ΑΝΩΤΑΤΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΠΕΙΡΑΙΑ ΤΕΧΝΟΛΟΓΙΚΟΥ ΤΟΜΕΑ

ΑΠΟΣΤΟΛΗ ΜΕΤΡΗΣΕΩΝ ΣΕ ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ ΜΕ ΧΡΗΣΗ ΜΙΚΡΟΕΛΕΓΚΤΗ ARDUINO

ΟΝΟΜΑΤΑ ΣΠΟΥΔΑΣΤΩΝ

ΒΕΪΣΛΛΑΡΙ ΑΛΓΚΕΡΤ

ΠΑΝΤΕΛΙΟΣ ΓΙΑΝΝΗΣ

ΟΝΟΜΑ ΕΠΙΒΛΕΠΟΝΤΑ ΚΑΘΗΓΗΤΗ

ΝΙΚΟΛΑΟΥ ΓΡΗΓΟΡΙΟΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Τμήμα Μηχανικών Αυτοματισμού Τ.Ε.

ΝΟΕΜΒΡΙΟΣ 2017

ΤΜΗΜΑ ΑΥΤΟΜΑΤΙΣΜΟΥ

Π. Ράλλη & Θηβών 250, 12244 Αιγάλεω , Αθήνα – Ελλάδα

Τηλ. 210-5381488

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΕΧΟΜΕΝΑ ΕΙΚΟΝΩΝ

ΠΕΡΙΕΧΟΜΕΝΑ ΠΙΝΑΚΩΝ

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Βε'ι'σλλάρι Αλγκέρτ του Ενβερ , με αριθμό μητρώου 38008 φοιτητής του Τμήματος **Μηχανικών Αυτοματισμού Τ.Ε.** του Α.Ε.Ι. Πειραιά Τ.Τ. πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε.) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε., ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

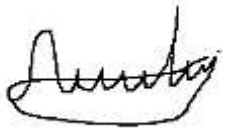
Πέραν των όποιων ποινικών ευθυνών του συγγραφέα σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασης της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση της Π.Ε. με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε. πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσης της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18, παρ. 5 του ισχύοντος Εσωτερικού Κανονισμού.»

Επίσης δηλώνω υπεύθυνα ότι έχω παρακολουθήσει το σεμινάριο συγγραφής και εκπόνησης πτυχιακής εργασίας που διοργανώνεται από το Τμήμα Μηχανικών Αυτοματισμού Τ.Ε. κατά το Χειμερινό/Εαρινό Εξάμηνο του Ακ. Έτους 2015/16

Ο Δηλών

Ημερομηνία

29/11/2017



ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Παντελιός Ιωάννης του Γεωργίου , με αριθμό μητρώου 38690 φοιτητής του Τμήματος **Μηχανικών Αυτοματισμού Τ.Ε.** του Α.Ε.Ι. Πειραιά Τ.Τ. πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε.) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε., ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

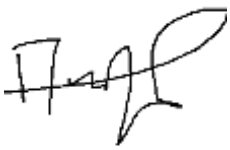
Πέραν των όποιων ποινικών ευθυνών του συγγραφέα σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασης της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση της Π.Ε. με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε. πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσης της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18, παρ. 5 του ισχύοντος Εσωτερικού Κανονισμού.»

Επίσης δηλώνω υπεύθυνα ότι έχω παρακολουθήσει το σεμινάριο συγγραφής και εκπόνησης πτυχιακής εργασίας που διοργανώνεται από το Τμήμα Μηχανικών Αυτοματισμού Τ.Ε. κατά το Χειμερινό/Εαρινό Εξάμηνο του Ακ. Έτους 2015/16

Ο Δηλών

Ημερομηνία

29/11/2017



Περίληψη

Στην συγκεκριμένη πτυχιακή εργασία σκοπός μας είναι να στέλνουμε δεδομένα σε μια βάση δεδομένων μέσω Arduino. Ο αισθητήρας ο οποίος είναι συνδεδεμένος με το Arduino θα παίρνει μετρήσεις και αυτό με τη σειρά του θα τις ανεβάζει στη βάση δεδομένων μέσω μιας άλλης πλακέτας η οποία συνδέεται ασύρματα στο ίντερνετ. Επιπλέον θα υπάρχει η δυνατότητα σύνδεσης κινητού με Bluetooth, που είναι συνδεδεμένο στην πλακέτα, για επιπλέον λειτουργία όπως για παράδειγμα το άνοιγμα μιας πόρτας.

In this particular thesis, we aim to send data in a database via Arduino. The sensor which is connected to the Arduino takes values and upload them in the database via another board which is connected wireless to the net. We also have the capability to connect a mobile phone via Bluetooth to Arduino for further functions such us opening a door

Εισαγωγή

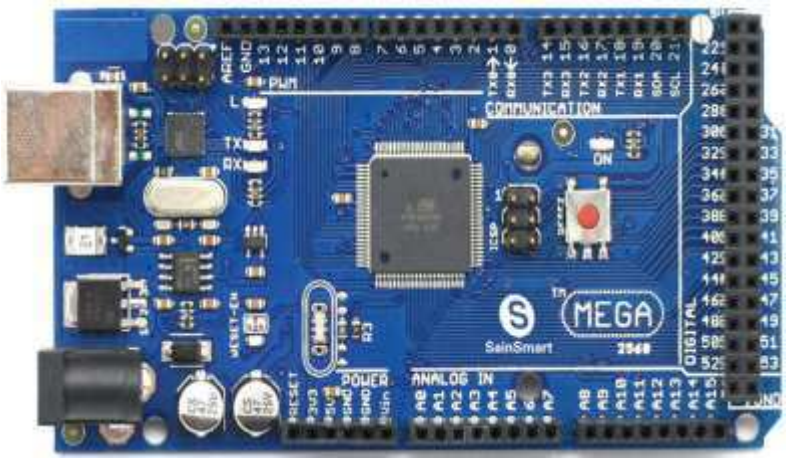
Η επιλογή ενασχόλησης μας με αυτό το συγκεκριμένο είδος πτυχιακής εργασίας ήταν για λογούς “παρατήρησης” της αγοράς. Αν στο μέλλον δημιουργηθεί κάτι αντίστοιχο θα μπορούσε να τοποθετηθεί σε κάποιο κατάστημα και σαν αποτέλεσμα να παίρναμε πληροφορίες για το πόσοι πελάτες το επισκέφτηκαν ,πόσοι αγοράσαν προϊόντα, πόσοι όχι, αλλά και γιατί δεν αγοράσαν ,που αυτό μπορεί να οφείλεται είτε στην κακή εξυπηρέτηση ,είτε στην έλλειψη προϊόντων.

ΚΕΦΑΛΑΙΟ 1
Hardware

1.1 Arduino Mega

1.1.1 Επισκόπηση

Η Arduino mega 2560 είναι μια πλακέτα μικροελεγκτή της οποίας η λειτουργία βασίζεται στον ATmega2560. Περιέχει 54 εισόδους - εξόδους σε μορφή Pin. οι 15 εξ αυτών μπορούν να χρησιμοποιηθούν σαν Pulse-Width Modulation έξοδοι. Επίσης έχει 12 αναλογικές εισόδους ,4 συριακές θύρες ,έναν κρυσταλλικό ταλαντωτή 16MHz μια θύρα USB , μια υποδοχή ρεύματος έναν σύνδεσμο ICSP και ένα κουμπί επανεκκίνησης .



ΕΙΚΟΝΑ 1 ΠΛΑΚΕΤΑ ARDUINO MEGA

1.1.2 Χαρακτηριστικά

Microcontroller026	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm
Width	53.3 mm
Weight	37 g

ΠΙΝΑΚΑΣ 1 ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ATMEGA2560

1.1.3 Τροφοδοσία

Το ARDUINO MEGA 2560 μπορεί να τροφοδοτηθεί από εξωτερικό τροφοδοτικό η μέσω σύνδεσης της θύρας usb που διαθέτει. Η πηγή τροφοδοσίας μπορεί να επιλεγεί αυτόματα. Η εξωτερική τροφοδοσία μπορεί να επέλθει είτε από έναν ac-to-dc ανάκτορα η από μπαταριά. Ο αναπτήρας μπορεί να συνδεθεί βάζοντας ένα βύσμα 2,1 χιλιοστών τύπου center-positive στην υποδοχή του ρεύματος. Η μπαταριά μπορεί να συνδεθεί με καλώδια τα οποία τοποθετούνται στα pins ground και power. Η πλακέτα μπορεί να λειτουργήσει με μια εξωτερική πηγή η οποία κυμαίνεται από 6 έως και 20 volts. Αν τροφοδοτηθεί λιγότερο από 7v τα πενταβολτα pins υπάρχει περίπτωση να τροφοδοτούν λιγότερο από 5v τη πλακέτα, και έτσι να μη λειτουργεί όπως θα πρέπει. Αν η πλακέτα τροφοδοτηθεί με 12v ο ρυθμιστής τάσης μπορεί κάλλιστα να υπερθερμανθεί και να προκαλέσει ζημιά στην πλακέτα μας. Άρα το συνιστώμενο εύρος είναι από 7 μέχρι 12 volts.

Τα pins τροφοδοσίας είναι τα ακόλουθα:

1) Vin. το pin για όταν χρησιμοποιείται η πλακέτα και τροφοδοτείται από εξωτερική πηγή. (σε αντίθεση με τα 5v όταν το τροφοδοτούμε με το usb η κάποια άλλη εξωτερική πηγή ρυθμιζόμενης τάσης) Μπορείς να το τροφοδοτήσεις μέσω αυτού του pin η εάν παρέχεται τάση από την υποδοχή ρεύματος η πρόσβαση παρέχεται μέσω αυτού του ακροδέκτη.
2) 5v. αυτό το pin παράγει μια ρυθμιζόμενη τάση στα 5v από τον ισοσταθμιστή στη πλακέτα. Η πλακέτα μπορεί να τροφοδοτηθεί είτε από συνεχή τάση με καλώδιο τροφοδοσίας που κυμαίνεται από 7μεχρι12v,από θύρα usb , είτε από το pin Vin. Τροφοδοτώντας μέσω των 5 η 3.3v pins μπορεί να σου προκαλέσει κακό και να βλάψει τη πλακέτα πράγμα το οποίο δεν πρέπει να γίνεται.
3) 3v3. Μια παροχή 3.3v που παράγεται από έναν ρυθμιστή τάσης ενσωματωμένο στη πλακέτα. Το μέγιστο ρεύμα που μπορεί να αντέξει είναι 50μα.
4) Gnd ground pins pin γείωσης
5) Ioref. Αυτό το pin το οποίο βρίσκεται στη πλακέτα παρέχει την τάση με την οποία ο μικροελεγκτής λειτουργεί. Ένα σωστά ρυθμισμένο shield μπορεί να διαβάσει το ioref pin και να διαλέξει την κατάλληλη τροφοδοσία δουλεύοντας στα 5η στα 3.3v

1.1.4 Είσοδοι Έξοδοι

Κάθε ένα από τα 54 ψηφιακά pins στην mega πλακέτα μπορούν να χρησιμοποιηθούν σαν είσοδο και σαν έξοδο χρησιμοποιώντας τις εξής "λειτουργίες"(functions). : pinMode(),digitalWrite(), and digitalRead(). Αυτές λειτουργούν στα 5v. Κάθε pin ξεχωριστά μπορεί να παρέχει η να λάβει 20μA σαν προτεινομένη λειτουργική κατάσταση έχοντας μια αντίσταση 20 με 50K. Τα 40mA είναι η τιμή την οποία δε πρέπει να ξεπεράσει για να αποφευχθούν βλάβες στον μικροελεγκτή.

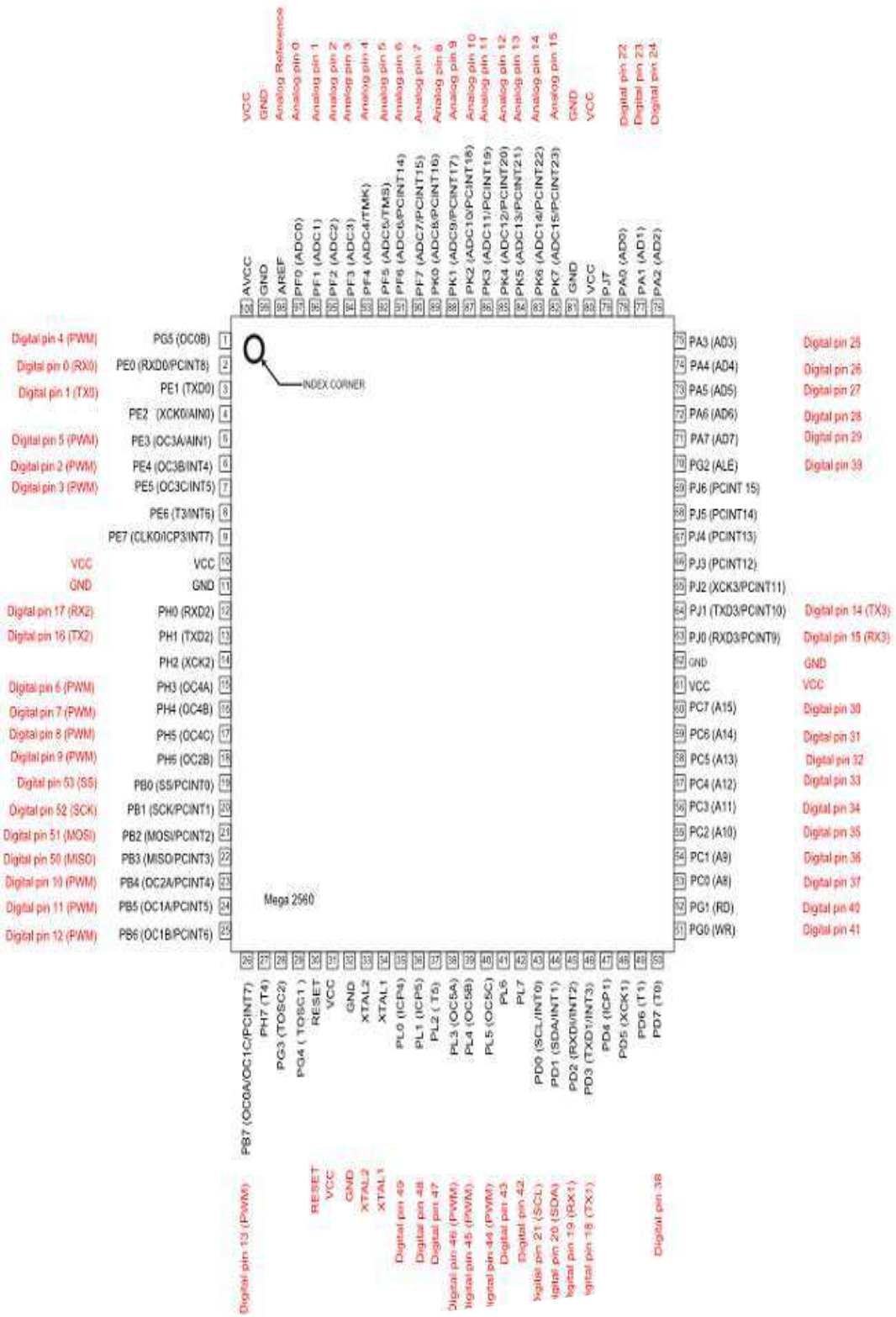
Όμως, μερικά pins έχουν τις παρακάτω εξειδικευμένες λειτουργίες:

- Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Χρησιμοποιείται για να εκπέμψει (RX) και να μεταδώσει (TX) TTL serial data. Τα pin 0και1 είναι επίσης συνδεδεμένα για να ανταποκρίνονται με τα Pins του ATmega16U2 USB-to-TTL Serial chip.
- PWM: 2 μέχρι και 13 και 44 μέχρι και 46 παρέχουν οχτάμπιτι έξοδο PWM με την συνάρτηση analogWrite() .
- Spi: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). Αυτά τα pin υποστηρίζουν SPI επικοινωνία χρησιμοποιώντας την SPI library.
- Led: Υπάρχει ένα ενσωματωμένο led που συνδέεται με το ψηφιακό pin 13. Όταν το pin αυτό είναι high τότε το Pin ανάβει αλλιώς σβήνει.
- Twi: 20(SDA) και 21(SCL). Υποστηρίζουν twi επικοινωνία χρησιμοποιώντας την Wire library.

Το Arduino mega 2560 έχει 16 αναλογικές εισόδους κάθε μια εκ των οποίων μπορεί να παρέχει 10μπιτ ανάλυσης . εξ ορισμού μετρούν από τη γείωση σε 5v , αν και είναι δυνατόν να αλλάξει το άνω άκρο του εύρους τους χρησιμοποιώντας τον ακροδέκτη AREF και τη συνάρτηση analogReference() .

Υπάρχουν όμως και κάποια άλλα pin πάνω στη πλακέτα.

-aref τάση αναφοράς για τις αναλογικές εισόδους χρησιμοποιούμενο με την συνάρτηση analogReference(). - reset . Αν φέρουμε αυτή τη γραμμή σε κατάσταση low κάνουμε reset τον μικροελεγκτή . Τυπικά χρησιμοποιείται για να προσθέσουμε ένα reset κουμπί στις ασπίδες το οποίο μπλοκάρει αυτό της πλακέτας.



ΕΙΚΟΝΑ 1 ΑΚΡΟΔΕΚΤΕΣ ARDUINO MEGA

1.1.5 Επικοινωνία

Η πλακέτα Arduino Mega2560 έχει αρκετές δυνατότητες για την επικοινωνία με τον ηλεκτρονικό υπολογιστή , με άλλες πλακέτες η και με άλλους μικροεπεξεργαστές. Η πλακέτα αυτή παρέχει TTL(5v) σειριακή επικοινωνία . Η ATmega16U2 περιέχει επικοινωνία από τα κανάλια της πλακέτας μια απ αυτές μέσω της θύρας usb και και εικονική θύρα com στο λογισμικό στον υπολογιστή. Το λογισμικό Arduino περιλαμβάνει μια σειριακή οθόνη η οποία μας επιτρέπει να στείλουμε απλά δεδομένα σε μορφή κειμένου από και προς την πλακέτα. Τα RX,TX LEDs πάνω στη πλακέτα αναβοσβήνουν όταν υπάρξει μεταφορά δεδομένων μέσω του τσιπ και της usb σύνδεσης με τον υπολογιστή.

Μια Software Serial library επιτρέπει να υπάρξει σειριακή επικοινωνία σε οποιοδήποτε ψηφιακό pin της πλακέτας αυτής.

Επίσης το Mega2560 υποστηρίζει twi, Spi επικοινωνία. Το λογισμικό της Arduino περιλαμβάνει μια Wire Library για να απλουστεύσει την χρήση της TWI. Για την Spi επικοινωνία χρησιμοποιούμε την Spi library.

1.1.6 Πρωτόκολλο SPI

Λίγα λογία για το πρωτόκολλο Spi.

Το πρωτόκολλο αυτό (Serial Peripheral Interface) χρησιμοποιείται από τους μικροελεγκτές για την επικοινωνία με μια η περισσότερες περιφερικές συσκευές μεταξύ μικρών αποστάσεων. Επίσης μπορεί να χρησιμοποιηθεί για την επικοινωνία μεταξύ 2 μικροελεγκτών. Με τη χρήση αυτού πάντα πρέπει να υπάρχει μια κυριά συσκευή , συνήθως ένας μικροελεγκτής η οποία ελέγχει τις υπόλοιπες περιφερικές συσκευές. Στην περίπτωση μας είναι η πλακέτα Arduino Mega.

Υπάρχουν 3 ακροδέκτες γενικά για τις συσκευές.

- MISO (Master In Slave Out) – Για την αποστολή δεδομένων από το Slave στον Master
- MOSI (Master Out Slave In) – Η γραμμή επικοινωνίας του Master για την αποστολή δεδομένων στις περιφερειακές συσκευές
- SCK (Serial Clock) – Ενεργοποιείται από τη Master συσκευή , το ρολόι στέλνει παλμό ο οποίος συγχρονίζει την εκπομπή δεδομένων

Τέλος υπάρχει ένας ακροδέκτης ειδικά για κάθε συσκευή:

- SS (Slave Select) – Το pin σε κάθε συσκευή το οποίο μπορεί να το χρησιμοποιήσει η κυριά συσκευή (Master) για να ενεργοποιεί και να απενεργοποιεί συγκεκριμένες συσκευές

Όταν το pin μιας συσκευής SS είναι σε κατάσταση low επικοινωνεί με την κύρια συσκευή. Όταν είναι high την αγνοεί. Αυτό επιτρέπει πολλαπλές συσκευές SPI να μοιραστούν τους ίδιους MISO,MOSI και CLK ακροδέκτες.

1.1.6 Προγραμματισμός.

Η πλακέτα αυτή μπορεί να προγραμματιστεί με το λογισμικό Arduino (ide) επιλέγοντας από το μενού του λογισμικού την πλακέτα αυτή.

Η ATmega2560 είναι προγραμματισμένη με έναν bootloader προγράμματος ο οποίος σου επιτρέπει να ανεβάσεις νέο κώδικα χωρίς να χρησιμοποιείς εξωτερικά υλικά προγραμματισμού. Επικοινωνεί χρησιμοποιώντας το αυθεντικό STK500 πρωτόκολλο.

Μπορεί επίσης να παρακαμφθεί ο bootloader και να προγραμματιστεί μέσω της icsp υποδοχής.

1.1.7 Αυτόματη επαναφορά.

Αντί να απαιτείται το πάτημα κουμπιού για να κάνουμε reset πριν ανεβάσουμε κώδικα ,η πλακέτα Mega2560 είναι σχεδιασμένη με τέτοιο τρόπο που επιτρέπει να γίνεται reset

, από το πρόγραμμα που τρέχει στον συνδεδεμένο υπολογιστή. Μια από τις γραμμές ελέγχου ροής υλικού (dtr) του AtMega8υ2 είναι συνδεδεμένη στη γραμμή επαναφοράς στου AtMega2560 μέσω ενός πυκνωτή 100nanofarad. Όταν η γραμμή αυτή βρίσκεται σε κατάσταση low η γραμμή επαναφοράς πέφτει για να επαναφέρει το chip. Το λογισμικό Arduino χρησιμοποιεί

αυτή τη δυνατότητα για να επιτρέψει να ανεβάσουμε κώδικα απλά πατώντας ένα κουμπί στο περιβάλλον της Arduino. Αυτό σημαίνει ότι ο bootloader μπορεί να έχει μικρότερο χρόνο λήξης όσο το dtr μικραίνει και μπορεί να συντονιστεί με την έναρξη της φόρτωσης προγράμματος.

Αυτή η εγκατάσταση έχει κάποιες επιπτώσεις. Όταν το Mega 2560 είναι συνδεδεμένο σε υπολογιστή χρησιμοποιημένο Mac η Linux κάνει επανεκκίνηση κάθε φορά που υπάρχει σύνδεση μέσω του usb. Για το επόμενο μισό δευτερόλεπτο περίπου ο bootloader τρέχει στο AtMega2560. Αν και αυτό είναι προγραμματισμένο να αγνοεί ακατάλληλα δεδομένα θα παρακολουθήσει τα πρώτα bytes των δεδομένων που έχουμε στείλει στη πλακέτα αφού έχει ανοίξει μια σύνδεση. Εάν ένα πρόγραμμά τρέχει στη πλακέτα και λάβει μια διαμόρφωση η ένα άλλο δεδομένο όταν αυτό ξεκινάει πρέπει να είμαστε σίγουροι ότι το λογισμικό με το οποίο επικοινωνεί αναμένει 1 δευτερόλεπτο μετρά το άνοιγμα της σύνδεσης και πριν την αποστολή δεδομένων. Η Mega2560 πλακέτα περιέχει ένα μονοπάτι το οποίο μπορεί να κοπεί ώστε να απενεργοποιηθεί η αυτόματη επανεκκίνηση. Οι άκρες του καλωδίου μπορούν να ενωθούν για να γίνει πάλι εφικτή η αυτόματη επανεκκίνηση. Έχει ετικέτα reset-en. Τέλος μπορείς επίσης να απενεργοποιήσεις την αυτόματη επανεκκίνηση συνδέοντας μια αντίσταση 110Ωm από το 5ν στη γραμμή επαναφοράς.

1.1.8 Φυσικά χαρακτηριστικά

Το μέγιστο μήκος και πλάτος του Mega2560 pcb είναι 4 και 2.1 ίντσες με την υποδοχή ρεύματος και της θύρας usb να εκτείνονται μακρύτερα από τις διαστάσεις αυτές. Η πλακέτα μπορεί να συνδεθεί η να κολληθεί σε μια θήκη για παράδειγμα μέσω των 3 τρυπών .Σημειώνεται ότι η απόσταση αναμεσά στα ψηφιακά pins 7και8 είναι 0,16μιλ.

1.2 ADAFRUIT CC3000

ΕΙΚΟΝΑ 2 ΠΛΑΚΕΤΑ ADAFRUIT CC3000

1.2.1 Επισκόπηση

Η λειτουργική μονάδα CC3000 της εταιρίας Texas Instrument είναι ένας αυτόνομος επεξεργαστής ασύρματου δικτύου που απλοποιεί την σύνδεση στο Internet με τη χρήση απλής τεχνολογίας Link Wi-Fi. Η Simple Link Wi-Fi CC3000 ελαχιστοποιεί τις απαιτήσεις του λογισμικού του μικροελεγκτή υποδοχής (MCU) και είναι επομένως μια ιδανική λύση για ενσωματωμένες εφαρμογές χρησιμοποιώντας οποιαδήποτε χαμηλού κόστους και χαμηλής ισχύος MCU.

1.2.2 Τεχνικά χαρακτηριστικά του cc3000

	CC3000's Characteristics
Standards	802.11 b/g, BSS Station
Wi-Fi Security Modes	WEP, WPA/WPA2 (AES and TKIP – Personal)
Embedded Wi-Fi	TCP/IP stack (IPv4 – DHCP client, DNS, mDNS, ARP), Wi-Fi driver, security supplicant, Auto-calibrated radio
Required Code size	As low as 5KB Flash & 360B RAM
Host Interface	SPI at 16MHz
Power Modes	Active Mode (92mA typ RX current), Shutdown Mode (<5uA)
Power Supply	2.9V – 4.8V
I/O Voltage	1.8V – 3.6V
Connections	4 Sockets (UDP or TCP)
Throughput (TCP)	~4 Mbps*
Headless Configuration	SmartConfig™ Technology

ΠΙΝΑΚΑΣ 2 ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΤΟΥ CC3000

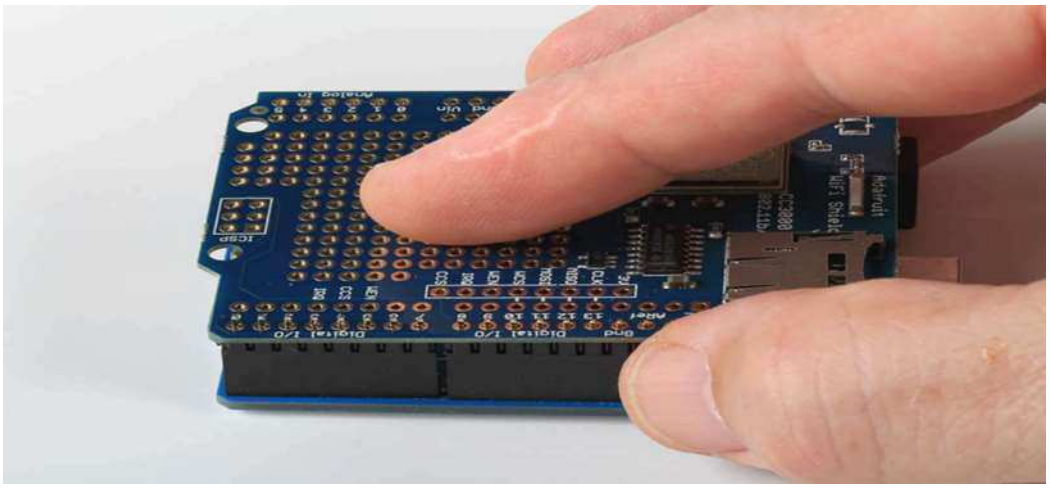
1.2.3 Συνδέσεις

Το CC3000 είναι (ηλεκτρικά) αρκετά απλό στη χρήση. Η μονάδα απαιτεί σύνδεση στο SPI, συμπεριλαμβανομένου ενός ρολογιού (CLK), τα δεδομένα από έναν μικροελεγκτή (MOSI) και τα δεδομένα εξόδου στον μικροελεγκτή (MISO). Χρησιμοποιεί επίσης μια επιλεγμένη γραμμή (CS) για SPI να δείχνει πότε μια μεταφορά δεδομένων όπως ξεκίνησε.

Μαζί με τη διεπαφή SPI, υπάρχει ένα εξωτερικά ενεργοποιούμενο pin που ονομάζεται VBAT_EN και χρησιμοποιούμε για να ξεκινήσουμε τη μονάδα σωστά και επίσης ένα pin IRQ, που είναι η διακοπή από το CC3000. Το pin IRQ απαιτεί να επικοινωνεί και πρέπει να συνδέεται με ένα pin εισόδου του Arduino. Από την Mega συνήθως χρησιμοποιούμε τις # 2 ή # 3

Συνδεσμολογία του cc3000

Η CC3000 συνοδεύεται με μια λωρίδα από καρφίτσες τις οποίες τις ξεχωρίζουμε σε τμήματα των 6,8 και 10 έτσι ώστε να χωρέσουν ακριβώς στις υποδοχές του Arduino μας.Τοποθετήσαμε το προστατευτικό κάλυμμα πάνω από τα pin, και κολλήσαμε προσεκτικά κάθε μία στη θέση του



ΕΙΚΟΝΑ 3 CC3000 ΚΑΙ ARDUINO

Η ασπίδα είχε επίσης μια υποδοχή θηλυκή των 2X3 pin. Αυτό το συνδέσαμε στο pin header 2X3 ICSP στο Arduino μας, για να φέρει SPI πάνω από το shield μας. Κόλληση καρφιτσών χρειάζεται να βάλουμε σύρματα συγκόλλησης για την SPI. Τοποθετήσαμε την υποδοχή στις οπές στο shield, στη συνέχεια, γυρίσαμε την ασπίδα ανάποδα και κολλήσαμε τις καρφίτσες.Από προεπιλογή το SPI δεν είναι συνδεδεμένο με την κεφαλίδα ICSP. Για να ενεργοποιήσουμε τις συνδέσεις κεφαλίδας ICSP, κάναμε τρεις κολλήσεις στο κάτω μέρος της CC3000.Οι συγκολλήσεις αυτές είναι απαραίτητες επειδή χρησιμοποιούμε το shield με το Mega αλλά δεν απαιτείται για χρήση με ένα UNO!Απλά λιώσαμε μια άμορφη μάζα της συγκόλλησης σε κάθε μία από τα 3 κουτάκια χωρίς να βγαίνουν εκτός των ορίων Μετά από αυτό το cc3000 είναι έτοιμο για χρήση και τοποθέτηση πάνω στο Arduino Mega που έχουμε

Στην μονάδα cc3000 χρησιμοποιούμε τα παρακάτω Pin

- SCK - #13
- MISO #12
- MOSI #11
- CS for CC3000 #10
- VBAT_EN #5
- CS for SD Card #4
- IRQ #3

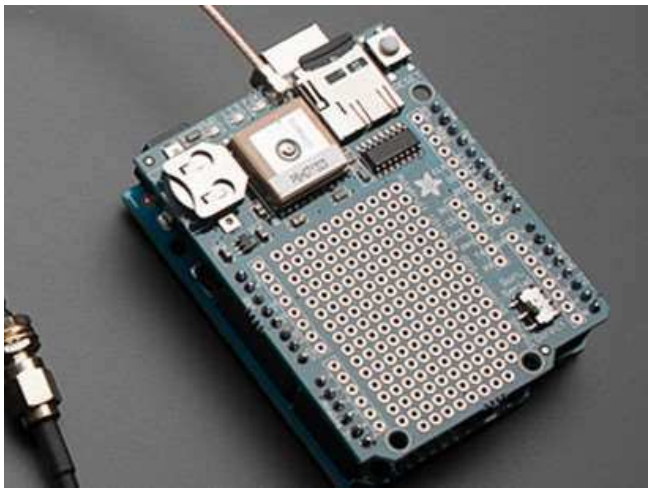
1.2.4 Προαιρετική Κεραία

Δεδομένου ότι το σύστημά μας είναι σε κουτί τοποθετήσαμε μια εξωτερική κεραία 2.4Ghz έτσι ώστε να έχουμε πιο δυνατή κεραία . Για να γίνει εφικτή η σύνδεση αυτή χρειάζεται και ένας προσαρμογέας (adaptor) UFL σε RP-SMA ή UFL σε SMA.

1.2.5 κάρτα μνήμης SD

ΕΙΚΟΝΑ 4 ΚΑΡΤΑ ΜΝΗΜΗΣ SD

Το παρόν εξάρτημα είναι κάρτα μνήμης η οποία έχει τοποθετηθεί σε ειδική θήκη πάνω στην πλακέτα Adafruit cc3000 όπως φαίνεται παρακάτω. Χρησιμοποιείται είτε για καταγραφή είτε για ανάγνωση πληροφοριών.



ΕΙΚΟΝΑ 5 ΤΟΠΟΘΕΤΗΜΕΝΗ ΚΑΡΤΑ ΜΝΗΜΗΣ SD ΠΑΝΩ ΣΤΟ CC3000

1.2.7 Wifi

Το Wifi το χρησιμοποιούμε για να συνδεθούμε ασύρματα στο διαδίκτυο. Αυτό το κάνουμε για να γίνει εφικτή η αποστολή των δεδομένων μας στη βάση.

Το Wifi είναι μια τεχνολογία για συσκευές οι οποίες βασίζονται στα IEEE 802.11 δεδομένα. Οι συσκευές που μπορούν να το χρησιμοποιήσουν είναι οι υπολογιστές , κινητά , tablets κ.α. . Οι συσκευές αυτές που είναι συμβώατες με τη τεχνολογία αυτή μπορούν να συνδεθούν στο διαδίκτυο μέσω WLAN και μέσω ενός wireless access point εύρους περίπου 20 μέτρων σε εσωτερικό χώρο και λίγο περισσότερο σε εξωτερικό. Το Wifi κυμαίνεται μεταξύ των συχνοτήτων των 2.4 και 5.8 gigahertz. Ο καθένας μπορεί να αποκτήσει πρόσβαση στο διαδίκτυο χρησιμοποιώντας modem βρισκόμενος σε αυτές τις συχνότητες. Εξαιτίας αυτού το ασύρματο δίκτυο είναι πιο ευάλωτο σε επιθέσεις

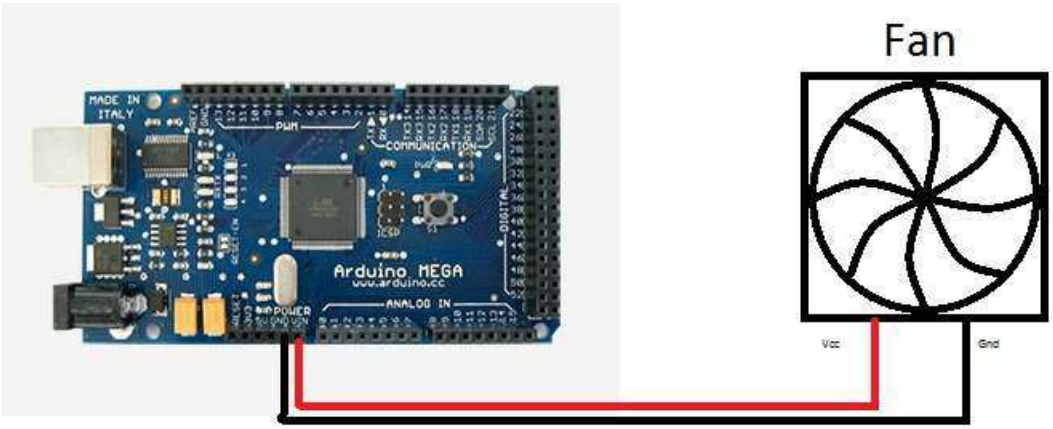
1.3 Brushless Cooling Fan

ΕΙΚΟΝΑ 6 ΑΝΕΜΙΣΤΗΡΑΣ

1.3.1 Επισκόπηση

Ο ανεμιστήρας είναι ένα εξάρτημα το οποίο το χρησιμοποιούμε για ψύξη. Αποτελείται από πτερύγια ενωμένα σε κυκλική μορφή γνωστά ως δρομέας. Λειτουργεί με τροφοδοσία που κυμαίνεται γύρω στα 12V. Με 0.12A και με ταχύτητα της φτερωτής να φτάνει τα 2500rpm (γύρους ανά λεπτό)

Ακολουθεί κύκλωμα:



ΕΙΚΟΝΑ 7 ΣΥΝΔΕΣΜΟΛΟΓΙΑ ΑΝΕΜΙΣΤΗΡΑ ΣΤΟ ARDUINO MEGA

1.4 Αισθητήρας υπερήχων HC-SR04

ΕΙΚΟΝΑ 8 HC-SR04

1.4.1 Επισκόπηση

Ο hc-sr04 είναι ένας αισθητήρας ο οποίος χρησιμοποιεί ραντάρ απόστασης για να μετρήσει απόσταση αναμεσά σε αυτό και σε ένα άλλο αντικείμενο οποιοδήποτε και αν είναι αυτό. Προσφέρει έναν εξαιρετικό ανιχνευτή με πολύ υψηλή ακρίβεια και σταθερές μετρήσεις σε ήλιο η σε σκοτάδι. Οι αποστάσεις τις οποίες μπορεί να μετρήσει κυμαίνονται από 2 εκατοστά του μέτρου μέχρι και 4 μέτρα. Αποτελείται από έναν πομπό και έναν δέκτη.

1.4.2 Τεχνικά χαρακτηριστικά

- Power Supply: +5V DC
- Quiescent Current: <2mA
- Working Current: 15mA
- Effectual Angle: <15°
- Ranging Distance: 2cm – 400 cm/1" - 13ft
- Resolution: 0.3 cm
- Measuring Angle: 30 degrees
- Trigger Input Pulse width: 10uS
- Dimension: 45mm x 20mm x 15mm

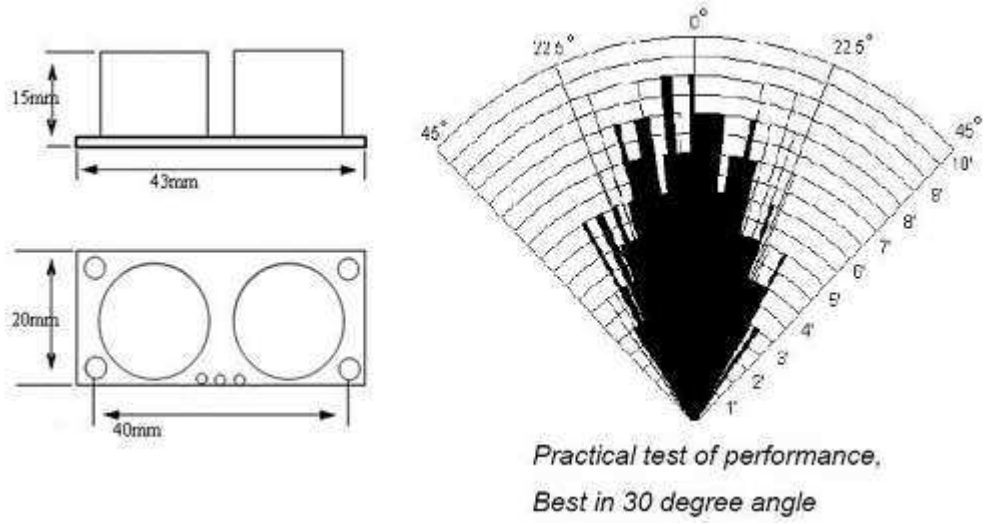
1.4.3 Συνδεσμολογία

Ο αισθητήρας έχει 4 ακροδέκτες. Οι ακροδέκτες είναι οι εξής:

1) VCC= +5VDC
2) Trig=trigger input of sensor
3) Echo= echo output of sensor
4) Gnd=Gnd

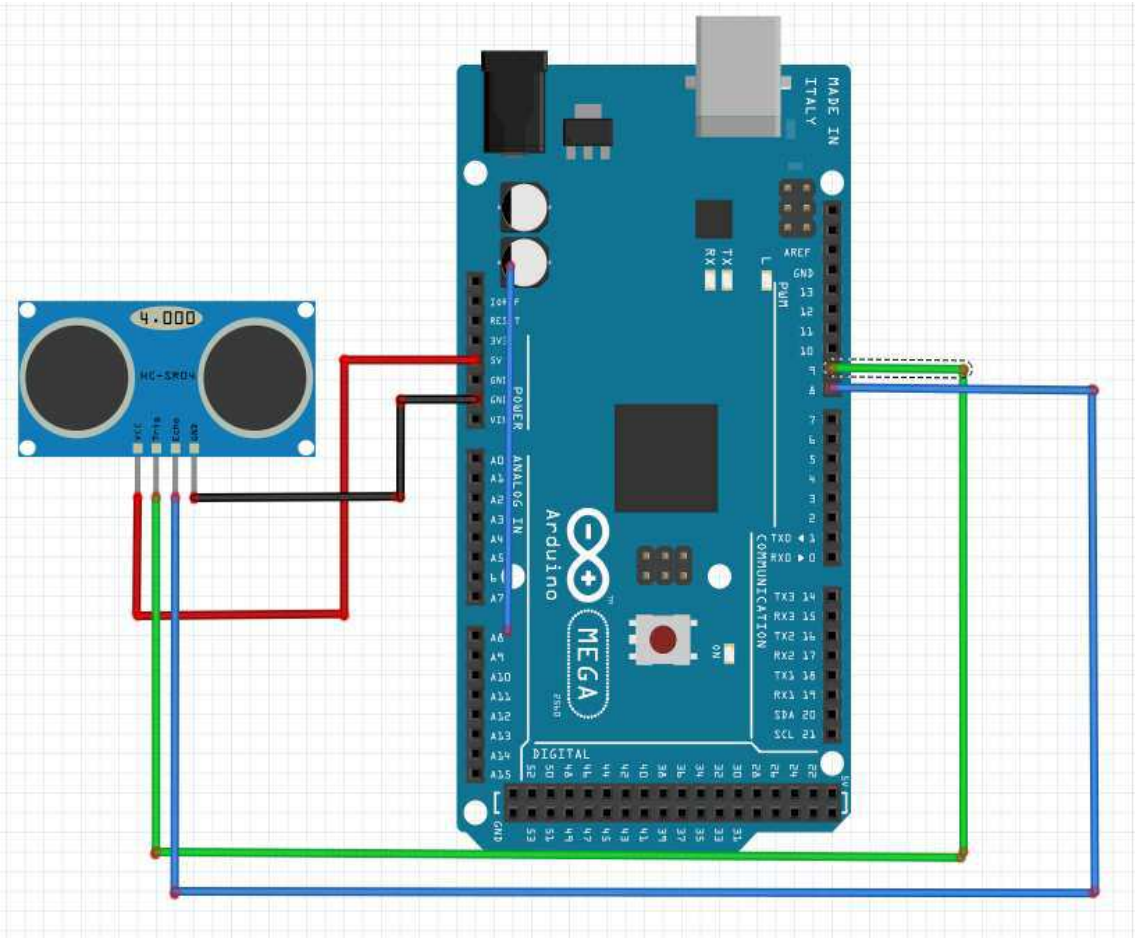
1.4.4 Λειτουργία

Για να αρχίσει η λειτουργία του, για να ξεκινήσει δηλαδή τις μετρήσεις το VCC πρέπει να δεχτεί παλμό high 5v το λιγότερο για 10us. Αυτό θα προκαλέσει τον σένσορα να μεταδώσει 8 "κύκλους" από υπέρηχους των 40khz και θα περιμένει την αντανάκλαση αυτών. Όταν ο σένσορας ανιχνεύσει τους υπερήχους από τον δέκτη αυτό θα θέσει το pin echo σε high 5v και θα καθυστερήσει για μια περίπου περίοδο ανάλογη της απόστασης. Για να κατανοήσουμε την απόσταση μετράμε το εύρος του echo pin.



ΕΙΚΟΝΑ 9 ΔΙΑΣΤΑΣΕΙΣ ΚΑΙ ΠΕΡΙΟΧΗ ΜΕΤΡΗΣΕΩΝ HC-SR04

Ακολουθεί απεικόνιση σύνδεσης:



ΕΙΚΟΝΑ 10 ΣΥΝΔΕΣΜΟΛΟΓΙΑ HC-SR04 ΜΕ ARDUINO MEGA

Παρατηρούμε ότι έχουμε τέσσερεις συνδέσεις. Το Vcc του αισθητήρα συνδέεται στο +5V , το trig στο 9, το echo στο 8 (ψηφιακά) και η γείωση στη γείωση

1.5 Seven Segment Display

ΕΙΚΟΝΑ 11 SEVEN SEGMENT DISPLAY

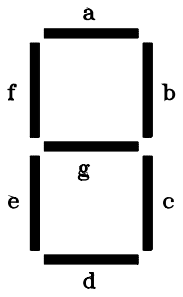
1.5.1 Επισκόπηση

Το εξάρτημα αυτό είναι μια φόρμα ηλεκτρονικής αναπαράστασης αριθμών. Χρησιμοποιείται ευρέως στα ψηφιακά ρολόγια για να παρέχει αριθμητική πληροφορία. Όπως καταλαβαίνουμε και από το όνομα του, αποτελείται από εφτά στοιχεία τα οποία συνδυάζονται μεταξύ τους και με λειτουργία on off (των στοιχείων) απεικονίζουν τα νούμερα.

Οι ενδείκτες δεκαδικών ψηφίων χρησιμοποιούν 7 τμήματα για να αναπαραστήσουν τους δεκαδικούς αριθμούς

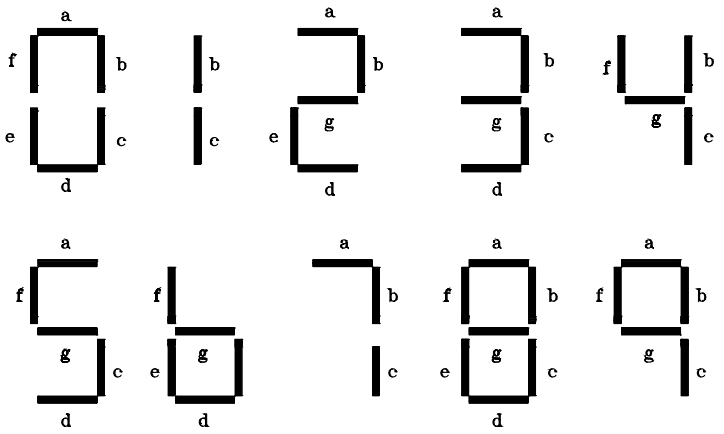
Υπάρχουν ενδείκτες όπου χρησιμοποιούνται οι δίοδοι εκπομπής φωτός (Light Emitting Diodes - LEDs) για την κατασκευή των τμημάτων τους. Η λειτουργία τους βασίζεται στο γεγονός ότι κάθε τμήμα αποτελείται από υλικό το οποίο εκπέμπει φως όταν διαρρέετε από ρεύμα.

Επίσης, υπάρχουν ενδείκτες υγρού κρυστάλλου (Liquide Crystal Displays - LCDs). Η λειτουργία τους βασίζεται στην ιδιότητα ενός ειδικού υγρού κρυστάλλου να διαδίδει διαφορετικά το φως υπό την επίδραση εναλλασσόμενου ηλεκτρικού πεδίου. Τα LCDs έχουν ιδιαίτερα χαμηλή κατανάλωση ισχύος και είναι ιδανικά για φορητές συσκευές.



ΕΙΚΟΝΑ 12 ΤΑ 7 ΤΜΗΜΑΤΑ ΤΟΥ SSD

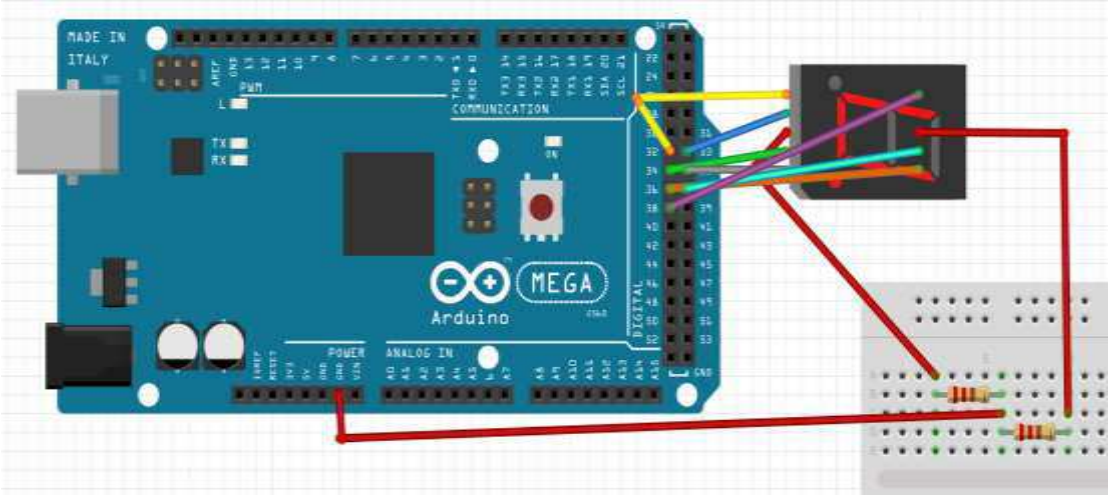
Οι δεκαδικοί αριθμοί σχηματίζονται όταν ανάβουν κάποια από τα τμήματα του ενδείκτη δεκαδικών ψηφίων (display). Παρακάτω παρουσιάζεται ο τρόπος εμφάνισης των δεκαδικών ψηφίων 0-9 στον ενδείκτη δεκαδικών ψηφίων (display).



ΕΙΚΟΝΑ 13 ΤΡΟΠΟΣ ΕΜΦΑΝΙΣΗΣ ΔΕΚΑΔΙΚΩΝ ΨΗΦΙΩΝ ΣΤΟΝ SSD

Για παράδειγμα για τον αριθμό για να ενεργοποιηθεί και να εμφανιστεί ο αριθμός 4 θα πρέπει να ενεργοποιηθούν τα b,c,g,f κ.ο.κ.

Ακολουθεί η σύνδεση:



ΕΙΚΟΝΑ 14 ΣΥΝΔΕΣΜΟΛΟΓΙΑ SSD ΜΕ ARDUINO

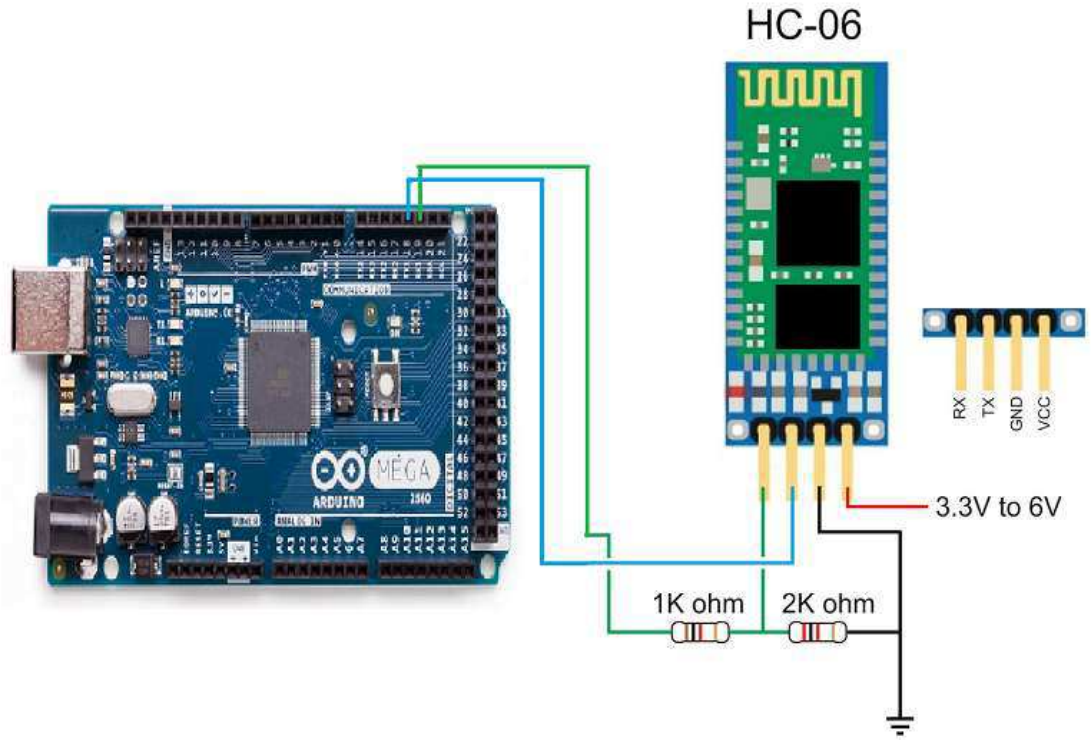
1.6 Bluetooth HC-05

ΕΙΚΟΝΑ 15 HC-05

1.6.1 Επισκόπηση

Το συγκεκριμένο εξάρτημα είναι ένα Bluetooth. Το Bluetooth είναι μια ασύρματη τεχνολογία η οποία χρησιμοποιείται για την ανταλλαγή δεδομένων σε μικρές αποστάσεις με μέγιστο περίπου τα 10 μέτρα. Χρησιμοποιεί UHF ραδιοκύματα μεταξύ των 2.4 και 2.485 GHz

Ακολουθεί κύκλωμα :



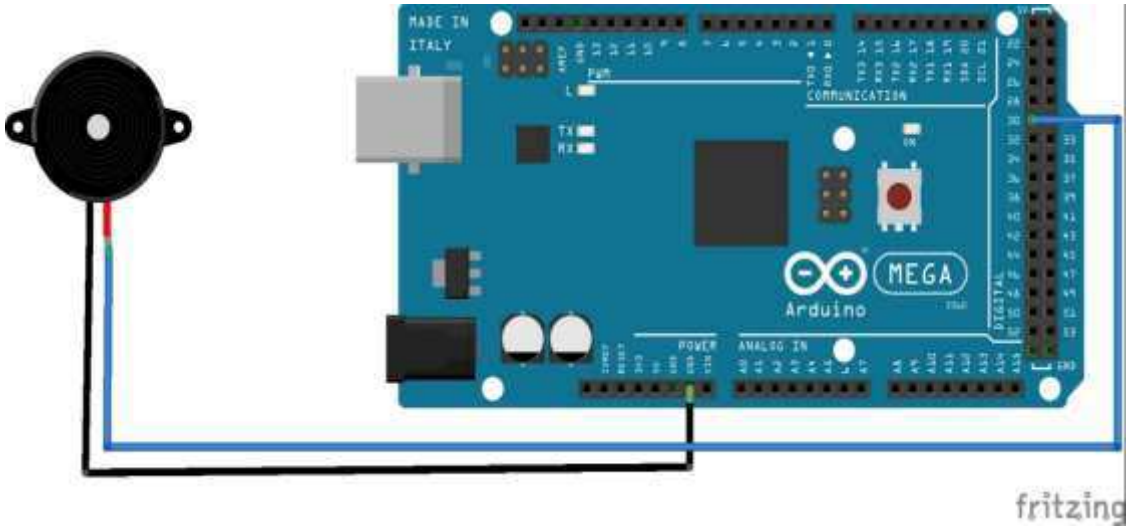
ΕΙΚΟΝΑ 16 ΣΥΝΔΕΣΜΟΛΟΓΙΑ HC-05 ΜΕ ARDUINO

1.7 buzzer

ΕΙΚΟΝΑ 17 BUZZER

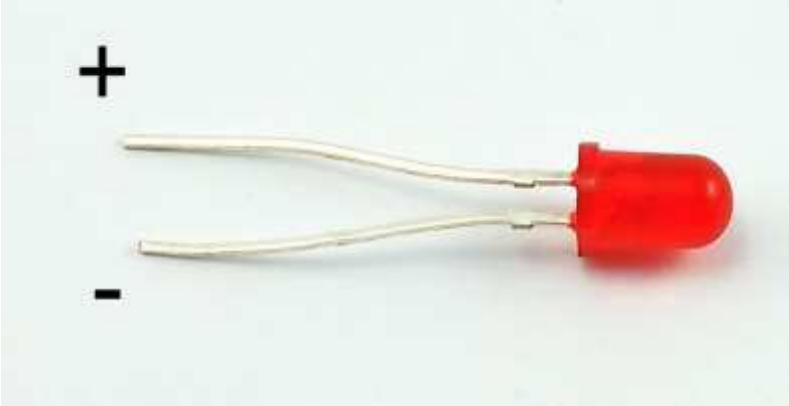
1.7.1 επισκόπηση

Το παρόν εξάρτημα είναι διαμέτρου 12mm και λειτουργεί σε εύρος των 2khz. Παράγει έναν τυπικό ήχο των 95dba και τροφοδοτείται από 3.5 έως 5 volt.



ΕΙΚΟΝΑ 18 ΣΥΝΔΕΣΜΟΛΟΓΙΑ BUZZER ΜΕ ARDUINO

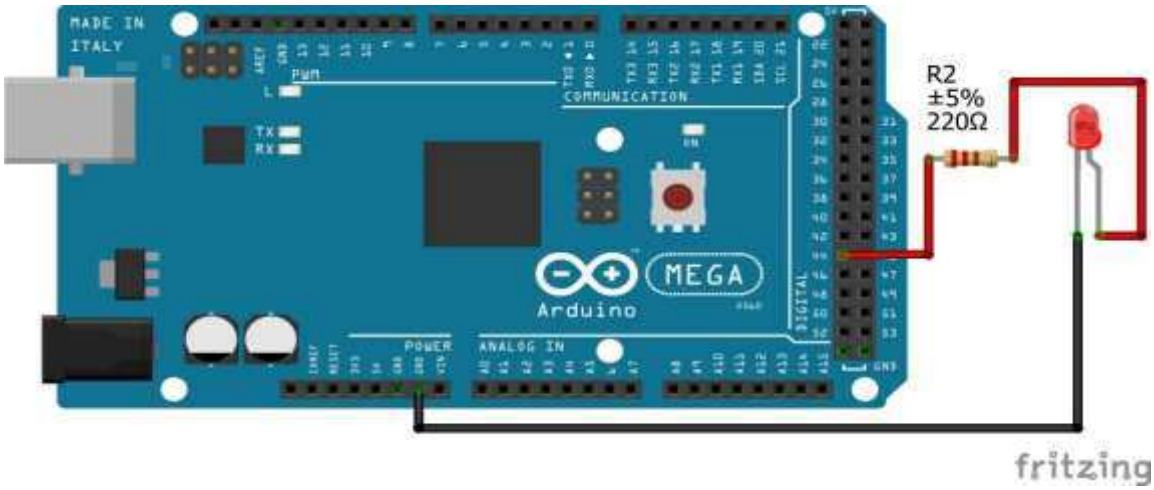
1.8 LED



ΕΙΚΟΝΑ 19 LED

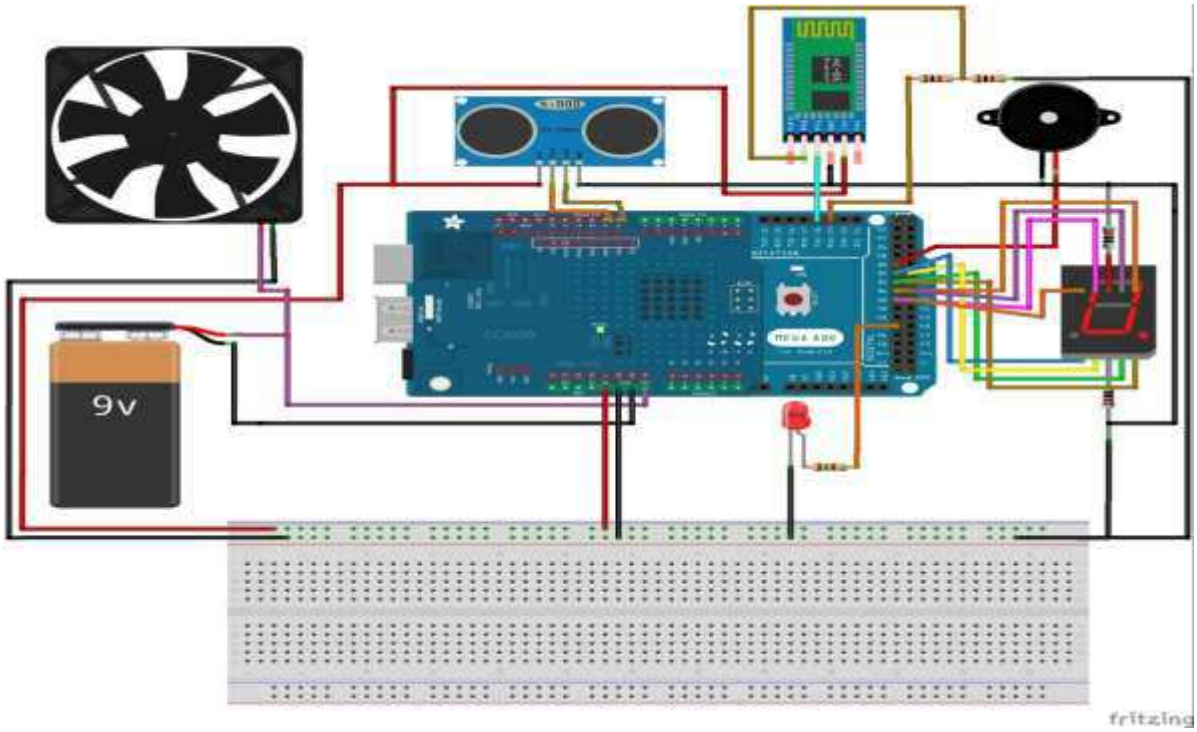
1.8.1 Επισκόπηση

Το παρόν στοιχείο είναι ένα led τύπου διόδου (άνοδος και κάθοδος) και τροφοδοτείται από 1.8 έως 2.4 βολτ



ΕΙΚΟΝΑ 20 ΣΥΝΔΕΣΜΟΛΟΓΙΑ LED ΜΕ ARDUINO

1.9 Ολοκληρωμένο κύκλωμα



ΕΙΚΟΝΑ 21 ΣΥΝΔΕΣΜΟΛΟΓΙΑ ΟΛΩΝ ΤΟΝ ΕΞΑΡΤΗΜΑΤΩΝ ΜΕ ΤΟ ARDUINO

ΚΕΦΑΛΑΙΟ 2
Software

2.1 Server , Γλώσσες και βάση δεδομένων

2.1.1 Server και βάση δεδομένων

Ο server η διακομιστής σε μια αρκετά απλή μορφή είναι ένας υπολογιστής ο οποίος τρέχει το κατάλληλο λογισμικό και σκοπός του είναι να εξυπηρετεί τους πελάτες του που συνδέονται σε αυτόν για έναν συγκεκριμένο λόγο. Ανάλογα λοιπόν με αυτό τον λόγο ο διακομιστής τρέχει τις αντίστοιχες υπηρεσίες και έχει και κατάλληλη ονομασία. Για παράδειγμα αν ο server εξυπηρετεί ιστοσελίδες τότε ονομάζεται web server. Η επικοινωνία μεταξύ του server και του client μπορεί να γίνει μέσω ενός local network η μέσω του διαδικτύου. Σε δίκτυα στα οποία ο διακομιστής εξυπηρετεί μεγάλο αριθμό πελατών χρησιμοποιούνται υπολογιστές οι οποίοι έχουν παραπάνω δυνατότητες από τους κοινούς υπολογιστές. Έχουν ανεβασμένα χαρακτηριστικά ώστε να παρέχουν μεγάλη αξιοπιστία και σιγουριά για τις υπηρεσίες που παρέχουν. Οι υπολογιστές αυτοί μπορούν και στις περισσότερες φορές πρέπει να τρέχουν όλη την ημέρα χωρίς διακοπή.

Ως Βάση Δεδομένων στην ουσία καλούμε τη συλλογή δεδομένων τα οποία είναι αποθηκευμένα σε έναν υπολογιστή και μπορούν να επεξεργαστούν από το άτομο που τα διαχειρίζεται. Η διαχείριση αυτή γίνεται μέσω του Συστήματος Διαχείρισης Βάσεων Δεδομένων. Οι βάσεις δεδομένων έχουν δημιουργηθεί για να διαχειρίζονται μεγάλο μέγεθος πληροφοριών που τα εισάγεις τα αποθηκεύεις τα τράβας και τα διαχειρίζεσαι. Συνήθως είναι σε πίνακες . Αποτελούνται από σειρές και στήλες. Κάθε στοιχείο που εισάγεται σε μια σειρά αποτελεί καταγραφή. Όταν καταγράφουν οι μετρήσεις και ολοκληρωθεί η βάση τα στοιχεία αυτά μπορούν να χρησιμοποιηθούν με πολλούς διαφορετικούς τρόπους, να τα διαχειριστούν ανάλογα με το λογισμικό που χρησιμοποιούμε και τις δυνατότητες που μας παρέχει.

2.1.2 Server Wamp

Το WAMP (Windows Apache , MySQL , PHP)είναι ένα πρόγραμμα το οποίο το χρησιμοποιήσαμε για να δημιουργήσουμε έναν τοπικό server στον υπολογιστή. Συνήθως

χρησιμοποιείται για ανάπτυξη εφαρμογών και απομακρυσμένο έλεγχο αλλά όπως αναφέραμε παραπάνω και για τη δημιουργία εξυπηρετητή.

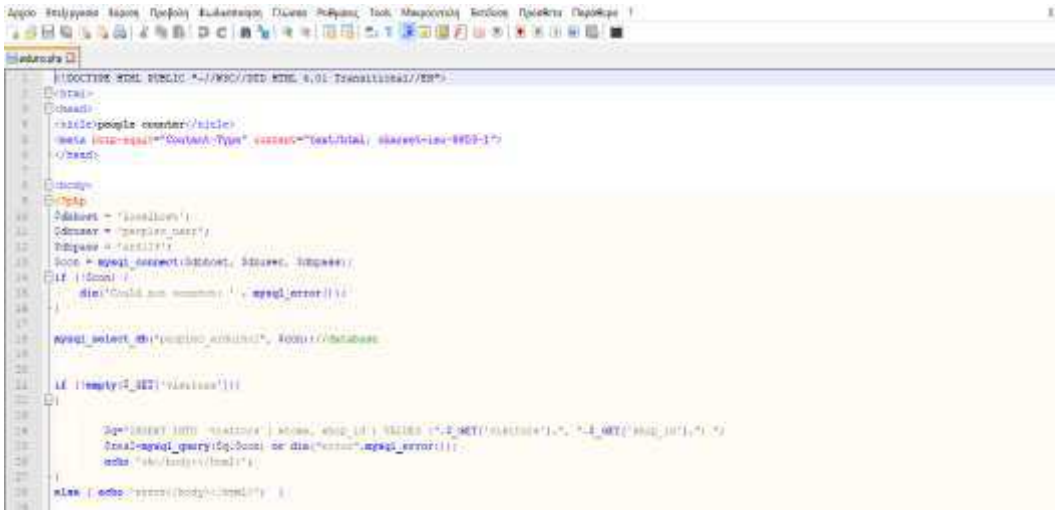
Το σημαντικότερο κομμάτι αυτού του προγράμματος είναι το Apache το οποίο είναι απαραίτητο για να τρέξει έναν web server μέσα στα windows. Τρέχοντας έναν τοπικό Apache εξυπηρετητή μέσα σε λογισμικό windows μπορούμε να τεστάρουμε ιστοσελίδες σε έναν πλοηγό χωρίς να τις δημοσιοποιήσουμε.

Επίσης το πρόγραμμα αυτό περιλαμβάνει και MySQL και PHP τα οποία είναι δυο από τις πιο κοινές λειτουργίες στη δημιουργία δυναμικών ιστοσελίδων. Το MySQL είναι μια ταχύτατη βάση δεδομένων ενώ το PHP είναι μια γλώσσα η οποία χρησιμοποιείται για να έχουμε πρόσβαση σε δεδομένα από τη βάση. Αν εγκαταστήσουμε αυτές τα δυο στοιχεία τοπικά στον υπολογιστή μας μπορούμε να φτιάξουμε και να τεστάρουμε μια δυναμική ιστοσελίδα πριν την ανεβάσουμε σε έναν δημόσιο web server.

ΕΙΚΟΝΑ 22 ΑΡΧΙΚΗ ΣΕΛΙΔΑ ΤΟΥ ΔΙΑΚΟΜΙΣΤΗ ΜΑΣ

2.1.3 Notepad ++

Πρόκειται για ένα πρόγραμμα το οποίο χρησιμοποιείται ευρέως από τους προγραμματιστές. Το γνωστό text editor. Είναι ανοιχτού λογισμικού , ελαφρύ αλλά με πολλές δυνατότητες και είναι εξαιρετικά απλό. Στα της δομής, περιέχει μια γραμμή εργαλείων που περιέχει τις κυριότερες λειτουργίες του. Ο υπόλοιπος χώρος αποτελεί τον χώρο συγγραφής κώδικα. Γύρω από αυτό εμφανίζονται διαφορά στοιχεία τα οποία μας παρέχουν διάφορες πληροφορίες. Ένα κύριο χαρακτηριστικό είναι ο χρωματισμός του κώδικα ανάλογα με τη δομή ώστε να γίνεται πιο εύκολη η ανάπτυξη του και ανεύρεση λαθών. Σε κάθε γλώσσα διαθέτει ξεχωριστό χρώμα το οποίο είναι έτοιμο απ το πρόγραμμα αλλά και διαχειριστικό από τον χρήστη. Στην συγκεκριμένη εργασία το χρησιμοποιήσαμε για να συγγράψουμε script σε pup, html,MySQL.Παρακάτω αναφέρονται αναλυτικά.



ΕΙΚΟΝΑ 23 ΠΕΡΙΒΑΛΛΟΝ ΤΟΥ NOTEPAD++

2.1.4 βάση δεδομένων MySQL

Η MySQL είναι μια βάση δεδομένων η οποία μπορεί να χρησιμοποιηθεί σε μικρές και σε μεγάλες επιχειρήσεις. Μερικά από τα χαρακτηριστικά της είναι τα εξής:

- Διατίθεται με δίπλωμα ανοιχτού κώδικα αρά δεν χρειάζεται πληρωμή
- Είναι μια ισχυρή βάση και μπορεί να διαχειριστή μεγάλο μέγεθος δεδομένων
- Χρησιμοποιεί SQL data language
- Είναι φιλική με την PHP , τη πιο διαδεδομένη γλώσσα για web developing

2.2 Γλώσσες προγραμματισμού

2.2.1 Η γλώσσα C#

Η c# είναι μια αντικειμενοστραφείς γλώσσα προγραμματισμού η οποία περιλαμβάνει δηλώσεις ,συναρτήσεις ,προστακτικές αρχές τα οποία είναι βασισμένες σε κλάσεις

Η γλώσσα αυτή δημιουργήθηκε από την Microsoft μέσα από την πλατφόρμα .NET και αργότερα πιστοποιήθηκε σαν πρότυπο από τους οργανισμούς Ecma (ECMA-334) και ISO (ISO/IEC 2327:2006).

Με ομάδα υπό την ηγεσία του Anders Hejlsberg και με την τελευταία έκδοση της που δημοσιεύτηκε στις 15 Αυγούστου του 2012 να είναι η 5.0 έχει σαν στόχο να είναι μια εξυγχρωνισμένη αντικειμενοστραφής γενικού σκοπού γλώσσα προγραμματισμού

Την ονομασία της την εμπνευστήκαν από την μουσική και μοιάζει αρκετά με την ονομασία της γλώσσας προγραμματισμού c++ η οποία στην ουσία εμφαίνει την αύξηση μιας τυχαίας μεταβλητής κατά 1 . Εξ ου και το σύμβολο «++»

Η σύνταξη της γλώσσας αυτής έχει αρκετά κοινά στοιχεία με τις γλώσσες c και c++ και java.

Πιο αναλυτικά:

- κάθε φορά που κάνουμε δήλωση, στο τέλος αυτής χρησιμοποιούμε ελληνικό ερωτηματικό

- οι δηλώσεις σχεδόν πάντα βρίσκονται μέσα σε συναρτήσεις οι οποίες βρίσκονται μέσα σε κολάσεις και τέλος μέσα σε namespaces. Για να λειτουργήσει αυτό χρειάζεται αγκύλες

- οι μεταβλητές λαμβάνουν τιμές όταν στη δήλωση υπάρχει το =

για την δήλωση πινάκων χρησιμοποιούνται τετραγωνικές αγκύλες

Παρακάτω ακολουθεί ένα απλό παράδειγμα προγράμματος που απλά θα εμφανίζει στην οθόνη το μήνυμα “hello World”.

```
1. using System;
2. namespace MyNamespace
4. {
5.    class Helloworld
6.    {
7.     static void Main()
a.     {
i.      Console.WriteLine("Hello World");
ii.       Console.ReadKey();
b.     }
8.    }
9.   }
```

Πιο αναλυτικά

1) NAMESPACES

Τα namespaces στην c# τα εμπεριέχουμε στον κώδικα μας για την οργάνωση του. Όπως βλέπουμε στο παραπάνω παράδειγμα κάναμε χρήση του system μέσω του using το οποίο περιέχει κλάσεις μεθόδους και συναρτήσεις με ολοκληρωμένους κώδικες από το .net.

Οι χώροι ονομάτων μπορούν να χρησιμοποιήσουν τα εξής :

• Namespaces • Classes • Delegates • Enums • Structs • Interfaces

Για να παράγουμε το δικό μας namespace κάνουμε χρήση του using και έτσι η "hello world" class συμπεριλαμβάνεται στο mynamespace που εμείς φτιάξαμε .όπως βλέπουμε στο κώδικα κάνουμε χρήση του console.writeline το οποίο εμπεριέχεται στη main() (βλέπε γραμμή 6 του κώδικα) για να εμφανίσουμε στην οθόνη μας το "hello world".

Είναι η εντολή την οποία χρησιμοποιούμε για να εμφανίζουμε ένα οποιοδήποτε αλφαριθμητικό μήνυμα στην σειριακή μας οθόνη.

2) Variables Statements (δήλωση μεταβλητών)

Στην c# για να μπορέσουμε να κάνουμε χρήση οποιασδήποτε μεταβλητής και να γίνει πράξη με αυτή πρέπει πρώτα να τη δηλώσουμε οπουδήποτε μέσα στο πρόγραμμα .

Για την δήλωση της αρχικά καθορίζουμε τον τύπο και έπειτα την ονομασία της

πχ:int y; float z;

3) Ανάθεση τιμών

Για να ορίσουμε τιμή σε μεταβλητή κάνουμε χρήση το σύμβολο ισότητας(=)

πχ: y=10; αλλιώς μπορούμε να το γράψουμε και ως εξής αν θέλαμε να γίνει και παράλληλα η δήλωση της μεταβλητής : Int y=10;

Παρατήρηση

Η γλώσσα που χρησιμοποιούμε κάνει διάκριση αναμεσά σε πεζά και κεφάλαια γράμματα. Για να δηλώσουμε την Main() αναγκαστικά την γράφουμε με κεφάλαιο "Μ" διότι ο μεταγλωττιστής αναγνωρίζει ότι το πρόγραμμα βρίσκεται στην μέθοδο Main().

4) Λέξεις-Κλειδιά

Στη c# υπάρχουν κάποιες λέξεις οι οποίες είναι δεσμευμένες και δεν είναι εφικτό να γίνουν χρήση αυτών σαν μεταβλητές.

5) ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ

a) integers (ακέραιοι)

Οι ακέραιες μεταβλητές χωρίζονται ανάλογα με το μέγεθος τους . Αναφορικά έχουμε τους εξής :

1. sbyte με μέγεθος 1byte
2. byte με μέγεθος 1byte
3. short με μέγεθος 2byte
4. unshort με μέγεθος 2byte
5. int με μέγεθος 4byte
6. uint με μέγεθος 4byte

7. long με μέγεθος 8byte

8. ulong με μέγεθος 8byte

1. Παράδειγμα (Byte):

```
static void Main()
      {
         byte mynumber1 = 254;
         Console.WriteLine(mynumber1); //Εμφανίζει 254
         mynumber1++;
          // Συν 1 -> 255
         Console.WriteLine(mynumber1); //Εμφανίζει 255
         mynumber1++;
          // Συν 1 -> 0 (όριο 255)
         Console.WriteLine(mynumber1); //Εμφανίζει 0
         mynumber1++;
         // Συν 1 -> 1
         Console.WriteLine(mynumber1); //Εμφανίζει 1
      }
```

2. Παράδειγμα (int)

Με Int

```
 static void Main()
  //Σε περίπτωση που Δηλώσουμε 2 ακεραίους INT
      {
```

```
    //Δηλώνουμε 2 ακεραίους INT
    int A = 10;
    int B = 3;
    //διαιρούμε 10/3
    int apotelesma = A/B;
    // Μας εμφανίζει 3 και Οχι 3,333...
    Console.WriteLine(apotelesma);
  }
```

b) Μεταβλητές float

Ο τύπος αυτός μας επιτρέπει την αλλαγή της μεταβλητής κατά την διάρκεια εκτέλεσης του κώδικα η οποία πρέπει αρχικά να δηλώνεται και να της δοθεί μια τιμή και να αποθηκευτεί.

3. Παράδειγμα (Float)

```
static void Main()
 {
   float apostasi;
   float xronos;
   float taxitita;
   apostasi = 0.1f; // 100m είναι 0.1 km
   // για 9.87 δευτερόλεπτα είναι 9.87/60*60 h
   xronos = 9.87f / 3600;
   taxitita = apostasi / xronos;
   Console.WriteLine("Η μέση ταχύτητα είναι {0} km/h", taxitita);  }
```

c) STRINGS/CHARS

Τα strings γνωστά και ως συμβολοσειρές αποτελούν μια σειρά από αλφαριθμητικούς χαρακτήρες. Με τον ορό σειρά αναφερόμαστε στης κατά σειρά θέσεις μνήμες που αντιμετωπίζονται ως ένα σύνολο διατεταγμένων στοιχειωδών δεδομένων.

Η συμβολοσειρά αυτή χρησιμοποιεί δεδομένα τύπου char .

Πχ

```
static void Main()
  {
   //το A = StudentGuru
   string A = "StudentGuru";
   //B = με το πρώτο χαρακτήρα της μεταβλητής A άρα = "S"
   char B = A[0];
   Console.WriteLine(B); // εκτυπώνει στην κονσόλα.
   Console.WriteLine(A); // εκτυπώνει στην κονσόλα.
   //Τώρα το B = με το τρίτο χαρακτήρα της μεταβλητής A άρα ="u"
   B=A[2];
   Console.WriteLine(B); // εκτυπώνει στην κονσόλα.
  }
```

d) BOOL

Η κατηγορία αυτή παίρνει τιμές .

1. True(αληθές)
2. false(ψευδές)

πχ

```
static void Main()
  {
    bool CheckNumber;
    Console.WriteLine("Typose studentguru");
    string  num = Console.ReadLine();
     //εάν η μεταβλητή = studentguru
      if (num =="studentguru")
      {
        CheckNumber = true; //τότε το checkNumber = true
      }
      else
      {
        CheckNumber = false;//διαφορετικά checkNumber = false
      }
    Console.WriteLine(CheckNumber);
  }
```

e) NULL

Σε αντίθεση με την αγγλική γλώσσα που σημαίνει μηδέν στην προγραμματιστική σημαίνει <<απροσδιόριστο>>.

Ο τύπος αυτός είναι ιδιαίτερα σημαντικός διότι μας δίνει την δυνατότητα να κάνουμε χρήση σε οποιοδήποτε σημείο του προγράμματος μιας μεταβλητής που μπορεί να έχει αξίες

1. Παράδειγμα σύνταξη:

```
1. char? letter = 'a';
2. int? i = null;
3. int? i = 10;
4. double? d1 = 3.14;
5. bool? flag = null;
6. int?[] arr = new int?[10];
```

Το "var" δεν είναι συμβατό με αυτό Nullable Type:

```
var? i = null; //Λάθος
```

Η σωστή σύνταξη για το "var" είναι:

```
var i = 4;
```

2. Παράδειγμα Nullable types

```
static void Main(string[] args)
  {
    int? num = null; //Nullable Types Σύνταξη
     //Ελεγχος εάν η μεταβλητή num έχει τιμή
     if (num.HasValue == true)
       //αν ναι τύπωσε μας την τιμή
       System.Console.WriteLine(num.Value);
     Else
       //αν όχι τύπωσε NULL
```

```
        System.Console.WriteLine("NULL");
    }
```

3. Άμεση Μετατροπή

```
int? n = null;
//int m1 = n;      // Λάθος
int m2 = (int)n;
int m3 = n.Value;
```

4. Έμμεση μετατροπή

```
int? n1 = null;
int? n2;
n2 = 10;  //Εμμεση μετατροπή
```

5. “??” Τελεστές Nullable

```
int? c = null;
// d = c, και το c είναι null, το d = -1.
int d = c ?? -1;
int? e = null;
int? f = null;
// g = e ή και f, εκτός εάν το e και το f = null, που για κάθε περίπτωση g = -1.
int g = e ?? f ?? -1;
bool? b = null;
if (b) // Λάθος CS0266
{
}
```

f) ΤΥΠΟΣ VAR

Η μεταβλητή αυτή είναι αρκετά εύχρηστη διότι ουσιαστικά καταλαμβάνουμε μια θέση μνήμης την οποία μπορούμε να την γεμίσουμε με δεδομένα οποιουδήποτε τύπου και ο μεταγλωττιστής να καθορίσει τον τύπο για μας .

Παράδειγμα Var Types

Normal 0 false false false EL X-NONE X-NONE

Normal 0 false false false EL X-NONE X-NONE

Παράδειγμα:

```
static void Main()
{
  var leksi = “studentGuru”;
  var Arithmos = 21;
  var MegArithmo = 219283746253;
  var dekadiko = 2,19283746253;
}
```

Παρόμοια κατάσταση, αν έχετε μια απλή κλάση με το όνομα Person:

```
public class Person
{
  public string Name { get; set; }
  public int Age { get; set; }
}
static void Main()
```

```
{
  var person = new Person();
  person.Age = 21;
  person.Name = “StudentGuru”;
}
```

6) Τελεστές

Οι τελεστές είναι διάφορα σύμβολα τα οποία μας είναι χρήσιμα για να εκτελέσουμε διάφορες διεργασίες στο πρόγραμμα μας μια η περισσότερες φορές. Μπορούν να είναι σταθερές ,μεταβλητές κλάσεις ακόμα και συναρτήσεις.

Σύμβολα τελεστών

Arithmetic (Αριθμητική)

+ - * / %

Logical (Λογική)

& | ^ ! ~ && ||

Increment, decrement (Αύξηση, μείωση)

-- ++

Shift (Αλλαγή)

<< >>

Relational (Σχεσιακή)

== != < > <= >=

Assignment (εκχώρηση)

= += -= *= /= %= &= |= ^= <<= >>=

Conditional (υπό όρους)

?:

Boolean

true false

7) ΕΠΑΝΑΛΗΨΕΙΣ

Εδώ θα δούμε πως μπορούμε να εκτελέσουμε εντολές ανάλογα με το αν ικανοποιούνται διάφορες συνθήκες που θέτουμε. Υπάρχουν διάφορες μορφές επαναλήψεων.

Αρχικά έχουμε:

1)If statement:

If(condition)

Statement;

Εάν ικανοποιείται η συνθήκη που έχουμε στην παρένθεση τότε εκτελείται η εντολή αλλιώς το πρόγραμμα προχωράει παρακάτω. Ένα απλό παράδειγμα :

int x = 10;

if (x >= 10)

Console.WriteLine("x is greater/equal 10");

Αν το χ είναι μεγαλύτερο από 10 τότε εκτελείται η εντολή

2) if-else

Η λειτουργία του είναι σχεδόν ιδιά με αυτή της if με τη διαφορά ότι αν δεν ισχύει η συνθήκη τότε εμείς θα ορίσουμε ποια εντολή θα εκτελεστεί. Για παράδειγμα:

```
int x = 42;
if (x == 0 || x < 0)
   Console.WriteLine("your number is negative or zero");
else
Console.WriteLine("your number is positive");
```

3) Switch

Η εντολή αυτή λειτουργεί ανάλογα με τις τιμές μια μεταβλητής. Συντάσσεται ως εξής:

```
switch (variable)
{
   case value_1:
      statement_1a;
      statement_1b;
      break;
   case value_2:
      statement_2a;
      statement_2b;
      break;
   case value_3:
      ...
      ...
   case value_n:
      statement_na;
      statement_nb;
```

```
        break;
    default:
        statement_default;
        break;
}
```

Μετά το switch υπάρχει η μεταβλητή με την ονομασία της. Στο case κάνουμε δήλωση την τιμή που ελέγχουμε. Ένα παράδειγμα:

```
int x = 2;
switch (x)
{
    case 0:
        Console.WriteLine("x equals 0 ");
        break;
    case 1:
    case 2:
        Console.WriteLine("x equals 1");
        Console.WriteLine("or x equals 2");
        break;
    case 3:
        Console.WriteLine("x equals 3");
        break;
    default:
```

```
        Console.WriteLine("x is negative or greater than 3");
        break;
    }
```

8) Κλάσεις

Ένας τρόπος που αναπτύσσεται ένας κώδικας είναι ο OOP. Πριν όμως από αυτό υπήρχε ένας άλλος τρόπος γνωστός και ως συναρτησιακός μέχρις ότου εμφανιστεί ο επόμενος που χρησιμοποιούμε

Μέχρι και σήμερα ο οποίος έχει λύσει αρκετά προβλήματα όπως για παράδειγμα ότι ο κώδικας γραφόταν σε πολλές ενότητες οι οποίες αλληλοεπηρεάζονταν και σε περίπτωση που έκανες λάθος τον κώδικα σε κάποια γραμμή ήταν πιθανό να το γράψεις όλο από την αρχή και σίγουρα όλη την ενότητα. Ο τρόπος προγραμματισμού για τον οποίο μιλάμε είναι ο αντικειμενοστραφής. Όπως αναφέραμε και παραπάνω ο τρόπος αυτός σου δίνει την επιλογή να αλλάξεις τον κώδικα σου οποιαδήποτε στιγμή χωρίς να χρειάζεται να αλλάξεις κάτι στο σύνολο του προγράμματος γιατί λειτουργεί με κλάσεις.

9) Κλάσεις στη c#

Η κλάση είναι το βασικό συστατικό για να δημιουργηθεί ένα πρόγραμμα, να χτιστεί δηλαδή από την αρχή. Τις χρησιμοποιούμε για να εισάγουμε μέσα σε αυτές ένα στοιχείο μαζί με τις ιδιότητες του. Επίσης ως πρότυπα για τη δημιουργία αντικειμένων. Για να το καταλάβουμε καλυτέρα ακολουθεί παράδειγμα και εξήγηση αυτού.

```
.class person
  {
    public string Name;
```

```
        public int Age;
    }
```

Με το παραπάνω δημιουργούμε το πρότυπο για έναν άνθρωπο στο οποίο καταχωρούμε το όνομα του και την ηλικία του. Για τη δήλωση λοιπόν της κλάσης αυτής γράφουμε τη λέξη-κλειδί class

Ακριβώς διπλά της το όνομα της και το περιεχόμενο της το ζητάμε μέσα σε αγκύλες. Πιο αναλυτικά:

```
    static void Main(string[] args)
        {
            Person Michael = new Person(); //δημιουργία αντικειμένου τύπου Person με το όνομα Michael
            Person Mary = new Person(); //αντίστοιχα
            // Ορισμός τιμών για τις ιδιότητες του Michael
            Michael.Age = 20;
            Michael.HairColor = "Brown";
            // Ορισμός τιμών για τις ιδιότητες της Mary
            Mary.Age = 25;
            Mary.HairColor = "Black";
            // εκτύπωση της ηλικίας τους
            Console.WriteLine("Michael's age = {0}, and Mary's age= {1}", Michael.Age, Mary.Age);
            Console.ReadLine();
        }
```

Εδώ αρχικά δημιουργούμε 2 αντικείμενα τύπου Person. Έπειτα γίνεται αρχικοποίηση των αντικειμένων mary και brown. Όταν λοιπόν γίνεται η δημιουργία του αντικειμένου michael ο compiler κρατάει 2 θέσεις μνήμης για τις δυο μεταβλητές . Κάθε αντικείμενο μπορεί να έχει δεδομένα διαφορετικά.

10) CONSTRUCTORS

Ένας κατασκευαστής είναι μια μέθοδος η οποία έχει την ιδιά ονομασία με τη κλάση ,παίρνει τα δεδομένα της αλλά δεν μπορεί να επιστρέψει τιμές. Κάθε κλάση πρέπει να έχει έναν κατασκευαστή ο οποίος τρέχει από μονός του όταν τη δημιουργούμε.

Ακολουθεί παράδειγμα

```
class Circle
{
   public Circle()  // default constructor
   {
      radius = 0.0;
   }
   public double Area()
   {
      return Math.PI * Math.Sqrt(radius);
   }
   private double radius;
}
Circle c;
c = new Circle();
```

```
double areaOfCircle = c.Area();
```

Το παραπάνω παράδειγμα δείχνει τη δημιουργία μιας κλάσης circle καθώς και τον κατασκευαστή της ο οποίος ορίζει την ακτίνα σε 0.

Στο συγκεκριμένο κομμάτι κώδικα ο κατασκευαστής μας είναι δημόσιος που σημαίνει ότι μπορεί να χρησιμοποιηθεί και έξω από την κλάση σε αντίθεση να μην μπορεί αν τον είχαμε κάνει private.

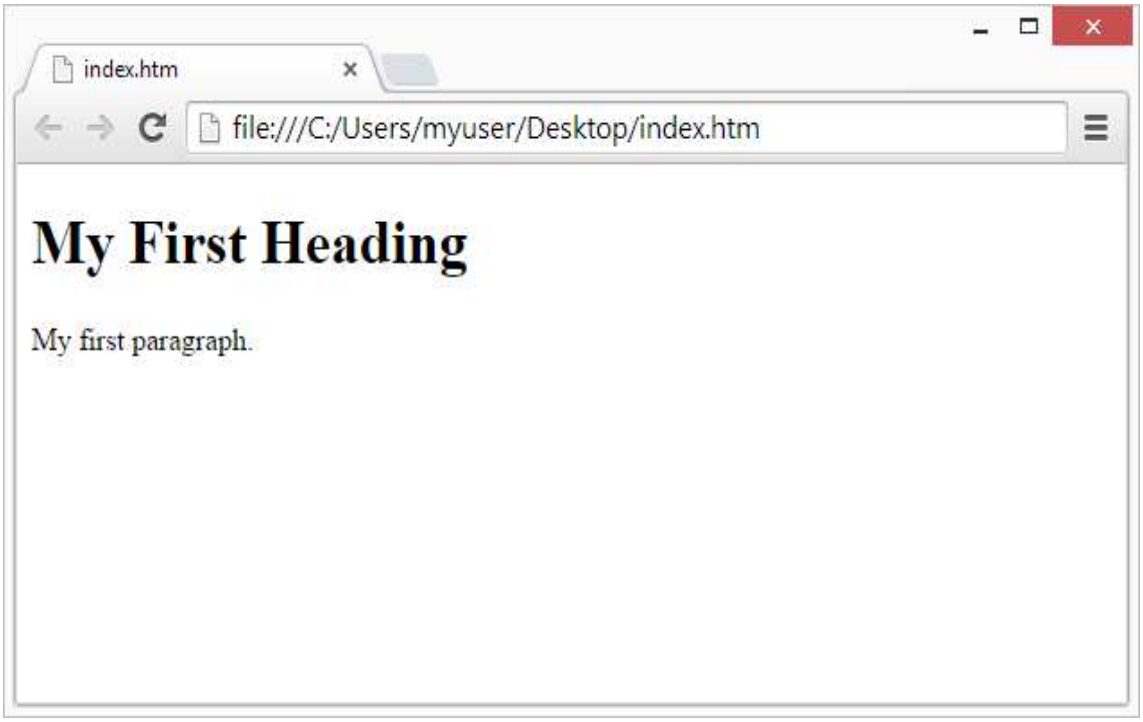
2.2.2 Η γλώσσα HTML

Η HTML είναι μια γλώσσα προγραμματισμού αρκετά απλά φτιαγμένη αποτελούμενη από στοιχεία τα οποία μπορούν να χρησιμοποιηθούν σε κομμάτια ενός κειμένου για να δώσουν διαφορετικό νόημα σε ένα αρχείο. Στην ουσία είναι μια γλώσσα η οποία συντάσσεται σε ένα text αρχείο και μια της λειτουργία είναι η ανάγνωση και η απεικόνιση της από έναν πλοηγό. Ο πλοηγός δεν εμφανίζει τις εντολές που είναι γραμμένες στο text αλλά τις χρησιμοποιεί για να καθορίσει το πως θα απεικονιστεί το κείμενο. Για την κατανόηση ακολουθεί ένα παράδειγμα.

Ανοίγοντας ένα text αρχείο και πληκτρολογώντας τα παρακάτω:

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Ο πλοηγός θα εμφανίσει το παρακάτω:



ΕΙΚΟΝΑ 24 ΠΑΡΑΔΕΙΓΜΑ HTML

2.2.3 Γλώσσα PHP

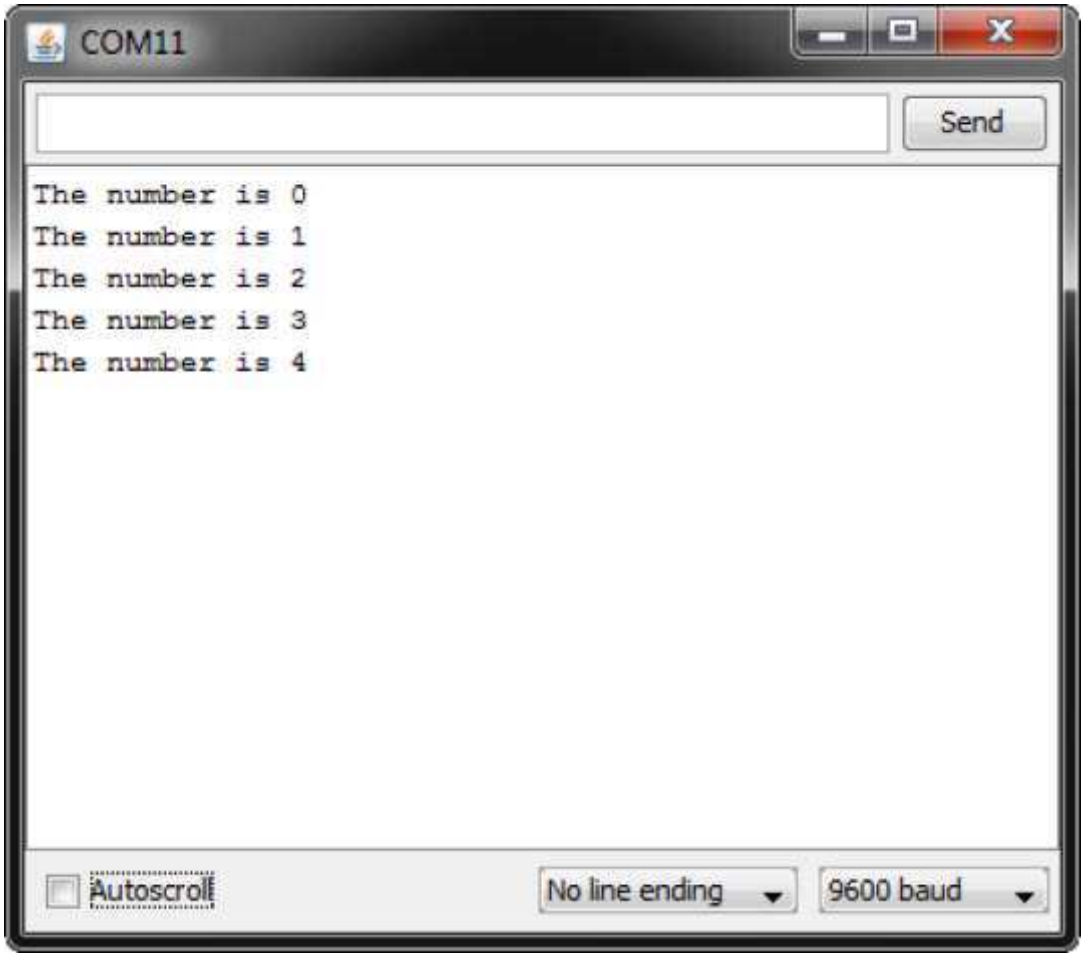
Η PHP είναι μια πολύ γνωστή γλώσσα προγραμματισμού η οποία μπορεί να εισχωρήσει σε HTML. Χρησιμοποιείται για προγραμματισμό διαδικτυακών εφαρμογών και ειδικά για τη δημιουργία ιστοσελίδων , dynamic ιστοσελίδων , δηλαδή σελίδων που μπορούν να επεξεργάζονται online από τον διαχειριστή ανάλογα με τα χαρακτηριστικά του χρήστη όπως αν είναι Windows , Linux κ.α. , τη φυσική διεύθυνση του κλπ. Είναι μια γλώσσα η οποία διαχειρίζεται λειτουργίες και εργασίες μιας σελίδας όπως για παράδειγμα για την δημιουργία λογαριασμού ενός χρήστη το οποίο το πραγματοποιεί σε συνεργασία με την HTML. Μια επιπλέον λειτουργία της είναι η συνεργασία με MySQL για τη επεξεργασία δεδομένων μέσα σε βάσεις όπως στο παράδειγμα που αναφέραμε , με σκοπό την αποθήκευση των usernames και των passwords.

2.3 Arduino ide

2.3.1 Εισαγωγή στο Arduino IDE

Το περιβάλλον του Arduino περιλαμβάνει έναν επεξεργαστή κειμένου στον οποίο γράφουμε τον κώδικα , στην συγκεκριμένη περίπτωση σε γλώσσα C# , μια περιοχή με μηνύματα , μια γραμμή εργαλείων με μια σειρά λειτουργιών και μενού. Συνδέεται με τη πλακέτα Arduino με σκοπό το ανέβασμα του κώδικα και την επικοινωνία μεταξύ τους. Το πρόγραμμα γράφεται στον επεξεργαστή κειμένου και γίνεται upload στην πλακέτα που χρησιμοποιούμε. Για να γίνει σωστό το ανέβασμα πρέπει να επιλέξουμε την com στην οποία έχουμε συνδέσει την πλακέτα με τον υπολογιστή ώστε να υπάρξει η επικοινωνία μεταξύ τους και να ανεβεί ο κώδικας όπως επίσης και την πλακέτα που χρησιμοποιούμε. Εκτός από το ανέβασμα του κώδικα απαιτείται επίσης, για την σωστή επικοινωνία, η χρήση βιβλιοθηκών. Οι βιβλιοθήκες παρέχουν επιπλέον λειτουργίες που μπορούμε να χρειαστούμε. Η εισαγωγή αυτών γίνεται από το μενού . Sketch > Import Library. Όταν λοιπόν πραγματοποιηθεί το ανέβασμα του κώδικα μπορούμε να δούμε το αποτέλεσμα του στο Serial Monitor στο οποίο εικονίζονται συριακά τα δεδομένα τα οποία έχουν σταλεί από την πλακέτα. Τέλος μπορούμε να στείλουμε δεδομένα μέσω του serial monitor στην πλακέτα αρκει να πληκτρολογήσουμε το μήνυμα στο command line και να πατήσουμε Send

ΕΙΚΟΝΑ 25 ΠΕΡΙΒΑΛΛΟΝ ARDUINO IDE



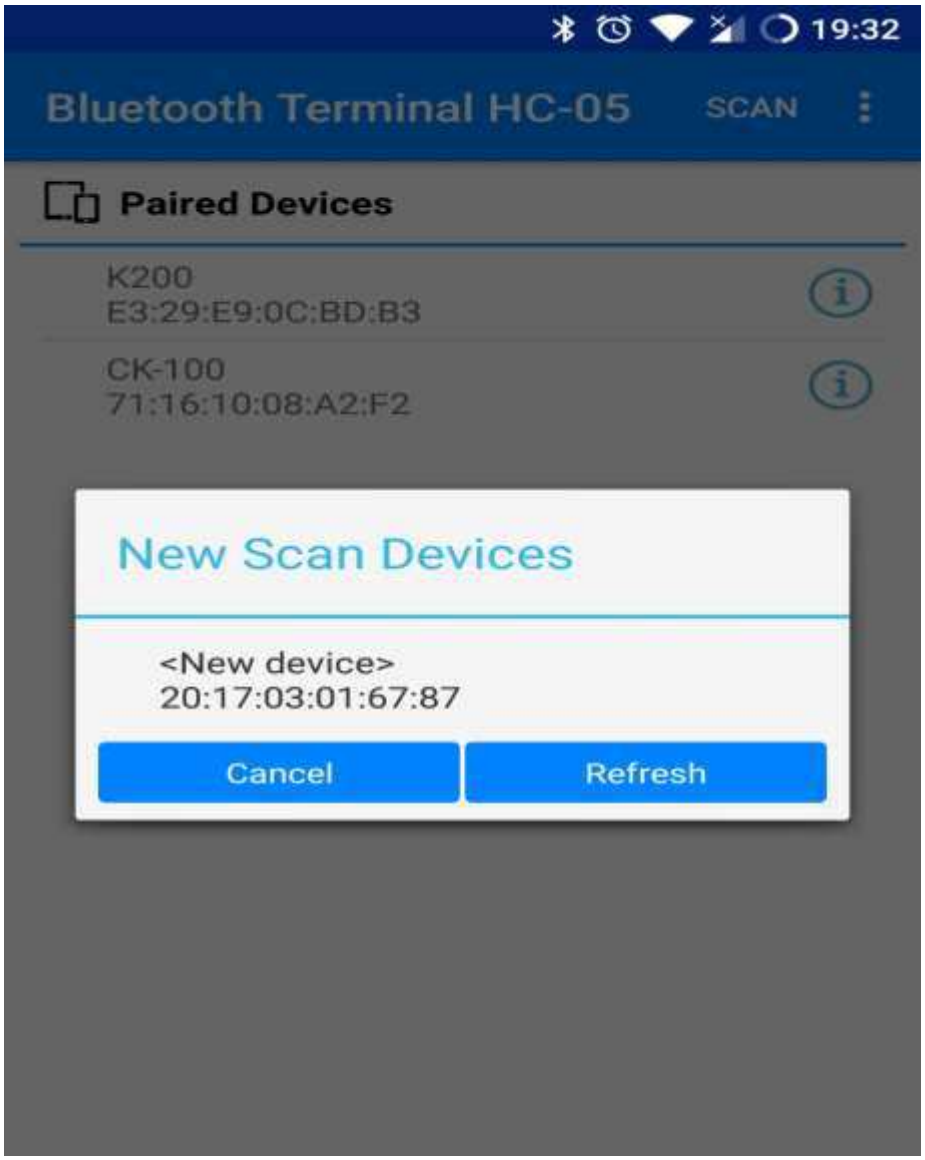
ΕΙΚΟΝΑ 26 SERIAL MONITOR ARDUINO IDE

2.4 Bluetooth Terminal HC-05

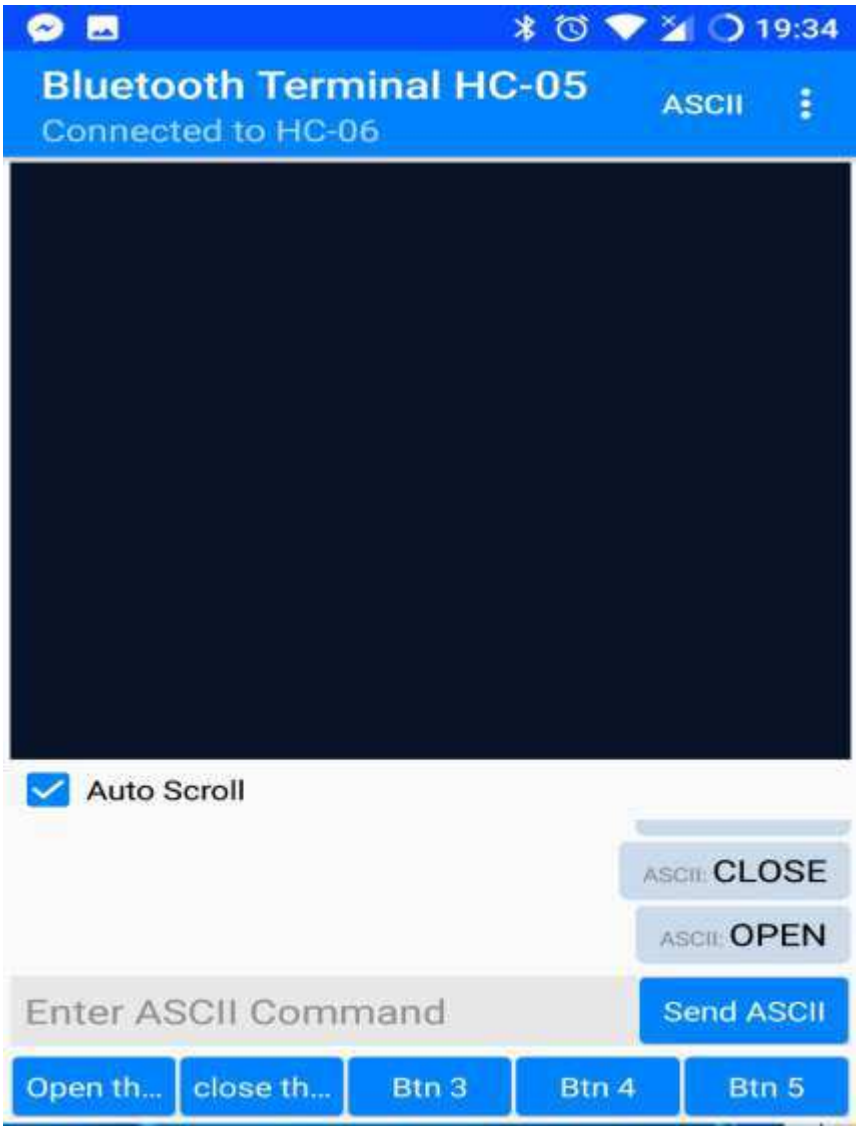
Στην παράγραφο αυτή θα παρουσιαστεί το πρόγραμμα που χρησιμοποιήθηκε για να πραγματοποιηθεί ο έλεγχος του bluetooth από το κινητό.

Το πρόγραμμα αυτό ονομάζεται Bluetooth Terminal HC-05 και διατίθεται δωρεάν στο Google Play Store. Για να επιτύχουμε σύνδεση ανοίγουμε την εφαρμογή η οποία ενεργοποιεί αυτόματα το bluetooth του κινητού μας. Πρώτη ενέργεια είναι η σάρωση για νέες συσκευές bluetooth. Όταν βρεθεί η συσκευή μας πατάμε connect και τη πρώτη φορά μας ζητά να πληκτρολογήσουμε τον κωδικό που ορίζουμε για να μπορέσουμε να το διαχειριστούμε αναλόγως. Με επιτυχή σύνδεση , για να ανοίξουμε τη πόρτα (στη περίπτωση μας να ενεργοποιήσουμε το led

λαμπάκι.) , πατάμε το κουμπί το οποίο το έχουμε ονομάσει open. Τα παραπάνω προϋποθέτουν και τη συγγραφή κατάλληλου κώδικα ο οποίος θα παρουσιαστεί παρακάτω.



ΕΙΚΟΝΑ 2 ΣΑΡΩΣΗ ΓΙΑ ΝΕΕΣ ΣΥΣΚΕΥΕΣ BLUETOOTH



ΕΙΚΟΝΑ 3 ΚΟΥΜΠΙ ΑΝΟΙΓΜΑΤΟΣ ΠΟΡΤΑΣ

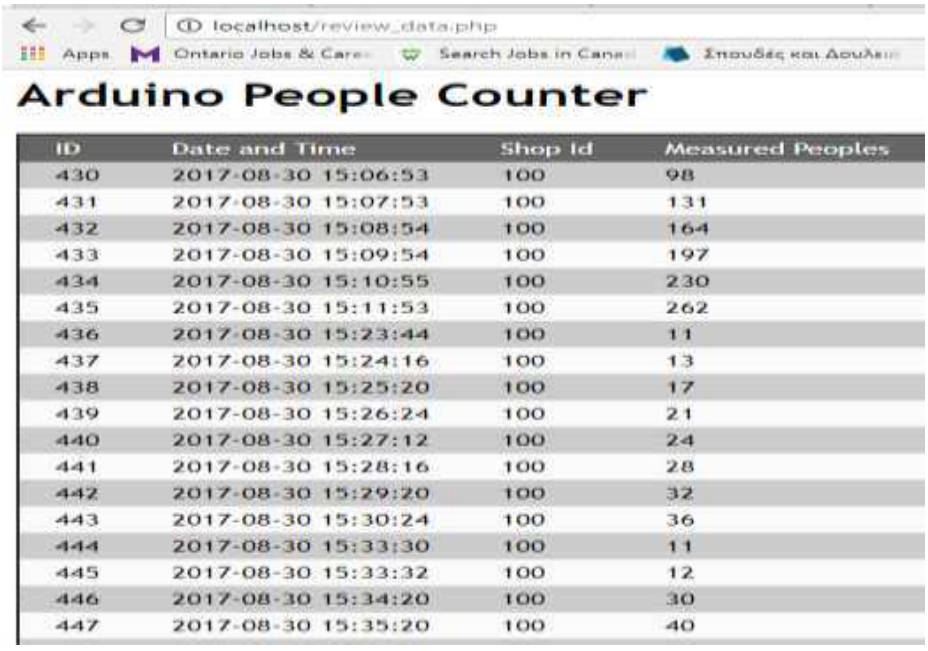
ΚΕΦΑΛΑΙΟ 3
Βάση δεδομένων και προγραμματισμός

3.1 Λογισμικό μέρος συστήματος

Στο κεφάλαιο αυτό θα διατυπώσουμε ακριβώς τη λειτουργία του συστήματος. θα αναλύσουμε δηλαδή τον ακριβή τρόπο επικοινωνίας του Arduino με τη βάση δεδομένων, την απεικόνιση των μετρήσεων σε ιστόχωρο αλλά και στη βάση δεδομένων.

3.1.1 Παρουσίαση πίνακα ισπροτόχωρου

Ο ιστόχωρος αποτελείται από τη μια και αρχική του σελίδα η οποία τραβάει τα δεδομένα , τις μετρήσεις δηλαδή του αισθητήρα και τις απεικονίζει σε έναν πίνακα με μερικά ακόμα στοιχεία τα οποία είναι ένας αύξων αριθμός που αποτελεί τον αριθμό των μετρήσεων , την ημερομηνία και ώρα που ανέβηκε η συγκεκριμένη μέτρηση, το shop id το οποίο μας χρησιμεύει αν έχουμε περισσότερα από ένα μαγαζιά στην ιδιά βάση και τέλος τον αριθμό της μέτρησης του αισθητήρα. Το link το οποίο χτυπάμε είναι η τοποθεσία στην οποία έχουμε αποθηκεύσει το αρχείο review_data.php στον τοπικό διακομιστή.



ID	Date and Time	Shop Id	Measured Peoples
430	2017-08-30 15:06:53	100	98
431	2017-08-30 15:07:53	100	131
432	2017-08-30 15:08:54	100	164
433	2017-08-30 15:09:54	100	197
434	2017-08-30 15:10:55	100	230
435	2017-08-30 15:11:53	100	262
436	2017-08-30 15:23:44	100	11
437	2017-08-30 15:24:16	100	13
438	2017-08-30 15:25:20	100	17
439	2017-08-30 15:26:24	100	21
440	2017-08-30 15:27:12	100	24
441	2017-08-30 15:28:16	100	28
442	2017-08-30 15:29:20	100	32
443	2017-08-30 15:30:24	100	36
444	2017-08-30 15:33:30	100	11
445	2017-08-30 15:33:32	100	12
446	2017-08-30 15:34:20	100	30
447	2017-08-30 15:35:20	100	40

ΕΙΚΟΝΑ 27 ΕΜΦΑΝΙΣΗ ΤΟΥ ΑΡΧΕΙΟΥ REVIEW_DATA ΣΤΟΝ ΔΙΑΚΟΜΙΣΤΗ ΜΑΣ

Στην συνέχεια παρουσιάζεται ο κώδικας της αρχικής σελίδας που είναι αποθηκευμένος στον τοπικό server

```
<html>
<head>
   <title>Arduino People Counter</title>
   <style type="text/css">
      .table_titles, .table_cells_odd, .table_cells_even {
            padding-right: 20px;
            padding-left: 20px;
            color: #000;
      }
      .table_titles {
         color: #FFF;
         background-color: #666;
      }
      .table_cells_odd {
         background-color: #CCC;
      }
      .table_cells_even {
         background-color: #FAFAFA;
      }
      table {
         border: 2px solid #333;
      }
      body { font-family: "Trebuchet MS", Arial; }
```

```
    </style>
</head>

    <body>
      <h1>Arduino People Counter</h1>
    <table border="0" cellspacing="0" cellpadding="4">
     <tr>
          <td class="table_titles">ID</td>
          <td class="table_titles">Date and Time</td>
          <td class="table_titles">Shop Id</td>
          <td class="table_titles">Measured Peoples</td>
        </tr>
```

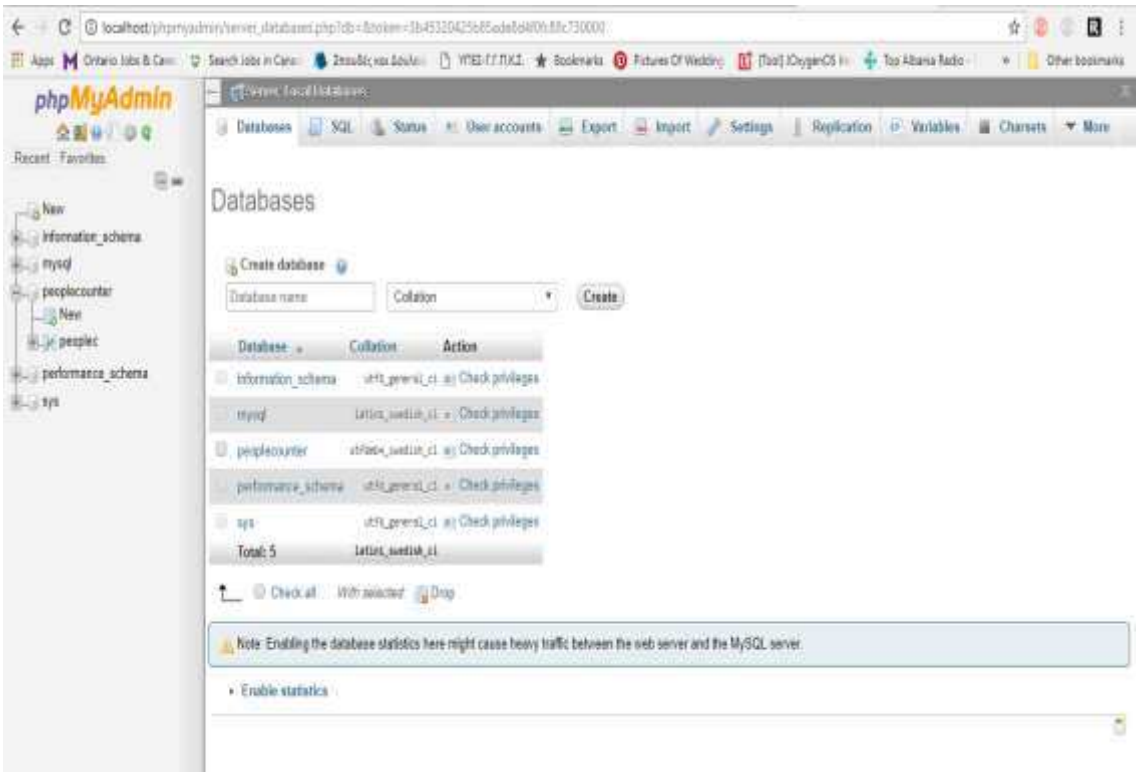
Το αρχείο review_data.php εκτός από το html που χρησιμοποιήσαμε για την παρουσία στου πίνακα περιέχει και κώδικα ο οποίος είναι βοηθητικός και συνεργάζεται με κάθε ένα αρχείο ιστοσελίδας με διάγραμμα. Οι ιστοσελίδες αυτού του είδους καλούν αυτό το αρχείο το οποίο είναι υπεύθυνο για την αποστολή των καταγεγραμμένων μετρήσεων από την βάση δεδομένων στις σελίδες απεικόνισης διαγραμμάτων. Ύστερα αυτές μετατρέπουν τα δεδομένα που τους παρέχει το αρχείο review_data.php σε διαγράμματα.

```
<?php
    // Start MySQL Connection
    include('dbconnect.php');
?>
// Retrieve all records and display them
    $result = mysql_query("SELECT * FROM peoplecounter ORDER BY id ASC");
```

```
    // Used for row color toggle
    $oddrow = true;
    // process every record
    while( $row = mysql_fetch_array($result) )
    {
        if ($oddrow)
        {
            $css_class=' class="table_cells_odd"';
        }
        else
        {
            $css_class=' class="table_cells_even"';
        }
        $oddrow = !$oddrow;
        echo '<tr>';
        echo '   <td'.$css_class.'>'.$row["id"].'</td>';
        echo '   <td'.$css_class.'>'.$row["log_time"].'</td>';
        echo '   <td'.$css_class.'>'.$row["shop_id"].'</td>';
        echo '   <td'.$css_class.'>'.$row["atoma"].'</td>';
        echo '</tr>';
    }
?>
    </table>
    </body>
</html>
```

3.2 Βάση δεδομένων

Η βάση δεδομένων που πραγματοποιήθηκε, έχει το όνομα peoplecounter και περιέχει έναν πίνακα που ονομάζεται people. Ο πίνακας αυτός απαρτίζεται από 4 στήλες που χρησιμοποιούνται για την καταγραφή δεδομένων , δηλαδή των μετρήσεων του αισθητήρα. Για να φτιάξουμε τη βάση ανοίγουμε την σελίδα phpmyadmin , δημιουργούμε μια νέα βάση δεδομένων με το όνομα που θέλουμε και τρέχουμε το παρακάτω αρχείο για να δημιουργηθεί ο πίνακας που θέλουμε.



ΕΙΚΟΝΑ 28 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ ΣΤΟΝ ΔΙΑΚΟΜΙΣΤΗ ΜΑΣ

```
CREATE TABLE `peoplecounter`.`peoplec` (
  `id` int(11) NOT NULL,
  `time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT 'Event Date and Time',
  `shop_id` varchar(30) NOT NULL COMMENT 'Unique ID of the shop',
  `atoma` varchar(10) NOT NULL COMMENT 'Measured peoples'
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

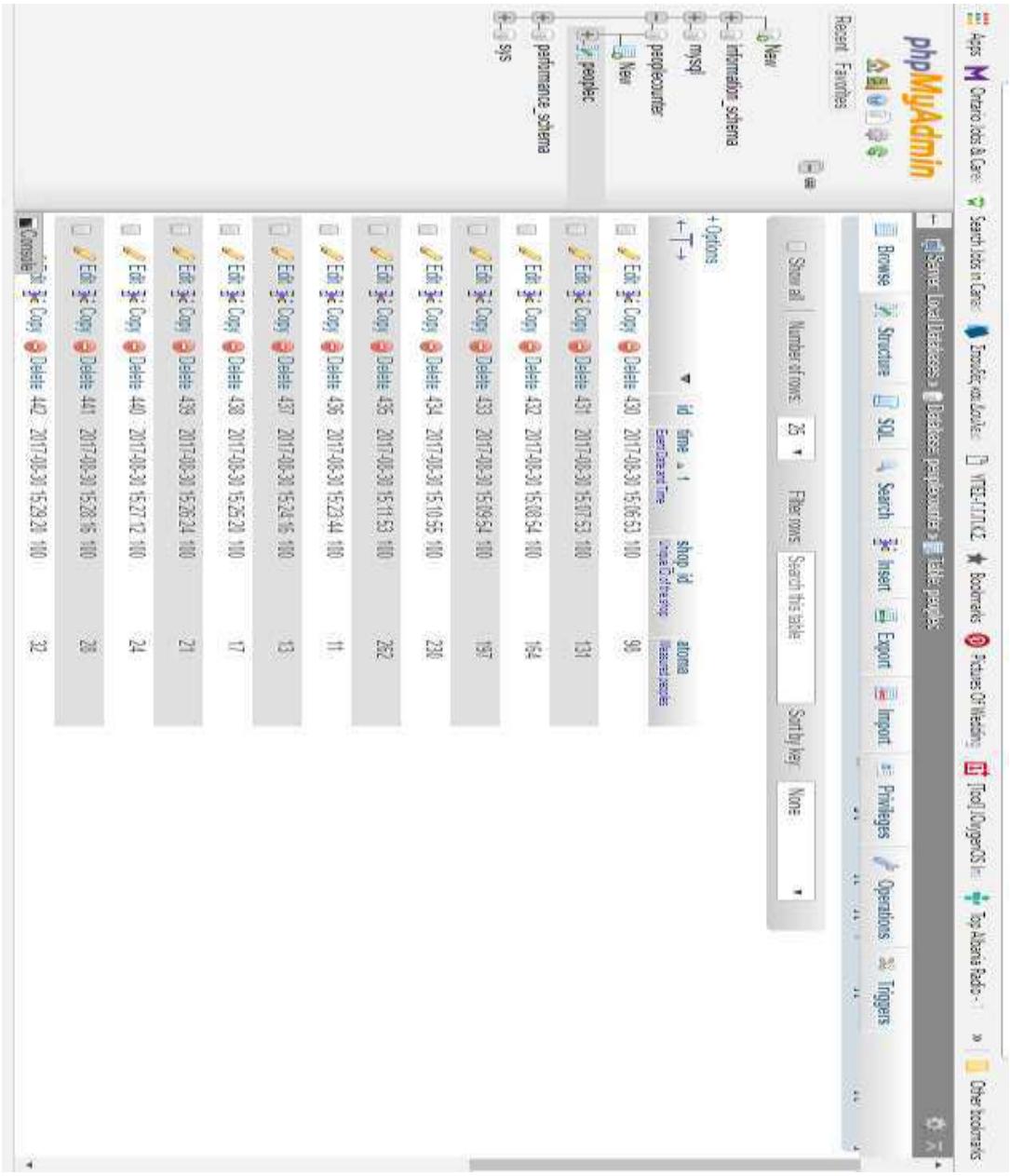
ΕΙΚΟΝΑ 29 ΚΩΔΙΚΑΣ ΓΙΑ ΤΗΝ ΔΗΜΙΟΥΡΓΙΑ ΤΟΥ ΠΙΝΑΚΑ ΜΑΣ ΣΤΗΝ ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

Ο παρακάτω πίνακας χρησιμοποιείται για την καταχώριση των μετρήσεων του αισθητήρα και την καταγραφή της ώρας κατά την οποία έγινε εγγραφή. Ακολουθεί αναλυτική περιγραφή των πεδίων του:

Όνομα	*Τύπος*	*Σύνθεση*	Προεπιλογή
Time	timestamp		CURRENT_TIMESTAMP
Shop_id	varchar(30)	latin1_swedish_ci	None
atoma	varchar(10)	latin1_swedish_ci	None

ΠΙΝΑΚΑΣ 3 ΠΙΝΑΚΑΣ ΚΑΤΑΧΩΡΗΣΕΩΝ ΜΕΤΡΗΣΕΩΝ ΣΤΗΝ ΒΑΣΗ

- **Time**: Εδώ γίνεται η καταγραφή της ημερομηνίας και της ώρας που ανέβηκε η μέτρηση
- **Shop_id**: Αποτελεί αναγνωριστικό του καταστήματος σε περίπτωση που υπάρχουν πάνω από ένα
- **Atoma**: τα atoma είναι η μέτρηση του αισθητήρα που αποθηκεύεται στη βάση δεδομένων



ΕΙΚΟΝΑ 30 ΕΜΦΑΝΙΣΗ ΤΩΝ ΚΑΤΑΧΩΡΗΜΕΝΩΝ ΜΕΤΡΗΣΕΩΝ ΣΤΗΝ PHPMYADMIN

3.3 Επικοινωνία βάσης δεδομένων

Η επικοινωνία μεταξύ του μικροελεγκτή Arduino και της βάσης δεδομένων πραγματοποιείται με αρχεία. php για να λαμβάνει και να αποτυπώνει τα δεδομένα. Τα αρχεία αυτά βρίσκονται μέσα στον τοπικό διακομιστή. Ακολουθεί η ανάλυση των αρχείων αυτών,

- **dbconnect.php.**

Το αρχείο αυτό είναι δημιουργήθηκε για να ξεκινήσει η σύνδεση με τη βάση δεδομένων peoplecounter.

```
<?php
$MyUsername = "root";  // όνομα χρήστη για τη MySQL
$MyPassword = "";  // κωδικός για την  MySQL
$MyHostname = "localhost";  // το όνομα του τοπικού διακομιστή
$dbh = mysql_pconnect ($MyHostname, $MyUsername, $MyPassword); // Σύνδεση
$selected = mysql_select_db ("peoplecounter», dbh);?>
```

- add_data.php.

Το αρχείο αυτό επικοινωνεί με το dB connect.php για να επέλθει η σύζευξη με τη βάση δεδομένων . Στη συνέχεια παίρνει τα δεδομένα του αισθητήρα και τα μεταφέρει στη βάση μας.

Ακολουθεί ο κώδικας

```
<?php
   // Σύνδεση με τη βάση δεδομένων βασιζόμενοι στο dbconnect.php file
   include("dbconnect.php");
   // Καταγραφή δεδομένων στη βάση μετρά από την λήψη.
   $SQL = "INSERT INTO peoplecounter. peoplec (shop_id, atoma) VALUES ('".$_GET["shop_id"]."',
   '".$_GET["atoma"]."')";
```

```
// Εδώ τρέχει η δήλωση SQL

mysql_query($SQL);

// Go to the review_data.php (optional)

header("Location: review_data.php");

?>
```

3.4 Προγραμματισμός Arduino

Οι βιβλιοθήκες του Arduino μας παρέχουν είτε drivers για κάποια συσκευή που συνδέσουμε πάνω στον μικροελεγκτή, είτε είναι κώδικας ο οποίος περιέχει κάποια functions τα οποία χρησιμοποιούνται πολύ συχνά. Υπάρχουν δυο ειδών βιβλιοθήκες. Οι πρώτες είναι αυτές που βρίσκονται μέσα στο Arduino από μονές τους, δηλαδή δεν τις κάνουμε εμείς install. , παρέχονται μέσα στο IDE και οι πιο βασικές παρέχουν την δυνατότητα επικοινωνίας με διαφορά κομμάτια hardware όπως είναι οι σερβοκινητηρες. Εκτός από αυτές έχουμε και τις βιβλιοθήκες που εγκαθιστούμε εμείς οι χρήστες για ειδική επικοινωνία μεταξύ της πλακέτας και ενός αλλού στοιχείου

3.5 Βασικές βιβλιοθήκες

3.5.1 Βιβλιοθήκη adafruit_cc3000.h

Η βιβλιοθήκη αυτή είναι υπεύθυνη για την επικοινωνία του Arduino mega με τη πλακέτα της Adafruit cc3000. Η πλακέτα αυτή χρησιμοποιείται για τη σύνδεση του Arduino στο δίκτυο. Παρέχει τη λειτουργία της πλακέτας ως διακομιστή η ως πελάτη. Ακολουθεί ανάλυση των συναρτήσεων που χρησιμοποιήθηκαν:

- wifiinit (): Η συνάρτηση αυτή μας παρουσιάζει τη διαδικασία σύνδεσης της πλακέτας στο διαδίκτυο μέσω μιας σειράς εντολών. Με τη δήλωση της mac address της πλακέτας προσπαθεί να συνδεθεί και σε περίπτωση σύνδεσης ζητάει DHCP

- connectServer() : Η συγκεκριμένη συνάρτηση επιχειρεί την επικοινωνία με τον server. Χρησιμοποιεί τη δήλωση ιστοσελίδας και την imp για να συνδεθεί και να μεταφέρει τις μετρήσεις του αισθητήρα.

- Connect (): Συνδέεται με την IP και την πόρτα που έχουμε επιλέξει κατά τη δημιουργία. Αν συνδεθεί και είναι true τότε εμφανίζει μήνυμα επιτυχίας αλλιώς αποτυχίας.

- miscInit(): Ενεργοποιεί τη πλακέτα για να ξεκινήσει η λειτουργία της.

- timeCheck() : Μέσω του Wi-Fi της πλακέτας adafruit γίνεται η αναζήτηση της ώρας η οποία υπολογίζεται από το 1970.

3.5.2 Βιβλιοθήκη SPI.h

Η βιβλιοθήκη αυτή μας δίνει την επιλογή σύνδεσης με άλλες SPI συσκευές με το Arduino ως κυριά συσκευή. Ως SPI (σειριακή περιφερική διεπαφή) ορίζουμε ένα πρωτόκολλο δεδομένων που το λειτουργούν οι μικροεπεξεργαστές για να έχουν τη δεινότητα επικοινωνίας με άλλες συσκευές η με άλλον μικροελεγκτή.

3.5.3 Βιβλιοθήκη DHT

Η βιβλιοθήκη αυτή χρησιμοποιείται για την μετάφραση του αναλογικού σήματος του αισθητήρα σε πραγματική τιμή.

3.5.4 Βιβλιοθήκη SD.h

Η βιβλιοθήκη αυτή επιτρέπει τη καταγραφή στην κάρτα αλλά και την ανάγνωση δεδομένων από αυτή. Υποστηρίζει τύπους αρχείων FAT16 και FAT32. Επιτρέπει την καταγραφή πολλών αρχείων μέσα σε έναν φάκελο χωρίζοντας τα με < / >. Για παράδειγμα

"directory/filename.txt". Ανάλογα με την έκδοση της βιβλιοθήκης υποστηρίζει το άνοιγμα πολλών αρχείων. Η επικοινωνία μεταξύ του μικροελεγκτή και της κάρτας γίνεται με χρήση SPI και λαμβάνει χωρά στα ψηφιακά pin 11,12,13 η στα 50,51,52.

3.6 Προγραμματισμός αισθητήρα υπερήχων

Στο πρώτο κεφάλαιο παρουσιάστηκε ο αισθητήρας ο οποίος χρησιμοποιήθηκε στην εργασία αυτή. Ο αισθητήρας αυτός είναι υπέρηχων. Αποτελείται από έναν πομπό και έναν δεκτή. Εκπέμπει δηλαδή υπέρηχους και ανάλογα με τον χρόνο που θα κάνει ο δέκτης να τους αντιληφθεί όταν βρει εμπόδιο και αρχίσει η επιστροφή τους, υπολογίζει την απόσταση στην οποία βρέθηκε το αντικείμενο. Μετατρέπει δηλαδή ψηφιακή είσοδο σε απόσταση. Για την λειτουργία του αισθητήρα χρειάζεται να καλέσουμε τη συνάρτηση readSensor()

Ακολουθεί ο κώδικας:

```
void readSensor () {
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  if (distance < 100) {
    Serial.println(distance);
    atoma=atoma+1;
    writeFile();
  }
```

```
  if (distance >= 200 || distance <= 0){
    Serial.println("Out of range");
  }
  else {
    Serial.println(distance);
    Serial.println(" cm");
  }
  delay(800);
}
```

3.7 Προγραμματισμός πλακέτας Bluetooth

Η συσκευή Bluetooth σκοπό έχει τον απομακρυσμένο έλεγχο ενός στοιχείου, στην παρούσα φάση ενός led . Αναλύοντας το , συνδέουμε το κινητό μας με την συγκεκριμένη πλακέτα μέσω ενός προγράμματος. Όταν η σύνδεση γίνει επιτυχής τότε μπορούμε να διαχειριστούμε το led. Σε άλλη περίπτωση θα μπορούσε να ανοίξει μια πόρτα για να εισέλθουμε σε ένα κατάστημα. Ακολουθεί η παρουσίαση του προγράμματος που χρησιμοποιήθηκε ,βήμα βήμα η σύνδεση του κινητού με το Bluetooth όπως και ο κώδικας της πλακέτας.

Φωτογραφίες και περιγραφή του προγράμματος για το Bluetooth πως συνδέεται πως ελέγχω το led κλπ.

Κώδικας:

```
void bluetooth()
{
    // listen for communication from the BT module and then write it
to the serial monitor
    if ( Serial1.available() )
    {
        c = Serial1.read();
        Serial.write( c );
```

```
        digitalWrite(led, HIGH);
        ssd();
        digitalWrite(led, LOW);
    }


    // listen for user input and send it to the HC-05
   if ( Serial.available() )
   {
        c = Serial.read();
        Serial1.write( c );
                digitalWrite(led, LOW);

   }

}
```

3.7.1 Αποστολή μετρήσεων στη βάση δεδομένων

Η πλακέτα μας (Arduino mega) έχει προγραμματιστεί για να συνδέεται και να ανεβάζει μεσω wifi τις μετρήσεις του αισθητήρα ανά τακτά χρονικά διαστήματα στη βάση. Για να πραγματοποιηθεί αυτό γίνεται χρήση της συνάρτησης connectServer. Η συγκεκριμένη συνάρτηση εκτελείται ανά χρονικό διάστημα το οποίο το έχουμε ορίσει εμείς. Για τον υπολογισμό του χρόνου από τη πλακέτα μας χρησιμοποιούμε μετρητή .Το Arduino διαχειρίζεται την ώρα η οποία έχει υπολογιστεί μέσω του διαδικτύου για το εσωτερικό του ρολόι και ο αντίστροφος αυτός μετρητής όταν μηδενίζει ξεκινάει τη σύνδεση με τη βάση. Ακολουθεί ο κώδικας.

Κώδικας connectServer()

```
void connectServer () {

  uint32_t ip;

  Serial.print(WEBSITE); Serial.print(F(" -> "));
  while (ip == 0) {
    if (! cc3000.getHostByName(WEBSITE, &ip)) {
      Serial.println(F("Couldn't resolve!"));
    }
    delay(500);
  }

  cc3000.printIPdotsRev(ip);
  Serial.println("");
```

```
  ip=cc3000.IP2U32(192, 168, 1, 7);

  Adafruit_CC3000_Client www = cc3000.connectTCP(ip, 80);
  if (www.connected()==1) {
    char temp[30];
    char temp1[30];
    sprintf(temp, "%d", atoma);
    sprintf(temp1, "%d", shop_id);

    Serial.println(F("Connected to server"));
    Serial.println(F("Connected !"));
    www.fastrprint(F("GET "));
    www.fastrprint(WEBPAGE);
    www.fastrprint(temp);
    www.fastrprint("&&shop_id=");
    www.fastrprint(temp1);
    www.fastrprint(F(" HTTP/1.1\r\n"));
    www.fastrprint(F("Host: "));
    www.fastrprint(WEBSITE);
    www.fastrprint(F("\r\n"));
    www.fastrprint(F("Connection: close\r\n\r\n"));
  }
  else {
    Serial.println(F("Connection failed"));
    return;
  }

  Serial.println(F("-------------------------------------"));

}
```

3.8 Καταγραφή δεδομένων σε κάρτα μνήμης SD

Μια ακόμα λειτουργία που έχουμε προσθέσει είναι η καταγραφή των μετρήσεων του αισθητήρα σε μια κάρτα μνήμης. Πιο αναλυτικά, έχουμε τοποθετήσει στην ειδική εσοχή της πλακέτας adafruit cc3000 μια κάρτα μνήμης SD.Μέσα σε αυτή δημιουργούμε ένα αρχείο μορφής text (.txt) στο οποίο και καταγράφονται οι μετρήσεις. Ο κώδικας που δημιουργήσαμε είναι ο παρακάτω:

```
void writeFile () {
   myFile = SD.open("example.txt", FILE_WRITE);

  // εάν ανοίξει ο φάκελος τότε γράφε σε αυτό
  if (myFile) {
    Serial.println("Writing to example.txt...");
    myFile.println("there are  ");
    myFile.println(atoma);
    myFile.println("people  ");
```

```
    Serial.write(myFile.println(atoma));
    // close the folder
    myFile.close();
  }
}
```

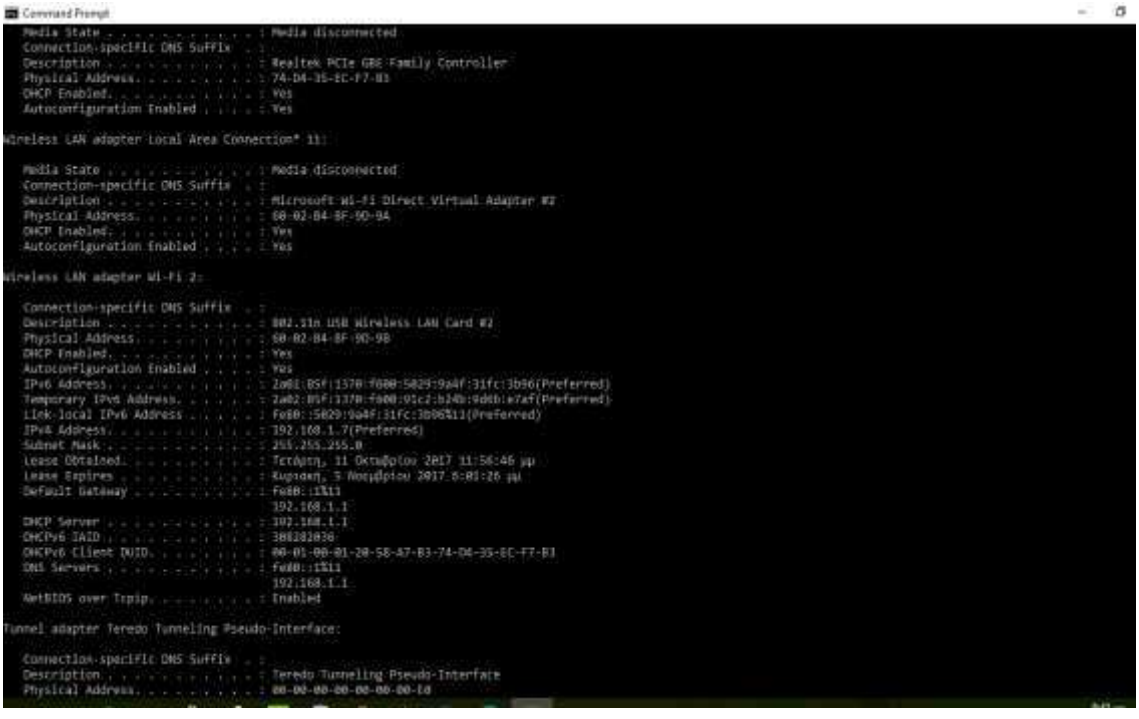
ΚΕΦΑΛΑΙΟ 4
Κώδικας C#

4.1 Κώδικας

Ο κώδικας θα παρουσιαστεί με σειρά εκτέλεσης.

//οι παρακάτω γραμμές κώδικα καλούν τις βιβλιοθήκες που χρειαζόμαστε. Κάθε βιβλιοθήκη χρησιμοποιείται για ξεχωριστή δουλειά η οποία πραγματοποιείται μέσα στον κώδικα. Χωρίς αυτές, οι λειτουργίες του κώδικα δε θα ήταν εφικτές.

Η εικόνα αυτή μας παρέχει χρήσιμες πληροφορίες , μερικές από τις οποίες θα χρειαστούμε.



ΕΙΚΟΝΑ 31 CMD ΓΙΑ ΝΑ ΒΡΟΥΜΕ ΤΗΝ IP ΤΟΥ ΔΙΑΚΟΜΙΣΤΗ ΜΑΣ

Ο κώδικας θα παρουσιαστεί με σειρά εκτέλεσης

```
//Οι παρακάτω γραμμές κώδικα καλούν τις βιβλιοθήκες που χρειαζόμαστε.
//Κάθε Βιβλιοθήκη χρησιμοποιείται για ξεχωριστή δουλειά που πραγματοποιείται μέσα στον κώδικα.
//Χωρίς αυτές, οι λειτουργίες του κώδικα δε θα ήταν εφικτές.


#include <Adafruit_CC3000.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <Adafruit_CC3000_Server.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <ccspi.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <SPI.h>  //βιβλιοθήκη πλακέτας arduino
#include <string.h>  //βιβλιοθήκη πλακέτας arduino
#include "utility/debug.h" //βιβλιοθήκη πλακέτας arduino
#include <stdio.h>  //βιβλιοθήκη πλακέτας arduino
#include <URL.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <Bridge.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <HttpClient.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <DHT.h>  //βιβλιοθήκη του αισθητήρα υπερήχων
#include <Ethernet.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <EthernetClient.h>  //βιβλιοθήκη πλακέτας adafruit cc3000
#include <SD.h>  //βιβλιοθήκη κάρτας μνήμης
```

Στην συνέχεια του κώδικα **:**

```
#define WLAN_SSID       "COSMOTE-C6C3CE" // εδώ βάζουμε το όνομα του wifi
#define WLAN_PASS       "ZNxGFTa7fQxjrDkv" // εδώ τον κωδικό του wifi
#define WEBSITE      "localhost" // ορίζουμε τον τοπικό διακομιστή
#define WEBPAGE      "/add_data.php?atoma="  // δηλώνουμε το url που θα χτυπήσει στον Server
// για να γίνει προσθήκη μέτρησης στη μεταβλητή "atoma"  στον πίνακα
#define WEBPAGE1      "/add_data.php?shop_id=" // url που θα χτυπήσει στον server
//για να γίνει προσθήκη του αριθμού καταστήματος  στην μεταβλητή "shop_id" στον πίνακα
#define WLAN_SECURITY   WLAN_SEC_WPA2 // ορίζουμε τύπο ασφάλειας της ασύρματης σύνδεσης
#define FIVEMIN (1000UL * 60 * 5)  // κάθε ποτέ θα γίνει η σύνδεση με τον διακομιστή. το "5" στο τέλος
//σημαίνει ανά  5 λεπτά.
#define LEAP_YEAR(Y) ( ((1970+Y)>0) && !((1970+Y)%4) && ( ((1970+Y)%100) || !((1970+Y)%400) ) )
//εδώ υπολογίζεται ο χρόνος από το 1970. Έτσι το υπολογίζει το Arduino.
#define DAYTOSEC  86400 //μετατροπή της ημέρας σε δευτερόλεπτα
```

```
#define ADAFRUIT_CC3000_IRQ   3
#define ADAFRUIT_CC3000_VBAT  5
#define ADAFRUIT_CC3000_CS    10
Adafruit_CC3000 cc3000 = Adafruit_CC3000(ADAFRUIT_CC3000_CS,
ADAFRUIT_CC3000_IRQ, ADAFRUIT_CC3000_VBAT,
                  SPI_CLOCK_DIVIDER);
Adafruit_CC3000_Client client;
//τα παραπάνω pin χρησιμοποιούνται για την επικοινωνία του cc3000 με το Arduino

#define trigPin 9
#define echoPin 8
/*τα 2 αυτά pin για τον αισθητήρα απόστασης το trigPin που το έχουμε συνδεδεμένο στο
pin 9 του Arduino
μας είναι ο πομπός και το echoPin το οποίο το έχουμε συνδέσει στο 8 είναι ο δέκτης μας
 */
 int led=44; //το pin στο οποίο είναι συνδεδεμένο το led
 const int buzzer1 = 30;  // δηλώνουμε το Pin του buzzer
```

ΔΗΛΩΜΕΝΕΣ ΜΕΤΑΒΛΗΤΕΣ:

```
unsigned long rolltime = millis() + FIVEMIN; //εδώ δηλώνουμε με όνομα rolltime την
μεταβλητή που
// χρησιμοποιείται για να μας βοηθήσει ώστε να καταλάβουμε ποτέ πέρασε ο χρόνος που
καθορίζουμε
//για να συνδεθεί το Arduino με την βάση δεδομένων μας χρησιμοποιώντας το εσωτερικό
ρολόι του
// Arduino. οι μεταβλητές αυτές τύπου unsigned long χρησιμοποιούνται για την
αποθήκευση νούμερων
// μεγάλου μεγέθους έως 4 bytes . δεν αποθηκεύουν αρνητικούς αριθμούς έχοντας εύρος
από 0 έως
//4,294,967,295

int atoma = 0; //μεταβλητή του αισθητήρα απόστασης για να μετράει τα άτομα
int shop_id=100; //μεταβλητή  αριθμού καταστήματος. Το ορίζει ο χρήστης

boolean tag = false; //εδώ δηλώνουμε την μεταβλητή tag ως boolean
// οι μεταβλητές boolean λαμβάνουν τιμές true ή false
int atomaentry=0; //μεταβλητή του αισθητήρα απόστασης που μετράει τα άτομα που
άνοιξαν τη πόρτα με bluetooth
File myFile; //το αρχείο που δημιουργείται για την καταγραφή μετρήσεων στην κάρτα
μνήμης

static const uint8_t monthDays[]={31,28,31,30,31,30,31,31,30,31,30,31}; // εδώ
δηλώνουμε τη μεταβλητή
// monthDays και καθορίζουμε τις μέρες που έχει ο κάθε μήνας του χρόνου έτσι ώστε
όταν δημιουργείται ο
```

```
//φάκελος στη κάρτα μνήμης sd , να κοιτάει την ημερομηνία και να δημιουργεί το αρχείο στον αντίστοιχο μηνά και
//για τις αντίστοιχες μέρες που έχει ο κάθε μήνας.
int tm_second, tm_minute, tm_hour, tm_wday, tm_day, tm_month, tm_year; // δήλωση του χρόνου


int serverConnect_h = 0, serverConnect_m = 5, serverConnect_s = 15;


unsigned long
  lastPolledTime  = 0L, // η τελευταία τιμή που πηρέ από τον server
  sketchTime           = 0L, // CPU milliseconds apo την τελευταία "ερώτηση" στον server


 byte seven_seg_digits[10][7] = { { 1,1,1,1,1,1,0 },  // = 0
                                  { 0,1,1,0,0,0,0 },  // = 1
                                  { 1,1,0,1,1,0,1 },  // = 2
                                  { 1,1,1,1,0,0,1 },  // = 3
                                  { 0,1,1,0,0,1,1 },  // = 4
                                  { 1,0,1,1,0,1,1 },  // = 5
                                  { 1,0,1,1,1,1,1 },  // = 6
                                  { 1,1,1,0,0,0,0 },  // = 7
                                  { 1,1,1,1,1,1,1 },  // = 8
                                  { 1,1,1,0,0,1,1 }   // = 9
                                  };
// Δήλωση για το SevenSegmentDisplay. Ανάλογα με το ποιο νούμερο θέλουμε να απεικονίσει ενεργοποιούνται
//και τα αντίστοιχα segments.
 void setup() {   // η συνάρτηση αυτή χρησιμοποιείται για αρχικοποίηση παραμέτρων. Παρουσιάζεται αναλυτικά
 //παρακάτω:
 Serial.begin(115200);  // Ρυθμός μετάδοσης πληροφορίας συριακής οθόνης
  Serial1.begin(9600);  //  Ρυθμός μετάδοσης πληροφορίας δεύτερης συριακής οθόνης
  pinMode(32, OUTPUT);
  pinMode(33, OUTPUT);
  pinMode(34, OUTPUT);
  pinMode(35, OUTPUT);
  pinMode(36, OUTPUT);
  pinMode(37, OUTPUT);
  pinMode(38, OUTPUT);
  pinMode(39, OUTPUT);
  // Τα παραπάνω pins προετοιμάζονται για να γίνουν εξόδιο του ssd
  pinMode(led, OUTPUT); // Προετοιμασία εξόδου του led
  pinMode(buzzer1, OUTPUT); //Προετοιμασία εξόδου  του buzzer.
  miscInit(); //κλήση της συνάρτησης miscInit
  wifiInit(); // κλήση της συνάρτησης wifiInitnit
```

```
  Serial.println("");
  Serial.println("wifi initializing done!"); // εκτύπωση μηνύματος
  Serial.println("");
  timeCheck();  //κλήση συνάρτησης timeCheck
  sdInit(); // κλήση συνάρτησης sdInit
}
void miscInit(){

  Serial.println("initializing  wifi shield");
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  // στη συνάρτηση καθορίζουμε ότι το trigPin θα είναι έξοδος σήματος του αισθητήρα απόστασης
//δηλαδή πομπός και το echoPin θα είναι  είσοδος του σήματος δηλαδή δέκτης
}
void wifiInit(){
        // Η συνάρτηση αυτή χρησιμοποιείται για να επιτευχθεί η σύνδεση της πλακέτας adafruit, ασύρματα στον δίκτυο.
        // Γίνεται έναρξη λειτουργίας  της πλακέτας  , αναζήτηση διαθέσιμων δικτυών, αναγνώριση ορισμένου δικτύου
        // και έναρξη , πραγματοποίηση σύζευξης. Πιο αναλυτικά:

  Serial.print("Free RAM: ");  //εμφάνιση διαθέσιμης μνήμης ram
  Serial.println(getFreeRam(), DEC);

  // Αρχή λειτουργίας της πλακέτας cc3000
  if (!cc3000.begin())
  {
    Serial.println(F("Couldn't begin()! Check your wiring?"));  //έλεγχος σωστής σύνδεσης της πλακέτας
    while(1);
  }
   uint8_t macAddress[6] = { 0x84, 0xA4, 0x66, 0xF8, 0x3F, 0x5E }; // δήλωση της macAddress της πλακέτας
   if (!cc3000.setMacAddress(macAddress)) //έλεγχος της MacAddress
   {
     Serial.println(F("Failed trying to update the MAC address")); // εκτύπωση μηνύματος σε περίπτωση
//σφάλματος ταυτοποίησης
     while(1);
   }
   void displayDriverMode(void) // συνάρτηση εσωτερικών ρυθμίσεων πλακέτας
{
  #ifdef CC3000_TINY_DRIVER
    Serial.println(F("CC3000 is configure in 'Tiny' mode"));
  #else
    Serial.print(F("RX Buffer : "));
```

```
    Serial.print(CC3000_RX_BUFFER_SIZE);
    Serial.println(F(" bytes"));
    Serial.print(F("TX Buffer : "));
    Serial.print(CC3000_TX_BUFFER_SIZE);
    Serial.println(F(" bytes"));
  #endif
}

uint16_t checkFirmwareVersion(void) //συνάρτηση ελέγχου έκδοσης λογισμικού
{
  uint8_t major, minor;
  uint16_t version;

#ifndef CC3000_TINY_DRIVER   // Εμφάνιση έκδοσης λογισμικού η εμφάνιση
σφάλματος σε περίπτωση
//που δεν μπορεί να γίνει λήψη της σωστής έκδοσης
  if(!cc3000.getFirmwareVersion(&major, &minor))
  {
    Serial.println(F("Unable to retrieve the firmware version!\r\n"));
    version = 0;
  }
  else
  {
    Serial.print(F("Firmware V. : "));
    Serial.print(major); Serial.print(F(".")); Serial.println(minor);
    version = major; version <<= 8; version |= minor;
  }
#endif
  return version;
}
displayMACAddress(); //κλήση συνάρτησης
void displayMACAddress(void) //συνάρτηση displayMACAddress. Ανάκτηση
macAddress η εμφάνιση λάθους σε περίπτωση
// αδύνατης  ανάκτησης
{
  uint8_t macAddress[6];

  if(!cc3000.getMacAddress(macAddress))
  {
    Serial.println(F("Unable to retrieve MAC Address!\r\n"));  //εκτύπωση μηνύματος σε
περίπτωση αδύνατης ανάκτησης
  }
  else
  {
    Serial.print(F("MAC Address: "));   //εκτύπωση της macAddress
    cc3000.printHex((byte*)&macAddress, 6);
  }
```

```
}
  Serial.println(F("\nDeleting old connection profiles"));
 if (!cc3000.deleteProfiles()) {
   Serial.println(F("Failed!"));
   while(1);
 }
 listSSIDResults();
 void listSSIDResults(void) // η συνάρτηση αυτή κάνει αναζήτηση και εμφάνιση διαθέσιμων ασυρμάτων συνδέσεων
{
 uint8_t valid, rssi, sec, index;
 char ssidname[33];  //μέγεθος ονόματος δικτύου μέχρι 33 χαρακτήρες

 Serial.print(F("Networks found: "));    //εκτύπωση διαθέσιμων δικτυών
 Serial.println(index);

Serial.println(F("================================================="));

 while (index) {
   index--; // μετρητής δικτυών

   valid = cc3000.getNextSSID(&rssi, &sec, ssidname); // αναζήτηση επομένου δικτύου

   Serial.print(F("SSID Name    : "));
       Serial.print(ssidname);
   Serial.println();
   Serial.print(F("RSSI         : "));
   Serial.println(rssi);
   Serial.print(F("Security Mode: "));
   Serial.println(sec);
   Serial.println();
 }


Serial.println(F("================================================="));

 cc3000.stopSSIDscan(); //διακοπή αναζήτησης
}
 Serial.print(F("\nAttempting to connect to "));  //εκτύπωση προσπαθείς σύνδεσης στο δίκτυο μας

 Serial.println(WLAN_SSID);  //εμφάνιση ονόματος δικτύου

 if (!cc3000.connectToAP(WLAN_SSID, WLAN_PASS, WLAN_SECURITY)) {   //αν συνδεθεί στο δίκτυο εμφανίζει το παρακάτω μήνυμα
   Serial.println(F("Failed!"));
   while(1);
```

```
  }

  Serial.println(F("Connected!"));
   Serial.println(F(""));


Serial.println(F("================================================"));

  Serial.println(F("Request DHCP"));  // εμφάνιση μηνύματος  για αίτημα του μηχανισμού διαχείρισης πρωτοκόλλων
  while (!cc3000.checkDHCP()) //κλήση συνάρτησης checkDHCP
  {
    delay(100);
  }

  // κλήση συνάρτησης displayConnectionDetails
  while (! displayConnectionDetails()) {
    delay(1000);
  }
bool displayConnectionDetails(void) //η συνάρτηση αυτή δημιουργήθηκε για να εμφανίζει τα  παρακάτω πρωτόκολλα
{
  uint32_t ipAddress, netmask, gateway, dhcpserv, dnsserv; //δήλωση μεταβλητών τύπου unit32_t

  if(!cc3000.getIPAddress(&ipAddress, &netmask, &gateway, &dhcpserv, &dnsserv))  // αν επιστρέψουν μηδενικές τιμές  τότε εμφανίζουμε το κατάλληλο
  //μήνυμα
  {
    Serial.println(F("Unable to retrieve the IP Address!\r\n"));
    return false;
  }
  else  // αλλιώς εμφανίζουμε τα παρακάτω στοιχεία.
  {
    Serial.print(F("\nIP Addr: "));
        cc3000.printIPdotsRev(ipAddress);
    Serial.print(F("\nNetmask: "));
        cc3000.printIPdotsRev(netmask);
    Serial.print(F("\nGateway: "));
        cc3000.printIPdotsRev(gateway);
    Serial.print(F("\nDHCPsrv: "));
        cc3000.printIPdotsRev(dhcpserv);
    Serial.print(F("\nDNSserv: "));
        cc3000.printIPdotsRev(dnsserv);
    Serial.println();
        //εμφάνιση
    return true;
```

```
  }

}
}



void timeCheck()  //συνάρτηση ανάκτησης ώρας από το διαδίκτυο
{
  Serial.println("");
  Serial.println("initializing  arduino time");  // εμφάνιση μηνύματος έναρξης διαδικασίας ανάκτησης ώρας
  Serial.println("");
  if(countdownFlag == 1) {          // η μεταβλητή μόλις γίνει άσσος τότε γίνεται η κλήση της παρακάτω συνάρτησης
    unsigned long t  = getTime(); // Ερώτηση στον Server για την ανάκτηση της ώρας

    if(t) {                   //  αν έχουμε επιτυχία και επιστραφεί η ώρα
      lastPolledTime = t;        // αποθήκευση της τιμής της (t)
      sketchTime     = millis();  // αποθήκευση τιμής ώρας εσωτερικού ρολογιού στη μεταβλητή sketchTime
      countdownFlag      = 0;   // μηδενισμός μεταβλητής για να μην παίρνει συνέχεια την ώρα
      ntpConnectedFlag = 1; // να γίνει άσσος η μεταβλητή αυτή
    }
  }

  if(ntpConnectedFlag) {  //αν η μεταβλητή είναι άσσος

    unsigned long currentTime = lastPolledTime + (millis() - sketchTime) / 1000; // υπολογισμός της τωρινής ώρας χρησιμοποιώντας τη τιμή της
        //μεταβλητής sketchTime  που είχε όταν λάβαμε για πρώτη φορά την ώρα από το διαδίκτυο συν τη διαφορά ώρα που έχει δημιουργηθεί από τη τωρινή
//χρησιμοποιώντας τη συνάρτηση millis που  είναι το εσωτερικό ρολόι του arduino.
    Serial.println(DAYTOSEC - (currentTime - lastPolledTime) );  // εκτύπωση και μετατροπή ώρας σε δευτερόλεπτα

    if( (currentTime - lastPolledTime) > DAYTOSEC ) {  //αν   υπάρχει διαφορά
      countdownFlag      = 1;   // κάνε τη μεταβλητή αυτή άσσο και ανάκτησε την ώρα ξανά από το ιντερνέτ
    }

    Serial.print(F("Current UNIX time: "));  //εμφάνιση τιμής ώρας
    Serial.print(currentTime);
    Serial.println(F(" (seconds since 1/1/1970 UTC)")); //εμφάνιση ότι υπολογίζεται από το 1970
```

```
   secTotimestamp(currentTime); // κλήση συνάρτησης με τι τιμή της μεταβλητής currentTime και επιστρέφει την ώρα τα δευτερόλεπτα και τα λεπτά

   Serial.print("Time in d/m/y h:m.s ");
   Serial.print(tm_day);
   Serial.print("/");
   Serial.print(tm_month);
   Serial.print("/");
   Serial.print(tm_year);
   Serial.print(" ");
   Serial.print(tm_hour);
   Serial.print(":");
   Serial.print(tm_minute);
   Serial.print(".");
   Serial.print(tm_second);
   Serial.println("");

  // εμφανίσεις μέρας , μηνά , έτους , ώρας ,λεπτών ,δευτερολέπτων
  }
 Serial.println("");
  Serial.println("initializing  arduino time done");  //εμφάνιση μηνύματος ολοκλήρωσης υπολογισμού ώρας
  Serial.println("");
}


void secTotimestamp(unsigned long timeInput){  // η συνάρτηση αυτή κάνει τις πράξεις για να βρεθούν χρονολογία, μήνας, μέρα, ώρα ,λεπτά
//και δευτερόλεπτα χρησιμοποιώντας τη τιμή της currentTime. Η ώρα ξεκινάει από το 1970

  uint8_t year;
  uint8_t month, monthLength;
  uint32_t time;
  unsigned long days;
  //δηλώσεις μεταβλητών

  time = (uint32_t)timeInput;
  tm_second = time % 60;
  time /= 60; // πράξη για τα δευτερόλεπτα
  tm_minute = time % 60;
  time /= 60; // πράξη για τις λεπτά
  tm_hour = time % 24;
  time /= 24; // πράξη για την ώρα
  tm_wday = ((time + 4) % 7) + 1; // και για τη μέρα. Κυριακή είναι η πρώτη μέρα
  year = 0;
```

```
  days = 0;

  while((unsigned)(days += (LEAP_YEAR(year) ? 366 : 365)) <= time) { //εδώ
υπολογίζεται η χρονολογία
    year++;
  }
  tm_year = year + 1970;
 // εδώ υπολογίζονται οι μήνες και οι μέρες του κάθε μηνά
  days -= LEAP_YEAR(year) ? 366 : 365;
  time -= days;
  days=0;
  month=0;
  monthLength=0;
  for (month=0; month<12; month++) { //ο πρώτος μήνας είναι ο Ιανουάριος και είναι ο
0
    if (month==1) { // αν η μεταβλητή month είναι 1 τότε είναι ο μήνας Φεβρουάριος και
έχει 28 η 29 ημέρες

      if (LEAP_YEAR(year)) {
        monthLength=29;
      }
      else {
        monthLength=28;
      }
    }
    else {
      monthLength = monthDays[month];
    }

    if (time >= monthLength) {
      time -= monthLength;
    }
    else {
      break;
    }
  }
  tm_month = month + 1; // εισαγωγή στη μεταβλητή tm_month τη  τιμή που έχει ο μήνας
  tm_day = time + 1; // εισαγωγή στη μεταβλητή tm_day τη τιμή που έχει η ημέρα
}


unsigned long getTime(void) { // η συνάρτηση αυτή χρησιμοποιείται για να επιτευχθεί η
σύνδεση με το διακομιστή
//της ώρας και να επιστρέψει την  ώρα

  uint8_t       buf[48];
  unsigned long ip, startTime, t = 0L;
```

```
  Serial.print(F("Locating time server..."));

  if(cc3000.getHostByName("pool.ntp.org", &ip)) { // η συνάρτηση αυτή επιστρέφει την Ip της διεύθυνσης pool.ntp.org
    static const char PROGMEM
      timeReqA[] = { 227,  0,  6, 236 },
      timeReqB[] = {  49, 78, 49,  52 };

    Serial.println(F("\r\nAttempting connection..."));
    startTime = millis();
    do {
      client = cc3000.connectUDP(ip, 123); // εδω ορίζουμε τον  client για επικοινωνήσει με τον διακομιστή ο οποίος
          //βρίσκεται στην ip που βρήκαμε προηγουμένως και χρησιμοποιώντας την πόρτα 123
    } while((!client.connected()) &&
          ((millis() - startTime) < connectTimeout)); // όσο βρίσκεται συνδεδεμένος ο client με τον διακομιστή και
                        //ο χρόνος που είναι συνδεδεμένος είναι μικρότερος από τον χρόνο διακοπής που έχουμε ορίσει τότε μην διακόψεις
                        //την σύνδεση αλλιώς διέκοψε την

    if(client.connected()) { //Αν συνδεθεί ο πελάτης με τον διακομιστή
      Serial.print(F("connected!\r\nIssuing request..."));  // εμφάνισε ότι συνδέθηκε

      memset(buf, 0, sizeof(buf));
      memcpy_P( buf    , timeReqA, sizeof(timeReqA));
      memcpy_P(&buf[12], timeReqB, sizeof(timeReqB));
      client.write(buf, sizeof(buf));

      Serial.print(F("\r\nAwaiting response...")); //
      memset(buf, 0, sizeof(buf));
      startTime = millis();
      while((!client.available()) &&
          ((millis() - startTime) < responseTimeout)); // όσο βρίσκεται συνδεδεμένος ο client  με τον διακομιστή και
                        //ο χρόνος που είναι συνδεδεμένος είναι μικρότερος από τον χρόνο διακοπής που έχουμε ορίσει
      if(client.available()) { // και αν ο πελάτης είναι διαθέσιμος
        client.read(buf, sizeof(buf)); // διάβασε το μέγεθος του buffer
        t = (((unsigned long)buf[40] << 24) |
            ((unsigned long)buf[41] << 16) |
            ((unsigned long)buf[42] <<  8) |
             (unsigned long)buf[43]) - 2208988800UL;
        Serial.print(F("OK\r\n")); //και εμφάνισε οκ
      }
```

```
    client.close(); //αλλιώς κλείσε σύνδεση
   }
  }
  if(!t) Serial.println(F("error")); //αν σου επιστρέψει λάθος
  return t; //επέστρεψε την ώρα
}
```

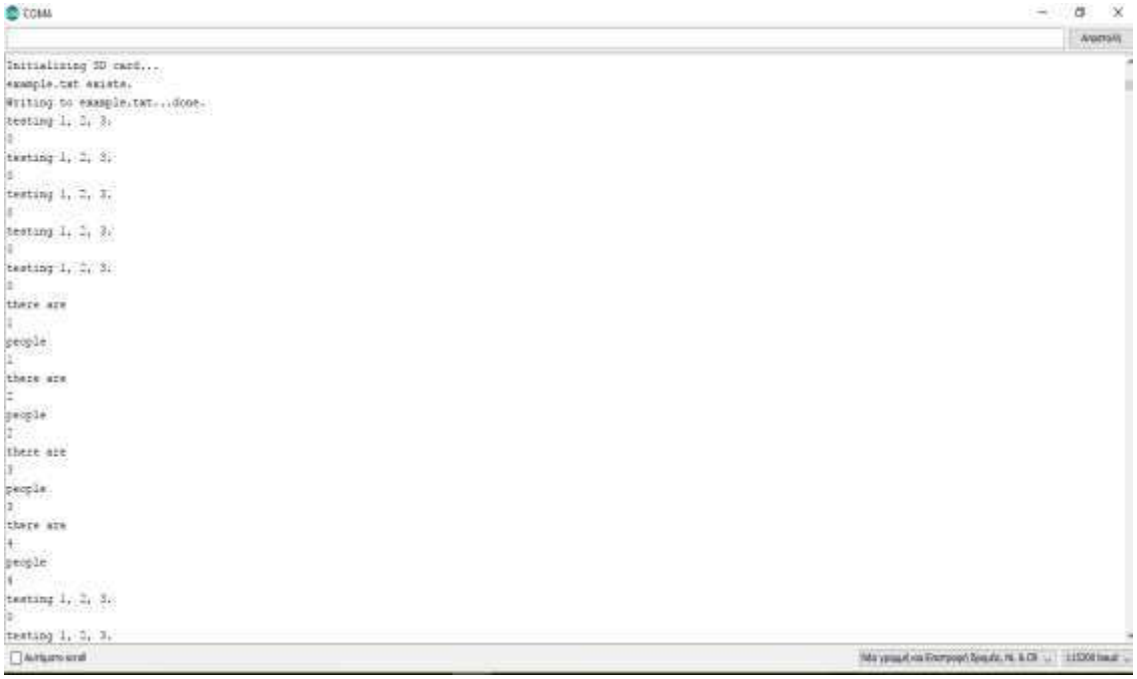
ΕΙΚΟΝΑ 32 SERIAL MONITOR ΣΥΝΔΕΣΗΣ ΤΟΥ CC3000 ΣΤΟ WIFI

ΕΙΚΟΝΑ 33 SERIAL MONITOR ΛΗΨΗΣ ΩΡΑΣ ΑΠΟ ΤΟ ΔΙΑΔΙΚΤΥΟ

```
//-------------------------------------------------------------
```

```
void sdInit(){  //συνάρτηση για την αρχικοποίηση της κάρτας μνήμης

  Serial.println("Initializing SD card..."); // εμφάνισε ότι έχει ξεκινήσει αρχικοποίηση

  pinMode(10, OUTPUT); // θέτουμε το pin 10 σε ετοιμότητα

  if (!SD.begin(4)) {  // αν δεν ξεκινήσει η διαδικασία σε έως 4 δευτερόλεπτα
    Serial.println("initialization failed!");  //επέστρεψε λάθος
    return;
  }


  if (SD.exists("example.txt")) {  //έλεγχος αν υπάρχει αρχείο example.txt
    Serial.println("example.txt exists.");  //αν υπάρχει εμφάνισε το μήνυμα αυτό
  }
  else {
    Serial.println("example.txt doesn't exist."); // αλλιώς εμφάνισε ότι δεν υπάρχει
  }


  myFile = SD.open("example.txt", FILE_WRITE); // άνοιξε καινούριο φάκελο,
δημιούργησε το example.txt μέσα σε αυτόν
  myFile.close();  //κλείσε το φάκελο

  myFile = SD.open("example.txt", FILE_WRITE);
  //άνοιξε τον φάκελο  και γράψε μέσα στο αρχείο

  if (myFile) { //αν υπάρχει το example.text
    Serial.print("Writing to example.txt..."); //εμφάνισε ότι ξεκινάει να γραφεί
    myFile.println("have passed "); //ότι πέρασαν χ άτομα
    myFile.println(atoma); //εκτύπωσε άτομα
    myFile.close(); //κλείσε φάκελο
    Serial.println("done.");
  }
  else {
    //αν υπάρξει κάποιο πρόβλημα εμφάνισε το παρακάτω
    Serial.println("error opening example.txt");
  }

  // άνοιξε πάλι το αρχείο
  myFile = SD.open("example.txt");
  if (myFile) {  //όταν το ανοίξει το αρχείο


    //τότε διάβασε ότι είναι καταγεγραμμένο στο αρχείο μερί το τέλος
    while (myFile.available()) {
```

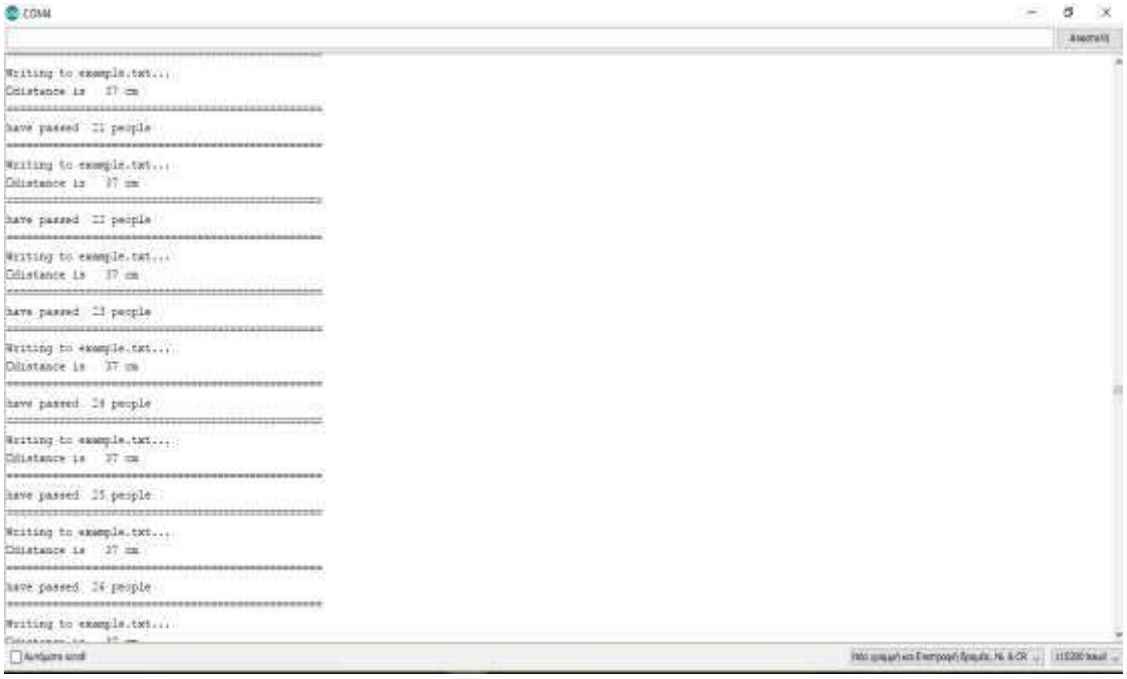
```
      Serial.write(myFile.read());
    }
    // κλείσε τον φάκελο
    myFile.close();
  }
  else {
    // αν δεν ανοίξει τον φάκελο εμφάνισε το παρακάτω
    Serial.println("error opening example.txt");
  }

}
```



ΕΙΚΟΝΑ 34 SERIAL MONITOR ΑΡΧΙΚΟΠΟΙΗΣΗΣ ΚΑΡΤΑΣ ΜΝΗΜΗΣ SD

```
void loop() {  // κύρια  συνάρτηση

  long duration, distance;   //δήλωση μεταβλητών, που χρησιμοποιούμε για τον αισθητήρα που παίρνει τις μετρήσεις
    digitalWrite(trigPin, LOW);  //το trigPin το κάνουμε μηδέν
    delayMicroseconds(2);   //μια αναγκαία καθυστέρηση που χρειάζεται ο αισθητήρας υπερήχων
    digitalWrite(trigPin, HIGH); // παροχή τάσης στο trigPin
    delayMicroseconds(10);   // μια αναγκαία καθυστέρηση που χρειάζεται ο αισθητήρας υπερήχων
    digitalWrite(trigPin, LOW);//το trigPin το κάνουμε μηδέν
```

```
duration = pulseIn(echoPin, HIGH); // παρέχουμε παλμό στο echoPin και το αποθηκεύουμε στη μεταβλητή
distance = (duration/2) / 29.1; // η απόσταση είναι η διάρκεια που κάνει το σήμα από τη στιγμή που στέλνεται μαζί με την αντανάκλαση του
//στο τυχόν εμπόδιο" και επιστρέφεται στον δεκτή δια δυο. ( το 29.1 σχετίζεται με την ταχύτητα του σήματος. Δίνεται δεδομένο για τον
//συγκεκριμένο αισθητήρα
if (distance < 100) {
```



ΕΙΚΟΝΑ 35 SERIAL MONITOR ΕΜΦΑΝΙΣΗΣ ΑΤΟΜΩΝ ΠΟΥ ΕΧΟΥΝ ΠΕΡΑΣΕΙ

```
// αν η απόσταση είναι μικρότερη από 100 εκατοστά

  atoma=atoma+1; //αύξησε τα άτομα κατά ένα
    delay(800);// μια μικρή καθυστέρηση που χρειάζεται έτσι ώστε να μην "πιάνει" πολλές φορές το ίδιο άτομο.
    tag==false; // εδώ μετατρέπουμε τη μεταβλητή tag σε false από true για να μην ανάβει συνέχεια το seven segments display

  writeFile(); // εδώ καλούμε την συνάρτηση writeFile
    void writeFile(){ //η συνάρτηση αυτή αποθηκεύει την μέτρηση στη κάρτα μνήμης
  myFile = SD.open("example.txt", FILE_WRITE); // μόλις ανοίξει το αρχείο τότε γράψε τη μέτρηση στο αρχείο example.txt
```

```
  if (myFile) { //εδώ εκτυπώνουμε κάποια μηνύματα με την καταγεγραμμένη τιμή στην κάρτα μνήμης και τα άτομα που έχουν περάσει
  // ως εκείνη την στιγμή
    Serial.println("Writing to example.txt...");
    myFile.println("there are  ");
    myFile.println(atoma);
    myFile.println("people  ");
    Serial.write(myFile.println(atoma));
    myFile.close(); // μόλις  γίνει η καταγραφή των ατόμων στην κάρτα μνήμης τότε κλείνουμε το αρχείο
  }
}
     Serial.print("distance is   ");
     Serial.print(distance); // εκτύπωση της απόστασης με το κατάλληλο μήνυμα
          Serial.print(" cm ");
          Serial.println(F(""));
          Serial.print("have passed  ");// εκτύπωση των ατόμων που έχουν περάσει με το κατάλληλο μήνυμα
     Serial.print(atoma);
          Serial.print(" people ");
          Serial.println(F(""));

Serial.println(F("================================================="));
          }

    // εδώ ξεκινάμε να ελέγχουμε αν κάποια συσκευή bluetooth έχει συνδεθεί με το module μας
    else if ( Serial1.available() )   { //μόλις υπάρξει η σύνδεση τότε το πρόγραμμα μας προχωράει μέσα στην if

    while (tag==false){ // για να γίνει η εκτέλεση του παρακάτω προγράμματος απαιτείται η μεταβλητή tag να είναι false
    {
        c = Serial1.read(); // εδώ διαβάζουμε εάν έχει σταλθεί κάτι από το Serial monitor και την εφαρμογή στην android
                  //συσκευή μας και την αποθηκεύουμε στην μεταβλητή c
        Serial.write( c ); // εμφάνιση της μεταβλητής c
        digitalWrite(led, HIGH);// μόλις υπάρξει η "εντολή' από την android συσκευή μας τότε ανάβουμε το λαμπάκι αυτό
                  // το οποίο χρησιμοποιούμε αντί για την πόρτα εισόδου του καταστήματος

        ssd(); // εδώ κάνουμε κλήση της συνάρτησης ssd
                  void sevenSegWrite(byte digit) { // η συνάρτηση αυτή χρησιμοποιείται για να κάνει τη αντιστροφή μέτρηση του seven segments display
  byte pin = 32;
  for (byte recount = 0; segCount < 7; ++segCount) {
    digitalWrite(pin, seven_seg_digits[digit][segCount]);
```

```
    ++pin;
  }
}

void ssd() { // η συνάρτηση αυτή χρησιμοποιείται κυρίως για να ενεργοποιηθεί το ssd και να αρχίζει η μέτρηση των ατόμων που
// εισήχθησαν από την στιγμή που άνοιξε η πόρτα μέχρι το πέρας των 9 δευτερολέπτων και σε περίπτωση που περάσουν πάνω από 1 άτομο
//τότε να χτυπήσει ο συναγερμός (δηλαδή το buzzer που έχουμε συνδέσει)
      Serial.println("");
      Serial.println("door is oppenned via bluetooth   ");
   for (byte count = 10; count > 0; --count) {
         Serial.println("");
         Serial.print("remaining :   ");
         Serial.print(count);
         Serial.print("  seconds   ");
         Serial.println("");



   delay(800);

   long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  if (distance < 100) {
    atomaentry=atomaentry+1; //η μεταβλητή αυτή είναι διαφορετική από την μεταβλητή άτομα και χρησιμοποιείται μόνο για να
        //μετράμε  τα άτομα που περνάνε μέσα από το bluetooth
```

ΕΙΚΟΝΑ 36 SERIAL MONITOR ΜΕΤΡΗΣΗΣ ΑΤΟΜΩΝ ΟΤΑΝ ΑΝΟΙΓΕΙ Η ΠΟΡΤΑ ΑΠΟ ΤΟ BLUETOOTH

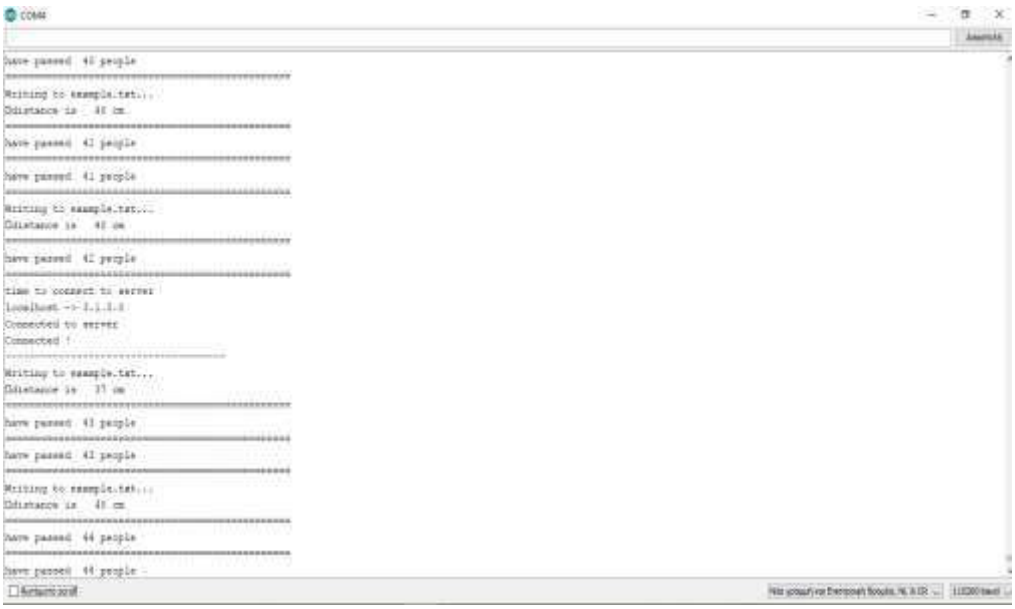
```
Serial.print("distance is   ");
Serial.print(distance);
     Serial.print(" cm ");
if (atomaentry>=2){ // αν τα άτομα είναι πάνω από 1 τότε ήχησε το buzzer
  tone(buzzer1, 10); //στείλε σήμα 10hz .
```

ΕΙΚΟΝΑ 37 SERIAL MONITOR ΟΤΑΝ ΑΝΟΙΓΕΙ Η ΠΟΡΤΑ ΑΠΟ ΤΟ BLUETOOTH

```
        Serial.println("");
        Serial.println("buzzer is on");
        Serial.println("");
        delay(500);        //καθυστέρηση 0.5 δευτερόλεπτα
       tag = true; //  μετατρέπουμε τη μεταβλητή tag   σε true  έτσι ώστε να βγει από την
while
     }
  }
       sevenSegWrite(count - 1); // μείωση κατά 1 του νούμερο στο ssd
if (count ==1){
   tag =true;} // εάν έχει περάσει ο χρόνος και έχει φτάσει η μεταβλητή count σε 1 χωρίς
να έχει μετρήσει παν από 1 άτομο τότε
   //κάνε τη μεταβλητή tag σε true για να βγει έξω από την while

Serial.print("got in "); // καταλληλά μηνύματα για να μας πληροφορήσουν για τα άτομα
που πέρασαν από την στιγμή που άνοιξε η πόρτα
        Serial.print(atomaentry);
     Serial.print(" people with door open ");
       Serial.println(F(""));

Serial.println(F("=================================================="));
       sevenSegWrite(count - 1);
if (count ==1){ // εάν φτάσει η μεταβλητή count  στο νούμερο 1  και έχει χτυπήσει το
buzzer  τότε  σταμάτησε τον ήχο του buzzer
   noTone(buzzer1);
   tag =true;} //και κάνε το tag πάλι σε true

}
// αυτά είναι τα pin του ssd τα οποία τα σβήνουμε μόλις τελειώσει η μέτρηση
 pinMode(32, LOW);
  pinMode(33, LOW);
  pinMode(34, LOW);
  pinMode(35, LOW);
  pinMode(36, LOW);
  pinMode(37, LOW);
  pinMode(38, LOW);
  pinMode(39, LOW);
  digitalWrite(led, LOW); // εδώ σβήνουμε και το led το οποίο σημαίνει ότι κλείνουμε την
πόρτα
  atomaentry=0; // και μηδενίζουμε την μεταβλητή atomaentry για να μετρήσει και την
άλλη φορά από το μηδέν εάν
  // υπάρξει πάλι σύνδεση με κάποια bluetooth  συσκευή
  tag ==true; // εξασφαλίζουμε ότι βγαίνουμε από την while
}

        digitalWrite(led, LOW);
         writeDot(0);  // εδώ ξεκινάμε με το ssd να είναι σβησμένο
```

```
                void writeDot(byte dot) {
        digitalWrite(39, dot);
}
   }
   }
  }
   if ( Serial.available() )  {  // διαβάζει αν υπάρχει κάποια απάντηση από το serial monitor προς την εφαρμογή της android συσκευής
        //έτσι ώστε να κλείσει και με αυτόν τον τρόπο το λαμπάκι
    c = Serial.read();
       Serial1.write( c );
            digitalWrite(led, LOW); //κλείνει το λαμπάκι
  delay(100);
   }

 if((long)(millis() - rolltime) >= 0) {  // εδώ ελέγχουμε αν έχει περάσει η καθορισμένη ώρα για να πραγματοποιηθεί η σύνδεση με τον διακομιστή
 // και να γίνει η αποστολή της μεταβλητής atoma στη βάση δεδομένων

   Serial.println("time to connect to server ");
   connectServer(); //κλήση συνάρτησης για σύνδεση με διακομιστή
       void connectServer() {   //συνάρτηση σύνδεσης με διακομιστή
  uint32_t ip;
  //προσπάθεια εύρεσης της ip του Server
  Serial.print(WEBSITE);  // εμφάνισε το WEBSITE
  Serial.print(F(" -> "));
  while (ip == 0) {  //αν δεν έχει βρεθεί η ip  να μην προχωρήσει
   if (! cc3000.getHostByName(WEBSITE, &ip)) {   //αν δεν τα βρει τότε εμφάνισε το παρακάτω μήνυμα
     Serial.println(F("Couldn't resolve!"));
   }
   delay(500);
  }

  cc3000.printIPdotsRev(ip); /// εκτύπωση ip
  Serial.println("");

  ip=cc3000.IP2U32(192, 168, 1, 7);  // δη δήλωση της ip του server μας

  Adafruit_CC3000_Client www = cc3000.connectTCP(ip, 80); //επικοινωνία πελάτη – διακομιστή στην πόρτα 80
  if (www.connected()==1) { //μόλις συνδεθεί
    char temp[30]; // φτιάξε πίνακα 30 θέσεων
    char temp1[30];     // φτιάξε άλλον ένα πίνακα 30 θέσεων
    sprintf(temp, "%d", atoma); // // προσθήκη μεταβλητής σε προσωρινό πίνακα και μετατροπή αυτής σε strings από integers
    sprintf(temp1, "%d", shop_id); //προσθήκη μεταβλητής σε προσωρινό πίνακα και
```

```
μετατροπή αυτής σε strings από integers

        //μηνύματα εκτύπωσης επιτυχούς σύνδεσης
   Serial.println(F("Connected to server"));
   Serial.println(F("Connected !"));
        // για να επιτύχουμε το ανέβασμα των τιμών στη βάση δεδομένων χρησιμοποιούμε τη μέθοδο "GET" .
        //η μέθοδος αυτή χρησιμοποιεί ένα url στο οποίο προσθέτουμε στο τέλος της μεταβλητής το "=" και μετά
        //το νούμερο το οποίο θέλουμε να ανεβάσουμε
   www.fastrprint(F("GET "));
   www.fastrprint(WEBPAGE);
   www.fastrprint(temp);
   www.fastrprint("&&shop_id=");
   www.fastrprint(temp1);
   www.fastrprint(F(" HTTP/1.1\r\n"));
   www.fastrprint(F("Host: "));
   www.fastrprint(WEBSITE);
   www.fastrprint(F("\r\n"));
   www.fastrprint(F("Connection: close\r\n\r\n"));  //κλείσιμο σύνδεσης
 }
```



ΕΙΚΟΝΑ 38 SERIAL MONITOR ΕΜΦΑΝΙΣΗΣ ΜΗΝΥΜΑΤΟΣ ΟΤΑΝ ΑΝΕΒΑΖΕΙ ΜΕΤΡΗΣΗ ΣΤΟΝ ΣΕΡΒΕΡ ΜΑΣ

```
 else { // αλλιώς αν δεν πραγματοποιηθεί η σύνδεση πελάτη διακομιστή
   Serial.println(F("Connection failed"));    //εμφάνισε το μήνυμα αυτό
   return;
 }
```

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

}

   rolltime += FIVEMIN;  // ανέβασε κάθε πέντε λεπτά
 }

 else {   // εδώ εμφανίζει τα άτομα
  delay(800);
   Serial.print("have passed  ");
        Serial.print(atoma);
        Serial.print(" people ");
        Serial.println(F(""));

Serial.println(F("=================================================="));
          delay(800);

       }
```

Βιβλιογραφία

1) https://www.wikipedia.org
2) https://www.stackoverflow.com
3) https://www.arduino.cc
4) https://www.adafruit.com
5) https://www.instructables.com