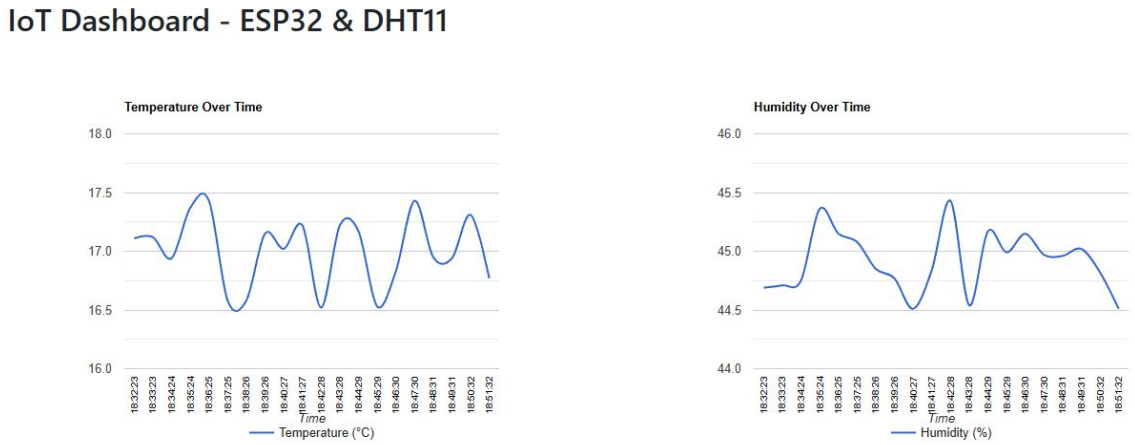
ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Απομακρυσμένη παρακολούθηση περιβαλλοντικών συνθηκών εσωτερικού χώρου μέσω IoT»



Φοιτητής

516088

ΠΑΝΑΓΙΩΤΗΣ ΜΟΚΑΛΗΣ

Επιβλέπων

Δρ. Κυριάκος Τσιακμάκης

Ιούνιος 2025

Απομακρυσμένη παρακολούθηση περιβαλλοντικών συνθηκών εσωτερικού χώρου μέσω IoT

Κωδικός: 24299

Φοιτητής: Όνομα

Εισηγητής: Δρ Κυριάκος Τσιακμάκης

Ημερομηνία ανάληψης Π.Ε. 30-10-2024

Ημερομηνία περάτωσης Π.Ε. 10-06-2025

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 εν λόγω μελέτη επικεντρώνεται στην απομακρυσμένη μέτρηση και παρακολούθηση των περιβαλλοντικών συνθηκών εσωτερικού χώρου, αξιοποιώντας τεχνολογίες IoT (Internet of Things). Χρησιμοποιούνται αισθητήρες για τη συλλογή δεδομένων όπως η θερμοκρασία, η υγρασία, επιτρέποντας τη συνεχή και αυτόματη καταγραφή τους σε πραγματικό χρόνο. Τα δεδομένα διαβιβάζονται σε κεντρικό σύστημα για ανάλυση και παρακολούθηση, με σκοπό τη βελτίωση της ποιότητας περιβάλλοντος και την υποστήριξη αυτόματων διαδικασιών προσαρμογής του χώρου στις ανάγκες των χρηστών.

«Remote Monitoring of Indoor Environmental Conditions via IoT»

Abstract

This study focuses on the remote measurement and monitoring of indoor environmental conditions, leveraging IoT (Internet of Things) technologies. Sensors are used to collect data, such as temperature and humidity, allowing for continuous and automated real-time logging. The data is transmitted to a central system for analysis and monitoring, aiming to improve environmental quality and support automated processes that adjust the space to meet users' needs.

Ευχαριστίες

Θέλω να ευχαριστήσω τους γονείς μου και τους φίλους μου για τη συμπαράσταση τους. Επίσης, να ευχαριστήσω τον επιβλέπων μου για την βοήθεια του και την καθοδήγηση του.

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1ο: Εισαγωγή

1.1 Εισαγωγή

Η παρακολούθηση των περιβαλλοντικών συνθηκών στους εσωτερικούς χώρους έχει αποκτήσει καλή σημασία τα τελευταία χρόνια κυρίως λόγω των μεγάλων απαιτήσεων για άνεση και υγεία στο σπίτι στους χώρους εργασίας αλλά και στους δημόσιους χώρους. Η ανάγκη αυτή είναι ιδιαίτερη σε κτίρια όπου διαμένουν ή εργάζονται πολλοί άνθρωποι καθώς η ποιότητα του αέρα και οι κατάλληλες συνθήκες θερμοκρασίας και υγρασίας επηρεάζουν άμεσα την υγεία των χρηστών.

Η ανάπτυξη του Διαδικτύου των Πραγμάτων (Internet of Things, IoT) έχει ανοίξει νέες προοπτικές στον τομέα της παρακολούθησης των συνθηκών εσωτερικών χώρων και παρέχει δυνατότητα συλλογής δεδομένων σε πραγματικό χρόνο. Οι IoT συσκευές επιτρέπουν τη συνεχή συλλογή και μετάδοση δεδομένων δημιουργώντας ένα αξιόπιστο ίσως σύστημα παρακολούθησης που παρέχει ακριβείς πληροφορίες.

Η χρήση αισθητήρων όπως οι αισθητήρες θερμοκρασίας και υγρασίας (π.χ. DHT11, DHT22) σε συνδυασμό με μικροελεγκτές όπως ο ESP32 καθιστά δυνατή τη συνεχή παρακολούθηση των συνθηκών αυτών. Η αποστολή των δεδομένων σε cloud πλατφόρμες όπως το δικό μας ThingSpeak προσφέρει δυνατότητες απομακρυσμένης διαχείρισης και ανάλυσης των πληροφοριών και επιτρέπει την έγκαιρη λήψη αποφάσεων για τη βελτίωση των συνθηκών του περιβάλλοντος.

Η δυνατότητα απομακρυσμένης παρακολούθησης μέσω διαδικτυακών εφαρμογών και η οπτικοποίηση των δεδομένων μέσω γραφημάτων μας δίνει σημαντικά εύκολη κατανόηση και αξιολόγηση των περιβαλλοντικών συνθηκών για τους χρήστες ή τους admin. Αυτή η τεχνολογία μπορεί να οδηγήσει σε καλύτερη ενεργειακή διαχείριση και μειωμένο κόστος και βελτιωμένη ποιότητα ζωής.

Η ανάγκη για τέτοια συστήματα γίνεται ακόμα πιο μεγάλη όταν αυξάνονται οι απαιτήσεις για υψηλότερη ποιότητα διαβίωσης και παράλληλα υπάρχει ανάγκη εξοικονόμησης ενέργειας και μείωσης του περιβαλλοντικού τυπώματος. Οι εφαρμογές IoT στη διαχείριση εσωτερικών χώρων αποτελούν ένα δυναμικά αναπτυσσόμενο πεδίο για περισσότερες επιχειρήσεις και οργανισμούς να επενδύουν στην ανάπτυξη και εφαρμογή τέτοιων λύσεων.

Η παρούσα μελέτη εστιάζει ειδικότερα στη χρήση του ESP32 ενός οικονομικού και αξιόπιστου μικροελεγκτή, για την παρακολούθηση της θερμοκρασίας και της υγρασίας μέσω του αισθητήρα DHT11. Στο πλαίσιο της εργασίας αυτής κάποιοι βασικοί στόχοι είναι η δημιουργία ενός ολοκληρωμένου συστήματος απομακρυσμένης παρακολούθησης και καταγραφής περιβαλλοντικών παραμέτρων, η αξιολόγηση της αξιοπιστίας και ακρίβειας των μετρήσεων και η διερεύνηση των δυνατοτήτων αυτόματης προσαρμογής του περιβάλλοντος για τη βελτίωση της άνεσης και της

ενεργειακής απόδοσης. Η συνεισφορά της εργασίας εντοπίζεται στη δημιουργία ενός απλού προσιτού και εύκολα επεκτάσιμου συστήματος Για να μπορεί να υλοποιηθεί σε πραγματικά σενάρια για την καλύτερη διαχείριση των συνθηκών εσωτερικού χώρου.

1.2 Δομή της εργασίας

Η εργασία είναι οργανωμένη ως εξής:

Στο δεύτερο κεφάλαιο παρουσιάζεται η βιβλιογραφική ανασκόπηση σχετικά με παρόμοια συστήματα. Στο τρίτο κεφάλαιο περιγράφεται αναλυτικά ο σχεδιασμός και η μεθοδολογία που χρησιμοποιήθηκε, συμπεριλαμβανομένων των αισθητήρων, των εργαλείων και των τεχνικών IoT.

Το τέταρτο κεφάλαιο περιλαμβάνει την υλοποίηση και τα αποτελέσματα των πειραμάτων, ενώ το πέμπτο κεφάλαιο παρουσιάζει τα συμπεράσματα, καθώς και προτάσεις για μελλοντική έρευνα.

Τέλος, στο παράρτημα παρατίθενται ενδεικτικά οι κώδικες που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής.

Κεφάλαιο 2ο: Παρόμοια Συστήματα και Βιβλιογραφική Ανασκόπηση

2.1 Βιβλιογραφική Ανασκόπηση

Στην εργασία με τίτλο "Design and Implementation of ESP32-Based IoT Devices" των Hercog, Lerher, Truntič και Težak (2023) παρουσιάζονται εκπαιδευτικά εργαλεία και τεχνολογίες του Διαδικτύου των Πραγμάτων (IoT) που απλοποιούν τον σχεδιασμό, την υλοποίηση και τη εφαρμογή εφαρμογών IoT. Οι συγγραφείς αναγνωρίζουν την αυξανόμενη παρουσία των συσκευών IoT και την ανάγκη ενσωμάτωσής τους στη διαδικασία εκπαίδευσης.

Η εργασία περιγράφει ένα εισαγωγικό μάθημα IoT που εκτελούν αρχικά οι φοιτητές χρησιμοποιώντας το ESP32, έναν μικροελεγκτή χαμηλού κόστους με δυνατότητες Wi-Fi και Bluetooth. Το μάθημα αυτό παρέχει στους φοιτητές τις βασικές γνώσεις και δεξιότητες για την ανάπτυξη εφαρμογών IoT, καλύπτοντας θέματα όπως η σύνδεση αισθητήρων και ενεργοποιητών, η συλλογή και μετάδοση δεδομένων και η χρήση πλατφορμών cloud για την αποθήκευση και ανάλυση των δεδομένων [1].

Μετά την ολοκλήρωση του εισαγωγικού μαθήματος, οι φοιτητές αναπτύσσουν και υλοποιούν δικά τους έργα IoT εφαρμόζοντας τις γνώσεις που απέκτησαν. Αυτή η προσέγγιση δίνει βάρος στη μάθηση μέσω της πράξης και ενισχύει την κατανόηση των φοιτητών σχετικά με την ανάπτυξη ολοκληρωμένων συστημάτων IoT, προετοιμάζοντάς τους για τις απαιτήσεις της σύγχρονης αγοράς εργασίας.

Στην εργασία με τίτλο "Smart Temperature Monitoring System Using ESP32 and DS18B20" του Santoso Budijono και της Felita (2021), παρουσιάζεται η ανάπτυξη ενός έξυπνου συστήματος παρακολούθησης θερμοκρασίας για καταψύκτες, με στόχο τη διασφάλιση της ποιότητας των αποθηκευμένων τροφίμων. Το σύστημα χρησιμοποιεί τον μικροελεγκτή ESP32 και τον αισθητήρα θερμοκρασίας DS18B20 για τη συνεχή καταγραφή της θερμοκρασίας εντός του καταψύκτη [2].

Η θερμοκρασία μετράται αυτόματα και τα δεδομένα αποστέλλονται σε πραγματικό χρόνο σε μια πλατφόρμα cloud επιτρέποντας την απομακρυσμένη παρακολούθηση μέσω διαδικτύου. Αυτό μειώνει την ανάγκη για χειροκίνητες μετρήσεις, αυξάνοντας την αποδοτικότητα και την αξιοπιστία του συστήματος. Το σύστημα μπορεί να ειδοποιεί τους υπεύθυνους σε περίπτωση που η θερμοκρασία υπερβεί τα προκαθορισμένα όρια, προλαμβάνοντας πιθανές αλλοιώσεις των τροφίμων.

Η υλοποίηση αυτού του συστήματος προσφέρει μια οικονομικά αποδοτική λύση για επιχειρήσεις στον τομέα των τροφίμων και ποτών, διασφαλίζοντας την ποιότητα των προϊόντων τους μέσω της συνεχούς και ακριβούς παρακολούθησης των συνθηκών αποθήκευσης.

Στην εργασία με τίτλο "Design of Standalone Asynchronous ESP32 Web-Server for Temperature and Humidity Monitoring" των Macheso et al. (2021) παρουσιάζεται η ανάπτυξη ενός αυτόνομου ασύγχρονου web server βασισμένου στον μικροελεγκτή ESP32 με στόχο την παρακολούθηση της θερμοκρασίας και της υγρασίας σε πραγματικό χρόνο. Το σύστημα αυτό σχεδιάστηκε για να παρέχει μια οικονομικά αποδοτική λύση για την παρακολούθηση περιβαλλοντικών παραμέτρων, αξιοποιώντας τον αισθητήρα DHT22 για τη μέτρηση της θερμοκρασίας και της σχετικής υγρασίας [3].

Η αρχιτεκτονική του συστήματος περιλαμβάνει τον ESP32 ο οποίος λειτουργεί ως αυτόνομος web server και είναι υπεύθυνος για τη συλλογή δεδομένων από τον αισθητήρα DHT22. Μέσω ασύγχρονων λειτουργιών ο ESP32 μπορεί να εξυπηρετεί πολλαπλούς web clients ταυτόχρονα. Επιτρέπει την πρόσβαση στα δεδομένα σε πραγματικό χρόνο χωρίς την ανάγκη συνεχούς ανανέωσης της σελίδας από τον χρήστη. Αυτό επιτυγχάνεται με τη χρήση βιβλιοθηκών όπως οι ESPAsyncWebServer και AsyncTCP, που διασφαλίζουν την αποδοτική και γρήγορη ανταπόκριση του συστήματος.

Η υλοποίηση του συστήματος έδειξε ότι είναι δυνατή η δημιουργία ενός οικονομικά αποδοτικού και ενεργειακά αποδοτικού web server για την παρακολούθηση περιβαλλοντικών συνθηκών. Η χρήση του ESP32 σε συνδυασμό με τον αισθητήρα DHT22 προσφέρει μια αξιόπιστη λύση για εφαρμογές όπου η συνεχής παρακολούθηση της θερμοκρασίας και της υγρασίας είναι κρίσιμη όπως σε βιομηχανικές διαδικασίες ή σε περιβάλλοντα όπου οι συνθήκες πρέπει να διατηρούνται εντός συγκεκριμένων ορίων.

Στην εργασία με τίτλο "Model Sistem Pengering Tenaga Surya Berbasis IoT dengan ESP32 dan DHT-11" των Gogor Christmass Setyawan και Leonard Joseph Setyawan (2025) εξετάζεται η ανάπτυξη ενός συστήματος ηλιακής ξήρανσης που ενσωματώνει τεχνολογία Διαδικτύου των Πραγμάτων (IoT) για την αύξηση της αποδοτικότητας στην ξήρανση γεωργικών προϊόντων. Η έρευνα επικεντρώνεται στα προβλήματα που αντιμετωπίζουν οι μικροκαλλιεργητές, ιδιαίτερα όσον αφορά τις αναποτελεσματικές μεθόδους ξήρανσης, και προτείνει μια λύση που χρησιμοποιεί τον μικροελεγκτή ESP32 και τον αισθητήρα DHT-11 για την παρακολούθηση της θερμοκρασίας και της υγρασίας σε πραγματικό χρόνο [4].

Στην εργασία με τίτλο "An IoT-Based Real Time Temperature and Humidity Monitoring and Data Logging System Using ESP32" των Myint, Aye και Hla (2024) παρουσιάζεται η ανάπτυξη ενός συστήματος παρακολούθησης και καταγραφής θερμοκρασίας και υγρασίας σε πραγματικό χρόνο, βασισμένου στην πλατφόρμα IoT με χρήση του μικροελεγκτή ESP32. Το σύστημα αυτό σχεδιάστηκε για να παρέχει συνεχή και αξιόπιστη παρακολούθηση περιβαλλοντικών συνθηκών, αξιοποιώντας τις δυνατότητες του ESP32 για ασύρματη επικοινωνία και επεξεργασία δεδομένων [5].

Η αρχιτεκτονική του συστήματος περιλαμβάνει τη χρήση αισθητήρων θερμοκρασίας και υγρασίας, οι οποίοι συνδέονται με τον ESP32. Ο μικροελεγκτής συλλέγει τα δεδομένα από τους αισθητήρες και τα αποστέλλει σε πραγματικό χρόνο σε μια διαδικτυακή πλατφόρμα, επιτρέποντας την απομακρυσμένη

παρακολούθηση μέσω διαδικτύου. Αυτό επιτρέπει στους χρήστες να έχουν πρόσβαση στα δεδομένα ανά πάσα στιγμή και από οποιαδήποτε τοποθεσία, διευκολύνοντας την έγκαιρη λήψη αποφάσεων σε περίπτωση ανωμαλιών στις περιβαλλοντικές συνθήκες.

Η υλοποίηση του συστήματος αυτού προσφέρει μια οικονομικά αποδοτική λύση για εφαρμογές όπου η συνεχής παρακολούθηση της θερμοκρασίας και της υγρασίας είναι κρίσιμη όπως σε βιομηχανικές εγκαταστάσεις, γεωργικές εφαρμογές ή χώρους αποθήκευσης ευαίσθητων προϊόντων. Η χρήση του ESP32 σε συνδυασμό με τα IoT τεχνολογίες κάνει το σύστημα εύκολα επεκτάσιμο και προσαρμόσιμο σε διαφορετικά περιβάλλοντα.

Στην εργασία με τίτλο "Prototype An ESP32-Based Room Humidity and Temperature Controller With IoT" των Mubarakah και Iddha (2022) παρουσιάζεται η ανάπτυξη ενός πρωτοτύπου συστήματος ελέγχου θερμοκρασίας και υγρασίας δωματίου, βασισμένου στην τεχνολογία IoT και τον μικροελεγκτή ESP32 [6].

Το σύστημα αποτελείται από τον μικροελεγκτή ESP32 όπου συνδέεται με τον αισθητήρα AM2301 για τη μέτρηση της θερμοκρασίας και της υγρασίας. Για τον έλεγχο της υγρασίας χρησιμοποιείται ένας υγραντήρας (mist maker) και για τον έλεγχο της θερμοκρασίας, αξιοποιείται η λειτουργία ενός κλιματιστικού. Ο ESP32 συλλέγει δεδομένα από τον αισθητήρα και μέσω ασύρματης σύνδεσης Wi-Fi επιτρέπει την απομακρυσμένη παρακολούθηση και ρύθμιση των συνθηκών του δωματίου μέσω διαδικτυακής εφαρμογής.

Η υλοποίηση αυτού του πρωτοτύπου συστήματος δίνει στη βελτίωση της ενεργειακής αποδοτικότητας και της άνεσης στους εσωτερικούς χώρους, παρέχοντας μια οικονομικά αποδοτική λύση για τον έλεγχο του κλίματος σε πραγματικό χρόνο. Η ενσωμάτωση της τεχνολογίας IoT επιτρέπει την ευκολότερη διαχείριση και παρακολούθηση των περιβαλλοντικών συνθηκών και κάνει το σύστημα προσαρμόσιμο σε διάφορες εφαρμογές όπως κατοικίες, γραφεία και βιομηχανικούς χώρους.

2.2 Κάποια συστήματα εμπορίου

2.2.1 Poseidon2 4002

Το Poseidon2 4002 είναι μια συσκευή παρακολούθησης περιβαλλοντικών συνθηκών, σχεδιασμένη για χρήση σε κέντρα δεδομένων και βιομηχανικές εφαρμογές. Διαθέτει δυνατότητα σύνδεσης έως και 40 αισθητήρων: 16 μέσω 1-Wire UNI/1-Wire και 24 μέσω RS-485. Επιπλέον, παρέχει 12 ψηφιακές εισόδους (dry contact) και δυνατότητα ελέγχου 4 ρελέ εξόδων (NO/NC). Η συσκευή διαθέτει

ενσωματωμένο web server για διαμόρφωση και μπορεί να παρακολουθείται απομακρυσμένα μέσω διαδικτύου, χρησιμοποιώντας την πύλη SensDesk Technology σε συνδυασμό με την εφαρμογή SensDesk Mobile για iOS και Android. Επιπλέον, υποστηρίζει πρωτόκολλα όπως HTTPs, IPv6, SNMPv3 και MQTT, επιτρέποντας την ενσωμάτωση σε λύσεις IoT [7].

2.2.2 Σύστημα Παρακολούθησης και Ελέγχου Θερμοκηπίων από τη Smartnet

Η Smartnet προσφέρει ένα σύστημα παρακολούθησης και ελέγχου θερμοκηπίων με βάρος στη διατήρηση των κατάλληλων επιπέδων θερμοκρασίας και υγρασίας για την ορθή ανάπτυξη των φυτών. Το σύστημα επιτρέπει τη σύνδεση διαφόρων τύπων αισθητήρων όπως θερμοκρασίας, υγρασίας και ανίχνευσης διαρροής νερού. Οι μετρήσεις αποστέλλονται σε πραγματικό χρόνο σε μια διαδικτυακή πλατφόρμα, επιτρέποντας την απομακρυσμένη παρακολούθηση και τον έλεγχο των συνθηκών του θερμοκηπίου. Αυτό συμβάλλει στη βελτίωση της παραγωγικότητας και της ποιότητας των καλλιεργειών και τη μείωση του κόστους λειτουργίας [8].

2.2.3 Milesight IoT Λύσεις για Έξυπνη Γεωργία

Η Milesight προσφέρει λύσεις IoT βασισμένες στην τεχνολογία LoRaWAN, ειδικά σχεδιασμένες για εφαρμογές έξυπνης γεωργίας. Η σειρά προϊόντων περιλαμβάνει αισθητήρες, πύλες (gateways), ελεγκτές και οθόνες. Οι αισθητήρες μπορούν να μετρούν παραμέτρους όπως θερμοκρασία, υγρασία, υγρασία εδάφους και επίπεδα φωτός δίνοντας χρήσιμα δεδομένα για τη διαχείριση των καλλιεργειών. Η τεχνολογία LoRaWAN επιτρέπει την ασύρματη μετάδοση δεδομένων σε μεγάλες αποστάσεις με χαμηλή κατανάλωση ενέργειας, καθιστώντας την ιδανική για γεωργικές εφαρμογές. Αυτές οι λύσεις βοηθούν τους αγρότες να λαμβάνουν ενημερωμένες αποφάσεις και βελτιώνουν την απόδοση των καλλιεργειών [9].

Κάθε ένα από αυτά τα συστήματα προσφέρει δικά του χαρακτηριστικά και πλεονεκτήματα προσαρμοσμένα σε διαφορετικές ανάγκες και εφαρμογές στον τομέα της παρακολούθησης περιβαλλοντικών συνθηκών και της έξυπνης γεωργίας.

Κεφάλαιο 3ο: Τεχνολογίες Υλοποίησης και Ανάπτυξης του Συστήματος

3.1 Υλικό

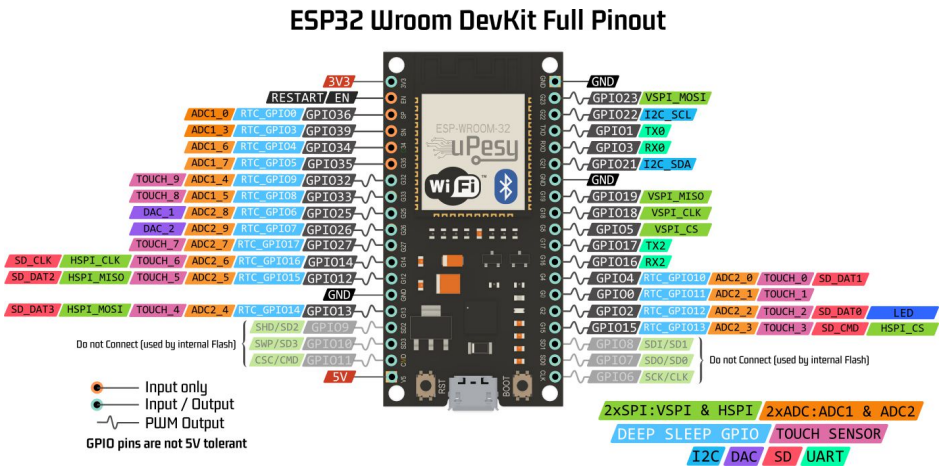
3.1.1 Esp32

Ο ESP32 είναι ένας ισχυρός χαμηλού κόστους μικροελεγκτής, αναπτυγμένος από την Espressif Systems. Είναι μια εταιρεία που εξειδικεύεται στην ανάπτυξη λύσεων για το Διαδίκτυο των Πραγμάτων (IoT). Από την πρώτη του κυκλοφορία ο ESP32 έχει γίνει ιδιαίτερα δημοφιλής λόγω της ενσωματωμένης υποστήριξης Wi-Fi και Bluetooth χαρακτηριστικά που τον κάνουν ιδανικό για εφαρμογές IoT [10].

Το Wi-Fi επιτρέπει στον ESP32 να συνδέεται εύκολα σε δίκτυα, παρέχοντας δυνατότητα απομακρυσμένης επικοινωνίας και ελέγχου. Η υποστήριξη Bluetooth προσθέτει δυνατότητες σύνδεσης σε μικρότερη εμβέλεια όπως είναι η επικοινωνία με κινητές συσκευές, αισθητήρες ή άλλες έξυπνες συσκευές. Ο συνδυασμός αυτών των δύο ασύρματων τεχνολογιών δίνει μεγάλη ευελιξία σε διάφορες εφαρμογές όπως είναι οικιακή αυτοματοποίηση και παρακολούθηση περιβαλλοντικών συνθηκών με τις φορητές συσκευές.

Η χαμηλή κατανάλωση ενέργειας και η υψηλή απόδοση του ESP32 επιτρέπουν την ενσωμάτωσή του ακόμα και σε εφαρμογές με περιορισμένους ενεργειακούς πόρους όπως φορητές ή αυτόνομες συσκευές που λειτουργούν με μπαταρία. Λόγω των χαρακτηριστικών αυτών ο ESP32 είναι από πολλούς ως ο πιο κατάλληλος μικροελεγκτής για ένα ευρύ φάσμα εφαρμογών IoT [11].

Τεχνικά χαρακτηριστικά του ESP32



Εικόνα 3.1: esp32 pinout

[https://www.upesy.com/cdn/shop/files/doc-esp32-pinout-reference-wroom-devkit.png?width=692]

Η αρχιτεκτονική του ESP32 είναι σχεδιασμένη για υψηλή απόδοση. Ο ESP32 βασίζεται σε έναν διπύρηνο επεξεργαστή Xtensa 32-bit LX6 με ταχύτητα έως και 240 MHz. Οι δύο πυρήνες μπορούν να λειτουργούν είτε ανεξάρτητα εκτελώντας διαφορετικές εργασίες ταυτόχρονα είτε συνεργατικά ώστε να μοιράζονται τη φόρτιση απαιτητικών εφαρμογών. Αυτό κάνει δυνατή την αποτελεσματική διαχείριση πολλαπλών εργασιών (multitasking) και την εκτέλεση σύνθετων διαδικασιών όπως επεξεργασία δεδομένων σε πραγματικό χρόνο και τη διαχείριση ασύρματης επικοινωνίας.

Ο ESP32 διαθέτει προηγμένη μονάδα διαχείρισης ενέργειας (power management unit) η οποία επιτρέπει τη βελτιστοποίηση της κατανάλωσης ενέργειας ανάλογα με τις απαιτήσεις της εφαρμογής. Αυτό περιλαμβάνει τη δυνατότητα λειτουργίας σε διάφορα επίπεδα ενεργειακής κατανάλωση όπως τα deep-sleep modes, τα οποία μειώνουν σημαντικά την κατανάλωση όταν ο μικροελεγκτής δεν εκτελεί κρίσιμες εργασίες κάνωντας τον ιδανικό για εφαρμογές με μπαταρία ή περιορισμένους ενεργειακούς πόρους.

Εκτός από τους πυρήνες επεξεργασίας, ο ESP32 περιλαμβάνει και διάφορα ενσωματωμένα περιφερειακά όπως ADC, DAC, GPIO pins, καθώς και ποικίλα πρωτόκολλα επικοινωνίας (UART, SPI, I^2C, I^2S) για επεκτασιμότητα για την ενσωμάτωση ποικίλων αισθητήρων και συσκευών.

Προγραμματιστικό περιβάλλον

Ο ESP32 ξεχωρίζει για την υποστήριξη πολλαπλών και διαφορετικών προγραμματιστικών περιβαλλόντων και παρέχει ευελιξία τόσο σε αρχάριους όσο και σε επαγγελματίες προγραμματιστές.

Το Arduino IDE αποτελεί το πιο δημοφιλές περιβάλλον για αρχάριους χρήστες. Είναι εύκολο στη χρήση, διαθέτει μεγάλη κοινότητα υποστήριξης και πλούσια βιβλιοθήκη έτοιμων παραδειγμάτων και κώδικα, διευκολύνοντας την ανάπτυξη εφαρμογών χωρίς ιδιαίτερη εμπειρία στον προγραμματισμό ενσωματωμένων συστημάτων. Η απλότητα του Arduino IDE επιτρέπει στους χρήστες να επικεντρωθούν στη γρήγορη υλοποίηση της ιδέας τους.

Το ESP-IDF (Espressif IoT Development Framework) αποτελεί ένα πιο προηγμένο και επαγγελματικό πλαίσιο ανάπτυξης. Είναι το επίσημο προγραμματιστικό περιβάλλον της Espressif, προσφέροντας πλήρη έλεγχο στις λειτουργίες και στο υλικό του ESP32. Παρέχει εργαλεία για λεπτομερή παραμετροποίηση του μικροελεγκτή, δυνατότητες multitasking μέσω FreeRTOS και καλύτερη διαχείριση πόρων και είναι κατάλληλο για απαιτητικές ή επαγγελματικές εφαρμογές που απαιτούν υψηλή αξιοπιστία και αποτελεσματικότητα.

Το MicroPython παρέχει μια εύκολη και απλή εισαγωγή στον προγραμματισμό του ESP32. Είναι μια έκδοση της γλώσσας Python προσαρμοσμένη για μικροελεγκτές, επιτρέποντας στους αρχάριους αλλά και σε όσους είναι ήδη εξοικειωμένοι με την Python να αναπτύξουν γρήγορα εφαρμογές χωρίς

δύσκολες διαδικασίες μεταγλώττισης. Το MicroPython είναι ιδιαίτερα χρήσιμο για την εκμάθηση βασικών εννοιών προγραμματισμού IoT και για τη γρήγορη υλοποποίηση εφαρμογών.

Εφαρμογές και δυνατότητες

Χάρη στις δυνατότητες ασύρματης επικοινωνίας και το μικρό του μέγεθος, ο ESP32 είναι ιδανικός για:

- Συστήματα έξυπνου σπιτιού και αυτοματισμούς
- Συστήματα παρακολούθησης και τηλεμετρίας
- Εφαρμογές φορητών συσκευών και αισθητήρων
- Βιομηχανικά συστήματα IoT

Σύνδεση με αισθητήρες (DHT11)

Στην παρούσα εργασία, ο ESP32 συνδέεται με τον αισθητήρα DHT11 έναν απλό/οικονομικό και ευρέως διαδεδομένο αισθητήρα για τη μέτρηση της θερμοκρασίας και της σχετικής υγρασίας σε περιβάλλοντα εσωτερικού ή εξωτερικού χώρου. Η σύνδεση γίνεται μέσω ενός μόνο ψηφιακού GPIO pin, χρησιμοποιώντας ένα πρωτόκολλο επικοινωνίας single-wire, το οποίο μειώνει την πολυπλοκότητα της καλωδίωσης και διευκολύνει σημαντικά τη διαδικασία εγκατάστασης και συντήρησης του συστήματος [12].

Ο αισθητήρας DHT11 παρέχει ψηφιακή έξοδο και δεν εχει την ανάγκη για χρήση επιπλέον κυκλωμάτων μετατροπής και εξασφαλίζοντας σταθερές και αξιόπιστες μετρήσεις. Λόγω της απλότητάς του και του χαμηλού κόστους είναι ιδανικός για εφαρμογές όπως η απομακρυσμένη παρακολούθηση συνθηκών σε σπίτια, γραφεία, θερμοκήπια. Η χρήση του αισθητήρα DHT11 σε συνδυασμό με τον ESP32 παρέχει τη δυνατότητα για γρήγορη και εύκολη ανάπτυξη συστημάτων IoT με αξιόπιστη συλλογή δεδομένων.

Επικοινωνία και αποστολή δεδομένων

Τα δεδομένα που συλλέγονται από τον αισθητήρα αποστέλλονται μέσω HTTP σε πλατφόρμες cloud, όπως το ThingSpeak. Το HTTP είναι ένα απλό πρωτόκολλο επικοινωνίας που επιτρέπει εύκολη ενσωμάτωση με διάφορες διαδικτυακές πλατφόρμες. Η χρήση πλατφορμών cloud έχει δυνατότητες ανάλυσης δεδομένων σε πραγματικό χρόνο, οπτικοποίησης μέσω διαδραστικών γραφημάτων και απομακρυσμένης πρόσβασης στα δεδομένα από οποιοδήποτε σημείο και μας δίνει τη δυνατότητα για άμεση λήψη αποφάσεων και την καλύτερη διαχείριση.

Πλεονεκτήματα και μειονεκτήματα

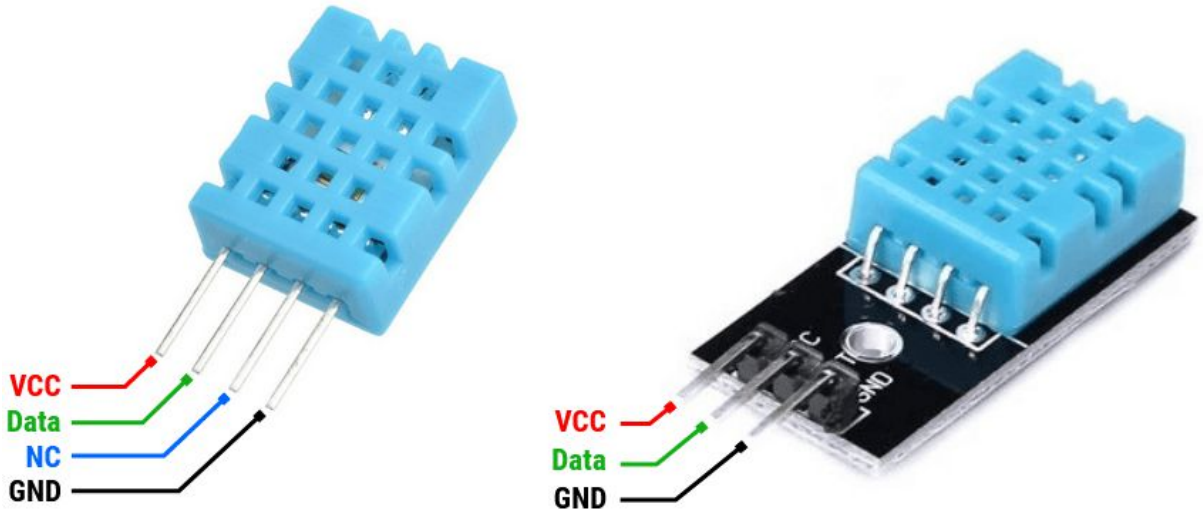
Ο ESP32 ξεχωρίζει για την υψηλή του απόδοση το χαμηλό κόστος και την ενεργειακή του αποδοτικότητα στοιχεία που τον καθιστούν εξαιρετική επιλογή για εφαρμογές IoT και ενσωματωμένα συστήματα. Η δυνατότητά του να διαχειρίζεται αποτελεσματικά ασύρματες επικοινωνίες και να εκτελεί παράλληλα πολλαπλές εργασίες τον κάνουν αποτελεσματικό στην υλοποίηση σύνθετων εφαρμογών.

Κάποια μειονεκτήματα περιλαμβάνουν περιορισμένη εσωτερική μνήμη RAM (έως 520 KB), η οποία μπορεί να αποδειχθεί ανεπαρκής για πολύ απαιτητικές εφαρμογές που διαχειρίζονται μεγάλα δεδομένα ή σύνθετες διαδικασίες. Οι αναλογικοί μετατροπείς (ADC) του ESP32 ενδέχεται να παρουσιάζουν ορισμένους περιορισμούς στην ακρίβεια και τη σταθερότητα των μετρήσεων και απαιτεί προσεκτική επιλογή και ρύθμιση για εφαρμογές με υψηλές απαιτήσεις ακριβείας.

Όμως ο ESP32 αποτελεί ιδανική λύση για πληθώρα εφαρμογών IoT και είναι μια εξαιρετική λύση. Η πρακτική του αξία επιβεβαιώνεται στην ενσωμάτωσή του σε συστήματα όπως αυτό που αναπτύχθηκε στην παρούσα εργασία όπου αποδείχθηκε ιδιαίτερα αποτελεσματικός και εύχρηστος σε πραγματικές εφαρμογές.

3.1.2 DHT11

Ο αισθητήρας DHT11 είναι ένας ψηφιακός αισθητήρας θερμοκρασίας και υγρασίας χαμηλού κόστους και καλής αξιοπιστίας. Αποτελεί μια από τις πιο δημοφιλείς λύσεις για εφαρμογές που απαιτούν βασική παρακολούθηση των περιβαλλοντικών συνθηκών.



Εικόνα 3.2: DHT11

[https://cdn.circuitgeeks.com/wp-content/uploads/2021/11/DHT11-Pinout.png]

Ο DHT11 προσφέρει μέτρηση θερμοκρασίας από 0 έως 50 βαθμούς Κελσίου με ακρίβεια ±2°C και σχετική υγρασία από 20% έως 90% με ακρίβεια ±5%. Τα δεδομένα παρέχονται σε ψηφιακή μορφή, διευκολύνοντας την επεξεργασία από μικροελεγκτές όπως ο ESP32.

Η επικοινωνία του αισθητήρα DHT11 με τον μικροελεγκτή πραγματοποιείται μέσω πρωτοκόλλου single-wire. Το πρωτόκολλο αυτό απαιτεί τη χρήση ενός μόνο ψηφιακού GPIO pin, απλοποιώντας σημαντικά τη διαδικασία σύνδεσης και εγκατάστασης.

Ο αισθητήρας περιλαμβάνει ένα θερμίστορ για τη μέτρηση της θερμοκρασίας και έναν αισθητήρα υγρασίας, που συνδυάζονται για την παραγωγή ενός ψηφιακού σήματος εξόδου. Κάθε κύκλος μέτρησης διαρκεί περίπου 2 δευτερόλεπτα, γεγονός που περιορίζει τη συχνότητα ενημέρωσης των δεδομένων αλλά επαρκεί για τις περισσότερες εφαρμογές παρακολούθησης περιβάλλοντος.

Το βασικό πλεονέκτημα του DHT11 είναι το χαμηλό του κόστος και η ευκολία χρήσης. Είναι ιδιαίτερα κατάλληλος για εκπαιδευτικές εφαρμογές, έργα DIY και βασική παρακολούθηση σε οικιακούς ή επαγγελματικούς χώρους.

Οι περιορισμοί στην ακρίβεια και το εύρος μέτρησης μπορεί να είναι ανεπαρκείς για επαγγελματικές ή βιομηχανικές εφαρμογές που απαιτούν υψηλή ακρίβεια ή συνεχή και ταχεία ενημέρωση δεδομένων. Σε αυτές τις περιπτώσεις, πιο προηγμένοι αισθητήρες όπως ο DHT22 είναι προτιμότεροι.

Εφαρμογές του DHT11

Ο DHT11 χρησιμοποιείται ευρέως σε:

- Συστήματα έξυπνου σπιτιού
- Εφαρμογές αυτοματισμού θερμοκηπίων
- Συστήματα παρακολούθησης περιβάλλοντος σε κτίρια
- Εκπαιδευτικά προγράμματα και έργα DIY

Στην παρούσα εργασία, ο αισθητήρας DHT11 επιλέχθηκε λόγω του χαμηλού κόστους και της απλής ενσωμάτωσής του με τον μικροελεγκτή ESP32, προσφέροντας μια αξιόπιστη λύση για την απομακρυσμένη παρακολούθηση των βασικών περιβαλλοντικών παραμέτρων, της θερμοκρασίας και της υγρασίας.

3.2 Λογισμικό

3.2.1 Python

Η Python είναι μια υψηλού επιπέδου διερμηνευόμενη γλώσσα προγραμματισμού γενικού σκοπού, γνωστή για την απλότητά της, την καθαρότητα της σύνταξης και την ευκολία εκμάθησης. Δημιουργήθηκε από τον Guido van Rossum και κυκλοφόρησε το 1991, έχοντας ως βασικό στόχο να ενθαρρύνει την αναγνωσιμότητα και την ευκολία γραφής κώδικα. Χάρη στη δομή και την ευελιξία της, η Python έχει καταστεί μία από τις πιο δημοφιλείς γλώσσες προγραμματισμού στον κόσμο [13-16].

Χαρακτηριστικά της Python

- Ευκολία εκμάθησης και χρήσης: Η σύνταξη της Python είναι απλή και κατανοητή, γεγονός που την καθιστά ιδανική για αρχάριους προγραμματιστές.

- Αντικειμενοστραφής Προγραμματισμός: Υποστηρίζει αντικειμενοστραφή, διαδικαστικό και συναρτησιακό προγραμματισμό.

- Δυναμική Τυποποίηση: Δεν απαιτεί δήλωση τύπου για τις μεταβλητές, διευκολύνοντας την ανάπτυξη κώδικα.

- Μεγάλη βιβλιοθήκη και κοινότητα: Η Python διαθέτει μια εκτεταμένη συλλογή βιβλιοθηκών και μια δυναμική κοινότητα προγραμματιστών, που παρέχει υποστήριξη και εργαλεία για κάθε είδους εφαρμογές.

Χρήσεις και Εφαρμογές της Python

Η Python χρησιμοποιείται σε ένα ευρύ φάσμα τομέων, όπως:

• **Ανάπτυξη Ιστοσελίδων:** Με τη χρήση πλαισίων όπως τα Django και Flask, η Python επιτρέπει την εύκολη και γρήγορη ανάπτυξη δυναμικών και ασφαλών ιστοσελίδων.

• **Επιστήμη Δεδομένων και Μηχανική Μάθηση:** Χρησιμοποιείται ευρέως για ανάλυση δεδομένων, δημιουργία μοντέλων μηχανικής μάθησης και στατιστικής ανάλυσης με βιβλιοθήκες όπως Pandas, NumPy, Scikit-learn και TensorFlow.

• **Αυτοματοποίηση Εργασιών:** Διευκολύνει την αυτοματοποίηση επαναλαμβανόμενων διαδικασιών, όπως η διαχείριση αρχείων, η αποστολή email, η συλλογή δεδομένων από το διαδίκτυο κ.ά.

• **Ανάπτυξη Παιχνιδιών:** Με βιβλιοθήκες όπως το Pygame, η Python επιτρέπει την ανάπτυξη απλών 2D παιχνιδιών.

• **Τεχνητή Νοημοσύνη (AI) και Ρομποτική:** Χρησιμοποιείται για την ανάπτυξη έξυπνων συστημάτων, αναλύοντας μεγάλα σύνολα δεδομένων για τη λήψη αποφάσεων.

• **Διαχείριση Δικτύου και Ασφάλεια:** Χρησιμοποιείται για την ανάλυση δικτύου, την ανάπτυξη συστημάτων ασφάλειας και την ανίχνευση κακόβουλου λογισμικού.

Πλεονεκτήματα της Python

1. **Απλή και Εύκολη στη Μάθηση:** Η Python διαθέτει καθαρή και κατανοητή σύνταξη, καθιστώντας την ιδανική για αρχάριους προγραμματιστές. Η δομή της γλώσσας επιτρέπει την ταχύτερη εκμάθηση σε σχέση με άλλες γλώσσες προγραμματισμού.
2. **Μεγάλη Κοινότητα και Υποστήριξη:** Η Python έχει μια από τις μεγαλύτερες κοινότητες προγραμματιστών παγκοσμίως. Αυτό σημαίνει ότι υπάρχουν πολλά διαθέσιμα resources, βιβλιοθήκες και εργαλεία, όπως και εκτενής τεκμηρίωση.
3. **Ευρύ Οικοσύστημα Βιβλιοθηκών:** Διαθέτει πληθώρα βιβλιοθηκών για επιστημονικούς υπολογισμούς, ανάπτυξη εφαρμογών web, τεχνητή νοημοσύνη, μηχανική μάθηση, και πολλά άλλα.
4. **Διαλειτουργικότητα:** Η Python μπορεί να ενσωματωθεί εύκολα με άλλες γλώσσες προγραμματισμού όπως C, C++, Java, επιτρέποντας την ανάπτυξη σύνθετων και αποδοτικών εφαρμογών.
5. **Πλατφόρμα Ανεξαρτησίας:** Ο κώδικας Python είναι φορητός και μπορεί να εκτελεστεί σε διαφορετικά λειτουργικά συστήματα, όπως Windows, Linux, και macOS.
6. **Εύκολη Ενσωμάτωση με Web Services:** Προσφέρει ισχυρή υποστήριξη για την ενσωμάτωση με διαδικτυακές υπηρεσίες και APIs, κάνοντάς την ιδανική για την ανάπτυξη εφαρμογών που απαιτούν διαδικτυακή επικοινωνία.
7. **Ισχυρή Υποστήριξη για Ανάπτυξη Prototypes:** Η Python επιτρέπει τη γρήγορη ανάπτυξη και δοκιμή πρωτοτύπων, καθιστώντας τη κατάλληλη για startups και ερευνητικά έργα.

Μειονεκτήματα της Python

1. **Χαμηλότερη Ταχύτητα Εκτέλεσης:** Ως διερμηνευόμενη γλώσσα, η Python είναι πιο αργή σε εκτέλεση σε σύγκριση με γλώσσες όπως η C ή η Java, ειδικά σε εφαρμογές που απαιτούν υψηλή απόδοση.
2. **Κατανάλωση Μνήμης:** Η Python τείνει να χρησιμοποιεί περισσότερη μνήμη σε σύγκριση με άλλες γλώσσες, κάτι που μπορεί να είναι πρόβλημα σε εφαρμογές με περιορισμένους πόρους.
3. **Περιορισμοί σε Κινητές Συσκευές:** Η Python δεν είναι ιδανική για την ανάπτυξη εφαρμογών κινητών τηλεφώνων λόγω περιορισμένης υποστήριξης και χαμηλότερης απόδοσης σε τέτοια περιβάλλοντα.
4. **Περιορισμένη Χρήση σε Παιχνίδια και Συστήματα Επιπέδου Πυρήνα:** Η Python δεν προτιμάται για την ανάπτυξη παιχνιδιών υψηλής απόδοσης ή συστημάτων επιπέδου πυρήνα, όπου οι γλώσσες χαμηλού επιπέδου είναι πιο αποτελεσματικές.
5. **Λιγότερο Ικανή για Πολυνηματικές Εφαρμογές:** Το Global Interpreter Lock (GIL) περιορίζει τη δυνατότητα της Python να εκμεταλλεύεται πλήρως τα πλεονεκτήματα της πολυνηματικής επεξεργασίας σε πολυπύρηνους επεξεργαστές.

Στην παρούσα εργασία η Python χρησιμοποιείται για την ανάπτυξη ενός web server χρησιμοποιώντας το πλαίσιο Flask. Ο server αυτός είναι υπεύθυνος για την ανάκτηση και την οπτικοποίηση των δεδομένων που αποστέλλονται από τον αισθητήρα DHT11 στο ThingSpeak. Η Python επιλέχθηκε λόγω της ευκολίας που προσφέρει στην ανάπτυξη web εφαρμογών και της πλούσιας συλλογής βιβλιοθηκών για χειρισμό δεδομένων και της ευελιξίας της στην ενσωμάτωση APIs και εξωτερικών υπηρεσιών.

Μέσω της Python αναπτύχθηκε η λογική για την ανάκτηση δεδομένων από το ThingSpeak μέσω HTTP αιτήσεων. Στη συνέχεια τα δεδομένα επεξεργάζονται και παρουσιάζονται στον τελικό χρήστη με τη μορφή πινάκων και διαγραμμάτων. Χάρη στην ευκολία ενσωμάτωσης βιβλιοθηκών όπως το Pandas για την επεξεργασία δεδομένων και το Matplotlib για την απεικόνιση, η Python διευκόλυνε σημαντικά την ανάπτυξη και την παρουσίαση της εφαρμογής.

Η Python προσφέρει δυνατότητες για την εύκολη παραμετροποίηση της web εφαρμογής για την αποδοτική διαχείριση δεδομένων και την απλή προσαρμογή στις απαιτήσεις της εφαρμογής. Η επιλογή της Python και του Flask ήταν πολύ καλή λόγω της απλότητας της ταχύτητας ανάπτυξης και της ευελιξίας της.

3.2.1.1 Χρήση της Python

Η Python αποτελεί μια γλώσσα προγραμματισμού με χαμηλό βαθμό δυσκολίας για αρχάριους διαθέτει απλή σύνταξη και εκτενές οικοσύστημα εργαλείων και βιβλιοθηκών. Ακολουθεί ένας συνοπτικός οδηγός εκμάθησης για όσους επιθυμούν να ξεκινήσουν τη χρήση της Python.

Εγκατάσταση Python

Για να εγκατασταθεί η Python, το πρώτο βήμα είναι η επίσκεψη στην επίσημη ιστοσελίδα https://www.python.org.

Από εκεί, μπορεί να γίνει λήψη και εγκατάσταση της τελευταίας έκδοσης, ανάλογα με το λειτουργικό σύστημα του χρήστη. Κατά την εγκατάσταση, προτείνεται η ενεργοποίηση της επιλογής **"Add Python to PATH"**.

Για τη δημιουργία ενός απλού προγράμματος, ο χρήστης μπορεί να δημιουργήσει ένα αρχείο με κατάληξη `.py` και να γράψει τον εξής κώδικα:

```
print("Γεια σου, Κόσμε!")
```

Για την εκτέλεση του προγράμματος σε περιβάλλον Windows, η εντολή είναι η εξής:

```
python app.py
```

Μεταβλητές και Τύποι Δεδομένων:

```
name = "Python"
version = 3.12
is_popular = True
print(name, version, is_popular)
```

Συνθήκες (if statements):

```
age = int(input("Ποια είναι η ηλικία σου; "))
if age >= 18:
    print("Είσαι ενήλικας.")
else:
    print("Είσαι ανήλικος.")
```

Βρόχοι (Loops):

```
for i in range(5):
    print("Επαναλαμβάνω:", i)
```

Συναρτήσεις

```
def greet(name):
    return "Γεια σου, " + name + "!"

print(greet("Python"))
```

Χρήση Βιβλιοθηκών

Για την πραγματοποίηση HTTP αιτήσεων, μπορεί να χρησιμοποιηθεί η βιβλιοθήκη `requests`:

```
import requests
response = requests.get("https://jsonplaceholder.typicode.com/posts")
print(response.json())
```

Δημιουργία Απλού Web Server με Flask

Αρχικά, απαιτείται η εγκατάσταση του Flask:

```
pip install flask
```

Στη συνέχεια, δημιουργείται ο εξής κώδικας:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return "Καλώς ήρθατε στον Flask Server!"

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

Η εκτέλεση του κώδικα οδηγεί στη λειτουργία ενός τοπικού διακομιστή στη διεύθυνση http://127.0.0.1:5000/.

3.2.2 Flask

Το Flask είναι ένα μικρό και ευέλικτο πλαίσιο ανάπτυξης web εφαρμογών για την Python. Δημιουργήθηκε το 2010 από τον Armin Ronacher, έναν προγραμματιστή που αναζητούσε έναν απλό και ελαφρύ τρόπο για την ανάπτυξη εφαρμογών ιστού. Το Flask ξεκίνησε ως ένα "πείραμα" για να αξιοποιηθεί η βιβλιοθήκη Werkzeug (για τον διακομιστή HTTP) και η βιβλιοθήκη Jinja2 (για την επεξεργασία πρότυπων HTML)[17-19].

Σε αντίθεση με άλλα δημοφιλή πλαίσια όπως το Django, το Flask ακολουθεί τη φιλοσοφία "micro-framework", προσφέροντας μόνο τα βασικά εργαλεία και αφήνοντας στον προγραμματιστή την

επιλογή να ενσωματώσει επιπλέον βιβλιοθήκες ανάλογα με τις ανάγκες του έργου. Αυτή η προσέγγιση προσφέρει μεγαλύτερη ελευθερία και ευελιξία, κάνοντάς το ιδανικό για μικρές και μεσαίες εφαρμογές.

Γιατί Flask;

Το Flask προτιμάται για πολλούς λόγους, κυρίως λόγω της απλότητάς του και της δυνατότητας που προσφέρει στους προγραμματιστές να ελέγχουν κάθε πτυχή της εφαρμογής τους. Είναι ιδανικό για:

- **Ταχεία ανάπτυξη πρωτοτύπων** λόγω της απλής σύνταξης και της εύκολης ενσωμάτωσης νέων λειτουργιών.
- **Μικρές και ευέλικτες εφαρμογές**, όπου δεν απαιτείται η πολυπλοκότητα ενός μεγαλύτερου πλαισίου.
- **Εκπαίδευση και πειραματισμό**, καθώς η απλότητα του το καθιστά ιδανικό για όσους μαθαίνουν web development.

Εφαρμογές και Δυνατότητες του Flask

Το Flask προσφέρει βασικές λειτουργίες για την ανάπτυξη web εφαρμογών, ενώ μέσω της χρήσης εξωτερικών βιβλιοθηκών, οι δυνατότητές του μπορούν να επεκταθούν σημαντικά. Οι βασικές εφαρμογές του Flask περιλαμβάνουν:

- **Δημιουργία δυναμικών ιστοσελίδων** με χρήση πρότυπων Jinja2.
- **Διαχείριση βάσεων δεδομένων** με εργαλεία όπως SQLAlchemy.
- **Δημιουργία RESTful APIs** για αποστολή και λήψη δεδομένων μέσω διαδικτύου.
- **Ενσωμάτωση με εξωτερικές υπηρεσίες**, όπως APIs τρίτων κατασκευαστών.
- **Διαχείριση χρηστών** με λειτουργίες εγγραφής, σύνδεσης και εξουσιοδότησης.

Παράδειγμα βασικής εφαρμογής Flask:

```
from flask import Flask, render_template


app = Flask(__name__)


@app.route('/')
def home():
    return "Καλώς ήρθατε στο Flask!"
```

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

Πλεονεκτήματα του Flask

1. **Απλότητα και Ευελιξία:** Το Flask είναι εξαιρετικά απλό και παρέχει μεγάλη ευελιξία, επιτρέποντας την προσαρμογή της εφαρμογής σύμφωνα με τις ανάγκες του έργου.
2. **Ελαφρύ και Γρήγορο:** Δεν επιβαρύνει τον προγραμματιστή με επιπλέον λειτουργίες που μπορεί να μην χρειάζεται.
3. **Ευκολία στην Εκμάθηση:** Λόγω της απλής σύνταξης, οι νέοι προγραμματιστές μπορούν να μάθουν γρήγορα πώς να το χρησιμοποιούν.
4. **Επεκτασιμότητα:** Μπορεί να ξεκινήσει ως μικρή εφαρμογή και να εξελιχθεί σε πιο σύνθετη, με την ενσωμάτωση επιπλέον βιβλιοθηκών.
5. **Μεγάλη Κοινότητα και Τεκμηρίωση:** Υπάρχει πλούσιο υλικό για εκμάθηση, παραδείγματα και έτοιμα εργαλεία.

Μειονεκτήματα του Flask

1. **Απουσία Ενσωματωμένων Εργαλείων:** Σε αντίθεση με το Django, το Flask δεν παρέχει ενσωματωμένα εργαλεία για διαχείριση χρηστών, φόρμες, ή ORM, γεγονός που απαιτεί περισσότερη εργασία από τον προγραμματιστή.
2. **Διαχείριση Πολυπλοκότητας:** Σε μεγαλύτερα έργα, η διαχείριση της πολυπλοκότητας μπορεί να γίνει δύσκολη, καθώς απαιτείται η επιλογή και ενσωμάτωση επιπλέον εργαλείων.
3. **Αντιμετώπιση Σφαλμάτων:** Ορισμένα σφάλματα ενδέχεται να είναι δύσκολα στην ανίχνευση λόγω της απλότητας της αρχιτεκτονικής του Flask.

Γιατί Flask στην Παρούσα Εργασία

Στην παρούσα εργασία, το Flask επιλέχθηκε ως το βασικό πλαίσιο για την ανάπτυξη της web εφαρμογής λόγω της απλότητας, της ευελιξίας και της ταχύτητας ανάπτυξης που προσφέρει. Ειδικότερα, το Flask διευκόλυνε την υλοποίηση των παρακάτω λειτουργιών:

1. **Ανάκτηση Δεδομένων:** Η δυνατότητα εύκολης ενσωμάτωσης της βιβλιοθήκης requests για την ανάκτηση δεδομένων από την πλατφόρμα ThingSpeak.

2. **Επεξεργασία και Παρουσίαση Δεδομένων:** Η χρήση βιβλιοθηκών όπως pandas για την επεξεργασία των δεδομένων και matplotlib για την οπτικοποίησή τους μέσω γραφημάτων.
3. **Ευελιξία στην Ανάπτυξη:** Το Flask επέτρεψε την ανάπτυξη της εφαρμογής με ευκολία και προσαρμοστικότητα, διατηρώντας τον κώδικα καθαρό και κατανοητό.
4. **Απλότητα στην Επέκταση:** Το έργο μπορεί να αναπτυχθεί περαιτέρω στο μέλλον με την προσθήκη νέων λειτουργιών, όπως η αποστολή ειδοποιήσεων ή η δημιουργία επιπλέον API endpoints.

Το Flask αποδείχθηκε το καλό εργαλείο για την επίτευξη των στόχων της παρούσας εργασίας για ταχύτητα ανάπτυξης και ευκολία ενσωμάτωσης εξωτερικών εργαλείων.

3.2.2.1 Το Πλαίσιο Flask

Εγκατάσταση Flask

Πριν την έναρξη, είναι απαραίτητη η εγκατάσταση του Flask. Αυτό μπορεί να γίνει εύκολα μέσω του pip:

pip install flask

Δημιουργία της Πρώτης Εφαρμογής Flask

Η πρώτη απλή εφαρμογή Flask μπορεί να υλοποιηθεί με τα παρακάτω βήματα:

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def home():
    return "Καλώς ήρθατε στην πρώτη σας εφαρμογή Flask!"

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

- Το @app.route('/') δηλώνει την αρχική διαδρομή (route) της εφαρμογής.
- Η μέθοδος app.run(debug=True) εκκινεί τον διακομιστή σε κατάσταση ανάπτυξης, επιτρέποντας την άμεση ανίχνευση λαθών.

Δημιουργία Δυναμικών Διαδρομών

Οι διαδρομές μπορούν να γίνουν δυναμικές, λαμβάνοντας παραμέτρους από τον χρήστη:

```
@app.route('/user/<username>')
def show_user(username):
    return f"Γεια σου, {username}!"
```

Χρήση Πρότυπων (Templates)

Το Flask υποστηρίζει πρότυπα HTML μέσω του Jinja2. Δημιουργήστε ένα αρχείο templates/home.html:

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Αρχική Σελίδα</title>
</head>
<body>
    <h1>Καλώς ήρθατε, {{ name }}!</h1>
</body>
</html>
```

Και ανακαλέστε το από το Flask:

```
from flask import render_template

@app.route('/home/<name>')
def home(name):
    return render_template('home.html', name=name)
```

Διαχείριση Φόρμας και Εισαγωγή Δεδομένων

Η διαχείριση δεδομένων από φόρμες είναι απλή στο Flask.

```
from flask import request

@app.route('/submit', methods=['POST'])
def submit():
    username = request.form['username']
    return f"Υποβλήθηκε από τον χρήστη: {username}"
```

Και το αντίστοιχο πρότυπο HTML:

```
<form method="POST" action="/submit">
    <input type="text" name="username">
    <input type="submit" value="Υποβολή">
</form>
```

Ενσωμάτωση με Βάση Δεδομένων (SQLite)

Το Flask μπορεί να ενσωματωθεί εύκολα με SQLite:

```
import sqlite3


@app.route('/add/<name>')
def add_user(name):
    connection = sqlite3.connect('database.db')
    cursor = connection.cursor()
    cursor.execute("INSERT INTO users (name) VALUES (?)", (name,))
    connection.commit()
    connection.close()
    return f"Ο χρήστης {name} προστέθηκε!"
```

Χειρισμός Σφαλμάτων

Ο Flask παρέχει ενσωματωμένες λειτουργίες για την αντιμετώπιση σφαλμάτων:

```
@app.errorhandler(404)
def page_not_found(e):
    return "Η σελίδα δεν βρέθηκε.", 404
```

Εξωτερικά APIs και Ενσωμάτωση

Μπορεί να γίνει εύκολη ενσωμάτωση με εξωτερικά APIs χρησιμοποιώντας τη βιβλιοθήκη requests:

```
import requests


@app.route('/api')
def api():
    response = requests.get('https://jsonplaceholder.typicode.com/posts')
    return response.json()
```

Επέκταση Εφαρμογής με Blueprints

Τα Blueprints διευκολύνουν την οργάνωση μεγάλων εφαρμογών:

```
from flask import Blueprint

bp = Blueprint('users', __name__)

@bp.route('/profile')
def profile():
    return "Προφίλ Χρήστη"

app.register_blueprint(bp, url_prefix='/users')
```

Βέλτιστες Πρακτικές Ανάπτυξης με Flask

- **Χρήση Εικονικών Περιβαλλόντων (Virtual Environments):**
- **Οργάνωση του Έργου:** Δημιουργία δομής με φακέλους για templates, static αρχεία και modules.
- **Χρήση Περιβαλλοντικών Μεταβλητών:** Για ασφαλή διαχείριση κλειδιών API και διαπιστευτηρίων.

Τεστ και Ανάπτυξη

Για την ανάπτυξη και τον έλεγχο της εφαρμογής:

```
export FLASK_APP=app.py
export FLASK_ENV=development
flask run
```

Ο παραπάνω οδηγός καλύπτει τα βασικά βήματα για την ανάπτυξη μιας απλής αλλά επεκτάσιμης εφαρμογής Flask. Η πρακτική εξάσκηση και η συνεπής εμβάθυνση στις δυνατότητες του πλαισίου οδηγούν στην αποτελεσματικότερη ανάπτυξη εφαρμογών web.

3.2.3 Bootstrap

Το Bootstrap είναι ένα από τα πιο δημοφιλή και ισχυρά πλαίσια ανάπτυξης ιστοσελίδων (front-end framework). Δημιουργήθηκε από τους Mark Otto και Jacob Thornton στο Twitter το 2011. Στόχος του ήταν να παρέχει μια συνεπή και αποδοτική προσέγγιση για την ανάπτυξη responsive και mobile-first ιστοσελίδων. Χρησιμοποιήθηκε εσωτερικά από την ομάδα του Twitter και σύντομα διατέθηκε ως open-source έργο και έχει τεράστια αποδοχή από την κοινότητα των προγραμματιστών [20-22].

Η επιτυχία του Bootstrap βασίζεται στην απλότητα, την ευκολία χρήσης και την εκτενή συλλογή προκαθορισμένων στοιχείων UI (User Interface). Παρέχει μια ολοκληρωμένη λύση για τη δημιουργία καλαίσθητων και λειτουργικών ιστοσελίδων με ελάχιστο προγραμματισμό CSS και JavaScript.

Το Bootstrap επιλέγεται συχνά λόγω της ευκολίας που προσφέρει στην ανάπτυξη responsive και ελκυστικών ιστοσελίδων. Οι βασικοί λόγοι χρήσης του περιλαμβάνουν:

- **Ευκολία και Ταχύτητα Ανάπτυξης:** Παρέχει έτοιμα components και προκαθορισμένα styles, μειώνοντας το χρόνο ανάπτυξης.
- **Responsive Σχεδίαση:** Υποστηρίζει εξ ορισμού την ανάπτυξη ιστοσελίδων που προσαρμόζονται σε όλες τις συσκευές (mobile, tablet, desktop).
- **Συμβατότητα με Πολλά Προγράμματα Περιήγησης:** Το Bootstrap λειτουργεί άψογα σε όλα τα σύγχρονα browsers.
- **Εκτενής Τεκμηρίωση:** Διαθέτει πλούσιο υλικό με παραδείγματα και οδηγίες χρήσης.

Εφαρμογές και Δυνατότητες του Bootstrap

Το Bootstrap παρέχει ένα ευρύ φάσμα εργαλείων και δυνατοτήτων για τη δημιουργία σύγχρονων ιστοσελίδων:

- **Responsive Grid System:** Ένα ευέλικτο σύστημα πλεγμάτων για την οργάνωση περιεχομένου σε σελίδες.
- **Προκαθορισμένα Components:** Έτοιμα στοιχεία UI, όπως κουμπιά, κάρτες, modals, φόρμες και πλοήγηση (navbar).
- **Utilities:** Κλάσεις για εύκολη προσαρμογή περιθωρίων, γραμματοσειρών, χρωμάτων και άλλων στοιχείων.

- **JavaScript Plugins:** Προκαθορισμένα plugins για λειτουργίες όπως carousels, modals, tooltips και alerts.

Παράδειγμα Βασικής Δομής Bootstrap

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
  <title>Απλή Σελίδα με Bootstrap</title>
</head>
<body>
  <div class="container">
    <h1 class="text-center">Καλώς ήρθατε στο Bootstrap</h1>
    <button class="btn btn-primary">Κουμπί Bootstrap</button>
  </div>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Πλεονεκτήματα του Bootstrap

1. **Γρήγορη Ανάπτυξη:** Παρέχει έτοιμα templates και components που επιταχύνουν τη διαδικασία ανάπτυξης.
2. **Ανταπόκριση (Responsive Design):** Επιτρέπει την εύκολη ανάπτυξη ιστοσελίδων που προσαρμόζονται σε όλες τις συσκευές.
3. **Συμβατότητα με Πολλαπλούς Browsers:** Οι ιστοσελίδες με Bootstrap λειτουργούν άψογα σε Chrome, Firefox, Safari, Edge κ.ά.
4. **Εύκολη Εκμάθηση:** Η απλή σύνταξη και η τεκμηρίωση καθιστούν το Bootstrap ιδανικό για αρχάριους προγραμματιστές.
5. **Εκτενής Τεκμηρίωση:** Διαθέτει πληθώρα παραδειγμάτων, tutorials και υποστηρικτική κοινότητα.

6. **Ευκολία Προσαρμογής:** Παρέχει τη δυνατότητα προσαρμογής των προκαθορισμένων styles μέσω custom CSS.

Μειονεκτήματα του Bootstrap

1. **Ομοιομορφία Σχεδίασης:** Οι εφαρμογές που βασίζονται αποκλειστικά σε Bootstrap μπορεί να μοιάζουν μεταξύ τους, εκτός αν γίνει εκτενής παραμετροποίηση.
2. **Μεγάλο Μέγεθος Αρχείων:** Η χρήση όλων των components μπορεί να οδηγήσει σε αυξημένο μέγεθος αρχείων, επηρεάζοντας την ταχύτητα φόρτωσης.
3. **Περιορισμένη Ευελιξία:** Αν και το Bootstrap είναι ευέλικτο, μπορεί να περιορίσει την πλήρη προσαρμογή για custom σχεδιασμούς.
4. **Περίπλοκος Κώδικας:** Η υπερβολική χρήση προκαθορισμένων κλάσεων μπορεί να κάνει τον κώδικα δύσκολα αναγνώσιμο.

Γιατί Bootstrap στην Παρούσα Εργασία

Η επιλογή του Bootstrap στην παρούσα εργασία έγινε για λόγους ευκολίας, ταχύτητας ανάπτυξης και προσαρμοστικότητας. Συγκεκριμένα:

1. **Γρήγορη Ανάπτυξη UI:** Το Bootstrap επέτρεψε την άμεση δημιουργία ενός καλαίσθητου και λειτουργικού περιβάλλοντος χρήστη, χωρίς την ανάγκη συγγραφής εκτεταμένου CSS.
2. **Responsive Design:** Η δυνατότητα για responsive σχεδίαση εξασφάλισε ότι η ιστοσελίδα λειτουργεί άψογα σε διαφορετικές συσκευές, όπως κινητά τηλέφωνα, tablets και desktops.
3. **Εύκολη Ενσωμάτωση με Flask:** Το Bootstrap ενσωματώθηκε στην Flask εφαρμογή, παρέχοντας τα απαιτούμενα στοιχεία UI χωρίς την ανάγκη επιπλέον παραμετροποίησης.
4. **Συμβατότητα με Πολλαπλούς Browsers:** Η χρήση του Bootstrap συμβαλλε στη σωστή λειτουργία της εφαρμογής σε όλα τα σύγχρονα προγράμματα περιήγησης.
5. **Εκτενής Τεκμηρίωση και Υποστήριξη:** Καθώς το Bootstrap διαθέτει μεγάλη κοινότητα, η επίλυση προβλημάτων και η ανεύρεση παραδειγμάτων ήταν άμεση.

Το Bootstrap αποτέλεσε το ιδανικό εργαλείο για την ανάπτυξη ενός λειτουργικού, ευέλικτου και προσαρμοστικού περιβάλλοντος χρήστη, επιτυγχάνοντας τους στόχους της παρούσας εργασίας.

3.2.4 Charts

Το Google Charts είναι μια ισχυρή και δωρεάν βιβλιοθήκη γραφημάτων που παρέχεται από την Google. Δημιουργήθηκε με στόχο να παρέχει στους προγραμματιστές μια εύκολη και αποτελεσματική λύση για την οπτικοποίηση δεδομένων σε εφαρμογές ιστού [23-24].

Από την κυκλοφορία του, το Google Charts έχει γίνει ένα από τα πιο δημοφιλή εργαλεία για τη δημιουργία διαδραστικών και προσαρμόσιμων γραφημάτων, καθώς ενσωματώνεται εύκολα σε HTML σελίδες, υποστηρίζει μεγάλο εύρος τύπων γραφημάτων και προσφέρει εκτενή παραμετροποίηση.

Γιατί Google Charts;

Το Google Charts επιλέγεται για την ανάπτυξη διαδραστικών γραφημάτων λόγω των εξής λόγων:

- **Ευκολία Χρήσης:** Παρέχει απλή και κατανοητή σύνταξη για την ανάπτυξη γραφημάτων.
- **Δωρεάν και Πλήρως Παραμετροποιήσιμο:** Είναι δωρεάν και παρέχει πολλές επιλογές προσαρμογής.
- **Διαδραστικότητα:** Υποστηρίζει διαδραστικά γραφήματα με δυνατότητες zoom, hover και επιλογής δεδομένων.
- **Εκτενής Τεκμηρίωση:** Παρέχει πλούσια τεκμηρίωση και παραδείγματα.
- **Ενσωμάτωση με APIs:** Ενσωματώνεται εύκολα με άλλες υπηρεσίες της Google και APIs.

Εφαρμογές και Δυνατότητες του Google Charts

Το Google Charts παρέχει ευρύ φάσμα τύπων γραφημάτων και εργαλείων για την οπτικοποίηση δεδομένων. Μερικές βασικές εφαρμογές και δυνατότητες περιλαμβάνουν:

- **Δημιουργία Διαδραστικών Γραφημάτων:** Όπως γραμμικά, ραβδογράμματα, πίτες, θερμικοί χάρτες, γεωγραφικά γραφήματα.
- **Υποστήριξη Δυναμικών Δεδομένων:** Δυνατότητα ενημέρωσης των γραφημάτων σε πραγματικό χρόνο.
- **Υποστήριξη Εξαγωγής:** Δυνατότητα εξαγωγής γραφημάτων σε μορφές εικόνας (PNG, SVG).
- **Προσαρμογή Εμφάνισης:** Παρέχει επιλογές για την προσαρμογή χρωμάτων, γραμματοσειρών, τίτλων και υποτίτλων.

Παράδειγμα Βασικής Χρήσης Google Charts

```
<!DOCTYPE html>
<html>
 <head>
  <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
  <script type="text/javascript">
   google.charts.load('current', {'packages':['corechart']});
   google.charts.setOnLoadCallback(drawChart);

   function drawChart() {
    var data = google.visualization.arrayToDataTable([
     ['Έτος', 'Πωλήσεις'],
     ['2019', 1000],
     ['2020', 1170],
     ['2021', 660],
     ['2022', 1030]
    ]);

    var options = {
     title: 'Πωλήσεις Ανά Έτος',
     curveType: 'function',
     legend: { position: 'bottom' }
    };

    var chart = new google.visualization.LineChart(document.getElementById('curve_chart'));

    chart.draw(data, options);
   }
  </script>
 </head>
 <body>
  <div id="curve_chart" style="width: 900px; height: 500px"></div>
 </body>
```

```
</html>
```

Πλεονεκτήματα του Google Charts

1. **Δωρεάν και Προσβάσιμο:** Δεν απαιτείται άδεια ή πληρωμή για τη χρήση του.
2. **Εύκολη Ενσωμάτωση:** Μπορεί να ενσωματωθεί άμεσα σε οποιαδήποτε ιστοσελίδα.
3. **Διαδραστικότητα:** Παρέχει δυναμικές και διαδραστικές δυνατότητες απεικόνισης.
4. **Πολλαπλές Επιλογές Γραφημάτων:** Υποστηρίζει πληθώρα τύπων γραφημάτων για κάθε ανάγκη.
5. **Υποστήριξη Πολλαπλών Γλωσσών:** Υποστηρίζει διεθνείς γλώσσες και μορφοποιήσεις.
6. **Συμβατότητα με Πολλαπλούς Browsers:** Διασφαλίζει τη σωστή λειτουργία σε όλους τους σύγχρονους browsers.

Μειονεκτήματα του Google Charts

1. **Απαιτεί Σύνδεση στο Διαδίκτυο:** Χρειάζεται ενεργή σύνδεση στο διαδίκτυο για τη φόρτωση της βιβλιοθήκης.
2. **Περιορισμοί Παραμετροποίησης:** Παρόλο που παρέχει πολλές επιλογές, κάποιες εξειδικευμένες προσαρμογές μπορεί να είναι δύσκολες.
3. **Εξάρτηση από την Google:** Η απόδοση και η λειτουργικότητα εξαρτώνται από τις υπηρεσίες της Google.
4. **Διαχείριση Μεγάλων Δεδομένων:** Η οπτικοποίηση πολύ μεγάλων συνόλων δεδομένων μπορεί να είναι απαιτητική σε πόρους.

Γιατί Google Charts στην Παρούσα Εργασία

Η χρήση του Google Charts στην παρούσα εργασία κρίθηκε απαραίτητη για τους εξής λόγους:

1. **Εύκολη Οπτικοποίηση Δεδομένων:** Το Google Charts διευκόλυνε την οπτικοποίηση δεδομένων θερμοκρασίας και υγρασίας από την πλατφόρμα ThingSpeak.
2. **Διαδραστικά Γραφήματα:** Προσέφερε τη δυνατότητα στους χρήστες να αλληλεπιδρούν με τα γραφήματα, κάνοντας zoom ή προβολή συγκεκριμένων περιόδων.
3. **Απλότητα Ενσωμάτωσης:** Το Google Charts ενσωματώθηκε άμεσα στη Flask εφαρμογή, βελτιώνοντας την αισθητική και λειτουργικότητα της ιστοσελίδας.

4. **Αξιοπιστία και Απόδοση:** Η υποστήριξη από την Google εγγυάται σταθερή απόδοση και σωστή λειτουργία της οπτικοποίησης.
5. **Προσαρμογή Εμφάνισης:** Παρείχε δυνατότητες για την προσαρμογή χρωμάτων, γραφημάτων και τίτλων, εξασφαλίζοντας την ομοιομορφία στο design της εφαρμογής.

Η επιλογή του Google Charts ενίσχυσε σημαντικά την ποιότητα παρουσίασης των δεδομένων, προσφέροντας μια αξιόπιστη και αποδοτική λύση για την ανάλυση και απεικόνιση των αποτελεσμάτων.

3.2.5 Data Tables

Το DataTables είναι μια δημοφιλής βιβλιοθήκη JavaScript που χρησιμοποιείται για τη δημιουργία διαδραστικών και ευέλικτων πινάκων δεδομένων σε ιστοσελίδες. Αναπτύχθηκε από τον Allan Jardine το 2007, με στόχο την απλοποίηση της διαχείρισης και παρουσίασης μεγάλων συνόλων δεδομένων σε περιβάλλοντα ιστού [25].

Από την κυκλοφορία του, το DataTables έχει εξελιχθεί σε ένα από τα πιο ισχυρά εργαλεία για την παρουσίαση δεδομένων, υποστηρίζοντας χαρακτηριστικά όπως αναζήτηση, ταξινόμηση, σελιδοποίηση και εξαγωγή δεδομένων.

Γιατί DataTables;

Η επιλογή του DataTables βασίζεται σε πολλούς λόγους που σχετίζονται με την ευκολία χρήσης και την ευελιξία του:

- **Ευκολία Εγκατάστασης και Χρήσης:** Απλή ενσωμάτωση μέσω CDN ή εγκατάσταση μέσω npm.
- **Δυνατότητα Αναζήτησης, Ταξινόμησης και Σελιδοποίησης:** Παρέχει αυτοματοποιημένα χαρακτηριστικά που βελτιώνουν τη διαχείριση των δεδομένων.
- **Υποστήριξη AJAX:** Επιτρέπει τη δυναμική φόρτωση δεδομένων από APIs.
- **Εκτενής Παραμετροποίηση:** Παρέχει πλήθος επιλογών για την προσαρμογή της εμφάνισης και της λειτουργικότητας.
- **Εξαγωγή Δεδομένων:** Υποστηρίζει εξαγωγή σε μορφές CSV, Excel και PDF.
- **Συμβατότητα με Πολλαπλούς Browsers:** Λειτουργεί άψογα σε όλους τους σύγχρονους browsers.

Εφαρμογές και Δυνατότητες του DataTables

Το DataTables χρησιμοποιείται ευρέως για τη δημιουργία και διαχείριση δυναμικών πινάκων δεδομένων σε:

- **Εφαρμογές Διαχείρισης Δεδομένων:** Όπως συστήματα CRM, ERP και πλατφόρμες διαχείρισης προϊόντων.
- **Εφαρμογές Διαχείρισης Χρηστών:** Παρουσίαση λιστών χρηστών με δυνατότητες επεξεργασίας και αναζήτησης.
- **Οικονομικές Αναφορές και Πίνακες Στατιστικών:** Για την απεικόνιση και διαχείριση πολύπλοκων συνόλων δεδομένων.
- **Δυναμικές Πλατφόρμες Web:** Όπου απαιτείται η διαχείριση μεγάλου όγκου δεδομένων.

Παράδειγμα Βασικής Χρήσης DataTables

```
<!DOCTYPE html>
<html>
<head>
  <title>Παράδειγμα DataTables</title>
  <link rel="stylesheet" href="https://cdn.datatables.net/1.11.3/css/jquery.dataTables.min.css">
  <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
  <script src="https://cdn.datatables.net/1.11.3/js/jquery.dataTables.min.js"></script>
  <script>
    $(document).ready(function() {
      $('#example').DataTable();
    });
  </script>
</head>
<body>
  <table id="example" class="display" style="width:100%">
    <thead>
      <tr>
        <th>Όνομα</th>
        <th>Επώνυμο</th>
        <th>Ηλικία</th>
```

```
            <th>Πόλη</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>Γιάννης</td>
            <td>Παπαδόπουλος</td>
            <td>30</td>
            <td>Αθήνα</td>
        </tr>
        <tr>
            <td>Μαρία</td>
            <td>Κωνσταντίνου</td>
            <td>25</td>
            <td>Θεσσαλονίκη</td>
        </tr>
    </tbody>
  </table>
</body>
</html>
```

Πλεονεκτήματα του DataTables

1. **Εύκολη Χρήση:** Η απλή σύνταξη και ενσωμάτωση διευκολύνουν την ταχεία ανάπτυξη.
2. **Διαδραστικότητα:** Παρέχει αυτόματες δυνατότητες ταξινόμησης, αναζήτησης και σελιδοποίησης.
3. **Υποστήριξη AJAX:** Δυνατότητα φόρτωσης δεδομένων από APIs.
4. **Επεκτασιμότητα:** Παρέχει πλήθος επεκτάσεων για επιπλέον λειτουργίες.
5. **Εξαγωγή Δεδομένων:** Υποστηρίζει εξαγωγή σε διαφορετικές μορφές, όπως CSV, Excel και PDF.
6. **Υποστήριξη Πολλαπλών Browsers:** Λειτουργεί απρόσκοπτα σε όλους τους σύγχρονους browsers.

Μειονεκτήματα του DataTables

1. **Απαιτεί Χρήση jQuery:** Εξαρτάται από τη βιβλιοθήκη jQuery, κάτι που μπορεί να περιορίσει τη χρήση σε ορισμένα σύγχρονα frameworks.
2. **Απόδοση σε Μεγάλα Δεδομένα:** Σε πολύ μεγάλα σύνολα δεδομένων, ενδέχεται να απαιτείται πρόσθετη βελτιστοποίηση.
3. **Περιορισμοί Παραμετροποίησης:** Παρόλο που είναι παραμετροποιήσιμο, ορισμένες εξειδικευμένες προσαρμογές μπορεί να είναι περίπλοκες.
4. **Μάθηση API:** Απαιτεί χρόνο για την πλήρη κατανόηση των δυνατοτήτων του API του DataTables.

Γιατί DataTables στην Παρούσα Εργασία

Η χρήση του DataTables στην παρούσα εργασία κρίθηκε κατάλληλη για τους εξής λόγους:

1. **Εύκολη Οργάνωση και Παρουσίαση Δεδομένων:** Διευκόλυνε την παρουσίαση των δεδομένων θερμοκρασίας και υγρασίας από την πλατφόρμα ThingSpeak σε μορφή πίνακα.
2. **Διαδραστικότητα:** Παρείχε τη δυνατότητα στους χρήστες να αναζητούν, να ταξινομούν και να σελιδοποιούν τα δεδομένα.
3. **Εύκολη Ενσωμάτωση με Flask:** Το DataTables ενσωματώθηκε άμεσα στην Flask εφαρμογή.
4. **Υποστήριξη Εξαγωγής Δεδομένων:** Παρείχε την δυνατότητα εξαγωγής των δεδομένων σε μορφή Excel ή CSV για περαιτέρω ανάλυση.
5. **Αξιοπιστία και Απόδοση:** Παρείχε σταθερή απόδοση και ευκολία στη διαχείριση μεγάλων ποσοτήτων δεδομένων.

Η επιλογή του DataTables ενίσχυσε τη λειτουργικότητα της εφαρμογής, προσφέροντας μια δυναμική και αξιόπιστη λύση για την απεικόνιση και τη διαχείριση δεδομένων.

3.3 Πλατφόρμα ThingSpeak

Το ThingSpeak είναι μια ανοικτή πλατφόρμα Internet of Things (IoT) που επιτρέπει τη συλλογή, αποθήκευση, ανάλυση και οπτικοποίηση δεδομένων από αισθητήρες και άλλες συσκευές IoT. Δημιουργήθηκε το 2010 από την εταιρεία ioBridge και αργότερα αποκτήθηκε από την MathWorks, την εταιρεία που αναπτύσσει το MATLAB [26-27].

Η πλατφόρμα σχεδιάστηκε για να παρέχει μια εύκολη λύση για την αποστολή και ανάλυση δεδομένων από συνδεδεμένες συσκευές. Με την πάροδο των ετών το ThingSpeak εξελίχθηκε σε ένα από τα πιο δημοφιλή εργαλεία για την παρακολούθηση και διαχείριση IoT δεδομένων υποστηρίζοντας επίσης την ανάλυση δεδομένων με MATLAB.

Το ThingSpeak επιλέγεται συχνά για IoT εφαρμογές λόγω των παρακάτω χαρακτηριστικών:

- **Ευκολία Χρήσης:** Παρέχει ένα απλό API για την αποστολή και ανάκτηση δεδομένων.
- **Δωρεάν Πρόσβαση:** Διατίθεται δωρεάν για μη εμπορική χρήση με περιορισμούς.
- **Ενσωμάτωση με MATLAB:** Επιτρέπει την ανάλυση δεδομένων με χρήση MATLAB scripts απευθείας στην πλατφόρμα.
- **Διαδραστική Οπτικοποίηση:** Προσφέρει εργαλεία για τη δημιουργία γραφημάτων και την παρακολούθηση των δεδομένων σε πραγματικό χρόνο.
- **Υποστήριξη APIs:** Παρέχει RESTful API για αποστολή και ανάκτηση δεδομένων.

Το ThingSpeak χρησιμοποιείται ευρέως σε IoT έργα για την παρακολούθηση και διαχείριση δεδομένων. Οι βασικές του δυνατότητες περιλαμβάνουν:

- **Αποστολή Δεδομένων από Συσκευές IoT:** Μέσω HTTP POST αιτήσεων.
- **Ανάκτηση Δεδομένων:** Μέσω HTTP GET αιτήσεων για προβολή ή περαιτέρω ανάλυση.
- **Οπτικοποίηση Δεδομένων:** Δημιουργία δυναμικών γραφημάτων και widgets.
- **Εκτέλεση MATLAB Scripts:** Αυτόματη ανάλυση και επεξεργασία δεδομένων.
- **Δημιουργία Ειδοποιήσεων:** Ρύθμιση triggers για αποστολή ειδοποιήσεων.

Παράδειγμα Βασικής Χρήσης ThingSpeak με Python

```
import requests


def send_data(api_key, field1_value):
    url = "https://api.thingspeak.com/update"
    payload = {'api_key': api_key, 'field1': field1_value}
    response = requests.post(url, data=payload)
    return response.status_code
```

```
# Παράδειγμα αποστολής δεδομένων
api_key = "YOUR_API_KEY"
status = send_data(api_key, 25)
print("Status Code:", status)
```

Ανάκτηση Δεδομένων από ThingSpeak

```
def get_data(channel_id, api_key):
    url = f"https://api.thingspeak.com/channels/{channel_id}/feeds.json?api_key={api_key}"
    response = requests.get(url)
    return response.json()

# Παράδειγμα ανάκτησης δεδομένων
data = get_data("CHANNEL_ID", "API_KEY")
print(data)
```

Πλεονεκτήματα του ThingSpeak

1. **Ευκολία Χρήσης:** Εύκολη ενσωμάτωση με IoT συσκευές και απλή διαδικασία αποστολής/ανάκτησης δεδομένων.
2. **Δωρεάν Πρόσβαση:** Παρέχει βασικές υπηρεσίες χωρίς κόστος για μη εμπορική χρήση.
3. **Ενσωμάτωση με MATLAB:** Δυνατότητα ανάλυσης δεδομένων μέσω MATLAB scripts.
4. **Διαδραστική Οπτικοποίηση:** Παροχή δυναμικών γραφημάτων για την παρακολούθηση δεδομένων σε πραγματικό χρόνο.
5. **Ευκολία στην Ανάπτυξη Πρωτοτύπων:** Ιδανικό για ανάπτυξη και δοκιμή IoT έργων.
6. **Ασφάλεια και Αξιοπιστία:** Υποστηρίζει ασφαλή αποστολή δεδομένων μέσω API keys.

1. Μειονεκτήματα του ThingSpeak

2. **Περιορισμοί Δωρεάν Έκδοσης:** Περιορισμένος αριθμός updates ανά ημέρα στη δωρεάν έκδοση.
3. **Απαίτηση Σύνδεσης στο Διαδίκτυο:** Απαιτεί σύνδεση στο διαδίκτυο για την αποστολή και ανάκτηση δεδομένων.
4. **Περιορισμένες Επιλογές Παραμετροποίησης στην Οπτικοποίηση:** Αν και παρέχει βασικά γραφήματα οι δυνατότητες παραμετροποίησης είναι περιορισμένες.

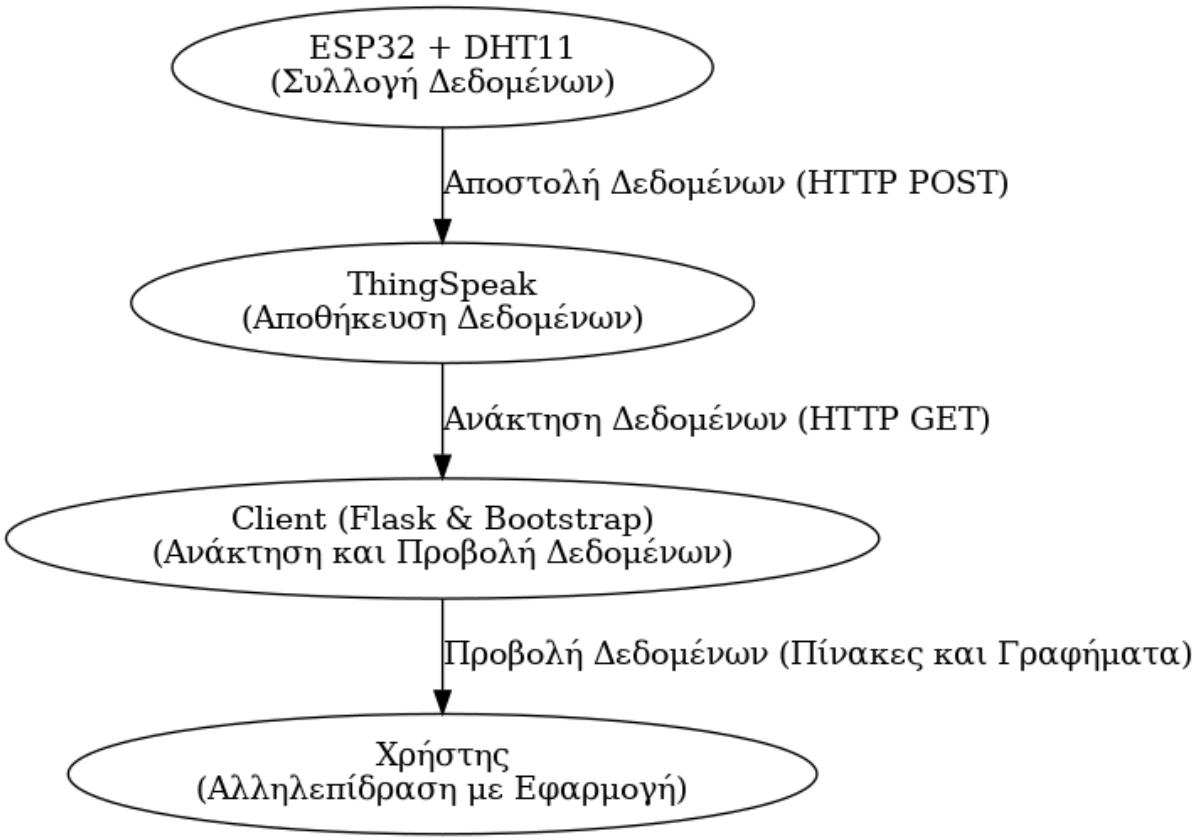
5. **Εξάρτηση από Εξωτερικό Πάροχο:** Η λειτουργία εξαρτάται από τις υπηρεσίες της MathWorks.

Η επιλογή του ThingSpeak για την παρούσα εργασία βασίστηκε σε πολλούς παράγοντες:

1. **Ευκολία Ενσωμάτωσης με ESP32:** Το ThingSpeak υποστηρίζει απλή αποστολή δεδομένων μέσω HTTP POST, καθιστώντας την ενσωμάτωση με τον αισθητήρα ESP32 εξαιρετικά απλή.
2. **Απλότητα Ανάκτησης Δεδομένων με Flask:** Η ανάκτηση δεδομένων και η προβολή τους σε Flask εφαρμογή ήταν γρήγορη και αποδοτική.
3. **Διαδραστική Οπτικοποίηση:** Η χρήση δυναμικών γραφημάτων επέτρεψε την άμεση παρακολούθηση των τιμών θερμοκρασίας και υγρασίας.
4. **Αξιοπιστία και Απόδοση:** Παρείχε σταθερή και αξιόπιστη λειτουργία καθ' όλη τη διάρκεια της ανάπτυξης της εργασίας.
5. **Ευκολία Ανάλυσης:** Η δυνατότητα ανάλυσης δεδομένων με MATLAB προσέφερε επιπλέον δυνατότητες εμβάθυνσης και επεξεργασίας.

Κεφάλαιο 4ο: Το σύστημα μέτρησης

4.1 Εισαγωγή στη διαδικασία



Εικόνα 4.1: Ανάλυση Ροής Δεδομένων

Το παραπάνω διάγραμμα απεικονίζει την πλήρη ροή δεδομένων στο σύστημα απομακρυσμένης παρακολούθησης περιβαλλοντικών συνθηκών μέσω IoT. Κάθε στάδιο αναλύεται λεπτομερώς παρακάτω.

ESP32 + DHT11 (Συλλογή Δεδομένων)

Η διαδικασία ξεκινά με τη συλλογή δεδομένων από τον αισθητήρα **DHT11** που είναι συνδεδεμένος με το μικροελεγκτή **ESP32**. Ο DHT11 μετρά δύο βασικές περιβαλλοντικές παραμέτρους:

- **Θερμοκρασία**
- **Υγρασία**

Ο ESP32, μέσω των GPIO pins και του ενσωματωμένου του προγραμματισμού, ανακτά περιοδικά τα δεδομένα από τον αισθητήρα και τα προετοιμάζει για αποστολή στο διαδίκτυο μέσω Wi-Fi.

Αποστολή Δεδομένων στο ThingSpeak (HTTP)

Τα συλλεγόμενα δεδομένα αποστέλλονται μέσω HTTP αιτήσεων στην πλατφόρμα ThingSpeak, η οποία λειτουργεί ως κεντρικό αποθετήριο για τα δεδομένα IoT. Η αποστολή περιλαμβάνει:

- Το μοναδικό API Key για την αυθεντικοποίηση.
- Τα πεδία δεδομένων που αντιπροσωπεύουν τις μετρήσεις (θερμοκρασία και υγρασία).

Η χρήση του ThingSpeak προσφέρει ένα αξιόπιστο και ασφαλές μέσο αποθήκευσης και διαχείρισης των δεδομένων σε πραγματικό χρόνο.

ThingSpeak (Αποθήκευση Δεδομένων)

Μόλις τα δεδομένα φτάσουν στο ThingSpeak, αποθηκεύονται σε ειδικά κανάλια οργανωμένα ανά πεδίο μέτρησης. Η πλατφόρμα:

- Παρέχει οπτικοποίηση των δεδομένων μέσω γραφημάτων.
- Επιτρέπει την ανάκτηση των δεδομένων μέσω API.
- Υποστηρίζει περαιτέρω ανάλυση μέσω MATLAB scripts.

Ανάκτηση Δεδομένων από Client (HTTP GET)

Ο client (η εφαρμογή Flask με Bootstrap) ανακτά τα δεδομένα από το ThingSpeak χρησιμοποιώντας HTTP GET αιτήσεις. Η διαδικασία περιλαμβάνει:

Ανάκτηση των πιο πρόσφατων μετρήσεων από το API του ThingSpeak.

Επεξεργασία των δεδομένων στο client-side για προετοιμασία απεικόνισης.

Η ευκολία ανάκτησης δεδομένων μέσω API καθιστά τη διαδικασία αποδοτική και απλή.

Προβολή Δεδομένων (Πίνακες και Γραφήματα)

Τα δεδομένα που ανακτώνται παρουσιάζονται στον τελικό χρήστη μέσω:

- Πινάκων (DataTables): Παρέχουν οργανωμένη και εύχρηστη παρουσίαση των δεδομένων, επιτρέποντας αναζήτηση, ταξινόμηση και σελιδοποίηση.
- Γραφημάτων (Google Charts): Οπτικοποιούν την εξέλιξη των μετρήσεων με ευδιάκριτο και κατανοητό τρόπο.

Αυτή η προσέγγιση βελτιώνει την αναγνωσιμότητα και τη χρηστικότητα της εφαρμογής για τον τελικό χρήστη.

Χρήστης (Αλληλεπίδραση με Εφαρμογή)

Ο χρήστης αλληλεπιδρά με την εφαρμογή μέσω ενός φιλικού περιβάλλοντος χρήστη, το οποίο επιτρέπει:

- Την παρακολούθηση των μετρήσεων σε πραγματικό χρόνο.
- Την αναζήτηση και φιλτράρισμα των δεδομένων μέσω των πινάκων.
- Την κατανόηση των τάσεων και μεταβολών των μετρήσεων μέσω των γραφημάτων.

Η προσέγγιση αυτή διασφαλίζει μια ευχάριστη εμπειρία για τον χρήστη, παρέχοντας άμεση πρόσβαση στις πληροφορίες που τον ενδιαφέρουν.

Περιγραφή Ροής Δεδομένων ESP32

Η διαδικασία συλλογής και αποστολής δεδομένων από το ESP32 στην πλατφόρμα ThingSpeak περιγράφεται στην Εικόνα 4.2 και περιλαμβάνει τα εξής στάδια:

1. **Ανάγνωση Δεδομένων από τον Αισθητήρα DHT11**

   Ο αισθητήρας DHT11 είναι υπεύθυνος για τη μέτρηση της θερμοκρασίας και της υγρασίας στον περιβάλλοντα χώρο. Το ESP32 διαβάζει αυτά τα δεδομένα μέσω ενός GPIO pin.

2. **Επεξεργασία Δεδομένων στο ESP32**

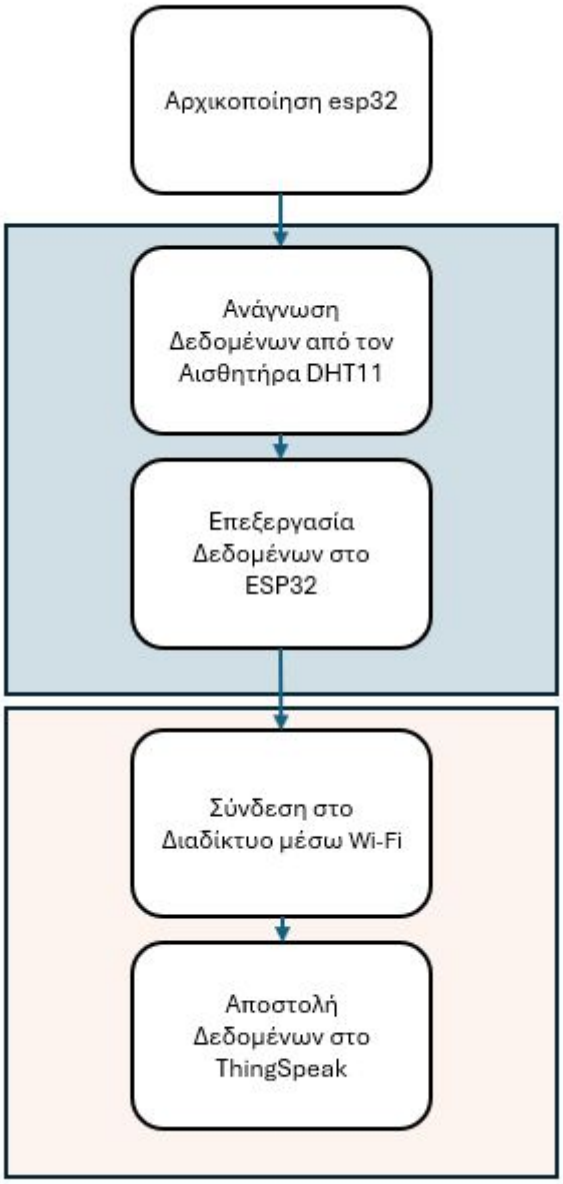
   Μετά την ανάγνωση, τα δεδομένα επεξεργάζονται εσωτερικά στο ESP32 για να εξασφαλιστεί η ακρίβεια και η ορθότητα των μετρήσεων. Το ESP32 διασφαλίζει ότι τα δεδομένα είναι έτοιμα για αποστολή.

3. **Σύνδεση στο Διαδίκτυο μέσω Wi-Fi**

   Το ESP32 αξιοποιεί την ενσωματωμένη του υποστήριξη Wi-Fi για να συνδεθεί στο τοπικό δίκτυο και, κατ' επέκταση, στο διαδίκτυο. Η σύνδεση αυτή είναι απαραίτητη για την αποστολή δεδομένων στην πλατφόρμα ThingSpeak.

4. **Αποστολή Δεδομένων στο ThingSpeak**

   Τα δεδομένα θερμοκρασίας και υγρασίας αποστέλλονται στο ThingSpeak μέσω HTTP POST αιτήσεων. Κάθε αίτηση περιλαμβάνει ένα μοναδικό API Key που διασφαλίζει την αυθεντικότητα και την ασφάλεια της αποστολής.



Εικόνα 4.2: Περιγραφή Ροής Δεδομένων ESP32

4.2 Περιγραφή του συστήματος μέσω κώδικα

4.2.1 Από την πλευρά του esp32

Παρακάτω παρουσιάζεται η αναλυτική επεξήγηση του κώδικα που χρησιμοποιείται στο ESP32 για τη συλλογή δεδομένων από τον αισθητήρα DHT11 και την αποστολή τους στην πλατφόρμα ThingSpeak.

Εισαγωγή Βιβλιοθηκών και Ορισμός Μεταβλητών

```
#include <WiFi.h>      // Βιβλιοθήκη για τη σύνδεση Wi-Fi
#include "DHT.h"       // Βιβλιοθήκη για την επικοινωνία με τον αισθητήρα DHT11
#include <HTTPClient.h> // Βιβλιοθήκη για αποστολή HTTP αιτήσεων

#define DHTPIN 4         // Ορισμός του GPIO pin όπου συνδέεται ο αισθητήρας DHT11
#define DHTTYPE DHT11    // Ορισμός του τύπου του αισθητήρα (DHT11)

DHT dht(DHTPIN, DHTTYPE); // Δημιουργία αντικειμένου DHT για την ανάγνωση δεδομένων

// Πληροφορίες σύνδεσης Wi-Fi
const char* ssid = "mokalis";
const char* password = "mok1234";

// Πληροφορίες σύνδεσης με το ThingSpeak
String apiKey = "KWJMLH3JEMD4LPN1";
const char* server = "api.thingspeak.com";
```

Επεξήγηση

- Οι βιβλιοθήκες WiFi.h, DHT.h, και HTTPClient.h εισάγονται για τη διαχείριση της σύνδεσης στο Wi-Fi, της επικοινωνίας με τον αισθητήρα και την αποστολή HTTP αιτήσεων αντίστοιχα.
- Το DHTPIN δηλώνει το GPIO pin στο οποίο είναι συνδεδεμένος ο αισθητήρας.
- Το DHTTYPE καθορίζει τον τύπο του αισθητήρα, που σε αυτή την περίπτωση είναι DHT11.
- Ορίζονται τα στοιχεία του Wi-Fi και οι πληροφορίες σύνδεσης με το ThingSpeak.

Ρύθμιση Αρχικής Σύνδεσης (setup)

```
void setup() {
  Serial.begin(115200); // Εκκίνηση της σειριακής επικοινωνίας για αποσφαλμάτωση
```

```
  WiFi.begin(ssid, password); // Έναρξη σύνδεσης στο Wi-Fi


  Serial.print("Connecting to WiFi...");
  while (WiFi.status() != WL_CONNECTED) { // Αναμονή μέχρι να ολοκληρωθεί η σύνδεση
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected.");
  dht.begin(); // Εκκίνηση του αισθητήρα DHT
}
```

Επεξήγηση

- Η συνάρτηση setup() εκτελείται μία φορά κατά την εκκίνηση της συσκευής.
- Ξεκινάει η σειριακή επικοινωνία με ταχύτητα 115200 baud για την εμφάνιση μηνυμάτων στο Serial Monitor.
- Πραγματοποιείται σύνδεση στο δίκτυο Wi-Fi και συνεχίζει την προσπάθεια μέχρι η σύνδεση να είναι επιτυχής.
- Ο αισθητήρας DHT εκκινείται με την dht.begin().

Κύριος Βρόχος (loop)

```
void loop() {
  float humidity = dht.readHumidity(); // Ανάγνωση υγρασίας
  float temperature = dht.readTemperature(); // Ανάγνωση θερμοκρασίας


  if (isnan(humidity) || isnan(temperature)) { // Έλεγχος εγκυρότητας μετρήσεων
    Serial.println("Αποτυχία ανάγνωσης από τον αισθητήρα DHT!");
    delay(2000);
    return;
```

```
}


Serial.printf("Θερμοκρασία: %.2f °C, Υγρασία: %.2f %%\n", temperature, humidity);
```

Επεξήγηση

- Οι τιμές θερμοκρασίας και υγρασίας ανακτώνται από τον αισθητήρα.
- Ελέγχεται αν οι τιμές είναι NaN (μη έγκυρες). Αν είναι, εμφανίζεται μήνυμα σφάλματος.
- Οι τιμές εκτυπώνονται στο Serial Monitor για παρακολούθηση.

Αποστολή Δεδομένων στο ThingSpeak

```
if (WiFi.status() == WL_CONNECTED) { // Έλεγχος σύνδεσης Wi-Fi
  HTTPClient http; // Δημιουργία αντικειμένου HTTPClient


  String url = "http://";
  url += server;
  url += "/update?api_key=";
  url += apiKey;
  url += "&field1=";
  url += String(temperature);
  url += "&field2=";
  url += String(humidity);


  http.begin(url); // Έναρξη σύνδεσης με το URL
  int httpResponseCode = http.GET(); // Αποστολή GET αιτήματος


  if (httpResponseCode > 0) {
    Serial.print("Αποστολή στο ThingSpeak επιτυχής, κωδικός: ");
    Serial.println(httpResponseCode);
```

```
  } else {
    Serial.print("Αποτυχία αποστολής, σφάλμα: ");
    Serial.println(httpResponseCode);
  }
  http.end(); // Τερματισμός σύνδεσης HTTP
}
```

Επεξήγηση

- Ελέγχεται αν η συσκευή είναι συνδεδεμένη στο Wi-Fi.
- Δημιουργείται το URL για αποστολή των δεδομένων στο ThingSpeak μέσω HTTP GET αιτήματος.
- Αποστέλλεται το αίτημα και ελέγχεται ο κωδικός απόκρισης.
- Αν το αίτημα είναι επιτυχές, εμφανίζεται μήνυμα επιτυχίας. Διαφορετικά, εμφανίζεται μήνυμα σφάλματος.
- Η σύνδεση HTTP κλείνει με την http.end().

Χρονική Καθυστέρηση

```
  delay(20000); // Καθυστέρηση 20 δευτερολέπτων πριν την επόμενη αποστολή
}
```

Επεξήγηση

- Ορίζεται καθυστέρηση 20 δευτερολέπτων μεταξύ των αποστολών δεδομένων, ώστε να τηρούνται τα όρια του ThingSpeak (15 δευτερόλεπτα ελάχιστο διάστημα).

Ο παραπάνω κώδικας παρέχει μια αξιόπιστη και αποδοτική λύση για την παρακολούθηση και αποστολή δεδομένων θερμοκρασίας και υγρασίας από έναν αισθητήρα DHT11 στο ThingSpeak μέσω ESP32. Με τη χρήση βασικών βιβλιοθηκών και τη σωστή διαχείριση των συνδέσεων, διασφαλίζεται η σταθερή λειτουργία και η ακρίβεια των μετρήσεων.

4.2.2 Από την μεριά του Python Κώδικα για λήψη δεδομένων από το ThingSpeak

Αναλύεται ο κώδικας Python που χρησιμοποιεί το Flask framework για την ανάκτηση και παρουσίαση των δεδομένων από την πλατφόρμα ThingSpeak.

Εισαγωγή Βιβλιοθηκών και Ρύθμιση Flask

```
from flask import Flask, render_template  # Βιβλιοθήκη Flask για ανάπτυξη web εφαρμογών
import requests                           # Βιβλιοθήκη για αποστολή HTTP αιτήσεων
from datetime import datetime             # Βιβλιοθήκη για χειρισμό ημερομηνιών και χρόνου


app = Flask(__name__)                     # Δημιουργία Flask εφαρμογής


# Στοιχεία σύνδεσης με το ThingSpeak
CHANNEL_ID = ****
READ_API_KEY = '***'
```

Επεξήγηση

- Η βιβλιοθήκη Flask χρησιμοποιείται για την ανάπτυξη της web εφαρμογής.
- Η requests χρησιμοποιείται για την αποστολή αιτήσεων HTTP προς το ThingSpeak API.
- Η datetime βοηθά στη μορφοποίηση και διαχείριση των χρονικών δεδομένων.
- Το app = Flask(__name__) δηλώνει τη δημιουργία της Flask εφαρμογής.
- Καθορίζονται το CHANNEL_ID και το READ_API_KEY για την πρόσβαση στο κανάλι του ThingSpeak.

Δημιουργία της Διαδρομής (Route) και Ανάκτηση Δεδομένων

```
@app.route('/')
def index():
    url = f'https://api.thingspeak.com/channels/{CHANNEL_ID}/feeds.json?api_key={READ_API_KEY}&results=20'
```

```
response = requests.get(url)        # Αποστολή GET αιτήματος προς το ThingSpeak
data = response.json()              # Μετατροπή της απάντησης σε μορφή JSON
```

Επεξήγηση

- Η @app.route('/') δηλώνει τη διαδρομή για την κεντρική σελίδα της εφαρμογής.
- Δημιουργείται το URL για την ανάκτηση των τελευταίων 20 εγγραφών από το ThingSpeak.
- Η requests.get(url) αποστέλλει την αίτηση, ενώ το response.json() μετατρέπει την απάντηση σε μορφή JSON.

Επεξεργασία των Δεδομένων

```
feeds = data['feeds']               # Ανάκτηση της λίστας των feeds
data_table = []                     # Δημιουργία λίστας για την αποθήκευση των δεδομένων


for feed in feeds:
    time_raw = feed['created_at']     # Ανάκτηση αρχικής ημερομηνίας
    time = datetime.strptime(time_raw, "%Y-%m-%dT%H:%M:%SZ").strftime("%d-%m-%Y %H:%M:%S")
    temperature = float(feed['field1']) if feed['field1'] else None
    humidity = float(feed['field2']) if feed['field2'] else None
    data_table.append((time, temperature, humidity))
```

Επεξήγηση

- Τα δεδομένα ανακτώνται από το πεδίο 'feeds' της JSON απάντησης.
- Ορίζεται η λίστα data_table για την αποθήκευση των δεδομένων σε μορφή πινάκων.
- Για κάθε εγγραφή (feed):
    - Το created_at μετατρέπεται σε μορφή dd-mm-yyyy HH:MM:SS με τη βοήθεια της datetime.
    - Οι τιμές θερμοκρασίας (field1) και υγρασίας (field2) μετατρέπονται σε float εάν υπάρχουν.

- Κάθε σύνολο δεδομένων (χρόνος, θερμοκρασία, υγρασία) αποθηκεύεται ως tuple στη λίστα data_table.

Απόδοση του Template με τα Δεδομένα

```
return render_template('index.html', data_table=data_table)
```

Επεξήγηση

- Η συνάρτηση render_template() αποδίδει το index.html template, μεταφέροντας τα δεδομένα μέσω της μεταβλητής data_table.

Εκκίνηση της Εφαρμογής

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

Επεξήγηση

- Το if __name__ == '__main__': διασφαλίζει ότι η εφαρμογή Flask θα εκκινήσει μόνο αν εκτελεστεί απευθείας.
- Η app.run(debug=True) εκκινεί την εφαρμογή με ενεργοποιημένο το debug mode, το οποίο είναι χρήσιμο για την εντόπιση σφαλμάτων κατά την ανάπτυξη.

Ο παραπάνω κώδικας παρέχει μια ολοκληρωμένη λύση για την ανάκτηση, επεξεργασία και παρουσίαση δεδομένων θερμοκρασίας και υγρασίας από την πλατφόρμα ThingSpeak. Μέσω της χρήσης της βιβλιοθήκης Flask, διασφαλίζεται η εύκολη και αποτελεσματική δημιουργία web εφαρμογής, με οργανωμένη δομή και ευέλικτη παρουσίαση των δεδομένων.

4.2.3 Παρουσίαση αποτελεσμάτων

Παρακάτω αναλύεται ο HTML κώδικας που χρησιμοποιείται για την παρουσίαση των δεδομένων θερμοκρασίας και υγρασίας που ανακτώνται από το ThingSpeak.

Δομή και Εισαγωγή Βιβλιοθηκών

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>IoT Dashboard - ThingSpeak</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
  <script src="https://www.gstatic.com/charts/loader.js"></script>
  <link rel="stylesheet" type="text/css" href="https://cdn.datatables.net/1.11.3/css/jquery.dataTables.min.css">
  <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
  <script src="https://cdn.datatables.net/1.11.3/js/jquery.dataTables.min.js"></script>
</head>
```

Επεξήγηση

- **Bootstrap**: Εισάγεται για την ενίσχυση της εμφάνισης και τη δημιουργία responsive σχεδίασης.
- **Google Charts**: Χρησιμοποιείται για την απεικόνιση γραφημάτων (θερμοκρασίας και υγρασίας).
- **DataTables**: Βιβλιοθήκη για το γέμισμα των HTML πινάκων με λειτουργίες όπως ταξινόμηση, αναζήτηση και σελιδοποίηση.
- **jQuery**: Απαραίτητο για τη λειτουργικότητα των DataTables.

Κύριο Περιεχόμενο της Σελίδας

```
<div class="container">
  <h2 class="mb-4">IoT Dashboard - ESP32 & DHT11</h2>

  <div class="row mb-5">
    <div class="col-md-6">
```

```
        <div id="temp_chart" style="height: 400px; width: 100%;"></div>
    </div>
    <div class="col-md-6">
        <div id="hum_chart" style="height: 400px; width: 100%;"></div>
    </div>
</div>

<table id="dataTable" class="display" style="width:100%">
    <thead>
        <tr>
            <th>Time</th>
            <th>Temperature (°C)</th>
            <th>Humidity (%)</th>
        </tr>
    </thead>
    <tbody>
        {% for time, temp, hum in data_table %}
            <tr>
                <td>{{ time }}</td>
                <td>{{ temp }}</td>
                <td>{{ hum }}</td>
            </tr>
        {% endfor %}
    </tbody>
</table>
</div>
```

Επεξήγηση

- **Χώρος Γραφημάτων**: Δημιουργούνται δύο διαγράμματα για θερμοκρασία και υγρασία χρησιμοποιώντας το div με τα id temp_chart και hum_chart.
- **Πίνακας DataTables**: Ορίζεται ένας πίνακας με τις στήλες Time, Temperature, και Humidity. Τα δεδομένα γεμίζονται δυναμικά μέσω της data_table λίστας από το Flask.

Ενεργοποίηση DataTables και Google Charts

```
$(document).ready(function() {
    $('#dataTable').DataTable();
});
```

Επεξήγηση

- Αρχικοποιείται το DataTable για τον πίνακα για λειτουργίες όπως ταξινόμηση και αναζήτηση.

Δημιουργία και Ρύθμιση Γραφημάτων

```
google.charts.load('current', {'packages':['corechart']});
google.charts.setOnLoadCallback(drawCharts);

function drawCharts() {
    // Temperature Chart
    var tempData = new google.visualization.DataTable();
    tempData.addColumn('string', 'Time');
    tempData.addColumn('number', 'Temperature (°C)');
    tempData.addRows([
        {% for time, temp, hum in data_table %}
            ['{{ time[11:] }}', {{ temp }}],
        {% endfor %}
    ]);

    var tempOptions = {
```

```
    title: 'Temperature Over Time',
    curveType: 'function',
    legend: { position: 'bottom' },
    hAxis: {
        title: 'Time',
        format: 'HH:mm:ss',
        slantedText: true,
        slantedTextAngle: 90,
        maxAlternation: 1,
        textStyle: {
            fontSize: 10,
            color: '#000'
        }
    }
};

var tempChart = new google.visualization.LineChart(document.getElementById('temp_chart'));
tempChart.draw(tempData, tempOptions);

// Humidity Chart
var humData = new google.visualization.DataTable();
humData.addColumn('string', 'Time');
humData.addColumn('number', 'Humidity (%)');
humData.addRows([
    {% for time, temp, hum in data_table %}
        ['{{ time[11:] }}', {{ hum }}],
    {% endfor %}
]);
```

```
  var humOptions = {
    title: 'Humidity Over Time',
    curveType: 'function',
    legend: { position: 'bottom' },
    hAxis: {
      title: 'Time',
      format: 'HH:mm:ss',
      slantedText: true,
      slantedTextAngle: 90,
      maxAlternation: 1,
      textStyle: {
        fontSize: 10,
        color: '#000'
      }
    }
  };

  var humChart = new google.visualization.LineChart(document.getElementById('hum_chart'));
  humChart.draw(humData, humOptions);
}
```

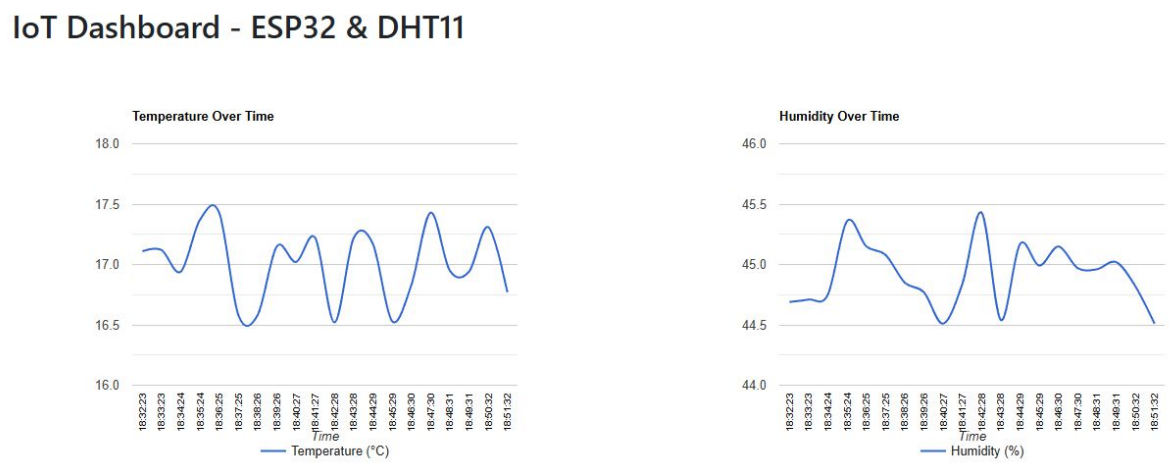
Επεξήγηση

- Οι πίνακες δεδομένων (DataTable) δημιουργούνται για κάθε γράφημα.
- Οι ετικέτες στον οριζόντιο άξονα (χρόνος) μορφοποιούνται μόνο για ώρα (HH:mm:ss).
- Ρυθμίσεις όπως slantedText και slantedTextAngle διασφαλίζουν κάθετη απεικόνιση για αποφυγή επικάλυψης.
- Η γραμματοσειρά των ετικετών μειώνεται για μεγαλύτερη ανάγνωση.

Ο παραπάνω HTML κώδικας συνδυάζει τις δυνατότητες των Bootstrap, Google Charts και DataTables για τη δημιουργία μιας λειτουργικής και αισθητικά ευχάριστης ιστοσελίδας. Οι χρήστες μπορούν να παρακολουθούν τις μετρήσεις θερμοκρασίας και υγρασίας σε πραγματικό χρόνο με δυναμική οπτικοποίηση και άμεση πρόσβαση στα δεδομένα μέσω του πίνακα DataTables.

4.3 Περιγραφή της σελίδας προβολής των δεδομένων

Τα γραφήματα στην Εικόνα 4.3 απεικονίζουν την πορεία της θερμοκρασίας και της υγρασίας σε βάθος χρόνου για μια άμεση οπτική αναπαράσταση των μετρήσεων.



Εικόνα 4.3: Διαγράμματα για προβολή των δεδομένων

- **Γράφημα Θερμοκρασίας (Temperature Over Time)**:
    - Στον οριζόντιο άξονα (X-axis) παρουσιάζεται ο χρόνος με τη μορφή HH:mm:ss.
    - Στον κάθετο άξονα (Y-axis) παρουσιάζεται η τιμή της θερμοκρασίας σε βαθμούς Κελσίου (°C).
    - Η καμπύλη γραμμής δείχνει τη μεταβολή της θερμοκρασίας μέσα στο συγκεκριμένο χρονικό διάστημα, υποδεικνύοντας τις αυξομειώσεις που καταγράφηκαν.

- **Γράφημα Υγρασίας (Humidity Over Time)**:

- Παρόμοια δομή με το γράφημα της θερμοκρασίας, με τον οριζόντιο άξονα να απεικονίζει τον χρόνο και τον κάθετο την τιμή της υγρασίας (%).
- Η καμπύλη απεικονίζει τις αυξομειώσεις της υγρασίας με την πάροδο του χρόνου, παρέχοντας πληροφορίες για την περιβαλλοντική μεταβλητότητα.

Σημαντικά Χαρακτηριστικά

- **Διαβάθμιση Ετικετών**: Οι ετικέτες χρόνου έχουν προσαρμοστεί ώστε να εμφανίζονται κάθετα, αποφεύγοντας την επικάλυψη και διατηρώντας τη σαφήνεια της πληροφορίας.
- **Οπτική Ευκρίνεια**: Η χρήση λεπτών γραμμών και καθαρών χρωμάτων συμβάλλει στη διακριτή απεικόνιση κάθε μεταβλητής.

  Μέσα από την παρατήρηση των γραφημάτων, μπορούμε να εντοπίσουμε τάσεις ή ανωμαλίες στις μετρήσεις, κάτι που είναι πολύτιμο για την ανάλυση των περιβαλλοντικών συνθηκών.

Ο πίνακας της Εικόνας 4.4. που εμφανίζεται στην εφαρμογή παρέχει μια οργανωμένη και λεπτομερή απεικόνιση των δεδομένων, επιτρέποντας την εύκολη ανάγνωση και αναζήτηση των πληροφοριών.

Show 10 entries Search:

Time	Temperature (°C)	Humidity (%)
16-03-2025 18:32:23	17.11	44.69
16-03-2025 18:33:23	17.12	44.71
16-03-2025 18:34:24	16.94	44.75
16-03-2025 18:35:24	17.37	45.36
16-03-2025 18:36:25	17.43	45.15
16-03-2025 18:37:25	16.58	45.08
16-03-2025 18:38:26	16.58	44.85

Εικόνα 4.4: Πίνακας με τα δεδομένα

Ανάλυση Δομής

- **Στήλη Time**: Παρουσιάζει τον ακριβή χρόνο καταγραφής κάθε μέτρησης με τη μορφή dd-mm-yyyy HH:mm:ss. Η ακριβής χρονική σήμανση επιτρέπει την αναλυτική εξέταση των μετρήσεων.
- **Στήλη Temperature (°C)**: Εμφανίζει τις μετρήσεις θερμοκρασίας με ακρίβεια δύο δεκαδικών ψηφίων, προσφέροντας σαφή εικόνα των μεταβολών.
- **Στήλη Humidity (%)**: Παρομοίως, απεικονίζει τις μετρήσεις υγρασίας με υψηλή ακρίβεια.
- **Δυνατότητες και Χαρακτηριστικά**

- **Δυναμική Ταξινόμηση**: Ο πίνακας επιτρέπει την ταξινόμηση των δεδομένων ανά στήλη, ώστε ο χρήστης να μπορεί να εντοπίσει άμεσα τις υψηλότερες ή χαμηλότερες τιμές.
- **Αναζήτηση Δεδομένων**: Το πεδίο αναζήτησης επιτρέπει την άμεση εύρεση συγκεκριμένων χρονικών στιγμών ή τιμών, διευκολύνοντας την εστίαση σε συγκεκριμένες μετρήσεις.
- **Σελιδοποίηση**: Ο πίνακας επιτρέπει την εμφάνιση συγκεκριμένου αριθμού εγγραφών ανά σελίδα, παρέχοντας μια πιο οργανωμένη προβολή μεγάλων συνόλων δεδομένων.

  Ο πίνακας παρέχει έναν άμεσο και αποτελεσματικό τρόπο παρουσίασης και ανάλυσης των δεδομένων. Η δυνατότητα φιλτραρίσματος και ταξινόμησης καθιστά εύκολη την εστίαση σε κρίσιμες πληροφορίες.

  Η συνδυασμένη παρουσίαση των δεδομένων σε πίνακα και γραφήματα επιτρέπει την πιο καλή παρακολούθηση των περιβαλλοντικών συνθηκών.

Κεφάλαιο 5ο: Συμπεράσματα - προτάσεις βελτίωσης

Παρουσιάστηκε μια ολοκληρωμένη λύση για την απομακρυσμένη παρακολούθηση περιβαλλοντικών συνθηκών εσωτερικού χώρου, με τη χρήση τεχνολογιών IoT και πλατφορμών ανάλυσης δεδομένων. Η εργασία επικεντρώθηκε στην ανάπτυξη ενός συστήματος που συνδυάζει το ESP32 με τον αισθητήρα DHT11 για τη συλλογή δεδομένων θερμοκρασίας και υγρασίας, και την αποστολή τους στην πλατφόρμα ThingSpeak. Στη συνέχεια μέσω μιας εφαρμογής Flask, τα δεδομένα ανακτώνται, αναλύονται και προβάλλονται με τη βοήθεια εργαλείων όπως τα Google Charts και DataTables.

Το σύστημα που αναπτύχθηκε είχε στόχο την αποτελεσματικότητά του στην αυτοματοποιημένη συλλογή και διαχείριση δεδομένων περιβαλλοντικών συνθηκών. Η ενσωμάτωση του ESP32 με τον αισθητήρα DHT11 και η αξιοποίηση της πλατφόρμας ThingSpeak προσέφεραν μια αξιόπιστη και ευέλικτη υποδομή αποστολής και αποθήκευσης δεδομένων. Επιπλέον, η χρήση της Python και του Flask για την ανάκτηση και προβολή των δεδομένων ενίσχυσε τη λειτουργικότητα του συστήματος, επιτρέποντας την εύκολη παρουσίαση και ανάλυση πληροφοριών.

Απλότητα και Ευκολία Εγκατάστασης:

- Η επιλογή του ESP32 και του αισθητήρα DHT11 αποδείχθηκε ιδανική λόγω της απλής συνδεσμολογίας και της ευκολίας προγραμματισμού.
- Η χρήση του ThingSpeak για την αποθήκευση δεδομένων απλοποίησε την υλοποίηση του backend συστήματος.

Αποτελεσματική Ανάλυση Δεδομένων:

- Η ενσωμάτωση των Google Charts επέτρεψε την άμεση και οπτικά κατανοητή απεικόνιση των δεδομένων θερμοκρασίας και υγρασίας.
- Το εργαλείο DataTables διευκόλυνε τη δυναμική ανάλυση, αναζήτηση και ταξινόμηση των δεδομένων.

Αξιοπιστία και Επεκτασιμότητα:

- Το σύστημα απέδειξε την αξιοπιστία του μέσω σταθερών μετρήσεων και συνεπούς αποστολής δεδομένων.
- Η αρχιτεκτονική που χρησιμοποιήθηκε μπορεί να επεκταθεί με την προσθήκη επιπλέον αισθητήρων ή τη διασύνδεση με άλλες πλατφόρμες.

Εκπαιδευτική και Ερευνητική Αξία:

- Η εργασία αυτή αποτελεί ένα χρήσιμο εργαλείο για την εκπαιδευτική διαδικασία, προσφέροντας ένα πρακτικό παράδειγμα υλοποίησης IoT συστημάτων.

- Παράλληλα, προσφέρει ένα σημείο εκκίνησης για περαιτέρω έρευνα πάνω σε συστήματα παρακολούθησης περιβαλλοντικών συνθηκών.

Κατά τη διάρκεια της υλοποίησης, αναδείχθηκαν ορισμένες προκλήσεις και περιορισμοί:

- **Ακρίβεια Μετρήσεων**: Ο αισθητήρας DHT11 παρουσιάζει περιορισμούς στην ακρίβεια των μετρήσεών του, γεγονός που μπορεί να επηρεάσει την αξιοπιστία των δεδομένων.
- **Χρονικοί Περιορισμοί Αποστολής**: Το ThingSpeak επιβάλλει περιορισμούς στη συχνότητα αποστολής δεδομένων (15 δευτερόλεπτα ανά εγγραφή), περιορίζοντας την ταχύτητα ενημέρωσης.
- **Ανάγκη για Σταθερή Σύνδεση**: Το σύστημα απαιτεί σταθερή σύνδεση στο διαδίκτυο για την απρόσκοπτη αποστολή και λήψη δεδομένων.

Κάποιες από τις προτάσεις για μελλοντική έρευνα είναι η αναβάθμιση αισθητήρων. Η χρήση πιο προηγμένων αισθητήρων (όπως DHT22 ή BME280) θα μπορούσε να βελτιώσει την ακρίβεια και το εύρος των μετρήσεων. Η ενσωμάτωση άλλων πλατφορμών ανάλυσης δεδομένων όπως το InfluxDB ή το Grafana, θα μπορούσε να προσφέρει περισσότερες δυνατότητες οπτικοποίησης. Η ανάπτυξη αλγορίθμων ανάλυσης και ανίχνευσης ανωμαλιών στα δεδομένα θα ενίσχυε την λειτουργικότητα του συστήματος. Η ενσωμάτωση μηχανισμών ασφάλειας στη μεταφορά δεδομένων όπως η χρήση κρυπτογραφημένων επικοινωνιών.Η προσθήκη λειτουργιών που θα επιτρέπουν την προσαρμογή των συσκευών ανάλογα με τις περιβαλλοντικές συνθήκες.

Η παρούσα εργασία αποτελεί ένα παράδειγμα ανάπτυξης συστήματος IoT για την παρακολούθηση περιβαλλοντικών συνθηκών. Για τη διόρθωση του συντακτικού του κειμένου χρησιμοποιήθηκαν TN εργαλεία. Από τη σωστή επιλογή τεχνολογιών και εργαλείων έγινε η δημιουργία ενός συστήματος που συνδυάζει απλότητα, αξιοπιστία και λειτουργικότητα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Hercog, D., Lerher, T., Truntič, M., & Težak, O. (2023). Design and implementation of ESP32-based IoT devices. Sensors, 23(15), 6739.

[2] Budijono, S. (2021, July). Smart temperature monitoring system using ESP32 and DS18B20. In IOP Conference Series: Earth and Environmental Science (Vol. 794, No. 1, p. 012125). IOP Publishing.

[3] Macheso, P., Chisale, S., Daka, C., Dzupire, N., Mlatho, J., & Mukanyirigira, D. (2021, March). Design of standalone asynchronous ESP32 web-server for temperature and humidity monitoring. In 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS) (Vol. 1, pp. 635-638). IEEE.

[4] Setyawan, G. C., & Setyawan, L. J. (2025). Model Sistem Pengering Tenaga Surya Berbasis IoT dengan ESP32 dan DHT-11. Progresif: Jurnal Ilmiah Komputer, 21(1).

[5] Myint, K. Z. Z., Aye, M., & Hla, T. T. (2024, November). An IoT-Based Real Time Temperature and Humidity Monitoring and Data Logging System Using ESP32. In 2024 5th International Conference on Advanced Information Technologies (ICAIT) (pp. 1-6). IEEE.

[6] Mubarakah, N., & Iddha, F. (2022, November). Prototype An ESP32-Based Room Humidity and Temperature Controller With IoT. In 2022 6th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM) (pp. 121-126). IEEE.

[7] https://www.hw-group.com/device/poseidon2-4002

[8] https://www.smartnet.gr/gr/efarmoges/thermokipia-parakoloythisi-kai-elegxos-tis-thermokrasias-kai-ygrasias/62-parakoloythisi-elegxos-thermokipion

[9] https://www.milesight.com/iot/#smart-agriculture

[10] https://www.espressif.com/en/products/socs/esp32

[11] https://www.nabto.com/guide-to-iot-esp-32/

[12] https://www.elprocus.com/a-brief-on-dht11-sensor/

[13] https://www.geeksforgeeks.org/what-is-python/

[14] https://aws.amazon.com/what-is/python/

[15] https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python

[16] https://www.geeksforgeeks.org/python-language-advantages-applications/

[17] https://www.mytaskpanel.com/what-is-flask/

[18] https://careerfoundry.com/en/blog/web-development/what-is-flask/

[19] https://dev.to/detimo/python-flask-pros-and-cons-1mlo

[20] https://uk.indeed.com/career-advice/career-development/what-is-bootstrap

[21] https://www.geeksforgeeks.org/what-are-the-benefits-of-implementing-the-bootstrap/

[22] https://owdt.com/article/the-pros-and-cons-of-using-bootstrap-for-front-end-development/
[23] https://developers.google.com/chart
[24] https://www.softwareadvice.com/bi/google-charts-profile/reviews/
[25] https://www.samgalope.dev/2023/10/13/datatables-pros-and-cons/
[26] https://thingspeak.mathworks.com/pages/commercial_learn_more
[27] https://www.zuehlke.com/en/insights/the-iot-platform-thingspeak-in-practical-test-0

ΠΑΡΑΡΤΗΜΑ Α

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

esp32

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

#define DHTPIN 4         // GPIO όπου συνδέεται το DHT11
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

// Πληροφορίες Wi-Fi
const char* ssid = "mokalis";
const char* password = "mok1234";

// ThingSpeak πληροφορίες
String apiKey = "KWJMLH3JEMD4LPN1";
const char* server = "api.thingspeak.com";

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);

  Serial.print("Connecting to WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected.");
  dht.begin();
}

void loop() {
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature();

  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Αποτυχία ανάγνωσης από τον αισθητήρα DHT!");
    delay(2000);
    return;
  }

  Serial.printf("Θερμοκρασία: %.2f °C, Υγρασία: %.2f %%\n", temperature, humidity);
```

```
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    String url = "http://";
    url += server;
    url += "/update?api_key=";
    url += apiKey;
    url += "&field1=";
    url += String(temperature);
    url += "&field2=";
    url += String(humidity);

    http.begin(url);
    int httpResponseCode = http.GET();

    if (httpResponseCode > 0) {
      Serial.print("Αποστολή στο ThingSpeak επιτυχής, κωδικός: ");
      Serial.println(httpResponseCode);
    } else {
      Serial.print("Αποτυχία αποστολής, σφάλμα: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  }

  delay(20000); // Στέλνει δεδομένα κάθε 20 δευτερόλεπτα (όριο του ThingSpeak: 15 sec)
}
```

Python Flask

```
from flask import Flask, render_template
import requests
from datetime import datetime


app = Flask(__name__)


# Αντικατάστησε με τις δικές σου πληροφορίες
CHANNEL_ID = ##
READ_API_KEY = '###'


@app.route('/')
def index():
    url                                                                                    =
f'https://api.thingspeak.com/channels/{CHANNEL_ID}/feeds.json?api_key={READ_API_KEY}&results=20'
```

```
    response = requests.get(url)
    data = response.json()

    feeds = data['feeds']
    data_table = []
    for feed in feeds:
        time_raw = feed['created_at']
        time = datetime.strptime(time_raw, "%Y-%m-%dT%H:%M:%SZ").strftime("%d-%m-%Y %H:%M:%S")
        temperature = float(feed['field1']) if feed['field1'] else None
        humidity = float(feed['field2']) if feed['field2'] else None
        data_table.append((time, temperature, humidity))

    return render_template('index.html', data_table=data_table)

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

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>IoT Dashboard - ThingSpeak</title>
    <link               href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <script src="https://www.gstatic.com/charts/loader.js"></script>
    <link                                rel="stylesheet"                                type="text/css"
href="https://cdn.datatables.net/1.11.3/css/jquery.dataTables.min.css">
    <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
    <script src="https://cdn.datatables.net/1.11.3/js/jquery.dataTables.min.js"></script>
</head>
<body class="p-4">

<div class="container">
    <h2 class="mb-4">IoT Dashboard - ESP32 & DHT11</h2>

    <div class="row mb-5">
        <div class="col-md-6">
            <div id="temp_chart" style="height: 400px; width: 100%;"></div>
        </div>
```

```
        <div class="col-md-6">
            <div id="hum_chart" style="height: 400px; width: 100%;"></div>
        </div>
    </div>

    <table id="dataTable" class="display" style="width:100%">
        <thead>
            <tr>
                <th>Time</th>
                <th>Temperature (°C)</th>
                <th>Humidity (%)</th>
            </tr>
        </thead>
        <tbody>
            {% for time, temp, hum in data_table %}
                <tr>
                    <td>{{ time }}</td>
                    <td>{{ temp }}</td>
                    <td>{{ hum }}</td>
                </tr>
            {% endfor %}
        </tbody>
    </table>
</div>

<script>
    $(document).ready(function() {
        $('#dataTable').DataTable();
    });

    google.charts.load('current', {'packages':['corechart']});
    google.charts.setOnLoadCallback(drawCharts);

    function drawCharts() {
        // Temperature Chart
        var tempData = new google.visualization.DataTable();
        tempData.addColumn('string', 'Time');
        tempData.addColumn('number', 'Temperature (°C)');
        tempData.addRows([
            {% for time, temp, hum in data_table %}
```

```
        ['{{ time[11:] }}', {{ temp }}],
    {% endfor %}
]);

var tempOptions = {
    title: 'Temperature Over Time',
    curveType: 'function',
    legend: { position: 'bottom' },
    hAxis: {
        title: 'Time',
        format: 'HH:mm:ss',
        slantedText: true,
        slantedTextAngle: 90,
        maxAlternation: 1,
        textStyle: {
            fontSize: 10,
            color: '#000'
        }
    }
};

var tempChart = new google.visualization.LineChart(document.getElementById('temp_chart'));
tempChart.draw(tempData, tempOptions);

// Humidity Chart
var humData = new google.visualization.DataTable();
humData.addColumn('string', 'Time');
humData.addColumn('number', 'Humidity (%)');
humData.addRows([
    {% for time, temp, hum in data_table %}
        ['{{ time[11:] }}', {{ hum }}],
    {% endfor %}
]);

var humOptions = {
    title: 'Humidity Over Time',
    curveType: 'function',
    legend: { position: 'bottom' },
    hAxis: {
        title: 'Time',
```

```
                format: 'HH:mm:ss',
                slantedText: true,
                slantedTextAngle: 90,
                maxAlternation: 1,
                textStyle: {
                    fontSize: 10,
                    color: '#000'
                }
            }
        };

        var humChart = new google.visualization.LineChart(document.getElementById('hum_chart'));
        humChart.draw(humData, humOptions);
    }
</script>

</body>
</html>
```