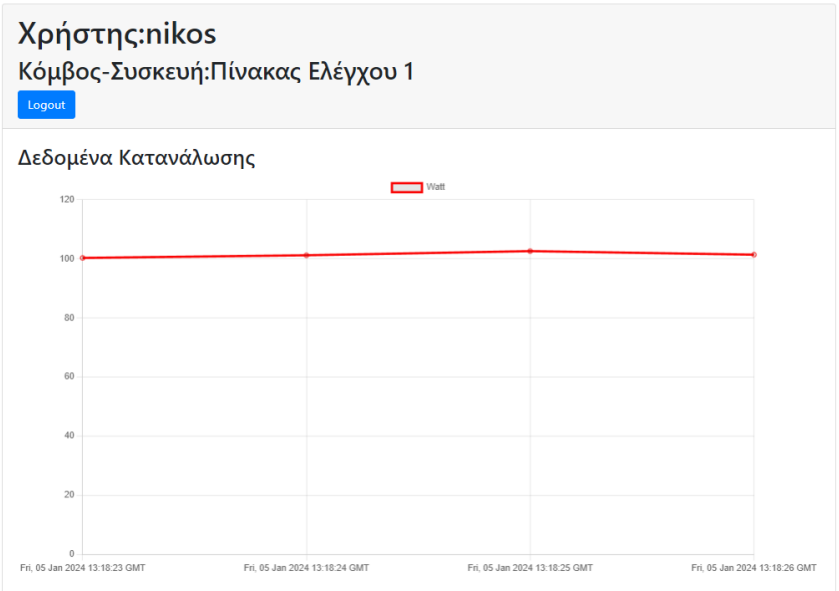
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Παρακολούθηση κατανάλωσης ρεύματος μέσω διαδικτύου»



Φοιτητής

514087

ΘΕΟΔΩΡΟΣ ΜΕΤΑΞΑΣ

Επιβλέπων

Δρ. Κυριάκος Τσιακμάκης

Φεβρουάριος 2024

Παρακολούθηση κατανάλωσης ρεύματος μέσω διαδικτύου

Κωδικός: 23305

Φοιτητής: ΘΕΟΔΩΡΟΣ ΜΕΤΑΞΑΣ

Εισηγητής: Δρ Κυριάκος Τσιακμάκης

Ημερομηνία ανάληψης Π.Ε. 31-10-2023

Ημερομηνία περάτωσης Π.Ε. 31-10-2025

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

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

Περίληψη

Η εργασία αφορά μια εφαρμογή Ιστού που βασίζεται σε Python Flask και έχει σχεδιαστεί για παρακολούθηση ενέργειας σε πραγματικό χρόνο και αλληλεπίδραση με τον χρήστη. Περιγράφει ένα ενοποιημένο σύστημα παρακολούθησης ενέργειας που ενσωματώνει μια εφαρμογή web που βασίζεται σε Python Flask με έναν μικροελεγκτή ESP32 και έναν αναλογικό αισθητήρα ρεύματος. Το σύστημα διασφαλίζει την ασφαλή πρόσβαση του χρήστη μέσω ελέγχου ταυτότητας, μετά την οποία προσφέρει αλληλεπίδραση σε πραγματικό χρόνο με δεδομένα ηλεκτρικού ρεύματος που παρέχονται από το ESP32 μέσω διαδικτύου. Οι χρήστες μπορούν να οπτικοποιήσουν την κατανάλωση ενέργειας μέσω δυναμικών γραφημάτων, χάρη στο Chart.js, απευθείας στη διεπαφή ιστού. Η εφαρμογή υποστηρίζει ενέργειες χρήστη όπως ο καθορισμός ορίων και η ανάλυση ανωμαλιών χρήσης ενέργειας, ενισχύοντας την ειδοποίηση των χρηστών. Αυτή η ενοποίηση ασφαλούς τεχνολογίας Ιστού, προγραμματισμού μικροελεγκτών και λήψης δεδομένων αισθητήρων παρέχει μια ολοκληρωμένη λύση για αποτελεσματική διαχείριση και παρακολούθηση ενέργειας.

« Online power consumption monitoring »

Abstract

The work concerns a Python Flask-based web application designed for real-time energy monitoring and user interaction. It describes an integrated energy monitoring system that integrates a Python Flask-based web application with an ESP32 microcontroller and an analog current sensor. The system ensures secure user access through authentication, after which it offers real-time interaction with electrical current data provided by the ESP32 over the internet. Users can visualize energy consumption through dynamic graphs, thanks to Chart.js, directly in the web interface. The application supports user actions such as setting limits and analyzing power usage anomalies, enhancing user notification. This integration of secure web technology, microcontroller programming and sensor data capture provides a complete solution for efficient energy management and monitoring.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ευχαριστίες μου στους γονείς μου για την αμέριστη υποστήριξή τους και στον επόπτη μου κ. Τσιακμάκη Κυριάκο για την εκπαιδευτική καθοδήγηση, τις επιστημονικές κατευθύνσεις και τη σημαντική συμβολή του στον κώδικα.

Περιεχόμενα

Περίληψη iv
Abstract v
Ευχαριστίες vi
Περιεχόμενα vii
Κατάλογος Σχημάτων viii
Κεφάλαιο 1ο: Εισαγωγή 9
1.1 Εισαγωγή 9
1.2 Δομή της εργασίας 10
Κεφάλαιο 2ο: Αναπτυξιακά και Αισθητήρας 11
2.1 ESP 32 11
2.2 Ο αναλογικός αισθητήρας ρεύματος DFRobot 14
Κεφάλαιο 3ο: Γλώσσες Προγραμματισμού και Τεχνολογικά Εργαλεία 17
3.1 Arduino IDE 17
3.2 Python 18
3.2.1 Flask 20
3.2.2 Restful EndPoint API 23
3.3 MySQL 25
3.4 Chart.js 28
Κεφάλαιο 4ο: Το σύστημα παρακολούθησης κατανάλωσης ρεύματος μέσω διαδικτύου 32
4.1 Η διαδικασία 32
4.2 Η εφαρμογή - ιστοσελίδα 39
4.3 Η Βάση 43
4.4 Ασφάλεια στο σύστημα και στα δεδομένα 46
Κεφάλαιο 5ο: Συμπεράσματα και προτάσεις βελτίωσης 47
ΒΙΒΛΙΟΓΡΑΦΙΑ 49
ΠΑΡΑΡΤΗΜΑ Α 50

Κατάλογος Σχημάτων

Εικόνα 2.1: esp32 wroom-32 [3] 12
Εικόνα 2.2: Gravity Αναλογικός Αισθητήρας Ρεύματος AC (20A) [4] 14
Εικόνα 3.1: Arduino IDE για esp32 [8] 17
Εικόνα 4.1: Σχέδιο του συστήματος μέτρησης με τον esp32 και τον αισθητήρα 32
Εικόνα 4.2: Το ολοκληρωμένο σχέδιο του συστήματος παρακολούθησης 33
Εικόνα 4.3: Διάγραμμα για τη λειτουργία στον esp32 35
Εικόνα 4.4: Διάγραμμα για τη λειτουργία στον python server 36
Εικόνα 4.5: Διάγραμμα για τη λειτουργία με χρήση python flask 37
Εικόνα 4.6: Διάγραμμα για τη λειτουργία υπολογισμού απόκλιση 38
Εικόνα 4.7: Διάγραμμα για τη λειτουργία υπολογισμού ορίων 39
Εικόνα 4.8: Σύνδεση με τα διαπιστευτήρια 40
Εικόνα 4.9: Κεντρική Σελίδα που βλέπει ο εξουσιοδοτημένος χρήστης 40
Εικόνα 4.10: Διάγραμμα με την κατανάλωση 41
Εικόνα 4.11: Τα πεδία για να θέσει όρια και ζώνες 41
Εικόνα 4.12: Οι πίνακες της βάσης δεδομένων nodewattdb 43
Εικόνα 4.13: Η δομή του πίνακα User 44
Εικόνα 4.14: Το περιεχόμενο του πίνακα User 44
Εικόνα 4.15: Η δομή του πίνακα mvalues 44
Εικόνα 4.16: Το περιεχόμενο του πίνακα mvalues 44
Εικόνα 4.17: Η δομή του πίνακα node 45
Εικόνα 4.18: Το περιεχόμενο του πίνακα node 45
Εικόνα 4.19: Η δομή του πίνακα settings 45
Εικόνα 4.20: Το περιεχόμενο του πίνακα settings 45
Εικόνα 4.21: Η δομή του πίνακα notifications 45
Εικόνα 4.22: Το περιεχόμενο του πίνακα notifications 45

Κεφάλαιο 1ο: Εισαγωγή

1.1 Εισαγωγή

Η έλευση του Διαδικτύου των Πραγμάτων (IoT) άνοιξε το δρόμο για την ενσωμάτωση των έξυπνων συσκευών στον ιστό της καθημερινής ζωής. Κεντρικό στοιχείο αυτής της ολοκλήρωσης είναι η ικανότητα παρακολούθησης και ελέγχου διαφόρων παραμέτρων του περιβάλλοντός μας. Αυτή η εργασία εισάγει ένα εξελιγμένο διαδικτυακό σύστημα που αποτελεί παράδειγμα αυτής της ολοκλήρωσης εστιάζοντας στην παρακολούθηση της κατανάλωσης ηλεκτρικού ρεύματος. Το σύστημα αξιοποιεί τις ισχυρές δυνατότητες του μικροελεγκτή ESP32 και την ακρίβεια του αναλογικού αισθητήρα ρεύματος DFRobot για την παροχή δεδομένων σε πραγματικό χρόνο σχετικά με τη χρήση ηλεκτρικού ρεύματος. Το ESP32 διαβάζει συνεχώς τις τρέχουσες τιμές RMS, οι οποίες στη συνέχεια μεταδίδονται σε έναν διακομιστή υποστήριξης για επεξεργασία και ανάλυση.

Το backend αυτού του συστήματος τροφοδοτείται από έναν διακομιστή Python Flask, ένα ελαφρύ αλλά ισχυρό πλαίσιο που υπερέχει στον χειρισμό εφαρμογών web και υπηρεσιών API. Ο διακομιστής Flask είναι υπεύθυνος για το κρίσιμο έργο του ελέγχου ταυτότητας χρήστη, διασφαλίζοντας ότι η πρόσβαση στα δεδομένα είναι ασφαλής και εξατομικευμένη. Μόλις ένας χρήστης συνδεθεί στο σύστημα, του παρέχεται πρόσβαση στα τρέχοντα δεδομένα κατανάλωσης που σχετίζονται με τον λογαριασμό του, τα οποία εμφανίζονται μέσω μιας διαδραστικής και φιλικής προς τον χρήστη διεπαφής ιστού. Η ευελιξία του πλαισίου Flask του επιτρέπει να διαχειρίζεται αποτελεσματικά τις περιόδους λειτουργίας χρήστη, διατηρώντας ένα ασφαλές περιβάλλον για τις αλληλεπιδράσεις του χρήστη με το σύστημα.

Στο μπροστινό μέρος, το σύστημα χρησιμοποιεί το Chart.js για την απόδοση των δεδομένων κατανάλωσης σε ζωντανά, κινούμενα γραφήματα που ενημερώνονται σε πραγματικό χρόνο. Η χρήση του Chart.js ζωντανεύει τα δεδομένα, μετατρέποντας τους ακατέργαστους αριθμούς σε μια ελκυστική οπτική αφήγηση που οι χρήστες μπορούν εύκολα να κατανοήσουν και να αλληλεπιδράσουν με αυτά. Αυτή η δυναμική αναπαράσταση δεδομένων δεν είναι μόνο αισθητικά ευχάριστη, αλλά ενισχύει επίσης την ικανότητα του χρήστη να παρακολουθεί τα πρότυπα κατανάλωσης ενέργειας με την πάροδο του χρόνου.

Ένα βασικό χαρακτηριστικό αυτού του συστήματος είναι η διαδραστική φύση του. Οι χρήστες δεν είναι παθητικοί αποδέκτες δεδομένων. μπορούν να ασχοληθούν ενεργά με το σύστημα θέτοντας κατώφλια για την κατανάλωση ενέργειας και λαμβάνοντας ειδοποιήσεις σε περίπτωση υπέρβασης αυτών των ορίων. Αυτή η διαδραστικότητα επεκτείνει τη λειτουργικότητα του συστήματος πέρα από την απλή παρουσίαση δεδομένων για να περιλαμβάνει στοιχεία ελέγχου και προσαρμογής. Οι χρήστες μπορούν να προσαρμόσουν το σύστημα ώστε να ανταποκρίνεται στις συγκεκριμένες ανάγκες

παρακολούθησης τους, θέτοντας παραμέτρους που αντικατοπτρίζουν τους προσωπικούς ή επιχειρηματικούς τους στόχους χρήσης ενέργειας.

Η ενσωμάτωση αισθητήρων υλικού με εξελιγμένα αναλυτικά στοιχεία λογισμικού προαναγγέλλει μια νέα εποχή στα συστήματα παρακολούθησης ενέργειας. Το σύστημα που παρουσιάζεται σε αυτό το έργο αποτελεί απόδειξη των δυνατοτήτων της τρέχουσας τεχνολογίας να παρέχει στους χρήστες λεπτομερείς πληροφορίες σχετικά με την κατανάλωση ενέργειας. Εξουσιοδοτεί τους χρήστες με γνώσεις, επιτρέποντας τη λήψη ενημερωμένων αποφάσεων σχετικά με τη χρήση ενέργειας, η οποία μπορεί να οδηγήσει σε σημαντική εξοικονόμηση κόστους και να προωθήσει την ενεργειακή απόδοση. Από τις ακριβείς μετρήσεις ρεύματος των αισθητήρων υλικού μέχρι την ασφαλή και διαδραστική εφαρμογή Ιστού, κάθε στοιχείο διαδραματίζει ζωτικό ρόλο στην παροχή μιας απρόσκοπτης εμπειρίας χρήστη.

1.2 Δομή της εργασίας

Στο πρώτο κεφάλαιο παρουσιάζεται μια μικρή εισαγωγή της εργασίας.

Στο δεύτερο κεφάλαιο παρουσιάζονται το αναπτυξιακό και ο αισθητήρας μέτρησης ρεύματος που χρησιμοποιήθηκε.

Στο τρίτο κεφάλαιο περιγράφεται η τεχνολογία που χρησιμοποιήθηκε και τα βοηθητικά εργαλεία.

Στο τέταρτο κεφάλαιο αναλύεται το σύστημα παρακολούθησης ρεύματος.

Στο πέμπτο κεφάλαιο παρουσιάζονται τα συμπεράσματα της εργασίας και θέματα για βελτίωση.

Στο τέλος της εργασίας έχει το παράρτημα με τους κώδικες που χρησιμοποιήθηκαν στην εργασία.

Κεφάλαιο 2ο: Αναπτυξιακά και Αισθητήρας

2.1 ESP 32

Ο ESP32 είναι ένας ευέλικτος και ισχυρός μικροελεγκτής που κερδίζει δημοτικότητα μεταξύ των χομπίστων, των εκπαιδευτικών και των επαγγελματιών λόγω των ισχυρών χαρακτηριστικών του και της οικονομικής του τιμής. Ο πυρήνας του ESP32 είναι ο διπύρηνος επεξεργαστής του, ο οποίος επιτρέπει την εκτέλεση πολλαπλών εργασιών που απλά δεν είναι δυνατή με πολλούς άλλους μικροελεγκτές. Αυτή η ισχύς επεξεργασίας, σε συνδυασμό με τη σχετικά γενναιόδωρη χωρητικότητα μνήμης, καθιστά το ESP32 κατάλληλο για πιο σύνθετα έργα που απαιτούν επεξεργασία σε πραγματικό χρόνο, χειρισμό δεδομένων και συνδεσιμότητα. Η ενσωμάτωση των λειτουργιών Wi-Fi και Bluetooth ενισχύει περαιτέρω τις δυνατότητές του, επιτρέποντας στο ESP32 να συνδέεται απρόσκοπτα στο διαδίκτυο και σε άλλες συσκευές, καθιστώντας το ιδανική επιλογή για έργα IoT.[1]

Το περιβάλλον ανάπτυξης και τα εργαλεία που είναι διαθέσιμα για το ESP32 είναι φιλικά προς το χρήστη και υποστηρίζονται ευρέως. Το πλαίσιο ESP-IDF, μαζί με τη συμβατότητα Arduino IDE, παρέχει μια ευέλικτη πλατφόρμα για προγραμματιστές όλων των επιπέδων δεξιοτήτων. Αυτή η προσβασιμότητα όχι μόνο ανοίγει την πόρτα για τους επαγγελματίες να αναπτύξουν εξελιγμένες εφαρμογές, αλλά επιτρέπει επίσης στους χομπίστες και τους εκπαιδευτικούς να ενσωματώσουν το ESP32 στα έργα και τα μαθήματά τους, καλλιεργώντας έτσι μια αυξανόμενη κοινότητα χρηστών που συμβάλλουν στην τεράστια βιβλιοθήκη πόρων, σεμιναρίων και ανοιχτών έργα πηγής.

Ένα από τα ξεχωριστά χαρακτηριστικά του ESP32 είναι η ενεργειακή του απόδοση. Παρά την ισχυρή του απόδοση, το ESP32 έχει σχεδιαστεί για να είναι ενεργειακά αποδοτικό, καθιστώντας το κατάλληλο για εφαρμογές που τροφοδοτούνται με μπαταρία. Η λειτουργία βαθύ ύπνου επιτρέπει τη δημιουργία εφαρμογών χαμηλής κατανάλωσης, ένα κρίσιμο χαρακτηριστικό για συσκευές IoT που πρέπει να λειτουργούν για εκτεταμένες περιόδους σε μια μικρή πηγή ενέργειας. Αυτό καθιστά το ESP32 μια φιλική προς το περιβάλλον επιλογή για διάφορες εφαρμογές, από οικιακούς αυτοματισμούς μέχρι περιβαλλοντική παρακολούθηση και φορητές συσκευές. Από απλά έργα DIY έως πολύπλοκα βιομηχανικά συστήματα IoT, το ESP32 παρέχει μια αξιόπιστη βάση. Η ικανότητά του να χειρίζεται πολλές εργασίες ταυτόχρονα και να αλληλεπιδρά με μια ποικιλία αισθητήρων και συσκευών το καθιστά εξαιρετική επιλογή για συστήματα οικιακού αυτοματισμού, όπου μπορεί να ελέγχει τα φώτα, τη θερμοκρασία και τις συσκευές ασφαλείας με ευκολία. Σε βιομηχανικά περιβάλλοντα, η ευρωστία και οι επιλογές συνδεσιμότητας επιτρέπουν την παρακολούθηση και τον έλεγχο των διαδικασιών, συμβάλλοντας στην αποτελεσματικότητα και την ασφάλεια.

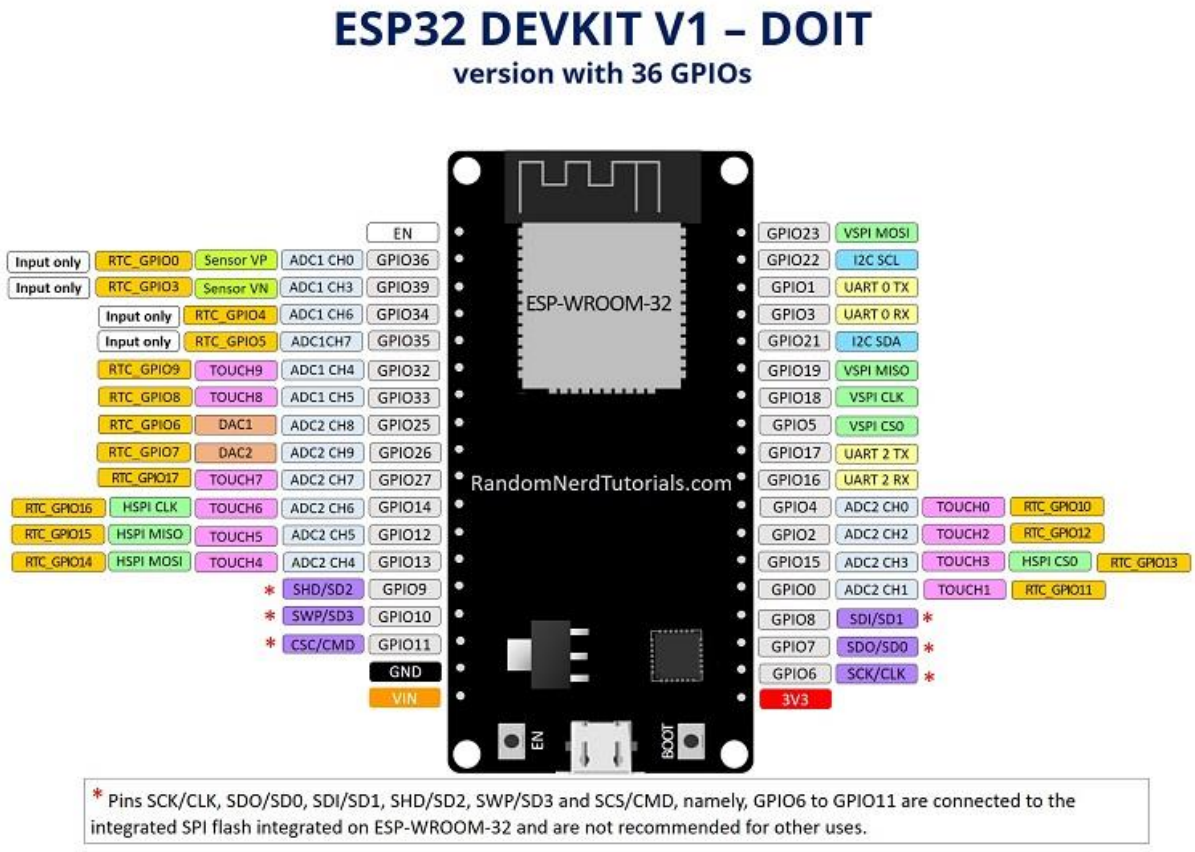
Η ασφάλεια αποτελεί πρωταρχικό μέλημα στην ανάπτυξη συνδεδεμένων συσκευών και το ESP32 το αντιμετωπίζει με μια σειρά λειτουργιών που έχουν σχεδιαστεί για την προστασία των δεδομένων και τη διασφάλιση της ακεραιότητας των επικοινωνιών. Η κρυπτογράφηση με επιτάχυνση υλικού, η

ασφαλής εκκίνηση και η κρυπτογράφηση flash είναι μόνο μερικές από τις προηγμένες δυνατότητες ασφαλείας του ESP32, διασφαλίζοντας ότι οι εφαρμογές που έχουν δημιουργηθεί σε αυτήν την πλατφόρμα μπορούν να θεωρηθούν ασφαλείς και ασφαλείς.

Το ESP32 είναι κάτι περισσότερο από έναν απλό μικροελεγκτή. είναι μια ολοκληρωμένη λύση για ένα ευρύ φάσμα εφαρμογών, από απλές έως πολύπλοκες. Οι ισχυρές του δυνατότητες επεξεργασίας, οι επιλογές συνδεσιμότητας, η ενεργειακή απόδοση και τα ισχυρά χαρακτηριστικά ασφαλείας το καθιστούν ιδανική επιλογή για προγραμματιστές και καινοτόμους που θέλουν να πραγματοποιήσουν τις ιδέες τους. Καθώς το ESP32 συνεχίζει να εξελίσσεται και η κοινότητα γύρω του μεγαλώνει, μπορούμε να περιμένουμε να δούμε ακόμα πιο εκπληκτικές εφαρμογές και εξελίξεις που προκύπτουν από αυτήν την αξιοσημείωτη τεχνολογία. [2]

Το τσιπ ESP32 διαθέτει 48 ακίδες με πολλαπλές λειτουργίες. Δεν είναι εκτεθειμένες όλες οι ακίδες σε όλες τις πλακέτες ανάπτυξης ESP32 και ορισμένες ακίδες δεν μπορούν να χρησιμοποιηθούν.

Το παρακάτω σχήμα απεικονίζει το pinout ESP-WROOM-32. Μπορείτε να το χρησιμοποιήσετε ως αναφορά εάν χρησιμοποιείτε γυμνό τσιπ ESP32 για να δημιουργήσετε μια προσαρμοσμένη πλακέτα:



Εικόνα 2.1: esp32 wroom-32 [3]

Τα περιφερειακά ESP32 περιλαμβάνουν:

- 18 Analog-to-Digital Converter (ADC) channels
- 3 SPI interfaces
- 3 UART interfaces
- 2 I2C interfaces
- 16 PWM output channels
- 2 Digital-to-Analog Converters (DAC)
- 2 I2S interfaces
- 10 Capacitive sensing GPIOs

Οι δυνατότητες ADC (μετατροπέας αναλογικού σε ψηφιακό) και DAC (μετατροπέας ψηφιακού σε αναλογικό) αντιστοιχίζονται σε συγκεκριμένες στατικές ακίδες. Ωστόσο, μπορείτε να αποφασίσετε ποιες ακίδες είναι UART, I2C, SPI, PWM κ.λπ. – απλά πρέπει να τις αντιστοιχίσετε στον κωδικό. Αυτό είναι δυνατό λόγω της δυνατότητας πολυπλεξίας του τσιπ ESP32.

Παρόλο που μπορείτε να ορίσετε τις ιδιότητες των ακίδων στο λογισμικό, υπάρχουν ακίδες που έχουν εκχωρηθεί από προεπιλογή όπως φαίνεται στην παρακάτω εικόνα (αυτό είναι ένα παράδειγμα για την πλακέτα ESP32 DEVKIT V1 DOIT με 36 ακίδες – η θέση της ακίδας μπορεί να αλλάξει ανάλογα με τον κατασκευαστή).

Επιπλέον, υπάρχουν καρφίτσες με συγκεκριμένα χαρακτηριστικά που τα καθιστούν κατάλληλα ή όχι για ένα συγκεκριμένο έργο. Ο παρακάτω πίνακας δείχνει ποιες ακίδες είναι καλύτερο να χρησιμοποιηθούν ως είσοδοι, έξοδοι και ποιες πρέπει να είστε προσεκτικοί.

Οι πινέζες που επισημαίνονται με πράσινο είναι OK για χρήση. Αυτά που επισημαίνονται με κίτρινο χρώμα είναι εντάξει για χρήση, αλλά πρέπει να δώσετε προσοχή γιατί μπορεί να έχουν μια απροσδόκητη συμπεριφορά κυρίως κατά την εκκίνηση. Οι ακίδες που επισημαίνονται με κόκκινο δεν συνιστάται να χρησιμοποιούνται ως είσοδοι ή έξοδοι.

Οι ακίδες ADC2 δεν μπορούν να χρησιμοποιηθούν όταν χρησιμοποιείται Wi-Fi. Επομένως, εάν χρησιμοποιείτε Wi-Fi και αντιμετωπίζετε προβλήματα με τη λήψη της τιμής από ένα ADC2 GPIO, μπορείτε να εξετάσετε το ενδεχόμενο να χρησιμοποιήσετε ένα ADC1 GPIO. Αυτό θα πρέπει να λύσει το πρόβλημά σας.

Τα κανάλια εισόδου ADC έχουν ανάλυση 12 bit. Αυτό σημαίνει ότι μπορείτε να λάβετε αναλογικές μετρήσεις από 0 έως 4095, στις οποίες το 0 αντιστοιχεί σε 0V και 4095 έως 3,3V. Μπορείτε επίσης να ορίσετε την ανάλυση των καναλιών σας στον κώδικα και την περιοχή ADC.

Οι ακίδες ADC ESP32 δεν έχουν γραμμική συμπεριφορά. Πιθανότατα δεν θα μπορείτε να διακρίνετε μεταξύ 0 και 0,1 V ή μεταξύ 3,2 και 3,3 V. Πρέπει να το έχετε υπόψη σας όταν χρησιμοποιείτε τις ακίδες ADC.

2.2 Ο αναλογικός αισθητήρας ρεύματος DFRobot

Ο αναλογικός αισθητήρας ρεύματος DFRobot αντιπροσωπεύει μια σημαντική πρόοδο στον τομέα της τεχνολογίας ηλεκτρικών μετρήσεων. Αυτός ο αισθητήρας έχει σχεδιαστεί ειδικά για να μετράει έως και 20A εναλλασσόμενου ή συνεχούς ρεύματος, προσφέροντας μια ευέλικτη λύση για διάφορες εφαρμογές. Η ακρίβεια αυτού του αισθητήρα έγκειται στην ικανότητά του να παράγει αναλογική τάση εξόδου που είναι ευθέως ανάλογη με το ρεύμα που διαρρέει το καλώδιο. Είναι αυτή η δυνατότητα που επιτρέπει την παρακολούθηση του ρεύματος σε πραγματικό χρόνο, μια ουσιαστική πτυχή για κάθε σύστημα που στοχεύει να παρέχει λεπτομερείς πληροφορίες σχετικά με τη χρήση ενέργειας. Η αναλογική φύση του αισθητήρα διευκολύνει την εύκολη ενσωμάτωση με μικροελεγκτές, όπως ο ESP32, ο οποίος μπορεί να δειγματίσει το αναλογικό σήμα και να το μετατρέψει σε ψηφιακή μορφή για περαιτέρω επεξεργασία και ανάλυση. [4]

Εικόνα 2.2: Gravity Αναλογικός Αισθητήρας Ρεύματος AC (20A) [4]

Ο σχεδιασμός του αισθητήρα επικεντρώνεται στην ευκολία χρήσης και την ασφάλεια. Χρησιμοποιώντας έναν μη επεμβατικό αισθητήρα εφέ Hall, μετρά το μαγνητικό πεδίο που

δημιουργείται από τη ροή ρεύματος χωρίς να απαιτείται άμεση επαφή με τον ηλεκτρικό αγωγό υψηλής τάσης. Αυτή η προσέγγιση όχι μόνο απλοποιεί τη διαδικασία εγκατάστασης, εξαλείφοντας την ανάγκη για πολύπλοκη καλωδίωση, αλλά επίσης ενισχύει την ασφάλεια των χρηστών μειώνοντας τον κίνδυνο ηλεκτρολογικών κινδύνων. Η λειτουργία clamp-on του αισθητήρα επιτρέπει τη γρήγορη και ασφαλή σύνδεση σε ένα καλώδιο τροφοδοσίας, καθιστώντας τον κατάλληλο τόσο για τους λάτρεις των DIY όσο και για τους επαγγελματίες. [5]

Η ενσωμάτωση του αναλογικού αισθητήρα ρεύματος DFRobot με έναν μικροελεγκτή ESP32 αναδεικνύει πλήρως τις δυνατότητές του. Οι μετατροπείς αναλογικού σε ψηφιακό (ADC) του ESP32 διαβάζουν την έξοδο στάθμης τάσης από τον αισθητήρα, η οποία συσχετίζεται με το ρεύμα που διέρχεται από τον αγωγό. Με τη συνεχή δειγματοληψία αυτού του σήματος, το ESP32 μπορεί να υπολογίσει την τιμή RMS του ρεύματος, μια πολύτιμη μέτρηση τόσο σε οικιακές όσο και σε βιομηχανικές ρυθμίσεις. Αυτή η τιμή RMS παρέχει έναν τυποποιημένο τρόπο αναπαράστασης του ενεργού ρεύματος, ο οποίος είναι ιδιαίτερα χρήσιμος για εφαρμογές εναλλασσόμενου ρεύματος (AC) όπου το μέγεθος και η κατεύθυνση του ρεύματος ποικίλλουν με το χρόνο.

Η απόδοση του αισθητήρα δεν περιορίζεται σε απλές μετρήσεις ρεύματος. Όταν συνδυάζεται με το ESP32, γίνεται συστατικό ενός μεγαλύτερου συστήματος με δυνατότητα λεπτομερούς ανάλυσης ποιότητας ισχύος. Η υπολογιστική ισχύς του ESP32 μπορεί να αξιοποιηθεί για τον υπολογισμό πρόσθετων παραμέτρων όπως ο συντελεστής ισχύος, η πραγματική ισχύς και η άεργος ισχύς, προσφέροντας μια ολοκληρωμένη κατανόηση της απόδοσης ενός ηλεκτρικού συστήματος. Αυτό καθιστά τον αναλογικό αισθητήρα ρεύματος DFRobot απαραίτητο εργαλείο για συστήματα διαχείρισης ενέργειας, προγνωστική συντήρηση, ακόμη και ανίχνευση σφαλμάτων.

Επιπλέον, η αναλογική έξοδος του αισθητήρα μπορεί να βαθμονομηθεί για να διασφαλιστεί η ακρίβεια και η ακρίβεια στις μετρήσεις του. Αυτή η βαθμονόμηση μπορεί να αντισταθμίσει τυχόν μη γραμμικότητες ή ασυνέπειες στην έξοδο του αισθητήρα, παρέχοντας στους χρήστες αξιόπιστα δεδομένα στα οποία μπορούν να βασίσουν τις αποφάσεις τους για την κατανάλωση ενέργειας. Η βαθμονόμηση λαμβάνει επίσης υπόψη περιβαλλοντικούς παράγοντες που μπορεί να επηρεάσουν την απόδοση του αισθητήρα, όπως αλλαγές θερμοκρασίας, διασφαλίζοντας σταθερή λειτουργικότητα υπό διαφορετικές συνθήκες.

Η απλή αναλογική διεπαφή του επιτρέπει την ανάπτυξη πολλαπλών αισθητήρων σε ένα μόνο σύστημα χωρίς σημαντικές αυξήσεις πολυπλοκότητας. Αυτή η επεκτασιμότητα είναι ζωτικής σημασίας για εφαρμογές παρακολούθησης μεγάλης κλίμακας, όπως κέντρα δεδομένων ή εγκαταστάσεις παραγωγής, όπου απαιτούνται πολλαπλά σημεία τρέχουσας μέτρησης για να ληφθεί μια πλήρης εικόνα της χρήσης ενέργειας.

Ο αναλογικός αισθητήρας ρεύματος DFRobot είναι βασικό συστατικό στη μετάβαση προς πιο έξυπνες ενεργειακές λύσεις. Ο συνδυασμός ακρίβειας, ευκολίας χρήσης και ασφάλειας το καθιστά εξαιρετική

επιλογή για όποιον θέλει να εφαρμόσει την ανίχνευση ρεύματος στα έργα του. Όταν ενσωματώνεται με προηγμένους μικροελεγκτές όπως το ESP32, γίνεται ακρογωνιαίος λίθος έξυπνων συστημάτων παρακολούθησης ενέργειας που μπορούν να οδηγήσουν σε πιο αποτελεσματικές και βιώσιμες πρακτικές κατανάλωσης ενέργειας. Αυτός ο αισθητήρας αποτελεί ένα λαμπρό παράδειγμα του πώς η σύγχρονη τεχνολογία μπορεί να ενισχύσει την ικανότητά μας να κατανοούμε και να διαχειριζόμαστε τους ηλεκτρικούς πόρους που τροφοδοτούν τον κόσμο μας.

Τα χαρακτηριστικά του είναι: [6]

AC Current Signal Conversion Module

- Input Voltage (VCC): 3.3V-5.5V
- Interface: Gravity Analog (PH2.0-3P, analog voltage output 0.2-2.8V DC)
- AC Voltage Input Range: 0-1V (AC RMS)
- Relative Error: ±4%
- Dimension: 32×27 mm /1.26×1.06 in
- Weight: 5g

Open Type AC Transformer Probe

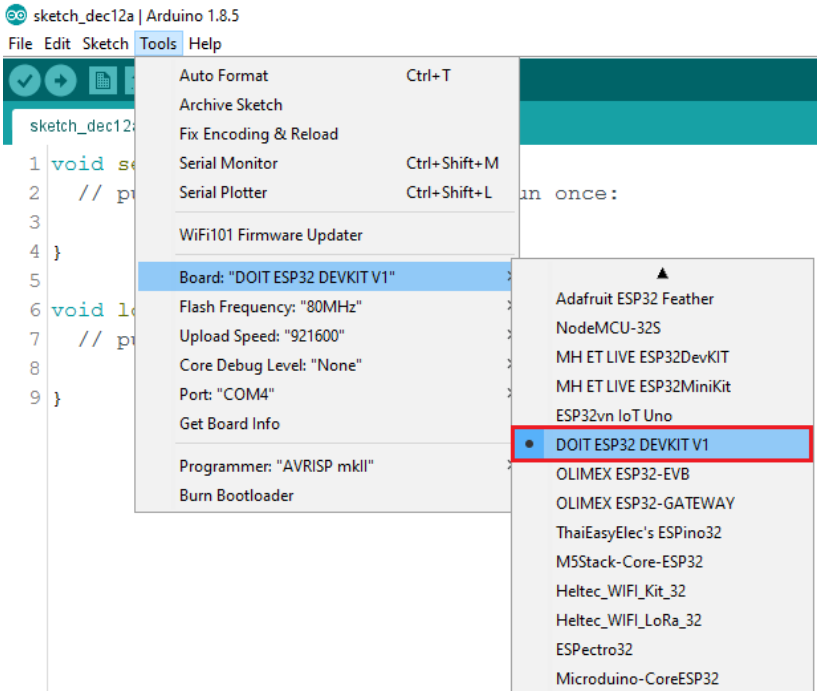
- AC Current Range: 0-20A
- Signal Output (standard Φ3.5mm 3P plug): 0-1V AC voltage, linear corresponding range 0-20A
- Accuracy: ±1%
- Non-linearity: ≤±0.2%
- Frequency Range: 50Hz~1kHz
- Cable Length: 1m
- Working Temperature: -25 ℃~+70 ℃
- Opening Size: 13×13 mm / 0.51×0.51 in
- Weight: 50g

Κεφάλαιο 3ο: Γλώσσες Προγραμματισμού και Τεχνολογικά Εργαλεία

Στο κεφάλαιο αυτό θα περιγράφει η τεχνολογία που χρησιμοποιήθηκε για την υλοποίηση του συστήματος παρακολούθησης κατανάλωσης ρεύματος μέσω διαδικτύου.

3.1 Arduino IDE

Το Arduino IDE, μαζί με τις εκτεταμένες δυνατότητές του, έχει φέρει επανάσταση στη σφαίρα του προγραμματισμού μικροελεγκτών, ιδιαίτερα όταν συνδυάζεται με ισχυρές μονάδες όπως το ESP32. Αυτός ο καινοτόμος συνδυασμός όχι μόνο παρέχει ένα φιλικό προς τον χρήστη περιβάλλον για κωδικοποίηση, αλλά επίσης ξεκλειδώνει τεράστιες δυνατότητες για τη δημιουργία εξελιγμένων έργων που κυμαίνονται από απλούς οικιακούς αυτοματισμούς έως πολύπλοκες εφαρμογές IoT. [7]



Εικόνα 3.1: Arduino IDE για esp32 [8]

Το Arduino IDE απλοποιεί τη διαδικασία ανάπτυξης προσφέροντας μια διαισθητική διεπαφή και μια πληθώρα βιβλιοθηκών και εργαλείων. Το απλό περιβάλλον προγραμματισμού του, που βασίζεται σε C/C++, το καθιστά προσβάσιμο σε αρχάριους, ενώ εξακολουθεί να είναι αρκετά ισχυρό για έμπειρους προγραμματιστές. Η ισχυρή κοινότητα και η εκτεταμένη τεκμηρίωση της πλατφόρμας ενισχύουν

περαιτέρω την εμπειρία μάθησης και ανάπτυξης, παρέχοντας έναν ανεκτίμητο πόρο για την αντιμετώπιση προβλημάτων και την καινοτομία.

Η ενσωμάτωση του ESP32 με το Arduino IDE είναι μια απλή διαδικασία, χάρη στον πυρήνα ESP32 που διατίθεται μέσω του Arduino Board Manager. Αυτή η ενοποίηση όχι μόνο επιτρέπει τις τυπικές λειτουργίες του Arduino, αλλά εκθέτει επίσης τα πλούσια χαρακτηριστικά του ESP32, επιτρέποντας στους προγραμματιστές να αξιοποιήσουν πλήρως τις δυνατότητές του. Από τον απλό έλεγχο GPIO έως τις πολύπλοκες λειτουργίες δικτύου, οι προγραμματιστές μπορούν να γράψουν κώδικα που αξιοποιεί τις δυνατότητες του ESP32 απολαμβάνοντας την απλότητα και την εξοικείωση του μοντέλου προγραμματισμού Arduino.

Ένα από τα αξιοσημείωτα χαρακτηριστικά του προγραμματισμού του ESP32 στο Arduino IDE είναι η δυνατότητα δημιουργίας εφαρμογών σε πραγματικό χρόνο. Ο επεξεργαστής διπλού πυρήνα του ESP32 επιτρέπει την εκτέλεση πολλαπλών εργασιών, όπου μπορούν να χειριστούν ξεχωριστές λειτουργίες ταυτόχρονα, οδηγώντας σε εφαρμογές με μεγαλύτερη απόκριση και αποτελεσματικότητα. Αυτό είναι ιδιαίτερα ωφέλιμο σε σενάρια όπου οι εργασίες κρίσιμες για το χρόνο πρέπει να διαχειρίζονται χωρίς να διακόπτεται η ροή της κύριας εφαρμογής.

Η συνδεσιμότητα είναι ένας άλλος ακρογωνιαίος λίθος του ESP32, με τις δυνατότητες Wi-Fi και Bluetooth να ανοίγουν πόρτες σε μυριάδες εφαρμογές IoT. Το Arduino Studio διευκολύνει την ανάπτυξη έργων που συνδέονται με το δίκτυο, επιτρέποντας στις συσκευές να επικοινωνούν μεταξύ τους, να ανταλλάσσουν δεδομένα με διακομιστές ή ακόμα και να έχουν πρόσβαση σε υπηρεσίες cloud. Αυτή η συνδεσιμότητα, σε συνδυασμό με την επεξεργαστική ισχύ του ESP32, ανοίγει το δρόμο για εξελιγμένα έργα όπως συστήματα οικιακού αυτοματισμού, αισθητήρες τηλεχειρισμού και ηλεκτρονική καταγραφή δεδομένων.

Η συγχώνευση του Arduino IDE και της μονάδας ESP32 προσφέρει μια ισχυρή πλατφόρμα τόσο για χομπίστες όσο και για επαγγελματίες. Εξισορροπεί την ευκολία χρήσης με τις ισχυρές δυνατότητες, επιτρέποντας τη δημιουργία μιας μεγάλης ποικιλίας έργων. Είτε πρόκειται για την εκμάθηση των βασικών αρχών του προγραμματισμού μικροελεγκτών είτε για την ανάπτυξη σύνθετων λύσεων IoT, αυτός ο συνδυασμός αποτελεί φάρο καινοτομίας και πρακτικότητας στον τομέα της ηλεκτρονικής και της ανάπτυξης λογισμικού.

3.2 Python

Η Python αποτελεί τον ακρογωνιαίο λίθο της σύγχρονης ανάπτυξης λογισμικού, μια ευέλικτη γλώσσα που εκτείνεται σε απλές δέσμες ενεργειών έως πολύπλοκα πλαίσια εφαρμογών ιστού και ανάλυση δεδομένων. Η σημασία του είναι ιδιαίτερα εμφανής στη σφαίρα της ανάπτυξης ιστού, όπου πλαίσια όπως το Flask έχουν γίνει συνώνυμα με την ταχεία, αποτελεσματική και επεκτάσιμη ανάπτυξη

εφαρμογών. Το Flask, ένα micro web πλαίσιο γραμμένο σε Python, προσφέρει μια ρεαλιστική και καθαρή προσέγγιση στις υπηρεσίες Ιστού, παρέχοντας στους προγραμματιστές τα εργαλεία που χρειάζονται για να δημιουργήσουν ισχυρές εφαρμογές Ιστού διατηρώντας παράλληλα έναν ελαφρύ πυρήνα. Αυτή η ευελιξία είναι ζωτικής σημασίας για την ανάπτυξη ενός διαδικτυακού συστήματος για την παρακολούθηση της κατανάλωσης ηλεκτρικού ρεύματος, καθώς επιτρέπει την απρόσκοπτη ενσωμάτωση με διάφορα στοιχεία υλικού και λογισμικού, καθώς και τη δυνατότητα κλιμάκωσης ανάλογα με τις αυξανόμενες απαιτήσεις των χρηστών και των δεδομένων. [9]

Το οικοσύστημα Python είναι πλούσιο με βιβλιοθήκες και ενότητες που επεκτείνουν τη λειτουργικότητά του, επιτρέποντας ένα ευρύ φάσμα εφαρμογών. Για το χειρισμό και την επεξεργασία δεδομένων, βιβλιοθήκες όπως οι NumPy και Pandas προσφέρουν ισχυρές δομές και λειτουργίες που απλοποιούν πολύπλοκες λειτουργίες δεδομένων. Όταν πρόκειται για εφαρμογές Ιστού, η ικανότητα της Python να διασυνδέεται με βάσεις δεδομένων και να εκτελεί λογική από την πλευρά του διακομιστή ενισχύεται από τη μινιμαλιστική αλλά επεκτάσιμη αρχιτεκτονική του Flask. Αυτή η συνέργεια είναι ιδιαίτερα αποτελεσματική όταν πρόκειται για επεξεργασία και οπτικοποίηση δεδομένων σε πραγματικό χρόνο, όπως απαιτείται στο σύστημα παρακολούθησης ηλεκτρικού ρεύματος που συζητείται σε αυτό το έργο.

Η σχεδιαστική φιλοσοφία της Python δίνει έμφαση στην αναγνωσιμότητα κώδικα και σε μια σύνταξη που επιτρέπει στους προγραμματιστές να εκφράζουν έννοιες σε λιγότερες γραμμές κώδικα από ό,τι θα ήταν δυνατό σε γλώσσες όπως η C++ ή η Java. Αυτή η αναγνωσιμότητα δεν είναι μόνο θέμα αισθητικής. συμβάλλει άμεσα στη συντηρησιμότητα και την επεκτασιμότητα της βάσης κωδικών. Όταν ασχολούμαστε με την περίπλοκη λογική που απαιτείται για το χειρισμό των περιόδων σύνδεσης των χρηστών, τον έλεγχο ταυτότητας αιτημάτων και την επεξεργασία δεδομένων αισθητήρων σε πραγματικό χρόνο, η σαφής σύνταξη και η πανίσχυρη τυπική βιβλιοθήκη της Python επιτρέπουν στους προγραμματιστές να εφαρμόζουν εξελιγμένες λειτουργίες με σαφήνεια και ακρίβεια.

Στο πλαίσιο του IoT, η Python χρησιμεύει ως γέφυρα μεταξύ του υλικού και των επιπέδων Ιστού. Ο μικροελεγκτής ESP32 μπορεί να προγραμματιστεί χρησιμοποιώντας υλικολογισμικό που βασίζεται σε Python, το οποίο απλοποιεί την εργασία ανάγνωσης από αισθητήρες και την επικοινωνία με τον διακομιστή web. Αυτή η ευκολία συνδεσιμότητας είναι ο λόγος που η Python είναι συχνά η γλώσσα επιλογής για έργα IoT, καθώς μπορεί να μειώσει την πολυπλοκότητα του προγραμματισμού συσκευών και να διευκολύνει την ταχεία ανάπτυξη και ανάπτυξη εφαρμογών IoT.

Επιπλέον, η δύναμη της Python αποδεικνύεται στον χειρισμό της ταυτόχρονης λειτουργίας, η οποία είναι απαραίτητη για έναν διακομιστή ιστού που αναμένεται να χειρίζεται πολλαπλά αιτήματα ταυτόχρονα. Το Flask, όταν συνδυάζεται με κατάλληλες βιβλιοθήκες Python, διαχειρίζεται τις ταυτόχρονες συνεδρίες με ασφάλεια, διατηρώντας τα δεδομένα κάθε χρήστη απομονωμένα και διαθέσιμα σε όλη την αλληλεπίδρασή τους με το σύστημα. Αυτό είναι ζωτικής σημασίας για τη

διατήρηση της ακεραιότητας της εμπειρίας χρήστη, διασφαλίζοντας ότι τα δεδομένα σε πραγματικό χρόνο παρουσιάζονται με συνέπεια στους χρήστες χωρίς καθυστερήσεις ή σφάλματα.

Το Flask επεκτείνει την εμβέλεια της Python στον τομέα ιστού, παρέχοντας μια σταθερή, επεκτάσιμη βάση για εφαρμογές web χωρίς να κλειδώνει τους προγραμματιστές σε μια μονολιθική δομή πλαισίου. Προσφέρει την ελευθερία επιλογής των στοιχείων που ταιριάζουν καλύτερα στις απαιτήσεις του έργου, από τη βάση δεδομένων ORM έως τη βιβλιοθήκη επικύρωσης φορμών, καθιστώντας την ιδανική επιλογή για ένα έργο που χρειάζεται ευελιξία και αξιοπιστία.

Η εκτεταμένη υποστήριξη κοινότητας της Python και το πλούσιο αποθετήριο τεκμηρίωσης και οι βιβλιοθήκες τρίτων συμβάλλουν σημαντικά στη δημοτικότητά της. Για το σύστημα παρακολούθησης ηλεκτρικού ρεύματος, η δυνατότητα αξιοποίησης των υπαρχουσών βιβλιοθηκών για χαρτογράφηση, κρυπτογράφηση δεδομένων και διαχείριση συνεδριών επιταχύνει την ανάπτυξη και διασφαλίζει ότι η εφαρμογή βασίζεται σε καλά δοκιμασμένες και ευρέως υιοθετημένες λύσεις. Αυτή η προσέγγιση με γνώμονα την κοινότητα στην ανάπτυξη λογισμικού σημαίνει ότι η εφαρμογή δεν είναι απλώς προϊόν μιας μεμονωμένης ομάδας, αλλά μέρος ενός ευρύτερου οικοσυστήματος προγραμματιστών και τεχνολογιών Python.

3.2.1 Flask

Το Flask, ένα ελαφρύ και ισχυρό πλαίσιο web για την Python, ξεχωρίζει στον κόσμο της ανάπτυξης ιστού για την απλότητα και την ευελιξία του. Αυτό το μικροπλαίσιο, αν και μινιμαλιστικό στον πυρήνα του, είναι αρκετά στιβαρό ώστε να υποστηρίζει την ανάπτυξη πολύπλοκων διαδικτυακών εφαρμογών. Προτιμάται ιδιαίτερα από προγραμματιστές που εκτιμούν τον καθαρό και διατηρήσιμο κώδικα, καθώς το Flask ενθαρρύνει τη χρήση απλής και ευανάγνωστης σύνταξης. Αυτό το χαρακτηριστικό το καθιστά εξαιρετική επιλογή τόσο για αρχάριους που μαθαίνουν τα σχοινιά της ανάπτυξης ιστού όσο και για έμπειρους επαγγελματίες που στοχεύουν στην αποτελεσματική κατασκευή αξιόπιστων διαδικτυακών υπηρεσιών. [10]

Η βασική φιλοσοφία του Flask περιστρέφεται γύρω από την απλότητα, τον μινιμαλισμό και τον λεπτό έλεγχο. Σε αντίθεση με άλλα πλαίσια που διαθέτουν πληθώρα ενσωματωμένων λειτουργιών, το Flask παρέχει το ελάχιστο δυνατό για την εκτέλεση μιας διαδικτυακής εφαρμογής, αφήνοντας την επιλογή πρόσθετων στοιχείων και προσθηκών στον προγραμματιστή. Αυτό όχι μόνο μειώνει τα γενικά έξοδα, αλλά δίνει επίσης στους προγραμματιστές την ελευθερία να προσαρμόσουν την εφαρμογή τους σύμφωνα με τις συγκεκριμένες ανάγκες τους. Ο ενσωματωμένος διακομιστής ανάπτυξης του πλαισίου και το γρήγορο πρόγραμμα εντοπισμού σφαλμάτων εξορθολογίζουν περαιτέρω τη διαδικασία ανάπτυξης, καθιστώντας το Flask ένα εξαιρετικά αποτελεσματικό εργαλείο για την ανάπτυξη ιστού.

Η ικανότητα του Flask να ενσωματώνεται απρόσκοπτα με άλλες βιβλιοθήκες και εργαλεία Python είναι ένα από τα πιο σημαντικά πλεονεκτήματά του. Για παράδειγμα, το Flask μπορεί εύκολα να συνδυαστεί με SQL Alchemy ή Django ORM για λειτουργίες βάσης δεδομένων ή με Marshmallow για σειριοποίηση και αποσειροποίηση σύνθετων τύπων δεδομένων. Αυτή η διαλειτουργικότητα επιτρέπει στο Flask να είναι απίστευτα ευέλικτο, καλύπτοντας ένα ευρύ φάσμα αναγκών ανάπτυξης ιστού, από τη δημιουργία απλών API έως την εξυπηρέτηση σύνθετων εφαρμογών ιστού.

Με λίγες μόνο γραμμές κώδικα, μπορείτε να εκτελείτε μια βασική εφαρμογή Ιστού. Εδώ είναι ένα απλό παράδειγμα:

```
from flask import Flask

app = Flask(__name__)


@app.route('/')

def hello_world():

    return 'Hello, World!'


if __name__ == '__main__':

    app.run(debug=True)
```

Αυτή η βασική εφαρμογή ρυθμίζει μια διαδρομή προς τη ριζική διεύθυνση URL ('/') και ορίζει μια συνάρτηση που θα καλείται κατά την πρόσβαση σε αυτήν τη διαδρομή, επιστρέφοντας τη συμβολοσειρά 'Hello, World!'. Με την εκτέλεση αυτής της εφαρμογής ξεκινά ένας τοπικός διακομιστής ανάπτυξης στον οποίο μπορείτε να έχετε πρόσβαση μέσω ενός προγράμματος περιήγησης Ιστού.

Η απλότητα του Flask επεκτείνεται και σε πιο σύνθετες λειτουργίες. Για παράδειγμα, η ενσωμάτωση του ελέγχου ταυτότητας χρήστη, ένα κρίσιμο συστατικό των σύγχρονων διαδικτυακών εφαρμογών, μπορεί να είναι μια αποθαρρυντική εργασία. Ωστόσο, με το Flask, αυτό μπορεί να διευκολυνθεί μέσω επεκτάσεων όπως το Flask-Login ή το Flask-Security. Αυτές οι επεκτάσεις αφαιρούν την πολυπλοκότητα που συνεπάγεται η διαχείριση των περιόδων σύνδεσης των χρηστών και παρέχουν ένα απλό μοτίβο για το χειρισμό των συνδέσεων, καθιστώντας έτσι την ανάπτυξη ασφαλών εφαρμογών πιο προσιτή.

Επιπλέον, η αποστολή αιτημάτων RESTful του Flask είναι ιδιαίτερα κατάλληλη για τη δημιουργία RESTful API. Χρησιμοποιώντας την επέκταση Flask-RESTful, οι προγραμματιστές μπορούν να

σχεδιάσουν και να δημιουργήσουν αποτελεσματικά κλιμακούμενα και διατηρούμενα API REST. Ακολουθεί ένα σύντομο παράδειγμα που δείχνει τη δημιουργία ενός απλού τερματικού σημείου API:

```
from flask import Flask
from flask_restful import Resource, Api

app = Flask(__name__)
api = Api(app)

class HelloWorld(Resource):
    def get(self):
        return {'hello': 'world'}

api.add_resource(HelloWorld, '/')

if __name__ == '__main__':
    app.run(debug=True)
```

Σε αυτό το απόσπασμα, ένας πόρος (HelloWorld) ορίζεται με μια μέθοδο λήψης, επιστρέφοντας μια απλή απόκριση JSON όταν αποκτάται πρόσβαση μέσω αιτήματος GET. Στη συνέχεια, ο πόρος προστίθεται στο API με ένα καθορισμένο τελικό σημείο ('/').

Το Flask ενσωματώνει την αρχή της απλότητας χωρίς να θυσιάζει τη δύναμη. Ο ελαφρύς πυρήνας του, σε συνδυασμό με την επεκτασιμότητα να περιλαμβάνει όσες ή λιγότερες δυνατότητες χρειάζεται, το καθιστά ένα υποδειγματικό πλαίσιο για όλα τα επίπεδα ανάπτυξης ιστού. Από απλές διαδικτυακές εφαρμογές έως πολύπλοκους ιστότοπους υψηλής επισκεψιμότητας, το Flask παρέχει τα εργαλεία και την ευελιξία που απαιτούνται για να πραγματοποιήσετε αποτελεσματικά τα έργα σας. Είτε είστε αρχάριος που μόλις ξεκινάτε το ταξίδι σας στην ανάπτυξη ιστού είτε έμπειρος προγραμματιστής που αναζητά ένα ρεαλιστικό και αποτελεσματικό πλαίσιο, το Flask αποτελεί μια συναρπαστική επιλογή.

3.2.2 Restful EndPoint API

Στο σύγχρονο ψηφιακό τοπίο, η αποτελεσματικότητα των εφαρμογών που βασίζονται σε δεδομένα εξαρτάται σε μεγάλο βαθμό από την ευρωστία των υπηρεσιών backend τους. Η Python, με την πληθώρα των πλαισίων της, έχει αναδειχθεί ως ηγετική γλώσσα για τη δημιουργία RESTful API, τα οποία είναι καίριας σημασίας στην αρχιτεκτονική των σύγχρονων διαδικτυακών υπηρεσιών. Τα RESTful API, ή τα Representational State Transfer API, συμμορφώνονται με ένα σύνολο αρχιτεκτονικών αρχών που επιτρέπουν σε διάφορες οντότητες λογισμικού να επικοινωνούν μέσω του Διαδικτύου με απρόσκοπτη και ανιθαγενή τρόπο. Λειτουργούν ως αγωγοί για την ανταλλαγή δεδομένων, διευκολύνοντας τις αλληλεπιδράσεις πελάτη-διακομιστή που είναι θεμελιώδεις για εφαρμογές ιστού, κινητών και IoT.[11]

Το Flask, ένα micro web πλαίσιο γραμμένο σε Python, είναι ιδιαίτερα κατάλληλο για τη δημιουργία RESTful API λόγω της απλότητας και της ευελιξίας του. Επιτρέπει στους προγραμματιστές να δημιουργούν γρήγορα επεκτάσιμες και διατηρούμενες υπηρεσίες web που μπορούν να χειριστούν πλήθος αιτημάτων από διάφορους πελάτες. Η ελαφριά φύση του Flask το καθιστά ιδανική επιλογή για τη δημιουργία τελικών σημείων που εκτελούν συγκεκριμένες λειτουργίες, όπως η ανάκτηση δεδομένων από μια βάση δεδομένων, η επεξεργασία τους και η επιστροφή τους σε δομημένη μορφή όπως το JSON. Με τη μόχλευση του Flask, οι προγραμματιστές μπορούν να κατασκευάσουν αποτελεσματικά τελικά σημεία που συμμορφώνονται με τις αρχές REST, όπως η ανικανότητα και η προσωρινή αποθήκευση, τα οποία είναι κρίσιμα για την επεκτασιμότητα και την απόδοση των εφαρμογών Ιστού.

Η κατασκευή ενός τερματικού σημείου RESTful ξεκινά με τον καθορισμό της διαδρομής και της μεθόδου HTTP στην οποία πρέπει να ανταποκρίνεται. Για παράδειγμα, ένα αίτημα GET σε μια διαδρομή Flask θα ανακτούσε δεδομένα, ενώ ένα αίτημα POST θα μπορούσε να χρησιμοποιηθεί για την υποβολή νέων δεδομένων στον διακομιστή. Το Flask απλοποιεί αυτή τη διαδικασία με διακοσμητές που συνδέουν λειτουργίες σε συγκεκριμένες διευθύνσεις URL, επιτρέποντας τη δημιουργία καθαρού και ευανάγνωστου κώδικα για κάθε τελικό σημείο API. Αυτά τα τελικά σημεία είναι τα δομικά στοιχεία μιας υπηρεσίας RESTful, καθένα από τα οποία εξυπηρετεί έναν ξεχωριστό σκοπό στη συνολική λογική της εφαρμογής.

Ένα RESTful τελικό σημείο στο Flask για την ανάκτηση όλων των βιβλίων θα μπορούσε να έχει ως εξής:

```
from flask import Flask, jsonify

app = Flask(__name__)
```

```
books = [
    {'id': 1, 'title': '1984', 'author': 'George Orwell'},
    {'id': 2, 'title': 'The Great Gatsby', 'author': 'F. Scott Fitzgerald'},
    # More books
]


@app.route('/books', methods=['GET'])
def get_books():
    return jsonify({'books': books})
```

Σε αυτό το παράδειγμα, όταν υποβάλλεται αίτημα GET στη διαδρομή '/books', καλείται η συνάρτηση get_books, επιστρέφοντας μια απάντηση JSON που περιέχει μια λίστα βιβλίων. Αυτό το τελικό σημείο είναι απλό και αποτελεσματικό, ενσωματώνοντας την αρχή REST του να είσαι ανιθαγενής και να μπορείς εύκολα να αποθηκευτεί στην κρυφή μνήμη.

Ομοίως, για να προσθέσετε ένα νέο βιβλίο στη βιβλιοθήκη, θα μπορούσε να δημιουργηθεί ένα τελικό σημείο POST:

```
from flask import request


@app.route('/books', methods=['POST'])
def add_book():
    new_book = request.get_json()
    books.append(new_book)
    return jsonify(new_book), 201
```

Εδώ, η συνάρτηση add_book εξάγει το ωφέλιμο φορτίο JSON από το αίτημα, το προσθέτει στη λίστα βιβλίων και, στη συνέχεια, επιστρέφει τα νέα δεδομένα βιβλίου με έναν κωδικό κατάστασης 201 που υποδεικνύει ότι ένας πόρος δημιουργήθηκε με επιτυχία.

Η ασφάλεια στα RESTful API είναι πρωταρχικής σημασίας, καθώς συχνά ασχολούνται με ευαίσθητα δεδομένα. Το Flask υποστηρίζει διάφορες μεθόδους για την εξασφάλιση τελικών σημείων, συμπεριλαμβανομένου του ελέγχου ταυτότητας βάσει διακριτικών και του ελέγχου πρόσβασης βάσει ρόλων. Χρησιμοποιώντας αυτά τα μέτρα ασφαλείας, τα τελικά σημεία API μπορούν να διασφαλίσουν ότι μόνο εξουσιοδοτημένοι χρήστες μπορούν να έχουν πρόσβαση ή να τροποποιούν πόρους.

Επιπλέον, ο χειρισμός σφαλμάτων είναι μια κρίσιμη πτυχή του σχεδιασμού ισχυρών RESTful API. Το Flask επιτρέπει στους προγραμματιστές να ορίζουν προσαρμοσμένους χειριστές σφαλμάτων που μπορούν να χειριστούν με χάρη διάφορα σενάρια αποτυχίας, όπως μη έγκυρα αιτήματα ή σφάλματα διακομιστή, διασφαλίζοντας ότι ο πελάτης λαμβάνει μια σαφή και ενημερωτική απάντηση ακόμη και όταν τα πράγματα πάνε στραβά.

Η τεκμηρίωση είναι ένα άλλο ζωτικής σημασίας στοιχείο ενός επιτυχημένου RESTful API. Εργαλεία όπως το Swagger μπορούν να ενσωματωθούν στο Flask για τη δημιουργία διαδραστικής τεκμηρίωσης που περιγράφει τα τελικά σημεία του API, τις αναμενόμενες εισόδους και εξόδους τους. Αυτή η τεκμηρίωση είναι ανεκτίμητη όχι μόνο για εξωτερικούς προγραμματιστές που μπορεί να καταναλώνουν το API αλλά και για τη συντήρηση και την εσωτερική κλιμάκωση της υπηρεσίας.

Η Python και το Flask μαζί προσφέρουν μια ισχυρή εργαλειοθήκη για τη δημιουργία RESTful API που είναι επεκτάσιμα, διατηρούμενα και ασφαλή. Ακολουθώντας τις αρχές REST και χρησιμοποιώντας τις δυνατότητες του Flask, οι προγραμματιστές μπορούν να δημιουργήσουν υπηρεσίες Ιστού που λειτουργούν αποτελεσματικά ως η ραχοκοκαλιά για σύγχρονες εφαρμογές Ιστού, παρέχοντας την απαραίτητη λειτουργικότητα για τη διαχείριση διαφορετικών αιτημάτων πελατών και εργασιών επεξεργασίας δεδομένων.

3.3 MySQL

Η MySQL, ένα από τα πιο δημοφιλή συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων ανοιχτού κώδικα στον κόσμο, είναι ευρέως αναγνωρισμένο για την αξιοπιστία, την απόδοση και την ευκολία χρήσης του. Αποτελεί τη ραχοκοκαλιά αμέτρητων διαδικτυακών εφαρμογών, που κυμαίνονται από έργα μικρής κλίμακας έως επιχειρηματικές λύσεις μεγάλης κλίμακας. Η συμβατότητά του με διάφορες γλώσσες προγραμματισμού, συμπεριλαμβανομένης της Python, το καθιστά μια απίστευτα ευέλικτη επιλογή για προγραμματιστές που θέλουν να δημιουργήσουν ισχυρές εφαρμογές που βασίζονται σε δεδομένα. [12]

Η Python, με την απλότητα και τις ισχυρές βιβλιοθήκες της, συνδυάζεται τέλεια με τη MySQL, επιτρέποντας στους προγραμματιστές να αλληλεπιδρούν με βάσεις δεδομένων με απλό αλλά αποτελεσματικό τρόπο. Βιβλιοθήκες όπως τη MySQL Connector/Python επιτρέπουν στους προγραμματιστές να γράφουν κώδικα Python που επικοινωνεί απρόσκοπτα με τις βάσεις δεδομένων

MySQL. Αυτή η ενοποίηση διευκολύνει εργασίες όπως η αναζήτηση δεδομένων, η εισαγωγή εγγραφών και η διαχείριση συναλλαγών, όλα με την άνεση της διαισθητικής σύνταξης της Python.

Μια κοινή εργασία κατά την εργασία με βάσεις δεδομένων είναι η ανάκτηση δεδομένων με βάση ορισμένες συνθήκες. Στην Python, αυτό μπορεί να επιτευχθεί χρησιμοποιώντας τη δήλωση SELECT σε συνδυασμό με την πρόταση WHERE. Για παράδειγμα, σκεφτείτε μια βάση δεδομένων που ονομάζεται Βιβλιοθήκη με έναν πίνακα που ονομάζεται Βιβλία. Για να λάβετε τους τίτλους όλων των βιβλίων που εκδόθηκαν μετά το έτος 2000, θα μπορούσατε να χρησιμοποιήσετε το ακόλουθο απόσπασμα κώδικα:

```
import mysql.connector


# Establish a connection to the database
cnx = mysql.connector.connect(user='username', password='password',
                              host='127.0.0.1',
                              database='Library')
cursor = cnx.cursor()


query = ("SELECT title FROM Books WHERE year > 2000")


cursor.execute(query)


for (title) in cursor:
    print("Title: {}".format(title))


cursor.close()
cnx.close()
```

Αυτός ο κώδικας δημιουργεί μια σύνδεση με τη βάση δεδομένων της Βιβλιοθήκης, εκτελεί το ερώτημα και εκτυπώνει τους τίτλους των βιβλίων που πληρούν την καθορισμένη συνθήκη.

Μια άλλη θεμελιώδης λειτουργία στη διαχείριση βάσεων δεδομένων είναι η εισαγωγή δεδομένων σε πίνακες. Η ενσωμάτωση της Python με τη MySQL καθιστά αυτή τη διαδικασία απλή. Ας υποθέσουμε ότι έχουμε έναν πίνακα με το όνομα Συγγραφείς στην ίδια βάση δεδομένων της Βιβλιοθήκης, με στήλες

για το αναγνωριστικό συγγραφέα, το όνομα και την ημερομηνία γέννησης. Για να εισαγάγετε έναν νέο συγγραφέα σε αυτόν τον πίνακα, μπορείτε να χρησιμοποιήσετε τον ακόλουθο κώδικα:

```
import mysql.connector
from datetime import date

cnx = mysql.connector.connect(user='username', password='password',
                              host='127.0.0.1',
                              database='Library')
cursor = cnx.cursor()

add_author = ("INSERT INTO Authors "
              "(author_id, name, birthdate) "
              "VALUES (%s, %s, %s)")
author_data = (1001, 'Jane Austen', date(1775, 12, 16))

cursor.execute(add_author, author_data)

# Commit the transaction
cnx.commit()

cursor.close()
cnx.close()
```

Σε αυτό το παράδειγμα, η πρόταση INSERT INTO προσθέτει μια νέα σειρά στον πίνακα Συγγραφείς. Τα σύμβολα κράτησης θέσης %s αντικαθίστανται με τις τιμές στην πλειάδα author_data, διασφαλίζοντας ότι τα δεδομένα εισάγονται με ασφάλεια χωρίς τον κίνδυνο εισαγωγής SQL.

Τα παραπάνω παραδείγματα δείχνουν μόνο ένα κλάσμα του τι μπορεί να επιτευχθεί με το συνδυασμό Python και MySQL. Αυτό το ισχυρό δίδυμο δίνει τη δυνατότητα στους προγραμματιστές να δημιουργούν αποτελεσματικά εξελιγμένες εφαρμογές που βασίζονται σε δεδομένα. Είτε διαχειρίζεστε δεδομένα χρήστη για μια εφαρμογή Ιστού, αναλύετε μεγάλα σύνολα δεδομένων είτε απλώς αυτοματοποιείτε μια διαδικασία που βασίζεται σε βάσεις δεδομένων, η συνέργεια μεταξύ Python και

MySQL παρέχει μια σταθερή βάση πάνω στην οποία μπορείτε να δημιουργήσετε ισχυρές και επεκτάσιμες λύσεις.

3.4 Chart.js

Το Chart.js είναι μια ευέλικτη και φιλική προς το χρήστη βιβλιοθήκη JavaScript που δίνει τη δυνατότητα στους προγραμματιστές να δημιουργούν μια ποικιλία από ανταποκρίσιμα και κινούμενα γραφήματα με ευκολία. Η ελκυστικότητά του έγκειται στην απλότητά του και στην ικανότητά του να ενσωματώνεται απρόσκοπτα με ιστοσελίδες, καθιστώντας το μια εξαιρετική επιλογή για την εμφάνιση δεδομένων με οπτικά ελκυστικό τρόπο. Η βιβλιοθήκη παρέχει ένα πλούσιο σύνολο τύπων γραφημάτων, όπως γραμμή, ράβδος, πίτα, ραντάρ και άλλα, καθένα με δυνατότητα προσαρμογής με μια σειρά επιλογών για την προσαρμογή της παρουσίασης σε συγκεκριμένες απαιτήσεις. Το στοιχείο καμβά στην HTML5 είναι ο πίνακας σχεδίασης για το Chart.js, διασφαλίζοντας ότι τα γραφήματα αποδίδονται με υψηλή απόδοση και συμβατότητα στα σύγχρονα προγράμματα περιήγησης ιστού. [13]

Το διαισθητικό API του Chart.js κάνει τη διαδικασία δημιουργίας γραφήματος απλή. Οι προγραμματιστές ξεκινούν ορίζοντας ένα στοιχείο καμβά στο έγγραφο HTML τους, το οποίο χρησιμεύει ως κοντέινερ για το γράφημα. Το επόμενο βήμα περιλαμβάνει τη σύνταξη μιας συνάρτησης JavaScript για τη διαμόρφωση του τύπου, των δεδομένων και των επιλογών του γραφήματος. Στη συνέχεια, το Chart.js παίρνει αυτήν τη διαμόρφωση και αποδίδει το γράφημα εντός του στοιχείου καμβά. Αυτή η διαδικασία επιτρέπει τη δημιουργία δυναμικών γραφημάτων που μπορούν να ενημερώνονται αμέσως, παρέχοντας μια διαδραστική εμπειρία στους τελικούς χρήστες καθώς αλληλεπιδρούν με τα δεδομένα που παρουσιάζονται.

Ένα από τα βασικά χαρακτηριστικά του Chart.js είναι η ανταπόκρισή του. Τα γραφήματα αλλάζουν αυτόματα το μέγεθος ώστε να ταιριάζουν στην οθόνη της συσκευής, διασφαλίζοντας ότι οι απεικονίσεις είναι ευανάγνωστες και ελκυστικές τόσο σε επιτραπέζιες οθόνες, tablet και smartphone. Αυτή η ανταπόκριση είναι ζωτικής σημασίας για την ανάπτυξη σύγχρονων διαδικτυακών εφαρμογών που στοχεύουν στην παροχή συνεπών εμπειριών χρήστη σε μια σειρά συσκευών με διαφορετικά μεγέθη οθόνης.

Ας δούμε ένα παράδειγμα όπου ένας προγραμματιστής επιθυμεί να δημιουργήσει ένα απλό γράφημα ράβδων για την εμφάνιση των μηνιαίων δεδομένων πωλήσεων. Η δομή HTML θα περιλαμβάνει ένα στοιχείο καμβά όπως αυτό:

```
<canvas id="myChart" width="400" height="200"></canvas>
```

Στη συνέχεια, η JavaScript για τη δημιουργία του γραφήματος ράβδων θα είναι:

```
var ctx = document.getElementById('myChart').getContext('2d');
var myChart = new Chart(ctx, {
    type: 'bar',
    data: {
        labels: ['January', 'February', 'March', 'April', 'May', 'June'],
        datasets: [{
            label: 'Monthly Sales',
            data: [12, 19, 3, 5, 2, 3],
            backgroundColor: [
                'rgba(255, 99, 132, 0.2)',
                'rgba(54, 162, 235, 0.2)',
                // More colors
            ],
            borderColor: [
                'rgba(255, 99, 132, 1)',
                'rgba(54, 162, 235, 1)',
                // More colors
            ],
            borderWidth: 1
        }]
    },
    options: {
        scales: {
            y: {
                beginAtZero: true
            }
        }
```

```
    }
});
```

Σε αυτό το απόσπασμα κώδικα, το σύνολο δεδομένων περιέχει τα δεδομένα πωλήσεων και το χρώμα κάθε γραμμής ορίζεται στους πίνακες backgroundColor και borderColor. Το γράφημα έχει διαμορφωθεί ώστε να ξεκινά τον άξονα y του στο μηδέν για καλύτερη αναπαράσταση δεδομένων.

Ένα άλλο παράδειγμα θα μπορούσε να περιλαμβάνει ένα γράφημα πίτας για να απεικονίσει τη διανομή δεδομένων, όπως ποσοστά μεριδίου αγοράς. Δείτε πώς μπορεί κανείς να ρυθμίσει το HTML:

```
<canvas id="marketShareChart" width="400" height="400"></canvas>
var ctx = document.getElementById('marketShareChart').getContext('2d');
var marketShareChart = new Chart(ctx, {
    type: 'pie',
    data: {
        labels: ['Product A', 'Product B', 'Product C'],
        datasets: [{
            data: [10, 20, 30],
            backgroundColor: [
                '#FF6384',
                '#36A2EB',
                '#FFCE56'
            ],
            hoverOffset: 4
        }]
    },
    options: {
        responsive: true,
        plugins: {
            legend: {
                position: 'top',
            },
```

```
        title: {
            display: true,
            text: 'Market Share by Product'
        }
    }
  }
});
```

Σε αυτό το παράδειγμα, ο πίνακας δεδομένων διατηρεί τις τιμές που καθορίζουν το μέγεθος κάθε τμήματος του γραφήματος πίτας, ενώ το backgroundColor ορίζει το χρώμα για κάθε τμήμα. Η ιδιότητα hoverOffset δίνει μια μικρή αύξηση σε ένα τμήμα όταν ένας χρήστης τοποθετείται πάνω του, προσθέτοντας μια ωραία διαδραστική δυνατότητα στο γράφημα.

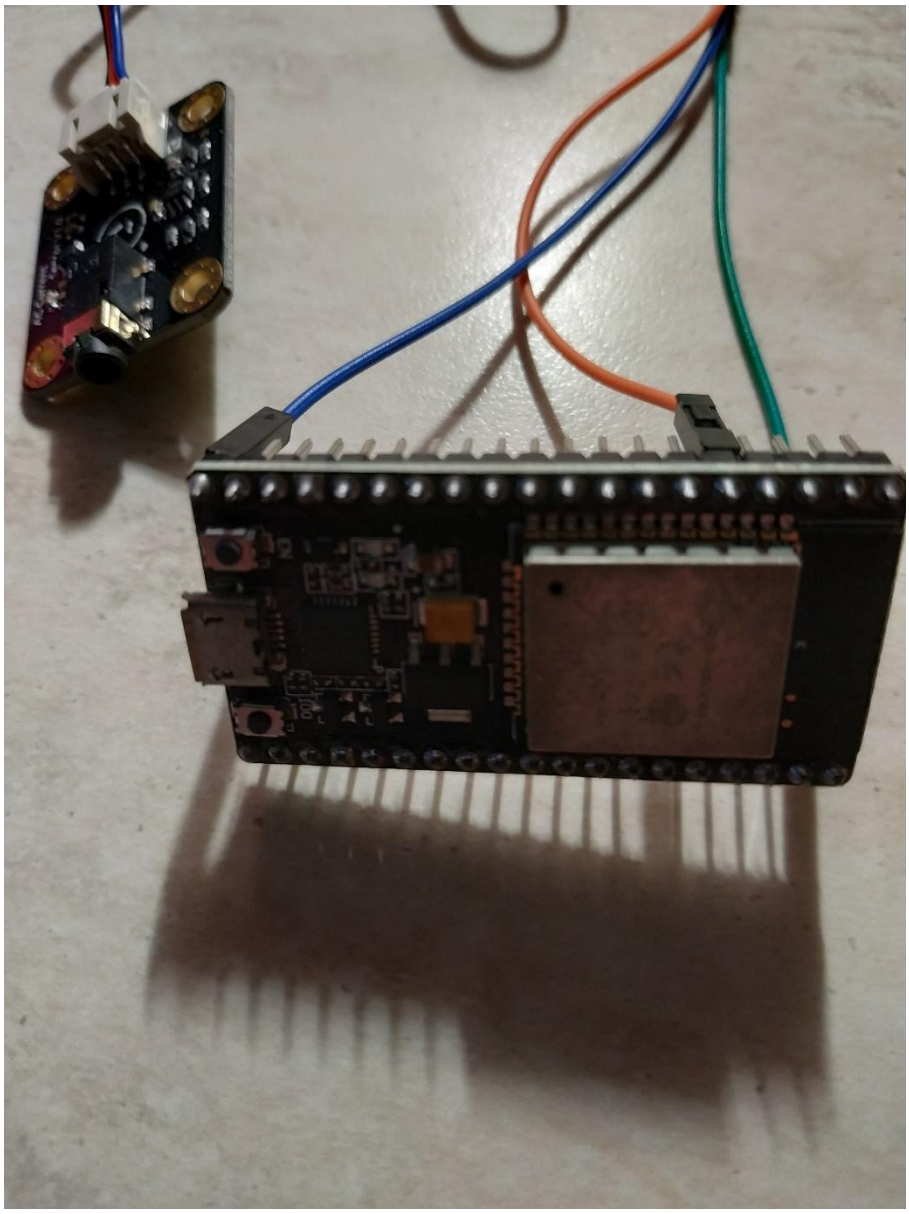
Πέρα από αυτά τα παραδείγματα, το Chart.js παρέχει εκτενείς επιλογές προσαρμογής, επιτρέποντας τη δημιουργία πολύπλοκων και όμορφων απεικονίσεων δεδομένων. Οι προγραμματιστές μπορούν να ελέγχουν πτυχές όπως κινούμενα σχέδια, κλίμακες, συμβουλές εργαλείων και θρύλοι για να βελτιώσουν την εμπειρία του χρήστη. Επιπλέον, η κοινότητα ανοιχτού κώδικα γύρω από το Chart.js συμβάλλει στην ευρωστία του, με πλήθος πόρων, προσθηκών και επεκτάσεων που διατίθενται για την επέκταση της βασικής του λειτουργικότητας.

Το Chart.js αποτελεί ένα ισχυρό εργαλείο στο οπλοστάσιο των προγραμματιστών ιστού, επιτρέποντας τη δημιουργία ελκυστικών και ενημερωτικών απεικονίσεων δεδομένων.

Κεφάλαιο 4ο: Το σύστημα παρακολούθησης κατανάλωσης ρεύματος μέσω διαδικτύου

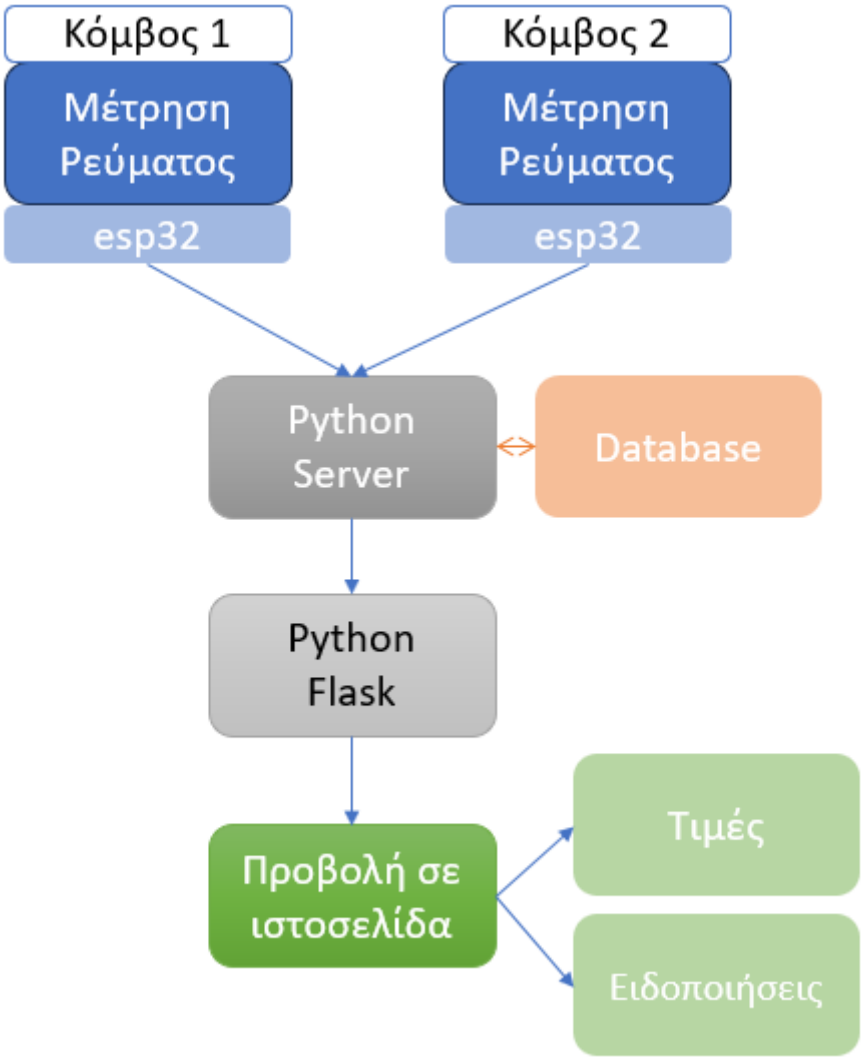
4.1 Η διαδικασία

Το σχέδιο του συστήματος μέτρησης παρουσιάζεται στην Εικόνα 4.1.



Εικόνα 4.1: Σχέδιο του συστήματος μέτρησης με τον esp32 και τον αισθητήρα

Το ολοκληρωμένο σχέδιο του συστήματος παρακολούθησης παρουσιάζεται στην Εικόνα 4.2.

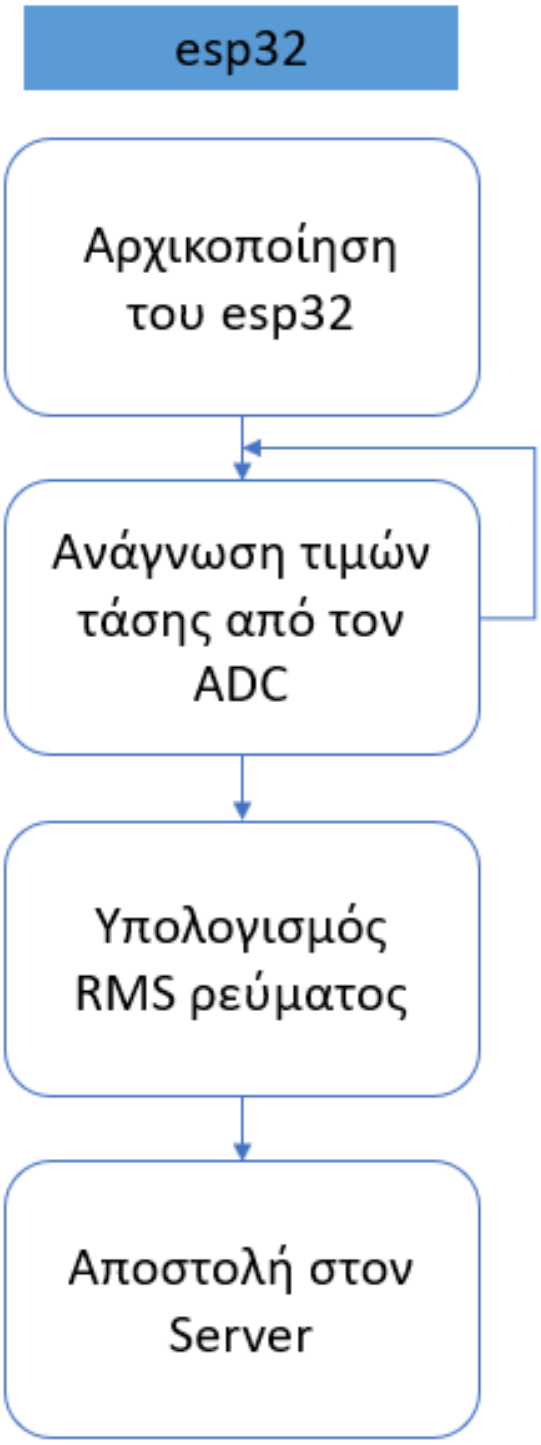


Εικόνα 4.2: Το ολοκληρωμένο σχέδιο του συστήματος παρακολούθησης

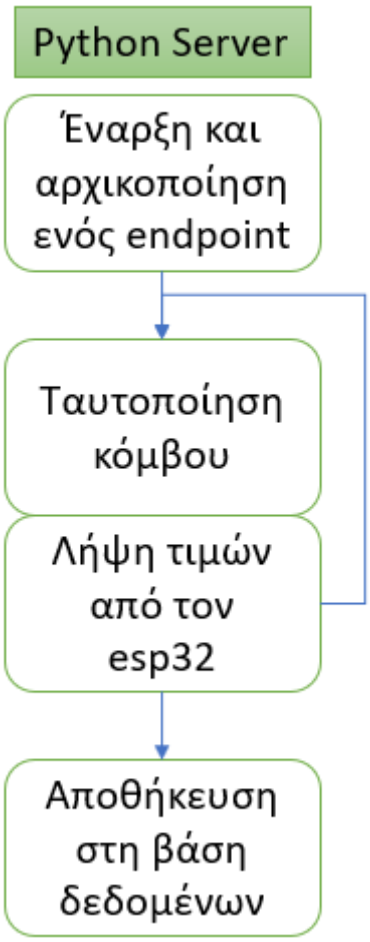
Αυτό το έργο περιλαμβάνει ένα ολοκληρωμένο σύστημα που ενσωματώνει αισθητήρες, μικροελεγκτές, διαδικτυακή επικοινωνία και τεχνολογίες Ιστού για τη δημιουργία μιας ισχυρής και διαδραστικής πλατφόρμας για την παρακολούθηση και την ανάλυση μετρήσεων ηλεκτρικού ρεύματος. Στον πυρήνα αυτού του συστήματος βρίσκεται ο αναλογικός αισθητήρας ρεύματος DFRobot (20A) που συνδέεται με έναν μικροελεγκτή ESP32. Το ESP32, γνωστό για τις ισχυρές του δυνατότητες επεξεργασίας και τις ασύρματες λειτουργίες του, διαβάζει αναλογικά σήματα που αντιπροσωπεύουν τη ροή ρεύματος μέσω του αισθητήρα. Μια κρίσιμη πτυχή αυτής της απόκτησης δεδομένων είναι η μετατροπή των ακατέργαστων μετρήσεων του αισθητήρα σε σημαντικές τιμές ρεύματος μέσου τετραγώνου ρίζας (RMS), ένα πρότυπο για την αξιολόγηση του ενεργού ρεύματος σε κυκλώματα AC. Οι υπολογισμένες τιμές RMS ενθυλακώνουν την ουσία των σημάτων κυμαινόμενου ρεύματος σε μια σταθερή μορφή, κατάλληλη για σκοπούς παρακολούθησης και ανάλυσης.

Μόλις υπολογιστούν οι τιμές RMS, το ESP32 χρησιμοποιεί τις δυνατότητές του Wi-Fi για τη μετάδοση των δεδομένων σε διακομιστή που βασίζεται σε Flask χρησιμοποιώντας τη μέθοδο GET, διασφαλίζοντας απρόσκοπτη ροή πληροφοριών από το υλικό στη διεπαφή ιστού. Αυτή η εφαρμογή από την πλευρά του διακομιστή, κατασκευασμένη σε Python με Flask, αποτελεί μια πολύπλευρη πλατφόρμα που προσφέρει λήψη, επεξεργασία και οπτικοποίηση δεδομένων. Στον πυρήνα του, ο διακομιστής Flask όχι μόνο λαμβάνει τις τρέχουσες μετρήσεις, αλλά διαχειρίζεται επίσης τις αλληλεπιδράσεις των χρηστών μέσω μιας ασφαλούς διεπαφής σύνδεσης, διασφαλίζοντας ότι η πρόσβαση στα δεδομένα και τα αναλυτικά στοιχεία είναι εξατομικευμένη και προστατευμένη. Ο διακομιστής ενσωματώνεται με μια βάση δεδομένων MySQL, σχολαστικά δομημένη με πίνακες για χρήστες, κόμβους και τρέχουσες τιμές, καθένας από τους οποίους εξυπηρετεί τον ξεχωριστό του σκοπό στην οργάνωση δεδομένων και τη διαχείριση σχέσεων. Αυτή η δομημένη προσέγγιση όχι μόνο διευκολύνει την αποτελεσματική ανάκτηση και αποθήκευση δεδομένων, αλλά επίσης θέτει τα θεμέλια για προηγμένη ανάλυση δεδομένων και παρουσίαση δεδομένων για συγκεκριμένους χρήστες.

Στο μέτωπο του χρήστη, το σύστημα παρουσιάζει μια διαισθητική διεπαφή ιστού, όπου οι εγγεγραμμένοι χρήστες μπορούν να οπτικοποιήσουν τις τρέχουσες μετρήσεις σε πραγματικό χρόνο. Αξιοποιώντας το Chart.js, ο ιστότοπος αποδίδει τις τρέχουσες τιμές RMS σε δυναμικά, διαδραστικά γραφήματα, προσφέροντας στους χρήστες μια λεπτομερή εικόνα των τρεχουσών τάσεων και διακυμάνσεων. Αυτή η οπτικοποίηση όχι μόνο βοηθά στην άμεση ερμηνεία των δεδομένων, αλλά εμπλουτίζει επίσης την αλληλεπίδραση του χρήστη με το σύστημα, καθιστώντας το ένα ανεκτίμητο εργαλείο για την παρακολούθηση, την ανάλυση και τις διαδικασίες λήψης αποφάσεων. Επιπλέον, η ενσωμάτωση προβολών δεδομένων για συγκεκριμένους χρήστες, με γνώμονα τη σχεσιακή δομή της βάσης δεδομένων MySQL, διασφαλίζει ότι το σύστημα είναι επεκτάσιμο και προσαρμόσιμο σε ένα ευρύ φάσμα εφαρμογών, που κυμαίνονται από μεμονωμένους λάτρεις του έργου έως συστήματα παρακολούθησης βιομηχανικής κλίμακας. σημαντικό άλμα στον τομέα του IoT και της ανάλυσης δεδομένων.

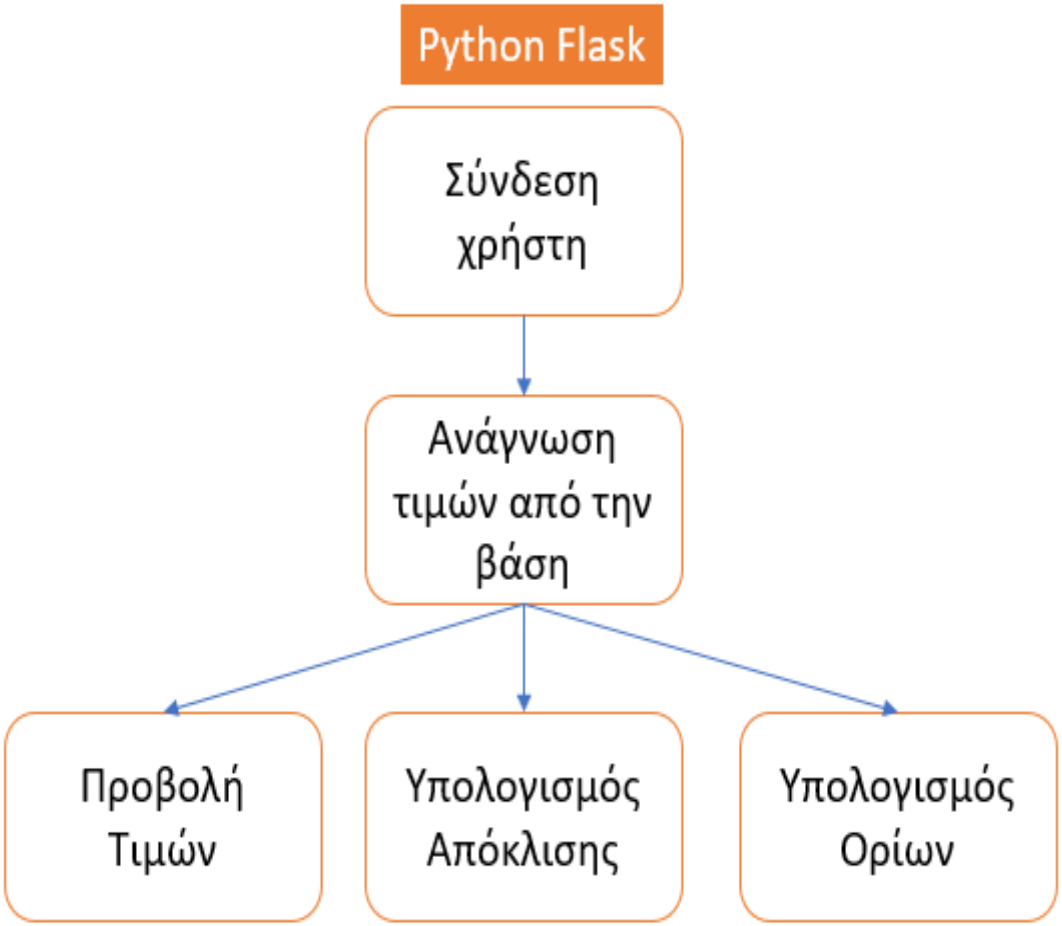


Εικόνα 4.3: Διάγραμμα για τη λειτουργία στον esp32



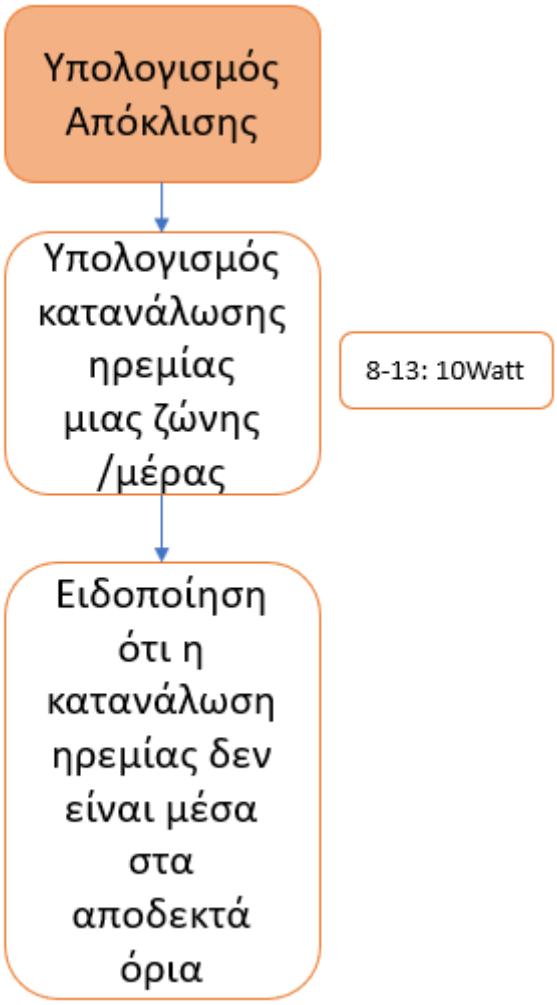
Εικόνα 4.4: Διάγραμμα για τη λειτουργία στον python server

Τα διαγράμματα 4.4-4.7 περιγράφουν την αρχιτεκτονική και τη ροή εργασίας ενός συστήματος που βασίζεται στο web που παρακολουθεί και αναλύει δεδομένα κατανάλωσης ενέργειας. Το πρώτο διάγραμμα περιγράφει τη ροή των δεδομένων από μια προοπτική διακομιστή Python, όπου η δημιουργία και η αρχική διαμόρφωση ενός τελικού σημείου είναι το πρώτο βήμα. Ακολουθεί αναγνώριση κόμβου και λήψη τρέχουσες τιμές από μια συσκευή ESP32. Αυτές οι τρέχουσες τιμές στη συνέχεια προωθούνται σε μια βάση δεδομένων με αναγνώσεις. Από την πλευρά του Flask, η εφαρμογή είναι υπεύθυνη για τη σύνδεση των χρηστών, την ανάκτηση δεδομένων από τη βάση δεδομένων και την περαιτέρω επεξεργασία για σκοπούς προβολής. Επισημαίνει την αναγκαιότητα μιας φιλικής προς τον χρήστη διεπαφής για τη σύνδεση και τη σημασία ενός ισχυρού backend ικανού να χειρίζεται εισερχόμενες ροές δεδομένων και να τις εξυπηρετεί αποτελεσματικά στους τελικούς χρήστες.



Εικόνα 4.5: Διάγραμμα για τη λειτουργία με χρήση python flask

Τα επόμενο διάγραμμα εμβαθύνει στις ιδιαιτερότητες της χρήσης δεδομένων, εστιάζοντας ιδιαίτερα στην ερμηνεία και εμφάνιση των δεδομένων κατανάλωσης. Περιγράφει τα υποσυστήματα για τις ώρες καταγραφής και τον υπολογισμό της συνολικής ενέργειας που καταναλώνεται, παρέχοντας πληροφορίες για τις ώρες αιχμής χρήσης. Μια σημαντική λειτουργία που επισημαίνεται είναι η δυνατότητα ειδοποίησης των χρηστών όταν η κατανάλωση είναι ασυνήθιστα υψηλή ή χαμηλή, κάτι που θα μπορούσε να είναι ενδεικτικό πιθανών προβλημάτων ή ευκαιριών εξοικονόμησης. Αυτή η προληπτική προσέγγιση στη διαχείριση ενέργειας υπογραμμίζει την ικανότητα του συστήματος όχι μόνο να ενημερώνει αλλά και να λειτουργεί ως συμβουλευτικό εργαλείο, ωθώντας τους χρήστες να βελτιστοποιήσουν τα πρότυπα κατανάλωσης ενέργειας.



Εικόνα 4.6: Διάγραμμα για τη λειτουργία υπολογισμού απόκλιση

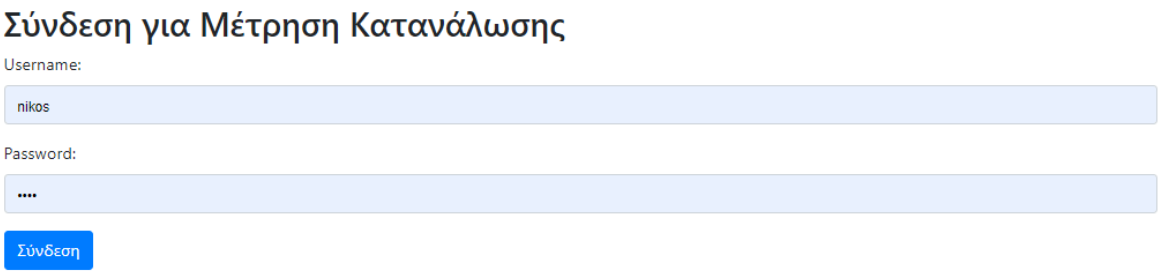
Ενσωματώνοντας αυτά τα διαγράμματα του συστήματος, μπορούμε να δούμε μια πιο ολιστική λύση που όχι μόνο αυτοματοποιεί την παρακολούθηση της κατανάλωσης ενέργειας αλλά επίσης συνεργάζεται ενεργά με τον χρήστη για ειδοποιήσεις όσον αφορά την ασφάλεια αλλά και την προώθηση της ενεργειακής απόδοσης. Ο διακομιστής Python και το υλικό ESP32 συνεργάζονται για να εξασφαλίσουν ακριβή και σε πραγματικό χρόνο συλλογή δεδομένων, τα οποία στη συνέχεια αναμεταδίδονται σε έναν διακομιστή Flask, όπου αποθηκεύονται, επεξεργάζονται και ετοιμάζονται για αλληλεπίδραση με τον χρήστη. Στην πλατφόρμα Flask, ο έλεγχος ταυτότητας χρήστη διασφαλίζει την πρόσβαση ενώ η εφαρμογή εξυπηρετεί τα επεξεργασμένα δεδομένα μέσω μιας διεπαφής. Η ευφυΐα του συστήματος αποδεικνύεται ως παράδειγμα στην ικανότητά του να αναλύει τάσεις, να εντοπίζει ανωμαλίες και να παρέχει χρήσιμες πληροφορίες, διασφαλίζοντας ταυτόχρονα ότι τα δεδομένα αναπαρίστανται με τρόπο που είναι τόσο ενημερωτικό όσο και προσβάσιμο στον χρήστη. Αυτή η

διπλής όψης προσέγγιση ενσωματώνει την ουσία ενός έξυπνου συστήματος παρακολούθησης που είναι αφενός στα δεδομένα και αφετέρου προσανατολισμένο στον χρήστη.

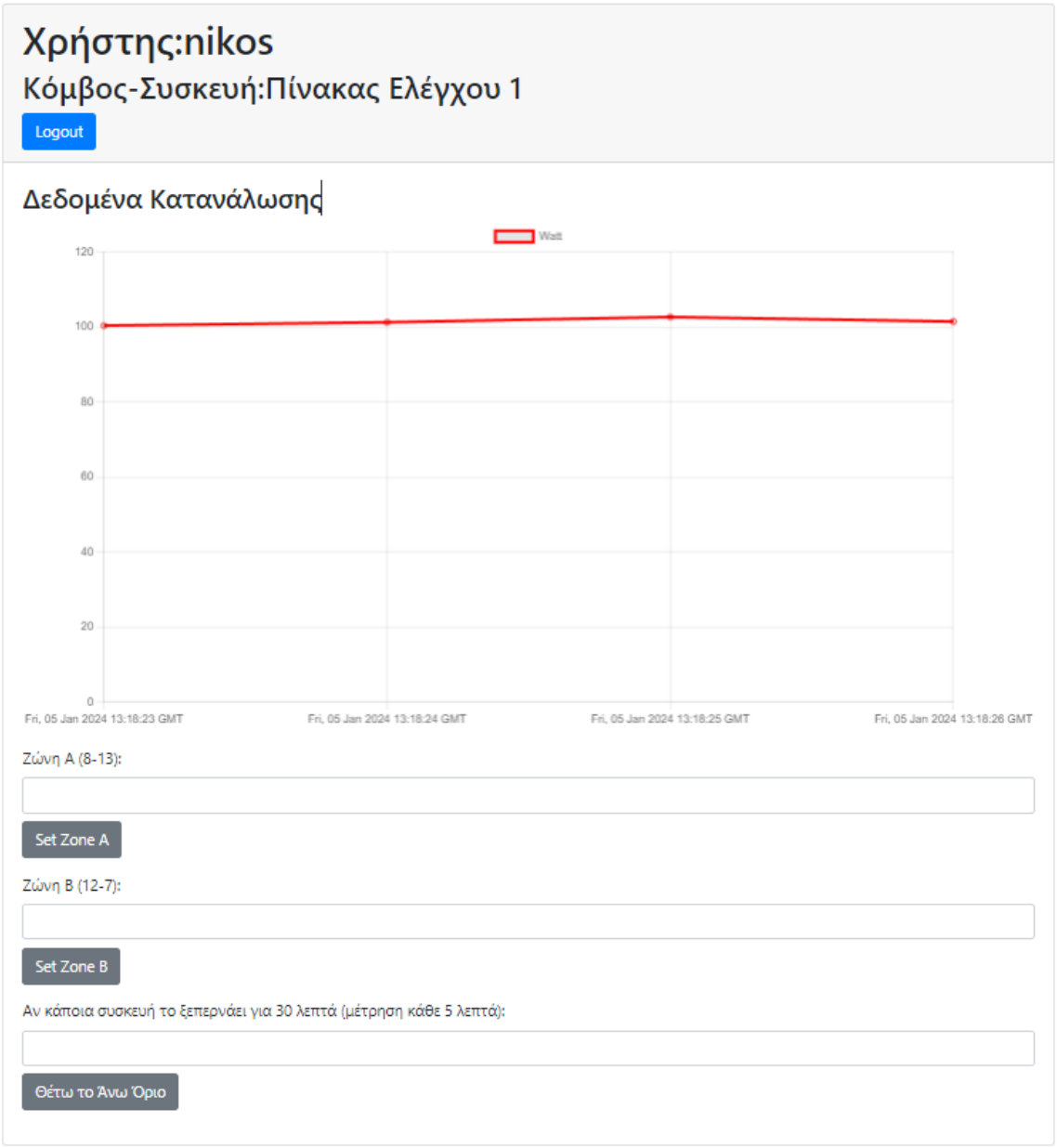
Εικόνα 4.7: Διάγραμμα για τη λειτουργία υπολογισμού ορίων

4.2 Η εφαρμογή - ιστοσελίδα

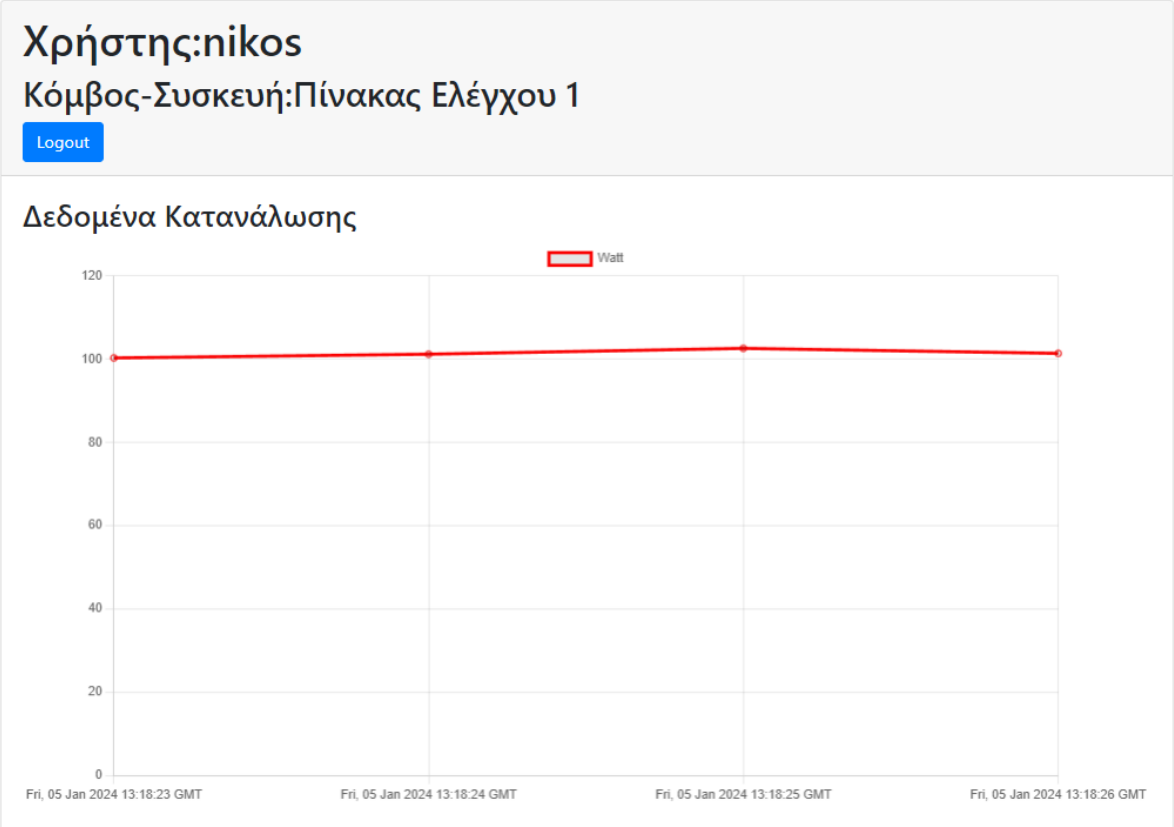
Ο χρήστης μπορεί να συνδεθεί στο σύστημα αφού πατήσει το κουμπί σύνδεση όπως φαίνεται στο παρακάτω Εικόνα 4.8.



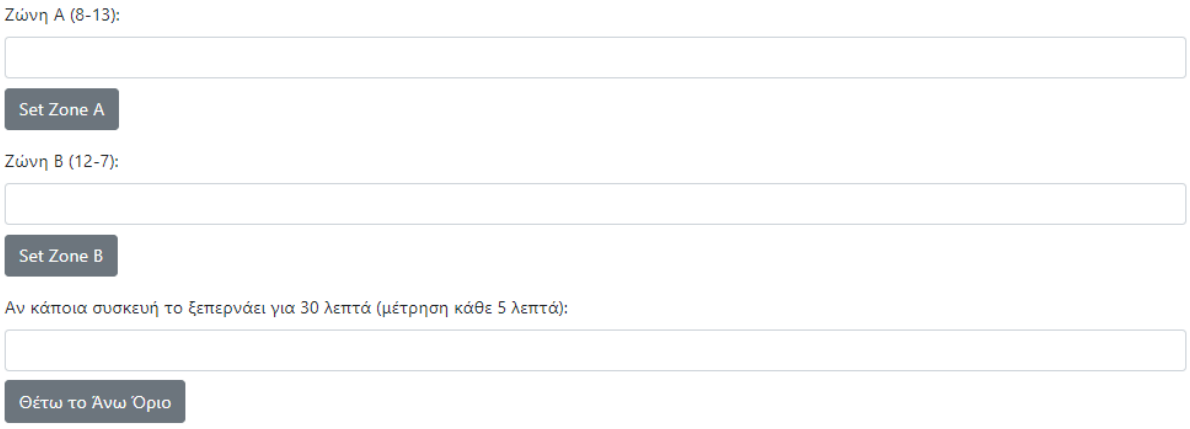
Εικόνα 4.8: Σύνδεση με τα διαπιστευτήρια



Εικόνα 4.9: Κεντρική Σελίδα που βλέπει ο εξουσιοδοτημένος χρήστης



Εικόνα 4.10: Διάγραμμα με την κατανάλωση



Εικόνα 4.11: Τα πεδία για να θέσει όρια και ζώνες

Η αναπτυγμένη διαδικτυακή εφαρμογή, που υποστηρίζεται από το Flask, λειτουργεί ως ένας διαδραστικός πίνακας εργαλείων που παρέχει στους χρήστες πληροφορίες σε πραγματικό χρόνο σχετικά με τα μοτίβα κατανάλωσης ενέργειας. Στην καρδιά αυτής της διεπαφής χρήστη βρίσκεται το

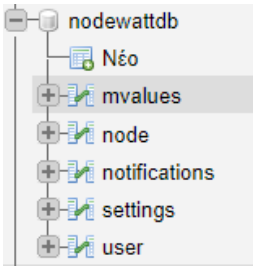
Chart.js, μια ισχυρή βιβλιοθήκη JavaScript που χρησιμοποιείται για τη δημιουργία αποκριτικών, κινούμενων και προσαρμόσιμων γραφημάτων. Το μπροστινό μέρος της εφαρμογής είναι σχολαστικά σχεδιασμένο για να καλύπτει τόσο την αισθητική όσο και τη λειτουργικότητα που απαιτούνται για μια διαισθητική εμπειρία χρήστη. Κατά τη σύνδεση —μια διαδικασία που εξασφαλίζεται από τους ισχυρούς μηχανισμούς ελέγχου ταυτότητας του Flask— οι χρήστες καλωσορίζονται με έναν πίνακα εργαλείων που εμφανίζει μια σειρά από γραφήματα και γραφήματα που προέρχονται από τις τρέχουσες τιμές που είναι αποθηκευμένες στη βάση δεδομένων MySQL. Αυτές οι απεικονίσεις δεν είναι στατικές εικόνες, είναι δυναμικές αναπαραστάσεις που ενημερώνονται σε πραγματικό χρόνο καθώς λαμβάνονται νέα δεδομένα από τη μονάδα αισθητήρα ESP32. Ο σχεδιασμός δίνει έμφαση στη σαφήνεια και την ευκολία κατανόησης, επιτρέποντας στους χρήστες να κατανοήσουν αμέσως την τρέχουσα και ιστορική χρήση ενέργειας.

Η διεπαφή έχει δημιουργηθεί με γνώμονα τον τελικό χρήστη, με διαδραστικά στοιχεία που επιτρέπουν στους χρήστες να διερευνούν τα δεδομένα τους για πιο λεπτομερή ανάλυση. Για παράδειγμα, οι χρήστες μπορούν να τοποθετήσουν το δείκτη του ποντικιού πάνω από τα στοιχεία του γραφήματος για να δουν ακριβή στοιχεία για συγκεκριμένα χρονικά σημεία ή μπορούν να κάνουν κλικ για μεγέθυνση σε ένα συγκεκριμένο παράθυρο δεδομένων. Αυτά τα γραφήματα εξυπηρετούν πολλαπλές λειτουργίες. Δεν εμφανίζουν μόνο τη συνολική κατανάλωση ενέργειας σε μια δεδομένη περίοδο, αλλά επισημαίνουν επίσης τους χρόνους αιχμής χρήσης, κάτι που είναι κρίσιμο για τη διαχείριση του ενεργειακού κόστους και της απόδοσης. Επιπλέον, η διεπαφή περιλαμβάνει προληπτικές ειδοποιήσεις που ειδοποιούν τους χρήστες όταν τα πρότυπα κατανάλωσης αποκλίνουν από τον κανόνα, υποδηλώνοντας είτε ένα πιθανό πρόβλημα είτε μια ευκαιρία προσαρμογής της χρήσης για εξοικονόμηση κόστους. Αυτή η προληπτική λειτουργία αποτελεί απόδειξη του ρόλου του μπροστινού μέρους όχι μόνο στην παρουσίαση δεδομένων, αλλά και στην καθοδήγηση της συμπεριφοράς των χρηστών προς την πιο αποδοτική χρήση ενέργειας. Συνολικά, η διαδικτυακή εφαρμογή αποτελεί απόδειξη του πώς μπορούν να αξιοποιηθούν οι προηγμένες τεχνολογίες Ιστού για τη δημιουργία μιας φιλικής προς τον χρήστη και ενημερωτικής πλατφόρμας που υπερβαίνει την απλή παρουσίαση δεδομένων για να διευκολύνει μια πιο αφοσιωμένη και συνειδητή προσέγγιση στην κατανάλωση ενέργειας.

4.3 Η Βάση

Η θεμελιώδης ραχοκοκαλιά της διαδικτυακής εφαρμογής Flask είναι η βάση δεδομένων MySQL MariaDB, όπου η ακεραιότητα των δεδομένων και η σχεσιακή δομή είναι υψίστης σημασίας. Ο σχεδιασμός της βάσης δεδομένων αρθρώνεται μέσω μιας σειράς καλά καθορισμένων πινάκων, ο καθένας από τους οποίους εξυπηρετεί έναν μοναδικό ρόλο μέσα στο σύστημα. Ο πίνακας «mvalues» είναι ο κεντρικός πίνακας στη λειτουργικότητα της εφαρμογής, αποθηκεύοντας μετρήσεις σε πραγματικό χρόνο των τιμών ηλεκτρικού ρεύματος από διάφορους κόμβους. Κάθε εγγραφή σε αυτόν τον πίνακα περιέχει ένα μοναδικό «id», ένα «nodeid» που συνδέεται με τον αισθητήρα, την τιμή μέτρησης «mvalue» και μια χρονική σήμανση «tstamp» που υποδηλώνει πότε έγινε η μέτρηση. Αυτός ο σχεδιασμός επιτρέπει στην εφαρμογή Flask να ανακτά και να χειρίζεται τα δεδομένα κατανάλωσης ενέργειας αποτελεσματικά, παρουσιάζοντάς τα στον χρήστη με προσιτό και ουσιαστικό τρόπο.

Εκτός από τον πίνακα «mvalues», η βάση δεδομένων περιλαμβάνει αρκετούς άλλους πίνακες που είναι ζωτικής σημασίας για ένα ολοκληρωμένο σύστημα παρακολούθησης της ενέργειας. Ο πίνακας "node", για παράδειγμα, αποθηκεύει πληροφορίες για κάθε κόμβο αισθητήρα, συμπεριλαμβανομένου ενός μοναδικού "id", του "userid" του χρήστη που κατέχει τον κόμβο και ενός περιγραφικού "nodename". Αυτός ο πίνακας διευκολύνει την παρακολούθηση δεδομένων για συγκεκριμένο χρήστη και επιτρέπει ένα επεκτάσιμο σύστημα όπου μπορούν να φιλοξενηθούν πολλοί χρήστες και κόμβοι. Ο πίνακας "notifications" διατηρεί ένα αρχείο καταγραφής ειδοποιήσεων και μηνυμάτων που σχετίζονται με τα μοτίβα κατανάλωσης των χρηστών, ενώ ο πίνακας "settings" αποθηκεύει όρια που ορίζονται από τον χρήστη για ειδοποιήσεις. Και οι δύο αυτοί πίνακες, μαζί με τον πίνακα "user", ο οποίος περιέχει διαπιστευτήρια χρήστη και προσωπικές πληροφορίες, περιλαμβάνουν πεδία "αναγνωριστικό" ως κύρια κλειδιά, χρονικές σημάνσεις "created_at" και "updated_at" για την παρακολούθηση της δημιουργίας αρχείων και των τροποποιήσεων, διασφαλίζοντας ότι το σύστημα διατηρεί μια ολοκληρωμένη διαδρομή ελέγχου αλληλεπιδράσεων και αλλαγών. Αυτά τα στοιχεία σχηματίζουν συλλογικά ένα ισχυρό και ασφαλές σύστημα διαχείρισης δεδομένων, επιτρέποντας στην εφαρμογή Flask να χρησιμεύσει ως αξιόπιστη και φιλική προς το χρήστη διεπαφή για παρακολούθηση και ανάλυση ενέργειας.



Εικόνα 4.12: Οι πίνακες της βάσης δεδομένων nodewattdb

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Σχόλια	Πρόσθετα
1	id	int(11)			Όχι	*Καμία*		AUTO_INCREMENT
2	username	varchar(30)	utf8mb4_general_ci		Όχι	*Καμία*		
3	password	varchar(30)	utf8mb4_general_ci		Όχι	*Καμία*		
4	lastname	varchar(30)	utf8mb4_general_ci		Όχι	*Καμία*		
5	firstname	varchar(30)	utf8mb4_general_ci		Όχι	*Καμία*		
6	created_at	datetime			Όχι	current_timestamp()		

Εικόνα 4.13: Η δομή του πίνακα User

id	username	password	lastname	firstname	created_at
1	nikos	6543	Τακάκης	Νίκος	2024-01-01 12:23:40
2	vlahos	6543	Βλάχος	Σπύρος	2024-01-01 12:23:44

Εικόνα 4.14: Το περιεχόμενο του πίνακα User

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Σχόλια	Πρόσθετα
1	id	int(11)			Όχι	*Καμία*		AUTO_INCREMENT
2	nodeid	int(11)			Όχι	*Καμία*		
3	mvalue	double			Ναι	*NULL*		
4	tstamp	datetime			Όχι	current_timestamp()		

Εικόνα 4.15: Η δομή του πίνακα mvalues

id	nodeid	mvalue	tstamp
1	1	100.3	2024-01-05 13:18:23
2	1	101.2	2024-01-05 13:18:24
3	1	102.6	2024-01-05 13:18:25
4	1	101.4	2024-01-05 13:18:26

Εικόνα 4.16: Το περιεχόμενο του πίνακα mvalues

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Σχόλια	Πρόσθετα
1	id	int(11)			Όχι	*Καμία*		AUTO_INCREMENT
2	userid	int(11)			Όχι	*Καμία*		
3	nodename	varchar(30)	utf8mb4_general_ci		Όχι	*Καμία*		

Εικόνα 4.17: Η δομή του πίνακα node

id	userid	nodename
1	1	Πίνακας Ελέγχου 1
2	2	Συσκευή 1

Εικόνα 4.18: Το περιεχόμενο του πίνακα node

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Σχόλια	Πρόσθετα
1	id	int(11)			Όχι	*Καμία*		AUTO_INCREMENT
2	oriotext	varchar(100)	utf8mb4_general_ci		Όχι	*Καμία*		
3	oriovalue	double			Όχι	*Καμία*		
4	created_at	datetime			Όχι	current_timestamp()		

Εικόνα 4.19: Η δομή του πίνακα settings

id	oriotext	oriovalue	created_at
1	Zone A (8-13)	10	2024-01-05 13:18:00
2	Zone B (12-7)	30	2024-01-05 13:25:23

Εικόνα 4.20: Το περιεχόμενο του πίνακα settings

#	Όνομα	Τύπος	Σύνθεση	Χαρακτηριστικά	Κενό	Προεπιλογή	Σχόλια	Πρόσθετα
1	id	int(11)			Όχι	*Καμία*		AUTO_INCREMENT
2	userid	int(11)			Όχι	*Καμία*		
3	nodeid	int(11)			Όχι	*Καμία*		
4	msg	varchar(200)	utf8mb4_general_ci		Ναι	*NULL*		
5	created_at	datetime			Όχι	current_timestamp()		

Εικόνα 4.21: Η δομή του πίνακα notifications

id	userid	nodeid	msg	created_at
1	1	1	Άνω Όριο	2024-01-04 09:21:25.000000
2	1	1	Άνω Όριο	2024-01-04 09:31:25.000000

Εικόνα 4.22: Το περιεχόμενο του πίνακα notifications

4.4 Ασφάλεια στο σύστημα και στα δεδομένα

Ενσωματώνοντας τα θεμελιώδη στοιχεία της εφαρμογής Ιστού που περιγράφηκε προηγουμένως, η ασφάλεια κωδικού πρόσβασης ιστότοπου είναι ένα κρίσιμο στοιχείο που διασφαλίζει ότι τα δεδομένα χρήστη παραμένουν προστατευμένα και εμπιστευτικά. Όταν ένας χρήστης επιχειρεί να αποκτήσει πρόσβαση στο σύστημα, πρέπει να παρέχει τα διαπιστευτήριά του—συνήθως μια διεύθυνση email και έναν κωδικό πρόσβασης. Μετά τον επιτυχή έλεγχο ταυτότητας, το σύστημα ξεκινά μια περίοδο λειτουργίας, μια προσωρινή κατάσταση αλληλεπίδρασης μεταξύ του χρήστη και του συστήματος που αποθηκεύει με ασφάλεια τις πληροφορίες του χρήστη. Αυτή η συνεδρία είναι υψίστης σημασίας για τη διατήρηση ενός ασφαλούς περιβάλλοντος, καθώς προσδιορίζει μοναδικά τον χρήστη σε όλη την αλληλεπίδρασή του με το σύστημα. Είναι μια κατασκευή αποθηκευμένη στον διακομιστή που ενσωματώνει την κατάσταση και το περιβάλλον του χρήστη, διασφαλίζοντας ότι οποιεσδήποτε ενέργειες πραγματοποιούνται στο σύστημα εκτελούνται από τον πιστοποιημένο χρήστη.

Ο μηχανισμός ασφαλείας που βασίζεται σε συνεδρία είναι θεμελιώδης για τη λειτουργία της εφαρμογής. Κάθε ενέργεια χρήστη, είτε είναι το πάτημα ενός κουμπιού για την εκκίνηση μιας εντολής είτε οποιαδήποτε άλλη αλληλεπίδραση με το σύστημα, επικυρώνεται μέσω αυτής της περιόδου λειτουργίας. Ο διακομιστής ελέγχει ότι το αίτημα προέρχεται από μια περίοδο λειτουργίας που σχετίζεται με έναν πιστοποιημένο χρήστη. Αυτό εγγυάται ότι μόνο εξουσιοδοτημένες ενέργειες επεξεργάζονται και εκτελούνται από το σύστημα, αποτρέποντας τη μη εξουσιοδοτημένη πρόσβαση και πιθανές παραβιάσεις της ασφάλειας. Για να διατηρηθεί ένα υψηλό επίπεδο ασφάλειας, αυτές οι περίοδοι λειτουργίας έχουν προκαθορισμένο χρόνο λήξης, μετά τον οποίο ο χρήστης πρέπει να πραγματοποιήσει εκ νέου έλεγχο ταυτότητας για να συνεχίσει να αλληλεπιδρά με το σύστημα.

Επιπλέον, η ακεραιότητα και το απόρρητο της μετάδοσης δεδομένων μεταξύ του χρήστη και του συστήματος προστατεύονται με τη χρήση της τεχνολογίας Secure Socket Layer (SSL). Το SSL είναι ένα τυπικό πρωτόκολλο ασφαλείας που κρυπτογραφεί πακέτα δεδομένων κατά τη μετάδοση στο Διαδίκτυο. Καθιερώνοντας μια κρυπτογραφημένη σύνδεση μεταξύ της συσκευής του χρήστη και του συστήματος, το SSL διασφαλίζει ότι ευαίσθητες πληροφορίες, όπως κωδικοί πρόσβασης και προσωπικά δεδομένα, δεν μπορούν να υποκλαπούν ή να αποκρυπτογραφηθούν από υποκλοπές. Αυτό το ασφαλές κανάλι επικοινωνίας είναι μια θεμελιώδης πτυχή της αρχιτεκτονικής ασφάλειας της διαδικτυακής εφαρμογής, παρέχοντας στους χρήστες τη σιγουριά ότι οι αλληλεπιδράσεις τους με το σύστημα είναι ιδιωτικές και προστατεύονται από πιθανές απειλές στον κυβερνοχώρο. Η χρήση του SSL υπογραμμίζει τη δέσμευση του συστήματος να τηρεί αυστηρά πρότυπα ασφαλείας σε κάθε πτυχή της λειτουργίας του, από τον έλεγχο ταυτότητας χρήστη έως τον ασφαλή χειρισμό των δεδομένων ηλεκτρικής κατανάλωσης.

Κεφάλαιο 5ο: Συμπεράσματα και προτάσεις βελτίωσης

Η ανάπτυξη αυτής της διαδικτυακής εφαρμογής αντιπροσωπεύει ένα σημαντικό βήμα στον τομέα της παρακολούθησης και ανάλυσης της κατανάλωσης ενέργειας. Στον πυρήνα του, το σύστημα ενσωματώνει εκ γενετής διάφορα τεχνολογικά στοιχεία, συμπεριλαμβανομένων των ακριβών μετρήσεων από τον αναλογικό αισθητήρα ρεύματος DFRobot που συνδέεται με το ESP32, τις ισχυρές δυνατότητες διαχείρισης και επεξεργασίας δεδομένων μιας βάσης δεδομένων MySQL MariaDB και μια ασφαλή και διαδραστική διεπαφή χρήστη που υποστηρίζεται από το Flask και Chart.js. Η εφαρμογή έχει σχεδιαστεί σχολαστικά για να διασφαλίζει ότι οι χρήστες όχι μόνο έχουν πρόσβαση σε δεδομένα πραγματικού χρόνου και ιστορικά σχετικά με την κατανάλωση ενέργειας, αλλά μπορούν επίσης να ερμηνεύουν και να αλληλεπιδρούν με αυτές τις πληροφορίες μέσω ενός δυναμικού και ανταποκρινόμενου πίνακα εργαλείων. Αυτή η προσέγγιση με επίκεντρο τον χρήστη, σε συνδυασμό με ειδοποιήσεις σε πραγματικό χρόνο και ανάλυση τάσεων, παρέχει ένα επίπεδο γνώσης και ελέγχου που είναι απαραίτητο για την αποτελεσματική διαχείριση ενέργειας.

Η ασφάλεια σε αυτό το οικοσύστημα είναι υψίστης σημασίας, καθώς περιλαμβάνει εγγυήσεις ελέγχου ταυτότητας χρήστη και μετάδοσης δεδομένων. Το σύστημα χρησιμοποιεί έλεγχο ταυτότητας βάσει περιόδου σύνδεσης, διατηρώντας την ακεραιότητα των ενεργειών του χρήστη διασφαλίζοντας ότι προέρχονται από επαληθευμένες και ενεργές περιόδους σύνδεσης. Οι περίοδοι λειτουργίας, οι οποίες παραμένουν στον διακομιστή για περιορισμένη διάρκεια, λειτουργούν ως διακριτικό συνεχούς επαλήθευσης, αντανακλώντας μια προληπτική στάση ασφαλείας. Παράλληλα, η εφαρμογή χρησιμοποιεί κρυπτογράφηση SSL για όλη την επικοινωνία δεδομένων, δημιουργώντας ένα ενισχυμένο κανάλι μεταξύ του χρήστη και του συστήματος. Αυτή η κρυπτογράφηση δεν είναι απλώς ένα εμπόδιο, αλλά μια δυναμική ασπίδα που προσαρμόζεται στο εξελισσόμενο τοπίο της ασφάλειας στο διαδίκτυο, προστατεύοντας ευαίσθητα δεδομένα χρήστη και την ακεραιότητα του συστήματος από πιθανές απειλές στον κυβερνοχώρο.

Η δομή της βάσης δεδομένων αποτελεί τη ραχοκοκαλιά του συστήματος, με κάθε πίνακα σχολαστικά σχεδιασμένο για όχι μόνο αποθήκευση αλλά και συσχέτιση δεδομένων χρήστη, πληροφοριών κόμβων και μετρήσεων με συνεκτικό και αποτελεσματικό τρόπο. Οι πίνακες «mvalues», «node», «notifications», «settings» και «user» συνδέονται μεταξύ τους μέσω καλά καθορισμένων σχέσεων, διασφαλίζοντας την ακεραιότητα των δεδομένων και διευκολύνοντας πολύπλοκα ερωτήματα που εξυπηρετούν τις αναλυτικές ανάγκες της εφαρμογής. Αυτός ο σχεσιακός σχεδιασμός επιτρέπει την επεκτασιμότητα, φιλοξενώντας έναν αυξανόμενο αριθμό χρηστών και κόμβων και διασφαλίζοντας ότι το σύστημα δεν είναι απλώς λειτουργικό αλλά και προσαρμόσιμο στις αυξανόμενες απαιτήσεις.

Κοιτάζοντας το μέλλον, υπάρχουν αρκετοί δρόμοι για βελτίωση που θα μπορούσαν να διερευνηθούν για να βελτιωθεί περαιτέρω το σύστημα. Μια σημαντική βελτίωση θα μπορούσε να είναι η ενσωμάτωση αλγορίθμων μηχανικής μάθησης για την παροχή προγνωστικών αναλύσεων σχετικά με τα πρότυπα κατανάλωσης ενέργειας. Αυτό θα μπορούσε να βοηθήσει τους χρήστες όχι μόνο να παρακολουθούν και να αντιδρούν στην τρέχουσα χρήση ενέργειας αλλά και να προβλέπουν τη μελλοντική τους κατανάλωση και να λαμβάνουν τεκμηριωμένες αποφάσεις για τη βελτιστοποίηση της ενεργειακής τους χρήσης. Ένας άλλος τομέας για βελτίωση είναι η εφαρμογή ενός πιο διευκρινισμένου συστήματος αδειών χρήστη, που επιτρέπει διαφορετικά επίπεδα πρόσβασης και ελέγχου εντός του συστήματος. Αυτό θα μπορούσε να διευκολύνει την εφαρμογή του συστήματος σε εμπορικά και βιομηχανικά περιβάλλοντα όπου πολλοί ενδιαφερόμενοι απαιτούν διαφορετικά επίπεδα πρόσβασης δεδομένων και αλληλεπίδρασης συστήματος.

Επιπλέον, λαμβάνοντας υπόψη τις αυξανόμενες ανησυχίες σχετικά με το απόρρητο και την ασφάλεια των δεδομένων, το σύστημα θα μπορούσε να επωφεληθεί από την ενσωμάτωση τεχνολογίας blockchain για συναλλαγές δεδομένων. Τα εγγενή χαρακτηριστικά της αποκέντρωσης, της διαφάνειας και της αμετάβλητης του blockchain θα μπορούσαν να προσθέσουν ένα επιπλέον επίπεδο ασφάλειας και εμπιστοσύνης στο σύστημα, διασφαλίζοντας ότι όλες οι συναλλαγές δεδομένων καταγράφονται σε ένα αμετάβλητο καθολικό. Επιπλέον, η υιοθέτηση πιο προηγμένων πρωτοκόλλων κρυπτογράφησης και τακτικών ελέγχων ασφαλείας θα μπορούσαν να ενισχύσουν περαιτέρω την άμυνα του συστήματος έναντι των ολοένα και πιο εξελιγμένων απειλών στον κυβερνοχώρο. Με τη συνεχή εξέλιξη και ενσωμάτωση αυτών των εξελίξεων, το σύστημα μπορεί να διατηρήσει την αιχμή του ως ένα αξιόπιστο και πρωτοποριακό εργαλείο για την παρακολούθηση και τη διαχείριση της ενέργειας. Ευχαριστούμε και το chatgpt για τη βοήθεια του.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] https://lastminuteengineers.com/esp32-pinout-reference/

[2] https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

[3] https://i0.wp.com/randomnerdtutorials.com/wp-content/uploads/2018/08/ESP32-DOIT-DEVKIT-V1-Board-Pinout-36-GPIOs-updated.jpg?w=750&quality=100&strip=all&ssl=1

[4] https://grobotronics.com/gravity-analog-ac-current-sensor-20a.html

[5] https://wiki.dfrobot.com/Gravity_Analog_AC_Current_Sensor__SKU_SEN0211_

[6] https://grobotronics.com/gravity-analog-ac-current-sensor-20a.html

[7] https://www.arduino.cc/en/software

[8] https://i0.wp.com/randomnerdtutorials.com/wp-content/uploads/2016/12/windows-select-board.png?resize=614%2C520&quality=100&strip=all&ssl=1

[9] https://www.learnpython.org/

[10] https://www.tutorialspoint.com/flask/index.htm

[11] https://www.smashingmagazine.com/2018/01/understanding-using-rest-api/

[12] https://www.python.org/

[13] https://www.w3schools.com/ai/ai_chartjs.asp

[14] https://chat.openai.com/chat

ΠΑΡΑΡΤΗΜΑ Α

Στο παράρτημα αυτό αναφέρονται τα βασικά κομμάτια του κώδικα που χρησιμοποιήθηκε.

Python Server

```
from flask import Flask, render_template, session, jsonify, request,url_for, redirect
import mysql.connector


from mysql.connector import Error


app = Flask(__name__)
app.secret_key = '344365324365'


def getDbCon():
    connection = None
    try:
        connection = mysql.connector.connect(
            host='localhost',
            user='root',
            password='',
            database='nodewattdb'
        )
    except Error as e:
        print("Error while connecting to MySQL", e)
    return connection


@app.route('/wattmain')
def main():
    if 'loggedin' in session:
        return render_template('index.html', username=session['username'], node=session['nodename'])
    return redirect(url_for('login'))


@app.route('/')
def index():
    if 'loggedin' in session:
        return redirect(url_for(''))
    return redirect(url_for('login'))


@app.route('/getdata')
def data():
```

```
    connection = getDbCon()
    if connection is not None and connection.is_connected():
        cursor = connection.cursor(dictionary=True)
        cursor.execute('SELECT nodeid,mvalue,tstamp FROM mvalues WHERE nodeid = 1 ORDER BY tstamp ASC
LIMIT 100')
        values = cursor.fetchall()
        cursor.close()
        connection.close()
        return jsonify(values)
    else:
        return jsonify([])


@app.route('/login', methods=['GET', 'POST'])
def login():
    msg = ''
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']


        hashed_password = password


        connection = getDbCon()
        if connection is not None and connection.is_connected():
            cursor = connection.cursor(dictionary=True)


            cursor.execute('SELECT * FROM user WHERE username = %s AND password = %s', (username,
hashed_password,))


            user = cursor.fetchone()


        if user:
            session['loggedin'] = True
            session['id'] = user['id']
            session['username'] = user['username']


            #get nodes
            cursorU = connection.cursor(dictionary=True)
            cursorU.execute('SELECT * FROM node WHERE userid = %s', (user['id'],))
            node = cursorU.fetchone()
            session['nodeid'] = node['id']
            session['nodename'] = node['nodename']
```

```
            return redirect(url_for('main'))
        else:
            msg = 'Incorrect username/password!'
        cursor.close()
        cursorU.close()
        connection.close()
    return render_template('login.html', message=msg)

@app.route('/logout')
def logout():

    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('username', None)
    session.pop('nodeid', None)
    session.pop('nodename', None)



    return redirect(url_for('login'))

from flask import request

@app.route('/setzone', methods=['POST'])
def set_zone():
    zone = request.args.get('zone')
    value = request.args.get('value')
    #Save it to database
    return 'Zone value set'

@app.route('/add')
def add_value():
    nodeid = request.args.get('node')
    mvalue = request.args.get('currentvalue')

    try:
        # Establish a database connection
        conn = mysql.connector.connect(**db_config)

        # Check if connection was successful
```

```
        if conn.is_connected():
            # Create a new cursor
            cursor = conn.cursor()

            # SQL query
            query = "INSERT INTO mvalues (nodeid, mvalue) VALUES (%s, %s)"

            # Execute the query
            cursor.execute(query, (nodeid, mvalue, tstamp))

            # Commit the transaction
            conn.commit()

            # Close the cursor and connection
            cursor.close()
            conn.close()

            return 'Data inserted successfully!'
        else:
            return 'Failed to connect to the database'

    except Error as e:
        return f"Error: {str(e)}"

    return 'Data received'

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=4000)
```

login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Σύνδεση</title>
    <!-- Bootstrap CSS -->
    <link               href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
```

```
    <div class="container mt-5">
        <h2>Σύνδεση για Μέτρηση Κατανάλωσης</h2>
        <form action="/login" method="post">
            <div class="form-group">
                <label for="username">Username:</label>
                <input type="text" name="username" id="username" class="form-control" required>
            </div>
            <div class="form-group">
                <label for="password">Password:</label>
                <input type="password" name="password" id="password" class="form-control" required>
            </div>
            <button type="submit" class="btn btn-primary">Σύνδεση</button>
        </form>
        {% if message %}
            <div class="alert alert-warning mt-2" role="alert">
                {{message}}
            </div>
        {% endif %}
    </div>
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Κατανάλωση</title>
    <!-- Bootstrap CSS -->
    <link              href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
rel="stylesheet">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <style>
        .chart-container {
            width: 100%;
            height: auto;
            margin-bottom: 20px;
```

```
        }
    </style>
</head>
<body>
    <div class="container mt-5">
        <div class="row justify-content-center">
            <div class="col-md-12">
                <div class="card">
                    <div class="card-header">
                        <h1>Χρήστης:{{ username }}</h1>
                        <h2>Κόμβος-Συσκευή:{{ node }}</h2>
                        <a href="/logout" class="btn btn-primary">Logout</a>
                    </div>
                    <div class="card-body">
                        <div class="chart-container">
                            <h3>Δεδομένα Κατανάλωσης</h3>
                            <canvas id="WattChart"></canvas>
                        </div>
                        <div class="form-group">
                            <label for="zoneA">Ζώνη A (8-13): </label>
                            <input type="number" id="zoneA" name="zoneA" class="form-control" min="8"
max="13">
                            <button onclick="sendZoneData('zoneA')" class="btn btn-secondary mt-2">Set
Zone A</button>
                        </div>
                        <div class="form-group">
                            <label for="zoneB">Ζώνη B (12-7): </label>
                            <input type="number" id="zoneB" name="zoneB" class="form-control" min="12"
max="7">
                            <button onclick="sendZoneData('zoneB')" class="btn btn-secondary mt-2">Set
Zone B</button>
                        </div>
                        <div class="form-group">
                            <label for="AnoOrio">Αν κάποια συσκευή το ξεπερνάει για 30 λεπτά (μέτρηση
κάθε 5 λεπτά): </label>
                            <input type="text" id="AnoOrio" name="AnoOrio" class="form-control">
                            <button  onclick="sendZoneData('AnoOrio')"  class="btn  btn-secondary  mt-
2">Θέτω το Άνω Όριο</button>
                        </div>
                    </div>
                </div>
            </div>
        </div>
```

```
</div>

    <script>
        async function sendZoneData(zoneId) {
    const value = document.getElementById(zoneId).value;
    const response = await fetch('/setzone?zone=${zoneId}&value=${value}', {
        method: 'POST'
    });

    if (response.ok) {
        console.log('Zone ${zoneId} value set to ${value}');
    } else {
        console.error('Error setting zone value');
    }
}

    const WattChartElem = document.getElementById('WattChart').getContext('2d');

    const WattChart = new Chart(WattChartElem, {
        type: 'line',
        data: {
            labels: [],
            datasets: [{
                label: 'Watt',
                borderColor: 'rgb(250, 0, 0)',
            }]
        },
        options: {
            scales: {
                y: {
                    beginAtZero: true
                }
            }
        }
    });

    async function fetchData() {
        const response = await fetch('/getdata');
        const data = await response.json();
        const tstamp = data.map(item => item.tstamp);
```

```
            const mvalue = data.map(item => item.mvalue);



            WattChart.data.labels = tstamp;
            WattChart.data.datasets[0].data = mvalue;



            WattChart.update();
        }

        window.onload = fetchData;
    </script>

    <!-- Optional JavaScript; choose one of the two! -->
    <!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Κώδικας esp32

(πηγή: https://github.com/DFRobot/DFRobot_AnalogACurrentSensor/blob/master/examples/readACCurrent/readACCurrent.ino)

```
#include <WiFi.h>
#include <HTTPClient.h>

const char* ssid = "yourSSID";
const char* password = "yourPASSWORD";
const char* serverName = "http://yourserver.com/add?node=1&currentvalue="; // Flask server URL



const int ACPin = A2;          // set signal read pin
#define ACTectionRange 20;     // set Non-invasive AC Current Sensor tection range (5A,10A,20A)

// VREF: Analog reference
#define VREF 3.3
```

```
float readACCurrentValue()
{
  float ACCurrtntValue = 0;
  float peakVoltage = 0;
  float voltageVirtualValue = 0;   // Vrms
  for (int i = 0; i < 50; i++)
  {
    peakVoltage += analogRead(ACPin);   // read voltage
        peakVoltage *=peakVoltage;
    delay(1);
  }
  peakVoltage = peakVoltage / 50;
  voltageVirtualValue = peakVoltage * 0.707;    // rms

  /* The circuit is amplified by 2 times, so it is divided by 2. */
  voltageVirtualValue = (voltageVirtualValue / 1024 * VREF ) / 2;

  ACCurrtntValue = voltageVirtualValue * ACTectionRange;

  return ACCurrtntValue;
}

void setup()
{
  Serial.begin(115200);
  pinMode(13, OUTPUT);

    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.println("Connecting to WiFi...");
    }
    Serial.println("Connected to WiFi");
}

void loop()
{
  float ACCurrentValue = readACCurrentValue();   // read AC Current Value
  Serial.print(ACCurrentValue);
  Serial.println(" A");
```

```
  digitalWrite(13, HIGH);
  delay(500);
  digitalWrite(13, LOW);
  delay(500);

     // Send data to server
    if(WiFi.status() == WL_CONNECTED){
        HTTPClient http;
        String serverPath = serverName + String(ACCurrentValue);
        http.begin(serverPath.c_str());
        int httpResponseCode = http.GET();
        if (httpResponseCode>0) {
            String response = http.getString();
            Serial.println(response);
        }
        http.end();
    }
}
```