UNIVERSIDADE
BEIRA INTERIOR

A Machine Learning-based Decision Support System for IoT Monitoring Solutions
(Versão Final Após Defesa)

Edgar Filipe Loureiro Ladeira

Relatório de Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor Bruno Miguel Correia da Silva

janeiro de 2025

Declaração de Integridade

Eu, Edgar Filipe Loureiro Ladeira, que abaixo assino, estudante com o número de inscrição M12437 do Mestrado de Engenharia Informática da Faculdade de Engenharia, declaro ter desenvolvido o presente trabalho e elaborado o presente texto em total consonância com o **Código de Integridades da Universidade da Beira Interior.**

Mais concretamente afirmo não ter incorrido em qualquer das variedades de Fraude Académica, e que aqui declaro conhecer, que em particular atendi à exigida referenciação de frases, extratos, imagens e outras formas de trabalho intelectual, e assumindo assim na íntegra as responsabilidades da autoria.

Universidade da Beira Interior, Covilhã 09/01/2025

Assinado por: **Edgar Filipe Loureiro Ladeira**
Num. de Identificação: 14758067
Data: 2025.01.10 14:38:45 +0000

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my advisor, Prof. Dr. Bruno Silva, for the invaluable support and guidance provided throughout the development of this project. His mentorship was crucial for my growth in the field of networks and IoT, and I am deeply grateful for the generosity with which he shared his knowledge throughout the entire academic process.

Additionally, I am very thankful to the entire RuraLTHINGS team, as well as the researchers at the SINS Lab, for their attention and support during the most delicate moments when I needed assistance. Their constant availability and presence were essential throughout this entire journey.

I would also like to acknowledge the support for this work provided by the Instituto de Telecomunicações (IT) Portugal, financed by the "la Caixa" Foundation, in partnership with the Foundation for Science and Technology (FCT) and BPI. This thesis was prepared at the Secure and Intelligent Networked Software Systems Laboratory (SINS-Lab).

Last but certainly not least, I would like to express my deep gratitude to my family and friends, who have always believed in me and my abilities. They stood by my side through both joyful and challenging moments, serving as my motivation and support to overcome obstacles and maintain a positive attitude. I am especially grateful to my parents, whose unconditional support was fundamental for me to embark on and overcome this challenge.

This research work was carried out at and with the partial support of the Instituto de Telecomunicações, Covilhã, Department of Computer Science of the University of Beira Interior. Specifically, at the Network Applications and Services-CV Research Cluster, sins-Lab.

Resumo

A nível global, os incidentes ambientais ocorrem diariamente devido à emissão de gases poluentes, sendo que muitos deles são as principais causas de desastres ambientais e mortes prematuras. O impacto significativo desses gases na saúde pública é uma grande preocupação, especialmente em regiões do interior. A inalação desses compostos frequentemente afeta a vida cotidiana e a saúde das pessoas que vivem nessas áreas, muitas vezes sem que tenham consciência dos riscos envolvidos.

O objetivo deste projeto é desenvolver um sistema de apoio à decisão, impulsionado por Inteligência Artificial (IA), e uma plataforma robusta para a visualização de estatísticas sobre gases poluentes. Utilizando um ecossistema de Internet das Coisas (IdC), o sistema tem o objetivo de notificar os moradores relativamente às condições ambientais e evitar quaisquer riscos, incluindo a exposição a gases perigosos. Os dados são recolhidos por sensores digitais, agregados por um microprocessador e transmitidos para uma base de dados centralizada. Essa base de dados alimenta a plataforma digital que disponibiliza gráficos abrangentes e de fácil navegação, ajudando os utilizadores a compreender claramente os níveis de poluição. Essas visualizações auxiliam autoridades e utilizadores a tomarem decisões informadas sobre a segurança ambiental.

Este trabalho visa encontrar o melhor modelo de ML para este tipo de dados (séries temporais), a fim de gerar previsões precisas, focando na investigação de diversas técnicas de ML. O sistema de apoio à decisão baseado em IA utiliza técnicas de *machine learning* (ML) para analisar os dados recolhidos e fornecer avaliações dos níveis de gases perigosos, bem como previsões futuras destes, utilizando camadas Long Short-Term Memory (LSTM) unidirecionais sobrepostas a camadas LSTM bidirecionais, uma arquitetura que apresentou os melhores resultados. Em áreas onde gases poluentes como o radão são comuns, essas previsões permitem gerar alertas oportunos e recomendações para reduzir os riscos à saúde.

De forma geral, este estudo contribui para o desenvolvimento de um sistema inovador, a partir da análise e tratamento de dados sensoriais, como gases poluentes e variáveis ambientais. Em um contexto real, a previsão é integrada na plataforma RuraLTHINGS, fornecendo aos utilizadores do sistema informações para tomar decisões mais conscientes sobre os seus espaços interiores, com o objetivo de promover melhores condições de vida.

Palavras-chave

Internet das Coisas, Monitorização Ambiental, Redes de Sensores, Processamento de Dados, Saúde Pública, Gases Ambientais, Sistema de Apoio à Decisão, Aprendizagem Automática, Previsão de Séries Temporais

Resumo alargado

Nos últimos anos, têm-se verificado sérios problemas de saúde pública, que têm contribuído para um aumento do número de mortes, especialmente entre a população idosa. Esta vulnerabilidade deve-se ao facto de passarem grande parte do tempo em casa ou em instituições de cuidado, muitas vezes expostos a condições ambientais inadequadas.

O presente projeto visa analisar e monitorizar as condições ambientais no interior das casas, com o objetivo de melhorar a qualidade de vida e reduzir os riscos à saúde pública. Neste contexto, a IdC associada ao *machine learning* desempenha um papel crucial. Por meio de dispositivos interconectados, é possível recolher dados em tempo real dos níveis de gases poluentes, permitindo respostas rápidas a variações significativas e utilizando sistemas de apoio à decisão, que são essenciais para minimizar a exposição a condições prejudiciais.

O objetivo principal deste trabalho é desenvolver e estudar mecanismos de suporte à decisão baseados em IA, integrados a uma plataforma digital, utilizando tecnologia IdC para monitorizar gases poluentes. Esses sistemas proporcionam à plataforma a capacidade de exibir gráficos interativos com dados em tempo real, apresentando informações relevantes sob a forma de gráficos, notificações e análises preditivas. Tais ferramentas ajudam os utilizadores a entender corretamente as medições e a tomar ações apropriadas para reduzir a exposição a gases perigosos, contribuindo assim para a tomada de decisões no âmbito da saúde pública.

Para selecionar o modelo preditivo mais adequado para este tipo de dados, foram considerados dados temporais e diversos algoritmos que melhor se ajustam a estas características, além de serem comparadas as suas métricas de desempenho para definir os parâmetros ideais. Após a escolha do modelo final, este foi integrado à plataforma digital. Esse poder preditivo é fundamental para orientar políticas de saúde pública em diferentes níveis, bem como para os próprios utilizadores. A capacidade do sistema de avaliar e prever ameaças ambientais cria as bases para um ambiente de vida mais seguro e saudável, promovendo melhorias contínuas na gestão da qualidade do ar interno e na tomada de decisões.

Este trabalho está inserido no projeto RuraLTHINGS, financiado pela Fundação la Caixa, no âmbito do Programa PROMOVE. Este projeto europeu visa o desenvolvimento de um sistema abrangente baseado em IoT para monitorizar condições ambientais em áreas rurais e remotas, focando-se em gases poluentes como o radão e o monóxido de carbono. A iniciativa tem como objetivo melhorar a saúde pública por meio de monitorização em tempo real, análises preditivas e mecanismos de apoio à decisão, melhorando a qualidade do ar interior, especialmente em áreas com infraestuturas reduzidas e ligações limitadas.

Este trabalho de investigação foi realizado com o apoio parcial do Instituto de Telecomunicações, Covilhã, Departamento de Engenharia Informática da Universidade da Beira Interior, especificamente, no Secure and Intelligent Networked Software Systems Laboratory, sins-Lab.

Abstract

Globally, environmental incidents occur daily due to the emission of pollutant gases, many of which are the main causes of environmental disasters and premature deaths. The significant impact of these gases on public health is a major concern, especially in rural regions. Inhalation of these compounds often affects the daily lives and health of people living in these areas, often without their awareness of the risks involved.

The goal of this project is to develop a decision support system powered by Artificial Intelligence (AI) and a robust platform for visualizing statistics on pollutant gases. Using an Internet of Things (IoT) ecosystem, the system aims to notify residents about environmental conditions and prevent any risks, including exposure to hazardous gases. Data is collected by digital sensors, aggregated by a microprocessor, and transmitted to a centralized database. This database feeds into the digital platform, which provides comprehensive and easy-to-navigate graphs, helping users to clearly understand pollution levels. These visualizations assist authorities and users in making informed decisions about environmental safety.

This work aims to find the best Machine Learning (ML) model for this type of data (time series), in order to generate accurate predictions, focusing on the exploration of various ML techniques. The AI-based decision support system uses ML techniques to analyze the collected data and provide assessments of hazardous gas levels, as well as future forecasts of these levels, using unidirectional LSTM layers overlaid on bidirectional LSTM layers, an architecture that has shown the best results. In areas where pollutant gases such as radon are common, these forecasts enable the generation of timely alerts and recommendations to reduce health risks.

Overall, this study contributes to the development of an innovative system through the analysis and processing of sensory data, such as pollutant gases and environmental variables. In a real-world context, the forecast is integrated into the RuraLTHINGS platform, providing system users with information to make more informed decisions about their indoor spaces, with the goal of promoting better living conditions.

Keywords

Internet of Things, Environmental Monitoring, Mobile Sensing Networks, Data Processing, Public Health, Environmental Gases, Decision Support System, Machine Learning, Time-series Forecasting

Contents

List of Figures

List of Tables

Acronyms List

ACF	AutoCorrelation Function
ACID	Atomicity, Consistency, Isolation, Durability
ADF	Augmented Dickey-Fuller
AI	Artificial Intelligence
AIC	Akaike Information Criterion
ANN	Artificial Neural Network
APA	*Agência Portuguesa do Ambiente*
API	Application Programming Interface
AQI	Air Quality Index
AR	Auto-Regressive
ARIMA	Auto-Regressive Integrated Moving Average
AWS	Amazon Web Services
BIC	Bayesian Information Criterion
BLE	Bluetooth Low Energy
BTEX	Benzene, Toluene, Ethylbenzene, and Xylenes
CNN	Convolutional Neural Network
CoAP	Constrained Application Protocol
DSS	Decision Support System
ECG	Electrocardiogram
EDA	Exploratory Data Analysis
EEA	European Environment Agency
EPA	Environmental Protection Agency
FCM	Firebase Cloud Messaging
GPRS	General Packet Radio Service
GPU	Graphics Processing Unit
GRU	Gated Recurrent Units
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IDE	Integrated Development Environment

IMAP Internet Message Access Protocol

IoT Internet of Things

IT Information Technology

JSON JavaScript Object Notation

LoRaWAN Long Range Wide Area Network

LPWAN Low-Power Wide-Area Network

LR-WPAN Low-Rate Wireless Personal Area Network

LSTM Long Short-Term Memory

MA Moving Average

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

ML Machine Learning

MQTT Message Queuing Telemetry Transport

MSE Mean Squared Error

MVC Model-View-Controller

NaN Not a Number

NB-IoT Narrowband Internet of Things

PACF Partial AutoCorrelation Function

PAN Personal Area Network

PCA Principal Component Analysis

PCS *Portal de Construção Sustentável*

POP3 Post Office Protocol3

RDBMS Relational Database Management System

RMSE Root Mean Square Error

RNN Recurrent Neural Network

SARIMA Seasonal Auto-Regressive Integrated Moving Average

sMAPE Symmetric Mean Absolute Percentage Error

SMS Short Message Service

SMTP Simple Mail Transfer Protocol

SQL Structured Query Language

SVM	Support Vector Machine
TCP/IP	Transmission Control Protocol / Internet Protocol
TLS	Transport Layer Security
UML	Unified Modeling Language
UNSCEAR	United Nations Scientific Committee on the Effects of Atomic Radiation
UWB	Ultra Wideband
VOC	Volatile Organic Compound
WHO	World Health Organization
Wi-Fi	Wireless Fidelity
WSN	Wireless Sensor Networks

Chapter 1

Introduction

This Chapter presents the factors that motivated the research conducted in this document and defines the objectives and potential contributions achieved through this investigation. Subsequently, the global structure of the current document will be described. Finally, a brief project description linked to the work mentioned in this document will be provided.

1.1 Context and Motivation

The elderly population often spends most of their time at home or in care facilities. This is alarming because of the poor indoor conditions, posing significant public health concerns. The occurrence of deaths due to poor environmental conditions is also a motivation for the development of this project, as the analysis and monitoring of indoor conditions are essential to provide the best possible quality of life and to reduce the health risks associated with the population.

The need for an innovative and effective approach to monitor and minimize exposure to unfavorable conditions is evident. It is in this context that IoT plays a fundamental role. By employing interconnected devices, it is possible to collect real-time data on pollutant gas levels.

Following the 2012 version of the Association for Computing Machinery (ACM) Computing Classification System (CCS), the scope of the master project is defined by the following categories:

- **Applied computing - Life and medical sciences - Health care information systems;**
- **Computing methodologies - Neural networks;**
- *Computing methodologies - Classification and regression trees;*
- *Computing methodologies - Feature selection;*
- **Computing methodologies - Supervised learning;**
- **Human-centered computing - User interface programming;**
- *Human-centered computing - Mobile computing;*
- *Hardware - Communication hardware, interfaces and storage - Sensor devices and platforms.*

1.2 Objectives

This project focuses on the study and development of machine learning-based decision-support mechanisms integrated into a digital platform employing IoT technology to monitor and analyze environmental conditions in houses, including pollution gases. These systems will drive the platform and provide interactive graphs with real-time data displaying smart dashboards. These tools help consumers to properly understand measurements and act wisely to lower their contact with dangerous gases. Apart from guaranteeing strong security through user authentication and data encryption, the system will offer predictive insights, send alarms, and assist public health decision-making.

The secondary objectives to achieve this primary goal are:

- **Digital platform development** - Develop a multi-platform application that ensures secure access for a variety of stakeholders, including households, healthcare providers, and policymakers, by incorporating user security through authentication, tokenization, and encryption;

- **Time-series data analysis** - Investigate how environmental data behaves over time, identifying patterns in pollutant levels and environmental conditions. This study will guide the adoption of techniques for manipulating and pre-processing time-series data to ensure accurate analysis;

- **Data preprocessing** - Ensure that the models and forecasts are founded on high-quality information by strategically cleaning and organizing the data through comprehensive preprocessing of acquired datasets;

- **Model testing and evaluation** - Evaluate and contrast a variety of machine learning models to ascertain which one is most effective in predicting pollutant levels. The objective is to select a final predictive model that improves the system's decision-support capabilities by evaluating metrics;

- **Platform adaptation for predictive insights** - Incorporate predictive capabilities that will allow users to obtain predictions of pollutant concentrations and preventive recommendations to mitigate health risks;

- **Alert and notification system** - Create a notification system that will notify users when pollutant levels surpass hazardous thresholds, thereby minimizing potential health risks by providing timely, actionable information;

Furthermore, the system aims to increase awareness of environmental risks, particularly for the elderly, who spend more time indoors. The platform will promote a healthier and safer environment by providing easily accessible tools that allow any user, regardless of technological literacy, to benefit from the available data and insights. In addition, the system will integrate ML techniques to facilitate decision-making by predicting pollutant levels.

1.3 Project RuraLTHINGS

This section provides a summary of the scope of the RuraLTHINGS project and the main stakeholders engaged in its development. It describes the specific objectives and contributions of the research, emphasizing the improvement of public health in rural regions via innovative environmental monitoring technologies.

1.3.1 Project Description

The RuraLTHINGS project aims to implement an intelligent IoT-based monitoring system in rural and remote areas, where Internet access is scarce or nonexistent. The pilot regions are Fundão and Pinhel, where around 10 smart homes in each region will serve as testbeds for monitoring harmful gases such as radon, carbon monoxide (CO), and carbon dioxide (CO_2) [4].

The system will not only monitor these gases but also temperature, humidity, and other environmental factors. The aim is to enhance the overall well-being of residents in these areas by mitigating health risks and preventing diseases like lung cancer, which is often associated with prolonged radon exposure. Ensuring the encryption and security of sensitive data during transmission between IoT devices and the central system is paramount. An additional objective is to develop communication protocols that operate efficiently in areas with limited or unreliable Internet access, thus connecting these zones more intelligently and effectively.

As illustrated in Figure 1.1, the RuraLTHINGS project involves a complex system architecture that requires the integration of several technologies and components, including connections made between devices utilizing varying communication protocols, depending on the situation. This architecture is designed to integrate environmental monitoring devices, forecasting algorithms, and decision support mechanisms, providing a clear view of the data flow from collection to analysis and decision-making. Through this architecture, the system ensures continuous data capture, the application of machine learning techniques for predictions, and efficient data-driven decision support.

The project emphasizes sustainable solutions by integrating low-energy, long-range communication networks such as Long Range Wide Area Network (LoRaWAN), which are ideal for rural deployment. This approach enables continuous monitoring and data collection even in areas with limited infrastructure, ensuring that residents in these regions receive the same level of environmental protection as those in urbanized areas. A network of sensors installed in smart houses plays a crucial role in this system, monitoring radon gas, carbon monoxide, humidity, and temperature levels. These sensors utilize Message Queuing Telemetry Transport (MQTT) over Transport Layer Security (TLS) for secure data transmission, which is then relayed via LoRaWAN and other communication networks. The collected data is subsequently sent to a centralized platform, accessible through web and mobile interfaces, allowing users to monitor data in real-time, review historical trends, receive alerts, and interact with statistical insights.

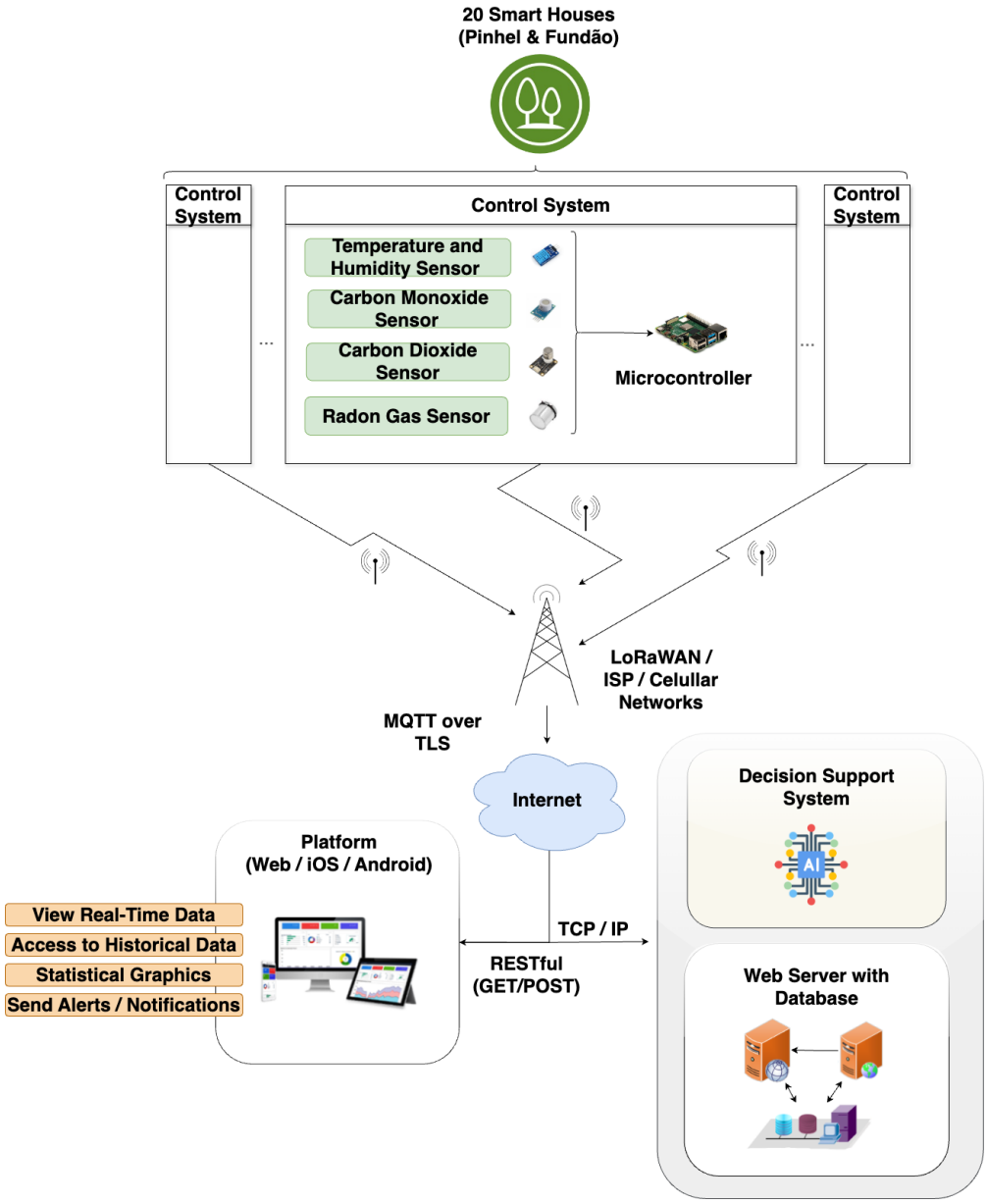

Figure 1.1: System Architecture of the System.

1.3.2 Objectives of the RuraLTHINGS project

This research actively contributes to several key areas:

- **Real-time environmental monitoring** - One of the biggest challenges of this project is to use IoT sensors and technologies to check the air quality and find particles that can cause cancer. The goal is to gather information about the environment and send it through the IoT network. This information will then be shown on a special platform. To do this, it is necessary to carefully think about how the homes are built, where they are located, and their sizes. Finding the right technologies for communication in places with poor connections will be the first step toward making a system that works well in remote areas;
- **Analysis of transmission protocols in rural and remote areas** - Data trans-

mission in an IoT network can be hard in places that are far away and do not have much or any Internet access. The system that needs to be built should not depend too much on being connected to the Internet all the time to send and receive data. The collection and transmission of data should happen in real-time, with few breaks. This will need strong communication protocols to make sure that the system works and is reliable, even when it is not connected to the Internet;

- **Public health improvement through environmental risk mitigation** - The project tackles several key public health concerns by monitoring not only radon gas, which can increase the risk of lung cancer, but also carbon monoxide (CO), a deadly gas produced by incomplete combustion from fires or heating systems. Additionally, it tracks environmental factors such as extreme temperatures, which can cause health issues like heatstroke or hypothermia, and humidity levels that affect respiratory health. By using IoT-based monitoring, especially in rural homes where these risks may be higher, the project aims to reduce exposure to these hazards and raise awareness of environmental dangers;
- **Technological and infrastructural advancements in rural areas** - Another contribution of RuraLTHINGS is the development of communication networks like LoRaWAN, which enable long-distance data transmission with low energy consumption. This contributes to a more inclusive infrastructure that supports technological development in rural and remote regions, making them more connected and self-sustaining;
- **Development of a new radon sensor** - A key aspect of the RuraLTHINGS project is the creation of an advanced radon detection system specifically designed for homes in rural and remote areas. The goal is to develop an active radon sensor that will monitor radon gas levels in real-time, helping to mitigate the health risks associated with prolonged exposure to this radioactive gas, which is a significant cause of lung cancer. This sensor will be installed in homes with IoT technology to help with gas accumulation and minimize them. This innovation is crucial to providing accurate and timely data and ensuring a healthier living environment, especially in areas where traditional solutions may be considered expensive and inefficient.

1.4 Main Contributions

Through the integration of IoT and ML, this project makes a substantial contribution by developing an innovative decision support system based on artificial intelligence for the purpose of gas concentration forecasting. Even users with limited technical knowledge can make informed decisions and take proactive efforts to prevent exposure to dangerous gases thanks to the real-time and predictive information provided by this system. Public health programs benefit from improved estimates and personalized suggestions regarding technology, which lead to better living circumstances.

The decision support system, which uses AI-driven models to predict pollutant concentrations and direct actions for reducing exposure risks, is the central component of this design.

This predictive power is essential for guiding public health policies at all levels and for individual users as well, especially in places where vulnerable populations, like the elderly, are present. In order to provide a safer and healthier living environment, the capacity of the system to monitor, evaluate, and forecast environmental threats lays the groundwork for ongoing improvements in indoor air quality management and decision-making.

As part of this project, a scientific paper titled "A Novel IoT Monitoring System for Remote and Rural Regions" was prepared and presented at the 20th Edition of the International Conference on Wireless Networks and Mobile Devices in Paris, France (WiMob 2024). The paper focuses on the analysis of transmission protocols in rural and remote areas, exploring different approaches and conducting exhaustive tests between various communication protocols. Moreover, it presents the digital platform developed in the scope of this dissertation.

1.5 Document Organization

The dissertation is composed of five chapters, organized as follows:

1. Chapter 1 – **Introduction** - introduces the scope, motivation, and objectives of the RuraLTHINGS project, along with an overview of the contributions of this research;
2. Chapter 2 - **Background and Related Work** - reviews relevant literature on environmental impacts on public health, IoT ecosystems, and decision support systems. It also provides an overview of existing work on digital platforms in IoT systems, setting the foundation for the proposed solution;
3. Chapter 3 - **RuraLTHINGS: Proposed Platform** - presents the system architecture and engineering of the RuraLTHINGS platform, explaining the use cases, system requirements, and design diagrams;
4. Chapter 4 - **Intelligent Decision Support System** - examines the implementation of decision-support functionalities, including data preprocessing, analysis, and machine learning model selection;
5. Chapter 5 - **System Evaluation: Viability and Validation** - evaluates the performance and viability of the proposed platform in real-world scenarios, focusing on the decision-support mechanisms and alert systems. The results are looked at and talked about based on how the system works and how users interact with it;
6. Chapter 6 - **Conclusions and Future Work** - presentation of the final thought about the dissertation and outline of possible future work.

Chapter 2

Background and Related Work

2.1 Introduction

This Chapter provides a brief contextualization, framing the main concepts related to IoT and the importance of its use in daily life, particularly for monitoring values important to public health. Section 2.2 is subdivided into different subsections, each describing a different topic, providing the necessary foundations for this thesis. The use of IoT sensors combined with Decision Support System (DSS) addresses various challenges in the field of public health to be addressed, as described throughout Section 2.2.1. Subsection 2.2.1.1 focuses on the pollutants present in the atmosphere and the importance of monitoring them. Subsection 2.2.1.2 addresses the environmental and physiological problems faced by citizens in many Portuguese homes due to a lack of insulation. Subsection 2.2.1.3 focuses on the monitoring and control of pollutant gases in Portugal, emphasizing the role of IoT technologies in tracking and managing air pollutants such as carbon dioxide (CO_2) and carbon monoxide (CO). It describes how real-time monitoring systems are essential in both urban and indoor environments, particularly in Portugal, where fluctuations in pollutant levels can pose serious health risks. Subsection 2.2.1.4 discusses the lack of infrastructure and communication means in remote and rural areas, which directly affects access to health services, infrastructure, and communication. Following this, Section 2.2.2 introduces the IoT technologies and sensors that enable the collection of environmental data and the monitoring of pollutants. For real-time analysis of the collected values, Section 2.2.3 focuses on the use of DSS to improve the decision-making process to minimize damage and precisely control the collected values. This subsection is divided into three sections: Subsection 2.2.3.1 considers the importance of time series analysis in this type of data, highlighting mechanisms for detecting patterns and trends in this data. Next, Subsection 2.2.3.2 briefly introduces ML concepts and how they can be applied to model and predict pollutant values. Finally, Subsection 2.2.3.3 explores regression algorithms used for prediction and analysis. Section 2.3 introduces the use of IoT technologies and how they can be used for the visualization, analysis, and control of sensory data through digital platforms. Finally, Section 2.4 provides an overall conclusion of the Chapter.

2.2 Background

This Chapter provides an overview of the key components of air quality monitoring, IoT technologies, and DSS. Furthermore, it reviews and compares relevant literature and methodologies that align with the objectives of the RuraLTHINGS project, focusing on the integration of IoT in environmental monitoring and the use of ML for predictive analysis.

2.2.1 Environmental Impacts on Public Health

This section discusses the impact of pollutant gases on public health, emphasizing monitoring and control measures in Portugal. It also underscores the environmental and geographic obstacles of home isolation, especially in rural and remote areas.

2.2.1.1 Pollutant Gases

Pollution levels have been a constant concern for both the global population and ecosystems [5]. Not only is it a problem that can translate into a threat to well-being, but it also jeopardizes the existence of certain species and ecosystems. Air quality in metropolitan areas is greatly impacted by heavy automotive traffic [6], which includes emissions from burning fossil fuels like coal and other harmful chemicals that raise the concentration of pollutants in the atmosphere. Uncontrolled and intensive farming is another activity that has a detrimental impact on climate change.

In 1979, the World Health Organization (WHO) issued its first statement on the risks of gas exposure in buildings. About nine years later, in 1988, based on studies related to radon gas exposure and lung cancer diagnosis, radon gas was characterized as the second leading cause of lung cancer, right after tobacco. Miners' high exposure to elevated concentrations of the gas over long periods, due to the depth of the mines, which creates an environment conducive to the presence of radioactive gases, led to these studies [7].

According to scientists, one of the most recent air quality assessments conducted by the European Environment Agency (EEA) indicates that air pollution in Europe is far above the limits set by the WHO [8]. According to this entity, in 2016, around 4.2 million premature deaths were associated with air pollution in urban and rural areas, categorizing it as the primary environmental risk to health. The report published by WHO states that air pollution is responsible for about 7% of the total number of deaths. Indoor air pollution, caused by the use of polluting fuels for cooking and heating, accounted for 3.8 million of these deaths [9].

The six main pollutants present almost everywhere are fine inhalable particles (PMx), ozone (O3), nitrogen dioxide (NO2), sulfur dioxide (SO2), and carbon monoxide (CO). Some of these well-known gases are produced by human activities. However, there are others that occur naturally in the environment, such as radon gas.

Radon gas (^{222}Rn) is one prominent member of the decay chain of the radioactive element uranium (^{238}U). Radon has a half-life of 3.8 days, and its parent compound (^{238}U) is present throughout the Earth's crust, easily passing into the environment through cracks or free spaces in the soil and rocks, as well as through water. Due to its natural origin, and being undetectable, colorless, odorless, and tasteless, this gas is quite dangerous when inhaled in enclosed and poorly ventilated spaces [10]. Constant monitoring of this component is essential to ensure the safety and comfort of residents indoors.

In 2001, several tests were conducted in different geographic areas to understand the behavior of the gas concerning factors that could influence its existence [11]. From a Hungarian study involving about 15,000 homes, it was concluded that the concentration of radon gas in enclosed environments depends on a series of factors, including air ventilation and the level of insulation [12]. A pioneering study in Europe by Darby et al. (2005, [13]), which included approximately 21.000 measurements from 13 countries, reported that for each 100 Bq/m³ increase in average radon gas concentration to which a person is exposed, the risk of developing lung cancer increases by approximately 16% [14]. Krewski et al. in 2005 corroborated this study and reached similar conclusions [15]. Researchers conducted these tests in an effort to gain a better understanding of the mechanisms causing this gas's rapid dispersion and emission. It was found that radon gas tends to accumulate in basements, garages, and any compartment in direct contact with the soil, as these are more prone to high concentration levels, especially when the soil is rich in uranium. Another fact is that radon gas is about eight times denser than air, so it naturally accumulates in lower areas [16].

According to a scientific assessment published by United Nations Scientific Committee on the Effects of Atomic Radiation (UNSCEAR) in 2006, radon gas is responsible for around 52% of the radiation exposure that people worldwide experience. There are notable differences between nations in the concentration of gas in enclosed spaces. Average values are reported to be less than 10 Bq/m³ in the Middle East, whereas they are more than 100 Bq/m³ in European nations [17].

Although lung cancer associated with radon exposure has been confirmed, it is believed that it may also be a cause of childhood leukemia. In a total of 12 studies conducted to evaluate the correlation between radon gas and childhood leukemia, 11 showed some cause-effect relationship, and 8 of these 12 presented a strong correlation, according to J. Tong et al. [18].

The risk of lung cancer associated with different levels of radon exposure for both non-smokers and smokers is based on data from studies and guidelines provided by the U.S. Environmental Protection Agency (EPA) [19]. The data reveals a clear trend that as radon levels decrease, the risk of lung cancer for non-smokers decreases proportionally, supporting the recommendation by the EPA to implement mitigation efforts when radon levels reach 148 Bq/m³ or higher. Additionally, the significantly higher risk for smokers at all levels of radon exposure compared to non-smokers is evident. The synergistic effect of smoking and radon exposure greatly increases the likelihood of developing lung cancer, and even at lower radon levels, the risk for smokers remains substantial, highlighting the importance of both smoking cessation and radon mitigation.

In Directive 2013/59/Euratom [20], proposed by the Council of the European Union on December 5, 2013, reference levels were established for radon concentrations indoors and for gamma radiation emitted by building materials. According to epidemiological data obtained from studies conducted in homes, the risk of lung cancer was higher with prolonged exposure to indoor radon levels of around 100 Bq/m³. The directive stipulates that these values should not exceed 300 Bq/m³ and requires the implementation of preventive measures in the construction of new buildings.

Unfortunately, a large part of the population is not aware of the existence of radon gas and its harmful effects on health. Citizens should always be concerned about air quality, and measurements should be made on a regular basis, particularly in regions where high concentrations of gases that are harmful to health are likely to occur. In order to preserve public health, the World Health Organization recommends that residents maintain their dwellings with values below 100 Bq/m³.

2.2.1.2 Monitoring and Control of Pollutant Gases in Portugal

The first study on gas concentrations in Portugal began in 1987, conducted by the Department of Radiological Protection and Safety of the National Laboratory of Industrial Engineering and Technology, which is now defunct. This research was conducted in cooperation with local secondary schools. Gas detectors were personally handed over to teachers, who managed their distribution based on a statistical ratio of one detector for every 2000 inhabitants. These sensors were exposed for periods ranging from 1 to 3 months, and whenever possible, measurements were taken twice in each household, totaling more than 4000 homes. Results were obtained from 276 Portuguese municipalities, allowing the verification of the influence of cold and ventilation on radon gas concentration. In summary, the study highlights the need for special attention to areas with high radon concentration and suggests implementing measures to reduce exposure levels, particularly in granite-rich, colder regions with poor ventilation [21].

In Figure 2.1, we can see the map based on a most recent study, which concludes the direct relationship between the type of rock (granite) and the higher concentration of radon and, consequently, radon gas.

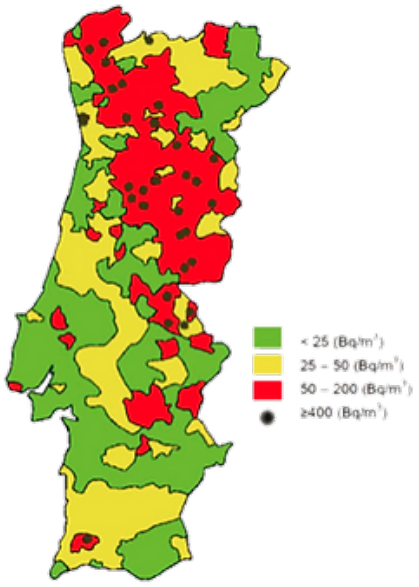

Figure 2.1: Radon Map of Portugal (Lusoradon, 2015, [1]).

The northern regions of Portugal, as well as the interior areas, are the most affected due to the types of rocks and soil present in these areas, as shown in Figure 2.1.

Another study conducted in Portugal aimed to evaluate the risk of lung cancer associated with radon exposure in indoor environments in the northern part of the country. Led by Veloso et al. [22] in 2011, this study estimated the number of lung cancer cases to be between 1.565 and 2.406 during the period from 1995 to 2004. These numbers indicate that of the 8,514 recorded deaths, approximately 18 to 28% could be related to radon exposure. This represents about 150 to 240 individuals annually who lose their lives just in the northern region of Portugal.

In Portugal, in 2013, legislation was enacted, specifically Decree-Law No. 118/2013 of August 20 [23], which establishes norms and requirements for the installation, operation, and maintenance of air conditioning systems in buildings intended for commerce and services. This initiative aims to promote energy performance in buildings and also addresses indoor air quality policies. Ordinance No. 353-A/2013 [24] was subsequently published, which established the minimum fresh air flow rates per space, as well as the protection limits and reference conditions for indoor air pollutants in buildings. This regulation enables the implementation of sustainability assessment systems and substantial interventions in existing buildings.

In indoor environments, the established limits for radon concentrations are 400 Bq/m^3, and continuous monitoring of this gas is necessary. This requirement is especially important for buildings constructed in granite areas, particularly in districts located in the Northern region of Portugal, including Braga, Castelo Branco, Porto, Guarda, Vila Real, and Viseu. These measures are intended to prevent or reduce the concentration of radon indoors [25].

Other studies have also been conducted to evaluate public health hazards, such as one that was conducted in the Euroregion of Galicia-North Portugal and concentrated on classrooms. One-third of the classrooms analyzed had elevated radon gas levels, and approximately 75% of the spaces exceeded 100 Bq/m^3 in these investigations. In areas where the concentration was more pronounced, this analysis enabled the implementation of mitigation measures [26].

National radon gas monitoring campaigns were implemented in 2020 by the *Agência Portuguesa do Ambiente* (APA) in collaboration with the University of Coimbra. Using a detector supplied by the organization, the objective of this campaign was to increase the awareness of Portuguese citizens regarding the detection of radon gas in their residences. During the three-month collection period, the equipment's mobility was restricted [1].

2.2.1.3 Environmental and Geographic Problems in Home Isolation

The insulation of residences is an essential component of the housing construction process, as it is essential for enhancing the quality of life, energy efficiency, and thermal comfort of the occupants. There is also a pertinent connection with pollutant gases, as a house that is inadequately insulated can readily permit the introduction of any pollutant gas. At present, there are insulating materials that can be used to decrease the concentration of radon gas indoors. Nevertheless, this is not the case in the vast majority of residences, which explains

the problem's existence.

In 2022, Portugal ranked among the four worst countries, with the fourth highest rate (17.5%) of people unable to adequately heat their homes, compared to the European Union average of 9.3%, according to Eurostat [27], [28].

According to the *Portal de Construção Sustentável* (PCS), through a survey conducted in March 2022, about 89% of Portuguese respondents do not feel comfortable in their own homes [29]. Based on this survey, the source indicates that only 1 in 10 Portuguese consider the temperature of their home satisfactory. The lack of insulation and window sealing are other issues highlighted in this study, which covers all districts of Portugal. Additionally, regarding the survey results, about 16% of respondents believe their health problems are due to poor thermal insulation, mostly related to insulation, humidity, and poorly insulated windows and doors.

Another study conducted in the regions of Lisbon and Porto concluded that excessive cold and heat negatively affect the quality of sleep, study, and telework performance, among other things. According to the source, it was concluded that the problem of thermal discomfort is felt throughout the year, mainly due to the constructive inefficiencies of homes [30], [31]. However, this situation is common in all regions of Portugal, although the cold may be more intense in interior areas due to higher temperatures experienced there.

Due to poor insulation in homes and the low level of comfort felt inside them, people die every year due to exposure to low temperatures. In 2017, a study conducted by researchers from the University of Coimbra found that residents in municipalities with the worst environmental conditions have a 70% increased risk of excess winter mortality [32]. Cold-related mortality is more common among the elderly, who have weaker defenses, making them the most vulnerable population. This risk is higher in winter, as lower temperatures are associated with the onset of epidemics such as flu and pneumonia due to the virus's ability to survive in lower temperatures [33], [34].

2.2.1.4 Rural and Remote Regions

It is important to observe that in many rural and distant regions, communication problems still pose a serious issue in an increasingly digitized society. While some rural and distant places struggle to access information, big cities are already planning for the arrival of 6G technology, which promises fast speeds and minimal latency [35].

Progress in social, economic, and educational domains is predicated on effective communication. However, maybe as a result of low population density, the lack of infrastructure is noticeable in remote and rural locations. Many issues, including unstable internet connections, inconsistent cellular networks, and a lack of efficient communication services, keep the people living in these locations isolated from the modern world. Due to communication failures, which present a substantial issue that has to be handled, this problem is also obvious in the creation of IoT projects, meaning more sophisticated and in-depth study in these circumstances.

In the development of projects in the field of IoT, this problem also occurs, requiring a more complex and in-depth investigation of these scenarios due to communication failures, which pose a significant challenge. Some of the solutions involve implementing caching mechanisms for cases where the network is unavailable, with data sent asynchronously when the network connection is reestablished [36]. In general, we can conclude that the lack of communication in rural and remote areas affects various aspects of life in these disadvantaged communities.

These issues are prevalent in many isolated and rural regions of Portugal, where an aging population finds it difficult to obtain basic services like hospitals, health centers, and other infrastructure that is vital to human survival. The case study selected two municipalities in the interior of Portugal, the regions of Fundão and Pinhel, in the districts of Castelo Branco and Guarda, respectively, as the locations to be monitored. These geographic areas were chosen because they represent two distinct zones in terms of access to communication means and infrastructure, as well as access to hospitals and health centers. The slightly different morphological characteristics of the soil are also important for analyzing the collected data and allowing for comparisons.

Fundão is a city in the Beira Baixa area in central Portugal. The municipality of Fundão is located in the district of Castelo Branco and has a population of around 29,000, based on data obtained in January 2022 [37]. Its area is roughly 700 km², and it is a region where mainly schists combined with granites and sedimentary formations can be found, playing an important role in the geological history of the region. The region of Fundão presents an urban environment, with the presence of industrial activities and moderate road traffic. Access to infrastructure and services in the Fundão area is relatively easy, as it is a city with a certain level of development, in addition to its favorable location compared to Pinhel. Fundão also has a mountainous area known as the Regional Protected Landscape of Serra da Gardunha, covering an area of about 10,000 hectares [38]. The area is mainly composed of granites and some thin layers, like ”blankets” of earth, called metasediments. These elements constitute the Serra da Gardunha and contain radioactive gases, such as radon gas.

The city of Pinhel, also known as Falcon City, is located in the district of Guarda, in the central region, and also belongs to the Beira Interior Norte sub-region. It has an approximate area of 480 km², about 8,000 inhabitants, and 18 parishes [39]. Regarding the landscape construction of Pinhel, this city has a large area covered by granite outcrops, from which the gray granite of Pinhel can be extracted, being one of the causes of the high concentration of radon gas in the region. Pinhel is a relatively isolated city, particularly due to its geographic separation from other urban areas and the various limitations that restrict its access to essential infrastructure and services. This isolation contributes to its underdeveloped status. The city’s population is approximately one-third that of Fundão, highlighting its smaller scale. Consequently, Pinhel faces significant disadvantages in accessing the tools and resources necessary to establish a monitoring system.

2.2.2 IoT in Monitoring Ecosystems

Nowadays, home networks allow a wide range of items to be connected to the Internet, including smartphones, smartwatches, televisions, refrigerators, and lamps, among many others, possibly globally. Wireless Sensor Networks (WSN), the foundation of these systems, allows devices to connect to the network and send data for quick and effective analysis. Users stand to gain greatly from the creation of new projects that explore the possibilities of IoT technology since it allows them to monitor and operate their gadgets remotely, which enhances their quality of life.

With its ability to detect different environmental components in a distributed manner and to seamlessly integrate this heterogeneous data into visualization applications, IoT technologies can be effectively applied in the context of environmental monitoring [35], [40], [41]. The notion of the 'Smart House' is still relatively new and has been expanding throughout time, with plenty of possibilities for growth and improvement [42].

When these technologies are employed in urban settings, sometimes known as 'Smart Cities', the goal is to promote the notion of a smart city while also significantly improving urban living. This covers the flow of traffic for both cars and pedestrians, monitoring environmental factors, and having parking available to avoid traffic jams and high traffic. Data accessibility, brought about by pervasive urban IoT, improves openness and residents' understanding of the state of the city. According to J.P. Lynch and J.L. Kenneth, based on distributed databases, urban IoT can give users access to a variety of data gathered from IoT sensors on the ambient conditions surrounding the deployed systems [43].

Health-related initiatives primarily focus on providing users with real-time condition monitoring, which enables them to be alerted about vital indicators like body temperature, heart rate, and Electrocardiogram (ECG). Periodic reports on the health of users may be obtained with the help of IoT platforms, often through the integration of smartphones with smartwatches, which collect data via Personal Area Network (PAN) connections. Additionally, this technology can be employed in professional contexts, facilitating an in-depth analysis of patients by healthcare professionals using statistical and temporal data related to them.

Thus, the scope of IoT technologies is extensive, encompassing a variety of domains, including sports and smart agriculture, demonstrating high success rates. The ability to integrate data from numerous IoT devices with effective processing strategies contributes to the 'Big Data' phenomenon, underscoring the ongoing and growing importance of IoT in the current technological landscape.

2.2.2.1 IoT Architectures

The IoT architecture is typically divided into three major parts, namely the physical or perception layer, the network layer, and the application layer. There is a fourth layer that can also be included, the middleware layer, where cloud computing operations and data processing take place, as well as access to the database, as depicted in Figure 2.2.

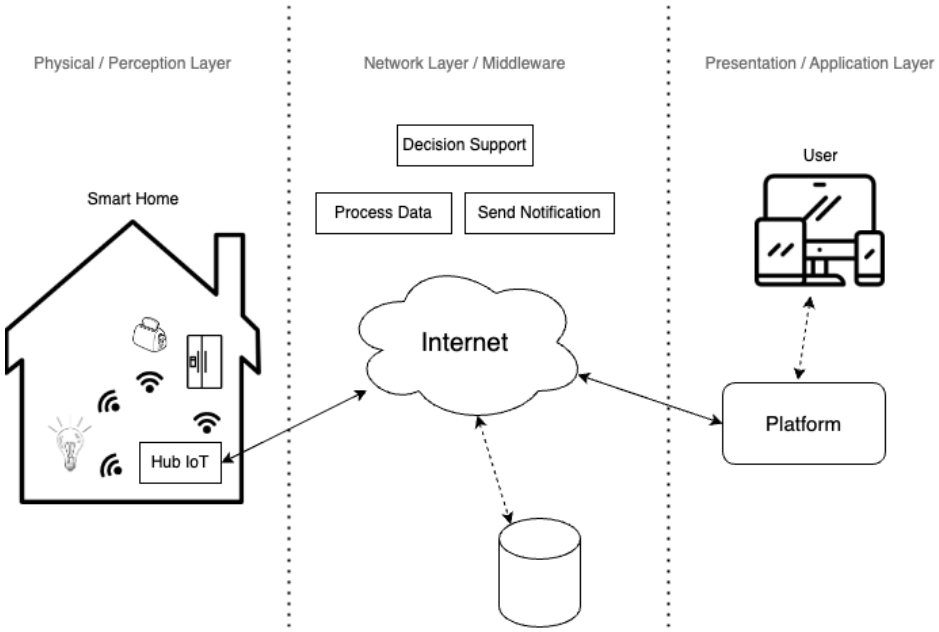

Figure 2.2: Representation of the different layers in IoT.

IoT sensors are devices that capture data from the environment and transform it into usable digital information, such as the reading of meteorological and environmental data. On the other hand, IoT actuators allow Internet-connected devices to remotely control functions such as opening doors or even activating an irrigation motor, for example. They are crucial for the automation and optimization of processes in various areas, including agriculture, healthcare, manufacturing, transportation, and smart homes [44]. The sensors and actuators are found in the perception layer for subsequent aggregation in the next layer. The network layer transmits the aggregated data from various devices to a generalized database.

It encompasses the stage in which data is aggregated and pre-processed, enabling the filtering, transformation, and formatting of this data, facilitating the transmission and analysis process.

On the other hand, the middleware layer plays a crucial role in the processing and interpretation of environmental data. This level of analysis is essential for extracting meaningful information from the data and identifying trends, patterns, and anomalies in environmental conditions. Additionally, the layer acts as a DSS, utilizing insights derived from data processing to create real-time alerts and notifications aimed at environmental management. This component is vital for the prompt response to critical environmental changes, enhancing the effectiveness of the actions taken.

The presentation function is performed by the application layer, which stands out by presenting processed and analyzed data in an accessible manner to users. This includes the creation of interactive dashboards, dynamic maps, and visual reports, providing users with a clear understanding of environmental parameters. This intuitive presentation capability not only facilitates data comprehension but also promotes informed decision-making.

2.2.3 Decision Support System

A DSS helps users make decisions with the analysis and provision of actionable insights from the analysis of large data sets. Within the environmental monitoring domain, a DSS has the capability to analyze information gathered from IoT sensors in order to predict future circumstances and initiate notifications upon exceeding specific benchmarks, like hazardous concentrations of harmful gases [45]. In circumstances where exposure to hazardous substances can have major repercussions, this enables prompt interventions that help reduce health risks [46].

Utilizing historical data to forecast future trends, Time Series Forecasting is a crucial technique in these kinds of systems (subsection 2.2.3.1). Because this method can detect trends over time and predict changes that could affect air quality or other environmental factors, it is especially helpful for environmental condition monitoring. Through continuous output refinement and learning from the data, machine learning models contribute significantly to improving the accuracy of these predictions (subsection 2.2.3.2). In order to enable more accurate forecasting and decision-making, algorithms for regression analysis are also frequently used to predict continuous variables, such as pollutant concentrations (subsection 2.2.3.3).

2.2.3.1 Time Series Forecasting

Forecasting time series data is essential in several sectors, as it forms the basis for making educated decisions in scientific, industrial, and commercial fields. Through the examination of sequential historical data collected at consistent intervals, we can predict forthcoming values, a crucial procedure in fields such as meteorology, telecommunications, and finance. Forecasting requirements may vary in scale, including both short-term and long-term timeframes, and ranging from projecting the next current data point to anticipating many future data points. Irrespective of the many uses, the fundamental difficulty remains consistent: forecasting future values using previous data, a process that might heavily rely on the characteristics of the time series [47].

Nevertheless, when we make predictions for the future, we increase the level of uncertainty. The growing complexity highlights the problem presented by time series forecasting: the need to effectively use established trends while also handling the uncertainty of future events [48].

Different patterns can be found in time series data over time. These patterns can be used to learn more about the time series or make predictions more accurate. A time series usually shows two main patterns: a trend and changes that happen with the seasons. It is called a trend pattern when the amount of a time series slowly goes up or down over a long period. People use the term ”trend cycle” to talk about changes in a time series that do not happen at set times. This is also known as a circular pattern. The yearly rhythm can be thought of as regular changes that are affected by factors that stay the same each season.

2.2.3.2 Machine Learning

Machine Learning (ML) is a data-driven approach that allows systems to iteratively learn from problem-specific data, automating the development of analytical models and the execution of associated tasks. It is a subfield of AI that enables computers to discover hidden insights and complex patterns without explicit programming [49].

The three primary categories of ML techniques are reinforcement learning, unsupervised learning, and supervised learning. All these categories deal with different kinds of issues and use different techniques to train models that can anticipate and make decisions based on information.

Supervised learning is distinguished by the utilization of labeled datasets, in which every input is linked to a predetermined result. Identifying patterns in the training data is essential for classification and regression jobs since it provides the foundation for precise predictions on fresh data. Tiwari (2022) emphasizes the value of supervised learning and lists a number of its numerous uses in time series forecasting, picture recognition, and medical diagnosis. Inherent difficulties are also covered by the author, including the requirement for substantial amounts of labeled data and the potential for overfitting, in which the model loses its capacity to generalize due to being overly tuned to the training set [50].

Conversely, unsupervised learning works with unlabeled data and looks for underlying patterns or structures in the data. This method works particularly well for problems like dimensionality reduction and clustering when the objective is to reduce or aggregate data representation without requiring preset results. In Zhang's (2022) exploration of unsupervised learning algorithms, he highlights the use of these algorithms in large data analysis, market segmentation, anomaly detection, and data compression. The author emphasizes the importance of techniques like k-means and Principal Component Analysis (PCA) for comprehending and examining huge, intricate datasets without the need for explicit labeling [51].

Lastly, using a system of rewards and penalties, reinforcement learning teaches agents to learn from the results of their activities and make successive judgments. This approach is very useful in situations when making decisions in real-time is essential, including in autonomous system control, gaming, and robotics. In their thorough examination of deep reinforcement learning, Mousavi, Schukat, and Howley (2018) show how this method helps agents acquire sophisticated behaviors via constant contact with their surroundings [52].

Precise forecasting of environmental gases provides useful insights that can assist persons in taking the required actions to avoid negative consequences. Data-driven methods, such as machine learning, can reveal linear connections between process parameters. This technique is not restricted to explaining the precise link between contaminants and environmental factors [53]. To make accurate predictions, relevant measurements are used to learn the underlying empirical reference model. With massive datasets readily available via sensors, data-driven methods seek to learn this link from the data using a variety of procedures, including statistical models and deep learning techniques [54], [55], [56].

A sub-branch of machine learning called 'Deep Learning' is concerned with modeling and solving complicated problems with the help of multi-layered artificial neural networks (ANNs). These networks are designed to simulate the human brain's ability to recognize patterns and learn from data. Each neuron in a layer is connected to neurons in the subsequent layer through weighted connections. During training, the network adjusts these weights based on the error of the output compared to the expected result, typically using a process called back-propagation combined with an optimization algorithm like gradient descent. This iterative process helps the network learn the optimal weights that minimize the error.

2.2.3.3 Algorithms for Regression Analysis

Deep Learning allows systems to learn different levels of abstraction for data representations, extracting high-level features straight from raw data without the need for manual feature engineering, unlike conventional ML algorithms. There are various neural network architectures for handling time series, with Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) being among the most commonly used [45].

The former are generally used in the field of computer vision and can also be applied to capture repetitive events in time series due to their ability to capture local patterns and spatial dependencies. On the other hand, the latter are ideal for time series forecasting because their internal loops allow information to persist, which is essential for detecting long-term temporal dependencies. Generally, RNNs maintain a hidden state that stores information from previous steps, although care must be taken with gradient problems. To minimize these problems, two variants (LSTM and GRU) have been developed, which have more effective mechanisms for capturing temporal dependencies. LSTM was initially proposed in 1997 for language models and is well known for its excellent ability to memorize long-term dependencies [57]. In 2014, the GRU model was an evolution of this model, introduced by Cho et al., designed to solve the gradient problem and with the advantage of being faster to compute than LSTM but with comparable performance [58]. More recently, another variant has emerged that has shown better results, consisting of the use of Bidirectional LSTM, capable of analyzing the data sequence in both directions, that is, from the past to the future and from the future to the past, processing two independent LSTMs simultaneously, which becomes more computationally complex but manages to produce more accurate and robust predictions [59].

These architectures have revolutionized time series analysis because they are excellent at collecting long-term dependencies and sequential information [60]. Although they present a high computational cost due to their complexity, they often require the use of GPUs and computers with high computational power.

In the analysis of environmental variables and pollutant gases read by IoT sensors, RNNs can be fundamental in this context for a better analysis of the underlying patterns in the data, aiding in the decision-making process, which minimizes possible disasters and health problems [61].

There is an alternative to deep learning architectures, which require more time and resources for training and tuning in the form of statistical models. These models are less prone to overfitting due to their simpler nature and the fewer parameters that need adjusting. Some recent investigations in the area of supervised techniques and high-dimensional models explore both linear and nonlinear methods for time series forecasting, utilizing components from ensemble models and hybrids of diverse methodologies [62].

Decision trees have also proved useful in time series analysis because of their capacity to capture nonlinear patterns and complicated relationships between variables. According to current research, decision tree modifications such as Random Forest and Gradient Boosting are very beneficial in time series forecasting when dealing with huge amounts of data and offering resilience against overfitting [63]. Furthermore, these models may be used with feature selection and dimensionality reduction strategies to increase prediction accuracy and interpretability. The versatility of decision trees in managing diverse types of data, as well as their ease of implementation, are key benefits for practical applications in disciplines such as finance, climatology, and public health.

However, statistical models are typically not as accurate in capturing variations, do not handle large datasets as effectively, and may not adapt as well to emerging changes. In 2004, Crespo Cuaresma et al. studied the forecasting abilities of a battery of univariate models, including AutoRegressive (AR) models [64]. The Auto-Regressive Integrated Moving Average (ARIMA) model was proposed in the 1970s to predict short-term freeway traffic data [65]. Over the years, other variant models of ARIMA have been proposed, which presented improvements. These models are based on the assumption of stationary variance and mean of time series as Kohonen-ARIMA (KARIMA), subset ARIMA and Seasonal ARIMA (SARIMA).

Hewamalage et al. [66] conducted a comprehensive empirical investigation on the current RNN models used for predicting. They concluded that RNNs can effectively predict seasonality when the series exhibits consistent seasonal patterns. However, if the series does not have homogeneous seasonal patterns, they advised doing a deseasonalization phase. Their demonstration showed that RNN models often achieve superior performance compared to the autoregressive integrated moving average (ARIMA) models. Nevertheless, RNN models need a larger amount of training data compared to ARIMA models because of their reduced reliance on assumptions about the time series structure and their limited interpretability.

Hybrid techniques are widely used because they improve results in both linear and nonlinear domains [48], [67], [68]. For example, ARIMA / Random Forest / Gradient Boosting models underperform when dealing with complex nonlinear issues, and ANNs do not necessarily produce the optimal outcomes while dealing with linear problems.

2.3 Related Works Review

The convergence of Information Technology (IT) with the advent of the IoT has revolutionized how we interact with the environment around us. In this dynamic landscape, the use of WSN, which consists of wireless sensor networks and low-power wide-area networks

(LPWAN), stands out as an essential facet in enabling advanced monitoring and automation solutions. This advancement is particularly evident in two crucial areas: the monitoring of environmental conditions and the implementation of smart homes.

WSN networks allow for the robust real-time collection and transmission of data from a myriad of sensors. In the context of environmental condition monitoring, WSNs play a fundamental role by enabling the deployment of sensors in remote, often hard-to-reach locations. The sensors can be strategically distributed, providing a holistic and precise view of the environment.

Low-Power Wide-Area Network (LPWAN) play a significant role, as they enable the transmission of small data packets over relatively large geographical areas. These networks are used in contexts where network connectivity is sparse or non-existent, and the use of cellular networks is neither viable nor available. LPWAN networks offer comprehensive coverage combined with low energy consumption, making them ideal for connecting devices in remote homes without Internet access. Any sensor can be interconnected within an IoT system through LPWANs, providing efficient and centralized management. The integration of these technologies not only covers underserved areas but also contributes to the advancement of the population residing in those locations.

2.3.1 Prototypes for Environmental Gases

These systems have an architecture based on IoT sensor networks, which then opt for various data transmission methods depending on the needs and availability of connection. Monitoring periods in IoT systems vary from days to months to obtain more accurate data.

The first prototypes that emerged in this field were for monitoring outdoor environments. Exposure Sense is the name of a prototype that allows the detection of gases such as CO, O3, and PM_{10}, but it requires the use of an Android smartphone to send the data (via cable) to the database [69]. In the end, it is possible to estimate users' exposure to polluted air and view these periods on a calendar available in the mobile application. The goal of this project is to advance research in this area and promote the use of the platform by more users, making the application more accurate and informative.

In Qatar, in 2013, another system, Hawa'ak ("Your Air" in Arabic), for monitoring outdoor gases like O3, CO, and H2S was developed, utilizing WSN and cellular networks (General Packet Radio Service (GPRS)) to send data to the Cloud. The mobile application and web platform allow for real-time data and statistics visualization, with the locations of different collection points represented on a map, and the testing period ran from March to June 2012, during which data was collected by distributed monitoring stations. This system uses an Short Message Service (SMS) system to send alerts to users during the monitoring period in case of detected anomalies. The mobile application and web platform allow for real-time data and statistics visualization, with the locations of different collection points represented on a map. To aid visualization, graphs showing the temporal variation of pollutant gases are displayed, with the ability to filter by hour, week, and month [70].

In March 2014, a pilot system was developed in Italy (Padova) [71] for environmental monitoring, using an IoT sensor network (nodes). Tests were conducted over a 7-day period, using 300 nodes to collect and send data to a centralized node. Besides temperature, humidity, and luminance, it also monitored C6H6 levels in the environment. The data packets were sent every 30 minutes using the IEEE 802.15.4 Low-Rate Wireless Personal Area Network (LR-WPAN) standard, with the devices powered by solar energy using batteries. For message transmission and management, the Constrained Application Protocol (CoAP) protocol is used at the application layer. The platform, which displays all data, is web-based and uses graphs and tables to visualize real-time and historical readings. In China, a large-scale monitoring system (UrbanAir20) was developed, covering several cities across the country, to monitor outdoor environmental variables (around 22 stations in 9 cities) [72]. The main focus of this prototype was to monitor ambient gas ($PM_{2.5}$), meteorological data, and vehicle and human traffic, with subsequent analysis of the collected data to draw some conclusions. The server or API, hosted in the Cloud, receives data from the sensors via communication networks, storing and processing the received information. For data visualization, a web platform and a mobile application are used, which is available for Windows Phone devices, though it is no longer in use. The publicly accessible web service features an interactive interface that displays air quality in different locations throughout the city, with additional functionalities such as viewing historical data and comparing air quality across various areas, aiding in informed decision-making. The system builds models that correlate air quality data from existing monitoring stations with other data sources, such as meteorology, traffic, and points of interest. In this way, it is possible to infer air quality in locations without monitoring stations.

One system that is, therefore, closer to the developed system is the Portuguese IoT system RnProbe. This prototype was developed in 2020 and can collect real-time data from 15 IoT devices located in rural and urban areas in the north of Portugal (Viana do Castelo and Barcelos) [73]. This system was connected with WSN, with the data being sent to the system via the MQTT protocol and HyperText Transfer Protocol (HTTP) protocol and the data was stored in PostgreSQL relational databases, which allows handling large volumes of data while maintaining high performance and scalability. In places where connection to the Internet is a problem, this prototype uses the LoRaWAN, utilizing LPWAN. The mobile application and web platform can receive data from the server and display it in real-time, and the most recent data is represented by graphs and regional maps with radon gas indicators. The platform allows user authentication and differentiates between *stakeholders*, namely the regular user, administrator, and guest user profiles. The architectural pattern used is Model-View-Controller (MVC), and the technologies used for the development of the platform are JavaServer Faces and Ionic technology, for *web* and mobile development, respectively. The first phase of testing took place over approximately two months. For storage, pre-processing, and data logic handling, the server (API) processes simultaneously, enabling a visualization platform to display the collected data in real-time. The internal web platform includes notifications to keep the user informed of high radon gas levels.

The obtained results allowed us to conclude that, in the collection areas, radon gas was inversely influenced by atmospheric pressure. This system demonstrated its importance as it can provide specialists with data that enables them to interpret the behavior of radon gas, with continuous study relating to this gas to combine its behaviors with external factors. It also sends notifications to them when the system detects high levels of radon gas.

2.3.2 Implementation of Decision Support in Environmental Monitoring Systems

More recently, in India, an external monitoring system (IoTMobair, 2019) was developed by Dhingra et al. [74] with the capability to collect data on pollutant gases such as CO, CO_2, NO_x, SO_x, $PM_{2.5}$, PM_{10}), using WSN to send data with HTTP protocol and database hosted in Cloud. The goal is to analyze real-time concentrations in urban areas of India, enabling people to navigate less polluted streets during their city journeys, and AI was utilized to indirectly measure the Air Quality Index (AQI) based on the environmental variables measured during the pilot phase. Artificial Intelligence is commonly used in IoT projects to predict future outcomes and detect patterns in time series data, even from data collected several years ago that was subsequently entered manually [75]. This was exemplified in the Airtify system [76], and in the system by Vanus et al. [77], where AI played a crucial role. In the former case, it alerted users based on estimated critical zones and indirectly predicted CO_2 levels. In 2021, Vanus et al. [78] presented an improved model that utilized Artificial Neural Network (ANN) in conjunction with the Bayesian regulation method to recognize patterns. In this case, the model was better able to analyze the CO_2 (indirectly predicted), temperature, and humidity. This model achieved better results, as an adaptive LMS filter (Sign, Sign-Sign, Sign-Regressor) was added to the model, which improved the accuracy in predicting CO_2 values.

Ma et al. developed an IoT system quite similar to this one, which collected data over six months in Northern China to infer that it was collecting real-time data. The monitored pollutant gases were NO_2, SO_2, O_3, CO, $PM_{1.0}$, $PM_{2.5}$, PM_{10}, and the location of the collection device. To complement the data, meteorological data such as temperature, humidity, and wind speed, which can influence the dispersion of gases, were aggregated. The data is sent from various devices to a Cloud server via cellular networks, which aggregates and stores the collected data for later visualization. From this data, a deep machine learning model was applied, with the chosen algorithm being ConvLSTM (Convolutional LSTM), to handle temporal data with the capability of being used in near real-time applications [79]. The developed system visualizes the data using accurate pollution maps even under irregular sampling conditions, incomplete data, and some climatic variations. The web platform allows any device with Internet access to access real-time and historical data, as well as display environmental pollution maps that are valuable for aiding decision-making and protecting public health.

Another system that also detects radon gas, in 2019, was a prototype developed by Tunyagi et al. [80] for the detection of radon gas (^{222}Rn) along with CO_2 and Volatile Organic Compound (VOC) detection. In addition to these capabilities, the prototype also collected data on ambient temperature, humidity, AQI, and relative pressure of an indoor environment. This

prototype was developed in a laboratory setting and under in-situ conditions to analyze air quality and enable the population to adopt mitigation measures against pollutant gases.

Another system quite similar to this one was developed in northwestern Pennsylvania. However, it does not include sensors for detecting radon gas, CO, and CO_2. Instead, it monitors indoor atmospheric gases such as Benzene, Toluene, Ethylbenzene, and Xylenes (BTEX) and $PM_{2.5}$. During the testing phase in 2016, 30 homes were equipped for gas detection, which took about seven months of testing. External factors such as temperature, humidity, and wind were also considered for a more comprehensive analysis [81].

2.3.3 Commercialized Systems

Some IoT systems now on the market have their interfaces, as do high-end items that require the same brand to function. There are several real-time environmental monitoring systems available, each with its own set of capabilities and restrictions. IQAir's AirVisual Pro Indoor gadget detects indoor gases and particles ($PM_{2.5}$ and CO_2), giving real-time and historical data access across several platforms [82]. UbiBot offers a paid system for environmental monitoring using various connectivity options (Wireless Fidelity (Wi-Fi), 2G, 3G, 4G, Zigbee, Ultra Wideband (UWB), Narrowband Internet of Things (NB-IoT)), with real-time data visualization and alerts via email and mobile/web platforms. The sensors detects the $_2$, $PM_{1.0}$, $PM_{2.5}$, PM_{10} and VOCs. The data is stored on the Cloud and has a free subscription with limited storage of 200MB, necessitating a paid plan for more extensive data storage [83].

There are very few products on the market capable of measuring indoor radon gas concentrations. In this context, due to the scarcity of sensors, two major companies stand out by providing an ecosystem capable of measuring radon gas (^{222}Rn) levels using IoT sensors (passive). Airthings is a renowned Norwegian company that provides sensors for monitoring radon gas, CO_2, $PM_{2.5}$, VOCs, temperature, air pressure, and humidity [84]. The radon sensor uses a passive diffusion chamber and requires a 7-day calibration period. The connection with the rest of the system occurs via Wi-Fi (2.4GHz) and Bluetooth Low Energy (BLE) (v4.2) connections, where the sensors send data to the server every 5 minutes, and this data is read on the Airthings mobile app. For this connection to occur, the data must first be sent to the mobile device and then sent to the cloud service if there is an Internet connection, and no data upload has occurred in the last hour. There is also the possibility of connecting to a hub that collects data (Airthings SmartLink) directly to the Cloud, though it incurs higher costs. In this case, the connection is made using an internal communication protocol, IEEE 802.11ah (Sub-1GHz), designed for long-range connections to replace the Bluetooth connection. With the hub functioning as an intermediary, the mobile device collects all data from the Cloud, including real-time data remotely. The Airthings platform allows users to visualize data, create radon mitigation graphs, and receive alerts when gas levels or environmental variables change. Another well-known brand in radon gas monitoring is RadonEye, manufactured by FTLab in Korea [85]. This brand features sensors that measure temperature and humidity in advanced models. Simpler devices offer BLE connectivity, while more expensive models include Wi-Fi and cloud service options. Initial findings are acquired in 10 minutes, with

reliable readings available in 60 minutes and updated counts at 30 per hour. Readings are stored in the device until connected to a mobile device for transfer, but when radon levels exceed recommended values, an alert is triggered on the device, accompanied by an integrated alarm.

2.3.4 Another Domains

In a slightly different approach, more focused on health and monitoring health levels, a system was developed by W. Hu et al. [86]. This system, using biometric sensors, aims to improve the thermal comfort of users based on machine learning mechanisms and sensors present in the user's residence. The data collection period lasted about two months, and the data included temperature, indoor humidity, lighting, and human presence, which serve as inputs to train the model to predict the thermal comfort of the occupants. The platform is part of the digital system, allowing the visualization of the collected and calculated data. ANNs are used, with a RNN architecture, as it learns based on complex patterns of sequential data. The Stochastic Gradient Descent optimization algorithm was also used to achieve the best performance.

Still in the health area, but related to sports, an IoT system was developed by Bhatia et al., in 2021, which monitors data collected by biosensors to study athletes' performance with the aid of artificial intelligence [87]. Machine learning techniques are used in this system for classification, pattern detection, and recommendation or decision-making to develop the best sports training strategies. Bayesian Modeling, LSTM, and Support Vector Machine (SVM) are used as they demonstrated the best results. For data visualization, the system uses a paid platform provided by Amazon Web Services (AWS) [88].

Also in the field of smart agriculture, in 2021, O. Ly et al. [89] developed a system capable of keeping farmers informed and connected with agronomists through a platform that facilitates the visualization of data obtained from sensors distributed across farms. The components measured by the sensors include soil and ambient humidity, soil pH, wind speed and direction, and the presence of harmful insects. It is an ecological system as it informs the user about the amount of water needed based on requirements. On the front-end, users can check real-time and historical data on a web platform using sockets. This platform was developed in NodeJs and EasyRTC/WebRTC, with authentication capabilities, allowing the identification of different users for an optimized platform tailored to the stakeholder. It contains real-time data obtained from soil moisture sensors, soil pH, wind speed and direction, and detection of harmful insects.

2.4 Conclusion

Table 2.1 shows a short overview of the studies referenced in Subsection 2.3. Despite the existence of various IoT prototypes for monitoring environmental variables and gases, none are entirely similar to the proposed solution, which specifically targets remote and rural areas, incorporating various communication strategies and caching mechanisms to meet these

unique needs. The system allows data visualization on multiple platforms, including mobile devices, for users and uses ML to predict future values, aiding in decision-making processes to minimize potential health risks. According to some authors, intelligent systems based on environmental monitoring, when including artificial intelligence and integrating artificial neural networks (ANNs), can address issues related to the physical sensors used, particularly temperature and humidity sensors [90].

Table 2.1: Prototypes Developed for Monitoring Pollutant Gases and RuraLTHINGS System. (*) - Meteorological Data, Traffic; Temp. - Temperature; Hum. - Humidity; Preci. - Precipitation; In. - Indoor; Out. - Outdoor; [-] - Not Mentioned

Ref. / Name	Type of Monitoring	Additional Parameters	Monitored Gases	Test Duration / Number of Nodes	Technologies	Data Visualization Platform
2013, [69] ExposureSense	Outdoor	Accelerometer Microphone	CO,O_3 PM_{10}	-	-	Smartphone
2013, [70] Hawa'ak	Outdoor	GPS	O_3, CO, H_2S	3 months / -	WSN, Alerts	Android Web
2014, [71] Padova	Outdoor	Temp. Hum., Luminance	C6H6	7 days / 300 nodes	WSN	Web
2014, [72] UrbanAir20	Outdoor	(*)	$PM_{2.5}$	- / 22 nodes, 9 cities	WSN, AI	Windows Phone Web
2016, [81] Pennsylvania	Indoor, Outdoor	Temp., Hum., Wind	BTEX e $PM_{2.5}$	7 months / 30 houses	WSN	Web
2018, [76] Airtify	Indoor	-	$PM_{2.5}$	-	AI, WSN, Alerts	Android Web
2018, [77] Prototype A	Indoor	Temp., Hum., Luminance	CO_2	2 months / -	AI	Android Web
2019, [80] Prototype B	Indoor	Temp., Hum., pressure	^{222}Rn, CO_2 COV	-	-	Android, iOS Web
2019, [74] IoTMobair	Outdoor	IAQ	CO, CO_2, CH_4	-	AI	Web
2020, [73] RnProbe	Indoor	Temp., Hum., pressure	^{222}Rn, CO_2	2 months/ 15 houses	-	Web
2024 ***RuraLTHINGS*** *System*	Indoor	Temp., Hum., rain	^{222}Rn, CO_2 CO	3 months / 20 houses 2 cities	WSN, AI, Alerts	Android, iOS, Web

A chronological review of Table 2.1 reveals that initial studies primarily focused on outdoor monitoring.

In subsequent years, the emphasis expanded to encompass interior spaces. A significant reason was the proliferation of remote work, leading individuals to spend more time in their residences, along with heightened studies on indoor air pollution. The gases detected differ considerably across prototypes, with carbon monoxide (CO) and carbon dioxide (CO_2) being the most prevalent. Environmental parameters are gathered for both real-time analysis and comprehensive examination of the monitored gases. Research is performed on the patterns discernible post-data collection, with the objective of concluding the influence of supplementary variables on the measured pollutant gases.

Certain investigations facilitate the study of gas concentrations and the evaluation of their effects and consequences. Nonetheless, this remains an emergent subject, necessitating greater investigation and the use of machine learning algorithms for predictive purposes.

Machine Learning is utilized in certain prototypes, facilitating enhanced insights and predictions derived from data acquired by IoT sensors. In the instance of the UrbanAir20 prototype [72], artificial intelligence is utilized just to deduce air quality in regions without monitoring stations. Airtify uses artificial intelligence techniques (classification) to evaluate data from sensors, facilitating the identification of crucial zones and then notifying users of such occurrences. Conversely, prototype A [77] and IoTMobair [74] assess the value of carbon dioxide (indirectly) and the air quality index, respectively. Regression techniques and artificial neural networks are employed to identify trends from historical data, with the objective of deriving insights from the pre-trained model.

The RuraLTHINGS system stands out from the others by allowing the monitoring of a lesser-known gas (radon gas) inside residential structures, similar to the prototypes [80] and [73]. Additionally, it can present graphs and historical data belonging to the gadgets that collect information inside the homes. The visualization platform enables data analysis across all operating systems (Android, iOS, and Web), a capability currently available only in prototype (B) [80], enhancing data interpretation. While alerts are integrated into several prototypes, none of them offer compatibility with iOS for this feature. The aim of using machine learning is to analyze sensory data over time, identify patterns and correlations, and predict future values. The model takes into account many aspects, including the seasonality (hour, month, and year) of the collected data and the indoor atmospheric conditions of the homes. The goal is for the platform to predict upcoming trends and issue notifications if the projected data indicates an upward trajectory. The details are comprehensively elucidated in the following chapters of this thesis.

Table 2.2 presents several commercially available systems that facilitate gas detection within a dedicated ecosystem.

Table 2.2: Commercialized Systems for Monitoring Environmental Factors/Pollutant Gases.

Reference	Environmental Parameters	Pollutant Gases	Technologies Used	Notifications / Platform	Extra Cost
AirVisual Pro Indoor, [82]	Temperature, humidity and AQI	$PM_{2.5}$ and CO_2	Wi-Fi	x (Android / iOS / Web)	✓
UbiBot, [83]	Temperature, humidity, luminance and atmospheric pressure	CO_2, PM1, $PM_{2.5}$, PM_{10}, VOCs	Wi-Fi / Cellular Network Zigbee / UWB / NB-IoT	✓ (Web App)	✓
Airthings, [84]	Temperature, humidity and atmospheric pressure	CO_2, $PM_{2.5}$, VOCs and ^{222}Rn	Wi-Fi and BLE (Airthings) Sub-1GHz (Air. SmartLink)	✓ (Android / iOS)	✓
RadonEye, [85]	Temperature and humidity	^{222}Rn	Wi-Fi and BLE	✓ (Android / iOS)	✓
RuraLTHINGS System	Temperature, humidity and atmospheric pressure	CO, CO_2 and ^{222}Rn	Wi-Fi / Ethernet / LoRaWAN / Cellular Network	✓ (Android / iOS / Web)	x

Airthings offers various models, but the cheaper ones are very limited in the functionalities they provide. These systems may require service subscriptions, paid software updates, and potential repair or part replacement charges, including capabilities such as Wi-Fi connectivity, cloud services, and additional temperature and humidity sensors. These combined constraints may make such systems financially unsustainable for some users, particularly those looking for a more economical and low-cost solution for radon monitoring in their homes or workplaces.

Commercialized monitoring solutions have numerous drawbacks. Airthings relies on a mobile device connection for initialization and data transmission to the Cloud and is limited to Wi-Fi 2.4GHz and BLE v4.2 connections, which might reduce range and cause interference. RadonEye provides simpler variants with Wi-Fi and BLE connectivity, although data transfer requires close contact with the mobile device. Furthermore, data is retained on the device until transferred, limiting the frequency of real-time updates, and alarms are generated locally, which may cause communication delays with the Cloud. In the RuraLTHINGS System, as shown in Table 2.2, the proposed system does not incur any extra cost. The citizens using this platform do not need to purchase additional storage or more devices to monitor environmental values.

The systems allowing for the detection of radon gas concentrations are very scarce due to the lack of knowledge about chemical components and their severity, especially when it is in contact with air and, consequently, with humans. Another reason is active sensors for detecting radon gas are difficult to access or construct, which is also a negative factor for the development of systems that need to monitor this gas. In addition to the high initial cost, commercially available IoT systems can represent a significant expense not only at the time of purchase but also in terms of ongoing maintenance and operation. Consequently, prototypes and commercialized systems that use sensors to detect radon gas are very limited, as demonstrated by tables 2.1 and 2.2. More recently, there has been increased research on this gas, which is quite dangerous at high concentrations.

In summary, the proposed system aims to monitor various pollutant gases, including radon gas, and environmental variables to identify possible abnormal values. The RuraLTHINGS system will use machine learning to identify patterns, considering various variables, thus enabling more informed decision-making.

Chapter 3

RuraLTHINGS: Implementation of the Digital Platform

3.1 Introduction

This chapter will describe the proposed IoT platform (RuraLTHINGS) that directs the data visualization and prediction process conducted in this research. The goal is to develop a platform for visualizing historical and real-time data collected from various homes and an intelligent system capable of accurately predicting pollutant gas concentrations and environmental variables. The data is collected by sensors and transmitted via suitable communication infrastructures, then stored and processed using machine learning algorithms and statistical analysis techniques to improve the prediction process, which consequently aids in decision-making. Within the objectives of the RuraLTHINGS project, this corresponds to a fundamental task that guides and promotes the detection of critical situations inside homes, in order to prevent and minimize health risks.

The following sections explore the software engineering of the platform, digital platform modeling, technologies used, and conclusions drawn based on the proposed platform.

3.2 Software Engineering

Subsequent to the Software Engineering subsection, several key components of the RuraLTHINGS platform development are examined. Firstly, Subsection 3.2.1, Stakeholders, identifies the primary stakeholders of the project and explains the use cases that detail how these stakeholders engage with the system to ensure their needs are addressed. Afterward, Subsection 3.2.2, User Requirements, outlines the user-centered requirements to ensure the system addresses the needs of the stakeholders effectively. Subsection 3.2.3, System Requirements, outlines the functional and non-functional requirements necessary for the effective and efficient implementation of the platform. Next, Subsection 3.2.4, Use Cases, elaborates on the specific scenarios that define interactions between the users and the system. Following this, Subsection 3.2.5, Activity Diagrams, depicts the dynamic flow of activities within the system. In addition, Subsection 3.2.6, Class Diagram, provides a detailed representation of the static structure of the system, illustrating the system classes and their relationships. Finally, Subsection 3.2.7, System Architecture, describes the system architecture, including the general structure, components, and their interactions, providing a coherent and scalable integration that supports the platform objectives of visualization, prediction, and alarm production.

3.2.1 Stakeholders

The stakeholders in this project represent fictional characters that embody the different types of end users of the RuraLTHINGS system. These personas allow us to better understand the needs, behaviors, and characteristics of real users, facilitating informed decision-making during the design and implementation of the system. The use cases will be based on these stakeholders, ensuring that the system addresses their specific requirements and interactions. The stakeholders involved are:

- **Administrator**: Responsible for managing the devices, assigning them to houses, and configuring specific rules for data collection;
- **Caregiver / Resident**: A user who interacts with the system to monitor sensor data and receive alerts;
- **Technician**: Handles the installation and maintenance of the devices, ensuring correct operation and integration with the platform.

3.2.2 User Requirements

User requirements describe what users need the system to do in order to meet their goals and expectations. These requirements form the foundation for the design and implementation of the RuraLTHINGS system, ensuring that it is built with a user-centered focus.

In this section, the following key aspects of user requirements will be addressed:

- **Functional Requirements**: These define the specific features and capabilities that the system must provide to meet the needs of users and effectively support their tasks.
- **Non-Functional Requirements**: These refer to the qualities and characteristics of the system, such as performance, usability, and security, which influence the user experience and the system's reliability.

By thoroughly addressing these requirements, the system will ensure that it not only delivers the necessary functionality but also meets the quality standards expected by the users.

3.2.2.1 Functional Requirements

The functional requirements define the specific actions and behaviors that the RuraLTHINGS system must support to meet the needs of its users. These requirements ensure that the system provides the necessary functionality for device management, data monitoring, and user notifications. The following requirements outline the core features that the system must implement to fulfill its intended purpose:

- The system must provide the option for login and registration (for new users) to the final user;
- The system must allow the association of one or more RuraLTHINGS devices to a user;

- The system must enable the viewing of one or more RuraLTHINGS devices associated with a user;
- The system must notify users when values exceed pre-established limits for collected environmental conditions;
- The system must notify the user if any sensor is not working;
- The system must allow the viewing of unread (warning) occurrences and already read occurrences;
- The system must allow real-time data visualization obtained from the communication system that sends data to the mobile platform/application;
- The system must allow filtering of data sent by RuraLTHINGS devices for more detailed analysis, presented in charts and widgets;
- The system must allow the Administrator to view the number of active devices and registered users in the system;
- The system must allow the Administrator to view all data;
- The system must assist the user in predicting values to minimize the consequences of elevated levels of air pollutants;
- The system must send notifications to the final users to alert them to abnormal values or any failures that may be occurring in the system.

3.2.2.2 Non-Functional Requirements

Non-functional requirements describe the qualities and performance standards that the system must meet. These requirements focus on system reliability, responsiveness, user interface design, and security, ensuring that the platform delivers a seamless and secure user experience. The following are key non-functional requirements that guide the overall system architecture and usability:

- The system must initialize the entire platform in less than three seconds;
- The system must load user options immediately;
- The system must present a clean interface with a pleasant color palette;
- The application should be intuitive and user-friendly;
- The system must have the ability to visualize and process large volumes of data;
- The system must not take more than four seconds to process the data to be displayed and the information to be stored;
- Application responsiveness must be an important factor regardless of the platform used;

- The application must have processes that ensure the highest possible security and, consequently, greater reliability of data to ensure data protection;
- The application must be stable to ensure that the user does not encounter bugs or failures that jeopardize the use of its functionalities.

3.2.3 System Requirements

In this subsection, the Software Engineering of the platform will be described, specifically its functional and non-functional system requirements. The requirements are divided by stakeholders since they differ for an End User and an Administrator.

This subsection describes the Software Engineering aspects of the platform, focusing on its functional and non-functional system requirements. These requirements specify the necessary features and performance standards that the system must meet. Since the needs and responsibilities vary between different types of stakeholders, such as End Users and Administrators, the requirements are divided accordingly.

3.2.3.1 Functional Requirements

The functional system requirements define the specific actions that the platform must perform for both End Users and Administrators. These requirements ensure that each type of user can interact with the system according to their role, managing devices, viewing data, and receiving notifications, as necessary.

RF001 - Login

The platform must authenticate users to ensure they can access only their devices and the data collected by them. This process involves verifying the credentials provided by the user and granting access to the platform.

The login process includes the following steps:

1. The system must display a form for user login and registration;
2. The system must allow entering the credentials of the user in the email and password fields;
3. If the email or password of the user does not match those stored in the database, an error message should be displayed;
4. After verifying that the credentials are correct, the system should redirect the user to the home page with the respective user data.

This process is essential for ensuring secure access, and further details can be found in Table 3.1.

RF002 - User Registration

The platform must allow users to register by providing their personal information and the device(s) they have in their home to access the data sent by these device(s). This process

RF001	Platform Authentication
Function	Authenticate the user in the system.
Inputs	Email and Password.
Outputs	Home page.
Required Information	Personal data of the User; User already registered in the database.
Action	Once the user enters the platform, they are directed to the page where they can log in if already registered in the system. Data verification is performed based on the information stored in the Database.
Preconditions	User registered in the database; Access to the home page.
Side Effects	No significant effects.
Layout	N/A

Table 3.1: Functional Requirement Specification Table No.001.

ensures that the user can interact with the platform and access relevant IoT data. The system facilitates this by guiding users through a registration form.

The registration process includes the following steps:

1. The system must display a form with the option for login and registration, the latter being for the case where the user is not yet registered in the database;
2. The system must allow entering the credentials of the user, such as email, password, name, and contact phone number;
3. If the email is already assigned, an error message should be displayed;
4. The system must ask for the password twice and allow its visibility to prevent possible mistakes;
5. After all fields are filled in correctly, and the verification code sent via email is entered, the system must register the user in the database and automatically assign a unique ID for identification within the system;
6. After registering the user, the system should redirect the user to the home page.

This flow is further detailed in Table 3.2, which outlines the system's functionality, inputs, and outputs during the registration process.

RF002	User System Registration
Function	Perform user registration in the system.
Inputs	Email, Password and phone contact.
Outputs	Addiction of a new user to the database followed by redirection to the home page.
Required Information	Personal data of the user.
Action	Once the user enters the platform, they are directed to the page where they can register in the system. Data verification is performed based on the information stored in the Database, to check if the email is not associated with another account.
Preconditions	Insertion of valid data; Access to the home page.
Side Effects	No relevant effects.
Layout	N/A

Table 3.2: Functional Requirement Specification Table No.002

RF003 - Password Recovery

The platform must provide a mechanism for users to recover access to the RuraLTHINGS platform if they forget their password. This feature is conveniently located near the login and registration forms and allows users to reset their password securely through an email verification process.

As described in Table 3.3, the recovery process follows a series of steps that ensure the user's identity is verified before allowing a password reset:

1. The system must display an option for password recovery near the login and registration forms;
2. The system must allow entering the email of the user in the email field;
3. If the email is incorrect, an error message should be displayed;
4. The system must automatically email the provided address with instructions for password recovery;
5. After entering the new password, the system must replace the old key with the hash of the new one in the Database;
6. The system must allow the user to perform the login process, as described in 3.2.3.1, to access the home page.

RF003	Reset Password
Function	Reset the password of the user.
Inputs	Email and updated password.
Outputs	Update of user data in the database.
Required Information	personal data of the User.
Action	Once the user enters their email on the password recovery form page, they receive an email at the email address provided. Data verification is performed based on the information stored in the Database to check if the email is associated with any account. If affirmative, the user receives an email directing them to a page where they can update their password in the system. Finally, a normal login can be performed. 3.2.3.1.
Preconditions	Insertion of a new password; Access to the home page.
Side Effects	No significant effects.
Layout	N/A

Table 3.3: Functional Requirement Specification Table No.003.

RF004 - Access to the Home Page and Various Pages

The system must allow access to various pages on both web and mobile platforms, providing key functionalities and displaying relevant information. This access is restricted to authenticated users, ensuring that only devices associated with their accounts are visible. The home page serves as the central hub for navigating the platform, offering a clear overview of the most recent data from IoT devices and facilitating access to other important sections. Below are the key features and functionalities available to users, as detailed in Table 3.4:

The platform allows access to various pages available to the user. This access is restricted to the devices of the user as a security measure.

1. A page with various widgets displaying the most recent data from the RuraLTHINGS devices of the user is presented.
2. This page should have a side menu that can navigate to other relevant pages (dashboard, charts and statistics, devices, and settings).
3. The user should be able to click on each presented chart to view it in higher resolution and analyze it in detail.

4. The user should be able to view alerts related to their devices, especially regarding value deviations.
5. The page should allow the addition of a new IoT device that the user already possesses.

RF004	Navigation through the Home Page and Platform Menu
Function	Allows the user to navigate and access the information available in the application.
Inputs	Authenticated user ID and associated data (HTTPS GET requests to the API).
Outputs	Access to the page corresponding to the selected option.
Required Information	None.
Action	The system allows access to other pages with information and analyses obtained from the data collected regarding the RuraLTHINGS device of the user.
Preconditions	Login performed; Access to the user-customized platform with the session initiated.
Side Effects	No significant effects.
Layout	N/A

Table 3.4: Functional Requirement Specification Table N°004.

RF005 - View the Collected Data from the Sensors

The system must allow users to view the data collected by their IoT sensors in real-time via the platform. The user can only view data related to the IoT device present in their home. Initially, the system presents data collected over the last 24 hours, but if no recent data is available, it defaults to the latest records. If no data is found for that period, the system automatically checks the last month, continuing this process until data is found.

The user interface provides various widgets that allow users to browse temporal records of sensor data. As described in Table 3.5, the user can filter the data by daily, weekly, or monthly intervals. A waiting page will be displayed while the API processes the request. Once the data is fetched, the system presents the records for values like radon gas, temperature, humidity, carbon dioxide, and carbon monoxide, each with a timestamp. For further analysis, the system can use an external API to append meteorological information to the sensor data records.

RF006	View the Collected Data from the Sensors
Function	Display various temporal records of values obtained by the sensors.
Inputs	HyperText Transfer Protocol Secure (HTTPS) GET requests to the API, based on the chosen time period in each widget.
Outputs	Records and their specifications.
Action	The user can browse all available records in various widgets. A waiting page will be displayed while the request to the API is processed. Then a list of records will be presented, each with a value for radon gas, temperature, humidity, carbon dioxide, and carbon monoxide, along with the date/time it was obtained. The user can filter the results based on their preferences (daily, weekly, and monthly). For collecting weather information at the time of data submission, an external API can be used to send a JavaScript Object Notation (JSON) file with meteorological information to add to the previously collected record.
Preconditions	Well-applied hardware system, stable connection between various components (sensors) connected to Arduino, stable connection from Arduino to the Internet, and stable connection from the API to the Cloud database.
Side Effects	N/A
Layout	N/A

Table 3.5: Functional Requirement Specification Table Nº005.

RF006 - Check Alerts Sent by the API

The system must notify users when sensor values exceed the limits recommended by the WHO, particularly for radon gas levels. These alerts are specific to the user's IoT devices, ensuring that they can monitor relevant environmental conditions in their homes.

Table 3.6 details the process of checking high concentration alerts. The system sends these alerts via an HTTPS GET request to the Application Programming Interface (API), and users can view both read and unread alerts. When a request is made, a waiting page will be displayed while the system retrieves the data. The user will then see a list of alerts, each specifying the abnormal sensor values along with the date and time the data was collected.

RF007	Check High Concentration Alerts
Function	Display various alerts sent by the API.
Inputs	HTTPS GET request to the API, based on the authenticated user.
Outputs	Viewed alerts and unread alerts.
Action	The user can browse all alerts related to their IoT devices to monitor their home(s). A waiting page will be displayed while the request to the API is processed. Then a list of alerts will be presented, each specifying the abnormal value and the date/time it was obtained.
Preconditions	Stable connection between the device and the API.
Side Effects	When the alert is read, a POST to the API is performed to mark the alert as read.
Layout	N/A

Table 3.6: Functional Requirement Specification Table Nº006.

RF007 - Obtain Statistics via API

The system must provide users with statistical data derived from the information collected by IoT sensors. The platform should process and present this data in the form of graphs, offering insights into sensor activity and alerts for high concentrations of pollutants.

As described in Table 3.7, users will be able to view these statistics through a series of graphs generated based on the data collected by their IoT devices. This data is fetched using an HTTPS GET request to the API, and the results are displayed after a brief loading period. During this time, a loading wheel will indicate that the request is being processed, ensuring the user is informed while the graphs are prepared.

RF008	Navigate through High Concentration Alerts
Function	Receive statistical data sent by the API.
Inputs	HTTPS GET request to the API, based on the authenticated user.
Outputs	Viewed alerts and unread alerts.
Action	The user should be able to view statistical data in the form of graphs obtained from environmental data collected by the sensors. A cyclic loading wheel will be displayed while the request to the API is processed. Then graphs related to the devices the user owns will be presented.
Preconditions	Stable connection between the device and the API.
Side Effects	N/A
Layout	N/A

Table 3.7: Functional Requirement Specification Table Nº007.

RF008 - Manage Devices

The system must allow users to manage the devices associated with their accounts. This includes viewing the devices, their locations, and the option to unlink a device. When a user wants to change the location of a device, it must go through an internal process to ensure data consistency, as any change can affect how data is handled across the platform.

Users can see a list of their associated IoT devices and manage their settings accordingly. As outlined in Table 3.8, the user is able to view the name assigned to their device and its current location and has the option to unlink the device from their account. However, if the user decides to relocate the device, they must follow a dedicated process that helps maintain the integrity of data across the platform.

RF009	Manage RuraLTHINGS Devices
Function	Receive information about devices associated with the user.
Inputs	HTTPS GET request to the API, based on the authenticated user.
Outputs	IoT devices associated with the authenticated user.
Action	The user should be able to view the name assigned to their IoT device, its location, and unlink from that device. If they want to change the location, they should go through an internal process to make the change.
Preconditions	Stable connection between the device and the API.
Side Effects	If the RuraLTHINGS device is unlinked or its location is changed, a POST to the database is required.
Layout	N/A

Table 3.8: Functional Requirement Specification Table Nº008.

RF009 - Add New Devices

The system must allow the association of RuraLTHINGS devices with one or more users. A user who wants to add a new device to their account on the platform should follow these steps:

1. By pressing the ”Add New Device” button, the user will see a pop-up with the possibility of adding the ID of the IoT device they own;

2. Next, a new text box will appear asking for the insertion of a confirmation code, a code that is physically provided to the user when they place the device in their home;
3. Once the code is confirmed, the user can view the data of that device on the platform. If it does not match, they should inform the system administrator.

The details of this process are further explained in Table 3.9, which outlines the key aspects of this functional requirement.

RF010	Add RuraLTHINGS Devices
Function	Associate new devices with the user.
Inputs	HTTPS POST request to the API, based on the authenticated user, device ID, and confirmation code.
Outputs	Success/failure in associating the device with the user.
Action	Once the form is filled out, the user should be able to view the data collected by the device since its placement.
Preconditions	Stable connection between the device and the API; verification code.
Side Effects	If the verification code is incorrect, the user will not be able to access the data of the RuraLTHINGS device.
Layout	N/A

Table 3.9: Functional Requirement Specification Table Nº009.

RF010 - Manage Personal Data

The system must allow the user to view their data (Name, Email, Contact, Default Language, Profile Picture) and also modify these as well as their Password, Name, Contact, Default Language, and Profile Picture. The RuraLTHINGS platform should only allow the modification of personal data related to the authenticated user. Details of the functionality can be found in Table 3.10.

1. When accessing the ”Personal Data” page, the user will see a widget with all their data;
2. They have the option to add or modify data already in the system;
3. Next, a text box will prompt confirmation for the new data entered;
4. Once the modification is confirmed, the user should be able to immediately see these changes on any device with their authenticated account.

RF011	Manage user data
Function	Allows viewing and modifying personal data of the user on the platform.
Inputs	HTTPS GET and POST to the API, with the sending ID of the authenticated user and data to be modified, if applicable.
Outputs	Success/failure in viewing and/or modifying the personal data of the user.
Action	Once the form is filled out, the user should be able to view their data and confirm changes if desired.
Preconditions	Stable connection between the device and the API; Data to be modified in the proper format.
Side Effects	Modification of data in the database if the modification is successful.
Layout	N/A

Table 3.10: Functional Requirement Specification Table Nº010.

RF011 - Notification Reception

The system must allow users to receive push notifications whenever a new message is received, such as an alert for changed radon gas values. The mobile application should enable real-time notification reception, while on the web platform, notifications should appear whenever the browser is open. The triggering of notifications is handled by the API, which

utilizes the token for sending notifications through the Firebase Cloud Messaging system (see Table 3.11).

RF012	Push Notifications
Function	Allows the user to be alerted to changes in any of the values collected by their IoT devices in real-time.
Inputs	Device token, previously stored for sending notifications by the API.
Outputs	Received notification.
Action	The API sends the notification as soon as the data to be inserted into the database has abnormal values, triggering a notification to the device(s) associated with the user.
Preconditions	Stable connection between the device and the API; Functioning of the Firebase Cloud Messaging (FCM) service; Abnormal data collected by sensors; User-activated notifications on the device.
Side Effects	N/A
Layout	N/A

Table 3.11: Functional Requirement Specification Table Nº011.

3.2.3.2 Non-Functional Requirements

1. The Database and the API must use hardware and software capable of providing high rates of data write and read, to enable smooth and fast use of platform features;
2. The system must be scalable so that the addition of new IoT devices does not interfere with the operation of the system;
3. The device running the platform/app must have a good Internet connection to be able to make all requests to the API;
4. Communication between the API and devices for data transmission must use tokens to ensure security in the process;
5. Obtaining and validating tokens must be done through secure connections, using HTTPS;
6. User-entered passwords must have a minimum length of 8 characters, include at least one special character, and combine uppercase and lowercase letters;
7. The platform must be responsive, and functioning correctly on any device;
8. The mobile application must present a simple and easily interpretable user interface that allows navigation through all features quickly and intuitively;
9. The system must ensure that notifications are delivered to users within 30 seconds whenever the device is connected to the Internet and has notifications enabled on the device;
10. The mobile application must remain stable to ensure smooth operation;
11. The device must have an Internet connection for the user to use the application;
12. In the case of the Android system, the operating system version must be greater than or equal to version 6.0 for the application to run. In the Apple ecosystem, the minimum version corresponds to iOS 12.0.

3.2.4 Use Cases

The use cases will be shown in the next subsections, firstly we expose the connection between the devices/sensors IoT and the system/API. In the second part, the use cases related to the platform will be addressed.

3.2.4.1 Collect Data from RuraLTHINGS Devices

The Figure 3.1 represents the aggregation and analysis of all data collected by the RuraLTHINGS IoT device, for subsequent sending to the database. In these representations, physical devices function as main actors.

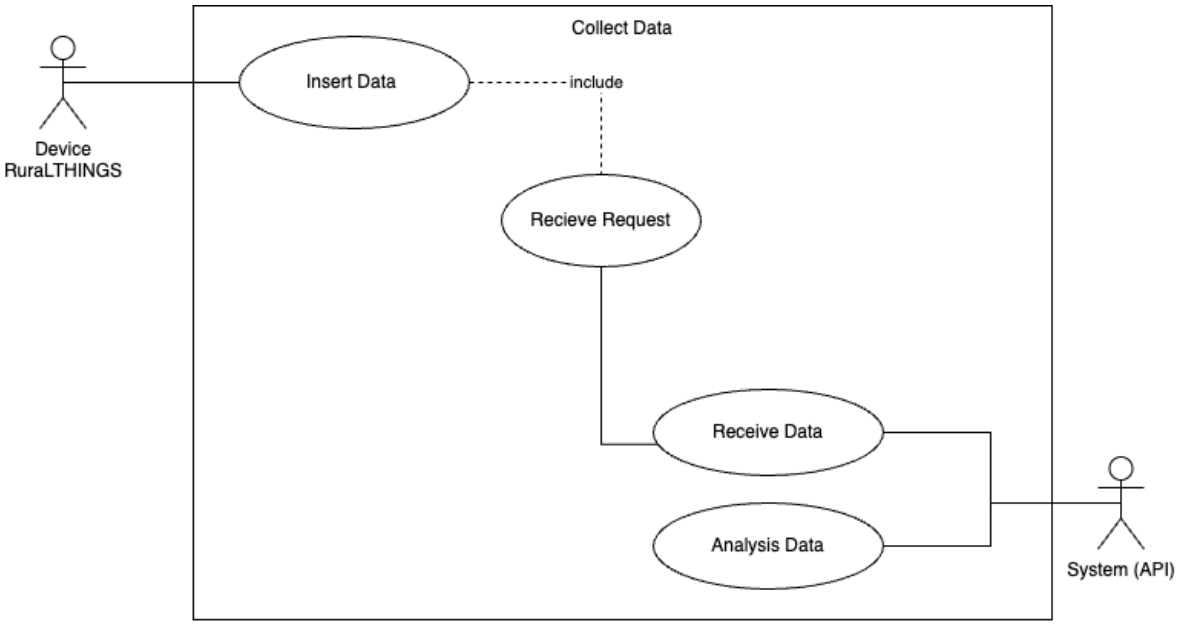

Figure 3.1: Diagram that describes the process of collecting data from all sensors to the server (API).

3.2.4.2 Get Records from System

The mobile application will be used by the different stakeholders responsible for the devices found in different homes, as shown in Figure 3.2 As the application does not have direct contact with the database, this entire process is also carried out via the API, which allows data to be received from different devices, as well as viewing these devices on the platform.

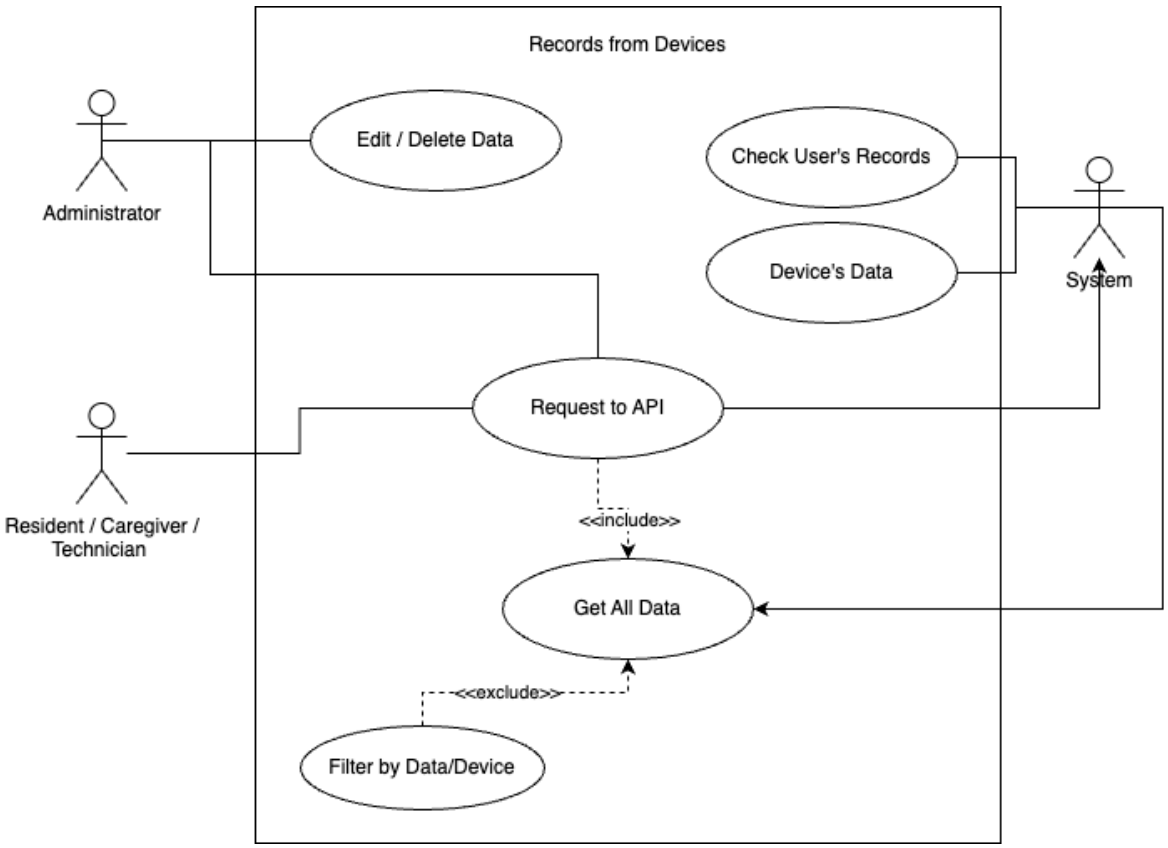

Figure 3.2: Diagram that describes the access to record data of the device on the platform.

The API has access to the database and the responsibility for responding to requests from different users and the Administration, as well as having the ability to process and treat data as necessary. Administration users are capable of accessing all data on the system, which allows them to edit/delete records. Table 3.12 outlines the specific actions for browsing device records, detailing the interactions for different user types on the platform.

Nome	Browse in Device Records
Actor	Platform (Different Users)
Preconditions	Establish connection with the system (API)
Data	Records saved on database.
Description	After gathering all the data from the API, with specification of selected device and data, the system is in conditions to show data records of the devices associated with Resident/Caretaker/Technician or all devices when the user is Administrator.

Table 3.12: Browse in Device Records / Data.

3.2.4.3 Access to Platform - Administration

The diagram presented in Figure 3.3 outlines the actions available to the Administrator on the platform, showcasing their ability to manage users, devices, and associated data. It provides a visual representation of the core functionalities, including user management, device administration, and data handling. Table 3.13 provides further details, describing the specific interactions the Administrator can have with the platform, along with the necessary preconditions and data involved.

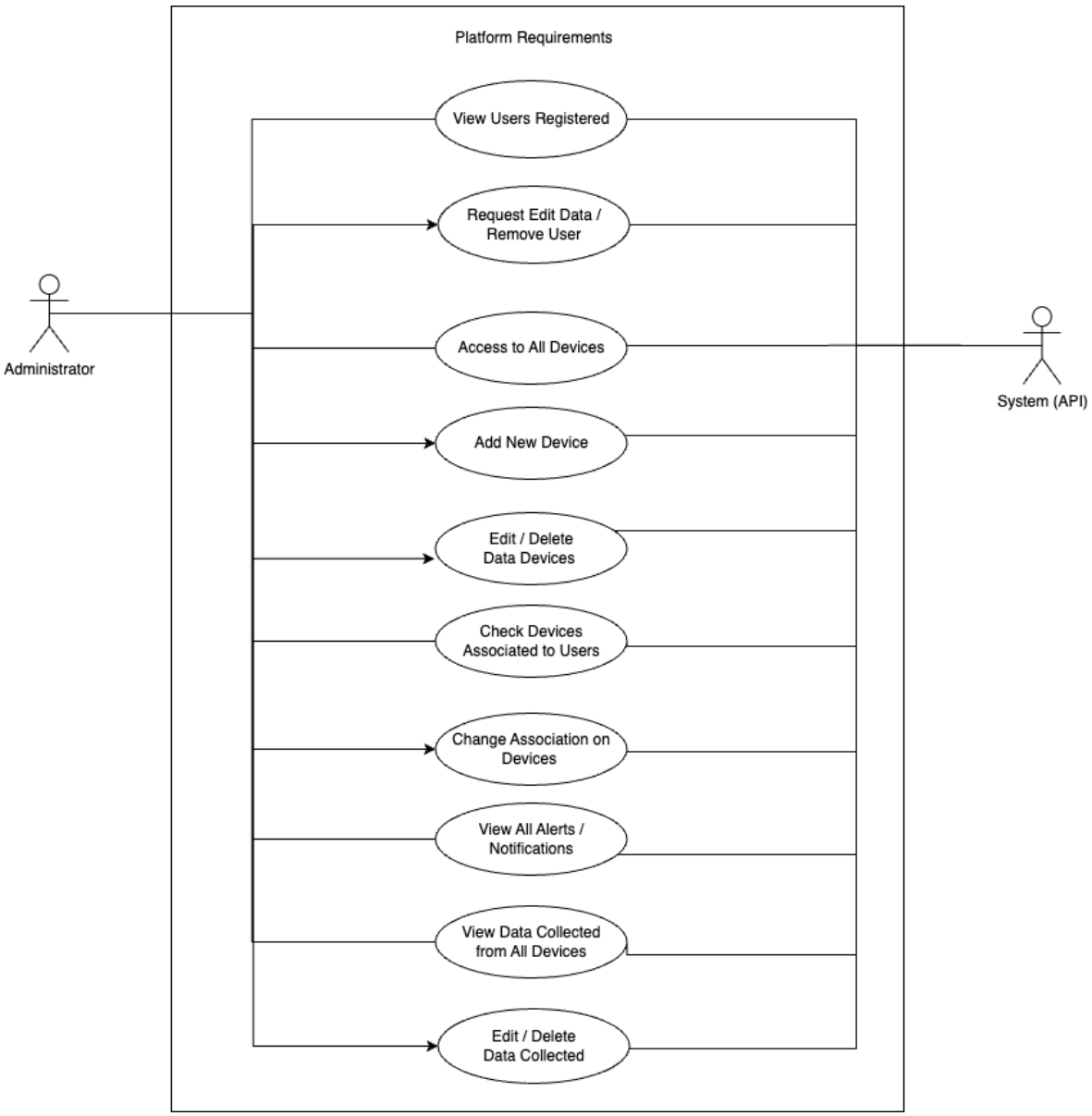

Figure 3.3: Diagram that describes what Administrator can do in the platform.

Nome	Browse in Platform like Administrator
Actor	Administrator
Preconditions	Establish connection with the system (API); Have permissions to access all platforms (admin).
Data	Records, users, devices, and alerts are saved on the database.
Description	After gathering all the data from the API, with specifications of the selected device and data, the system is in conditions to show data records as well as alerts associated to each user and all data recorded by sensors.

Table 3.13: Browse in Platform by Administrator.

3.2.4.4 Access to Platform - User

The diagram presented in Figure 3.4 illustrates the access and functionalities available to Residents, Caretakers, or Technicians on the platform. It outlines their ability to manage devices, view and handle alerts, and access collected data. Table 3.14 further elaborates on the specific actions these users can perform, detailing the conditions and data associated with their interactions on the platform.

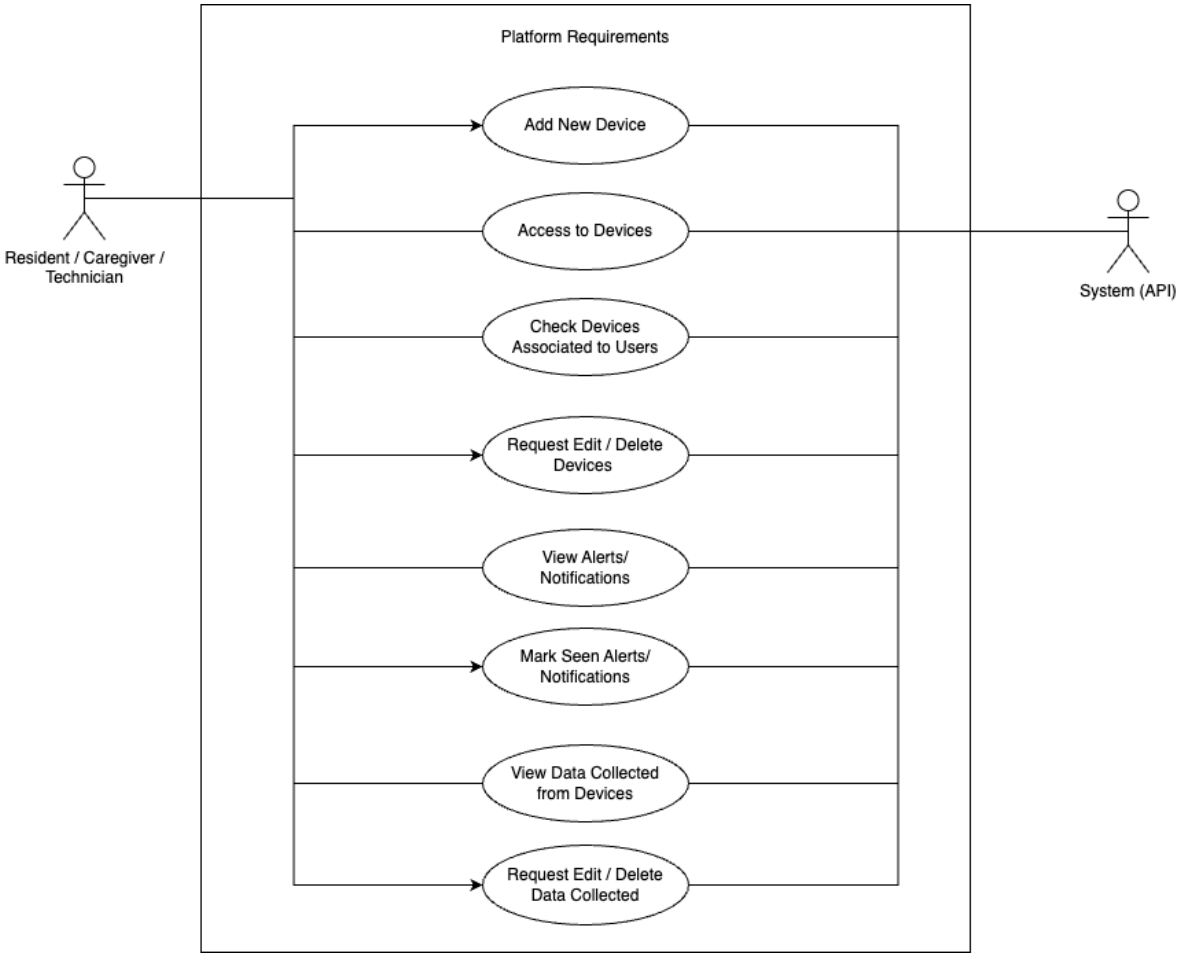

Figure 3.4: Diagram that describes the access to the platform by Resident / Caretaker / Technician.

Nome	Browse in Platform as Resident / Caretaker / Technician
Actor	Resident / Caretaker / Technician
Preconditions	Establish connection with the system (API); Have devices associated and operational.
Data	Records, devices, and alerts saved on the database, associated with the device of the user.
Description	After gathering all the data from the API, with the selected device of the user and data, the system is in conditions to show data records and verifying alerts associated with users and all data recorded by the sensors.

Table 3.14: Browse in Platform by Resident / Caretaker / Technician.

3.2.4.5 Authentication on Platform

The use case represented in Figure 3.5 intends to demonstrate the authentication process, in which the user can register if it is the first time on the platform or simply login. If it is the first time of registering the user, a verification code is sent via email to confirm their identity. If the data entered does not match the registration requirements, the verification code is incorrect, or, in the case of login, the password entered into the system, a warning message should be displayed.

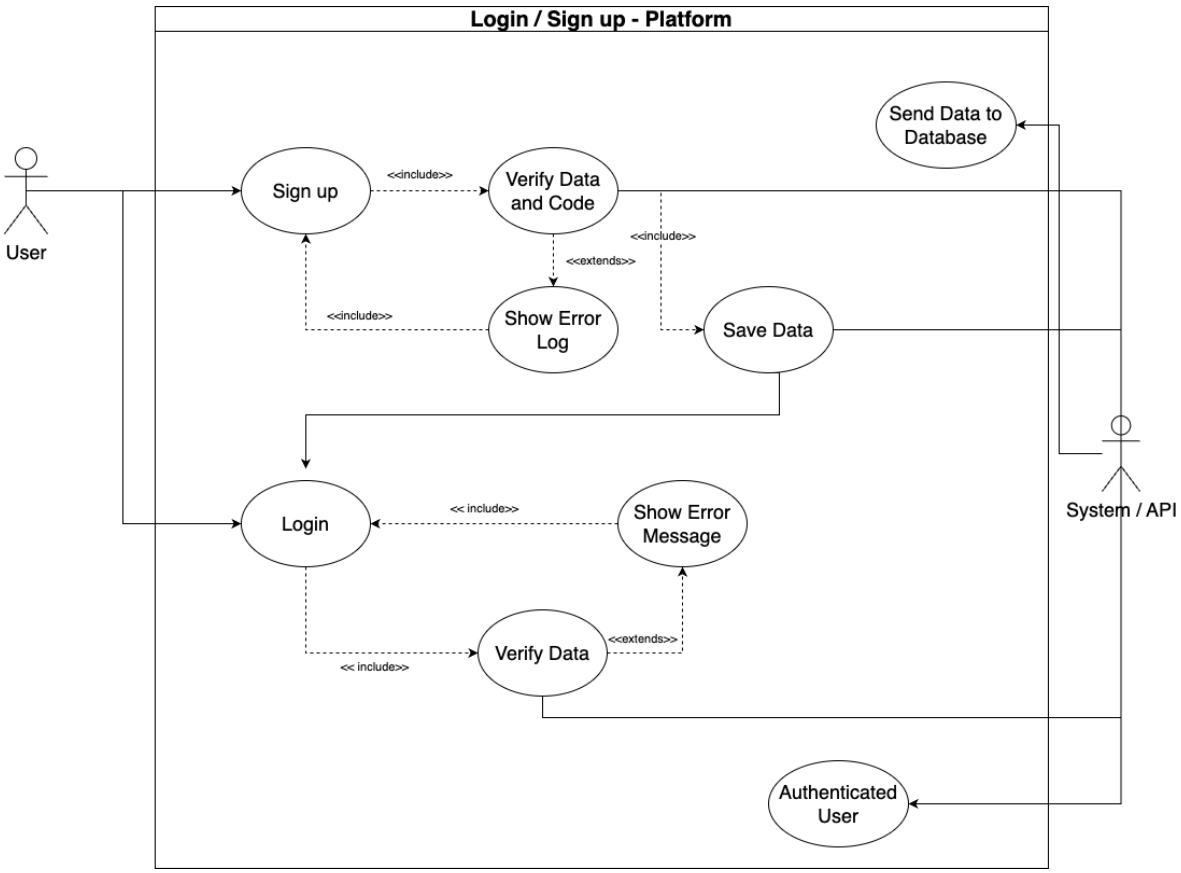

Figure 3.5: Diagram that describes the authentication process for the user in the platform.

3.2.4.6 Reset on Platform

The use case represented in Figure 3.6 aims to demonstrate the password reset process, where the user can request a reset if they have forgotten their current password. After the request, a verification code is sent via email to confirm the user's identity. If the email provided does not match any account registered in the system, or if the verification code is incorrect, an error message should be displayed.

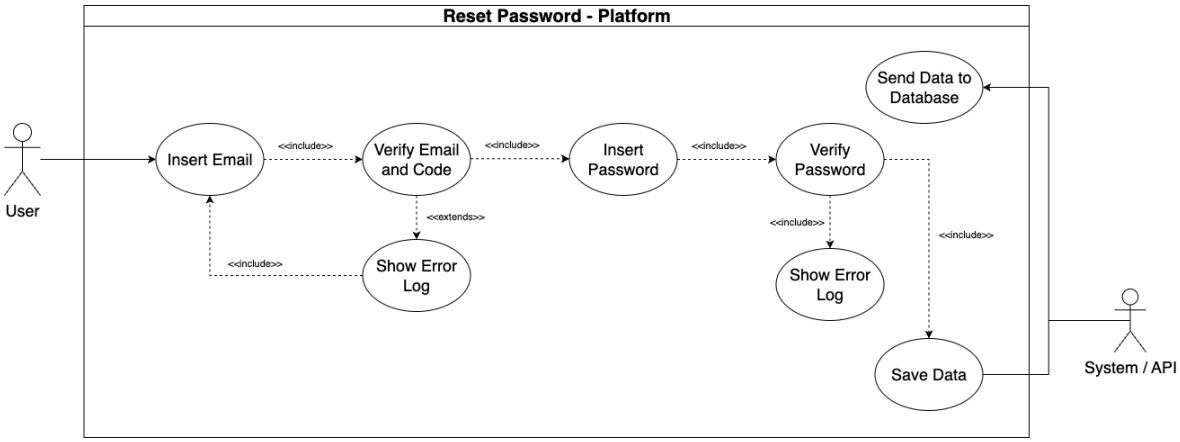

Figure 3.6: Diagram that describes the authentication process for the user in the platform.

3.2.4.7 Associate Devices to User

The process of associating devices in the system occurs when a user, such as a caretaker, resident, or technician, enters an admin code to link devices to a house. The administrator also plays a role by generating and managing the admin code, which is used to control device associations. Figure 3.7 illustrates this flow in detail, showing how users can insert the admin code, verify its validity, and assign device locations within a house. The API ensures constant communication with the database for verifying the code, retrieving device data, and storing the new configurations. Additionally, the administrator can access and modify the admin code as needed.

3.2.4.8 Manage User on Platform

Personal data management is possible when a registered user wants to modify their data. Another possibility is for the administrator to access and change any user's data. Figure 3.8 represents these cases in detail, as well as access to the API for checking and storing changes. Access to the database is constant via the API, to view and modify the data obtained.

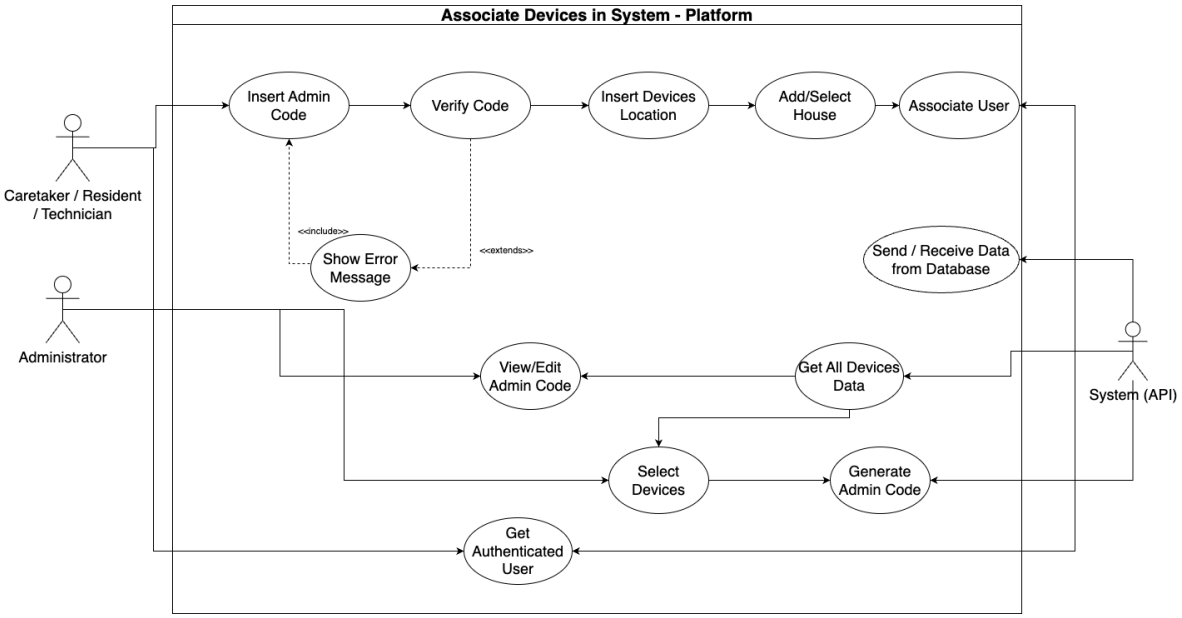

Figure 3.7: Diagram illustrating the process of device association in the system, detailing user interactions and administrative control over device management.

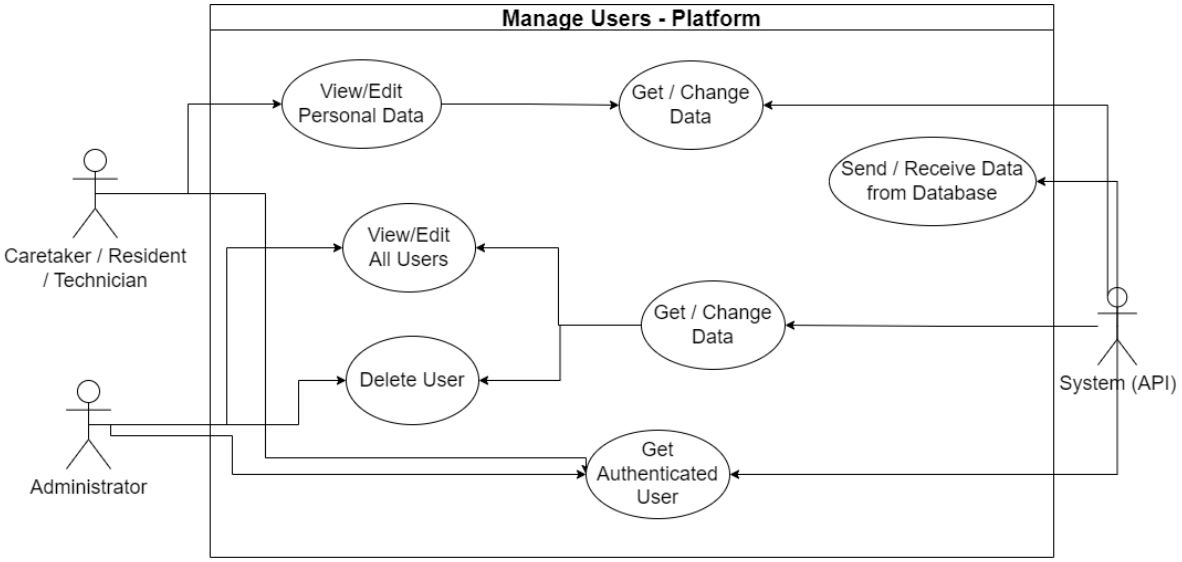

Figure 3.8: Diagram that describes the management of users, with the possibility to change their parameters by the user.

3.2.4.9 Notifications System - User

The Figure 3.9 aims to demonstrate the cases in a specific way, in which, similar to the visualization of records, notifications, and alerts shown to the user are only related to the devices associated with them. On the other hand, the administrator has access to any device associated with any user.

3.2.4.10 Notifications for Maintenance of System

System maintenance is performed automatically by the system itself, which detects all issues promptly as they arise. Administrators can view system reports and have the privilege to

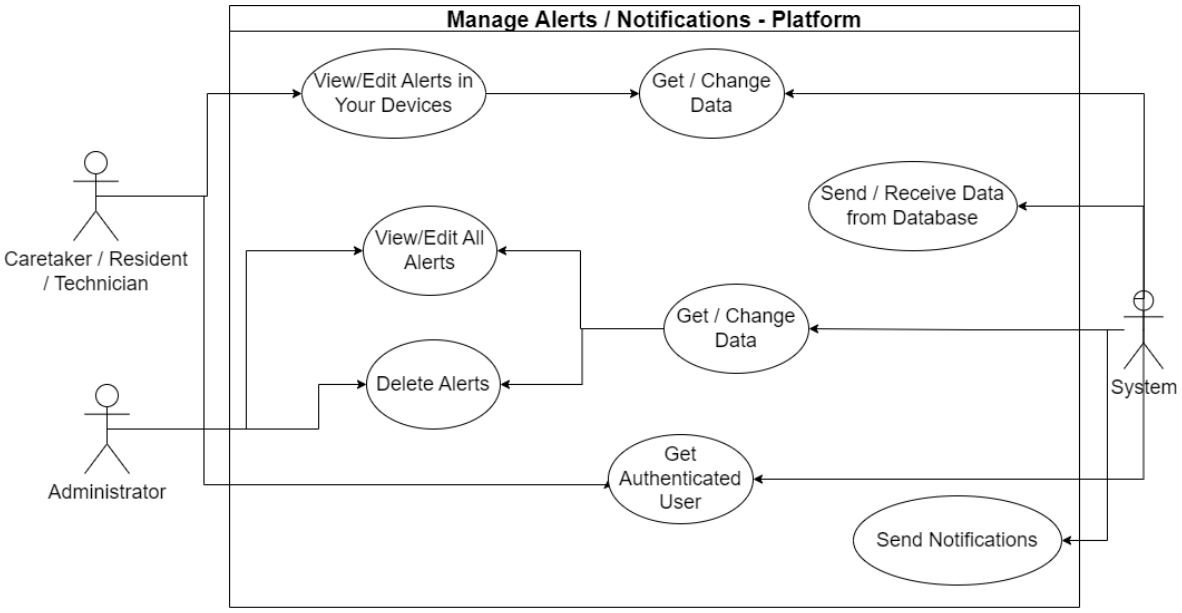

Figure 3.9: Diagram that describes the management of alerts or notifications, with the possibility to edit or delete the alerts.

resolve and remove these problems from the system once they are resolved. This case is demonstrated in Figure 3.10.

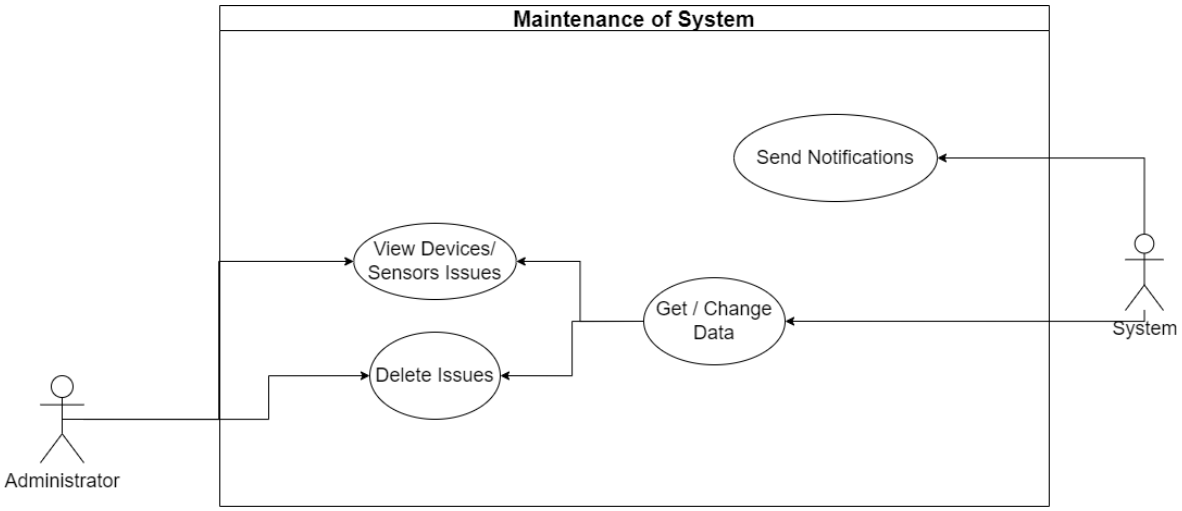

Figure 3.10: Diagram that describes the maintenance of the system, with the Administrator interaction.

3.2.4.11 Decision Support Mechanism

To assist in the analysis and decision-making process based on all collected data, the system utilizes machine learning mechanisms to provide predictions of future data. Figure 3.11 illustrates the system's influence in performing this operation and generating suggestions regarding pollutant gases.

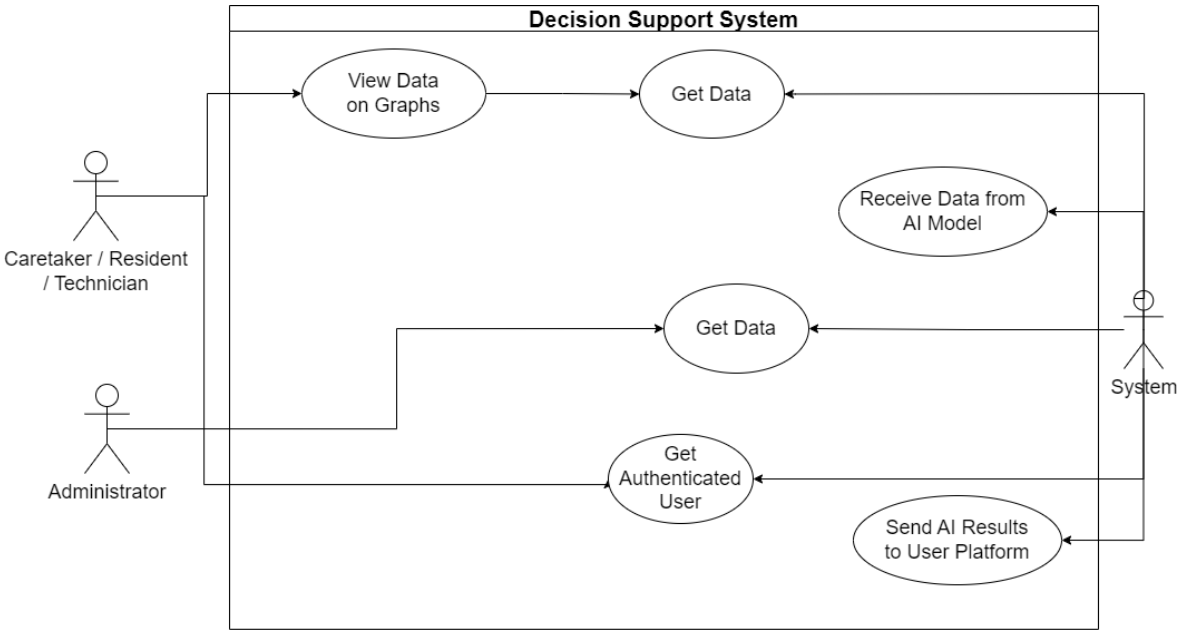

Figure 3.11: Diagram that describes decision support system, when AI model can predict the soon values to minimize the damage to high pollutant levels.

3.2.5 Activity Diagrams

Figure 3.12, 3.13, and 3.14 are activity diagrams to show the dynamism of the system. Figure 3.12 illustrates the sequential behavior of all the components of the system (local hardware, Cloud infrastructure, and Application) while Figure 3.13 focuses on illustrating the generation of notifications in the Cloud Infrastructure and Figure 3.14 demonstrates how the data can be viewed by the user in the application.

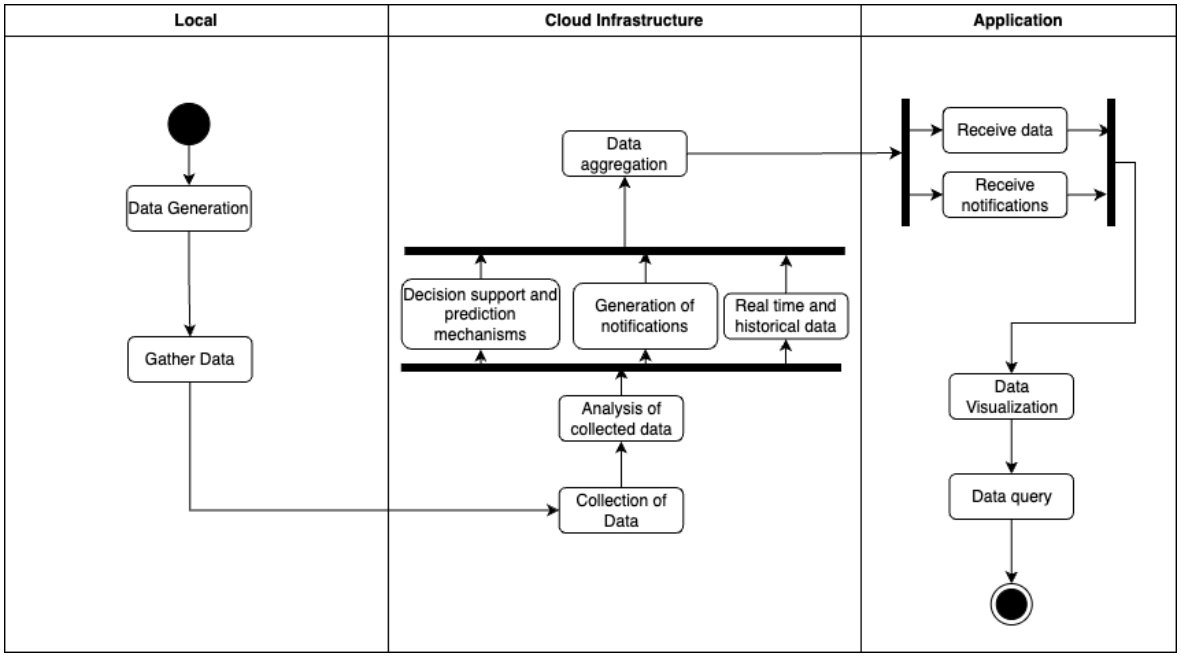

Figure 3.12: UML Activity Diagram for the System, Divided in Local, Cloud, and Application.

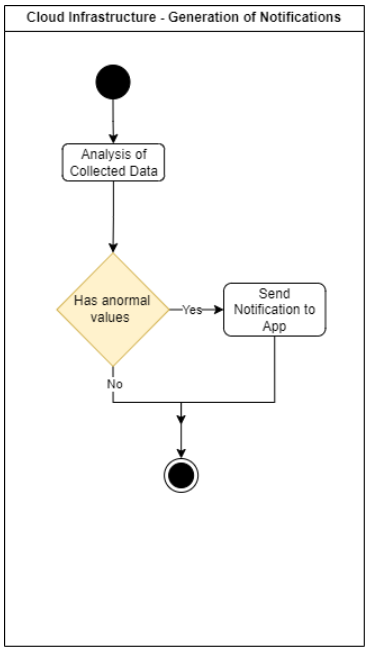

Figure 3.13: UML Activity Diagram for the Cloud Infrastructure, namely the Generation of Notifications.

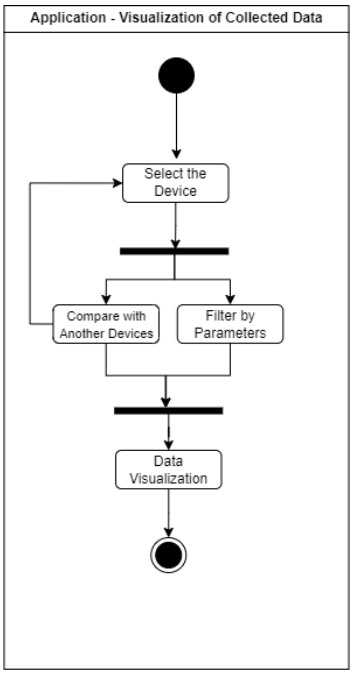

Figure 3.14: UML Activity Diagram for the Visualization of Collected Data on the platform.

3.2.6 Class Diagram

A class diagram is an essential tool in software engineering, representing the static structure of an object-oriented system. It illustrates the classes, their attributes, methods, and the relationships between them, including inheritance, association, and composition. The diagram shown in Figure 3.15 plays a crucial role in comprehending the architecture of the system, identifying the responsibilities assigned to each class, and facilitating communication between developers and stakeholders. Additionally, it serves as a valuable guide for code implementation. In summary, the class diagram aids in the clear and accurate organization and planning of the overall structure of the software.

In the diagram presented in Figure 3.15, it is possible to identify that various stakeholders can access the platform, as specified in the User class under type_of_user, which can take on different values such as 'resident' or 'admin'. Each user is required to enter their personal information into the platform (name, phone, email, password). Each user can be associated with a House, and a House can be associated with multiple Users, creating a many-to-many relationship. This necessitates the existence of an intermediary class (HouseUser) to establish this association. The House class includes information about the houses where RuraLTHINGS devices are installed, with the system storing details such as the street where the house is located, geographical coordinates (latitude and longitude), as well as a flag indicating the region where the house is situated (Fundão or Pinhel). Multiple devices can be installed in a house by the user using an AdminCode provided at the time of equipment purchase, which is referenced by the field admin_code_id in the DeviceInfo class.

The DeviceInfo class contains the date of the last insertion into the system (lastUpdate)

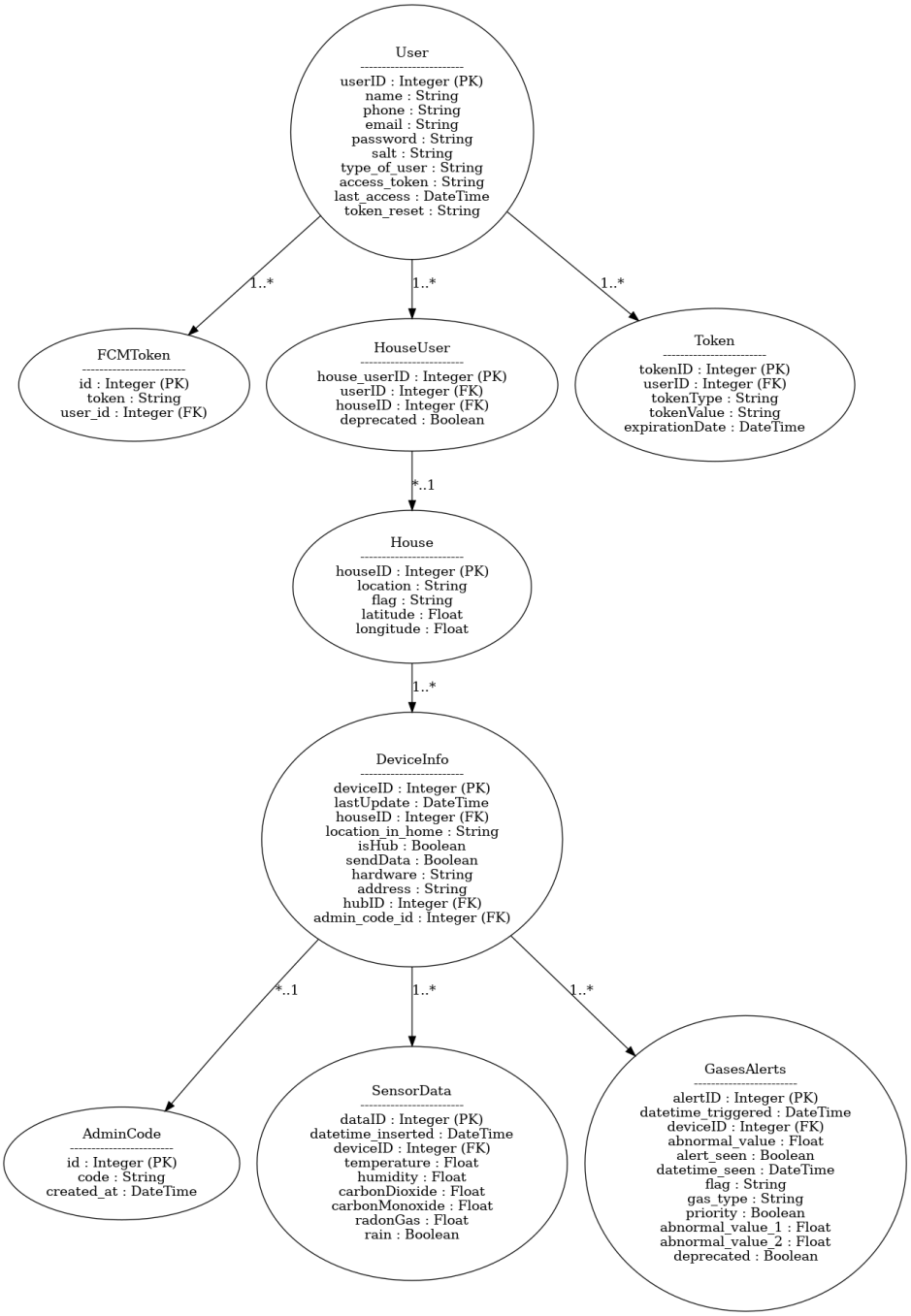

Figure 3.15: UML Class Diagram.

from this device. Specifying the device's location within the house is also essential (location_in_home). Given that there are various types of devices within the RuraLTHINGS ecosystem, it is necessary to specify whether it is a Hub (isHub), a device that collects data from multiple devices and sends it to the system. The boolean attribute sendData is useful because the hub itself can simultaneously send its own data, not just the data from connected devices. The hubID attribute allows the association of devices with hubs when they depend on the latter to function correctly.

The data collected by the sensors constitutes a significant portion of the attributes in the SensorData table, specifically temperature, humidity, carbonDioxide, carbonMonoxide, radon-

Gas. The remaining attributes indicate when the data was inserted into the system (datetime_inserted), as well as the device that collected the data (deviceID). As this data is introduced into the system, a trigger in the database automatically inserts the DateTime at the moment of entry.

Another key trigger verifies, at the moment of data insertion, whether the environmental gas levels exceed the acceptable thresholds. If they do, a new entry is automatically generated in the GasesAlerts table. This table monitors when the abnormal value occurred and when it was acknowledged by the user, as indicated by the respective datetime_triggered and datetime_seen attributes. To optimize the process of data visualization and alerting users, it is crucial to assign priority levels. Priority is also established with a trigger that continuously analyzes whether the altered values were persistent and whether more than one environmental gas was affected. In such cases, the actions to be taken are more urgent, and notifications to users are prioritized. In just one row of the table, multiple altered values can be included, if they exist, in each of the attributes (abnormal_value), with one attribute for each environmental gas. The gas_type attribute describes the name of the gas that shows values outside the limits recommended by the WHO.

The FCMToken class is crucial for the notification system, enabling the platform to send push notifications when abnormal values are detected. Each user can have multiple devices associated with different FCM tokens, allowing notifications to be sent to all devices linked to the same user. This ensures real-time alerts across multiple platforms.

The Token class manages user authentication tokens, such as session tokens and password reset tokens. This class stores information like the tokenType, tokenValue, and expirationDate, ensuring that user sessions are securely managed and that tokens expire after a certain period. These tokens are crucial for maintaining security within the platform, ensuring that users' identities are validated without needing to log in repeatedly.

The AdminCode is provided to the user at the time of device purchase, allowing them to register the devices to their account/platform later. This code enables the user to associate the devices with their account, ensuring access to the sensor data collected by the registered devices.

The deprecated attribute is present in some classes to filter out data that is to be removed, but in a way that it does not actually leave the database. Rather, it simply stops being displayed. This enables the system to keep historical records of inactive devices, users, or houses without displaying them in current views or reports.

3.2.7 System Architecture : Platform Component

The proposed IoT system collects data from various sensors deployed in rural environments, serving as a foundation for a central application. Due to the complexity of integrating multiple devices and architectures, understanding the system can be challenging without the aid of UML diagrams. Therefore, this section provides a detailed study of the RuraLTHINGS system, focusing on structural and behavioral UML diagrams. Figure 3.16, in particular,

illustrates the different layers of the RuraLTHINGS system and complements the diagram presented in Figure 1.1.

The component addressed in this document is entirely focused on the Platform Component and partially on the Cloud and Network Component. The other parts, although essential for the operation of the entire system, are not covered in this document. The diagram presented in Figure 3.16 addresses the user interface, the presence of data analysis mechanisms, notifications, data retrieval from an API, and security mechanisms that ensure secure data exchanges. In turn, the DSS mechanism is interconnected with the API for data exchange, as well as the notification system.

The API, through Transmission Control Protocol / Internet Protocol (TCP/IP) connections, access the database and can detect any triggers that may have occurred in order to send notifications to users. The connection/environment layer and physical layer are only briefly addressed to understand the origin of the data and the type of data expected by the platform.

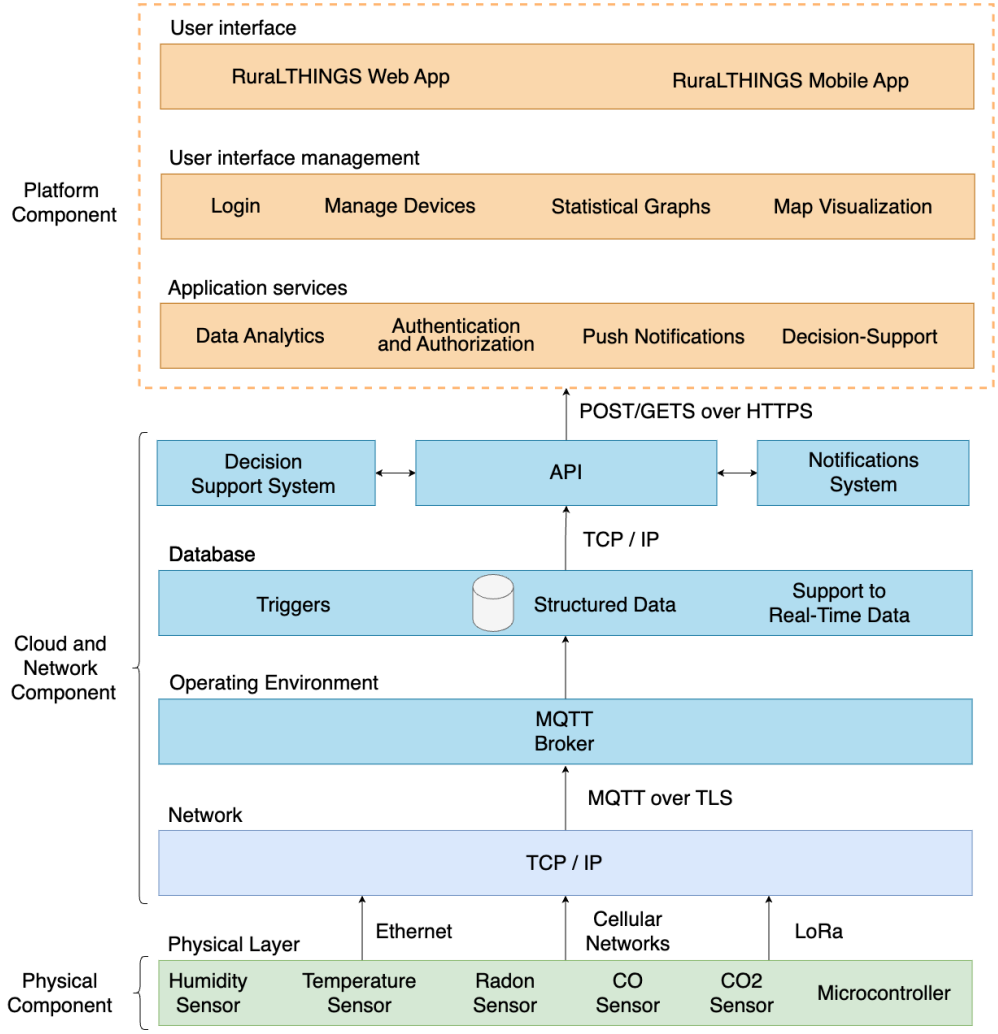

Figure 3.16: System Architecture.

3.3 Technologies Used

The technologies used will be described in the following subsections. A set of tools is required to construct the system and develop the necessary software. These tools enable the execution of various functionalities and meet the defined requirements.

3.3.1 FastAPI

FastAPI is a cutting-edge, high-performance web framework that is well-known for being quick and simple to use for creating Python APIs. It uses Pydantic for data validation and Starlette for the web framework to provide automated input and output validation using Python type hints. Async and await are two programming languages that FastAPI fully supports, which makes it perfect for high-concurrency applications like APIs that mostly rely on external I/O activities. Additionally, without the need for third-party tools like Postman, FastAPI automatically creates interactive API documentation using OpenAPI and Swagger. This documentation enables us to explore, test, and comprehend the API struct, including details like input types, authentication methods, and requested permissions.

Secure access control is made by FastAPI's simple integration with contemporary authentication solutions like OAuth2, which upholds data integrity and security. By confirming an authenticated user's permission status using tokens, FastAPI makes sure that only authorized users may access rights. The system can confirm that the user has authorization to access the requested data by using the user-specific data that each token can hold. Tools such as Uvicorn, an ASGI server for high-performance, low-latency deployment, are supported by the framework.

3.3.2 Mail Server

A mail server is a system for handling the delivery, receipt, and storage of electronic messages from one user to another. Acting as the primary route for mail processing, it ensures the effective distribution of messages between senders and recipients. Mail servers are mainly divided into two types. The first type is the receiving servers, such as Internet Message Access Protocol (IMAP) and Post Office Protocol3 (POP3), which allow users to access their inboxes and retrieve messages, and Simple Mail Transfer Protocol (SMTP), which is used to send and transmit emails.

The goal is to send emails from a service provided by dominios.pt, the cPanel email provider, which consists of a powerful tool for managing email servers. cPanel allows the automatic sending of emails, including registration verification tokens and password recovery, to custom email addresses associated with the domain, thus simplifying the configuration and use of SMTP. Using back-end endpoints called by the front-end, this functionality can be coupled to enable automatic email sending for activities such as user registration or password recovery. Leveraging the email provider provided at the time of domain establishment ensures additional control and integrity when sending important emails, thus improving contact with users more professionally and efficiently.

3.3.3 Firebase Cloud Messaging

With the service developed by Google, the FCM, it is possible to send notifications to users on many platforms such as web browsers, iOS and Android. FCM allows servers to send notifications to user devices in real-time, even while the application is not open. How it makes it easy to provide updates, warnings, and reminders, this feature is crucial to maintaining user engagement and knowledge. In addition to supporting visible notifications, the FCM also allows to send data messages in the background, allowing the application to synchronize or change data without requiring user input. FCM offers a comprehensive solution for real-time communication between servers and user devices due to their scalability and multi-platform support.

For the execution of this project, a specific table must be created in the database containing the FCM and associated user IDs. Due to this architecture, each user's FCM tokens are recorded for effective and personalized notifications. The front conveys the user's current FCM token to an API endpoint assigned to the login, and then the API checks that the database already contains this token. If it is not still in the table, the system automatically inserts it.

After searching the table for all tokens connected to a user's ID, the system can notify that user on the devices that are logged in. The API ensures that the user receives the notification in real-time by sending it to the registered device(s). This method offers a scalable solution to managing alerts on multiple devices connected to a single user as well as efficient FCM tokens management.

3.3.4 Flutter

Flutter is a free and open-source framework created by Google. This framework allows software developers to create native applications for both iOS and Android (mobile phones) and Websites with a single code written in just one programming language (Dart). Flutter consists of a vast library of widgets for use in building user interfaces, as well as a Software Development Kit. All of this allows the code initially written in the Dart language to be compiled into the operating system language on which the application will be used, enabling access to the native resources of each device, which enhances its performance.

3.3.5 Dart

The Dart language was used to make requests to the developed API and create the front-end of this project, particularly to create layouts and an appealing User Interface. The application was developed with the concept of Null Safety because this concept ensures that a variable cannot have a null value unless it is initialized as such. Therefore, all errors caused by referencing null values will be shown during the code compilation process. This is essential, especially when using certain widgets, and it also helps the programmer to develop more coherent code and to identify errors in it more quickly, especially when a significant portion of the application's back-end consists of making requests to an external API.

3.3.6 Android Studio

Android Studio is an Integrated Development Environment (IDE), based on the JetBrains Intellij IDEA platform, dedicated to mobile device programming, especially for Android. This IDE also provides the hot-reload and hot-restart functions. Hot-reload loads code changes in the emulator and rebuilds the widget tree, preserving the App's state. All of this happens in a very short amount of time. Hot-restart, on the other hand, also loads code changes into the emulator, but in doing so, the application's state is lost because it is restarted, so it ends up being a slower process as well.

3.4 Conclusion

This Chapter explored the software engineering concepts and technologies employed in the creation of the RuraLTHINGS platform. It examined the strategies implemented to provide a resilient and effective system for the integrated gathering, processing, and analysis of sensor data, along with the application of AI techniques for predicting pollutant levels and sending timely alerts. Chapter 4 will address the development of this DSS, while Chapter 5 will present the resultant software, emphasizing the actual implementation of the software engineering principles and methodologies discussed in this Chapter.

Chapter 4

Intelligent Decision Support System

4.1 Introduction

This Chapter presents in detail the machine learning-based decision support system that was developed on top of the above-described digital platform.

Providing users with an innovative platform that enables real-time data analysis and decision-making to minimize risk and improve quality of life requires system optimization and continuous communication to coordinate all components across platforms and network infrastructure. These elements are essential for the decision-making system to operate effectively and provide timely alerts.

The following subsections will cover the techniques commonly used in time series, including data analysis exploration, data preprocessing, model selection, and model evaluation metrics. In summary, this section will discuss the use of AI mechanisms for detecting environmental gases and the implementation of the DSS in the alert system. A study will be conducted to identify the most effective AI model for predicting values through training and subsequent evaluation, as Chapter 5 will show. The final model will be selected based on the evaluation metrics outlined in Section 4.1.5.

4.1.1 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an initial phase of data analysis that helps summarize key trends and aspects within a dataset. It plays a crucial role in recognizing patterns, understanding relationships between variables, and assessing data distribution before applying statistical techniques or predictive models. Furthermore, EDA lets one identify anomalies, missing values, and data discrepancies that, in improper handling, might skew the outcomes. Understanding data distribution and connection between variables allows EDA to also choose pertinent features for the model, thereby improving prediction performance. Therefore, EDA offers a strong basis for modeling as it guarantees that predictive models are grounded on excellent data and a clear awareness of the fundamental dynamics [91].

4.1.1.1 Data Collection

Within the scope of this project, the dataset consists of sensory recordings collected from inside homes using IoT devices. The environmental variables collected consist of ambient gases, namely carbon monoxide, carbon dioxide, and radon gas, as well as environmental conditions such as temperature and humidity. These values, after being collected by the RuraLTHINGS devices, are sent securely and encrypted to a database, which aggregates and

stores all the collected values. The purpose of carrying out these measurements is to be able to provide the data collected in real-time and, at the same time, predict risks that may arise with the emergence of altered values that are harmful to health. The active sensors enable real-time data collection, allowing for accurate and immediate analysis when integrated with a properly connected IoT system.

For the development of the decision support system, an external dataset by Ángel Rodés, provided in 2022 on GitHub [92], was used due to the lack of a robust dataset, since the time required for data collection from RuraLTHINGS devices has not yet been reached. This dataset was collected over a period of approximately 3 months from an Airthings Wave Air device [84] and exported from the platform for further manipulation. Although this dataset does not include carbon monoxide, it does include the remaining variables, which makes it quite similar to the data expected during the development and production phase of this project. As such, the dataset was prepared and modeled for training using machine learning algorithms and statistical models.

4.1.1.2 Data Preprocessing

Based on the dataset, the environmental measurements for the decision support system comprise radon concentration measured in Bq/m³, temperature measured in °C, humidity measured in percentage, atmospheric pressure, carbon dioxide (CO_2) measured in ppm and VOC measured in ppb. The latter was not used (VOC), as well as the atmospheric pressure values, as they are not among the objectives of the RuraLTHINGS project and, therefore, will not be used as input to the model.

In total, there are about 17,236 records, of which about 91% (15,783) are NaNs due to radon gas that is collected every hour, and the remaining variables are collected at five-minute intervals. Figure 4.1 represents the data provided by the dataset, without any manipulation of the data, which includes missing values, as can be seen. In the case of radon gas, this is where the largest absence of data is noticed, in the other variables it ends up not being noticed since they only constitute about 0.13% of the missing data (15 in temperature and 8 in carbon dioxide).

Given the decreasing frequency of radon accumulation, it is imperative to carefully review the data, especially for other features observed over shorter periods. For example, radon values are only present in the hourly dataset, which required the use of linear interpolation mechanisms to fill in the gaps, as shown in Figure 4.2. The initial dataset, before any preprocessing, contained missing values in almost all other variables except for 'Humidity', which had no null values. As such, missing value identification techniques and linear interpolation were important for the other variables (radon, temperature, CO_2, and pressure), which also had some missing data.

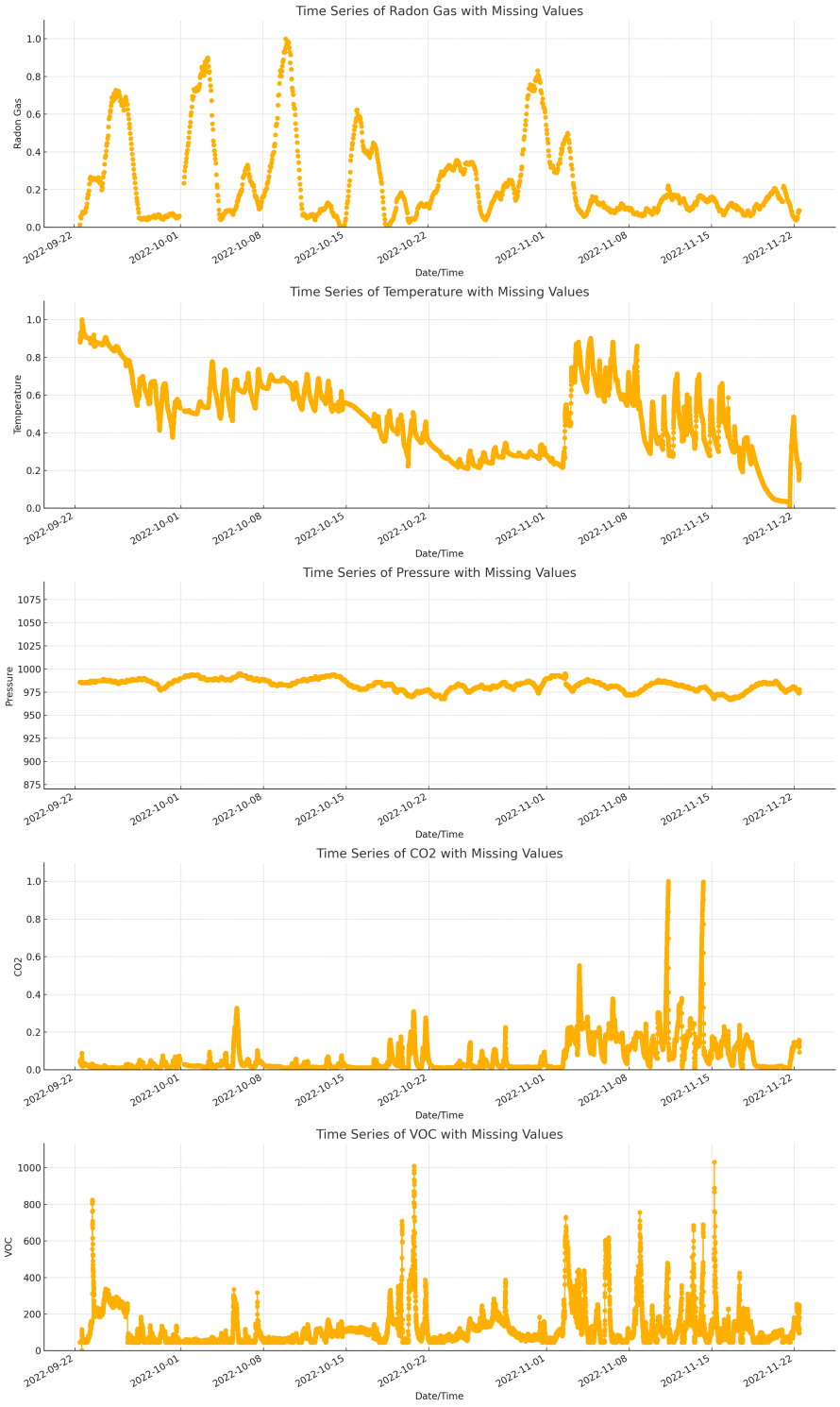

Figure 4.1: Representation of Dataset with Missing Values.

Time series data necessitated the use of several preprocessing techniques, including linear interpolation and NaNs removal, which are part of the Data Cleaning stage and deal with missing values explicitly. To guarantee data integrity before moving on to the following phases, these procedures are vital for retaining temporal patterns as they preserve the overall trend of the data. In addition, this helps prevent errors in later stages. Figure 4.3 shows the data following the proper processing of data cleaning and treatment.

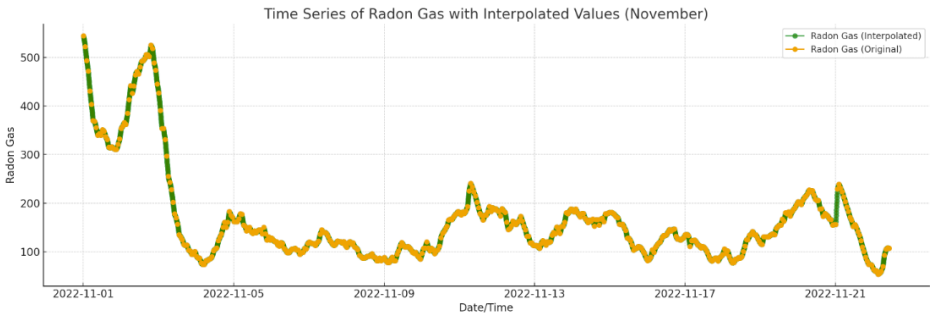

Figure 4.2: Interpolated Values on Radon Gas.

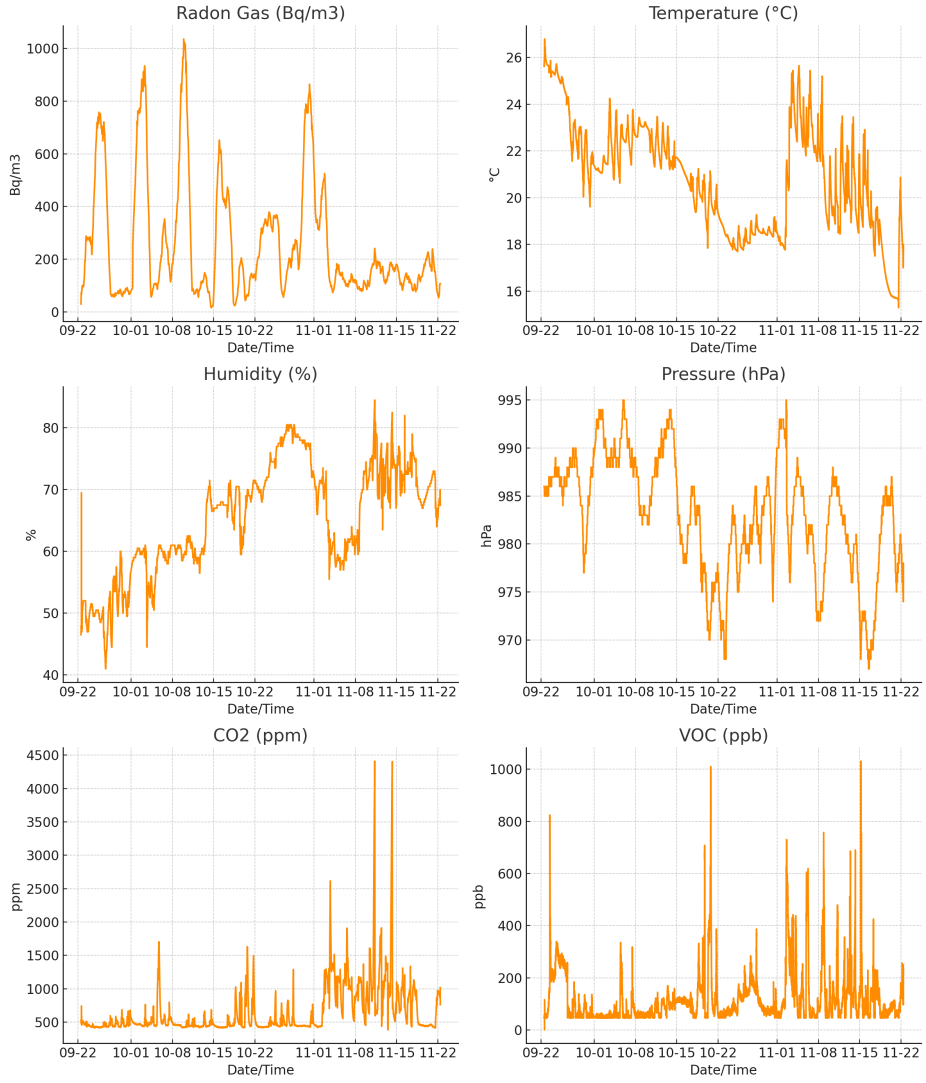

Figure 4.3: Representation of Dataset.

Based on the various graphs presented in Figure 4.3, some conclusions can be drawn. For temperature, the graph shows regular daily cycles and fluctuations over time. At larger intervals throughout the day, humidity also follows a cyclical trend. In terms of pressure, variations fit with environmental changes. CO_2 levels show fluctuations that seem to be impacted by external elements, perhaps of an environmental nature. On the other hand, radon is the most easily changeable factor and shows more sensitivity to the environment, as expected.

The next phase of data preparation involves scaling or normalizing the cleaned data. Usually between 0 and 1, scaling the data helps to change the variables to a common scale, so ensuring that all variables have values within a similar range. This is thus because scaling lowers the variance of the variables therefore enabling stable and faster convergence of optimization techniques.

All things considered, the data distribution is changed to scale the points to fluctuate within the same range, thereby reducing the influence of outliers and so lowering the danger of affecting data analysis. These methods also help to better comprehend the results so that ML models may concentrate on the patterns and trends shown in the data and prevent technical issues such as gradient explosion or vanishing during the training process.

Based on the possible connections between variables, it is necessary to decide which ones are significant and thus choose the variables to be used. In the study of time series that include environmental variables such as Temperature, Humidity, CO_2, and Radon Gas, it is possible to examine different trends and seasonality from the data collected.

According to these Figures, temperature indicates a general downward trend, while humidity shows a slow increase until mid-November, as expected as winter approaches, and then a smaller decline. On the other hand, radon gas showed consistent cyclical changes, with a peak in October followed by a decrease in November, suggesting specific environmental conditions affecting its concentration. In general, radon gas shows greater variability and significant peaks, as expected, due to its chemical non-reactivity with other elements in the air, which causes it to disperse rapidly in the air with ventilation/movement of ambient air. CO_2 showed a consistent increase with clear peaks at some intervals, possibly due to seasonal factors or human activities. With certain exposure events to waste influencing the readings, particularly in the case of CO_2 and radon gas, all variables revealed notable seasonal components representing daily cycles. These observations imply the need to keep an eye on the factors to detect and minimize anomalies.

Temporal aspects should be given relevance in the data since seasonality exists there and shows repeated patterns at regular intervals. Helping the prediction model to capture temporal trends depends on the development of additional characteristics. Daily, weekly, monthly, and annual levels of seasonality can be found, so extracting day, month, year, and hour helps the model to reflect these trends. Both nighttime and daytime should also be taken into account since sensors gather data around the clock and there can be notable variations.

The dataset indicates that the current seasonality requires developing additional characteristics stressing the date/time. This kind of feature engineering guarantees more accurate forecasts and helps to enhance the predictive model's performance.

Normally, time series analysis uses the AutoCorrelation Function (ACF) and the Partial AutoCorrelation Function (PACF) to find out how much new data depends on old data. It finds patterns like seasonality and trends by measuring the correlation between observations of a time series and their lags. The PACF, on the other hand, finds the direct correlation between the series and a specific lag, removing the effect of intermediate lags. This is important for

figuring out the right number of lags in Auto-Regressive (AR) models [93].

The Appendix A.1.0.1 contains ACF and PACF graphs, which show a substantial autocorrelation at initial lags, especially for variables like temperature, humidity, and radon gas. This indicates that a variable's present value is heavily reliant on its most recent values, even those from a few hours ago. This autocorrelation may reflect patterns in the data that occur naturally throughout the day, such as temperature readings that rise during the day and plummet at night. The observations make it easy to see that, for example, temperature and humidity tend to show substantial daily cycles.

Considering the temporal dependence patterns of the graphs, it is reasonable to think about how the events of the day affect sensor readings. Different parts of the day, such as 'night1', 'day', and 'night2', are used to explicitly represent these daily cyclic or seasonal changes. To account for this temporal fluctuation, the model can be taught to segment time (or create a timing variable) so that patterns during the day, night, and following variations in the night can be distinguished. To make it usable by machine learning algorithms, the category night1, day, and night2 labels on the time variable need to be transformed into a numerical format. By using Ordinal Encoding (OrdinalEncoder) in this case, we are assuming that these classes follow a logical path.

The events take place in chronological order, with night, day, night, etc., making up the ordinal encoding. This follows from the reasoning that measurements made at 'night1' might be more indicative of daytime values than those taken at 'night2', which could help catch daytime fluctuations that do not occur at night. Using this method, the model can more accurately reflect the daily and diurnal fluctuations indicated by the autocorrelation patterns.

All these mechanisms used in Data Preprocessing are crucial for ensuring data consistency and guaranteeing that the data analysis is comprehensive, accurate, and useful, thereby providing valuable insights and enabling the construction of reliable predictive models.

The Augmented Dickey-Fuller (ADF) tests are fundamental statistical tests when analyzing time series, particularly when using statistical models for prediction, as they allow for verifying the stationarity of the data. Essentially, a time series is said to be stationary if its characteristics do not depend on the time at which the data are observed (Hyndman et al. 2018) [94].

The lack of systematic trends, like a constant rise or fall over time, is what makes a stationary series practical. After running the Augmented Dickey-Fuller (ADF) test on a time series, we compare the resulting p-value to a predetermined significance level, typically 0.05, to see if the data is stationary. A p-value below 0.05 indicates that the null hypothesis regarding the series having a unit root is rejected, leading to the conclusion that the time series is stationary. This means that its statistical properties remain constant over the course of the analysis. In contrast, if the p-value is bigger than 0.05, the possibility that the series is not stationary and might, in fact, exhibit characteristics that change over time is not ruled out.

According to Table 4.1, the results of the ADF test show that humidity, with values closer to 0, can be considered non-stationary, although it is on the threshold. Therefore, it is important to handle humidity with care in future analyses, considering possible transformations to stabilize its statistical properties. All other variables are considered stationary, meaning that although the values vary, this variation occurs around a fixed mean with constant variability (standard deviation).

Variable	ADF Statistic	p-value
Temperature	-3.7144	0.0039
Humidity	-2.8251	0.0548
CO_2	-9.3002	1.1179e-15
Radon Gas	-5.2814	5.9820e-06

Table 4.1: ADF Statistics and p-values for Various Variables.

The Temperature, CO_2, and Radon Gas time series are stationary, meaning their statistical features, including mean and variance, remain consistent across time. The Humidity time series does not meet the criterion for stationary behavior while being close. Current research has to handle humidity carefully, taking into consideration prospective changes to stabilize its statistical properties.

4.1.1.3 Data Visualization

Data can be divided into three basic components in time series analysis: trend, seasonality, and residual. Referring to long-term behavior, the trend shows a constant pattern of increase or decrease over time, which is present in the different variables of the dataset. A time series without seasonality is one in which there are no trends that repeat themselves routinely over time. Likewise, a dataset devoid of residuals suggests that the series does not have appreciable random fluctuations.

Correlation matrices are highly effective tools in EDA for uncovering linear relationships among multiple variables in a dataset. They offer a concise and comprehensive summary of these connections, facilitating the rapid recognition of patterns, trends, and important correlations. These are crucial for improving the selection of variables and guiding well-informed decision-making in predictive modeling.

The linear relationships among numerous major variables from the dataset under examination are shown in Figure 4.4, which is the correlation matrix. This matrix is examined to give information regarding the interdependencies between elements such as temperature, humidity, CO_2, and radon gas.

The presented correlation matrix reveals the linear relationships between the variables Temperature, Humidity, CO_2, and Radon Gas. The analysis highlights a strong negative correlation between temperature and humidity (-0.78), suggesting that as temperature increases, humidity tends to decrease, and vice versa. The inverse relationship between temperature and humidity, as demonstrated in Figure 4.5, can be crucial for understanding and modeling climate behavior or environmental processes in specific scenarios.

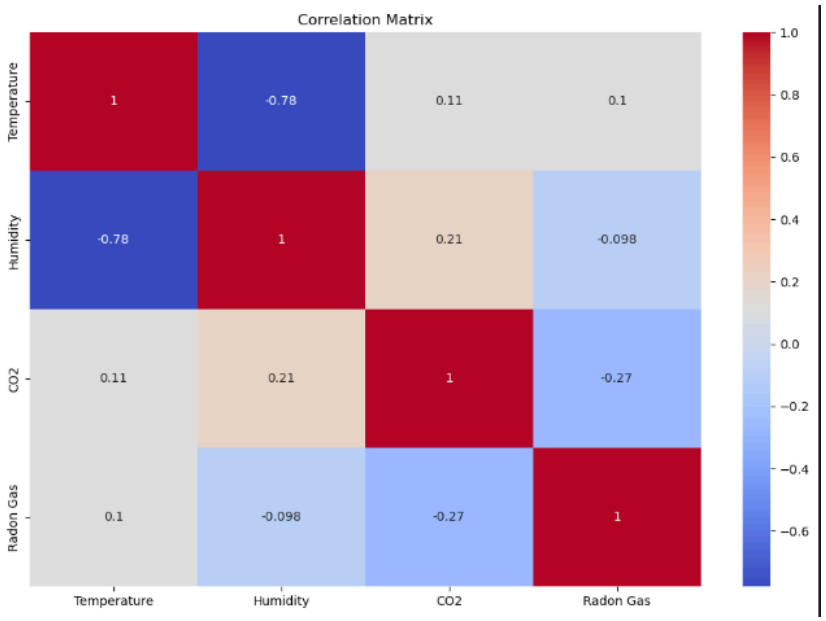

Figure 4.4: Correlation Matrix.

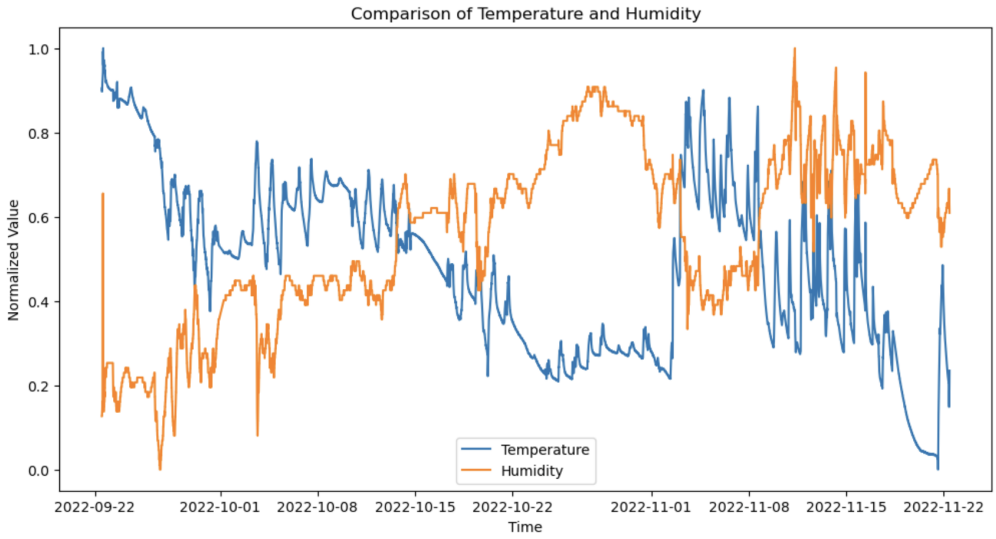

Figure 4.5: Correlation Analysis of Humidity and Temperature Values.

On the other hand, the correlations between the other variables (CO_2 and Radon Gas) or (CO_2 and Humidity) are relatively weak, with coefficients ranging from −0.27 to 0.21. These values indicate that there is no significant linear relationship between these variables in the dataset, implying that changes in CO_2 concentration or radon gas levels are not directly associated with variations in temperature or humidity. These findings suggest that, instead of relying solely on simple linear relationships, modeling these variables may benefit from techniques that capture more complex and dynamic interactions, such as nonlinear models, providing a richer and more detailed understanding of the behavior of these variables under different environmental conditions.

4.1.2 Data Splitting: Evaluation Methods

Data Splitting is one of the essential techniques for the development and evaluation of machine learning models. It is very necessary for the generation of precise forecasts that are capable of being properly applied to fresh data that has not been seen before. This ensures that a reliable evaluation of the model's efficacy and resilience in real-world settings is carried out [95].

This process is necessary to maintain the credibility of machine learning experiments and ensure that they will have practical effects. According to Arlot et al. 2010 [96], training a learning model and always using the same input data to evaluate its predictive ability produces biased results due to overfitting. Overfitting is a phenomenon in which the model captures superfluous noise and narrow patterns instead of accumulating knowledge of larger patterns. Data splitting helps identify and eliminate overfitting, which is another benefit of data splitting. Thus, within the scope of this project, two methodologies were used to divide the data:

- **Train-Validation-Test Split**;
- **Time Series Split**.

In the following sections, these methods will be explored to assess their efficacy and impact on the project.

4.1.2.1 Train-Validation-Test Split

Starting with the partition of the original dataset, a basic process of Data Splitting guarantees that the machine learning model generalizes properly, therefore evaluating the performance of such models. Starting with splitting the original dataset, this divide is the train-valuation-test split that verifies the model generalizes correctly. The given dataset might be split into many subsets usually comprising testing, validation, and training. Each of these parts has different objectives, but all together work towards a common goal:

- **Train Set**: This is the largest portion of the data, used to train the model. The objective is to adjust the model's parameters so that it learns the underlying patterns in the data;
- **Validation Set**: After training the model on the training set, it is evaluated on the validation set. This step is crucial to adjust model hyperparameters, identify possible problems such as overfitting and choose the best version of the model before the final evaluation. The model does not "see" this data during training, but it is used to fine-tune the learning process;
- **Test Set**: This is the portion of the data reserved for the final evaluation of the model, after tuning the hyperparameters and choosing the best model based on validation. The test set simulates the performance of the model on completely new data, providing a realistic estimate of how it should behave in production.

The Train-Validation-Test Split is a very successful technique for creating models that are

both robust and well-generalized. Nevertheless, it is crucial to exercise prudence while using it, especially in scenarios where the sequential arrangement of the data has importance, such as in time-based series.

4.1.2.2 Time Series Split

Designed especially for time series models, Time Series Split is a cross-validation method where maintaining the chronological order/temporality of data is crucial [97]. Unlike conventional methods, such K-Fold Cross-Valuation, which randomly splits and shuffles the data, the Time Series Split preserves the temporal sequence, so guaranteeing consistent time flow training and testing of the model. In predictive models, this is especially important since it avoids the use of future data to forecast the past, a practice that might introduce bias and cause an overestimation of the model's performance.

K-Fold Cross Validation is inadequate for time series because observations are linked in time and future events can not affect or be linked to past events [97]. As shown in Figure 4.6, K-Fold shuffles and splits the data into estimated folds. This breaks this temporal dependence and can cause wrong or unrealistic biases because information from the future can affect the model's training in a bad way. Instead, Time Series Split makes more sense because it divides the data into multiple runs. Each run moves the test set through time, while each run adds to the training set. Each iteration adds new data to the training set and tests the model in the next section. This way, the model can be tested over different periods. This approach allows the model to be tested over different periods, ensuring that its performance in future predictions is more reliable and robust, especially when it is crucial to accurately understand historical patterns and trends [94].

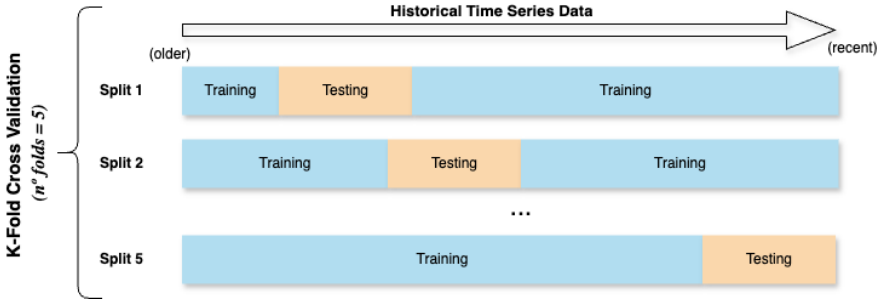

Figure 4.6: Illustration of the K-Fold Cross Validation Mechanism.

Figure 4.7 provides a visual representation of the process of this technique, where the data is partitioned into several folds, each representing a distinct period. What differentiates this division mechanism from K-Fold Cross-Validation is that after the validation fold, the space that is not used in Time Series Split is reused for training, thus becoming another train fold. In Time Series Split, this reuse cannot occur because it is necessary to maintain the integrity of the data, and there cannot be any randomness. In the end, the final performance is calculated by averaging the results of all folds, that is, the average of the evaluation metrics, which will be discussed in detail in Section 5.3. This process is designed to maximize data dependencies and capture patterns or trends to achieve more accurate results.

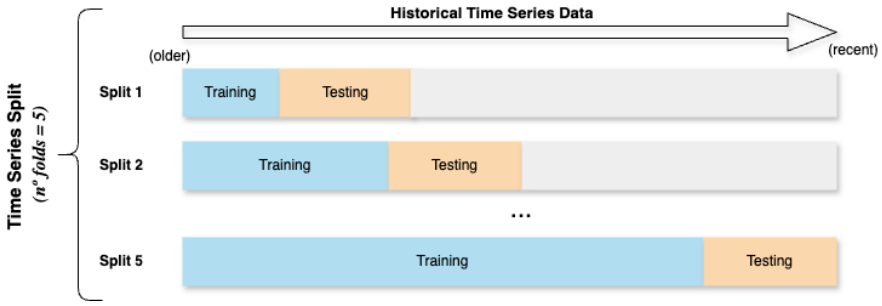

Figure 4.7: Illustration of the Time Series Split Mechanism.

4.1.3 Model Selection

Focusing on the many techniques that can be used based on the type of data and the purpose of the analysis, several ways to create predictive models will be covered in this subsection. With their special qualities, these techniques include conventional statistical models, neural networks, and decision tree-based models.

First, the focus will be on statistical models, which have been a mainstay of time series analysis and other forecasting applications. Since ARIMA can simulate both the autoregressive and moving average components of a time series, it is one of the most frequently employed of these. SARIMA (Seasonal ARIMA) is often used in areas such as banking and retail for data with seasonal trends because it offers an extension of ARIMA to manage seasonality.

Next, the neural network-based models will be discussed, and they will shine at gleaning intricate patterns from large amounts of data. LSTM can learn and remember long-term dependencies in sequential data, which is something that many other models struggle to do, it makes it very useful in time series forecasting. In multivariate forecasting situations, where multiple connected time series must be modeled simultaneously, LSTM is very successful.

On the other hand, decision tree-based models such as Gradient Boosting offer a distinct strategy by combining substantial predictive power with interpretive simplicity. Particularly good at modeling complicated interactions between input factors and the output variable, these models provide a sequence of binary choices that lead to a final prediction. Gradient Boosting allows methods to be modified in multivariate settings to capture interactions between many predictor variables, thereby improving forecast accuracy.

Finally, hybrid methods will be discussed, i.e., combinations of the strengths of many types of models aimed at increasing forecast accuracy. These methods provide exceptional flexibility and accuracy when combining statistical models with neural networks or when combining decision tree-based approaches with other techniques, thus allowing models to better fit the complexity of real data. Hybrid methods can be especially effective in multivariate forecasting by combining the robustness of methods such as Gradient Boosting with the ability of models such as LSTM to reflect both the complexity and interdependence of variables.

Each of these techniques will be discussed in the following sections, highlighting their benefits and drawbacks, as well as their internal dynamics, in multivariate time series. In Chapter 5, all these algorithms will be tested and then evaluated based on the performance metrics described in section 4.1.5.

4.1.3.1 Auto-Regressive Integrated Moving Average (ARIMA)

Time series forecasting models can be created using autoregressive (AR) methodologies. Box et al. (2015) have described ARIMA models as a common method for analyzing historical patterns in time series data and predicting future values [98]. This algorithm is frequently implemented to predict univariate time series data. The Seasonal Auto-Regressive Integrated Moving Average (SARIMA) can be incorporated to improve ARIMA models when the impact of seasonality in time series data is taken into account. One of the drawbacks of AR models is their propensity to experience a delay in predicting long-term dependencies and their inability to manage large quantities of training data as a result of the restrictive assumptions they place on time series [99].

The time series must exhibit stability prior to employing the ARIMA method. The ARIMA model asserts that the mean, variance, and autocovariance of the time series are invariant under the assumption of stationarity.

A series plot and statistical tests like the ADF test are used to determine if a time series is stationary. A unit root in a time series indicates non-stationarity, which the ADF test detects. If the ADF test rejects the null hypothesis of no unit root, the series stands still. Making sure changes like differencing are needed before using the ARIMA model is crucial. Detailed ADF test details are available in the Data Preprocessing section 4.1.1.2.

The ARIMA model is determined mostly by p, d, and q parameters. Analyzing the major lags in the PACF plot helps one to ascertain the sequence of the AR model, denoted by p. The variable d, usually found using ADF testing, indicates the necessary differencing for series stability. Counting significant lags in the ACF plot helps determine the value of the Moving Average (MA) term, denoted by q. Then, analyzing the effect of past values and errors on future forecasts helps one to decide the ARIMA model's structure.

The method of modeling time series using the SARIMA model starts with the identification of seasonality in the series. This may be accomplished by the use of visual analysis or ACF to find recurrent patterns that lead to the identification of a seasonal period s. After the period s has been determined, the ACF and PACF tests are carried out. These tests are essential for defining the seasonal (P, D, Q) as well as the non-seasonal (p, d, q) model parameters. The seasonal parameters are derived by observing specific lags corresponding to the period s in the ACF and PACF plots: P is determined by significant lags in the seasonal PACF, Q by lags in the seasonal ACF, and D by the need for seasonal differencing to achieve stationarity, verified through unit root tests like ADF. The values of p, d, and q are discovered similarly, with the exception that shorter cycles and regular changes are taken into consideration.

The process is shown in Figure 4.8, which gives a visual representation of the ARIMA/SARIMA modeling workflow.

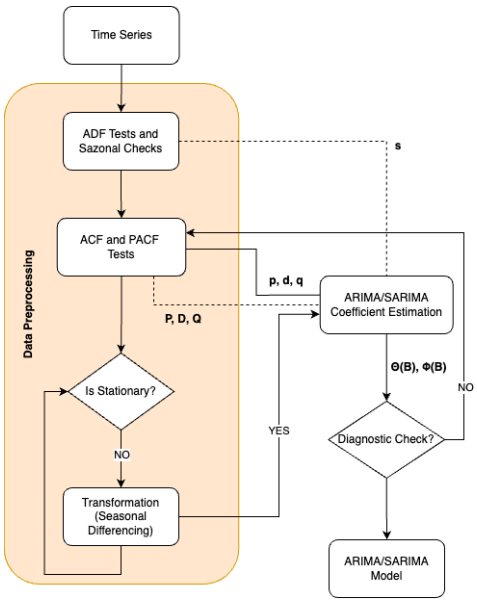

Figure 4.8: Workflow of the ARIMA/SARIMA Modeling Process.

4.1.3.2 Long Short Term Memory (LSTM)

Long Short-Term Memory Networks (LSTMs) are a particular form of RNN structure that addresses the issues of disappearing and inflating gradients often seen in regular RNNs. LSTMs are specifically built to excel at capturing long-term dependencies in sequential data. The long-term dependencies that these networks learn are enabled by the memory cells that make up the LSTMs. The state of an LSTM cell is updated at each time step, and vanishing gradients are prevented by the constant error flow caused by the cell state [57].

Three gates, namely the input gate, the output gate, and the forget gate regulate the flow of information into and out of the cell. This may be observed in Figure 4.9. The forget gate examines the significance of information from the previous cell state and decides what to retain and what to disregard. On another hand, the input gate regulates the passage of information into the cell state. The output gate is primarily responsible for determining and controlling the outputs. The collaboration of these gates enables the network to ascertain which input to retain or discard, resulting in a highly efficient system for processing and generating intricate patterns.

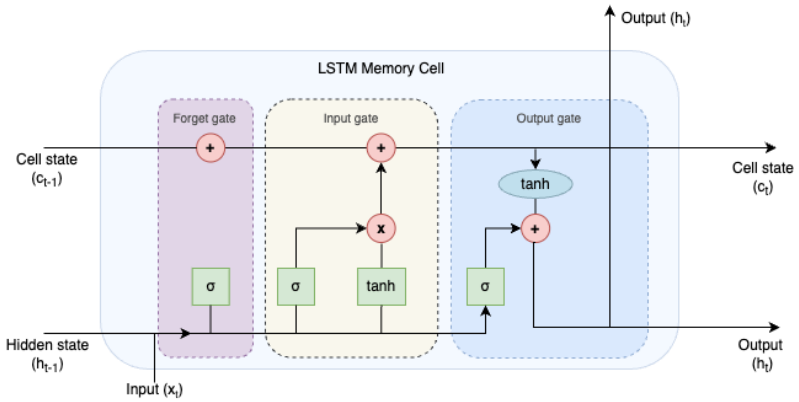

Figure 4.9: LSTM Architecture.

A bidirectional LSTM (BDLSM) layer is exploited to capture spatial features and bidirectional temporal dependencies from historical data. This allows the model to capture temporal dependencies from both the past and the future context. The BDLSTM is composed of two LSTMs: one that handles the sequence in its original temporal order and another that handles it in the opposite order. The integration of these two layers enables the model to get a more comprehensive understanding of the whole context of the time series. Despite its higher computational intensity and complexity in training, the BDLSTM can greatly enhance accuracy in tasks that require consideration of context from both ends of the sequence. The structure of the BDLSTM is seen in Figure 4.10.

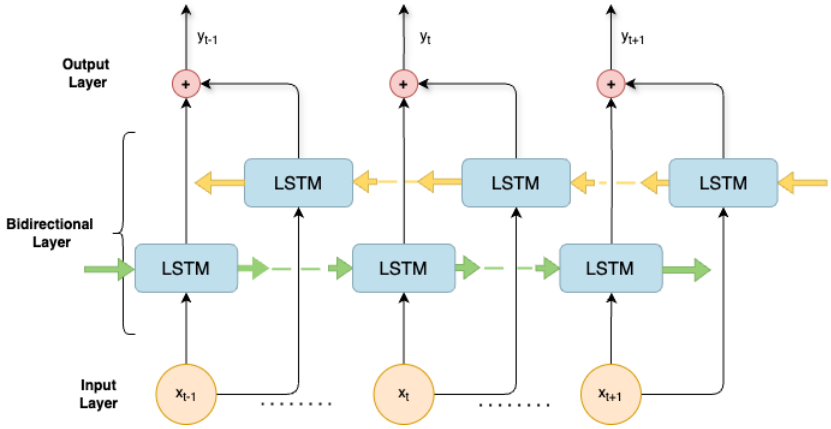

Figure 4.10: Bidirectional LSTM Architecture.

4.1.3.3 Gated Recurrent Units (GRU)

The GRU is a type of Recurrent Neural Networks RNNs that was created to help with the problem of learning how long-term relationships in sequential data work. The GRU was created as an easier version of the well-known Long Short-Term Memory LSTM. It keeps its high performance while using fewer settings, which makes it more efficient for computing. The GRU architecture transforms the cell state and hidden state into a unified hidden state vector, denoted as h_t, as seen in Figure 4.11. This vector changes at each time step and the

GRU can get temporal information without a different cell state because the structure is so simple.

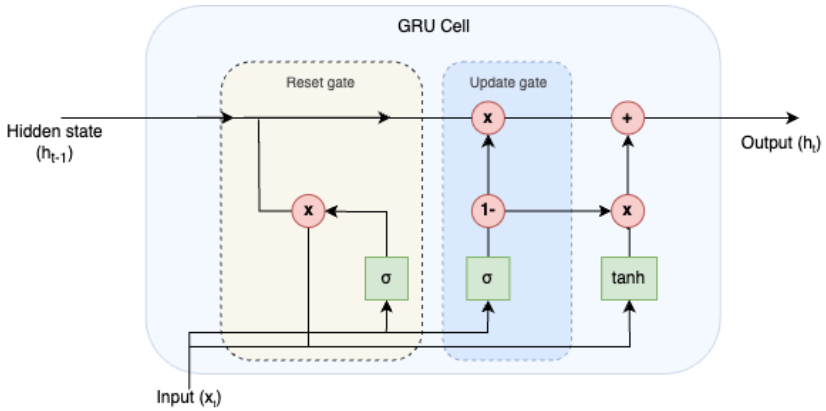

Figure 4.11: GRU Architecture.

It is noted that the LSTM employs 3 distinct gate networks while the GRU reduces the gate networks to two and these changes make training easier and speed up the inference process. They are the update gate (z_t) and the reset gate (r_t). They control the flow of information in the GRU. The reset gate controls how much of the previous hidden state should be kept, and the update gate controls how much of the previous hidden state should be used to create the new state.

GRU algorithm can quickly learn about long-term relationships with this set of gates because they can choose which information to ignore or update based on their needs. This algorithm is often used in natural language processing, time series analysis, and other problems with sequential data because its design is simpler and more efficient [100]. It is a strong alternative to LSTMs in many situations.

4.1.3.4 Gradient Boosting

Boosting is a highly effective technique in the tree-based ensemble method. It involves storing the labels and weights of leaf nodes, which helps interpret predictions. Gradient Boosting, proposed by Chen and Guestrin, is a pragmatic technique [101] that is widely regarded as one of the preferred algorithms in machine learning. A robust and precise model can be achieved by amalgamating the feeble models during the Gradient Boosting. This technique relies on the residuals from the previous iteration to determine classification. The impact of each feature is assessed sequentially until a desired level of accuracy is attained. The residuals are computed as the discrepancy between the observed values and the estimated values and are determined by minimizing a loss function (L(ϕ)) through the use of gradient descent. Gradient Boosting trains a subsequent model on the residuals of the preceding model, aiming to rectify the inaccuracies made by the preceding model, as shown in Figure 4.12. The process is iterated multiple times (based on the user-specified number of iterations), resulting in a final model that is the aggregate of all fitted models, weighted accordingly.

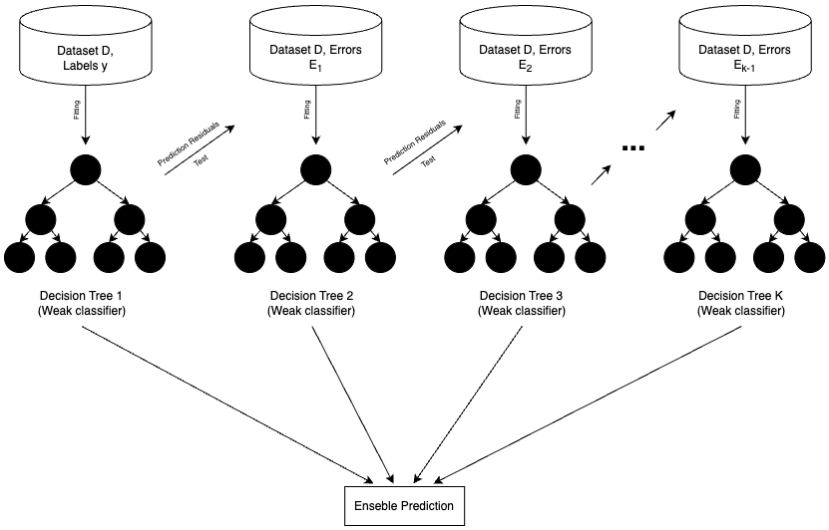

Figure 4.12: The Architecture of Gradient Boosting Decision Tree. Based on (Deng, 2021) [2].

The Gradient Boosting approach has proved very useful in time series analysis. Time series data often has complicated temporal interdependencies, seasonality, and patterns that make modeling difficult. Iteratively improving predictions using past mistakes is enough for Gradient Boosting to capture complicated patterns. This adaptability enables the management of non-linear correlations and anomalies in time series data. The capacity to include delayed elements and external inputs enhances work forecasting. Gradient Boosting is used in time series forecasting to improve accuracy in finance, energy, and supply chain management. Has the capacity to capture intricate relationships within data, making it particularly useful in scenarios where traditional time series models like ARIMA may fall short [102].

4.1.4 Mechanisms to Reduce Overfitting

Overfitting is a common issue in deep learning when a model gets too tailored to the training data, resulting in worse performance on new, unknown data. To mitigate this issue, many methodologies have been devised to augment the ability to apply learned knowledge in a broader context while still ensuring efficient acquisition of new information. These techniques include adjusting the training process in real-time based on how well the model performed on a distinct validation set. Methods such as Early Stopping may mitigate overfitting by terminating training when performance enhancements end, thereby circumventing superfluous over-training. Similarly, methods like ReduceLROnPlateau may automatically decrease the learning rate when performance reaches a plateau, enabling more accurate modifications to the weights.

In the following sections, we will thoroughly examine these and other fundamental processes, investigating how each might be efficiently used to mitigate overfitting and enhance the generalization of the model. By comprehending and using these tactics, it is feasible to construct models that not only exhibit high performance on the training data but also demonstrate good generalization to novel, unobserved data.

4.1.4.1 Early Stopping

Early Stopping in machine learning model training, usually neural networks, prevents overfitting and ensures model performance on new data. The idea is to monitor how the model performs on a validation set during training on training data. Knowledge improves model performance on training and validation datasets. However, beyond a certain threshold, the model may overfit, becoming too good at predicting outcomes using training data but too bad at predicting new data. Early stopping stops training when model accuracy on the validation set stops improving [103], [104].

This technique is implemented in practice by choosing a ”patience” parameter that determines how many epochs (full runs through the training dataset) the model may train for without seeing an increase in the validation performance before halting the training. For example, the training process is stopped if the performance of the model on the validation set is constant for five epochs in a row, with the patience set to five. The model is then put back to how it was at the period when its validation performance was at its best. This approach prevents overfitting and saves computational resources by removing unnecessary training beyond the point of peak performance. A well-performing model on training and validation data is the consequence of striking a balance between generalization and model complexity, which is facilitated by early stopping.

4.1.4.2 Reduce Learning Rate (Reduce LR)

Autonomous learning rate adjustment during neural network training using the ”ReduceLROnPlateau” technique is one of the most critical hyperparameters in model optimization. This approach is mainly based on how well the model does on a validation set when it is being trained. The technique decreases the learning rate when growth in performance, such as an increase in accuracy or a decrease in loss, reaches a plateau. The objective is for the model to avoid getting stuck in local minima or plateaus, which are situations where the progress in minimizing the loss function slows down or stops. To do this, the parameters of the model are further refined at a low learning rate. The idea is that by continuously fine-tuning its weights more precisely at a low learning rate, the model will be able to avoid local minima or plateaus, which happen when progress in minimizing the loss function slows down or stops [105].

In practice, ”ReduceLROnPlateau” is configured as a callback in frameworks such as TensorFlow [106] or Keras [107]. The model monitors a particular metric, such as validation loss, during the training process. The learning rate is automatically reduced by a predetermined factor, such as 0.1 or 0.5, if this metric fails to demonstrate substantial improvement after a specified number of consecutive epochs (the ”patience” parameter). This implies that the technique permits the model to continue converging at a slower and more controlled tempo if it ceases to improve by adjusting the learning speed. This automatic adjustment can eliminate the necessity of manually reconfiguring the learning rate, thereby saving time and effort during the training process.

This mechanism works effectively in reality when the model performs well quickly but still requires fine-tuning to get the highest accuracy. "ReduceLROnPlateau" smoothes the learning trajectory in challenging tasks which include image recognition and natural language processing with vast parameter spaces and local minima. This tactic is easy to implement in code, and by keeping the model from plateauing and letting it explore the solution space, it may enhance training efficiency and outcomes.

4.1.5 Metrics to Evaluate Model Performance

Performance evaluation is essential for guaranteeing the reliability and accuracy of the predictions produced by time series forecasting models. Metrics that are appropriately chosen facilitate the quantification of accuracy and the identification of the model's inherent limitations. Because of their simplicity and capacity to offer a clear understanding of the magnitude of errors, metrics like Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) are frequently employed, as per Hyndman and Athanasopoulos (2018) [94]. Nevertheless, as Makridakis et al. (2020) have emphasized, the selection of metrics must be meticulously evaluated, as factors such as seasonality and trend can have a substantial impact on the models' perceived performance [108]. Consequently, a more comprehensive analysis is necessary to prevent the drawing of premature conclusions.

In this analysis, the selected metrics to comprehensively evaluate the performance of the time series forecasting models were:

- **Coefficient of Determination (R^2)**;
- **Mean Absolute Error (MAE)**;
- **Mean Square Error (MSE)**;
- **Root Mean Square Error (RMSE)**;
- **Symmetric Mean Absolute Percentage Error (sMAPE)**.

4.1.5.1 Coefficient of Determination (R^2)

The Coefficient of Determination, commonly known as R-squared (R^2), is a statistical measure of the capacity of a model to explain dependent variable variance depending on independent variables. Depending on how well the model captures the relationship between the variables, R^2 values lie between -∞ and 1. As shown in Equation 4.1, in the numerator, the SSE consists of the sum of squared errors, meaning the difference between the observed values y_i and the predicted values $\hat{y}_i$ by the model. In the denominator, the total sum of squares (SST) is represented, which is the variation of the values y_i relative to the mean $\bar{y}$. The R^2 is calculated by subtracting the ratio between SSE and SST from 1, i.e., $R^2 = 1 - \frac{\text{SSE}}{\text{SST}}$.

$$R^2 = 1 - \frac{SSE}{SST} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (4.1)$$

An ideal predictive performance is indicated when R^2 equals one since it indicates that the model totally explains all the variation in the dependent variable. R^2 values close to 1 indicate a high degree of accuracy in the dependent variable prediction of the model. On the other hand, negative R^2 values suggest that the model is less efficient than one that merely forecasts the mean of the dependent variable. This happens when the model generates basically random predictions by failing to find any obvious trends or patterns in the data [109].

4.1.5.2 Mean Absolute Error

The Mean Absolute Error (MAE) represents the average magnitude of the absolute difference between ground truth and predicted values [110]. It is calculated as the mean of the absolute differences between the actual values and the values predicted by the model. The formula for MAE is

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \quad (4.2)$$

where y_i are the observed values, $\hat{y}_i$ are the predicted values and n is the number of observations.

The MAE is calculated by finding the absolute difference between each observed value and the predicted value, summing all these differences, and then dividing by the total number of data points, as Equation 4.2 demonstrates. This metric provides the mean absolute error, a measure of the model prediction accuracy. Because the MAE is stated in the same units as the data and gives a straightforward measure of the average error in absolute terms, it is simple to comprehend. Since it does not square the errors, it is resistant to outliers and helpful when a clear and concise representation of the model average prediction error is required. The performance of the model improves with a lower MAE.

4.1.5.3 Mean Square Error

Mean Squared Error MSE is one of the most frequently employed estimations of prediction quality [111]. Using the Euclidean distance, it illustrates the extent to which predictions deviate from the actual values. Whether during training, cross-validation, or post-deployment monitoring, it is exceedingly advantageous to have a single numerical value to evaluate model performance in machine learning. One of the most frequently employed metrics for this purpose is the MSE [110].

Mean Squared Error can be expressed as

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \quad (4.3)$$

where y_i represents the actual or observed value, $\hat{y}_i$ represents the predicted or estimated value, and n is the number of observations.

The residual (difference between the prediction and the truth) is evaluated for each data point to calculate MSE. Subsequently, the residuals are squared and averaged. For this reason, MSE is frequently implemented in supervised learning applications, as it requires accurate measurements at each prediction data point. Correlation coefficients are directly associated with the utilization of standardized observations and forecasts as MSE inputs. For instance, if the correlation coefficient is 1, the MSE will be zero, as all of the points are on the regression line and, as a result, there are no errors.

4.1.5.4 Root Mean Square Error

The Root Mean Squared Error (RMSE) is a commonly used variation of MSE, providing a more interpretable measure by maintaining the same unit of measurement as the predicted variable [111]. Similar to MSE, it assesses how far predictions deviate from the actual values using Euclidean distance. The RMSE is especially useful for interpreting prediction errors in machine learning models during training, validation, and post-deployment phases, where it offers a single summary statistic to evaluate model performance [110].

The RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \quad (4.4)$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of observations.

The RMSE provides a more interpretable score by taking the square root of the MSE, ensuring the error metric is in the same unit as the target variable. This characteristic makes it easier to comprehend in practical applications. To compute RMSE, the residuals are first squared and averaged, and then the square root is taken, penalizing larger errors similarly to MSE. The RMSE is widely used in supervised learning problems and closely relates to correlation coefficients: if the correlation coefficient is 1, the RMSE equals zero, indicating perfect prediction alignment with no error.

4.1.5.5 Symmetric Mean Absolute Percentage Error

Measuring the percentage difference between actual and anticipated values helps one to assess the accuracy of forecasting models using the Symmetric Mean Absolute Percentage difference sMAPE. Unlike conventional measures such as MAPE, sMAPE provides a symmetric approach wherein it equally penalizes overestimations and underestimations. Rather than only the actual value, this is accomplished by averaging the absolute difference between observed and anticipated values [112].

Symmetric Mean Absolute Percentage Error can be expressed as shown in Equation 4.5:

$$\text{sMAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{\frac{|y_i|+|\hat{y}_i|}{2}} \times 100\% \quad (4.5)$$

where y_i represents the actual or observed value, $\hat{y}_i$ represents the predicted or estimated value, and n is the number of observations.
Symmetric Mean Absolute Percentage Error (sMAPE) is hence especially helpful when working with data with values close to zero, when MAPE may provide falsely high or indeterminate faults. For evaluating model performance across a broad spectrum of data circumstances, sMAPE is, therefore, a more balanced and reliable metric.

When environmental data is utilized for decision-making, such as air quality control, sMAPE can measure forecast model reliability. This helps policymakers trust their requirements and make educated environmental and public health decisions. sMAPE usually gives an average percentage of a difference between real and projected environmental variables. This makes it easier to evaluate the prediction model and ensures that it can predict rapid and unexpected environmental changes.

4.2 Alerts and Warning System

An alert and warning system is intended to notify users or administrators of critical or abnormal conditions, such as exceeding safe limits or detecting errors. Timely notifications are a key feature of this system, enabling swift action to avoid harm, guarantee safety, and keep operations stable. Without such a system, problems may go undetected and cause expensive or even harmful consequences.

With the use of Structured Query Language (SQL), MySQL is a high-performance Relational Database Management System (RDBMS) that facilitates tabular data manipulation and management. In addition to guaranteeing transaction integrity by upholding the Atomicity, Consistency, Isolation, Durability (ACID) principles, it offers a number of sophisticated capabilities, including indexing, joins, views, and stored procedures. Especially in large-scale systems, these characteristics enable effective data management and the optimization of complex searches. The multi-threaded engine supported by the MySQL design allows for high concurrency and strong data security, making it an effective tool for managing databases in dynamic contexts.

In MySQL, named database objects known as triggers are intimately linked to a particular table and are triggered automatically upon the occurrence of a specified data manipulation event. Row-level actions like INSERT, UPDATE, and DELETE are examples of these events, which are referred to as trigger events. When data in a table is modified, triggers immediately carry out predetermined logic, guaranteeing data integrity and consistency. This is crucial for enforcing sophisticated business logic directly at the database level. An UPDATE trigger, on the other hand, can make sure that any changes made to a row are correctly cascaded across adjacent tables. For instance, an INSERT trigger can verify or alter data before it is put to the table. It is possible to specify when a trigger will operate in relation to the triggering event. Prior to the data being committed, a BEFORE trigger can be used to do operations like data validation, rule enforcement, or change. This trigger is executed before the actual data alteration happens. An AFTER trigger, on the other hand, is used for actions like logging changes, updating audit tables, or starting follow-up procedures that rely on the final state of the data. It is triggered after the data has been modified. When dealing with an INSERT event, for example, a trigger can be set to fire either before or after each entry is successfully added to the table.

Triggers play an important role in the RuraLTHINGS system since when data is inserted into the system (database), another action can be triggered. In this case, it would be beneficial if, as soon as the data is inserted, the values entered were checked to see if they are within safe limits. If the values are not within safe limits, the records are also inserted into another table (GasesAlerts), which is being monitored by a script that analyzes all the inserted data. There is another trigger that uses the same data insertion action in the 'SensorData' table to update the DateTime of the last record sent by a device. The latter is used to analyze the status of the devices and see if any are disconnected from the network due to external factors.

This script monitors the table every 5 seconds for new abnormal data. In order to be able to send an HTTPS request to the endpoint that handles sending notifications to the user, it sends an HTTP request to an API (using the requests library) for each alert, which provides comprehensive information about the alert, including gas categories, exceedances, and device details. The alert is marked as processed in the database to avoid reprocessing if the request is successful. This process ensures continuous real-time monitoring and alert management.

In system development, the use of Docker and phpMyAdmin assists in the process of viewing tables and introducing triggers. Docker allows for the rapid setup of isolated and consistent environments, ensuring that MySQL runs seamlessly on different systems without the need for complex installations [113]. PhpMyAdmin complements this by providing an intuitive web interface for database management, allowing easy viewing, manipulation, and maintenance of databases without having to rely solely on command line operations [114]. Together, they simplify the development and management process, saving time and reducing complexity.

4.3 Conclusion

Regarding pollutant gases and important environmental variables, the RuraLTHINGS platform, as suggested in this Chapter, aims to provide a complete solution for monitoring environmental conditions in homes. Leveraging IoT technologies, sensor networks, and machine learning algorithms, the platform enables visualization of real-time and historical data, equipping users with insights into indoor air quality and environmental factors.

At this point, the theoretical aspects of the platform, including data collection, preprocessing, and ML-based pollution concentration prediction, have been described. The possibilities of these models to support accurate predictions and aid decision-making are discussed, as well as their role in identifying important events and thus preventing health risks.

As hypothesized, the warning system of the platform will be very important in alerting users to dangerous conditions, enabling rapid intervention. Furthermore, a key part of the platform, the DSS, aims to provide actionable insights based on real-time and projected data, thereby improving the ability to respond appropriately to environmental risks.

With a focus on its ability to handle real-world data, prediction accuracy, and usability of the warning system, practical testing will be conducted in the next phases to validate the functionality and effectiveness of the decision support engine and the platform as a whole. Although such useful applications have not yet been realized, the theoretical framework presented in this chapter lays a solid foundation for future testing and improvements. With practical experimentation being the next important step to ensure its success in improving indoor air quality and protecting health using IoT and AI solutions, the RuraLTHINGS platform presents a promising theoretical approach to environmental monitoring and health risk mitigation.

Chapter 5

System Evaluation - Viability and Validation

5.1 Introduction

The system proposed in this document was validated in the following sections. The evaluation of the digital platform was conducted first, followed by that of the decision support system based on prediction.

5.2 Testbed Environment

Regarding the decision support system, due to the lack of sufficient data to test the decision support system, data from a dataset obtained from GitHub, as previously mentioned in this document, were used. Several tests were carried out to determine which model would best adapt to temporal data, which follows a time series. All tests were conducted on the machine described in Table 5.1.

Specification	Details
CPU	11-core (5 performance cores, 6 efficiency cores)
Graphics Processing Unit (GPU)	14-core
Neural Engine	16-core
Memory Bandwidth	150 GB/s
Unified Memory	18 GB
Storage	512 GB SSD

Table 5.1: PC Specifications.

The digital platform was developed using smartphone emulators (Android/iOS). The API, through its documentation, facilitated the use of endpoints by digital systems, which were then integrated into the platform to access the data stored in the database.

5.3 Proposed System Evaluation

To obtain a final model that best fits temporal data, which consists of time series, it is necessary to perform numerous tests among different models, always taking into account the values obtained from the performance metrics defined in 4.1.5.

On the other hand, the digital platform will also be evaluated as a whole, particularly the implementation of functionalities in the layouts, their correspondence with the use cases, and other software engineering diagrams.

5.3.1 Decision Support Mechanism Evaluation

The choice of certain parameters can be crucial for the accuracy of the models and how different approaches can be combined to achieve better results. The evaluation follows the workflow presented in Figure 5.1.

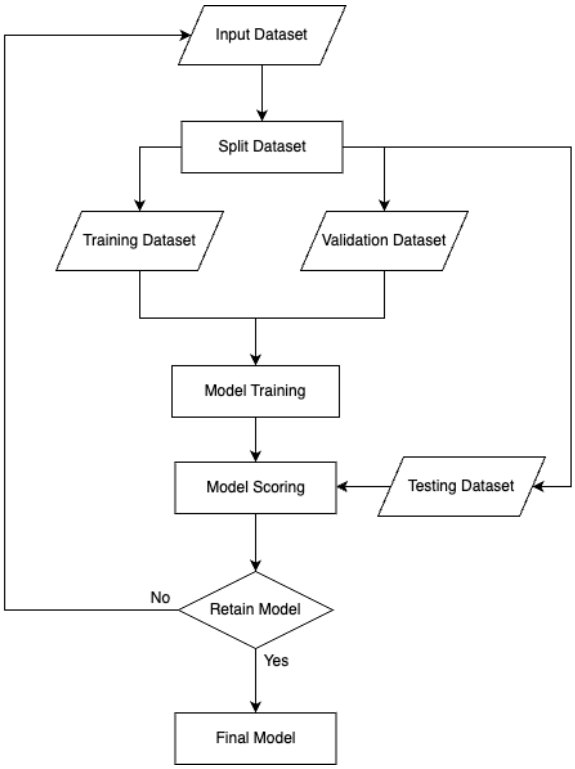

Figure 5.1: Evaluating Workflow Adapted from (Aziz,2020) [3].

5.3.1.1 Statistical Model - ARIMA

The ARIMA statistical model, as mentioned in section 4.1.3.1, requires the specification of some parameters used by the model to make predictions based on the provided data. The selection of these values can be obtained by interpreting the ACF and PACF plots, which can be found in appendix A.1.0.1. The properties of time series are fundamental, such as the presence of autocorrelation and seasonal or cyclical patterns, and are crucial for determining the appropriate number of lags in autoregressive models.

According to the values shown in Figures (Appendix A.1.0.1), it can be concluded that the values that appear to be ideal for the test dataset are those defined in Table 5.2.

Variable	ARIMA Model (p, d, q)
CO_2	ARIMA(2,0,2)
Humidity	ARIMA(2,1,2)
Radon	ARIMA(2,0,1)
Temperature	ARIMA(2,0,1)

Table 5.2: ARIMA Models for Different Variables.

The p-values were mainly assigned due to the CO_2 variable, in which the PACF graphs show significant peaks right at the first lags and seem to cut off after the second lag, suggesting that **p = 2** would be an appropriate value. For the remaining variables, although it is not as evident, this value seems to be a good choice for modeling these time series.

According to the ADF results and the ACF graphs, which allow us to obtain the value of d from the number of times the data need to become stationary, the value **d = 0** seems to be the most suitable for each of the variables. Although in the case of Humidity, which presents an ADF value quite close to the threshold, we can already consider **d = 1**, as it suggests a slight differentiation that may be necessary to ensure the complete stationarity of the time series.

On the other hand, the ACF graphs do not rule out intermediate correlations, and for this reason, it is possible to observe the decay of autocorrelation over the lags. The value of q, according to the interpretation of the graphs, can be considered **q = 1** or **q = 2**, as in all graphs, it is common to see a slow decay over the lags, suggesting a moving average process.

Another way to obtain these hyperparameters is through a mechanism called AutoARIMA, which automates the process of fitting ARIMA models, along with the use of other criteria, namely Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), allowing the selection of the model that best fits the data. This automated technique presents its advantages, especially in terms of automation and process flexibility. However, it is quite computationally expensive until the ideal model is found. In this technique, stopping criteria should be mentioned so that the model finds the optimal model until this criterion is reached.

```
model = auto_arima(
            residuals[var],
            start_p=1, start_q=1,
            test='adf',
            max_p=5, max_q=5,
            d=None,
            m=12,
            seasonal=True,
            start_P=0, D=1, start_Q=0,
            max_P=5, max_D=5, max_Q=5,
            trace=True,
            error_action='ignore',
            suppress_warnings=True,
            stepwise=True,
            n_jobs=1)
```

The result of this execution is described in Table 5.3, showing various possible values, with the model that presented the best results being **ARIMA (2,0,1)**.

Model	MSE	MAE
ARIMA(2,0,1)	0.007494	0.033231
ARIMA(3,0,1)	0.008218	0.034296
ARIMA(3,0,2)	0.008341	0.034688

Table 5.3: Results by AutoARIMA.

After training the model using the dataset, the graphical results are shown in Figure 5.2.

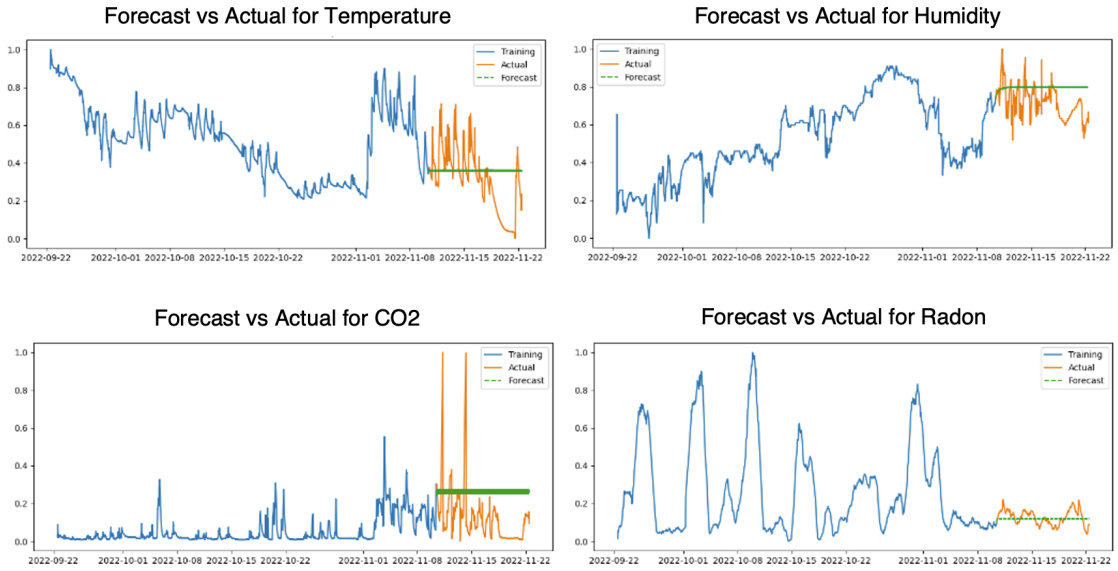

Figure 5.2: ARIMA Prediction vs Forecast Values (ARIMA(2,0,1)).

As can be seen from the graphs in Figure 5.2, the statistical model does not show good results in prediction, although the performance metric values are relatively good. One possible reason for this is that it represents the average error over the dataset without capturing the error distribution over time. The presence of peaks and abrupt drops in the data may also be a factor, as it prevents the model from accurately predicting in these cases, which justifies the smoothness in the predictions without significant highs and lows. Temperature and CO_2 are examples where the model did not follow the rapid changes in the real data.

The fact that the predictions are not very close to the real values, despite the relatively low Mean Squared Error (MSE) and MAE values, indicates that the model is capturing the general patterns of the time series but failing to predict the nuances or more abrupt and seasonal variations. The SARIMA model, which includes seasonality in the data, was also tested, but the results were the same as those of the ARIMA model, suggesting that seasonality was not detected using this approach.

5.3.1.2 Artificial Neuronal Networks - RNAs

In order to find a model that best suited the requirements, other mechanisms were used to better detect volatile data or include non-linear components, such as neural networks. These are suitable for time series and sequential problems but can suffer from long-term issues due to the vanishing gradient problem.

The seq_length, which consists of the sequence length, was set to a value of 100, which, in the context of time series, represents the time window used as input for the model. For example, assuming the data is sent every 5 minutes, a time window of 100 corresponds to 100 records, which is approximately 8 hours and 20 minutes. Again, it is crucial for seq_length to be well adapted to the data so that the model knows how many data points to consider when making predictions. The value should be adjusted to ensure it is not too short, as it may fail to capture dependencies, nor too long, as it may unnecessarily increase complexity.

On the other hand, the batch_size is the number of events processed simultaneously by the model during training. It allows for faster training by updating the weights simultaneously. For each batch, the model adjusts its parameters based on the average of the errors made, rather than predicting the entire batch dataset. A large batch size generally results in more precise weight updates but may require more memory. Conversely, a smaller batch size tends to converge faster and improve the model's generalization, but the weight updates are usually noisier.

In order to find the most appropriate batch size, experiments were conducted with various batch sizes, and the performance of the model was evaluated using several metrics, including MSE, Root Mean Square Error (RMSE), MAE, and the coefficient of determination (R^2), as shown in Table 5.4. In these results, the data provided to the model was not normalized.

Batch Size	**MSE**	**RMSE**	**MAE**	R^2
10	2676.57	51.74	7.23	0.985
20	538.36	23.20	2.54	0.978
32	4070.78	63.80	8.28	0.929

Table 5.4: Performance Metrics for Different Batch Sizes Using the RNN Model.

According to the results obtained from the metrics mentioned in section 4.1.5, we can observe an improvement in predictions when the batch_size is set to 20, based on the MSE, RMSE, and MAE metrics. On the other hand, it is noted that the model does not generalize as well as the model tested with batch_size equal to 10, which is expected since the model learns more general patterns by seeing smaller subsets of data at each iteration. Although it is more accurate, the other metrics (MSE, RMSE, and MAE) are not the best, indicating that with a batch_size of 20, the predictions are closer to the real values but do not generalize very well.

To improve this factor, a model was tested in which the number of neurons in the second RNN layer was doubled from 32 to 64 while keeping the batch_size at 20. The results can be seen in Table 5.5, where it is evident that the model with more neurons in the second Dense layer demonstrated better results in terms of MSE, RMSE, and R^2. In these results, the data provided to the model was not normalized.

Nº Neurons	**MSE**	**RMSE**	**MAE**	R^2
32	538.36	23.20	2.54	0.978
64	199.91	14.14	4.69	0.992

Table 5.5: Performance Metrics for Different Neurons on Second Dense Layer.

5.3.1.3 Artificial Neuronal Networks - LSTM

In neural networks with many layers or when working with long temporal data, such as LSTMs or GRUs, the gradient that propagates through the layers can vanish or explode, making training difficult. This requires the use of memory mechanisms (as in LSTMs and GRUs) to mitigate this problem. In time series problems, it is essential to address the model's ability to capture both long-term and short-term patterns. Models based on LSTM layers and GRU layers were designed for this purpose.

LSTMs are the most commonly used architecture for capturing trends and patterns in temporal data, especially in historical data, as LSTMs retain the "memory" of the network's state over long time windows. It is also common to use bidirectional layers in conjunction with the LSTMs architecture [59], as they improve the model's ability to learn patterns in data sequences.

In all results present in the next tests carried out, the model input data were normalized, that is, they are in a range between 0 and 1, in order to help the model adjust to the training data, resulting in better performance.

The learning_rate for all tests represented in Table 5.6 is 0.001, as well as the seq_length, which corresponds to 100. The tests were conducted with an 80/20 split (80% of the dataset was used for training, and 20% for testing).

Model	MAE	sMAPE	RMSE	R^2
LSTM (100) + Dropout (0.2) + **LSTM** (100) + Dropout (0.2) + Dense	0.1203	12.96%	0.2106	0.9725
Bi-LSTM (100) + Dropout (0.2) + **Bi-LSTM** (100) + Dropout (0.2) + Dense	0.0544	5.63%	0.085525	0.9875
Bi-LSTM (100) + LayerNormalization + **Dropout** (0.2) + Attention + Concatenate (drop_out, attention_out) + **Bi-LSTM** (100) + **Dropout** (0.2) + Dense (activation = 'sigmoid')	0.1404	17.46%	0.1797	0.9225
Bi-LSTM (100) + LayerNormalization + **Dropout** (0.3) + Attention + Concatenate (drop_out, attention_out) + **Bi-LSTM** (100) + **Dropout** (0.3) + Dense (activation = 'sigmoid')	0.2028	28.97%	0.235075	0.7575

Table 5.6: Model Performance Metrics, with batch_size = 32.

The model that showed the best results was the one that included a Bidirectional LSTM layer instead of a unidirectional LSTM, assisted by Dropout (0.2), a mechanism used to minimize overfitting. The 20% dropout rate corresponds to the proportion of neurons in the previous layer that will be deactivated during each training step. This layer is added twice: first, to reduce overfitting of the extracted features, and then to continue regularizing to prevent overfitting. The dropout rate was also adjusted to a slightly higher value, which directly affects the model's generalization capacity and resistance to overfitting, as a larger proportion of neurons is ignored in each iteration. This increase forces the model to learn more robust

representations, making it harder for the model to memorize the training dataset. However, this change was not positive, as the model stopped generalizing well and began showing signs of undertraining (underfitting), thus impairing its ability to learn meaningful patterns.

The main difference between the Bidirectional layer and the Unidirectional one is that features are extracted in both directions, effectively functioning as two LSTM layers, one backward and one forward (100 neurons in each direction). This layer is added twice to capture contextual information from both the past and the future (low-level features), while the second layer adds depth to the model and learns more complex and abstract representations. The final Dense layer, in turn, transforms these final features into predictions.

In more complex models, additional layers were included: normalization (LayerNormalization), attention (Attention), and concatenation (Concatenate). The first aims to stabilize and accelerate the training of neural networks by normalizing activation values and reducing sensitivity to the scale of inputs, ultimately improving model performance. Next, the attention layer allows the model to focus on specific parts of the input that are more relevant, thanks to the assignment of different weights to distinct parts of the sequence, highlighting the most important elements. This layer is essential for identifying relevant patterns and assigning less importance to values without trends, for example. The concatenation layer (Concatenate) is commonly used to combine the results of the Dropout layer and the Attention layer. It merges the information obtained after being processed by Dropout and weighted by attention, allowing the model to integrate different aspects of the sequence's representation.

The use of the sigmoid activation in the Dense layer serves to map the neuron's output to a range between 0 and 1. This is more common in classification tasks. However, since the values were normalized before being fed into the model, this activation function can be applied. Overall, the results were not better when this function was used.

According to the results described in Table 5.6, the more complex models were the ones that presented inferior results. Therefore, to understand if the batch_size value influenced the results since smaller values tend to force the model to converge and require more training time, it was reduced to 20, as shown in Table 5.7.

Model	**MAE**	**MSE**	**RMSE**	**R^2**
Bi-LSTM (100) + Dropout (0.2) + Bi-LSTM (100) + Dropout (0.2) + Dense	0.0217	9.73×10^{-4}	0.0312	0.9182
Bi-LSTM (100) + LayerNormalization + Dropout (0.2) + Attention + Concatenate (drop_out, attention_out) + Bi-LSTM (100) + Dropout (0.2) + Dense (activation = 'sigmoid')	0.0197	8.59×10^{-4}	0.0293	0.8603
Bi-LSTM (100) + LayerNormalization + Dropout (0.3) + Attention + Concatenate (drop_out, attention_out) + Bi-LSTM (100) + Dropout (0.3) + Dense (activation = 'sigmoid')	0.0183	7.45×10^{-4}	0.0273	0.8541

Table 5.7: Model Performance Metrics, with batch_size = 20.

Comparing the two tables, it is observed that the best model remains the same, although the results were different. With the larger `batch_size` (32), the model generalizes better and generally shows improved values in the R^2, MAE, and RMSE metrics. The R^2 value is a decisive factor, as it reflects the model's ability to generalize well to previously unseen data, and the priority is to establish a good balance between training performance and generalization capability.

Next, the cross-validation mechanism (`TimeSeriesSplit`) was used to create multiple training and validation sets to identify which model generalized better across different datasets. The goal is to achieve consistent results across the folds to determine if the model generalizes well at all times or only in specific cases.

Initially, 3 folds were used to understand how the models behaved when the training set was larger and smaller, as well as the test set. Table 5.8 presents the average of the metrics obtained in each fold to determine which model generalizes better, regardless of the size of the sets provided to the model.

Model (Mean All Folds)	**MSE**	**MAE**
LSTM (100) + Dropout (0.2) + LSTM (100) + Dropout (0.2) + Dense	0.0104	0.0664
Bi-LSTM (100) + Dropout (0.2) + Bi-LSTM (100) + Dropout (0.2) + Dense	0.0040	0.0421
Bi-LSTM (100) + Dropout (0.3) + Bi-LSTM (100) + Dropout (0.3) + Dense (activation = sigmoid)	0.0056	0.0496
Bi-LSTM (100) + Attention + Concatenate + Normalization + Bi-LSTM (100) + Dropout (0.3) + Dense	0.0051	0.0496
Bi-LSTM (100) + Dropout (0.2) + Attention + Concatenate + Bi-LSTM (100) + Dropout (0.2) + Dense (activation = sigmoid)	0.0055	0.0489
Bi-LSTM (50) + Normalization + Dropout (0.2) + Attention + Concatenate + Bi-LSTM (50) + Dropout (0.2) + Dense	0.0059	0.0490
Bi-LSTM (100) + Normalization + Dropout (0.3) + Attention + Concatenate + Bi-LSTM (100) + Dropout (0.3) + Dense	0.0064	0.0545

Table 5.8: Model Performance Metrics, with TimeSeriesSplit (n=3).

With the results presented in Table 5.8, we can conclude that the use of Bidirectional layers is beneficial for the model's training, representing improvements in the prediction of values and the approximation to the real values, as indicated by the MSE and MAE metrics. Once again, more complex models show inferior results when using `TimeSeriesSplit` to evaluate different splits of the dataset. The model with the architecture Bi-LSTM (100) + Dropout (0.2) + Bi-LSTM (100) + Dropout (0.2) + Dense is, once again, the one that has demonstrated the best performance and accuracy compared to the others.

Thereafter, to understand the effectiveness of the GRU model relative to the model considered the best so far (Bi-LSTM), the results obtained are shown in Table 5.9.

When deciding between LSTM and GRU for a specific task, the choice can be based on several factors related to the behavior, performance, and efficiency of these models. Based on the

results shown in Table 5.9, the models that included LSTM layers demonstrated more stability and accuracy in prediction. Since 10 folds were used, multiple tests were conducted on each model, and the metrics (MSE and MAE) indicate that the LSTM-based model performed better. In any case, the GRU model deserves further attention in terms of hyperparameter tuning. However, considering that the GRU architecture requires less computational time to train, and since this problem does not exhibit very noticeable short-term trends—with the main objective being to detect long-term trends—the learning time should ideally be longer to detect complex patterns.

Model (Mean All Folds)	**MSE**	**MAE**
Bi-LSTM (100) + Normalization + Dropout (0.2) + Attention + Concatenate + Bi-LSTM (100) + Dropout (0.2) + Dense	0.0054	0.0432
Bi-LSTM (100) + Dropout (0.2) + Bi-LSTM (100) + Dropout (0.2) + Dense	0.0053	0.0400
GRU (50, activation = 'tanh') + Dense (64, activation = 'relu') + Dense (activation = 'sigmoid')	0.0151	0.0806
GRU (50, activation = 'tanh') + Attention + GRU (50, activation = 'tanh') + Dense (64, activation = 'relu') + Dense (activation = 'sigmoid')	0.0058	0.0541

Table 5.9: Model Performance Metrics, with TimeSeriesSplit (n=10).

LSTMs are ideal when the task involves learning detailed sequence patterns and sufficient computational resources are available. On the other hand, GRUs are typically more efficient when patterns are less complex, or resources are limited.

Regarding the model that is already nearly finalized, some variants of it were tested with slight changes in the hyperparameters to find a model that could perform even better, as presented in Table 5.10.

Model	**MSE**	**MAE**	**Explained Variance**	**R^2**
Bi-LSTM(150) + Dropout (0.3) + LSTM(100) + Dropout (0.3) + Dense	0.0005	0.0149	0.9733	0.9521
Bi-LSTM(100) + Dropout (0.2) + LSTM(100) + Dropout (0.2) + Dense	0.0004	0.0112	0.9817	0.9693
Bi-LSTM(100) + Dropout (0.2) + LSTM(50) + Dropout (0.2) + Dense	**0.0002**	**0.0069**	**0.9820**	**0.9787**
Bi-LSTM(100) + Dropout (0.2) + Bi-LSTM(50) + Dropout (0.2) + Dense	0.0004	0.0136	0.9880	0.9510
Bi-LSTM(100, activation = 'tanh') + Dropout (0.2) + LSTM(50) + Dropout (0.2) + Dense	0.0007	0.0153	0.9597	0.9366

Table 5.10: Model Performance Metrics, with TimeSeriesSplit (n=5).

The hybrid approach of placing a bidirectional layer at the beginning of the model and then stacking a unidirectional layer coupled with the LSTM yielded the best results. The reason for this, as shown in Table 5.10, is that at the beginning, the general bidirectional informa-

tion is collected, followed by direct short-term dependencies. In other words, the first layer **Bi-LSTM (100)** captures rich representations of the sequence, including both short- and long-term dependencies in both directions. A subsequent unidirectional LSTM layer can then refine these features by focusing on just one direction (forward) rather than introducing another bidirectional layer that might ”dilute” the already processed information.

Another reason might be that the Bidirectional layer has more parameters due to processing in two directions, which results in longer training time and greater computational complexity. In contrast, the unidirectional LSTM layer can be more efficient in converging to a more stable and efficient model, thus favoring model optimization.

The final results obtained from the chosen model for each variable are represented in Figure 5.3.

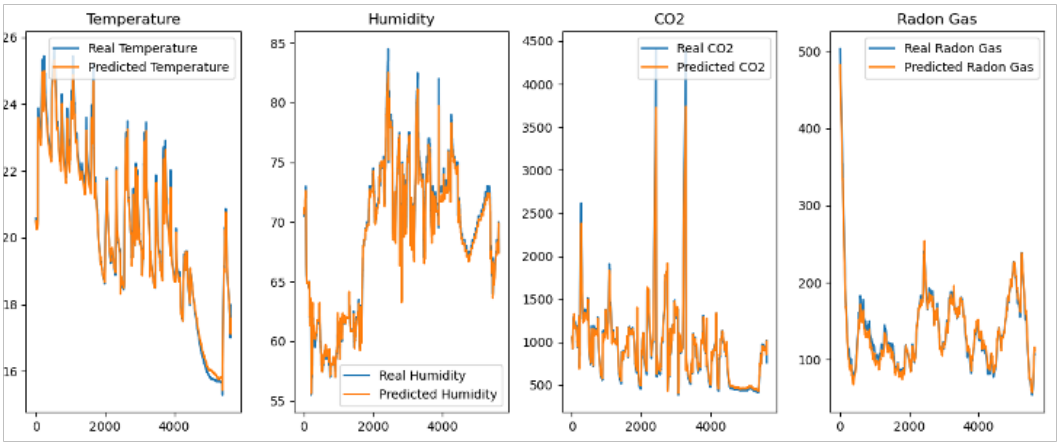

Figure 5.3: Prediction Results (Real vs Predicted Values).

The variable that showed the most deviation from the predicted values was radon gas, likely due to its known volatility. For the remaining variables, the model adapted well, although it is not as accurate at the peaks, it performs quite well when the values remain constant. Despite yielding good results, the model still struggles to reach extreme values in some cases, possibly due to the limited data (only 3 months).

5.3.1.4 Comparative Between Models, with Hybrid Approach

Since the performance of the models used depends directly on the real problem and the characteristics of the data, a hybrid methodology offers both linear and non-linear modeling capabilities, which can be a good strategy. For example, by combining different models, various aspects of the underlying patterns can be captured.

The hybrid approach, known as stacking, combines two models into one, which often leads to better performance due to its ability to predict using a statistical model (ARIMA), which captures linear patterns more effectively, together with a model based on binary trees or neural networks, which excels at detecting complex and non-linear patterns. To test this stacking system, several tests were conducted with different architectures, and the performance metric values were compared, as shown in Table 5.11.

Hybrid Model	MSE	MAE
ARIMA (3,0,1) + [Bi-LSTM (100) + Dropout (0.2) + Bi-LSTM (100) + Dropout (0.2)]	0.0177	0.0770
ARIMA (2,0,1) + [Bi-LSTM (100) + Dropout (0.2) + Bi-LSTM (100) + Dropout (0.2)]	0.0177	0.0770
ARIMA (2,0,1) + [RandomForest (max_depth = 10, min_samples_split = 2, n_estimators = 50)]	0.0010	0.0099
ARIMA (2,0,1) + [Gradient Boosting (max_depth = 10, n_estimators = 100, subsample = 0.8899)]	0.0061	0.0121
RandomForest (max_depth = 10, min_samples_split = 2, n_estimators = 50)	0.0018	0.0100
Gradient Boosting (max_depth = 10, n_estimators = 100, subsample = 0.8899)	0.0018	0.0099

Table 5.11: Hybrid Model Performance Metrics.

In Table 5.11, it can be seen that only the ARIMA (2,0,1) + [RandomForest (max_depth = 10, min_samples_split = 2, n_estimators = 50)] model stood out from the rest regarding the hybrid approach, as the results were better when using ARIMA with binary trees (RandomForest). This was the only hybrid approach tested that showed better results compared to the metrics obtained when the models were used separately. Figure 5.4 shows the actual and predicted values with the models based on decision trees.

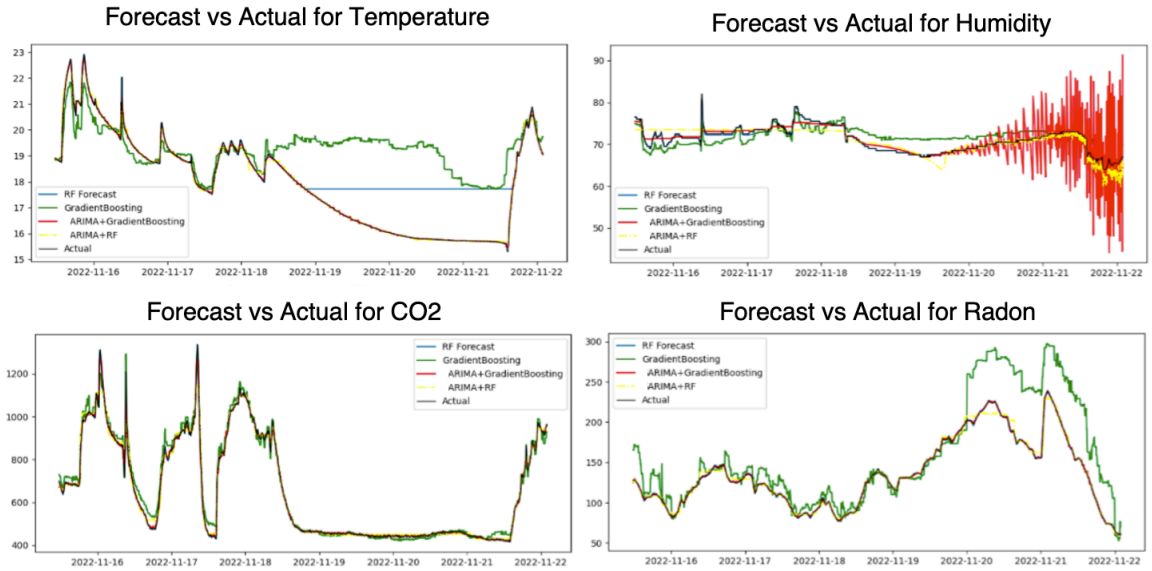

Figure 5.4: Hybrid Models.

On the other hand, based on the graphs, it can be immediately noticed that the results obtained from the hybrid model (SARIMA/ARIMA) with Gradient Boosting presented a significant amount of noise in the prediction of the Humidity variable, whereas the other models seemed more accurate. Tree-based models always have more difficulty reaching the limits, partly due to faster training and their inability to detect patterns and trends in the data, which explains the predicted values being farther from the actual values. Gradient Boosting performs better when not combined with a statistical model, as shown in the data in Table 5.11.

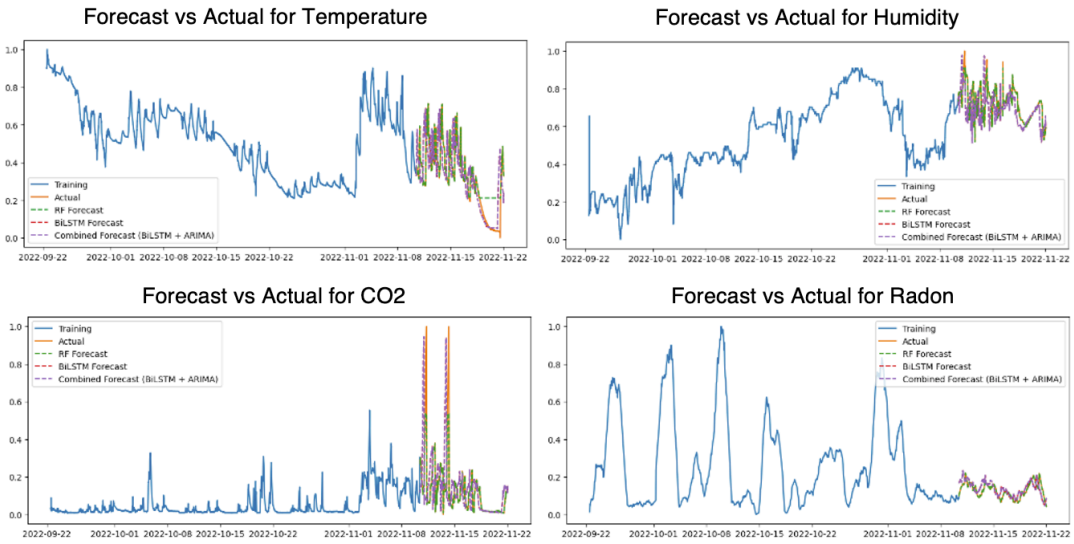

Figure 5.5: Hybrid Values.

The LSTM architecture combined with the statistical model (ARIMA) was less beneficial, as the model was less capable of learning the patterns, which resulted in lower presented values. Figure 5.5 displays several graphs, one for each variable, showing the predictions for each model.

According to the graphs, the model that came closest to the actual values, managing to approach the extremes, was the Bi-LSTM, without the hybrid approach, with predictions more aligned with the real values. The combination of Bi-LSTM with ARIMA, although slightly better than the tree-based model (Random Forest), was not ideal in the graph.

Among all the hybrid models, the one that showed the best results was ARIMA (2,0,1) + RandomForest (max_depth = 10, min_samples_split = 2, n_estimators = 50), as it most effectively captured the trends and came closest to the predicted values. However, the model that continues to predict the best is the Bi-LSTM(100) + Dropout (0.2) + LSTM(50) + Dropout (0.2) + Dense, as concluded in section 5.3.1.3. These findings demonstrate that using a hybrid approach between the statistical model and a neural network-based model was not beneficial for this type of problem.

5.3.2 Integration of ML-based Decision Support System

After working with neural networks, specifically those with LSTM architecture, it was noted that the model with the best results is the one that utilizes both unidirectional and bidirectional layers of this architecture. Subsequently, other approaches were tested, namely a hybrid approach, as discussed in Section 5.3.1.4, where the results of linear models were combined with the results of nonlinear models. This attempt did not yield better results compared to those obtained with the LSTM architecture, so this approach was discarded.

Thus, the model considered the final model is Bi-LSTM(100) + Dropout (0.2) + LSTM(50) + Dropout (0.2) + Dense. After arriving at the appropriate model for predicting future values,

it is necessary to iterate the prediction as many times as the desired number of future steps.

All these models were trained to examine sets of 100 data points (seq_length) and detect trends within that data when provided with a large dataset. The input provided always included the last 100 records to predict the next value. However, the purpose of this decision support system is to obtain predictions for the next few hours, not just for the next 5 minutes.

To make the model iterative, an autoregressive approach was used, where the model is called at each prediction step, and the previous prediction is used as input to predict the next value. In this case, the model only predicts one step ahead at a time and uses that prediction to inform the next, updating the input sequence with each iteration. This iterative model has disadvantages, as errors in early predictions can propagate and affect subsequent predictions. The idea is to use the model to predict not only the next value but a sequence of future values, one after another.

5.3.2.1 Iterative Forecasting Method

The iterative forecasting process can be described as follows:

- **Iteration 1:**
  - *Input:* $[X_1, X_2, X_3, \ldots, X_{100}]$
  - *Prediction 1:* P_1 (for time $T + 1$)
  - *Update Sequence:* $[X_2, X_3, \ldots, X_{100}, P_1]$
- **Iteration 2:**
  - *Input:* $[X_2, X_3, \ldots, X_{100}, P_1]$
  - *Prediction 2:* P_2 (for time $T + 2$)
  - *Update Sequence:* $[X_3, \ldots, X_{100}, P_1, P_2]$
- **Final Iteration:**
  - *Input:* $[X_{n-98}, \ldots, X_n, P_{n-1}]$
  - *Prediction N:* P_N (for time $T + N$)
  - *Update Sequence:* $[X_{n-97}, \ldots, P_{n-1}, P_N]$

Using a sliding series of values, this iterative forecasting approach makes forecasts across multiple time steps. The model initially predicts P_1 for the next moment ($T+1$), given a series of 100 historical values $[X_1, X_2, ..., X_{100}]$. Next, the sequence is updated by adding the new predicted value (P_1) at the end and removing the oldest value (X_1). The second prediction, P_2, for the subsequent moment ($T + 2$), is then made using this new sequence, which now consists of the 99 most recent historical values and the new prediction. Iterations of this procedure follow, with each iteration drawing on the prior predictions to guide the current one.

The technique maintains a constant window of 100 values throughout the iterations, gradually replacing historical data with predictions. This may be helpful in capturing long-term temporal trends, as each future forecast depends on the ones made before it. The previously trained model produces one future time step, which matches the input expected by the fixed window (seq_length) of 100.

The number of future time steps to anticipate may be specified, though it is recommended not to set this value too high since predictions naturally become less accurate as more iterations are conducted.

5.3.2.2 Send Alerts by Predicition

If the user grants permission for notifications related to forecasted values, they agree to be notified of any altered values that may appear in the forecasts for the upcoming hours. If there is any irregularity in the forecasted values, the user is alerted so that this situation can be minimized, and the forecasted values are seen as crucial for the user to mitigate natural disasters and the emergence of diseases.

For this system to function correctly, it is necessary to use a script that runs automatically, at set intervals, as described in section 5.3.2.1, to ensure that the system is always monitoring data changes in order to assist in decision-making by forecasting future patterns and detecting possible anomalous values that may occur in the near future.

5.3.3 Digital Platform Evaluation

In this section, the different pages that make up the RuraLTHINGS platform will be discussed, starting with the authentication page where the user can create an account if they do not have one. Next, the user's home page, where they can check the latest sensor readings and the status of various devices and receive feedback on the houses. There is a dedicated layout for the houses associated with the user ('Devices'), which allows viewing all the houses, the devices associated with each house, and, in turn, the ability to add new houses. The 'Sensor Data' page allows for reviewing all the data received from the IoT sensors, analyzing each environmental variable in detail over time, while the 'Statistics Data' page layout enables viewing the data through averages, understanding the hourly/daily/monthly average for each environmental gas. On the other hand, the 'Predictions' page allows for viewing predicted data according to the decision support system, providing insights into how the model tends to predict the values for the upcoming hours. The alerts page enables viewing any alerts that may have arisen based on the real data already sent to the system from the IoT devices. Lastly, the settings page allows the user to log out of their account, edit their profile information, and enable/disable notifications based on predictions.

5.3.3.1 Login and Register Page

In Figures 5.6 and 5.7, the user authentication process for the platform is illustrated. In the first figure, the user fills in their credentials if they are already registered in the system. If registration is required, the user first enters their information as shown in Figure 5.7. Then, a verification email is sent, which includes a code that must be entered on the next page to complete the registration process. If the user has forgotten their password, the steps are similar, involving the sending of a verification code to the email, followed by its verification and the introduction of a new password.

There are some validations performed on the email and password fields to ensure the user enters the correct email in the designated field and that the password meets certain requirements, such as having at least 6 characters.

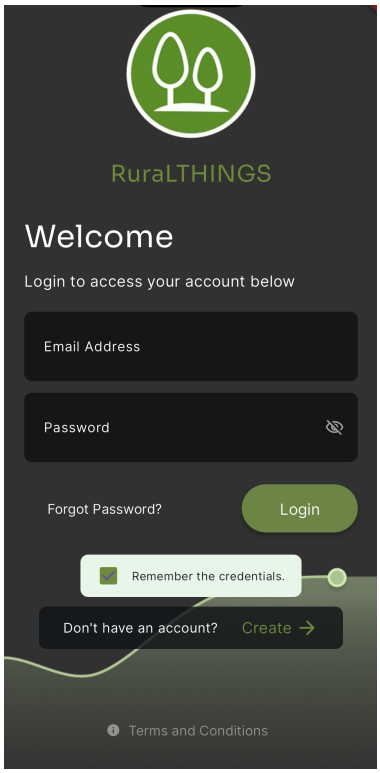

Figure 5.6: Login Page for User.

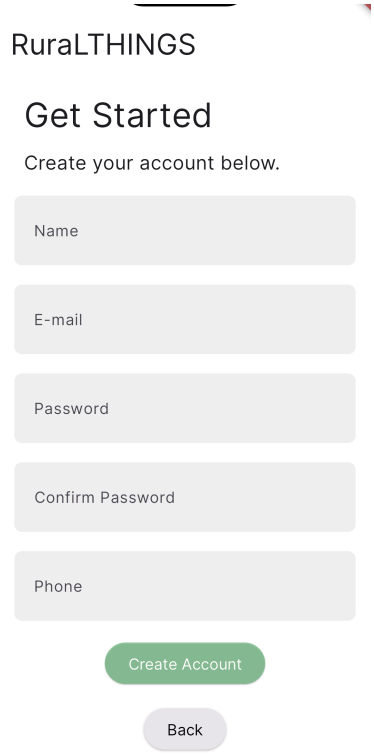

Figure 5.7: Register Page for User.

5.3.3.2 Home and Devices Page

The homepage, represented in Figure 5.8, is the first page the user encounters upon logging into the application, where they can check information related to their home and its status. If recent data is available, it will be displayed in the form of a graph within a ScrollView that shows various environmental gases. By clicking on any of these gases, the user is redirected to the page that shows detailed data collected by the IoT sensors. Other pages easily accessed from the homepage include the alerts page, as well as statistics, settings, and user-associated devices, all of which are present in the persistent bottom menu (BottomNavigation).

The devices associated with the user can be viewed on the 'DevicesInfo' page, as shown in Figure 5.9, where the user can see which devices are linked to each home, access specific device data, and also add new devices to homes already registered in the system. Figures 5.9 and 5.11 respectively show the list of homes without displaying the devices, followed by detailed information for each associated device, including the date of the last update, the location of the device within the home (as entered by the user), and direct access to the data collected by that device. Figure 5.10 shows a button at the bottom of the list that allows users to add a new house to the system. Once clicked, the user is redirected to the page represented in figure 5.12 to add devices to this new house.

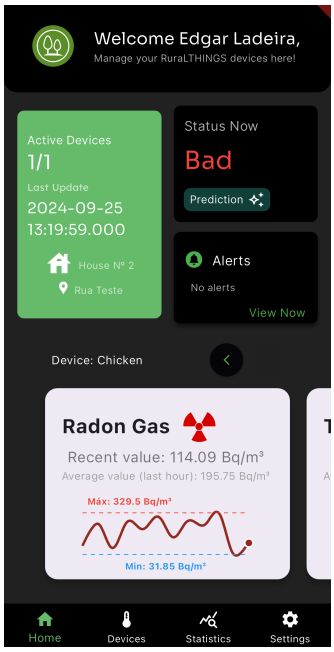

Figure 5.8: Home Page of Platform.

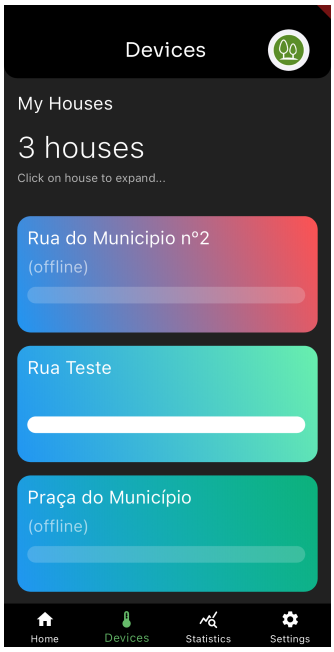

Figure 5.9: Devices Info.

Figure 5.10: Add House.

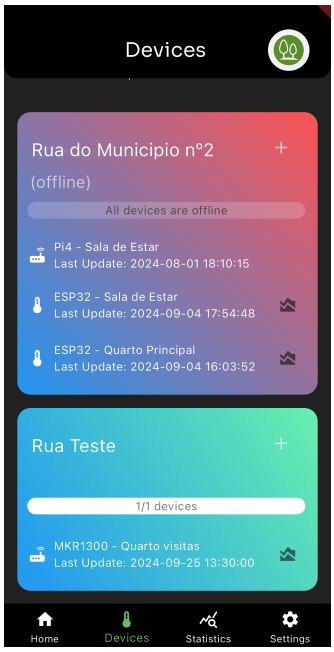

Figure 5.11: Device Info Expanded.

If the user wants to install new devices, they can associate them with homes already registered in the system or add a new home to the system. To add a new home, they must first provide the code issued by the administration, which is linked to all the devices assigned to the user. The page shown in Figure 5.12 requests this code (Admin Code). Once entered correctly, the user is redirected to a page where they can specify the areas of their home where each device will be installed.

This code is provided to the user when they acquire the RuraLTHINGS devices. The code groups the various devices either collected by the user or installed by the technical team. From a technical perspective, in the back office, this code is stored in a table along with a DateTime entry to record the acquisition of these devices. If the location is the same for all devices, the user can check the checkbox to copy the description of the first device and replicate it for the others if there is more than one.

Before allowing the user to adjust the map marker to specify the house location, the app requests permission to access the location. This helps position the map closer to the current location, aiding in selecting the correct area for house registration. Figure 5.14 shows the permission request dialog.

Figure 5.13 shows the location of the device (MAC ADDRESS MA:LD:EO:29:36:32) located in the bedroom. Then, by clicking 'Submit Locations' the user specifies the location of the house to which the device will be added. On the 'Register House' page, the user can specify the street where the house is located by entering the address, which triggers a search through the Flutter API to fetch the latitude and longitude coordinates after the street is input. The map marker will then be positioned on the specified street, allowing for pin adjustment to the exact location, as shown in Figure 5.15.

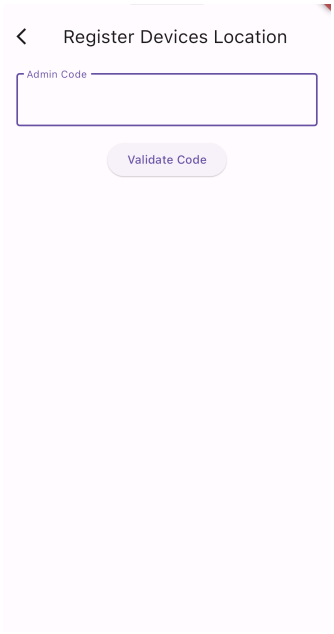

Figure 5.12: Register Device Location.

Figure 5.13: Device Info Expanded.

The marker can be fixed on the map by clicking the 'Fix Marker' button, making user interaction with the map easier. Finally, upon proceeding with the insertion ('Register House'), the user is redirected to the 'Devices' page, where the newly added home and device should already be present.

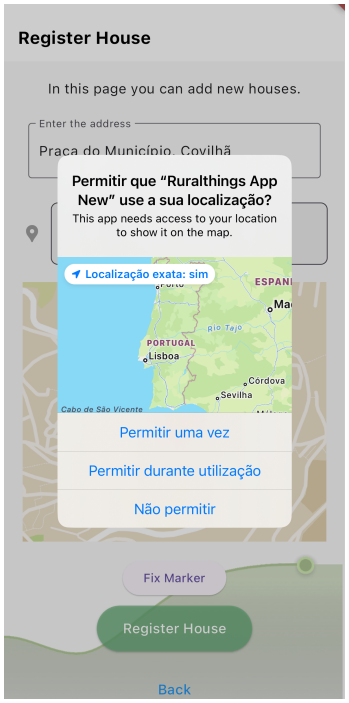

Figure 5.14: Location Permission.

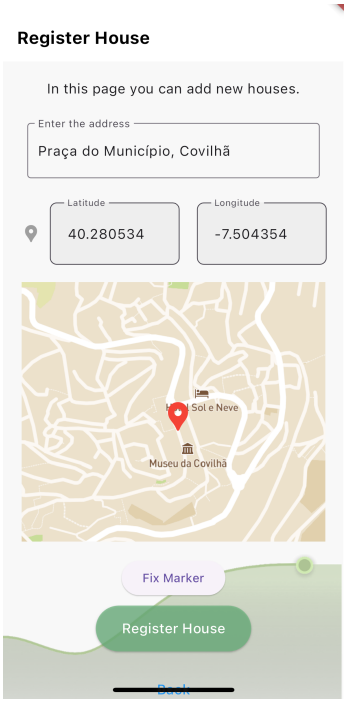

Figure 5.15: Add House.

5.3.3.3 Sensor Data Page

The page that displays sensor data provides a comprehensive and intuitive experience for real-time data visualization, with the most recent data appearing on the right side of the line graph and historical data on the left. Below the graph, a synchronized list offers a detailed and alternative view of the same data.

Users can customize the viewing period using a filter popup, with options to view the last 24 hours (default), 48 hours, or select two consecutive days from the calendar, ensuring the application remains lightweight and efficient. Additionally, the graphs have predefined limits for the sensors: 'Temperature' (5.0 - 25.0), 'Humidity' (30.0 - 60.0), 'CO' (0.0 - 10.0), 'CO_2' (0.0 - 1000.0), and 'Radon' (0.0 - 100.0). Values outside these parameters are highlighted with different colors in both the graph and the list, identifying them as abnormal or concerning values. In case of data gaps, whether due to connection failures or sensor issues, the list automatically adjusts the records, displaying them slightly separated to facilitate the clear understanding and interpretation of information, even in cases of data inconsistency.

Figure 5.16 represents the values received from the Temperature sensor, and Figure 5.17 shows the values for radon gas collected by the same device, meaning the data is gathered from the same environment. To switch between environmental variables, there is a navigation bar that allows the user to update the values shown in the graph and list.

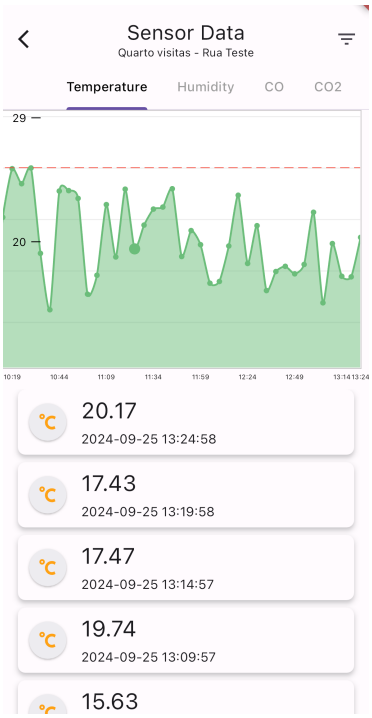

Figure 5.16: Sensor Data (Temperature).

Figure 5.17: Sensor Data (Radon) with Alert Values.

5.3.3.4 Statistics Page

The statistics page provides a comprehensive overview of sensor data, allowing users to track statistics in a detailed and organized manner. Users can view data by the hour for a single day (hourly), by the day for a specific month (daily), or by the month for an entire year (monthly). For daily and monthly visualizations, the values are based on the averages of all captured data, offering a more condensed and easy-to-interpret view of trends over time. This structure allows users to analyze sensor behavior across different time scales, making it easier to identify patterns and anomalies in both the short and long term.

In Figure 5.18, the values for the Temperature variable are shown for all days of the month with available data, whereas in Figure 5.19, the statistical values for Temperature are displayed according to the most recent hours. The monthly option allows viewing yearly averages, showing all the months during which sensor data was collected.

The radon gas statistics in Figure 5.20 for September 23rd show that, based on the data from the past 24 hours, all values were above the limits considered safe for this radioactive gas. The data was collected from a device located in the house on 'Rua Teste'. It is possible to select data from specific sensors or calculate the average across all sensors in that house ('All sensors'), as demonstrated in Figure 5.21. If the user wants to view the statistics for a different house, they can simply select the house in this popup.

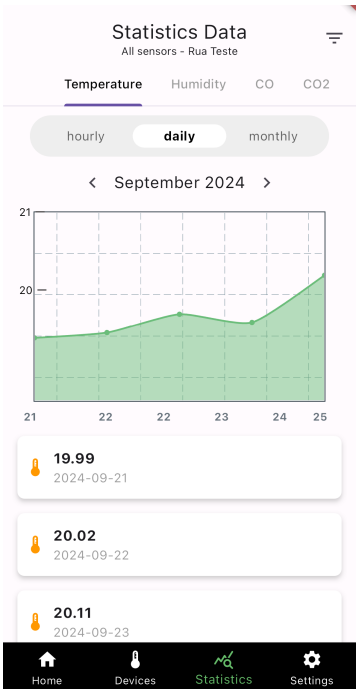

Figure 5.18: Statistics Page Daily (Temperature).

Figure 5.19: Statistics Page Hourly (Temperature).

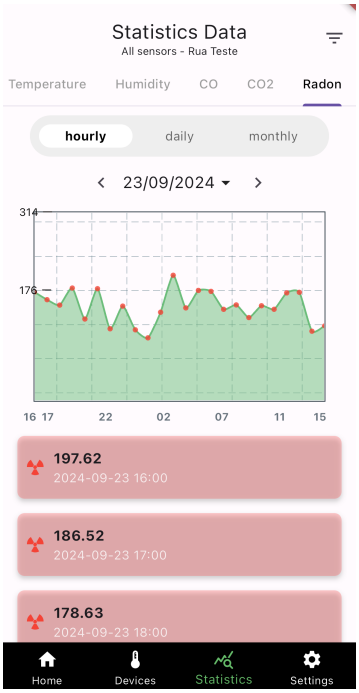

Figure 5.20: Statistics Page Hourly (Radon).

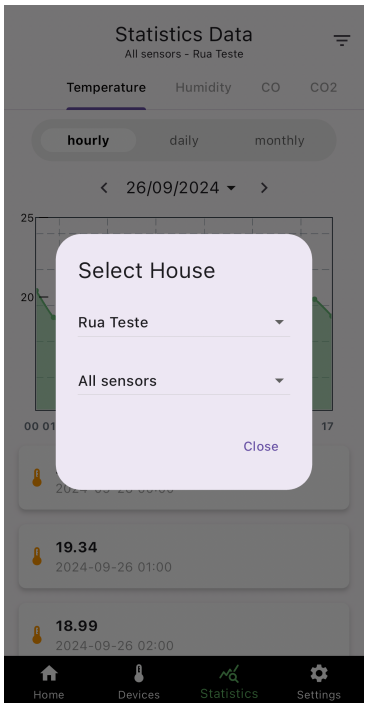

Figure 5.21: Popup to Filter Devices and Houses.

5.3.3.5 Predictions Page

The page aims to provide forecasts based on the data collected by the sensors, using a decision support model to help identify trends. Utilizing the iterative approach discussed in subsection 5.3.2.1, the page processes the last 100 real records and predicts the next 10, which correspond to approximately 50 minutes of future readings. This limitation on the number of predictions ensures greater accuracy, as long-term forecasts tend to be less reliable.

Visually, the graph on the Predictions Page is similar to the graph on the Sensor Data page, with one key distinction: it first displays the 100 real values that served as the model's basis, followed by the 10 predictions. Similarly, in the list below, the predicted values appear before the real values, offering the user a clear view of future trends. It is important to note that CO gas is not included in this predictions page, which focuses only on parameters such as temperature, humidity, CO_2, and radon.

Figure 5.22 shows the temperature predictions, while Figure 5.23 illustrates the humidity predictions.

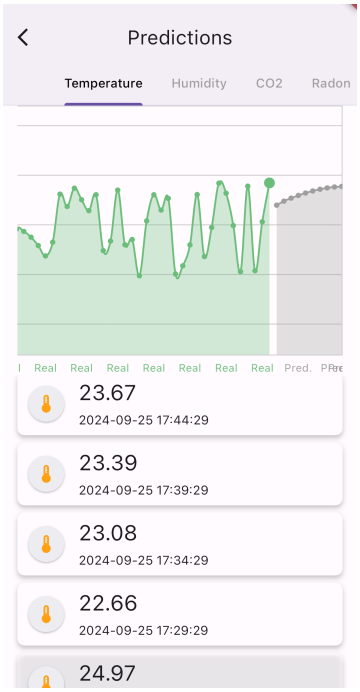

Figure 5.22: Predictions Page (Temperature).

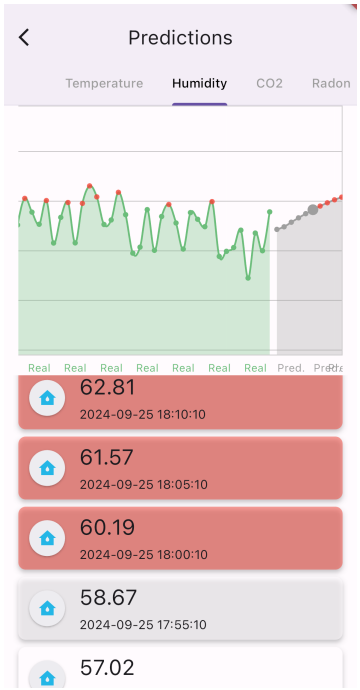

Figure 5.23: Predictions Page (Humidity).

5.3.3.6 Alerts Page

Alerts can be marked as seen by the user, allowing the administration to know that the user is aware of the occurrences and understands the potential danger of the collected values. It is also possible to mark all alerts as read or delete alerts in case the user has performed any activity nearby that might have altered those values, resulting in manipulated data.

First, the status is shown as sent by the API, based on all the devices the user possesses, reflecting occurrences from the last 24 hours. This status can be 'Critical', 'Under Surveillance', or 'Safe', depending on the severity of the latest occurrence. As defined in section 5.3.4, this corresponds to the number of consecutive occurrences, and if a given threshold is exceeded, all subsequent records become prioritized.

Priority alerts are displayed first, filtered from the most recent to the oldest. The user can disable this search filter to view all alerts. These alerts are stored in the Gases_Alerts table in the database.

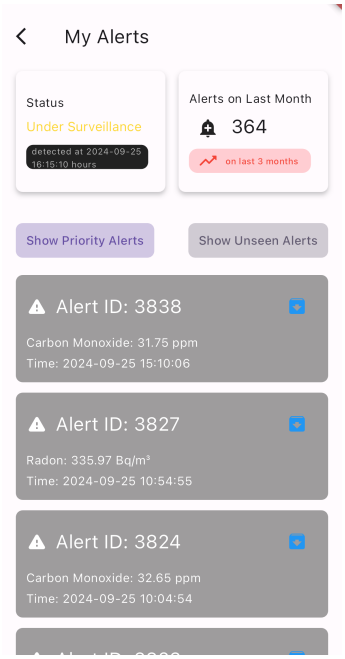

Figure 5.24: Alerts Page.

5.3.3.7 Settings

Settings are essential in the app, allowing users to control certain actions, as shown in Figure 5.25. By default, real-time alerts are enabled for all users. If they wish to disable them, they simply need to click the 'Disable Realtime Alerts' button, and the same applies to alerts based on the decision support system. Figure 5.26 allows users to edit their personal and app access information. The email is the only detail that cannot be changed, while all other information can be updated.

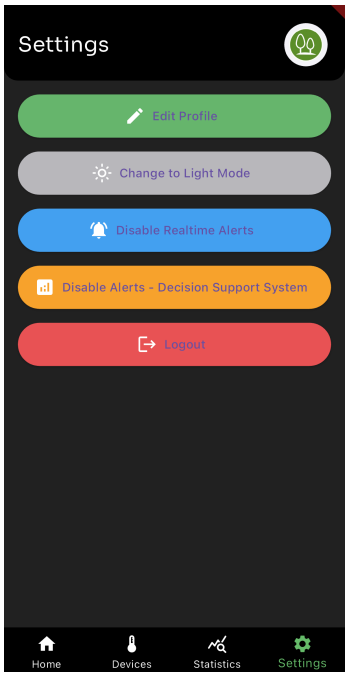

Figure 5.25: Settings Page.

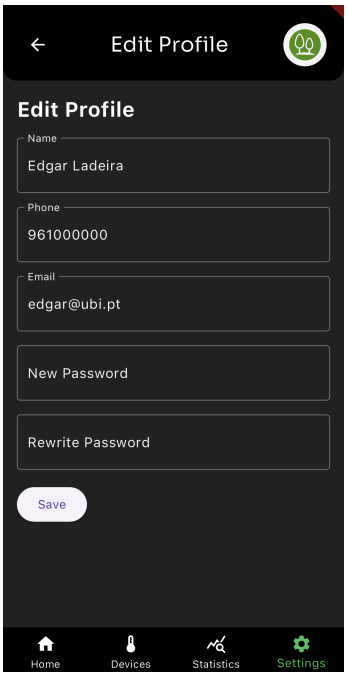

Figure 5.26: Edit Profile Page.

5.3.4 Alert Notification System Evaluation

This SQL trigger monitors gas levels (radon, carbon monoxide, and carbon dioxide) on a device. When new data is inserted, it checks if the gas levels exceed predefined thresholds or are invalid. If any abnormal values are detected, the corresponding gas is logged, and if multiple alerts have been generated for the same gas in the past hour, the alert priority is marked as true. The goal is to compare the values recorded in the new data inserted by the IoT device with predefined limits to identify any abnormal values and generate alerts as needed. These thresholds vary for each gas, based on research conducted by numerous organizations. For example, radon gas is considered concerning when its value exceeds 100.

The system checks whether the radon gas (radonGas) value is greater than 100 or equal to -1, which may indicate an invalid reading or a sensor failure. If either condition is met, radon is flagged as an abnormal gas. After this identification, the code performs a second check to count how many radon-related alerts have been generated in the past hour for the device. If two or more alerts exist in this time frame, the current alert is marked as a priority, signaling that immediate attention is needed.

Similarly, for carbon monoxide (carbonMonoxide), the code checks if the value exceeds 30 or equals -1. If either condition is true, the gas is considered abnormal. If an abnormal gas is already registered, carbon monoxide is added to the list of detected abnormal gases. Like radon, the code checks whether two or more carbon monoxide alerts have been logged in the past hour for the device. If the count is two or more, the alert is marked as a priority.

Lastly, carbon dioxide (carbonDioxide) is monitored, and if the value exceeds 80 or equals -1, it is flagged as abnormal. If other gases are already marked as anomalies, carbon dioxide is added to the list. If there are already two abnormal gases logged, carbon dioxide becomes the third abnormal gas. Additionally, the code checks if more than two alerts for carbon dioxide have occurred within the past hour, and if so, the current alert is marked as a priority.

If any abnormal values are detected, the code inserts a new alert into the Gases_Alerts table, including the device ID, abnormal values, gases involved, alert date and time, and priority status if applicable. This ensures continuous monitoring and signals dangerous conditions related to the monitored gases.

Once the Gases_Alerts database contains the data, a script runs every five seconds to identify fresh items in the table and then notifies the related users. If the alert has already generated a notification, it is marked as processed. If not, it is handled in a way that depends on what action caused it. A special notification is triggered for sensor malfunctions when an anomalous value of '-1' is detected. Different kinds of alerts are produced if the levels of every gas fall outside their safe ranges.

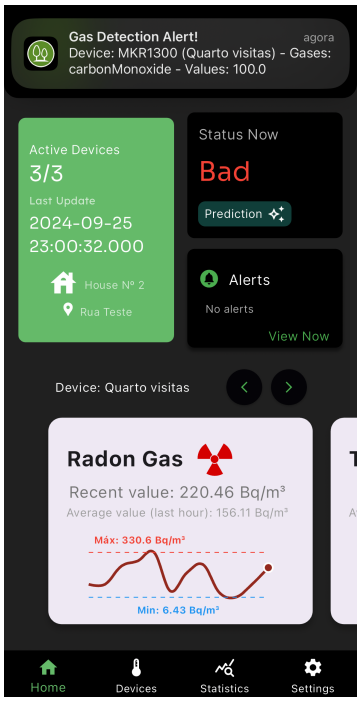

Figure 5.27: Notification with Alert of Abnormal Value on Gas (carbonMonoxide).

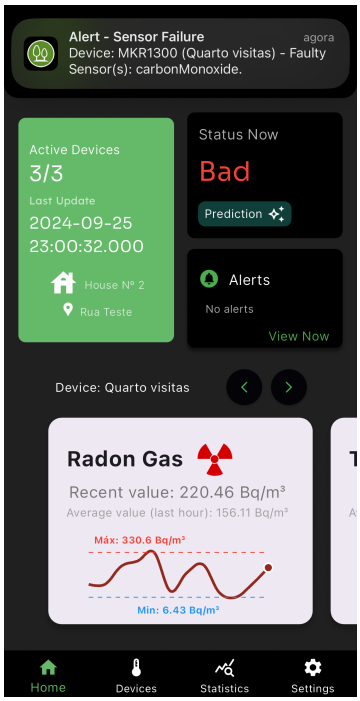

Figure 5.28: Notification with Alert of Sensor Failure (carbonMonoxide).

5.4 Conclusion

Applying this approach to the IoT devices requires careful consideration of the reality that the data gathered by sensors spread over several sites. It is probable that one will have to change the weights of the model or create particular adjustments for several settings. An important one is whether the model requires to be trained individually for every device or whether a single model based on data from all devices can fairly anticipate what would happen on each device.

Afterward, the finished model with both one-way and two-way LSTM layers was put into use, and the decision support system obtained good results. This model did better than statistical models and models based on decision trees. Good success was found in the review, even though it was based on only three months' worth of data. When the model was being trained, carbon monoxide was left out because there was not enough data in the collection for it. The final model worked well at predicting what would happen next when it was used repeatedly. However, future versions of the platform could include more advanced notifications and a better alert system that can work with new technologies like virtual assistants or more advanced predictive alerts.

The digital platform has many features that make it easier for users to work with it. For example, it lets us see sensor-collected data in simple graphs. Because the decision support system is built into the platform, users can get warnings about predictions of upcoming trends along with real-time data changes. This lets the user take preventative steps to lessen the harm that might happen.

In summary, the platform has made it much easier to add a decision support system to IoT settings. This gives users a smarter and more effective way to keep an eye on environmental data and guess what risks might happen. The model has some promise, but applying it to more devices and looking into different neural network designs could make the system work better and be more useful.

Chapter 6

Conclusions and Future Work

6.1 Conclusion

This dissertation introduced a novel approach to the surveillance of environmental conditions and pollutant gases in residential areas, utilizing IoT technologies and machine learning models to promote public health. By incorporating real-time alerts for anomalous values and predictive capabilities for pollutant levels, the digital platform that was developed exhibited substantial potential. This prognostic capacity is essential for the prevention of health hazards, particularly in vulnerable populations such as the elderly, who spend an increased amount of time indoors. In this sense, all the main objectives of this dissertation were accomplished.

A digital platform, directed for mobile platforms (Android and iOS) was developed with the essential features. Certain adjustments to optimize its performance on desktop devices guarantee a seamless and responsive user experience. The requirements reported in this document were met during the development of the platform to ensure the functionality of the system.

The implemented model has demonstrated satisfactory performance in predicting values with respect to the recommendation system. However, additional testing is necessary. The model would be able to learn more effectively from complex patterns, particularly when predicting extreme values, by utilizing a larger volume of data and expanding the global context of the system. Furthermore, the computational performance of an iterative approach has been demonstrated to be efficient, despite the probability that it may accrue some defects over time. This method speeds up the forecasting process, especially when recent data holds greater significance, as in this project. The integration of the DSS into the platform was successful, as users can view future predictions (insights) on the platform and receive real-time alerts, such as preventive measures to mitigate health risks. Notifications also appear when a device stops functioning, meaning there is no recent data available in the system.

Therefore, the integration of IoT technologies with machine learning has demonstrated potential, allowing users to make informed decisions in real-time, thereby promoting a safer and healthier environment. Specifically, the RuraLTHINGS initiative addresses the significant advancement in public health that the capacity to predict and alert about environmental hazards represents, particularly in rural areas with limited connectivity.

6.2 Future Work

Several areas for future improvement and expansion of the project have been identified. These include:

- **App adjustments and bug fixes:** Refining both the mobile and web applications to enhance user experience and address any identified issues or bugs;
- **Application in a real-world context:** Implementing the prediction system in the real-world context of the RuraLTHINGS project with a larger dataset; This will improve the accuracy and reliability of predictions as more data is collected;
- **Automatic model retraining and refinement:** Developing a script for automatic model retraining at regular intervals to keep the model updated and capable of detecting seasonal patterns. As the dataset grows in complexity and size, it will also be necessary to adjust model parameters such as batch size and the number of neurons to maintain optimal performance;
- **Multilingual support and data inclusion:** Adding support for multiple languages to make the platform more accessible, and integrating carbon monoxide (CO) data once available to further enhance prediction capabilities;
- **App distribution:** Expanding the platform's reach by releasing the mobile app on both the App Store and Google Play Store.

The improvements to the RuraLTHINGS project will substantially increase its utility and usability, while increasing its adaptability to the changing requirements of remote and rural communities, thereby promoting a safer and healthier environment for all citizens. The project seeks to increase operational efficiency and improve public health outcomes for citizens through the continuous improvement of the mechanisms used in the predictive analytics process.

Bibliography

[1] Lusoradon, "Radon map of portugal," https://www.lusoradon.com/mapa-de-radao-de-portugal, 2024, accessed: October 20, 2023. xv, 10, 11

[2] H. Deng, Y. Zhou, L. Wang, and C. Zhang, "Ensemble learning for the early prediction of neonatal jaundice with genetic features," *BMC Medical Informatics and Decision Making*, vol. 21, 12 2021. xvi, 72

[3] N. Aziz, E. A. P. Akhir, I. A. Aziz, J. Jaafar, M. H. Hasan, and A. N. C. Abas, "A study on gradient boosting algorithms for development of ai monitoring and prediction systems," in *2020 International Conference on Computational Intelligence (ICCI)*, 2020, pp. 11–16. xvi, 82

[4] RuraLTHINGS. (2023) Ruralthings site. Accessed: 2023-11-12. [Online]. Available: https://ruralthings.ubi.pt 3

[5] J. J. Zhang, Y. Wei, and Z. Fang, "Ozone pollution: A major health hazard worldwide," *Frontiers in Immunology*, vol. 10, p. 2518, 2019. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fimmu.2019.02518/full 8

[6] G. L. P.H. Ryan, "A review of land-use regression models for characterizing intraurban air pollution exposure." *Inhalation Toxicology*, no. 19, pp. 127–133, 2007. 8

[7] I. W. G. on the Evaluation of Carcinogenic Risks to Humans, I. A. for Research on Cancer, and N. C. I. (U.S.), *Man-made Mineral Fibres and Radon*, ser. Centre International de Recherche sur le Cancer Lyon: IARC monographs on the evaluation of carcinogenic risks to humans. International Agency for Research on Cancer, 1988. [Online]. Available: https://books.google.pt/books?id=3synw-4viyEC 8

[8] W. H. Organization, *WHO global air quality guidelines: particulate matter (PM2.5 and PM10), ozone, nitrogen dioxide, sulfur dioxide and carbon monoxide*. World Health Organization, 2021. 8

[9] W. H. O. (WHO). (2018) 9 out of 10 people worldwide breathe polluted air, but more countries are taking action. Accessed: 2023-11-12. [Online]. Available: https://www.who.int/news-room/detail/02-05-2018-9-out-of-10-people-worldwide-breathe-polluted-air-but-more-countries-are-taking-action 8

[10] B. Papp, F. Deák, A. Horváth, A. Kiss, G. Rajnai, and C. Szabó, "A new method for the determination of geophysical parameters by radon concentration measurements in bore-hole," *Journal of Environmental Radioactivity*, vol. 99, no. 11, pp. 1731–1735, Nov 2008, epub 2008 Sep 11. 8

[11] J. Kemski, A. Siehl, R. Stegemann, and M. Valdivia-Manchego, "Mapping the geogenic radon potential in germany," *Science of The Total Environment*, vol. 272, no. 1, pp. 217–230, 2001, radon in the Living Environment. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0048969701006969 9

[12] K. Hámori, E. Tóth, A. Losonci, and M. Minda, "Some remarks on the indoor radon distribution in a country," *Applied Radiation and Isotopes*, vol. 64, no. 8, pp. 859–863, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0969804306001527 9

[13] S. Darby, D. Hill, A. Auvinen, J. Barros-Dios, H. Baysson, F. Bochicchio, H. Deo, R. Falk, F. Forastiere, M. Hakama, I. Heid, L. Kreienbrock, M. Kreuzer, F. Lagarde, I. Mäkeläinen, C. Muirhead, W. Oberaigner, G. Pershagen, A. Ruano-Ravina, E. Ruosteenoja, A. Rosario, M. Tirmarche, L. Tomášek, E. Whitley, H. Wichmann, and R. Doll, "Radon in homes and risk of lung cancer: collaborative analysis of individual data from 13 european case-control studies," *BMJ*, vol. 330, no. 7485, p. 223, 2005. 9

[14] World Health Organization, *WHO Handbook on Indoor Radon: A Public Health Perspective*. Geneva: World Health Organization, 2009, pMID: 23762967. 9

[15] D. Krewski, J. H. Lubin, J. M. Zielinski, M. Alavanja, V. S. Catalan, R. W. Field, J. B. Klotz, E. G. Létourneau, C. F. Lynch, J. L. Lyon, D. P. Sandler, J. B. Schoenberg, D. J. Steck, J. A. Stolwijk, C. Weinberg, and H. B. Wilcox, "A combined analysis of north american case-control studies of residential radon and lung cancer," *J Toxicol Environ Health A*, vol. 69, no. 7, pp. 533–597, Apr 2006. 9

[16] P. S. R. Soares, "Determinação da concentração de radão num estabelecimento de ensino público," Master's thesis, Universidade da Beira Interior, 2011. 9

[17] United Nations Scientific Committee on the Effects of Atomic Radiation (UNSCEAR), *Report to the General Assembly: Scientific Annex A. Effects of ionizing radiation*, 1st ed. New York: United Nations, 2008, sales No. E.08.IX.6, ISBN: 978-92-1-142263-4. 9

[18] J. Tong, L. Qin, Y. Cao, J. Li, J. Zhang, J. Nie, and Y. An, "Environmental radon exposure and childhood leukemia," *Journal of Toxicology and Environmental Health, Part B*, vol. 15, no. 5, pp. 332–347, 2012, pMID: 22852813. [Online]. Available: https://doi.org/10.1080/10937404.2012.689555 9

[19] US Environmental Protection Agency, *A Citizen's Guide to Radon: The Guide to Protecting Yourself and Your Family from Radon*. Washington, DC: US Environmental Protection Agency, 2012. 9

[20] "Council directive 2013/59/euratom of 5 december 2013 laying down basic safety standards for protection against the dangers arising from exposure to ionising radiation, and repealing directives 89/618/euratom, 90/641/euratom, 96/29/euratom, 97/43/euratom and 2003/122/euratom," https://eur-lex.europa.eu/eli/dir/2013/59/oj, 2014, official Journal of the European Union, L 13, 1-73. 9

[21] M. Faisca, M. Teixeira, and A. Bettencourt, "Indoor Radon Concentrations in Portugal - A National Survey," *Radiation Protection Dosimetry*, vol. 45, no. 1-4, pp. 465–467, 12 1992. [Online]. Available: https://doi.org/10.1093/rpd/45.1-4.465 10

[22] B. Veloso, J. R. Nogueira, and M. F. Cardoso, "Lung cancer and indoor radon exposure in the north of portugal – an ecological study," *Cancer Epidemiology*, vol. 36, no. 1, pp. e26–e32, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877782111001561 11

[23] "Decreto-lei n.º 118/2013," https://diariodarepublica.pt/dr/detalhe/decreto-lei/118-2013-499237, 2013, Accessed: January 18, 2024. 11

[24] "Portaria n.º 353-a/2013," https://diariodarepublica.pt/dr/detalhe/portaria/353-a-2013-331868, 2013, Accessed: January 18, 2024. 11

[25] "Decreto-lei n.º 118/2013 de 20 de agosto," August 2013, Accessed: December 21, 2023. 11

[26] P. T. Branco, L. Martin-Gisbert, J. P. Sá, A. Ruano-Raviña, J. Barros-Dios, L. Varela-Lema, and S. I. Sousa, "Quantifying indoor radon levels and determinants in schools: A case study in the radon-prone area galicia–norte de portugal euroregion," *Science of The Total Environment*, vol. 882, p. 163566, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S004896972302185X 11

[27] Eurostat, "9% of eu population unable to keep home warm in 2022," 2023, accessed: 2024-01-23. [Online]. Available: https://ec.europa.eu/eurostat/web/products-eurostat-news/-/ddn-20230117-1 12

[28] European Economic and Social Committee, "Energy poverty – 42 million people in the eu cannot afford to heat their homes adequately," 2023, accessed: 2024-01-23. [Online]. Available: https://www.eesc.europa.eu/en/news-media/news/energy-poverty-42-million-people-eu-cannot-afford-heat-their-homes-adequately 12

[29] S. Architecture and C. Portal. (2023) Results of the survey conducted in mainland portugal and islands on thermal comfort at home. Accessed on 23/01/2024. [Online]. Available: https://www.csustentavel.com/wp-content/uploads/2023/04/resultados_inquerito-2023.pdf 12

[30] L. C. Council and P. C. Council. Study on energy poverty. Accessed on 23/01/2024. [Online]. Available: https://pobrezaenergetica.pt/ 12

[31] R. A. Paula Santana, Helena Nogueira, *A avaliação multidimensional da saúde da população: o caso do município de Estarreja*. Coimbra, Portugal: Imprensa da Universidade de Coimbra, 2017, pp. 219–238. 12

[32] H. N. Ricardo Almendra, Paula Santana, "Evidence of social deprivation on the spatial patterns of excess winter mortality in portugal," *International Journal of Environmental Research and Public Health*, vol. 14, no. 6, p. 637, 2017. [Online]. Available: https://www.mdpi.com/1660-4601/14/6/637 12

[33] A. Campos. (2018, January) Risco de morrer é maior no inverno. Accessed on 23/01/2024. [Online]. Available: https://www.publico.pt/2018/01/22/sociedade/noticia/risco-de-morrer-e-maior-no-inverno-1800132 12

[34] J. Fitzner, A. Tjon-Kon-Fat, and R. Brown, “Impact of cold ambient temperature in the pattern of influenza virus,” *Open Forum Infectious Diseases*, vol. 10, no. 2, p. ofad039, 2023. [Online]. Available: https://academic.oup.com/ofid/article/10/2/ofad039/7008468 12

[35] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, D. Niyato, O. Dobre, and H. V. Poor, “6g internet of things: A comprehensive survey,” *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 359–383, 2022. 12, 14

[36] B. M. C. Silva, J. J. P. C. Rodrigues, A. Ramos, K. Saleem, I. de la Torre, and R. L. Rabêlo, “A mobile health system to empower healthcare services in remote regions,” in *2019 IEEE International Conference on E-health Networking, Application Services (HealthCom)*, 2019, pp. 1–6. 13

[37] INE - Instituto Nacional de Estatística. [Online]. Available: https://www.ine.pt/ 13

[38] NaturalPT. Paisagem protegida regional da serra da gardunha. [Online]. Available: https://natural.pt/protected-areas/paisagem-protegida-regional-serra-gardunha?locale=pt 13

[39] Federação Portuguesa do Caminho de Santiago. Município de pinhel. [Online]. Available: https://fpcsantiago.pt/municipio-de-pinhel/ 13

[40] 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. 14

[41] H. Tran-Dang, N. Krommenacker, P. Charpentier, and D.-S. Kim, “Toward the internet of things for physical internet: Perspectives and challenges,” *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4711–4736, 2020. 14

[42] (2017) Smart homes market size, share| future, trends estimate 2025. Accessed on 27/12/2023. [Online]. Available: http://www.transparencymarketresearch.com/smart-homes-market.html 14

[43] J. Lynch and K. Loh, “A summary review of wireless sensors and sensor networks for structural health monitoring,” *The Shock and Vibration Digest*, vol. 38, pp. 91–128, 03 2006. 14

[44] A. Rayes and S. Salam, *The Things in IoT: Sensors and Actuators*. Cham: Springer International Publishing, 2019, pp. 67–87. [Online]. Available: https://doi.org/10.1007/978-3-319-99516-8_3 15

[45] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, pp. 436–444, 05 2015. 16, 18

[46] G. De Vito, “Assessing healthcare software built using iot and llm technologies,” in *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE ’24. New York, NY, USA: Association for

Computing Machinery, 2024, p. 476–481. [Online]. Available: https://doi.org/10.1145/3661167.3661202 16

[47] G. T. Wilson, "Time Series Analysis: Forecasting and Control, 5th Edition , by George E. P. Box , Gwilym M. Jenkins , Gregory C. Reinsel and Greta M. Ljung , 2015 . Published by John Wiley and Sons Inc. , Hoboken, N," *Journal of Time Series Analysis*, vol. 37, no. 5, pp. 709–711, September 2016. [Online]. Available: https://ideas.repec.org/a/bla/jtsera/v37y2016i5p709-711.html 16

[48] G. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231201007020 16, 19

[49] M. Méndez, M. G. Merayo, and M. Núñez, "Machine learning algorithms to forecast air quality: a survey," *Artif. Intell. Rev.*, vol. 56, no. 9, p. 10031–10066, feb 2023. [Online]. Available: https://doi.org/10.1007/s10462-023-10424-4 17

[50] S. Tiwari, "Supervised machine learning: A brief introduction," 12 2022, pp. 219–230. 17

[51] M. Zhang, *Unsupervised Learning Algorithms in Big Data: An Overview*, 01 2022, pp. 910–931. 17

[52] S. Mousavi, M. Schukat, and E. Howley, "Deep reinforcement learning: An overview," 06 2018, pp. 426–440. 17

[53] A. Yafouz, A. N. Ahmed, N. Zaini, and A. El-Shafie, "Ozone concentration forecasting based on artificial intelligence techniques: A systematic review," *Water, Air, & Soil Pollution*, vol. 232, no. 2, p. 79, Feb. 2021. [Online]. Available: https://doi.org/10.1007/s11270-021-04989-5 17

[54] S. Das, *Statistical Analysis of Environmental Data*. Cham: Springer International Publishing, 2023, pp. 183–195. [Online]. Available: https://doi.org/10.1007/978-3-031-42137-2_9 17

[55] Q. Zhou, H. Jiang, J. Wang, and J. Zhou, "A hybrid model for pm2.5 forecasting based on ensemble empirical mode decomposition and a general regression neural network," *Science of The Total Environment*, vol. 496, pp. 264–274, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S004896971401081X 17

[56] A. Russo, F. Raischel, and P. G. Lind, "Air quality prediction using optimal neural networks with stochastic variables," *ATMOSPHERIC ENVIRONMENT*, vol. 79, pp. 822–830, 2013. 17

[57] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, nov 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735 18, 69

[58] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014. 18

[59] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Stacked bidirectional and unidirectional lstm recurrent neural network for forecasting network-wide traffic state with missing values," *Transportation Research Part C: Emerging Technologies*, vol. 118, p. 102674, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X20305891 18, 86

[60] N. Ding, C. Benoit, G. Foggia, Y. Bésanger, and F. Wurtz, "Neural network-based model design for short-term load forecast in distribution systems," *IEEE Transactions on Power Systems*, vol. 31, no. 1, pp. 72–81, 2016. 18

[61] B. T. Ong, K. Sugiura, and K. Zettsu, "Dynamic pre-training of deep recurrent neural networks for predicting environmental monitoring data," in *2014 IEEE International Conference on Big Data (Big Data)*, 2014, pp. 760–765. 18

[62] R. Masini, M. Medeiros, and E. Mendes, "Machine learning advances for time series forecasting," *Journal of Economic Surveys*, vol. 37, no. 1, pp. 76–111, 2021, funding Information: We are very grateful for the insightful comments made by two anonymous referees. The second author gratefully acknowledges the partial financial support from CNPq. We are also grateful to Francis X. Diebold, Daniel Borup, and Andrii Babii for helpful comments. Publisher Copyright: © 2021 John Wiley Sons Ltd. 19

[63] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002, nonlinear Methods and Data Mining. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167947301000652 19

[64] J. Crespo Cuaresma, J. Hlouskova, S. Kossmeier, and M. Obersteiner, "Forecasting electricity spot-prices using linear univariate time-series models," *Applied Energy*, vol. 77, no. 1, pp. 87–106, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261903000965 19

[65] M. Ahmed, A. Cook, U. of Oklahoma. School of Civil Engineering, and E. Science, *Analysis of Freeway Traffic Time Series Data Using Box and Jenkins Techniques*, 1979. [Online]. Available: https://books.google.pt/books?id=7Do8HQAACAAJ 19

[66] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *International Journal of Forecasting*, vol. 37, no. 1, p. 388–427, Jan. 2021. [Online]. Available: http://dx.doi.org/10.1016/j.ijforecast.2020.06.008 19

[67] R. Singhal, N. K. Choudhary, and N. Singh, "Short-term load forecasting using hybrid arima and artificial neural network model," in *Advances in VLSI, Communication,*

and Signal Processing, D. Dutta, H. Kar, C. Kumar, and V. Bhadauria, Eds. Singapore: Springer Singapore, 2020, pp. 935–947. 19

[68] Q. Tao, F. Liu, Y. Li, and D. Sidorov, "Air pollution forecasting using a deep learning model based on 1d convnets and bidirectional gru," *IEEE Access*, vol. 7, pp. 76 690–76 698, 2019. 19

[69] B. Predić, Z. Yan, J. Eberle, D. Stojanovic, and K. Aberer, "Exposuresense: Integrating daily activities with air quality using mobile participatory sensing," in *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2013, pp. 303–305. 20, 25

[70] A. Kadri, E. Yaacoub, M. Mushtaha, and A. Abu-Dayya, "Wireless sensor network for real-time air pollution monitoring," in *2013 1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, 2013, pp. 1–5. 20, 25

[71] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014. 21, 25

[72] Y. Zheng, X. Chen, Q. Jin, Y. Chen, X. Qu, X. Liu, E. Chang, W.-Y. Ma, Y. Rui, and W. Sun, "A cloud-based knowledge discovery system for monitoring fine-grained air quality," Tech. Rep. MSR-TR-2014-40, March 2014. [Online]. Available: https://www.microsoft.com/en-us/research/publication/a-cloud-based-knowledge-discovery-system-for-monitoring-fine-grained-air-quality/ 21, 25, 26

[73] F. Pereira, S. I. Lopes, N. B. Carvalho, and A. Curado, "Rnprobe: A lora-enabled iot edge device for integrated radon risk management," *IEEE Access*, vol. 8, pp. 203 488–203 502, 2020. 21, 25, 26

[74] S. Dhingra, R. B. Madda, A. H. Gandomi, R. Patan, and M. Daneshmand, "Internet of things mobile–air pollution monitoring system (iot-mobair)," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5577–5584, 2019. 22, 25, 26

[75] S. Fang, L. D. Xu, Y. Zhu, J. Ahati, H. Pei, J. Yan, and Z. Liu, "An integrated system for regional environmental monitoring and management based on internet of things," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1596–1605, 2014. 22

[76] S. R. Garzon, S. Walther, S. Pang, B. Deva, and A. Küpper, "Urban air pollution alert service for smart cities," in *Proceedings of the 8th International Conference on the Internet of Things*, ser. IOT '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3277593.3277599 22, 25

[77] J. Vanus, J. Machac, R. Martinek, P. Bilik, J. Zidek, J. Nedoma, and M. Fajkus, "The design of an indirect method for human presence monitoring in intelligent buildings," *Human-centric Computing and Information Sciences*, vol. 8, no. 1, p. 28, 2018. [Online]. Available: https://doi.org/10.1186/s13673-018-0151-8 22, 25, 26

[78] J. Vanus, O. Majidzadeh Gorjani, P. Dvoracek, P. Bilik, and J. Koziorek, "Application of a new co2 prediction method within family house occupancy monitoring," *IEEE Access*, vol. 9, pp. 158 760–158 772, 2021. 22

[79] R. Ma, N. Liu, X. Xu, Y. Wang, H. Y. Noh, P. Zhang, and L. Zhang, "Fine-grained air pollution inference with mobile sensing systems: A weather-related deep autoencoder model," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 2, jun 2020. [Online]. Available: https://doi.org/10.1145/3397322 22

[80] A. Tunyagi, T. Dicu, A. Cucos, B. D. Burghele, G. Dobrei, A. Lupulescu, M. Moldovan, D. Nita, B. Papp, I. Pap, K. Szacsvai, A. Tenter, M. Beldean-Galea, M. Anton, S. Grecu, L. Cioloca, R. Milos, M. L. Botos, C. G. Chiorean, T. Catalina, M. A. Istrate, and C. Sainz Fernández, "An innovative system for monitoring radon and indoor air quality," 2020. [Online]. Available: http://hdl.handle.net/10902/21222 22, 25, 26

[81] J. Gabrys, H. Pritchard, and B. Barratt, "Just good enough data: Figuring data citizenships through air pollution sensing and data stories," *Big Data & Society*, vol. 3, no. 2, p. 2053951716679677, 2016. [Online]. Available: https://doi.org/10.1177/2053951716679677 23, 25

[82] IQAir. (2024) Iqair - airvisual series. [Online]. Available: https://www.iqair.com/air-quality-monitors 23, 26

[83] UbiBot. (2024) Ubibot - iot platform. [Online]. Available: https://www.ubibot.com/ 23, 26

[84] Airthings. (2024) Wave: understanding the connectivity and compatibility of your wave. [Online]. Available: https://help.airthings.com/en/articles/6009826-wave-understanding-the-connectivity-and-compatibility-of-your-wave 23, 26, 58

[85] RadonFTLab. (2024) Continuous radon detector - radoneye. [Online]. Available: http://radonftlab.com/radon-sensor-product/radon-detector/ 23, 26

[86] W. Hu, Y. Wen, K. Guan, G. Jin, and K. J. Tseng, "itcm: Toward learning-based thermal comfort modeling via pervasive sensing for smart buildings," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4164–4177, 2018. 24

[87] M. Bhatia, "Intelligent system of game-theory-based decision making in smart sports industry," *ACM Trans. Intell. Syst. Technol.*, vol. 12, no. 3, apr 2021. [Online]. Available: https://doi.org/10.1145/3447986 24

[88] Amazon. (2024) Aws iot - internet of things platform. [Online]. Available: https://aws.amazon.com/pt/iot/ 24

[89] O. Racine Ly, K. Gueye, S. Ouya, and G. Mendy, "Collaborative platform for supervision and advice of agricultural engineer to farmers thanks to iot and webrtc," in *Proceedings of the 9th International Conference on Computer and Communications Management*, ser. ICCCM '21. New York, NY, USA:

Association for Computing Machinery, 2021, p. 143–148. [Online]. Available: https://doi.org/10.1145/3479162.3479184 24

[90] O. A. Postolache, J. M. Dias Pereira, and P. M. B. Silva Girao, "Smart sensors network for air quality monitoring applications," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 9, pp. 3253–3262, 2009. 25

[91] C. C. Aggarwal, *Data Mining: The Textbook*. Springer Publishing Company, Incorporated, 2015. 57

[92] Ángel Rodés, "Radonsim," 2022, accessed: 20-03-2024. [Online]. Available: https://github.com/angelrodes/RadonSim 58

[93] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 2nd ed. Australia: OTexts, 2018. 62

[94] R. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018. [Online]. Available: https://books.google.pt/books?id=_bBhDwAAQBAJ 62, 66, 74

[95] A. Geron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. O'Reilly Media, Inc., 2019. 65

[96] S. Arlot, Alain and Celisse, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, no. none, Jan. 2010. [Online]. Available: http://dx.doi.org/10.1214/09-SS054 65

[97] S. Arlot and A. Celisse, "A survey of cross validation procedures for model selection," *Statistics Surveys*, vol. 4, 07 2009. 66

[98] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015. 68

[99] O. Triebe, N. Laptev, and R. Rajagopal, "Ar-net: A simple auto-regressive neural network for time-series," 2019. [Online]. Available: https://arxiv.org/abs/1911.12436 68

[100] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," *CoRR*, vol. abs/1502.02367, 2015. [Online]. Available: http://arxiv.org/abs/1502.02367 71

[101] J. H. Friedman, "Greedy function approximation: A gradient boosting machine." *The Annals of Statistics*, vol. 29, no. 5, pp. 1189 – 1232, 2001. [Online]. Available: https://doi.org/10.1214/aos/1013203451 71

[102] S. R. Karingula, N. Ramanan, R. Tahmasbi, M. Amjadi, D. Jung, R. Si, C. Thimmisetty, L. P. Cabrera, M. Sayer, and C. N. C. J. au2, "Boosted embeddings for time series forecasting," 2021. [Online]. Available: https://arxiv.org/abs/2104.04781 72

[103] L. Prechelt, *Early Stopping - But When?* Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 55–69. [Online]. Available: https://doi.org/10.1007/3-540-49430-8_3 73

[104] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org. 73

[105] J. Brownlee, *Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions*. Machine Learning Mastery, 2018. [Online]. Available: https://books.google.pt/books?id=T1-nDwAAQBAJ 73

[106] M. Abadi, A. Agarwal, P. Barham *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous systems," https://www.tensorflow.org/, 2015, software available from tensorflow.org. 73

[107] F. Chollet, "Keras," https://keras.io, 2015. 73

[108] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The m4 competition: 100,000 time series and 61 forecasting methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020, m4 Competition. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169207019301128 74

[109] D. Chicco, M. J. Warrens, and G. Jurman, "The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation," *PeerJ Computer Science*, vol. 7, p. e623, 2021. [Online]. Available: https://doi.org/10.7717/peerj-cs.623 75

[110] C. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate Research*, vol. 30, p. 79, 12 2005. 75, 76

[111] J. F. Kenney and E. S. Keeping, *Mathematics of Statistics*, 3rd ed. Princeton: Van Nostrand, 1964, vol. 1, ch. 4.15 Root Mean Square, pp. 59–60. 75, 76

[112] S. Makridakis, "Accuracy measures: theoretical and practical concerns," *International Journal of Forecasting*, vol. 9, no. 4, pp. 527–529, 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0169207093900793 76

[113] J. Turnbull, *The Docker Book: Containerization Is the New Virtualization*. James Turnbull, 2014. [Online]. Available: https://books.google.pt/books?id=4xQKBAAAQBAJ 78

[114] phpMyAdmin Development Team, "phpMyAdmin Documentation," https://docs.phpmyadmin.net/, n.d., accessed: 2023-10-10. 78

Appendix A

Appendix

A.1 Extended Analysis

A.1.0.1 ACF and PACF Graphs

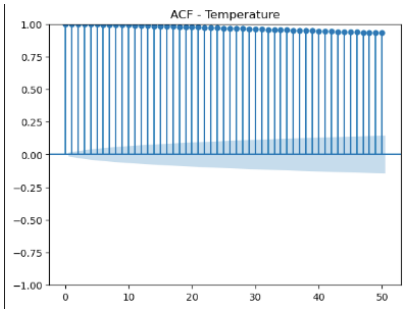

Figure A.1: Temperature ACF.

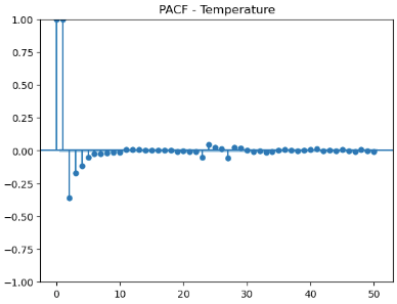

Figure A.2: Temperature PACF.

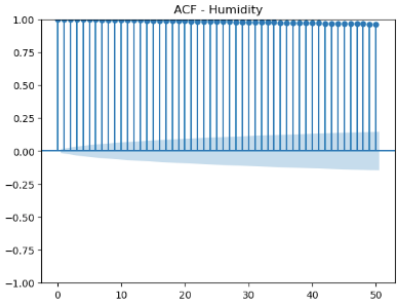

Figure A.3: Humidity ACF.

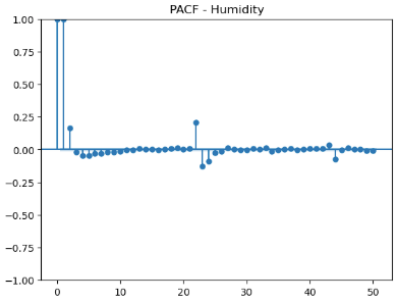

Figure A.4: Humidity PACF.

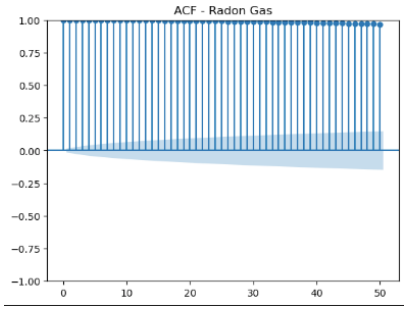

Figure A.5: Radon ACF.

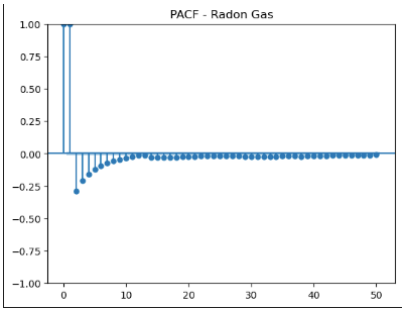

Figure A.6: Radon PACF.

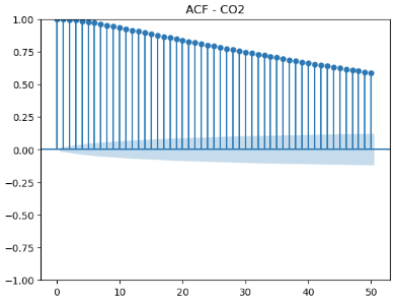

Figure A.7: CO2 ACF.

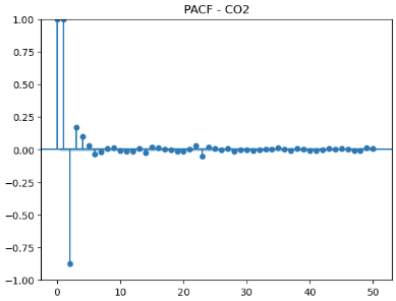

Figure A.8: CO2 PACF.