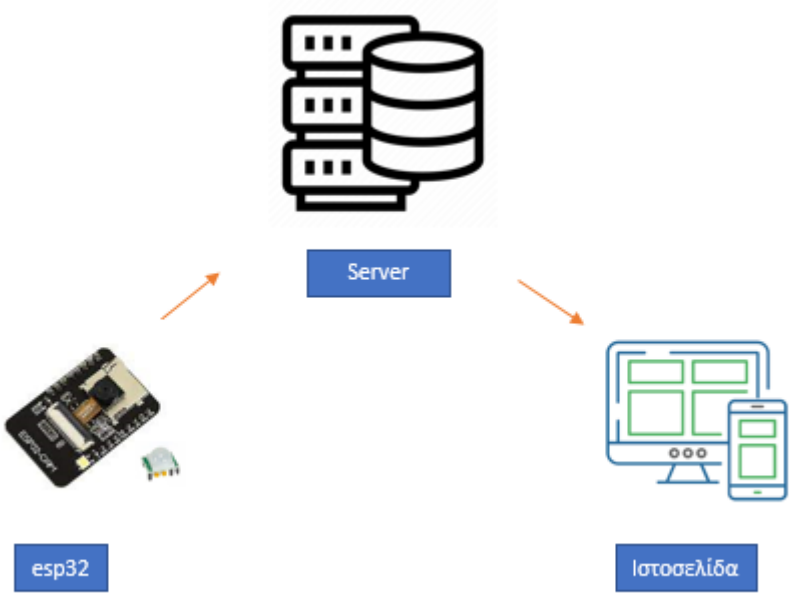
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Σύστημα παρακολούθησης χώρου με ανίχνευση κίνησης, κάμερα και ειδοποιήσεις»



Φοιτητής

Κλαίντι Κερόζι 516052

Επιβλέπων

Δρ. Κυριάκος Τσιακμάκης

Φεβρουάριος 2022

Φορητό σύστημα απομακρυσμένης επίβλεψης χώρου

Κωδικός: 21365

Φοιτητής: Κλαίντι Κερόζι

Εισηγητής: Δρ Κυριάκος Τσιακμάκης

Ημερομηνία ανάληψης Π.Ε. 15-10-2021

Ημερομηνία περάτωσης Π.Ε. 10-01-2022

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

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

Περίληψη

Η εργασία αυτή αφορά την κατασκευή ενός φορητού και φθηνού (κάτω απο 20 ευρώ) συστήματος παρακολούθησης που μπορεί να τοποθετηθεί εύκολα σε έναν χώρο και να ανιχνεύει και να αποστέλλει εικόνες και ειδοποίηση όταν υπάρχει κίνηση σε αυτόν. Με ένα esp32-cam και αισθητήρα ανίχνευσης να μπορεί ο χρήστης να το τοποθετεί στο δωμάτιο του ξενοδοχείου του ή σε κάποιον άλλο χώρο για να επιβλέπει παραβίαση του χώρου.

«Monitoring system with motion detection, camera and alerts»

Abstract

This work concerns the implementation of a portable and cheap (under 20 euros) monitoring system that can be easily placed in a space and detect motion and send images and alerts to the server. With an esp32-cam and detection sensor, the system can be placed at a hotel room or in another place to monitor the violation.

Ευχαριστίες

Να ευχαριστήσω τους γονείς μου για τη πολύτιμη συμπαράσταση τους.

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1ο: Εισαγωγή

1.1 Εισαγωγή

Οι κάμερες παρακολούθησης ή κάμερες ασφαλείας είναι βιντεοκάμερες που χρησιμοποιούνται για την παρατήρηση μιας περιοχής. Συχνά συνδέονται με μια συσκευή εγγραφής ή ένα δίκτυο IP και μπορεί να παρακολουθούνται από φύλακα ασφαλείας ή την αστυνομία. Οι κάμερες και ο εξοπλισμός εγγραφής είναι σχετικά ακριβοί και απαιτούν ανθρώπινο προσωπικό για την παρακολούθηση της κάμερας, αλλά η ανάλυση κάποιες φορές γίνεται ευκολότερη από κάποιο αυτοματοποιημένο λογισμικό που επεξεργάζεται το ψηφιακό βίντεο σε μια βάση δεδομένων με δυνατότητα αναζήτησης και λογισμικό ανάλυσης βίντεο. Το μέγεθος του βίντεο μειώνεται επίσης δραστικά από αισθητήρες κίνησης που καταγράφουν μόνο όταν ανιχνεύεται κίνηση. Με φθηνότερες τεχνικές παραγωγής, οι κάμερες παρακολούθησης είναι απλές και αρκετά φθηνές ώστε να χρησιμοποιούνται σε συστήματα οικιακής ασφάλειας και για καθημερινή παρακολούθηση. [1]

Οι κάμερες ασφαλείας στο σπίτι έχουν αλλάξει πολύ τα τελευταία χρόνια. Οι κάμερες ασφαλείας είναι πιο προσιτές από ό,τι στο παρελθόν, διατίθενται τόσο σε ενσύρματη όσο και σε ασύρματη έκδοση και μπορούν να συνδεθούν στο Διαδίκτυο, ώστε να είναι εφικτή η παρακολούθηση της ιδιοκτησία απομακρυσμένα.

Η οικιακή κάμερα ασφαλείας είναι μια συσκευή εγγραφής βίντεο που καταγράφει πλάνα του σπιτιού και της ιδιοκτησίας με προβολή σε smartphone, tablet ή υπολογιστή από οπουδήποτε χρησιμοποιώντας σύνδεση στο Διαδίκτυο. Κάποιες κάμερες ασφαλείας στο σπίτι ενεργοποιούνται με κίνηση και καταγράφουν όταν ανιχνεύουν κίνηση και κάποιες στέλνουν μια ειδοποίηση. Μερικές μπορούν να κάνουν εγγραφή 24/7, η οποία είναι γνωστή ως συνεχής εγγραφή βίντεο (CVR).

Υπάρχουν διάφοροι τρόποι με τους οποίους μπορεί να παρακολουθείται το υλικό που παράγεται από ένα σύστημα παρακολούθησης βίντεο. Η πιο παραδοσιακή και οικεία μέθοδος είναι να έχουμε έναν φύλακα ή ομάδα που είναι υπεύθυνη για την διαχείριση του βίντεο. Όταν επιλέγονται αναλογικά συστήματα/κάμερες χρησιμοποιούνται ομοαξονικά καλώδια για τη σύνδεση των καμερών με τα DVR και τις μονάδες προβολής τους.

Η συντριπτική πλειονότητα των σημερινών καμερών ασφαλείας είναι ψηφιακές κάμερες με ή όχι σύνδεση με το διαδίκτυο ώστε ο χρήστης να μπορεί να παρακολουθεί σε μια οθόνη τον χώρο. Επιπλέον, ορισμένα συστήματα και κάμερες έχουν τη δυνατότητα να μπορούν να ανιχνεύσουν κίνηση και, στη συνέχεια, να στείλουν ειδοποιήσεις μέσω κινητού σε εξουσιοδοτημένο προσωπικό που μπορεί στη συνέχεια να ελέγξει τη ζωντανή ροή.

Όταν μια κάμερα CCTV ανιχνεύσει κίνηση, θα αρχίσει να καταγράφει το υλικό, ώστε να μπορεί να ελεγχθεί αργότερα. Μπορεί να επιλεχθεί να καταγράφει το σύστημα κάθε δευτερόλεπτο πλάνα, αλλά αυτό καταλαμβάνει τεράστιο χώρο αποθήκευσης. Για συστήματα που χρησιμοποιούν αναλογικές κάμερες, το υλικό θα σταλεί από την κάμερα σε ένα DVR μέσω ομοαξονικού καλωδίου. [2]

Για συστήματα παρακολούθησης βίντεο που χρησιμοποιούν κάμερες IP χρησιμοποιείται NVR. Οι κάμερες και το NVR συνδέονται μέσω δρομολογητή δικτύου. Το εγγεγραμμένο υλικό κρυπτογραφείται και αποθηκεύεται σε σκληρό δίσκο και μπορεί να προβληθεί μέσω μιας συνδεδεμένης οθόνης ενός προγράμματος περιήγησης ιστού ή μιας εφαρμογής για κινητά.

Οι κάμερες παρακολούθησης βίντεο μπορούν πλέον να προσφέρουν αναγνώριση προσώπου, λέγονται έξυπνες κάμερες, όπως είναι οι κάμερες PTZ με έξυπνη παρακολούθηση που επιτρέπει να αναγνωρίζουν και να παρακολουθούν άτομα ή οχήματα μέχρι να είναι εκτός εμβέλειας. Υπάρχουν οι θερμικές κάμερες με νυχτερινή όραση, με πλήρη έγχρωμη υψηλή ευκρίνεια και μια ποικιλία από έξυπνες τεχνολογίες που επιτρέπουν στις κάμερες να στέλνουν άμεσες ειδοποιήσεις σχετικά με συγκεκριμένους τύπους δραστηριοτήτων.

Η εργασία αυτή αφορά την υλοποίηση φορητού συστήματος που θα ανιχνεύει κίνηση-παραβίαση με σε ένα χώρο, ένα φορητό και φθηνό (κάτω απο 20 ευρώ) σύστημα παρακολούθησης που μπορεί να τοποθετηθεί εύκολα σε έναν χώρο και να καταγράφει ή/και να αποστέλλει εικόνες/βίντεο όταν υπάρχει κίνηση σε αυτόν. Χρησιμοποιώντας ένα esp32-cam και έναν αισθητήρα PIR ο χρήστης μπορεί να το τοποθετήσει στο δωμάτιο που διανέμει για να ελέγξει τυχόν παραβίαση ή κίνηση σε αυτόν.

Θα αναλυθεί το πρόγραμμα που υλοποιήθηκε σε ένα ειδικού σκοπού με ενσωματωμένη κάμερα esp32 που τοποθετείται στο χώρο παρακολούθησης και στη συνέχεια θα περιγραφεί ο server που υλοποιήθηκε με python και η βάση MySQL με την οποία συνδέεται όπου αποθηκεύονται οι εικόνες και οι ειδοποιήσεις που λαμβάνει από το esp32. Επίσης, θα περιγραφεί η ιστοσελίδα στην οποία παρακολουθεί ο χρήστης τις ειδοποιήσεις-εικόνες που λαμβάνει από τον python server. Να σημειωθεί ότι η ιστοσελίδα υλοποιήθηκε με Flask και το πρόγραμμα στον esp32 με C.

1.2 Δομή της εργασίας

Στο πρώτο κεφάλαιο παρουσιάζεται μια μικρή εισαγωγή της εργασίας και οι στόχοι της εργασίας.

Στο δεύτερο κεφάλαιο περιγράφεται το σύστημα παρακολούθησης.

Στο τρίτο κεφάλαιο περιγράφεται τα εργαλεία και η τεχνολογία που χρησιμοποιήθηκαν για την υλοποίηση του έργου.

Στο τέταρτο κεφάλαιο αναλύονται ο server, η βάση και ο τρόπος απομακρυσμένης διαχείρισης μέσω ιστοσελίδας.

Στο πέμπτο κεφάλαιο παρουσιάζονται τα συμπεράσματα της εργασίας και οι προτάσεις βελτιώσεων. Στο τέλος της εργασίας παρατίθενται τα παραρτήματα με τους κώδικες που χρησιμοποιήθηκαν στην εργασία.

Κεφάλαιο 2ο: Το σύστημα παρακολούθησης

2.1 Εισαγωγή

Σήμερα τα συστήματα επιτήρησης αποτελούν σημαντικό μέρος της ασφάλειας του σπιτιού ή μιας επιχείρησής. Αυτά τα συστήματα κυμαίνονται από ασύρματες οικιακές κάμερες ασφαλείας έως εξελιγμένα συστήματα συναγερμού που ειδοποιούν τις αρχές του νόμου με την πρώτη ειδοποίηση. Η παρουσία καμερών ασφαλείας μπορεί να χρησιμεύσει ως αποτρεπτικός παράγοντας για τους επίδοξους κλέφτες, ενώ οι κρυφές κάμερες μπορούν να προστατεύουν διακριτικά. [3-4]

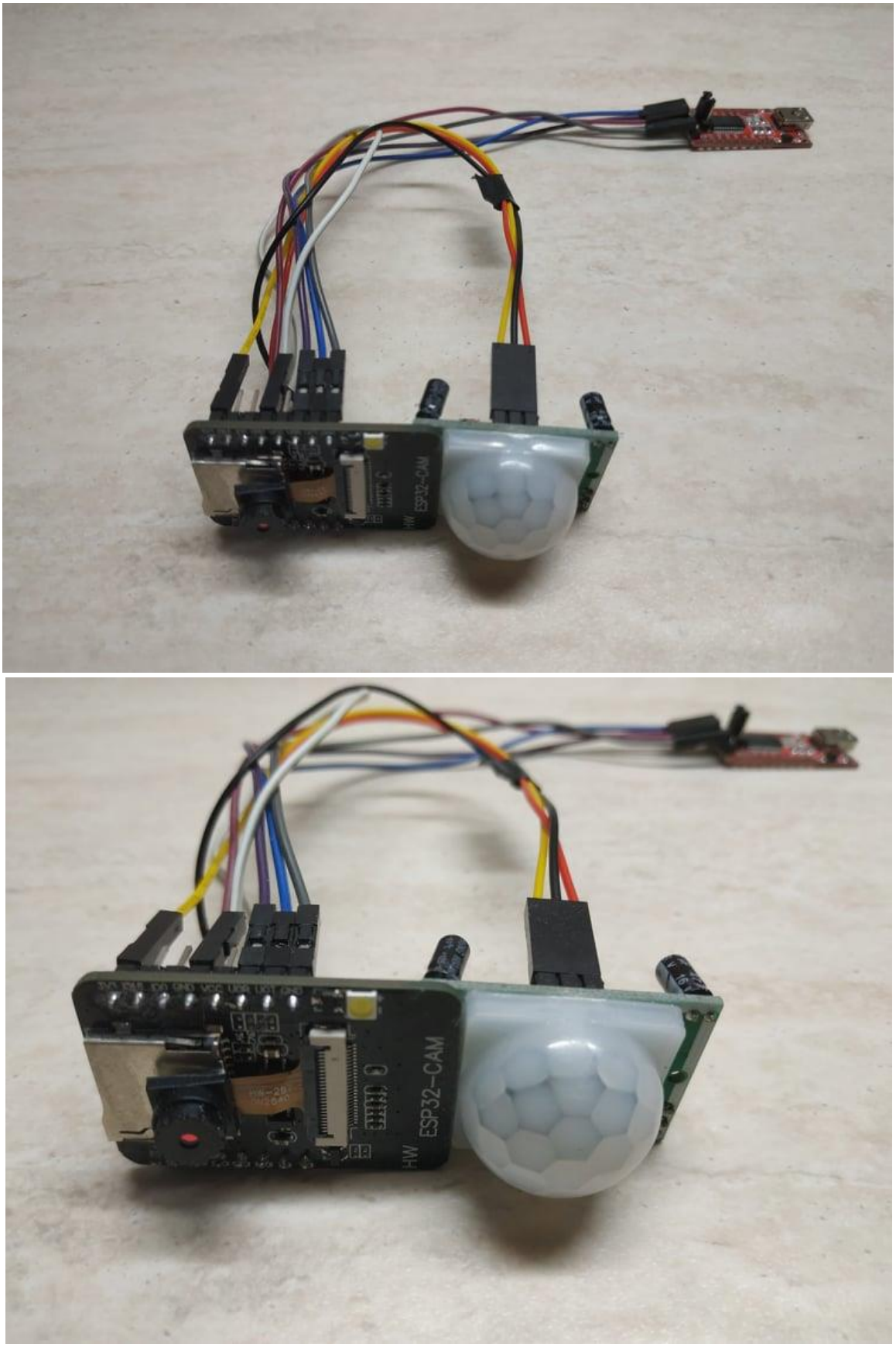
Είτε θέλουμε να παρακολουθήσουμε τους υπαλλήλους σε κοντινές εγκαταστάσεις είτε να τακτοποιήσουμε κάθε άτομο που πλησιάζει στην εξώπορτα του σπιτιού μας, ο κατάλληλος εξοπλισμός μπορεί να προσφέρει την απαραίτητη προστασία.

Η παρακολούθηση ενός δωματίου ή ενός χώρου μπορεί να γίνει μέσω μιας απλής κάμερας με καταγραφή σε sd κάρτα ή με ip κάμερα αν θέλουμε να παρακολουθούμε απομακρυσμένα μέσω διαδικτύου. Στη περίπτωση μας θέλουμε να παρακολουθήσουμε ένα δωμάτιο ενός ξενοδοχείου όταν λείπουμε ή έναν εργασιακό χώρο όταν λείπουμε. Με την λέξη ‘παρακολουθήσουμε’ εννοούμε την ανίχνευση κίνησης-παραβίασης στο χώρο και άμεση φωτογράφηση του χώρου και αποστολή στον server για αποθήκευση και προβολή.

Στην εργασία αυτή κατασκευάστηκε ένα σύστημα παρακολούθησης χώρου ανιχνεύοντας κίνηση με αισθητήρα PIR και στη συνέχεια στέλνει ειδοποίηση μέσω διαδικτύου σε server.

Η ανίχνευση κίνησης PIR πραγματοποιείται με ανίχνευση θερμότητας. Οι αισθητήρες κίνησης PIR (Passive Infrared) είναι σχεδιασμένοι για ασφάλεια και βελτιστοποιημένοι για να εντοπίζουν αξιόπιστα άτομα, μεγάλα κατοικίδια και άλλα μεγάλα θερμά κινούμενα αντικείμενα. Αυτό σημαίνει ότι η κάμερα γίνεται αρκετά έξυπνη ώστε να ανιχνεύει ανθρώπους και όχι φύλλα ή σκιές χωρίς να επηρεάζεται από αν είναι μέρα ή νύχτα..

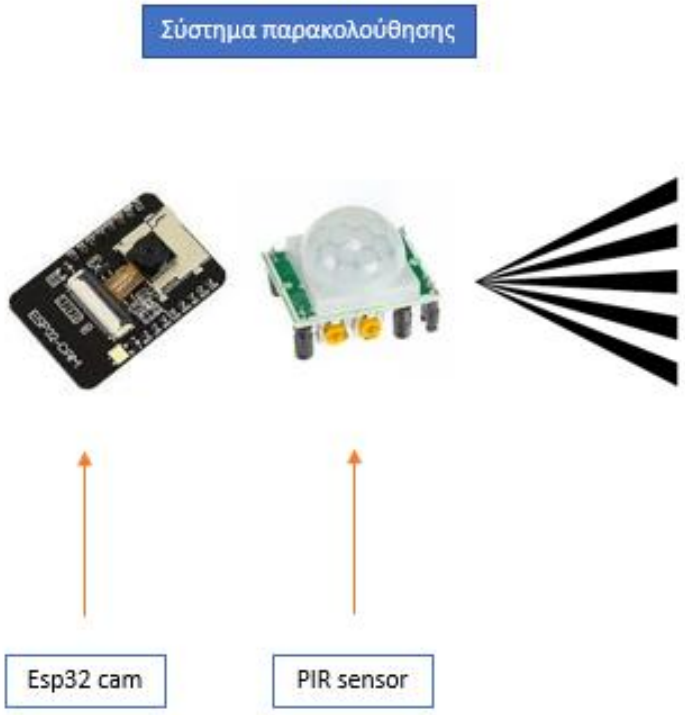
Το σύστημα που υλοποιήθηκε παρουσιάζεται στην Εικόνα 2.1.



Εικόνα 2.1: Σύστημα παρακολούθησης - Διάταξη

Η συσκευή-σύστημα τοποθετείται στο χώρο επίβλεψης (πχ ξενοδοχείο) με τον PIR αισθητήρα να επιβλέπει το πεδίο. Συνήθως εστιάζει σε μια είσοδο κάποιου δωματίου ή κάποιο παράθυρο.

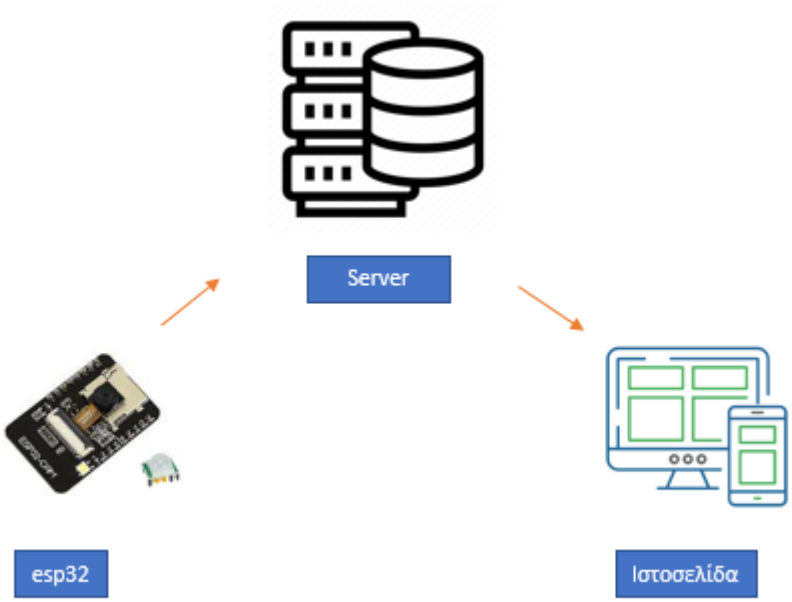
Στην Εικόνα 2.2. παρουσιάζεται πεδίο παρακολούθησης από τον PIR.



Εικόνα 2.2: Πεδίο παρακολούθησης από το PIR

Μόλις το PIR ανιχνεύσει κίνηση ενεργοποιηθεί την κάμερα στο esp32 και λαμβάνει μια φωτογραφία.

Στην Εικόνα 2.3 παρουσιάζεται το σύστημα ανίχνευσης - ειδοποίησης - αποστολής ειδοποίησης.



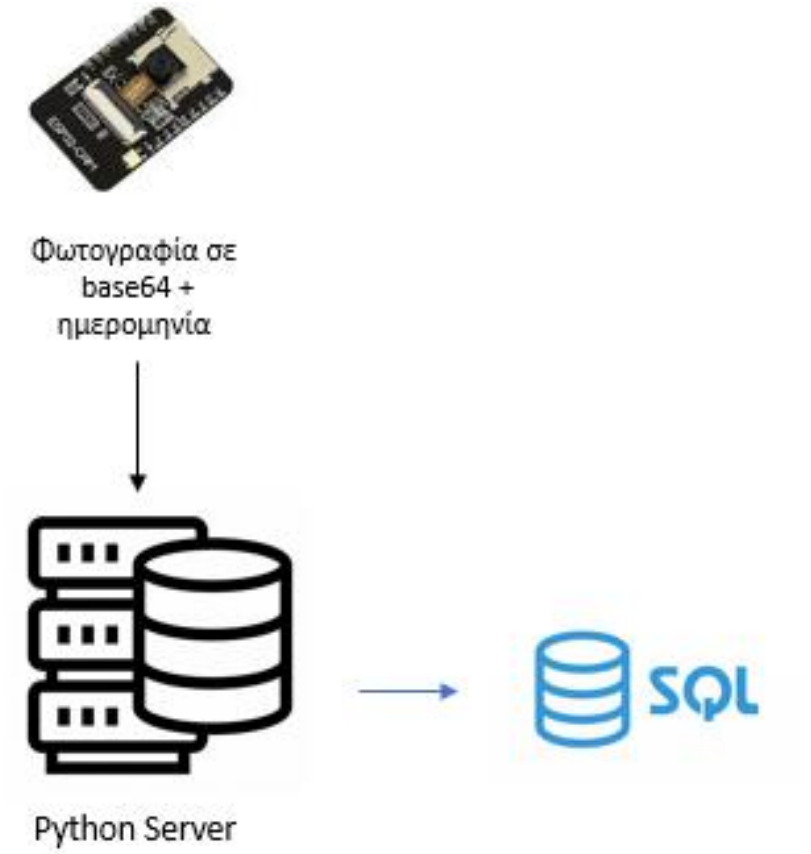
Εικόνα 2.3: Σύστημα ανίχνευσης - ειδοποίησης - αποστολής ειδοποίησης

Για την επικοινωνία του esp32 με τον server πρέπει να υπάρχει σύνδεση με το διαδίκτυο μέσω του Wi-Fi.

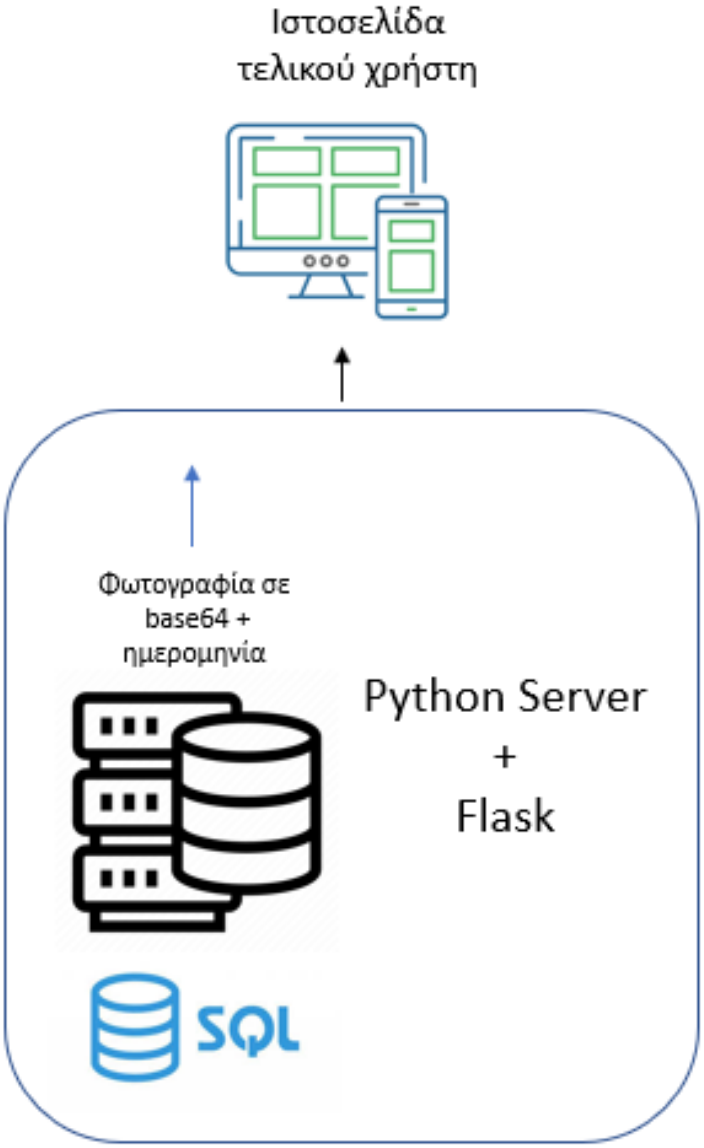
Στην Εικόνα 2.4 παρουσιάζεται η επικοινωνία του esp32 με το server. Όταν το esp32 μέσω του PIR ανιχνεύσει κίνηση στο πεδίο δίνει σήμα και το φωτογραφίζει και την αποστέλλει στο python server με την ειδοποίηση.

Η φωτογραφία μετατρέπεται σε base64 μορφή και ο server την αποθηκεύει σε βάση δεδομένων MySQL.

Η φωτογραφία είναι σε base64 δηλαδή ένα απλό κείμενο από χαρακτήρες και αυτό γίνεται για να είναι δυνατή η αποθήκευση και η ανάγνωση της από βάση δεδομένων.



Εικόνα 2.4: Επικοινωνία esp32 με τον python server – Αποστολή εικόνας σε base64 μορφή



Εικόνα 2.5: Διαχείριση και προβολή ειδοποιήσεων σε ιστοσελίδα

Στην Εικόνα 2.5 παρουσιάζεται ο τρόπος διαχείρισης και προβολής ειδοποιήσεων σε ιστοσελίδα που υποστηρίζεται από τον python server και το framework flask.

Κεφάλαιο 3ο: Εργαλεία και τεχνολογίες που χρησιμοποιήθηκαν

Σε αυτό το κεφάλαιο θα περιγραφούν ποια εργαλεία και συσκευές χρησιμοποιήθηκαν όσον αφορά το ηλεκτρονικό υλικό και ποιες τεχνολογίες και πλατφόρμες χρησιμοποιήθηκαν για το λογισμικό.

3.1 Esp32

Το ESP32 είναι ένας μικροελεγκτής χαμηλού κόστους και χαμηλής κατανάλωσης με ενσωματωμένο Wi-Fi και Bluetooth. Είναι ο διάδοχος του ESP8266 που είναι επίσης ένα μικροτσίπ Wi-Fi χαμηλού κόστους, αν και με περιορισμένη εξαιρετικά περιορισμένη λειτουργικότητα.

Έχει μια ενσωματωμένη κεραία, ενισχυτής ισχύος, ενισχυτές χαμηλού θορύβου, φίλτρα και μονάδα διαχείρισης ενέργειας. Η πλακέτα της καταλαμβάνει τη μικρότερη έκταση της πλακέτας τυπωμένου κυκλώματος. Αυτή η πλακέτα χρησιμοποιείται με τσιπ διπλής λειτουργίας Wi-Fi 2,4 GHz και Bluetooth από την τεχνολογία χαμηλής ισχύος TSMC 40nm, τις καλύτερες ιδιότητες ισχύος και ραδιοσυχνοτήτων, η οποία είναι ασφαλής, αξιόπιστη και μπορεί να χρησιμοποιηθεί σε διάφορες εφαρμογές.

Τα χαρακτηριστικά του ESP32 περιλαμβάνουν τα ακόλουθα:

Επεξεργαστής:

CPU: Μικροεπεξεργαστής Xtensa διπλού πυρήνα (ή μονοπύρηνος) 32-bit LX6, που λειτουργεί στα 160 ή 240 MHz και αποδίδει έως και 600 DMIPS

Συνεπεξεργαστής εξαιρετικά χαμηλής ισχύος (ULP).

Μνήμη: 320 KiB RAM, 448 KiB ROM

Ασύρματη συνδεσιμότητα:

Wi-Fi: 802.11 b/g/n

Bluetooth: v4.2 BR/EDR και BLE (μοιράζεται το ραδιόφωνο με Wi-Fi)

Περιφερειακές διεπαφές:

34 × προγραμματιζόμενα GPIO

12-bit SAR ADC έως 18 κανάλια

2 × 8-bit DAC

10 × αισθητήρες αφής (GPIO με χωρητική ανίχνευση)

4 × SPI

2 × διεπαφές I²S

2 × διεπαφές I²C

3 × UART

Αναλογικός προενισχυτής εξαιρετικά χαμηλής ισχύος

Ασφάλεια:

Υποστηρίζονται όλες οι τυπικές λειτουργίες ασφαλείας IEEE 802.11, συμπεριλαμβανομένων των WPA, WPA2, WPA3 (ανάλογα με την έκδοση)[5] και WAPI

Κρυπτογράφηση Flash

Ελεγκτής κεντρικού υπολογιστή SD/SDIO/CE-ATA/MMC/eMMC

Υποτελής ελεγκτής SDIO/SPI

Διεπαφή MAC Ethernet με αποκλειστικό DMA και προγραμματισμένη υποστήριξη πρωτοκόλλου χρόνου ακριβείας IEEE 1588

CAN bus 2.0

Υπέρυθρες (TX/RX, έως 8 κανάλια)

Μοτέρ PWM

LED PWM (έως 16 κανάλια)

Αισθητήρας εφέ Hall

Διαχείριση ενέργειας:

Εσωτερικός ρυθμιστής χαμηλής πτώσης

Μεμονωμένος τομέας ισχύος για RTC

Ρεύμα βαθέως ύπνου 5 μA

Αφύπνιση από διακοπή GPIO, χρονοδιακόπτη, μετρήσεις ADC, διακοπή χωρητικού αισθητήρα αφής

Το ESP32 υποστηρίζει τρεις τύπους λειτουργιών I/O με κάθε GPIO Pin: Ψηφιακούς, Αναλογικούς και Εσωτερικούς Αισθητήρες

Αναλογικά: Χρησιμοποιείται για την αποστολή/λήψη αναλογικών δεδομένων χρησιμοποιώντας τις ακόλουθες λειτουργίες:

analogRead();

analogWrite();

Ψηφιακά: Χρησιμοποιείται για την αποστολή/λήψη ψηφιακών δεδομένων χρησιμοποιώντας τις ακόλουθες λειτουργίες:

digitalRead();

digitalWrite();

Εσωτερικοί αισθητήρες: Αυτή η λειτουργία μας επιτρέπει να ανακτούμε δεδομένα εσωτερικού αισθητήρα από το ίδιο το ESP32.

Οι τρεις διαθέσιμοι αισθητήρες είναι οι εξής:

Internal Temperature Sensor

Hall Effect Sensor

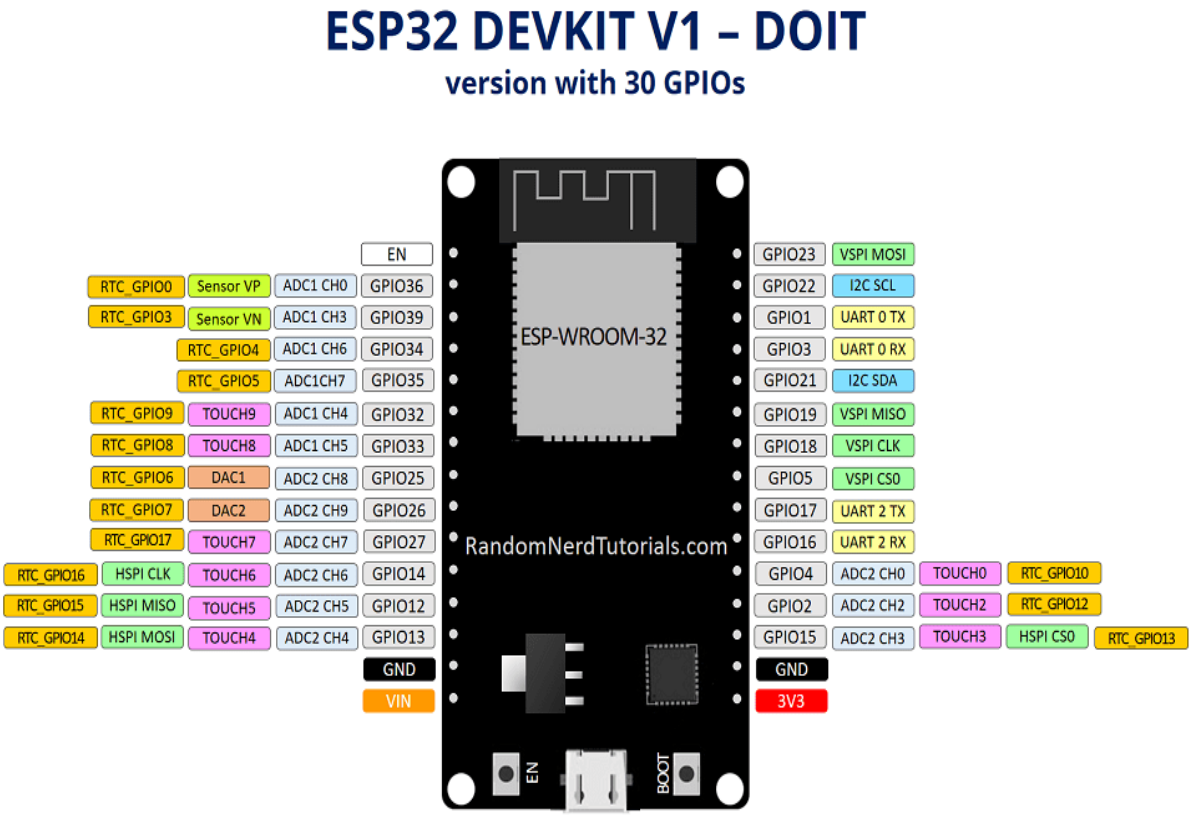
Touch Sensor

Η πρόσβαση σε αυτούς τους αισθητήρες είναι δυνατή με τις ακόλουθες λειτουργίες:

temprature_sens_read()

hallRead()

touchRead()



Εικόνα 3.1: Esp32 ακροδέκτες [4]

3.1.1 Esp32-cam

Το ESP32-CAM μπορεί να χρησιμοποιηθεί ευρέως σε διάφορες εφαρμογές IoT. Είναι κατάλληλο για οικιακές έξυπνες συσκευές, βιομηχανικό ασύρματο έλεγχο, ασύρματη παρακολούθηση, ασύρματη αναγνώριση για QR, ασύρματα σήματα συστήματος εντοπισμού θέσης και άλλες εφαρμογές ιδίως για IoT.

Το ESP32-CAM είναι μια πλακέτα ανάπτυξης WIFI+ bluetooth διπλής λειτουργίας που χρησιμοποιεί ενσωματωμένη κεραία PCB και πυρήνες που βασίζονται σε τσιπ ESP32. Μπορεί να λειτουργήσει ανεξάρτητα ως σύστημα.

Το ESP ενσωματώνει το WiFi, παραδοσιακό bluetooth και BLE Beacon, έχει 2 επεξεργαστές υψηλής απόδοσης 32-bit LX6, εύρος ρύθμισης κύριας συχνότητας 80MHz έως 240MHz, αισθητήρα on-chip, αισθητήρα Hall, αισθητήρα θερμοκρασίας και άλλους.

Πλήρως συμβατό με τα πρότυπα WiFi 802.11b/g/n/e/i και το bluetooth 4.2. Μπορεί να χρησιμοποιηθεί ως κύρια λειτουργία για τη δημιουργία ενός ανεξάρτητου ελεγκτή δικτύου ή ως slave σε άλλα MCU κεντρικού υπολογιστή για να προσθέσει δυνατότητες δικτύωσης σε υπάρχουσες συσκευές.

Το ESP32-CAM μπορεί να χρησιμοποιηθεί ευρέως σε διάφορες εφαρμογές IoT. Είναι κατάλληλο για οικιακές έξυπνες συσκευές, βιομηχανικό ασύρματο έλεγχο, ασύρματη παρακολούθηση και άλλα.

Μερικά από τα χαρακτηριστικά του παρουσιάζονται παρακάτω και στον Πίνακα 3.1:

- Μικρή μονάδα 802.11b/g/n Wi-Fi + BT/BLE SoC
- CPU διπλού πυρήνα 32-bit χαμηλής κατανάλωσης
- Έως 240 MHz, έως 600 DMIPS
- Ενσωματωμένη SRAM 520 KB, εξωτερικό PSRAM 4M
- Υποστηρίζει διεπαφές όπως UART/SPI/I2C/PWM/ADC/DAC
- Υποστήριξη καμερών OV2640 και OV7670 με ενσωματωμένο φλας
- Υποστήριξη κάρτας TF
- Υποστήριξη πολλαπλών λειτουργιών ύπνου
- Υποστήριξη λειτουργίας STA/AP/STA+AP
- Υποστήριξη δικτύου διανομής Smart Config/AirKiss
- Υποστήριξη για σειριακή τοπική αναβάθμιση και απομακρυσμένη αναβάθμιση υλικού - λογισμικού (FOTA)

Πίνακας 3.1

SPI Flash	32Mbit
RAM	Εσωτερική 520KB+εξωτερική 4M PSRAM
Serial port rate	Προεπιλεγμένη 115200 bps
Image output format	JPEG (only supported by OV2640), BMP, GRAYSCALE

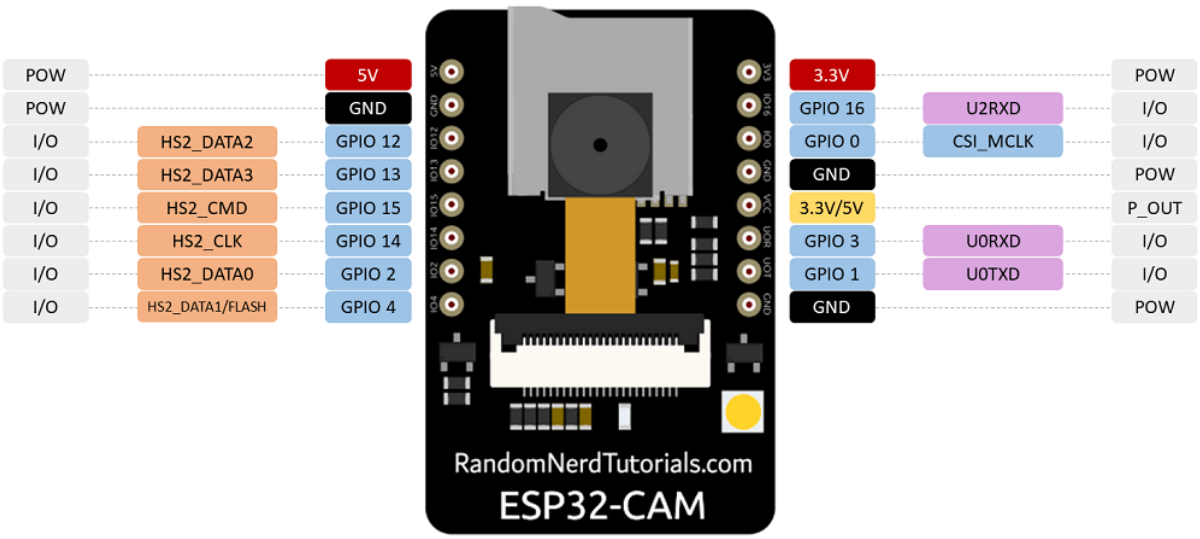
Power consumption	Χωρίς flash: 180mA@5V Με flash: 310mA@5V Sleep: 6mA-6.7mA @5V
Security	WPA/WPA2/WPA2-Enterprise/WPS
Power supply range	5V

Ο Πίνακας 3.2 παρουσιάζει πόσες εικόνες μπορεί να μεταδώσει σε ένα δευτερόλεπτο ανάλογα τα χαρακτηριστικά της εικόνας.

Πίνακας 3.2

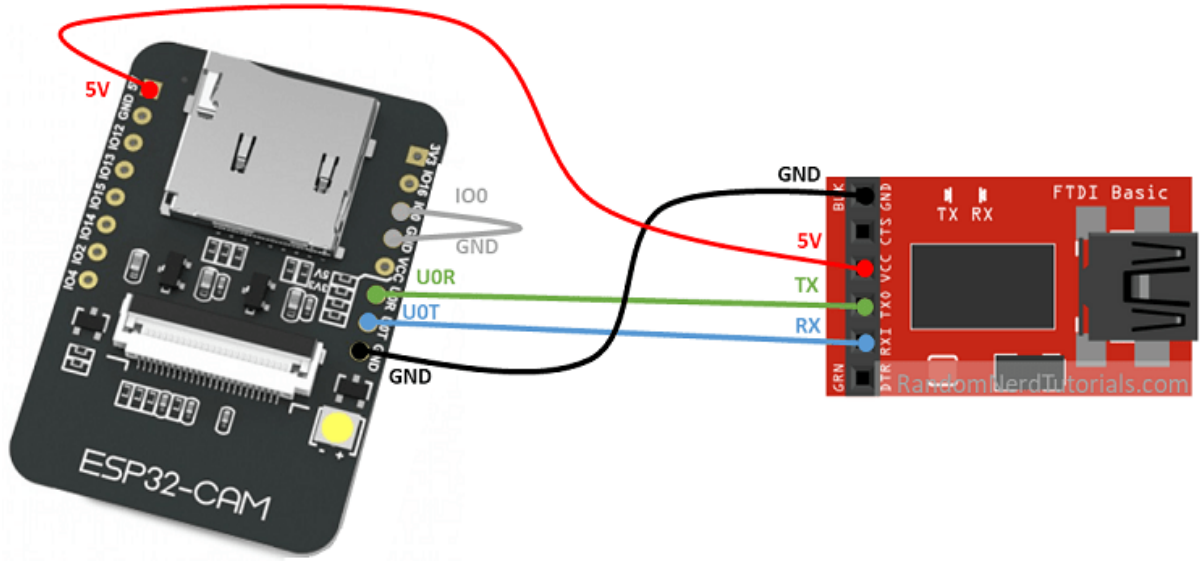
ESP32-CAM Picture Output Format Rate [6]

	QQVGA	QVGA	VGA	SVGA
JPEG	6	7	7	8
BMP	9	9	-	-
GRAYSCALE	9	8	-	-



Εικόνα 3.2: Esp32 -CAM ακροδέκτες [5]

Στην Εικόνα 3.2 παρουσιάζονται οι ακροδέκτες του esp32-cam που είναι προφανώς λιγότεροι διαθέσιμοι από το αντίστοιχο esp32.



Εικόνα 3.3: Σύνδεση Esp32 -CAM με FTDI [5]

Για τον προγραμματισμό/φόρτωση του προγράμματος του esp32-cam χρειάζεται μια ειδική συσκευή, το FTDI και στην Εικόνα 3.3 παρουσιάζεται ο τρόπος σύνδεσης.

3.1.2 Αισθητήρας PIR

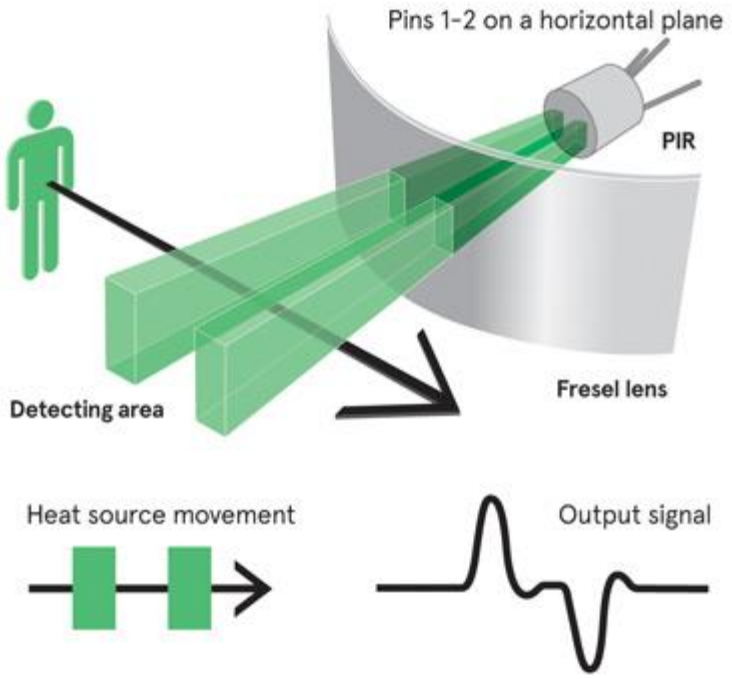
Ο αισθητήρας PIR είναι επίσης γνωστός ως παθητικός αισθητήρας υπερύθρων είναι ένας ηλεκτρονικός αισθητήρας.

Μετράει το υπέρυθρο φως που εκπέμπεται από αντικείμενα στο πεδίο..

Τα υπέρυθρα κύματα μπορούν να ανιχνεύσουν τη θερμοκρασία. Οι αισθητήρες κίνησης υπερύθρων ανιχνεύουν την παρουσία ενός ατόμου ή αντικειμένου ανιχνεύοντας την αλλαγή της θερμοκρασίας μιας δεδομένης περιοχής.

Ο αισθητήρας PIR έχει δύο υποδοχές, κάθε υποδοχή είναι κατασκευασμένη από ειδικό υλικό που είναι ευαίσθητη στο IR. Ο φακός που χρησιμοποιείται δεν κάνει πραγματικά πολλά και έτσι βλέπουμε ότι οι δύο υποδοχές μπορούν να «βλέπουν» πέρα από κάποια απόσταση (βασικά την ευαισθησία του αισθητήρα). Όταν ο αισθητήρας είναι αδρανής, και οι δύο υποδοχές ανιχνεύουν την ίδια ποσότητα

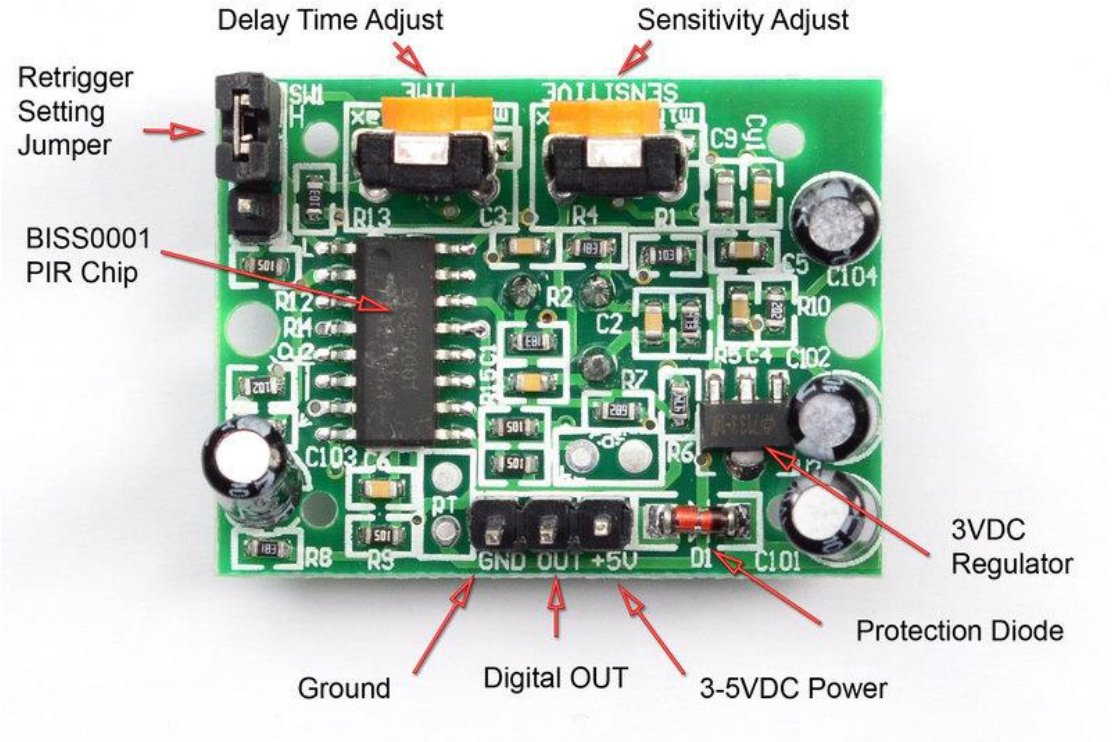
υπερύθρων, την περιβαλλοντική ποσότητα που εκπέμπεται από το δωμάτιο ή τους τοίχους ή εξωτερικούς χώρους. Όταν περνάει ένα ζεστό σώμα όπως ένας άνθρωπος ή ένα ζώο, ανακόπτει πρώτα το μισό του αισθητήρα PIR, το οποίο προκαλεί μια θετική αλλαγή διαφοράς μεταξύ των δύο μισών. Όταν το θερμό σώμα φεύγει από την περιοχή αίσθησης, συμβαίνει το αντίστροφο, οπότε ο αισθητήρας δημιουργεί μια αρνητική αλλαγή διαφορικού. Αυτοί οι παλμοί αλλαγής είναι αυτό που ανιχνεύεται, όπως φαίνεται στην Εικόνα 3.4.



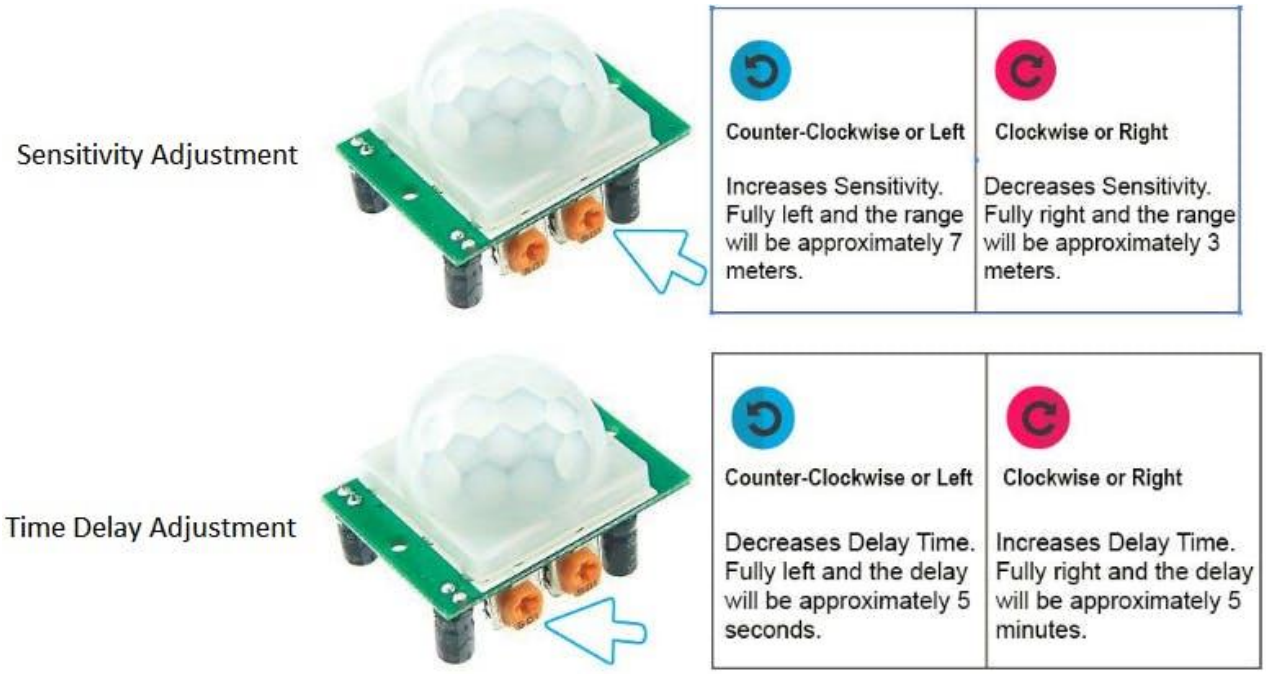
Εικόνα 3.4: Αρχή λειτουργίας PIR [7]

Στην Εικόνα 3.5 παρουσιάζεται ο αισθητήρας PIR που χρησιμοποιήθηκε στην εργασία και υπάρχει στην Ελληνική αγορά σε πολλά καταστήματα. Στην Εικόνα 3.6 παρουσιάζεται η πίσω όψη της πλακέτας PIR.

Εικόνα 3.5: Ο αισθητήρας PIR

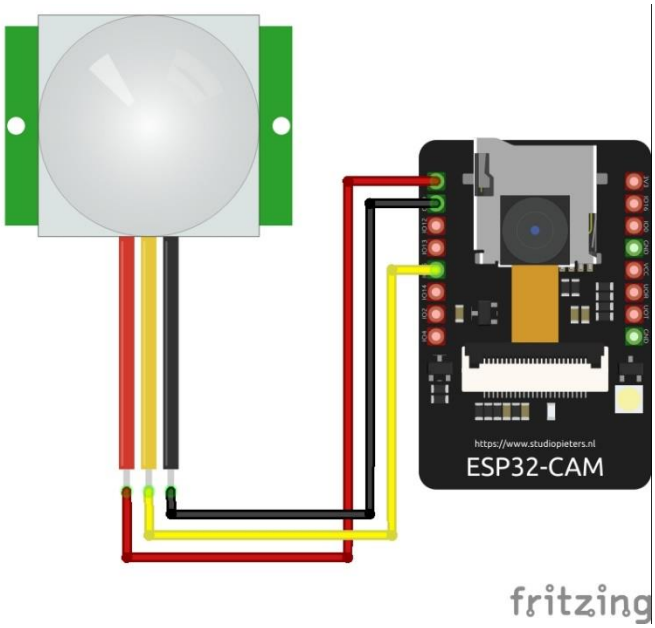


Εικόνα 3.6: Η πίσω όψη της πλακέτας PIR [8]



Εικόνα 3.7: Ρύθμιση του PIR [9]

Όπως φαίνεται στην Εικόνα 3.7 το PIR έχει δύο κουμπιά ρύθμισης, το ένα για τον χρόνο καθυστέρησης ανάμεσα στην ενεργοποίησης ανίχνευσης κίνησης και το άλλο για την ευαισθησία του.



Εικόνα 3.8: Σύνδεση του PIR με το esp32-cam [10]

Στην Εικόνα 3.8 παρουσιάζεται η σύνδεση του PIR με το esp32-cam.

3.2 Arduino IDE

Το Arduino Integrated Development Environment είναι ένα IDE πολλαπλών πλατφορμών που έχει σχεδιαστεί για μικροελεγκτές Arduino. Το IDE χρησιμοποιεί έναν συνδυασμό της τυπικής βιβλιοθήκης C και C++. Χρησιμοποιείται για τη σύνταξη και τη μεταφόρτωση προγραμμάτων σε πλακέτες συμβατές με Arduino και με τη βοήθεια τρίτων και άλλων πλακετών ανάπτυξης προμηθευτών.

Ο πηγαίος κώδικας για το IDE κυκλοφορεί με τη Γενική Δημόσια Άδεια GNU, έκδοση 2. Το Arduino IDE υποστηρίζει τις γλώσσες C και C++ χρησιμοποιώντας ειδικούς κανόνες δόμησης κώδικα. Το Arduino IDE παρέχει μια βιβλιοθήκη λογισμικού από το έργο Wiring, η οποία παρέχει πολλές κοινές διαδικασίες εισόδου και εξόδου. Ο κώδικας που γράφουμε απαιτεί μόνο δύο βασικές συναρτήσεις για την εκκίνηση του και του κύριου βρόχου. Αυτά μεταγλωττίζονται και συνδέονται με ένα πρόγραμμα main() σε ένα εκτελέσιμο κυκλικό εκτελεστικό πρόγραμμα με την αλυσίδα εργαλείων GNU, που περιλαμβάνεται επίσης στη διανομή IDE.

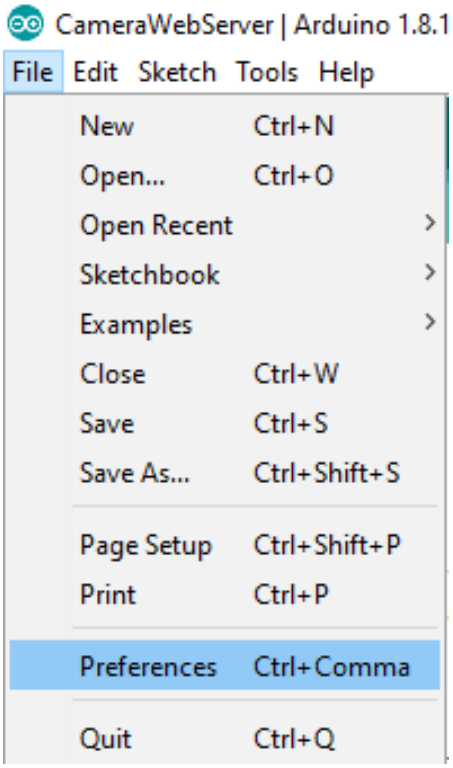
Το Arduino IDE χρησιμοποιεί το πρόγραμμα avr dude για να μετατρέψει τον εκτελέσιμο κώδικα σε αρχείο κειμένου με δεκαεξαδική κωδικοποίηση που φορτώνεται στην πλακέτα Arduino από ένα πρόγραμμα φόρτωσης στην πλακέτα. Από προεπιλογή, το avr dude χρησιμοποιείται ως εργαλείο μεταφόρτωσης.

Το Arduino IDE είναι παράγωγο του Processing IDE αντικαταστάθηκε με το Eclipse Theia IDE που βασίζεται σε κώδικα του Visual Studio. Με την αυξανόμενη δημοτικότητα του Arduino ως πλατφόρμας λογισμικού, άλλοι προμηθευτές άρχισαν να εφαρμόζουν δικούς τους μεταγλωττιστές ανοιχτού κώδικα και εργαλεία (πυρήνες) που μπορούν να δημιουργήσουν και να ανεβάσουν κώδικες σε άλλους μικροελεγκτές που δεν υποστηρίζονται από την επίσημη σειρά μικροελεγκτών του Arduino. Από το 2019 ο οργανισμός Arduino άρχισε να παρέχει έγκαιρη πρόσβαση σε ένα νέο Arduino Pro IDE με εντοπισμό σφαλμάτων και άλλη προηγμένη δυνατότητα.

Για να προγραμματίσουμε την πλακέτα ESP32-CAM με Arduino IDE, πρέπει να έχουμε εγκαταστήσει το Arduino IDE καθώς και το πρόσθετο ESP32. Πριν ξεκινήσουμε αυτήν τη διαδικασία εγκατάστασης, να δούμε ότι έχουμε εγκατεστημένη στον υπολογιστή την πιο πρόσφατη έκδοση του Arduino IDE. Εάν όχι κάνουμε απεγκατάσταση και εγκατάσταση ξανά.

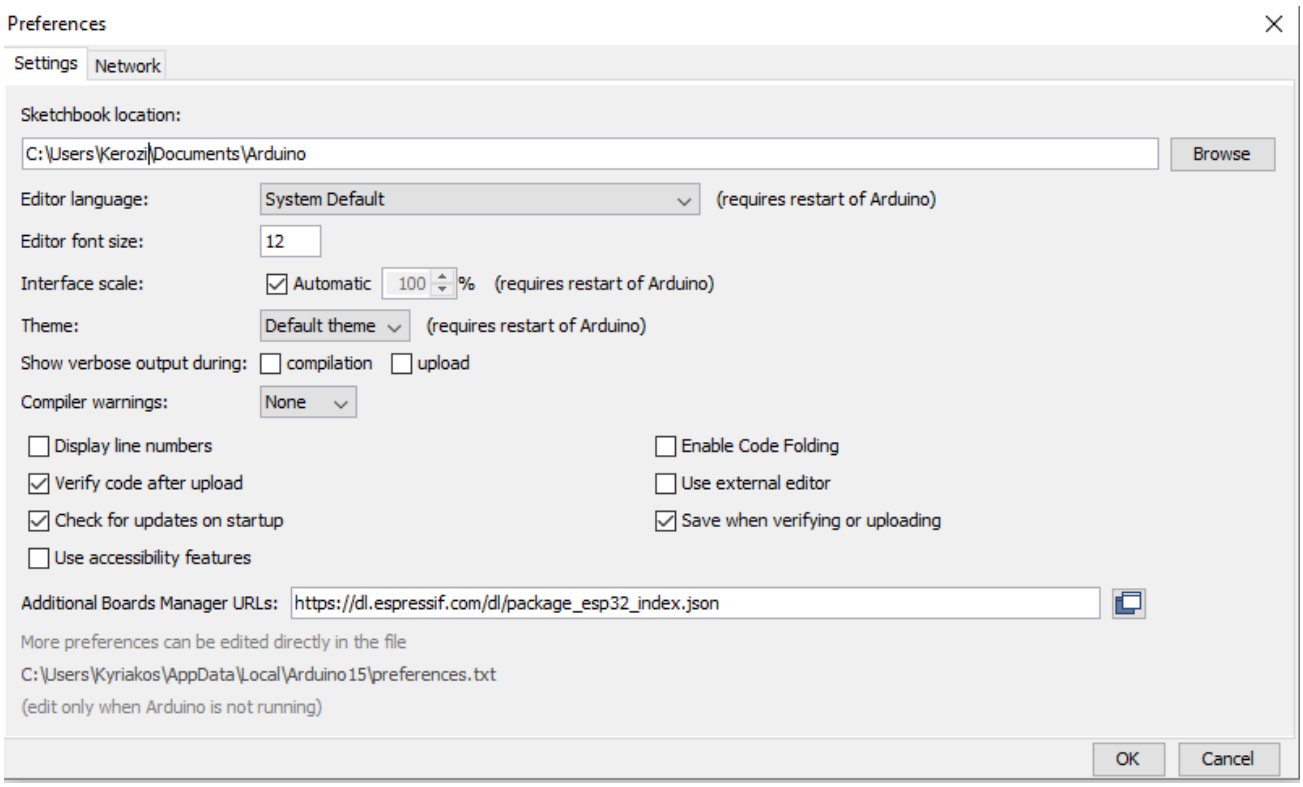
Για να εγκαταστήσουμε την πλακέτα ESP32 στο Arduino IDE, ακολουθούμε την παρακάτω σειρά:

Στο Arduino IDE μεταβαίνουμε στο File -> Preferences



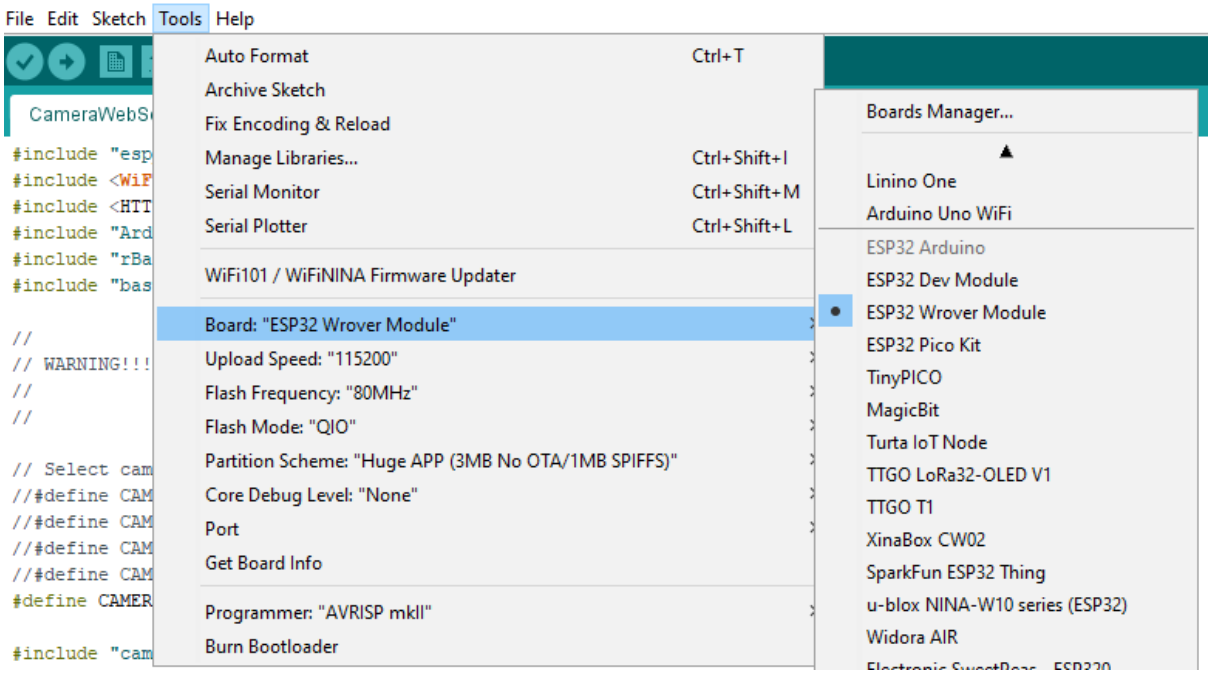
Εικόνα 3.9: Προτιμήσεις στο Arduino IDE

Βάζουμε https://dl.espressif.com/dl/package_esp32_index.json στο "Additional Board Manager URLs" όπως φαίνεται στην Εικόνα 3.10.



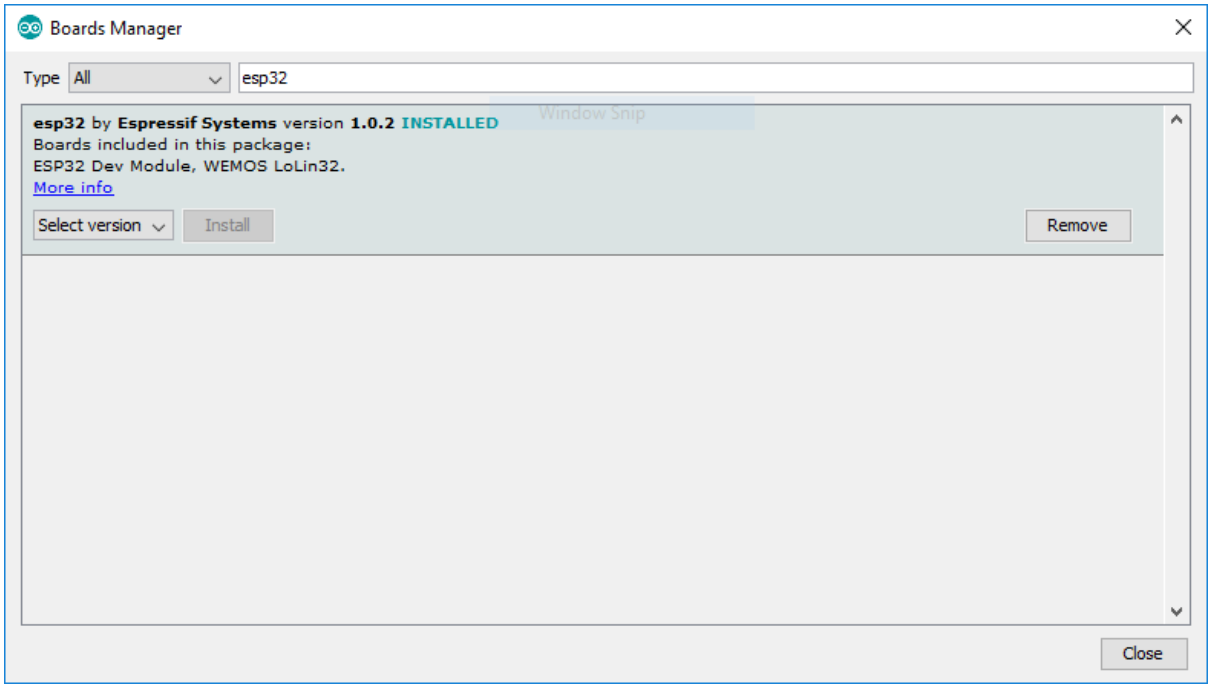
Εικόνα 3.10: Πρόσθεση Board

Ανοίγουμε το Boards Manager και πηγαίνουμε στο Tools > Board > Boards Manager



Εικόνα 3.11: Επιλογή Board

Γράφουμε στην αναζήτηση το ESP32 και κάνουμε εγκατάσταση το “ESP32 by Espressif Systems“.

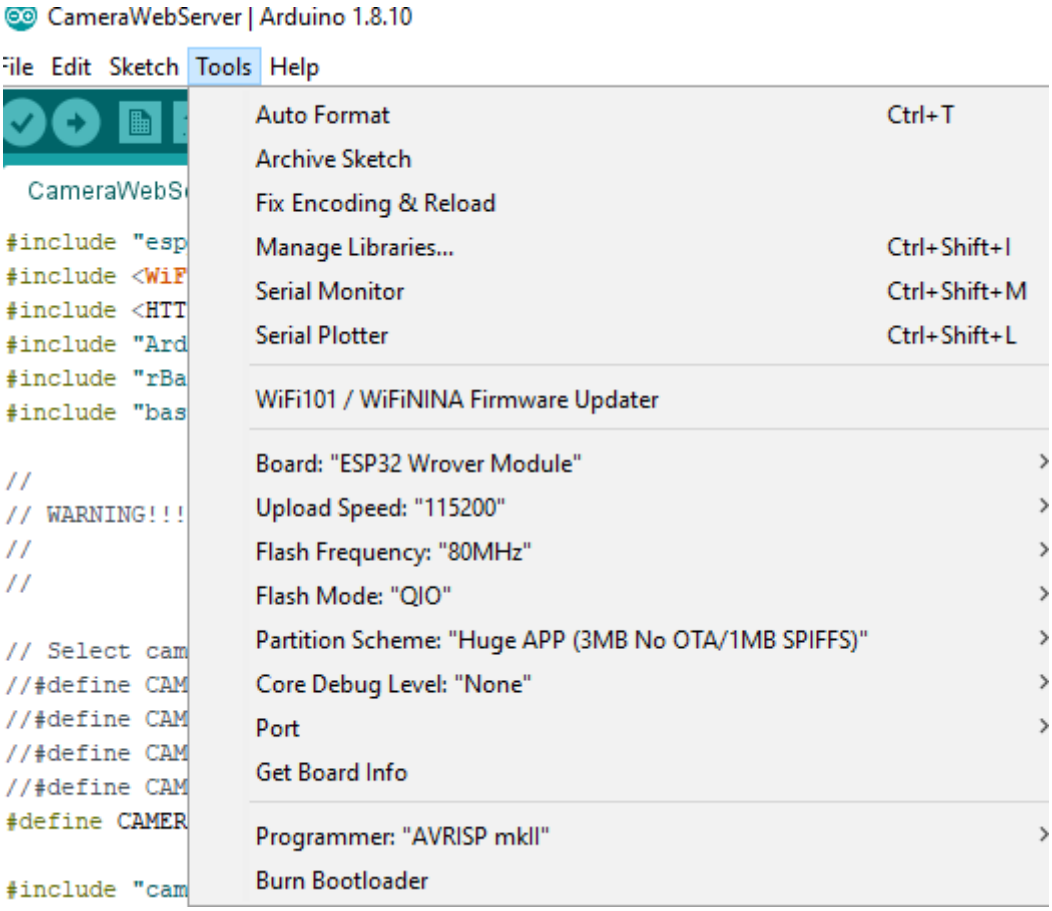


Εικόνα 3.12: Εγκατάσταση ESP32 by Espressif Systems

Για να ανεβάσουμε κώδικα στο ESP32-CAM χρησιμοποιώντας Arduino IDE ακολουθούμε τα ακόλουθα βήματα:

Πάμε στο στο Tools -> Board και επιλέγουμε το AI-Thinker ESP32-CAM. Πρέπει να έχουμε εγκατεστημένο το πρόσθετο ESP32.

Πάμε στο Tools -> Port και επιλέγουμε τη θύρα COM στην οποία είναι συνδεδεμένο το ESP32-CAM.



Εικόνα 3.13: Τελικές Ρυθμίσεις για προγραμματισμό esp32-cam

3.3 C++ για Esp32

Το ESP32 μπορεί να προγραμματιστεί χρησιμοποιώντας πολλά διαφορετικά περιβάλλοντα ανάπτυξης. Ο κώδικας μπορεί να γραφτεί σε C++ (όπως το Arduino) ή σε MicroPython.

Ένα παράδειγμα παριτίθεται παρακάτω όπου φαίνονται οι δύο βασικές συναρτήσεις setup() και loop().

```
// LED on GPIO2
int ledPin = 2;
void setup()
{
   // Set LED as output
   pinMode(ledPin, OUTPUT);

   // Serial monitor setup
   Serial.begin(115200);
}
void loop()
{
   Serial.print("Hello");
   digitalWrite(ledPin, HIGH);
   delay(500);
   Serial.println(" world!");
   digitalWrite(ledPin, LOW);
   delay(500);
}
```

3.4 Python

Σήμερα υπάρχουν πολλές διαθέσιμες γλώσσες προγραμματισμού. Η επιλογή των εργαλείων ανάπτυξης μερικές φορές βασίζεται σε μοναδικούς περιορισμούς ή προσωπικές προτιμήσεις. Οι κύριοι παράγοντες που αναφέρονται για την Python είναι οι εξής:

Ποιότητα λογισμικού:

Για πολλούς, η εστίαση της Python στην αναγνωσιμότητα, τη συνοχή και την ποιότητα του λογισμικού γενικά τη διαφοροποιεί από άλλα εργαλεία στον κόσμο του script. Ο κώδικας Python έχει σχεδιαστεί για να είναι αναγνώσιμος και επαναχρησιμοποιήσιμος και διατηρήσιμο πολύ περισσότερο από τις παραδοσιακές γλώσσες.

Η ομοιομορφία του κώδικα Python καθιστά εύκολη την κατανόηση, ακόμα κι αν δεν τον γράψουμε εμείς. Επιπλέον, η Python έχει βαθιά υποστήριξη για πιο προηγμένους μηχανισμούς επαναχρησιμοποίησης λογισμικού, όπως ο αντικειμενοστραφής προγραμματισμός (OOP).

Παραγωγικότητα προγραμματιστή:

Η Python ενισχύει την παραγωγικότητα των προγραμματιστών πολλές φορές πέρα από μεταγλωττισμένες ή στατικά πληκτρολογημένες γλώσσες όπως η C, η C++ και η Java. Ο κώδικας Python είναι συνήθως το ένα τρίτο έως το ένα πέμπτο του μεγέθους ενός ισοδύναμου κώδικα C++ ή Java. Αυτό σημαίνει ότι γράφουμε λιγότερο κώδικα και χρησιμοποιούμε λιγότερο χρόνο για εντοπισμό σφαλμάτων και για διατήρηση. Τα προγράμματα Python εκτελούνται επίσης αμέσως, χωρίς τα μακρά βήματα μεταγλώττισης και σύνδεσης που απαιτούνται από ορισμένα άλλα εργαλεία, ενισχύοντας περαιτέρω την ταχύτητα του προγραμματιστή.

Φορητότητα προγράμματος:

Τα περισσότερα προγράμματα Python εκτελούνται σε όλες τις μεγάλες πλατφόρμες υπολογιστών. Η μεταφορά κώδικα Python μεταξύ Linux και Windows, για παράδειγμα, είναι συνήθως απλώς θέμα αντιγραφής του κώδικα ενός σεναρίου μεταξύ μηχανών. Επιπλέον, η Python προσφέρει πολλαπλές επιλογές για την κωδικοποίηση φορητών γραφικών διεπαφών χρήστη, προγραμμάτων πρόσβασης σε βάσεις δεδομένων, συστημάτων που βασίζονται στο web και πολλά άλλα. Ακόμη και οι διεπαφές του λειτουργικού συστήματος, συμπεριλαμβανομένων των εκκινήσεων προγραμμάτων και της επεξεργασίας καταλόγου, είναι όσο το δυνατόν πιο φορητές στην Python.

Βιβλιοθήκες υποστήριξης:

Η Python συνοδεύεται από μια μεγάλη συλλογή προκατασκευασμένων και λειτουργιών, γνωστή ως τυπική βιβλιοθήκη. Αυτή η βιβλιοθήκη υποστηρίζει μια σειρά εργασιών προγραμματισμού σε επίπεδο εφαρμογής, από αντιστοίχιση μοτίβων κειμένου έως script δικτύου. Επιπλέον, η Python μπορεί να επεκταθεί τόσο με εγχώριες βιβλιοθήκες όσο και με μια τεράστια συλλογή λογισμικού υποστήριξης

εφαρμογών τρίτων. Ο τομέας τρίτου μέρους της Python προσφέρει εργαλεία για την κατασκευή ιστοσελίδων, αριθμητικό προγραμματισμό, πρόσβαση σε σειριακή θύρα, ανάπτυξη παιχνιδιών και πολλά άλλα. Η επέκταση NumPy, για παράδειγμα, έχει περιγραφεί ως δωρεάν και πιο ισχυρό ισοδύναμο με το σύστημα αριθμητικού προγραμματισμού Matlab.

Ενοποίηση στοιχείων:

Τα σενάρια Python μπορούν εύκολα να επικοινωνήσουν με άλλα μέρη μιας εφαρμογής, χρησιμοποιώντας μια ποικιλία μηχανισμών ολοκλήρωσης. Τέτοιες ενσωματώσεις επιτρέπουν στην Python να χρησιμοποιείται ως εργαλείο προσαρμογής και επέκτασης προϊόντος. Σήμερα, ο κώδικας Python μπορεί να καλέσει βιβλιοθήκες C και C++, μπορεί να κληθεί από προγράμματα C και C++, μπορεί να ενσωματωθεί με στοιχεία Java και .NET, μπορεί να επικοινωνεί μέσω πλαισίων όπως το COM, μπορεί να διασυνδέεται με συσκευές μέσω σειριακών θυρών και μπορεί να αλληλεπιδρά μέσω δικτύων με διεπαφές.

Λόγω της ευκολίας χρήσης και του ενσωματωμένου συνόλου εργαλείων της Python, μπορεί να κάνει την πράξη του προγραμματισμού ευχαρίστηση. Η επίδρασή του στην παραγωγικότητα είναι ένα σημαντικό πλεονέκτημα.

Παράγοντες

Ποιότητα Λογισμικού:

Με βάση το σχεδιασμό, η Python εφαρμόζει μια απλή και ευανάγνωστη σύνταξη και ένα εξαιρετικά συνεκτικό μοντέλο προγραμματισμού. Τα χαρακτηριστικά της γλώσσας αλληλεπιδρούν με συνεπή και περιορισμένους τρόπους και απορρέουν φυσικά από ένα μικρό σύνολο βασικών εννοιών. Αυτό διευκολύνει την εκμάθηση, την κατανόηση και την απομνημόνευση της γλώσσας. Στην πράξη, οι προγραμματιστές Python δεν χρειάζεται να αναφέρονται συνεχώς σε εγχειρίδια όταν διαβάζουν ή γράφουν κώδικα. είναι ένα σταθερά σχεδιασμένο σύστημα που πολλοί βρίσκουν ότι αποδίδει κανονικό κώδικα. Η Python υιοθετεί μινιμαλιστική προσέγγιση. Αυτό σημαίνει ότι παρόλο που υπάρχουν συνήθως πολλοί τρόποι για να ολοκληρωθεί μια εργασία κωδικοποίησης, υπάρχει συνήθως μόνο ένας προφανής τρόπος.

Επιπλέον, η Python δεν λαμβάνει αυθαίρετες αποφάσεις. Πέρα από τέτοια θέματα σχεδίασης, η Python περιλαμβάνει εργαλεία όπως μονάδες και OOP που προωθούν φυσικά την επαναχρησιμοποίηση κώδικα. Και επειδή η Python επικεντρώνεται στην ποιότητα, έτσι είναι και οι προγραμματιστές Python.

Παραγωγικότητα προγραμματιστών:

Η Python έχει λάμψει ως εργαλείο που επιτρέπει στους προγραμματιστές να κάνουν περισσότερα με λιγότερη προσπάθεια. Έχει βελτιστοποιηθεί σκόπιμα για ταχύτητα ανάπτυξη όπως η απλή σύνταξη, η δυναμική πληκτρολόγηση, η έλλειψη βημάτων μεταγλώττισης και το ενσωματωμένο σύνολο εργαλείων επιτρέπουν στους προγραμματιστές να αναπτύσσουν προγράμματα σε ένα κλάσμα του χρόνου που απαιτείται όταν χρησιμοποιούν κάποια άλλα εργαλεία. Το καθαρό αποτέλεσμα είναι ότι η Python συνήθως ενισχύει την παραγωγικότητα των προγραμματιστών πολλές φορές πέρα από τα επίπεδα που υποστηρίζονται από τις παραδοσιακές γλώσσες. Αυτά είναι καλά νέα τόσο σε περιόδους άνθησης όσο και σε περιόδους ύφεσης.

Η Python είναι μια γλώσσα προγραμματισμού γενικής χρήσης που χρησιμοποιείται συχνά σε ρόλους δέσμης ενεργειών. Συνήθως ορίζεται ως αντικειμενοστραφής γλώσσα δέσμης ενεργειών, ένας ορισμός που συνδυάζει την υποστήριξη για OOP με έναν συνολικό προσανατολισμό προς τους ρόλους δέσμης ενεργειών. Στην πραγματικότητα, οι άνθρωποι χρησιμοποιούν συχνά τη λέξη "script" αντί για "πρόγραμμα" για να περιγράψουν ένα αρχείο κώδικα Python. Συνήθως υπερισχύει η προτίμηση για "script" για την περιγραφή ενός απλούστερου αρχείου ανώτατου επιπέδου και "program" για αναφορά σε μια πιο εξελιγμένη εφαρμογή πολλαπλών αρχείων. Επειδή ο όρος "γλώσσα δέσμης ενεργειών" έχει τόσες πολλές διαφορετικές έννοιες για διαφορετικούς παρατηρητές, ορισμένοι θα προτιμούσαν να μην εφαρμοστεί καθόλου στην Python. Στην πραγματικότητα, οι άνθρωποι τείνουν να κάνουν τρεις πολύ διαφορετικούς συσχετισμούς, μερικοί από τους οποίους είναι πιο χρήσιμοι από άλλους, όταν ακούν την Python να χαρακτηρίζεται ως τέτοια:

Εργαλεία κελύφους:

Τα προγράμματα Python μπορούν και εξυπηρετούν πολλούς ρόλους. Δεν είναι απλώς μια καλύτερη γλώσσα σεναρίου κελύφους.

Γλώσσα ελέγχου:

Για άλλους, η δέσμη ενεργειών αναφέρεται σε ένα επίπεδο που χρησιμοποιείται για τον έλεγχο και τη διεύθυνση άλλων στοιχείων εφαρμογής. Τα προγράμματα Python όντως αναπτύσσονται συχνά στο πλαίσιο μεγαλύτερων εφαρμογών. Για παράδειγμα, για τη δοκιμή συσκευών υλικού, τα προγράμματα Python ενδέχεται να καλούν στοιχεία που παρέχουν πρόσβαση χαμηλού επιπέδου σε μια συσκευή. Ομοίως, τα προγράμματα μπορεί να εκτελούν κομμάτια κώδικα Python σε στρατηγικά σημεία για να υποστηρίξουν την προσαρμογή του προϊόντος στον τελικό χρήστη χωρίς την ανάγκη αποστολής και εκ νέου μεταγλώττισης του πηγαίου κώδικα ολόκληρου του συστήματος. Η απλότητα της Python την καθιστά ένα φυσικά ευέλικτο εργαλείο ελέγχου.

Πολλοί προγραμματιστές Python κωδικοποιούν αυτόνομα σενάρια χωρίς ποτέ να χρησιμοποιούν ή να γνωρίζουν για κανένα ενσωματωμένο στοιχείο.

Ευκολία στη χρήση της:

Πιθανώς ο καλύτερος τρόπος για να σκεφτούμε τον όρο «script» είναι ότι αναφέρεται σε μια απλή γλώσσα που χρησιμοποιείται για γρήγορη κωδικοποίηση εργασιών. Αυτό ισχύει ιδιαίτερα όταν ο όρος εφαρμόζεται στην Python, η οποία επιτρέπει πολύ πιο γρήγορη ανάπτυξη προγραμμάτων από μεταγλωττισμένες γλώσσες όπως η C++. Ο γρήγορος κύκλος ανάπτυξής του ενθαρρύνει έναν διερευνητικό, σταδιακό τρόπο προγραμματισμού που πρέπει να βιωθεί για να εκτιμηθεί. Η Python δεν είναι μόνο για απλές εργασίες αλλά κάνει τις εργασίες απλές λόγω της ευκολίας χρήσης και της ευελιξίας της. Η Python έχει ένα απλό σύνολο χαρακτηριστικών, αλλά επιτρέπει στα προγράμματα να κλιμακώνονται σε εξελιγμένα επίπεδα όπως απαιτείται. Εξαιτίας αυτού, χρησιμοποιείται συνήθως για γρήγορες τακτικές εργασίες και μακροπρόθεσμη στρατηγική ανάπτυξη. [11]

3.5 Server - Rest

Το REST, ή το Representational State Transfer, είναι ένα αρχιτεκτονικό στυλ για την παροχή προτύπων μεταξύ συστημάτων υπολογιστών στον Ιστό, διευκολύνοντας τα συστήματα να επικοινωνούν μεταξύ τους. Τα συστήματα συμβατά με REST, που συχνά ονομάζονται συστήματα RESTful, χαρακτηρίζονται από το πώς είναι stateless και διαχωρίζουν πελάτη και διακομιστή.

Στο αρχιτεκτονικό στυλ REST, η υλοποίηση του πελάτη και η υλοποίηση του διακομιστή μπορούν να γίνουν ανεξάρτητα χωρίς ο καθένας να γνωρίζει για τον άλλον. Αυτό σημαίνει ότι ο κώδικας στην πλευρά του πελάτη μπορεί να αλλάξει ανά πάσα στιγμή χωρίς να επηρεαστεί η λειτουργία του διακομιστή και ο κώδικας στην πλευρά του διακομιστή μπορεί να αλλάξει χωρίς να επηρεαστεί η λειτουργία του πελάτη.

Εφόσον κάθε πλευρά γνωρίζει ποια μορφή μηνυμάτων να στείλει στην άλλη, μπορούν να διατηρηθούν ως χωριστά. Διαχωρίζοντας τις λειτουργίες της διεπαφής χρήστη από τις λειτουργίες για την αποθήκευση δεδομένων, βελτιώνουμε την ευελιξία της διεπαφής σε όλες τις πλατφόρμες και βελτιώνουμε την επεκτασιμότητα απλοποιώντας τα στοιχεία διακομιστή. Επιπλέον, ο διαχωρισμός επιτρέπει σε κάθε στοιχείο την ικανότητα να εξελίσσεται ανεξάρτητα.

Χρησιμοποιώντας μια διεπαφή REST, διαφορετικοί πελάτες επιτυγχάνουν τα ίδια τελικά σημεία REST, εκτελούν τις ίδιες ενέργειες και λαμβάνουν τις ίδιες απαντήσεις.

Τα συστήματα που ακολουθούν το παράδειγμα REST είναι χωρίς κατάσταση, που σημαίνει ότι ο διακομιστής δεν χρειάζεται να γνωρίζει τίποτα σχετικά με την κατάσταση στην οποία βρίσκεται ο πελάτης και το αντίστροφο. Με αυτόν τον τρόπο, τόσο ο διακομιστής όσο και ο πελάτης μπορούν να κατανοήσουν οποιοδήποτε μήνυμα λαμβάνεται, ακόμη και χωρίς να δουν προηγούμενα μηνύματα. Αυτός ο περιορισμός του stateless επιβάλλεται μέσω της χρήσης πόρων και όχι εντολών. Οι πόροι του

περιγράφουν οποιοδήποτε αντικείμενο, έγγραφο ή πράγμα που μπορεί να χρειαστεί να αποθηκεύσουμε ή να στείλουμε σε άλλες υπηρεσίες.

Επειδή τα συστήματα REST αλληλεπιδρούν μέσω τυπικών λειτουργιών σε πόρους, δεν βασίζονται στην υλοποίηση διεπαφών. Αυτοί οι περιορισμοί βοηθούν τις εφαρμογές RESTful να επιτύχουν αξιοπιστία, γρήγορη απόδοση και επεκτασιμότητα, ως στοιχεία που μπορούν να διαχειριστούν, να ενημερωθούν και να επαναχρησιμοποιηθούν χωρίς να επηρεαστεί το σύστημα στο σύνολό του, ακόμη και κατά τη λειτουργία του συστήματος.

Στην αρχιτεκτονική REST, οι πελάτες στέλνουν αιτήματα για ανάκτηση ή τροποποίηση πόρων και οι διακομιστές στέλνουν απαντήσεις σε αυτά τα αιτήματα.

Το REST απαιτεί από έναν πελάτη να υποβάλει αίτημα στον διακομιστή προκειμένου να ανακτήσει ή να τροποποιήσει δεδομένα στον διακομιστή. Ένα αίτημα γενικά αποτελείται από:

η HTTP μέθοδος, το οποίο καθορίζει τι είδους λειτουργία πρέπει να εκτελεστεί

μια κεφαλίδα-header, η οποία επιτρέπει στον πελάτη να μεταβιβάσει πληροφορίες σχετικά με το αίτημα

μια διαδρομή προς έναν πόρο ένα προαιρετικό σώμα μηνύματος που περιέχει δεδομένα. [12]

HTTP μέθοδοι

Υπάρχουν 4 βασικές μέθοδοι HTTP που χρησιμοποιούμε σε αιτήματα για αλληλεπίδραση με πόρους σε ένα σύστημα REST:

GET - ανακτήστε έναν συγκεκριμένο πόρο ή μια συλλογή πόρων

POST - δημιουργήστε έναν νέο πόρο

PUT - ενημερώστε έναν συγκεκριμένο πόρο (κατά αναγνωριστικό)

DELETE - καταργήστε έναν συγκεκριμένο πόρο κατά αναγνωριστικό

3.6 Flask

Το Flask είναι ένα μικροπλαίσιο Python που χρησιμοποιείται για τη δημιουργία εφαρμογών Ιστού και REST API. Το Flask παρέχει μια σταθερή ραχοκοκαλιά για τις εφαρμογές και αφήνει πολλές επιλογές σχεδιασμού. Η κύρια δουλειά του Flask είναι να χειρίζεται αιτήματα HTTP και να τα δρομολογεί στην κατάλληλη λειτουργία της εφαρμογής.

Το Flask είναι ένα micro web πλαίσιο γραμμένο σε Python. Ταξινομείται ως μικροπλαίσιο επειδή δεν απαιτεί συγκεκριμένα εργαλεία ή βιβλιοθήκες. Δεν έχει επίπεδο αφαίρεσης βάσης δεδομένων, επικύρωση φόρμας ή άλλα στοιχεία όπου προϋπάρχουσες βιβλιοθήκες τρίτων παρέχουν κοινές λειτουργίες. Ωστόσο, το Flask υποστηρίζει επεκτάσεις που μπορούν να προσθέσουν δυνατότητες εφαρμογής σαν να είχαν υλοποιηθεί στο ίδιο το Flask. Υπάρχουν επεκτάσεις για αντικειμενοσχεσιακούς αντιστοιχιστές, επικύρωση φόρμας, χειρισμό μεταφόρτωσης, διάφορες ανοιχτές τεχνολογίες ελέγχου ταυτότητας και πολλά κοινά εργαλεία που σχετίζονται με αυτό. [12]

Το παρακάτω περιγράφει ένα απλό πρόγραμμα σε Flask-python που όταν ο χρήστης τρέξει την απλή σελίδα θα δει το μήνυμα HI

```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def hello() -> str:
    return " HI"
if __name__ == "__main__":
    app.run(debug=False)
```

3.7 MySql

Η SQL σημαίνει γλώσσα δομημένης αναζήτησης.

Η SQL είναι η τυποποιημένη γλώσσα που χρησιμοποιείται για την πρόσβαση στη βάση δεδομένων.

Η SQL περιλαμβάνει τρία μέρη:

Η γλώσσα ορισμού δεδομένων περιλαμβάνει δηλώσεις που σας βοηθούν να ορίσετε τη βάση δεδομένων και τα αντικείμενά της, π.χ. πίνακες, προβολές, κανόνες ενεργοποίησης, αποθηκευμένες διαδικασίες κ.λπ.

Η γλώσσα χειρισμού δεδομένων περιέχει δηλώσεις που σας επιτρέπουν να ενημερώνετε και να ρωτάτε δεδομένα.

Η γλώσσα ελέγχου δεδομένων σάς επιτρέπει να εκχωρήσετε τα δικαιώματα σε έναν χρήστη για πρόσβαση σε συγκεκριμένα δεδομένα στη βάση δεδομένων.

Το όνομα της MySQL είναι ο συνδυασμός My και SQL, MySQL.

Η MySQL είναι ένα σύστημα διαχείρισης βάσεων δεδομένων που σας επιτρέπει να διαχειρίζεστε σχεσιακές βάσεις δεδομένων. Είναι λογισμικό ανοιχτού κώδικα που υποστηρίζεται από την Oracle.

Σημαίνει ότι μπορείτε να χρησιμοποιήσετε τη MySQL δωρεάν. Επίσης, αν θέλετε, μπορείτε να αλλάξετε τον πηγαίο κώδικα του για να ταιριάζει στις ανάγκες σας.

Παρόλο που η MySQL είναι λογισμικό ανοιχτού κώδικα, μπορείτε να αγοράσετε μια έκδοση εμπορικής άδειας από την Oracle για να λάβετε υπηρεσίες υποστήριξης premium.

Η MySQL είναι πολύ εύκολο να κυριαρχήσει σε σύγκριση με άλλο λογισμικό βάσης δεδομένων όπως το Oracle Database ή ο Microsoft SQL Server.

Το MySQL μπορεί να τρέξει σε διάφορες πλατφόρμες UNIX, Linux, Windows κ.λ.π. Μπορείτε να το εγκαταστήσετε σε διακομιστή ή ακόμα και σε επιτραπέζιο υπολογιστή. Επιπλέον, η MySQL είναι αξιόπιστη, επεκτάσιμη και γρήγορη.

Χαρακτηριστικά

Γραμμένη σε C και C++.

Δοκιμασμένη με ένα ευρύ φάσμα διαφορετικών μεταγλωττιστών.

Λειτουργεί σε πολλές διαφορετικές πλατφόρμες.

Χρησιμοποιεί σχεδιασμό διακομιστή πολλαπλών επιπέδων με ανεξάρτητες μονάδες.

Σχεδιασμένο για να είναι πλήρως πολυνηματική χρησιμοποιώντας νήματα πυρήνα, για εύκολη χρήση πολλαπλών CPU εάν είναι διαθέσιμες.

Παρέχει μηχανές αποθήκευσης συναλλαγών και μη συναλλαγών.

Χρησιμοποιεί πολύ γρήγορους πίνακες δίσκων B-tree (MyISAM) με συμπίεση ευρετηρίου.

Σχεδιασμένο για να κάνει σχετικά εύκολη την προσθήκη άλλων μηχανών αποθήκευσης. Αυτό είναι χρήσιμο εάν θέλετε να παρέχετε μια διεπαφή SQL για μια εσωτερική βάση δεδομένων.

Χρησιμοποιεί ένα πολύ γρήγορο σύστημα εκχώρησης μνήμης που βασίζεται σε νήματα.

Εκτελεί πολύ γρήγορες συνδέσεις χρησιμοποιώντας μια βελτιστοποιημένη ένωση ένθετου βρόχου.

Υλοποιεί πίνακες κατακερματισμού στη μνήμη, οι οποίοι χρησιμοποιούνται ως προσωρινοί πίνακες.

Υλοποιεί λειτουργίες SQL χρησιμοποιώντας μια εξαιρετικά βελτιστοποιημένη βιβλιοθήκη κλάσεων που θα πρέπει να είναι όσο το δυνατόν πιο γρήγορη. Συνήθως δεν υπάρχει καθόλου εκχώρηση μνήμης μετά την προετοιμασία του ερωτήματος.

Παρέχει τον διακομιστή ως ξεχωριστό πρόγραμμα για χρήση σε περιβάλλον δικτύου πελάτη/διακομιστή και ως βιβλιοθήκη που μπορεί να ενσωματωθεί (συνδεθεί) σε αυτόνομες εφαρμογές. Τέτοιες εφαρμογές μπορούν να χρησιμοποιηθούν μεμονωμένα ή σε περιβάλλοντα όπου δεν υπάρχει διαθέσιμο δίκτυο.

Κεφάλαιο 4ο: Ανάλυση του συστήματος παρακολούθησης

4.1 Εισαγωγή

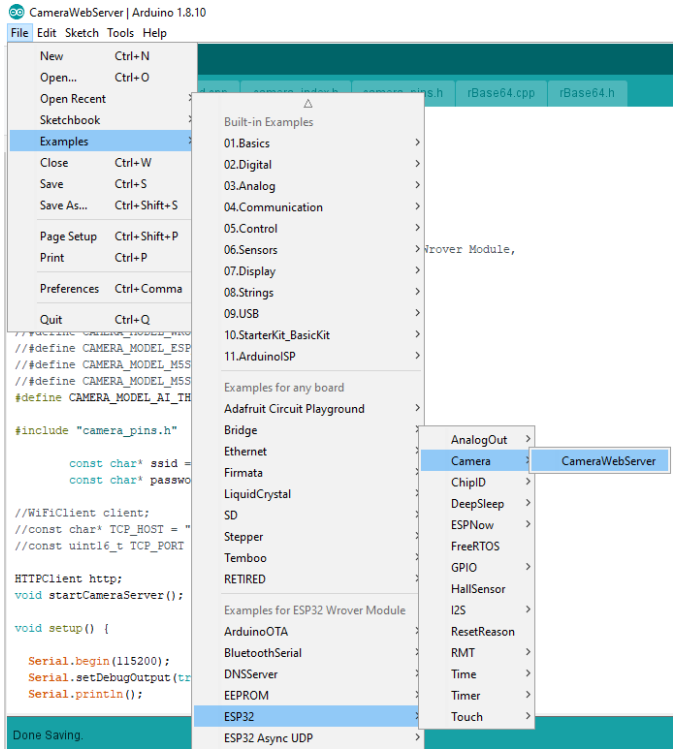
Στο κεφάλαιο αυτό θα αναλυθεί το σύστημα παρακολούθησης και κάθε έργο που το αποτελεί. Θα αναλυθεί το πρόγραμμα στο esp32 που τοποθετείται στο χώρο παρακολούθησης και στη συνέχεια θα περιγραφεί ο server που υλοποιήθηκε με python και η βάση MySQL με την οποία συνδέεται όπου αποθηκεύονται οι εικόνες και οι ειδοποιήσεις που λαμβάνει από το esp32. Τέλος θα αναλυθεί η ιστοσελίδα που παρακολουθεί ο χρήστης τις ειδοποιήσεις από τον python server όταν χρειάζεται.

Η ιστοσελίδα υλοποιήθηκε με Flask python και το πρόγραμμα esp32 με C.

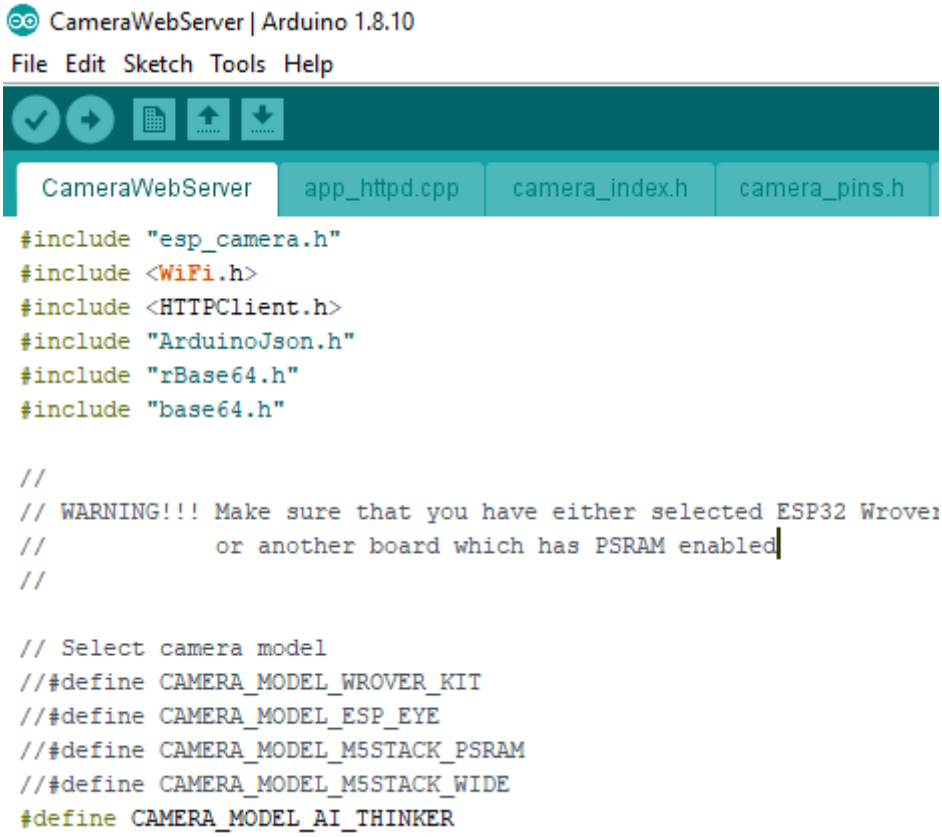
4.2 Ανάλυση προγράμματος στο esp32

Ξεκινώντας θα περιγραφεί πως μπορεί κάποιος να ενεργοποιήσει απευθείας την κάμερα και να δει το βίντεο σε ένα απλό browser.

Στο Arduino IDE, πηγαίνουμε στο File -> Examples -> ESP32 -> Camera και ανοίγουμε το CameraWebServer παράδειγμα, όπως φαίνεται στην Εικόνα 4.1.



Εικόνα 4.1: Έναρξη παραδείγματος CameraWebServer



Εικόνα 4.2: Αρχική

Το άνοιγμα του παραδείγματος θα εμφανίσει την Εικόνα 4.2.

Πριν ανεβεί (φορτωθεί στο eps32) ο κώδικας, πρέπει να εισαχθούν τα στοιχεία του δικτύου στις ακόλουθες μεταβλητές:

```
const char* ssid = "REPLACE_WITH_YOUR_SSID";

const char* password = "REPLACE_WITH_YOUR_PASSWORD";
```

Στη συνέχεια, βεβαιωθείτε ότι έχετε επιλέξει τη σωστή μονάδα κάμερας.

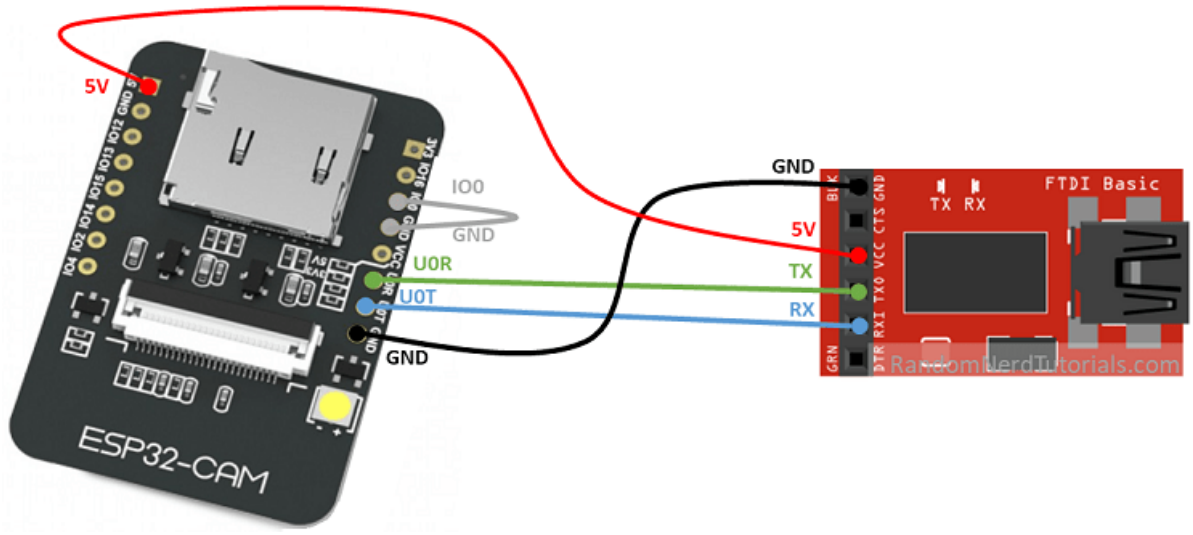
Σε αυτήν την περίπτωση, χρησιμοποιούμε το μοντέλο AI-THINKER.

Αφήνουμε μόνο αυτό όπως φαίνεται παρακάτω:

```
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER
```

Εάν κανένα από αυτά δεν αντιστοιχεί στην κάμερα που πρόκειται να χρησιμοποιηθεί τότε πρέπει να ρυθμιστεί μέσω του camera_pins.h.

Στη συνέχεια συνδέουμε την πλακέτα ESP32-CAM στον υπολογιστή χρησιμοποιώντας έναν προγραμματιστή FTDI, όπως φαίνεται στην Εικόνα 4.3.



Εικόνα 4.3: Σύνδεση esp32-cam με FTDI [5]

Κάποιοι προγραμματιστές FTDI διαθέτουν βραχυκυκλωτήρα που επιτρέπει να επιλεγεί 3,3V ή 5V. Πρέπει ο βραχυκυκλωτήρας να βρίσκεται στη θέση 5V.

Το GPIO 0 πρέπει να συνδεθεί στο GND, ώστε να μπορούμε να ανεβάσουμε/φορτώσουμε κώδικα.

Οι συνδέσεις με το FTDI παρουσιάζονται παρακάτω:

ESP32-CAM	FTDI Programmer
GND	GND
5V	VCC (5V)
U0R	TX
U0T	RX
GPIO0	GND

Για να ανεβάσουμε τον κώδικα, ακολουθούμε τα παρακάτω βήματα:

1) Πάμε στο Tools -> Board και επιλέγουμε AI-Thinker ESP32-CAM

2) Πάμε στο Tools -> Port και επιλέγουμε τη θύρα COM η οποία είναι συνδεδεμένη το ESP32

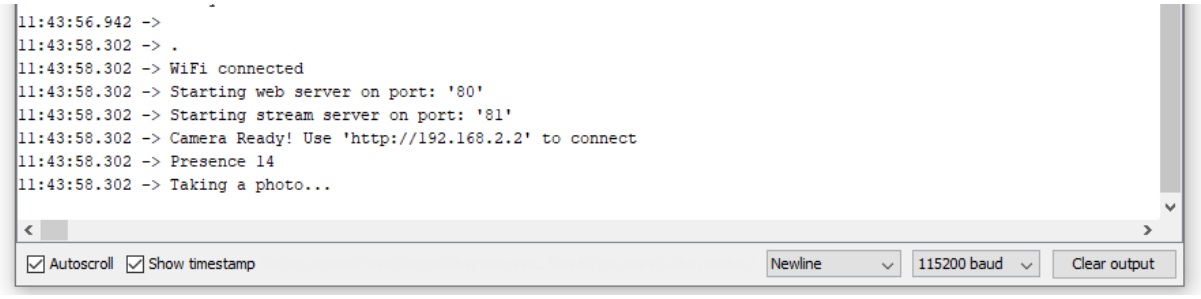
3) Κάνουμε κλικ στο Upload

4) Περιμένουμε να φορτωθεί όταν δείξει 100% στο παράθυρο terminal.

Αφού φορτωθεί ο κώδικας/πρόγραμμα, αποσυνδέουμε το GPIO 0 από το GND

Στη συνέχεια ανοίγουμε τη σειριακή οθόνη από Tools -> Serial Monitor και βάζουμε ρυθμό baud 115200.

Η διεύθυνση IP του ESP32 θα εκτυπωθεί στη Σειριακή οθόνη.



Εικόνα 4.4: Σειριακή οθόνη για προβολή επικοινωνίας

Στην Εικόνα 4.5 φαίνεται η αρχική οθόνη του eps32 – cam όταν κάποιος θέλει να έχει πρόσβαση στα χαρακτηριστικά της κάμερας μέσω browser.

Στο συγκεκριμένο παράδειγμα απλά πατάει

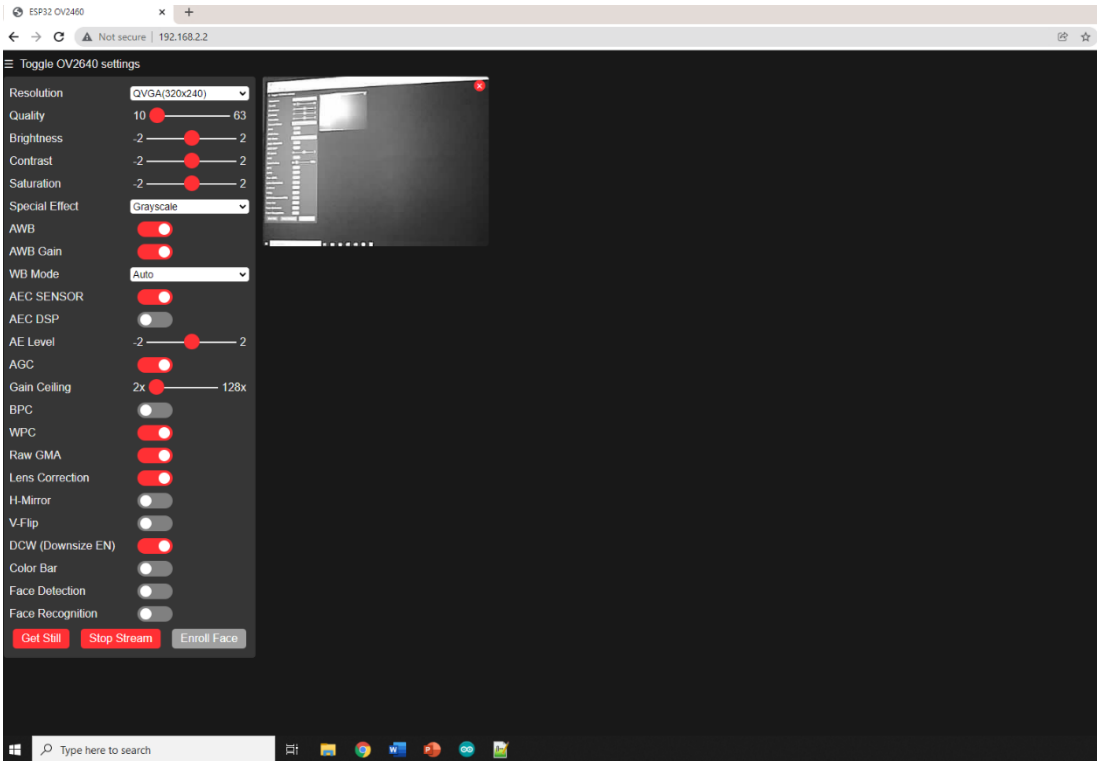
http://192.168.2.2/

και στη συνέχεια το

Start Stream

Για να δει το live video στην οθόνη του.

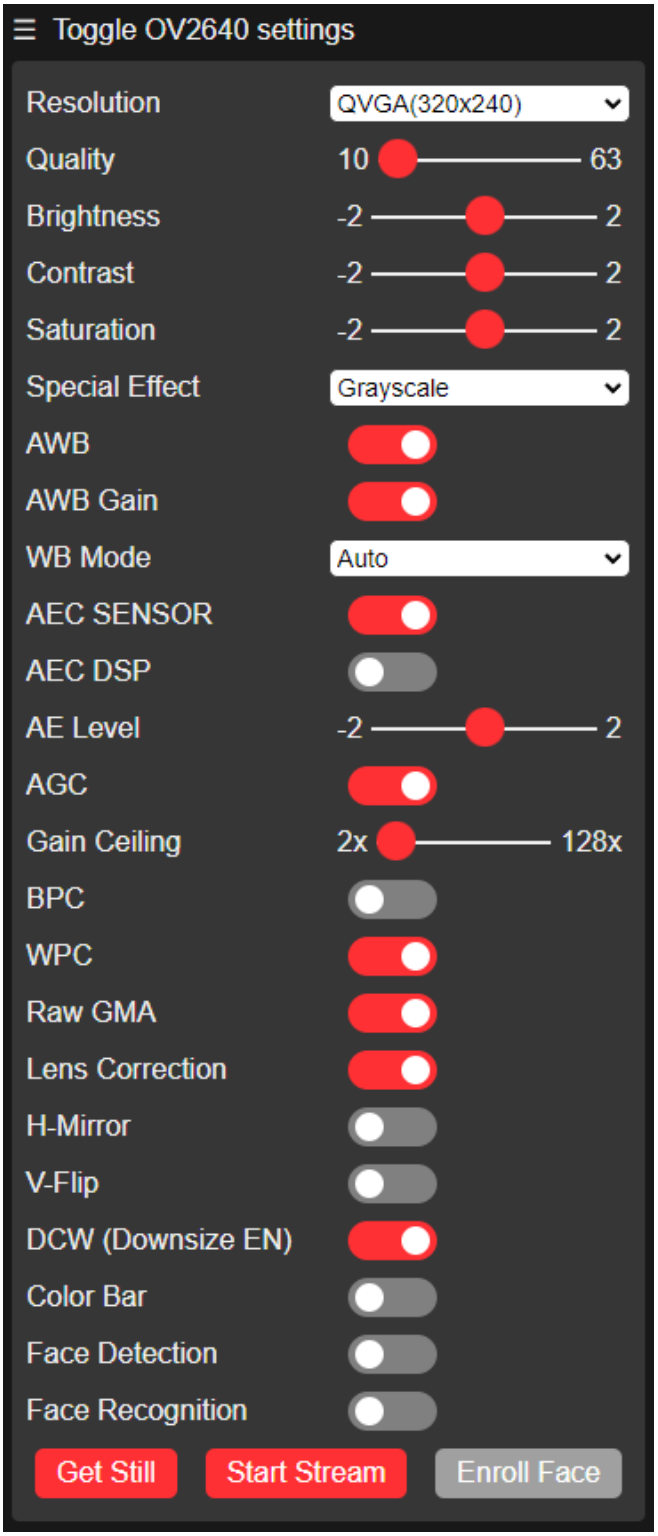
Μπορεί να βλέπει την εικόνα στο χώρο μόνο τοπικά.



Εικόνα 4.5: Αρχική οθόνη όταν χρησιμοποιείται ο ενσωματωμένος τοπικός server του esp32-cam

Στην Εικόνα 4.6 παρουσιάζεται το πάνελ ρυθμίσεων της κάμερας που παρέχεται από τον κατασκευαστή της βιβλιοθήκης – παράδειγμα.

Από τα πιο σημαντικά είναι η ανάλυση και η δυνατότητα για ανίχνευση και αναγνώριση προσώπου αλλά θα πρέπει να είναι ενεργοποιημένη συνέχεια η κάμερα.



Εικόνα 4.6: Αρχική οθόνη της εφαρμογής με ενεργοποιημένο το μενού ρύθμισης του φλας

Αν θέλει να την βλέπει απομακρυσμένα πρέπει να γνωρίζει την ip του router στο χώρο που έχει συνδεθεί και να κάνει port forward μέσω αυτού. Αυτό είναι δύσκολο για κάποιον που βρίσκεται στο σπίτι του και αδύνατο για κάποιον που είναι σε ένα ξένο χώρο όπως ξενοδοχείο.

Όμως όταν το σύστημα esp32-cam είναι συνδεδεμένο στο διαδίκτυο μπορεί να στείλει δεδομένα σε έναν εξωτερικό server.

Η ανίχνευση κίνησης στο πεδίο παρακολούθησης μέσω PIR πραγματοποιείται χωρίς πρόσβαση στο διαδίκτυο αλλά η αποστολή δεδομένων, όπως εικόνα, γίνεται μέσω διαδικτύου στον server και στη συνέχεια ο χρήστης μπορεί να παρακολουθήσει τις ειδοποιήσεις μέσω ιστοσελίδας.

Αρχικά ενεργοποιείται ο εσωτερικός server στον esp32 για την υποστήριξη της κάμερας με την παρακάτω εντολή:

```
void startCameraServer();
```

Κατά την εκτέλεση του προγράμματος καλείται η πρώτη συνάρτηση του esp32.

```
void setup() {
}
```

Σε αυτήν την συνάρτηση περιέχεται η αρχικοποίηση της σειριακής επικοινωνίας του esp32 με τον server.

```
Serial.begin(115200);
```

ρύθμιση των pins για το esp32

```
camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
```

```
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
```

με ένα από τα σημαντικότερα Pins για την σύνδεση με τον αισθητήρα PIR είναι το

```
pinMode(14, INPUT);
```

Στη συνέχεια ρυθμίζεται η κάμερα

```
// camera Αρχικοποίηση
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }
  sensor_t * s = esp_camera_sensor_get();
  s->set_special_effect(s, 2);


  //Χρώμα και θέση κάμερας
  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1);//flip it back
```

```
    s->set_brightness(s, 1);//up the blightness just a bit
    s->set_saturation(s, -2);//lower the saturation
  }
  s->set_framesize(s, FRAMESIZE_QVGA);

#if defined(CAMERA_MODEL_M5STACK_WIDE)
  s->set_vflip(s, 1);
  s->set_hmirror(s, 1);
#endif
```

Θα πρέπει να ρυθμιστεί και να γίνουν προσπάθειες σύνδεσης με το τοπικό δίκτυο μέσω Wi-Fi

```
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
```

Στο τέλος αυτής της συνάρτησης καλείται ο εσωτερικός Server της κάμερας-esp32.

```
  startCameraServer();

  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  Serial.println("' to connect");
```

Μετά την αρχικοποίηση του προγράμματος καλώντας τη setup υπάρχει ακόμα μια συνάρτηση που καλεί το esp32 σε ατέρμων επανάληψη:

```
void loop() {
…
}
```

Αρχικά ελέγχεται το pin 14 αν έχει σήμα με λογικό 1, δηλαδή ανιχνεύτηκε κίνηση από τον αισθητήρα.

```
int isDetected14 = digitalRead(14);
    if(isDetected14 == 1)
     {
     Serial.println("Presence 14");
     camera_fb_t * fb = NULL;
      ……
```

Στη συνέχεια πραγματοποιείται λήψη frame και ελέγχεται αν λήφθηκε με επιτυχία

```
fb = esp_camera_fb_get();


   if (!fb) {
     Serial.println("Camera capture failed");
     return;
   }
```

Αν το frame λήφθηκε σωστά τότε το μετατρέπουμε αρχικά σε πίνακα χαρακτήρων – 1 byte το καθένα

```
const char *data = (const char *)fb->buf;
size_t size = fb->len;
String dataString = base64::encode(fb->buf, fb->len);
```

και στη συνέχεια μετατρέπουμε όλο τον πίνακα που έχει το frame σε base64 συμβολοσειρά.

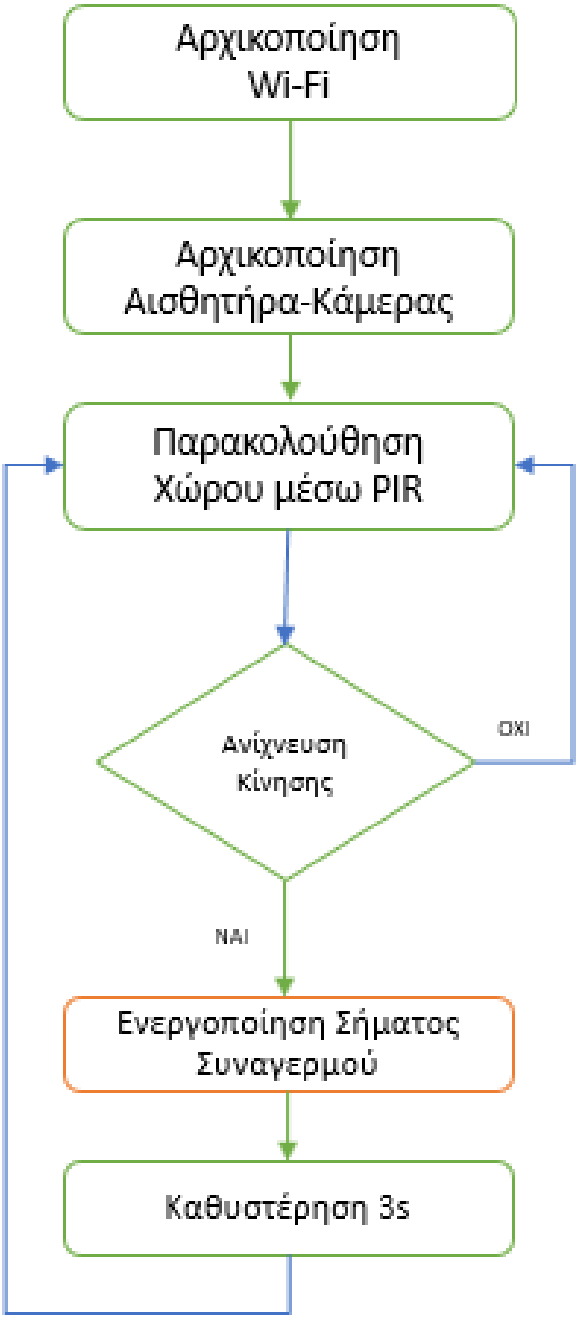
Αμέσως μετά το στέλνουμε στον Server μας

```
//POST
 String serverName ="http://192.168.2.200:4000/postimage";
 http.begin(serverName.c_str());
http.addHeader("Content-Type", "application/x-www-form-urlencoded");
String httpRequestData = "field1="+dataString;
int httpResponseCode = http.POST(httpRequestData);
 //POST
```

Μπορούμε να αποστείλουμε και άλλες πληροφορίες, όπως ποια συσκευή το στέλνει και token.

Επίσης, μπορούμε να ελέγξουμε αν το httpResponseCode που λαμβάνουμε ως επιστροφή από τον server είναι σωστό/επιθυμητό, δηλαδή είναι 200 για ένα σωστό κάλεσμα, όπως φαίνεται παρακάτω

```
 if (httpResponseCode>0) {
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);      }
    else {
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }
```

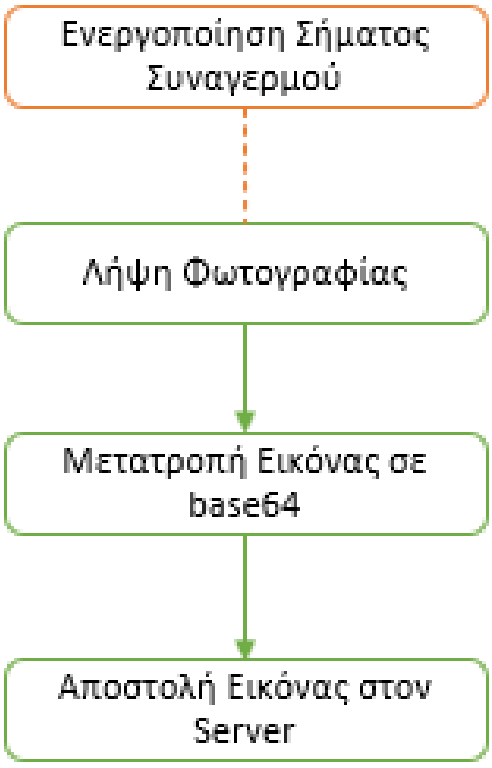


Εικόνα 4.7: Περιγραφή συστήματος συναγερμού στο esp32-cam

Στο τέλος κλείνουμε την επικοινωνία με το server και την κάμερα και βάζουμε καθυστέρηση 3 sec μετά από κάθε ανίχνευση για να μην στέλνει συνέχεια ειδοποιήσεις στον server όταν εντοπίζει συνεχόμενη κίνησης στον χώρο.

```
http.end();
 }
esp_camera_fb_return(fb);
 }
delay(3000);
```

Οι διαδικασίες περιγράφονται με διαγράμματα στις Εικόνες 4.7 και 4.8 και ο κώδικας του esp32 παρατίθεται στο Παράρτημα Β.

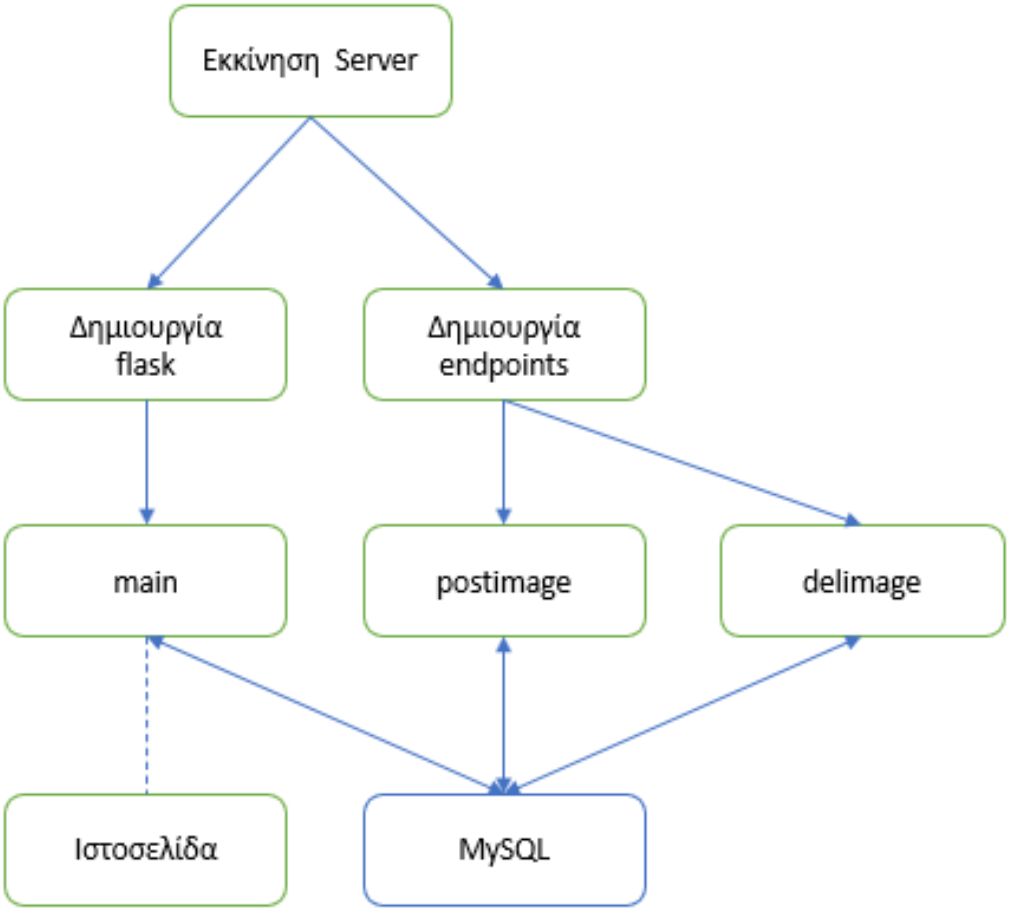


Εικόνα 4.8: Διαδικασία διαχείρισης εικόνας κατά την ενεργοποίηση συναγερμού

4.3 Ο Python server και η βάση δεδομένων

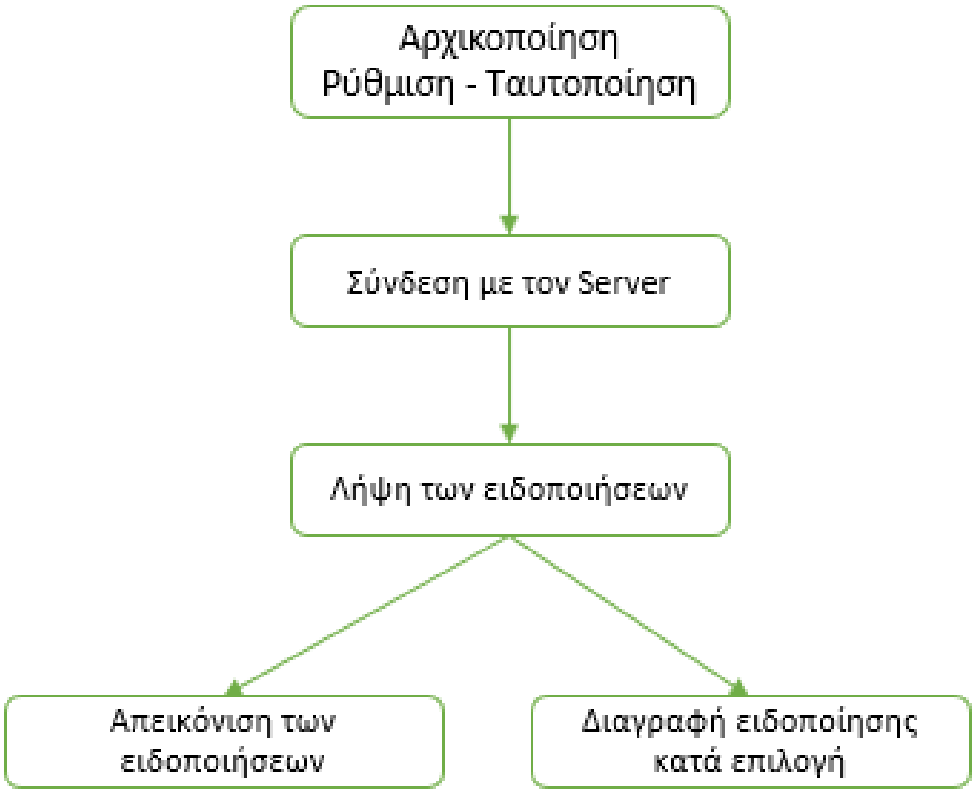
Χρησιμοποιήθηκε python server για την λήψη και αποστολή των εικόνων-ειδοποιήσεων. Η συνεργασία με τη βάση δεδομένων MySql χρησιμοποιείται για την αποθήκευση και ανάγνωση των ειδοποιήσεων. Η εγκατάσταση του framework flask στη python χρησιμοποιείται για την Ιστοσελίδα και τη διαχείριση των ειδοποιήσεων.

Το Διάγραμμα λειτουργίας για τον python server παρουσιάζεται στην Εικόνα 4.9.



Εικόνα 4.9: Διάγραμμα λειτουργίας για τον python server

Στην Εικόνα 4.10 παρουσιάζεται το διάγραμμα ροής για την ιστοσελίδα.



Εικόνα 4.10: Διάγραμμα λειτουργίας για την Flask ιστοσελίδα

Στο παρακάτω κομμάτι κώδικα παρουσιάζονται οι βιβλιοθήκες που χρειάζονται για την εκτέλεση του Flask framework, της σύνδεσης με τη βάση και json μετατροπές.

```
from flask import Flask, Response
from flask import request,jsonify
import json
import mysql.connector
from flask import render_template
app = Flask(__name__)
```

Στη συνέχεια παρουσιάζονται τα δύο σημεία πρόσβασης – endpoint για την λήψη και αποθήκευση της εικόνας που στέλνει το esp32 στον server και για τη διαγραφή μιας ειδοποίησης.

```
@app.route('/postimage', methods=['POST'])
def postimage():
    field1 = request.form['field1']
    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="espcamdb"
    )
    mycursor = mydb.cursor()

   sql = "INSERT INTO espalert (imagedata, datedata,espdevice) VALUES (%s, %s, %s)"
   val = (field1, "1", 1)
   mycursor.execute(sql, val)
   mydb.commit()

    return Response("{'ok':'1'}", status=200, mimetype='application/json')
```

Ενδεικτικά, πραγματοποιείται λήψη στο field1 και αποθηκεύεται στη βάση.

Μια ειδοποίηση διαγράφεται όταν καλείται το σημείο πρόσβασης delimage, όπως φαίνεται παρακάτω.

```
@app.route('/delimage', methods=['GET'])
def getimage():
    id = request.args.get("id")
    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
```

```
    database="espcamdb"
    )
    mycursor = mydb.cursor()
    sql = "DELETE FROM espalert WHERE id = %s"
    adr = (id, )
    mycursor.execute(sql,adr)
    mydb.commit()

    if mycursor.rowcount>0 :
        return Response("1", status=200, mimetype='application/json');
    else:
        return Response("0", status=200, mimetype='application/json');
```

Στη συνέχεια παρουσιάζεται το σημείο πρόσβασης – endpoint για την υποστήριξη ιστοσελίδας

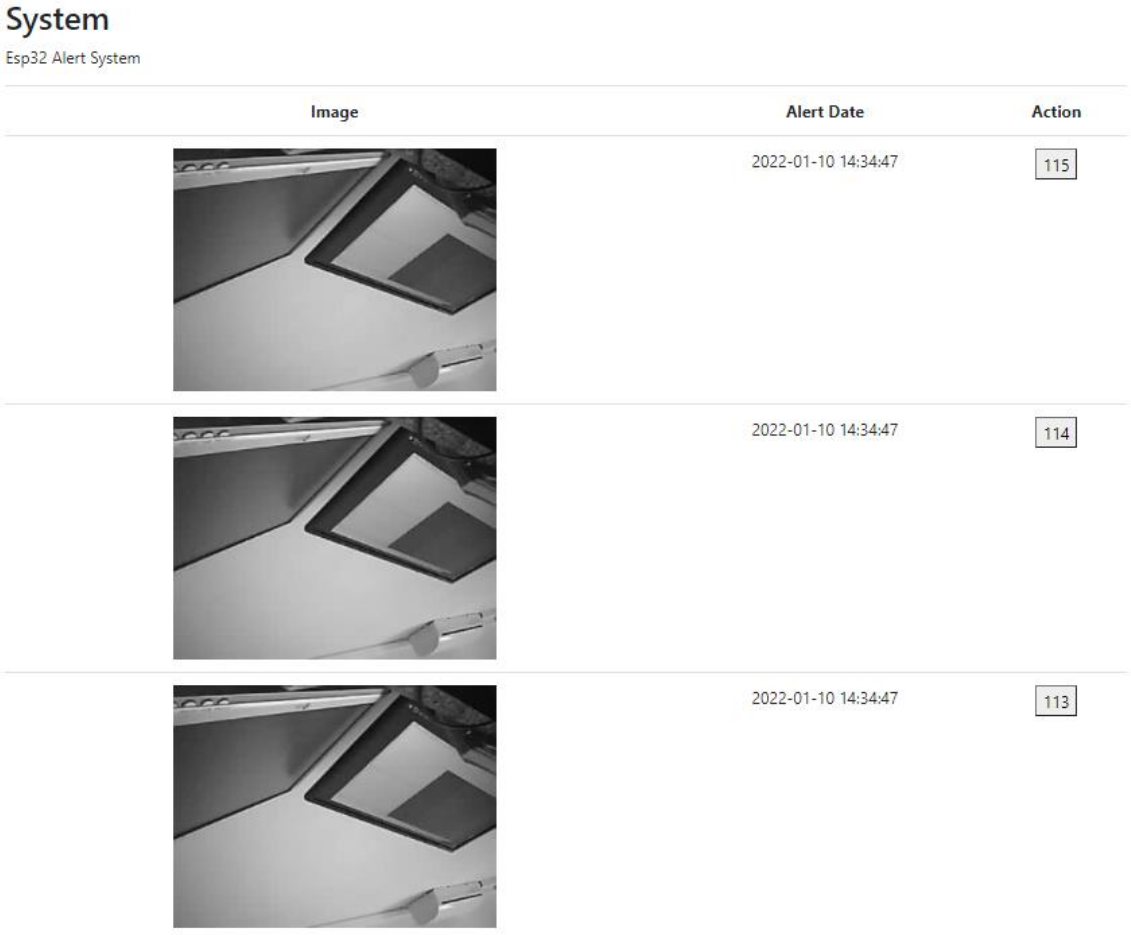
```
@app.route('/', methods=['GET'])
def main():
    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="espcamdb"
    )
    mycursor = mydb.cursor()
    mycursor.execute("SELECT * FROM espalert ORDER BY id DESC")
    records = mycursor.fetchall()
    return render_template('index.html', title='Esp32 Alert System',records=records,
imagedata=records[0][1], alertdate=records[0][2])
```

Στην επιστροφή του φορτώνει μια ιστοσελίδα και της στέλνει όσες παραμέτρους θεωρεί αναγκαία.

Μέρος της ιστοσελίδας παρουσιάζεται στον επόμενο κώδικα:΄

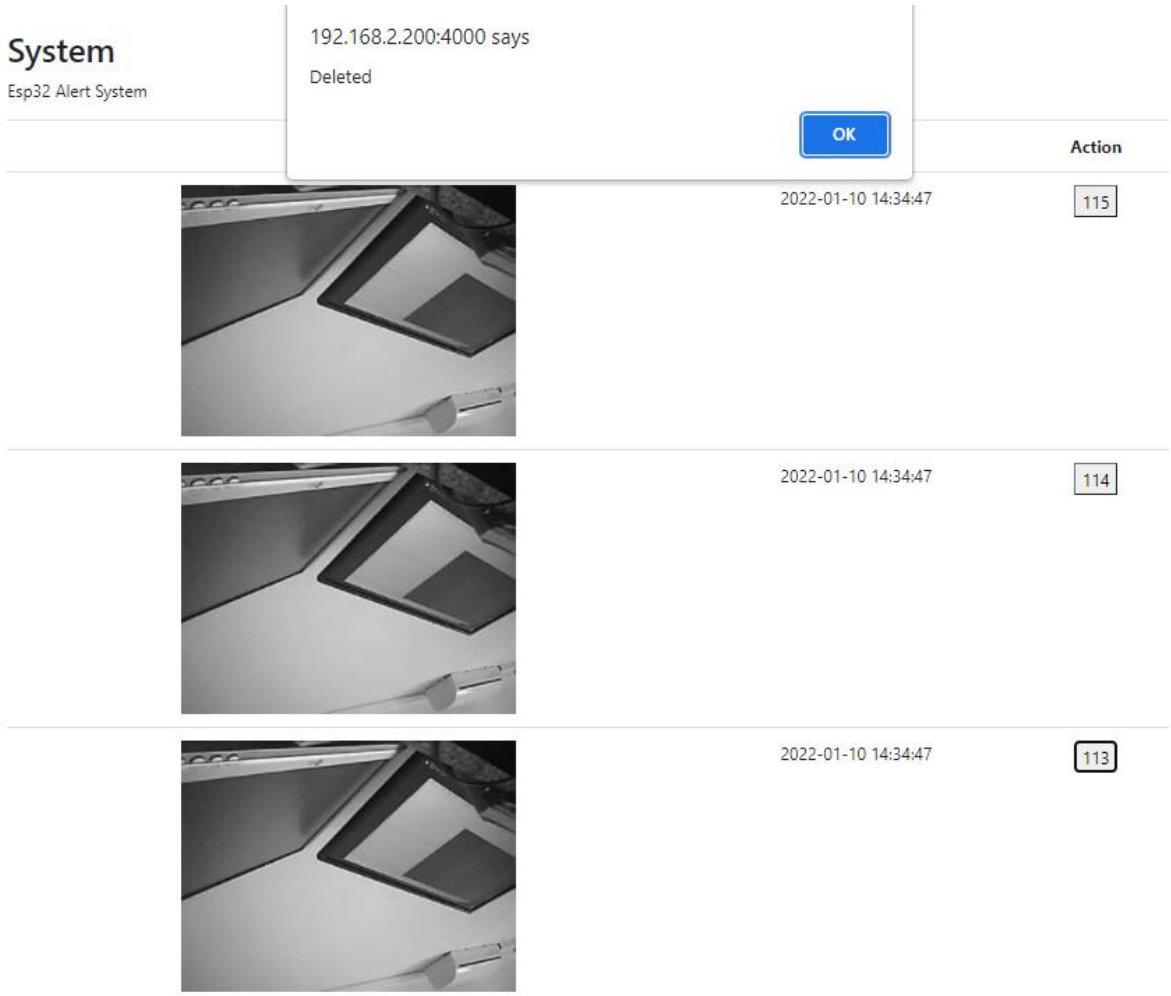
```
<body>
<div class="container">
 </br>
 <h2>System</h2>
 <p>{{title}}</p>

 <table class="table">
  <thead>
   <tr>
    <th>Image</th>
    <th>Alert Date</th>
                <th>Action</th>
   </tr>
  </thead>
  <tbody>
        {% for row in records %}
   <tr>
    <td><img src=" {{row[1]}}" alt="alert image" /></td>
                <td>{{row[2]}}</td>
                <td><button type="button" id="{{row[0]}}"
class="delimg">{{row[0]}}</button></td>
   </tr>
        {% endfor %}
  </tbody>
 </table>
</div>
```

```
</body>
<script>
$('.delimg').click(
function() {
$.ajax({ url: 'http://192.168.2.200:4000/ delimage?id='+this.id, type : 'GET', success : function(result) { if (result == 0) alert("Not Deleted"); if (result == 1) alert("Deleted"); }, error : function(error) { alert("Error"); } })
});
</script>
</html>
```



Εικόνα 4.11: Η ιστοσελίδα διαχείρισης των ειδοποιήσεων

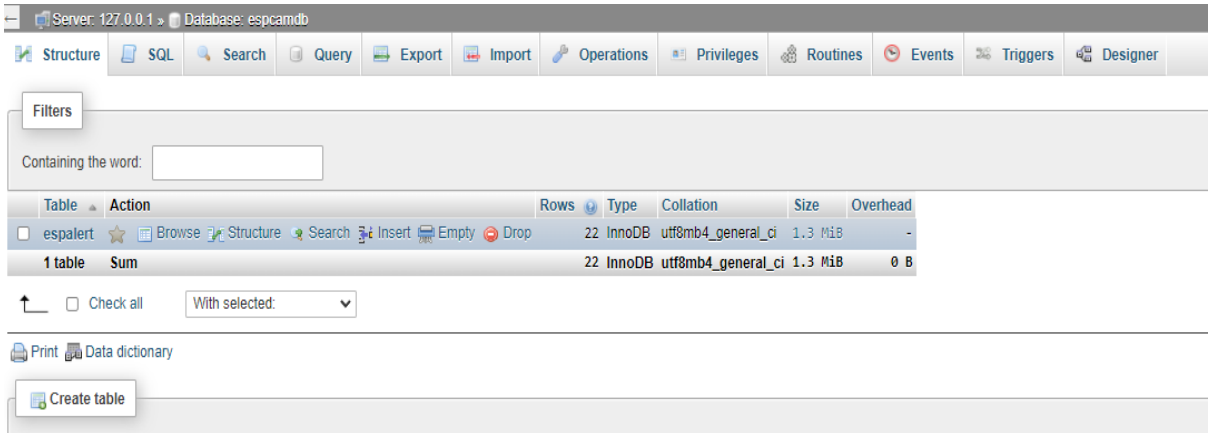
Στην Εικόνα 4.11 παρουσιάζεται η ιστοσελίδα διαχείρισης των ειδοποιήσεων. Η ιστοσελίδα υποστηρίζεται από τον Python server με χρήση του Flask framework και του mySQL server.



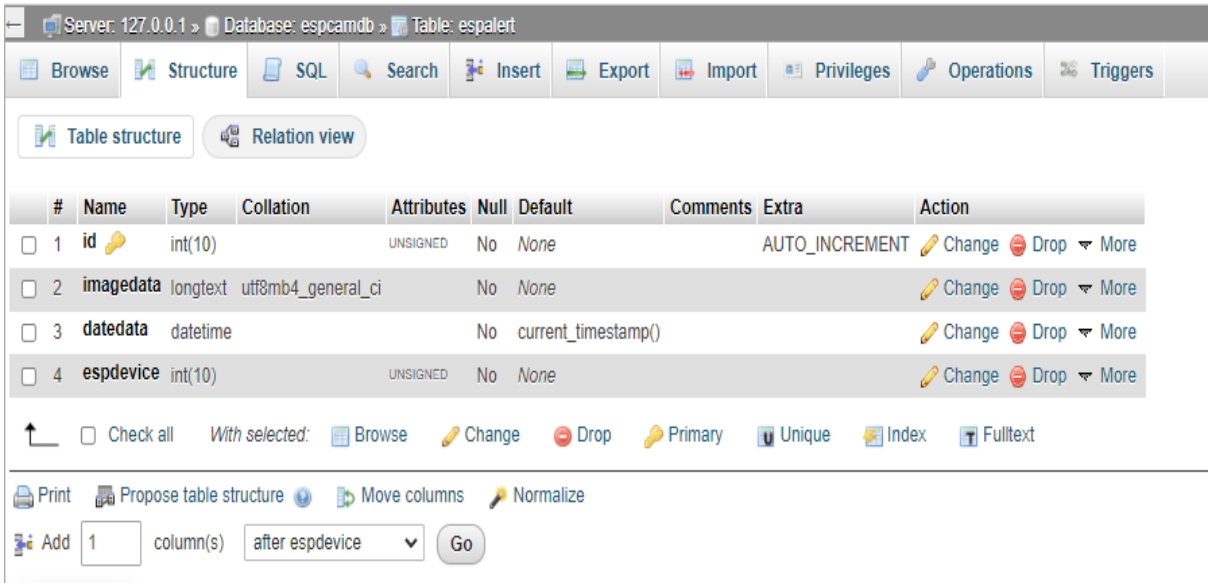
Εικόνα 4.12: Διαγραφή ειδοποίησης

Στην Εικόνα 4.12 παρουσιάζεται η ιστοσελίδα με την εκτέλεση της ενέργειας διαγραφής μιας ειδοποίησης η οποία πραγματοποιήθηκε με χρήση Javascript κώδικα.

Η βάση που δημιουργήθηκε φαίνεται στην Εικόνα 4.13.

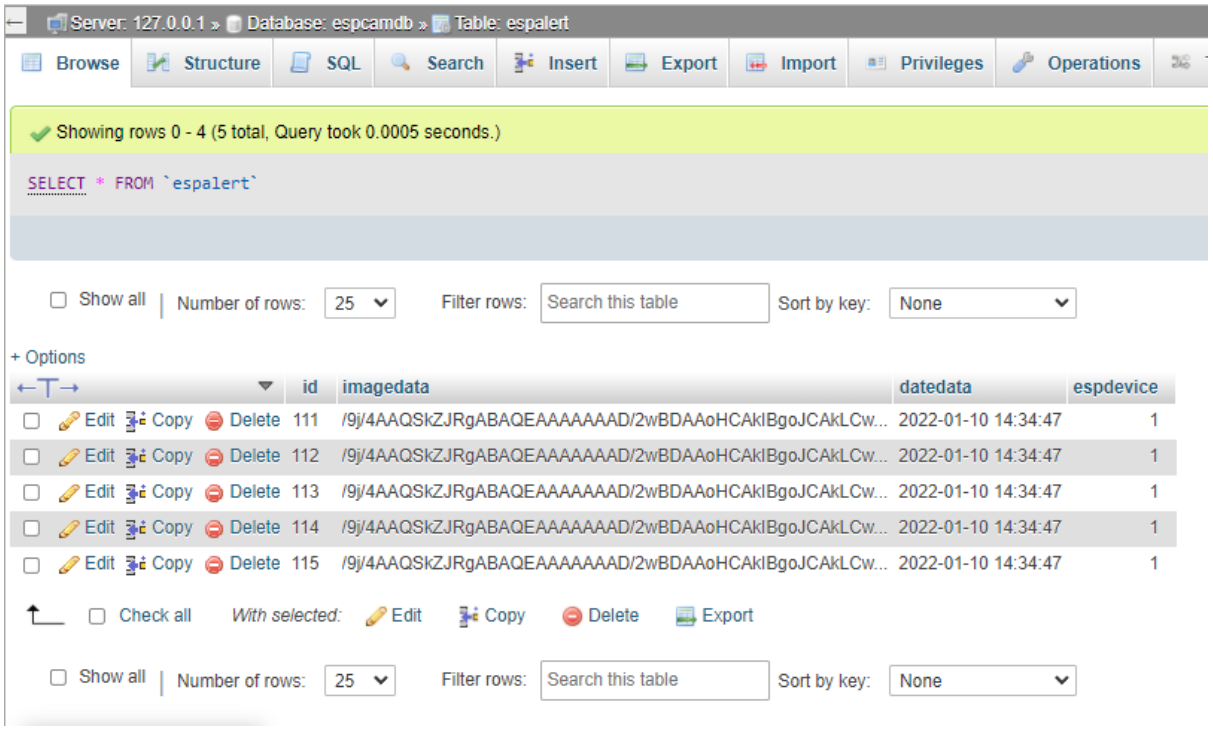


Εικόνα 4.13: Η βάση με τον πίνακα espalert



Εικόνα 4.14: Η δομή του πίνακα espalert

Στην Εικόνα 4.14 φαίνεται η δομή του πίνακα espalert και τι τύπος χρησιμοποιήθηκε σε κάθε πεδίο.



Εικόνα 4.15: Το περιεχόμενο του πίνακα espalert με τις ειδοποιήσεις του

Στην Εικόνα 4.15 παρουσιάζεται δείγμα περιεχόμενου του πίνακα espalert με κάποιες ειδοποιήσεις. Η εικόνα αποθηκεύεται σε base64 μορφοποίηση.

4.4 Συζήτηση για την ασφάλεια στο σύστημα

Ασφάλεια στην αποστολή δεδομένων στον server

Αν κάποιος παραβιάσει τον χώρο και κλέψει το esp32 δεν μπορεί να έχει πρόσβαση στις πληροφορίες ή σε κάποιο κομμάτι κώδικα. Για να λειτουργήσει σωστά η παρακολούθηση το esp32 πρέπει να έχει πρόσβαση στο διαδίκτυο και να επιβλέπει υπό σωστή γωνία τον χώρο. Δεν έχει σημασία αν δεν έχει καλό φωτισμό αφού το PIR ανιχνεύει κίνηση ανθρώπινης παρουσίας. Επίσης, το esp32 πρέπει τροφοδοτείται από powerbank ώστε σε περίπτωση διακοπής ρεύματος σε κάποιο δωμάτιο ξενοδοχείου να μπορεί να έχει ενέργεια για να λειτουργεί αδιάλειπτα.

Κάθε eps32 μπορεί να στέλνει-επικοινωνεί με τον server με ένα token και με ένα μοναδικό id της συσκευής.

Ασφάλεια πρόσβασης στην ιστοσελίδα

Η πρόσβαση στην ιστοσελίδα μπορεί να γίνει από κάποιο χρήστη που διαθέτει το url του server. Δεν έχει υλοποιηθεί πιστοποιημένη πρόσβαση με στοιχεία του χρήστη, ούτε σύνδεση κάποιας συσκευής esp32 με κάποιον χρήστη μέσω της βάσης.

Κεφάλαιο 5ο: Συμπεράσματα και προτάσεις βελτίωσης

Στην εργασία αυτή παρουσιάστηκε και κατασκευάστηκε ένα φορητό και φθηνό (κάτω απο 20 ευρώ) σύστημα παρακολούθησης που μπορεί να τοποθετηθεί εύκολα σε έναν χώρο και να καταγράφει ή/και να αποστέλλει εικόνες/βίντεο όταν υπάρχει κίνηση σε αυτόν. Με ένα esp32-cam και αισθητήρα ανίχνευσης ο χρήστης το τοποθετεί στο δωμάτιο του ξενοδοχείου του ή σε κάποιον άλλο χώρο για να επιβλέπει παραβίαση του χώρου.

Η ασφάλεια του συστήματος περιγράφθηκε στο προηγούμενο κεφάλαιο. Όταν κάποιος παραβιάσει τον χώρο και κλέψει το esp32 δεν μπορεί να έχει πρόσβαση στις πληροφορίες ή σε κάποιο κομμάτι κώδικα. Επίσης το PIR λειτουργεί μέρα/νύχτα αφού ανιχνεύει κίνηση ανθρώπινης παρουσίας.

Η ταυτοποίηση και η σύνδεση κάθε συσκευής με τον χρήστη γίνεται με id και token. Το eps32 μπορεί να στέλνει-επικοινωνεί με τον server με ένα token και με ένα μοναδικό id της συσκευής.

Μια από τις βελτιώσεις είναι η υλοποίηση πιστοποιημένης πρόσβασης στην ιστοσελίδα με στοιχεία του χρήστη και η σύνδεση κάποιας συσκευής esp32 με κάποιον χρήστη μέσω της βάσης.

Να προστεθεί ότι μια καλή βελτίωση της εφαρμογής θα ήταν και η πραγματικού χρόνου βίντεο από την κάμερα μέσω της ιστοσελίδας χωρίς να χρειάζεται port forwarding.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Sreenu, G., Saleem Durai, M.A. Intelligent video surveillance: a review through deep learning techniques for crowd analysis. J Big Data 6, 48 (2019). https://doi.org/10.1186/s40537-019-0212-5

[2] Welsh BC, Piza EL, Thomas AL, Farrington DP. Private Security and Closed-Circuit Television (CCTV) Surveillance: A Systematic Review of Function and Performance. *Journal of Contemporary Criminal Justice*. 2020;36(1):56-69. doi:10.1177/1043986219890192

[3] Hobbs MT, Brehme CS (2017) An improved camera trap for amphibians, reptiles, small mammals, and large invertebrates. PLoS ONE 12(10): e0185026. https://doi.org/10.1371/journal.pone.0185026

[4] Haitham Abbas Khalaf, A.S. Tolba, M.Z. Rashid, Event triggered intelligent video recording system using MS-SSIM for smart home security, Ain Shams Engineering Journal, Volume 9, Issue 4, 2018, Pages 1527-1533, ISSN 2090-4479

[5] https://randomnerdtutorials.com/

[6] https://loboris.eu/ESP32/ESP32-CAM%20Product%20Specification.pdf

[7] https://www.avnet.com/wps/portal/abacus/resources/article/adapting-pir-sensor-technology-to-new-applications/]

[8] https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor

[9] https://www.hackster.io/Grensom/motion-controlled-bed-lights-58610c

[10] https://create.arduino.cc/projecthub/WillMakesTV/discord-security-camera-with-an-esp32-7c4621

[11] https://cfm.ehu.es/ricardo/docs/python/Learning_Python.pdf

[12] https://realpython.com/api-integration-in-python/]

ΠΑΡΑΡΤΗΜΑ Α: Esp32-cam κώδικας

```
#include "esp_camera.h"
#include <WiFi.h>
#include <HTTPClient.h>
#include "ArduinoJson.h"
#include "rBase64.h"
#include "base64.h"

//
// WARNING!!! Make sure that you have either selected ESP32 Wrover Module,
//            or another board which has PSRAM enabled
//

// Select camera model
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"

const char* ssid = "aaa";
const char* password = "bbb";

HTTPClient http;
void startCameraServer();

void setup() {

  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
```

```
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;

  if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }

  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
  }

  sensor_t * s = esp_camera_sensor_get();
  s->set_special_effect(s, 2);

  //initial sensors are flipped vertically and colors are a bit saturated
  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1);//flip it back
    s->set_brightness(s, 1);//up the blightness just a bit
    s->set_saturation(s, -2);//lower the saturation
  }
  //drop down frame size for higher initial frame rate
  s->set_framesize(s, FRAMESIZE_QVGA);

#if defined(CAMERA_MODEL_M5STACK_WIDE)
  s->set_vflip(s, 1);
  s->set_hmirror(s, 1);
#endif

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
```

```
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  startCameraServer();

  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  Serial.println("' to connect");
  //startTCP();

  pinMode(14, INPUT);
  pinMode(15, INPUT_PULLUP);

}

void loop() {
  // put your main code here, to run repeatedly:
     int isDetected14 = digitalRead(14);
     if(isDetected14 == 1)
      {
      Serial.println("Presence 14");
      camera_fb_t * fb = NULL;

    // Take a photo with the camera
    Serial.println("Taking a photo...");

    fb = esp_camera_fb_get();

    if (!fb) {
      Serial.println("Camera capture failed");
      return;
    }

    else {
      const char *data = (const char *)fb->buf;
      size_t size = fb->len;
      String dataString = base64::encode(fb->buf, fb->len);

      //POST
      String serverName ="http://192.168.2.200:4000/postimage";
      http.begin(serverName.c_str());
      http.addHeader("Content-Type", "application/x-www-form-urlencoded");
      String httpRequestData = "field1="+dataString;
      int httpResponseCode = http.POST(httpRequestData);
      //POST
```

```
      Serial.println("------------");
      Serial.println(dataString);
      Serial.println("------------");


      if (httpResponseCode>0) {
        Serial.print("HTTP Response code: ");
        Serial.println(httpResponseCode);
      }
      else {
        Serial.print("Error code: ");
        Serial.println(httpResponseCode);
      }
      // Free resources
      http.end();

      Serial.print("The picture has been saved in ");
    }

    esp_camera_fb_return(fb);

     }

    delay(3000);
}
```

ΠΑΡΑΡΤΗΜΑ Β: Server - Python + html κώδικας

Python Server

```
from flask import Flask, Response
from flask import request,jsonify
import json
import mysql.connector

from flask import render_template
app = Flask(__name__)

#rest endpoint (restfull api)
@app.route('/', methods=['GET'])
def main():
    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="espcamdb"
    )
    mycursor = mydb.cursor()
    mycursor.execute("SELECT * FROM espalert ORDER BY id DESC")
    records = mycursor.fetchall()

    #for row in records:
    #print("id = ", row[0])

    #print("id = ", records[0][0])
    return render_template('index.html', title='Esp32 Alert
System',records=records, imagedata=records[0][1], alertdate=records[0][2])
    #return Response("{'ok':'1'}", status=200, mimetype='application/json')

#@app.route('/index')
#def index():

@app.route('/delimage', methods=['GET'])
def delimage():
    print('getimage')
    id = request.args.get("id")

    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="espcamdb"
    )
```

```
    mycursor = mydb.cursor()

    sql = "DELETE FROM espalert WHERE id = %s"
    adr = (id, )
    mycursor.execute(sql,adr)

    mydb.commit()

    if mycursor.rowcount>0 :
        return Response("1", status=200, mimetype='application/json');
    else:
        return Response("0", status=200, mimetype='application/json');

@app.route('/postimage', methods=['POST'])
def postimage():
    #print(request.data)
    #print(jsonify(request.form))
    field1 = request.form['field1']
    #print(field1)
    field1 = field1.replace(" ","+")

    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="espcamdb"
    )

    mycursor = mydb.cursor()

    #sql = "INSERT INTO espalert (imagedata, datedata,espdevice) VALUES (%s,
%s, %s)"
    #val = (field1, "1", 1)
    #mycursor.execute(sql, val)

    #mydb.commit()

    print(mycursor.rowcount, "record inserted.")

    return Response("{'ok':'1'}", status=200, mimetype='application/json')

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

HTML code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title> {{title}} </title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></scrip
t>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js
"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></sc
ript>

  <style>

table {
  border-collapse: collapse;
  width: 100%;
}

td, th {
  text-align: center;
}
</style>
</head>

<body>

<div class="container">
  </br>
  <h2>System</h2>
  <p>{{title}}</p>

  <table class="table">
    <thead>
      <tr>
        <th>Image</th>
        <th>Alert Date</th>
            <th>Action</th>
```

```
      </tr>
    </thead>
    <tbody>
       {% for row in records %}
      <tr>
        <td><img src=" {{row[1]}}" alt="alert image"
/></td>
            <td>{{row[2]}}</td>
            <td><button type="button" id="{{row[0]}}"
class="delimg">{{row[0]}}</button></td>
      </tr>
        {% endfor %}
    </tbody>
  </table>

</div>

</body>

<script>
$('.delimg').click(

function() {
//alert(this.id)
$.ajax({ url: 'http://192.168.2.200:4000/delimage?id='+this.id, type : 'GET',
success : function(result) { if (result == 0) alert("Not Deleted"); if (result
== 1) alert("Deleted"); }, error : function(error) { alert("Error"); } })
}
);
</script>
</html>
```