Instituto Superior de Engenharia

Politécnico de Coimbra

DEPARTMENT OF SYSTEMS AND COMPUTER ENGINEERING

Smart Cabinet for Small Agricultural Products Using Low-Cost IoT and Open-Source Technologies

To fulfill the Master's degree in Informatics Engineering

Specialization in Software Engineering

Author

José Ricardo da Cunha Guimarães

Supervisors

Acácio Manuel Raposo Amaral

Filipe Alexandre Almeida Ningre de Sá

INSTITUTO POLITÉCNICO DE COIMBRA

INSTITUTO SUPERIOR DE ENGENHARIA DE COIMBRA

Coimbra, December 2024

Acknowledgment

I would like to express my deepest gratitude to the Coimbra Institute of Engineering and its dedicated professors, who have played a pivotal role in shaping my academic journey, from my bachelor's degree to the pursuit of my master's in Computer Engineering. Their commitment to excellence in teaching and mentorship has provided the foundation for my professional and personal growth.

A special acknowledgment goes to my supervisors, Doctor Filipe Sá and Doctor Acácio Amaral, whose invaluable guidance, constructive feedback and determined support were crucial throughout this project. Their expertise and encouragement inspired me to overcome challenges and achieve meaningful results.

I would also like to extend my sincere thanks to André Aleixo, representative of *Verdes do Mondego*, whose profound expertise in biology and microgreens cultivation was instrumental in the validation of this system. His invaluable insights and practical experience in the field not only ensured the system's relevance but also significantly contributed to its alignment with real-world needs and standards.

I am also profoundly thankful to João Gonçalves and Pedro Batalha, both graduates in Computer Engineering, for their insightful feedback during the system's development, particularly in refining the mobile application. Their contributions significantly enhanced the quality and usability of the final product, ensuring its success.

Finally, I owe my deepest appreciation to my family, with a special mention to my parents, Maria Cunha e António Guimarães, and my sister, Mafalda Guimarães. Their persistent belief in me, constant encouragement and emotional support have been a source of strength throughout this journey. Their love and resilience made it possible for me to persevere and complete this work.

To everyone who has contributed to this journey in any way, thank you for being a part of this milestone in my life.

Abstract

The integration of Internet of Things (IoT) technologies into agriculture has opened new opportunities to enhance productivity and sustainability, particularly in small-scale and domestic farming operations. This thesis describes the development of a prototype for a smart farming system designed to automate plant cultivation through the integration of IoT systems and intuitive applications, providing easy use for any user with a smartphone. The primary objective is to enable an efficient and accessible solution by leveraging open-source technologies and low-cost materials. Using an agile methodology and in collaboration with *Verdes do Mondego*, a Coimbra-based company specializing in indoor microgreens production, the development process focused on creating a prototype for a smart agricultural cabinet. The system's architecture integrates hardware components such as microcontrollers, sensors and actuators with software solutions, including a system orchestrator and a dual-database strategy. The system's core functionalities include real-time environmental monitoring through mobile and web applications and the control of critical factors for plant growth, such as temperature, humidity, ventilation and irrigation. Additionally, leveraging the capabilities of MongoDB, the system stores historical data, enabling users to analyze trends and make data-driven decisions to improve plant health and growth. The results demonstrate the potential of IoT-based systems to improve efficiency and sustainability in agriculture. By automating critical processes and providing actionable insights, the smart agricultural cabinet reduces the need for human intervention, optimizes the use of vital resources like water, and promotes healthy plant growth. This project contributes to the growing field of smart agriculture by presenting a practical, economical, and scalable solution for domestic agricultural applications. Thus, this final project of the Master's in Computer Engineering presents a prototype of the agricultural cabinet as a proof of concept, serving as a demonstration and validation of the research conducted.

Keywords: Smart Farming, Internet of Things, Sensors, Actuators, NoSQL Databases

Resumo

A integração de tecnologias de Internet das Coisas (IoT) na agricultura abriu novas oportunidades para melhorar a produtividade e a sustentabilidade, particularmente em operações agrícolas de pequena escala e domésticas. Esta tese descreve o desenvolvimento de um protótipo de um sistema de agricultura inteligente projetado para automatizar o cultivo de plantas através da integração de sistemas IoT e aplicações intuitivas, proporcionando uma fácil utilização por qualquer utilizador com um smartphone. O objetivo principal é viabilizar uma solução eficiente e acessível, utilizando tecnologias open-source e materiais de baixo custo. Utilizando uma metodologia ágil e em colaboração com a *Verdes do Mondego*, uma empresa de Coimbra especializada na produção de microvegetais em ambiente *indoor*, o processo de desenvolvimento concentrou-se na criação de um protótipo de um armário de agricultura inteligente. A arquitetura do sistema integra componentes de hardware, como microcontroladores, sensores e atuadores, com soluções de *software*, incluindo um orquestrador de sistema e uma estratégia de base de dados dupla. As principais funcionalidades do sistema incluem a monitorização ambiental em tempo real através de aplicações móveis e *web* e o controlo de fatores críticos para o crescimento das plantas, como a temperatura, humidade, ventilação e irrigação. Adicionalmente, com as capacidades do MongoDB, o sistema armazena dados históricos, permitindo aos utilizadores analisar tendências e tomar decisões baseadas em dados para melhorar a saúde e o crescimento das plantas. Os resultados demonstram o potencial dos sistemas baseados em IoT para melhorar a eficiência e a sustentabilidade na agricultura. Ao automatizar processos críticos e fornecer informações relevantes e aplicáveis, o armário agrícola inteligente reduz a necessidade de intervenção humana, otimiza o uso de recursos vitais, como a água, e promove o crescimento saudável das plantas. Este projeto contribui para o crescente campo da agricultura inteligente ao apresentar uma solução prática, económica e escalável para aplicações agrícolas domésticas. Assim, este projeto final do Mestrado em Engenharia Informática apresenta um protótipo do armário como prova de conceito, servindo para efeitos de demonstração e validação da investigação realizada.

Palavras-chave: Agricultura Inteligente, Internet das Coisas, Sensores, Atuadores, Bases de Dados NoSQL

Index

Index of Tables

Index of Figures

List of Acronyms and Abbreviations

ADC	Analog-to-Digital Converter
DB	Database
DBMS	Database Management System
EC	Electrical Conductivity
ER	Entity-Relationship
GSM	Global System for Mobile Telecommunications
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
ISEC	*Instituto Superior de Engenharia de Coimbra*
I/O	Input/Output
K	Potassium
LCD	Liquid Crystal Display
LDR	Light Dependent Resistor
ML	Machine Learning
MQTT	Message Queuing Telemetry Transport
NPK	Nitrogen, Phosphorus, Potassium
N	Nitrogen
P	Phosphorus
RDBMS	Relational Database Management System
SQL	Structured Query Language
TDS	Total Dissolved Solids
UI	User Interface
UEQ	User Experience Questionnaire
UX	User Experience

List of Symbols

°C	Degrees Celsius
μmol $m^{-2}s^{-1}$	Micromoles Of Photons Per Square Meter Per Second

1 Introduction

According to the United Food and Agriculture Organization, agricultural production will need to increase by approximately 75% to sustain the global population by 2050 [1]. To support this growth, the concept of Smart Agriculture has emerged. Smart agriculture is an evolving field that utilizes technological innovations to transform traditional farming practices. The integration of digital technologies within agricultural practices has opened up new opportunities and possibilities, revolutionizing the way farmers manage their crops, resources and operations [2]. The increasing integration of Internet of Things (IoT) technologies into various domains has revolutionized how humans interact with systems. Agriculture, a crucial sector for food security and sustainability, has embraced these technologies to tackle the challenges of resource limitations and the growth of urban areas. This thesis presents the design and development of a Smart Agricultural Cabinet, a prototype aimed at automating plant cultivation for small-scale and domestic use.

The proposed system integrates low-cost IoT components, open-source technologies and mobile and web applications to manage the system. Its core functionalities include real-time environmental monitoring, automated irrigation and ventilation, all managed through mobile and web applications.

The project was conceived with the support of the *Verdes do Mondego*, a company based in Coimbra specializing in microgreens cultivation. This partnership played a key role in the system design, enabling the definition of the system requirements that incorporate the stakeholders' perspective. By adopting an agile development methodology, the project ensured adaptability and iterative refinement throughout its lifecycle.

This work is built within a large research in IoT and smart farming, aiming to contribute a sustainable solution to indoor agriculture. By integrating technologies with a focus on affordability and ease of use, the Smart Agricultural Cabinet represents a step forward in making precision farming techniques available to broader audiences, particularly in urban environments where traditional farming is often impractical [3].

This document details the development of a Smart Agricultural Cabinet, created within the scope of the Project curricular unit, as part of the Master's program in Computer Engineering at the Instituto Superior de Engenharia de Coimbra (ISEC). The following sections of this chapter aim to introduce the context and objectives of this project, the development methodology applied and the document structure, laying a foundation for the detailed discussions in subsequent chapters.

1.1 Coimbra Institute of Engineering

The Coimbra Institute of Engineering (Instituto Superior de Engenharia de Coimbra - ISEC) [4], was established in 1974, evolving from the former Coimbra Industrial and Commercial Institute, which dates back to 1921. Over the decades, ISEC has cemented its position as a leader in engineering education, consistently adapting to technological advancements and societal needs.

ISEC is dedicated to its mission of fostering the creation, dissemination, and application of knowledge in the fields of science, technology, and engineering. The institution strives to provide an education that balances academic excellence with practical applicability, ensuring its graduates are well-prepared to meet the demands of the modern workforce. With a focus on innovation and global competitiveness, ISEC has established itself as a center of distinction both nationally and internationally.

The academic offerings at ISEC include a broad range of undergraduate and postgraduate programs, spanning disciplines such as Civil, Mechanical, Electrical and Computer Engineering. The Master's program in Computer Science, under which this project was developed, reflects the institution's commitment to advancing technical expertise and innovation. Specifically, this project aligns with the specialization in Software Engineering, a branch aimed at equipping students with the skills necessary to design, develop and maintain software systems. On the other hand, the skills provided by the study program in Informatics Engineering were highly valuable. These skills encompassed expertise in Software Integration and Development, with a particular emphasis on mobile and web application development, along with knowledge in electronics, communication networks and protocols, and information security.

ISEC places significant emphasis on fostering partnerships with industries, enabling students to engage in real-world applications of their studies. This collaborative approach not only enriches the academic experience but also facilitates the transition from education to professional practice.

1.2 *Verdes do Mondego*

Verdes do Mondego is a company based in Coimbra dedicated to growing microgreens using vertical farming techniques. Founded by André Aleixo, a Biology student at the University of Coimbra, the company produces over twenty types of microgreens, including radish, arugula, broccoli, coriander and chives [5].

The company uses controlled-environment agricultural techniques without pesticides to produce microgreens with high nutritional value. These products are sold weekly at local markets, such as the Coimbra Municipal Market and supplied to approximately eighteen restaurants in the Coimbra region.

They has been involved in research projects and collaborations with the University of Coimbra, emphasizing sustainable farming practices and innovative approaches. Looking ahead, *Verdes do Mondego* plans to expand into new facilities, where visitors will be welcome to learn more about its work and its mission to bring farming closer to the community. The company aims to establish itself as a national reference in microgreen production by integrating traditional agricultural methods with modern, sustainable practices.

The implementation of a smart agricultural cabinet could be particularly interesting for *Verdes do Mondego*, as it can automate their system, improving efficiency and consistency in production while reducing manual effort. This aligns with their commitment to innovation and sustainability, enhancing their ability to meet the increasing demand and maintain good quality in microgreen cultivation.

1.3 Project Context

The rapid advancement of IoT technologies has transformed multiple sectors, especially agriculture. By facilitating seamless interconnectivity and enabling real-time data exchange, IoT has revolutionized the management and optimization of various systems [6]. In agriculture, innovations caused by IoT offer solutions to overcome challenges such as resource inefficiency and the spatial constraints of urban environments.

Urban environments, where traditional farming is often impractical due to limited space and resources, stand to benefit significantly from such solutions [7]. The increasing global emphasis on sustainability and efficient use of resources highlights the importance of innovative approaches like this one. By leveraging controlled environments and applying smart farming principles, this project seeks to make advanced agricultural practices accessible for small-scale agricultural production.

This project aims to design and develop a smart agricultural cabinet specifically for domestic and small-scale use. Using low-cost IoT components and open-source technologies, the system automates environmental control while integrating mobile and web applications for real-time monitoring and management of plant growth. The primary goal is to minimize human intervention while ensuring optimal conditions for small-scale agricultural production, making it particularly helpful for users with limited time, space or resources.

1.4 Problem Definition

Agriculture is the foundation of global food security, but it faces numerous challenges that are becoming increasingly difficult to tackle. Climate change, rapid urbanization and limited resources are putting significant pressure on traditional farming practices.

With the world's population expected to grow and food demand likely to rise substantially by 2050, these issues are testing the limits of agriculture [7]. Irregular weather patterns and inefficient resource use not only have serious economic implications but also pose significant environmental risks.

The incorporation of IoT technologies in agriculture has opened up new opportunities for farmers to closely monitor and manage environmental factors, leading to enhanced yields and more efficient resource utilization. However, factors such as connectivity troubles, device compatibility difficulties and cybersecurity risks present significant obstacles [8].

In the study conducted in [9], the authors highlight the primary challenges associated with the adoption of IoT in smart farming, as depicted in Figure 1.1. They point out that, in addition to the high costs related to implementing IoT and its energy consumption, there are significant security and privacy concerns. The integration of advanced hardware and software elevates the risk of cyberattacks and unauthorized access, highlighting the necessity for the development of robust safeguarding protocols.

They also underline system scalability challenges, as the growing number of IoT devices generates vast quantities of real-time data, requiring seamless integration and refined analysis to maintain system performance. Additionally, the complexity of IoT systems often demands a level of technical expertise that many users may lack, highlighting the importance of developing intuitive and user-friendly solutions to enhance accessibility and facilitate wider adoption among farmers.

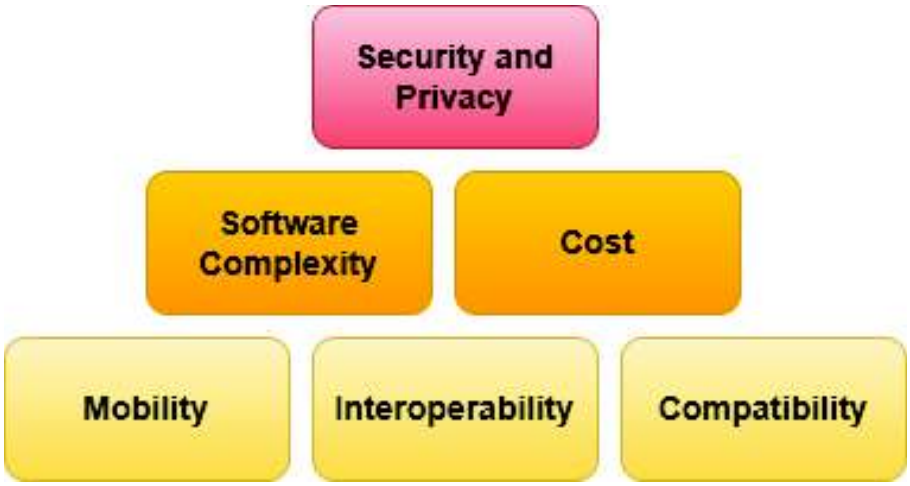


Figure 1.1: Challenges of IoT in smart farms presented by the authors in [9]

This project aims to bridge that gap. By developing a smart agricultural cabinet designed specifically for small-scale and domestic agriculture, it seeks to provide a practical and accessible solution. Using low-cost IoT components and open-source technologies, the cabinet automates essential agricultural tasks, making advanced farming techniques available to a wider audience.

1.5 Objectives

The objectives of this project are structured to ensure the successful design, implementation, and validation of a smart agricultural cabinet that meets the challenges outlined in Section 1.4. The project's goals are divided into two main categories: general objectives, which outline the overarching aim of the system, and specific objectives, which define the technical and functional requirements necessary to achieve this purpose.

1.5.1 General Objectives

The objective of this project is to create a prototype of a smart agricultural cabinet, which seeks to resolve the issues specified in Section 1.4. This will be achieved by combining user-friendly technologies to enable real-time monitoring, automate environmental control, and facilitate straightforward management. The cabinet is designed to provide a practical and cost-effective solution for domestic agriculture, bridging the gap between advanced precision farming and its adaptation for small-scale users.

At its core, the project seeks to implement a system capable of autonomously managing the entire crop lifecycle with minimal human intervention. By leveraging IoT technologies, the cabinet continuously monitors and optimizes essential environmental parameters, including soil moisture, temperature, humidity, and lighting. The emphasis on affordability and usability ensures that the system remains accessible to non-expert users, demonstrating how sophisticated agricultural techniques can be seamlessly adapted for sustainable and efficient home farming.

1.5.2 Specific Objectives

To achieve the general objective, the project delineates specific objectives that address the technical and functional requirements of the system. As illustrated in Figure 1.2, the proposed system's architecture, created based on researches, integrates various components to achieve seamless functionality. A modular hardware infrastructure is central to the design, incorporating sensors, actuators and controllers to monitor and manage environmental parameters with precision. Each section of the cabinet is equipped with dedicated microcontrollers, connected to a centralized orchestrator that ensures efficient coordination and real-time data processing.

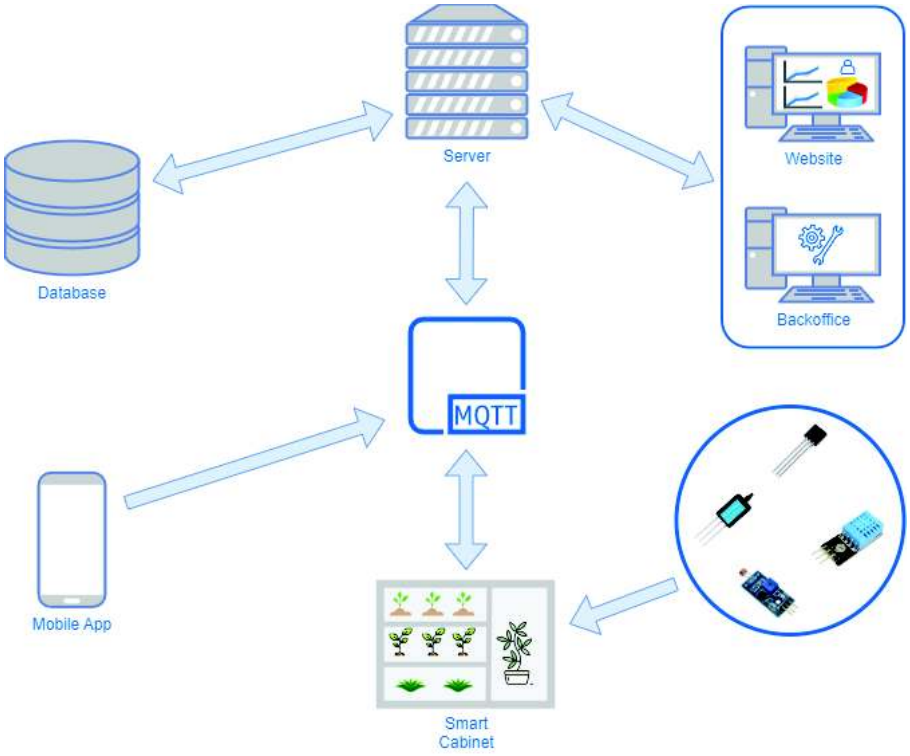


Figure 1.2: Proposed concept

The system also features a mobile application to enable users to monitor and control the cabinet in real-time. This application is complemented by a web-based dashboard, which provides detailed visualizations of collected data, allowing users to track trends and make informed decisions. Furthermore, the system incorporates automated control mechanisms to optimize irrigation, ventilation, and lighting cycles based on real-time sensor inputs, enhancing both efficiency and plant growth outcomes. Ensuring data security and reliability is a critical objective, achieved through robust communication protocols and database solutions.

By addressing these specific objectives, the project aims to deliver a reliable, user-friendly, and sustainable solution for domestic smart agriculture, bridging the gap between advanced agricultural technologies and their practical application in smaller-scale environments.

1.6 Methodology

The development methodology adopted to build the smart agricultural cabinet was an agile methodology. The choice of this methodology was due to the fact that there was a need for adaptability and flexibility required to develop a robust system, allowing adjustments to be made based on feedback, primarily derived from collaboration with *Verdes do Mondego*. Their input during some meetings ensured that the system addressed real-world agricultural needs effectively. Agile methodologies also facilitated

continuous testing and validation, allowing for improvement of the functionalities and ensure the system met both functional and non-functional requirements [10]. This methodology provided the flexibility needed to adapt to emerging challenges and incorporate new features without compromising the project's overall goals.

The Figure 1.3 illustrates the work plan followed to develop the smart agricultural system.

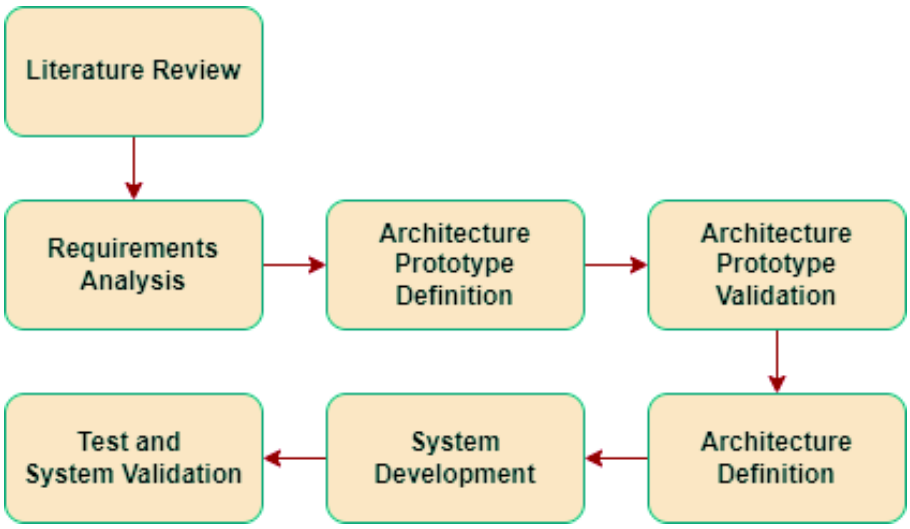


Figure 1.3: Work plan for the development of the smart agricultural cabinet

The process started with a literature review, which involved investigating and analyzing existing knowledge and advancements in the field. These articles were sourced from reputable academic databases, including IEEE, Springer Link, ScienceDirect, ACM, MDPI and b-on, ensuring that the review was grounded in high-quality, peer-reviewed research. This phase provided a solid foundation for the subsequent stages of the development process.

Following the literature review, a requirements analysis was conducted to identify and specify both functional and non-functional requirements. This phase was enriched by collaboration with *Verdes do Mondego*, ensuring that the system aligned with real-world agricultural needs. The next step involved the Architecture Prototype Definition, which established the initial design of the system's architecture.

This prototype was subjected to a comprehensive prototype validation by specialists to ensure it meets the identified requirements and is suitable for practical application. Upon successful validation, the workflow progressed to the system development, during which the smart agricultural cabinet was implemented. The process culminated in test and system validation, where the system's functionality and effectiveness were evaluated.

1.7 Document Structure

This document is organized into six chapters, each focusing on a critical aspect of the project's development and implementation. Chapter 1 introduces the project, providing an overview of its context and objectives while explaining the motivation behind the creation of the smart agricultural cabinet. Chapter 2 explores the principles of IoT technologies and their applications in smart systems, focusing on the frameworks and tools that support the development of this project. Additionally, it examines the state of the art in smart agriculture, providing an analysis of existing solutions, identifying their limitations and highlighting opportunities for innovation, especially in the context of small-scale agricultural systems.

Chapter 3 explores the system architecture, presenting a detailed account of how hardware, software and IoT technologies are integrated to meet the project's objectives. Chapter 4 describes the development process, outlining the implementation of key components, including controllers, sensors, mobile and web applications, and the system orchestrator. Chapter 5 evaluates the system's performance, providing insights into its functionality and reliability based on functional and non-functional tests and user experience questionnaires. Finally, Chapter 6 concludes the document by summarizing the project's findings, reflecting on its contributions to smart agriculture and suggesting potential directions for future development and improvement.

2 State of the Art

This section presents a comprehensive review of the latest advancements, methodologies, and technologies essential to the development of smart agricultural systems leveraging IoT. The review focuses on recent systems and approaches that enhance agricultural practices through automation, real-time data monitoring, and data-driven analytics, thereby optimizing efficiency and sustainability. To ensure that the analysis remains current and relevant, only articles published within the last 10 years were considered. These articles were sourced from reputable academic databases, including IEEE, Springer Link, ScienceDirect, ACM, MDPI and b-on, ensuring a foundation rooted in high-quality, peer-reviewed research.

The analysis highlights the integration of IoT-enabled sensors and automated control systems, such as as ventilation, irrigation and lighting, to improve processes such as precision irrigation, soil nutrient management and climate control. Furthermore, this section critically examines existing literature to identify strengths, challenges, and areas for further innovation. By evaluating best practices and uncovering gaps in current implementations, the review informs the development of more efficient and scalable solutions for the proposed smart agricultural system.

2.1 Internet of Things

According to the authors in [1], the term Internet of Things (IoT) refers to a vast network of hardware and software, including microcontrollers, sensors, graphical interfaces, and data processing capabilities that enable communication between hardware and the cloud. This study also reveals that the use of IoT began with Coca-Cola machines, experiencing exponential growth in the 21st century as the concept became the focus of numerous studies and investigations, and is now utilized in various fields, such as agriculture.

In a similar line of research, Arindom, Monirul, Animesh, and Mohammad [11] present several analyses related to the IoT market that highlight its growth. They cite a study conducted by Gartner, a leading global research company, which predicts that the IoT market will have an annual growth rate of approximately 7% between 2020 and 2030, leading to an increase of up to about 3.2 billion IoT devices.

Additionally, [12] discusses a Gartner study, which states that in 2015, there were around 5 billion connected IoT devices, with a strong tendency to increase this num-

ber in the next years. The study also mentions that numerous definitions have been proposed for the IoT concept, varying from one entity to another. The definition proposed by the authors states that IoT consists of a network of connected objects collecting data and communicating with other connected objects using communication protocols without human intervention. This definition is supported by an architecture proposed by the authors, represented in Figure 2.1, where they consider IoT to be composed of four layers:

A. **Physical Layer**
The physical layer of the IoT is composed of a diverse range of devices, such as sensors, actuators, RFID tags, GPS units, and cameras. These devices play a critical role in gathering data from the environment they are connected to, enabling the collection of various types of information, such as ambient temperature, soil humidity, and even image capture. This real-time data collection process is essential for efficient monitoring and analysis of the environment, allowing for data-driven decision-making based on accurate and up-to-date information.

B. **Communication Layer**
The communication layer of IoT is composed of various technologies, such as Wi-Fi, GSM, Ethernet, and 4G and 5G networks. These technologies play a crucial role in establishing a bridge between the physical layer of IoT and the middleware layer, enabling the transmission of collected data for future processing and analysis.

C. **Middleware Layer**
The middleware layer of IoT plays a critical role in managing the data generated by connected devices. It acts as a bridge between the communication layer and applications, being responsible for storing, organizing, and processing the data collected from the physical layer. In addition to ensuring the integrity and security of the information, the middleware analyzes this data, extracting valuable insights that can be used to optimize processes, predict trends and support decision-making. It is within this layer that raw data is transformed into useful, actionable information, facilitating integration and interoperability between different IoT devices and systems.

D. **Application Layer**
The application layer of IoT is where the information extracted and processed by the middleware is used to create smart solutions. This layer is directly related to user interaction, developing applications and services that transform the analyzed data into practical actions. The applications created at this level allow businesses and individuals to make more informed decisions and optimize their operations based on the insights generated by IoT devices.

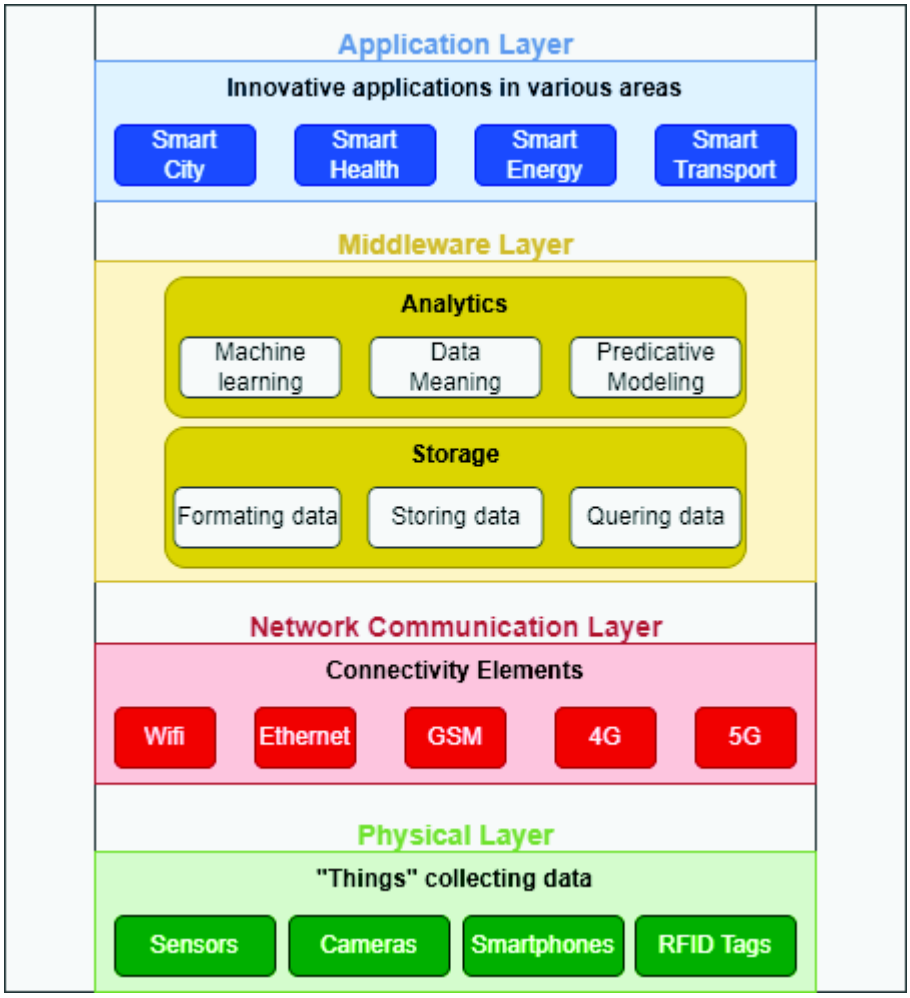


Figure 2.1: IoT architecture proposed by the authors in [12]

As further elaborated by Lin et al. [13], IoT systems are inherently layered, consisting of a perception layer, responsible for data collection using devices such as sensors and actuators, a network layer that facilitates data transmission and an application layer that provides user-facing insights and controls. This multi-layered architecture ensures scalability and interoperability, making IoT indispensable in domains such as agriculture, where it enables efficient resource management and real-time decision-making.

2.2 Sensors and Actuators

As referenced in the study conducted by Arshi and Mondal [14], sensors and actuators are critical components in the development of smart systems, including smart cities and agricultural solutions, forming the foundation of IoT architectures. Sensors gather real-time data by detecting environmental and physical properties such as temperature, humidity and light intensity. This data is essential for understanding system dynamics and enabling responsive decision-making processes. On the other hand, actuators use the information received from sensors to perform actions, such as adjust-

ing lighting, activating irrigation systems or controlling ventilation settings, thereby automating processes and improving system efficiency.

In the same study, the authors also provide an example of the combined use of sensors and actuators, as is the case in the monitoring of traffic flow, where sensors are the responsibles to monitoring traffic flow while actuators dynamically adjust traffic signals to reduce congestion and enhance mobility. Similarly, in agricultural systems, sensors measure soil moisture and environmental conditions, and actuators respond by adjusting irrigation or ventilation to ensure optimal plant growth conditions. The authors conclude that the integration of advanced sensors and actuators is crucial for achieving responsive and sustainable solutions across various domains, such as the development of modern smart systems.

After analyzing the article written by Arshi and Mondal, Zeng et al. in [15] also consider that the primary purpose of sensors can be identified as the real-time collection of various types of data, such as temperature, relative humidity and soil moisture. On the other hand, actuators transform electrical signals into non-eletrical actions, such as activating an irrigation system when a sensor detects low moisture levels. The use of sensors and actuators in smart cities, emphasizing their importance for urban management.

In agriculture, sensors are mostly used to monitor soil moisture and environmental conditions, providing data to optimize irrigation and manage greenhouse climates. Actuators receive an electrical signal and convert them into mechanical actions to adjust water flow or regulate temperature, ensuring suitable conditions for plant growth. These systems enhance resource efficiency and contribute to more sustainable practices [15].

The authors highlight that integrating sensors and actuators in IoT systems enhances sustainability and efficiency by automating processes and minimizing the need for manual intervention. This integration enables intelligent systems to adapt to changing conditions, supporting innovation across various fields.

2.3 Smart Agriculture

The concept of smart farming, also known as precision agriculture or smart farming, represents a shift in the agricultural sector driven by the development of technologies such as the IoT, artificial intelligence and data analytics. This approach seeks to improve productivity and sustainability in farming by using devices and systems that enable real-time monitoring and control of essential variables like irrigation, nutrients and soil conditions. Smart farming technologies apply to both large-scale agriculture and controlled environments like indoor farming, where conditions can be adjusted to support plant growth.

Indoor farming is a modern agricultural practice that allows crops to be grown in controlled environments like vertical farms, greenhouses and hydroponic systems. These systems create optimal conditions for plants by carefully regulating factors such as temperature, humidity and artificial lighting, often eliminating the need for soil or natural sunlight. With the integration of IoT, indoor farming ensures precise monitoring and adjustments to maintain ideal growing environments. Vertical farming, a prominent type of indoor farming, takes efficiency a step further by recycling water and nutrients, reducing the use of pesticides and fertilizers, and cutting transportation emissions through localized production in urban areas. This innovative approach offers a sustainable and practical solution to many challenges faced by traditional farming methods [16].

Smart farming brings numerous benefits, such as using water and energy more efficiently, reducing waste and achieving more stable crop yields, which help protect against unexpected losses. It also makes it easier to manage pests and diseases, cutting down on the need for chemical treatments and supporting more sustainable farming methods. However, adopting these technologies comes with its own set of challenges, including high initial costs, the need for strong infrastructure and confidence on reliable internet connectivity to work effectively [17].

Several studies have developed and tested smart farming systems to address these issues. Chakraborty et al. [11] created a smart agriculture system with three subsystems: an offline monitoring system, an IoT-enabled monitoring system, and a smart irrigation system. The first subsystem displays data from sensors on an LCD screen using an Arduino Uno. The second subsystem uses an ESP32 microcontroller to transmit sensor data to a Blynk-based[1] platform and control a water pump when moisture falls below 25%. The third subsystem, connected to an Arduino Uno, automates irrigation by activating the water pump when moisture levels are low.

In related work, Yasin, Zeebaree, and Zebari [18] proposed an automatic irrigation system using an Arduino Mega 2560. This system uses GSM technology to allow SMS-based control of irrigation, with sensors that detect soil moisture and automatically activate irrigation when needed. Rain sensors prevent irrigation during precipitation, and water level sensors monitor the storage tank. Their study showed significant water savings and highlighted the benefits of SMS-based remote control for cost-effective irrigation.

Soniya et al. [19] developed a system to automatically water plants and feed animals. They used an ESP8266 microcontroller connected to an FC-28 soil moisture sensor, a servo motor, and a relay for pump control. An ESP32 with a camera module was also used for real-time video monitoring, enabling remote observation of the system.

[1]System that uses the Blynk platform for monitoring and management

Kodali and Valdas in [20] focused on low-cost urban cultivation systems to address potential food shortages due to migration from rural to urban areas. Their automated system, designed for urban environments, uses sensors for temperature, soil moisture, and light intensity to monitor and manage plant growth.

The study conducted by Tan et al. [3] use a hydroponic technique for urban farming. The authors consider that this technique is emerging as a sustainable response to increasing population and limited arable land. The study presents an IoT-based framework that automates farming tasks like monitoring and adjusting nutrients, temperature and humidity. Using sensors for pH, Total Dissolved Solids (TDS) and temperature, along with camera-based growth tracking, the system provides real-time monitoring via web and mobile applications. It use MQTT for data transmission and MongoDB for storage, ensuring scalability and good performance. This system was tested on a hydroponic farm, where it demonstrated increased efficiency and minimal human intervention, contributing significantly to the sustainability of urban agriculture.

Finnaly, still within the area of smart agriculture but with a focus on crop protection rather than ensuring healthy plant growth, the authors Manikandan et al. [21] developed a system to detect animal and human intrusions in agricultural fields, helping protect crops. Their system combines a motion sensor with an ESP32-CAM to capture images of intruders, sending alerts to the farmer to enhance field security.

The Table 2.1 presents a summary of the key systems identified and analyzed in the context of smart agriculture. Only the most relevant articles for the project are included in this table, as they were considered the most insightful and aligned with the project's objectives, while other similar studies were not detailed due to their overlap with these key contributions.

Table 2.1: Summary of studies on smart agriculture

Authors	**Type of system**	**Summary**
Chakraborty et al. [11]	IoT-enabled smart agriculture system	Developed three subsystems: offline monitoring, IoT-based monitoring with Blynk and smart irrigation using an ESP32 microcontroller. Automated irrigation triggered when soil moisture falls below a threshold.
Yasin et al. [18]	Automatic irrigation system	Used Arduino Mega 2560 and GSM technology for control via SMS. Sensors detected soil moisture and rain, preventing unnecessary irrigation during precipitation.

Continued on next page

Authors	Type of system	Summary
Soniya et al. [19]	Automated plant watering and animal feeding system	Integrated ESP8266 microcontroller with soil moisture sensors and servo motors for pump control. Included real-time video monitoring via ESP32-CAM.
Kodali and Valdas [20]	Urban smart farming system	Designed a low-cost system for urban agriculture. Monitored temperature, soil moisture, and light intensity using IoT sensors to optimize plant growth.
Tan et al. [3]	Hydroponic urban farming with IoT	Proposed an IoT-based hydroponic framework using MQTT and MongoDB. Included sensors for pH, TDS and temperature, along with a camera for plant monitoring. Demonstrated increased efficiency with minimal human intervention.
Manikandan et al. [21]	Field security for agricultural crops	Combined motion sensors with ESP32-CAM for detecting animal and human intrusions in fields. Sent alerts to farmers for enhanced security.

2.4 Materials and Methods

This section provides a comprehensive review of the materials and methodologies explored in the literature to guide the development of a reliable and efficient smart systems, such as smart agricultural systems. Emphasis is placed on the hardware components, technological tools and strategies that align with the principles of precision agriculture. Drawing on insights from existing research and industry practices, this project adopts approaches designed to prioritize efficiency, adaptability, and long-term sustainability.

2.4.1 Hardware

To create an effective smart agriculture system, which is defined as an automated solution that minimizes the need for human intervention, several different types of components are required for specific purposes. Table 2.2 summarizes the hardware components used in the studies examined.

Table 2.2: Hardware used for the construction of smart systems

Component	**Utility**	**Reference**
Air Fans	Component responsible for air circulation in the system	[22]
Arduino Mega	Microcontroller for reading and processing sensor data	[22], [18]
Arduino Uno	Microcontroller for reading and processing sensor data	[3], [11], [23], [24]
DHT11	Digital sensor for collecting ambient temperature and humidity values	[11], [20]
DHT22	Digital sensor for collecting ambient temperature and humidity values	[1], [22]
ESP32	Microcontroller with integrated Bluetooth and Wi-Fi modules for sensor data processing	[11], [20]
ESP32-CAM	Microcontroller with integrated camera, Bluetooth, and Wi-Fi modules, for image capture	[19], [23], [21]
ESP8266	Microcontroller with integrated Wi-Fi module for reading and processing sensor data	[1], [19], [25]
FC-28 Soil Moisture Sensor	Analog sensor to measure soil moisture content	[19]
Light Dependent Resistor (LDR)	Resistor whose value changes according to light intensity on its surface, providing an analog signal	[11], [20]
LM35	Sensor for collecting ambient temperature values and provides an analog voltage output	[11]
MAX485	Low-power, half-duplex RS-485 transceiver for long-distance communication	[1]
MG811 CO2 Sensor	Analog sensor for measuring carbon dioxide concentration in the air	[22]
Motion Sensor	Digital sensor for motion detection in the system	[22], [21]
nRF52840	Microcontroller with integrated Bluetooth for sensor data processing	[26]

Continued on next page

Component	Utility	Reference
NPK Sensor	Analog sensor to measure the concentration of Nitrogen, Phosphorus, and Potassium in the soil	[1]
Raspberry Pi	Microcomputer primarily used for data processing	[3], [22], [20], [26]
Raindrop Sensor	Digital sensor capable of detecting precipitation	[18]
Relay	Electromechanical switch that controls higher power circuits	[19], [18], [24]
SG90 Servomotor	Actuator for angular position control	[19]
SIM900 GSM Shield	Component for reading a SIM card	[18]
Soil Moisture Sensor	Analog sensor to measure moisture content in the soil	[11], [20], [18], [25]
Water Pump	Component used for system irrigation	[11], [19], [18]

The review of hardware used in various analyzed studies revealed a clear preference for specific devices in the construction of smart systems, particularly in the field of agriculture:

- **Microcontroller**
  The analysis of hardware used in various studies shows a clear preference for the ESP8266 and ESP32 microcontrollers. This preference is mainly due to their integrated Wi-Fi modules, with the ESP32 also including Bluetooth functionality. These microcontrollers are programmable, facilitating communication between the physical layer and middleware layer in IoT applications. Their low cost and energy efficiency further contribute to their suitability for smart systems. The ESP32 also has a version with an integrated camera module, the ESP32-CAM, which is normally used with an OV2640 camera for direct image capture.

- **Arduino**
  Different Arduino models, such as the Arduino Mega and Arduino Uno, are commonly used in the projects analyzed. While these models do not have built-in Wi-Fi, they effectively support data collection and processing from various sensors. The Arduino Mega, with more input/output pins and memory capacity, is suited for projects requiring multiple sensor inputs, while the Arduino Uno offers a simpler solution for less complex projects. Both models support a wide range of sensors, making them widely used for data acquisition and processing.

According to the study conducted by the authors in [27], the ESP32 stands out as a highly versatile and powerful controller, particularly well-suited for IoT applications and advanced computational tasks. In comparison, the ESP8266 provides a more affordable option, ideal for simpler IoT projects with moderate requirements. On the other hand, the Arduino Mega and Uno are designed to hardware integration and prototyping applications, with the Mega being a good choice for larger-scale projects due to its extensive input/output (I/O) capabilities. Meanwhile, the Arduino Uno serves as an accessible and user-friendly device, making it a good option for small-scale projects. The comparison of SP32, ESP8266, Arduino Mega and Arduino Uno is presented in Table 2.3.

Table 2.3: Summary of the comparison of ESP32, ESP8266, Arduino Mega and Arduino Uno realized in [27]

Attribute	**ESP32**	**ESP8266**	**Arduino Mega**	**Arduino Uno**
Processor	Dual-core Xtensa LX6 @ 240 MHz	Single-core Xtensa LX106 @ 80 MHz	ATmega2560 @ 16 MHz	ATmega328P @ 16 MHz
Memory (SRAM / Flash)	520 KB / 4 MB	80 KB / 512 KB–4 MB	8 KB / 256 KB	2 KB / 32 KB
Digital I/O Pins	32 (16 PWM)	10	54 (14 PWM)	14 (6 PWM)
Analog Input Channels	18 (2 ADCs @ 12-bit)	1 (10-bit)	16 (10-bit)	6 (10-bit)
Connectivity	Wi-Fi, Bluetooth	Wi-Fi	None	None
Best Application	Advanced IoT and real-time tasks	Budget IoT systems	Large-scale control projects	Small-scale and beginner projects

- **Raspberry Pi**
  The Raspberry Pi is a commonly used device in smart systems, functioning as a small computer with data processing and analysis capabilities [26]. Its compact design and cost-effectiveness allow integration into many IoT applications. Often used in the middleware layer of the IoT architecture (Figure 2.1), the Raspberry Pi processes and analyzes data from connected devices, supporting real-time decision-making. It also supports multiple programming languages and connectivity options, allowing developers to create specific solutions for different applications.

- **DHT Sensor**
  The DHT11 and DHT22 sensors measure temperature and humidity. The DHT11 sensor has measurement ranges of 20% to 80% for relative humidity and 0 °C to 50 °C for temperature. In the other hand, the DHT22 provides more accurate readings than the DHT11, measuring humidity levels from 0% to 100% and temperatures from -40 °C to 80 °C, with a 0.5°C variation [28]. These sensors are typically connected to microcontrollers for data collection.

- **Soil Moisture Sensor**
  Sensors to measure soil moisture levels are often used in smart agriculture systems [29]. These readings help assess whether soil moisture is optimal for plant growth, informing irrigation decisions and preventing overwatering or underwatering.

- **NPK Sensor**
  The NPK sensor, similar to soil moisture sensors, is commonly used in smart agriculture systems to measure soil nutrient levels, specifically Nitrogen, Phosphorus and Potassium. This data supports soil fertility assessments, determining if the soil contains sufficient nutrients for healthy plant growth. The NPK sensor is often paired with the MAX485 module, which converts its TTL signal to RS485, making it compatible with ESP microcontrollers [1].

2.4.2 Technological Resources

The selection of technologies and tools is a critical factor in building an effective smart agriculture system. Given that IoT is a vast field, there are numerous technologies and tools designed specifically for this area. However, the articles reviewed in this study focus on specific technologies and tools that have proven to be effective in agricultural applications. This section delves into these selected technologies, highlighting their roles and contributions to the development of robust smart farming systems. The selection of technologies and tools is a critical factor in building an effective smart agriculture system. Given that IoT is a vast field, there are numerous technologies and tools designed specifically for this area. However, the articles reviewed in this study focus on specific technologies and tools that have proven to be effective in agricultural applications. This section will delve into these selected technologies, highlighting their roles and contributions to the development of robust smart farming systems.

A. **Blynk**
   The configuration and management of components in the systems developed in [19] and [21] were performed using Blynk software, specifically designed for IoT applications. Blynk provides libraries that facilitate interaction with various de-

vices, including Raspberry Pi and microcontrollers. This software detects devices connected to the same Wi-Fi network and offers various features, such as widgets for displaying sensor values and data storage. The authors conclude that this type of platform represents an effective solution for smart agriculture environments.

B. **Message Queuing Telemetry Transport (MQTT)**
The system in [20] uses the MQTT communication protocol to transmit data collected by an ESP32 to a central component, where it is stored in a MySQL database and subsequently processed. The dashboard for presenting the collected and processed data was developed using NodeRed.

C. **Data storage**
The study conducted by the authors in [12] aims to determine the most suitable NoSQL database for use in IoT applications, based on the following requirements: scalability (R1), real-time processing (R2), security (R3), spatial data handling (R4), and data aggregation (R5). For this study, the authors selected five NoSQL databases based on their performance and popularity. After aligning the individual characteristics of the selected databases with the presented requirements, the authors created a table evaluating each database against each requirement. Table 2.4 summarizes the authors' conclusions regarding the quality of the databases under study, considering their performance for each requirement:

Table 2.4: NoSQL databases analysis conducted by the authors in [12]

Database	**Scalability (R1)**	**Real-Time Processing (R2)**	**Security (R3)**	**Spatial Data Handling (R4)**	**Data Aggregation (R5)**
Redis	Poor	Medium	Medium	Medium	Medium
Cassandra	Good	Medium	Medium	Poor	Medium
MongoDB	Medium	Medium	Medium	Good	Good
Couchbase	Good	Good	Medium	Good	Medium
Neo4j	Medium	Medium	Medium	Good	Medium

In terms of scalability (R1), Couchbase and Cassandra showed the best results. For real-time processing (R2), Couchbase stood out with higher performance compared to the others. Regarding security (R3), all databases performed similarly, with a median evaluation. For spatial data handling (R4), Couchbase, MongoDB, and Neo4j received the highest ratings. Finally, for data aggregation (R5), MongoDB was the top performer.

The authors concluded that Couchbase, MongoDB, and Neo4j are among the best choices for IoT applications. Couchbase was identified as the most suitable for

scenarios requiring scalability, while MongoDB and Neo4j were noted for their capabilities in spatial data handling, with MongoDB excelling in data aggregation performance.

In another study, the authors in [30] compared four different time series databases using five evaluation criteria: write speed (R1), read speed (R2), database size (R3), support for time series data tables (R4), and API usability (R5). The results of the analysis are presented in Table 2.5.

Table 2.5: Results of the performance tests conducted by the authors in [30]

Database	**Write Speed (s)**	**Read Speed (s)**	**Database Size (MB)**	**Time Series Support**	**API Usability**
InfluxDB	200	0.16	100	Yes	3
MongoDB	89	1.56	0.004	Yes	5
TimescaleDB	97	0.389	0.136	Yes	1
ClickHouse	0.889	0.245	26.25	No	5

According to the findings, ClickHouse demonstrated the fastest write speed (R1), with MongoDB being the second fastest. For read speed (R2), InfluxDB performed best, followed by ClickHouse. Regarding database size (R3), MongoDB required the least storage space, while InfluxDB had the largest footprint. For support of time series data tables (R4), all databases except ClickHouse were suitable. In terms of API usability (R5), both MongoDB and ClickHouse were rated highly, while TimescaleDB received the lowest score.

The study concluded that ClickHouse is a leading option for time series data storage, although MongoDB, despite slower read and write speeds, excels in other criteria and remains a viable choice.

In another study, Paethong et al. [31] examined the advantages and limitations of utilizing a Raspberry Pi as a database server for IoT systems. The study compared the performance of MongoDB and MySQL on both a Raspberry Pi 2 Model B and an x86 machine. The detailed results of this comparison are presented in Table 2.6.

Table 2.6: MySQL and MongoDB comparison

Database (Platform)	**Insert (s)**	**Select All (s)**	**Select Where (s)**	**Update All (s)**	**Update Where (s)**	**Delete All (s)**	**Delete Where (s)**
MySQL (x86)	76.64	6.72	0.85	8.93	1.98	0.03	0.50
MySQL (Raspberry Pi)	765.54	4.98	2.14	31.62	6.15	3.03	2.00
MongoDB (x86)	1084.24	8.87	0.28	3.70	0.29	6.40	0.28
MongoDB (Raspberry Pi)	1086.61	7.42	4.27	45.88	4.48	126.20	4.31

The results demonstrated that MySQL on the x86 platform performed better than on the Raspberry Pi in most cases. However, for specific tasks like insertion and updating with conditions, MongoDB on the Raspberry Pi was faster than MySQL. In other cases, the x86 machine significantly outperformed the Raspberry Pi.

2.4.3 Machine Learning

The authors responsible for the research documented in [32] analyze various viable methods for identifying certain types of plant leaves. To achieve good performance in leaf classification, the algorithm considered several features, such as leaf shape, texture, and venation. In the study, the authors state that automated plant classification significantly reduces the time spent on manual identification performed by people. However, they also caution that a substantial number of resources are required, including databases, in-depth knowledge in the field, and strong software development skills to automate this process.

With the study conducted in [33], the authors highlight the importance of integrating Big Data and Machine Learning technologies in rice agriculture. They emphasize that by combining data from multiple sources, such as temperature sensors, soil moisture, precipitation, wind speed and direction, as well as satellite imagery and soil data, it is possible to significantly improve the accuracy of yield prediction, growth monitoring, and disease detection.

The study also identifies optimal IoT devices for specific agricultural applications. To estimate rice yield, the authors recommend utilizing data collected from sensors that monitor soil moisture, temperature, pH levels, and light intensity. For tracking rice growth, temperature and soil pH sensors are suggested. Furthermore, for the development of intelligent irrigation systems, the implementation of water level sensors is proposed. The findings reveal that a well-designed irrigation system can reduce water consumption in rice fields by up to 19.91% while simultaneously enhancing produc-

tion by fostering a more favorable environment for cultivation. In the area of disease detection, the study explores the use of various Machine Learning algorithms that have shown promising results. The following algorithms are highlighted for their effectiveness:

- **Support Vector Machine (SVM)**
  Frequently employed for classification and pattern recognition tasks, SVM can be applied to classify images of rice leaves and detect the presence of diseases such as rice rust.
- **Convolutional Neural Networks (CNNs)**
  Particularly effective in image analysis, CNNs can be trained to identify visual features associated with plant diseases, allowing for early detection and the precise identification of disease types in rice crops.
- **K-Nearest Neighbors (KNN)**
  A straightforward yet effective algorithm, KNN can classify plant samples based on similar features, aiding in the identification and differentiation of rice diseases.
- **Decision Trees and Random Forests**
  These algorithms are powerful tools for classifying data and can be applied to analyze environmental variables and plant characteristics, providing accurate predictions regarding disease outbreaks.

The authors conclude that these technological innovations not only improve the efficiency of agricultural practices but also promote sustainability by optimizing water usage and enhancing soil health. The integration of Big Data, IoT, and Machine Learning in rice farming has the potential to transform production systems, making them more resilient to climate change and better equipped to meet the growing global demand for food.

According to Araújo et al. [34], today's agriculture has been revolutionized by machine learning, which provides cutting-edge approaches to enhance crop management, allocate resources more efficiently and advance environmental sustainability. By employing advanced algorithms like Random Forest, SVM, and CNN, machine learning facilitates precise identification of diseases, enhances yield prediction and optimizes irrigation management practices. These models process data from IoT sensors to make precise adjustments to parameters such as soil moisture and nutrient levels, ensuring efficient resource usage and healthier crops.

Machine learning also plays a crucial role in managing water and soil resources by optimizing irrigation timetables and tracking soil status continuously. This anticipatory strategy minimizes resource waste and enables farmers to make knowledgeable choices, resolving potential problems before they affect productivity. Despite chal-

lenges such as the need for high-quality datasets and specialized expertise, advancements in ML continue to unlock its vast potential, paving the way for more sustainable and efficient farming practices [34].

With the study carried out in [3], the authors demonstrate the use of machine learning to improve urban farming processes. It integrates image processing with Python and OpenCV to monitor plant growth, using green pixel analysis to track changes over time. Sensor data on pH, TDS and temperature is also analyzed to automate fertigation, which is a technique used in hydroponic systems that involves fertilizing the water used to irrigate the crops, and regulate environmental factors. By reducing manual intervention and enhancing resource efficiency, the platform showcases the potential of combining machine learning with IoT to advance precision agriculture.

2.4.4 Key Conclusions From Analyzed Solutions

The analysis of the state of the art in smart agriculture systems offers valuable insights into innovative approaches designed to integrate IoT, automation and user-friendly interfaces. Table 2.7 presents a concise summary of the key features and outcomes of these solutions, providing a comparative analysis that serves as a foundation for identifying best practices for the development of the proposed smart agricultural cabinet.

Table 2.7: Key components and their usage in the smart agricultural cabinet

Component	**Description / Usage**
Blynk Platform	Provides a user-friendly interface for real-time monitoring of the smart agricultural system
DHT22 Sensor	Measures temperature and humidity within the cabinet, ensuring environmental parameters are optimized for plant growth
ESP32 Microcontroller	Used for sensor data collection and communication with the system
MongoDB Database	It is used for storing unstructured sensor data, including real-time measurements, to facilitate comprehensive trend analysis and generate valuable historical insights
MySQL Database	Stores structured data, such as user preferences and environmental configurations, ensuring consistency and reliability
Phytolamp	Simulates sunlight to promote photosynthesis in indoor farming environments
	Continued on the next page

Component	Description / Usage
Python	Used to develop the logic for system orchestration and data processing
Raspberry Pi	Used for handles computational tasks, real-time data processing and communication with sensors and user applications. It also will serve as host of the MySQL and MongoDB databases, as well as the Mosquitto MQTT broker, ensuring seamless data management and IoT communication
Soil Moisture Sensor	Used to collect the soil's water content
Water Pump	Used for the irrigation system

3 System Architecture

This chapter introduces the architecture of the smart agricultural cabinet, highlighting how it combines hardware and software to enable real-time monitoring of plant growth and implement automated controls that maintain ideal environmental conditions. The development process followed agile methodologies, allowing for continuous improvements and ensuring the system could adapt to changing project requirements and agricultural challenges. The design emphasizes efficiency and remote accessibility, ensuring it remains versatile and responsive to the dynamic needs of modern agriculture. Key elements for the system are microcontrollers, the mobile application and a dual-database approach that employs MySQL and MongoDB for optimized data management. The sections that follow delve into these components, showing how the architecture supports sustainable plant growth through constant data collection, analysis and adaptive adjustments.

3.1 System Requirements

The development of the smart agriculture cabinet requires a detailed definition of system requirements to guide the design, implementation and evaluation phases.

The knowledge acquired in the Requirements Analysis curricular unit was applied to create a storyboard, presented in Figure 3.1. This storyboard, created with Canva [35], was significantly important in visualizing the imagined functionality of the smart agricultural cabinet. By outlining key user interactions and operational scenarios, it provided a clear representation of the system's objectives, facilitating effective communication with *Verdes do Mondego*.

Figure 3.1: Storyboard for the Smart Agricultural Cabinet

Building upon the insights gained from the storyboard, the next step was to establish a structured approach to prioritizing system functionalities. Given the constraints of resources, time and development scope, the MoSCoW method was employed. According to Ahmad et al. [36], the MoSCoW method is among the most frequently cited techniques for prioritizing software requirements, especially in scenarios where limited resources, time and budget restrict the scope of system development, ensuring that indispensable functionalities were prioritized while less critical elements could be deferred to later stages.

This method categorizes requirements into four priority levels:

- **Must have:** Critical requirements that are essential for the system's operation. The absence of these features would render the system non-functional.
- **Should have:** Important requirements that significantly enhance the system but are not strictly necessary for core functionality.
- **Could have:** Non-essential features that add value but can be omitted if time or resources are limited.

- **Won't have:** Features that are not considered for the current version but may be considered for future iterations.

This requirements were defined with the support of the stakeholder of this project, *Verdes do Mondego*, which played a key role in aligning the project with the specific needs of plant cultivation. Two meetings were held with the company's representative to gather insights and refine the system's focus.

The first meeting, held remotely on March 14, 2024, included a presentation of the project and an initial version of the proposed architecture. Based on the feedback provided, several decisions were made for the development of the system, such as using microgreens as test plants and prioritizing the control of temperature, humidity and soil irrigation as the system's primary features.

The second meeting, conducted on April 11, 2024, at *Verdes do Mondego*'s facilities, provided an opportunity to explore their current microgreens cultivation system. The representative outlined improvements they wanted to see, many of which were incorporated into this project. The main suggestion was to replace their current setup—three separate rooms with distinct environments for different growth phases—with a single, fully automated and adaptable space capable of adjusting to the specific growth phase.

These discussions with *Verdes do Mondego*, experts in biology and indoor microgreens cultivation, informed the system requirements described in the following subsections.

3.1.1 Functional Requirements

The functional requirements define the essential capabilities of the system. These requirements are categorized based on the primary components of the system: mobile application, web application and the smart cabinet itself.

Table 3.1 outlines the functional requirements for the mobile application, focusing on user interaction, system control, and remote monitoring capabilities. Key features include secure registration and authentication (FR1, FR2), real-time monitoring and control of environmental conditions such as humidity and temperature (FR6), and the management of water levels (FR4). The ability to configure sensor thresholds (FR3) is considered essential for automated responses to environmental changes, making it a "must have" feature.

Table 3.1: Functional Requirements - Mobile Application

ID	**Description**	**Priority (MoSCoW)**
FR1	The system must allow users to register a new account	Should have
FR2	The system must authenticate users through a login mechanism using email and password	Must have
FR3	Users must be able to configure threshold values for sensors to trigger alerts and automated actions	Must have
FR4	The application must enable users to monitor and control the water levels in each section of the cabinet	Must have
FR5	Users should have the ability to adjust the ventilation intensity.	Should have
FR6	The system must provide real-time monitoring and control of humidity and temperature in each section	Must have
FR7	Users should be able to adjust light intensity for each section of the cabinet	Should have
FR8	The application should allow control over nutrient levels in each section to optimize plant growth	Could have
FR9	The mobile application must establish and maintain a secure connection with the smart cabinet	Must have

The functional requirements specific to the web application are presented in the Table 3.2. These requirements emphasize secure access (FR10, FR11) and comprehensive system management capabilities, such as displaying real-time sensor data on a dashboard (FR12) and providing tools for component control (FR13). The ability to analyze historical sensor data (FR14) enables users to make data-driven decisions, enhancing the overall effectiveness of the system.

Table 3.2: Functional Requirements - Web Application

ID	**Description**	**Priority (MoSCoW)**
FR10	The web application must support user registration with secure password management.	Must have
FR11	The system must authenticate users securely during the login process.	Must have
FR12	Real-time sensor data must be displayed on the dashboard for monitoring.	Must have

ID	Description	Priority (MoSCoW)
FR13	The web interface should provide controls for managing system components such as lighting, ventilation, and irrigation.	Could have
FR14	Historical sensor data must be accessible for analysis and reporting.	Could have

Table 3.3 focuses on the core functionalities of the smart cabinet itself. It includes the system's capabilities to continuously collect, transmit, and monitor environmental data such as humidity, temperature, and light intensity (FR15, FR16). Automated adjustments to ventilation (FR18) are critical for maintaining optimal growing conditions, while the system's ability to respond to commands from the mobile application (FR19) ensures seamless integration with user inputs. Additionally, the system's capacity to capture periodic images (FR20) supports remote visual inspections, which can aid in disease detection.

Table 3.3: Functional Requirements - Smart Cabinet

ID	Description	Priority (MoSCoW)
FR15	The cabinet must continuously collect and transmit humidity and temperature data from each section.	Must have
FR16	The system should monitor and report the light intensity levels in each section.	Should have
FR17	Nutrient concentration levels in each section must be monitored for optimal plant health.	Could have
FR18	The ventilation system must automatically adjust based on predefined thresholds.	Must have
FR19	The system must receive control commands from the mobile application and execute them promptly.	Must have
FR20	The cabinet should capture periodic images of each section for monitoring plants and detecting diseases	Could have

3.1.2 Non-Functional Requirements

The non-functional requirements are concerned with the system's performance, security, and usability. These requirements ensure that the system not only functions correctly, but also meets the quality standards expected in terms of reliability, efficiency, and user experience.

Table 3.4 specifies the non-functional requirements that the system must meet to ensure its robustness and efficiency. The system's uptime is crucial (NFR1) to maintain

continuous operation, especially in a smart agriculture context where disruptions can impact crop health. Data security (NFR2) is a top priority to protect sensitive information transmitted between the smart cabinet and the user interfaces. Additionally, optimizing resource usage (NFR3) is essential for sustainable operations, while scalability (NFR4) ensures the system can accommodate future growth in terms of users and cabinet sections. Cross-platform compatibility (NFR5) and user-friendly interfaces (NFR6) are also essential to enhance accessibility and ease of use across different devices.

Table 3.4: Non-Functional Requirements

ID	**Description**	**Priority (MoSCoW)**
NFR1	The system must maintain a minimum uptime of 99.9% to ensure reliability	Must have
NFR2	All data transmissions must be encrypted to ensure confidentiality and integrity	Must have
NFR3	Resource efficiency must be optimized, reducing water and energy consumption	Should have
NFR4	The system should be designed to be scalable, allowing it to handle an increasing number of users and sensors without compromising performance.	Should have
NFR5	The system should be cross-platform, supporting both Android and iOS devices, as well as modern web browsers	Could have
NFR6	The user interfaces must be user-friendly, minimizing the learning curve	Could have

All these requirements were thoroughly evaluated and validated by *Verdes do Mondego*, ensuring that the proposed solution satisfies the practical needs and standards for effective and sustainable plant growth.

3.1.3 Use Cases

The smart agricultural cabinet system is designed to seamlessly blend user interaction with automated operations, making plant care more efficient and intuitive. By integrating a range of thoughtful functionalities, the system supports users in creating optimal conditions for plant growth while minimizing manual effort. After the validation of functional and non-functional requirements with the person in charge at *Verdes do Mondego*, the design of the use cases commenced. These use cases, also validated by *Verdes do Mondego*, are illustrated in the use case diagram shown in the Figure 3.2, highlight the key ways users and the system interact.

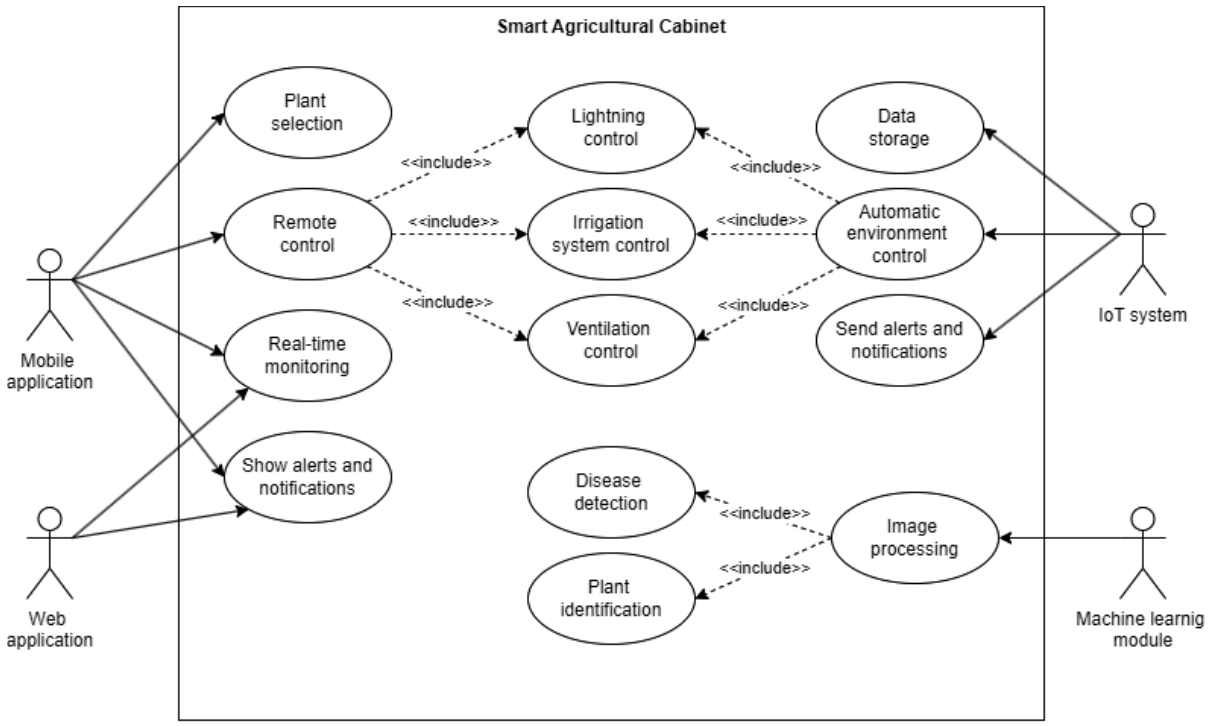


Figure 3.2: Use cases diagram for the smart agricultural cabinet

- **Plant Selection**
  Users can select a specific plant from a preloaded database to initiate its cultivation cycle. Based on this selection, the system retrieves optimal environmental parameters tailored to the plant's needs. This ensures precise care and customization for the plant's growth requirements. Access to the mobile application is required for this feature.

- **Real-Time Monitoring**
  The system provides users with live updates on environmental conditions such as temperature, humidity, and soil moisture. These quantities are accessible via the mobile and web applications, empowering users to track and manage their plants' environment in real time.

- **Remote Control**
  Users have the flexibility to adjust key settings such as lighting intensity, irrigation schedules, and ventilation levels through the mobile application. This use case ensures that users maintain full control over the system.

- **Show Alerts and Notifications**
  The system monitors environmental conditions continuously. If deviations from predefined thresholds are detected, such as low soil moisture or high temperatures, it sends alerts and notifications to the user through the mobile application, allowing for prompt action.

- **Automatic Environment Control**
  This comprehensive functionality automatically adjusts temperature, humidity, lighting, and irrigation based on sensor data. The IoT system maintains all parameters within optimal ranges, reducing the need for human intervention.
- **Irrigation System Control**
  By monitoring soil moisture levels, the system activates the irrigation mechanism when necessary. This ensures consistent hydration while preventing overwatering and avoid wasting a vital resource such as water. Users can adjust irrigation thresholds or activate at any time through the application.
- **Lighting Control**
  Lighting is automatically adjusted based on the growth stage of the plant and the predefined parameters. Users can also override this functionality through the mobile application, ensuring flexibility in meeting specific plant needs.
- **Ventilation Control**
  The system adjusts ventilation levels dynamically to regulate temperature and humidity. This functionality helps maintain a stable environment and reduces the risk of plant stress caused by overheating or excess moisture.
- **Environmental Adaptation**
  Based on the feedback from *Verdes do Mondego*, the system can adapt its settings dynamically to accommodate various stages of plant growth within a single chamber. This use case replaces the need for separate chambers with distinct environmental settings, optimizing space and simplifying operation.
- **Send Alerts and Notifications**
  The system proactively identifies potential issues, such as resource shortages or environmental imbalances, and notifies users via the mobile application. This ensures timely interventions and prevents disruptions to plant growth.
- **Data Storage**
  The system stores historical environmental data and provides detailed reports via the web dashboard. This data enables users to analyze trends and refine their plant care strategies, fostering informed, data-driven decisions.
- **Image Processing**
  The system incorporates image processing capabilities to support plant identification and disease detection. Images are captured by the ESP32-CAM and sent to the system for processing. This functionality enables precise monitoring of plant conditions, allowing the system to make adjustments or provide actionable insights based on the analysis.
- **Plant Identification**
  The system uses a machine learning module to identify plant types based on pre-

defined characteristics. This functionality enhances adaptability by ensuring that the system adjusts its configurations to suit specific plant requirements.

- **Disease Detection**
  By analyzing plant images through the machine learning module, the system can detect signs of disease or abnormalities. This enables users to take corrective action promptly, protecting the health of the plants.

Similar to the system requirements, the use cases were also presented to and validated by *Verdes do Mondego*. This process ensured that the proposed solution aligned with the needs for a robust and complete system.

3.2 General Architecture

The proposed system architecture for the smart agricultural cabinet, represented in Figure 3.3, is designed to seamlessly integrate hardware components, IoT technologies, and software solutions to optimize the monitoring and management of plant growth. This architecture is composed of multiple layers, each responsible for specific functionalities that ensure efficient communication, data processing, and user interaction.

At its core, the architecture is structured around three primary compartments, each equipped with an independent ESP32 microcontroller. The microcontrollers in this system oversee a variety of sensors and actuators. Among them are DHT22 sensors that measure temperature and humidity, as well as soil nutrient sensors (NPK sensors) that track the levels of essential nutrients such as nitrogen, phosphorus, and potassium. Furthermore, each section is fitted with an ESP32-CAM module to capture real-time images of the plants, facilitating effective visual monitoring and analysis.

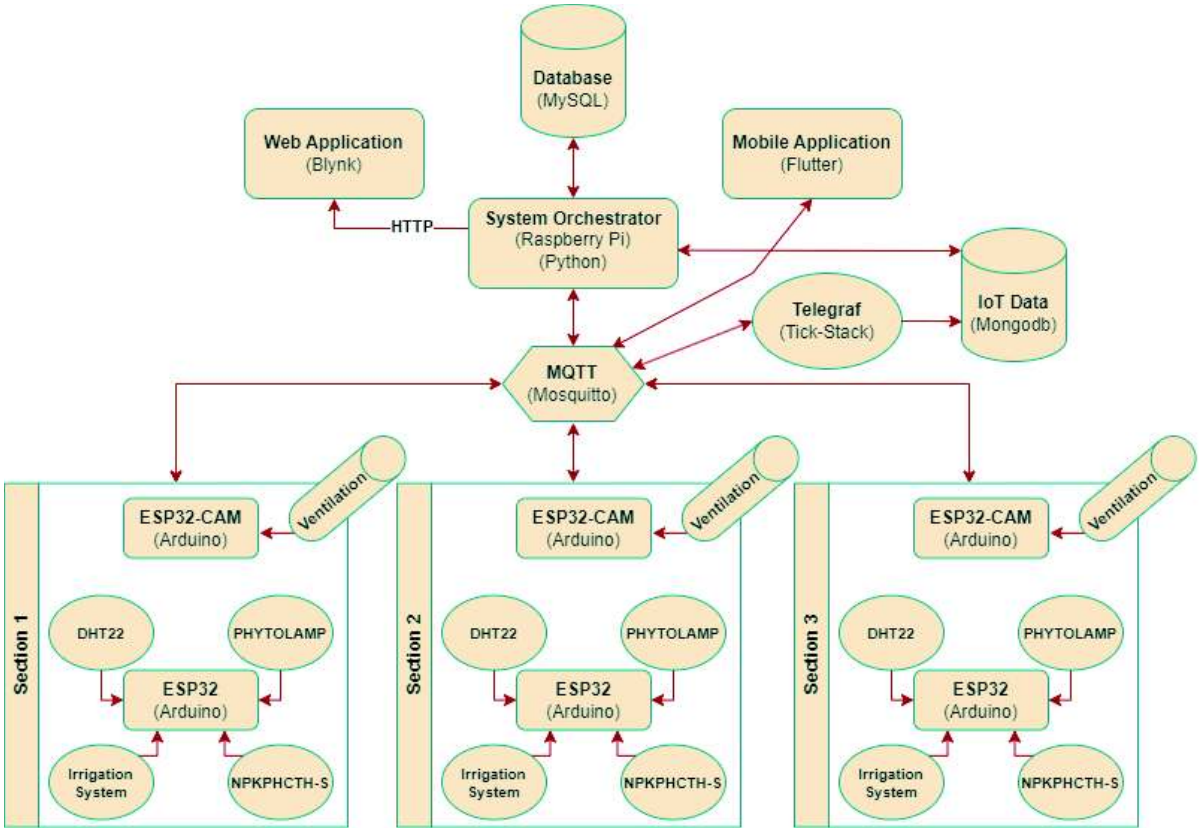


Figure 3.3: General architecture of the smart agricultural cabinet

The data collected from the sensors in each compartment is transmitted to a central control unit, a Raspberry Pi, which serves as the system orchestrator. The system orchestrator uses the MQTT protocol to facilitate efficient, low-latency communication between the ESP32 devices and itself. The Mosquitto MQTT broker, running in a Docker container, ensures reliable data exchange, leveraging the lightweight publish-subscribe model to reduce network congestion and optimize bandwidth usage.

Data processing on the Raspberry Pi involves real-time analysis of sensor readings to adjust environmental conditions automatically, such as activating the irrigation system or adjusting the ventilation intensity based on predefined thresholds. The processed data is stored in two databases: MySQL for structured data, like user configurations and system settings, and MongoDB for unstructured IoT data, such as sensor values, to visualize sensor data history.

The backend infrastructure, developed using Python, supports both the web application and mobile application. These interfaces allow users to remotely monitor and control the cabinet, providing real-time feedback on plant conditions, environmental metrics and system status. In this system, Telegraf is used to receive sensor data from the microcontrollers and forward them to MongoDB.

In addition to ensuring operational efficiency, the architecture is designed for scalability and cross-platform compatibility. The use of Docker containers allows for seam-

less deployment across various environments, while the MQTT protocol ensures a lightweight and efficient method for communication between components.

3.3 Data Storage

The architecture of the smart agricultural cabinet incorporates a dual-database strategy to optimize data management, improve system performance, and ensure scalability. This approach leverages the complementary strengths of both MySQL and MongoDB, chosen for their distinct capabilities in handling different types of data.

3.3.1 MySQL

MySQL is a highly reliable and well-established relational database management system (RDBMS) selected to manage structured data, such as user profiles and system configurations. The primary advantage of using MySQL lies in its robust handling of transactional data, ensuring high levels of consistency and integrity, which are essential for reliable system performance in critical IoT applications. According to Eyada et al. [37], MySQL is particularly effective in scenarios where structured data and predefined schemas are crucial for maintaining data consistency.

The literature review played a key role in selecting MySQL as the database for the system, however, another important reason for choosing MySQL was its high ranking on DB-Engines in December 2024, a resource discussed in the Big Data module of the Master in Computer Engineering. DB-Engines evaluates database management systems (DBMS) based on popularity metrics [38]. As can be seen in Figure 3.4, MySQL secures the second position, trailing only Oracle. Even so, among open-source databases, MySQL ranks as the leading solution.

Rank			DBMS	Database Model	Score		
Dec 2024	Nov 2024	Dec 2023			Dec 2024	Nov 2024	Dec 2023
1.	1.	1.	Oracle	Relational, Multi-model	1263.79	-53.22	+6.38
2.	2.	2.	MySQL	Relational, Multi-model	1003.76	-14.04	-122.88
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	805.69	+5.80	-98.14
4.	4.	4.	PostgreSQL	Relational, Multi-model	666.37	+12.04	+15.47
5.	5.	5.	MongoDB	Document, Multi-model	400.39	-0.54	-18.76
6.	6.	6.	Redis	Key-value, Multi-model	150.27	+1.63	-8.08

Figure 3.4: DB-Engines ranking [39]

The MySQL database is the core of the system's structured data management. Its schema has been thoughtfully prepared to facilitate key functionalities such as user authentication, section management and environmental configuration. The following subsections outline the main tables within the database and their specific roles.

The *users* table is responsible for managing user authentication and store user information. Each user entry includes a unique username and email, along with a hashed password to ensure the protection of sensitive data. The inclusion of unique constraints on the username and email columns ensures that duplicate entries are prevented, enhancing the integrity of the database. Furthermore, timestamps automatically log the creation time of each user entry, creating an audit trail. The schema is defined as follows:

```
CREATE TABLE IF NOT EXISTS users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(255) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Listing 3.1: SQL code to create *users* table

The *sections* table is designed to store information about individual cabinet sections, including their names, descriptions, and configuration data. To provide flexibility, the table supports JSON fields for storing additional metadata. Each section is associated with an environmental configuration through a foreign key reference to the *env_configs* table. This relationship enables dynamic application of environmental parameters, ensuring that each section operates under optimal conditions. The ON DELETE SET NULL clause ensures that if an environmental configuration is deleted, the associated sections remain intact by nullifying the reference. The schema is as follows:

```
CREATE TABLE IF NOT EXISTS sections (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    data_configs JSON,
    env_config_id INT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    CONSTRAINT fk_env_config
        FOREIGN KEY (env_config_id)
        REFERENCES env_configs(id)
        ON DELETE SET NULL
);
```

Listing 3.2: SQL code to create *sections* table

The *env_configs* table holds predefined environmental configurations that include thresholds for temperature, humidity, soil moisture, electrical conductivity (EC), and pH levels. These configurations play a vital role in automating the control of the cabinet's

systems, such as irrigation and ventilation. By defining these thresholds as float values, the database guarantees accurate and consistent management of the parameters. A unique constraint on the name column allows for easier identification and reuse of configurations. The schema is as follows:

```
CREATE TABLE IF NOT EXISTS env_configs (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL UNIQUE,
    temp_min FLOAT(5, 2) NOT NULL,
    temp_max FLOAT(5, 2) NOT NULL,
    humidity_min FLOAT(5, 2) NOT NULL,
    humidity_max FLOAT(5, 2) NOT NULL,
    soil_moist_min FLOAT(5, 2) NOT NULL,
    soil_moist_max FLOAT(5, 2) NOT NULL,
    ec_min FLOAT(5, 2) NOT NULL,
    ec_max FLOAT(5, 2) NOT NULL,
    ph_min FLOAT(5, 2) NOT NULL,
    ph_max FLOAT(5, 2) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Listing 3.3: SQL code to create *env_configs* table.

Figure 3.5 illustrates the Entity-Relationship (ER) diagram for the smart agricultural cabinet database. The diagram highlights the relationships between the tables sections, env_configs and users. The sections table is linked to the env_configs table through a many-to-one relationship. This setup enables multiple sections to utilize the same environmental configurations, promoting consistency and reusability. The users table is linked to the sections table through a one-to-many relationship, as a single user can manage and control multiple sections within the system.

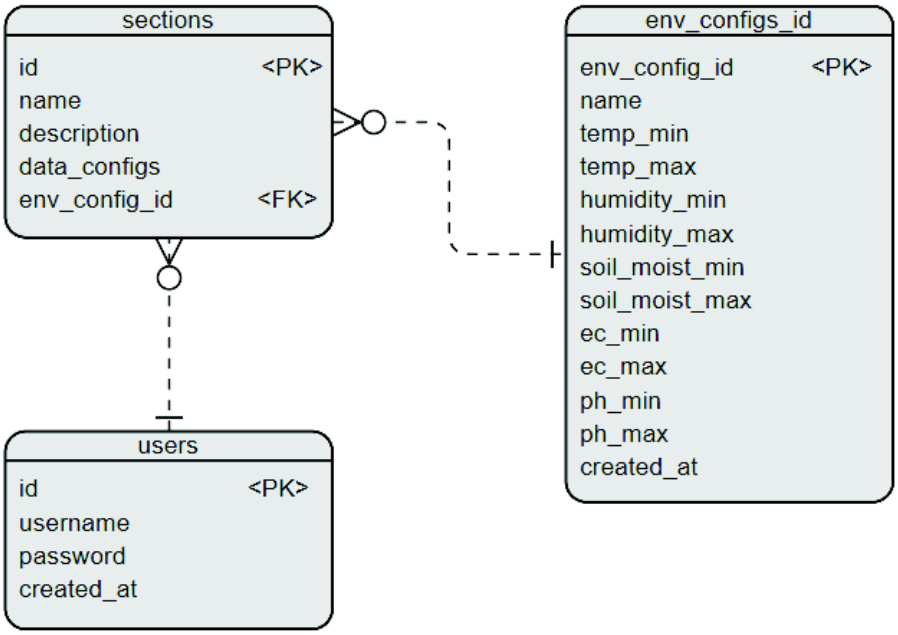


Figure 3.5: ER diagram of the smart agricultural cabinet database

3.3.2 MongoDB

Complementing MySQL, MongoDB is employed as the system's NoSQL database, specifically designed to handle the vast amounts of data generated by the IoT sensors embedded in the smart cabinet. MongoDB excels in managing high volumes of varied data types, which is particularly advantageous for storing sensor readings, time-series data and environmental metrics [37].

The flexibility of MongoDB's document-oriented architecture allows it to efficiently store dynamic datasets, such as continuous measurements of temperature, humidity and soil moisture, as well as images captured for plant health monitoring. This structure not only facilitates real-time access to data but also enables efficient historical analysis, which is critical for detecting patterns and trends over time. As noted by Eyada et al., MongoDB significantly outperforms traditional relational databases like MySQL in handling high-frequency IoT data streams due to its schema-less design and lower latency [37].

As with MySQL, the DB-Engines ranking was also considered a key factor in selecting MongoDB as the database. Among the NoSQL databases listed in the ranking, MongoDB is the most popular, followed by Redis, as shown in the Figure 3.4.

3.4 Technologies and Tools Used

The design of the smart agricultural cabinet integrated a curated suite of technologies and tools to deliver a reliable, scalable and effective IoT solution. Here in this sec-

tion will discuss the detailed about chosen key technologies and software tools with their importance towards the project requirement, ability to integrate well and level of performance. The association of hardware platforms with communication protocols, programming languages and development environments supported effective data acquisition, processing, and remote execution.

3.4.1 Python

Python was selected as the main programming language for the development of backend scripting and data processing algorithm development programming language due to its flexible and efficient framework for data processing and component coordination [40]. Such large libraries enable efficient data analysis and also assist or merge the machine learning model in such a way as to offer minimum impedance. Further, paho-mqtt being an implementation of MQTT communication in Python, Python code can be made to run on any system with Python and implemented libraries, thereby enabling easy integration with IoT systems for real-time data processing. The Python version chosen for the development was the latest available at the start of the project: Python 3.12, released in October 2023.

3.4.2 Flutter

Flutter was chosen as the technology to build the mobile app because it can create high-performance cross-platform apps with a single codebase [25]. The other reason for selection is that it is a framework introduced in 2018 by Google, which allows the developers to create applications for Android and iOS devices with near-native performance [41]. The mobile application communicates with backend systems over a standard MQTT connection, allowing users to monitor sensor data in real-time and control the smart cabinet.

3.4.3 Message Queuing Telemetry Transport

The implementation of MQTT in this project ensures low-latency communication and reliable data delivery. Features such as retained messages and the Last Will and Testament enhance the robustness of the system, aligning with the requirements of smart agricultural systems [42]. MQTT's publish/subscribe model minimizes network overhead and supports three Quality of Service (QoS) levels, allowing it to handle diverse IoT scenarios efficiently. These characteristics make it a widely adopted protocol for real-time IoT applications [42].

The Figure 3.6 illustrates the simplicity of the architecture of the MQTT publish/subscribe architecture. This communication model operates on a straightforward princi-

ple: publishers send data to specific topics and subscribers receive this data by subscribing to the corresponding topics.

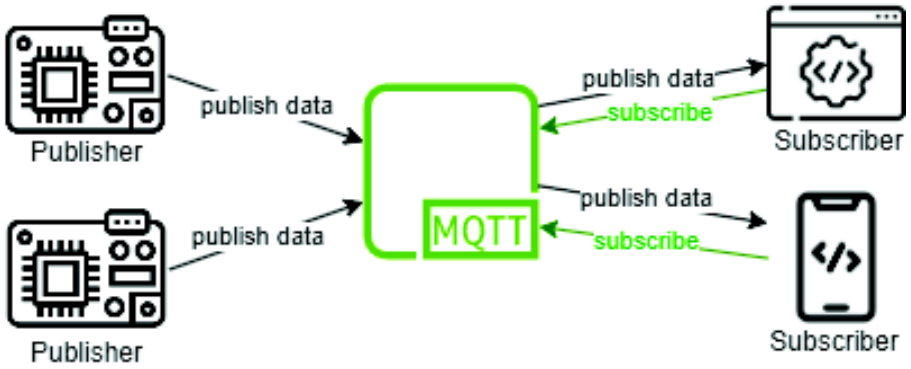


Figure 3.6: General MQTT architecture

To integrate MQTT into this system, the Mosquitto broker was chosen to centrally manage communication between all components. Mosquitto stands out for its performance and efficiency, especially in resource-constrained environments. Mishra et al. [43] demonstrated that Mosquitto consistently outperformed other MQTT brokers, such as HiveMQ and EMQX, in single-threaded environments. It exhibited lower CPU usage, reduced latency and higher throughput across various QoS levels. This is particularly important for this smart system, because a consistent communication between sensors, actuators and user interfaces is crucial to maintain optimal environmental conditions.

The Mosquitto MQTT broker was deployed within a Docker container, enabling an easier setup and efficient management while maintaining performance in diverse environments.

3.4.4 Blynk

The Blynk platform was used to visualize real-time data from the smart agricultural system, providing an intuitive and efficient way to monitor environmental conditions remotely. Blynk connects IoT devices, such as the raspberry pi or the microcontrollers, to its cloud server via Wi-Fi, where data can be processed and displayed in a dashboard, like is illustrated in the Figure 3.7. For this system, key agricultural parameters like soil moisture, temperature and humidity were transmitted to the Blynk cloud, enabling real-time monitoring through a dashboard accessible via mobile or web applications.

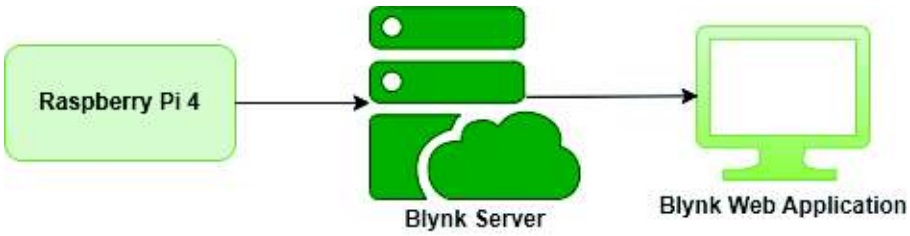


Figure 3.7: Blynk platform working diagram

The choice to use this platform was due to the fact that it was extensively explored in some of the studies on smart systems analyzed during the literature review. Studies have demonstrated Blynk's effectiveness in improving smart farming practices by offering live data visualization, thereby empowering the users to make informed decisions in real-time [44].

3.4.5 Arduino

The Arduino programming language, which is a simplified version of C/C++, was used to program the ESP32 microcontroller, responsible for controlling sensors and actuators in the smart cabinet. The Arduino IDE was employed for writing, compiling and uploading code to the ESP32 microcontroller. The choice of the Arduino IDE was driven by its user-friendly interface, ESP32 libraries and extensive community support [45], which streamlined the prototyping and testing stages. The ESP32 microcontroller was programmed to communicate with the backend via the MQTT protocol, ensuring real-time data transmission and control based on sensor inputs.

3.4.6 Visual Studio Code

Visual Studio Code was chosen as the primary Integrated Development Environment (IDE) for coding and debugging. Its powerful extensions, including those for Python, Flutter, and MQTT, enabled a seamless development process. The built-in Git support facilitated efficient version management, issue tracking and collaborative development, ensuring smooth project progression.

3.4.7 Git and GitHub

Git [46] was used for version control to manage changes to the project files, while GitHub [47] was used to store the source code and track progress. This ensured that the project maintained an organized and structured development process. The use of Git enabled efficient tracking of updates and facilitated rollback if necessary, contributing to project stability and code integrity.

3.4.8 Docker

Docker [48] was employed to containerize various components of the application, including the Mosquitto MQTT broker and MySQL database. This containerization ensured that the components could be easily deployed, scaled and maintained across different environments. By using Docker Compose, the deployment of multiple containers was streamlined, ensuring that all services operated seamlessly together. Docker's portability simplified the setup process, reducing the likelihood of compatibility issues across different deployment platforms.

4 Development of the Smart Agricultural Cabinet

This chapter presents the process of transforming the conceptual design of the smart agricultural cabinet into a functional prototype. The development phase, guided by agile methodologies, prioritized flexibility and iterative improvements to address the practical challenges of small-scale agriculture. This approach ensured that the system not only met its planned objectives but also adapted effectively to evolving requirements identified during implementation.

The next sections detail each stage of the development process, including the programming of microcontrollers, the integration of sensors and the implementation of the system orchestrator. Additionally, the development of the mobile and web applications is explored, highlighting their role in providing intuitive and efficient access to the cabinet's functionalities. This chapter offers a view of the technical and methodological efforts undertaken to realize the smart agricultural cabinet as a practical solution for indoor agriculture.

4.1 Controllers and Sensors

The design of the smart agricultural cabinet involves the integration of a range of sensors and actuators to enable comprehensive monitoring and automated management of the system. Each section is equipped with its own set of controllers and sensors, carefully chosen based on a detailed review of the literature to ensure optimal performance.

4.1.1 Hardware Used

A. **Raspberry Pi 4 model B**
The Raspberry Pi 4 model B was selected for the system because of its improved processing power and energy efficiency, making it well-suited for IoT applications that demand real-time data handling. With built-in wireless Internet capabilities and seamless integration with IoT frameworks, it offers the flexibility needed for such tasks. Like its predecessor, the Raspberry Pi 3 Model B+, it combines low energy consumption with broad software compatibility, ensuring reliable performance for managing computational tasks and wireless communication [49]. Additionally, its cost-effectiveness relative to cloud-based solutions was a critical consideration, offering an economical yet efficient alternative for the implemen-

tation of the system.

Figure 4.1: Raspberry Pi 4 model B used in the smart agricultural cabinet

B. **ESP32 microcontroller**
After the literature review carried out, the ESP32 microcontroller was selected as the primary controller due to its versatility, low power consumption and integrated Wi-Fi and Bluetooth capabilities [27]. These features make the ESP32 ideal for real-time communication with other components of the system while maintaining efficient energy use. Its powerful processing capabilities allow it to handle multiple sensor readings and control actuators effectively, ensuring reliable operation of the system. To further enhance the robustness of the system, an additional ESP32 with an integrated camera module, known as the ESP32-CAM, was incorporated. This microcontroller is primarily responsible for capturing periodic images of the plants, enabling visual monitoring for growth assessment and early detection of issues such as diseases or pest infestations. The ESP32-CAM is also connected to the ventilation system, which it controls independently to reduce the load on the primary ESP32 microcontroller, thus optimizing resource allocation.

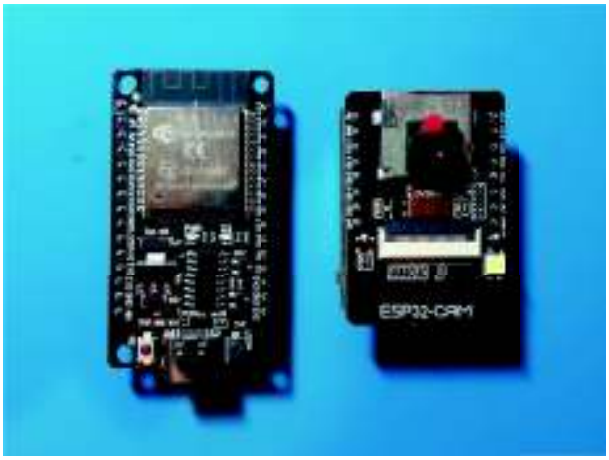


Figure 4.2: ESP32 and ESP32-CAM used in the smart agricultural cabinet

C. **DHT22**

The DHT22 sensor was selected to measure temperature and relative humidity of the section. Compared to its predecessor, the DHT11, the DHT22 offers a wider range of measurement values, with temperature readings from -40°C to 80°C and humidity levels from 0% to 100%, ensuring precise monitoring [28]. These measurements are crucial for maintaining optimal growing conditions within the cabinet.

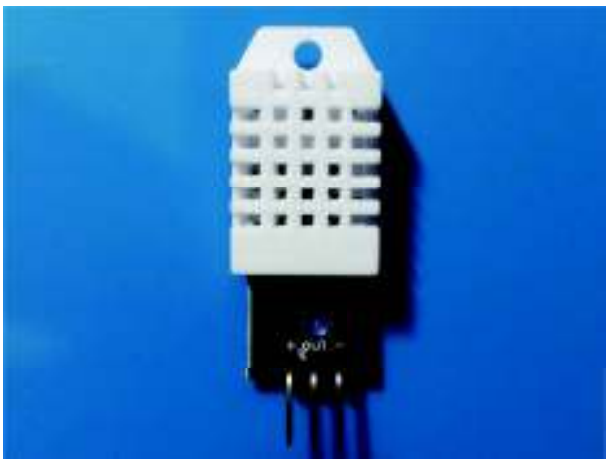


Figure 4.3: DHT22 sensor used in the smart agricultural cabinet

D. **Soil sensors**

To evaluate soil quality, the NPKPHCTH-S sensor has been incorporated to deliver comprehensive information on vital soil nutrients, such as nitrogen (N), phosphorus (P), and potassium (K). Additionally, it evaluates soil pH, electrical conductivity (EC), temperature and moisture content. This sensor facilitates a thorough understanding of soil health, enabling more informed decisions regarding nutrient management and irrigation, thereby promoting enhanced crop growth and sustainability.

For efficient and cost-effective soil moisture monitoring, a capacitive soil moisture sensor was chosen. Unlike resistive sensors, the electrodes of capacitive sensors are coated with a corrosion resistant material providing a long service life when compared with resistive sensors that tend to corrode [29]. This sensor is particularly advantageous for the automatic operation of irrigation systems, allowing for more effective water usage by adjusting irrigation levels according to real-time soil moisture data.

Figure 4.4: Soil moisture sensor used in the smart agricultural cabinet

E. **Irrigation system**
The irrigation system utilizes a water pump that delivers precise amounts of water to the plants. By integrating this pump with a soil moisture sensor, the system automates irrigation based on the real-time moisture levels in the soil. This effectively prevents both overwatering and underwatering, conserving water resources while ensuring that the plants receive the optimal hydration needed for healthy growth [18].

Figure 4.5: Water pump used in the smart agricultural cabinet

F. **Lightning system**
The cabinet's lighting system is equipped with a full-spectrum phytolamp that emits pink and purple light, specifically designed to enhance plant photosynthesis [50]. This optimal light spectrum balances red and blue wavelengths, which are vital for various stages of plant development. By choosing this type of lighting, we not only foster healthier plant growth but also achieve greater energy efficiency compared to conventional lighting systems [51].

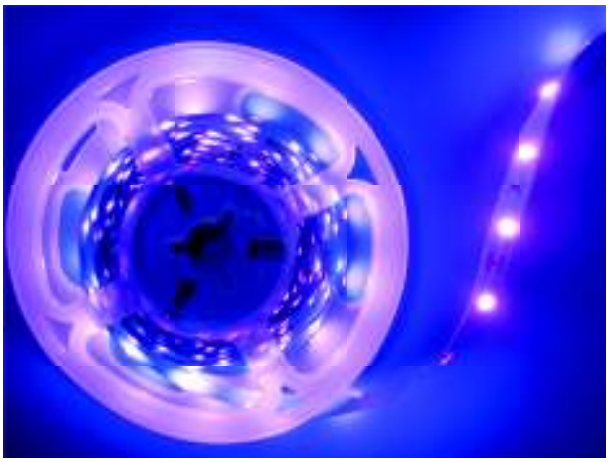


Figure 4.6: Phytolamp used in the smart agricultural cabinet

4.1.2 Development

The software implemented on the ESP32 microcontroller was developed utilizing the Arduino framework, with the Arduino Integrated Development Environment (IDE) serving as the primary code editor. This selection promoted an efficient development process, given the extensive array of libraries and comprehensive documentation available within the Arduino ecosystem.

The development process started with establishing a connection between the ESP32 and the Wi-Fi network created by the Raspberry Pi 4. This connection was vital for allowing the ESP32 to access the MQTT broker deployed locally on the Raspberry Pi.

Once the Wi-Fi and MQTT connections were successfully configured, the integration of sensors responsible for environmental data collection for each cabinet section was implemented. The key sensors include the DHT22, which measures ambient temperature and humidity, and a soil moisture sensor for monitoring soil conditions. These sensors were carefully selected for their accuracy and reliability in IoT applications.

Figure 4.7 illustrates the connections between the ESP32 and the DHT22 sensor, which collects temperature and humidity data. In this diagram, the red and black wires correspond to the VCC and GND pins, respectively, providing power to the sensor. The green wire is connected to the data pin of the DHT22, which is mapped to pin 27 on

the ESP32. This setup allows the ESP32 to read real-time temperature and humidity data from the sensor.

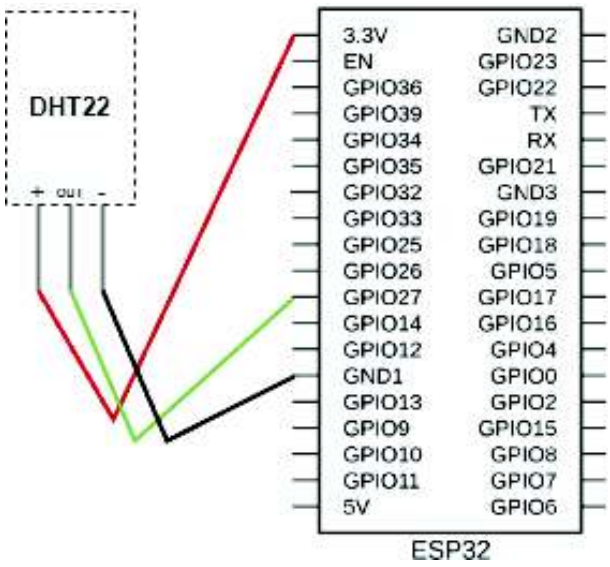


Figure 4.7: Connection between DHT22 sensor and ESP32

The soil moisture sensor, which monitors the hydration levels of soil, is connected to pin 34 of the ESP32, as depicted in Figure 4.8. This sensor operates based on a capacitive principle, offering enhanced resistance to corrosion and improved longevity compared to resistive alternatives. In the diagram, the red and black wires are designated for the VCC and GND connections to power the sensor. Meanwhile, the green wire transmits the analog output from the soil moisture sensor to pin 34 on the ESP32, allowing for real-time readings of soil moisture levels. This pin was specifically programmed to receive analog input values, taking advantage of its built-in 12-bit Analog-to-Digital Converter (ADC). This setup allows the sensor's output, which varies between 0 and 4.2V, to be precisely converted into 4096 distinct digital values [29]. The calculation to determine this result follows the formula:

$$2^{12} = 4096$$

This shows how the 12-bit resolution of the ADC converts the sensor's voltage range into 4096 unique digital values, allowing for accurate measurement and monitoring.

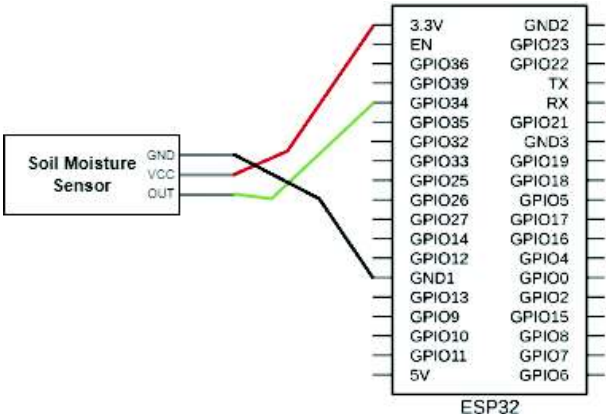


Figure 4.8: Connection between soil moisture sensor and ESP32

Thus, an Arduino function was developed to process the analog values captured by the sensor. These values are then converted into percentages, as shown in the following code:

```
float readSoilMoisture() {
    int rawValue = analogRead(34);

    float moisturePercentage = (rawValue / 4095.0) * 100;

    return moisturePercentage;
}
```

Listing 4.1: Arduino function to read Soil Moisture on ESP32

Figure 4.9 presents the initial stage of the system assembly, showcasing the integration of the DHT22 sensor alongside the Soil Moisture sensor, both of which have been successfully connected to the ESP32 microcontroller.

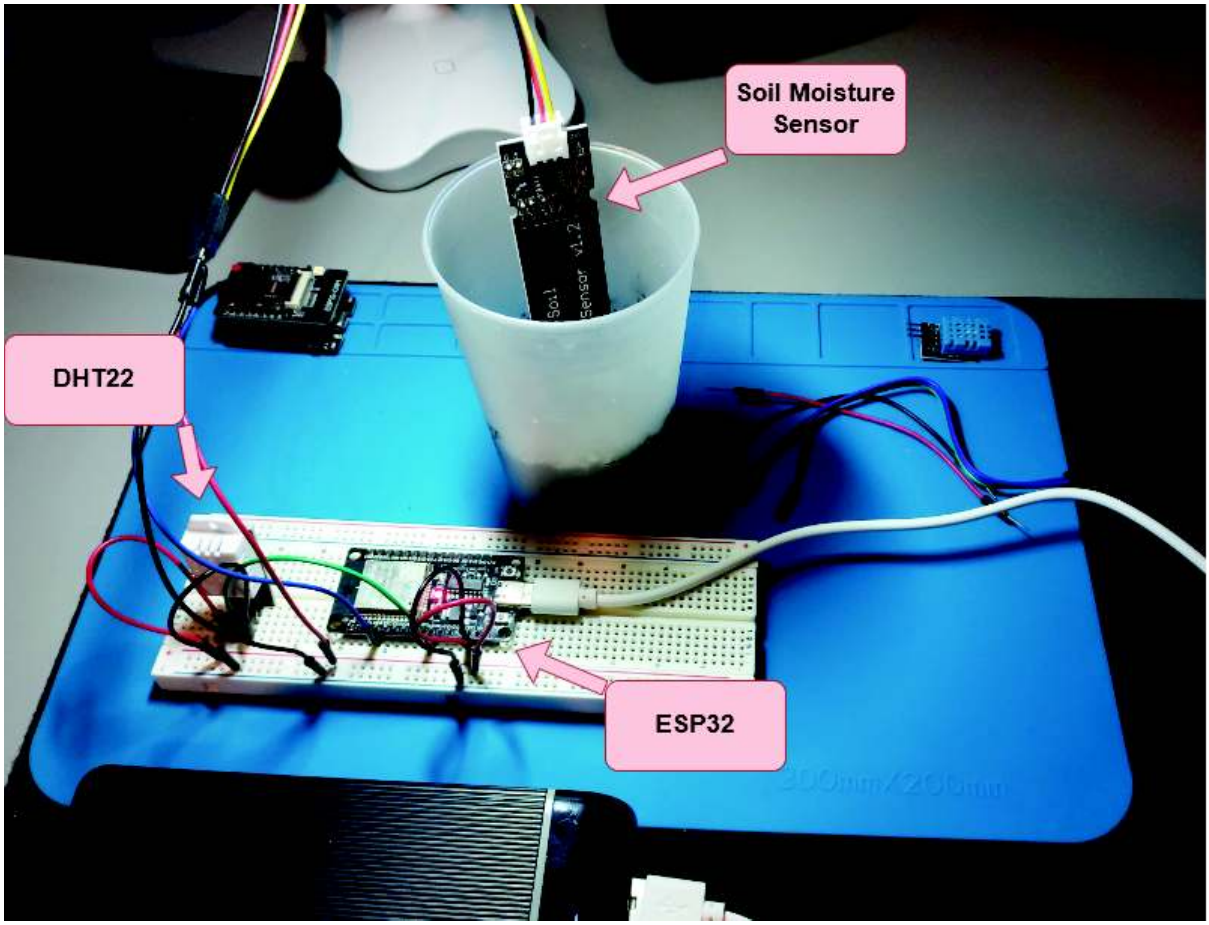


Figure 4.9: Soil moisture and DHT22 sensors setup with ESP32 on a breadboard

The ESP32 processes the data collected from both sensors and transmits it via MQTT to the system orchestrator. The orchestrator analyzes this data against predefined thresholds to determine when to activate the irrigation system. For instance, if the soil moisture level falls below the designated threshold, the orchestrator will activate the irrigation system to maintain optimal conditions.

The process of the irrigation system, illustrated in Figure 4.10 begins with the soil moisture sensor, which continuously measures the water content in the soil. When the detected moisture level falls below a predefined threshold, the sensor generates a signal that is subsequently transmitted to a relay module for further processing [11]. The relay, functioning as an electrically controlled switch, regulates the power supply to the water pump. Upon receiving an activation signal, the relay transitions from its default open state to a closed state, thereby completing the electrical circuit and supplying power to the pump [18]. This activation initiates the irrigation process, delivering water to the designated area. Once the soil moisture sensor detects that the moisture content has reached or exceeded the configured threshold, it transmits a deactivation signal to the relay. In response, the relay reverts to its open state, disconnecting the power supply to the pump and stopping the irrigation.

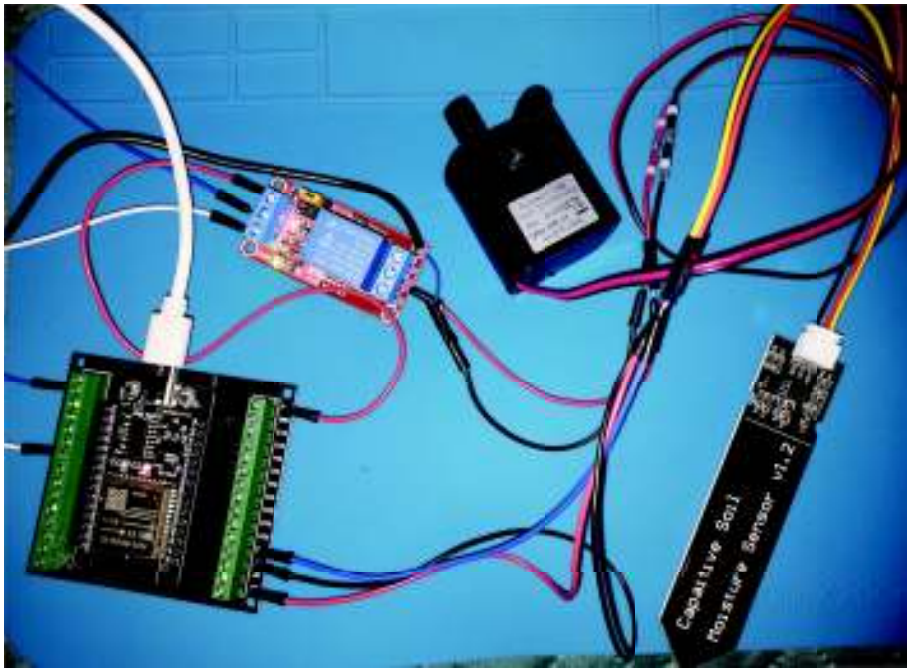


Figure 4.10: Irrigation system assembly

The lighting system's assembly, illustrated in Figure 4.11, is designed to ensure optimal light exposure for plant growth within the smart agricultural cabinet. The LEDs emit specific light tailored to support photosynthesis and enhance plant development.

As in the irrigation system, a relay module is used to regulate the power supply to the LED grow lights, controlled by the ESP32 microcontroller. When the user activates the lights via the app, a signal is sent to the ESP32 microcontroller, which triggers the relay module to power the LEDs. Similarly, the app can be used to turn off the lights when desired, ensuring complete user control over the lighting system. This integration of the relay ensures automated and precise control of the lighting system, optimizing energy efficiency while providing the necessary light conditions for healthy and sustainable plant growth.

Figure 4.11: Lightning system assembly

Moreover, sensor readings are relayed to a mobile application, enabling users to monitor real-time conditions remotely.

4.2 System Orchestrator

The system orchestrator serves as the central management unit of the smart agricultural cabinet, responsible for supervise the operation and coordination of all system components. Its primary function is to ensure seamless integration across sections, managing everything from section registration, user administration and real-time environmental control. As depicted in Figure 4.12, the orchestrator acts as the hub of the core communication, connecting microcontrollers, the mobile application and the MySQL database using the MQTT protocol.

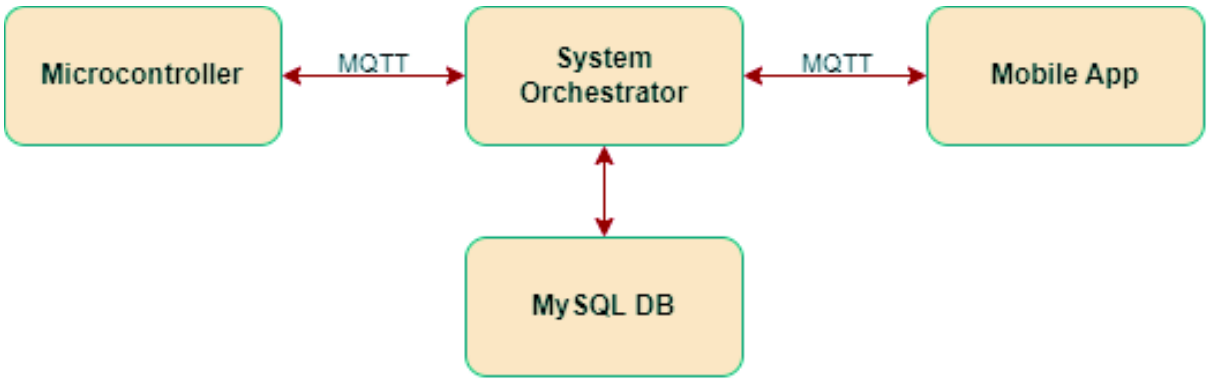


Figure 4.12: System orchestrator interaction with system components

The development of the system orchestrator was implemented using Python, chosen for its versatility and extensive support for various libraries that facilitate rapid development. Using the paho-mqtt library, the orchestrator connects with the MQTT broker, enabling efficient communication with the microcontrollers and the mobile application. The following code snippet demonstrates the core logic implemented to establish communication with the MQTT broker:

```
import paho.mqtt.client as mqtt

Mqtt.host = mqtt_config.get('host', 'localhost')
Mqtt.port = int(mqtt_config.get('port', '1883'))
Mqtt.client_id = mqtt_config.get('client_id', 'iot_handler')

Mqtt.mqtt_client = mqtt.Client(Mqtt.client_id)
result = Mqtt.mqtt_client.connect(Mqtt.host, Mqtt.port)
```

Listing 4.2: Python code with MQTT configuration

Using MQTT for data transmission between the system's components ensures efficient communication. Its lightweight protocol reduces latency and conserves bandwidth, essential for IoT applications requiring real-time responsiveness [42].

Once communication with the MQTT broker is established, the orchestrator leverages the mysql.connector library to establish a connection with the MySQL database. The code snippet below summarizes the core logic implemented for this database connection:

```
import mysql.connector

MySQLMng.host = mysql_config.get('host', 'localhost')
MySQLMng.port = int(mysql_config.get('port', '3306'))
MySQLMng.user = mysql_config.get('user', 'root')
MySQLMng.password = mysql_config.get('password', 'smcabinet_db')
MySQLMng.db_name = mysql_config.get('db_name', 'smart_cabinet')
MySQLMng.connect()
```

Listing 4.3: Python code to connect to MQTT

One of the core responsibilities of the orchestrator is the automatic control of the cabinet's environmental parameters, such as irrigation and ventilation. By analyzing sensor readings in real-time and comparing them with predefined threshold values for each section stored in the MySQL DB, the system can autonomously activate or deactivate the irrigation and ventilation systems. Based on the analysis of the study conducted in [52], specific threshold values for certain microgreens, including lettuce and Brussels sprouts, were collected (Table 4.1). After the collection of these values, they were submitted for validation by the representative of *Verdes do Mondego*, who verified their accuracy and confirmed that they closely correspond to the values utilized in their current production processes.

Table 4.1: Environmental factors levels for lettuce and brussels sprouts [52]

Environmental factor	**Lettuce (Range)**	**Brussels Sprouts (Range)**
Light Intensity	500 μmol $m^{-2}s^{-1}$	300 $\pm$ 15 μmol $m^{-2}s^{-1}$
Temperature	20 $\pm$ 2 °C	18 - 24 °C
Relative Humidity	80 $\pm$ 5 %	70 - 80 %
pH	6.3 $\pm$ 0.4	6.0 $\pm$ 0.2
Electrical Conductivity	1.8 $\pm$ 0.2 mS	1.8 $\pm$ 0.2 mS

For instance, if soil moisture levels fall below a defined threshold, the orchestrator automatically activates the water pump to irrigate the plants. Similarly, using the predefined environmental thresholds for lettuce as an example, if the temperature exceeds 20 °C and the humidity surpasses 80%, the ventilation system is triggered to restore environmental balance These thresholds can be remotely configured by users through a dedicated mobile application, providing enhanced flexibility and adaptability to accommodate various plant species and cultivation conditions. The orchestrator processes these threshold adjustments in real-time, ensuring their immediate application, while also storing them securely in the MySQL database for future reference and system integrity.

Beyond environmental control, the orchestrator also handles user management and section registration. Users can register through the mobile application and their information is stored in the MySQL database. The orchestrator manages user authentication and access control, ensuring that only authorized users can modify system settings or adjust section parameters. Additionally, new sections can be registered into the system, with each section having its own unique configuration and thresholds tailored to the specific requirements of the plants being grown.

Finnaly, the transmission of data to the Blynk platform for monitoring is also handled by the system orchestrator. This process involves making an HTTP request containing the value of the virtual pin associated with a specific type of data, as predefined in the Blynk platform, along with the corresponding metric value. For example, temperature data is linked to the virtual pin V0, and if the temperature is 23 °C, the code to send a message with this information would be as illustrated in the following code snippet:

```
def send_to_blynk():
    try:
        pin = "V0"
        value = "23"
        blynk_url = f"https://blynk.cloud/external/api/update?token=<BLYNK_TOKEN>&{pin}={value}"

        response = requests.get(blynk_url)
        if response.status_code == 200:
            Logger.info(f"Data sent to Blynk successfully: {value}")
        else:
            Logger.error(f"Failed to send data to Blynk: {response.text}")
    except Exception as e:
        Logger.error(f"Error forwarding data to Blynk - {e}")
```

Listing 4.4: Python code to send sensor data to the Blynk platform

4.3 Mobile Application

The mobile application serves as the primary interface for users to monitor and control the smart agricultural cabinet, offering a convenient way to interact with the system remotely. Developed using the Flutter framework, the application provides a unified experience across both Android and iOS platforms, ensuring accessibility and ease of use for a diverse range of users. The authors in [25] demonstrate Flutter's effectiveness in creating cross-platform IoT applications, highlighting how Flutter facilitates real-

time monitoring and control capabilities in IoT-based systems, making it particularly suitable for smart agricultural solutions.

The application's architecture was designed to be intuitive, facilitating seamless communication with the backend system via the MQTT protocol. The use of Flutter also enhances development efficiency by enabling a single codebase for multiple platforms, a feature noted in the previous study [25] as crucial for reducing development time and ensuring consistent user experiences. These characteristics establish Flutter as an optimal framework for systems demanding real-time user engagement and seamless IoT integration.

4.3.1 User Interface (UI) Design and Mockups

The design process for the mobile application prioritized user experience and ease of navigation. A series of wireframes and mockups were developed to visualize the layout and flow of the application before actual development commenced. As shown in Figure 5.2, the UI focuses on providing clear and intuitive access to system features, such as real-time monitoring of sensor data, control of irrigation and ventilation systems, and customization of environmental thresholds.

Key screens include a dashboard displaying current environmental metrics (such as temperature, humidity, soil moisture and nutrient levels) and a control panel where users can adjust settings for each section of the cabinet. Additionally, notifications and alerts are integrated to inform users of significant changes, such as when soil moisture levels fall below the set threshold or when the irrigation system has been automatically activated.

Figure 4.13: Mockups for the mobile application

To ensure the development of a mobile application that is both visually appealing and functional, the mockups were carefully reviewed in collaboration with *Verdes do Mondego*. Their feedback was positive, expressing full support for the application design presented. Beyond their approval, additional feedback was gathered from two experienced computer engineers. One, a project manager with expertise in front-end development, recommended incorporating features like loading progress bars to enhance the user experience during times when the application is processing or waiting for data. The second, a software engineering graduate, also reviewed the mockups and found no significant areas for improvement, fully supporting the proposed design.

4.3.2 Application Development

The development of the mobile application leveraged the Flutter framework, chosen for its ability to create high-performance, cross-platform applications from a single codebase. Flutter's extensive library of pre-built widgets and robust support system significantly streamlined the development process, ensuring that the application remained both lightweight and efficient.

To facilitate smooth communication with the system, the application employs the MQTT protocol alongside the *mqtt-client* library to establish a connection with the system orchestrator. This integration enables real-time data exchange between the microcontrollers and the mobile application, allowing users to receive immediate updates on environmental conditions and make necessary adjustments. For example, if the system identifies that soil moisture levels have dipped below a predetermined threshold, the application promptly notifies the user, who can then remotely modify the irrigation settings to restore optimal conditions.

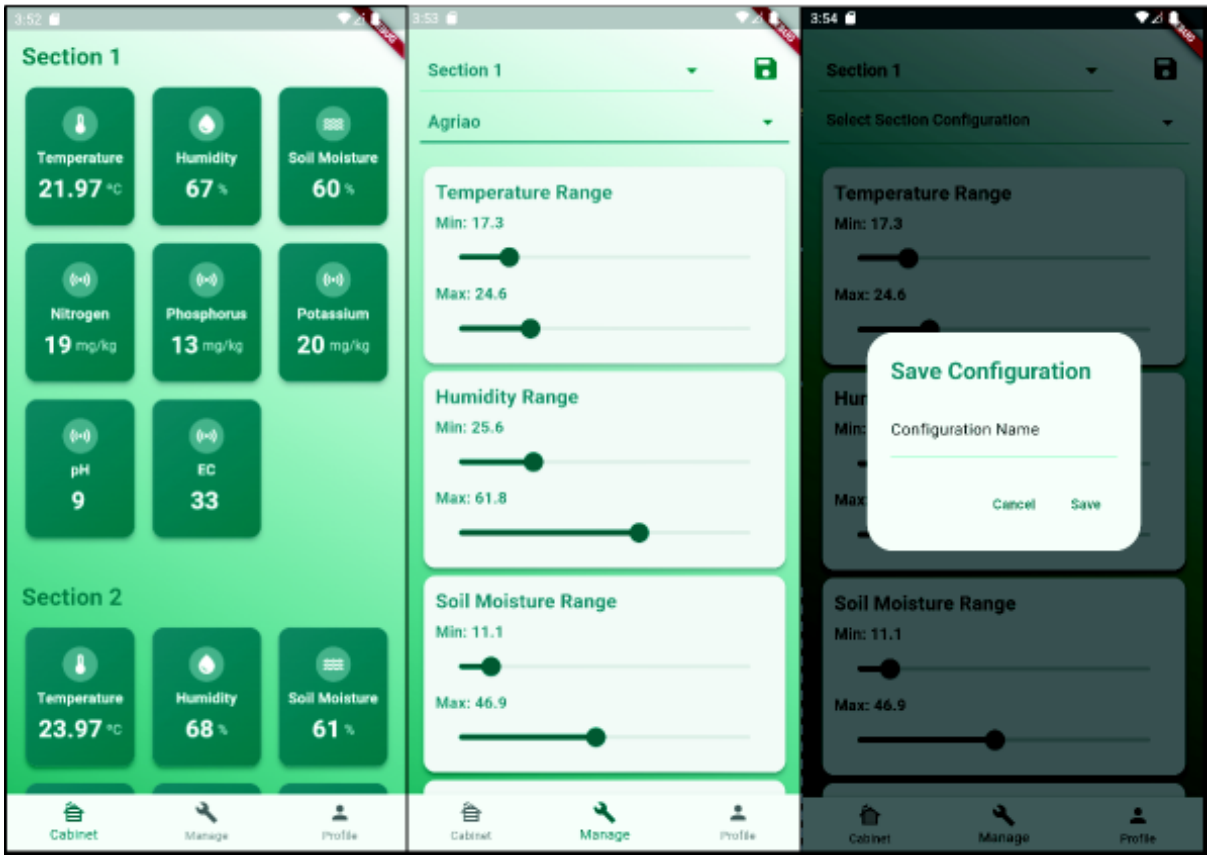


Figure 4.14: Screenshots from the mobile application

This application (Figure 4.14) is designed with the user in mind, offering features for system management and monitoring. A key functionality of the mobile application is its integration with the system orchestrator to manage automated controls. Users can define specific thresholds for crucial environmental parameters directly through the mobile interface. For example, if a section is configured with a minimum soil moisture threshold of 15%, the system automatically activates the irrigation system whenever the sensor readings fall below this set value. When users adjust these thresholds through the mobile app, an MQTT message is sent to the system orchestrator. Upon receiving this message, the orchestrator promptly updates the relevant values in the system's database, associating them with the specified section. This workflow is represented in Figure 4.15. This communication allows the system to dynamically adapt to changing conditions based on real-time inputs, ensuring precise control over irrigation and other automated processes.

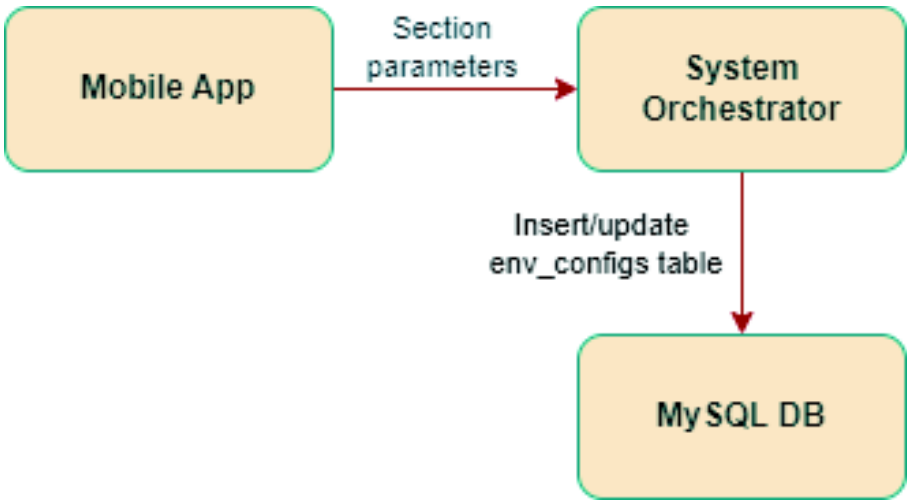


Figure 4.15: Workflow for updating the environmental parameters of a section

4.4 Web Application

The web application was developed using Blynk, a low-code platform widely employed in intelligent systems for monitoring IoT devices. This platform uses a cloud-based communication service [11] to communicate with the system orchestrator. Blynk offers a free usage plan with some limitations, such as a maximum limit of 30000 messages per month that can be received from a device. However, with efficient management of the messages sent to this platform, it is feasible to use Blynk as a monitoring tool for proof-of-concept systems.

To create the monitoring dashboard, the first step involved adding the type of device that would transmit data to the dashboard, a Raspberry Pi in this case, which would send data via WiFi. Figure 4.16 illustrates the creation of a Blynk template, which served as the foundation for building the monitoring dashboard.

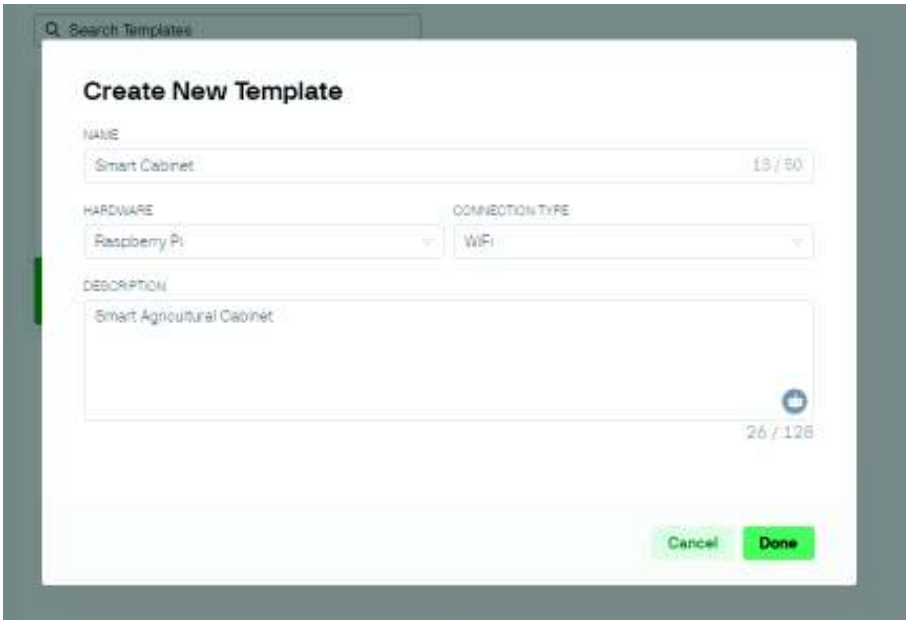


Figure 4.16: Creation of the web application dashboard using Blynk

For displaying sensor values, such as temperature and humidity, Gauge widgets were selected for each type of measurement. Within each widget, the type of value to be received was defined, along with a virtual pin to associate with the data and the corresponding metric unit, as shown in Figure 4.17.

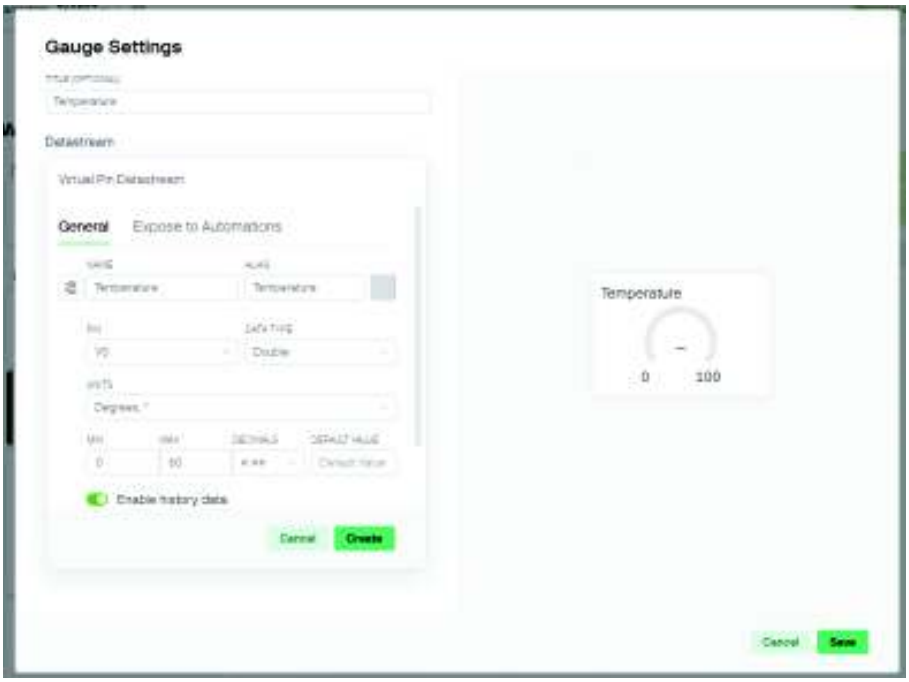


Figure 4.17: Create a gauge graphic to display sensor data

After designing a chart for each type of sensor data collected, the system orchestrator was programmed to route the sensor data from microcontrollers to the web application.

This was achieved using the requests library, which facilitates the sending of HTTP requests. These type of communication were necessary to transmit sensor values to their respective virtual pins. The HTTP request used by the system orchestrator to send the collected sensor data is structured as follows:

https://blynk.cloud/external/api/update?token=BLYNK-AUTH-TOKEN&VIRTUAL-PIN=VALUE

Here, BLYNK-AUTH-TOKEN corresponds to the unique token required for authentication and connection to the platform.

By sending this GET request to Blynk, each type of data, properly linked to a virtual pin, is kept updated with the latest sensor readings. This enables real-time analysis of the data from the smart agriculture system from any location, overcoming the limitation of local-only communication when the system is restricted to the WiFi network created by the Raspberry Pi.

Figure 4.18 illustrates the final web application developed using the Blynk platform, displaying real-time data from the system.

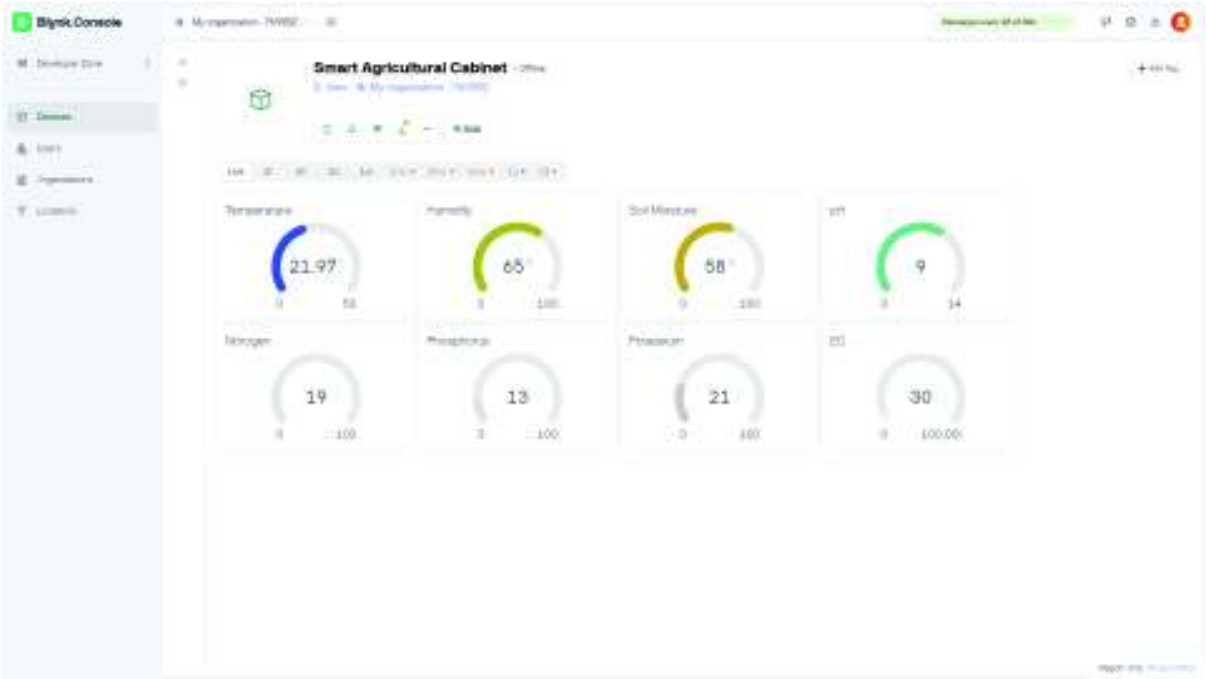


Figure 4.18: Web application displaying sensor data

5 Tests and Results Analysis

This section presents the tests and results analysis of the smart agricultural cabinet prototype, combining functional and non-functional testing to assess its effectiveness. The results are analyzed within the context of the system's objectives and its practical application in real-world scenarios.

The final smart cabinet prototype, illustrated in Figure 5.1, marks the consummation of the project's design and development. This prototype integrates different components, such as sensors, actuators, microcontrollers and a system orchestrator, forming a robust farming solution.

Using this type of system and corroborating the observations made during the conducted literature review, the resource wastage could be decreased, especially water, while maintaining stable and optimal growing conditions. These successes illustrate the growing potential for IoT-enabled solutions to drive efficiency and sustainability in agriculture.

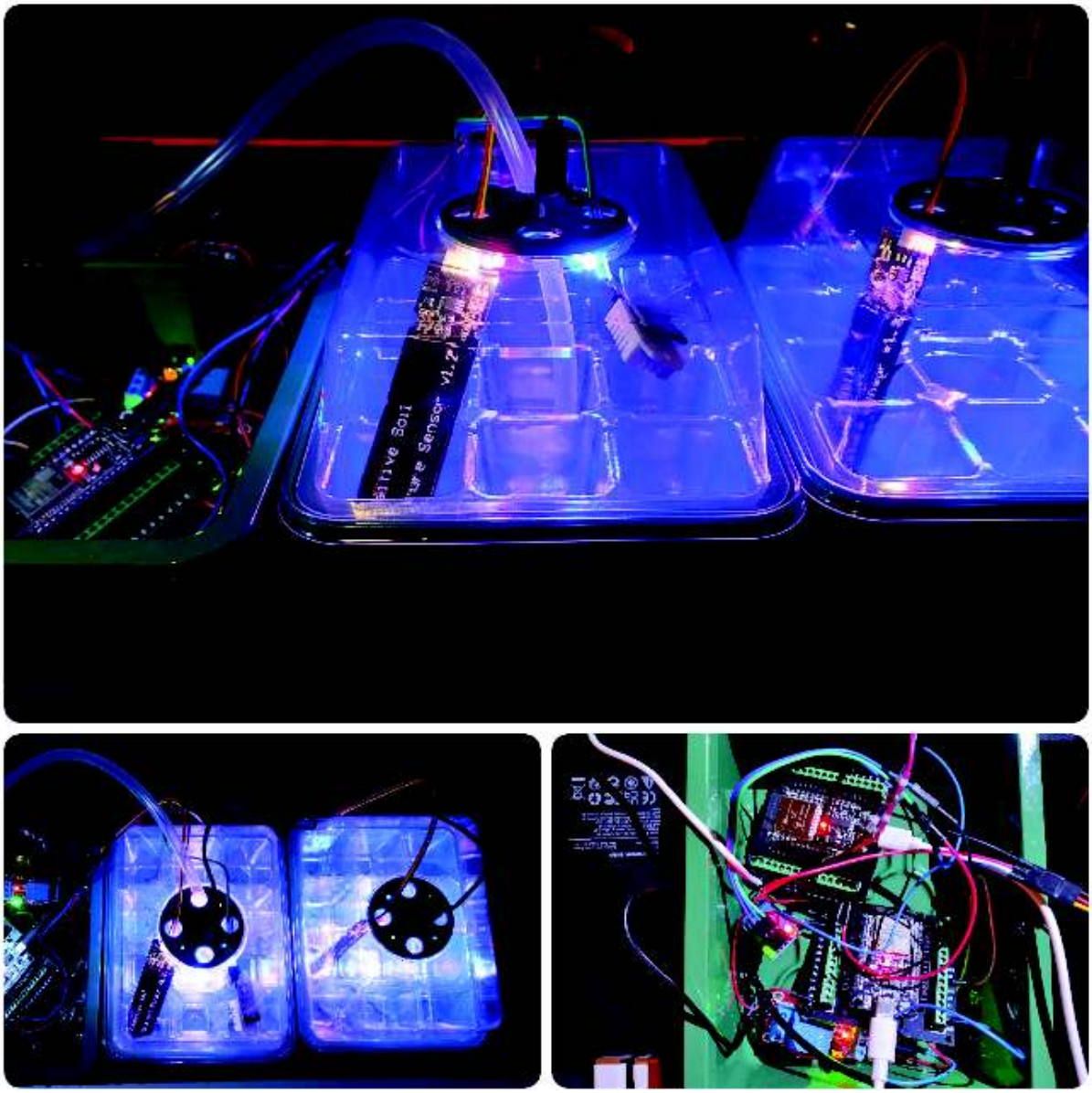


Figure 5.1: Final smart agricultural cabinet prototype

5.1 Partnership with *Verdes do Mondego*

The collaboration with *Verdes do Mondego*, a company specializing in the cultivation of microgreens through vertical farming techniques, played a pivotal role in the development of the smart agricultural cabinet. Founded by André Aleixo and embedded in sustainable and innovative agricultural practices, *Verdes do Mondego* provided invaluable insights that informed the design and functionality of the system. Their experience in managing controlled agricultural environments ensured that the system addressed practical needs while aligning with industry best practices.

Regular interactions with the representative of the company allowed for iterative refinements of the prototype, ensuring that both functional and non-functional requirements were met. Specific recommendations from the company included the calibration

of environmental thresholds such as temperature and humidity, which were integral to maintaining optimal conditions for the microgreens.

The Figure 5.2 illustrates the different growth stages of cress, showcasing the effectiveness of the system used for cultivating microgreens. The seeds, provided by *Verdes do Mondego*, were observed at three distinct intervals: 5 days, 8 days and 10 days. This progression highlights the system's efficiency in supporting the consistent and healthy growth of microgreens, highlighting its potential for sustainable small-scale agriculture.

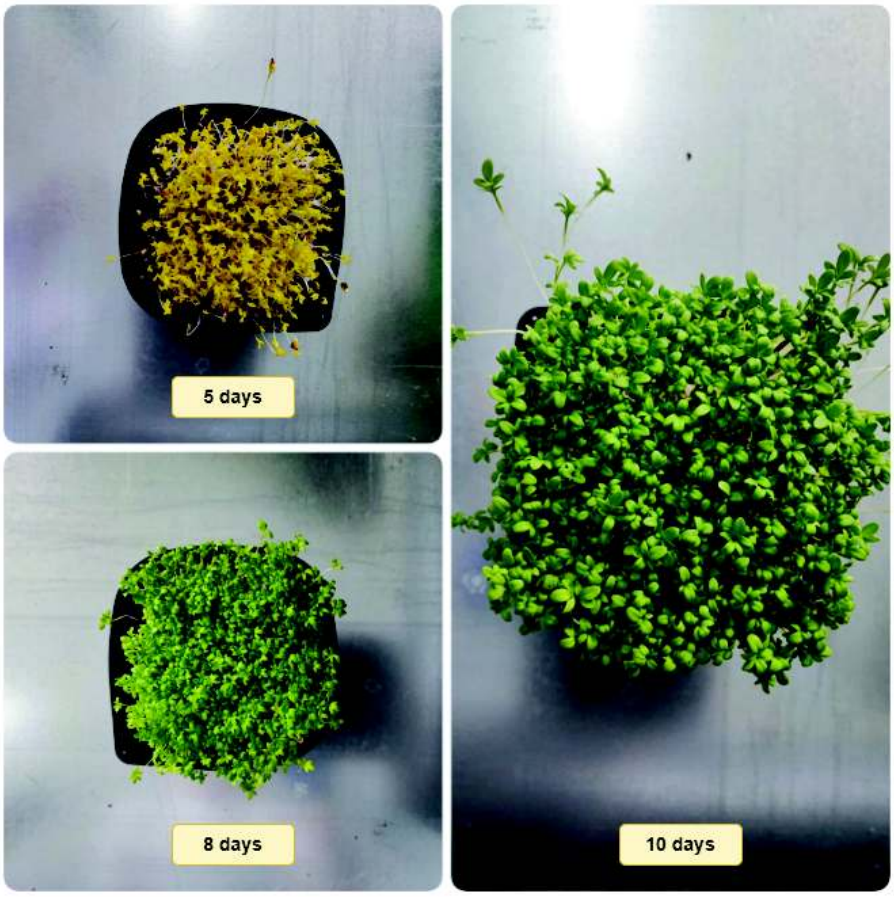


Figure 5.2: Growth stages of cress using the smart agricultural cabinet

This collaboration not only validated the system's relevance but also reinforced its alignment with real-world agricultural processes. By automating key aspects of cultivation, the smart cabinet aims to enhance operational efficiency and consistency, contributing to *Verdes do Mondego's* commitment to sustainable and high-quality production.

5.2 System Validation

System validation was a critical phase in assessing the functionality and reliability of the smart agricultural cabinet. This process involved comprehensive testing to ensure that the system met both functional and non-functional requirements defined during the development phase. The validation process was divided into three key aspects: functional tests, non-functional tests and user experience evaluations.

- **Functional validation:** The functional validation phase focused on ensuring that all core features operated as intended. Tests included verifying the accuracy of sensor readings, as well as the responsiveness of actuators controlling irrigation, lighting and ventilation systems. Real-time data transmission via the MQTT protocol was also validated, ensuring seamless communication between the hardware and software components. These tests confirmed that the system could autonomously monitor and regulate environmental conditions to optimize plant growth.
- **Non-functional validation:** Non-functional validation assessed attributes such as system reliability, scalability and performance under various conditions. Tests were conducted to evaluate the system's capacity to handle simultaneous sensor inputs and user interactions without performance loss.
- **User Experience validation:** The final stage of validation involved usability testing with input from Verdes do Mondego. Feedback was gathered on the intuitiveness and responsiveness of the mobile application, with particular emphasis on remote configuration and real-time monitoring features.

5.2.1 Functional and Non-functional Tests

One of the main methods used to validate the developed system was the execution of functional tests, an approach also adopted by the authors in the work described in [3]. The primary goal of these tests is to verify whether the system meets the functional requirements and use cases previously outlined in Chapter 3. To achieve this, the system was prepared and configured to ensure a comprehensive and accurate evaluation, enabling the application of this testing method and confirming whether all functionalities adhered to the criteria established for this system. The Table 5.1 presents all the functional tests covered, the components involved in the test and the final result.

Table 5.1: Functional tests cases for the smart agricultural system

ID	Components Involved	Test Case	Result
TC1	Sensors	Collect temperature, humidity and soil moisture data	Passed
TC2	Sensors	Image capture of plants	Failed
TC3	Sensors, System Orchestrator	Automated lighting control	Passed
TC4	Sensors, System Orchestrator	Automated ventilation control	Failed
TC5	Sensors, System Orchestrator	Automated irrigation based on soil moisture levels	Passed
TC6	System Orchestrator, Mobile Application	Display sensor data in the mobile application	Passed
TC7	System Orchestrator, Web Application	Display sensor data in the web application	Passed
TC8	Mobile Application, Web Application	Display sensor data history	Passed
TC9	System Orchestrator, Mobile Application	Send alerts and notifications	Passed
TC10	Mobile Application, Sensors	Remote control of ventilation system	Passed
TC11	Mobile Application, Sensors	Remote control of irrigation system	Passed
TC12	Mobile Application, Sensors	Remote control of lightning system	Passed
TC13	Mobile Application, System Orchestrator	Overwrite environment settings manually	Passed

Functional tests focused on evaluating whether the system's core features worked as intended. Each test case had the purpose of validating a specific aspect of the cabinet's functionality:

- **Sensor data collection (TC1 and TC2)**
  These tests were conducted to validate the data collection from the sensors and the image capture functionality of the ESP32-CAM. Both requirements were met, successfully passing the tests.
- **System automation (TC3, TC4 and TC5)**
  The tests were performed to validate the automation of the system by assessing whether it effectively activated the ventilation, irrigation and lighting systems based on the values received from the sensors. Unfortunately, the automation requirements for the ventilation system was not fulfilled, as the necessary logic was

not implemented, resulting in failures in these feature. In contrast, the irrigation and lighting systems functioned as expected and successfully passed the test.

- **Data monitoring (TC6, TC7, TC8 and TC9)**
  These tests aimed to validate the monitoring of data collected by the sensors and its accurate display on the mobile and web applications. Additionally, they evaluated the system's ability to send notifications when sensor values deviated from the predefined thresholds for each data type. The tests were successfully completed.
- **Mobile remote control (TC10, TC11, TC12 and TC13)**
  Finally, tests were conducted to validate the requirements related to remote system control via the mobile application, including managing the irrigation system and configuring the environmental values for each section of the cabinet. These requirements are implemented in the system and successfully passed the functional tests.

To validate the non-functional requirements outlined in Table 3.4, non-functional tests were defined and are presented in Table 5.2:

Table 5.2: Non-functional tests cases for the smart agricultural system

ID	Components Involved	Test Case	Result
TC14	System Orchestrator, Sensors	System scalability	Passed
TC15	Mobile Application	Mobile application usability	Passed
TC16	System Orchestrator	Security of user credentials	Failed
TC17	System Orchestrator	Data security	Failed
TC18	Web Application, Mobile Application	Cross-platform compatibility	Passed

- **System scalability (TC14)**
  This test aimed to validate the system's ability to expanding cabinet sections. The test confirmed that the system's architecture could scale without degradation in performance, successfully passing the test.
- **Mobile application usability (TC15)**
  Usability testing focused on evaluating the mobile application's user interface and user experience. For this validation, a form was created with some questions about the user's experience using the mobile application. The answers confirmed that the application was intuitive and user-friendly. This requirement was successfully validated.
- **Security of user credentials (TC16)**
  This test was conducted to ensure the protection of user credentials during reg-

istration and authentication. The system failed this test due to the lack of use of encryption mechanisms.

- **Data security (TC17)**
  As in TC16, data security testing verified the system's ability to protect data, such as sensor data, during transmission and storage. This test failed, highlighting vulnerabilities in the system.
- **Cross-platform compatibility (TC18)**
  This test aimed to validate the operation of the web and mobile applications across different devices and platforms. The system passed this test, demonstrating full compatibility with Android, iOS and various web browsers.

  Availability testing assessed the system's uptime and reliability during continuous operation for 48 hours. The cabinet achieved 100% uptime, successfully passing the test.

5.2.2 User Experience

To evaluate the User Experience (UX) of the mobile application, the User Experience Questionnaire (UEQ) was employed as the primary evaluation tool. The UEQ is a method for measuring user perceptions of a product across six essential dimensions: Attractiveness, Perspicuity, Efficiency, Dependability, Stimulation and Novelty [53], making it a good choice for evaluating the mobile application developed in this project.

The Test Case 15, described in Table 5.2, aimed to collect feedback on the user interactions with the application. The survey, based on the UEQ framework and illustrated in the Figure 5.3, was distributed to a group of six participants with diverse backgrounds, including technological and healthcare sectors. This diverse group was selected to capture a wide range of perspectives, ensuring a comprehensive understanding of the application's usability. The results of the UEQ can be found in the Appendix A of this document.

obstructive	o o o o o o o	supportive
complicated	o o o o o o o	easy
inefficient	o o o o o o o	efficient
confusing	o o o o o o o	clear
boring	o o o o o o o	exciting
not interesting	o o o o o o o	interesting
conventional	o o o o o o o	inventive
usual	o o o o o o o	leading edge

Figure 5.3: User Experience Questionnaire

After collecting user responses, the data were input into the UEQ Data Analysis Tool [54]. This process generated a benchmark graph, represented in Figure 5.4, allowing the results to be compared with a reference dataset. Based on this analysis, it was observed that the user experience of the mobile application was rated as positive.

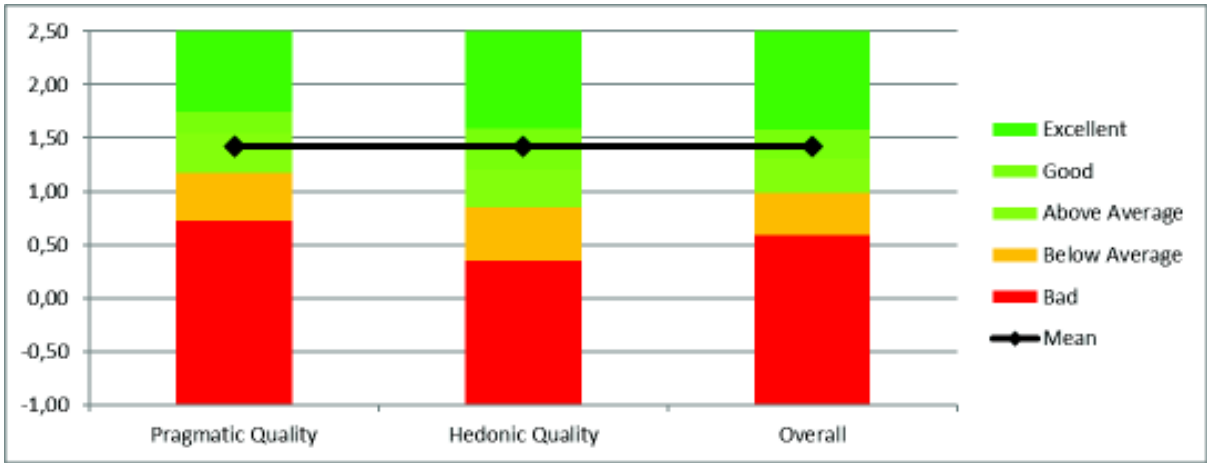


Figure 5.4: Benchmark graph with the User Experience results

6 Conclusion and Future Work

In this final chapter, the conclusions drawn from the implementation of this project will be presented. Additionally, future work will be outlined, focusing on potential enhancements and the integration of innovative technologies to expand the system's functionality.

6.1 Conclusion

This document has presented the study and development of a smart system prototype, whose objective is to minimize human intervention and make agricultural practice more efficient, especially domestic agriculture, integrating low-cost IoT technologies and open-source solutions. The project aimed to deliver an automated system capable of monitoring and controlling environmental parameters for plant growth.

The testing liberated the actual performance of the system in achieving the tasks set. The cabinet displayed accurate real-time reporting, effective management of resources and easy access and control with the mobile and web applications. The processes which were automated including irrigation, ventilation and data collection show the prospect of using IoT-based systems in controlling the environment with minimal human intervention to support growth and development of plants.

The partnership with *Verdes do Mondego* turned out to be most beneficial in the desire to customize the system to the practical farming context since having feedback from someone experienced in the field of biology and microgreens production in domestic systems provided a more technical perspective on plants and their requirements for healthy growth. Their insights during the development process ensured that the cabinet was specifically designed and optimized for indoor farming, which makes the system more appropriate and capable of being successful in real-world scenarios.

However, there were a few issues that were noted in the testing of the system. For instance, concerns were raised on the security of sensitive information such as user authentication and sensor data. In the same sense, and although the application of machine learning was something that was investigated, this functionality was also not applied in the developed system, which is why it also presents itself as a flaw in this project. These features will be addressed in the following section, forming part of the future work to be considered for this system.

This project represents a significant contribution to the advancing field of smart agri-

culture by delivering a scalable and practical solution that successfully connects advanced precision farming technologies with their practical implementation in domestic settings. The successful development and implementation of this prototype function as a robust proof-of-concept, affirming the feasibility of designing smart agricultural systems that are not only accessible but also capable of making a meaningful impact on sustainable and efficient small-scale farming practices.

6.2 Future Work

The smart agricultural cabinet is a promising development in terms of automating and improving small-scale farming. However, there is potential to improve it further by incorporating new technologies and resolving some more general agricultural issues. There is also the additional opportunity for further advancements to develop the system to be more comprehensive and efficient.

To mention some improvements, the possible addition of machine learning algorithms to the system would be one of them. These would enable the system to perform such functions as early disease detection, based on data provided by sensors and cameras. Such detection would facilitate adequate response measures, thereby saving crops from the losses they could have otherwise suffered. Furthermore, machine learning might make it possible for the system to leverage past data, such as key temperature and humidity levels or irrigation, and implement the best conditions for growth processes while maximally economizing on resources.

Another improvement for the system could be the predicting analytics to implement the anticipation of plants and environmental factors. For instance, it could be able to look through a bunch of data and recognize when plants were last watered which could then predict when the next watering would take place or when the temperature and humidity might shift. This sort of anticipation would enhance the reliability and efficiency of the system as potential risk factors for the health of the plants would be minimized.

Finally, cloud-based technologies could significantly add features to this system. With cloud-based storage and analytics, users can monitor and control the cabinet remotely from anywhere. Advanced analytical tools can give insights about the system's performance, thus allowing users to make more timely decisions for plant care and resource management.

Submitting this project to initiatives like the Poliempreende competition could showcase its innovative potential while providing valuable feedback and guidance for improvement. Participation could also support scaling the system and adapting it for broader agricultural and commercial applications.

By focusing on these areas for future development, the smart agricultural cabinet has the potential to become a more robust and user-friendly solution. Figure 6.1, created using Blender [55], presents a 3D prototype that envisions a more solid and durable structure for the cabinet. This conceptual design highlights future possibilities for improving its functionality and reliability, further enhancing its value as a practical tool for small-scale farmers.

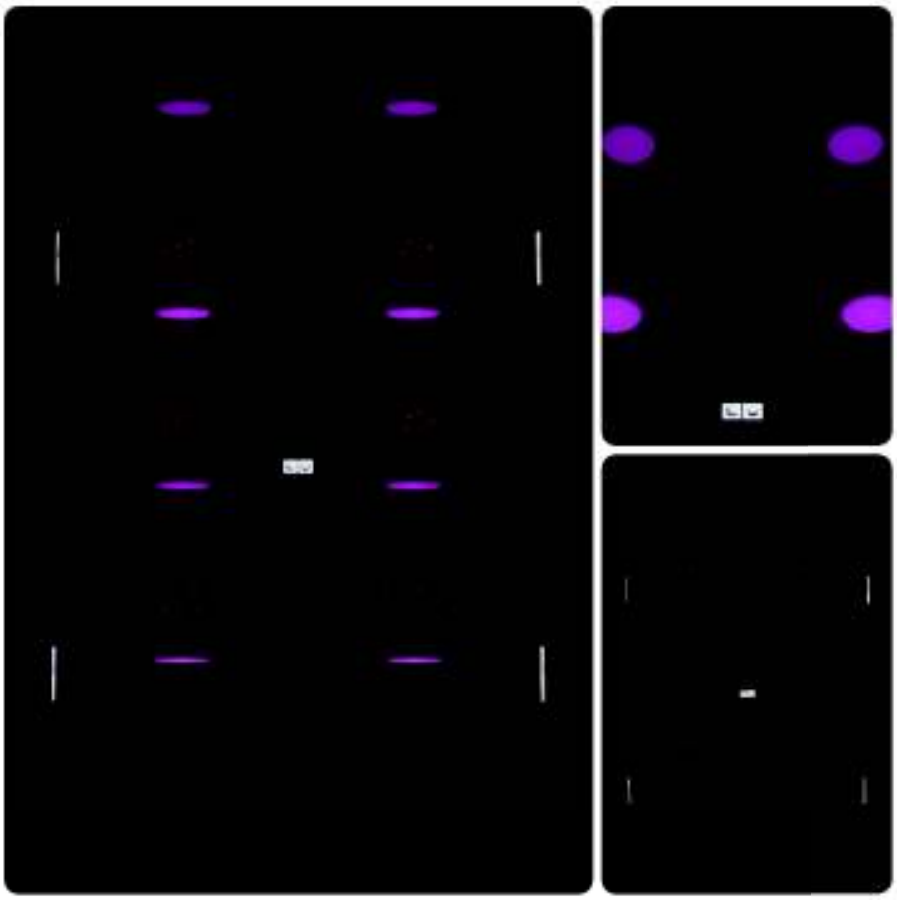


Figure 6.1: 3D prototype of the smart agricultural cabinet

Bibliographic References

[1] B. Cheruvu, S. B. Latha, M. Nikhil, H. Mahajan, and K. Prashanth, "Smart farming system using npk sensor," in *2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 3 2023, pp. 957–963.

[2] E. M. Karunathilake, A. T. Le, S. Heo, Y. S. Chung, and S. Mansoor, "The path to smart farming: Innovations and opportunities in precision agriculture," 2023.

[3] E. K. Tan, Y. W. Chong, M. Niswar, B. Y. Ooi, and A. Basuki, "An iot platform for urban farming," in *Proceedings - 2020 International Seminar on Intelligent Technology and Its Application: Humanification of Reliable Intelligent Systems, ISITIA 2020*, 2020.

[4] Instituto Superior de Engenharia de Coimbra, "Instituto superior de engenharia de coimbra," 2024. [Online]. Available: https://www.isec.pt

[5] C. Coolectiva. (2023) Quero ser considerado um agricultor, mas um agricultor do futuro. [Online]. Available: https://coimbracoolectiva.pt/historias/quero-ser-considerado-um-agricultor-mas-um-agricultor-do-futuro/?doing_wp_cron=1733481892.3973760604858398437500

[6] M. Kulkarni, M. Ramamoorthy, G. Ramsankar, V. P., V. Bhagyalakshmi, and C. Sathish, "Sustainable crop monitoring and management for enhanced agricultural productivity through iot, ai&ml: Case studies and innovations," pp. 182–199, 2024.

[7] M. Pitesky, A. Gunasekara, C. Cook, and F. Mitloehner, "Adaptation of agricultural and food systems to a changing climate and increasing urbanization," 2014.

[8] B. Ahmed, H. Shabbir, S. R. Naqvi, and L. Peng, "Smart agriculture: Current state, opportunities and challenges," *IEEE Access*, 2024.

[9] A. Aborujilah, A. Alashbi, I. Shayea, A. H. Mohamed, A. Alhammadi, A. A. El-Saleh, and A. Ahad, "Iot integration in agriculture: Advantages, challenges, and future perspectives: Short survey," in *Proceedings - 10th International Conference on Wireless Networks and Mobile Communications, WINCOM 2023*, 2023.

[10] G. Guerrero-Ulloa, C. Rodríguez-Domínguez, and M. J. Hornos, "Agile methodologies applied to the development of internet of things (iot)-based systems: A review," 2023.

[11] A. Chakraborty, M. Islam, A. Dhar, and M. S. Hossain, "Iot based greenhouse environment monitoring and smart irrigation system for precision farming tech-

nology," in *2022 International Conference on Innovations in Science, Engineering and Technology (ICISET)*. IEEE, 2 2022, pp. 123–128.

[12] S. Amghar, S. Cherdal, and S. Mouline, "Which nosql database for iot applications?" in *2018 International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*. IEEE, 6 2018, pp. 131–137.

[13] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, 2017.

[14] O. Arshi and S. Mondal, "Advancements in sensors and actuators technologies for smart cities: a comprehensive review," *Smart Construction and Sustainable Cities*, vol. 1, 2023.

[15] F. Zeng, C. Pang, and H. Tang, "Sensors on internet of things systems for the sustainable development of smart cities: A systematic literature review," *Sensors*, vol. 24, no. 7, p. 2074, 2024.

[16] C. Gnauer, H. Pichler, C. Schmittner, M. Tauber, K. Christl, J. Knapitsch, and M. Parapatits, "A recommendation for suitable technologies for an indoor farming framework," *Elektrotechnik und Informationstechnik*, vol. 137, 2020.

[17] E. Navarro, N. Costa, and A. Pereira, "A systematic review of iot solutions for smart farming," 2020.

[18] H. M. Yasin, S. R. M. Zeebaree, and I. M. I. Zebari, "Arduino based automatic irrigation system: Monitoring and sms controlling," in *2019 4th Scientific International Conference Najaf (SICN)*. IEEE, 4 2019, pp. 109–114.

[19] V. Soniya, K. R. Shankar, S. Karishma, D. Vamsi, and R. V. H. Prasad, "Iot based smart way of watering plants and feeding pets," in *2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 3 2023, pp. 744–749.

[20] R. K. Kodali and A. Valdas, "Mqtt based monitoring system for urban farmers using esp32 and raspberry pi," in *2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT)*. IEEE, 8 2018, pp. 395–398.

[21] P. Manikandan, G. Ramesh, P. Lokesh, P. Raju, M. Prasad, and P. Madhu, "Iot based farm protection system from animals and humans theft using esp32 with camera module," in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*. IEEE, 4 2022, pp. 1861–1864.

[22] F. Lachhab, M. Bakhouya, R. Ouladsine, and M. Essaaidi, "Towards an intelligent approach for ventilation systems control using iot and big data technologies," *Procedia Computer Science*, vol. 130, pp. 926–931, 2018.

[23] N. Kumar, K. Kumar, and A. Kumar, "Application of internet of things in image processing," in *2022 IEEE Delhi Section Conference (DELCON)*. IEEE, 2 2022, pp. 1–5.

[24] C. Yoon, M. Huh, S.-G. Kang, J. Park, and C. Lee, "Implement smart farm with iot technology," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2 2018, pp. 749–752.

[25] K. Manaf, A. B. A. Rahman, Y. Setiawan, F. M. Kaffah, N. Lukman, and D. Pitoyo, "Designing a smart garden for automated plant watering using flutter and internet of things in the context of industry 4.0," in *Proceeding of 2023 17th International Conference on Telecommunication Systems, Services, and Applications, TSSA 2023*, 2023.

[26] N. Nakayama, H. Yamamoto, M. Arakawa, and I. Naka, "Sensor network system for observing environmental information and growth condition to support indoor agriculture," in *2022 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 1 2022, pp. 1–6.

[27] O. E. Amestica, P. E. Melin, C. R. Duran-Faundez, and G. R. Lagos, "An experimental comparison of arduino ide compatible platforms for digital control and data acquisition applications," in *IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies, CHILECON 2019*, 2019.

[28] J. Saini, M. Dutta, and G. Marques, "Sensors for indoor air quality monitoring and assessment through internet of things: a systematic review," 2021.

[29] J. Teixeira and R. C. dos Santos, "Exploring the applicability of low-cost capacitive and resistive water content sensors on compacted soils," *Geotechnical and Geological Engineering*, vol. 39, 2021.

[30] V. Rudakov, M. Timur, and A. Yedilkhan, "Comparison of time series databases," in *2023 17th International Conference on Electronics Computer and Computation (ICECCO)*. IEEE, 6 2023, pp. 1–4.

[31] P. Paethong, M. Sato, and M. Namiki, "Low-power distributed nosql database for iot middleware," in *Proceedings of the 2016 5th ICT International Student Project Conference, ICT-ISPC 2016*, 2016.

[32] K. Pushpanathan, M. Hanafi, S. Mashohor, and W. F. F. Ilahi, "Machine learning in medicinal plants recognition: a review," *Artificial Intelligence Review*, vol. 54, pp. 305–327, 1 2021.

[33] R. Alfred, J. H. Obit, C. P. Y. Chin, H. Haviluddin, and Y. Lim, "Towards paddy rice smart farming: A review on big data, machine learning, and rice production tasks," 2021.

[34] S. O. Araújo, R. S. Peres, J. C. Ramalho, F. Lidon, and J. Barata, "Machine learning applications in agriculture: Current trends, challenges, and future perspectives," 2023.

[35] Canva, "Canva: Design and visual content creation platform," https://www.canva.com, 2024.

[36] K. S. Ahmad, N. Ahmad, H. Tahir, and S. Khan, "Fuzzy-moscow: A fuzzy based moscow method for the prioritization of software requirements," in *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2017*, vol. 2018-January, 2017.

[37] M. M. Eyada, W. Saber, M. M. E. Genidy, and F. Amer, "Performance evaluation of iot data management using mongodb versus mysql databases in different cloud environments," *IEEE Access*, vol. 8, 2020.

[38] DB-Engines, "Db-engines: Knowledge base management systems," https://db-engines.com/en/, 2024.

[39] DB-Engines Ranking, "Db-engines ranking of database management systems," https://db-engines.com/en/ranking, 2024.

[40] P. Parial, "Python the game changer in the field of machine learning, data science and iot: A review," *International Journal for Research in Applied Science and Engineering Technology*, vol. 9, 2021.

[41] Google, "Flutter: Beautiful native apps in record time," https://flutter.dev.

[42] B. Mishra and A. Kertesz, "The use of mqtt in m2m and iot systems: A survey," *IEEE Access*, vol. 8, 2020.

[43] B. Mishra, B. Mishra, and A. Kertesz, "Stress-testing mqtt brokers: A comparative analysis of performance measurements," *Energies*, vol. 14, 2021.

[44] P. Serikul, N. Nakpong, and N. Nakjuatong, "Smart farm monitoring via the blynk iot platform : Case study: Humidity monitoring and data recording," in *International Conference on ICT and Knowledge Engineering*, vol. 2018-November, 2018.

[45] D. Hercog, T. Lerher, M. Truntič, and O. Težak, "Design and implementation of esp32-based iot devices," *Sensors*, vol. 23, 2023.

[46] "Git - distributed version control system." [Online]. Available: https://git-scm.com

[47] "Github - where the world builds software." [Online]. Available: https://github.com

[48] D. Inc., "Docker: Empowering app development for developers," https://www.docker.com.

[49] A. Manowska, A. Wycisk, A. Nowrot, and J. Pielot, "The use of the mqtt protocol in measurement, monitoring and control systems as part of the implementation of energy management systems," *Electronics (Switzerland)*, vol. 12, 2023.

[50] C. Nájera, V. M. Gallegos-Cedillo, M. Ros, and J. A. Pascual, "Role of spectrum-light on productivity, and plant quality over vertical farming systems: Bibliometric analysis," 2023.

[51] L. Yudina, E. Sukhova, E. Gromova, M. Mudrilov, Y. Zolin, A. Popova, V. Nerush, A. Pecherina, A. A. Grishin, A. A. Dorokhov, and V. Sukhov, "Effect of duration of led lighting on growth, photosynthesis and respiration in lettuce," *Plants*, vol. 12, 2023.

[52] P. I. Moraru, T. Rusu, and O. S. Mintas, "Trial protocol for evaluating platforms for growing microgreens in hydroponic conditions," 2022.

[53] M. Schrepp, A. Hinderks, and J. Thomaschewski, "Applying the user experience questionnaire (ueq) in different evaluation scenarios," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8517 LNCS, 2014.

[54] U. E. Questionnaire, "User experience questionnaire (ueq) online resource," https://www.ueq-online.org, 2024.

[55] B. Foundation, "Blender - a 3d modelling and rendering software," https://www.blender.org, 2024.

Appendix

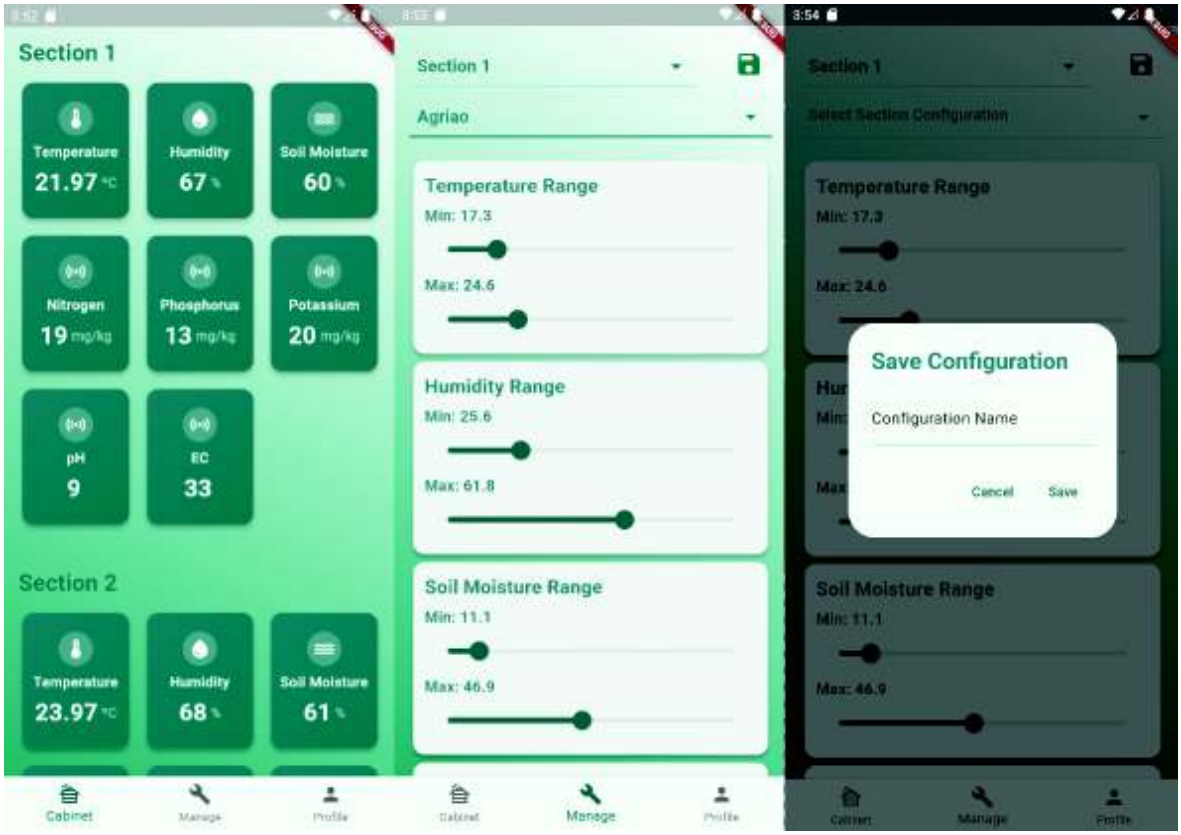
Appendix A - Responses of the User Experience Questionnaire

User Experience Questionnaire

User Experience with the mobile application for controlling and monitoring the smart agriculture cabinet, developed for the final project of the Master's degree in Computer Engineering

Application Demonstration

Mobile Application Developed



What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Obstructive	○	○	○	○	◉	○	○	Supportive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Complicated	○	○	○	○	○	○	◉	Easy

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Inefficient	○	○	○	○	◉	○	○	Efficient

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Confusing	○	○	○	○	○	◉	○	Clear

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Boring	○	○	○	○	◉	○	○	Exciting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Not interesting	○	○	○	○	○	◉	○	Interesting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Conventional	○	○	○	○	○	○	◉	Inventive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Usual	○	○	○	○	○	◉	○	Leading edge

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Obstructive	○	○	○	○	◉	○	○	Supportive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Complicated	○	○	○	○	○	○	◉	Easy

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Inefficient	○	○	○	○	○	◉	○	Efficient

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Confusing	○	○	○	○	○	○	◉	Clear

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Boring	○	○	○	○	◉	○	○	Exciting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Not interesting	○	○	○	○	○	◉	○	Interesting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Conventional	○	○	○	○	◉	○	○	Inventive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Usual	○	○	○	○	◉	○	○	Leading edge

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Obstructive	○	○	○	○	○	◉	○	Supportive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Complicated	○	○	○	○	◉	○	○	Easy

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Inefficient	○	○	○	○	◉	○	○	Efficient

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Confusing	○	○	○	○	◉	○	○	Clear

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Boring	○	○	○	○	○	○	◉	Exciting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Not interesting	○	○	○	○	◉	○	○	Interesting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Conventional	○	○	○	◉	○	○	○	Inventive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Usual	○	○	○	○	◉	○	○	Leading edge

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Obstructive	○	○	○	◉	○	○	○	Supportive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Complicated	○	○	○	○	○	◉	○	Easy

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Inefficient	○	○	○	◉	○	○	○	Efficient

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Confusing	○	○	○	○	○	◉	○	Clear

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Boring	○	○	○	○	◉	○	○	Exciting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Not interesting	○	○	○	○	○	◉	○	Interesting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Conventional	○	○	○	○	◉	○	○	Inventive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Usual	○	○	○	○	○	◉	○	Leading edge

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Obstructive	○	○	○	◉	○	○	○	Supportive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Complicated	○	○	○	○	○	◉	○	Easy

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Inefficient	○	○	○	○	◉	○	○	Efficient

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Confusing	○	○	○	○	○	◉	○	Clear

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Boring	○	○	○	○	○	◉	○	Exciting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Not interesting	○	○	○	○	○	◉	○	Interesting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Conventional	○	○	○	○	○	◉	○	Inventive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Usual	○	○	○	◉	○	○	○	Leading edge

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Obstructive	○	○	○	◉	○	○	○	Supportive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Complicated	○	○	○	○	◉	○	○	Easy

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Inefficient	○	○	○	◉	○	○	○	Efficient

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Confusing	○	○	○	○	○	○	◉	Clear

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Boring	○	○	○	◉	○	○	○	Exciting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Not interesting	○	○	○	○	○	○	◉	Interesting

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Conventional	○	○	○	◉	○	○	○	Inventive

What do you think about the mobile app? *

	1	2	3	4	5	6	7	
Usual	○	○	○	○	◉	○	○	Leading edge

Instituto Superior
de Engenharia
Politécnico de Coimbra