ΠΑΝΕΠΙΣΤΉΜΙΟ ΔΥΤΙΚΉΣ ΑΤΤΙΚΉΣ

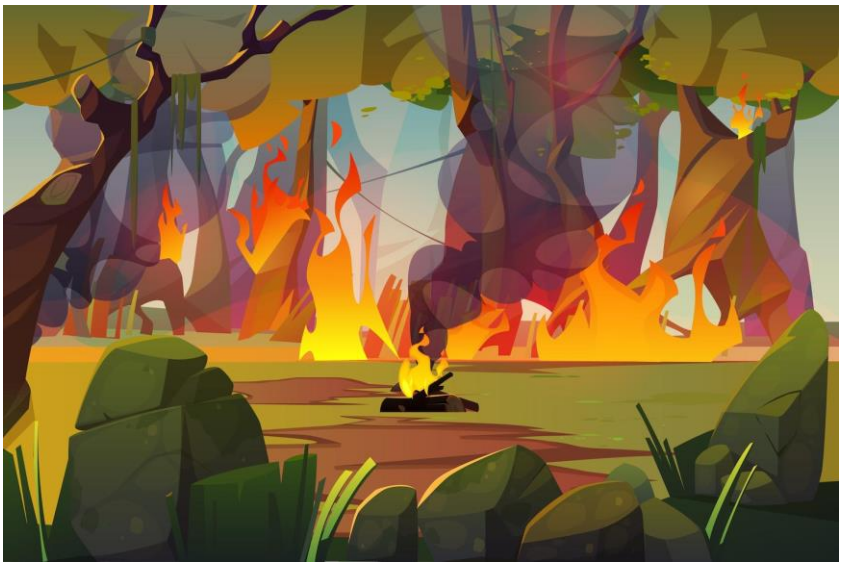
ΣΧΟΛΉ ΜΗΧΑΝΙΚΏΝ

ΤΜΉΜΑ ΗΛΕΚΤΡΟΛΌΓΩΝ & ΗΛΕΚΤΡΟΝΙΚΏΝ ΜΗΧΑΝΙΚΏΝ

ΚΑΤΕΥΘΥΝΣΗ ΗΛΕΚΤΡΟΝΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Διπλωματική Εργασία

Ανάπτυξη Πλατφόρμας Έγκαιρης Προειδοποίησης Δασικών Πυρκαγιών Βασισμένη σε Τεχνολογία IoT



Φοιτητής: Μάντζιος Παύλος

ΑΜ: 46366

Επιβλέπων Καθηγητής:

Παπαγέωργας Παναγιώτης

Καθηγητής

ΑΘΗΝΑ-ΑΙΓΑΛΕΩ, ΣΕΠΤΕΜΒΡΙΟΣ 2022

UNIVERSITY OF WEST ATTICA

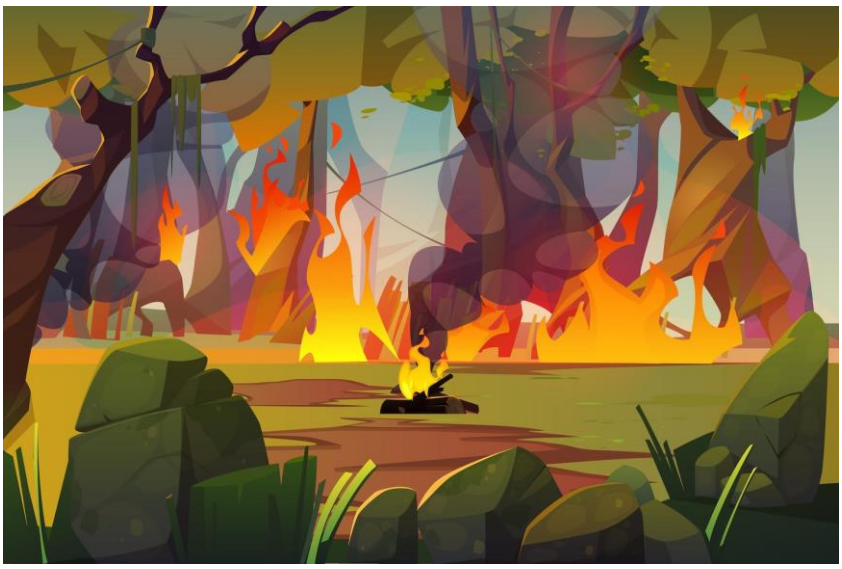
FACULTY OF ENGINEERING

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

SPECIALIZATION AT ELECTRONICS AND COMPUTER SYSTEMS

Diploma Thesis

Development of an Early Warning Platform for Prevention of Forest Fires Based on IoT Technologies



Student: Mantzios Pavlos

Registration Number: 46366

Supervisor:

Papageorgas Panagiotis

Professor

ATHENS-EGALEO, SEPTEMBER 2022

Η Διπλωματική Εργασία έγινε αποδεκτή και βαθμολογήθηκε από την εξής τριμελή επιτροπή:

(Ονοματεπώνυμο), (βαθμίδα)	(Ονοματεπώνυμο), (βαθμίδα)	(Ονοματεπώνυμο), (βαθμίδα)
Παναγιώτης Παπαγέωργας Καθηγητής Digitally signed by Panagiotis Papageorgas Date: 2022.10.11 22:23:46 +03'00' (Υπογραφή)	Γεώργιος Βόκας Καθηγητής Georgios Vokas Digitally signed by Georgios Vokas Date: 2022.10.11 23:00:09 +03'00' (Υπογραφή)	Δημήτριος Πυρομάλης Επίκουρος Καθηγητής Dimitrios Pyromalis Digitally signed by Dimitrios Pyromalis Date: 2022.10.12 08:21:02 +03'00' (Υπογραφή)

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Παύλος Μάντζιος του Αλεξάνδρου, με αριθμό μητρώου 46366 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής ΜΗΧΑΝΙΚΩΝ του Τμήματος ΗΛΕΚΤΡΟΛΟΓΩΝ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ,

δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του διπλώματός μου.

Επιθυμώ την απαγόρευση πρόσβασης στο πλήρες κείμενο της εργασίας μου μέχρι …… και έπειτα από αίτησή μου στη Βιβλιοθήκη και έγκριση του επιβλέποντος καθηγητή.»

Ο Δηλών

Παύλος Μάντζιος

Ευχαριστίες

Ευχαριστώ τον καθηγητή μου, κ. Παναγιώτη Παπαγέωργα, που μου έδωσε την ευκαιρία να πραγματοποιήσω διπλωματική εργασία υπό την επίβλεψή του. Θέλω να εκφράσω την αγάπη μου στην οικογένειά μου για τη στήριξη, συμπαράσταση και κατανόηση που μου έδωσαν καθ΄ όλη την διάρκεια της πανεπιστημιακής μου σταδιοδρομίας. Τέλος, θα ήθελα να ευχαριστήσω τον θείο μου, Παναγιώτη Τριανταφύλλου, για τη βοήθεια που μου προσέφερε όποτε τη χρειαζόμουν στη διάρκεια της φοίτησής μου.

Περίληψη

Στην εργασία αυτή εξετάζεται η ανάπτυξη μιας πλατφόρμας έγκαιρης προειδοποίησης δασικών πυρκαγιών βασισμένη σε τεχνολογία IoT. Οι δασικές πυρκαγιές αποτελούν μια από τις πιο επικίνδυνες φυσικές καταστροφές που απειλούν το φυσικό περιβάλλον και την ανθρωπότητα. Οι αιτίες για τις οποίες δημιουργούνται εξαρτώνται από πολλούς παράγοντες. Ανθρωπογενείς παράγοντες, όπως η παράνομη καύση νέκρας ύλης, ψησταριές, κάμπινγκ, δημόσια έργα και κακή πολεοδομία, καθώς και φυσικούς παράγοντες όπως η γεωγραφική τοποθεσία μιας περιοχής, τα καιρικά φαινόμενα, η βλάστηση και η μορφολογία του δάσους. Μια δασική πυρκαγιά είναι μια ανεξέλεγκτη φωτιά, άρα, για να προβλεφθεί η εμφάνιση της, θα πρέπει να ανιχνευθούν τα σημάδια που χαρακτηρίζουν μια φωτιά. Αυτά είναι ο καπνός, η μεγάλη θερμοκρασία, η μείωση της υγρασίας στο περιβάλλον, έντονη φωτεινή πηγή και μονοξείδιο και διοξείδιο του άνθρακα. Για να επιτευχθεί η πρόληψη μιας πυρκαγιάς, σχεδιάστηκε ένα σύστημα βασισμένο στην τεχνολογία του Διαδικτύου των Πραγμάτων (IoT) και την τεχνολογία LoRa. Η τεχνολογία του Διαδίκτυού των Πραγμάτων σε συνδυασμό με την τεχνολογία LoRa δίνει την δυνατότητα σε έναν χρήστη να λάβει δεδομένα από τους αισθητήρες ανίχνευσης της πυρκαγιάς σε μεγάλη απόσταση και να μπορεί να τα βλέπει από οποιοδήποτε σημείο και σε οποιαδήποτε χρονική στιγμή. Το σύστημα αυτό ακολουθεί την αρχιτεκτονική LoRa, δηλαδή αποτελείται από κόμβους (nodes). Στη συγκεκριμένη κατασκευή υπάρχει ένας κόμβος. Αυτός αποτελείται από τέσσερις αισθητήρες που συλλέγουν δεδομένα θερμοκρασίας, υγρασίας, μονοξείδιου και διοξείδιο του άνθρακα από το περιβάλλον, ένα Arduino, το LoRa module (transmitter), κεραία και μια μπαταρία. Έπειτα, τα δεδομένα αυτά στέλνονται μέσω ενός κόμβου - δέκτη που έχει κατασκευαστεί από ένα δεύτερο Arduino, LoRa module (receiver) και κεραία σε μια βάση δεδομένων από την οποία ο χρήστης μέσα από ένα περιβάλλον-application server μπορεί να καλέσει τα δεδομένα, να τα επεξεργαστεί μέσα από την οθόνη του υπολογιστή του ή του κινητού του τηλεφώνου.

Λέξεις – κλειδιά

Φωτιά, πυρκαγιά, κόμβος, δέκτης LoRa, LoRa Transmitter, LoRa, ThingSpeak

Abstract

This work examines the development of a forest fire early warning platform based on IoT technology. Forest fires are one of the most dangerous natural disasters that threaten the natural environment and humanity. The causes for which they are created depend on many factors. Anthropogenic factors such as illegal incineration, barbecues, campsites, public works and poor urban planning, and also natural factors such as the geographical location of an area, weather, vegetation and forest morphology. A forest fire is an uncontrollable fire, so in order to predict its appearance, the signs that characterize a fire must be detected. These are: smoke, high temperature and reduction of humidity in the environment. A system based on Internet of Things (IoT) technology and LoRa technology was designed to prevent a fire. Internet of Things technology combined with LoRa technology enables a user to receive data from fire detection sensors over long distances and to be able to see them from anywhere, at any time. This system follows the LoRa architecture, it consists of nodes. In this project there is only one node. It consists of four sensors that collect temperature data, humidity data, monoxide and carbon dioxide data from the environment, an Arduino, a LoRa module, an antenna and a battery. This data is then sent via a receiver built from a second Arduino, LoRa module and antenna to a database from which the user -through an application server environment- can call the data, process it through a computer screen or mobile phone.

Keywords

Fire, forest fire, node, LoRa Receiver, LoRa Transmitter, LoRa, ThingSpeak

ΠΕΡΙΕΧΟΜΕΝΑ

Κατάλογος Πινάκων

Κατάλογος Εικόνων

ΕΙΣΑΓΩΓΗ

Οι δασικές πυρκαγιές αποτελούν μια από τις σημαντικότερες καταστροφές που φέρνουν σε κίνδυνο τους ανθρώπους, τα ζώα και το φυσικό περιβάλλον σε ολόκληρο τον κόσμο. Κάθε καλοκαίρι, πάνω από 5.000 πυρκαγιές καταστρέφουν περίπου 600.000 με 800.000 εκτάρια γης στην περιοχή της Μεσογείου Θάλασσας, δημιουργώντας πολλά προβλήματα για τη χλωρίδα, την πανίδα και τους ανθρώπους που επηρεάζονται [1].

Η ετήσια αύξηση της θερμοκρασίας του πλανήτη στο τελευταίο τέταρτο του 20ου αιώνα έχει ανεβάσει την πιθανότητα δημιουργίας πυρκαγιών λόγω των σπανιότερων βροχοπτώσεων που έχουν ως αποτέλεσμα να δημιουργούνται μεγάλες ποσότητας νεκράς ύλης από την ξηρασία της βλάστησης. Επιπρόσθετα, η αύξηση της θερμοκρασίας μειώνει το ποσοστό υγρασίας στο αέρα και το περιβάλλον με αποτέλεσμα τα φυτά, τα δέντρα και τα χόρτα να καίγονται πολύ πιο εύκολα και πολύ πιο γρήγορα, το οποίο έχει ως αποτέλεσμα να αυξάνονται οι πιθανότητες για τη δημιουργία ανεξέλεγκτων πυρκαγιών [2].

Στην Ελλάδα οι πυρκαγιές είναι σχεδόν πάντα αναμενόμενες κάθε χρόνο. Τον Ιούλιου του 2018 και τον Αύγουστο του 2021 ξέσπασαν οι μεγαλύτερες και πιο θανατηφόρες πυρκαγιές στην Ελληνική ιστορία. Οι πυρκαγιές το καλοκαίρι του 2021 έπληξαν την Αττική, την Πελοπόννησο και την Εύβοια. Δύο από τα μέτωπα που εκδηλώθηκαν ήταν τα πιο καταστροφικά του καλοκαιριού. Την Τρίτη 3 Αυγούστου του 2021, στη 13:25 εμφανίστηκε δασική πυρκαγιά στην περιοχή της Βαρυμπόμπης, η οποία επεκτάθηκε από τους δυνατούς ανέμους στις γύρω περιοχές και έφτασε και στην ίδια την πόλη της Βαρυμπόμπης, οπού η φωτιά έκαψε την πλατεία, τα σπίτια, τα καταστήματα και το δάσος που την περιτριγυρίζει. Το δεύτερο μέτωπο εκδηλώθηκε στη βόρεια Εύβοια την ίδια μέρα λίγες ώρες αργότερα, όπου η φωτιά λόγω των ισχυρών ανέμων βγήκε πολύ γρήγορα εκτός ελέγχου, με αποτέλεσμα να κατευθυνθεί με μεγάλη ταχύτητα στη θάλασσα, από τη μία πλευρά του Ευβοϊκού μέχρι την άλλη πλευρά του Αιγαίου. Συνολικά, η πυρκαγιά αυτή κατέστρεψε 500.000 στρέμματα δάσους και μαζί τους σπίτια και ολόκληρα χωριά.

Τη Δευτέρα 23 Ιουλίου του 2018, στις 12:03 μ.μ. ξέσπασε φωτιά στα Γεράνεια Όρη στην περιοχή της Κινέτας και λίγες ώρες αργότερα στα σύνορα της Πεντέλης με Καλλιτεχνούπολης Ραφήνας – Πικερμίου μεταξύ Νταού Πεντέλης και Διώνης [3]. Οι δύο φωτιές πολύ γρήγορα βγήκαν εκτός ελέγχου και παρά τις προσπάθειες της πυροσβεστικής και των πολιτών, επεκτάθηκαν πολλά χιλιόμετρα από όπου εκδηλώθηκαν και κατέστρεψαν τα πάντα στο δρόμο τους, αγροτικές περιοχές, δάση, σπίτια και αυτοκίνητα και κατέληξαν στη θάλασσα. Συγκεκριμένα:

- Η φωτιά στα Γέρανα λόγω της μεγάλης ταχύτητας των ανέμων έγινε γρήγορα ανεξέλεγκτη και κινήθηκε προς τους οικισμούς, καθώς πέρασε ακόμη και πάνω από την Ολυμπία Οδό καίγοντας συνολικά 60.000 στρέμματα γης.
- Η δεύτερη φωτιά βγήκε εκτός ελέγχου την ίδια ώρα που είχε βγει και η φωτιά στα Γέρανα και λόγω των ισχυρών ανέμων η φωτιά γρήγορα κατευθύνθηκε βόρια στην ευρύτερη περιοχή της Ραφήνας έως τους οικισμούς στο Κόκκινο Λιμανάκι, το Μάτι και κατέληξε στη θάλασσα. Η πυρκαγιά έκαψε ολοκληρωτικά το Κόκκινο Λιμανάκι, ενώ και στο Μάτι εκατοντάδες σπίτια και οχήματα τυλίχθηκαν στις φλόγες. Δυστυχώς, 102 άνθρωποι έχασαν τη ζωή τους και χιλιάδες άλλοι τα σπίτια και τις περιουσίες τους.

Τα αίτια αυτών των δύο τραγικών καταστροφών ήταν ο συνδυασμός περιβαλλοντικών και ανθρωπογενών παραγόντων. Ειδικότερα, η γεωγραφία των δύο περιοχών ήταν ευάλωτη, αφού υπήρχαν μεγάλες εκτάσεις από χωράφια και δασάκια με αποξηραμένη βλάστηση, καθιστώντας εύκολη την αναζωπύρωση και την επέκταση της πυρκαγιάς. Οι δυνατοί άνεμοι που έφτασαν 12 μποφόρ και οι υψηλές θερμοκρασίες που άγγιξαν τους 40°C, όπως και η σχετική υγρασία του περιβάλλοντος η οποία βρισκόταν σε χαμηλά επίπεδα κοντά στο 19% [4], οδήγησαν την πυρκαγιά με μεγάλη ταχύτητα και πολύ υψηλές θερμοκρασίες δεκάδες χιλιόμετρα βόρεια προς τις κατοικημένες περιοχές.

Σύμφωνα με τις αρχές και την αναφορά του 6ου Τακτικού Ανακριτή Αθηνών [5], η πυρκαγιά που εμφανίστηκε στο Μάτι προκλήθηκε από ανθρώπινη αμέλεια, για την οποία κατηγορείται ένας 65χρονος κάτοικος της Νταού Πεντέλης, ο οποίος έβαλε φωτιά για να κάψει κλαδιά και δεν το διαχειρίστηκε υπεύθυνα. Από την άλλη πλευρά, η πυρκαγιά που εκδηλώθηκε στην Κινέτα θεωρείται πως ξεκίνησε από βραχυκύκλωμα σε κολώνα της ΔΕΗ [6]. Όμως δεν υπάρχουν ακριβείς αιτίες για τις δύο πυρκαγιές και οι παραπάνω κατηγορίες δεν αποτελούν οριστικές αιτίες για την καταστροφή που προέκυψε το καλοκαίρι του 2018.

Υπάρχουν πολλοί παράγοντες που συμβάλλουν στη δημιουργία μιας πυρκαγιάς. Σίγουρα ο πιο συνήθης λόγος είναι η ανθρώπινη αμέλεια, όπως είναι οι φωτιές για την καύση χόρτων, τα πεταμένα τσιγάρα, οι ψησταριές στην εξοχή, η κακή πολεοδομία, τα εκτεθειμένα καλώδια ή/και σπινθήρες από εργαλεία. Όμως για να μετατραπεί η φωτιά σε πυρκαγιά σημαντικό ρόλο παίζει ο τύπος της βλάστησης, ο οποίος επηρεάζει τη συμπεριφορά της πυρκαγιάς, όπως και τα καιρικά φαινόμενα που συμβάλλουν στην ταχύτητα εξάπλωσης.

Αντικείμενο της Διπλωματικής Εργασίας

Το πρόβλημα που χρήζει αντιμετώπισης στην εργασία αυτή είναι ο έγκαιρος εντοπισμός μιας πυρκαγιάς.

- ***Τι είναι όμως μια πυρκαγιά;***

Μια πυρκαγιά είναι η μια ανεξέλεγκτη φωτιά και προκαλείται από μια μη ελεγχόμενη καύση ύλης που τροφοδοτείται με οξυγόνο. Εφόσον η φωτιά και, κατ' επέκταση, η πυρκαγιά είναι μια χημική αντίδραση, αυτή έχει ως αποτέλεσμα να παράγει φως, μεγάλη ποσότητα θερμότητας και καπνό.

Τα αποτελέσματα αυτής της χημικής εξώθερμης αντίδρασης μπορούμε να τα μετρήσουμε με ειδικούς αισθητήρες και να ανιχνεύσουμε μια πυρκαγιά σε πρώιμο στάδιο, πριν αυτή βγει εκτός ελέγχου.

Σκοπός και στόχοι

Σκοπός αυτής της εργασίας είναι η κατασκευή μιας πλατφόρμας έγκαιρης προειδοποίησης δασικών πυρκαγιών, βασισμένης σε τεχνολογία IoT. Η κατασκευή αυτή θα πρέπει να πετύχει τους εξής τρεις στόχους:

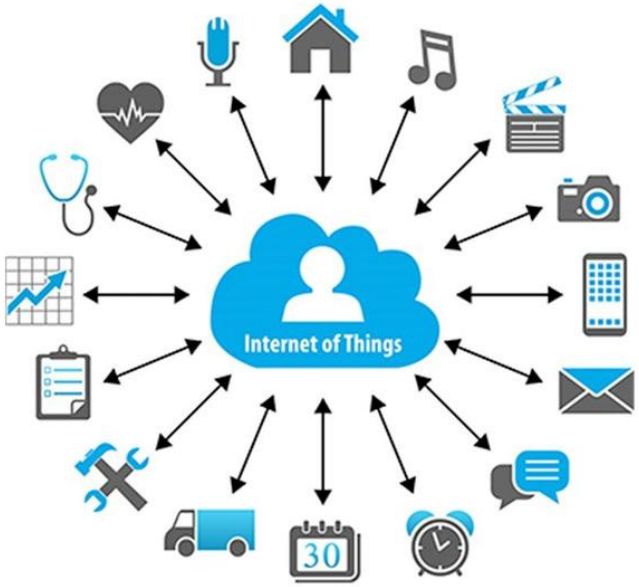
1. Χαμηλό συνολικό κόστος.
2. Θα πρέπει να είναι αξιόπιστη στα δεδομένα που θα μας αποδίδει.
3. Αυτόνομη, δηλαδή θα πρέπει να μην χρειάζεται τον συνεχή έλεγχο από τον χρήστη.
4. Με μεγάλη διάρκεια ζωής.
5. Και να έχει ένα φιλικό περιβάλλον ανάγνωσης των δεδομένων και προειδοποιήσεις σε περίπτωση πυρκαγιάς.

ΚΕΦΑΛΑΙΟ 1ο: Περιγραφή του συστήματος

1.1 Αρχιτεκτονική του συστήματος

Η αρχιτεκτονική του συστήματος που κατασκευάστηκε βασίζεται στην τεχνολογία LoRa. Η τεχνολογία LoRa είναι ένα ασύρματο πρωτόκολλο επικοινωνίας μηχανής με μηχανή «M2M». Έχει σχεδιαστεί ειδικά για επικοινωνίες μεγάλης εμβέλειας και χαμηλής κατανάλωσης ενέργειας, όπως και για IoT συστήματα.

1.2 Τι είναι το Διαδίκτυο των Πραγμάτων (IoT)



Εικόνα 1*: Διαδίκτυο των πραγμάτων (Internet of Thinks – IoT)*

Διαδίκτυο των Πραγμάτων (Internet of Thinks – IoT) είναι ένα δίκτυο από φυσικές οντότητες ή πράγματα τα οποία έχουν ενσωματωμένα ηλεκτρονικά μέρη, λογισμικό, αισθητήρες και δυνατότητα σύνδεσης στο internet, όπως και την ικανότητα να συλλέγουν και να μεταδίδουν δεδομένα. Αυτά μπορούν να γίνουν αντιληπτά από άλλα πράγματα και να «επικοινωνήσουν» μέσω κάποιων υποδομών ή προτύπων επικοινωνίας. Τα πράγματα - οντότητες είναι αισθητήρες (π.χ. υγρασίας, θερμοκρασίας, πίεσης, φωτός), ιατρικά εμφυτεύματα, biochips σε ζώα, αυτοκίνητα με ενσωματωμένους αισθητήρες, κάμερες, κλιματιστικά, φώτα, συστήματα ασφαλείας, smartwatches, συσκευές παρακολούθησής δεδομένων RFID κτλ.

Η φιλοσοφία του Διαδικτύου των Πραγμάτων (IoT) είναι η σύνδεση όλων των αντικειμένων ή ηλεκτρονικών συσκευών μεταξύ τους και με τη βοήθεια του διαδικτύου να μπορεί ένας χρήστης να τα διαχειριστεί από οπουδήποτε και οποτεδήποτε, χωρίς να χρειάζεται η φυσική του παρουσία.

Σε αυτή την κατασκευή, το Διαδίκτυο των Πραγμάτων μας επιτρέπει να επεξεργαστούμε δεδομένα, τα οποία συλλέγονται από ένα κόμβο με αισθητήρες με τη βοήθεια ενός δέκτη. Ο κόμβος θα βρίσκεται πολλά χιλιόμετρα μακριά σε ένα δάσος και δεν θα χρειάζεται συντήρηση ή αλλαγή μπαταρίας για μεγάλο χρονικό διάστημα.

1.3 Τι είναι το LoRa;

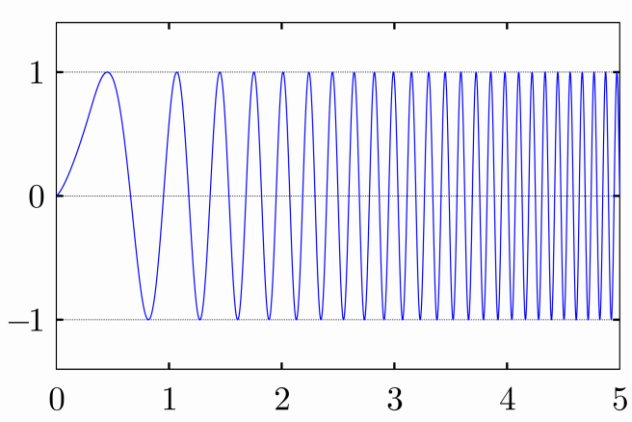
LoRa είναι το φυσικό επίπεδο στο οποίο γίνεται η διαμόρφωση ενός ραδιοσήματος με σκοπό να μεταφέρουμε πάνω του δεδομένα σε μεγάλες αποστάσεις με χαμηλή κατανάλωση και χαμηλές απώλειες.

1.3.1 Τεχνολογία LoRa

Η τεχνολογία LoRa, της οποίας το όνομα αποτελείται από τα αρχικά της συντομογραφίας “Long Range” αναφέρεται στην ικανότητα της τεχνολογίας αυτής για επικοινωνία σε μεγάλες αποστάσεις. Αποτελεί μια τεχνική διαμόρφωσης φάσματος που προέρχεται από την τεχνολογία του φάσματος διασποράς chirp (CSS*) (Εικόνα 2)*, κατά την οποία η κωδικοποίηση των πληροφοριών σε ένα ραδιοσήμα γίνεται με πολλές συνεχόμενες κορυφές που θυμίζουν το τιτίβισμα των πουλιών [7]. Αυτός ο τρόπος διαμόρφωσης καθιστά το σήμα πολύ ανθεκτικό σε παρεμβολές και μπορεί να ληφθεί από μεγάλες αποστάσεις.

Η τεχνολογία LoRa, την οποία ανέπτυξε η Semtech, είναι μια ασύρματη πλατφόρμα μεγάλης εμβέλειας και χαμηλής ισχύος, η οποία αποτελεί ένα από τα πιο σημαντικά πρωτόκολλα επικοινωνίας για την τεχνολογία του Διαδικτύου των Πραγμάτων (IoT).

Οι συσκευές και τα δίκτυα LoRa είναι χρήσιμα για IoT τεχνολογίες που επιλύουν μερικές από τις μεγαλύτερες προκλήσεις που αντιμετωπίζει η ανθρωπότητα π.χ. διαχείριση ενέργειας, μείωση χρήσης φυσικών πόρων και εξοικονόμηση τους, έλεγχος της ρύπανσης, πρόληψη και αντιμετώπιση καταστροφών. Οι συσκευές LoRa της Semtech έχουν συγκεντρώσει αρκετές εκατοντάδες γνωστές περιπτώσεις χρήσεων για έξυπνες πόλεις, σπίτια και κτίρια, κοινότητες, μετρήσεις, εφοδιαστική αλυσίδα και logistics, γεωργία και πολλά άλλα. Με εκατοντάδες εκατομμύρια συσκευές συνδεδεμένες σε δίκτυα σε περισσότερες από 100 χώρες [8].



Εικόνα 2: ***Σήμα διαμορφωμένο με την τεχνική chirp spread spectrum (CSS)***

1.3.2 Κριτήριο επιλογής της τεχνολογίας LoRa

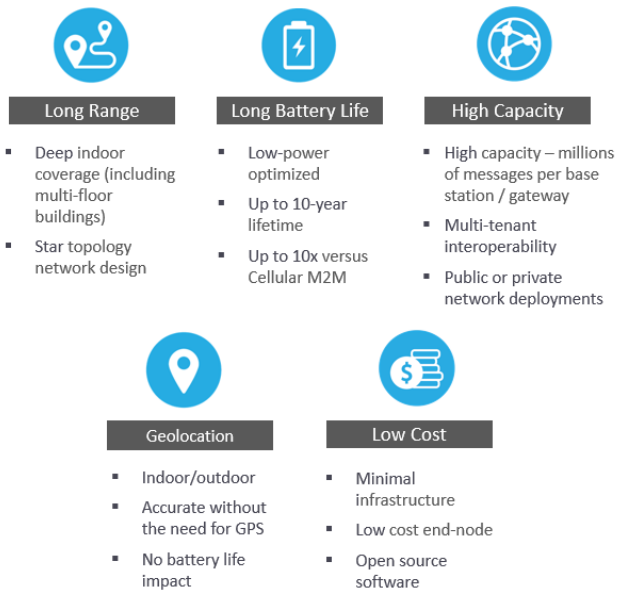
Οι συσκευές LoRa προσφέρουν μεγάλες δυνατότητες για εφαρμογές που χρησιμοποιούν IoT, όπως μεγάλη εμβέλεια επικοινωνίας. Ένα βασικό χαρακτηριστικό ενός συστήματος LoRa είναι οι εξαιρετικά χαμηλές απαιτήσεις ισχύος, οι οποίες επιτρέπουν τη δημιουργία συσκευών που λειτουργούν με μπαταρία ή και φωτοβολταϊκό πάνελ που μπορούν να διαρκέσουν έως και 10 χρόνια αφού έχουν χαμηλή κατανάλωση ενέργειας.

Ακόμη, έχουμε ασφαλή μετάδοση δεδομένων, τα οποία δεν κινδυνεύουν να αλλοιωθούν από άλλες συχνότητες, αφού δεν βρίσκονται στο εύρος της ευρείας τηλεφωνικής επικοινωνίας. Το LoRa χρησιμοποιεί ελεύθερες ραδιοσυχνότητες που δεν απαιτούν άδεια, οι οποίες χωρίζονται σε γεωγραφικές ζώνες[9] όπως βλέπουμε παρακάτω:

Region	Band / channels	Channel Plan
Europe	433.05-434.79 MHz	EU433
	863–870/873 MHz	EU863-870
North America	902–928 MHz	US902-928
India	865–867 MHz	IN865-867
Southeast Asia	433.05-434.79 MHz	EU433
	915–928 MHz	AU915-928/AS923-1
Australia	915–928 MHz	AU915-928/AS923-1

Πίνακας 1: ***Ζώνες συχνοτήτων LoRa***

Η τεχνολογία χρησιμοποιείται από δημόσια δίκτυα και παρέχει τη μεγαλύτερη εμβέλεια που φτάνει μέχρι και 15 km και έχει ρυθμούς μετάδοσης δεδομένων μεταξύ 0,3 kbit/s και 300 kbit/s, ανάλογα με τον παράγοντα διασποράς.

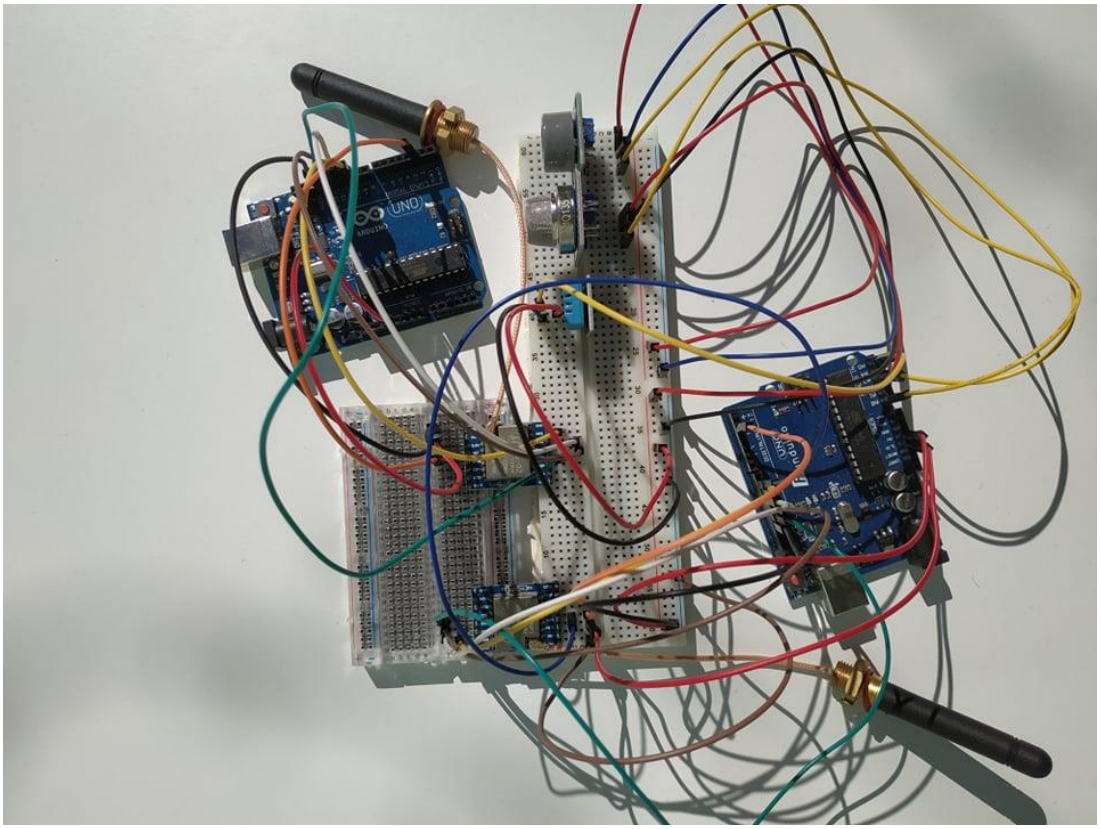


Εικόνα 3*: Πλεονεκτήματα χρήσης της τεχνολογίας LoRa*

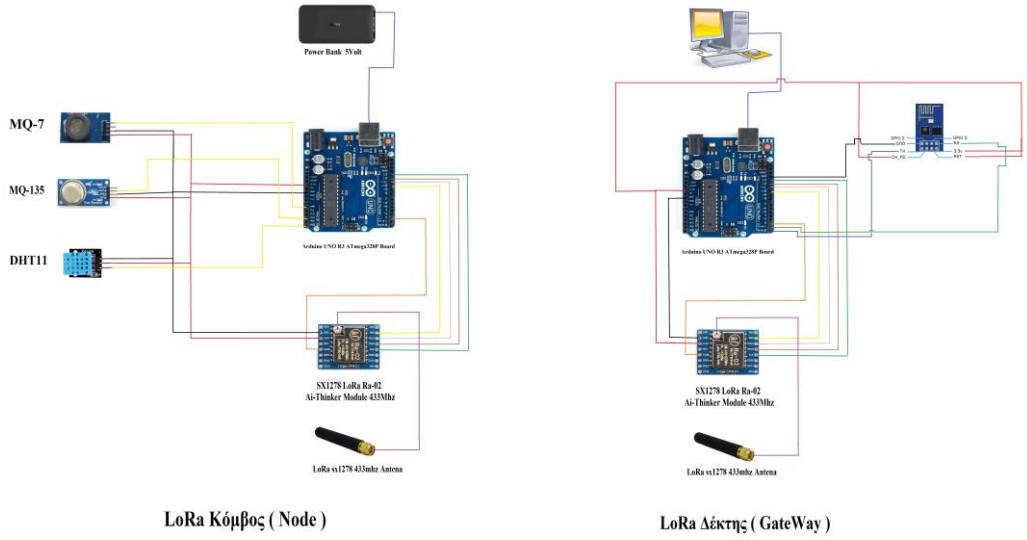
ΚΕΦΑΛΑΙΟ 2ο: Η κατασκευή – Περιγραφή και ανάλυση

Σε αυτό το κεφάλαιο θα αναλυθεί το φυσικό κομμάτι της κατασκευής, από τι υλικά αποτελείται, πώς συνδέονται το ένα με το άλλο και ο ρόλος τους στην ανίχνευση μιας πυρκαγιάς.

Το υλικό κομμάτι της κατασκευής αυτής της διπλωματικής αποτελείται από δύο οντότητες - κόμβους. Ο **κόμβος (Transmitter)** αυτής της κατασκευής βρίσκεται σε μια περιοχή ή σε ένα δάσος σε μεγάλη απόσταση από τον δέκτη και έχει τη δυνατότητα να συλλέγει δεδομένα και με την τεχνολογία LoRa να τα στέλνει χωρίς παρεμβολές σε ένα μακρινό **κόμβο - δέκτη (Receiver**). Με τη βοήθεια αυτού μπορούμε να επεξεργαστούμε τις πληροφορίες για τη θερμοκρασία, την υγρασία του δάσους, καθώς και τα ποσοστά μονοξειδίου και διοξειδίου του άνθρακα στο περιβάλλον. Από αυτά μπορούμε να βγάλουμε ένα συμπέρασμα με τη βοήθεια ενός αλγορίθμου, ο οποίος θα μας αποδίδει ως αποτέλεσμα το αν έχει προκύψει πυρκαγιά κοντά στην περιοχή που καλύπτει ο κόμβος. Στο *Κεφάλαιο 3* θα γίνει ανάλυση του software μέρους που αφορά την επικοινωνία του LoRa, τη μετάδοση των δεδομένων από τον δέκτη και την απεικόνιση των δεδομένων.



Εικόνα 4: Πρώτη κατασκευή



Εικόνα 5*: Απεικόνιση της πρώτης κατασκευής*

2.1 Κόμβος – LoRa Transmitter

Ο κόμβος αυτής της κατασκευής αποτελεί το κομμάτι που έρχεται σε επαφή με το περιβάλλον και συλλέγει κάποια δεδομένα από αυτό. Τα δεδομένα που συλλέχθηκαν, έπειτα από επεξεργασία, μπορούν να μας προσδιορίσουν τα χαρακτηριστικά για την ύπαρξη πυρκαγιάς. Η πυρκαγιά είναι ουσιαστικά μια ανεξέλεγκτη φωτιά, άρα οι ενδείξεις που μας υποδηλώνουν την ύπαρξη μιας φωτιάς είναι οι εξής: Η θερμοκρασία είναι υψηλή και ξεπερνά εκείνη του περιβάλλοντος σε πολύ μεγάλους βαθμούς. Ο καπνός που παράγεται από την καύση της ύλης αποτελείται από μονοξείδιο και διοξείδιο του άνθρακα. Υπάρχει έλλειψη νερού σε μορφή υγρασίας στο περιβάλλον, αφού οι πολύ υψηλές θερμοκρασίες θα έχουν εξατμίσει το νερό που βρίσκεται στον αέρα και φως που παράγουν οι φλόγες. Στην κατασκευή αυτή ανιχνεύεται: η θερμοκρασία, η υγρασία, το μονοξείδιο και διοξείδιο του άνθρακα στο περιβάλλον γύρω από τον κόμβο. Ο κόμβος (LoRa Transmitter) έχει κατασκευαστεί από τα εξής πράγματα:

a) Αισθητήρας Θερμοκρασίας και Υγρασίας DHT11

b) Αισθητήρας Μονοξείδιου του άνθρακα (CO) MQ7

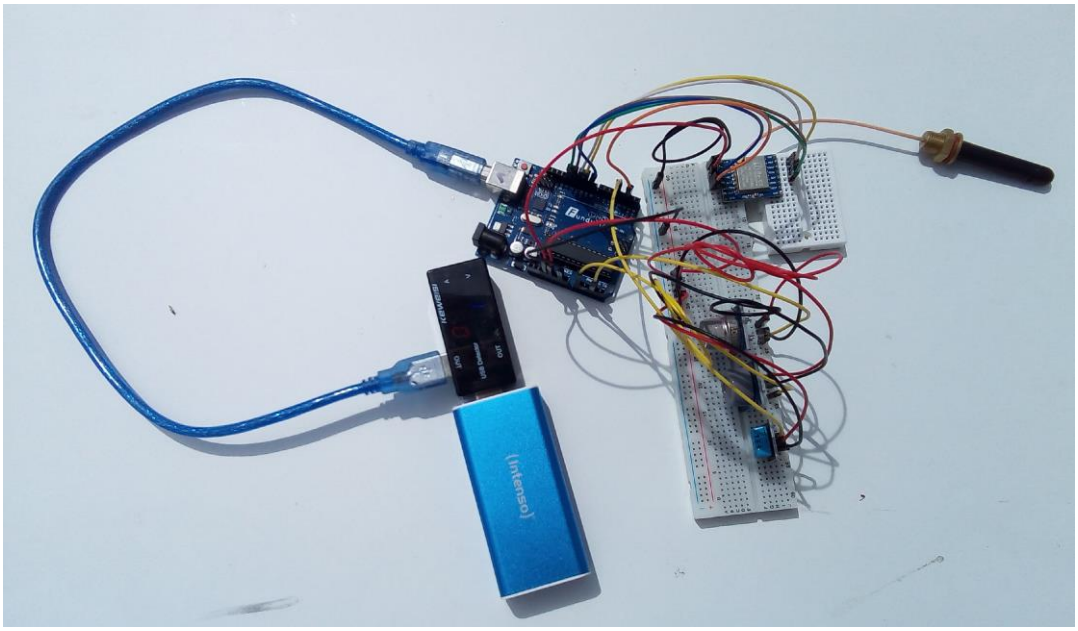
c) Αισθητήρας Διοξειδίου του άνθρακα (CO2) MQ135

d) Arduino Uno

e) LoRa module

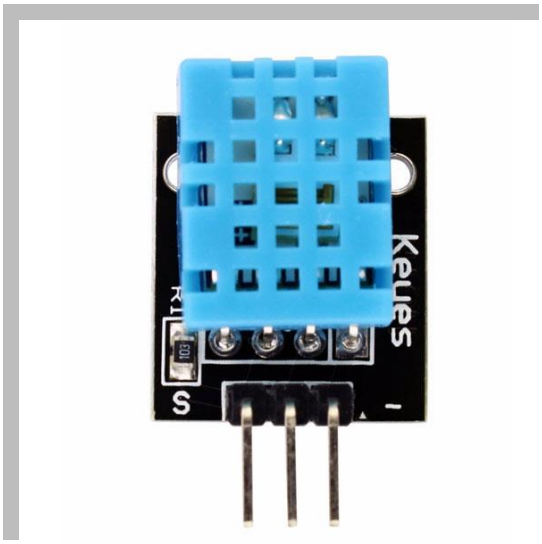
f) Κεραία συντονισμένη στα 433Mhz

g) Μπαταρία – Τροφοδοσία



Εικόνα 6*: Ο κόμβος LoRa Transmitter*

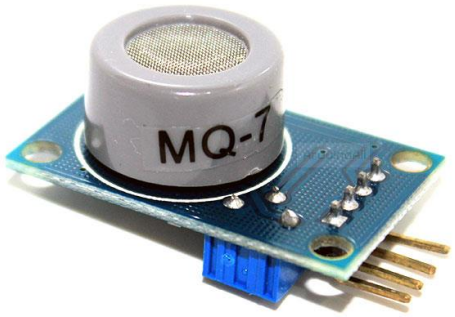
a) Αισθητήρας Θερμοκρασίας και Υγρασίας DHT11



Εικόνα 7*: DHT11 Digital Temperature and Humidity Sensor*

Ο αισθητήρας DHT11 είναι αισθητήρας θερμοκρασίας και υγρασίας. Είναι ένας οικονομικός αισθητήρας ευρείας χρήσης, με εύρος τιμών θερμοκρασίας από 0 έως 50 °C με σφάλμα μέτρησης ±2°C, και εύρος τιμών υγρασίας 20-90%RH (Relative Humidity) με σφάλμα μέτρησης ±5%RH. Η κατανάλωση του αισθητήρα είναι πολύ μικρή στην ενεργή λειτουργία του, στα 0,3mA και 60μA σε κατάσταση ετοιμότητας.

b) Αισθητήρας Μονοξείδιου του άνθρακα (CO) MQ7



Εικόνα 8*: MQ-7 Carbon Monoxide CO Gas Sensor For Arduino (MQ7)*

Ο αισθητήρας MQ-7 είναι αισθητήρας μονοξειδίου του άνθρακα. Πρόκειται για αισθητήρα με μεγάλο εύρος ανίχνευσης, γρήγορη απόκριση και υψηλή ευαισθησία, όπως και σταθερή και μεγάλη διάρκεια ζωής. Το εύρος τιμών ανίχνευσης του μονοξείδιου του άνθρακα είναι από 10 έως 2000ppm (part per million), η τιμή του οποίου εξαρτάται από τη σχετική θερμοκρασία που κυμαίνεται στους ±20 °C και τη σχετική υγρασία που κυμαίνεται 65%±5%RH. Η κατανάλωση του αισθητήρα εξαρτάται από την ισχύ που καταναλώνεται κατά τη θέρμανση μιας ειδικής αντίστασης μέσα στον αισθητήρα, η οποία δίνει την τιμή του μονοξείδιου του άνθρακα. Αυτή είναι κοντά στα 350mW.

c) Αισθητήρας Διοξειδίου του άνθρακα (CO2) MQ135

Εικόνα 9*: MQ-135 Air Quality Sensor – Hazardous Gas Detection Module For Arduino (MQ135)*

Ο αισθητήρας MQ-135 εκτός των άλλων ανιχνεύει και διοξείδιο του άνθρακα, το οποίο θα χρησιμοποιηθεί στην παρούσα εργασία. Ένας αισθητήρας με μεγάλο εύρος ανίχνευσης, γρήγορη απόκριση, υψηλή ευαισθησία, σταθερή και μεγάλη διάρκεια ζωής. Το εύρος τιμών ανίχνευσης του διοξειδίου του άνθρακα είναι από 10 έως 1000ppm, η τιμή του οποίου εξαρτάται από την σχετική θερμοκρασία που κυμαίνεται στους ±20 °C και την σχετική υγρασία που κυμαίνεται 65%±5%RH. Η κατανάλωση του αισθητήρα εξαρτάται από την ισχύ που καταναλώνετε κατά την θέρμανση μιας ειδικής αντίστασης μέσα στον αισθητήρα, η οποία δίνει την τιμή του διοξειδίου του άνθρακα. Αυτή κυμαίνεται κοντά στα 800mW.

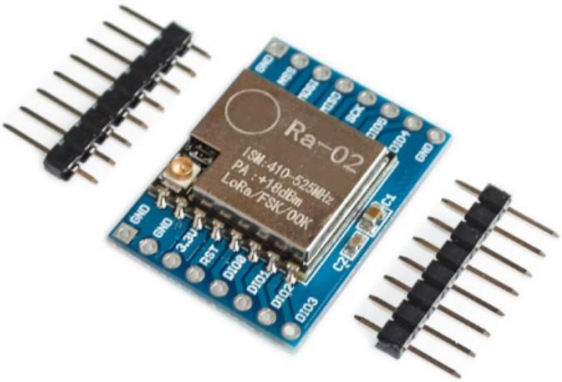
d) Arduino Uno

Εικόνα 10*: Arduino UNO R3 Atmega328P Board with USB Cable (DIP Version)(OEM)*

Το Arduino UNO R3 είναι μια διάταξη που είναι εύκολη για χρήση με πλατφόρμα ανοικτού κώδικα και με ενσωματωμένο μικροελεγκτή ATmega328P με ταχύτητα ρολογιού στα 20Mhz. Η πλακέτα είναι βασισμένη στη γλώσσα προγραμματισμού C και μπορεί να προγραμματιστεί με το πρόγραμμα Arduino IDE για υπολογιστή με λειτουργικό Windows. Λειτουργεί στα 5Volt με προτεινόμενη τάση λειτουργείας τα 7-12Volt σε θερμοκρασίες που κυμαίνονται από

-40°C έως 85°C και καταναλώνει 40mA για κάθε I/O pin και 50mA για τα 3.3Volt pin.

e) LoRa module



Εικόνα 11*: SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz*

Το Ra-02 είναι μια μονάδα ασύρματης μετάδοσης που βασίζεται στον ασύρματο πομποδέκτη SX1278 της SEMTECH, βασίζεται στην τεχνολογία LoRa, και έχει απόσταση επικοινωνίας 10 χιλιόμετρα. Λειτουργεί στη συχνότητα των 433MHz με ευαισθησία κάτω στα -148dBm και προγραμματιζόμενο ρυθμό μετάδοσης μέχρι τα 300kbit. Η μονάδα SX1278 έχει τάση λειτουργίας 3,3Volt και αντέχει σε θερμοκρασίες από -40°C έως 80°C.

f) Κεραία συντονισμένη στα 433Mhz

Εικόνα 12*: Antenna RF 434MHz 2dBi 63.6mm u.FL*

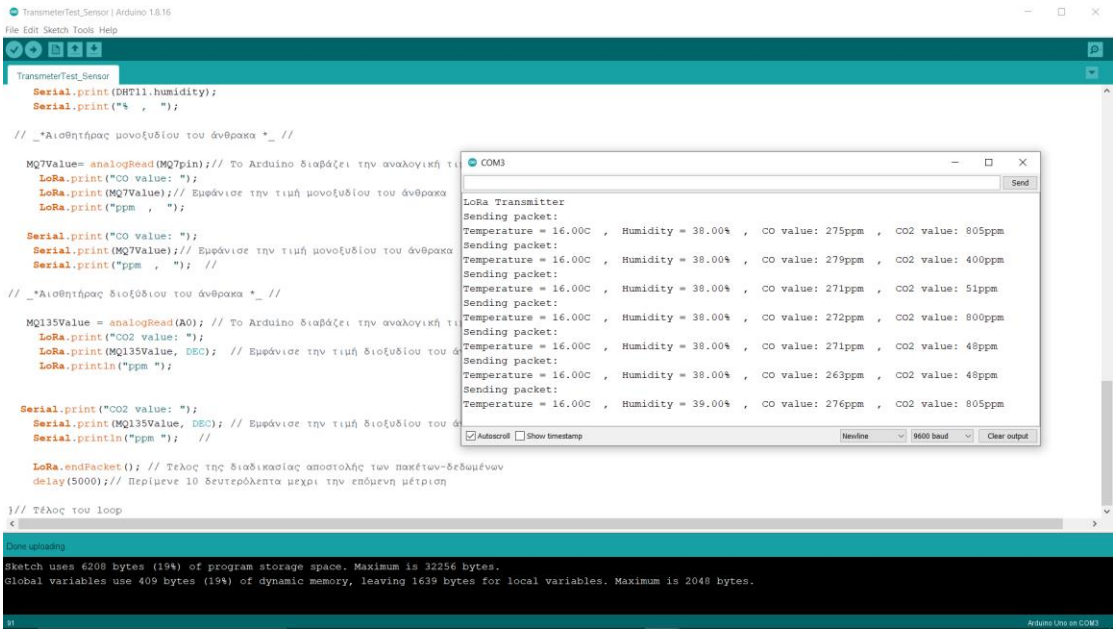
Η επιλογή της κεραίας παίζει πολύ σημαντικό ρόλο στη μετάδοση δεδομένων όταν χρησιμοποιούμε την τεχνολογία LoRa. Όσο μεγαλύτερη είναι η απολαβή της κεραίας που χρησιμοποιούμε τόσο μεγαλύτερη είναι η εμβέλεια της μονάδας LoRa. Ένας άλλος παράγοντας που βοηθάει είναι ο συντονισμός της κεραίας. Μια καλά συντονισμένη κεραία βοηθά στην αύξηση της εμβέλειας της μονάδας LoRa (LoRa module). Στη συγκεκριμένη κατασκευή, η κεραία είναι συντονισμένη στην περιοχή συχνοτήτων που έχει προκαθοριστεί για την τεχνολογία LoRa από την LoRa alliance για την περιοχή της Ευρώπης στα 433.05-434.79 MHz.

g) Μπαταρία – Τροφοδοσία

Εικόνα 13*: Intenso PowerBank 5200 mAh*

Στην παρούσα κατασκευή θα χρησιμοποιηθεί για την τροφοδοσία της μια μπαταρία – PowerBank, που θα τροφοδοτεί το Arduino Uno με 5V και θα παρέχει ενέργεια 5200mAh στον Lora Transmitter. Η κατασκευή έχει τη δυνατότητα να αναβαθμιστεί με διαφορετική μπαταρία και έναν μικρό ηλιακό συλλέκτη για να επιτευχθεί ακόμη μεγαλύτερη αυτονομία.

2.1.2 Ανάπτυξη προγράμματος για την εύρυθμη λειτουργεία του κόμβου - LoRa Transmitter



Εικόνα 14*: Πρόγραμμα για τον έλεγχο της διάταξης LoRa Transmitter*

Στην *Εικόνα 14* βλέπουμε την επιτυχή λειτουργία του κόμβου κατά την οποία αυτός συλλέγει δεδομένα από το περιβάλλον γύρω του και τα εμφανίζει στην οθόνη του υπολογιστή στο περιβάλλον του προγράμματος Arduino IDE.

2.2 Κόμβος - Δέκτης (LoRa Receiver)

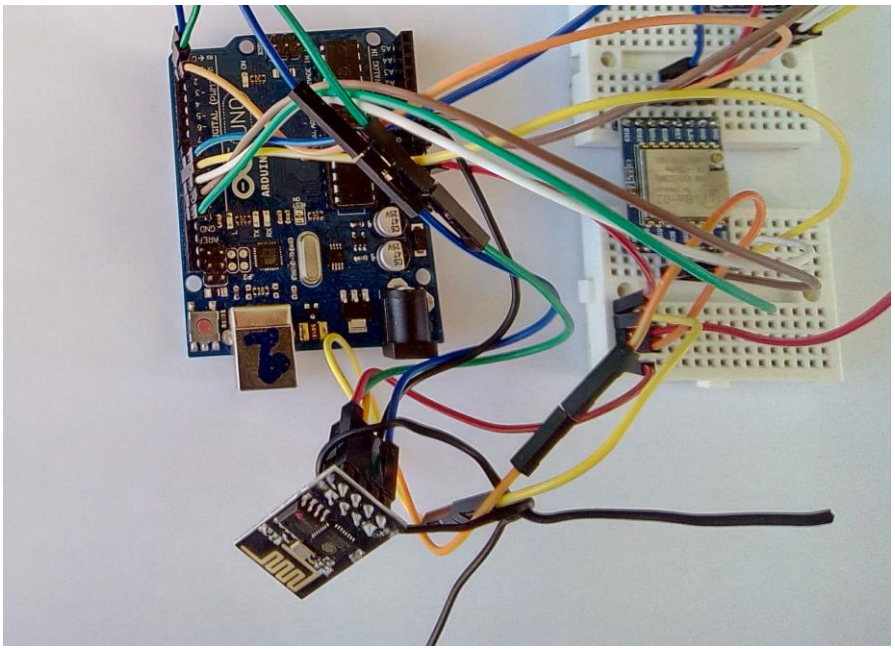
Ο κόμβος δέκτη (LoRa Receiver) βρίσκεται στο άλλο άκρο του συστήματος και είναι το κομμάτι του συστήματος που θα συλλέξει τα δεδομένα που έχει στείλει ο κόμβος (LoRa Transmitter) και θα τα μεταφέρει σε μια βάση δεδομένων για αποθήκευση και επεξεργασία τους σε ένα άλλο χρόνο. Το κομμάτι αυτό αποτελείται από τα εξής πράγματα:

a) **Arduino Uno**

b) **LoRa Module - Receiver**

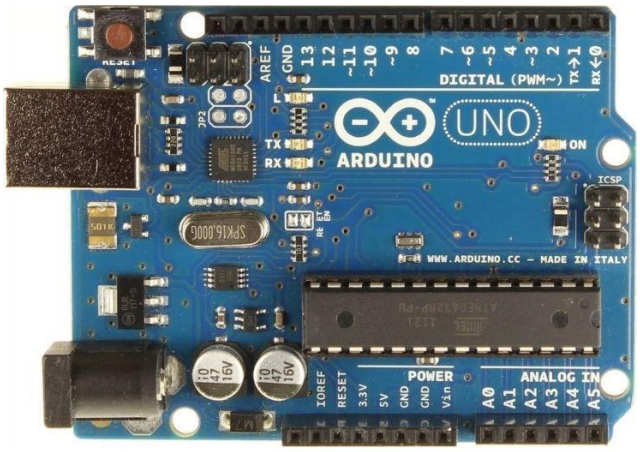
c) **Κεραία συντονισμένη στα 433Mhz**

d) **WiFi Module**



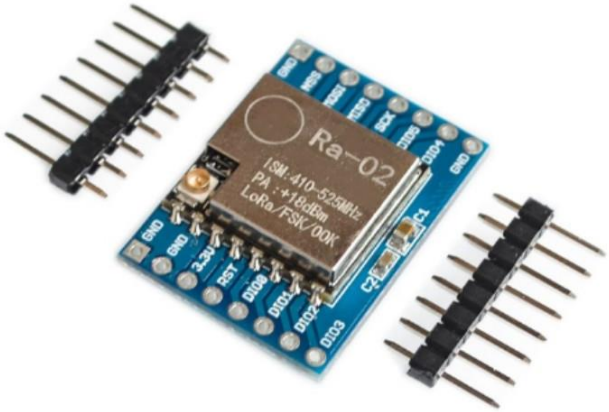
Εικόνα 15*: Κόμβος - δέκτη (LoRa Receiver)*

a) Arduino Uno



Εικόνα 16*: Arduino UNO R3 Atmega328P Board with USB Cable (DIP Version)(OEM)*

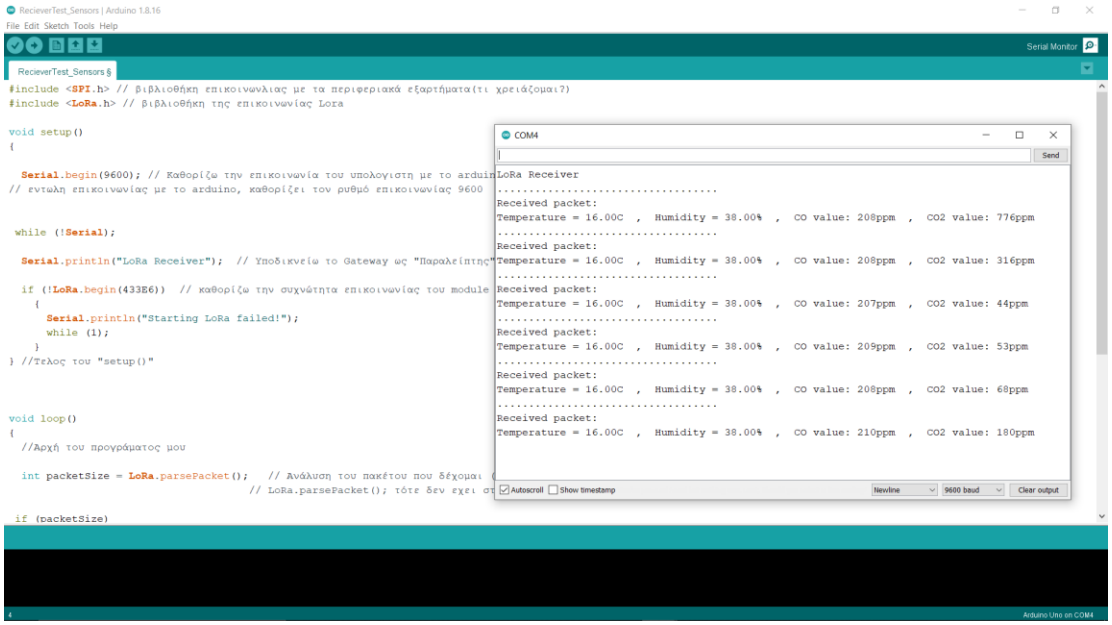
b) LoRa Module



Εικόνα 17*: SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz*

c) Κεραία συντονισμένη στα 433Mhz

Εικόνα 18*: Antenna RF 434MHz 2dBi 63.6mm u.FL*

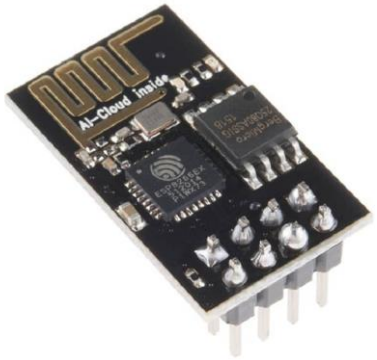


Εικόνα 19*: Αποτέλεσμα του κώδικα του δέκτη, ορθή επικοινωνία μεταξύ LoRa Transmitter και LoRa Receiver*

Μετά από την επιτυχή λειτουργεία της επικοινωνίας LoRa μεταξύ του κόμβου (LoRa Transmitter) και του κόμβου - δέκτη (LoRa Receiver), όπως φαίνεται στην *Εικόνα 19*, προστέθηκε ένα Wifi module για να σταλθούν τα δεδομένα που δεχόμαστε στο LoRa Receiver, σε μια βάση δεδομένων από την οποία μπορούμε να τα επεξεργαστούμε. Το Wifi module που επιλέχθηκε είναι το ESP8266 ESP-01. Η μονάδα αυτή μπορεί να δώσει σε έναν μικροελεγκτή πρόσβαση στο ιντερνέτ, αφού έχει ενσωματωμένο το πρωτόκολλο επικοινωνίας TCP/IP. Έχει χαμηλή τιμή, ασύρματη μετάδοση δεδομένων, είναι συμβατό με το Arduino Uno, μπορεί να λειτουργήσει ως πομπός ή δέκτης ή πομποδέκτης και προγραμματίζεται και με AT commands.

Η μονάδα ESP8266 ESP-01 λειτουργεί στα 3.3Volt και καταναλώνει σε κατάσταση ετοιμότητας 1.0mW. Στην ενεργή λειτουργία του καταναλώνει ανάλογα με τον ρυθμό μετάδοσης που θα ορίσει ο χρήστης μέχρι και 215mA. Αντέχει σε θερμοκρασίες που κυμαίνονται από -40°C έως 125°C.

d) WiFi Module



Εικόνα 20*: ESP 8266 ESP-01 WiFi Module*

2.3 Λίστα των εξαρτημάτων και το κόστος τους

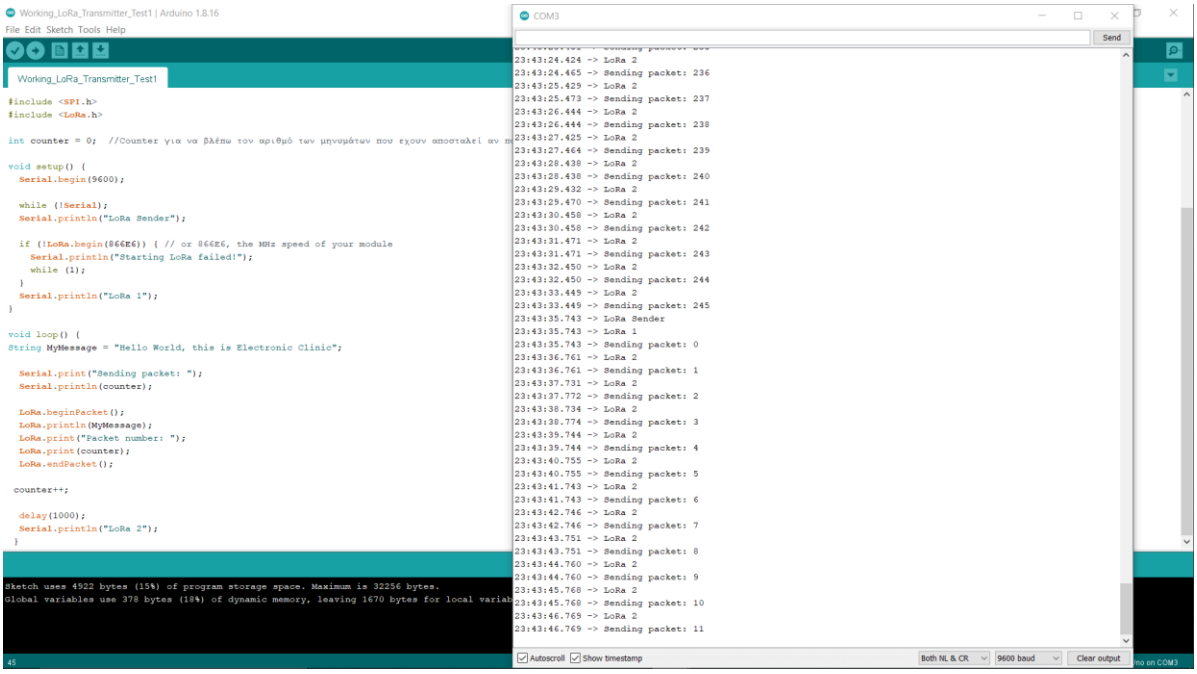
Υλικά	Κόστος
DHT11 Digital Temperature and Humidity Sensor	2,23 €
MQ-7 Carbon Monoxide CO Gas Sensor For Arduino (MQ7)	3,79 €
MQ-135 Air Quality Sensor – Hazardous Gas Detection Module For Arduino (MQ135)	2,70 €
Arduino UNO R3 Atmega328P Board with USB Cable (DIP Version)(OEM) (x2)	14,52€
SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz (x2)	21,20 €
Antenna RF 434MHz 2dBi 63.6mm u.FL (x2)	5,80 €
Μπαταρία, *Intenso PowerBank 5200 mAh*	9 €
ESP 8266 ESP-01	4,20 €
Συνολικό κόστος	63,44 €

Πίνακας 2*: Τιμοκατάλογος της κατασκευής*

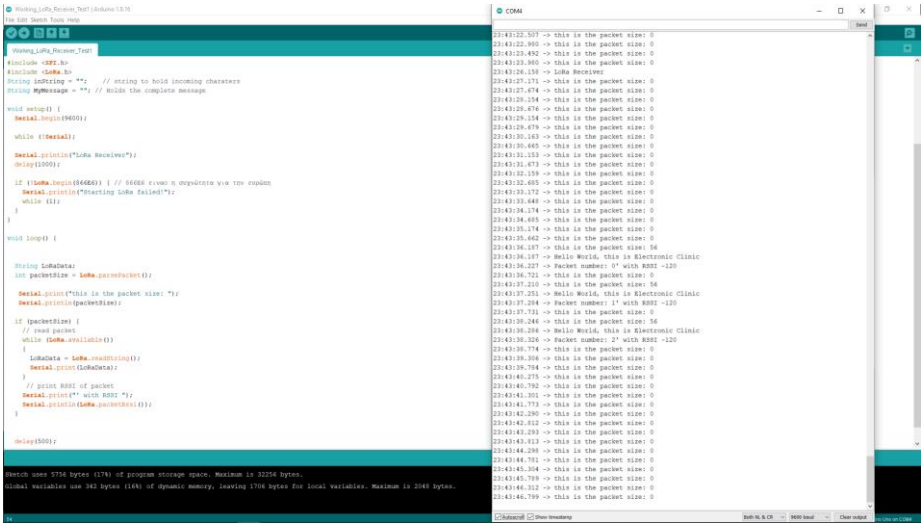
2.4 Παρατήρηση

Σε αυτό το σημείο χρησιμοποιήθηκε ένας δοκιμαστικός κώδικας για τον δέκτη (LoRa receiver), όπου τα δεδομένα που θα δέχεται από τον πομπό θα πρέπει να τα στείλει με τη χρήση AT commands στον server που επιλέχθηκε με σκοπό τη διαχείριση των δεδομένων. Το πρόβλημα όμως που συναντήθηκε ήταν ότι έπρεπε να αναλυθεί το πακέτο που έρχεται από τον κόμβο και έπειτα να σταλθεί με το WiFi module στον server με τη χρήση AT commands. Παρουσιάστηκαν πολλά προβλήματα στην προσπάθεια να συνδυαστεί ο κώδικας για την επικοινωνία LoRa με τις εντολές AT και να αντιμετωπιστούν ασύμβατες και λανθασμένες βιβλιοθήκες που δεν λειτουργούσαν σωστά ή, σε πολλές περιπτώσεις, δεν ήταν συμβατές. Για τον λόγο αυτό, λήφθηκε η απόφαση να μην χρησιμοποιηθεί το ESP8266 ESP-01 WiFi module και να βρεθεί άλλη λύση, που θα επιτρέψει την κατασκευή ενός LoRa Receiver, ο οποίος θα μπορεί να συνεργάζεται πιο εύκολα με το WiFi module.

Στις παρακάτω φωτογραφίες αναφέρονται κάποια από τα προβλήματα που συναντήθηκαν κατά τον προγραμματισμό του LoRa Transmitter και LoRa Receiver.

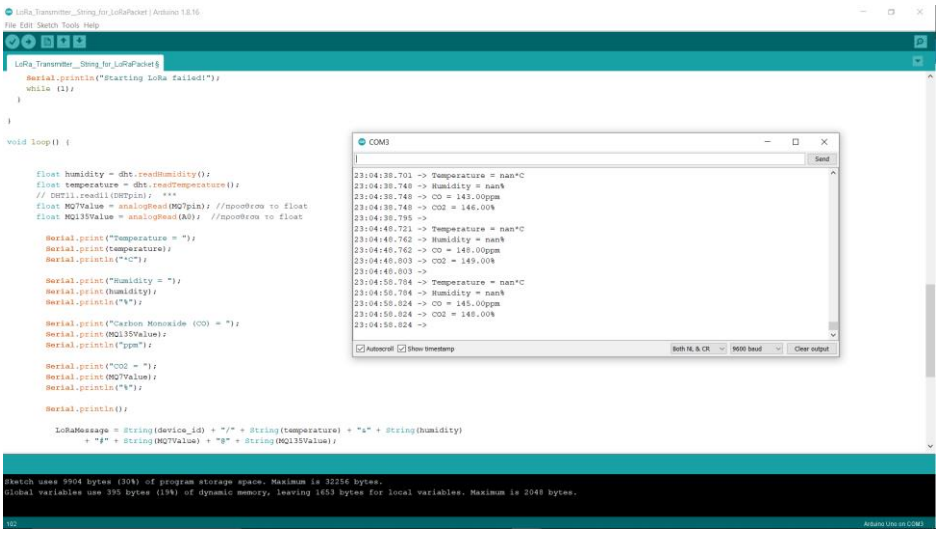


Εικόνα 21*: Ο LoRa Transmitter στέλνει ένα απλό πακέτο στο LoRa Receiver*



Εικόνα 22*: LoRa Receiver δέχεται το απλό πακέτο σε μη τακτά χρονικά διαστήματα*

Στην *Εικόνα 22* παρατηρούμε την έξοδο του serial com4 από όπου φαίνεται ότι κατά την επικοινωνία LoRa των δύο κόμβων υπάρχουν πολλά πακέτα που χάνονται κατά την αποστολή από τον κόμβο LoRa Transmitter στον κόμβο – δέκτη LoRa Receiver. Αυτό γίνεται αντιληπτό από τον αριθμό που έχουμε βάλει στο κάθε πακέτο που στέλνει ο LoRa transmitter (Packet number). Για ένα μεγάλο χρονικό διάστημα ο LoRa receiver δεν δέχεται κανένα πακέτο. Μέχρι που κάποια στιγμή λαμβάνει ένα ή δύο πακέτα και στη συνέχεια χάνει τα επόμενα. Δεν υπάρχει σταθερή και συνεχόμενη λήψη των δεδομένων που συλλέγει ο δέκτης LoRa receiver. Τέλος, στο Παράρτημα ΣΤ αναφέρεται και το πρόβλημα στην επικοινωνία κόμβου – δέκτη LoRa Receiver με την πλατφόρμα ThingSpeak.



Εικόνα 23*: Πρόβλημα στο LoRa transmitter με τις τιμές θερμοκρασίας και υγρασίας*

Εικόνα 24*: Πρόβλημα του LoRa transmitter με τις τιμές θερμοκρασίας και υγρασίας όπως φαίνεται στο ThingSpeak*

2.5 Προσπάθεια 2η: Κατασκευή του δέκτη (LoRa Receiver) με το ESP8266 ESP-12E

Στη δεύτερη προσπάθεια για την κατασκευή του δέκτη (LoRa Receiver) αποφάσισα να χρησιμοποιήσω ένα διαφορετικό εξάρτημα που θα μου προσφέρει τη δυνατότητα σύνδεσης στο internet και θα είναι πιο εύκολο στη χρήση και τον προγραμματισμό του. Για τον λόγο αυτό επιλέχθηκε ένας μικροϋπολογιστής ο οποίος περιέχει WiFi module ενσωματωμένο. Αυτός ο μικροϋπολογιστής είναι ο NodeMCU ESP8266 WiFi module και είναι εύκολος στον προγραμματισμό του, τροφοδοτείται από 3.3v και διαθέτει επεξεργαστή 106 Mhz Tensilica Xtensa LX80, 64 KB SRAM και 96 KB flash μνήμη, 16 ακροδέκτες GPIO, έχει μεγάλες εφαρμογές και δυνατότητες για κατασκευές που βασίζονται στην τεχνολογία IoT, όπως και πολλές δυνατότητες για τη χρήση του σε κόμβους και δίκτυα που χρησιμοποιούν API. Ο δέκτης μπορεί να τροφοδοτηθεί με μια μπαταρία. Στη συγκεκριμένη κατασκευή, ο δέκτης τροφοδοτείται μέσω ενός ηλεκτρονικού υπολογιστή. Το κομμάτι αυτό αποτελείται από τα εξής πράγματα:

a) **ESP8266 ESP-12E**

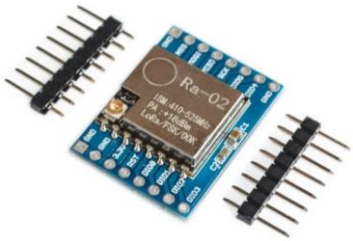
b) **LoRa Module**

c) **Κεραία συντονισμένη στα 433Mhz**

a)ESP8266 ESP-12E

Εικόνα 25*: ESP8266 ESP-12E NodeMcu V3 Lua Board CH340 WIFI IoT*

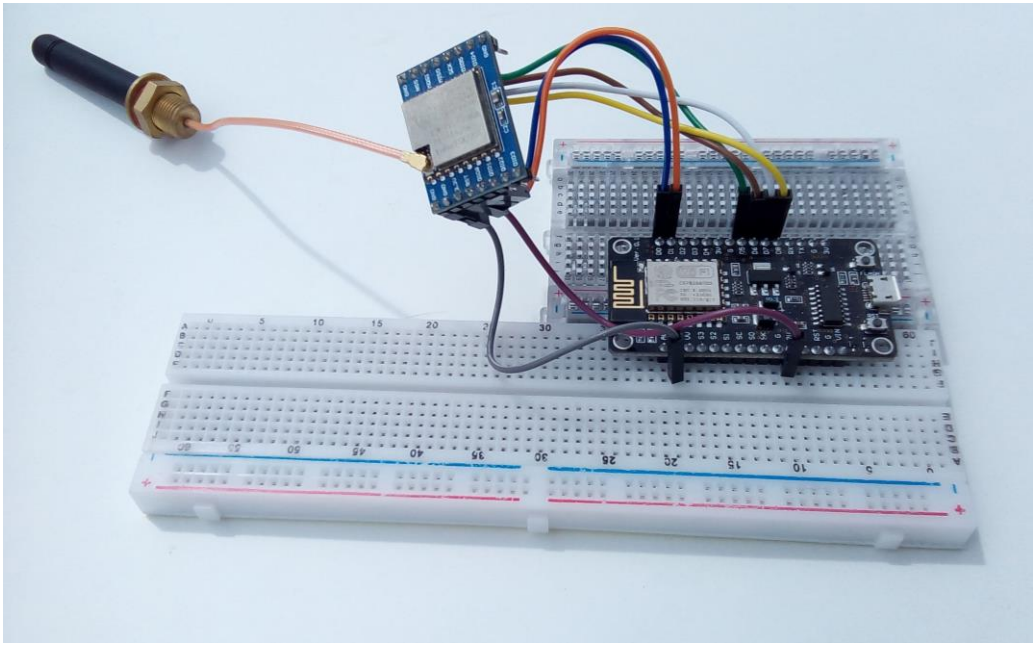
b) LoRa Module



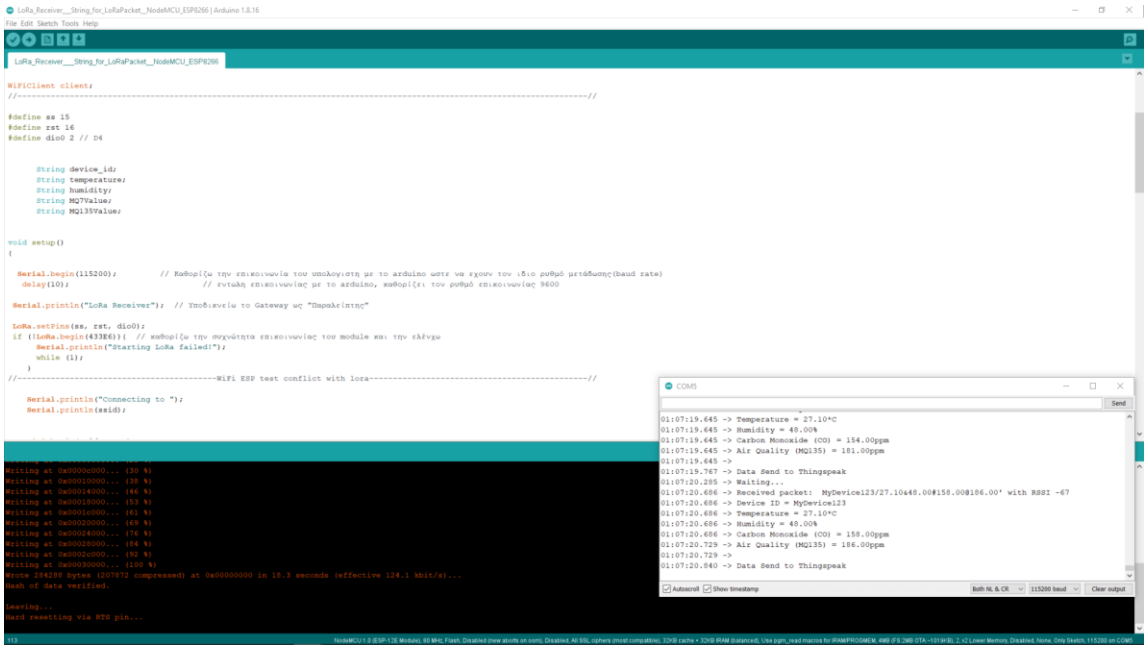
Εικόνα 26*: SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz*

c) Κεραία συντονισμένη στα 433Mhz

Εικόνα 27*: Antenna RF 434MHz 2dBi 63.6mm u.FL*

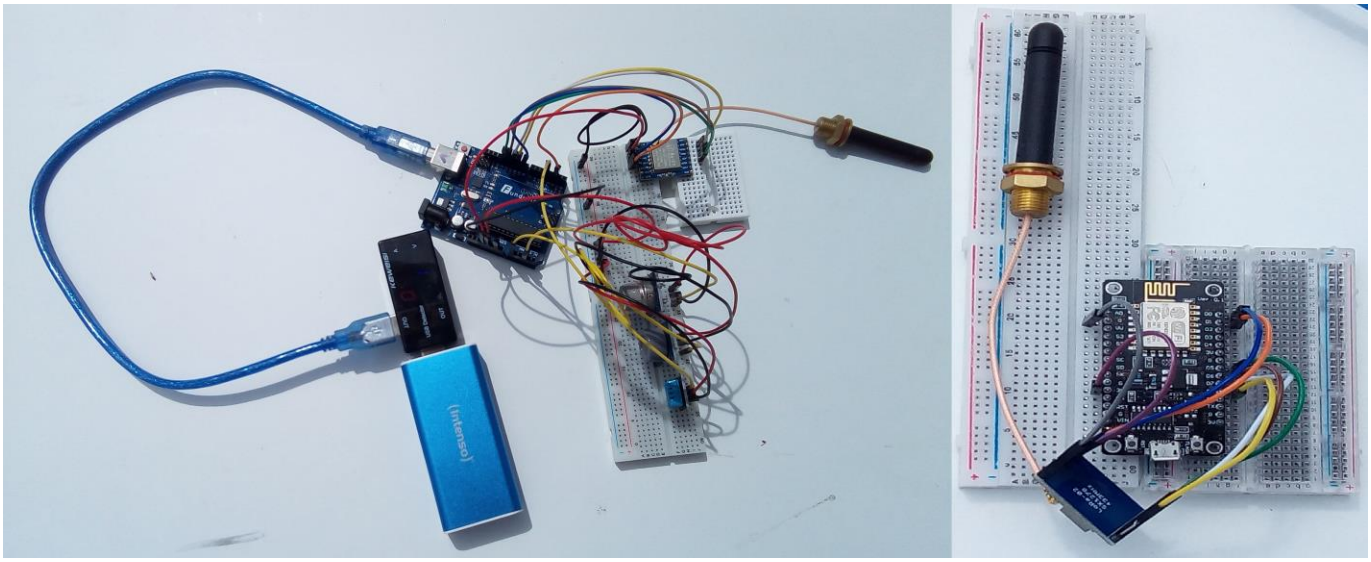


Εικόνα 28*: Χρήση του NodeMCU ESP8266 ως δέκτη (LoRa Receiver)*

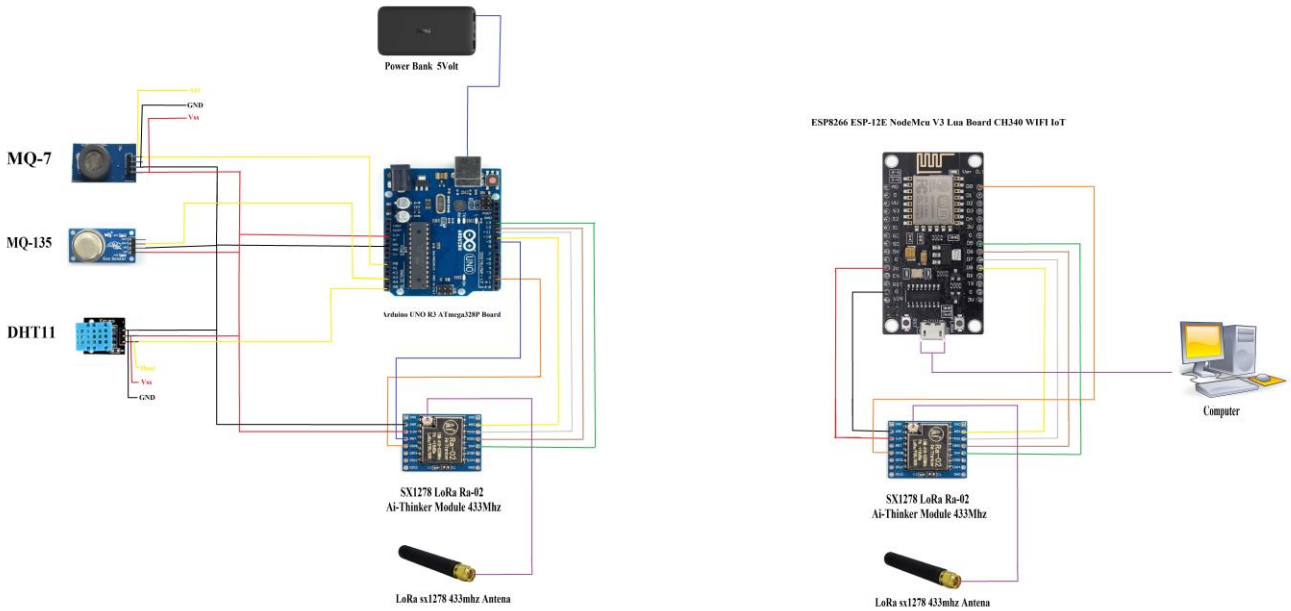


Εικόνα 29*: Επιτυχής επαναλαμβανόμενη λήψη δεδομένων στο LoRa receiver από τον LoRa Transmitter (πρόγραμμα Arduino IDE)*

Εικόνα 30: ***Επιτυχής επαναλαμβανόμενη ανάρτηση των δεδομένων που λαμβάνει ο LoRa receiver στο ThingSpeak***



Εικόνα 31: ***Τελική κατασκευή με χρήση του NodeMCU ESP8266 ως δέκτη (LoRa Receiver)***



Εικόνα 32*: Απεικόνιση της τελικής κατασκευής*

Υλικά	Κόστος
DHT11 Digital Temperature and Humidity Sensor	2.23 €
MQ-7 Carbon Monoxide CO Gas Sensor For Arduino (MQ7)	3.79 €
MQ-135 Air Quality Sensor – Hazardous Gas Detection Module For Arduino (MQ135)	2.70 €
Arduino UNO R3 Atmega328P Board with USB Cable (DIP Version)(OEM)	14.99€
SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz (x2)	21.20 €
Antenna RF 434MHz 2dBi 63.6mm u.FL (x2)	5.80 €
Μπαταρία, *Intenso PowerBank 5200 mAh*	9 €
ESP8266 ESP-12E NodeMcu V3 Lua Board CH340 WIFI IoT	6.80 €
Συνολικό κόστος	66,51€

Πίνακας 3*: Τιμοκατάλογος της τελικής κατασκευής*

Κεφάλαιο 3ο: Ανάλυση της κατασκευής από πλευράς κώδικα

Το NodesMCU ESP8266 WiFi module είναι ένας πολύ ευέλικτος μικροελεγκτής που έχει βρει μεγάλη χρήση σε συστήματα που βασίζονται στο διαδίκτυο των πραγμάτων λόγω της ευκολίας προγραμματισμού και προσβασιμότητάς του για τους χρήστες του. Χρησιμοποιεί τον ίδιο τρόπο προγραμματισμού με το Arduino uno, μέσω του προγράμματος Arduino IDE που παρέχει υποστήριξη μέσα από τις βιβλιοθήκες του, για να έχει πρόσβαση στο WiFi module χωρίς τη χρήση AT commands.

3.1 Σταθμός αποστολής δεδομένων: Κώδικας Κόμβου (LoRa Transmitter)

Παρακάτω βλέπουμε τον κώδικα για τον σταθμό αποστολής των δεδομένων - κόμβο (LoRa Transmitter) από όπου καλούμαστε να συλλέξουμε και να μεταφέρουμε στον χρήστη τα δεδομένα του περιβάλλοντος που παρακολουθούμε, χρησιμοποιώντας την τεχνολογία LoRa μέσα από το περιβάλλον του Arduino IDE.

Εικόνα 33*: Κώδικας κόμβου (LoRa Transmitter) μέρος 1ο*

Στην *Εικόνα 33* βλέπουμε το πρώτο μέρος του κώδικα, στην αρχή του οποίου δηλώνονται οι βιβλιοθήκες που χρειάζονται για την κατασκευή του κόμβου LoRa Transmitter. Έπειτα καθορίζονται οι μεταβλητές και ο τύπος του αισθητήρα θερμοκρασίας και υγρασίας. Αυτές είναι:

1) DHT.h και DHT_U : βιβλιοθήκες για τον αισθητήρα θερμοκρασίας και υγρασίας

2) SPI.h : βιβλιοθήκη για την επικοινωνία με περιφερειακές συσκευές

3) LoRa.h : βιβλιοθήκη για το LoRa (sx1278 lora ra-02)

4) MQ135.h : βιβλιοθήκη για τον αισθητήρα διοξειδίου του άνθρακα

Εικόνα 34*: Κώδικας κόμβου (LoRa Transmitter) μέρος 2ο*

Στην *Εικόνα 34* βλέπουμε το δεύτερο μέρος του κώδικα, από όπου συνεχίζεται το void setup στο οποίο πραγματοποιείται η εκκίνηση και ο έλεγχος λειτουργείας του LoRa. Στη συνέχεια, φτάνουμε στο κομμάτι του setup όπου πραγματοποιείται η διαδικασία συλλογής των δεδομένων που συλλέγουν οι αισθητήρες και η αποθήκευση αυτών στις αντίστοιχες μεταβλητές που έχουμε ορίσει (Temperature, Humidity, MQ7Value, MQ135Value).

Εικόνα 35*: Κώδικας κόμβου (LoRa Transmitter) μέρος 3ο*

Στην *Εικόνα 35* βλέπουμε το τρίτο και τελευταίο μέρος του κώδικα, όπου τα δεδομένα που συνέλεξε ο κόμβος πρώτα παρουσιάζονται στον χρήστη. Θα πρέπει να επισημανθεί ότι οι εντολές "serial.print" βρίσκονται εδώ μόνο για να μπορεί ο χρήστης να ελέγξει τα αποτελέσματα των αισθητήρων μέσο σειριακής από το πρόγραμμα Arduino IDE. Οι εντολές αυτές δεν θα βρίσκονται στον τελικό κώδικα, γιατί ο κόμβος (LoRa Transmitter) θα πρέπει να βρίσκεται σε ένα σημείο που θα έχει δύσκολη πρόσβαση ο χρήστης, όπως είναι ένα δάσος στην κορυφή ενός βουνού.

Μετά τις εντολές "serial.print" δηλώνεται ένα string LoRaMessage μέσα στο οποίο βρίσκονται μια σειρά από μεταβλητές οι οποίες είναι τα δεδομένα που καλούμαστε να συλλέξουμε, μαζί με το μοναδική ταυτότητα "id" του κόμβου (LoRa Transmitter). Η χρήση της εντολής string χρησιμοποιήθηκε γιατί μας δίνει την ικανότητα ο δέκτης (LoRa Receiver) να ξεχωρίσει τα δεδομένα που δέχεται από τον κόμβο (LoRa Transmitter) με βάση τους αλφαριθμητικούς χαρακτήρες του μηνύματος. Στον κώδικα του δέκτη θα γίνει περαιτέρω ανάλυση και επεξήγηση της διαδικασίας αυτής.

3.2 Σταθμός βάσης δεδομένων: κώδικας Δέκτη (LoRa Receiver)

Παρακάτω βρίσκεται ο κώδικας για τον σταθμό λήψης των δεδομένων ή δέκτη (LoRa Receiver) που καλούμαστε να συλλέξουμε και να μεταφέρουμε στον χρήστη, χρησιμοποιώντας την τεχνολογία LoRa μέσα από το περιβάλλον του Arduino ID.

Εικόνα 36*: Κώδικας δέκτη (LoRa Receiver) μέρος 1ο*

Στην *Εικόνα 36* βλέπουμε το πρώτο μέρος του κώδικα, στην αρχή του οποίου δηλώνονται οι βιβλιοθήκες που είναι απαραίτητες για τον προγραμματισμό του κόμβου - δέκτη LoRa Receiver. Αυτές είναι:

1) DHT.h και DHT_U : βιβλιοθήκες για τον αισθητήρα θερμοκρασίας και υγρασίας

2) SPI.h : βιβλιοθήκη για την επικοινωνία με περιφερειακές συσκευές

3) LoRa.h : βιβλιοθήκη για το LoRa (sx1278 lora ra-02)

4) ESP8266WiFi.h : βιβλιοθήκη για το NodeMCU ESP8266 WiFi module

Έπειτα καθορίζονται οι μεταβλητές που χρειαζόμαστε για να συνδεθεί το WiFi module με το router, έτσι ώστε να ο κόμβος - δέκτης (LoRa Receiver) να έχει πρόσβαση στο διαδίκτυο. Μαζί με αυτές τις εντολές, ορίζουμε μια μεταβλητή string με τον μοναδικό κωδικό API key του καναλιού που θα εμφανίζονται τα δεδομένα μας στο ThingSpeak. Επιπρόσθετα, ορίζουμε τη λειτουργία του ESP8266 module σαν WiFi. Στις επόμενες γραμμές γίνονται define τα «ποδαράκια» στα οποία θα χρησιμοποιεί το WiFi module από την πλατφόρμα NodeMCU. Κατόπιν, γίνεται η δήλωση πέντε μεταβλητών string, οι οποίες θα χρησιμοποιηθούν για να τοποθετήσουμε τα δεδομένα που θα έρχονται από τον LoRa Transmitter (string device_id, string temperature, string humidity, string MQ7Value, string MQ135Value).

Εικόνα 37*: Κώδικας κόμβου - δέκτη (LoRa Receiver) μέρος 2ο*

Στην *Εικόνα 37* βλέπουμε το δεύτερο μέρος του κώδικα, από όπου συνεχίζεται το void setup στο οποίο πραγματοποιούμε την εγκατάσταση των pins που ορίσαμε στην αρχή του προγράμματος για την εκκίνηση και τον έλεγχο λειτουργείας του LoRa. Στη συνέχεια, πραγματοποιείται η διαδικασία σύνδεσης του WiFi module στο modem χρησιμοποιώντας τον κωδικό και το όνομα του. Κατόπιν, στο while, όταν θα έχει γίνει σύνδεση στο WiFi router που επιλέξαμε, με τις εντολές serial.print εμφανίζεται στη σειριακή η IP διεύθυνση του δικτύου WiFi στο οποίο είναι συνδεδεμένο ο κόμβος - δέκτης LoRa Receiver και στο σημείο αυτό τελειώνει το setup.

Εικόνα 38*: Κώδικας κόμβου - δέκτη (LoRa Receiver) μέρος 3ο*

Στην *Εικόνα 38* βλέπουμε στο τρίτο μέρος του κώδικα την αρχή του **void loop.** Στην αρχή του void loop δηλώνουμε πέντε μεταβλητές τύπου int (pos1, pos2, pos3, pos4, pos5), αυτές οι μεταβλητές θα καθορίσουν παρακάτω τέσσερα σημεία στο μήνυμα που θα δέχεται ο δέκτης (LoRa Receiver) μέσα από το LoRa. Αφού ελέγξουμε αν έχει ληφθεί από το δέκτη ένα LoRa μήνυμα, το τοποθετούμε στην μεταβλητή packetSize. Έπειτα, έχουμε μια δομή if, στην αρχή της οποίας όταν θα ληφθεί ένα μήνυμα packetSize, αρχικά θα τυπώνεται στο serial το περιεχόμενο της μεταβλητής - string μηνύματος LoRaMessage που φτιάξαμε στον κόμβο, μαζί με την ένταση του σήματος επικοινωνίας (RSSI).

Στο σημείο αυτό παρουσιάζεται η διαδικασία με την οποία ξεχωρίζουμε και παίρνουμε τα δεδομένα από τη μεταβλητή - string LoRaMessage που δέχεται ο δέκτης για να τα στείλουμε στο ThingSpeak. Με τη χρήση των τεσσάρων μεταβλητών τύπου int (pos1, pos2, pos3, pos4) που ορίσαμε παραπάνω στον κώδικα, θα ξεχωρίσουμε τα δεδομένα της θερμοκρασίας, της υγρασίας, του

μονοξειδίου και του διοξειδίου του άνθρακα που βρίσκονται στη σειρά από χαρακτήρες από την οποία απαρτίζεται το string LoRaMessage. Ο τρόπος με τον οποίο ξεχωρίζουμε αυτά τα πέντε δεδομένα σε αυτή τη σειρά από αλφαριθμητικούς χαρακτήρες είναι ο εξής:

Για το πρώτο δεδομένο ορίζουμε μια μεταβλητή devise_id μέσα στην οποία θα τοποθετηθεί ένα substring το οποίο αποτελείται από τους χαρακτήρες του αρχικού μηνύματος LoRaMessage που βρίσκονται μεταξύ δύο ορίων, στην προκειμένη περίπτωση για το πρώτο δεδομένο, το πρόγραμμα θα ξεκινήσει να διαβάζει χαρακτήρες από αριστερά προς τα δεξιά μέχρι να συναντήσουμε το χαρακτήρα που θέσαμε ως όριο παραπάνω στον κώδικα. Το όριο αυτό ονομάζετε **pos1** και ο χαρακτήρας του είναι « / ».

Το μήνυμα έχει ένα συγκεκριμένο μέγεθος από χαρακτήρες σύμφωνα με αυτό που έγινε η κατασκευή στον κόμβο LoRa Transmitter. Το πρόγραμμα πηγαίνει και διαβάζει την πρώτη θέση του αντικειμένου στο μήνυμα LoRaMessage από αριστερά προς τα δεξιά. Άρα θα ξεκινήσει από τη θέση μηδέν «0».

Οι χαρακτήρες που ξεκινάνε από την πρώτη θέση του μηνύματος η οποία είναι το μηδέν, μέχρι τον χαρακτήρα που θέσαμε ως όριο, θα αποτελούν την ταυτότητα ID του κόμβου LoRa Transmitter. Επόμενος στο serial θα πρέπει να δούμε για το πρώτο στοιχείο την έξοδο: Device ID = MyDevice123.

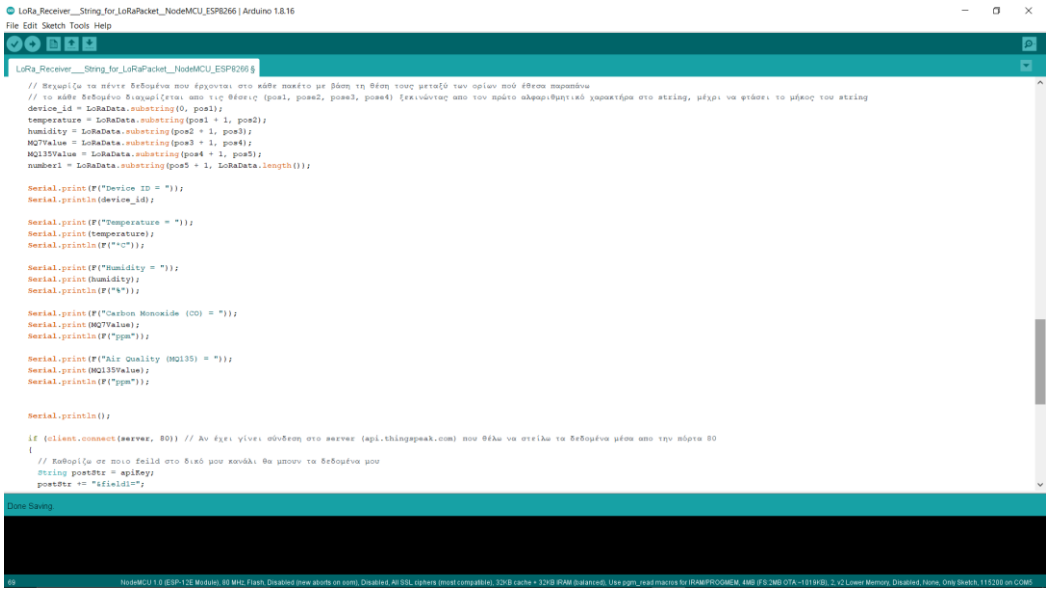
Για το δεύτερο δεδομένο ορίζουμε μια μεταβλητή **temperature** μέσα στην οποία θα τοποθετηθεί ένα substring το οποίο αποτελείται από τους χαρακτήρες του αρχικού μηνύματος LoRaMessage που βρίσκονται μια θέση αμέσως μετά από το προηγούμενο όριο μέχρι να συναντήσουμε το χαρακτήρα που θέσαμε ως το επόμενο όριο, στην προκειμένη περίπτωση το όριο αυτό ονομάζετε **pos2** και ο χαρακτήρας του είναι « **&** ». Επομένως, στο serial θα πρέπει να δούμε για την δεύτερη τιμή στην έξοδο: temperature = (την αριθμητική τιμή που έστειλε ο LoRa Transmitter).

Για το τρίτο δεδομένο ορίζουμε μια μεταβλητή **humidity** μέσα στην οποία θα τοποθετηθεί ένα substring το οποίο αποτελείται από τους χαρακτήρες του αρχικού μηνύματος LoRaMessage που βρίσκονται μια θέση αμέσως μετά από το προηγούμενο όριο μέχρι να συναντήσουμε το χαρακτήρα που θέσαμε ως το επόμενο όριο, στην προκειμένη περίπτωση το όριο αυτό ονομάζετε **pos3** και ο χαρακτήρας του είναι « # ». Επομένως στο serial θα πρέπει να δούμε για την τρίτη τιμή στην έξοδο: humidity = (την αριθμητική τιμή που έστειλε ο LoRa Transmitter).

Για το τέταρτο δεδομένο ορίζουμε μια μεταβλητή **MQ7Value** μέσα στην οποία θα τοποθετηθεί ένα substring το οποίο αποτελείται από τους χαρακτήρες του αρχικού μηνύματος LoRaMessage που βρίσκονται μια θέση αμέσως μετά από το προηγούμενο όριο **pos3** μέχρι να

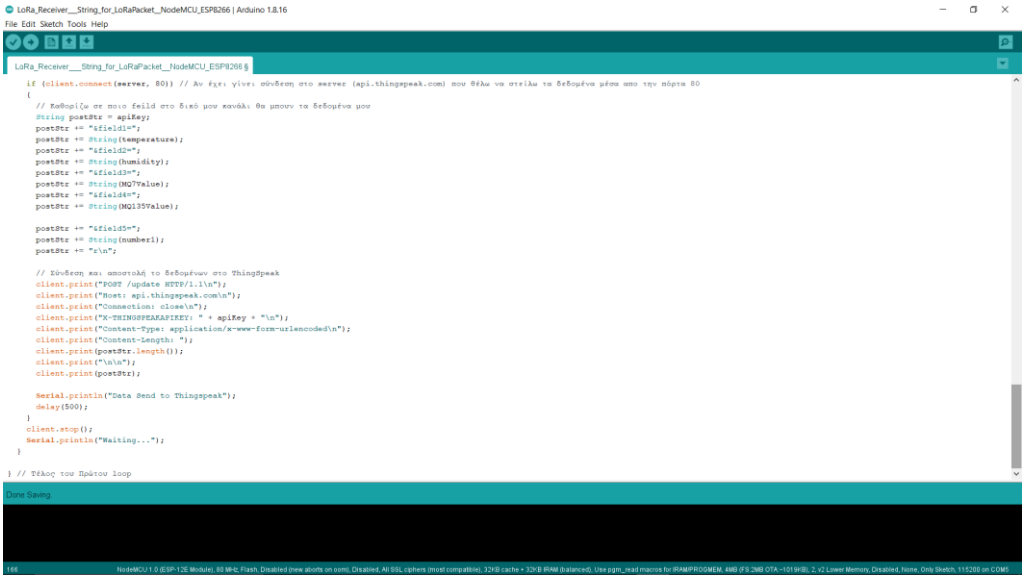
συναντήσουμε το χαρακτήρα που θέσαμε ως το επόμενο όριο, στην προκειμένη περίπτωση το όριο αυτό ονομάζετε **pos4** και ο χαρακτήρας του είναι « **@** ». Επόμενος στο serial θα πρέπει να δούμε την τιμή στην έξοδο: MQ7Value = (την αριθμητική τιμή που έστειλε ο LoRa Transmitter). Για το πέμπτο δεδομένο το διοξείδιο του άνθρακα (**MQ135Value**), τα δεδομένα που βρίσκονται μια θέση αμέσως μετά από το προηγούμενο όριο **pos4** μέχρι το **pos5**.

Αυτή η διαδικασία θα επαναλαμβάνεται για όλα τα δεδομένα (θερμοκρασία, υγρασία, μονοξείδιο και διοξείδιο του άνθρακα), για το κάθε πακέτο που θα δέχεται ο δέκτης.



Εικόνα 39*: Κώδικας δέκτη (LoRa Receiver) μέρος 4ο*

Στην *Εικόνα 39* έχουμε τις εντολές serial.print για τις τιμές των δεδομένων που πήραμε από το LoRa μήνυμα, αυτές υπάρχουν στον κώδικα μόνο για να μπορεί ο χρήστης να κάνει επαλήθευση των δεδομένων που δέχεται από τον LoRa Transmitter.



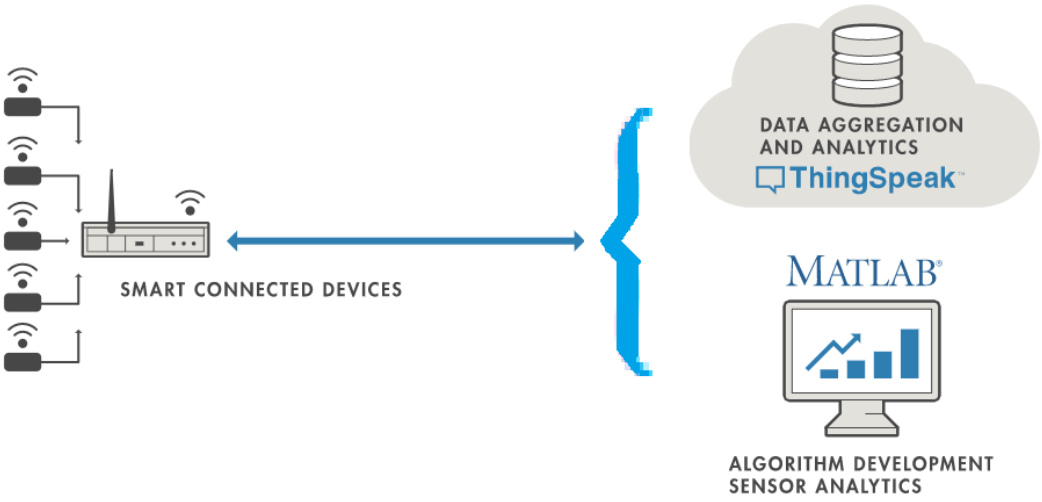
Εικόνα 40*: Κώδικα δέκτη (LoRa Receiver) μέρος 5ο*

Στην *Εικόνα 40* βλέπουμε το πέμπτο και τελευταίο μέρος του κώδικα της αποστολής και τοποθέτησης τον δεδομένων στο προσωπικό κανάλι στο ThingSpeak. Ανοίγουμε μια δομή ελέγχου if με την οποία θα στέλνουμε τα δεδομένα που πήραμε από την επικοινωνία LoRa. Αν η δομή επανάληψης ισχύει δηλαδή έχουμε συνδεθεί στο server που έχουμε ορίσει στην αρχή του προγράμματος μέσω της πόρτας 80 του router τότε με τις εντολές **postStr** θα τοποθετήσουμε τα δεδομένα μας στα πεδία – field του καναλιού μας στο ThingSpeak από όπου θα τα βλέπει ο κάθε χρήστης.

Κεφάλαιο 4ο: Γραφικό περιβάλλον χρήστη

Σημαντικό κομμάτι της κατασκευής μιας πλατφόρμας για την έγκαιρη προειδοποίηση ενός χρήστη είναι η κατασκευή ενός περιβάλλοντος το οποίο θα ενημερώνει τον χρήστη για την κατάσταση την οποία καλείται να αντιμετωπίσει προσφέροντας του όλες τις πληροφορίες που χρειάζεται με απλό και κατανοητό τρόπο. Έτσι ώστε ο χρήστης να αντιλαμβάνεται άμεσα και με ευκολία την κατάσταση για την οποία είναι υπεύθυνος να παρακολουθεί, για να μπορεί να αντιδράσει κατάλληλα.

4.1 ThinkSpeak



Εικόνα 41*: ThingSpeak model*

Το ThingSpeak είναι μια υπηρεσία ανάλυσης ενός δικτύου IoT που επιτρέπει την συγκέντρωση, την οπτικοποίηση και την ανάλυση ζωντανών ροών δεδομένων στο cloud. Η υπηρεσία αυτή παρέχει άμεσες απεικονίσεις των δεδομένων που δημοσιεύονται από τις συσκευές που είναι συνδεδεμένες στο δίκτυο IoT του ThingSpeak, με μεγάλη ευκολία [10]. Επίσης το ThingSpeak έχει την δυνατότητα εκτέλεσης κώδικα MATLAB, από όπου είναι δυνατό να εκτελείται διαδικτυακά η ανάλυση και η επεξεργασία δεδομένων όπως αυτά εισέρχονται από το φυσικό περιβάλλον.

Εφόσον οι συσκευές που θέλω να συνδέσω στο διαδίκτυο έχουν πρόσβαση στο internet, μπορώ να στείλω δεδομένα από οποιαδήποτε συσκευή απευθείας στο ThingSpeak χρησιμοποιώντας ένα μοναδικό Rest API κλειδί ή MQTT της σελίδας.

Επιπλέον, οι ενσωματώσεις cloud-to-cloud με το The Things Network, το Senet, την πύλη Libelium Meshlium και το Particle.io επιτρέπουν στα δεδομένα αισθητήρων να φτάσουν στο ThingSpeak μέσω συνδέσεων κινητής τηλεφωνίας LoRaWAN® και 4G/3G.

Κατόπιν με το ThingSpeak, μπορώ να αποθηκεύσω και να αναλύσω τα δεδομένα μου στο cloud χωρίς να διαμορφώνεται ένας διακομιστής ιστού όπως επίσης υπάρχει και η δυνατότητα να δημιουργήσω εξειδικευμένες ειδοποιήσεις email που βασίζονται σε συμβάντα που ενεργοποιούνται με βάση τα δεδομένα που προέρχονται από τις συνδεδεμένες συσκευές μου [11].

Η επιλογή της υπηρεσίας ThingSpeak έγινε λόγω της δυνατότητας δωρεάν χρήσης, της μεγάλης προσβασιμότητας και ευκολίας που προσφέρει να χειρίζεται ένας χρήστης, έως και 90.000 μηνύματα την ημέρα και να τα αποθηκεύει για ένα χρόνο. Ταυτόχρονα προσφέρει ένα user-friendly περιβάλλον με πολλά εξαρτήματα, πίνακες, διαγράμματα και πολλά άλλα γραφικά που καθιστούν εύκολη και κατανοητή την παρακολούθηση τον δεδομένων που μπορεί να δει ο κάθε χρήστης εφόσον έχει το σύνδεσμό ή link, για την πρόσβαση στον συγκεκριμένο ιστότοπο της κατασκευής ή είναι εγγεγραμμένος να δέχεται δεδομένα στο προσωπικό του email.

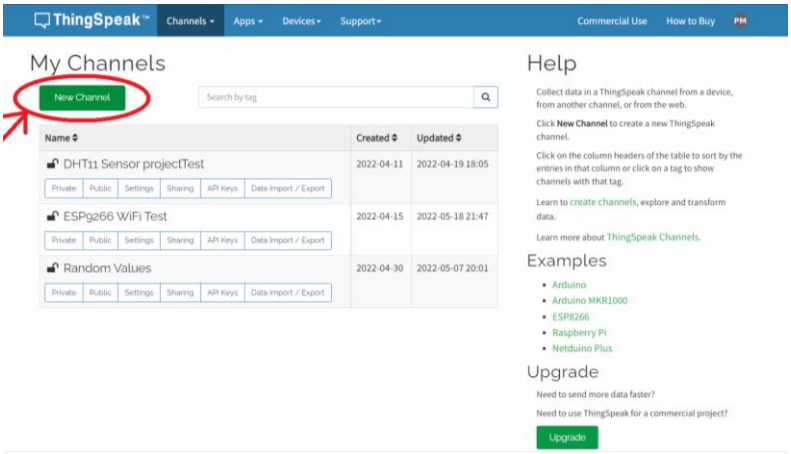
4.2 Ανάλυση του σχεδιασμού περιβάλλοντος του ThingSpeak

4.2.1 Κατασκευή προσωπικού καναλιού στο ThingSpeak

Σε αυτό το κομμάτι της κατασκευής γίνεται η κατασκευή του προσωπικού καναλιού στο ThingSpeak server στο οποίο θα τοποθετήσουμε τα δεδομένα που δέχεται ο κόμβος – δέκτης LoRa Receiver για τις μετρήσεις των φαινομένων που παρουσιάζονται στην περιοχή που βρίσκεται ο κόμβος LoRa Transmitter. Πρώτα συνδεόμαστε με τον προσωπικό μας λογαριασμό στην σελίδα του ThingSpeak και φτιάχνουμε ένα νέο κανάλι στο οποίο θα παρακολουθούμε τα δεδομένα από τον κόμβο LoRa transmitter.

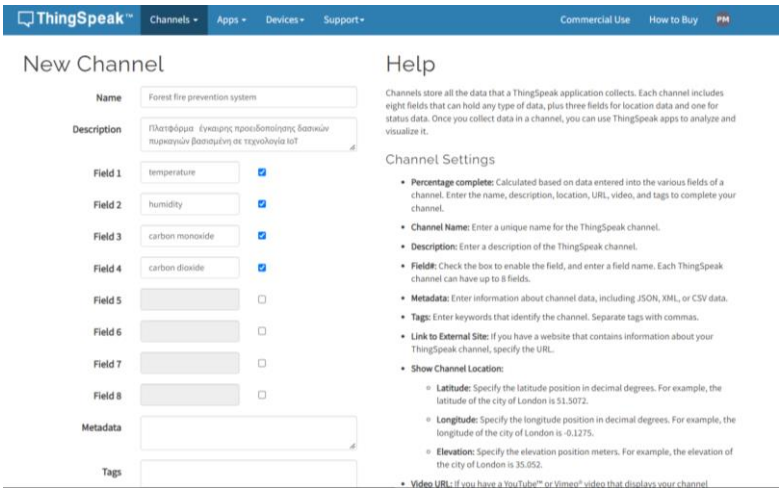
Στην συνέχεια, *Εικόνα 42,* από τη μπάρα επιλογών, πηγαίνουμε στο Channels και έπειτα στο My channels και φτιάχνουμε ένα νέο κανάλι για την κατασκευή.

Στο νέο κανάλι που φτιάξαμε, συμπληρώνουμε το όνομα, την περιγραφή και τα πεδία που θέλουμε να χρησιμοποιήσουμε (*Εικόνα 43)*. Στην παρούσα κατασκευή θα χρειαστούμε τέσσερα πεδία για τα τέσσερα δεδομένα που συλλέγει ο κόμβος – δέκτης (LoRa Receiver).



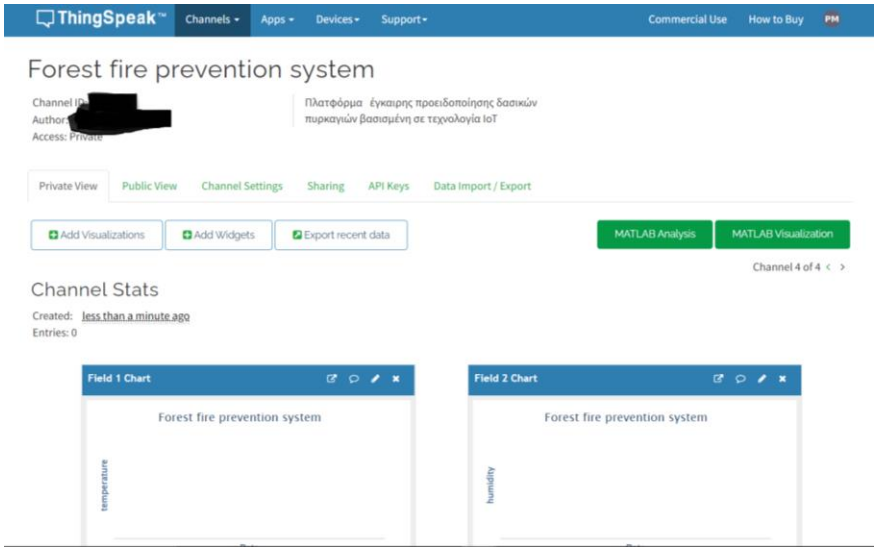
Εικόνα 42*: Κατασκευή νέου καναλιού*

Στο νέο κανάλι που φτιάξαμε, συμπληρώνουμε το όνομα, την περιγραφή και τα πεδία που θέλουμε να χρησιμοποιήσουμε (*Εικόνα 43)*. Στην παρούσα κατασκευή θα χρειαστούμε τέσσερα πεδία για τα τέσσερα δεδομένα που συλλέγει ο κόμβος – δέκτης (LoRa Receiver).



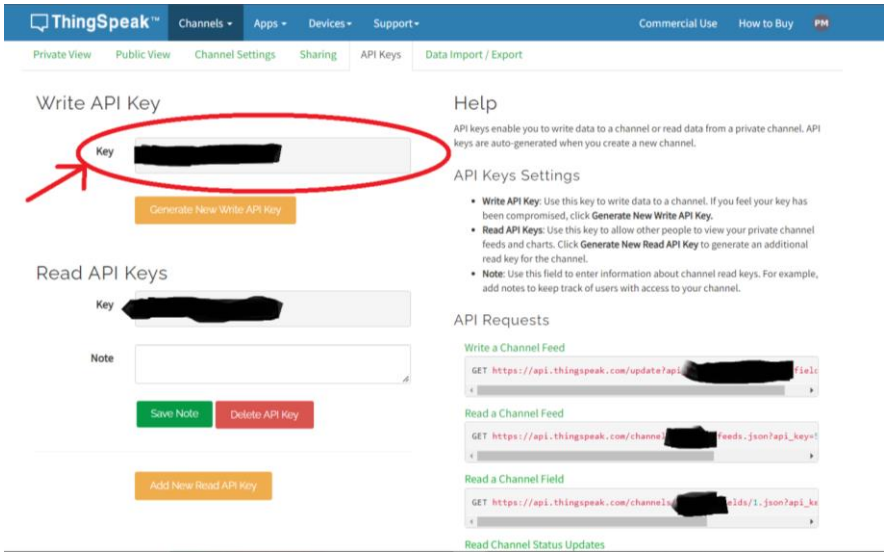
Εικόνα 43*: Χαρακτηριστικά καναλιού*

Στην Εικόνα 44, αφού φτιάξαμε το κανάλι με την χρήση των γραφικών που προσφέρει η πλατφόρμα ThingSpeak, μπορούμε να οπτικοποιήσουμε τα δεδομένα με όποιο τρόπο θέλουμε.



Εικόνα 44*: Γραφικό περιβάλλον του καναλιού*

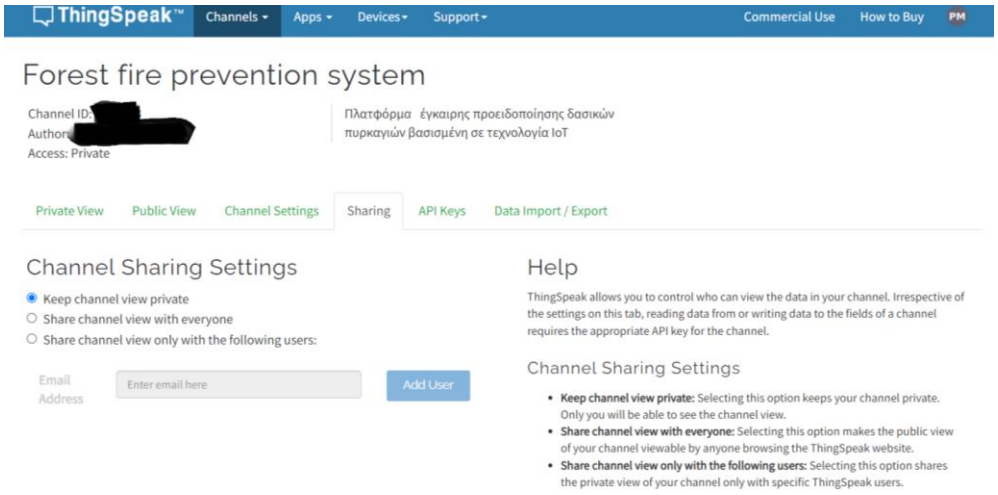
Στην *Εικόνα 45*, στη μπάρα στοιχείων επιλέγουμε «API Keys». Στη σελίδα αυτή μπορούμε να αλλάξουμε ή να χρησιμοποιήσουμε αλλά κλειδιά που επιθυμούμε. Στη δική μας περίπτωση θα κρατήσουμε το API key που μας δίνεται στο σημείο «Write API Key». Το κλειδί αυτό θα μας επιτρέψει να έχουμε πρόσβαση στο κανάλι μας για να μπορέσουμε να τοποθετήσουμε τα δεδομένα που έχει συλλέξει ο δέκτης LoRa Receiver.



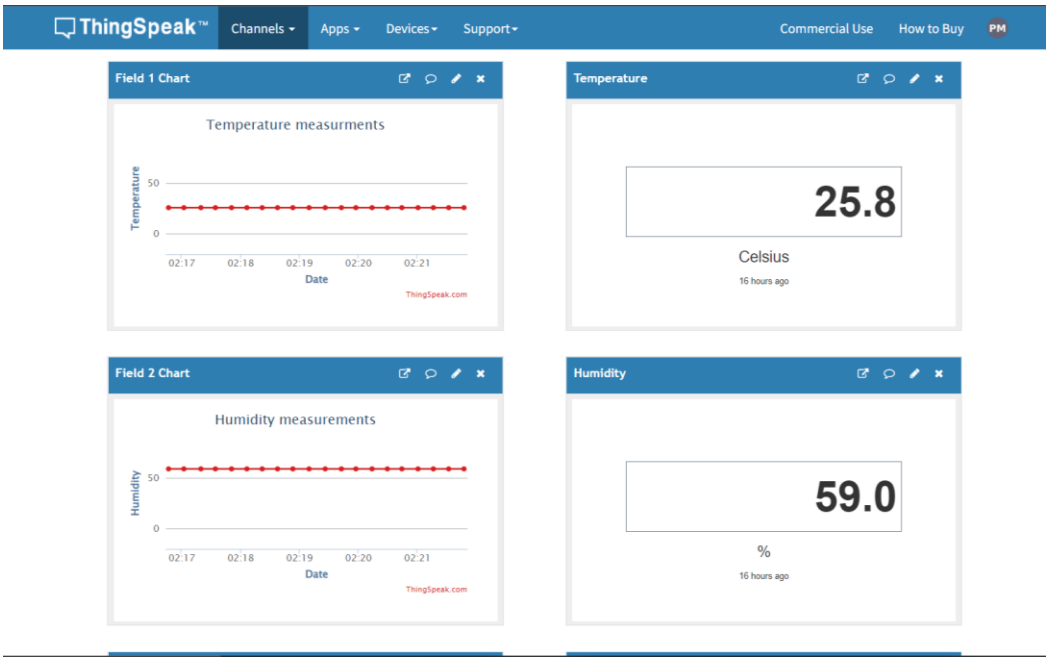
Εικόνα 45*: Μοναδικό κλειδί του καναλιού που φτιάξαμε*

Στην *Εικόνα 46* βρισκόμαστε στην καρτέλα sharing στην οποία ρυθμίζουμε την προσβασιμότητα στο κανάλι που φτιάξαμε. Σε αυτή μπορούμε να θέσουμε το κανάλι σε ιδιωτική ή δημόσια προβολή. Στη δική μας περίπτωση θέτουμε το κανάλι σε ιδιωτική προβολή. Στις παρακάτω εικόνες βλέπουμε το γραφικό περιβάλλον του καναλιού με όλα τα δεδομένα προς παρακολούθηση.

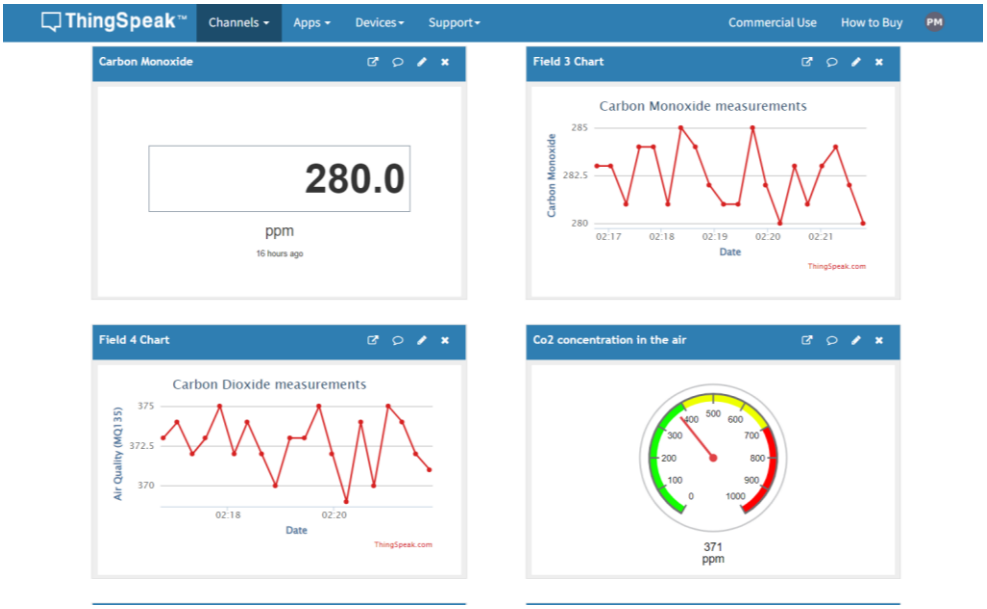
Η πλατφόρμα ThingSpeak επιτρέπει την προβολή δεδομένων με καθυστέρηση δεκαπέντε δευτερολέπτων. Αυτό μας δίνει τη δυνατότητα για συνεχόμενη παρακολούθηση δεδομένων.



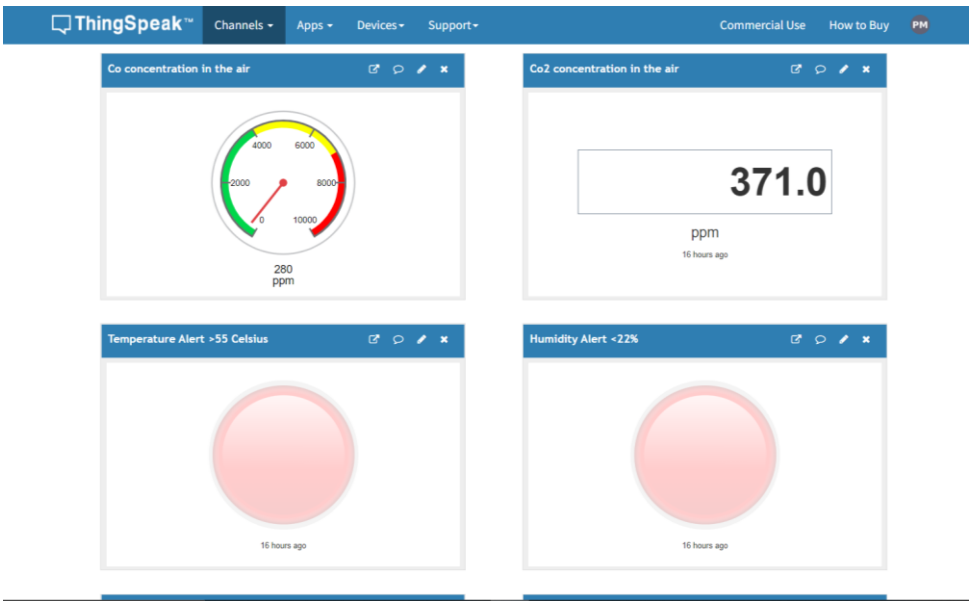
Εικόνα 46: *Προσβασιμότητα στο κανάλι*



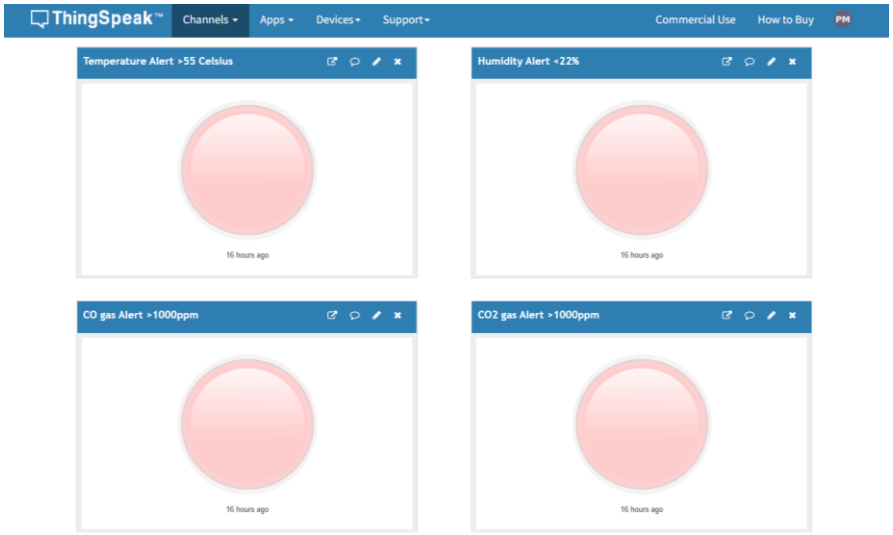
Εικόνα 47: *Γραφικό περιβάλλον του προσωπικού μας καναλιού*



Εικόνα 48: ***Γραφικό περιβάλλον του προσωπικού μας καναλιού***



Εικόνα 49: ***Γραφικό περιβάλλον του προσωπικού μας καναλιού***



Εικόνα 50*: Γραφικό περιβάλλον του προσωπικού μας καναλιού – λυχνίες προειδοποίησης*

Το τελευταίο κομμάτι της κατασκευής είναι η υλοποίηση του συστήματος έγκαιρης προειδοποίησης του χρήστη σε περίπτωση δασικής πυρκαγιάς. Για να κατασκευάσουμε ένα τέτοιο σύστημα θα πρέπει να επιλέξουμε τον τρόπο με τον οποίο ο χρήστης θα προειδοποιείται για την ύπαρξη πυρκαγιάς. Ο τρόπος με τον οποίο θα ενημερώνεται ο χρήστης στην κατασκευή αυτή είναι:

α) Βλέποντας τις λυχνίες προειδοποίησης που τοποθετήσαμε στο γραφικό περιβάλλον (*Εικόνα 50)* του καναλιού οι οποίες θα ανάβουν όταν οι τιμές των αισθητήρων έχουν ξεπεράσει τα όρια που θέσαμε.

β) Μέσω ενός προσωπικού μηνύματος (email) το οποίο θα στέλνεται κάθε φόρα που οι συνθήκες για την ύπαρξη πυρκαγιάς πληρούνται βάσει ενός κώδικα που θα συγκρίνει τα δεδομένα που έχει αποθηκευμένα το κανάλι μας στο ThingSpeak.

4.2.2 Κατασκευή του μηνύματος προειδοποίησης

Για την κατασκευή του μηνύματος προειδοποίησης που θα στέλνεται στο χρήστη θα χρησιμοποιήσουμε δυο εφαρμογές, την MatLab TimeControl και την MatLab Analysis. Για τη δημιουργία των συνθηκών για την ύπαρξη πυρκαγιάς πρέπει να αναλύσουμε τα δεδομένα στο κανάλι. Αυτό επιτεύχθηκε δημιουργώντας ένα MATLAB Analysis. Η πλατφόρμα ThingSpeak έχει ενσωματωμένη ένα MATLAB compiler το οποίο δίνει την δυνατότητα στον χρήστη για:

α) Την επεξεργασία των δεδομένων που συλλέγονται σε ένα κανάλι

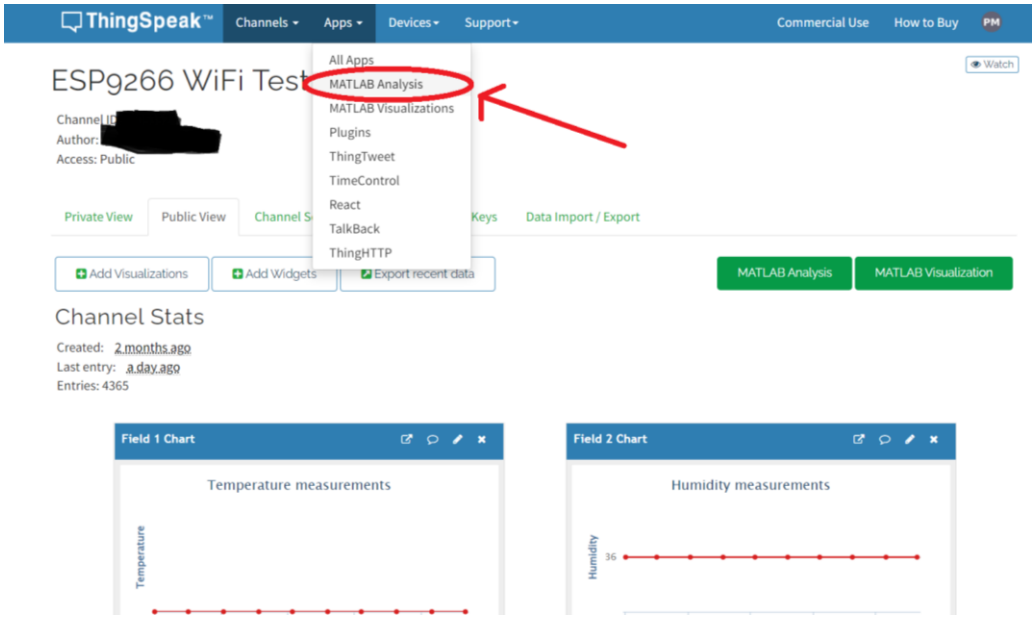
β) Την εύρεση και κατάργηση μη χρήσιμων δεδομένων από το κανάλι

γ) Τη μετατροπή δεδομένων σε διαφορετικές μονάδες

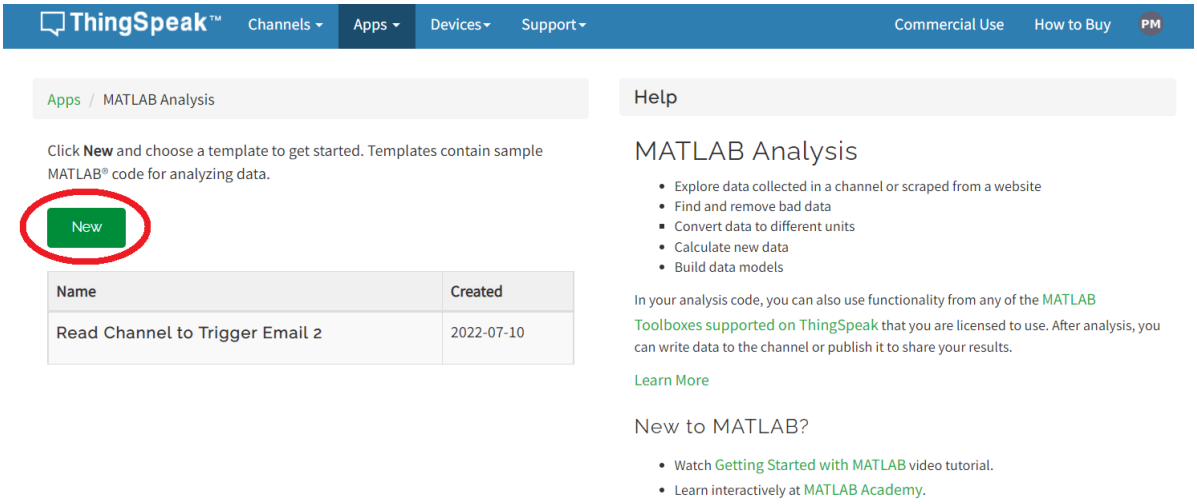
δ) Τον υπολογισμό νέων δεδομένων

ε) Τη δημιουργία μοντέλων δεδομένων

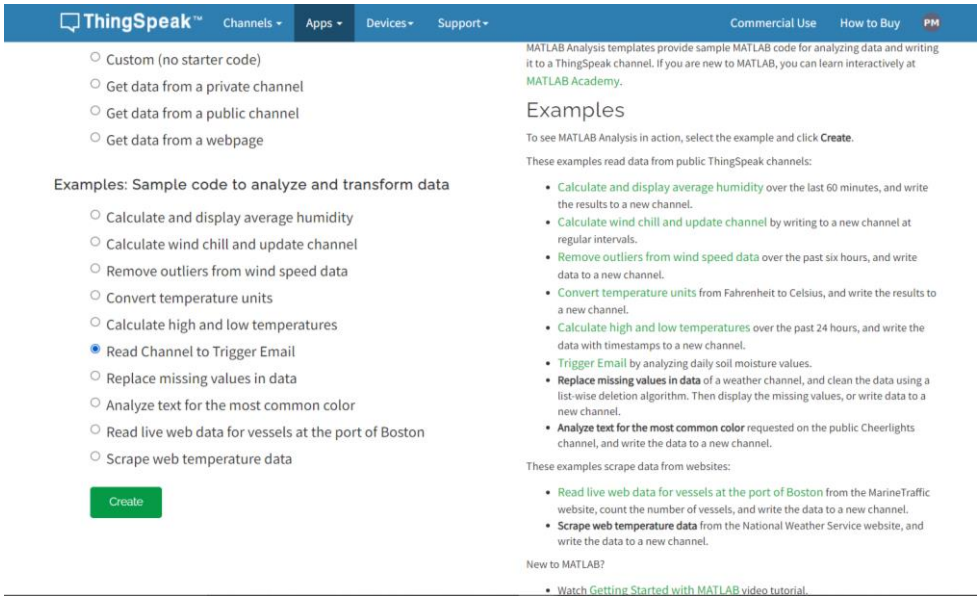
Στις παρακάτω εικόνες βλέπουμε τη διαδικασία που πραγματοποιήθηκε για την κατασκευή του MatLab Analysis.



Εικόνα 51*: Επιλέγω από την γραμμή εργασιών το app MATLAB Analysis*

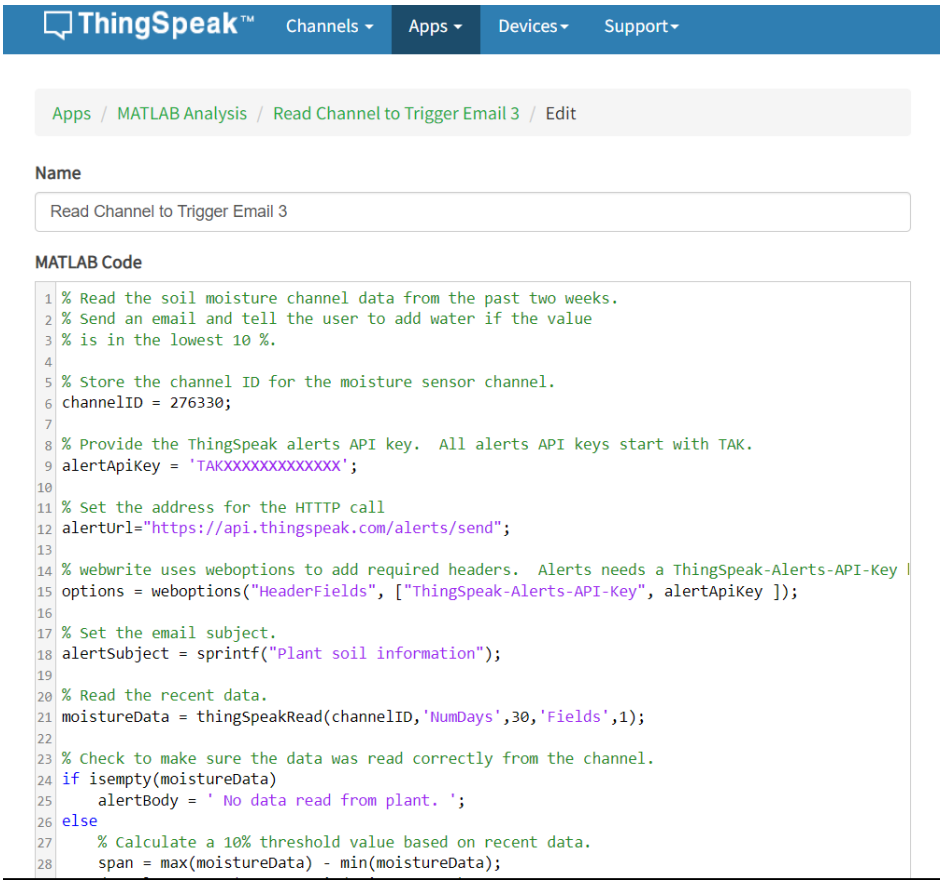


Εικόνα 52*: Φτιάχνω νέα ανάλυση*



Εικόνα 53*: Επιλογές από έτοιμα παραδείγματα κώδικα*

Στο σημείο αυτό που δείχνει η *Εικόνα 53* μπορούμε να φτιάξουμε διάφορα είδη ανάλυσης των δεδομένων από παραδείγματα που προσφέρει το ThingSpeak. Εδώ επιλέχθηκε η ανάγνωση δεδομένων από ένα κανάλι και αποστολή email μηνύματος.



Εικόνα 54*: Παράδειγμα Matlab κώδικα*

Ο κώδικας Matlab που μας δίνεται αποτελεί ένα παράδειγμα για την αποστολή ενός μηνύματος όταν ένα φυτό χρειάζεται πότισμα *Εικόνα 54*. Στη διπλωματική αυτή δεν μου χρησιμεύει ο κώδικας αυτός, για τον λόγο αυτό διέγραψα το πρόγραμμα που μου δόθηκε και έγραψα ένα νέο κώδικα που θα με εξυπηρετήσει για τους λόγους που ανέφερα προηγουμένως.

Για τη δημιουργία του προσωπικού μηνύματος που θα στέλνεται στον χρήστη έπρεπε να γράψω ένα κώδικα ο οποίος θα ενημερώνει ξεκάθαρα τον χρήστη, χωρίς περιττές πληροφορίες. Αυτό επιτεύχθηκε με την χρήση τεσσάρων συνθηκών που έθεσα για τα δεδομένα που παρακολουθώ, οι οποίες, αν είναι αληθείς , θα υποδηλώνουν την ύπαρξη φωτιάς.

Οι συνθήκες οι οποίες υποδηλώνουν την πιθανότητα για την ύπαρξη μιας φωτιάς - πυρκαγιάς βασίζονται σε ένα πρόγραμμα το οποίο θα συσχετίσει τα τέσσερα δεδομένα που συλλέγει ο κόμβος. Η συνθήκη που πρέπει να ισχύει είναι:

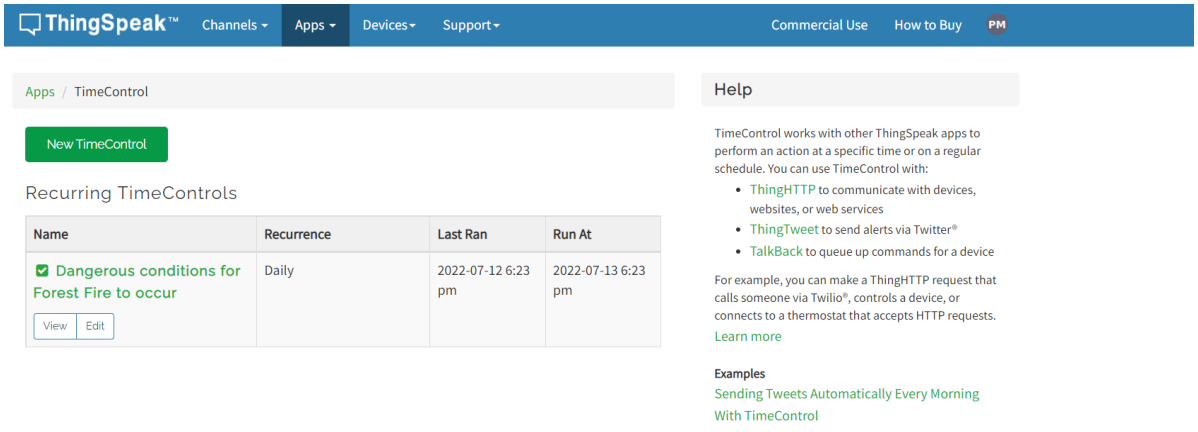
Θερμοκρασία > 55Celsius AND Υγρασία < 22% AND Μονοξείδιο του άνθρακα > 1000ppm AND Διοξείδιο του άνθρακα > 1000ppm

Όταν η συνθήκη αυτή είναι αληθής τότε το πρόγραμμα θα στέλνει ένα μήνυμα όπως βλέπουμε στην *Εικόνα 59*. Για τη σωστή λειτουργία του συστήματος χρησιμοποιήθηκε η εφαρμογή χρονοελεγκτή (TimeControl app) η οποία λειτουργεί με άλλες εφαρμογές και τις πληροφορίες που έχει ένα κανάλι για να εκτελέσει μια ενέργεια, σε ένα συγκεκριμένο χρονικό διάστημα ή σε ένα καθημερινό πρόγραμμα που επιθυμούμε. Στη συγκεκριμένη περίπτωση θα χρησιμοποιήσουμε τον χρονοελεγκτή (TimeControl app) για να εκτελεί κάθε πέντε λεπτά το πρόγραμμα που φτιάξαμε στο MatLab analysis, δηλαδή θα ελέγχει με βάση τα δεδομένα από το κανάλι “Forest fire prevention system” τις συνθήκες που υποδηλώνουν την ύπαρξη φωτιάς.

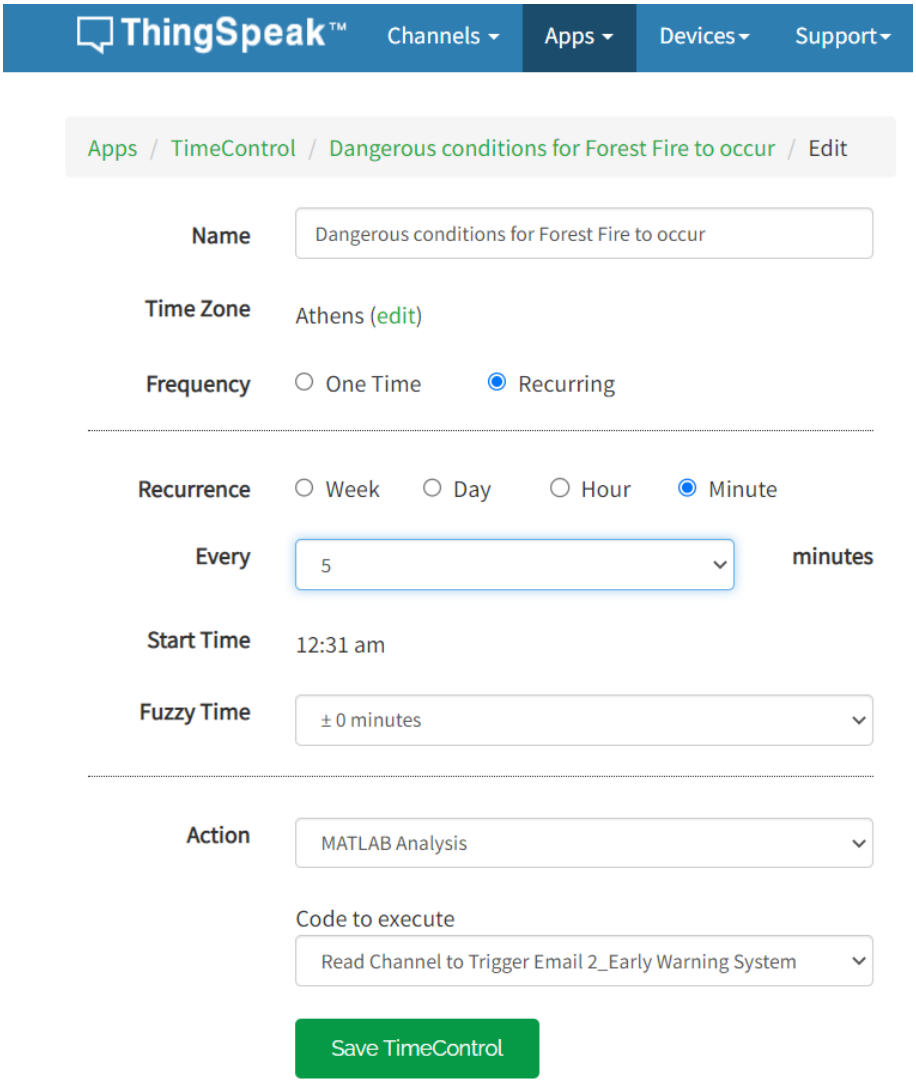
MATLAB Code

```
channelID = [redacted]
alertApiKey = [redacted];

alertUrl="https://api.thingspeak.com/alerts/send";
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key", alertApiKey ]);



temperature = thingSpeakRead(channelID,'Fields',1);

humidity = thingSpeakRead(channelID,'Fields',2);

COvalue = thingSpeakRead(channelID,'Fields',3);

CO2value = thingSpeakRead(channelID,'Fields',4);


alertSubject = sprintf("*** ΚΙΝΔΥΝΟΣ - ΦΩΤΙΑ ***");

alertBody = sprintf(" Οι ενδείξεις έχουν υπερβεί τα όρια ασφαλείας!");

%if temperature > 10  % γραμμή προγράμματος για επίδειξη

  if temperature > 55 & humidity < 22 & COvalue > 1000 & CO2value > 1000

  webwrite(alertUrl , "body", alertBody, "subject", alertSubject, options);   %  γραμμή προγρά

 end

```

Εικόνα 55*: Κώδικας για το μήνυμα προειδοποίησης στο MatLab Analysis*

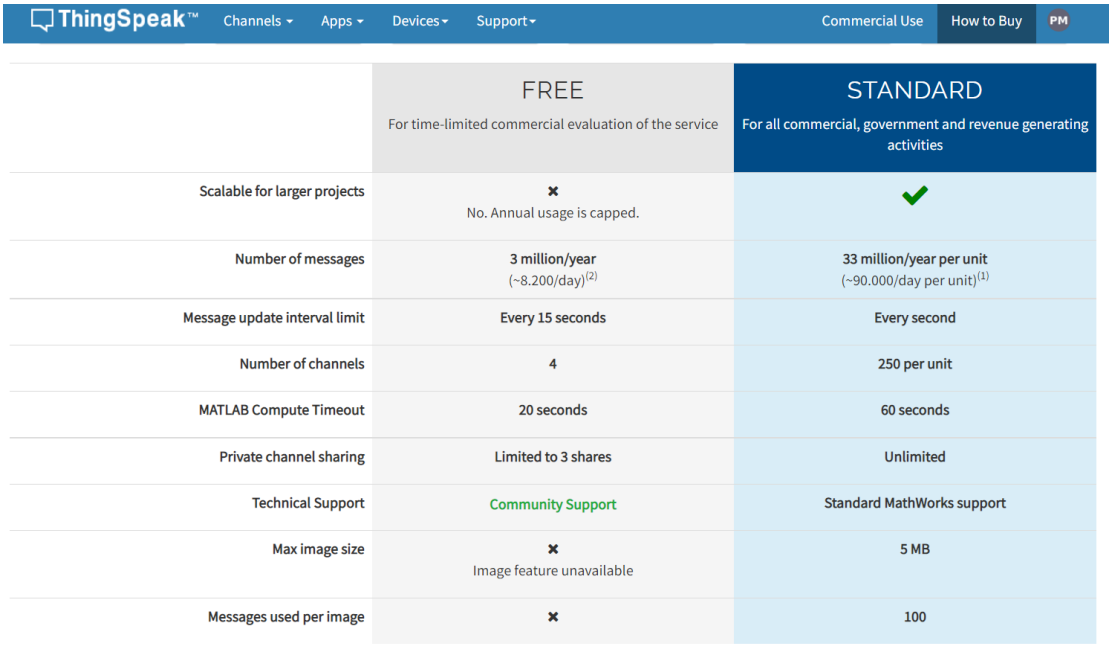


Εικόνα 56*: Κατασκευή νέου χρονοελεγκτή – TimeControl*



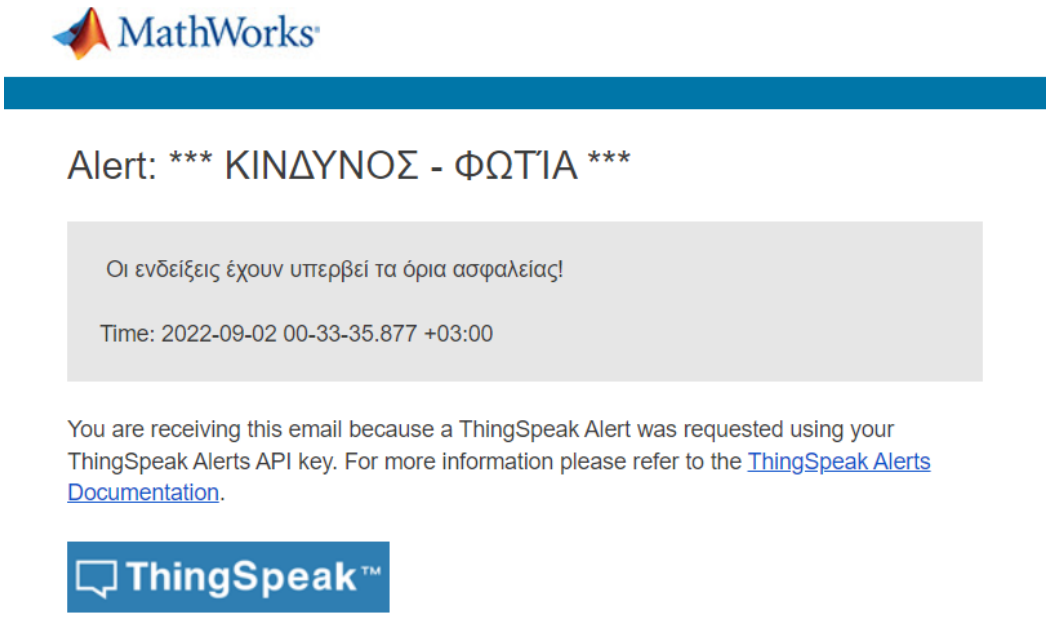
Εικόνα 57*: Ορισμός του χρόνου για την εκτέλεση του MatLab Analysis*

Πρώτα κατασκευάστηκε μια νέα εφαρμογή τύπου χρονοελεγκτή Time Control, όπως φαίνεται στην *Εικόνα 56* και *Εικόνα 57*. Ο χρονοελεγκτής εκτελεί την εφαρμογή MATLAB Analysis κάθε πέντε λεπτά σύμφωνα με το χρονικό όριο που τέθηκε. Η χρήση του χρονοελεγκτή έγινε για να περιορίσουμε τον αριθμό των ειδοποιήσεων που έρχονται, γιατί υπάρχει περιορισμένος αριθμός μηνυμάτων που μπορεί να στείλει η πλατφόρμα λόγω του γεγονότος ότι γίνεται χρήση δωρεάν λογαριασμού, χωρίς τις δυνατότητες ενός standard χρήστη [12].



Εικόνα 58*: Τύπος χρήστη συμφώνα με την πλατφόρμα ThingSpeak Free License vs Standard*

Η εφαρμογή Time Control κάθε 5 λεπτά ενεργοποιεί το MATLAB Analysis που με την σειρά του ελέγχει τις προϋποθέσεις για την αποστολή ή όχι του μηνύματος. Όταν οι προϋποθέσεις πληρούνται, βλέπουμε το μήνυμα που στάλθηκε όπως φαίνεται στην *Εικόνα 59*.



Εικόνα 59*: Μήνυμα προειδοποίησης του χρήστη για την ύπαρξη φωτιάς*

ΚΕΦΑΛΑΙΟ 5ο: ΣΥΜΠΕΡΑΣΜΑΤΑ

Το θέμα της παρούσας διπλωματικής εργασίας ήταν η προειδοποίηση ενός χρήστη για την ύπαρξη μιας δασικής πυρκαγιάς. Οι δασικές πυρκαγιές είναι ένα συχνό και επικίνδυνό φαινόμενο που πλήττει τις δασικές περιοχές στην Ελλάδα και ολόκληρο τον κόσμο. Στόχος της εργασίας ήταν η ανάπτυξη μιας πλατφόρμας για την έγκαιρη προειδοποίηση της ύπαρξης μιας φωτιάς, χρησιμοποιώντας την τεχνολογία του διαδικτύου των πραγμάτων (Internet of Things – IoT), έτσι ώστε να μπορούμε να έχουμε στοιχεία για το αν αυτή η φωτιά θα εξελιχθεί σε πυρκαγιά.

Οι προδιαγραφές που τέθηκαν για τη δημιουργία της κατασκευής αυτής ήταν να υπάρχει χαμηλό κόστος κατασκευής, να έχει επεκτασιμότητα, να είναι αξιόπιστο, να έχει μικρό μέγεθος, να έχει αυτονομία, να έχει μεγάλη εμβέλεια και να είναι ευανάγνωστο στις πληροφορίες που θα δίνει. Θα πρέπει να κατασκευαστεί ένα σύστημα το οποίο θα συλλέγει κάποια δεδομένα που σχετίζονται με τις δασικές πυρκαγιές. Εφόσον μιλάμε για δασικές πυρκαγιές, τα δεδομένα αυτά μπορεί να βρίσκονται σε απομακρυσμένες και μη προσβάσιμες περιοχές λόγω του εδάφους, της μορφολογίας και της βλάστησης της περιοχής.

Τα βήματα που πραγματοποιήθηκαν για την κατασκευή ενός συστήματος προειδοποίησης ξεκίνησαν από την κατανόηση του προβλήματος που κλήθηκα να λύσω. Το πρώτο βήμα που έγινε ήταν η κατανόηση της φύσης μιας δασικής πυρκαγιάς. Άρα αναλύοντας το φαινόμενο αυτό βγήκε το συμπέρασμα ότι για να ανιχνευθεί η ύπαρξη μιας φωτιάς, πριν αυτή εξελιχθεί σε πυρκαγιά, θα πρέπει να ανιχνευθεί πρώτα η ύπαρξη μεγάλης θερμοκρασίας, χαμηλού επιπέδου υγρασίας και τα υψηλά επίπεδα μονοξειδίου και διοξειδίου του άνθρακα στον αέρα. Αυτά τα τέσσερα πράγματα αποτελούν τα δεδομένα που θα πρέπει να παρακολουθήσουμε σε αρχικό στάδιο.

Ο τρόπος με τον οποίο θα παρακολουθούνται αυτά τα δεδομένα είναι με τη χρήση αισθητήρων που θα τηρούν τις προδιαγραφές που τίθενται. Άρα, το δεύτερο βήμα ήταν ο προσδιορισμός των χαρακτηριστικών των αισθητήρων και των περιφερειακών εξαρτημάτων που είναι απαραίτητα και η έρευνα αγοράς τους.

Έχοντας υπόψιν τα εμπόδια που αναγνωρίσαμε και τα δεδομένα που θέλουμε να ερευνήσουμε, κατέληξα στην κατασκευή ενός IoT συστήματος το οποίο θα έχει ένα σύνολο από αισθητήρες που θα βρίσκονται στην περιοχή που θέλουμε να ελέγξουμε, οι οποίοι συλλέγουν τα δεδομένα και θα τα μεταδίδουν ασύρματα, μέσα από μια μεγάλη απόσταση στο χρήστη. Αυτός θα προειδοποιείται για την ύπαρξη μιας φωτιάς βάση ενός αλγόριθμου ο οποίος θα επεξεργάζεται τα δεδομένα και θα τα παρουσιάζει με απλό και ευανάγνωστο τρόπο.

Το σύστημα IoT που υλοποιήσαμε βασίζεται στην τεχνολογία LoRa και αποτελείται από δύο μέρη. Το σταθμό συλλογής και αποστολής των δεδομένων, κόμβος LoRa Transmitter και το σταθμό λήψης δεδομένων κόμβος - δέκτη LoRa Receiver. Η τεχνολογία LoRa προσφέρει μεγάλη εμβέλεια μετάδοσης δεδομένων σε μεγάλες αποστάσεις, χαμηλή κατανάλωση ισχύος, και προστασία από παρεμβολές. Άρα είναι ιδανική γιατί μας δίνει την δυνατότητα να τοποθετήσουμε κόμβους σε δύσβατες περιοχές που δεν μπορεί να πάει ευκολά κάποιος, ενώ ταυτόχρονα να προσφέρει απομακρυσμένη επικοινωνία του χρήστη με την περιοχή που

περιτριγυρίζει τον κόμβο LoRa Transmitter. Τα δεδομένα που θα συλλέγει ο κόμβος LoRa Transmitter θα μεταδίδονται μέσω ενός LoRa module στο κόμβο - δέκτη LoRa Receiver ο οποίος είναι υπεύθυνος για την αποστολή τους μέσα από ένα router στο διαδίκτυο.

Στόχος είναι να προειδοποιήσω έναν χρήστη για την ύπαρξη μιας πυρκαγιάς, πρέπει να κατασκευάσω και να χρησιμοποιήσω μια πλατφόρμα που θα επιτρέπει να αποθηκεύουμε, επεξεργαστούμε και να αναλύσουμε τα δεδομένα. Σε αυτή την κατασκευή επέλεξα μια υπάρχουσα πλατφόρμα για την ανάλυση, αποθήκευση και επεξεργασία δεδομένων που ονομάζεται ThingSpeak και λόγο του ότι έχουμε δωρεάν χρήση. Η πλατφόρμα αυτή είναι μια IoT cloud πλατφόρμα που μας δίνει την δυνατότητα να αποθηκεύσουμε, να επεξεργαστούμε και να αναλύσουμε ροές δεδομένων με μεγάλη ευκολία. Τα δεδομένα που στέλνει ο κόμβος - δέκτης LoRa Receiver μέσα από ένα router στο διαδίκτυο πηγαίνουν στο ThingSpeak και από εκεί κάποιος χρήστης θα μπορεί να ενημερώνεται για το περιβάλλον που περιτριγυρίζει τον κόμβο LoRa Transmitter και να προειδοποιείται για την ύπαρξη φωτιάς.

Έχει επιτευχθεί η κατασκευή μιας βασικής δομής που επιτρέπει την επιτήρηση ενός σημείου του δάσους για την ύπαρξη πυρκαγιάς. Με τον όρο βασική δομή εννοούμε ότι η κατασκευή έχει την δυνατότητα για μελλοντικές επεκτάσεις και κυρίως με την προσθήκη περισσότερων κόμβων.

5.1 Προτάσεις Μελλοντικής Επέκτασης και βελτίωσης

Μέσα από την διαδικασία εκπόνησης της παρούσας διπλωματικής εργασίας για την κατασκευή μιας πλατφόρμας έγκαιρης προειδοποίησης δασικών πυρκαγιών βασισμένη σε τεχνολογία IoT, η πλατφόρμα αυτή έχει την δυνατότητα να εξελιχθεί σε πολύ μεγάλο επίπεδο. Οι περιορισμοί που εμφανίστηκαν στην έρευνα και τη κατασκευή της, ανέπτυξαν ιδέες για την πιθανή βελτιστοποίηση και επέκτασης της. Μερικές προτάσεις για την βελτιστοποίηση της πλατφόρμας αυτής αναφέρονται παρακάτω:

- Σύστημα τροφοδοσίας με Φωτοβολταϊκό για χρήση σε μεγάλο χρονικό διάστημα.
- Προσθήκη και άλλων αισθητήρων όπως είναι αισθητήρας υπέρυθρης ακτινοβολίας για την ανίχνευση του υπέρυθρου φάσματος που εκπέμπεται κατά την καύση, ανεμόμετρο για την μέτρηση της ταχύτητας του αέρα η οποία συμβάλει στην επέκταση του πύρινου μετώπου κτλ.
- GPS γεωγραφική τοποθέτηση τον κόμβων της πλατφόρμας έτσι ώστε ο χρήστης να μπορεί να ενημερώσει τις ειδικές πυροσβεστικές δυνάμεις με ακρίβεια για την τοποθεσία στην περιοχή όπου ανιχνεύθηκε φωτιά με ακριβείς συντεταγμένες [13].
- Ανάπτυξη περισσότερων κόμβων για την κάλυψη μιας μεγαλύτερης δασικής έκτασης. Αυτό θα επιτευχθεί με την ανάπτυξη των κόμβων σε τοπολογία αστέρα, χρησιμοποιώντας

πρωτόκολλο επικοινωνίας που μπορώ να αναπτύξω ή να χρησιμοποιηθεί το υπάρχων πρωτόκολλο LoRaWAN [14].

- Σύστημα πρόληψης για τα φαινόμενα που θα συμβάλουν στην δημιουργία μιας πυρκαγιάς όπως ενσωμάτωση μετεωρολογικού συστήματος που θα προβλέπει την θερμοκρασία, την υγρασία, την ταχύτητα του ανέμου και το ποσοστό ηλιοφάνειας για της επόμενες μέρες που προβλέπει ο Παγκόσμιος Μετεωρολογικός Οργανισμός.
- Αγορά του standard license για την χρήση όλων των διαθέσιμων πόρων που προσφέρει η πλατφόρμα ThingSpeak, όπως την ικανότητα αποστολής έως και 90.000 μηνυμάτων την ημέρα, ανανέωση των δεδομένων και αποστολή μηνυμάτων κάθε δευτερόλεπτο, δημιουργία μέχρι και 250 κανάλια, χρήση φωτογραφιών για την απεικόνιση δεδομένων ή την προβολή ενός χάρτη.
- Χρήση ισχυρότερης κεραίας για την επέκταση της εμβέλειας επικοινωνίας μεταξύ των κόμβων π.χ.: Yagi κεραία ή παραβολική κεραία.

Παράρτημα Α

Κώδικας Κόμβου – LoRa transmitter (πρώτη προσπάθεια)

```
#include <SPI.h> // βιβλιοθήκη επικοινωνίας με τα περιφερειακά εξαρτήματα
#include <LoRa.h> // βιβλιοθήκη της επικοινωνίας Lora
#include <dht.h> // Βιβλιοθήκη του αισθητήρα DHT11
#include "MQ135.h" // Βιβλιοθήκη του αισθητήρα MG135
#define DHTpin A5     // Καθορίζω το pin του Arduino απο όπου διαβάζω τον
//αισθητήρα Θερμοκρασίας DHT11
dht DHT11;         //Καθορίζω τον τύπο του αισθητήρα που έχω σύμφωνα με την
//βιβλιοθήκη, στην περίπτωση μας ο DHT11
#define MQ7pin A3  // Καθορίζω το pin του Arduino απο όπου διαβάζω τον
//αισθητήρα μονοξειδίου του άνθρακα
#define MQ135pin A0 //Καθορίζω το pin του Arduino απο όπου διαβάζω τον αισθητήρα διοξειδίου
//του άνθρακα
                //------------------  Variables  ----------------------//
float hum;
float temp;
int MQ7Value;
int MQ135Value;
//-------------------------------------------------------------------------------------------//
void setup()
{
  Serial.begin(9600); // Καθορίζω την επικοινωνία του υπολογιστή με το arduino ώστε
//να έχουν τον ίδιο ρυθμό μετάδοσης (baud rate)
// εντολή επικοινωνίας με το arduino, καθορίζει τον ρυθμό επικοινωνίας 9600
```

```
 while (!Serial);
  Serial.println("LoRa Transmitter");  // Υποδεικνύω το κόμβο ως "Αποστολέα"  στην οθόνη μου
//στην περίπτωση που το βλέπω

  if (!LoRa.begin(433E6))  // καθορίζω την συχνότητα επικοινωνίας του module και την ελέγχω
    {
      Serial.println("Starting LoRa failed!"); // Στη περίπτωση που αποτύχει η επικοινωνία με τον
//κόμβο τότε εκτύπωσε: Starting LoRa failed!
      while (1); // επανάλαβε τον έλεγχο επικοινωνίας μέχρι να βρεθεί σφάλμα, το έβαλα για να
//ολοκληρώσω τον συνεχή έλεγχο επικοινωνίας
      // παράδειγμα επικοινωνίας με LoRa από την βιβλιοθήκη LoRa.h
    }
 } //Τέλος του "setup()"
//-------------------------------------------------------------------------------------------//
void loop()
{
  //Αρχή του προγράμματος μου
 Serial.println("Sending packet: "); //Υποδηλώνω την αποστολή δεδομένων από τον //κόμβο

   DHT11.read11(DHTpin); //Το Arduino διαβάζει από τον αισθητήρα DHT11 τα //δεδομένα της
θερμοκρασίας και της υγρασίας σύμφωνα με τα πρωτόκολλα τη
//βιβλιοθήκη dht.h
     LoRa.beginPacket(); // Αρχή της διαδικασίας αποστολής των πακέτων-δεδομένων
     LoRa.print("Temperature = ");
     LoRa.print(DHT11.temperature); //Θερμοκρασία
     LoRa.print("C  ,  ");
     LoRa.print("Humidity = "); //Υγρασία
     LoRa.print(DHT11.humidity);
     LoRa.print("%  ,  ");  //
```

```
//   Serial.print("Temperature = ");  // Εδώ βλέπω στο δικό μου κομμάτι του trasmitter τι στέλνω ( δεν θα υπάρχει αυτό στο τέλος)
  //  Serial.print(DHT11.temperature); //Θερμοκρασία
    // Serial.print("C  ,  ");
     //Serial.print("Humidity = "); //Υγρασία
     //Serial.print(DHT11.humidity);
     //Serial.print("%  ,  ");

 // _*Αισθητήρας μονοξείδιου του άνθρακα *_ //

   MQ7Value= analogRead(MQ7pin);// To Arduino διαβάζει την αναλογική τιμή από
//την έξοδο του αισθητήρα και την αποδίδει στην ακέραια τιμή MQ7Value
      LoRa.print("CO value: ");
      LoRa.print(MQ7Value);// Εμφάνισε την τιμή μονοξείδιου του άνθρακα
      LoRa.print("ppm  ,  ");

 //  Serial.print("CO value: ");
   // Serial.print(MQ7Value);// Εμφάνισε την τιμή μονοξείδιου του άνθρακα
   // Serial.print("ppm  ,  ");

// _*Αισθητήρας διοξειδίου του άνθρακα *_ //

   MQ135Value = analogRead(A0); // To Arduino διαβάζει την αναλογική τιμή από
//την έξοδο του αισθητήρα και την αποδίδει στην ακέραια τιμή MQ135Value
      LoRa.print("CO2 value: ");
      LoRa.print(MQ135Value, DEC);  // Εμφάνισε την τιμή διοξειδίου του άνθρακα
      LoRa.println("ppm ");
```

```
  //Serial.print("CO2 value: ");
   // Serial.print(MQ135Value, DEC); // Εμφάνισε την τιμή διοξειδίου του άνθρακα
    //Serial.println("ppm ");

    LoRa.endPacket(); // Τέλος της διαδικασίας αποστολής των πακέτων-δεδομένων
    delay(5000);// Περίμενε 10 δευτερόλεπτα μέχρι την επόμενη μέτρηση

}// Τέλος του loop
```

Παράρτημα Β

Κώδικας Δέκτη – LoRa Receiver (πρώτη προσπάθεια)

```
#include <SPI.h> // βιβλιοθήκη επικοινωνίας με τα περιφερειακά εξαρτήματα
#include <LoRa.h> // βιβλιοθήκη της επικοινωνίας Lora
void setup()
{
  Serial.begin(9600); // Καθορίζω την επικοινωνία του υπολογιστή με το arduino ώστε να έχουν τον ίδιο
//ρυθμό μετάδοσης (baud rate)
// εντολή επικοινωνίας με το arduino, καθορίζει τον ρυθμό επικοινωνίας 9600
while (!Serial);

  Serial.println("LoRa Receiver");  // Υποδεικνύω το Gateway ως "Παραλήπτης"
  if (!LoRa.begin(433E6))  // καθορίζω την συχνότητα επικοινωνίας του module και την
//ελέγχω
    {
      Serial.println("Starting LoRa failed!");
      while (1);
    }
} //Τέλος του "setup()"

void loop()
{
  //Αρχή του προγράμματος μου
  int packetSize = LoRa.parsePacket();   // Ανάλυση του πακέτου που δέχομαι (Note: το
//μέγεθος του πακέτου είναι στην παρένθεση και είναι πάντα ">0" αλλιώς αν έχω
//LoRa.parsePacket(0); τότε δεν έχει στείλει τίποτα ο transmitter και δεν έχουμε δεχθεί τίποτα
 if (packetSize)
  {
    // Το πακέτο ήρθε

    Serial.println("....................................."); //αφήνω κενό ανάμεσα στα πακέτα
```

```
    Serial.println("Received packet: ")
 while (LoRa.available())   // Ανάγνωση του πακέτου που ήρθε όταν το LoRa εκτελέστηκε με επιτυχία
{
     Serial.print((char)LoRa.read()); // Εμφάνισε στην οθόνη τα δεδομένα που παίρνουμε απο τον κόμβο με το LoRa.read σύμφωνα με την βιβλιοθήκη
    }
// εκτύπωσε στην οθόνη το RSSI του κάθε πακέτου, δλδ την ένταση του σήματος
//επικοινωνίας LoRa
    Serial.print(" with RSSI: ");
    Serial.println(LoRa.packetRssi());
  } // Τέλος του Δεύτερου loop
delay(500);
} // Τέλος του Πρώτου loop
```

Παράρτημα Γ

Κώδικας Κόμβου – LoRa transmitter (Τελική κατασκευή)

```
#include <DHT.h> // Βιβλιοθήκη του αισθητήρα DHT11
#include <DHT_U.h>

#include <SPI.h>    //Βιβλιοθήκη που χειρίζεται τις περιφερειακές συσκευές που συνδέονται στο aduino
#include <LoRa.h> // Βιβλιοθήκη του LoRa
#include "MQ135.h" // Βιβλιοθήκη του αισθητήρα MQ135
#define DHTPIN 4
#define MQ7pin A3  // Καθορίζω το pin του Arduino από όπου διαβάζω τον
//αισθητήρα μονοξειδίου του άνθρακα
#define MQ135pin A0 //Καθορίζω το pin του Arduino από όπου διαβάζω τον
//αισθητήρα διοξειδίου του άνθρακα

DHT dht(DHTPIN, DHT11); // Ορίζω τον τύπο του αισθητήρα θερμοκρασίας και
//υγρασίας (υπάρχουν δυο τύποι DHT11 και DHT12)



//------------------  Variables  ---------------------------------------------------------------//

float MQ7Value;

float MQ135Value;

String LoRaMessage = "";
```

```
char device_id[12] = "MyDevice123";

//--------------------------------------------------------------------------------------------//

void setup() {
  Serial.begin(9600);
  delay(10);
  dht.begin();

  while (!Serial);

  Serial.println("LoRa Sender");

  if (!LoRa.begin(433E6)) { // Η προκαθορισμένη συχνότητα για την λειτουργεία της
//τεχνολογίας LoRa στην Ευρώπη-Ελλάδα είναι 433.05 - 434.79 Mhz άρα 433E6
    Serial.println("Starting LoRa failed!");
    while (1);
  }

}

void loop() {

     float humidity = dht.readHumidity(); // float την μεταβλητή humidity όπου θα
//βάζω την τιμή της υγρασίας που διαβάζει ο αισθητήρας DHT11 σύμφωνα με τη
//βιβλιοθήκη του αισθητήρα
     float temperature = dht.readTemperature(); // float την μεταβλητή temperature
//όπου θα βάζω την τιμή της θερμοκρασίας που διαβάζει ο αισθητήρας DHT11
//σύμφωνα με την βιβλιοθήκη του αισθητήρα
```

```
    float MQ7Value = analogRead(MQ7pin); // float την μεταβλητή MQ7Value όπου
//θα βάζω την τιμή του διοξειδίου του άνθρακα που διαβάζει ο αισθητήρας MQ7
    float MQ135Value = analogRead(A0);  // float τη μεταβλητή MQ135Value όπου
//θα βάζω την τιμή του μονοξείδιου του άνθρακα που διαβάζει ο αισθητήρας MQ135


      Serial.print("Temperature = ");
      Serial.print(temperature); // εκτύπωση της θερμοκρασίας στο Serial Monitor για
//τον έλεγχο της ακρίβειας του αισθητήρα
      Serial.println("*C");


      Serial.print("Humidity = ");
      Serial.print(humidity); // εκτύπωση της υγρασία στο Serial Monitor για τον
//έλεγχο της ακρίβειας του αισθητήρα
      Serial.println("%");


      Serial.print("Carbon Monoxide (CO) = ");
      Serial.print(MQ7Value); // εκτύπωση του μονοξείδιου του άνθρακα στο Serial
//Monitor για τον έλεγχο της ακρίβειας του αισθητήρα
      Serial.println("ppm");


      Serial.print("CO2 = ");
      Serial.print(MQ135Value); // εκτύπωση του διοξειδιού του άνθρακα στο Serial
//Monitor για τον έλεγχο της ακρίβειας του αισθητήρα
      Serial.println("%");


      Serial.println();


        LoRaMessage = String(device_id) + "/" + String(temperature) + "&" + String(humidity)
```

```
        + "#" + String(MQ7Value) + "@" + String(MQ135Value) + "^" + String(1);
//δημιουργία string LoRaMessage όπου θα βάλω τις 4 τιμές από τους αισθητήρες, μαζί με το id του
//κόμβου
 // για να μπορέσω να τα μεταφέρω με ευκολία στο δέκτη από όπου θα τα μεταδώσω //στο Internet


  LoRa.beginPacket();  // εδώ δηλώνω την αρχή του πακέτου που θα μεταδώσει ο //κόμβος
  LoRa.print(LoRaMessage); //το πακέτο προς αποστολή, στην δική μου περίπτωση //το string
περιέχει τα δεδομένα του κόμβου
  LoRa.endPacket(); // εδώ δηλώνω το τέλος του πακέτου που θα μεταδώσει ο //κόμβος


  delay(1000);
}
```

Παράρτημα Δ

Κώδικας Δέκτη – LoRa Receiver (Τελική κατασκευή)

```
#include <SPI.h> // βιβλιοθήκη επικοινωνίας με τα περιφερειακά εξαρτήματα
#include <LoRa.h> // βιβλιοθήκη της επικοινωνίας Lora
//-----------------------WiFi ESP test conflict with lora----------------------------//
#include <ESP8266WiFi.h>
String apiKey = "********************";     //  Καθορίζω σε string το μοναδικό //κωδικό API key από την σελίδα του ThingSpeak
const char *ssid =  "************";     // Στις μεταβλητές αυτές καθορίζω το όνομα //και τον κωδικό του router μου και έπειτα το server που θα στείλω τα δεδομένα μου
const char *pass =  "************";
const char* server = "api.thingspeak.com";

WiFiClient client; // Ενεργοποιώ την σύνδεση ως ένα δίκτυο/client για να μπορώ να //στείλω δεδομένα
//-------------------------------------------------------------------------------------------------//
#define ss 15   // Ορίζω τα ποδαράκια (ss, rst, dio0) όπου το ESP8266 module θα //χρησιμοποιεί το LoRa module
#define rst 16
#define dio0 2


    String device_id;  // Καθορίζω τις μεταβλητές string που θα διαβάζω από το κάθε //πακέτο που στέλνει ο κόμβος
    String temperature;
    String humidity;
    String MQ7Value;
    String MQ135Value;
```

```
void setup()
{
  Serial.begin(115200);         // Καθορίζω την επικοινωνία του υπολογιστή με το //arduino ώστε να έχουν τον ίδιο ρυθμό μετάδοσης (baud rate)
   delay(10);

 Serial.println("LoRa Receiver");  // Υποδεικνύω το κόμβο ως δέκτης – LoRa Receiver, αυτή η γραμμή κώδικα υπάρχει μόνο για να κάνω έλεγχο της σωστής λειτουργείας του δέκτη από το μόνιτορ

 LoRa.setPins(ss, rst, dio0);  // Καθορίζω τα ποδαράκια (ss, rst, dio0) του NodeMCU //ESP8266 που θα χρησιμοποιεί το LoRa module
 if (!LoRa.begin(433E6)){  // Καθορίζω την συχνότητα επικοινωνίας του LoRa module και την ελέγχω
     Serial.println("Starting LoRa failed!");
     while (1);
   }


//----------------------------------------WiFi ESP test conflict with lora--------------------//

  // Αρχή κώδικα για την σύνδεση του ESP8266 module στο internet
   Serial.println("Connecting to ");
   Serial.println(ssid);
   WiFi.begin(ssid, pass);

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

```
    Serial.println("");
    Serial.println("WiFi connected..!");
    Serial.print("Got IP: ");
    Serial.println(WiFi.localIP());

  // Τέλος κώδικα για τη σύνδεση του ESP8266 module στο internet
//-------------------------------------------------------------------------------------------//


} //Τέλος του "setup()"

// Στον σημείο αυτό έχω τον κώδικα για την λείψει και αποκωδικοποίηση του LoRa //πακέτου που θα έρθει από τον κόμβο (LoRa Transmitter) και έπειτα αποστολής των //δεδομένων στην ιστοσελίδα ThingSpeak

void loop(){

  int pos1, pos2, pos3, pos4, pos5; // Ορίζω μεταβλητές που θα χρησιμοποιήσω για να //ξεχωρίσω τα //δεδομένα που παίρνω από το πακέτο που θα έρθει στο LoRa module

  int packetSize = LoRa.parsePacket(); // Έλεγχος αν έχει ληφθεί πακέτο LoRa
  if (packetSize)
  {
    // Λήψη πακέτου LoRa
    Serial.print("Received packet:  ");
    String LoRaData = LoRa.readString();
    Serial.print(LoRaData);

    // Ανάγνωση του πακέτου που λήφθηκε
    while (LoRa.available())
```

```
    {
      Serial.print((char)LoRa.read());
    }
    // Τύπωσε το RSSI της σύνδεσης LoRa, όπου RSSI η δύναμη του σήματος //επικοινωνίας, όσο πιο
//κοντά στο μηδέν τόσο πιο δυνατό το σήμα
    Serial.print("' with RSSI ");
    Serial.println(LoRa.packetRssi());

    // Ορίζω τα όρια από όπου θα ξεχωρίσω το κάθε δεδομένο μέσα στην σειρά από  //χαρακτήρες
του κάθε πακέτο, που θα δέχεται ο LoRa receiver
    pos1 = LoRaData.indexOf('/');
    pos2 = LoRaData.indexOf('&');
    pos3 = LoRaData.indexOf('#');
    pos4 = LoRaData.indexOf('@');
   pos5 = LoRaData.indexOf('^');

    // Ξεχωρίζω τα πέντε δεδομένα που έρχονται στο κάθε πακέτο με βάση την θέση //τους, μεταξύ
των ορίων πού έθεσα παραπάνω
    // το κάθε δεδομένο διαχωρίζεται από τις θέσεις (pos1, pos2, pos3, pos4,pos5) ξεκινώντας από
//τον πρώτο αλφαριθμητικό χαρακτήρα στο string, μέχρι να φτάσει το pos5
    device_id = LoRaData.substring(0, pos1);
    temperature = LoRaData.substring(pos1 + 1, pos2);
    humidity = LoRaData.substring(pos2 + 1, pos3);
    MQ7Value = LoRaData.substring(pos3 + 1, pos4);
    MQ135Value = LoRaData.substring(pos4 + 1, pos5);


    Serial.print(F("Device ID = "));
    Serial.println(device_id);
```

```
    Serial.print(F("Temperature = "));
    Serial.print(temperature);
    Serial.println(F("*C"));

    Serial.print(F("Humidity = "));
    Serial.print(humidity);
    Serial.println(F("%"));

    Serial.print(F("Carbon Monoxide (CO) = "));
    Serial.print(MQ7Value);
    Serial.println(F("ppm"));

    Serial.print(F("Air Quality (MQ135) = "));
    Serial.print(MQ135Value);
    Serial.println(F("ppm"));



    Serial.println();

    if (client.connect(server, 80)) // Αν έχει γίνει σύνδεση στο server //(api.thingspeak.com) που θέλω να στείλω τα δεδομένα μέσα από την πόρτα 80
    {
      // Καθορίζω σε ποιο field στο δικό μου κανάλι θα μπουν τα δεδομένα μου
      String postStr = apiKey;
      postStr += "&field1=";
      postStr += String(temperature);
      postStr += "&field2=";
      postStr += String(humidity);
      postStr += "&field3=";
```

```
    postStr += String(MQ7Value);
    postStr += "&field4=";
    postStr += String(MQ135Value);
    postStr += "r\n";

    // Σύνδεση και αποστολή το δεδομένων στο ThingSpeak
    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(postStr.length());
    client.print("\n\n");
    client.print(postStr);

    Serial.println("Data Send to Thingspeak");
    delay(500);
  }
  client.stop();
  Serial.println("Waiting...");

 }

} // Τέλος του Πρώτου loop
```

Παράρτημα Ε

Κώδικας της εφαρμογής Matlab app για την αποστολή μηνύματος προειδοποίησης

```
channelID = *******;
alertApiKey = '***************';

alertUrl="https://api.thingspeak.com/alerts/send";
options = weboptions("HeaderFields", ["ThingSpeak-Alerts-API-Key", alertApiKey ]);

temperature = thingSpeakRead(channelID,'Fields',1);

humidity = thingSpeakRead(channelID,'Fields',2);
COvalue = thingSpeakRead(channelID,'Fields',3);
CO2value = thingSpeakRead(channelID,'Fields',4);

alertSubject = sprintf("*** ΚΙΝΔΥΝΟΣ - ΦΩΤΙΑ ***");

alertBody = sprintf(" Οι ενδείξεις έχουν υπερβεί τα όρια ασφαλείας!");

%if temperature > 10  % γραμμή προγράμματος για επίδειξη

 if temperature > 55 & humidity < 22 & COvalue > 1000 & CO2value > 1000

 webwrite(alertUrl , "body", alertBody, "subject", alertSubject, options);

 end
```

Παράρτημα ΣΤ

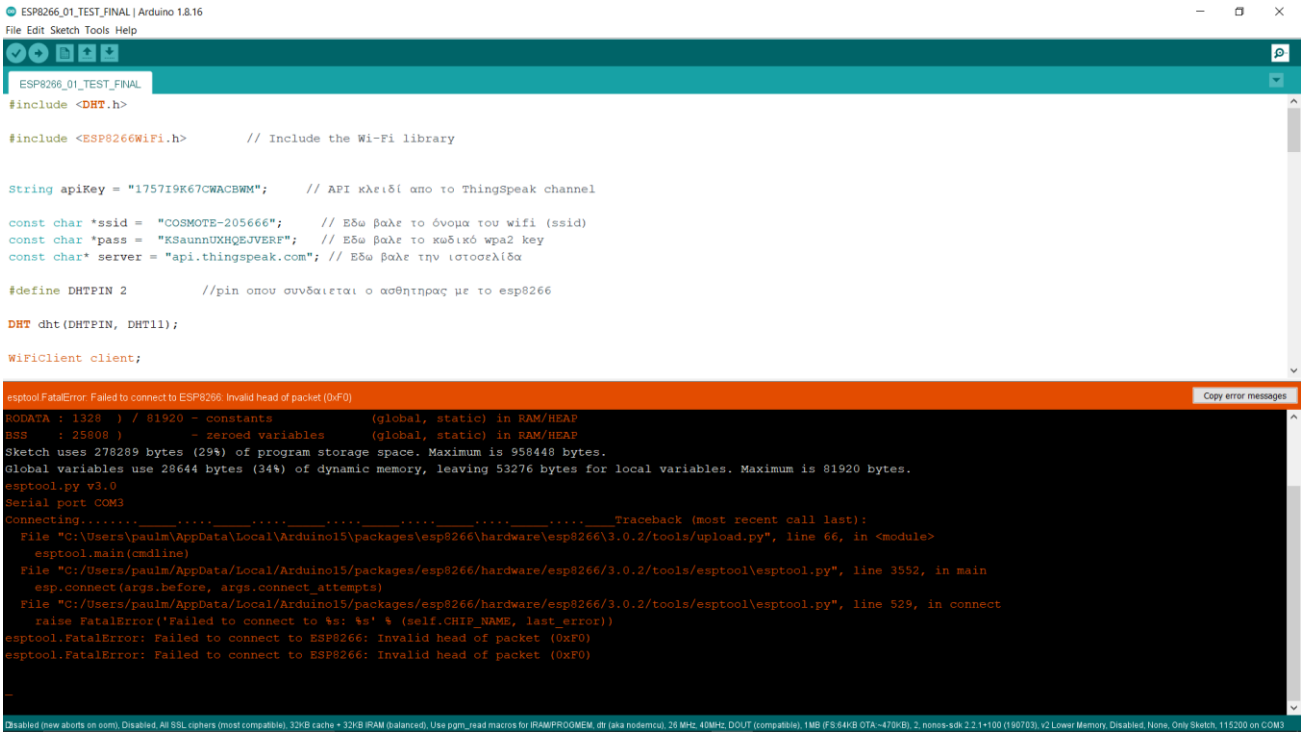
Κώδικας Κόμβου – δέκτης LoRa receiver (πρώτη προσπάθεια) χρησιμοποιώντας ESP8266 – ESP 01 για την αποστολή δεδομένων

Σε αυτό τον κώδικα έγινε η προσπάθεια να σταλθούν μόνο οι τιμές θερμοκρασίας και υγρασίας με σκοπό να εξασφαλιστεί η επικοινωνία μεταξύ του κόμβου – δέκτη LoRa Receiver με την πλατφόρμα ThingSpeak χρησιμοποιώντας AT commands. Μετά από πολλές προσπάθειες αλλαγής του κώδικα και των βιβλιοθηκών δεν επιτεύχθηκε η σύνδεση.

```
#include <DHT.h>
#include <ESP8266WiFi.h>        // Include the Wi-Fi library
String apiKey = "****************";     // API κλειδί απο το ThingSpeak channel
const char *ssid =  "***************";     // Εδώ βάλε το όνομα του wifi (ssid)
const char *pass =  "******************";   // Εδώ βάλε τον κωδικό wpa2 key
const char* server = "api.thingspeak.com"; // Εδώ βάλε την ιστοσελίδα
#define DHTPIN 2          //pin όπου συνδέεται ο ασθητήρας με το esp8266
DHT dht(DHTPIN, DHT11);
WiFiClient client;
void setup()
{
  Serial.begin(115200);
  delay(10);
  Serial.println('\n');

  WiFi.begin(ssid, pass);             // Εδώ γίνεται η έναρξη σύνδεσης στο WiFi
  Serial.print("Connecting to ");
  Serial.print(ssid);
  Serial.println(" ...");
```

```
    while (WiFi.status() != WL_CONNECTED)    // Αναμονή σύνδεσης στο WiFi
    {
        delay(1000);
        Serial.print("*");
    }
    Serial.println('\n');
    Serial.println("Connection established!");
    Serial.print("IP address:\t");
    Serial.println(WiFi.localIP());         // Αποστολή της IP address του ESP8266 στο υπολογιστή
}
void loop()
{
    float h = dht.readHumidity();
    float t = dht.readTemperature();
 Serial.print("test");
                  if (client.connect(server,80))   //   "184.106.153.149" or api.thingspeak.com
                {

                    String postStr = apiKey;
                    postStr +="&field1=";
                    postStr += String(h);
                    postStr +="&field2=";
                    postStr += String(h);
                    postStr += "\r\n\r\n";

                    client.print("POST /update HTTP/1.1\n");
                    client.print("Host: api.thingspeak.com\n");
                    client.print("Connection: close\n");
                    client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
```

```
                    client.print("Content-Type: application/x-www-form-urlencoded\n");
                    client.print("Content-Length: ");
                    client.print(postStr.length());
                    client.print("\n\n");
                    client.print(postStr);
                     Serial.print("Number: ");
                     Serial.print(y);
                    Serial.print("Temperature: ");
                    Serial.print(t);
                    Serial.print(" degrees Celcius, Humidity: ");
                    Serial.print(h);
                    Serial.println("%. Send to Thingspeak.");
                }
        client.stop();
        Serial.println("Waiting...");
  delay(5000);
}
//Τέλος προγράμματος
```



Εικόνα 60: ***Πρόβλημα αποστολής δεδομένων στην πλατφόρμα ThingSpeak με τον πρωτότυπο κώδικα***

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] San-Miguel-Ayanz, J. (14 Μαΐου, 2004). The European Forest Fire Information System (EFFIS). Joint Research Centre. European Commission. https://ec.europa.eu/environment/forests/pdf/meeting140504_presjrc.pdf

[2] Turco, M., Rosa-Cánovas J.J., Bedia J., Jerez S., Montávez, J.P., Llasat, M.C., Antonello Provenzale, A. (02 Οκτωβρίου, 2018). "Exacerbated fires in Mediterranean Europe due to anthropogenic warming projected with non-stationary climate-fire models"**.** Nature Communications. https://www.nature.com/articles/s41467-018-06358-z

[3] Τζιανίδης, Ν. (23 Ιουλίου, 2021). «Φωτιά στο Μάτι-Τρία χρόνια: Οι νεκροί περισσότεροι και πυρόπληκτοι στο δρόμο». Έθνος. https://www.ethnos.gr/greece/article/167213/fotiastomatitriaxroniaoinekroiperissoteroikaipyroplhktoistodromo

[4] "Wind gusts from automatic meteorological stations". (23 Ιουλίου, 2018). Meteo. Ανακτήθηκε από: https://meteo.gr/UploadedFiles/articlePhotos/JUL18/MaxGusts_Attica_23072018.png

[5] «Αποκαλύψεις για τη φωτιά στο Μάτι: Καθυστερήσεις και παραλήψεις της Πυροσβεστικής – Η αιτία της τραγωδίας». (25 Ιουλίου, 2020). In.gr. https://www.in.gr/2020/07/25/greece/apokalypseis-gia-ti-fotia-sto-mati-kathysteriseis-kai-paralipseis-tis-pyrosvestikis-aitia-tis-tragodias/

[6] Καλαφάτης, Α. (14 Σεπτεμβρίου, 2018). «Αποκλειστικό θέμα HuffPost Greece: Από καλώδια της ΔΕΗ η φωτιά στην Κινέτα». HuffPost Greece.

https://www.huffingtonpost.gr/entry/apokleistiko-thema-huffpost-greece-apo-kalodia-tes-dee-e-fotia-sten-kineta_gr_5b9b7df5e4b046313fb9e775

[7][11] *What are LoRa and LoRaWAN?*.TheThingsNetwork. Ανακτήθηκε από: https://www.thethingsnetwork.org/docs/lorawan/what-is-lorawan/

[8] "What Is LoRa®?". *LoRa PHY*. Semtech. Ανακτήθηκε από: https://www.semtech.com/lora/what-is-lora

[9] LoRa Alliance Technical Committee Regional Parameters Workgroup. (05 Μαΐου, 2021). "LoRa Alliance, RP002-1.0.3 LoRaWAN® Regional Parameters". LoRa Alliance. https://lora-alliance.org/wp-content/uploads/2021/05/RP002-1.0.3-FINAL-1.pdf

[10] *Learn More About ThingSpeak.* ThingSpeak. Ανακτήθηκε από: https://thingspeak.com/pages/learn_more

[12] ThingSpeak™ Licensing FAQ. Ανακτήθηκε από: https://thingspeak.com/pages/license_faq

[13] Ahmad A. A. Alkhatib. (5 Μαρτίου 2014). «A Review on Forest Fire Detection Techniques». *https://journals.sagepub.com/doi/pdf/10.1155/2014/597368*

[14] What is LoRaWAN® Specification. Ανακτήθηκε από: https://lora-alliance.org/about-lorawan/

Παρακάτω παρατίθενται οι εικόνες που πάρθηκαν από εξωτερικές πηγές:

Εικόνα 1: Διαδίκτυο των πραγμάτων (Internet of Thinks – IoT). Ανακτήθηκε από: https://schoolpress.sch.gr/3lykmagazine/2019/05/%CF%84%CE%BF-%CE%B4%CE%B9%CE%B1%CE%B4%CE%AF%CE%BA%CF%84%CF%85%CE%BF-%CF%84%CF%89%CE%BD-%CF%80%CF%81%CE%B1%CE%B3%CE%BC%CE%AC%CF%84%CF%89%CE%BD-internet-of-things/

Εικόνα 2: Σήμα διαμορφωμένο με την τεχνική chirp spread spectrum (CSS). Ανακτήθηκε από: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/

Εικόνα 3: Πλεονεκτήματα χρήσης της τεχνολογίας LoRa. Ανακτήθηκε από: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/

Εικόνα 7: DHT11 Digital Temperature and Humidity Sensor. Ανακτήθηκε από: https://grobotronics.com/humidity-sensor-dht11-module.html?sl=en

Εικόνα 8: MQ-7 Carbon Monoxide CO Gas Sensor For Arduino (MQ7). Ανακτήθηκε από: https://topelectronics.gr/electronics/sensors/gas/mq-7-carbon-monoxide-co-gas-sensor-for-arduino-mq7/

Εικόνα 9: MQ-135 Air Quality Sensor – Hazardous Gas Detection Module For Arduino (MQ135). Ανακτήθηκε από: https://topelectronics.gr/electronics/sensors/gas/mq-135-air-quality-sensor-hazardous-gas-detection-module-for-arduino-mq135/

Εικόνα 10: Arduino UNO R3 Atmega328P Board with USB Cable (DIP Version)(OEM). Ανακτήθηκε από: https://topelectronics.gr/arduino/compatible-boards/arduino-uno-r3-atmega328p-board-with-usb-cable-dip-version/

Εικόνα 11: SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz. Ανακτήθηκε από: https://topelectronics.gr/iot/sx1278-lora-ra-02-ai-thinker-module-433mhz/

Εικόνα 12: Antenna RF 434MHz 2dBi 63.6mm u.FL. Ανακτήθηκε από: https://grobotronics.com/antenna-rf-434mhz-2dbi-63.6mm-u.fl.html

Εικόνα 13: Intenso PowerBank 5200 mAh. Ανακτήθηκε από: https://www.e-jumbo.gr/eidi-technologias/kalodia-fortistes-kiniton-tablet/powerbanks/power-bank-intenso-5200mah-mavro_1434788/

Εικόνα 16: Arduino UNO R3 Atmega328P Board with USB Cable (DIP Version)(OEM). Ανακτήθηκε από: https://topelectronics.gr/arduino/compatible-boards/arduino-uno-r3-atmega328p-board-with-usb-cable-dip-version/

Εικόνα 17: SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz. Ανακτήθηκε από: https://topelectronics.gr/iot/sx1278-lora-ra-02-ai-thinker-module-433mhz/

Εικόνα 18: Antenna RF 434MHz 2dBi 63.6mm u.FL. Ανακτήθηκε από: https://grobotronics.com/antenna-rf-434mhz-2dbi-63.6mm-u.fl.html

Εικόνα 20: ESP 8266 ESP-01 WiFi Module. Ανακτήθηκε από: https://www.devobox.com/el/wifi/717-esp-8266-esp-01-wifi-module-gr.html?search_query=esp8266&results=5

Εικόνα 25: ESP8266 ESP-12E NodeMcu V3 Lua Board CH340 WIFI IoT. Ανακτήθηκε από: https://www.devobox.com/el/espressif/642-esp8266-esp-12e-nodemcu-v3-lua-board-ch340-wifi-iot.html

Εικόνα 26: SX1278 LoRa Ra-02 Ai-Thinker Module 433Mhz. Ανακτήθηκε από: https://topelectronics.gr/iot/sx1278-lora-ra-02-ai-thinker-module-433mhz/

Εικόνα 27: Antenna RF 434MHz 2dBi 63.6mm u.FL. Ανακτήθηκε από: https://grobotronics.com/antenna-rf-434mhz-2dbi-63.6mm-u.fl.html