INSTITUTO
UNIVERSITÁRIO
DE LISBOA

CityIOT – Air Quality in the city supported by an IoT ecosystem

Gonçalo Franco Morgado

Master in Telecommunications and Computer Engineering

Supervisor:
Octavian Adrian Postolache, Associate Professor with aggregation
ISCTE - Instituto Universitário de Lisboa

Co-Supervisor:
José Miguel Costa Dias Pereira, Coordinating Professor
IPS - Instituto Politécnico de Setúbal

December, 2023

TECNOLOGIAS
E ARQUITETURA

DCTI - Department of Information Sciences and Technologies

CityIOT - Air Quality in the city supported by an IoT ecosystem

Gonçalo Franco Morgado

Master in Telecommunications and Computer Engineering

Supervisor:
Octavian Adrian Postolache, Associate Professor with aggregation
ISCTE - Instituto Universitário de Lisboa

Co-Supervisor:
José Miguel Costa Dias Pereira, Coordinating Professor
IPS - Instituto Politécnico de Setúbal

December, 2023

Acknowledgements

I would like to start by expressing my gratitude to Professor Octavian Postolache and Prof. José Miguel Dias Pereira who acted as supervisors and guided me throughout the development of this dissertation. I would also like to thank him for the time he gave me to clarify and exchange truly relevant opinions about the project.

A special thank you to invited assistant Bruno Mataloto, who despite not acting as supervisor or co-supervisor, was an important support in this project. His ideas and opinions on certain topics helped me a lot in the development of this project and in clarifying both theoretical and practical aspects.

Special acknowledgements go to Instituto de Telecomunicações and ISCTE – Instituto Universitátio de Lisboa, that supported the present research associated with my master thesis.

I would like to thank all my friends, both outside and inside the university. They were the primary motivation for all the effort devoted to this project. In the not-so-good moments, they were always by my side to support and encourage me, and they never gave up on me. They are the ones who make me smile every day and would like to dedicate this dissertation mainly to them.

Lastly, a heartfelt thank you to my family, especially my mother and father. All the sacrifices they've made for me means a lot and is something I'll never forget, and I hope one day I'll be able to return. They were always the light at the end of the tunnel to guide me through challenging times and they have taught me to never give up, no matter how difficult the path is, if we walk together, anything is possible. Thank you for everything.

Resumo

A poluição do ar representa um problema global, que afeta milhões de pessoas, podendo trazer graves consequências para a saúde pública, para a economia e também sociais. A constante emissão de poluentes em estado gasoso, resultante de operações industriais, combustão de combustíveis fósseis e fogos florestais, afetam a qualidade do ar e contribuem para o aumento deste tipo de poluição. A Internet das Coisas (IoT) emergiu como resposta para a criação de sistemas inteligentes de monitorização que podem ser usados para a gestão e identificação de fontes de poluição e para a proteção da saúde pública. O desenvolvimento de novos equipamentos e tecnologias de comunicação, têm vindo a permitir a criação de aplicações de monitorização da qualidade do ar, em tempo real. Esta dissertação apresenta uma implementação de um sistema IoT de monitorização de qualidade do ar exterior, baseado num sistema multisensor com capacidade de comunicação LoRa. O sistema incorpora sensores de baixo custo capazes de detetar diferentes poluentes atmosféricos, como também parâmetros atmosféricos, como temperatura e humidade relativa. De modo a ser obtida uma transmissão de leituras em longa distância e com baixo consumo energético, o uso da tecnologia LoRa foi adotado. A frequência do envio de leituras para a rede, é feita com base no nível de tráfego urbano, numa certa localização. Ainda é apresentada uma aplicação web/móvel, que permite ao utilizador acompanhar em tempo real e proceder a uma análise temporal, das medições efetuadas pelo sistema.

Palavras-chave: Sensores inteligentes, Internet das Coisas (IdC), Qualidade do ar exterior, LoRaWAN, Monitorização inteligente, Aplicação web/móvel

Abstract

Air pollution is a worldwide problem, which affects millions of people and can have serious consequences for public health, economy and for social issues. Constant emissions of gaseous pollutants, as a result of industrial operations, combustion of fossil fuels and forest fires, affect air quality and contribute to an increase in this type of pollution. Internet of Things (IoT) has emerged as a response to the creation of intelligent monitoring systems that can be used for management and identification of pollution sources, and for health protection. The development of new equipment and communication technologies has allowed the creation of real-time air quality monitoring applications. This dissertation presents an implementation of an IoT system for monitoring outdoor air quality, based on a multisensory system with LoRa communication capabilities. The system incorporates low-cost sensors capable of detecting different air pollutants, as well as atmospheric parameters, such as temperature and relative humidity. To achieve a long-distance transmission of readings with low energy consumption, the use of LoRa technology was adopted. The frequency of sending readings to the network is based on the level of urban traffic in a certain location. Furthermore, a web/mobile application is presented, which allows the user to monitor in real time and to carry out a temporal analysis of the measurements taken by the system.

Keywords: Smart sensors, Internet of Things (IoT), Outdoor air quality, LoRaWAN, Smart monitoring, Web/Mobile app

Contents

List of Tables

List of Figures

List of Acronyms

3GPP - Third-Generation Partnership Project

API - Application Programming Interface

AQI - Air Quality Index

ADC - Analog to Digital Converter

ABP - Activation By Personalization

ADR – Adaptive Data Rate

BLE - Bluetooth Low Energy

BDS - Beidou Satellite Navigation System

CSS - Chirp Spread Spectrum

CSMA/CA - Carrier Sense Multiple Access with Collision Avoidance

CPU - Central Processing Unit

DSRM - Design Science Research Methodology

DSP - Digital Signal Processor

EEA - European Environment Agency

EPA - Environmental Protection Agency

EEPROM - Electrically Erasable Programmable Read-only Memory

FSK - Frequency Shift Keying

GFSK - Gaussian Frequency Shift Keying

GPRS - General Packet Radio Service

GPS - Global Positioning System

GLONASS - Global Navigation Satellite System

IoT - Internet of Things

ISM - Industrial, Scientific and Medical

IaaS - Infrastructure-as-a-Service

I2C - Inter-Integrated Circuit

IDE - Integrated Development Environment

IEEE - Institute of Electrical and Electronics Engineers

LoRa - Long Range

LoRaWAN - Long Range Wide Area Network

LPWAN - Low Power Wide Area Network

LTE - Long Term Evolution

LPG - Liquefied Petroleum Gas

MQTT - Message Queuing Telemetry Transport

MOS - Metal Oxide Semi-Conductors

MAC - Medium Access Control

MSK - Minimum-Shift Keying

NB-IoT - Narrowband Internet of Things

NDIR - Non-dispersive Infrared Radiation

NMEA - National Marine Electronics Association

OFDMA - Orthogonal Frequency Division Multiple Access

OTAA - Over-The-Air-Activation

OOK - On-Off Keying

PM - Particulate Matter

PaaS - Platform-as-a-Service

PPM - Parts per Million

RFM - Radio Frequency Module

RSSI - Received Signal Strength Indicator

RFID - Radio-Frequency Identification

SC-FDMA - Single Carrier Frequency Division Multiple Access

SF - Spreading Factor

SaaS - Software-as-a-Service

SRAM - Static Random Access Memory

SPI - Serial Peripheral Interface

SNR - Signal-to-Noise Ratio

TCP/IP - Transmission Control Protocol/Internet Protocol

TTN - The Things Network

UNII - Unlicensed National Information Infrastructure

UART - Universal Asynchronous Receiver/Transmitter

USB - Universal Serial Bus

Wi-Fi - Wireless Fidelity

WHO - World Health Organization

CHAPTER 1

Introduction

The population's well-being is a crucial factor, both socially and economically. Air pollution is an aspect that has a major influence, not only on the economy, but mainly on public health, leading people to stop living a healthy lifestyle. To overcome this problem, measures have been imposed to decrease pollutants and systems have been used to monitor air quality.

To introduce and address this problem, a motivation associated with the subject in discussion, will be given. The next part will present the context of the dissertation topic and will describe the developed project. The questions which led to the development of this project will also be presented. The goals of the project will be presented after the questions. The methodology that was used to carry out this project will be also presented, and finally, the organization of the chapters of the dissertation will be described.

1.1. Motivation

One of the world's biggest problems, which affects millions of people all over the world, is air pollution, which can come in two forms: indoor and outdoor pollution. The emission of chemical and physical substances from industrial operations, combustion-powered cars, certain domestic equipment, forest fires and the burning of fossil fuels are the leading causes of this type of pollution [1].

Long-term exposure to this kind of substances has negative impacts on public health and can lead to serious problems such as cardiovascular and respiratory diseases and even cancers. In 2019, around 99% of the world's population breathed air that did not meet the requirements imposed by the WHO (World Health Organization) [1] and it was estimated that around 4.5 million people died prematurely due to outdoor air pollution [2]. Small particles, or particulate matter (PM), and ground-level ozone (O_3) are the main causes of the high mortality rate [2]. In the case of particulate matter, there are several types according to their size. The most common are the particles with a diameter of 10 (PM_{10}) and 2.5 ($PM_{2.5}$) micrometres. Although these components are the main pollutants, gases such as carbon monoxide (CO), nitrogen dioxide (NO_2) and sulphur dioxide (SO_2) also contribute to air pollution [3].

This problem not only has serious consequences for human health, but also has negative economic and social impacts. As pollution increases, the costs associated with health increase and productivity in certain sectors, such as agriculture, decreases [4]. Air pollution also has an impact on people's well-being and satisfaction and is associated with disorders in the cognitive development of young people and the appearance of mental problems such as depression, schizophrenia, and anxiety [5].

To tackle this type of pollution, several measures are implemented to reduce the level of pollutant concentration in the air. Within this set of measures, the most common are the use of renewable energies, better management of urban and agricultural waste and opting for public transport methods [3]. With the implementation of this set of measures, mortality rates due to outdoor pollution have been falling over the years, as can be seen in Figure 1.1:

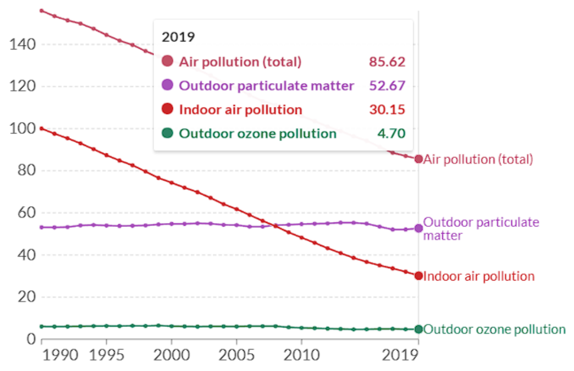


Figure 1.1. Number of deaths associated with air pollution, per 100,000 people [2]

In 2022, the European Commission proposed new measures to combat the concentration of pollutants, with the aim of achieving a cleaner ambient air by 2030, in Europe [6]. These measures involve strengthening plans and applying air quality monitoring models [6]. This monitoring can be conducted using monitoring station systems, which consist of a physical construction containing various components, including sensors capable of measuring the concentration of pollutants, such as carbon monoxide, particulate matter, ground-level ozone, nitrogen oxides and sulphur dioxide.

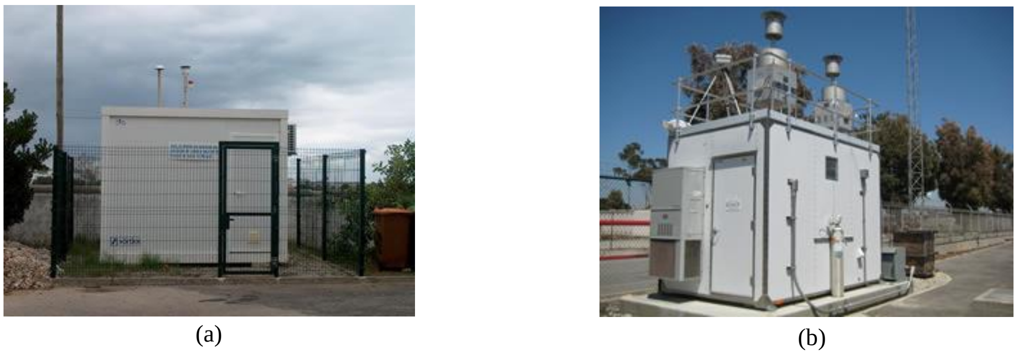


Figure 1.2. Conventional monitoring stations in Portugal (QualAR, 2019) and in USA (The Port of Los Angeles, 2023)

These types of stations can be used to obtain data on the concentrations of various pollutants and can be used to later analyse and build maps that make it possible to assess the concentrations of these pollutants, as can be seen in Figure 1.3. By accessing this information from monitoring stations and applications that allow their data to be visualised, anti-pollution measures can be imposed to reduce the concentration of pollutants in an outdoor context.

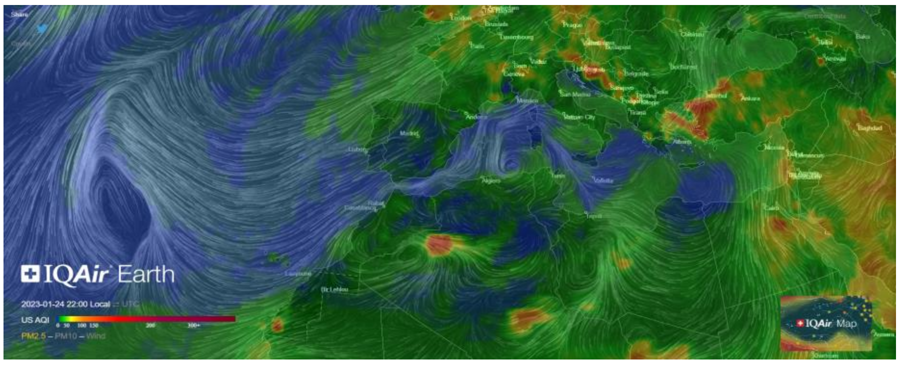


Figure 1.3. Map of pollutant concentration, associated with $PM_{2.5}$ concentration (IQAir, 2023)

1.2. Context

Conventional monitoring stations, as shown in Figure 1.2, offer a high degree of accuracy in their readings because they use a combination of techniques and some calibrated analytical instruments, like gas chromatograph-mass spectrometers [7], to measure several air pollutants, present in an air sample. Although they are expensive to build, require a high level of energy consumption and some of them take up a lot of physical space [7].

With progress in the areas of electronics, there are now low-cost, energy-efficient, and small sensors capable of detecting several types of pollutants, with a high response time (in the order of seconds or minutes) [8], [9].

The IoT landscape has made it possible for sensors of this type to interact with each other and, using certain communication technologies, build air quality monitoring systems. With these systems, the readings obtained by the set of sensors can be sent to a base station or gateway and then forwarded to a cloud service [10], giving the user easy access to the readings.

In this kind of IoT monitoring applications, the requirement is for energy-efficient communication technologies, due to the continuous operation of these systems, which are accessible to implement and therefore have a low implementation cost, and since there may be clusters of monitoring nodes spread across several locations, a wide coverage of this type of technology is required. LPWAN's (Low Power Wide Area Networks) were created to fill the requirements mentioned above [11], and therefore new technologies have emerged that can offer the characteristics required for this type of monitoring applications, such as LoRa and Sigfox [12], [13].

These IoT air quality monitoring systems, capture information about the air quality, so it can be processed, analysed, and treated to be used to visualise and build maps of the level of pollutant concentration in each area.

The aim of this project is to develop an IoT system for intelligent monitoring of outdoor air quality, but unlike conventional monitoring stations, this prototype aims to create a low-cost implementation with reduced energy consumption, using low-cost sensors to detect concentrations of certain pollutants, as well as the temperature and relative humidity level present in the air. These sensor readings will be sent to a LoRaWAN server using LoRa technology, due the need for long-range communication, and then sent via MQTT to a development tool to help process the data and integrate API’s and online services, and finally inserted into a database. To avoid network congestion, the frequency at which the readings are sent considers the amount of urban traffic in each location in real time, using a traffic API. Finally, all the data obtained will be visualised and analysed using a web/mobile application.

1.3. Research Questions

The development of this project generates several questions that will be answered, based on the results obtained. Some of these questions are:

1. Can the results obtained by low-cost IoT air quality monitoring systems be reliable, in such a way as to provide a reference for further measures to prevent outdoor air pollution?
2. Is the use of LoRa technology, in this context of air pollution monitoring, viable and scalable enough to create a larger network with multiple monitoring nodes geographically spread over a larger area?
3. Which advantages does the visualisation and analysis of sensor data, in real time, bring to stakeholders?

1.4. Objectives

The main goal of this project is the intelligent monitoring of air quality in an outdoor context, using LoRa technology to send the sensor readings, to a LoRaWAN server, for further data visualization and analysis on a web and android application.

Another objective is the implementation of a mechanism, on the air quality monitoring system, that changes the frequency at which the data obtained from the sensors, is sent via LoRa, according to urban traffic conditions at the location where the monitoring is being conducted.

Lastly, the project aims to verify whether LPWAN technologies, in this case LoRa technology, can be scaled up in the context of IoT monitoring systems.

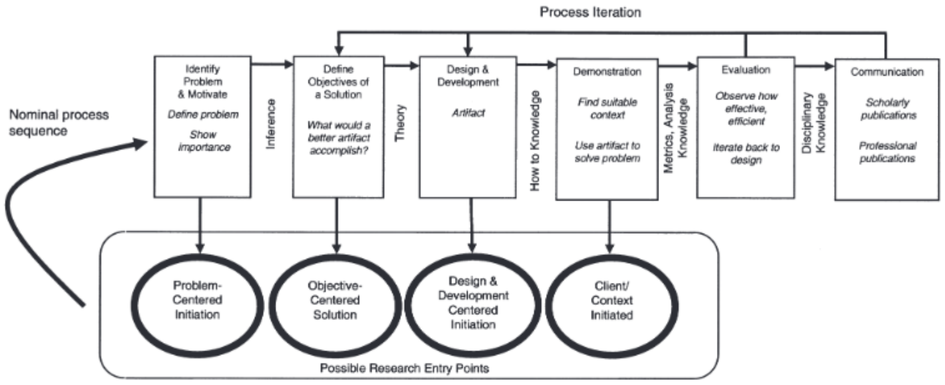
To achieve the above objectives, the following requirements were considered:

1. Development of a monitoring node capable of measuring pollutant concentrations and other atmospheric parameters such as temperature and relative humidity;
2. Implementation of long-range communication using LoRa technology;
3. Storage of sensor readings in a cloud-hosted database;
4. Processing the data obtained for later visualisation;
5. Development of a web and android application to visualise and analyse the data.

1.5. Methodology

The methodology used to develop this project was DSRM (Design Science Research Methodology). This methodology, presented by Peffers et al. [14], is characterized by a process of developing artefacts to address known research problems, contributing to the project's research, evaluating the implementations developed, and demonstrating the results to the public. These artefacts can be prototypes, models, or experimental procedures [14].

It is a methodology that works in a sequential and iterative way and is divided into seven activities: "problem identification and motivation", "definition of objectives for a solution", "planning and development", "demonstration", "evaluation" and "communication". Figure 1.4. shows the order flow of the activities, as well as the several "entry points":



Process Iteration
Nominal process sequence
Identify Problem & Motivate
Define problem
Show importance
Inference
Define Objectives of a Solution
What would a better artifact accomplish?
Theory
Design & Development
Artifact
How to Knowledge
Demonstration
Find suitable context
Use artifact to solve problem
Metrics, Analysis Knowledge
Evaluation
Observe how effective, efficient
Iterate back to design
Disciplinary Knowledge
Communication
Scholarly publications
Professional publications
Problem-Centered Initiation
Objective-Centered Solution
Design & Development Centered Initiation
Client/ Context Initiated
Possible Research Entry Points

Figure 1.4. Structure of the DSRM methodology [14]

The entry points make it possible to determine at which activity of this methodology the development process of the project will begin. Since in this case the project problem has been identified, it will start from the "Problem-Centred Initiation" point, which represents the "Problem identification and motivation" phase. This phase involves a literature study on the problem involved, which is presented in the motivation and context and then in the state of the art.

When this first phase is over, the objectives for the solution in question will be defined, and these are presented in section 1.4. The design and creation of the artefact is the next phase, which represents the development of the solution to the problem. In the case of this project, it represents the development of an IoT monitoring system for outdoor air quality. Once the previous phase has been completed, in the "Demonstration" phase, the solution will be put into operation to measure the presence of pollutants in the air.

In the following "Evaluation" phase, the prototype will be examined for its accuracy in taking readings. If all the previous phases have been completed, we proceed to the "Communication" phase, where all the research, development and results are presented to the public. During the "Evaluation" or "Communication" phase, if new objectives are to be set or the accuracy of the solution needs to be improved, a new iteration of some of the previously phases can be started, as shown in the Figure 1.4.

1.6. Dissertation Structure

This dissertation is structured as follows:

- Chapter 1 - Introduction: the motivation and context that gave rise to the development of this prototype, some questions to be answered, objectives to be achieved with this implementation and the methodology used in the development of this project are presented;
- Chapter 2 – Literature Review: this chapter describes concepts related to outdoor pollution, LPWAN technologies, IoT architectures for this type of system and lastly, it provides a description and analysis of related work on the topic of the dissertation;
- Chapter 3 - System Description: all the hardware and software used in the development of this IoT system is described, as well as a description of how the system works, its architecture and how communication between the different components is carried out;
- Chapter 4 – Results and Discussion: this chapter analyses the data obtained by the IoT system, including a detailed discussion and remarks;
- Chapter 5 - Conclusions and Future Work: finally, based on the results obtained, the conclusions and future work, regarding air quality monitoring and data analysis are considered, for this IoT system are described.

CHAPTER 2

Literature Review

This section describes the state of the art, related to all the research carried out on the topics covered in this dissertation. This chapter is separated into two sections. Section 2.1. is related to the background concepts associated with the project. Concepts such as outdoor air pollution, sensors for pollutants, Internet of Things (IoT), architectures used in IoT systems, communication technologies and cloud computing are presented and described. Section 2.2. presents work related to the dissertation topic. They are described and finally the choices made by the authors in implementing their IoT systems are examined.

2.1. Background Concepts

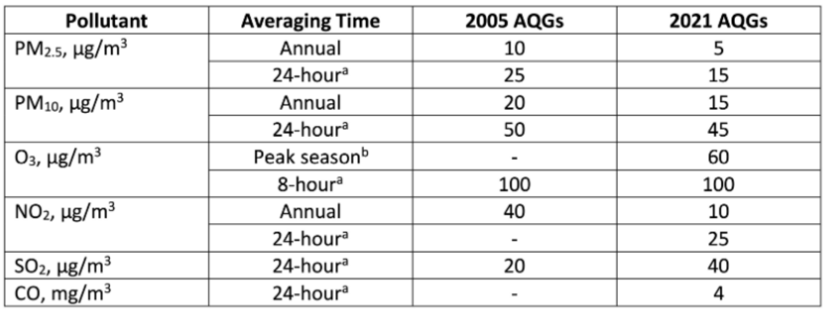
2.1.1 Outdoor Air Pollution

To achieve public well-being, it is necessary to create the conditions to do so, from an economic and social level to health in general. In the case of health, a key factor in achieving well-being is the air we breathe, but this air is subject to pollution. One of the types of air pollution that has a major impact on human health is outdoor air pollution. This type of pollution is caused by human activities such as industrial operations, fossil fuel combustion, forest fires, construction, and agriculture [15], which emit a variety of pollutants into the atmosphere. Pollutants such as carbon monoxide (CO), nitrogen oxides (NOx), sulphur oxides (SOx), ground-level ozone (O_3), particulate matter with diameters of 2.5 μm and 10 μm ($PM_{2.5}$ and PM_{10}), are declared by the WHO as the main atmospheric pollutants [3].

Long-term exposure to this type of pollution can cause serious illness and, consequently, premature death. In 2019, the WHO estimated that heart-related diseases, such as strokes, were associated with 37% of premature deaths worldwide [3]. Respiratory cancers, respiratory infections and other lung-related diseases are also associated as major causes of premature death [15].

To control the emission of these air pollutants and create a healthier environment in different countries, WHO created a list of guidelines [16] with the recommended concentrations of each pollutant. Table 2.1 shows these limits imposed by the WHO, where the different types of pollutants, that have the most impact on outdoor air pollution are listed, and for each of these pollutants, there is a limit concentration, over a certain period, that must be considered to achieve a cleaner outdoor environment, with a decrease in the emission of atmospheric pollutants. In this way, the governments of each country can manage the activities that contribute to pollution and take measures to reduce the emission of pollutants.

Table 2.1. WHO guidelines for each pollutant [16]



Pollutant	Averaging Time	2005 AQGs	2021 AQGs
$PM_{2.5}$, μg/m^3	Annual	10	5
	24-hour[a]	25	15
PM_{10}, μg/m^3	Annual	20	15
	24-hour[a]	50	45
O_3, μg/m^3	Peak season[b]	-	60
	8-hour[a]	100	100
NO_2, μg/m^3	Annual	40	10
	24-hour[a]	-	25
SO_2, μg/m^3	24-hour[a]	20	40
CO, mg/m^3	24-hour[a]	-	4

2.1.2 Air Quality Index (AQI)

The Air Quality Index is a measure designed to detect the level of air pollution, in a certain location [17]. It is a numerical indicator that depends on the concentration of various pollutants. The European Environment Agency (EEA) defines the pollutants considered for the AQI as being: microparticles ($PM_{2.5}$ and PM_{10}), ground-level ozone (O_3), nitrogen dioxide (NO_2) and sulphur dioxide (SO_2) [17].

Table 2.2. AQI levels and respective colors, declared by Environmental Protection Agency (EPA) [18]

AQI	AQI Levels	AQI Level Colors
0-50	Good	Green
51-100	Moderate	Yellow
101-150	Unhealthy for Sensitive Groups	Orange
151-200	Unhealthy	Red
201-300	Very Unhealthy	Purple
301-500	Hazardous	Maroon

Table 2.2 shows the different AQI limits and their associated levels. This index is divided into 6 levels and each level is represented by a different colour [18], [19]. The higher the AQI value, higher the level of pollution at a particular location and the values used for the limits of each AQI level are based on an equation that considers several parameters, presented in [19], one of these parameters being the concentration of the pollutant. AQI's are calculated for each pollutant and the one with the highest AQI value is the one that has the greatest influence on atmospheric pollution, and consequently this will be the representative AQI value in the location under study.

2.1.3 Gas and Particulate Matter Sensors

In IoT systems, the use of sensors has become increasingly important for collecting and processing data [20], for subsequent intervention by some service, actuator, or even human beings.

In the context of air quality monitoring, gas and particulate matter sensors are used to measure the concentration of certain air pollutants and the concentration of particles suspended in the air. The following subsections describe some of the types of sensors used in this type of monitoring system and their operating principles.

2.1.3.1. Metal Oxide Semi-Conductors (MOS) Sensors

In the implementations [21]–[24], Metal Oxide Semi-Conductors sensors are used to detect pollutants such as CO, NO_2 and SO_2. These sensors are known as MOS sensors and are very popular among the range of gas sensors [25]. They consist of an oxidisable metal (depending on the gas being detected), a heating filament, a substrate where the oxidisable metal is deposited and a set of electrodes [7]. Initially, when the sensor is powered on, the filament heats up to keep the sensor at a certain temperature and obtain more accurate results. The flow of electrons between the sensor's electrodes changes according to the gas it is exposed to. When the sensor is in the presence of fresh air, more electrons are accumulated on the surface of the oxidisable metal, resulting in a lower flow of electrons, and consequently, lower conductivity and higher resistance. If the sensor is exposed to polluting gases, propane, methane, carbon monoxide, among others, the electrons are gradually released from this surface, ensuring a greater flow, and resulting in a lower sensor resistance and greater conductivity between the electrodes. The concentration of the gas being measured can be obtained by measuring the electrical conductivity of the metal [7]. As the electrical conductivity increases, the resistance value decreases and the concentration of the studied gas increases.

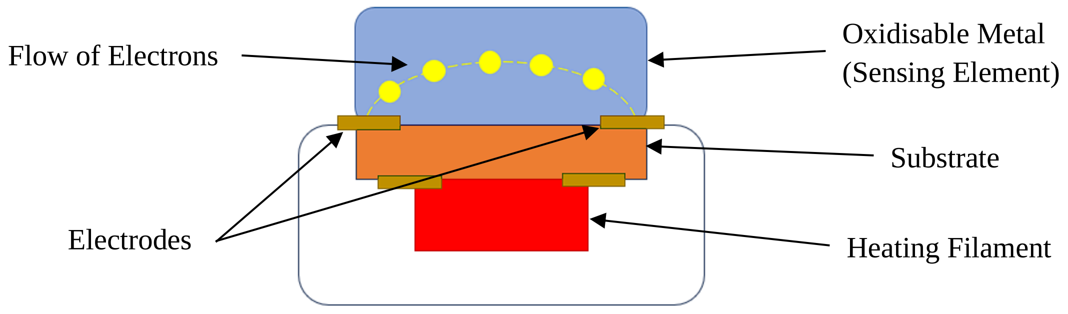


Figure 2.1. MOS sensors inside constitution

They are low-cost sensors, and are resilient to extreme conditions, making them reliable for long-term deployments and implementations that are low budget. These sensors can detect a wide range of gases, but despite this property, they are not very selective, i.e. the presence of a certain gas can influence the concentration of another gas being measured by the sensor, which can lead to misinterpretations of the current concentration of a specific gas. Despite being robust, their sensitivity is affected by changing atmospheric conditions and to obtain more accurate readings they need to be recalibrated [25].

2.1.3.2. Non-dispersive Infrared Radiation (NDIR) Sensors

The CO_2 detection in the implementations of [22], [26] is conducted using Non-dispersive Infrared Radiation (NDIR) Sensors. These types of sensors consist of an infrared light source, a sampling chamber containing the gas to be detected, an optical filter and an infrared detector [25].

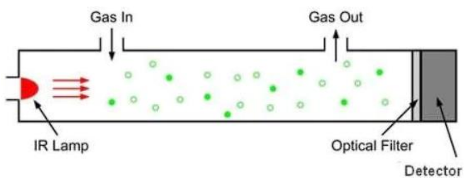


Figure 2.2. NDIR sensors constitution and its operation [27]

The infrared light beam travels through the entire gas chamber to the optical filter. The gas to be detected enters the sampling chamber and the infrared ray is absorbed by the gas molecules in the chamber. Since the gases that enter the sampling chamber have a certain absorption band in the infrared spectrum, the closer the radiation band produced by the infrared ray is to the absorption band of the gas, higher will be the absorption by these molecules, derived from the wavelengths of the infrared spectrum that are characteristic of the structure of the gas [27]. The wavelength of the infrared ray that has not been absorbed by the gas molecules is absorbed by the optical filter and subsequently by the detector, which measures the infrared laser intensity, after the absorption made by gas molecules. The detector calculates the gas concentration through the difference between the intensity of the radiation emitted by the infrared light source and the intensity of the infrared ray received by the detector [27].

NDIR sensors are subject to spectral interference from other gases, which can affect the accuracy of the readings, but these sensors have a long service life and low energy consumption [25].

2.1.3.3. Particulate Matter Sensors

Particulate matter sensors detect microparticles in diverse ways, depending on the sensor used. There are a variety of techniques such as optical detection, gravimetry, microbalance oscillation [28] and beta attenuation to detect the concentration and quantity of microparticles [7], [29]. In the implementations presented in [21], [22] and [26], particulate matter sensors are used, based on the light scattering principle. Sensors using this technique consist of an infrared light source, a photo detector, and a detection chamber.

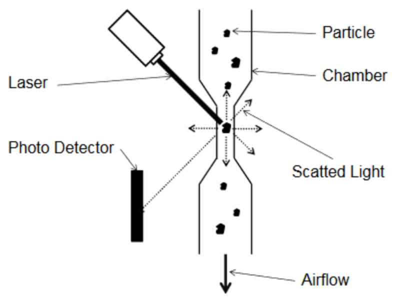


Figure 2.3. Light Scattering PM sensors and its operation [7]

The infrared light beam is in continuous operation, and when a microparticle travels past the detection chamber and intercepts the infrared beam, the beam is scattered inside the detection chamber, and this scattering is detected by the photo detector [7]. Depending on the intensity of the scattered beam and the scattering patterns caused by this scattering, an electrical signal is produced, making it possible to determine the density and how many microparticles are contained in that air sample. Sensors using this technique can detect particulate matter with diameters around 2.5 μm, 10 μm or smaller [25].

Parameters such as temperature and relative humidity can affect the reading of these sensors. Despite being susceptible to these parameters, they are suitable for long-term deployments because they don't require much maintenance [25].

2.1.4 Internet of Things (IoT)

With the increasing number of electronic devices in society, there is a need for connectivity between them and to the internet. It is expected by 2030, 80 billion devices will be connected within a network and 20.5 billion will be connected per person [9].

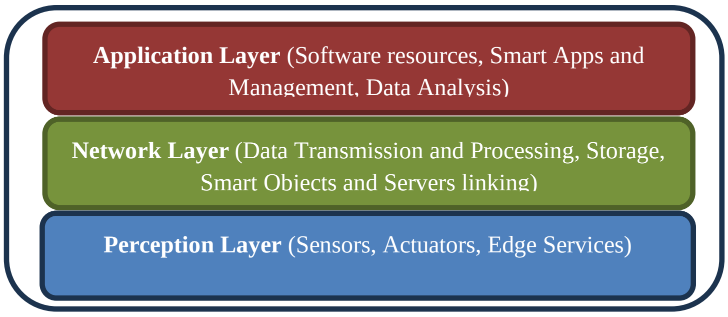
To fulfil these needs, the Internet of Things has emerged. This concept is based on the idea of a large network of physical devices with integrated software, electronics, actuators, and sensors, which are connected to the internet, enabling communication between devices, collection, and exchange of data [26].

The Internet of Things has become an aid in diverse areas, such as health, agriculture, industry, transport and combating certain global problems, such as the management of natural resources, the energy crisis and, air pollution [30].

To obtain information about air pollution, IoT monitoring ecosystems have been created that allow data to be collected and analysed. With advances in the field of electronics, new sensors have emerged with low cost, consumption and size [8], [9], and in combination with certain communication technologies, it is possible to create IoT ecosystems for monitoring parameters, which collect and exchange information about the environment, for subsequent action by humans [31].

2.1.5 IoT Architectures

There are numerous components that comprise an IoT system, from sensors and actuators to web platforms, and each of these has a function associated with it. The architecture of these systems includes different components that can be associated with different layers. The perception, network and application layers make up the three layers of the basic architecture model [32], [33]. In the works presented in section 2.2.1, the architecture used for their IoT systems follows the pattern of this IoT architecture model.

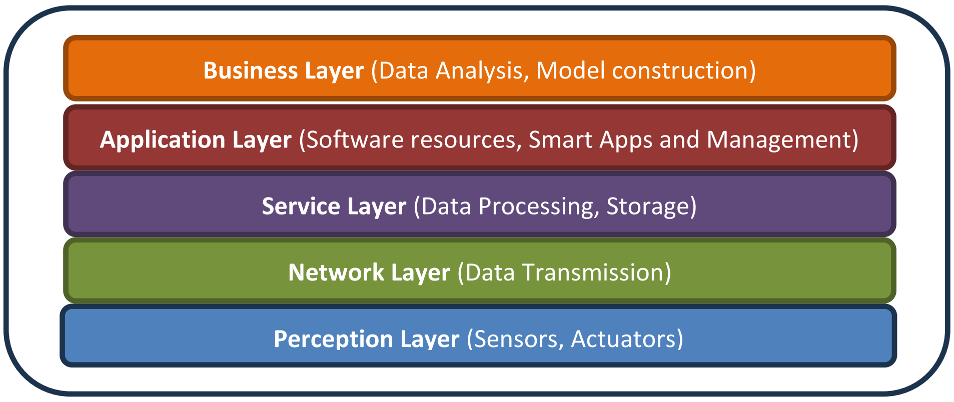

Application Layer (Software resources, Smart Apps and Management, Data Analysis)
Network Layer (Data Transmission and Processing, Storage, Smart Objects and Servers linking)
Perception Layer (Sensors, Actuators, Edge Services)
Figure 2.4. 3-layer architecture model for IoT systems

The perception layer, or sensing layer, is the first layer of this architectural model, and its primary function is to collect data about the surrounding environment. This layer includes sensors, actuators, and RFID tags, and is also responsible for interconnecting these physical components to the IoT network [26], [33].

The second layer is the network layer and is the layer just above the perception layer. It is responsible for processing and receiving data from the perception layer and forwarding this information via gateways to external services such as cloud platforms, databases, or visualisation platforms. Communication is achieved through technologies such as Wi-Fi, Bluetooth, mobile networks (3G or 4G) or LPWANs [26], [33].

Finally, the last layer is the application layer. This receives all the information from the network layer and is used for specific applications (data visualisation, database storage, cloud storage, intelligent action), acting as an interface between users and applications or services [26].

Depending on the user's needs and specific use cases, other models for IoT system architecture have started to emerge. Another example is the 5-layer model, composed by the perception, network, service, application, and business layer. Between the application and network layers, there is another layer, the service layer (or middleware layer). Similar to the 3-layer model, the perception layer is designed to collect data from the surrounding environment. The functions associated with processing, analysing, and storage of data, are part of the service layer, leaving the network layer only responsible for the data transmission. In this model, the application layer is only responsible for intelligent action and providing data and services when the user requests them. Finally, the business layer is designed for data analysis and model and chart construction for further interpretation [32]–[34].


Business Layer (Data Analysis, Model construction)
Application Layer (Software resources, Smart Apps and Management)
Service Layer (Data Processing, Storage)
Network Layer (Data Transmission)
Perception Layer (Sensors, Actuators)
Figure 2.5. 5-layer architecture model for IoT systems

2.1.6 LPWAN (Low Power Wide Area Network)

The emergence of IoT systems has overcome some of the problems of traditional systems based on cellular networks, such as 3G or 4G. Their objectives are to make a more efficient use of the frequency spectrum, to achieve lower latency, to reduce costs associated with hardware and its implementation, to achieve a greater range of coverage and reach lower energy consumption for longer battery life [9]. To achieve these objectives, Low Power Wide Area Networks (LPWAN's) have been created.

LPWAN's use a star topology, where the devices connect directly to base stations. In this way, the cost of implementing gateways is minimised and devices don't have to waste energy "listening" to other devices that want to route traffic through them [30]. Unlike the traditional use of the 2.4 GHz band by technologies such as Wi-Fi, LPWAN's use sub-GHz bands, allowing for less signal attenuation, reduced energy consumption and greater communication range [30].

LPWA networks consume less energy as they don't require much bandwidth to operate. Some IoT devices have a strong need for long-range communications and the use of LPWAN's makes it possible to satisfy this need and find a less expensive solution.

With the emergence of LPWAN's, technologies have also begun to be developed that allow the characteristics required for IoT systems to be achieved. In accordance with this dissertation, the use of this type of technology is very much required, as the project is based on building an IoT ecosystem for monitoring air quality, which sends only sensor information over a long-range network. The following subsections describe the characteristics of some of these technologies.

2.1.6.1. Sigfox

Sigfox is an LPWAN technology that was developed in France in 2010 by the Sigfox group. From proprietary base stations, it is possible to connect to backend servers via an IP-based network [30].

It is a technology based on unlicensed bandwidth, i.e., it is isolated from costs and any organisation can use this frequency spectrum and communication medium [35], where it uses the 868.180 - 868.220MHz bands in Europe, with 400 100Hz channels, 915MHz in North America and 433MHz in Asia [12]. It is also characterised by its very narrow bandwidth, which allows for efficient use of the spectrum and reduced noise and energy consumption [11], [12], [30]. As it has a very low data rate (100 bps), Sigfox is very limited when it comes to sending information.

Initially, only uplink communication was supported, but later downlink communication was implemented. The range associated with this technology is 30-50 kilometres in rural areas and 3-10 kilometres in urban areas [11].

To fulfil the regional spectrum usage regulations, only 140 uplink 12-byte messages can be sent per day. From the 400 available channels, devices can choose any of the available channels to send their messages. These messages are sent several times to increase the probability of reception by the base stations [30].

2.1.6.2. Narrow-Band IoT (NB-IoT)

NB-IoT is a narrowband-based technology developed by the Third-Generation Partnership Project (3GPP) group in 2016. Unlike Sigfox, NB-IoT uses a licensed frequency band, i.e., the user is subject to a cost if he wants to use the NB-IoT frequency band. This technology is compatible with LTE, since it is a technology that comes from LTE [30].

In uplink communication, NB-IoT uses the Single Carrier Frequency Division Multiple Access (SC-FDMA) method, where multiple sub-carriers carry the same symbol, and in the case of downlink, Orthogonal FDMA (OFDMA) is used, which has the same operating principle as FDMA, except that each symbol is carried by a sub-carrier [12], [30]. On the uplink, communication is limited to 20 kbps and on the downlink, it is limited to 250 kbps. In systems that have implemented NB-IoT, battery life can reach 10 years if an average of 200 bytes are transmitted per day.

It's a technology that allows the interconnection of several devices within a cell, allowing the connectivity of 50 000 devices, and this number can be increased adding more NB-IoT carriers [30].

2.1.6.3. Long Range (LoRa)/Long Range Wide Area Network

LoRa is a technology developed by the company Semtech, which acts on the physical layer, where signals are modulated via unlicensed frequency sub-bands in the order of MHz. In Europe, LoRa works with the 868 MHz and the 463 MHz band [36], with these two, being divided in sub-bands. LoRa uses a chirp spread spectrum (CSS) technique, which consists on spreading the signal over a wide bandwidth, using a chirp signal, in which the frequency of the signal can increase or decrease over time, making the signal more difficult to be detected and with less interference [12], [36], and also, it enables the possibility of achieving long-range transmission with low energy consumption.

This technology can change the binary rate of the data transfer, its range, the lifetime of the batteries and the sensitivity of the receiver, by using Spreading Factors (SF). The SF’s used in LoRa range from SF7 to SF12. The higher the SF, the greater the range and sensitivity of the receiver, but the data rate and the lifetime of the batteries, decreases [12], [30], [37]. The choice of SF’s is based on network conditions. The mechanism that allows the choice of SF's, and consequently, the adaptation of communication parameters, is called Adaptive Data Rate (ADR). ADR ensures an approach for optimizing data rates, the time that a message takes to arrive to a gateway, and energy usage in the network, automatically adapting communication parameters [38], in accordance with network conditions. This adjustment is performed by the LoRaWAN server or infrastructure. Throughout signal transmission, communication metrics (packet loss, signal strength, signal-to-noise ratio) are exchanged between the device, gateway and LoRaWAN server/infrastructure. The LoRaWAN server analyses these communication metrics, and based on this data, it uses the ADR mechanism, to dynamically adjust several communication parameters to optimise the device's performance on the network.

As LoRa acts on the physical layer, the technology responsible for communication is LoRaWAN. LoRaWAN it's open-source technology, from the LoRa-Alliance association, that deals with the MAC (Medium Access Control) layer.

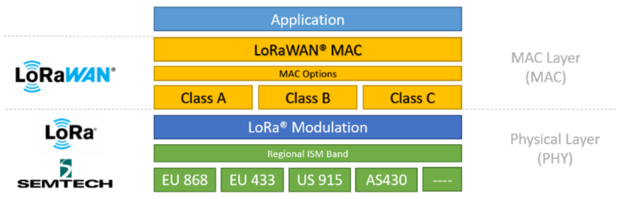


Figure 2.6. Constitution of the physical and MAC layers, related to LoRa and LoRaWAN, respectively (LoRa and LoRaWAN – Semtech LoRa, 2023)

LoRaWAN operates between 0.3kbps and 50kbps [12] and is not a very desirable technology for transmitting content that requires higher binary rates, such as video or images. Two-way communication is available on all LoRa devices. However, there are some limitations associated with sending and receiving messages from the devices. Three classes of devices have therefore been created: Class A, associated with devices that are powered by batteries but have a high latency. Downlink messages can only be sent when an uplink message is first sent by the device, after which the device opens two small reception windows for downlink communication; class B, which has the same features as class A, except that has a lower latency and downlink messages are sent in scheduled reception windows at a predefined time interval. This feature is achieved by synchronising the devices with the base stations via signalling messages sent by the base station. Finally, class C, where devices are always listening, in exchange for minimal latency in receiving downlink messages and very high energy consumption, making this class more suitable for devices with their own power supply [11], [12], [30].

The architecture of a LoRaWAN network is characterised by having physical devices that send information via LoRa to gateways, and then the information from the gateways is sent via TCP/IP to a network server, and then forwarded to application platforms. The architecture diagram used in this type of technology is shown in the Figure 2.7:

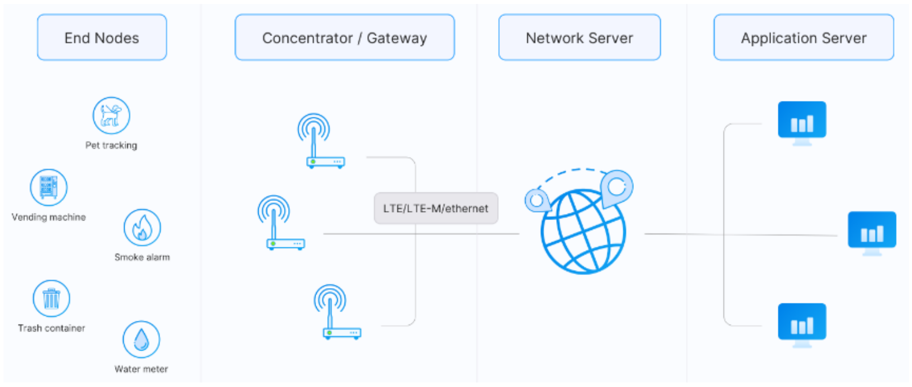


Figure 2.7. Architecture of a LoRaWAN network (LoRaWAN Architecture – TTN)

2.1.7 Cloud Computing

IoT monitoring systems generate large amounts of data and use external services or tools for further analysis or processing.

Cloud computing has made it possible to optimise these IoT systems by providing access to resources, applications, tools, and data storage integrated into remote servers over the Internet [39]. This allows entities to reduce the cost associated with installing and configuring infrastructures, such as physical servers, provides flexibility in accessing multiple services, without the need to install software on local devices and access to resources, like data or tools, and these tasks can be done from any location, as long as there is internet access.

There are 3 types of cloud services: SaaS (Software-as-a-Service), in which entities access cloud software or applications (such as email applications) via their browser, with no need to maintain or manage the services; PaaS (Platform-as-a-Service), which offers access to remote servers, databases, operating system software, storage and tools for developing, managing and running applications; IaaS (Infrastructure-as-a-Service), which allows online access to computing resources/infrastructures, such as access to physical and remote servers, data storage and networks, without the need to implement this type of infrastructure locally [39].

2.2. Related Work

With the emergence of these communication technologies and sensors capable of detecting different types of polluting gases and microparticles suspended in the air, several IoT implementations for monitoring air quality have started to appear.

In a project carried out by Simitha K. M. and Subodh Raj M. S. [21], the concentration of pollutants in a city in India is studied using an IoT system that takes readings from sensors and uses the LoRa and Wi-Fi technology to transmit the data acquired. This system studies the concentration of pollutant gases like CO, NO_2, SO_2, using the MQ-7, MQ-135, MQ-136 sensors and also measures the concentration of particulate matter, with the usage of GP2Y1010AUF optical sensor. In this monitoring system, the sensors provide information about the pollutants levels, with these being connected to the analog inputs of the Arduino Mega board. An SX1278 RA-02 LoRa module is connected to this Arduino board to send the sensor readings via LoRa. A LoRa module is connected to a ESP32 Wi-Fi module to receive the readings sent, with the ESP32 module, acting as a LoRa gateway. It should be noted that the LoRa modules are separated by 1.5 kilometres. The values obtained by the sensors are sent through LoRa to the gateway and later sent over Wi-Fi, from the gateway to the ThingSpeak platform. Finally, a Raspberry Pi 3 Model B+ is used to read the pollutant readings from the ThingSpeak platform through a Python script and then visualized with a Photoshop CS6 template.

The authors in [40] proposed a mobile monitoring system for detecting certain pollutants, such as carbon monoxide (CO), nitrogen dioxide (NO_2) and sulphur dioxide (SO_2), which can be placed in any public transport vehicle. The monitoring node uses an array of sensors to detect the above-mentioned pollutants, a GPS module and a modem using General Packet Radio Service (GPRS) technology for 2G and 3G cellular communication. The measurements obtained by the sensors are then transmitted to a public GPRS base station and then forwarded to a Pollution server, defined by the author as a "an off-the-shelf standard personal computer with accessibility to the Internet", which provides a graphic interface for the user to access the pollution levels, air quality index and location of the monitoring node. Some programming made on the server, is made in PHP, with this server being connected to a MySQL database to store the readings.

Aziz A. A. et al. [22] created a portable IoT air quality monitoring system that captures the concentration of certain pollutants (CO, CO_2), particulate matter, and parameters such as temperature and relative humidity using sensors that are also connected to an Arduino UNO board with an integrated LoRa Shield, so that the readings can then be sent via LoRa to a gateway consisting of a WeMos D1R1 board and a LoRa Shield. The WeMos D1R1 board then sends the readings via Wi-Fi to the ThingSpeak platform so that the data can later be viewed. This implementation was conducted in Malaysia, and initially, two scenarios were carried out, measurements indoors and measurements outdoors, specifically at a construction site. The concentration of particulate matter was higher at the construction site, due to being a space that is heavily polluted by dust particles. The concentration of CO_2 was also found to be much higher at the construction site than indoors. An experiment was also carried out with cigarettes in an indoor space to check the behaviour of the CO concentration. It was found that the more cigarettes that burned, the higher the CO concentration. To compare the concentration of pollutants with the pre-defined values for the AQI (Air Quality Index), measurements were taken in two different locations in Malaysia, Ipoh and Kampung Jalan Kebun. Ipoh has a higher population density and several industries and Kampung Jalan Kebun is a rural location. AQI values are defined by calculating an equation based on the CO and particulate matter concentrations. It was found that the AQI is higher in an urban centre, such as Ipoh, meaning that there is a higher pollution in this location.

The system proposed by Husein N. A. A. et al. [23] is based on a network of monitoring nodes that transmit readings from sensors that detect carbon monoxide (CO), carbon dioxide (CO_2) and nitrogen oxides (NO_x), using LoRa technology, to a gateway and then display the readings on a web interface created by the authors. The monitoring nodes are composed by an Arduino UNO R3 board, MQ-2, MQ-7 and MQ-135 sensors to detect CO_2, CO, and NO_x respectively, a portable rechargeable battery and a LoRa Shield RFM module, attached to the Arduino board, for LoRa transmission. The gateway, on the other hand, is made up of an Arduino UNO R3 board and a LoRa Shield RFM module to receive the readings. In this IoT system, the data is updated every minute. Two outdoor scenarios and one indoor scenario were evaluated. In the first outdoor scenario, the nodes were placed in distinct locations within a Malaysian university campus (UKM campus). In the second scenario, the nodes were placed in the city centre of Kajang, in Malaysia. In the indoor scenario, the nodes monitored three areas of the university campus library. A last study was carried out regarding the maximum coverage distance for LoRa transmission between the nodes and the gateway, carried out in the university campus stadium.

Another implementation of IoT systems for monitoring air quality, created by Thu M. Y. et al. [26], collects pollutant readings from sensors, which are connected to a The Things Uno board. The sensor readings are transmitted using LoRa technology, and these readings are directed to a LoRaWAN gateway so that the data can then be sent to The Things Network (TTN) platform. Concentrations of CO_2, particulate matter only between 1-2 μm in size and the level of temperature and relative humidity are measured using the T6713, SM-PWM-01C and T9602 sensors, respectively. From the TTN, the sensor values are sent to a database, InfluxDB, via the MQTT protocol, so that this data can be visualized on the Grafana platform. A machine learning algorithm written in Python is also applied to predict the temperature and humidity values, to be compared with the values obtained by the sensors. The tests were conducted in the Yangon region of Myanmar.

An IoT air quality monitoring system [24] was built to measure just two parameters: carbon monoxide (CO), using an MQ-9 sensor, and air quality in general, using an MQ-135 sensor. Several monitoring nodes were built using an Arduino UNO board, a portable battery, a Bluetooth module (HC-05) a LoRa module and MQ-9 and MQ-135 sensors. A gateway is also used, consisting of an Arduino Uno board, a LoRa receiver module, to receive the readings from the monitoring nodes, a SIM800L module, which allows communication via General Packet Radio Service (GPRS) technology and an ESP8266 module, which allows Wi-Fi connection. The readings are sent from the monitoring nodes to the gateway and then to the ThingSpeak platform. SMS warnings are sent to the user from the SIM800L module if the predefined air quality thresholds are exceeded. The user can also receive the sensor readings on his mobile device via Bluetooth by installing the "Arduino Bluetooth App". The user pairs his mobile device with the HC-05 module, and each time measurements arrive at the gateway, they are sent to the user via the Bluetooth module. Each monitoring node was placed in 5 different locations throughout a university campus in an Indian city to detect pollution levels. In a final study, it was compared the use of LoRa technology with Wi-Fi to send readings, with monitoring nodes placed along routes between the 5 locations of the university campus, in an outdoor and indoor scenario.

NDIR sensors have been used to detect pollutants such as CO_2 in implementations including [22], [26]. There are bands in the absorption spectrum of infrared radiation that make up the absorption spectrum of CO_2 [41], so the characteristic wavelength of the infrared ray is easily absorbed by CO_2 molecules. This is why NDIR sensors are widely used for CO_2 detection. As a result, the readings obtained are highly accurate, using NDIR sensors, compared to CO_2 detection using MOS sensors, due to their different operating mode.

MQ sensors can detect a large range of gases and are very affordable. In implementations [21], [23] and [24], where it is necessary to measure several pollutant gases concentrations, the adoption of these sensors in monitoring systems is suitable. Despite being able to detect multiple gases, the presence of different gases in the surrounding environment can affect the accuracy of the sensor's readings.

The sensors used for particulate matter detection in [21], [22], [26] are based on optical detection using the principle of light scattering. This type of technique achieves good precision and, as we are talking about small mobile deployments, it is appropriate to have sensors that do not require a great deal of maintenance. Techniques such as gravimetry, in this type of deployment, are not very suitable because they require constant maintenance due to the use of filters.

In the systems presented, the systems' architecture follows the predefined pattern for an IoT system architecture, where the perception layer contains the physical devices, the network layer collects and forwards information to external platforms and the application layer uses external services and platforms provided by the user.

In the implementation carried out by [24], in addition to LoRa technology, Wi-Fi was also used to communicate readings. In the case where the monitoring nodes were deployed in outdoor areas at distances of over 874-1130 metres from the gateway, it was found that using Wi-Fi required several nodes to communicate the readings to the gateway. Wi-Fi, as a short-range technology [34], is not very scalable in long-range systems. It is necessary to implement more nodes so that information can be exchanged via multiple hops to the gateway, which is something that is not very appropriate due to the additional costs of implementing more nodes and the excessive energy consumption. For the indoor deployment, the use of Wi-Fi was more reliable, achieving a lower implementation cost compared to the use of LoRa. In [40], GPRS technology was used to send sensor readings. GPRS is a cellular communication technology, which is why infrastructures such as base stations and sector antennas [42] are needed to enable communication between the monitoring node and the server. For a mobile implementation, using this technology is a viable option because, as it is an older technology, there are several infrastructures already installed for cellular communication, unlike LoRa, which requires its own gateways with LoRa reception modules to receive data. In terms of network coverage, GPRS can't achieve as much coverage as LoRa [42], due to the infrastructure of its cellular network. LoRa, on the other hand, is a long-range technology that offers greater coverage and is a practical choice for long-range deployments. As the system is implemented in a vehicle in motion, the accuracy of the readings may not be very precise, resulting in unreliable data.

In the implementation [23], the study conducted to test Lora transmission coverage showed that the further away from the gateway the monitoring node was, the less packets were received at the gateway. By choosing spreading factors (SF), LoRa can adapt various communication parameters [37], with the use of ADR mechanism. In this case, as the node moves further away from the gateway, it is necessary to increase the spreading factor to increase the communication range. In exchange for this increase in range, the data rate decreases, and the signal is more susceptible to interference. Between the gateway and the node, there were several buildings and trees, which interfered with signal propagation and, consequently, communication lost quality, leading to a packet loss and consequently, to a lower data reception rate.

In [21], [22], [24], the readings are sent to the ThingSpeak platform and in the case of [26], the readings are sent to TTN, to be subsequently stored in InfluxDB and visualised in Grafana. These are two different approaches that depend on the user's needs. InfluxDB offers the possibility of being used by several services, giving the user great flexibility when it comes to retrieving data for use in other real time applications. Grafana platform has a wide range of options when it comes to data visualisation and allows data to be visualised in several dashboards from some databases [43]. ThingSpeak can offer real-time data analysis, allows data visualisation and the user can also act on the data obtained.

CHAPTER 3

System Description

Considering the main objective of the research, which is to monitor outdoor air quality, a prototype of a monitoring node was created, made up of low-cost sensors, using LoRa technology, due to the need for long-range communication to send readings to a LoRaWAN server, and due to its low energy consumption. The readings are collected from various locations in the city of Lisbon, Portugal, and for this reason, the monitoring node integrates a GPS module to analyse where the readings were taken. To achieve an implementation to prevent network congestion, a feature is implemented which, depending on the level of urban traffic, changes the frequency of sending readings to the LoRaWAN server. A web and mobile application have been created so that the user can analyse the readings in real time and in specific time periods.

Throughout this section, a description is given of the system's architecture, the components that make up the monitoring node, how the readings are sent, and all the software used to run this system. This section is subdivided as follows:

- System Architecture;
- Perception Layer;
- Network Layer;
- Application Layer.

3.1. System Architecture

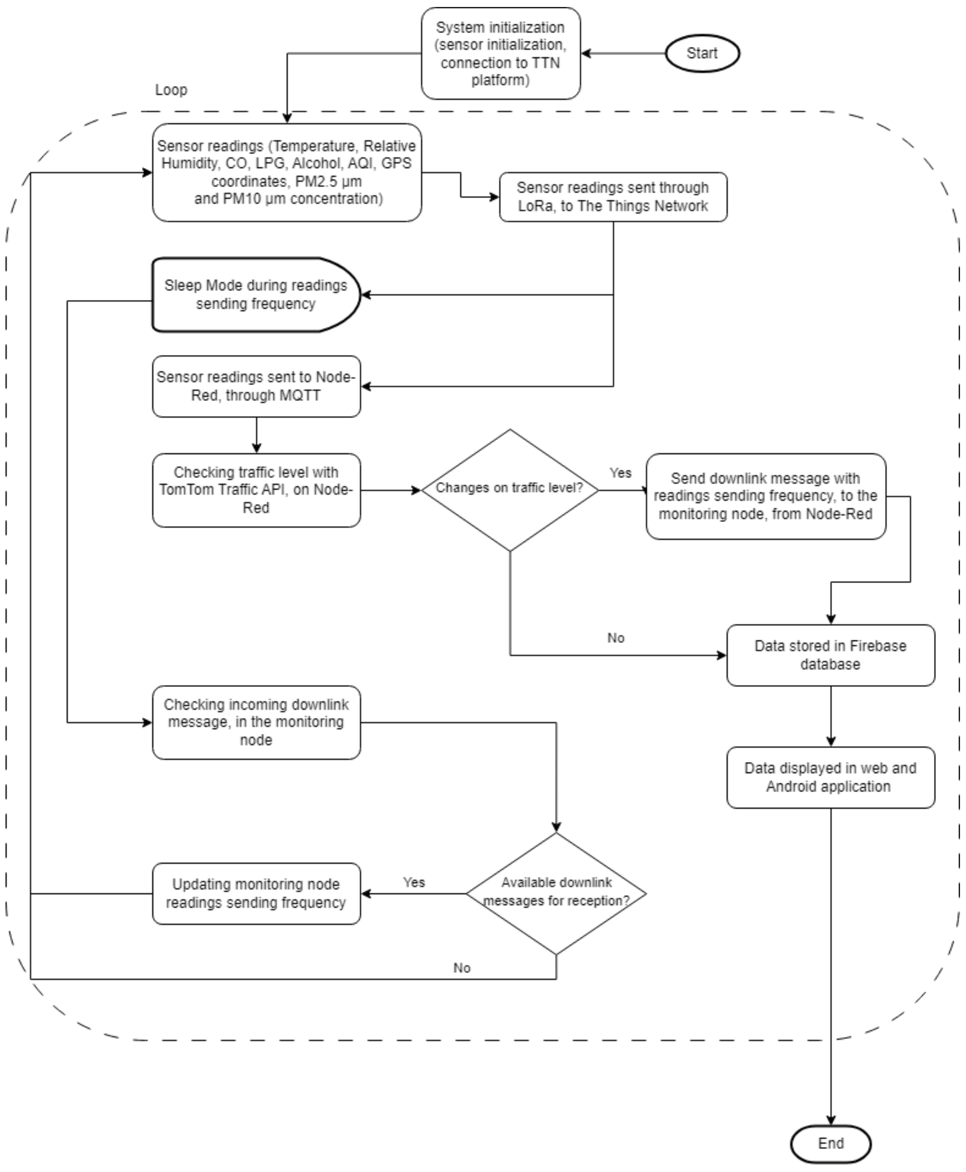
This project mainly describes the creation of an outdoor air quality monitoring node prototype with the integration of LoRa technology. The architecture of this system is divided into three layers. In the perception layer, the node consists of several gas sensors, an air quality sensor, a temperature and humidity sensor, a particulate matter sensor, a GPS module, and a portable battery. A Raspberry Pi is also used to run a software instance that acts as a data receiver and processes the incoming data.

In the network layer, readings are sent periodically from the monitoring node to a LoRa gateway. In this transmission, the data is sent regarding the concentration of certain pollutant gases obtained by the gas sensors, data on the temperature and humidity values, the air quality indicator, the concentration of microparticles, the latitude and longitude of the node's location, and finally the frequency at which the readings are sent to the LoRa gateway. This data then forwarded to a LoRaWAN server. The Things Network (TTN) platform is used to create an instance of a LoRa network that can run a server that receives the readings from the monitoring node. After receiving the readings on the TTN, they are sent via the MQTT protocol to an instance of a programmable tool, Node-Red, inside a Raspberry Pi 4 Model B. Besides receiving the readings from the TTN, via the MQTT integration with Node-Red, the data is also filtered, some API's and online services are integrated and the functionality for changing the frequency of the readings is implemented.

Changing the frequency at which readings are sent is done by exchanging downlink messages (from Node-Red to the microcontroller). The frequency is changed when the level of urban traffic in the location under study changes.

The readings obtained are stored in Firebase, a set of Google computing services that host a database in the cloud [44]. This stored data is used for later visualisation and analysis.

In the application layer, a graphical interface was created in the form of a web and mobile application that allows the user to visualise all the readings obtained by the sensors in real time. Charts can also be generated by selecting fields such as location, date of measurement and the parameter to be studied. This application runs on web and Android platforms. Figure 3.1 shows a flowchart with all the system logic and in Figure 3.2 is shown a diagram summarising the communication of the entire IoT ecosystem created.



Start
System initialization (sensor initialization, connection to TTN platform)
Loop
Sensor readings (Temperature, Relative Humidity, CO, LPG, Alcohol, AQI, GPS coordinates, PM2.5 µm and PM10 µm concentration)
Sensor readings sent through LoRa, to The Things Network
Sleep Mode during readings sending frequency
Sensor readings sent to Node-Red, through MQTT
Checking traffic level with TomTom Traffic API, on Node-Red
Changes on traffic level?
Yes
Send downlink message with readings sending frequency, to the monitoring node, from Node-Red
No
Data stored in Firebase database
Data displayed in web and Android application
Checking incoming downlink message, in the monitoring node
Available downlink messages for reception?
Yes
Updating monitoring node readings sending frequency
No
End

Figure 3.1. Air Quality Monitoring System logic

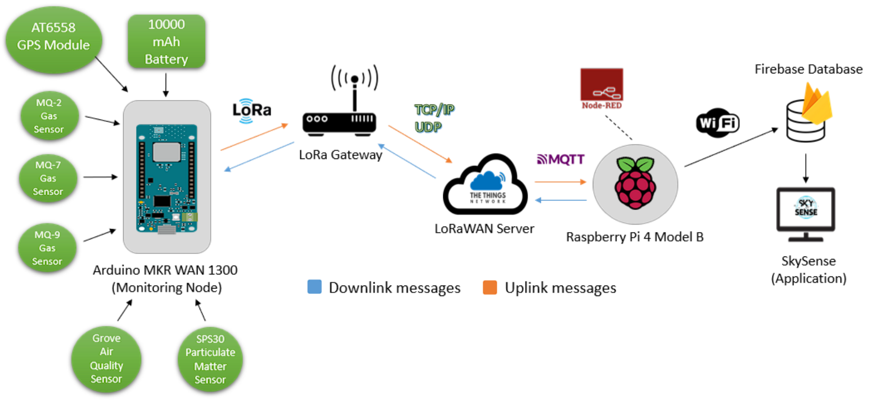


Figure 3.2. Air Quality Monitoring System architecture and its components

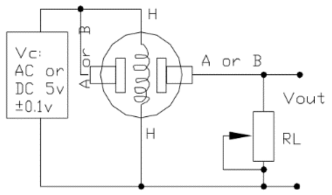
3.2. Perception Layer

3.2.1. Gas Sensors

Fuel combustion is one of the biggest contributors to outdoor pollution, releasing gases that are harmful to public health. In an urban context, some of these gases are released by the combustion of fuel inside the engine of vehicles when they are in circulation.

To detect the presence of certain types of these pollutant gases, this system uses MQ sensors, based on Solid-State Metal Oxide Sensors. Figure 3.3 (a) shows an example of a MQ sensor and Figure 3.3 (b) the circuit that constitutes this type of sensor:

(a)



(b)

Figure 3.3. MQ sensor (a) and its schematic (b)

As explained in Section 2.1.3.1, this type of sensor consists of a metal filament (H - Heater), electrodes, an oxidisable metal, which is usually referred as sensing material, and a substrate on which these components are installed. The sensor also includes a potentiometer, used to adjust its sensitivity when detecting gases. The variation in electrical conductivity can be translated into a variation in resistance; the lower the resistance of the sensor, the higher the electrical conductivity and, consequently, the higher the concentration of the gas concerned [7]. These sensors have limitations in terms of their selectivity. Measurements of a particular gas concentration can be influenced by the presence of another gas, and it can sometimes be assumed that this gas is present when it was just the interaction between another gas and the sensor, leading to misleading interpretations.

The voltage value (V_{OUT}) of the Load Resistor (R_L) is used to calculate the concentration of the gases concerned. To obtain the most accurate measurements, manufacturers recommend that the MQ sensors are switched on so that they enter a warm-up period of 48 hours [45], [46], [47].

3.2.1.1. Sensors Calibration

In this system, the MQ-2, MQ-7, and MQ-9 sensors are used, and in their respective datasheets it can be found the gases that can be captured by the sensors [45]–[47]. These were manually calibrated to capture alcohol, carbon monoxide (CO) and liquefied petroleum gas (LPG), respectively.

Each of the sensors is used to capture a different gas, so each sensor must go through a calibration process to be accurate in obtaining measurements according to the respective gas.

By analysing the circuit diagram of the MQ sensors (Figure 3.3 (b)), and using Ohm's law ($V = I \times R$), the following formula is deduced:

$$I = \frac{V}{R} \Leftrightarrow I = \frac{V_{CC}}{R_S \times R_L} \tag{1}$$

where R_S is the resistance of the sensor which changes according to the concentration of the gas, V_{CC} is the supply voltage of the sensor and R_L is the load resistor. Considering the deduction from the previous formula, substituting the terms of Ohm's law gives the following expression:

$$V = I \times R \Leftrightarrow V_{RL} = \left[\frac{V_{CC}}{(R_S + R_L)}\right] \times R_L \tag{2}$$

From this expression, the value of the sensor's output voltage (V_{RL}) can be calculated, and then the value of R_S can be obtained. By solving Equation 1 in function of R_S, the following formula is obtained:

$$R_S = \left[\left(\frac{V_{CC}}{V_{RL}}\right) - 1\right] \times R_L \tag{3}$$

The V_{RL} value is acquired by converting the analog value obtained by the sensor into a digital value. The Analog to Digital Converter (ADC) on the MKRWAN 1300 board has a resolution of 8/10/12 bits, and the higher the resolution, the more accurate the result. Therefore, the resolution chosen for the ADC was 12 bits, which results in a range of values that can be obtained via the ADC, between 0 and 4095 ($2^{12\text{ bits}} - 1 = 4095$). To convert the value obtained by the ADC to a voltage value, in this case to calculate V_{RL}, the following conversion is made:

$$V_{RL}\,[V] = \frac{ADC\ value \times V_{cc}}{4095} \tag{4}$$

To determine the gas concentration, the R_S/R_0 ratio must be considered. The value of R_0 is calculated using Equation 3 to obtain the R_S value and considering the value of the R_S/R_0 ratio in the presence of fresh air:

$$R_0 = \frac{R_S}{resistance\ ratio\ in\ fresh\ air} \tag{5}$$

Using the datasheets [45]–[47] and the characteristic curves of the respective sensors, the values of the R_S/R_0 ratio in fresh air are taken to calculate R_0. Figure 3.4 (a), (b) and (c) show that the values for this ratio for the MQ-2, MQ-7 and MQ-9 sensors are 9.7, 27 and 9.8 respectively.

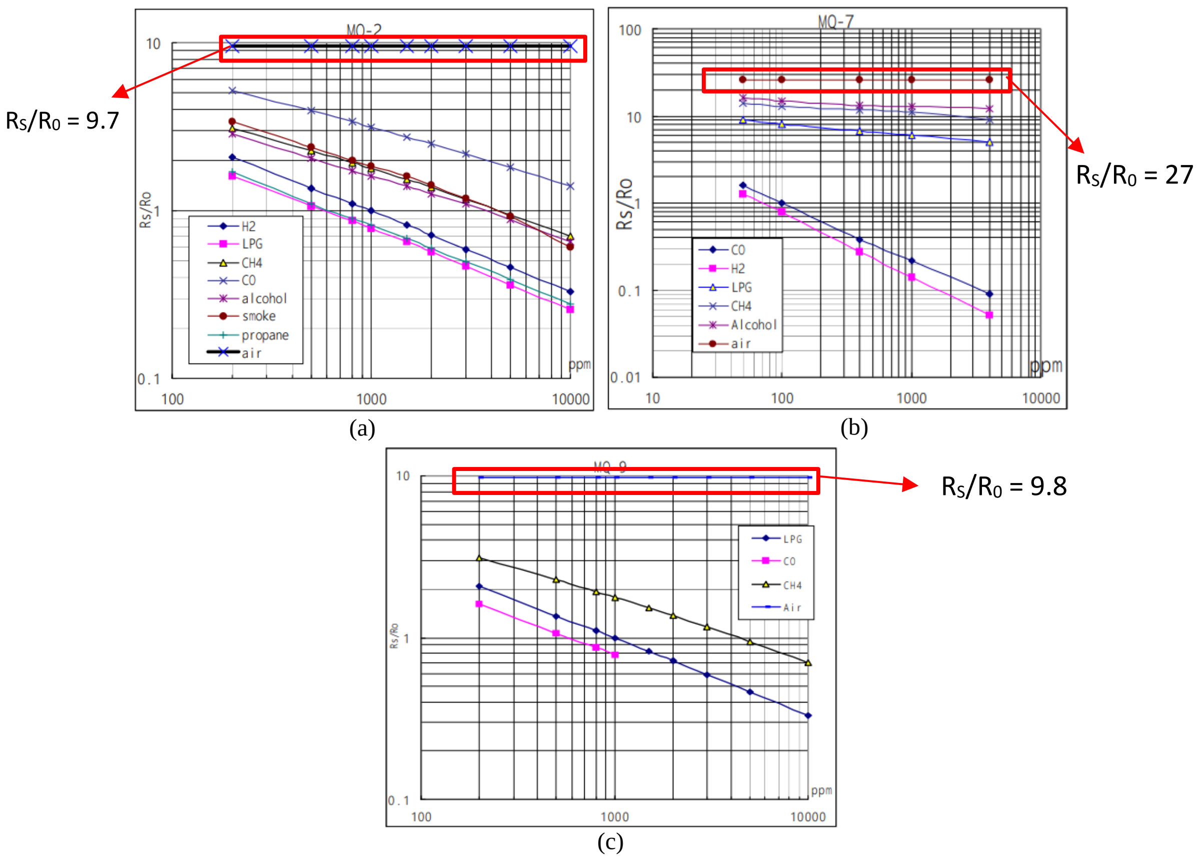


Figure 3.4. MQ-2 (a), MQ-7 (b) and MQ-9 (c) characteristic curves, with the R_S/R_0 value for fresh air

Based on a study carried out by [48], the process described below was used to determine the best formula for calculating gas concentration as a function of R_S/R_0. The characteristic curves of the sensors are on a logarithmic scale, and to find a formula that linearly relates the value of the R_S/R_0 ratio to the concentration of a gas, points were extracted from these curves relating to the gases being studied. Each MQ sensor was assigned a gas to capture, as shown below:

- MQ-2: Alcohol.
- MQ-7: Carbon Monoxide (CO).
- MQ-9: Liquefied petroleum gas (LPG).

The points taken from the characteristic curves of the respective gases are shown in the 3.1 (a), (b) and (c) tables.

Table 3.1. Extracted points on the MQ-9 LPG (a), MQ-2 Alcohol (b), MQ-7 CO (c) characteristic curves

(a)

Extracted values from MQ-9 characteristic curves, related to LPG concentration	
R_S/R_0	Concentration [ppm]
2,1	200
1,4	500
1,2	800
1	1000
0,82	1500
0,72	2000
0,59	3000
0,47	5000
0,33	10000

(b)

Extracted values from MQ-2 characteristic curves, related to Alcohol concentration	
R_S/R_0	Concentration [ppm]
2,8	200
2,1	500
1,7	800
1,6	1000
1,45	1550
1,35	2000
1,2	3000
0,89	5000
0,65	10000

(c)

Extracted values from MQ-7 characteristic curves, related to CO concentration	
R_S/R_0	Concentration [ppm]
1,7	50
1	100
0,38	400
0,22	1000
0,09	4000

Using the points extracted from the characteristic curves, a trendline is created to find the equation that relates the gas concentration to the R_S/R_0 ratio, using the Excel tool. The choice of the best trendline is based on the R-Squared (R^2) value. This value, between 0 and 1, measures how strong the relationship is between the data model and the model points [49]. The one with the highest R^2 value is the power trendline. The curves and equations generated for each sensor are shown below:

➢ **MQ-9: Liquefied petroleum gas (LPG)**

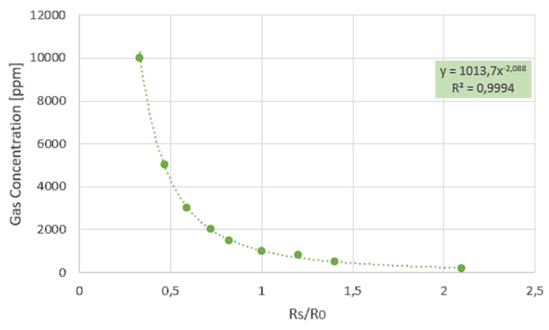


Figure 3.5. Characteristic curve for LPG, related to MQ-9 sensor, in a linear scale

$$LPG\ Concentration\ [ppm]\ = 1013{,}7 \times \left(\frac{R_S}{R_0}\right)^{-2{,}088} \quad (6)$$

➢ **MQ-2: Alcohol**

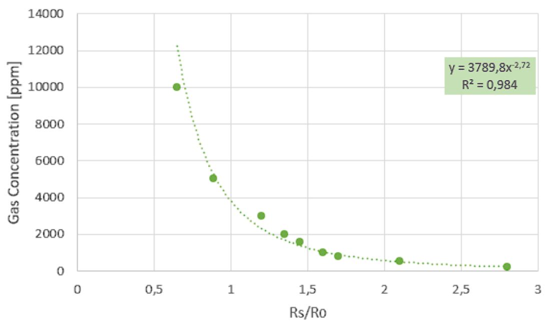


Figure 3.6. Characteristic curve for Alcohol, related to MQ-2 sensor, in a linear scale

$$Alcohol\ Concentration\ [ppm] = 3789{,}8 \times \left(\frac{R_S}{R_0}\right)^{-2{,}72} \quad (7)$$

➢ **MQ-7: Carbon Monoxide (CO)**

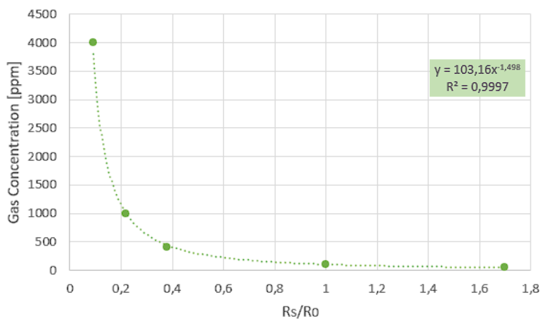


Figure 3.7. Characteristic curve for CO, related to MQ-7 sensor, in a linear scale

$$CO\ Concentration\ [ppm]\ = 103{,}16 \times \left(\frac{R_S}{R_0}\right)^{-1{,}498} \quad (8)$$

3.2.2. Air Quality Sensor

For general air quality monitoring, the Grove v1.3 air quality sensor is used. This sensor monitors ambient air quality, being responsive to gases that are harmful to health (carbon monoxide, acetone, alcohol, among others), but due to its reading mechanism, it is not able to reproduce a quantitative result for each concentration of each gas. Instead, the sensor returns a qualitative result on the current air quality condition. This result is obtained by considering a quantitative result obtained by measuring the voltage measured by the sensor and the higher the value obtained, the lower is the air quality, and vice versa. The sensor is exposed to the surrounding environment, and when a pollutant gas comes directly into contact with the sensor, an electrical signal is generated which is subsequently processed by the microprocessor with the signal output being analog. This sensor is compatible with 3.3 and 5V and has low power consumption. Initially, to achieve good accuracy, the sensor needs to be exposed to fresh air for some time. One of the limitations of this sensor is the decrease in sensitivity if it is exposed to polluted air for a long period of time [50].

Figure 3.8. Grove Air Quality sensor v1.3

3.2.3. Temperature and Relative Humidity Sensor

Parameters such as temperature and relative humidity are obtained using the DHT22 sensor. This uses a capacitive humidity sensor and a thermistor, which measures the surrounding environment and produces a digital signal for the data collected. It is a digital sensor, where data is transmitted over a bidirectional single-bus interface, where the microcontroller sends a request to receive data that is subsequently sent by the sensor. When the initial connection between sensor and microcontroller is made, the signal on the data bus switches from a high voltage state to a low voltage state. When the DHT22 sensor detects this signal and is ready to send data (after a few microseconds), it switches the data bus from low-voltage to high-voltage, transmitting temperature and humidity data. At the end of this process, the sensor is released from the bus, causing the data bus to return to a low-voltage state. It has a reading range of 0 to 100% for relative humidity and -40ºC to 80ºC for temperature and can operate with a voltage of between 3 and 5V. It has low power consumption and better accuracy (around ±0.5°C and ±2%RH measurement error) compared to the DHT11 sensor [51], which is one of the factors for choosing this sensor over the DHT11.

Figure 3.9. DHT22 sensor

3.2.4. Particulate Matter Sensor

Outdoor air pollution is also characterised by the presence of particulate matter, a mixture of solid particles and liquid droplets present in the air [52]. Among the existing types of microparticles, $PM_{2.5}$ µm and PM_{10} µm stand out. $PM_{2.5}$ is one of the most dangerous types of microparticle for public health, due to its small size. PM_{10} also has a strong presence in outdoor air but is larger than $PM_{2.5}$. These microparticles deposit in the lung area, causing serious lung diseases and even death [52]. These are the two types of microparticles being studied, and the SPS30 sensor is used to detect them.

Figure 3.10. SPS30 Particulate Matter sensor

The SPS30 is a particulate matter sensor that uses a detection technique based on the light scattering principle, as described in Section 2.1.3.3. The sensor consists of a photo detector, an infrared laser, a microparticle detection camera and a mini fan, as can be seen in Figure 3.11:

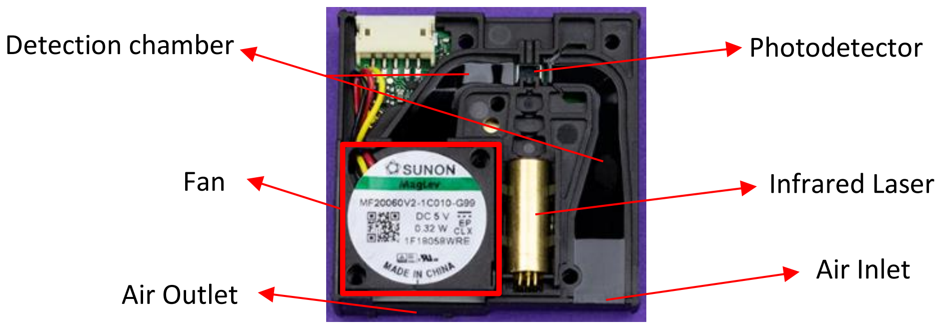


Figure 3.11. SPS30 hardware components

Measurement accuracy can be affected by variables such as temperature and humidity, and the manufacturer recommends that the sensor operates at a temperature between 10 - 40ºC and a humidity in the range of 20 - 80%, to obtain greater measurement accuracy [53]. It can detect microparticles of the following sizes: 1, 2.5, 4 and 10 µm.

This module has three operating modes, idle, sleep and measurement. Idle and sleep modes are designed to reduce the sensor's energy consumption when microparticle detection is not required [54].

When the sensor is in measurement mode, all its electronic components are switched on and the processing and measurement of new data is continuous [54]. In this mode, the laser is in continuous operation, and when microparticles enter the detection chamber, they intercept the laser beam, scattering it, which is then detected by the photo detector [7]. Depending on the intensity of the scattered beam and the patterns caused by this scattering, an electrical signal is produced by the photo detector. This signal is processed by the microcontroller/digital signal processor (DSP) and in this way, information is generated about the concentration and size of the microparticles present in the detection chamber. In the end, the microparticles are blown out of the sensor outlet through the fan [55].

The sensor has two communication interfaces, I2C and UART. Selecting one of the interfaces used for measurements is done by connecting the cables between the pins on the sensor and the microcontroller. In the case of this system, the interface used is I2C, and as indicated by the manufacturer in the datasheet, pins 4 and 5 from the SPS30 sensor, are connected to the microcontroller's GND to select this interface [53].

3.2.5. GPS Module

A GPS module is used in this system to obtain the location associated with the measurements. This module is the AT6558 Mini GPS and provides the exact coordinates (latitude and longitude) of the respective location of the monitoring node. It has an AT6558 navigation chip that supports various satellite navigation systems, such as Global Positioning System (GPS), Global Navigation Satellite System (GLONASS), Beidou Satellite Navigation System (BDS), among others. It uses a UART interface to exchange data with the microcontroller and returns an NMEA (National Marine Electronics Association) string, which can be used to retrieve information about several parameters, such as latitude, longitude, time, date, number of satellites to be used for the GPS signal, among others. It also consists of a MAX2659 chip that amplifies the signal from its antenna. Initially, when the module is connected to the power supply, it takes between 32-35 seconds to obtain localisation data. It can operate at temperatures between -40 and 85ºC [56].

Figure 3.12. AT6558 Mini GPS module

3.2.6. Microcontroller

To build the monitoring node, the main component chosen for programming, interconnecting all the components (sensors, GPS module and battery) and LoRa communication was an Arduino board. Within the range available for this system, two different types of board were tested: Arduino UNO and Arduino MKRWAN 1300. The Arduino UNO board is a robust and simple board to program for various types of projects. It features an ATmega328P microcontroller, which integrates a microprocessor with a frequency of 16 MHz, 2 KB of SRAM memory, 32 KB of flash memory and 1 KB of EEPROM memory [57]. This board does not integrate LoRa communication, but to overcome this problem, a component that can be incorporated into the board was tested. This component is the Dragino LoRa Shield v1.4. It has an RF transmitter/receiver, with a high sensitivity of -148 dBm and through modulation modes such as FSK, GFSK, MSK, GMSK and LoRa^{TM} Modulation, it allows data communication over long distances, with low interference and low data rates through LoRa technology [58].

(a) (b)

Figure 3.13. Arduino UNO R3 board (a) and Dragino LoRa Shield v1.4 (b)

Another board was tested for this implementation, the MKRWAN 1300 board. Unlike the Arduino UNO board, MKRWAN board has already a built-in module that allows communication via LoRa. It's a practical solution for IoT-related projects, it can establish connections with other boards, to its own LoRa network or to existing LoRaWAN infrastructures, and by using a library, it's possible to lower energy consumption and thus increase battery life [59]. It has a SAMD21 Cortex M0+ 32bit microcontroller, which incorporates a microprocessor that operates at a speed of up to 48 MHz and contains SRAM and Flash memory of 32 KB and 256 KB respectively, and a LoRaWAN Murata CMWX1ZZABZ module, also with SRAM memory of 20 KB and Flash memory of 192 KB, which has a receiver sensitivity, that can reach up to -135.5dBm and works with modulations such as FSK, OOK and LoRa™ Modulation [59].

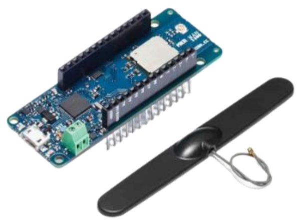


Figure 3.14. Arduino MKRWAN 1300 board and an antenna for LoRa communication

Table 3.2. Comparison between Arduino Uno + Dragino LoRa Shield and Arduino MKRWAN 1300 boards

	Arduino UNO + Dragino LoRa Shield v1.4	Arduino MKRWAN 1300
Microcontroller	ATmega328P	AMD21 Cortex-M0+ 32bit
CPU Clock speed	16 MHz	48 MHz
ADC Resolution	10 bits	8/10/12 bits
UART interface	Yes	Yes
I2C interface	Yes	Yes
SPI interface	Yes	Yes
SRAM Memory	2KB	32KB
Flash Memory	32KB	256KB
LoRa Connectivity	Yes (only with the integration of the Dragino Shield)	Yes
LoRaWAN Module	RFM95W (Dragino Shield)	Murata CMWX1ZZABZ
Carrier Frequency	433/868/915 MHz	433/868/915 MHz

Each one of the microcontrollers supports the Arduino IDE development environment, the software used to program communication via LoRa and collect data from all the components incorporated into the monitoring node. The Arduino IDE supports several libraries containing numerous functions that can be used to program the microcontroller. The use of library functions when programming the microcontroller takes up some memory and therefore it is necessary to have enough memory, in this case Flash and SRAM memory, to use these functions. This was the main criteria for choosing between Arduino boards, and so the MKRWAN 1300 board was chosen. The MKRWAN 1300 board's facility to integrate with existing LoRaWAN infrastructures (such as The Thing Network) was another factor that helped to choose between the two boards. The speed of the microprocessor is decisive for the execution of instructions, and when comparing the two boards, the microprocessor integrated on the MKRWAN 1300 board has a higher speed (48 MHz), which is a feature that allows for faster and more efficient system development.

From the MKRWAN 1300 board, it was designed the system schematic between all the components which constitute the monitoring node. Figure 3.15 shows a schematic of the monitoring node circuit connections:

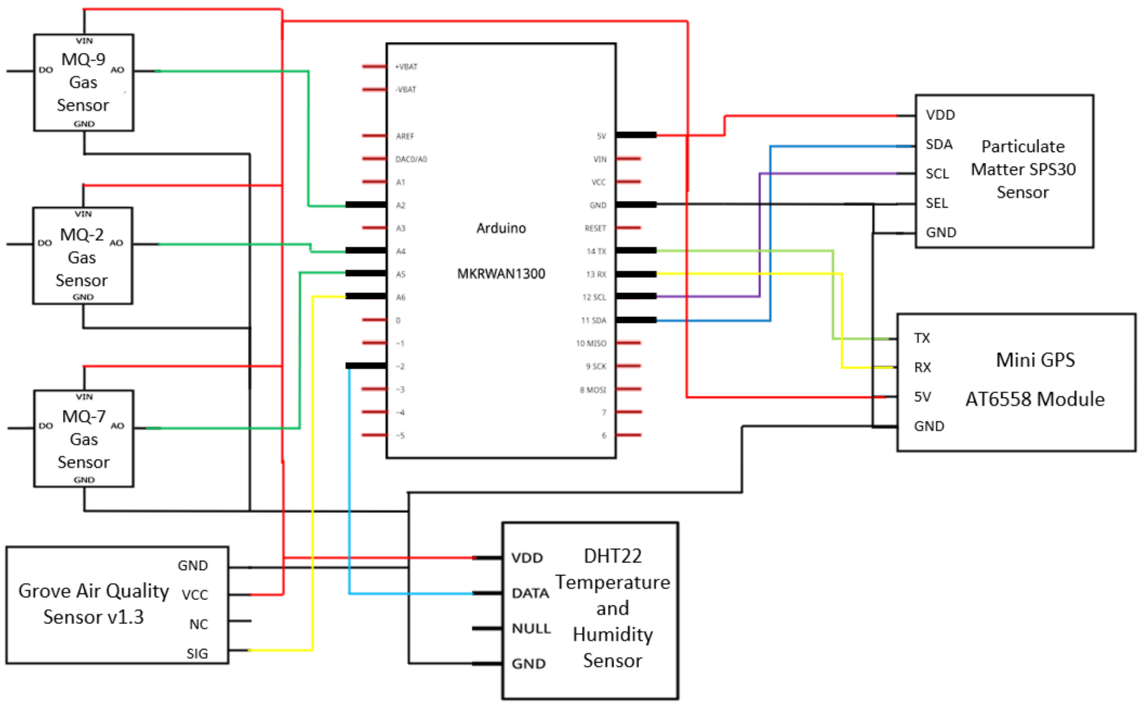


Figure 3.15 Monitoring node circuit connections

A 3D printed box was created to ensure better storage of the components that make up the monitoring node and to make it more accessible when taking readings outdoors.

The monitoring node is powered by a power bank, made up of a lithium battery with a 10000 mAh (milliampere-hour) capacity. Figure 3.16 shows the box with all the components that make up the monitoring node:

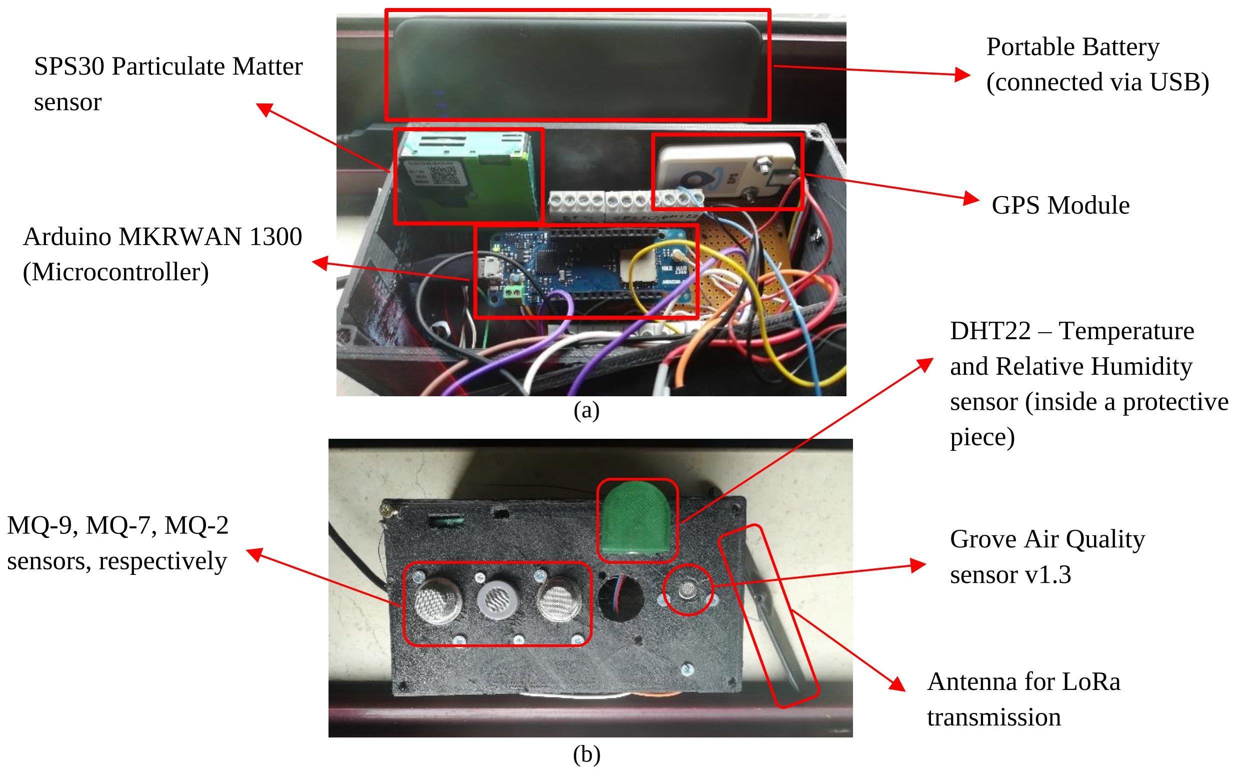

SPS30 Particulate Matter sensor
Portable Battery (connected via USB)
GPS Module
Arduino MKRWAN 1300 (Microcontroller)
DHT22 – Temperature and Relative Humidity sensor (inside a protective piece)
(a)
MQ-9, MQ-7, MQ-2 sensors, respectively
Grove Air Quality sensor v1.3
Antenna for LoRa transmission
(b)

Figure 3.16. Monitoring node hardware components and arrangement, (a) and (b)

3.2.7. Raspberry Pi

The Node-Red tool is an important part of the operation of this IoT system and needs to be running continuously for this implementation to be scalable. To ensure that the Node-Red instance is always operating, a Raspberry Pi 4 Model B was utilized for this purpose. This component has a Broadcom BCM2711 quad-core Cortex-A72 (ARM v8) processor with a speed of 1.5 GHz, 4GB of RAM, 2.4 GHz and 5GHz Wi-Fi connectivity, Bluetooth 5.0 and Bluetooth Low Energy and has a Micro SD slot [60]. From this slot, and with a Micro SD card, it is possible to install the operating system, but also store some data.

Figure 3.17. Raspberry Pi 4 Model B module

3.2.8. Programming the microcontroller (Arduino MKRWAN)

The microcontroller needs to be programmed to be able to take readings from all the components that make up this monitoring node and also to achieve communication via LoRa. As mentioned in section 3.2.6, the microcontroller is compatible with the Arduino IDE software, which was chosen to do all the microcontroller programming. The Arduino IDE is open-source software that allows programmes to be developed and then uploaded to the Arduino boards via a USB connection [61]. These programmes are called sketches and are implemented using a combination of languages, C/C++. In each one of the sketches, there are two main methods: *setup()* and loop(). *Setup()* is the first method to run in the sketch, it only runs once and is used to initialise certain objects or system conditions. The *loop()* method, on the other hand, runs continuously after the *setup()* method and this is where the main part of the code is inserted, such as logic for reading sensors, processing information, logic for communication with other devices/components. In addition to the manual programming done by the user, there is a wide range of libraries that provide extra functionalities/functions to help program the microcontroller. When programming this system, functions from some libraries were used to obtain data from the integrated components and allow the transmission of data through LoRa.

3.2.8.1. Acquiring data from the monitoring node components

Code created as microcontroller firmware is shown in Appendix A. The programming is based on the use of some libraries provided by the Arduino IDE.

Table 3.3 shows the libraries used to program the microcontroller, as well as the functions used and their respective descriptions, to retrieve the values from each monitoring node component:

Table 3.3. Arduino libraries and functions used for the monitoring node programming with their respective description

Used libraries	Used functions	Function description
Sps30.h [62]	*sensirion_i2c_init()*	Hardware and software components initialisation
	sps30_sleep()	Sensor enters sleep mode, switching off most of its internal electronic components [54]
	sps30_wake_up()	Sensor "wakes up" from sleep mode and enters idle mode; it is ready to receive any command; most of its internal electronics are switched on [54]
	sps30_start_measurement()	Sensor switches to measurement mode and all its internal electronics are switched on; sensor is ready to take measurements [54]
	sps30_read_measurement(&m)	Particulate matter values read out and placed inside a structure (m)
	sps30_stop_measurement()	Sensor goes into idle mode [54]
	m.mc_2p5 e m.mc_10p0	Particulate matter values obtained ($PM_{2.5}$ µm and PM_{10} µm respectively)
DHT.h [63]	*dht.begin()*	Initialising the internal components of the DHT22 sensor
	dht.readTemperature()	Obtaining the temperature value
	dht.readHumidity()	Obtaining the relative humidity value
Air_Quality_Sensor.h [64]	*sensor.init()*	Initialisation of the internal components of the Grove Air Quality sensor
	sensor.getValue()	Obtaining a voltage value associated with "air quality"
TinyGPSPlus.h [65]	*gpsSerial.begin(GPSBaud)*	Initialising a Serial channel for data exchange between the GPS module and the microcontroller, with a baud rate of "GPSBaud" bits/s
	gps.location.lat()	Obtaining the latitude value
	gps.location.lng()	Obtaining the longitude value

Using methods such as *sps30_stop_measurement()* and *sps30_sleep()* allows the reduction in the SPS30 sensor energy consumption by switching operating modes, as mentioned in Section 3.2.4.

As mentioned in section 3.2.1.1, to obtain the concentration values for the MQ sensors, it is necessary to calibrate the sensors to first obtain the sensor's resistance value without the presence of gases (R_0). Once the R_0 value has been obtained for each of the MQ sensors, the calculation of the gas concentration is performed. To obtain greater precision when capturing the readings, the ADC was configured to use a resolution of 12 bits, using the *analogReadResolution(12)* method. Once the sensor's analog value has been detected, it is converted to a voltage value using Equation 4. Next, the value of the sensor resistance that varies with the gas (R_S) is calculated using Equation 3, and then the value of the R_S/R_0 ratio is calculated using Equation 5. To calculate R_0, the MQ sensors were switched on for 48 hours, as advised by the manufacturers, and were in continuous operation in a fresh air scenario during this period. The values used for R_0 were obtained from the code shown in Appendix B. Table 3.4 shows the values obtained for R_0 and used for the further calculation of the gas concentration:

Table 3.4. R_0 calculated values for each MQ sensor

	MQ-2	**MQ-7**	**MQ-9**
R_0 [kΩ]	1.30	4.78	8.21

Considering the value of R_0 for each sensor, equations 5, 6 and 7 are used to calculate the gas concentration value for sensors MQ-9, MQ-2, and MQ-7, respectively. Appendix C shows the code for this gas concentration calculation process.

3.2.8.2. Communication through LoRa

Once the data has been collected by the sensors, it is then sent via LoRa to the gateways that are within communication range of the monitoring node. To achieve this communication, the MKRWAN library [66] is used. Using the methods provided by this library, it is possible to construct packets containing sensor readings, send and receive data via uplink and downlink messages, respectively, and adapt certain parameters characteristic of LoRa communication. Table 3.5 shows the methods used from this library:

Table 3.5. Used functions from MKRWAN Arduino library and their description

Used functions	**Function description**
modem.begin(EU868)	Initialising the modem instance with the regional parameter EU868, assigned to each region where LoRa communication takes place [67].
modem.joinOTAA(appEui, appKey)	Connection to the TTN platform, using the *appEui* and *appKey* parameters
modem.beginPacket();	Initiating the process of sending the information packet
modem.write(sensorData, dataSize);	Inserting data into the packet to send via LoRa; *sensorData* represents the array with the data and *dataSize*, the size of this array.
modem.endPacket(true);	Ends the process of sending the information packet
modem.available()	Returns the number of bytes available for reading
modem.read()	Reading incoming downlink data sent by LoRa
modem.sleep()	Puts the microcontroller's LoRa module into a sleep state

Immediately after sending the data, the monitoring node waits for a predefined periodicity at the start of the code, in this case 5 minutes, before sending another packet. When this time interval is up, the *modem.available()* method checks for downlink messages. If there are no downlink messages, new readings are made and another packet is sent within the same time interval, otherwise the downlink message is decoded. The downlink messages that are received by the monitoring node are related to the new time interval that the node must wait after sending a packet. When the downlink message is decoded, the time interval is updated with the decoded value, and then the node's normal operating cycle is repeated, but with the new time interval.

3.2.8.3. Low Power Mode

The monitoring node prototype is powered by a portable battery. Continuous operation of this type of implementation can lead to excessive energy consumption and, consequently, reduce the battery's lifespan. To overcome this problem, a low energy consumption mode was introduced in the microcontroller. By using the Arduino MKRWAN 1300 board, the ArduinoLowPower.h library [68] can be employed. This library offers methods for putting certain internal components of the board into a low-power state. The method used to lower power consumption was *LowPower.sleep(milliseconds).* This method deactivates the microcontroller for a time defined by the user and only the digital peripherals chosen by the user are active.

Despite using this method to put the microcontroller into sleep mode, the LoRa module, which is used for communication, is still active. Using the *modem.sleep()* method allows the LoRa module to switch to a sleep state, optimizing the consumption of the monitoring node.

3.3. Network Layer

This section describes all the software configured for this IoT system and the communication tasks between the various elements of the system's architecture.
Once the data has been obtained by the sensors, it is sent via LoRa technology to the gateways that are within communication range, and from there it is forwarded via IP transport protocols (TCP or UDP) to the LoRaWAN server/infrastructure. The infrastructure used for this system is The Things Network (TTN). TTN is a community ecosystem that uses LoRaWAN technology to create IoT networks and applications. It is part of The Things Stack, a decentralised and free LoRaWAN infrastructure that allows users to register gateways and create applications in this ecosystem [69].

The monitoring node initially needs to be registered on the network. This task is achieved by creating an application, created by the user himself, where he can register any type of device that communicates via LoRa. In the case of this system, the MKRWAN 1300 board was registered in the TTN application. To establish communication between the board and the TTN, the application provides two identifiers: AppKey and AppEUI. AppEUI and AppKey are identifiers that can be generated randomly, where AppEUI is the application identifier and AppKey is an identifier that allows the device to access the TTN application [70]. In addition to these two identifiers, another identifier is required to fully register the device in the application, the DevEUI. This is an identifier that recognises the device itself and is generated by the manufacturer.

By using the MKRWAN library when programming the microcontroller, it is only needed to declare the AppEUI and AppKey inside the script and then the access to the TTN platform is done. This access can be done in two ways, via the Over-The-Air-Activation (OTAA) or Activation By Personalisation (ABP) method. By default, the method used to register the monitoring node is OTAA. The device performs a procedure to join the TTN's LoRaWAN network, where security keys are generated and exchanged with the device [70]. Figure 3.18 shows the usual process of the device joining the network:

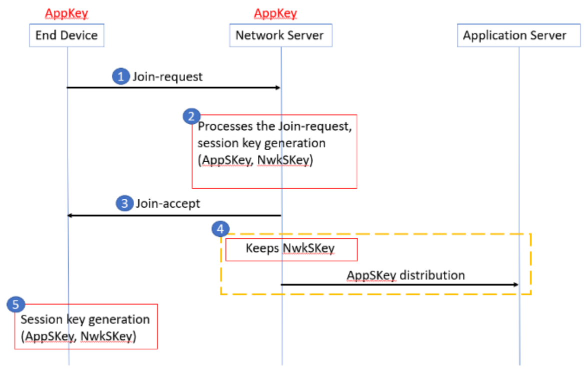


Figure 3.18. Join process with OTAA, between end device and TTN LoRaWAN/Network server [70]

To send data from the monitoring node to the TTN platform, LoRaWAN network coverage is required, and this coverage is achieved through the use of gateways. TTN offers the possibility for the any user, that is registered in TTN, to register gateways on the platform. As LoRaWAN operates in unlicensed bands, any user can install and register a gateway on TTN. These gateways are available for public access and can be used freely, under certain conditions, by other users. These conditions are related to the TTN Fair Use Policy. This policy declares limits to the number of uplink and downlink messages exchanged during the day, to create a fair communication environment for each user. There is a parameter called "consumed airtime" that indicates how long it took for the message to reach the gateway. Taking this parameter into consideration, the Fair Use Policy indicates the following limit: for the uplink messages, 30 seconds of airtime per day (24 hours), per node and for the downlink, 10 messages per day, per node [71]. The users that want to use the public TTN LoRaWAN network, must try to meet these conditions.

The main reason for selecting TTN as the LoRaWAN infrastructure to be used, is the number of gateways dispersed around the city of Lisbon, offering LoRaWAN coverage in several areas. A map of the gateways registered in the TTN can be found on the TTN website [72].

As mentioned in Section 2.1.6.3, the ADR mechanism allows control over certain parameters relating to LoRa communication (spreading factors, bandwidth, transmission power), depending on the current network conditions. TTN uses ADR by default and in the present system this mechanism is being considered. This way, the parameters mentioned above are automatically adapted.

The information that reaches the TTN comes in binary format and is decoded, by default, into hexadecimal format. To better understanding of the data, a customised JavaScript formatter has been created that can be inserted into TTN to decode the readings received and convert it into a structured and human-readable format. The code used for this task is in Appendix D.

3.4. Application Layer

This section describes the integration of a programmable tool, Node-Red, for using certain APIs that support the system, processing the readings from the monitoring node and storing them in a database. Finally, the web/mobile application used to visualise and analyse the data is presented.

3.4.1. Node-Red

Node-Red is a programmable tool based on flows, which allow the integration of APIs, online services, and the linking of physical devices with programmable nodes. It is built on Node.js and allows rules to be created from JavaScript scripts [73]. The Node-Red instance can run locally on a personal computer, on cloud platforms or on devices such as Raspberry Pi. In the system developed, Node-Red was installed on the Raspberry Pi 4 Model B, so that its instance can run continuously. One drawback of some cloud platforms is the need to pay after the trial period. As the Raspberry Pi is a personal component and easy to configure, it was the choice for running the Node-Red instance. The Node-Red instance running on the Raspberry Pi integrates a few APIs and online services for receiving, sending, and processing data and storing it in a Firebase database.

Once the sensor readings have reached the TTN, they are forwarded to the Node-Red using the MQTT protocol. This is a lightweight protocol based on publish/subscribe messages that don't require a large amount of bandwidth [74]. Since the data coming from the monitoring node are only sensor values, they don't need much bandwidth to be transmitted, therefore, MQTT protocol is the ideal protocol for transmitting this data. There are two key elements for running this protocol, MQTT client and MQTT broker. The broker is responsible for forwarding messages from clients to other clients. Clients, on the other hand, can publish messages with a particular topic and if another client wants to receive that message, they must subscribe to that topic.

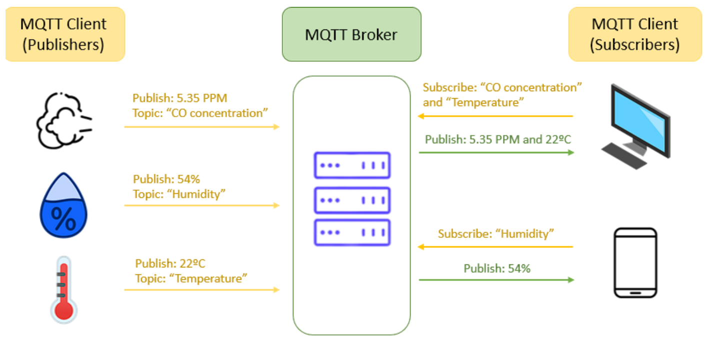

MQTT Client (Publishers)
MQTT Broker
MQTT Client (Subscribers)
Publish: 5.35 PPM Topic: "CO concentration"
Publish: 54% Topic: "Humidity"
Publish: 22ºC Topic: "Temperature"
Subscribe: "CO concentration" and "Temperature"
Publish: 5.35 PPM and 22ºC
Subscribe: "Humidity"
Publish: 54%

Figure 3.19. MQTT communication example between MQTT clients and MQTT broker

In the Appendix E, is shown the workflow created in Node-Red for this system, and on the table 3.6, there is a description about the "Function" nodes created in the workflow. The workflow is read from left to right and from top to bottom.

Table 3.6. Description of each function node used in Node-Red workflow

Function nodes in the workflow	Description
Set Sensor Topic	Sets the propriety "topic" as a string with the devEUI from the Arduino MKRWAN board
Get Values	Creates payload fields for each value from the monitoring node (for example, msg.temperature, msg.humidity, msg.pm2_5, msg.lat,…)
Formatting URL	Formats a web URL to make a http request to the TomTom Traffic API
Giving Thresholds for traffic	Gathers the value from two payload fields retrieved from the traffic API, *currentSpeed* and *freeFlowSpeed*; with these two values, new frequency to send readings through LoRa, according to traffic levels, is set
Setting String "Location"	Gathers the location address obtained from "Getting location" node, and an address is made with some of the fields from the node "Getting location"
Measure Timestamp	Creates a separate time and date field from a timestamp field
Send packet	Creates a packet with the new sending readings frequency, to send via downlink to the monitoring node
Insert on Firebase	Creates a customized payload with the different fields associated with measurements of the monitoring node, location address, date, and time of measurement
creating docID	Creates a custom document ID, based on time and date fields, to store the payload created by "Inserto n Firebase" node, on the Firestore Database

TTN supports the MQTT protocol, which makes communication between TTN and Node-Red easier. Two parameters are required to establish MQTT communication between TTN and Node-Red, a username and an api key. The username is related to the ID used in the TTN application and the api key can be generated automatically, in TTN. In Node-Red, the only thing required is to integrate an "MQTT IN" (with the label "Values from TTN", on the workflow presented above) node into the workflow and insert the parameters mentioned above so that the sensor readings are received in that instance. Some APIs are used in this flow to convert coordinates to an exact address (Google Geolocation, represented in the workflow with the "Getting location" node), to obtain certain atmospheric parameters (OpenWeatherMap, represented with the "Get weather description" node) and acquire speed of urban traffic on a certain section of a given location (TomTom Traffic API). From the TomTom API, the functionality of changing the frequency of sending sensor readings was created. This API returns the current average speed of the urban traffic flow and the speed expected under ideal conditions, on a section of a specific location. The further away the current average speed value is from the speed expected under ideal conditions, the more urban traffic there is at the location where the monitoring is happening. The flowchart shown in Figure 3.20 explains the logic behind the implementation of the mechanism for changing the periodicity of sending readings, inside the node “Giving Thresholds for traffic”:

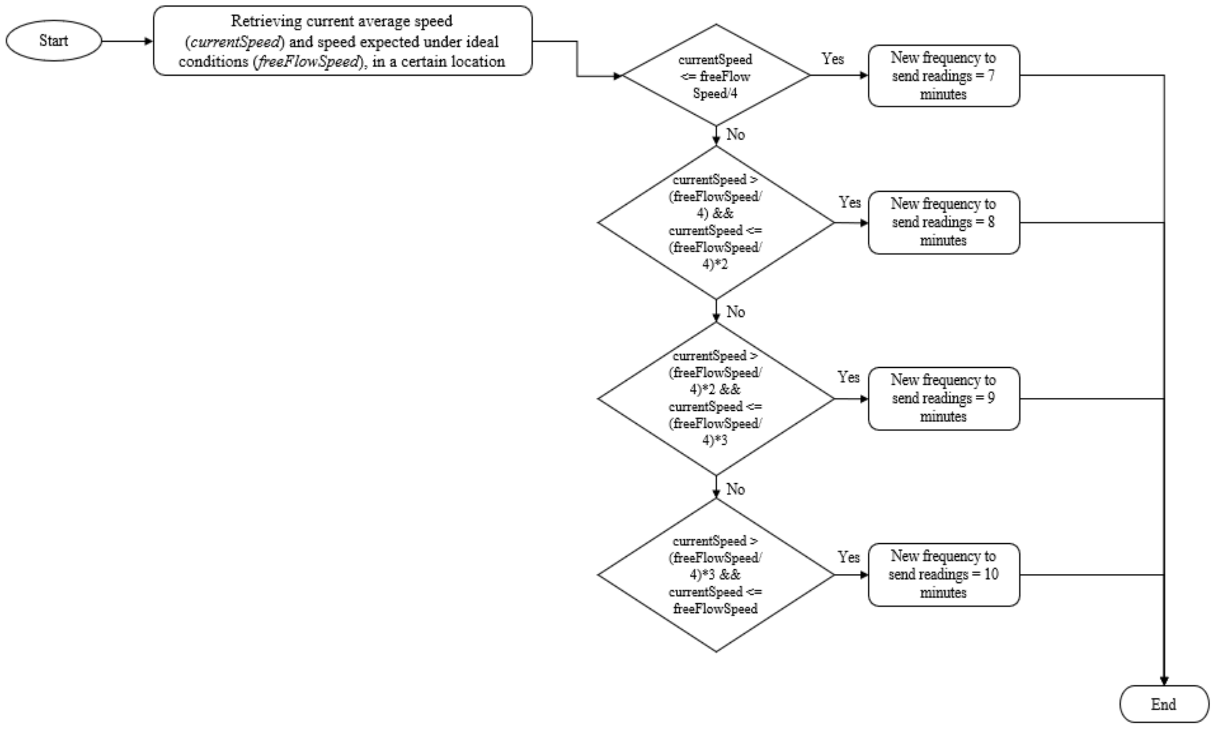

Start
Retrieving current average speed (currentSpeed) and speed expected under ideal conditions (freeFlowSpeed), in a certain location
currentSpeed <= freeFlow Speed/4
Yes
New frequency to send readings = 7 minutes
No
currentSpeed > (freeFlowSpeed/ 4) && currentSpeed <= (freeFlowSpeed/ 4)*2
Yes
New frequency to send readings = 8 minutes
No
currentSpeed > (freeFlowSpeed/ 4)*2 && currentSpeed <= (freeFlowSpeed/ 4)*3
Yes
New frequency to send readings = 9 minutes
No
currentSpeed > (freeFlowSpeed/ 4)*3 && currentSpeed <= freeFlowSpeed
Yes
New frequency to send readings = 10 minutes
End

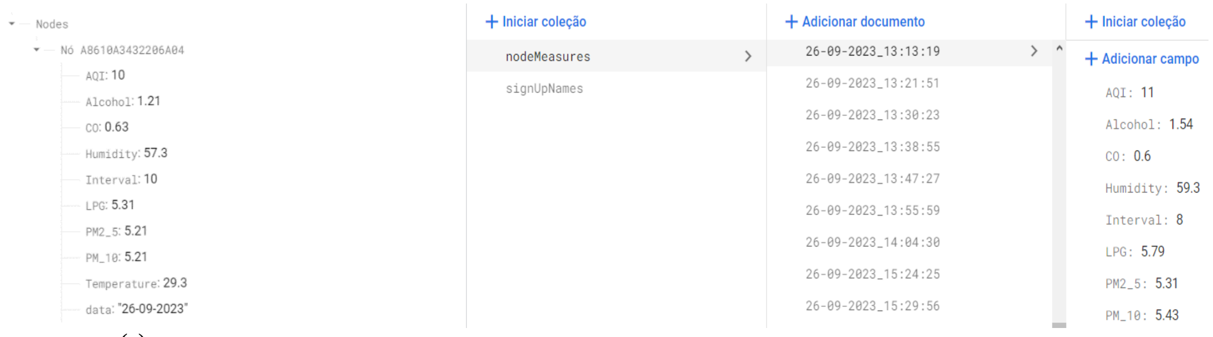
Figure 3.20. Changing sending readings frequency flowchart

The possible periodicities for this system are 7, 8, 9 and 10 minutes. These values were chosen to fulfil the conditions of the TTN Fair Use Policy, mentioned in the Section 3.3. To calculate these values, an Airtime calculator for LoRaWAN [75] was used. For this calculation, it was considered a SF of 8 and a payload size of 24 bytes (considering the amount of data to be sent from the monitoring node), plus 13 bytes for packet overhead. The number of uplink messages that could be sent was 8.7 messages per hour, which means that the minimum periodicity is every 7 minutes to send messages through LoRa. In other words, anything longer than 7 minutes fulfils the requirements of the Fair Use Policy, for the respective SF, and to create some interval levels for urban traffic, periodicities between 7 and 10 minutes were considered, associated with the highest and lowest levels of urban traffic, respectively. This value is then compared with the previous periodicity value, which is also sent together with the sensor readings to the TTN. This is carried out in the "Check interval" node. If the value of the old periodicity is different from the new periodicity, it means that there has been a change in the level of traffic, and a downlink message is sent to the monitoring node.

The "Send packet" function node creates the packet with the new frequency to send readings and the "Arduino MKR WAN 1300" node sends this packet to the TTN via the MQTT protocol, which is then sent to the available gateways within range, and finally to the monitoring node. By doing this, the node changes the frequency at which readings are sent.

Sensor readings and other parameters (measurement site address, frequency of sending readings, weather conditions) are stored in Firebase. Firebase is a set of Google computing services that host databases and other services in the cloud [44]. There are two databases hosted by this platform, that were used, the Realtime Database and the Firestore Database. Realtime Database is used to store the most recent readings and Firestore is based on a document-organised database, where each document is a key-value pair that can be grouped into collections.

In the "Insert on Firebase" node, the readings and other parameters obtained by the API's are combined to create the body of the payload, which is stored in the Realtime Database via the "UPDATE" node. Each time new readings arrive, the "UPDATE" node updates the value of the fields. From the "creating docID" node, an ID is created for the document of the readings, based on the time and date of the reading, which is then stored in the Firestore Database, inside “nodeMeasures” collection, via the "Add values in Firestore 2" node. Each document is associated with a set of readings at a given time.


Nodes
Nó A8610A3432206A04
AQI: 10
Alcohol: 1.21
CO: 0.63
Humidity: 57.3
Interval: 10
LPG: 5.31
PM2_5: 5.21
PM_10: 5.21
Temperature: 29.3
data: "26-09-2023"

(a)

+ Iniciar coleção
nodeMeasures
signUpNames

+ Adicionar documento
26-09-2023_13:13:19
26-09-2023_13:21:51
26-09-2023_13:30:23
26-09-2023_13:38:55
26-09-2023_13:47:27
26-09-2023_13:55:59
26-09-2023_14:04:30
26-09-2023_15:24:25
26-09-2023_15:29:56

+ Iniciar coleção
+ Adicionar campo
AQI: 11
Alcohol: 1.54
CO: 0.6
Humidity: 59.3
Interval: 8
LPG: 5.79
PM2_5: 5.31
PM_10: 5.43

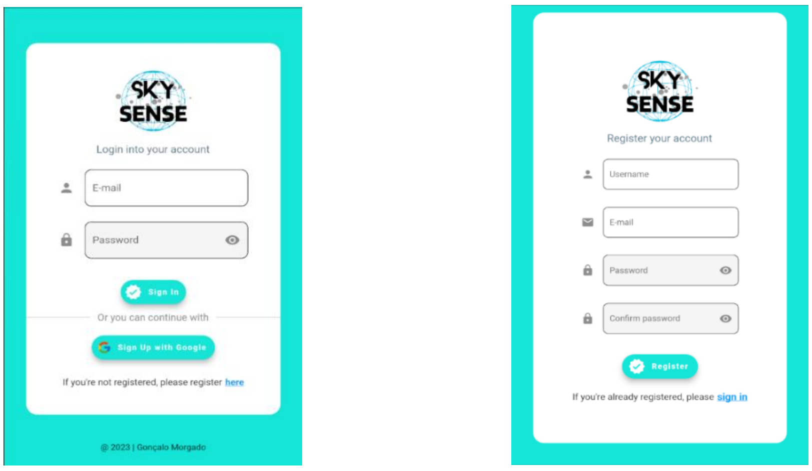
(b)

Figure 3.21. Monitoring node values in the RealTime database (a) and all the collected measures in Firestore database, inside de “nodeMeasures” collection (b)

3.4.2. Web/Mobile Application

The data obtained by the monitoring node is visualised and analysed using a web and Andoid application. The "SkySense" application was developed in Flutter, a framework produced by Google that allows applications to be built from a single code base and is based on the Dart programming language [76]. Another option considered when developing the application was using a combination of Javascript for the backend and React for the frontend. The fact that all the programming can be done from native code and the ease with which applications can be created on different platforms was the reason for choosing Flutter, over other options, to create this app.

When the user accesses the application, they are asked to log in. If the user is not already registered, they can do so, but they also have the option of logging into the application with their Google account (if they have one), as can be seen in Figure 3.22:



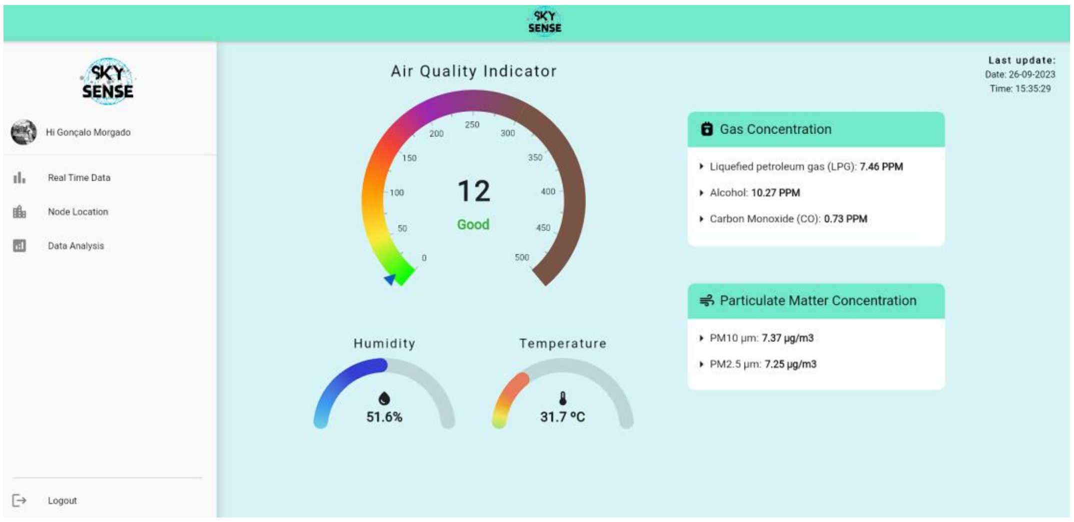
(a) (b)

Figure 3.22. Application Login page (a) and Register page (b)

As Firebase offers authentication services, these were used to create the login and user registration functionalities in the application. Gathering data from the Firebase databases, authentication and other functional and visual elements are achieved by installing packages, provided by Flutter, which allow diverse functionalities to be built.

After logging in or signing up, the user enters the application's main page relating to the monitoring node's real-time data. This page shows data about an air quality indicator, temperature and relative humidity, the concentrations of different pollutants (carbon monoxide (CO), liquefied petroleum gas (LPG), alcohol), $PM_{2.5}$ and PM_{10} concentration and the time and date of the last reading from the monitoring node. The numerical value displayed in the application, referring to “Air Quality Indicator”, relates to an analog value obtained by the Grove Air Quality sensor, and therefore can't be considered the true AQI value. It is merely an indicator of how polluted the air is, when monitoring is being conducted, and as mentioned in section 3.2.2, the higher the value displayed, the lower the air quality.

All the data presented in this page is updated every time new readings arrive in the Realtime Database.



Last update:
Date: 26-09-2023
Time: 15:35:29

12
Good

Gas Concentration
- Liquefied petroleum gas (LPG): 7.46 PPM
- Alcohol: 10.27 PPM
- Carbon Monoxide (CO): 0.73 PPM

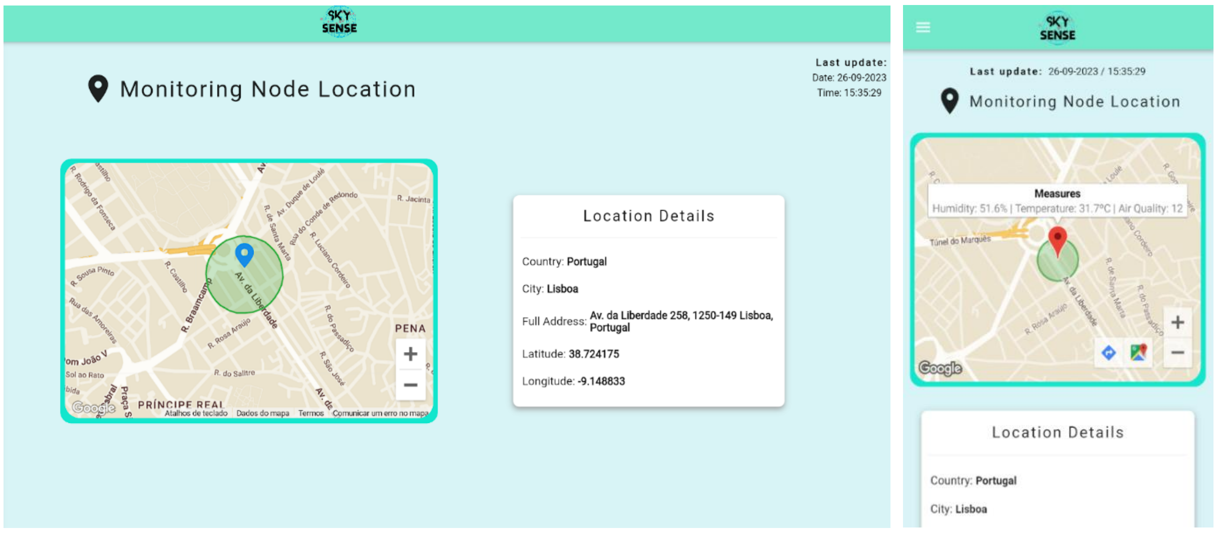
Particulate Matter Concentration
- PM10 µm: 7.37 µg/m3
- PM2.5 µm: 7.25 µg/m3

51.6%

31.7 ºC

Figure 3.23. "Real Time Data" page, from web application view

From the drawer on the left-hand side of the application page (Figure 3.23), the user can navigate between the "Real Time Data", "Node Location" and "Data Analysis" pages. The "Node Location" page contains information on the location of the monitoring node, such as latitude, longitude, site address, city, and country. Just like the "Real Time Data" page, it is possible to see the date and time of the monitoring node's latest readings.

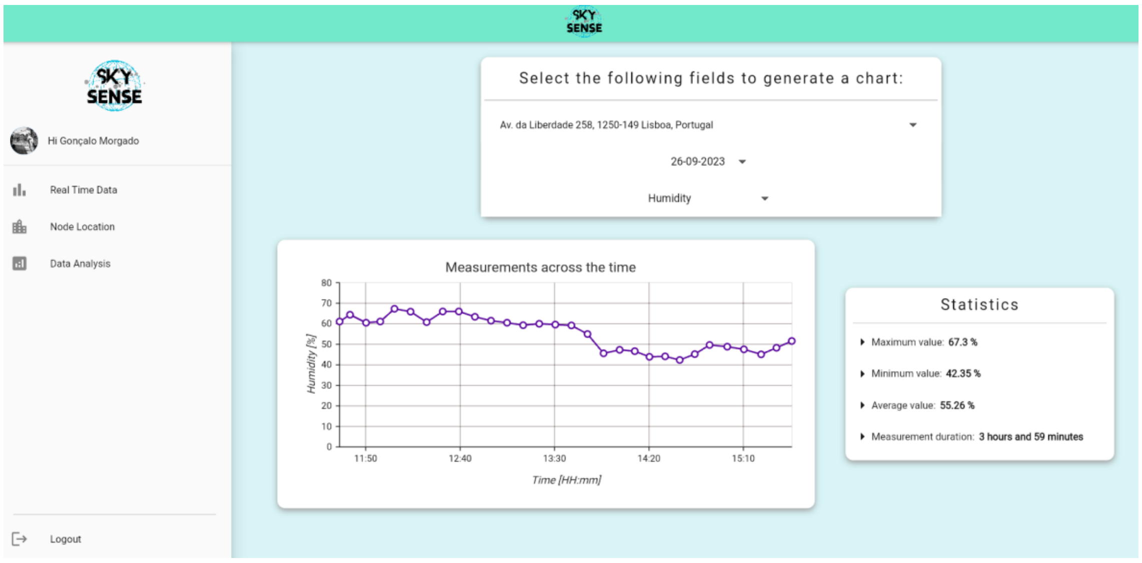


(a) (b)

Figure 3.24. "Node Location" page from web (a) and Android (b) view

The map used to visualise the location is provided via the Google Maps API. The marker marks the exact location of the node and around it is a circle that updates its colour depending on the air quality level. If the user clicks on the marker, they can also get readings of some parameters such as temperature, relative humidity, and the air quality indicator value (Figure 3.24 (b)).

On the "Data Analysis" page, the user can choose to carry out a temporal analysis of the measured parameters based on the location and date at which the monitoring node has been taking measurements. Firstly, a location must be selected from the "Select a location" field, and only when this location has been selected, a second field, "Select a date", appears for the user to fill in the date on which the measurements were taken. Finally, the user chooses the parameter they want to analyse (temperature, relative humidity, CO, LPG, Alcohol, $PM_{2.5}$, PM_{10}, air quality value) from the "Select a measurement" field and only when this field is selected a chart is generated showing all the measurements taken by the monitoring node. Statistics such as the maximum, minimum and average values, and the time the node has been monitoring that location, can also be found on this page.



Select the following fields to generate a chart:

Av. da Liberdade 258, 1250-149 Lisboa, Portugal

26-09-2023

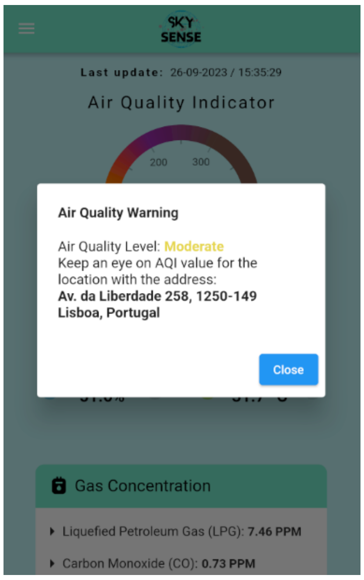
Humidity

Statistics

- Maximum value: **67.3 %**
- Minimum value: **42.35 %**
- Average value: **55.26 %**
- Measurement duration: **3 hours and 59 minutes**

Figure 3.25. "Data analysis" page from web view

A final feature of this application is the display of warnings regarding the level of the air quality. The Grove Air Quality sensor library contains a file with thresholds for the quantitative value, used to return a qualitative value indicating the level of air quality (e.g., "Fresh Air", "Low Pollution", "High Pollution" ...). Some of the thresholds presented in the file, regarding the quantitative value obtained by the Grove Air Quality sensor, are similar to those imposed by EPA, and therefore the air quality levels presented on the application's "Real Time Data" page were based on the AQI levels declared by EPA [18]. Each time the air quality changes its level, the user receives a warning in the application with the current air quality level and the specific location where the level has changed.



Last update: 26-09-2023 / 15:35:29

Air Quality Indicator

Air Quality Warning

Air Quality Level: Moderate
Keep an eye on AQI value for the location with the address:
Av. da Liberdade 258, 1250-149 Lisboa, Portugal

Gas Concentration

- Liquefied Petroleum Gas (LPG): **7.46 PPM**
- Carbon Monoxide (CO): **0.73 PPM**

Figure 3.26. Air Quality Level Warning mechanism, from Android view

CHAPTER 4

Results and Discussion

This chapter presents the results obtained from the monitoring tests carried out by the monitoring node. Several tests were conducted, the first being associated with the calibration of the sensors when they are capturing the same gas inside a gas test chamber. Afterwards, measurements were taken outdoors, in different locations, for a later comparison between the readings obtained by the sensors and the surrounding environment. Finally, the energy consumption of the monitoring node is presented when it is in operation and in low energy consumption mode.

4.1. Gas Test Chamber

To test the calibration of the sensors, mentioned in Section 3.2.1.1, and their sensitivity when they are exposed to a particular gas, a test was conducted in a gas test chamber. The component used for this task was the Figaro SR-3 Bench Top Test Chamber. It is a box made of acrylic resin that allows the gas under study to be injected by inserting a syringe with the desired gas through a gas inlet in the chamber [77]. The sensor can be placed inside the chamber to measure the gas concentration being considered. Inside the box there is a fan which, when switched on, spreads the gas so that a uniform concentration is achieved throughout the test chamber [77]. This test chamber has a maximum capacity of 5,400 ml of injected gas.

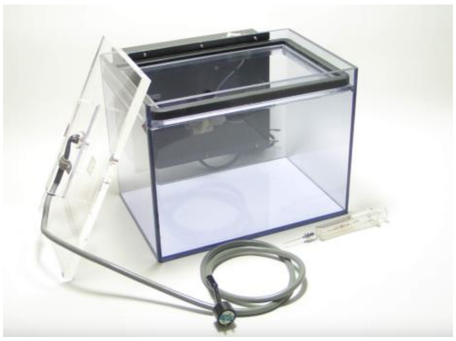


Figure 4.1. Figaro SR-3 Test Chamber

The MQ-2, MQ-7 and MQ-9 sensors were calibrated to detect concentrations of alcohol, carbon monoxide (CO) and liquefied petroleum gas (LPG), respectively. Short-term inhalation of CO and LPG, especially CO, is very dangerous and can cause death within a few moments. Ethanol (ethyl alcohol) is the only safer gas to test, so only the results from the MQ-2 sensor were considered for this experiment.

In the test chamber's datasheet, reference is made to the equation used to determine the volume of gas needed to be injected to reach a particular concentration of the gas being tested:

$$Volume\ [ml] = 5{,}400\ ml\ \times \frac{Gas\ concentration\ [ppm]}{1{,}000{,}000} \tag{9}$$

The injected gas volume is given in millilitres (ml) and the gas concentration is given in parts per million (ppm). Three tests were carried out, both with different concentrations of injected alcohol. Equation 9 gives the expected concentration for the different amounts of alcohol:

- 1st Test: 1 ml of alcohol ≈ 185.19 ppm;
- 2nd Test: 1.5 ml of alcohol ≈ 277.78 ppm;
- 3rd Test: 2 ml of alcohol ≈ 370.37 ppm.

Both experiments lasted 15 minutes, and at the end of each test, more alcohol was added to the test chamber, except for the last case. The alcohol concentration obtained by the MQ-2 sensor is obtained by the process presented in section 3.2.1.1. The chart in Figure 4.2 shows the results obtained:

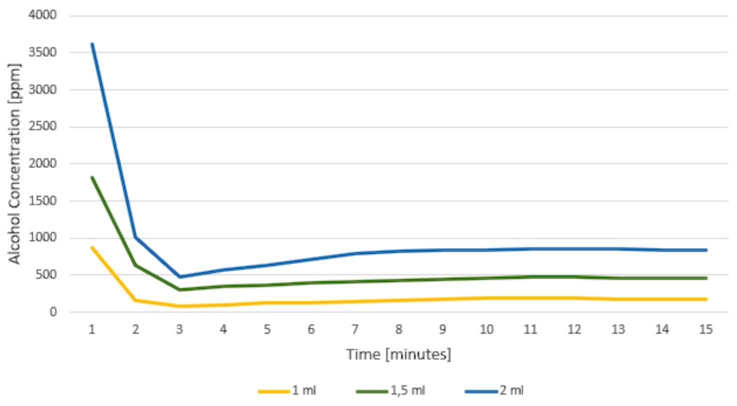


Figure 4.2. Alcohol concentration behaviour, across the time regarding different alcohol volumes injected

During the test, it was noted that there was a change in the concentration obtained by the other MQ gas sensors. The presence of other gases, apart from the gas being studied (such as the presence of ethanol in the detection of CO and LPG), can affect the MQ sensors' reaction and cause incorrect inductions about the gas concentration.

In the first instant, the alcohol was injected into the test chamber and for this reason, a very high concentration value was captured. The built-in fan was switched on for 2 minutes at the start of each test to disperse the alcohol throughout the chamber, which is why there was a sudden drop in concentration initially. From the third minute onwards, the alcohol was spread throughout the chamber and the sensor readings began to stabilise. At around 10-11 minutes, the concentration value was stabilised. Values obtained by the sensor were expected, considering the values obtained by equation 7. As soon as the 15 minutes of a test were up, the volume associated with the next test was inserted, i.e., the volume injected into the test chamber was being accumulated. The total amount injected into the chamber was 4.5 ml, which corresponds to a concentration of approximately 833.33 ppm. As can be seen in the last test, the stabilised value is practically equivalent to the alcohol concentration mentioned above.

4.2. Outdoor Monitoring Tests

Three different locations were chosen to test the monitoring node in an outdoor context, to understand what influence the surrounding environment has on the readings obtained by the sensors. The locations selected to carry out the tests are associated with Lisbon city centre, in Portugal. Figure 4.3 shows the locations chosen for the measurements:

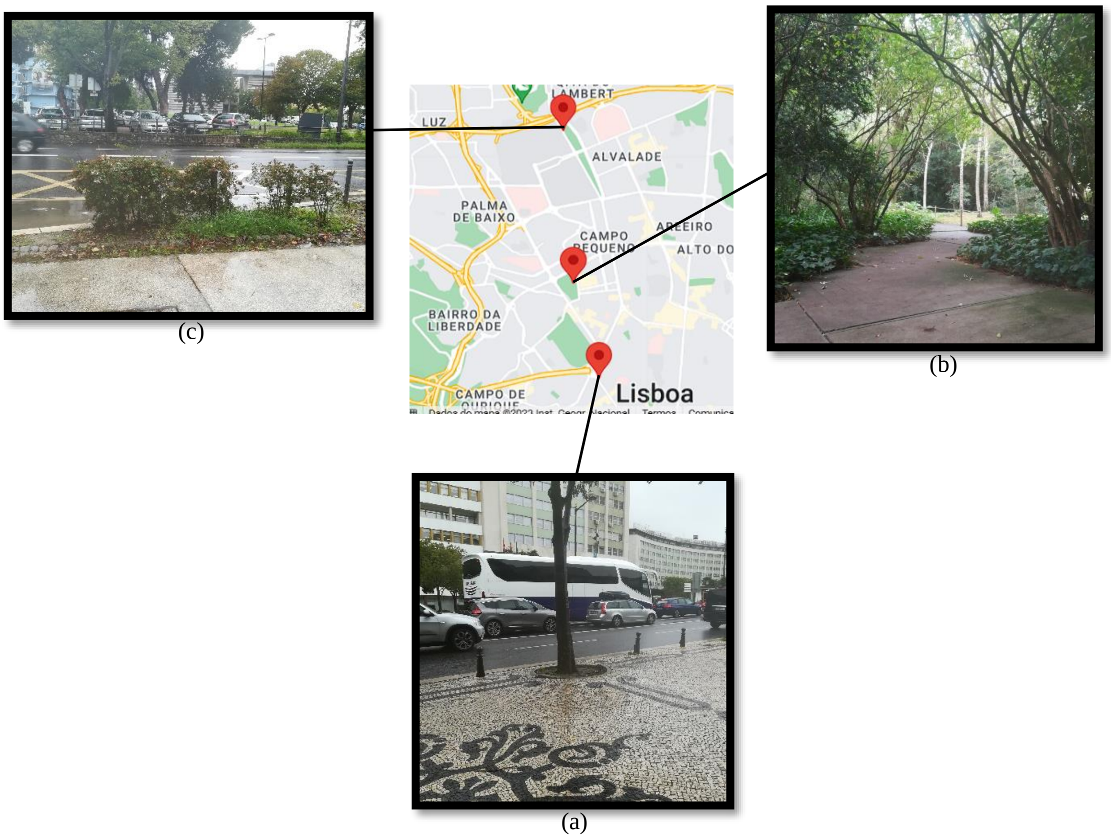


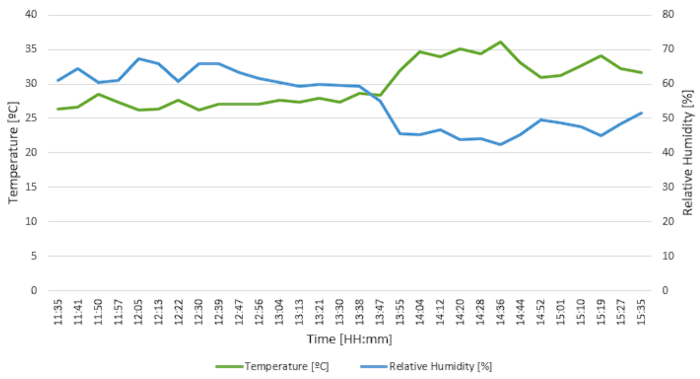
Figure 4.3. Locations where the monitoring has taken place: Avenida da Liberdade (a), Jardim da Fundação Calouste Gulbenkian (b) and Campo Grande (c)

The first location (Av. da Liberdade 258, 1250-149 Lisboa, Portugal – Figure 4.3 (a)) relates to an avenue in the centre of Lisbon, more specifically, near the Marquês de Pombal roundabout. The second location (Av. de Berna 45, 1050-078 Lisboa, Portugal – Figure 4.3 (b)) is the garden of the Calouste Gulbenkian Foundation. Finally, the last location where tests were conducted was an avenue near the Mário Soares Garden, Campo Grande (Campo Grande C3, 1700-162 Lisboa, Portugal – Figure 4.3 (c)).

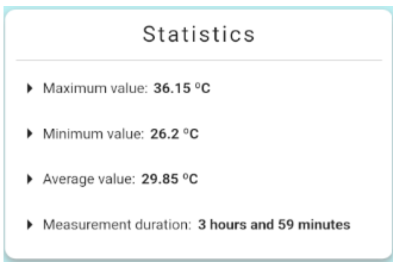
It should be noted that the monitoring performed for each of the previously mentioned locations was carried out in different days, but the tests lasted the same amount of time, approximately 4 hours for each location. Due to the impossibility and lack of security in leaving the monitoring node in an outdoor location, measurements could not be taken over a long period of time.

4.2.1. 1st Location – Avenida da Liberdade

At this location, measurements were taken between 11:35 and 15:35. During the day, the weather was clear, sunny, with low wind intensity and there was no rainfall. The weather conditions can be obtained by the OpenWeatherMap API integrated on the Node-Red instance, but also on the OpenWeatherMap website. It should be noted that the measurements were taken on a weekday, which is favourable for urban traffic.



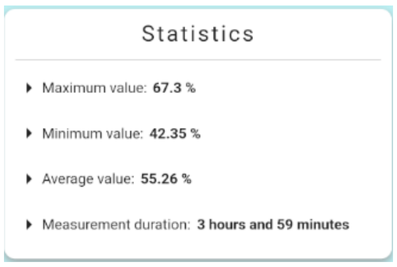
(a)



Statistics

- Maximum value: **36.15 ºC**
- Minimum value: **26.2 ºC**
- Average value: **29.85 ºC**
- Measurement duration: **3 hours and 59 minutes**

(b)



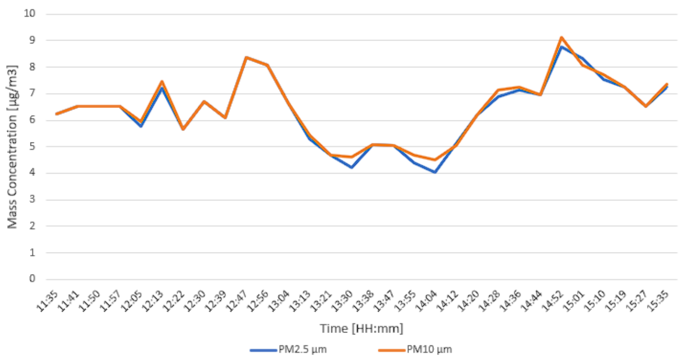
Statistics

- Maximum value: **67.3 %**
- Minimum value: **42.35 %**
- Average value: **55.26 %**
- Measurement duration: **3 hours and 59 minutes**

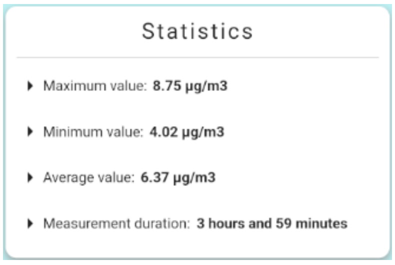
(c)

Figure 4.4. Temperature and Relative Humidity variation during the monitoring period (a), in the 1st location; statistics about Temperature (b) and Relative Humidity (c)

During the morning, the relative humidity was above 60%, but from 12:39 onwards, the humidity level began to fall, with a significant drop from 13:38 onwards. From this moment on, there was a greater incidence of sunlight and, consequently, a gradual increase in temperature, reaching its maximum value at 14:36. It was between the period of 13:38 and 15:35 that the temperature showed the highest values and, consequently, the relative humidity showed the lowest ones.

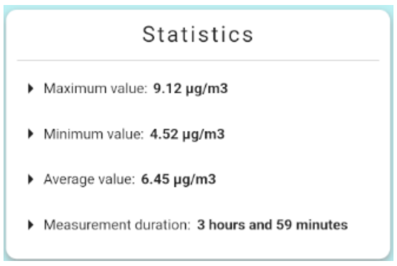


(a)


Statistics

- Maximum value: **8.75 μg/m3**
- Minimum value: **4.02 μg/m3**
- Average value: **6.37 μg/m3**
- Measurement duration: **3 hours and 59 minutes**

(b)


Statistics

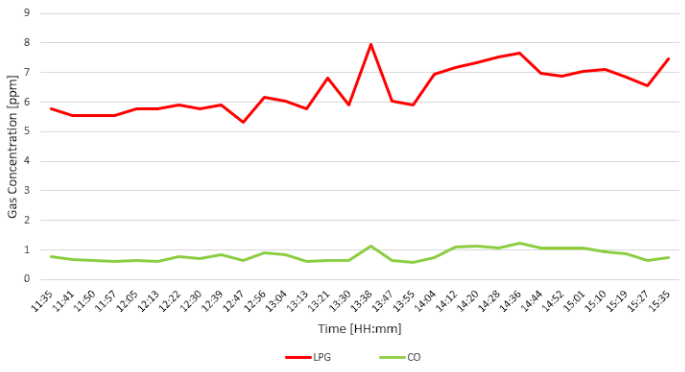
- Maximum value: **9.12 μg/m3**
- Minimum value: **4.52 μg/m3**
- Average value: **6.45 μg/m3**
- Measurement duration: **3 hours and 59 minutes**

(c)

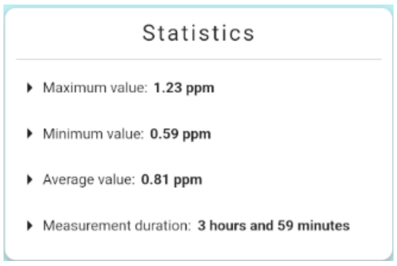
Figure 4.5. $PM_{2.5}$ and PM_{10} variation across the monitoring period (a), in the 1st location; statistics about $PM_{2.5}$ (b) and PM_{10} (c) concentration

Particulate matter concentrations, $PM_{2.5}$ and PM_{10}, throughout the monitoring period showed practically the same behaviour. Near the location where the measurements were taken, there was a construction site, which may have influenced the concentration of these particles in the air, as well as the number of vehicles travelling along the avenue. At 13:30 and 14:04, the concentration of particulate matter showed the lowest values. At 12:47 and 14:52, the highest values were found.

On this day, the wind speed was minimal, which might have contributed to the particles not being so widely spread through the air, resulting in an accumulation of particulate matter in certain places, thus increasing their concentration, although there were slight wind breezes present at certain times. As a result, the particles can disperse more easily through the air. Considering this fact and the density of traffic flow on the avenue, it can explain the large variations that occurred during the monitoring period.

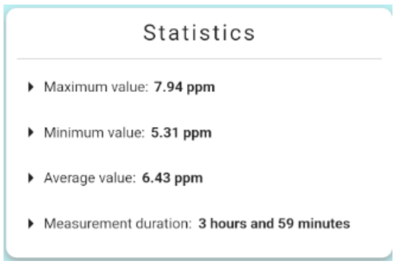


(a)


Statistics

- Maximum value: **1.23 ppm**
- Minimum value: **0.59 ppm**
- Average value: **0.81 ppm**
- Measurement duration: **3 hours and 59 minutes**

(b)


Statistics

- Maximum value: **7.94 ppm**
- Minimum value: **5.31 ppm**
- Average value: **6.43 ppm**
- Measurement duration: **3 hours and 59 minutes**

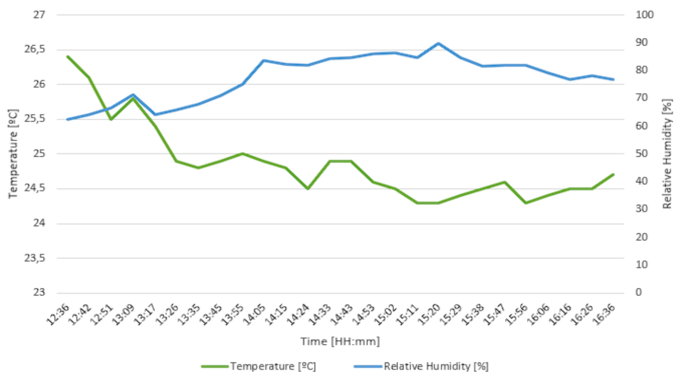
(c)

Figure 4.6. CO and LPG variation across the monitoring period (a), in the 1st location; statistics about CO (b) and LPG (c) concentration

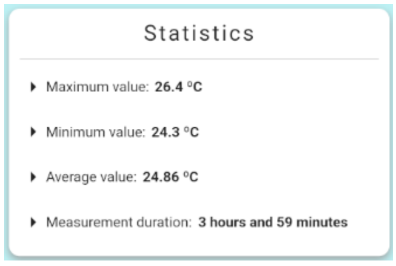
The concentration of pollutant gases such as CO and LPG showed an increasing behaviour for most of the monitoring period. The variations found are related to the number of vehicles in circulation at the site, and since there is a traffic light at the top of the avenue, the spikes in values can be explained by this fact. The LPG values vary more than the CO values, but both have peak values at 13:38 and 14:36.

4.2.2. 2nd Location – Jardim da Fundação Calouste Gulbenkian

The monitoring in this location was conducted between 12:36 and 16:36. During the monitoring period, the weather was quite cloudy, with a tendency for precipitation. Figure 4.7 shows the temperature and relative humidity values obtained during the monitoring period:

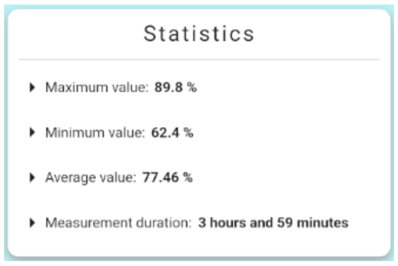


(a)



Statistics

- Maximum value: **26.4 ºC**
- Minimum value: **24.3 ºC**
- Average value: **24.86 ºC**
- Measurement duration: **3 hours and 59 minutes**



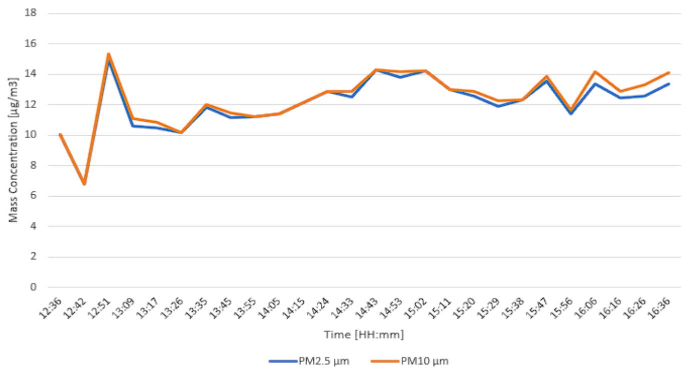
Statistics

- Maximum value: **89.8 %**
- Minimum value: **62.4 %**
- Average value: **77.46 %**
- Measurement duration: **3 hours and 59 minutes**

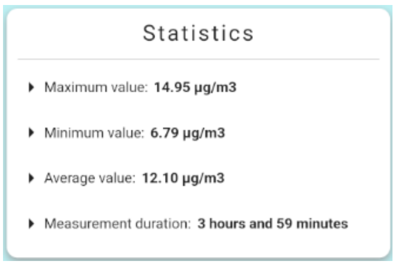
(b) (c)

Figure 4.7. Temperature and Relative Humidity variation during the monitoring period (a), in the 2nd location; statistics about Temperature (b) and Relative Humidity (c)

The temperature shows an inverse behaviour to the relative humidity, which is expectable given the temperate climate in Portugal. Throughout the monitoring period, it became increasingly cloudy, and the wind speed also slowly increased, which contributed to a gradual decrease in temperature and an increase in relative humidity. Around 15:20, the relative humidity reached its maximum value at approximately 90%. At this point, very light raindrops were felt, which explains the high humidity value.

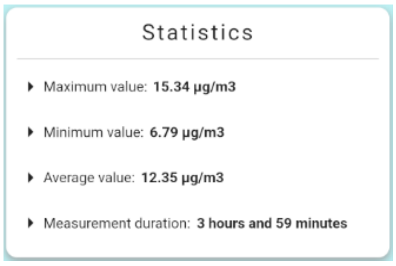


(a)


Statistics

- Maximum value: **14.95 µg/m3**
- Minimum value: **6.79 µg/m3**
- Average value: **12.10 µg/m3**
- Measurement duration: **3 hours and 59 minutes**

(b)

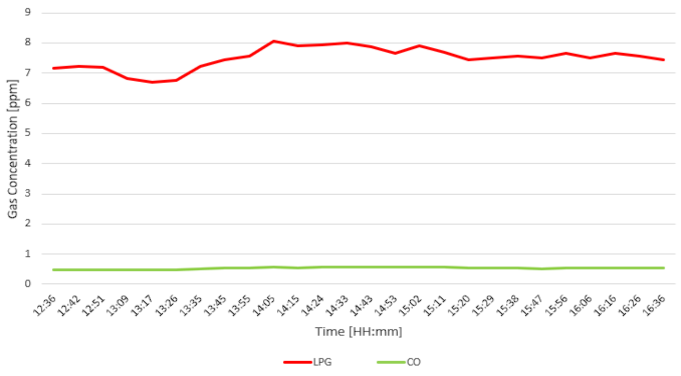

Statistics

- Maximum value: **15.34 µg/m3**
- Minimum value: **6.79 µg/m3**
- Average value: **12.35 µg/m3**
- Measurement duration: **3 hours and 59 minutes**

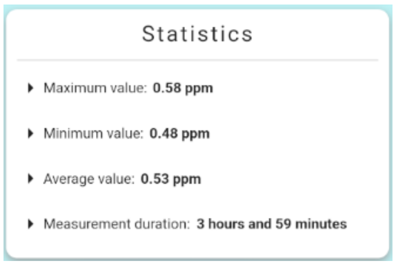
(c)

Figure 4.8. $PM_{2.5}$ and PM_{10} variation across the monitoring period (a), in the 2nd location; statistics about $PM_{2.5}$ (b) and PM_{10} (c) concentration

During the measurements, there were some variations in the particulate matter concentration in the garden. These variations may be explained by the fact that there were several people walking around, and since there were some dirt trails within the garden, the concentration of particulate matter could vary depending on the number of people walking on these trails. The period between 12:42 and 12:51 was when the highest increase in particulate matter concentration was observed, and it was during this period that most people were moving around inside the garden. Factors such as the presence of wind and vegetation can also influence the dispersion and concentration of these particles at the area.

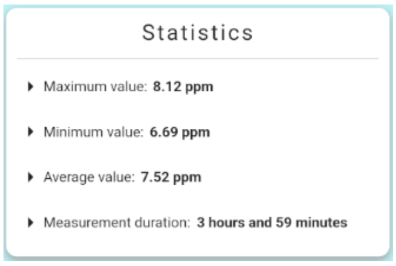


(a)


Statistics

- Maximum value: **0.58 ppm**
- Minimum value: **0.48 ppm**
- Average value: **0.53 ppm**
- Measurement duration: **3 hours and 59 minutes**

(b)


Statistics

- Maximum value: **8.12 ppm**
- Minimum value: **6.69 ppm**
- Average value: **7.52 ppm**
- Measurement duration: **3 hours and 59 minutes**

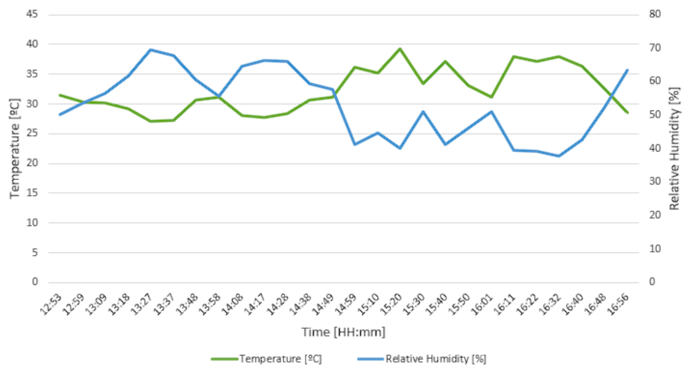
(c)

Figure 4.9. CO and LPG variation across the monitoring period (a), in the 2nd location; statistics about CO (b) and LPG (c) concentration

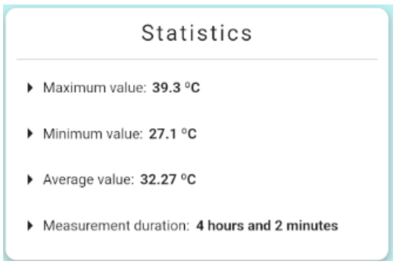
Regarding the concentrations of pollutant gases, particularly CO and LPG, only CO showed a practically constant behaviour during the monitoring period, unlike LPG, which exhibited slight variations. The tests were conducted in the central area of the garden, but the garden itself is located between several avenues. Despite the presence of large amounts of vegetation, wind speed can influence the distribution of gases in the air and generate some variations.

4.2.3. 3rd Location – Campo Grande

The readings obtained by the monitoring node relate to the period between 12:53 and 16:56 and were conducted during the weekend. During this day, there was no precipitation, the sky was slightly cloudy and there were periods of high solar incidence.

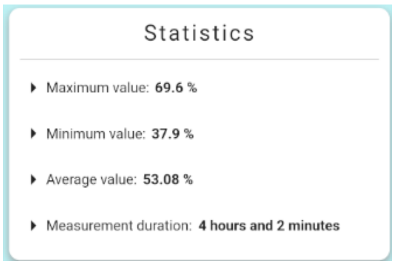


(a)



Statistics

- Maximum value: **39.3 ºC**
- Minimum value: **27.1 ºC**
- Average value: **32.27 ºC**
- Measurement duration: **4 hours and 2 minutes**



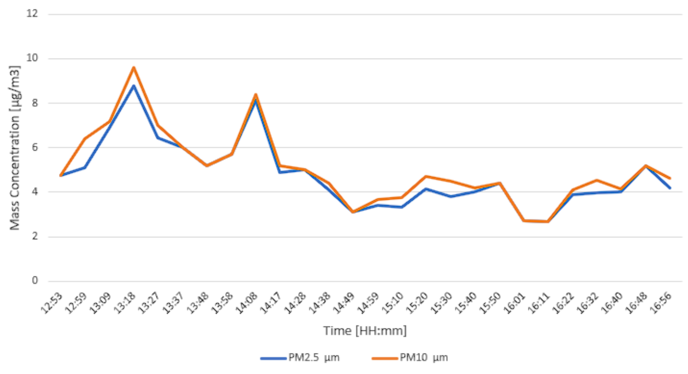
Statistics

- Maximum value: **69.6 %**
- Minimum value: **37.9 %**
- Average value: **53.08 %**
- Measurement duration: **4 hours and 2 minutes**

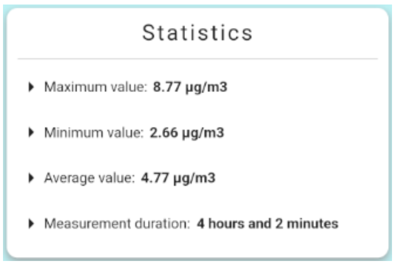
(b) (c)

Figure 4.10. Temperature and Relative Humidity variation during the monitoring period (a), in the 3rd location; statistics about Temperature (b) and Relative Humidity (c)

The chart shown in Figure 4.10 (a) demonstrates an almost symmetrical behaviour between temperature and relative humidity. Throughout the monitoring period, there were huge variations in the values of these two atmospheric parameters. Periods in which relative humidity showed higher values were related to periods when the weather was slightly cloudy. When sunlight was strongest, temperature values increased, peaking at around 15:20.

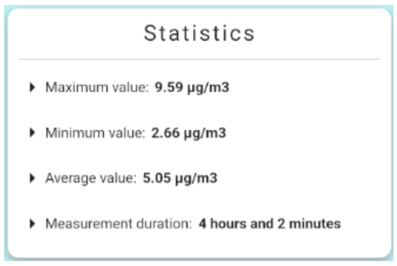


(a)


Statistics

- Maximum value: **8.77 µg/m3**
- Minimum value: **2.66 µg/m3**
- Average value: **4.77 µg/m3**
- Measurement duration: **4 hours and 2 minutes**

(b)


Statistics

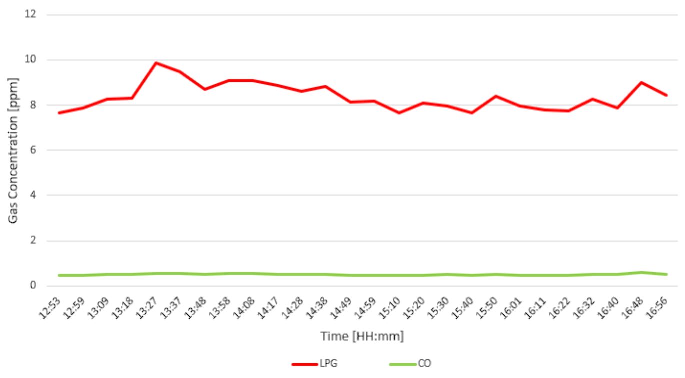
- Maximum value: **9.59 µg/m3**
- Minimum value: **2.66 µg/m3**
- Average value: **5.05 µg/m3**
- Measurement duration: **4 hours and 2 minutes**

(c)

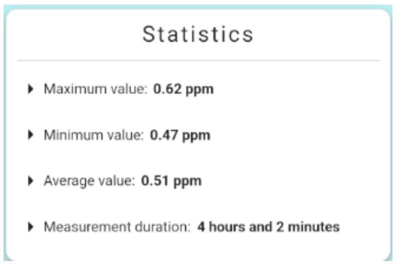
Figure 4.11. $PM_{2.5}$ and PM_{10} variation across the monitoring period (a), in the 3rd location; statistics about $PM_{2.5}$ (b) and PM_{10} (c) concentration

The concentration of particulate matter, both $PM_{2.5}$ and PM_{10}, at certain times, showed some variations during the monitoring period. In the chart shown in Figure 4.11 (a), there are two moments when the concentration of particulate matter increased significantly, specifically at 13:18 and 14:08. Since this location is near an avenue, there is a constant flow of vehicles. It should be noted that there is a traffic light controlling traffic at this location, and the spikes shown on the chart correspond to the times when traffic is most concentrated at the monitoring site, due to the action of the traffic light. As a result, the concentration of particulate matter is higher at the two instants mentioned above.

The concentration over the rest of the monitoring period was dependent on the traffic conditions along the avenue, which led to some variations. It should be noted that the period between 16:01 and 16:11 was the period when there was the least concentration of vehicles travelling, and therefore the least concentration of particulate matter.

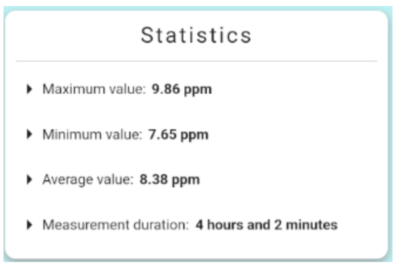


(a)


Statistics

- Maximum value: **0.62 ppm**
- Minimum value: **0.47 ppm**
- Average value: **0.51 ppm**
- Measurement duration: **4 hours and 2 minutes**

(b)


Statistics

- Maximum value: **9.86 ppm**
- Minimum value: **7.65 ppm**
- Average value: **8.38 ppm**
- Measurement duration: **4 hours and 2 minutes**

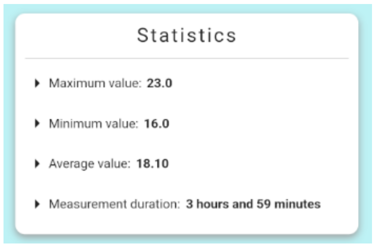
(c)

Figure 4.12. CO and LPG variation across the monitoring period (a), in the 3rd location; statistics about CO (b) and LPG (c) concentration

Similarly, to the second location, the CO concentration shows a practically constant behaviour throughout the monitoring period. On the other hand, the LPG concentration shows some variations, due to the traffic conditions mentioned above, with the maximum LPG concentration at 13:27.

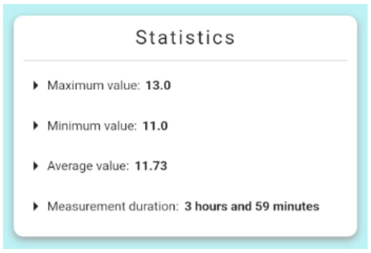
4.2.4. Air Quality Indicator values comparison

The air quality indicator obtained by the Grove Air Quality sensor changed according to the concentration of polluting gases in the locations where the monitoring was conducted. From the developed application, statistics about this index can be observed.


Statistics

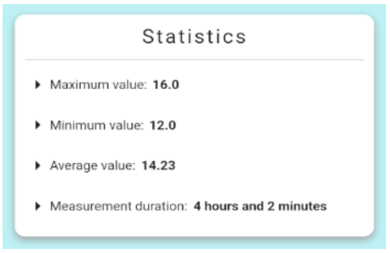
- Maximum value: **23.0**
- Minimum value: **16.0**
- Average value: **18.10**
- Measurement duration: **3 hours and 59 minutes**

(a)


Statistics

- Maximum value: **13.0**
- Minimum value: **11.0**
- Average value: **11.73**
- Measurement duration: **3 hours and 59 minutes**

(b)


Statistics

- Maximum value: **16.0**
- Minimum value: **12.0**
- Average value: **14.23**
- Measurement duration: **4 hours and 2 minutes**

(c)

Figure 4.13. Statistics about air quality indicator values in the 1st location (a), 2nd location (b) and 3rd location (c)

The measured air quality indicator value over the monitoring time at the first location (Avenida da Liberdade) had the highest average value. The fact that the monitoring on this avenue was carried out on a weekday was favourable for a higher urban traffic flow, which is a propitious factor for a higher concentration of pollutant gases and, consequently, an increase in the air quality indicator value. In the second location, the values were the lowest, mainly due to the nature of the environment where the monitoring was conducted. Regarding the third location, relating to the other avenue, the values obtained were lower compared to the first location, but higher when compared to the second location. Since this location is also an avenue, despite having some urban traffic, there wasn't as much when comparing it to the first location. The fact that the monitoring was carried out on a weekend contributed to less urban traffic, so it can be assumed that there wasn't as much concentration of pollutant gases, making the air quality indicator value a little lower compared to the first location.

4.2.5. Frequency of Readings

Over the monitoring period for the different locations, there were changes to the frequency at which readings were sent to the LoRaWAN server. Initially, a 5-minute interval was manually imposed between sending the first measurement and the second measurement. From the second measurement onwards, the sending frequency is adjusted according to urban traffic levels, by sending downlink messages from the Node-Red to the monitoring node, as explained in Section 3.4.1. In the second and third location, relating to the garden and one of the avenues, respectively, 25 measurements were taken, and the interval between them mainly ranged between 9 and 10 minutes, with some measurements also taken every 8 minutes. In the case of the 1st location, also referring to the other avenue, 29 measurements were taken.

The tests were conducted at both locations for the same length of time (4 hours), but more measurements were taken at the first location. The tests that were conducted at the first location were carried out on an avenue that is usually characterised by heavy urban traffic on weekdays. In comparison to the test that was conducted at the third location, during a weekend, even though it was on an avenue where there is also a lot of traffic, there wasn't as much urban traffic going through it. The choice of day to perform monitoring tests had an influence on the number of measurements taken by the monitoring node.

4.3. Monitoring Node Energy Consumption

Since the monitoring node incorporates several sensors, its energy consumption can be quite high. A study was conducted to analyse its energy consumption. The node is powered by a battery with a capacity of 10000 mAh (milliampere-hour) and to optimise its consumption, a low power mode was implemented in the monitoring node to lower its energy consumption and increase battery life. Measurements were taken when the low consumption mode is not present and when it is activated.

4.3.1. Monitoring Node Power Consumption

With all the sensors connected to the monitoring node, the energy consumption related to normal cycle of operation, was measured, i.e., current measurements were taken during the period where the monitoring node was taking measurements, sending them through LoRa and waiting to repeat the cycle.

Current measurements were taken using a Keithley 2110 Digit Multimeter, with the selected option "DCI" to measure DC current. The results obtained can be seen in the table below:

Table 4.1. Current consumption during the taking of measurements, measurements transmission through LoRa and delay period before sending other measurements, without and with Low Power Mode

	Current Drawn (*Amps*)		
	Sensors Measuring	**LoRa Transmission**	**Delay Period**
Without Low Power Mode	1.0582 A	1.1814 A	1.0422 A
With Low Power Mode	0.85346 A	1.05636 A	0.69252 A

With this low power mode active, it is possible to observe a decrease in the node's current draw, especially when the node is in the delay period. Initially, in low power mode, the node consumes 0.85346 A, a lower value also due to the implementation of the low consumption mode for the SPS30 sensor, described in Section 3.2.8.3. After the measurements taken by the sensors, the sensor reaches a value of 1.05636 A, when information is sent via LoRa. When the node enters its delay period, sleep mode is activated, and its consumption remains at around 0.69252 A.

Table 4.2. Monitoring node power consumption in measurement, transmission and sleep modes

	Power Consumption ($P\ [Watts] = V\ [Volts] \times I\ [Amps]$)		
	Sensors Measuring	**LoRa Transmission**	**Delay Period**
Without Low Power Mode	5.291 W	5.907 W	5.211 W
With Low Power Mode	4.267 W	5.282 W	3.463 W

As the current decreases, so does the power consumption associated with the system. This is most evident when the monitoring node is in its delay period before sending another packet with sensor measurements, which is when the LoRa module and microcontroller are in sleep mode.

The value of the battery's autonomy can be obtained from the quotient between the battery's capacity and the current drawn by the monitoring node. Table 4.3 shows the autonomy calculated for each period of activity of the monitoring node:

Table 4.3. Battery autonomy during several periods of activity, without and with Low Power Mode

	Battery autonomy ($\text{Autonomy [hours]} = \frac{\text{Battery capacity (mAh)}}{Current\ Drawn\ (mA)}$)		
	Sensors Measuring	**LoRa Transmission**	**Delay Period**
Without Low Power Mode	9 hours and 27 minutes	8 hours and 27 minutes	9 hours and 35 minutes
With Low Power Mode	11 hours and 43 minutes	9 hours and 28 minutes	14 hours and 26 minutes

The introduction of sleep mode in the moment when the monitoring node is waiting to send another data packet results in the highest autonomy value, but the autonomy of the system, considering a normal operating cycle, with the all the activities describes in the Table 4.3, can be obtained with the following formula:

$$System\ autonomy\ [hours] = \frac{Battery\ capacity\ (mAh)}{\frac{I_M \times T_M}{T_M + T_T + T_S} + \frac{I_T \times T_T}{T_M + T_T + T_S} + \frac{I_S \times T_S}{T_M + T_T + T_S}} \qquad (10)$$

This formula considers the current values when measurements are being made by the sensors (I_M), transmission via LoRa (I_T) and when the node is in the delay period waiting to send another packet (I_S). Additionally, the time when the node is taking measurements (T_M), transmitting via LoRa (T_T) and in the delay period (T_S), is used. The values used for I_M, I_T and I_S are shown in the Table 4.1. It is assumed

3 seconds and 1 second for T_M and T_T, respectively. For T_S the values it is assumed the worst case, where the information is sent via LoRa every 7 minutes.

Table 4.4. System autonomy without and with Low Power mode

	System autonomy [hours]
Without Low Power Mode	9 hours and 35 minutes
With Low Power Mode	14 hours and 24 minutes

It can be observed that the introduction of methods to reduce the energy consumption of the monitoring node has a big influence on the autonomy of a battery, and in this type of systems it is important to conserve battery lifespan as much as possible to ensure a longer monitoring periods.

4.3.2. MQ Sensors Power Consumption

MQ sensors have a metallic filament in their composition, which is heated when the sensor is switched on, and is used to maintain a high operating temperature. The constant operation of the sensor can result in high energy consumption. Measurements were carried out to see the impact that the constant operation of the MQ sensors has on the consumption of the monitoring node. The measurements were made only with the MQ sensors connected to the microcontroller.

Table 4.5. Current consumption with 1, 2 and 3 MQ sensors connected to the microcontroller

	1 MQ sensor	**2 MQ sensors**	**3 MQ sensors**
Current Drawn [Amps]	0.20474 A	0.36455 A	0.71404 A

Table 4.6. Power consumption for consumption for each MQ sensor connected to the microcontroller

	1 MQ sensor	**2 MQ sensors**	**3 MQ sensors**
Power Consumption [Watts]	1.0237 W	1.82275 W	3.5702 W

The results obtained are associated with one, two and three MQ sensors connected. Each time a MQ sensor was connected to the microcontroller, its current, and consequently, the power consumption increased. It can be said that the operation of each sensor in this system consumes around 160-350 mA (milliamps). When three MQ sensors are connected, the total consumption of the microcontroller is more than half the consumption observed in the first scenario, where low power mode is not used, which means that most of the system consumption is generated by the constant operation of the MQ sensors.

CHAPTER 5

Conclusions and Future Work

5.1. Conclusions

A smart monitoring IoT system was proposed, consisting of several sensors to extract information about air quality condition in an outdoor context, across some places of Lisbon city. In this system, sensors were used to capture concentrations from some of pollutant gases like, carbon monoxide, liquefied petroleum gas, organic compounds like alcohol. Particulate matter, air quality and atmospheric factors such as temperature and humidity, were also measured to provide more data about the conditions at the monitoring site. The integration of a GPS module allowed the localization of the monitoring system, for later analysis of results.

Initially, a test was carried out to check the sensors' calibration. Since it was only possible to use ethanol, only the MQ-2 sensor could be used to test its calibration. In this type of gas sensor, manufacturers advise that the sensor should be exposed to a defined gas concentration, for a certain period, so that the readings obtained are more accurate. The measurements taken by the MQ-7 and MQ-9 sensors, to capture CO and LPG respectively, showed that the values obtained in the locations where the outdoor tests were conducted, varied somewhat depending on the surrounding environment. Despite their reactiveness to different conditions, it cannot be concluded that the concentrations obtained by the sensors are very precise. MOS sensors, like the MQ sensors, are sensitive to multiple gases, which means that the presence of other gases, around the monitoring environment, can have influence on the concentration of a gas that is being studied. Exposing the gas directly to the sensor is also an important step in obtaining accurate readings, but in the case of this system, this was not the possible with CO and LPG. Environmental conditions, like humidity and temperature, can also have an influence in the accuracy of readings from MQ and particulate matter sensors, like the SPS30 sensor.

This system is designed for an outdoor scenario, where monitoring is conducted in several dispersed locations, with varying distances from the monitoring node to the gateways, requiring the adoption of a technology that offered long range communication and wide coverage. For this reason, LoRa technology was used. By spreading the signal over a wide bandwidth, it is possible to achieve transmission with wide coverage and low energy consumption, which is required for systems that need to monitor large areas. A basic LoRa communication network only requires gateways and LoRa modules for transmitting and receiving data, which also makes deployment costs lower when compared to cellular networks. The possibility of adding more components, such as gateways and monitoring nodes, to the LoRa network, makes the technology highly scalable in the monitoring context, allowing the network to expand to more geographical locations as well as ensuring that the technology is viable for large-scale deployments. The usage of LoRa technology fulfilled all the communication requirements, needed for the system present in this dissertation.

To obtain a smart implementation, allowing the change of readings sending frequency, via LoRa, an urban traffic API was used. With this API it was possible to set levels of traffic, manually and define thresholds for each of these levels, and every time the level of traffic changes, the Node-Red application sends a downlink message to the monitoring node, updating the frequency of sending readings.

During transmission through LoRa, there were occasions in which parameters such data rate, the time taken for a message to reach the gateway and the receiver's sensitivity changed according to network conditions, due to the Adaptive Data Rate (ADR) mechanism that adapts these parameters automatically, and consequently, during the monitoring period, the limits imposed by the Fair Use Policy could not be complied with. ADR has its limitations, and the automatic adaptation of the communication parameters means that the Fair Use Policy limits are changed throughout the monitoring period, making it difficult to respect these limits when using ADR.

A web and Android application were created for data visualisation, allowing the user to view real time data about the air quality, but also obtain historical data from measurements taken by the monitoring node. The creation of applications that allow the visualisation of data in real time and the integration of warnings associated with changes in the level of air quality, such as the one implemented, makes it possible to evaluate air quality conditions and in this way give people information about places to avoid when these are highly polluted. Analysing the historical data also makes it possible to understand certain trends or variations associated with air quality and pollutant concentrations.

To conclude, by using a technology with a long range, low implementation cost and low energy consumption, like LoRa, the integration of low-cost sensors capable of monitoring pollutant gases, particulate matter, temperature and humidity and the use of LoRaWAN infrastructures and cloud services and platforms, it was possible to create an air quality monitoring system, fulfilling all the objectives established for this dissertation.

The development of this work resulted in a scientific paper "Smart City Air Quality Monitoring supported by IoT ecosystem" (Appendix F) for 16th International Conference on Sensing Technology.

5.2. Future Work

Looking ahead, we hope to expand the air quality monitoring network by integrating more nodes spread across different geographical regions and gateways to create our own LoRa monitoring network.

A system calibration using a variety of gases, exposed directly to the sensors to increase their accuracy, is required to make the system more reliable, and be less cross-sensitive and also the acquisition of sensors with lower energy consumption, to make the system more energy efficient. The integration of solar panels can also be considered, for better sustainability and lowering the costs.

Ultimately, the creation of predictive models is another task for the future, which will make it possible to forecast pollutant gas and particulate matter concentrations and extending presentation of data to represent air quality conditions in the monitored areas.

References

[1] ‘Air pollution’. Accessed: Jul. 31, 2023. [Online]. Available: https://www.who.int/health-topics/air-pollution

[2] H. Ritchie and M. Roser, ‘Outdoor Air Pollution’, *Our World in Data*, Nov. 2019, Accessed: Jan. 24, 2023. [Online]. Available: https://ourworldindata.org/outdoor-air-pollution

[3] ‘Ambient (outdoor) air pollution’. Accessed: Jan. 24, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health

[4] OECD, *The Economic Consequences of Outdoor Air Pollution*. OECD, 2016. doi: 10.1787/9789264257474-en.

[5] ‘The psychological, economic, and social costs of air pollution’, MIT Sloan. Accessed: Jan. 24, 2023. [Online]. Available: https://mitsloan.mit.edu/ideas-made-to-matter/psychological-economic-and-social-costs-air-pollution

[6] ‘Commission proposes rules for cleaner air and water’, European Commission - European Commission. Accessed: Jan. 24, 2023. [Online]. Available: https://ec.europa.eu/commission/presscorner/detail/en/ip_22_6278

[7] W. Y. Yi, K. M. Lo, T. Mak, K. S. Leung, Y. Leung, and M. L. Meng, ‘A Survey of Wireless Sensor Network Based Air Pollution Monitoring Systems’, *Sensors*, vol. 15, no. 12, Art. no. 12, Dec. 2015, doi: 10.3390/s151229859.

[8] J. Lozano *et al.*, ‘Personal electronic systems for citizen measurements of air quality’, in *2019 5th Experiment International Conference (exp.at’19)*, Jun. 2019, pp. 315–319. doi: 10.1109/EXPAT.2019.8876471.

[9] L. Chettri and R. Bera, ‘A Comprehensive Survey on Internet of Things (IoT) Toward 5G Wireless Systems’, *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 16–32, Jan. 2020, doi: 10.1109/JIOT.2019.2948888.

[10] M. L. Liya and M. Aswathy, ‘LoRa technology for Internet of Things(IoT):A brief Survey’, in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Oct. 2020, pp. 8–13. doi: 10.1109/I-SMAC49090.2020.9243449.

[11] M. Abbasi, S. Khorasanian, and M. H. Yaghmaee, ‘Low-Power Wide Area Network (LPWAN) for Smart grid: An in-depth study on LoRaWAN’, in *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)*, Feb. 2019, pp. 022–029. doi: 10.1109/KBEI.2019.8735089.

[12] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, ‘Overview of Cellular LPWAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB-IoT’, in *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, Mar. 2018, pp. 197–202. doi: 10.1109/PERCOMW.2018.8480255.

[13] F. Van den Abeele, J. Haxhibeqiri, I. Moerman, and J. Hoebeke, ‘Scalability Analysis of Large-Scale LoRaWAN Networks in ns-3’, *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2186–2198, Dec. 2017, doi: 10.1109/JIOT.2017.2768498.

[14] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, ‘A Design Science Research Methodology for Information Systems Research’, *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, Dec. 2007, doi: 10.2753/MIS0742-1222240302.

[15] M. C. Turner *et al.*, ‘Outdoor air pollution and cancer: An overview of the current evidence and public health recommendations’, *CA: A Cancer Journal for Clinicians*, vol. 70, no. 6, pp. 460–479, 2020, doi: 10.3322/caac.21632.

[16] ‘What are the WHO Air quality guidelines?’ Accessed: Oct. 15, 2023. [Online]. Available: https://www.who.int/news-room/feature-stories/detail/what-are-the-who-air-quality-guidelines

[17] ‘European Air Quality Index’. Accessed: Oct. 15, 2023. [Online]. Available: https://airindex.eea.europa.eu/Map/AQI/Viewer/#

[18] O. US EPA, ‘Air Data Basic Information’. Accessed: Oct. 09, 2023. [Online]. Available: https://www.epa.gov/outdoor-air-quality-data/air-data-basic-information

[19] "Technical Assistance Document for the Reporting of Daily Air Quality – the Air Quality Index (AQI)". Accessed: Oct. 15, 2023. [Online]. Available: https://www.airnow.gov/sites/default/files/2020-05/aqi-technical-assistance-document-sept2018.pdf

[20] D. Sehrawat and N. S. Gill, ‘Smart Sensors: Analysis of Different Types of IoT Sensors’, in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Apr. 2019, pp. 523–528. doi: 10.1109/ICOEI.2019.8862778.

[21] K. M. Simitha and M. S. Subodh Raj, ‘IoT and WSN Based Air Quality Monitoring and Energy Saving System in SmartCity Project’, in *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, Jul. 2019, pp. 1431–1437. doi: 10.1109/ICICICT46008.2019.8993151.

[22] A. A. Aziz *et al.*, ‘Portable Outdoor Air Quality Monitoring Using A Wireless Sensor Network (WSN)’, in *2021 4th International Conference on Computing & Information Sciences (ICCIS)*, Nov. 2021, pp. 1–5. doi: 10.1109/ICCIS54243.2021.9676381.

[23] N. A. A. Husein, A. Hadi, and D. Putri, ‘Evaluation of LoRa-based Air Pollution Monitoring System’, *IJACSA*, vol. 10, no. 7, 2019, doi: 10.14569/IJACSA.2019.0100753.

[24] S. Walling, J. Sengupta, and S. Das Bit, ‘A Low-cost Real-time IoT based Air Pollution Monitoring using LoRa’, in *2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Dec. 2019, pp. 1–6. doi: 10.1109/ANTS47819.2019.9117963.

[25] F. Concas *et al.*, ‘Low-Cost Outdoor Air Quality Monitoring and Sensor Calibration: A Survey and Critical Analysis’, *ACM Trans. Sen. Netw.*, vol. 17, no. 2, p. 20:1-20:44, May 2021, doi: 10.1145/3446005.

[26] M. Y. Thu, W. Htun, Y. L. Aung, P. E. E. Shwe, and N. M. Tun, ‘Smart Air Quality Monitoring System with LoRaWAN’, in *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*, Nov. 2018, pp. 10–15. doi: 10.1109/IOTAIS.2018.8600904.

[27] ‘How does an NDIR CO2 Sensor Work?’, CO2 Meter. Accessed: Feb. 03, 2023. [Online]. Available: https://www.co2meter.com/blogs/news/how-does-an-ndir-co2-sensor-work

[28] H. Patashnick and E. G. Rupprecht, ‘Continuous PM-10 Measurements Using the Tapered Element Oscillating Microbalance’, *Journal of the Air & Waste Management Association*, vol. 41, no. 8, pp. 1079–1083, Aug. 1991, doi: 10.1080/10473289.1991.10466903.

[29] 'Method IO-1.2 Determination of PM10 in Ambient Air Using the Thermo Environmental Instruments (formerly Wedding) Continuous Beta Attenuation Monitor'.

[30] U. Raza, P. Kulkarni, and M. Sooriyabandara, 'Low Power Wide Area Networks: An Overview', *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 855–873, 2017, doi: 10.1109/COMST.2017.2652320.

[31] Ahmed. S. Elselini, Hamed. R. Eleribi, M. Sanaani, and A. Alwerfalli, 'A Performance Study of an IoT System Using LoRa Access Network Technology', in *Proceedings of the 6th International Conference on Engineering & MIS 2020*, in ICEMIS'20. New York, NY, USA: Association for Computing Machinery, Sep. 2020, pp. 1–7. doi: 10.1145/3410352.3410807.

[32] P. Sethi and S. R. Sarangi, 'Internet of Things: Architectures, Protocols, and Applications', *Journal of Electrical and Computer Engineering*, vol. 2017, pp. 1–25, 2017, doi: 10.1155/2017/9324035.

[33] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, 'A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications', *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017, doi: 10.1109/JIOT.2017.2683200.

[34] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, 'Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications', *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015, doi: 10.1109/COMST.2015.2444095.

[35] 'Difference between Licensed Band and Unlicensed Band'. Accessed: Jan. 30, 2023. [Online]. Available: https://www.rfwireless-world.com/Terminology/Difference-between-Licensed-band-and-Unlicensed-band.html

[36] W. Ayoub, M. Mroue, F. Nouvel, A. E. Samhat, and J. Prévotet, 'Towards IP over LPWANs technologies: LoRaWAN, DASH7, NB-IoT', in *2018 Sixth International Conference on Digital Information, Networking, and Wireless Communications (DINWC)*, Apr. 2018, pp. 43–47. doi: 10.1109/DINWC.2018.8356993.

[37] 'Spreading Factors', The Things Network. Accessed: Jan. 31, 2023. [Online]. Available: https://www.thethingsnetwork.org/docs/lorawan/spreading-factors/

[38] 'Adaptive Data Rate', The Things Network. Accessed: Oct. 27, 2023. [Online]. Available: https://www.thethingsnetwork.org/docs/lorawan/adaptive-data-rate/

[39] 'What is cloud computing? | IBM'. Accessed: Feb. 03, 2023. [Online]. Available: https://www.ibm.com/topics/cloud-computing

[40] A. R. Al-Ali, I. Zualkernan, and F. Aloul, 'A Mobile GPRS-Sensors Array for Air Pollution Monitoring', *IEEE Sensors Journal*, vol. 10, no. 10, pp. 1666–1671, Oct. 2010, doi: 10.1109/JSEN.2010.2045890.

[41] 'NOVT_Whitepaper_Alternative_CO2_Laser_Wavelengths.pdf'. Accessed: Feb. 06, 2023. [Online]. Available: https://novantaphotonics.com/wp-content/uploads/2021/12/NOVT_Whitepaper_Alternative_CO2_Laser_Wavelengths.pdf

[42] B. Vejlgaard, M. Lauridsen, H. Nguyen, I. Z. Kovacs, P. Mogensen, and M. Sorensen, 'Coverage and Capacity Analysis of Sigfox, LoRa, GPRS, and NB-IoT', in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, Sydney, NSW: IEEE, Jun. 2017, pp. 1–5. doi: 10.1109/VTCSpring.2017.8108666.

[43] 'What is Grafana?' Accessed: Feb. 06, 2023. [Online]. Available: https://www.redhat.com/en/topics/data-services/what-is-grafana

[44] ‘Noções básicas | Documentação do Firebase’, Firebase. Accessed: Jul. 31, 2023. [Online]. Available: https://firebase.google.com/docs/guides?hl=pt-br

[45] "MQ-2 Datasheet", Hanwei Eletronics. Accessed: Jul. 29, 2023. [Online]. Available: https://www.mouser.com/datasheet/2/321/605-00008-MQ-2-Datasheet-370464.pdf

[46] "MQ-7 Datasheet", Hanwei Eletronics. Accessed: Jul. 29, 2023. [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf

[47] "MQ-9 Datasheet", Hanwei Eletronics. Accessed: Jul. 29, 2023. [Online]. Available: https://www.electronicoscaldas.com/datasheet/MQ-9_Hanwei.pdf

[48] Department of Materials and Mechanical Technology, Faculty of Technology, University of Sri Jayewardenepura, N. Kobbekaduwa, W. R. De Mel, Department of Materials and Mechanical Technology, Faculty of Technology, University of Sri Jayewardenepura, P. Oruthota, and Department of Materials and Mechanical Technology, Faculty of Technology, University of Sri Jayewardenepura, ‘Calibration and Implementation of Heat Cycle Requirement of MQ-7 Semiconductor Sensor for Detection of Carbon Monoxide Concentrations’, *ait*, vol. 1, no. 2, Aug. 2021, doi: 10.31357/ait.v1i2.5068.

[49] ‘R-Squared’, Corporate Finance Institute. Accessed: Oct. 09, 2023. [Online]. Available: https://corporatefinanceinstitute.com/resources/data-science/r-squared/

[50] ‘Grove - Air Quality Sensor v1.3 | Seeed Studio Wiki’. Accessed: Jul. 29, 2023. [Online]. Available: https://wiki.seeedstudio.com/Grove-Air_Quality_Sensor_v1.3/

[51] T. Liu, ‘Capacitive-type humidity and temperature module/sensor’.

[52] O. US EPA, ‘Particulate Matter (PM) Basics’. Accessed: Oct. 09, 2023. [Online]. Available: https://www.epa.gov/pm-pollution/particulate-matter-pm-basics

[53] "Datasheet SPS30 Particulate Matter Sensor for Air Quality Monitoring and Control", Sensirion, The Sensor Company. Accessed: Jul. 20, 2023. [Online]. Available: https://sensirion.com/media/documents/8600FF88/616542B5/Sensirion_PM_Sensors_Datasheet_SPS30.pdf

[54] "Low-Power Operation Of the SPS30 Particulate Matter Sensor", Sensirion, The Sensor Company. Accessed: Oct. 09, 2023. [Online]. Available: https://sensirion.com/media/documents/188A2C3C/6166F165/Sensirion_Particulate_Matter_AppNotes_SPS30_Low_Power_Operation_D1.pdf

[55] ‘Teardown: Sensirion SPS30 Particle Matter Sensor | MistyWest’. Accessed: Oct. 09, 2023. [Online]. Available: https://www.mistywest.com/posts/teardown-sensirion-particle-matter-sensor/

[56] "Mini GPS/BDS Unit (AT6558)", M5Stack. Accessed: Oct. 09, 2023. [Online]. Available: https://media.digikey.com/pdf/Data%20Sheets/M5Stack%20PDFs/U032_Web.pdf

[57] ‘UNO R3 | Arduino Documentation’. Accessed: Oct. 09, 2023. [Online]. Available: https://docs.arduino.cc/hardware/uno-rev3

[58] "Long Range Wireless Transceiver for Arduino - Lora Shield", Dragino. Accessed: Oct. 09, 2023. [Online]. Available: https://www.dragino.com/downloads/downloads/LoraShield/Datasheet_LoraShield.pdf

[59] ‘MKR WAN 1300 | Arduino Documentation’. Accessed: Jul. 29, 2023. [Online]. Available: https://docs.arduino.cc/hardware/mkr-wan-1300

[60] "Raspberry Pi 4 Model B - Datasheet", Raspberry Pi. Accessed: Oct. 09, 2023. [Online]. Available: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf

[61] ‘Overview of the Arduino IDE 1 | Arduino Documentation’. Accessed: Oct. 09, 2023. [Online]. Available: https://docs.arduino.cc/software/ide-v1/tutorials/Environment

[62] ‘Arduino library for the SPS30 particulate matter sensor’. Sensirion AG, Jul. 04, 2023. Accessed: Oct. 09, 2023. [Online]. Available: https://github.com/Sensirion/arduino-sps

[63] ‘DHT sensor library’. Adafruit Industries, Oct. 08, 2023. Accessed: Oct. 09, 2023. [Online]. Available: https://github.com/adafruit/DHT-sensor-library

[64] ‘Seeed-Studio/Grove_Air_quality_Sensor’. Seeed Studio, May 09, 2023. Accessed: Oct. 09, 2023. [Online]. Available: https://github.com/Seeed-Studio/Grove_Air_quality_Sensor

[65] M. Hart, ‘TinyGPSPlus’. Oct. 05, 2023. Accessed: Oct. 09, 2023. [Online]. Available: https://github.com/mikalhart/TinyGPSPlus

[66] ‘MKRWAN’. Arduino Libraries, Aug. 23, 2023. Accessed: Oct. 09, 2023. [Online]. Available: https://github.com/arduino-libraries/MKRWAN

[67] ‘LoRaWAN® Regional Parameters’.

[68] ‘Arduino Low Power library’. Arduino Libraries, Oct. 08, 2023. Accessed: Oct. 27, 2023. [Online]. Available: https://github.com/arduino-libraries/ArduinoLowPower

[69] ‘The Things Network’. Accessed: Oct. 09, 2023. [Online]. Available: https://www.thethingsindustries.com/docs/reference/ttn/

[70] ‘End Device Activation’, The Things Network. Accessed: Oct. 09, 2023. [Online]. Available: https://www.thethingsnetwork.org/docs/lorawan/end-device-activation/

[71] ‘Duty Cycle’, The Things Network. Accessed: Oct. 27, 2023. [Online]. Available: https://www.thethingsnetwork.org/docs/lorawan/duty-cycle/

[72] T. T. Network, ‘Lisbon’, The Things Network. Accessed: Jul. 31, 2023. [Online]. Available: https://www.thethingsnetwork.org/community/lisbon/

[73] ‘Node-RED’. Accessed: Oct. 09, 2023. [Online]. Available: https://nodered.org/#get-started

[74] ‘MQTT - The Standard for IoT Messaging’. Accessed: Oct. 09, 2023. [Online]. Available: https://mqtt.org/

[75] ‘Airtime calculator for LoRaWAN’. Accessed: Oct. 27, 2023. [Online]. Available: https://avbentem.github.io/airtime-calculator/ttn/eu868

[76] ‘Flutter - Build apps for any screen’. Accessed: Oct. 09, 2023. [Online]. Available: //flutter.dev/

[77] "SR-3 - Bench Top Test Chamber", Figaro Accessed: Oct. 19, 2023. [Online]. Available: https://www.figaro.co.jp/en/product/docs/sr-3_product%20information%28en%29_rev02.pdf

APPENDIX A

Arduino Microcontroller Code

```
//Imported libraries
#include <MKRWAN.h>
#include <Wire.h>
#include <TinyGPSPlus.h>
#include <avr/dtostrf.h>
#include <sps30.h>
#include "DHT.h"
#include "Air_Quality_Sensor.h"
#include "ArduinoLowPower.h"
#include "arduino_secrets.h"  // File with the TTN appEui and appKey

#define gpsSerial Serial1 //Instance for UART communication between gps module
and arduino board
#define DHTPIN 2 //Declaring the pin which dht22 sensor is connected
#define DHTTYPE DHT22 //Declaring the type of dht sensor used

//Intances for gps module, dht22 and grove air quality sensors
TinyGPSPlus gps;
DHT dht(DHTPIN, DHTTYPE);
AirQualitySensor sensor(A3);

static const uint32_t GPSBaud = 9600; //GPS BaudRate 9600 bits/s

//Declaring the pins for MQ sensor connection
int gas_sensor_mq2 = A4;
int gas_sensor_mq7 = A5;
int gas_sensor_mq9 = A2;

//R0 values used for MQ-2, MQ-7, and MQ-9, respectively
float R0_mq2 = 1.30;
float R0_mq7 = 4.78;
float R0_mq9 = 8.21;

//Gathering appEui and appKey from TTN application, stored in a private file
String appEui = SECRET_APP_EUI;
String appKey = SECRET_APP_KEY;

//Instance for LoRa communication
LoRaModem modem;

//Default frequency of sending readings via LoRa (5 minutes)
```

```
int interval = 300;

void setup() {
  int16_t ret;
  uint8_t auto_clean_days = 4;
  uint32_t auto_clean;

  Serial.begin(115200);
  analogReadResolution(12); //Changing MKRWAN board bit resolution to 12 bits
  delay(3000);

  gpsSerial.begin(GPSBaud); //GPS module initialization with baud rate defined
in "GPSBaud" variable

  dht.begin(); //DHT22 sensor initialization

  if (sensor.init()) { //Grove Air Quality v1.3 sensor initialization check
        Serial.println("Sensor ready.");
    } else {
        Serial.println("Sensor ERROR!");
  }

  //Declaring pins associated with MQ sensors, as INPUT (gathering values)
  pinMode(gas_sensor_mq2, INPUT);
  pinMode(gas_sensor_mq7, INPUT);
  pinMode(gas_sensor_mq9, INPUT);

  sensirion_i2c_init(); //SPS30 sensor initialization
  while (sps30_probe() != 0) { //SPS30 sensor initialization check
    Serial.print("SPS sensor probing failed\n");
    delay(500);
  }

  //Declaring a value for SPS30 sensor automatically cleaning interval (every
4 days)
  ret = sps30_set_fan_auto_cleaning_interval_days(auto_clean_days);
  if (ret) {
    Serial.print("error setting the auto-clean: ");
    Serial.println(ret);
  }

  //Putting the SP30 sensor into sleep state
  sps30_sleep();

  while (!Serial);
  if (!modem.begin(EU868)) { //Checking LoRa module initialization (with
regional parameter for the reagion where the communication is being made, in
this case EU868)
    Serial.println("Failed to start module");
    while (1) {}
  };

  Serial.println(modem.deviceEUI()); //Obtaining MKRWAN 1300 board DevEUI

  int connected = modem.joinOTAA(appEui, appKey); //Connection to TTN
platform, with user TTN application appEui and appKey
  if (!connected) { //Checking connection to TTN platform
    Serial.println("Something went wrong with connection");
    while (1) {}
  }
  delay(10000);
}

//Function to check if there is any available data on the GPS module
static void gpsDelay(unsigned long ms)
{
  unsigned long start = millis();
  do
  {
    while (gpsSerial.available())
      gps.encode(gpsSerial.read());
  } while (millis() - start < ms);
}

void loop() {
  float sensor_volt_mq2, sensor_volt_mq7, sensor_volt_mq9;
  float RS_gas_mq2, RS_gas_mq7, RS_gas_mq9;
  float ratio_mq2, ratio_mq7, ratio_mq9;
  struct sps30_measurement m;
  uint16_t data_ready;
  int16_t ret;
  float pm2_5, pm_10;

  //Checking for gps data
  gpsDelay(2000);

  // //Reading the analog MQ sensor values for later conversion
  float sensorValue_mq2 = analogRead(gas_sensor_mq2);
  float sensorValue_mq7 = analogRead(gas_sensor_mq7);
  float sensorValue_mq9 = analogRead(gas_sensor_mq9);

  //Analog sensor readings to voltage conversion (Vrl=(ADC value x Vcc)/2^12
bits)
  sensor_volt_mq2 = (sensorValue_mq2*3.3)/4095.0;
  sensor_volt_mq7 = (sensorValue_mq7*3.3)/4095.0;
  sensor_volt_mq9 = (sensorValue_mq9*3.3)/4095.0;

  //Getting RS value for each MQ sensor (RS=[(Vcc/Vrl)-1]xRL)
  RS_gas_mq2 = ((3.3/sensor_volt_mq2)-1)*5.0;
  RS_gas_mq7 = ((3.3/sensor_volt_mq7)-1)*10.0;
  RS_gas_mq9 = ((3.3/sensor_volt_mq9)-1)*10.0;

  //Rs/R0 ratio calculation for each MQ sensor
  ratio_mq2 = RS_gas_mq2/R0_mq2;
  ratio_mq7 = RS_gas_mq7/R0_mq7;
  ratio_mq9 = RS_gas_mq9/R0_mq9;

  //Gas concentration calculation
  double ppm_mq2 = 3789.8 * pow(ratio_mq2, -2.72); //Alcohol
  double ppm_mq7 = 103.16 * pow(ratio_mq7, -1.498); //CO
  double ppm_mq9 = 1013.7 * pow(ratio_mq9, -2.088); //LPG

  Serial.println(ppm_mq2);
  Serial.println(ppm_mq7);
  Serial.println(ppm_mq9);

  //SPS30 sensor wake up from sleep mode to idle mode
  ret = sps30_wake_up();
  if (ret < 0) {
    Serial.print("Error waking UP");
  } else {
    Serial.println("SUCESS WAKING UP");
  }

  //SPS30 sensor change from idle mode to measurement mode
  ret = sps30_start_measurement();
  if (ret < 0) {
    Serial.print("error starting measurement\n");
  }

  //Delay to obtain more accuracy readings before the taking of PM2.5 and PM10
readings in measurement mode
  delay(30000);

  //Loop tp check available data on the SPS30 sensor
  do {
    ret = sps30_read_data_ready(&data_ready);
    if (ret < 0) {
      Serial.print("error reading data-ready flag: ");
    } else if (!data_ready)
      Serial.print("data not ready, no new measurement available\n");
    else
      break;
    delay(100);
  } while (1);

  //Gathering measurements and store them in a structure "m"
  ret = sps30_read_measurement(&m);
  if (ret < 0) {
    Serial.print("error reading measurement\n");
  }

  //Store PM2.5 and PM10 values in variables
  pm2_5 = m.mc_2p5;
  pm_10 = m.mc_10p0;

  //SPS30 change from measurement mode to idle mode
  ret = sps30_stop_measurement();
  if (ret < 0) {
    Serial.print("Error STOPPING");
  } else {
    Serial.println("SUCESS STOPPING");
  }

  //SPS30 change from idle mode to sleep mode
  ret = sps30_sleep();
  if (ret < 0) {
    Serial.print("Error Sleeping");
  } else {
    Serial.println("SUCESS Sleeping");
  }

  Serial.println(pm2_5);
  Serial.println(pm_10);

  //Gathering temperature and relative humidity values and store them in
variables
  float temp = dht.readTemperature();
  float hum = dht.readHumidity();

  Serial.println(temp);
  Serial.println(hum);

  //Gathering latitude and longitude values and store them in variables
  float latitudeValue = gps.location.lat();
  float longitudeValue = gps.location.lng();

  Serial.println(latitudeValue);
  Serial.println(longitudeValue);

  ////Gathering air condition value store it in variable
  int aqi = sensor.getValue();
  Serial.println(aqi);

  //Storing all the measures in unsigned 16-bit and 32-bit integer varibles
for packet size otimization
```

```
uint16_t pm2_5_x = pm2_5 * 100;
uint16_t pm_10_x = pm_10 * 100;
uint16_t temperature = temp * 100;
uint16_t humidity = hum * 100;
uint32_t latitudeBinary = ((latitudeValue + 90) / 180) * 16777215;
uint32_t longitudeBinary = ((longitudeValue + 180) / 360) * 16777215;
uint16_t ppm_mq2_x = ppm_mq2 * 100;
uint16_t ppm_mq7_x = ppm_mq7 * 100;
uint16_t ppm_mq9_x = ppm_mq9 * 100;

//Unsigned 8-bit array to store all the measurements and its size
uint8_t sensorData[28];
size_t dataSize = sizeof(sensorData);

//Storing all the measurements in the "sensorData" array
sensorData[0] = temperature >> 8;
sensorData[1] = temperature;
sensorData[2] = humidity >> 8;
sensorData[3] = humidity;
sensorData[4] = pm2_5_x >> 8;
sensorData[5] = pm2_5_x;
sensorData[6] = pm_10_x >> 8;
sensorData[7] = pm_10_x;
sensorData[8] = ppm_mq2_x >> 8;
sensorData[9] = ppm_mq2_x;
sensorData[10] = ppm_mq7_x >> 8;
sensorData[11] = ppm_mq7_x;
sensorData[12] = ppm_mq9_x >> 8;
sensorData[13] = ppm_mq9_x;
sensorData[14] = aqi >> 8;
sensorData[15] = aqi;
sensorData[16] = interval >> 8;
sensorData[17] = interval;
sensorData[18] = latitudeBinary >> 24;
sensorData[19] = latitudeBinary >> 16;
sensorData[20] = latitudeBinary >> 8;
sensorData[21] = latitudeBinary;
sensorData[22] = longitudeBinary >> 24;
sensorData[23] = longitudeBinary >> 16;
sensorData[24] = longitudeBinary >> 8;
sensorData[25] = longitudeBinary;

int err;
//Initializing packet sending via LoRa
modem.beginPacket();
modem.write(sensorData, dataSize);
err = modem.endPacket(true);

//Checking if the packet was correctly sent
```

```
  if (err > 0) {
    Serial.println("Binary message sent correctly!");
  } else {
    Serial.println("Error sending binary message");
    Serial.println(err);
  }

  //Putting LoRa module into sleep mode to conserve energy
  modem.sleep();
  //Putting microcontroller and some internal peripherals into sleep mode to
conserve energy, during frequency of sending readings
  LowPower.sleep(interval*1000);
  //Delay introduced for wake up time of the microcontroller
  delay(15000);

  Serial.println("Avaliação de DW");
  //Checking if there is any downlink messages available
  if (!modem.available()) {
    Serial.println("No downlink message received at this time.");
    return;
  }
  //If there is a downlink message available, decode that message
  char rcv[64];
  int i = 0;
  while (modem.available()) {
    rcv[i++] = (char)modem.read();
  }
  rcv[i]=0;
  Serial.print("Received: ");
  for (unsigned int j = 0; j < i; j++) {
    Serial.println(rcv[j], DEC);
  }
  int number = (int)rcv[0];
  Serial.println(number);
  int new_delay = number*60;
  Serial.print("New delay: ");
  Serial.println(new_delay);

}
```

APPENDIX B

Arduino Code for R_0 calculation of each MQ sensor

```
//Imported libraries
#include <SPI.h>
#include <Wire.h>

void setup() {
  Serial.begin(9600);
  analogReadResolution(12); //Changing MKRWAN board bit resolution to 12 bits
  delay(10000);
}

void loop() {
  float sensor_volt2, sensor_volt7, sensor_volt9;
  float RS_air2, RS_air7, RS_air9;
  float R0_2, R0_7, R0_9;
  float sensorValue2, sensorValue7, sensorValue9;

  for(int x = 0 ; x < 500 ; x++) //Reading the analog MQ sensor values for
later conversion
  {
    sensorValue2 = sensorValue2 + analogRead(A4);
    sensorValue7 = sensorValue7 + analogRead(A5);
    sensorValue9 = sensorValue9 + analogRead(A2);
  }

  //Average of readings for each MQ sensor
  sensorValue2 = sensorValue2/500.0;
  sensorValue7 = sensorValue7/500.0;
  sensorValue9 = sensorValue9/500.0;

  //Average of readings to voltage conversion (Vrl=(ADC value x Vcc)/2^12
bits)
  sensor_volt2 = (sensorValue2*3.3)/4095.0;
  sensor_volt7 = (sensorValue7*3.3)/4095.0;
  sensor_volt9 = (sensorValue9*3.3)/4095.0;

  //Getting RS value for each MQ sensor (RS=[(Vcc/Vrl)-1]xRL)
  RS_air2 = ((3.3/sensor_volt2)-1)*5.0;
  RS_air7 = ((3.3/sensor_volt7)-1)*10.0;
  RS_air9 = ((3.3/sensor_volt9)-1)*10.0;

  //Getting RS value for each MQ sensor (R0=RS/(RS/R0 in fresh air))
  R0_2 = RS_air2/9.7;
  R0_7 = RS_air7/27.0;
  R0_9 = RS_air9/9.8;

  //Displaying R0 values
  Serial.print("R0_2 = ");
  Serial.println(R0_2);
  Serial.print("R0_7 = ");
  Serial.println(R0_7);
  Serial.print("R0_9 = ");
  Serial.println(R0_9);
  Serial.println("-------------------------------");
  delay(60000);
}
```

APPENDIX C

Arduino script to calculate gas concentration

```
 // //Reading the analog MQ sensor values for later conversion
  float sensorValue_mq2 = analogRead(gas_sensor_mq2);
  float sensorValue_mq7 = analogRead(gas_sensor_mq7);
  float sensorValue_mq9 = analogRead(gas_sensor_mq9);

  //Analog sensor readings to voltage conversion (Vrl=(ADC value x Vcc)/2^12
bits)
  sensor_volt_mq2 = (sensorValue_mq2*3.3)/4095.0;
  sensor_volt_mq7 = (sensorValue_mq7*3.3)/4095.0;
  sensor_volt_mq9 = (sensorValue_mq9*3.3)/4095.0;

  //Getting RS value for each MQ sensor (RS=[(Vcc/Vrl)-1]xRL)
  RS_gas_mq2 = ((3.3/sensor_volt_mq2)-1)*5.0;
  RS_gas_mq7 = ((3.3/sensor_volt_mq7)-1)*10.0;
  RS_gas_mq9 = ((3.3/sensor_volt_mq9)-1)*10.0;

  //Rs/R0 ratio calculation for each MQ sensor
  ratio_mq2 = RS_gas_mq2/R0_mq2;
  ratio_mq7 = RS_gas_mq7/R0_mq7;
  ratio_mq9 = RS_gas_mq9/R0_mq9;

  //Gas concentration calculation
  double ppm_mq2 = 3789.8 * pow(ratio_mq2, -2.72); //Alcohol
  double ppm_mq7 = 103.16 * pow(ratio_mq7, -1.498); //CO
  double ppm_mq9 = 1013.7 * pow(ratio_mq9, -2.088); //LPG
```

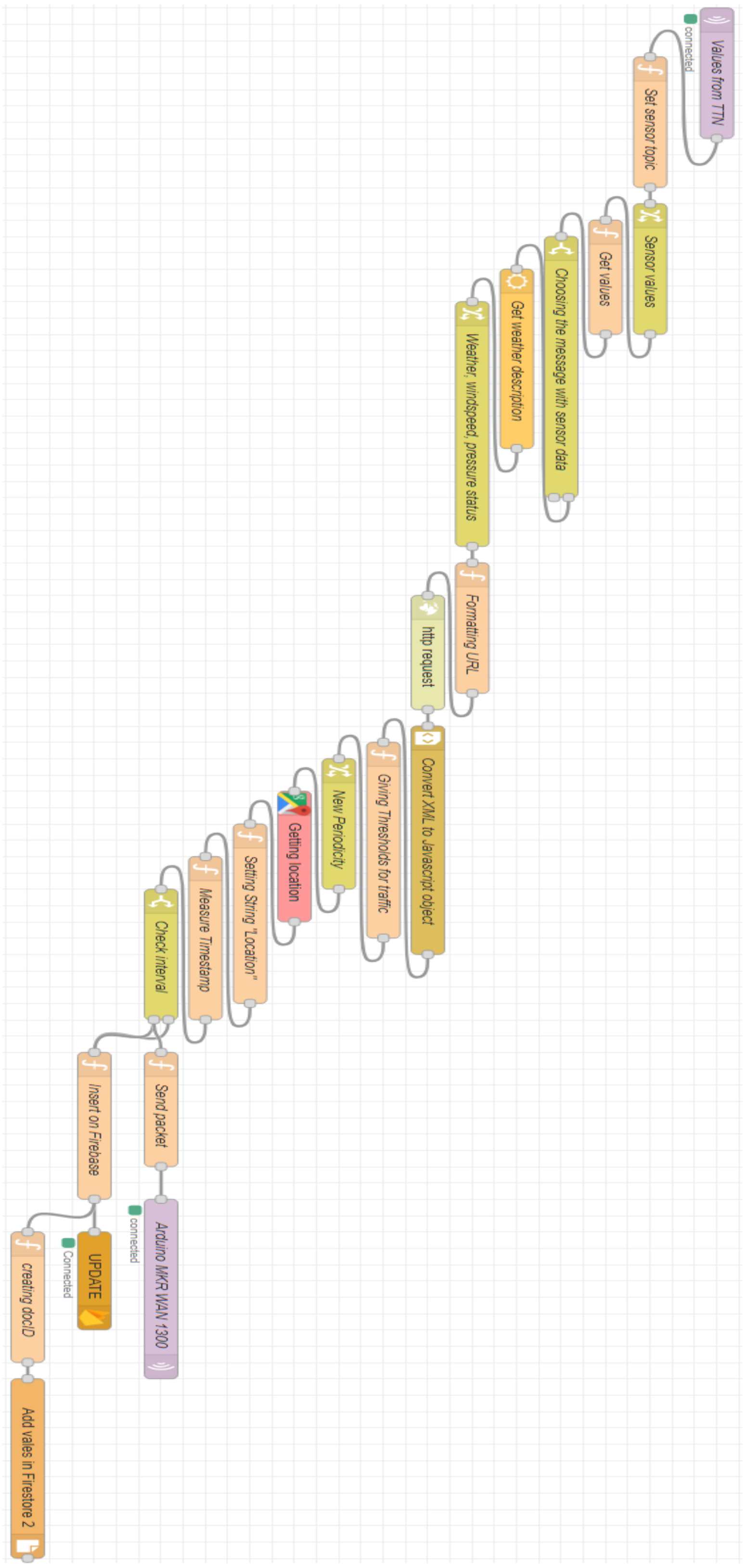
APPENDIX D

Javascript decoder formatter for TTN uplink messages

```
function Decoder(bytes, port) {
  var decoded = {};

  //Decoding temperature value
  var  celciusInt  =  (bytes[0]  &  0x80  ?  0xFFFF<<16  :  0)  |  bytes[0]<<8  |
   bytes[1];
  decoded.temperature = celciusInt / 100;

  //Decoding relative humidity value
  var humInt = (bytes[2] & 0x80 ? 0xFFFF<<16 : 0) | bytes[2]<<8 | bytes[3];
  decoded.humidity = humInt / 100;

  //Decoding PM2.5 µm value
  var pm2_5_x = (bytes[4] << 8) + bytes[5];
  decoded.pm2_5 = pm2_5_x / 100;

  //Decoding PM10 µm value
  var pm_10_x = (bytes[6] << 8) + bytes[7];
  decoded.pm_10 = pm_10_x / 100;

  //Decoding Alcohol value
  var ppm_mq2_x = (bytes[8] << 8) + bytes[9];
  decoded.ppm_mq2 = ppm_mq2_x / 100;

  //Decoding CO value
  var ppm_mq7_x = (bytes[10] << 8) + bytes[11];
  decoded.ppm_mq7 = ppm_mq7_x / 100;

  //Decoding LPG value
  var ppm_mq9_x = (bytes[12] << 8) + bytes[13];
  decoded.ppm_mq9 = ppm_mq9_x / 100;
  //Decoding AQI value
  decoded.aqi = (bytes[14] << 8) + bytes[15];

  //Decoding frequency of sending readings value
  decoded.interval = (bytes[16] << 8) + bytes[17];

  //Decoding Latitude value
  var lat = (bytes[18] << 24) + (bytes[19] << 16) + (bytes[20] << 8) +
   bytes[21];
  decoded.latitude = (((lat / 16777215.0) * 180.0) - 90).toFixed(6);

  //Decoding Longitude value
  var lon = (bytes[22] << 24) + (bytes[23] << 16) + (bytes[24] << 8) +
   bytes[25];
  decoded.longitude = (((lon / 16777215.0) * 360.0) - 180).toFixed(6);

  return decoded;
}
```

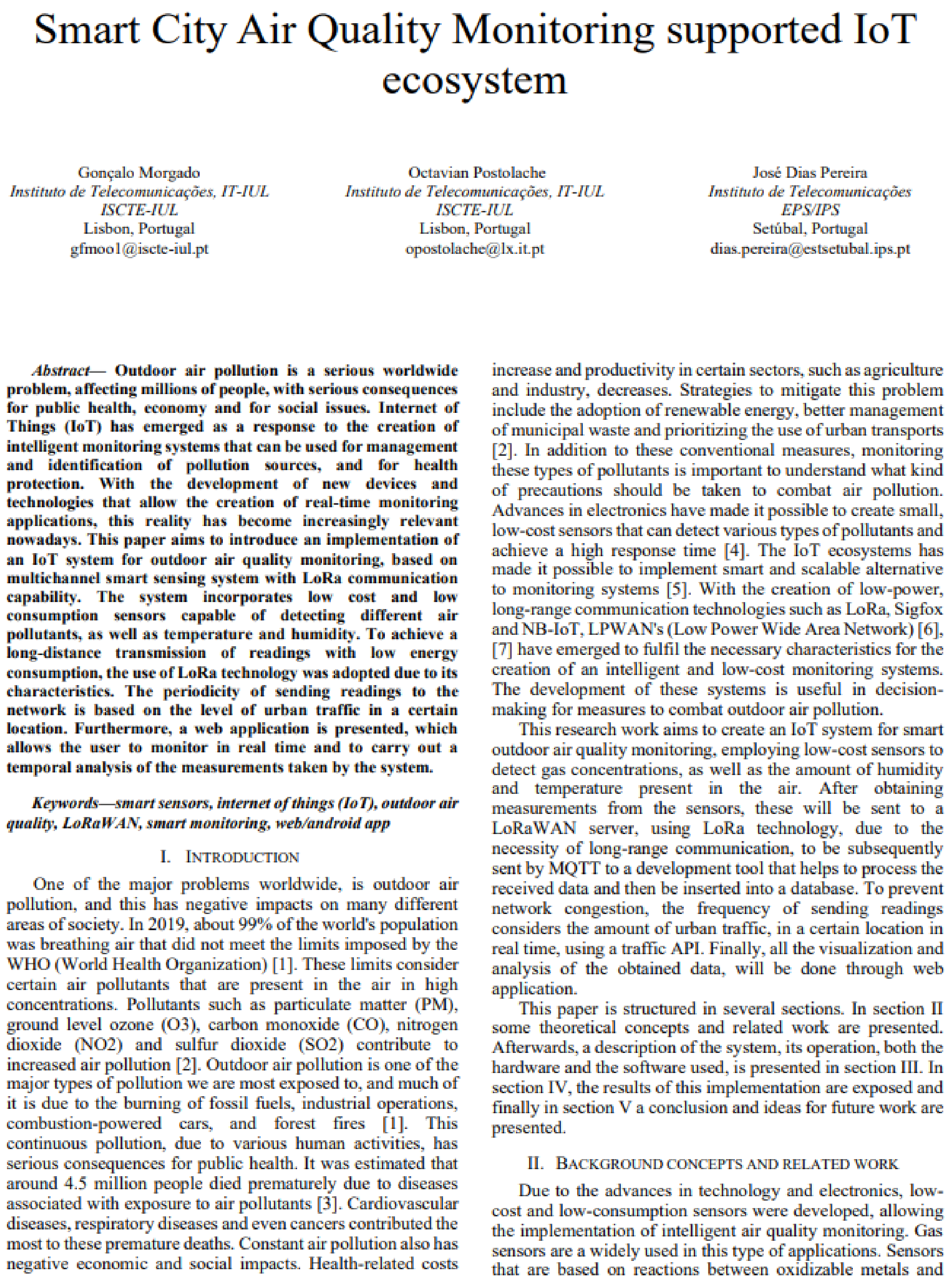
APPENDIX E

Node-Red Flow



APPENDIX F

Scientific Paper



Smart City Air Quality Monitoring supported IoT ecosystem

Gonçalo Morgado
Instituto de Telecomunicações, IT-IUL ISCTE-IUL
Lisbon, Portugal
gfmoo1@iscte-iul.pt

Octavian Postolache
Instituto de Telecomunicações, IT-IUL ISCTE-IUL
Lisbon, Portugal
opostolache@lx.it.pt

José Dias Pereira
Instituto de Telecomunicações EPS/IPS
Setúbal, Portugal
dias.pereira@estsetubal.ips.pt

***Abstract*— Outdoor air pollution is a serious worldwide problem, affecting millions of people, with serious consequences for public health, economy and for social issues. Internet of Things (IoT) has emerged as a response to the creation of intelligent monitoring systems that can be used for management and identification of pollution sources, and for health protection. With the development of new devices and technologies that allow the creation of real-time monitoring applications, this reality has become increasingly relevant nowadays. This paper aims to introduce an implementation of an IoT system for outdoor air quality monitoring, based on multichannel smart sensing system with LoRa communication capability. The system incorporates low cost and low consumption sensors capable of detecting different air pollutants, as well as temperature and humidity. To achieve a long-distance transmission of readings with low energy consumption, the use of LoRa technology was adopted due to its characteristics. The periodicity of sending readings to the network is based on the level of urban traffic in a certain location. Furthermore, a web application is presented, which allows the user to monitor in real time and to carry out a temporal analysis of the measurements taken by the system.**

Keywords—smart sensors, internet of things (IoT), outdoor air quality, LoRaWAN, smart monitoring, web/android app

I. Introduction

One of the major problems worldwide, is outdoor air pollution, and this has negative impacts on many different areas of society. In 2019, about 99% of the world's population was breathing air that did not meet the limits imposed by the WHO (World Health Organization) [1]. These limits consider certain air pollutants that are present in the air in high concentrations. Pollutants such as particulate matter (PM), ground level ozone (O3), carbon monoxide (CO), nitrogen dioxide (NO2) and sulfur dioxide (SO2) contribute to increased air pollution [2]. Outdoor air pollution is one of the major types of pollution we are most exposed to, and much of it is due to the burning of fossil fuels, industrial operations, combustion-powered cars, and forest fires [1]. This continuous pollution, due to various human activities, has serious consequences for public health. It was estimated that around 4.5 million people died prematurely due to diseases associated with exposure to air pollutants [3]. Cardiovascular diseases, respiratory diseases and even cancers contributed the most to these premature deaths. Constant air pollution also has negative economic and social impacts. Health-related costs increase and productivity in certain sectors, such as agriculture and industry, decreases. Strategies to mitigate this problem include the adoption of renewable energy, better management of municipal waste and prioritizing the use of urban transports [2]. In addition to these conventional measures, monitoring these types of pollutants is important to understand what kind of precautions should be taken to combat air pollution. Advances in electronics have made it possible to create small, low-cost sensors that can detect various types of pollutants and achieve a high response time [4]. The IoT ecosystems has made it possible to implement smart and scalable alternative to monitoring systems [5]. With the creation of low-power, long-range communication technologies such as LoRa, Sigfox and NB-IoT, LPWAN's (Low Power Wide Area Network) [6], [7] have emerged to fulfil the necessary characteristics for the creation of an intelligent and low-cost monitoring systems. The development of these systems is useful in decision-making for measures to combat outdoor air pollution.

This research work aims to create an IoT system for smart outdoor air quality monitoring, employing low-cost sensors to detect gas concentrations, as well as the amount of humidity and temperature present in the air. After obtaining measurements from the sensors, these will be sent to a LoRaWAN server, using LoRa technology, due to the necessity of long-range communication, to be subsequently sent by MQTT to a development tool that helps to process the received data and then be inserted into a database. To prevent network congestion, the frequency of sending readings considers the amount of urban traffic, in a certain location in real time, using a traffic API. Finally, all the visualization and analysis of the obtained data, will be done through web application.

This paper is structured in several sections. In section II some theoretical concepts and related work are presented. Afterwards, a description of the system, its operation, both the hardware and the software used, is presented in section III. In section IV, the results of this implementation are exposed and finally in section V a conclusion and ideas for future work are presented.

II. Background Concepts and Related Work

Due to the advances in technology and electronics, low-cost and low-consumption sensors were developed, allowing the implementation of intelligent air quality monitoring. Gas sensors are a widely used in this type of applications. Sensors that are based on reactions between oxidizable metals and polluting gases (Metal Oxide Semi-Conductors (MOS)), electrochemical reactions through the contact between gases and electrodes (Electrochemical Sensors) or sensors that use infrared light sources, that are absorbed by gas molecules and the detection is made based on the intensity of that same source (Non-dispersive Infrared Sensors (NDIR)), are some examples used in these implementations [8], [9]. To detect particulate matter, there are several techniques. The most common ones are gravimetry, the use of tapered element oscillating micro-balance analyzers and modules that are based on the light scattering principle, while this last one is most commonly used [8], [9].

The Internet of Things concept is based around the idea of a large network of hardware devices with integrated software, electronics, actuators, and sensors, enabling communication among devices and the gathering and exchange of data [5].

To minimize hardware and deployment costs, achieve greater coverage range and reduce energy consumption, LPWANs are commonly used. Short range technologies such as Bluetooth and Wi-Fi work with very high frequency bands (2.4GHz and 5GHz, with the 5 GHz band only for Wi-Fi), which makes the transmission range quite short and becomes more susceptible to interference. Since LPWAN technologies don't need to work with a large bandwidth value, their energy consumption is greatly reduced [5], as they use a frequency band between 863-870 MHz, in Europe, allowing for less attenuation and greater signal range.

New technologies have begun to be implemented to address the objectives of LPWANs to be used in IoT systems. Technologies such as NB-IoT, which operates in licensed bands, i.e., you need to pay or have permission to use the frequency band to make transmissions. Other LPWAN technologies used in this type of IoT systems are Sigfox and LoRa/LoRaWAN. Both operate in unlicensed bands, i.e., the use of the band for information transmission is free. LoRa technology acts on the physical layer, meaning that it does not transmit the signal, but only modulates it. The technology responsible for communication is LoRaWAN. It deals with the Medium Access Control (MAC) layer, establishing a bidirectional communication. To increase the capacity of the network and the battery life of end devices, a mechanism is used, Adaptive Data Rate (ADR), that adapts the parameters of the transmission according to the environment conditions, like data rate, bandwidth, symbol duration and rate [7], [10].

With the emergence of these technologies, numerous implementations of IoT monitoring systems have started to appear. In a project carried out by Simitha K. M. and Subodh Raj M. S. [11], is proposed a real-time water quality and air quality monitoring, that also incorporates a street light energy saving system, that takes readings from sensors and then sends them via LoRa to a cloud service. The pollutants studied are CO, NO2, SO2 and particulate matter. The monitoring node is composed by an Arduino Mega board, using the MQ7, MQ135, MQ136 sensors, a GP2Y1010AUF optical sensor, and a LoRa transmitter module. An ESP32 Wi-Fi module that is connected to a LoRa receiver module, acts as a gateway, to receive the readings. These values are subsequently sent over Wi-Fi via the ESP32 module to the ThingSpeak cloud platform, to later visualization. Also, important to note that a Raspberry Pi 3 Model B+ is used to read the readings.

In an implementation by A. James et al. [12], a system for monitoring temperature, relative humidity and the number of pedestrians is presented. This data is sent via LoRa technology to The Things Network (TTN) platform. The temperature and relative humidity are obtained using an AM2302 sensor, the number of pedestrians using the Adafruit PIR sensor and an Arduino MKRWAN 1300 board for LoRa communication. The Adafruit IO platform was used for data storage and analysis. The If This Then That (IFTTT) API was used to send the data from the TTN to the Adafruit platform.

In the implementation by S. Walling et al. [13], an air quality monitoring IoT system was built for only measuring two parameters, carbon monoxide (CO) from an MQ 9 sensor and air quality using an MQ 135 sensor. Sensor readings are sent to a gateway via LoRa and then sent to a cloud platform (ThingSpeak). An SMS notification system was also developed to alert mobile devices if the limits imposed for the monitoring parameters are exceeded. Sensor readings can be sent to a mobile application via a Bluetooth module. Each monitoring node was placed in 5 different locations throughout a university campus and an Indian city. In a latter study, the use of LoRa technology with Wi-Fi was compared in an outdoor and indoor scenario for sending readings, with the monitoring nodes at different distances from the gateway.

The authors [14] proposed an outdoor air quality monitoring system for areas with high population density and vehicle traffic in the Taipei city of Taiwan, based on a WSN. The monitoring nodes communicate with each other using the ZigBee protocol. The main pollutant measured in this implementation was carbon monoxide (CO). Other atmospheric parameters, such as temperature, relative humidity, barometric pressure, wind speed and direction and precipitation, are measured with a weather station module. All the measurements are sent via Global System for Mobile Communications (GSM) technology, to a gateway, which consists of a GSM module, the weather station module, and a microcontroller. A control system based on Lab-View receives the readings via Short Message Service (SMS) and stores them in a MySQL database. The use of GSM makes the system's energy consumption and implementation cost high compared to other technologies such as LoRa.

This paper presents an implementation of a monitoring node, for air quality, pollutants, temperature, and humidity measurement, in the outdoor context, where communication via LoRa is used to send readings to a cloud LoRa server. A programmable tool is also implemented on a Raspberry Pi 4 Model B, to receive by the Message Queuing Telemetry Transport (MQTT) protocol, the readings coming from the LoRa server and send them to a database. A mechanism for changing the periodicity of measurements sent to the LoRa server is also implemented, using a traffic index API. Finally, a web application is presented that allows the visualization and analysis of sensor data in real time.

III. SYSTEM DESCRIPTION AND OPERATION

The work mainly describes the LoRa outdoor air quality monitoring. A monitoring node, composed by several sensors, a GPS module, a portable battery, and a power supply board, all connected to a single microcontroller, was created for this purpose. Information about gas concentrations, particulate matter concentration, temperature, relative humidity, air quality index, GPS coordinates and the frequency with which readings are transmitted, is sent in a single transmission via LoRa. These readings are sent to a public gateway and then forwarded to a LoRa server. To create an instance of a LoRa network, which allows running a server that receives the readings coming from the monitoring node, The Things Network (TTN) platform is used. The gateways used for this implementation, are deployed by users who belong to the TTN community, and as LoRaWAN operates in unlicensed bands, any user can install and register the gateway on the TTN platform. The map of TTN registered gateways in the Lisbon area, where the measurements will be gathered, can be found on the TTN webpage [15]. After receiving the measurements at the TTN, they are sent through the MQTT protocol to an instance of a programmable tool, Node-Red, which is running on a Docker, inside a Raspberry Pi 4 Model B. Besides the reception of the readings from TTN, in Node-red is also done the filtering of the data, the integration of some API's and online services and the implementation of the functionality of change of periodicity of samples. From downlink messages (from Node-Red to the microcontroller), the sample rate of measures is changed when any of the imposed levels changes. The readings obtained are stored in Firebase [16], a set of computing services from Google, which hosts a database in the cloud. The application created allows the user to view all readings obtained by the sensors in real time and readings obtained from previous days. The application runs on web and Android platforms. Below, a schematic summarizing the communication of the entire IoT ecosystem created, is presented:

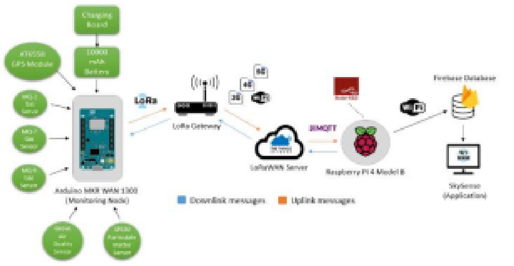


Fig. 1. Workflow and architecture of the IoT system

A. *Hardware description*

The main component of the monitoring node is the Arduino MKR WAN 1300 board. It allows communication via LoRa technology and can be connected to existing LoRa cloud platforms such as TTN. It has a Cortex-M0+ 32-bit SAMD21 microcontroller, which operates with a voltage of 3.3V. The power supply of the board is capped at 5V and can be supplied with a battery or through a micro-USB connection. Finally, it features a clock speed on the processor of 48MHz and on the RTC of 32,768kHz. The antenna power with this board can reach up to 2 dBs.

For the detection of certain polluting gases, in this case CO, LPG and Alcohol, MQ gas sensors are used. These sensors have a metal filament that is heated to get more accurate readings. When the gas encounters the oxidizable metal, it modifies his conductivity, which translates into a resistance value and the lower the resistance of the sensor, the higher the concentration of that gas, and vice versa [8], [9].

The MQ sensors used for this monitoring node and the gases each one is capturing, are listed below:

- MQ-7: Carbon Monoxide (CO).
- MQ-2: Alcohol.
- MQ-9: Liquefied petroleum gas (LPG).

For a general air quality control, the Grove Air Quality Sensor v1.3 is used. It controls the air quality of the environment by detecting gases that are harmful to health, but due to its reading mechanism, it can only reproduce one result and not several quantitative results for each concentration of each gas. One of the limitations of this sensor is the decreased sensitivity if it is exposed to polluted air for a long time.

When it comes to particulate matter concentration measurement, the SPS30 sensor was chosen to perform this function. The working principle of the SPS30 is based on the scattering of the internal infrared laser, which is then detected by the photo detector [8]. Factors such as temperature and humidity can affect the measurement accuracy. It can be used to detect particles with 1, 2.5, 4 and 10 µm [17]. In the case of this implementation, only particles with 2.5 and 10 µm are of concern to us for further analysis.

Parameters such as temperature and relative humidity are obtained through the DHT22 sensor. It uses a capacitive humidity sensor and a thermistor, that measures the surrounding environment in which it is located, producing a digital signal for the collected data. It's a sensor that has a reading range of 0 to 100% for relative humidity and -40ºC to 80ºC for temperature and can operate at a voltage between 3 to 5V. It has a low power consumption and has a better accuracy (about ±0.5°C and ±2%RH 171 measurement error) compared to the DHT11 sensor [18].

Since this is a portable monitoring node, the main supply is through a battery with 10000 mAh and a charging board.

For the Node-Red instance to be running 24 hours a day, the Raspberry Pi 4 Model B was chosen for this role. It has a quad-core 64-bit ARM-CORTEX A72 processor running at 1.5 GHz and 4 GB of RAM.

B. *Software description*

The system firmware that includes sensor data acquisition, fetching GPS coordinates and transmitting packets, with all this information, via LoRa, was developed using the Arduino IDE. It has a huge variety of libraries which helps the construction of important functions to perform certain monitoring implementations.

To obtain more accurate results, MQ sensors undergo a calibration process. In this process, the signal related to the output voltage obtained by the sensor, is analog, but this value is later converted to a digital value. The MKR WAN 1300 board has an ADC with a resolution of 12 bits, obtaining a range of digital values from 0 to 4095. For the determination of the gas concentration, the value of R_S/R_0 is considered, therefore it is necessary to calculate these two values. From the schematic of the MQ sensors and the calibration curve of known gases, the following equations are originated:

$$R_S = \left(\frac{VCC}{VRL} - 1\right) RL \qquad (1)$$

$$R_0 = \frac{Rs}{resistance\ ratio\ in\ fresh\ air} \qquad (2)$$

where Rs is the sensor resistance that varies with the gas concentration, V_{CC} is the sensor supply voltage, V_{RL} is the output voltage of the sensor circuit and R_0 is the value of the sensor resistance, at a known concentration, without the presence of gases. The resistance ratio in fresh air is a value taken from the sensitivity curves of each sensor.

As the characteristic sensitivity curves are on a logarithmic scale, points were extracted from the respective lines for each gas to be measured. From these points, charts were generated, on a linear scale, also relating gas concentration and the R_S/R_0 ratio. To obtain the equation that allows the gas concentration to be calculated more precisely, a trendline was used. From the available trendlines, the one with the smallest error and therefore the most accurate results were a power trendline. Based on this trendline, the following expression was obtained for calculating the gas concentration:

$$c\,[\text{ppm}] = a \times \left(\frac{Rs}{R0}\right)^{-b} \quad (3)$$

a and b are coefficients, that vary according to the points on the generated trendline, and c is the concentration of the gas, in parts per million (ppm). From this equation, substituting the value of R_S/R_0 for the value obtained by the sensor, gives a concentration value.

TTN was chosen as the basis for the LoRa server because due to its capabilities, is easy to configure with the Arduino MKR WAN 1300 board, and offers integrations with different services and protocols, such as the MQTT protocol. To register the board in the LoRa network, created by TTN, some parameters are required, AppEUI, DevEUI and AppKey, being the AppEUI and AppKey randomly generated by TTN and DevEUI represents the device identifier, which is predefined by the manufacturer.

The software used on the Raspberry Pi, is Node-Red [19]. It's built on Node.js and is deployed locally (localhost).

In Node-Red, APIs are integrated in the instance that is running on the Raspberry Pi, to obtain certain parameters, the level of traffic in each location, the exact address of the location where the monitoring node is located through the coordinates of the GPS module and connection to the Firebase database for storing the readings. The data storage was separated into two databases. The Realtime Database is used to store only the latest readings obtained by the sensors and in the Firestore Database all readings associated with the node, are stored.

From the TomTom API, the smart sampling rate feature was created. This API retrieves the speed of the traffic flow and the ideal speed of the traffic flow. The further away the value of the traffic flow speed is from the ideal speed, the more traffic flow there is at the specific location. For defining traffic levels, the ideal speed value was divided into 4 intervals. Each interval indicates a traffic level, and each of these levels has an associated value for the frequency of sending readings (in minutes), which is set manually. Every time the traffic level changes, a downlink message is sent to the Arduino board, and the frequency to send readings, is updated.

The application used for data visualization and analysis was developed based on Flutter. This framework is based on the Dart programming language. An authentication service from Firebase, Firebase SDK Authentication, is also used in this application, where the user can register in the application, log in or access the application with their Google account.

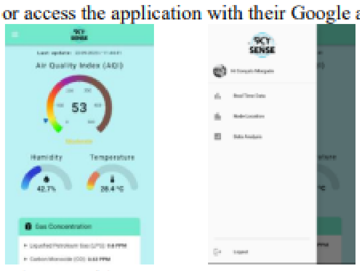


Fig. 2. Main screen of the app

After logging in, the user will be able to find the real-time measurements on the "Real Time Data" page and can also navigate between the "Real Time Data", “Node Location” and "Data Analysis" pages, from a drawer, which can be seen in the Figure 2.

To generate a chart with the measurements obtained, the user selects the option "Data Analysis" from the drawer, and after selecting this tab, he has 3 fields to pick, shown in the Figure 3. The fields “Select a location”, “Select a date” and “Select a measurement” are related to location, date, and measurement to be displayed, respectively. Only after these fields are selected, a line chart is generated with all the readings related to the selected measure, in the desired location and date. It is also possible to visualize some statistical data (average, maximum and minimum value) about these measurements.

In the tab “Node Location”, the user can check the last measurement location from the monitoring node, and if he clicks on the marker, he gets information about the temperature, relative humidity, and air quality index. The circle around the marker changes color according to the measured air quality index value.

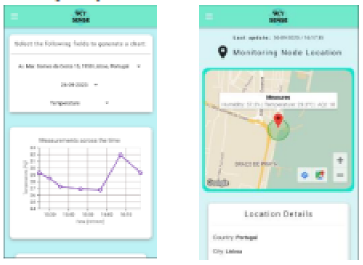


Fig. 3. "Data Analysis" and "Node Location" pages

IV. Results

A. MQ sensors calibration

The process of calibration takes into account the calibration curves of each of the sensors [20], [21], [22].

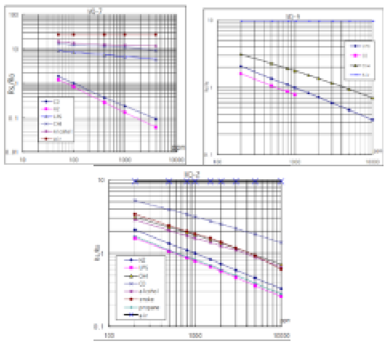


Fig. 4. Calibration curves specified by the manufacturer for MQ-7, MQ-9, and MQ-2 sensors (variation of the sensor resistance ratio [R_S/R_0] and gas concentration)

Considering the selected gases for each sensor to detect, the R_L (load resistance) value used for each sensor is considered. In the case of the MQ-7 and MQ-9 sensors, R_L=10kΩ was considered and for the MQ-2, R_L=5kΩ. The values considered for R_S/R_0 ratio in fresh air, were the following:

TABLE I. R_S/R_0 RATIO IN FRESH AIR FOR EACH MQ SENSOR

	MQ-2	MQ-7	MQ-9
R_S/R_0	9.7	27	9.8

Initially, the sensors were left to warm up for 24 hours (time recommended by the manufacturers) to obtain more accurate results. After this period, the R0 value was measured for 15 minutes, and it was found that the result during this time interval maintained a steady oscillation. The values used for the R_0 of each sensor are shown in Table 2:

TABLE II. R_0 VALUES FOR THE RESPECTIVE SENSORS

	MQ-2	MQ-7	MQ-9
R_0 [kΩ]	4.08	22.11	33.34

This value is then used to calculate the concentration (using the Equation (3)) of the gases under consideration.

B. Measurements results

The monitoring tests were conducted in an outdoor context, and therefore 2 locations were chosen for the value comparison, in the same day, but at different times of the day. The locations can be seen on the Figure 5:

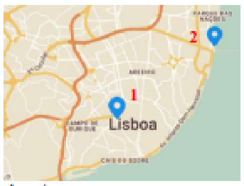


Fig. 5. Measurement's locations

The first location (Av. da Liberdade 258, 1250-149 Lisboa, Portugal), which is labeled on the map with the number 1, relates to an avenue located in the center of Lisboa. The measurements at this location were taken from 11:35 to 14:15. The second location (Av. Mar. Gomes da Costa 15, 1950 Lisboa, Portugal) corresponds to a park located near the Tejo River, labeled on the map with the number 2, also in Lisbon. Compared to the first location, fewer measurements were taken at this place, with results only being obtained between 15:24 and 16:30.

In the charts, a 5-minute interval is imposed between the first and second measurement. This is a manually defined default value. From the second measurement onwards, there is a change in the measurement's frequency, generated by a downlink message sent from the Node-Red to the monitoring node. In the measurements taken on the avenue, the interval between measurements was 8 minutes, and sometimes decreased to 7 minutes due to the level of traffic in the area. In the park located near the Tejo River, there was practically no traffic, so the interval between measurements was longer - 10 minutes for this area.

The more traffic there is in each location, the shorter the interval between measurements, and vice versa.

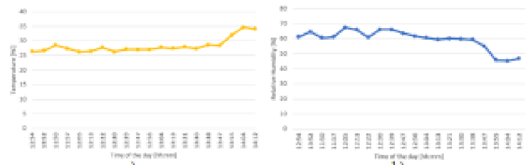


Fig. 6. Temperature a) and Relative Humidity b) for first location (Lat: 38.724175, Lon: -9.148876)

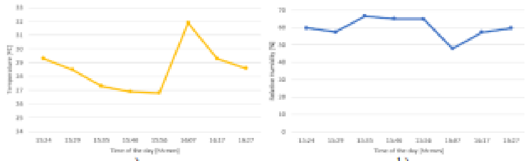


Fig. 7. Temperature a) and Relative Humidity b) for second location (Lat: 38.753357, Lon: -9.094631)

For the first location, the temperature remained stable between 11:35 and 13:45. Around 13:45, it was noted that the temperature began to increase, since the sun began to be more intense at the monitoring location, causing the temperature to rise. As for humidity, throughout the measurement period, there were oscillations between 11:35 and 12:30, but from that moment, the value began to decrease, reaching its lowest value at 14:04. As the temperature increased, relative humidity decreased.

The second location's temperature values didn't vary much either, with the maximum value during the measurement period being reached at 16:07. The variation that occurred between 15:56 and 16:07 can be explained by the absence of wind, which caused the temperature to increase. When it comes to relative humidity, like what was observed at the first measurement point, it behaves in the opposite way to the temperature. The lowest relative humidity value (around 48%) and the highest temperature value (31.9ºC) were obtained at 16:07, justifying this inverse behavior.

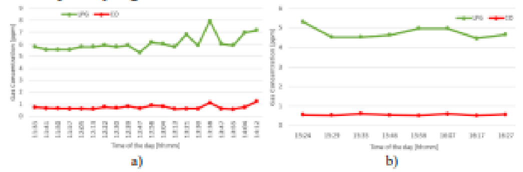


Fig. 8. LPG and CO concentration for first location a) (Lat: 38.724175, Lon: -9.148876) and for second location b) (Lat:38.753357, Lon: -9.094631)

As this second location refers to a park by the Tejo River, the relative humidity values are usually higher than those obtained at the first location.

Over the period of measurements, the behavior of pollutants such as LPG and CO were very similar: every time the concentration of one pollutant increased, the concentration of the other increased as well. These measurements were carried out on an avenue, where there was constant traffic, which made it favorable for the existence of pollutants in the air. At certain times during the measurement period, there were some peaks in the concentration of pollutants. These peaks can be explained by the fact that this avenue has traffic lights, and every time there was a stoppage in traffic, the concentration of pollutants increased. Despite the stoppage, the vehicles' engines kept running, causing the concentration of LPG and CO to increase.

In the second period of measurements, at the park next to the Tejo River, the concentration of LPG and CO was lower, since there were no cars traveling near the monitoring area. However, this park is also next to a dock, where there were a small number of boats in operation, but which contributes to the concentration of these pollutants, which can be seen by some higher values observed in the Fig. 8 and 9.

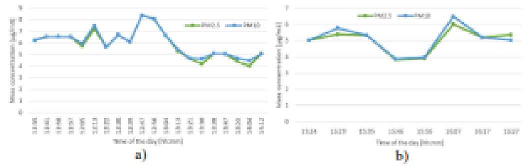


Fig. 9. PM2.5 and PM10 concentration for first location a) (Lat: 38.724175, Lon: -9.148876) and for second location b) (Lat:38.753357, Lon: -9.094631)

As in the case of LPG and CO, the concentration of PM2.5 µm and PM10 µm is higher at the first location, on the avenue, due to the high number of vehicles on the road. The peaks seen in figure 10 can be explained by the large amount of traffic flowing through the area during the time periods.

As the gas concentration and particulate matter values was increasing, the air quality index was also increasing. This index considers the values from many pollutants, and with the higher concentration of pollutants, the value of the index increases, indicating higher air pollution.

V. Conclusion and Future Work

A smart monitoring IoT system was proposed, consisting of several sensors to extract information about air quality condition in the city. In this system, sensors were used to capture concentrations of pollutant gases, particulate matter, air quality and atmospheric factors such as temperature and humidity. The integration of a GPS module allowed the localization of the monitoring system, for later analysis of results. Since this system is designed for an outdoor scenario, it was required that the readings could be sent from any location. For this reason, LoRa technology was used, since it has a low energy consumption and a long range. The use of LoRa technology for outdoor monitoring contexts was a good choice, mainly due to its wide coverage.

To obtain a smart implementation, allowing the change the periodicity for sending readings, an urban traffic API was used. With this API it was possible to set levels of traffic, manually and define thresholds for each of these levels.

For future work, is intended to implement a deep sleep state in the monitoring node, to decrease its energy consumption.

Acknowledgment

The work was supported by Fundação para a Ciência e Tecnologia, UIDB/50008/2020, project, ISCTE – Instituto Universitário de Lisboa and Instituto de Telecomunicações, Lisbon, Portugal

References

[1] 'Air pollution'. Accessed: Jan. 24, 2023. [Online]. Available: https://www.who.int/health-topics/air-pollution

[2] 'Ambient (outdoor) air pollution'. Accessed: Jan. 24, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health

[3] H. Ritchie and M. Roser, 'Outdoor Air Pollution', *Our World in Data*, Nov. 2019, Accessed: Jan. 24, 2023. [Online]. Available: https://ourworldindata.org/outdoor-air-pollution

[4] J. Lozano *et al.*, 'Personal electronic systems for citizen measurements of air quality', in *2019 5th Experiment International Conference (exp.at'19)*, Jun. 2019, pp. 315–319. doi: 10.1109/EXPAT.2019.8876471.

[5] L. Chettri and R. Bera, 'A Comprehensive Survey on Internet of Things (IoT) Toward 5G Wireless Systems', *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 16–32, Jan. 2020, doi: 10.1109/JIOT.2019.2948888.

[6] M. Abbasi, S. Khorasanian, and M. H. Yaghmaee, 'Low-Power Wide Area Network (LPWAN) for Smart grid: An in-depth study on LoRaWAN', in *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)*, Feb. 2019, pp. 022–029. doi: 10.1109/KBEI.2019.8735089.

[7] U. Raza, P. Kulkarni, and M. Sooriyabandara, 'Low Power Wide Area Networks: An Overview', *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 855–873, 2017, doi: 10.1109/COMST.2017.2652320.

[8] W. Y. Yi, K. M. Lo, T. Mak, K. S. Leung, Y. Leung, and M. L. Meng, 'A Survey of Wireless Sensor Network Based Air Pollution Monitoring Systems', *Sensors*, vol. 15, no. 12, Art. no. 12, Dec. 2015, doi: 10.3390/s151229859.

[9] F. Concas *et al.*, 'Low-Cost Outdoor Air Quality Monitoring and Sensor Calibration: A Survey and Critical Analysis', *ACM Trans. Sen. Netw.*, vol. 17, no. 2, p. 20:1-20:44, May 2021, doi: 10.1145/3446005.

[10] G. Gupta and R. V. Zyl, 'Energy harvested end nodes and performance improvement of LoRa networks', *International Journal on Smart Sensing and Intelligent Systems*, vol. 14, no. 1, pp. 1–15, Jan. 2021, doi: 10.21307/ijssis-2021-002.

[11] K. M. Simitha and M. S. Subodh Raj, 'IoT and WSN Based Air Quality Monitoring and Energy Saving System in SmartCity Project', in *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, Jul. 2019, pp. 1431–1437. doi: 10.1109/ICICICT46008.2019.8993151.

[12] A. James, A. Seth, and S. C. Mukhopadhyay, 'IoT enabled sensor node: a tutorial paper', *International Journal on Smart Sensing and Intelligent Systems*, vol. 13, no. 1, pp. 1–18, Jan. 2020, doi: 10.21307/ijssis-2020-022.

[13] S. Walling, J. Sengupta, and S. Das Bit, 'A Low-cost Real-time IoT based Air Pollution Monitoring using LoRa', in *2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Dec. 2019, pp. 1–6. doi: 10.1109/ANTS47819.2019.9117963.

[14] J.-H. Liu *et al.*, 'An Air Quality Monitoring System for Urban Areas Based on the Technology of Wireless Sensor Networks', *International Journal on Smart Sensing and Intelligent Systems*, vol. 5, no. 1, pp. 191–214, Jan. 2012, doi: 10.21307/ijssis-2017-477.

[15] T. T. Network, 'Lisbon', The Things Network. Accessed: Jul. 31, 2023. [Online]. Available: https://www.thethingsnetwork.org/community/lisbon/

[16] 'Noções básicas | Documentação do Firebase', Firebase. Accessed: Jul. 31, 2023. [Online]. Available: https://firebase.google.com/docs/guides?hl=pt-br

[17] 'Sensirion_PM_Sensors_Datasheet_SPS30.pdf'. Accessed: Jul. 20, 2023. [Online]. Available: https://sensirion.com/media/documents/8600FF88/616542B5/Sensirion_PM_Sensors_Datasheet_SPS30.pdf

[18] 'Liu - Capacitive-type humidity and temperature modulese.pdf'. Accessed: Jul. 29, 2023. [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf

[19] 'About: Node-RED'. Accessed: Jul. 29, 2023. [Online]. Available: https://nodered.org/about/

[20] '605-00008-MQ-2-Datasheet-370464.pdf'. Accessed: Jul. 29, 2023. [Online]. Available: https://www.mouser.com/datasheet/2/321/605-00008-MQ-2-Datasheet-370464.pdf

[21] 'MQ-7.pdf'. Accessed: Jul. 29, 2023. [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf

[22] 'MQ-9_Hanwei.pdf'. Accessed: Jul. 29, 2023. [Online]. Available: https://www.electronicoscaldas.com/datasheet/MQ-9_Hanwei.pdf