του.

- **Add User To House :** Προσθήκη χρήστη σε κάποιο σπίτι.



Εικόνα 5.14 : Προσθήκη χρήστη σε κάποιο σπίτι με Postman

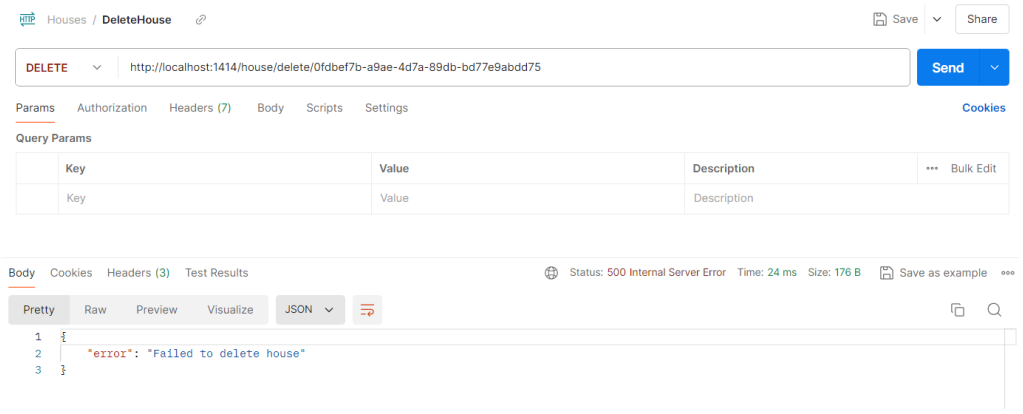
Στην επόμενη εικόνα φαίνεται ότι αν ζητήσουμε να μας επιστραφούν όλοι οι χρήστες που είναι συνδεδεμένοι με κάποιο σπίτι, θα τους επιστρέψει και τους δύο σωστά:



Εικόνα 5.15 : Επιβεβαίωση ανάκτησης όλων των χρηστών που συνδέονται με ένα συγκεκριμένο σπίτι με Postman

5.2.1.4. DELETE

- **Delete House**



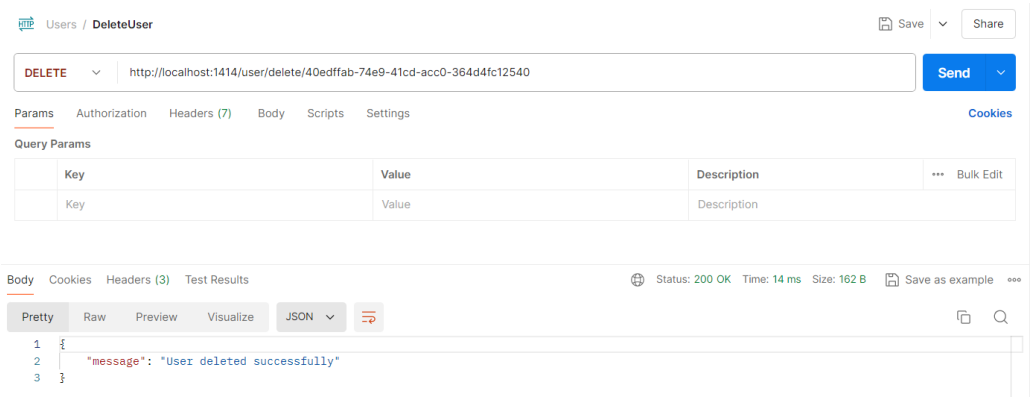
Εικόνα 5.16 : Διαγραφή σπιτιού με Postman

Εξαιτίας της ύπαρξης του id του σπιτιού που θέλουμε να διαγράψουμε στον ενδιάμεσο πίνακα των user_houses, δεν μπορούμε να διαγράψουμε το σπίτι αν πρώτα δεν έχει διαγραφεί αυτή η σχέση. Αντίστοιχα, δεν μπορούμε να διαγράψουμε ούτε το χρήστη με τον οποίο είναι συνδεδεμένο το σπίτι , για τον ίδιο ακριβώς λόγο. Για να το πετύχουμε αυτό πρέπει να πάει κάποιος διαχειριστής στη βάση, από τη γραμμή εντολών ,και να διαγράψει τη σχέση κατευθείαν στον ενδιάμεσο πίνακα.

Παρόμοια, μια εγγραφή στα rooms , που έχει FK σε μια εγγραφή στον πίνακα των houses, δεν μπορεί να διαγραφεί αν πρώτα δε διαγραφεί η εγγραφή από την οποία εξαρτάται , δηλαδή η 2η.

- **Delete User :**

  Αντίθετα, μπορούμε κατευθείαν να διαγράψουμε κάποιο χρήστη που δεν συνδέεται με κανένα σπίτι :



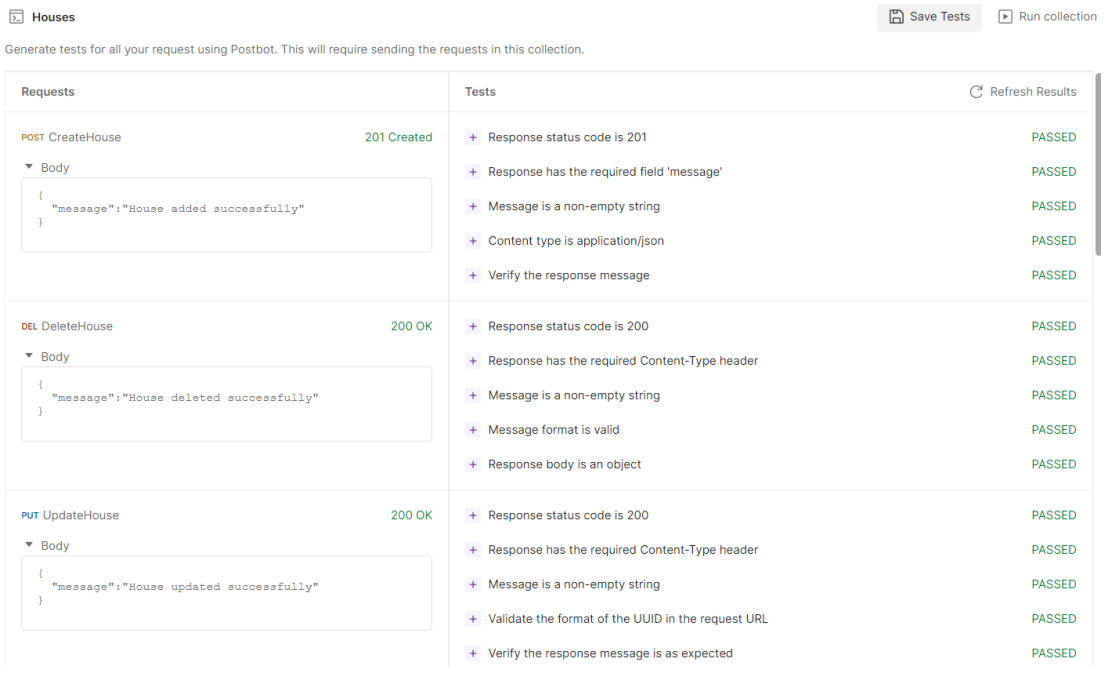
Εικόνα 5.17 : Διαγραφή χρήστη με Postman

5.2.2. ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΟ TESTING

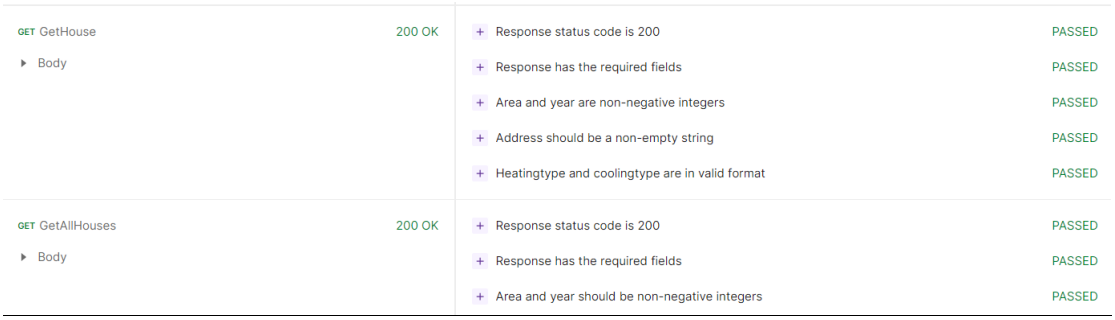
Το Postman παρέχει ένα ισχυρό εργαλείο για την αυτοματοποίηση της δημιουργίας σεναρίων ελέγχου, τα οποία μπορούν να χρησιμοποιηθούν για την επαλήθευση της απόκρισης του API μας στα διάφορα αιτήματα των χρηστών, διασφαλίζοντας τη σωστή λειτουργία του.

Για να το πετύχουμε αυτό δημιουργούμε αρχικά τα αιτήματα στα endpoints του API που θέλουμε να δοκιμάσουμε. Στη συνέχεια , με τη λειτουργία “Generate Tests” το Postman , με τη βοήθεια κάποιου μοντέλου AI, μπορεί να δημιουργήσει σενάρια ελέγχου για κάθε endpoint. Στην ουσία, εκτελεί τα αιτήματα στα endpoints με τις παραμέτρους που έχουμε ορίσει και στη συνέχεια δημιουργεί και τεστάρει συνθήκες που θα πρέπει να ικανοποιούνται σε κάθε ένα, όπως η εγκυρότητα της μορφής των δεδομένων και των παραμέτρων, ο χρόνος απόκρισης , το μήνυμα που θα πρέπει να επιστρέφεται κ.ά.

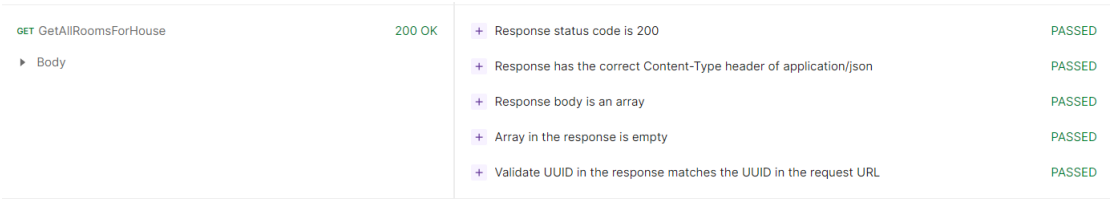
- **Houses**



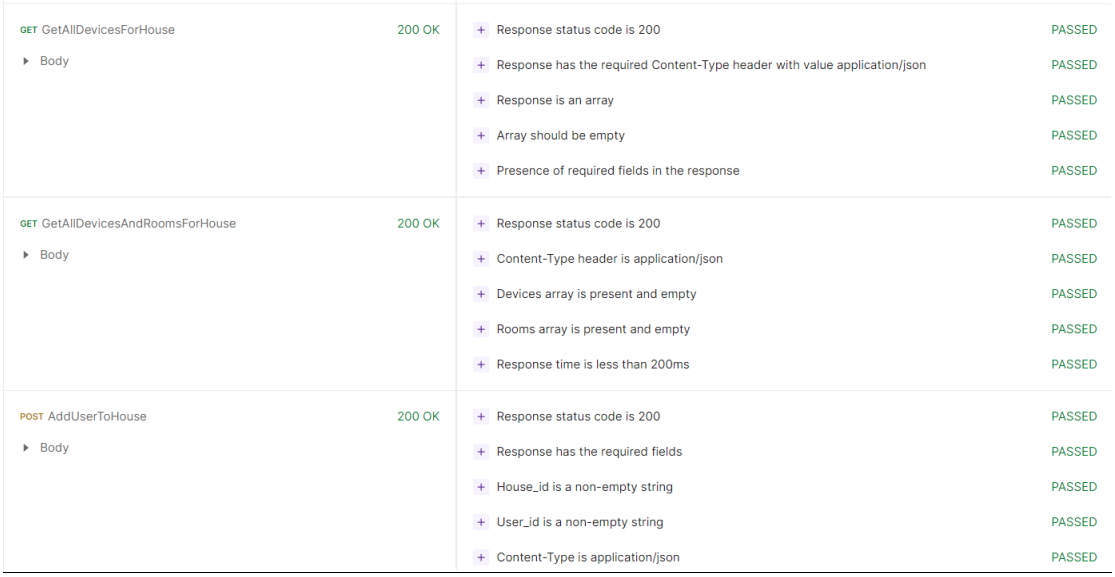
Εικόνα 5.18 : AI Testing για τον πίνακα των σπιτιών με Postman (CreateHouse,DeleetHouse,UpdateHouse)



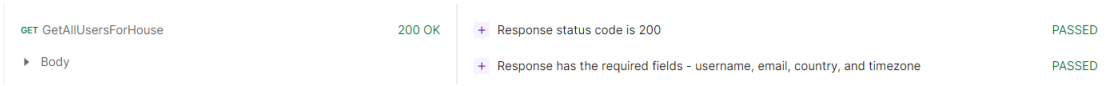
Εικόνα 5.19 : AI Testing για τον πίνακα των σπιτιών με Postman (GetHouse,GetAllHouses)



Εικόνα 5.20 : AI Testing για τον πίνακα των σπιτιών με Postman (GetAllRoomsForHouse)



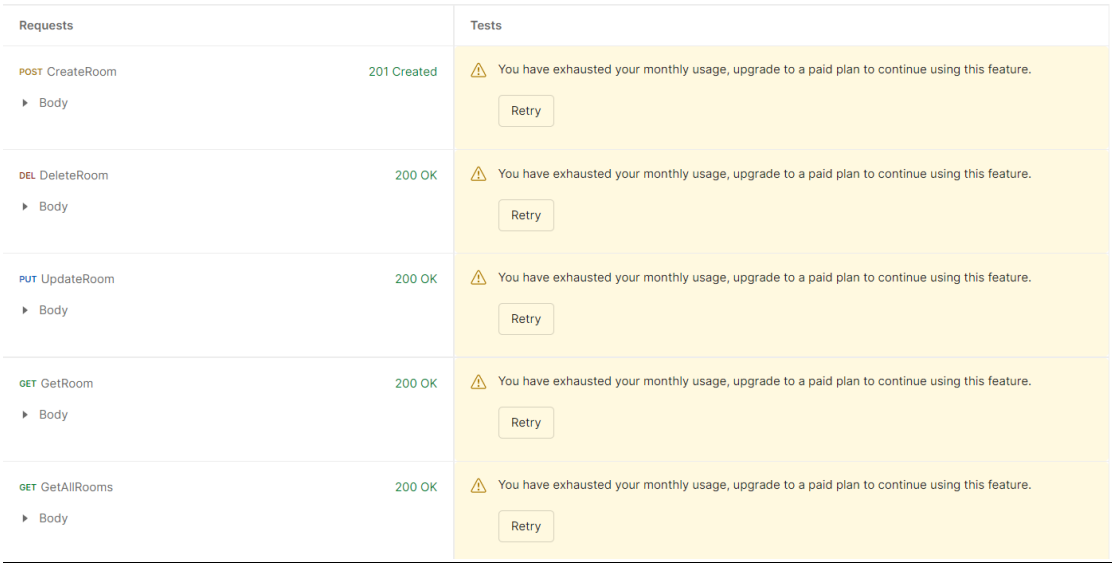
Εικόνα 5.21 : AI Testing για τον πίνακα των σπιτιών με Postman (GetAllDevicesForHouse,GetAllDevicesAndRoomsForHouse,AddUserToHouse)



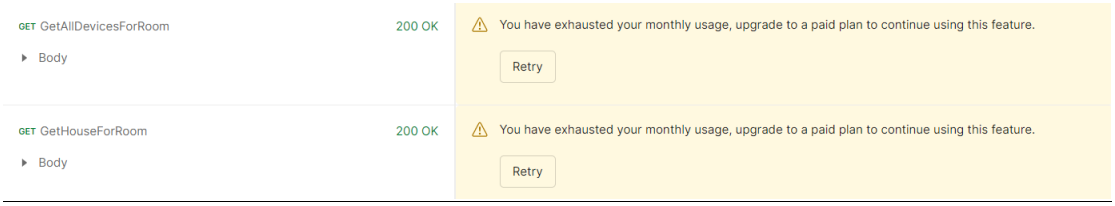
Εικόνα 5.22 : AI Testing για τον πίνακα των σπιτιών με Postman(GetAllUsersForHouse)

- Με τον ίδιο τρόπο θα μπορούσαμε να ελέγξουμε και τους υπόλοιπους πίνακες, και κυρίως τους βασικούς των Users,Devices και Rooms.

  Ωστόσο, καθώς το free trial του Postman μας επιτρέπει συγκεκριμένο αριθμό για τη δημιουργία tests με AI , οπότε , για παράδειγμα για το collection των Rooms παίρνουμε το παρακάτω μήνυμα :



Εικόνα 5.23 : AI Testing για τον πίνακα των δωματίων με Postman(CreateRoom,DeleteRoom,UpdateRoom,GetRoom,GetAllRooms)



Εικόνα 5.24 : AI Testing για τον πίνακα των δωματίων με Postman(GetAllDevicesForRoom,GetHouseForRoom)

Παρ'όλα αυτά , αν και δεν φαίνονται αναλυτικά τα PASS και FAIL για κάθε endpoint, είναι ξεκάθαρο , και από την εικόνα της γραμμής εντολών παρακάτω, πως όλα εκτελούνται σωστά και έχουν κατάσταση "200 OK" χωρίς να εμφανίζεται κάποιο error.

```
[GIN] 2024/07/06 - 18:39:54 | 201 |    7.908672ms |             ::1 | POST   "/room/create"
[GIN] 2024/07/06 - 18:39:54 | 200 |    8.554486ms |             ::1 | DELETE "/room/delete/c8abd637-d017-4afc-9bfc-3e8bd8303cef"
[GIN] 2024/07/06 - 18:39:54 | 200 |      7.8067ms |             ::1 | PUT    "/room/update/185bb343-58e4-4282-aa15-f7ea851d59bd"
[GIN] 2024/07/06 - 18:39:54 | 200 |    3.028175ms |             ::1 | GET    "/room/get/185bb343-58e4-4282-aa15-f7ea851d59bd"
[GIN] 2024/07/06 - 18:39:54 | 200 |    2.072809ms |             ::1 | GET    "/room/getall"
[GIN] 2024/07/06 - 18:39:54 | 200 |    4.847574ms |             ::1 | GET    "/room/getalldevices/185bb343-58e4-4282-aa15-f7ea851d59bd"
[GIN] 2024/07/06 - 18:39:54 | 200 |     1.77999ms |             ::1 | GET    "/room/gethouse/185bb343-58e4-4282-aa15-f7ea851d59bd"
```

Εικόνα 5.25 : Room endpoints από τη γραμμή εντολών

Με τον ίδιο τρόπο μπορούμε να ελέγξουμε τα endpoints και για τους υπόλοιπους πίνακες.

ΚΕΦΑΛΑΙΟ 6

6.ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ

6.1. ΣΥΜΠΕΡΑΣΜΑΤΑ

Η εργασία αυτή ασχολήθηκε με την ανάπτυξη μιας PostgreSQL βάσης δεδομένων για τη αποθήκευση και τη διαχείριση πληροφοριών που αφορούν τη δομή ενός έξυπνου σπιτιού και των στοιχείων από τα οποία αυτό αποτελείται. Η βάση αυτή ενεργεί και αλληλοεπιδρά συμπληρωματικά με μια βάση δεδομένων χρονοσειρών Timescale, στην οποία αποθηκεύονται αποκλειστικά οι μετρήσεις που προέρχονται από τις συσκευές και τους αισθητήρες. Το σύστημα αυτό εξυπηρετεί, συνολικά, όπως αναφέρθηκε και στις προηγούμενες ενότητες, την παρακολούθηση και τον έλεγχο της ενεργειακής κατανάλωσης του σπιτιού από τους ίδιους τους χρήστες , τόσο για δική τους οικονομία όσο και για περιβαλλοντικούς σκοπούς .

Τα αποτελέσματά μας ,ωστόσο, θα επικεντρωθούν στην PostgreSQL βάση η οποία υλοποιήθηκε στα πλαίσια αυτής της εργασία, προκειμένου να δούμε αν εξυπηρετεί τους σκοπούς τους και βοηθάει τους χρήστες στην διαχείριση των δεδομένων τους.

Από τις πλήρεις δοκιμές και τον έλεγχο που πραγματοποιήθηκε στο προηγούμενο κεφάλαιο μπορούν να εξαχθούν τα παρακάτω συμπεράσματα σχετικά με τα επιθυμητά χαρακτηριστικά που διαθέτει η βάση μας :

1. **Λειτουργικότητα :** Υποστηρίζει πλήρως τις βασικές λειτουργίες της αποθήκευσης, ανάκτησης, ενημέρωσης και διαγραφής (CRUD) δεδομένων. Οι δοκιμές που εκτελέστηκαν απέδειξαν ότι όλες οι λειτουργίες εκτελούνται χωρίς σφάλματα και επιστρέφουν τα επιθυμητά αποτελέσματα.

2. **Αξιοπιστία :** Η χρήση περιορισμών ,όπως Foreign Keys, Unique Ids, και Not null πεδία) διασφαλίζει την ακεραιότητα των δεδομένων. Κατά τις δοκιμές, δεν παρατηρήθηκαν περιπτώσεις όπου εισήχθησαν εσφαλμένα δεδομένα. Συγκεκριμένα:
    - μια εγγραφή σε πίνακα , ο οποίος διαθέτει πεδίο FK προς κάποιον άλλον πίνακα, δεν μπορεί να εισαχθεί εάν δεν οριστεί η τιμή ενός έγκυρου FK, εκτός αν αυτή η σχέση ορίζεται ως προαιρετική , όπως ανάμεσα στα Devices και στα Rooms.
    - Έχουμε φροντίσει για τη διατήρηση της μοναδικότητας του PK σε κάθε νέα εγγραφή ενός πίνακα, σε αυτό να δίνεται αυτόματα μια τυχαία τιμή uuid ,ενώ δεν μπορεί να εισαχθεί νέα εγγραφή με χειροκίνητο καθορισμό του PK σε κάποιον πίνακα στον οποίο υπάρχει ήδη άλλη εγγραφή με το ίδιο PK.
    - Μια νέα εγγραφή σε οποιονδήποτε πίνακα δεν μπορεί να εισαχθεί αν δεν έχει τιμές για πεδία που δεν μπορούν να είναι null, όπως για παράδειγμα το όνομα.
    - Ένας πίνακας που το PK του εντάσσεται σαν FK σε κάποιον άλλον πίνακα δεν μπορεί να διαγραφεί αν δεν διαγραφεί πρώτα ο 2^{ος} πίνακας .

3. **Επιδόσεις :** Οι απαντήσεις από τη βάση δεδομένων ήταν άμεσες και γρήγορες και οι λειτουργίες εκτελούνται αποδοτικά. Σε αυτό συνέβαλε και η χρήση της βιβλιοθήκης GORM , όπως αναφέρθηκε παραπάνω, έναντι της κλασσικής γλώσσας ερωτημάτων SQL.

4. **Επεκτασιμότητα:** Η βάση μας μπορεί να υποστηρίξει την προσθήκη νέων πινάκων , με έναν απλό ορισμό του πίνακα struct και ένα Automigrate () εκ νέου του σχήματος, χωρίς να χρειάζεται να γίνουν όλα από την αρχή. Επίσης, μπορεί να δεχθεί την προσθήκη νέων χαρακτηριστικών, όπως νέων εντολών και δυναμικών χαρακτηριστικών για τις συσκευές, χωρίς να απαιτείται εξ ολοκλήρου αλλαγή στη δομή της, χάρη και στη χρήση του τύπου json που προσδίδει ευελιξία στα δεδομένα. Συνεπώς, έχει τη δυνατότητα να υποστηρίξει εύκολα νέους τύπους συσκευών , αλλά και οποιονδήποτε εξοπλισμό έχει να κάνει με τα έξυπνα σπίτια.

5. **Ασφάλεια:** Η βάση μας εξασφαλίζει την προστασία των προσωπικών δεδομένων των χρηστών και την ιδιωτικότητά τους. Το βασικότερο από όλα, ο κωδικός πρόσβασης κάθε χρήστη για να εισέλθει στο λογαριασμό του κρυπτογραφείται κατά την αποθήκευσή του στη βάση, έτσι ώστε κανένας , ούτε καν οι διαχειριστές, να μην τον γνωρίζουν. Μάλιστα, η κρυπτογράφηση γίνεται κάθε φορά με διαφορετικό κλειδί , που προκύπτει τυχαία από μια γεννήτρια κλειδιών, έτσι ώστε να είναι αδύνατο να βρεθεί το κλειδί που χρησιμοποιήθηκε για κάθε κωδικό και να είναι αδύνατο να γίνει αποκωδικοποίησή του. Ακόμη, αν και δεν αναλύεται στα πλαίσια αυτής της διπλωματικής, έχουμε φροντίσει έτσι ώστε για τα αιτήματα των χρηστών προς το σύστημα να μεσολαβεί ένα API Gateway που εξασφαλίζει πως μόνο εξουσιοδοτημένοι χρήστες έχουν δικαίωμα πρόσβασης στα δεδομένα και εκτέλεσης συγκεκριμένων ενεργειών στη βάση .

Τα παραπάνω επιτεύχθηκαν χάρη σε διαδοχικές βελτιστοποιήσεις που έγιναν τόσο στις τεχνικές που χρησιμοποιήθηκαν, όπως η χρήση της βιβλιοθήκης GORM αντί για την γλώσσα ερωτημάτων SQL, όσο και στη δημιουργία του σχήματος της βάσης.

Αναφορικά με το δεύτερο κομμάτι, η πρώτη βελτιστοποίηση αφορά τους πίνακες των έξυπνων συσκευών. Στην αρχή δημιουργήθηκαν ξεχωριστοί πίνακες για κάθε κατηγορία συσκευών ,όπως Air Condition, Shelly Sensors, Shelly Controllers. Αυτό , ωστόσο, εμπόδιζε την επεκτασιμότητα της βάσης μας, καθώς περιοριζόταν μόνο σε αυτά τα μοντέλα συσκευών, με συγκεκριμένα χαρακτηριστικά το καθένα, και συνεπώς οποιαδήποτε νέα συσκευή διαφορετική από αυτές δεν μπορούσε να ενταχθεί στο δίκτυο. Ως λύση σε αυτό προτάθηκε η δημιουργία ενός ενιαίου πίνακα για όλες τις συσκευές και η αποθήκευση των διαφορετικών χαρακτηριστικών για κάθε συσκευή με ένα πεδίο json, το οποίο προσδίδει ευελιξία.

Μια επόμενη μοντελοποίηση αφορούσε τη δημιουργία αυτού του ενιαίου πίνακα για όλες τις συσκευές, μαζί με μερικούς πίνακες για τους χρήστες, τα δωμάτια, τα σπίτια, τις εργασίες και τις άδειες πρόσβασης, με σχεδόν όλες τις δυνατές συνδέσεις μεταξύ τους. Ωστόσο, ούτε κάτι τέτοιο ήταν αποδοτικό, αφού μερικές συνδέσεις ήταν περιττές και ,επίσης, υπήρχαν πληροφορίες αποθηκευμένες πάνω από 1 φορά, γεγονός που είχε επίπτωση στην ταχύτητα και στον αποθηκευτικό χώρο του συστήματος.

Τελικά, καταλήξαμε στο ERD μοντέλο που παρουσιάστηκε και υλοποιήθηκε στην εργασία μας, με τη δημιουργία σταθερών χαρακτηριστικών , όπως οι τύποι συσκευών, οι κατηγορίες χρηστών, τα μοντέλα, η μάρκα κλπ σε ξεχωριστούς πίνακες, προκειμένου να μην επαναλαμβάνεται η πληροφορία, αλλά να δηλώνεται με μια απλή σύνδεση με την απαιτούμενη οντότητα.

Συνεπώς, γίνεται σαφές πως η υλοποίηση της βάσης δεδομένων μας για έξυπνα σπίτια είναι επιτυχής . Οι δοκιμές απέδειξαν ότι το σύστημα είναι αξιόπιστο, αποδοτικό και ασφαλές. Η δυνατότητα επέκτασης και η ευκολία χρήσης το καθιστούν ιδανικό για εφαρμογές που απαιτούν τη

διαχείριση μεγάλου όγκου δεδομένων σχετικά με έξυπνα σπίτια. Συνολικά, το σύστημα παρέχει μια ολοκληρωμένη λύση για την αποθήκευση και διαχείριση των δεδομένων αυτών, διευκολύνοντας τους χρήστες στην καθημερινή τους διαχείριση.

6.2. ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

Η βάση δεδομένων που υλοποιήσαμε στην παρούσα εργασία για έξυπνα σπίτια μπορεί να επεκταθεί σε μελλοντική εργασία για την συμπερίληψη περισσότερων συσκευών και μετρήσεων.

Με την εισαγωγή κατάλληλων πινάκων και την αντικατάσταση ή τη διαγραφή ορισμένων από τους υπάρχοντες που έχουν οριστεί στο σχήμα μας, ανάλογα με τις ανάγκες του κάθε έργου, η ίδια βάση μπορεί να χρησιμοποιηθεί για την αποθήκευση και τη διαχείριση δεδομένων που εξυπηρετούν ένα ευρύτερο πλαίσιο, όπως είναι η ενεργειακές κοινότητες. Ακόμη, η βάση μας μπορεί να επεκταθεί εύκολα έτσι ώστε να υποστηρίξει διαφορετικούς τύπους συσκευής και οποιονδήποτε εξοπλισμό έχει να κάνει με ένα έξυπνο σπίτι, όπως είναι για παράδειγμα τα φωτοβολταϊκά συστήματα.

Οι ενεργειακές κοινότητες ορίζονται ως κοινότητες που επιτρέπουν στους πολίτες να συμμετέχουν στην παραγωγή ενέργειας, μέσα από καθαρές ή ανανεώσιμες πηγές, ενισχύοντας το εισόδημά τους, προστατεύοντας το περιβάλλον και προωθώντας την ενεργειακή δημοκρατία, σύμφωνα με την οποία η παραγωγή ενέργειας είναι ένα θέμα που αφορά κάθε πολίτη και όχι έναν μόνο πάροχο [98] . Τα δεδομένα που σχετίζονται με αυτές τις κοινότητες στην Ελλάδα μπορεί να αφορούν φωτοβολταϊκά συστήματα σε κτίρια ή σε πάρκα, αιολικά πάρκα, σταθμούς βιομάζας ή βιοαερίου για παραγωγή ενέργειας από αγροτική κοινότητα, θερμοκήπια και εγκαταστάσεις ΑΠΕ με σύστημα αποθήκευσης ενέργειας. Η ενέργεια που παράγεται από αυτά μπορεί να αξιοποιηθεί είτε για πώληση είτε από νοικοκυριά ή μικρές επιχειρήσεις για να παράγουν ένα μέρος ή το σύνολο της ηλεκτρικής ενέργειας που χρειάζονται για τη λειτουργία τους, μειώνοντας έτσι την εξάρτησή τους από κάποιον πάροχο ενέργειας. Ένα τέτοιο σύστημα αποτελεί μια γενίκευση της βάσης μας, σε συνδυασμό με τη βάση χρονοσειρών μας , με την απαραίτητη προσαρμογή του σχήματός της, καθώς περιλαμβάνει πληροφορίες για τις εμπλεκόμενες οντότητες και μετρητές για τη συλλογή δεδομένων. Επίσης, η βάση μας μπορεί να χρησιμοποιηθεί και για την αποθήκευση δεδομένων που προέρχονται μεμονωμένα από κάποια από τα παραπάνω συστήματα, όπως φωτοβολταϊκά και αιολικά πάρκα [98].

Η PostgreSQL βάση μας , επίσης, μπορεί να χρησιμοποιηθεί για εφαρμογές όπως η δημιουργία προσωποποιημένης εμπειρίας χρήστη, σύμφωνα με την οποία το σύστημα μπορεί να μελετήσει τα προγράμματα Jobs που έχει δημιουργήσει κάποιος χρήστης, να κατανοήσει τις ανάγκες του και να του προτείνει και άλλα παρόμοια προγράμματα.

Ακόμη, η βάση μας μπορεί να χρησιμοποιηθεί και για την εύκολη και γρήγορη εφαρμογή ερωτημάτων με σκοπό τη δημιουργία αναφορών . Μπορεί κάποιος να ανακτήσει γρήγορα και εύκολα , για παράδειγμα, πόσες συσκευές μέσα στο σπίτι ανήκουν σε ένα συγκεκριμένο μοντέλο ή ακόμη, σε συνδυασμό με την Timescale βάση, να ζητήσει να ενημερωθεί σχετικά με τον χώρο του σπιτιού που είχε τη μεγαλύτερη ενεργειακή κατανάλωση για κάποιο χρονικό διάστημα.

Αναφορικά με τη βελτιστοποίηση του χρόνου αναζήτησης ανάμεσα σε έναν μεγάλο όγκο δεδομένων που θα έχουν αποθηκευτεί σε αυτήν, μπορούμε να χρησιμοποιήσουμε τα ευρετήρια (indexes), τα οποία είναι δομές που επιταχύνουν την ανάκτηση δεδομένων από τους πίνακες, χωρίς να απαιτείται ολόκληρη η σάρωσή τους. Με τη βοήθειά τους θα δημιουργήσουμε αναφορές σε συγκεκριμένα σημεία σε κάποιον πίνακα, έτσι ώστε να γίνεται γρηγορότερη η αναζήτηση ,ενώ παράλληλα ενημερώνεται αυτόματα κατά την εισαγωγή, ενημέρωση ή διαγραφή δεδομένων, διασφαλίζοντας τη συνέπεια των πληροφοριών.

Μελλοντικές εργασίες και ιδέες σχετίζονται και με τη Timescale βάση χρονοσειρών μας, η οποία αποτελεί ένα microservice με δυνατότητα να συνδεθεί με άλλες υπηρεσίες για την επίτευξη ενός σκοπού. Ορισμένα έργα στα οποία μπορούμε να την αξιοποιήσουμε στα έξυπνα σπίτια , και όχι μόνο, είναι τα παρακάτω :

1. Σε επικοινωνία με κατάλληλο API, προκειμένου να παρέχει live ενημέρωση των χρηστών σχετικά με την ενεργειακή τους κατανάλωση. Αυτό μπορεί να συνδυαστεί και με αντίστοιχα εργαλεία οπτικοποίησης, έτσι ώστε να φαίνεται ευκολότερα η εξέλιξη της κατανάλωσης του σπιτιού κατά τη διάρκεια της ημέρας και να διευκολύνονται οι διαμένοντες στην κατανόησή της.

2. Ανάλυση των χρονοσειρών που προκύπτουν από τις μετρήσεις με κατάλληλα Machine Learning μοντέλα , έτσι ώστε να εξάγονται στατιστικά στοιχεία ανά χρονικές περιόδους και τα συμπεράσματα που θα προκύψουν θα μπορούν να χρησιμοποιηθούν για τη λήψη αποφάσεων.

3. Επίσης, τα δεδομένα μπορούν να αξιοποιηθούν για την έκδοση των τιμολογίων ενέργειας από τις εταιρίες προς τους καταναλωτές , τα οποία είναι σημαντικά για τη διαχείριση δαπανών ενέργειας από τους καταναλωτές και την ενημέρωσή τους , αφού περιλαμβάνουν τις χρεώσεις της κατανάλωσης ενέργειας για συγκεκριμένο χρονικό διάστημα. Σε συνέχεια αυτού, οι χρήστες μπορούν να κατηγοριοποιηθούν με βάση τα τιμολόγιά τους και να αναλυθούν τα προφιλ κατανάλωσης κάθε κατηγορίας. Το γεγονός αυτό μπορεί να αποκαλύψει διαφορές μεταξύ των ομάδων, να αναδείξει ποιες κατηγορίες έχουν υψηλότερη ή χαμηλότερη κατανάλωση, και με αυτό τον τρόπο να προσφερθούν πιο προσαρμοσμένες υπηρεσίες και προϊόντα ανάλογα με τις ανάγκες κάθε κατηγορίας, αλλά και να προταθούν διαφορετικά μέτρα εξοικονόμησης για κάθε ομάδα, για την καλύτερη διαχείριση κόστους.

4. Παρόμοια μοντέλα , τα οποία θα τροφοδοτηθούν με τα δεδομένα των μετρήσεων, είναι πιθανό να συμβάλουν στο σχηματισμό προβλέψεων για μελλοντική κατανάλωση ενέργειας σε σύντομα χρονικά διαστήματα.

5. Η Timescale βάση που διαθέτουμε είναι σχεσιακή και αποθηκεύει δομημένα δεδομένα μετρήσεων που προέρχονται από τους αισθητήρες. Ωστόσο, μελλοντικά, για εξοικονόμηση χρόνου και πόρων χρειάζεται να συνδεθεί με μια NoSQL βάση στην οποία θα μεταφέρονται τα δεδομένα μετά από κάποια προκαθορισμένη χρονική περίοδο , οπότε θα θεωρούνται παλιά και δεν θα μας είναι πια χρήσιμα. Η συγκέντρωση δεδομένων (data aggregation) ,πχ ανά μήνα για μια κατοικία, μπορεί να βοηθήσει στη μείωση του πλήθους τους, αφού έχοντας εξαχθεί τα απαραίτητα αναλυτικά συμπεράσματα όσο αυτά βρίσκονται στην αρχική μας βάση, αυτά αντί να απορριφθούν εντελώς συγκεντρώνονται σε μία $2^{η}$ πιο ευέλικτη βάση , σε περίπτωση που χρειαστεί κάποια εποπτεία τους.

Ο κώδικας για την PostgreSQL βάση δεδομένων που αναπτύχθηκε στα πλαίσια της παρούσας διπλωματικής βρίσκεται στον εξής σύνδεσμο :
https://github.com/DimitraKaroutsou/EHS_Database

ΚΕΦΑΛΑΙΟ 7

7. ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] : Harper R (2003) , Inside the Smart Home https://link.springer.com/chapter/10.1007/1-85233-854-7_2

[2] : TechTarget (2023), Definition of smart home
https://www.techtarget.com/iotagenda/definition/smart-home-or-building

[3] : Hayes A. (2024), Smart Home: Definition,How They Work, Pros and Cons
https://www.investopedia.com/terms/s/smart-home.asp

[4] :Ding D. , Cooper R. , Pasquina P., Fici-Pasquina L.(2011) , Sensor technology for smart homes
https://www.sciencedirect.com/science/article/pii/S0378512211000983

[5] : Allioui H., Mourdi Y. (2023), Exploring the Full Potentials of IoT for Better Financial Growth and Stability: A Comprehensive Survey
https://www.mdpi.com/1424-8220/23/19/8015#B30-sensors-23-08015

[6] : Big Blue Academy (2022) , Τι Είναι το Internet of Things (IoT) και Πώς Λειτουργεί
https://bigblue.academy/gr/internet-of-things-iot

[7] : Kenton W. (2022), What Is the Internet of Things (IoT)? How It Works and Benefits
https://www.investopedia.com/terms/i/internet-things.asp

[8] : Ren Y.,Huang D.,Wang W., Yu X.(2023), BSMD:A blockchain-based secure storage mechanism for big spatio-temporal data
https://www.sciencedirect.com/science/article/abs/pii/S0167739X22002928

[9] : Orfanos V., Kaminaris S. , Papageorgas P. , Piromalis D. , Kandris D.(2023) , A Comprehensive Review of IoT Networking Technologies for Smart Home Automation Applications
https://www.mdpi.com/2224-2708/12/2/30

[10] Zeng, Q., Lv, Z., Li, C. (2023) FedProLs: federated learning for IoT perception data prediction
https://link.springer.com/article/10.1007/s10489-022-03578-1

[11] : Mansour N. , Bujosa Vadell L. (2022) The Financial Sphere in the Era of COVID-19 :Trends and Perspectives of Artificial Intelligence

https://link.springer.com/chapter/10.1007/978-3-030-89416-0_3

[12] : Allioui H. , Allioui A. , Mourdi Y. (2023) , AI-Based Logistics Solutions to Tackle Covid-19 Pandemic and Ensure a Sustainable Financial Growth in Advanced AI and Internet of Health Things for Combating Pandemics, Lahby M., Pilloni V., Sekhar Banerjee J. , Mahmud M.

https://link.springer.com/chapter/10.1007/978-3-031-28631-5_17

[13] : Yavari A., Korala H., Georgakopoulos D., Kua J., Bagha H. (2023), Sazgar IoT: A Device-Centric IoT Framework and Approximation Technique for Efficient and Scalable IoT Data Processing.

https://www.mdpi.com/1424-8220/23/11/5211

[14] :Pliatsios A., Kotis K., Goumopoulos C. (2023), A systematic review on semantic interoperability in the IoE-enabled smart cities.

https://www.sciencedirect.com/science/article/pii/S254266052300077X

[15] :Lemus-Zúñiga,L.-G., Félix J.M., Fides-Valero A., Benlloch-Dualde J.-V., Martinez-Millana A. (2022) A Proof-of-Concept IoT System for Remote Healthcare Based on Interoperability Standards.

https://www.mdpi.com/1424-8220/22/4/1646

[16] : Abounassar E.M., El-Kafrawy P., Abd El-Latif A. (2022) , A. Security and Interoperability Issues with Internet of Things (IoT) in Healthcare Industry: A Survey. In Security and Privacy Preserving for IoT and 5G Networks, Abd El-Latif, A.A., Abd-El-Atty, B., Venegas-Andraca, S.E., Mazurczyk, W., Gupta, B.B., Eds.

https://link.springer.com/chapter/10.1007/978-3-030-85428-7_7

[17] : Zubaydi H.D., Varga P., Molnár S. (2023), Leveraging Blockchain Technology for Ensuring Security and Privacy Aspects in Internet of Things: A Systematic Literature Review. Sensors 2023.

https://www.mdpi.com/1424-8220/23/2/788

[18] : Pal A., Rath H.K., Shailendra S., Bhattacharyya A. (2018)IoT standardization: The road ahead. In Internet of Things-Technology, Applications, and Standardization

https://books.google.gr/books?hl=el&lr=&id=2XaQDwAAQBAJ&oi=fnd&pg=PA53&ots=rW04cCTwnJ&sig=CqCfPbu18taLuzda--OvDEyv0Eo&redir_esc=y#v=onepage&q&f=false

[19] :Bharany S., Sharma S., Khalaf O.I., Abdulsahib G.M., Al Humaimeedy A.S., Aldhyani T.H.H., Maashi M., Alkahtani H. (2022) A Systematic Survey on Energy-Efficient Techniques in Sustainable Cloud Computing.

https://www.mdpi.com/2071-1050/14/10/6256

[20] : Lake (2023) , Definition: What is a Smart Home?

https://www.lake.com/help/glossary/smart-home/

[21] : Molina-Solana M., Ros M. , M. Dolores Ruiz , Gómez-Romero J., M.J. Martin-Bautista (2017), Data science for building energy management: A review

https://www.sciencedirect.com/science/article/pii/S1364032116308814?casa_token=DuXJEge%200Rr%C7%AAAAAAA:31R-YoaxVxgJ_UYN3pS_DY2xj4WbZkCH8xYbBYAWD9d-bLw8AGWtnd_sale36ysh7J21kwIF#s0005)

[22] : Leitão J. ,Gil P. , Ribeiro B. , Cardoso A. (2020), A Survey on Home Energy Management

https://ieeexplore.ieee.org/abstract/document/8948036

[23] : Miorandi D., Sicari S., De Pellegrini F., Chlamtac I. (2012) , Internet of things: Vision, applications and research challenges.

https://www.sciencedirect.com/science/article/pii/S1570870512000674

[24] : Pérez S., Garcia-Carrillo D., Marín-López R., Ramos J.L.H., Perez R.M., Skarmeta A.F. (2019), Architecture of security association establishment based on boot-strapping technologies for enabling secure IoT infrastructures.

https://www.sciencedirect.com/science/article/pii/S0167739X18325573

[25] : Sicari S., Cappiello C., De Pellegrini F., Miorandi D., Porisini A.C. (2016), A security-and quality-aware system architecture for Internet of Things.

https://link.springer.com/article/10.1007/s10796-014-9538-x

[26] : San Emeterio de la Parte, M., Martínez-Ortega J.F., Hernández Díaz V., Martinez N.L. (2023) , Big Data and precision agriculture: A Novel spatio-temporal Semantic IoT Data Management Framework for Improved Interoperability.

https://link.springer.com/article/10.1186/s40537-023-00729-0

[27] : Lemus-Zúñiga L.-G., Félix J.M., Fides-Valero A., Benlloch-Dualde J.-V., Martinez-Millana A. (2022), A Proof-of-Concept IoT System for Remote Healthcare Based on Interoperability Standards.

https://www.mdpi.com/1424-8220/22/4/1646

[28] : Kiljander J., D'elia A., Morandi F., Hyttinen P., Mattila J.T., Oja A.Y., Soininen J.P., Cinotti T.S. (2014) , Semantic interoperability architecture for pervasive computing and Internet of Things.

https://ieeexplore.ieee.org/abstract/document/6879461

[29] : Cleber M., Sadok D., Kelner J. (2019) , An IoT sensor and scenario survey for data researchers.

https://link.springer.com/article/10.1186/s13173-019-0085-7

[30] : Vijaya R.S., Nalluri S., Ramasubbareddy S., Govinda K., Swetha E. (2020) , Brilliant corp yield prediction utilizing Internet of Things. In Data Engineering and Communication Technology;

https://link.springer.com/chapter/10.1007/978-981-15-1097-7_75

[31] : Bi Z., Jin Y., Maropoulos P., Zhang W.-J., Wang L. (2023), Internet of things (IoT) and big data analytics (BDA) for digital manufacturing (DM).

https://www.tandfonline.com/doi/abs/10.1080/00207543.2021.1953181

[32] : Zeng Q., Lv Z., Li C., Shi Y., Lin Z., Liu C., Song G. (2023), FedProLs: Federated learning for IoT perception data prediction.

https://link.springer.com/article/10.1007/s10489-022-03578-1

[33] : Yavari A., Korala H., Georgakopoulos, D., Kua J., Bagha H.(2023) , Sazgar IoT: A Device-Centric IoT Framework and Approximation Technique for Efficient and Scalable IoT Data Processing.

https://www.mdpi.com/1424-8220/23/11/5211

[34] : Vijaya R.S., Nalluri S., Ramasubbareddy S., Govinda K., Swetha E. (2020), Brilliant corp yield prediction utilizing Internet of Things. In Data Engineering and Communication Technology.

https://link.springer.com/chapter/10.1007/978-981-15-1097-7_75

[35] : Elijah O., Rahman T.A., Orikumhi I., Leow C.Y., Hindia M.N. (2018), An overview of Internet of Things (IoT) and data analytics in agriculture: Benefits and challenges.

https://ieeexplore.ieee.org/abstract/document/8372905

[36] : Wang K., Wang Y., Sun Y., Guo S., Wu J. (2016), Green industrial Internet of Things architecture: An energy-efficient perspective.

https://ieeexplore.ieee.org/abstract/document/7785890

[37] : Ahad M.A., Biswas R. (2019), Request-based, secured, and energy-efficient (rbsee) architecture for handling IoT big data.

https://journals.sagepub.com/doi/abs/10.1177/0165551518787699

[38] : Naranjo P., Baccarelli E., Scarpiniti, M. (2018) , Design and energy-efficient resource management of virtualized networked fog architectures for the real-time support of IoT applications.

[39] :Molina-Solana M. , Ros M. , M. Dolores Ruiz , Gómez-Romero J. , M.J. Martin-Bautista (2017) ,Data science for building energy management: A review

https://www.sciencedirect.com/science/article/pii/S1364032116308814?casa_token=DuXJEge0RrQAAAAA:31R-YoaxVxgJ_UYN3pS_DY2xj4WbZkCH8xYbBYAWD9d-bLw8AGWtnd_sale36ysh7J21kwIF#s0005

[40] : A. Kusiak, M. Li, Z. Zhang (2010), A data-driven approach for steam load prediction in buildings Appl Energy.

https://www.sciencedirect.com/science/article/pii/S0306261909003808

[41] : Prahastono I, King D, Ozveren CS. (2007), A review of electricity load profile classification methods. In: Proceedings of the 42nd international universities power engineering conference.

https://ieeexplore.ieee.org/abstract/document/4469120

[42] : Z.Y. Li (2012), An empirical study of knowledge discovery on daily electrical peak load using decision tree

https://www.scientific.net/AMR.433-440.4898

[43] : Yang G, Tumwesigye E, Cahill B, Menzel K. (2010), Using data mining in optimisation of building energy consumption and thermal comfort management. In: Proceedings of the 2nd international conference on software engineering and data mining (SEDM).

https://ieeexplore.ieee.org/abstract/document/5542881

[44] : P.T. May-Ostendorp, G.P. Henze, B. Rajagopalan, C.D. Corbin (2011), Extraction of supervisory building control rules from model predictive control of windows in a mixed mode building

https://www.tandfonline.com/doi/abs/10.1080/19401493.2012.665481

[45] : Z. Yu, F. Haghighat, B.C.M. Fung, L. Zhou (2012), A novel methodology for knowledge discovery through mining associations between building operational data

https://www.sciencedirect.com/science/article/pii/S0378778811006281

[46] : F. Xiao, C. Fan (2014) , Data mining in building automation system for improving building operational performance

https://www.sciencedirect.com/science/article/pii/S0378778814001169

[47] : A. Kusiak, M. Li, F. Tang (2010), Modeling and optimization of HVAC energy consumption

https://www.sciencedirect.com/science/article/pii/S0306261910001157

[48] : C. Morbitzer, P. Strachan, C. Simpson (2004), Data mining analysis of building simulation performance data

https://journals.sagepub.com/doi/abs/10.1191/0143624404bt098oa

[49] : A. Ahmed, N.E. Korres, J. Ploennigs, H. Elhadi, K. Menzel (2011), Mining building performance data for energy-efficient operation

https://www.sciencedirect.com/science/article/pii/S147403461000100X

[50] : A. Ahmed, M. Otreba, N.E. Korres, H. Elhadi, K. Menzel (2011), Assessing the performance of naturally day-lit buildings using data mining

https://www.sciencedirect.com/science/article/pii/S1474034610000959

[51] : Shao H, Marwah M, Ramakrishnan N.(2012), A temporal motif mining approach to unsupervised energy disaggregation: applications to residential and commercial buildings. In: Proceedings of the 1st international non-intrusive load monitoring workshop.

https://ojs.aaai.org/index.php/AAAI/article/view/8485

[52] : A. Capozzoli, F. Lauro, I. Khan (2015), Fault detection analysis using data mining techniques for a cluster of smart office buildings

https://www.sciencedirect.com/science/article/pii/S0957417415000251

[53] : Javier Sedano, Leticia Curiel, Emilio Corchado, Enrique de la Cal, José R. Villar (2010), A soft computing method for detecting lifetime building thermal insulation failures

https://content.iospress.com/articles/integrated-computer-aided-engineering/ica00337

[54] : Stergiopoulos G., Kotzanikolaou P. , Theocharidou M., Gritzalis D. (2015) , Risk mitigation strategies for critical infrastructures based on graph centrality analysis

https://www.sciencedirect.com/science/article/pii/S1874548215000414

[55] : J. Meléndez, O. Quiroga, S. Herraiz (2012), Analysis of sequences of events for the characterization of faults in poser systems

https://www.sciencedirect.com/science/article/pii/S0378779612000120

[56] : Gómez-Romero J. , Bobillo F. , Ros M. , Molina-Solana M., M. Dolores Ruiz, M.J. Martin-Bautista (2015), A fuzzy extension of the semantic building information model

https://www.sciencedirect.com/science/article/pii/S0926580515000850

[57]: Zimmerman R. (2004) ,Decision-making and the vulnerability of interdependent critical infrastructure. In: Proceedings of the 2004 IEEE international conference on systems, man and cybernetics.

https://ieeexplore.ieee.org/abstract/document/1401166

[58] : G. Chicco, R. Napoli, F. Piglione, P. Postolache, M. Scutariu, C. Toader (2004), Load pattern-based classification of electricity customers

https://ieeexplore.ieee.org/abstract/document/1295037

[59] : D. De Silva, X. Yu, D. Alahakoon, G. Holmes (2011), A data mining framework for electricity consumption analysis from meter data

https://ieeexplore.ieee.org/abstract/document/5893960

[60] :D.F. Motta-Cabrera, H. Zareipour (2013), Data association mining for identifying lighting energy waste patterns in educational institutes

https://www.sciencedirect.com/science/article/pii/S0378778813001436

[61] : M. Santamouris, G. Mihalakakou, P. Patargias, N. Gaitani, K. Sfakianaki, M. Papaglastra, C. Pavlou, P. Doukas, E. Primikiri, V. Geros, M.N. Assimakopoulos, R. Mitoula, S. Zerefos (2007),

Using intelligent clustering techniques to classify the energy performance of school buildings

https://www.sciencedirect.com/science/article/pii/S0378778806001502

[62] : Galván J, Elices E, Noz AM, Czernichow T, Sanz-Bobi M. (1998), System for detection of abnormalities and fraud in customer consumption. In: Proceedings of the 12th IEEE/PES .conference electric power supply industry.

http://www.mat.ucm.es/~aelices/conferences/Elices_1998_System_for_dectection_abnormalities_fraud_customer_consumption.pdf

[63] : C. León, F. Biscarri, I. Monedero, J.I. Guerrero, J. Biscarri, R. Millán (2011), Integrated expert system applied to the analysis of non-technical losses in power utilities

https://www.sciencedirect.com/science/article/pii/S0957417411002685

[64] : Cabral JE, Pinto JOP, Martins EM, Pinto AMAC. (2008), Fraud detection in high voltage electricity consumers using data mining. In: Proceedings of transmission and distribution conference and exposition.

https://ieeexplore.ieee.org/abstract/document/4517232

[65] : M. Sforna (2000), Data mining in a power company customer database

https://www.sciencedirect.com/science/article/pii/S0378779600000869

[66] : Adam Zielonka,Marcin Woźniak, Sahil Garg,Georges Kaddoum,Md. Jalil Piran,Ghulam Muhammad (2021) ,Smart Homes: How Much Will They Support Us? A Research on Recent Trends and Advances

https://ieeexplore.ieee.org/abstract/document/9335602

[67] :Shaikh K.A., Nazir A.,Khan I., Shah A.S. (2022), Short term energy consumption forecasting using neural basis expansion analysis for interpretable time series

https://www.nature.com/articles/s41598-022-26499-y

[68] : Dave Keating (2024) , Getting smarter? Europe struggling with smart energy efficiency interconnectivity

https://www.euractiv.com/section/energy-environment/news/getting-smarter-europe-struggling-with-smart-energy-efficiency-interconnectivity/

[69] :Cuncu E. , Manca M.M. ,Pes B., Riboni D. (2022) , Towards Context-aware Power Forecasting in Smart-homes

https://www.sciencedirect.com/science/article/pii/S1877050921024741

[70] : Jaouhari S.E. , Palacios-Garcia E.J. , Anvari-Moghaddam A., Bouabdallah A. (2019) , Integrated Management of Energy, Wellbeing and Health in the Next Generation of Smart Homes

https://www.mdpi.com/1424-8220/19/3/481

[71] : Molina-Solana M., Ros M. , M. Dolores Ruiz , Gómez-Romero j., M.J. Martin-Bautista (2017), Data science for building energy management: A Review

https://www.sciencedirect.com/science/article/pii/S1364032116308814?casa_token=DuXJEge0RrQAAAAA:31R-YoaxVxgJ_UYN3pS_DY2xj4WbZkCH8xYbBYAWD9d-bLw8AGWtnd_sale36ysh7J21kwIF#bbib71

[72] : P. McDaniel, S. McLaughlin (2009) , Security and privacy challenges in the smart grid.

https://ieeexplore.ieee.org/abstract/document/5054916

[73] : Efthymiou C, Kalogridis G.(2010) Smart grid privacy via anonymization of smart metering data. In: Proceedings of the first IEEE international conference on smart grid communications (SmartGridComm).

https://ieeexplore.ieee.org/abstract/document/5622050

[74] : Laney D. (2001) 3-D data management: Controlling data volume, velocity and variety.

[75] : Teradata. Siemens and Teradata form global strategic partnership for big data in the utility sector.

https://www.sciencedirect.com/science/article/pii/S1364032116308814

[76] : Eyada M.M., Saber W. , M.M. El Genidy, Fathy Amer (2020), Performance Evaluation of IoT Data Management Using MongoDB Versus MySQL Databases in Different Cloud Environments

https://ieeexplore.ieee.org/abstract/document/9116940

[77] : Abu-Elkheir M., Hayajneh M. , Abu Ali N. (2013), Data Management for the Internet of Things: Design Primitives and Solution

https://www.mdpi.com/1424-8220/13/11/15582

[78] : Eyada M.M., Saber W. , M.M. El Genidy, Fathy Amer (2020), Performance Evaluation of IoT Data Management Using MongoDB Versus MySQL Databases in Different Cloud Environments

https://ieeexplore.ieee.org/abstract/document/9116940

[79] : Mustafa Utku KALAY (2018) , DATABASE SYSTEM SUGGESTIONS FOR THE INTERNET OF THINGS (IOT) SYSTEMS

https://dergipark.org.tr/en/download/article-file/489220

[80] : Diène B. , Joel J.P.C. Rodrigues, Diallo O., EL Hadji Malick Ndoye , Valery V. Korotaev (2020),Data management techniques for Internet of Things

https://www.sciencedirect.com/science/article/pii/S088832701930785X#s0110

[81] : Tingli Li, Yang Liu, Ye Tian, Shuo Shen, Wei Mao (2012),A Storage Solution for Massive IoT Data Based on NoSQL

https://ieeexplore.ieee.org/abstract/document/6468294

[82] : influxdata,Time series database (TSDB) explained

https://www.influxdata.com/time-series-database/

[83] : Syeda Noor Zehra Naqvi ,Sofia Yfantidou (2017) ,Time Series Databases and InfluxDB

https://www.devopsschool.com/blog/wp-content/uploads/2022/09/influxdb_2017.pdf

[84] : Wang M. , Zhang Q.(2020) , Optimized data storage algorithm of IoT based on cloud computing in distributed system

https://www.sciencedirect.com/science/article/pii/S0140366420304643#sec3

[85] : Databricks, Hadoop Distributed File System (HDFS)

https://www.databricks.com/glossary/hadoop-distributed-file-system-hdfs

[86] : IoT Business News (2023) ,The Smart Home Revolution: How IoT Is Transforming Modern Living

https://iotbusinessnews.com/2023/11/10/89810-the-smart-home-revolution-how-iot-is-transforming-modern-living/

[87] : Zielonka A., Woźniak M., Garg S., Kaddoum G,, Md. Jalil Piran, Muhammad G. (2021) ,Smart Homes: How Much Will They Support Us? A Research on Recent Trends and Advance

https://ieeexplore.ieee.org/abstract/document/9335602

[88] : Oracle, What is a Relational Database (RDBMS)? https://www.oracle.com/in/database/what-is-a-relational-database/

[89] : Astera (2023) , PostgreSQL vs. SQL Server – Everything You Need to Know https://www.astera.com/knowledge-center/postgresql-sql-server/

[90] : Percona (2024) , What is PostgreSQL? Everything You Need to Know https://www.percona.com/blog/what-is-postgresql-used-for/

[91] : Docker documentation https://docs.docker.com/guides/docker-overview/

[92] : Lucidchart, What is an Entity Relationship Diagram (ERD)? https://www.lucidchart.com/pages/er-diagrams

[93] : Hermann Rösch (2023) , Using GORM Versus Plain SQL to Interact with Databases in Go https://medium.com/hyperskill/using-gorm-versus-plain-sql-to-interact-with-databases-in-go-39728974edc8

[94] : F5, What Is an API Gateway? https://www.f5.com/glossary/api-gateway

[95] : Boomi (2023), API Management vs. API Gateways https://boomi.com/blog/api-gateway-vs-api-management/

[96] : Sennovate , The MSSP Guide to Keycloak https://sennovate.com/the-mssp-guide-to-keycloak/

[97] : Apidog, David Demir (2023) What is Postman (A Tutorial for Beginners) https://apidog.com/blog/what-is-postman/

[98] : Greenpeace , Τι είναι οι ενεργειακές κοινότητες; https://koinitasi.greenpeace.gr/ti-einai-oi-energeiakes-koinotites/

[99] : Timescale (2024) , Building Columnar Compression for Large PostgreSQL Databases

https://www.timescale.com/blog/building-columnar-compression-in-a-row-oriented-database/

[100] : Ryan Booz ,Timescale (2023), Guide to Postgres Data Management
https://www.timescale.com/blog/guide-to-postgres-data-management/