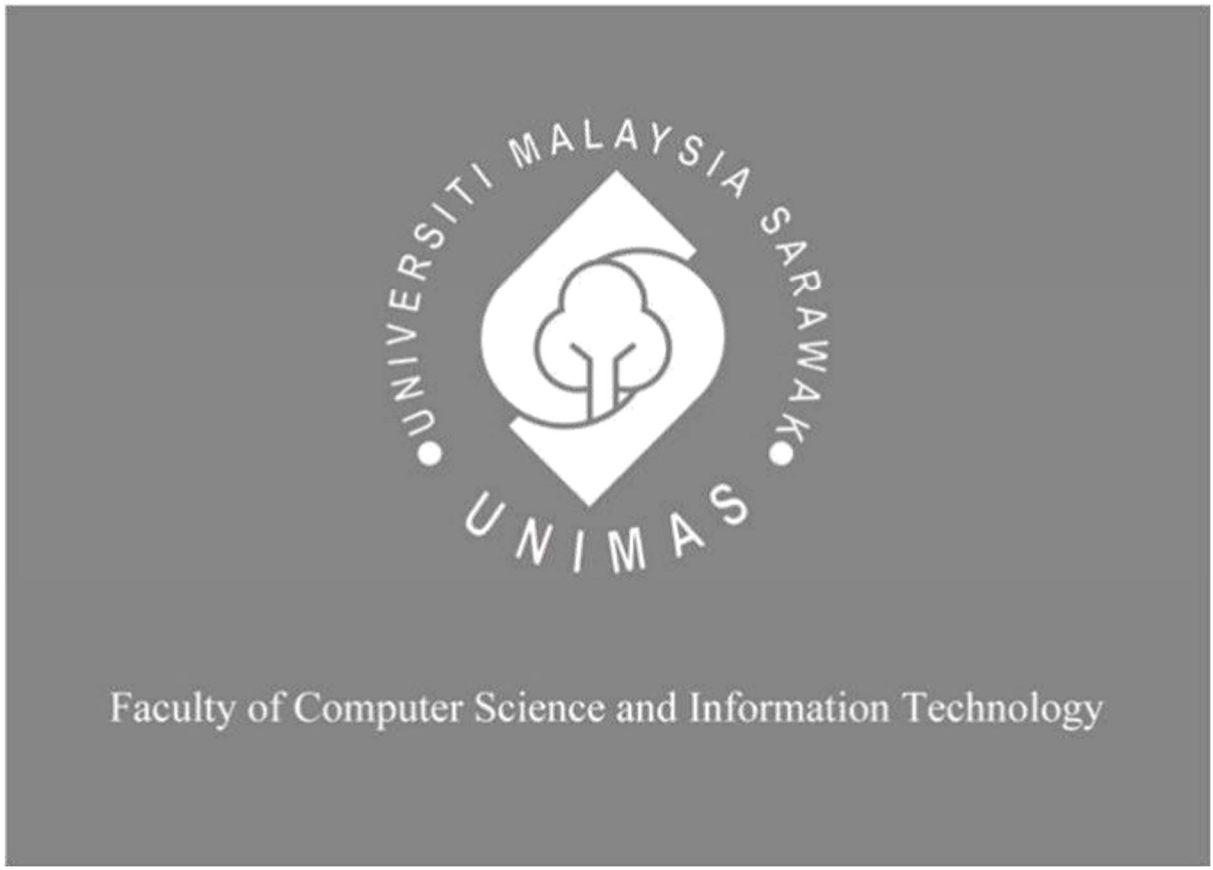


Greenhouse IoT Monitoring System Using MQTT

Ahmad Syafiq bin Salleh

Bachelor of Software Engineering with Honours

2024

Thesis Status Endorsement

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE **Greenhouse IOT Monitoring System Using MQTT**

ACADEMIC SESSION: 2024/2025

_____

(CAPITAL LETTERS)

hereby agree that this Thesis* shall be kept at the Centre for Academic Information Services, Universiti Malaysia Sarawak, subject to the following terms and conditions:

1. The Thesis is solely owned by Universiti Malaysia Sarawak
2. The Centre for Academic Information Services is given full rights to produce copies for educational purposes only
3. The Centre for Academic Information Services is given full rights to do digitization in order to develop local content database
4. The Centre for Academic Information Services is given full rights to produce copies of this Thesis as part of its exchange item program between Higher Learning Institutions [or for the purpose of interlibrary loan between HLI]
5. ** Please tick (√)

☐	CONFIDENTIAL	(Contains classified information bounded by the OFFICIAL SECRETS ACT 1972)
☐	RESTRICTED	(Contains restricted information as dictated by the body or organization where the research was conducted)
☑	UNRESTRICTED	

(AUTHOR'S SIGNATURE)

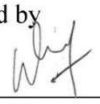
Ahmad Syafiq Bin Salleh

Permanent Address

Lot 531,
Lorong Sentosa Timur 3A,
Jalan Salim,96000 Sibu,Sarawak

Date: 24/07/2025

Validated by



(SUPERVISOR'S SIGNATURE)

Dr Wang Hui Hui
Senior Lecturer
Faculty of Computer Science and Information Technology
Universiti Malaysia Sarawak

Date: 24/7/2025

Note * Thesis refers to PhD, Master, and Bachelor Degree
** For Confidential or Restricted materials, please attach relevant documents from relevant organizations / authorities

Declaration

I hereby declare that this research together with all of its content is none other than that of my own work, with consideration of the exception of research-based information and relative materials that were adapted and extracted from other resources, which have evidently been quoted or stated respectively.

Signed,

.....
AHMAD SYAFIQ BIN SALLEH

Faculty of Computer Science and Information Technology

27 July 2025

Universiti Malaysia Sarawak (UNIMAS)

Acknowledgement

I would like to express my heartfelt gratitude to all those who supported and contributed to the successful completion of this Greenhouse IoT Monitoring System Using MQTT project.

Firstly, I would like to thank my supervisor, Dr Wang Hui Hui for her continuous guidance, valuable feedback, and unwavering support throughout this journey. Her advice and guidance were pivotal in shaping the direction and outcomes of this project.

I also extend my sincere appreciation to my fellow friends who provided both technical assistance and moral support, making this process more enriching and enjoyable.

A special mention goes to SAINS and Mr. Chin Sin Kian for providing opportunities during my internship, where I gained critical experience that was essential to the development of the system. Their mentorship contributed significantly to my understanding of IoT technologies, especially MQTT, and its application in real-world scenarios.

Finally, I would like to thank my family for their encouragement and support. Their belief in my abilities gave me the strength to persevere and successfully complete this project.

Thank you to everyone who has played a role, big or small, in making this project a reality.

Table of Contents

Thesis Status Endorsement ii
Declaration iii
Acknowledgement iv
List of Figure viii
List of Tables x
List of Abbreviations xi
ABSTRACT xiii
ABSTRAK xiv
Chapter 1: Introduction 1
1.1 Background 1
1.2 Problem Statement 2
1.3 Objective 3
1.4 Scope 3
1.5 Brief Methodology 4
1.6 Significance of Project 6
1.7 Project Schedule 8
1.8 Expected Outcome: 10
1.9 Project Outcome 11
Chapter 2: Background Study 12
2.0 Introduction 12
2.1 Review on similar existing system 13
2.2 Greenhouse Monitoring System for Chili Plant via IoT Cloud on Ubidots Platform 14
2.3 Greenhouse Monitoring System Using IOT 19
2.4 Greenhouse Monitoring And Control System With An Arduino System 21
2.5 Summary for past related projects 24
2.6 Advantages of MQTT Compared to Wi-Fi, GSM, and Other Protocols 25
2.7 Conclusion 27
Chapter 3: Requirement Analysis and Design 30
3.1 Introduction 30
3.2 Requirements Elicitation 32
3.2.1 User Requirements 32

3.2.2 Section A: Existing Challenges 34
3.2.3 Section B: Existing Challenges 36
3.2.3 Section C: Expectations and Features for the Proposed System 43
3.2.4 Section D: Impact and Feedback on Proposed System 47
3.3 System Design 50
3.3.1 Block Diagram 50
3.3.2 Greenhouse Monitoring System Use Case Diagram 53
3.3.3 Greenhouse Monitoring System Sequence Diagram 63
3.3.4 Greenhouse Monitoring System Class Diagram 65
3.3.5 Interface Design 68
3.4 System Requirements 3.4.1 Hardware 70
3.4.2 NodeMCU ESP32 71
3.4.3 Sensors 72
3.4.4 Control Equipment 75
3.4.5 Power System 79
3.4.6 Software 81
3.5 Conclusion 85
Chapter 4: Methodology and Implementation 86
4.1 Introduction 86
4.2 Tools and Technologies 87
4.3 Implementation Details 90
4.3.1 User Authentication and Authorization 90
4.3.2 Plant Management 93
4.3.3 Sensor Management 96
4.3.4 Dashboard and Data Visualization 99
4.3.5 Alerting System and Settings 100
4.3.6 Control of Output Devices 102
4.4 Database Implementation 103
4.5 Hardware Implementation 105
4.6 Chapter Summary 108
Chapter 5: Testing 109
5.1 Introduction 109
5.2 Functional Testing 109
5.2.1 Test Cases 110
5.3 Non-Functional Testing 119

5.4 Conclusion 121
Chapter 6: Requirement Analysis and Design 123
6.1 Introduction 123
6.2 Objective & Achievement 123
6.3 Limitation 125
6.4 Future Works 126
6.5 Conclusion 128
Reference 129
Appendix A: Manual UAT Checklist 131

List of Figure

Figure 1 Type Of Methodology....4
Figure 2 Block diagram of greenhouse monitoring system. 15
Figure 3 The flowchart 1 of greenhouse monitoring system (Che et al., 2023)....16
Figure 4 The flowchart 2 of greenhouse monitoring system (Che et al., 2023)....17
Figure 5 Block diagram of the “Greenhouse Monitoring System Using IOT” (Kumar et al., 2020) 19
Figure 6 Green House Monitoring and Control System Architecture (Yahaya et al., 2019) 22
Figure 7 Use Case Diagram for the Greenhouse System (Yahaya et al., 2019)....23
Figure 8 The first question for Section A....34
Figure 9 The Second question for Section A 35
Figure 10 The first question for Section B....36
Figure 11 The second question for Section B 37
Figure 12 The third question for Section B....38
Figure 13 The fourth question for Section B 40
Figure 14 The fifth question for Section B 41
Figure 15 The first question for Section C....43
Figure 16 The second question for Section C....44
Figure 17 The third question for Section C....45
Figure 18 The First question for Section D....47
Figure 19 The second question for Section D....48
Figure 20 The third question for Section D 49
Figure 21 Block diagram Greenhouse Monitoring using MQTT 50
Figure 22 Greenhouse Monitoring System Use Case Diagram 53
Figure 23 Greenhouse Monitoring System Sequence Diagram....63
Figure 24 Greenhouse Monitoring System Class Diagram....65
Figure 25 Greenhouse iot monitoring UI....68
Figure 26 NodeMCU ESP32 71
Figure 27 Soil Moisture Sensor 72
Figure 28 DHT22 Sensor....73
Figure 29 12V Cooling Fan for ESP 32 75
Figure 30 Micro Submersible Water Pump DC 3V-5V....76
Figure 31 5V 2 Channel Relay Module 10A 77
Figure 32 5V 2.5A / 3A Micro USB Power Supply Adapter for Raspberry Pi 3 80
Figure 33 Topic List....83
Figure 34 Json Data Handling....84
Figure 35 loginpage.php 90
Figure 36 add user.php....91
Figure 37 Manage User.php....92
Figure 38 add_plant.php 93
Figure 39 manage_plants.php 94
Figure 40 Edit plant.php 95
Figure 41 add_sensor.php 96
Figure 42 edit sensor.php....97
Figure 43 manage sensor.php....98
Figure 44 Dashboard.php....99

Figure 45 alert setup.php 100
Figure 46 Alert in dashboard.php 101
Figure 47 greenhouse_iot diagram design 103
Figure 48 Hardware Setup 105
Figure 49 Published to topics from arduino 106
Figure 50 subscribe topic 107

List of Tables

Table 1 Summary for past related project and project in development comparison 24
Table 2 Test Cases for User Authentication & Authorization Functionality 110
Table 3 Test Cases for Plant Management Functionality 111
Table 4 Test Cases for Sensor Management Functionality 112
Table 5 Test Cases for Sensor Data & MQTT Communication Functionality 113
Table 6 Test Cases for Dashboard & Real-time Visualization Functionality 114
Table 7 Test Cases for Alerting System Functionality 115
Table 8 Test Cases for Output Device Control Functionality 116
Table 9 Usability Testing Checklist for Web-based Dashboard 118
Table 10 Objective & Achievement 124

List of Abbreviations

IoT	Internet of Things
MQTT	Message Queuing Telemetry Transport
ESP32	Embedded Serial Peripheral 32-bit
Wi-Fi	Wireless Fidelity
GSM	Global System for Mobile Communications
ADC	Analog-to-Digital Converter
DHT	Digital Humidity and Temperature
LDR	Light Dependent Resistor
RH	Relative Humidity
VDC	Volts Direct Current
GPIO	General Purpose Input/Output
LCD	Liquid Crystal Display
DC	Direct Current
PSU	Power Supply Unit
JSON	JavaScript Object Notation

API	Application Programming Interface
PCB	Printed Circuit Board
LED	Light Emitting Diode
USB	Universal Serial Bus
ADC	Analog-to-Digital Converter
GUI	Graphical User Interface

ABSTRACT

The agricultural industry has increasingly embraced technological advancements to enhance productivity and efficiency. The Greenhouse IoT Monitoring System Using MQTT addresses key challenges in greenhouse management by utilizing the lightweight MQTT protocol to facilitate real-time monitoring of critical environmental parameters such as temperature, humidity, and soil moisture. Maintaining these conditions at optimal levels is essential for ensuring healthy plant growth and resource efficiency. Traditional greenhouse monitoring systems often rely on manual processes or outdated automation, which lack the precision and responsiveness required to adapt to rapidly changing conditions. These limitations result in resource inefficiencies and compromised crop quality. By integrating IoT sensors and leveraging MQTT for low-latency communication, this system provides reliable, continuous data transmission and supports proactive, data-driven decision-making. The project implements a user-friendly, web-based dashboard powered by Python, enabling greenhouse operators to visualize real-time data, receive automated alerts, and optimize environmental control. The system's scalability and adaptability to greenhouses of varying sizes make it a versatile solution for modern agriculture. This innovative approach minimizes operational costs, enhances resource management, and boosts agricultural productivity, offering a significant leap forward in sustainable farming practices.

ABSTRAK

Industri pertanian semakin memanfaatkan kemajuan teknologi untuk meningkatkan produktiviti dan kecekapan. Sistem Pemantauan Rumah Hijau IoT Menggunakan MQTT menangani cabaran utama dalam pengurusan rumah hijau dengan menggunakan protokol MQTT, yang ringan dan efisien, untuk memantau secara masa nyata parameter persekitaran kritikal seperti suhu, kelembapan, dan kelembapan tanah. Penyelenggaraan parameter ini pada tahap optimum adalah penting untuk memastikan pertumbuhan tanaman yang sihat dan penggunaan sumber yang cekap. Sistem pemantauan rumah hijau tradisional sering bergantung kepada proses manual atau automasi lama yang tidak mempunyai ketepatan dan responsiviti yang diperlukan untuk menyesuaikan diri dengan keadaan yang berubah-ubah. Kekurangan ini menyebabkan pembaziran sumber dan kualiti tanaman yang terjejas. Dengan mengintegrasikan sensor IoT dan menggunakan MQTT untuk komunikasi berlatensi rendah, sistem ini menyediakan penghantaran data yang berterusan dan boleh dipercayai, serta menyokong pengambilan keputusan secara proaktif berdasarkan data. Projek ini melaksanakan papan pemuka berasaskan web yang mesra pengguna, dikuasakan oleh Python, membolehkan pengendali rumah hijau memvisualisasikan data masa nyata, menerima amaran automatik, dan mengoptimumkan kawalan persekitaran. Keupayaan sistem yang berskala dan boleh disesuaikan untuk rumah hijau dengan pelbagai saiz menjadikannya penyelesaian serba boleh untuk pertanian moden. Pendekatan inovatif ini meminimumkan kos operasi, meningkatkan pengurusan sumber, dan meningkatkan produktiviti pertanian, memberikan kemajuan besar ke arah amalan pertanian yang lestari.

Chapter 1: Introduction

1.1 Background

The agricultural sector increasingly benefits from advancements in information technology, driving efforts toward greater efficiency and productivity (Baccay et al., 2021). The greenhouse IoT monitoring system using MQTT utilises the MQTT protocol to enable a reliable, real-time solution for monitoring essential greenhouse conditions, including temperature, humidity, and soil moisture. Maintaining these parameters at optimal levels is critical for promoting healthy plant growth and efficient resource usage. However, traditional monitoring methods often rely on manual processes, which can lead to delays and suboptimal growing conditions.

MQTT is a lightweight and efficient messaging protocol tailored for real-time data transfer in IoT applications, facilitating quick, secure, and low-latency communication between sensors and a central server. In this system, IoT sensors continually gather data on crucial environmental factors, while MQTT enables rapid data transmission across low-bandwidth networks. The data is then analyses with Python and displayed on a user-friendly, web-based dashboard, providing greenhouse operators with a real-time picture of environmental conditions.

By leveraging the MQTT protocol, this system minimizes communication overhead and enhances scalability, making it adaptable to greenhouses of various sizes and configurations. This seamless data flow empowers operators to make timely, data-driven decisions, optimizing conditions for plant growth while promoting resource efficiency.

1.2 Problem Statement

Greenhouses play a critical role in agriculture by providing controlled growing environments that optimize plant health and yield. Technology like AI and IoT has been employed in farming for some time now, along with other forms of cutting-edge computer science. In recent years, there has been a shift towards thinking about how to put this new technology to use (AlZubi & Galyna, 2023). However, managing these environments is complex, as optimal conditions for plant growth such as temperature, humidity, and soil moisture are subject to frequent fluctuations due to factors like weather, equipment efficiency, and crop demands. Conventional greenhouse monitoring systems often rely on periodic manual checks or outdated automated solutions that lack real-time capabilities. These methods fall short of delivering the precision and speed needed to respond effectively to changing conditions, leading to water waste, energy inefficiencies, and ultimately, reduced crop quality.

The absence of real-time data in such systems limits the ability to make timely adjustments, hindering resource efficiency and compromising plant health. Given these limitations, an IoT-based solution that provides continuous monitoring and instant data transmission could significantly improve greenhouse operations. This project introduces a real-time greenhouse monitoring system that utilizes the MQTT protocol for rapid, reliable communication between IoT sensors and a central server. By offering a scalable and user-friendly platform, this system empowers greenhouse operators to make proactive, data-driven decisions, optimizing environmental conditions and enhancing resource use across diverse greenhouse settings.

1.3 Objective

The objectives of this project are designed to tackle the current challenges in greenhouse management and propose innovative solutions using IoT technology:

- To design and develop a comprehensive greenhouse monitoring system using Internet of Things (IoT) technologies.
- To implement the MQTT (Message Queuing Telemetry Transport) protocol as the primary communication method between IoT sensors and a centralized server.
- To assess and optimize the usability of a web-based dashboard designed for real-time data visualization.

1.4 Scope

This project aims to design and implement an IoT-based monitoring system for greenhouses, tracking key environmental factors such as temperature, humidity, and soil moisture. Real-time data will be accessible through a web-based dashboard, allowing users to monitor conditions remotely and respond promptly to fluctuations that may affect plant health. The system is designed to be scalable and adaptable to different greenhouse sizes and configurations, providing flexibility for various setups and facilitating future expansion.

1.5 Brief Methodology

This project follows the Waterfall methodology, a linear, sequential approach where each phase must be completed before moving to the next. The project workflow is structured into distinct stages, each building upon the previous, ensuring comprehensive planning, implementation, and validation of each component before final deployment. The phases are as follows:

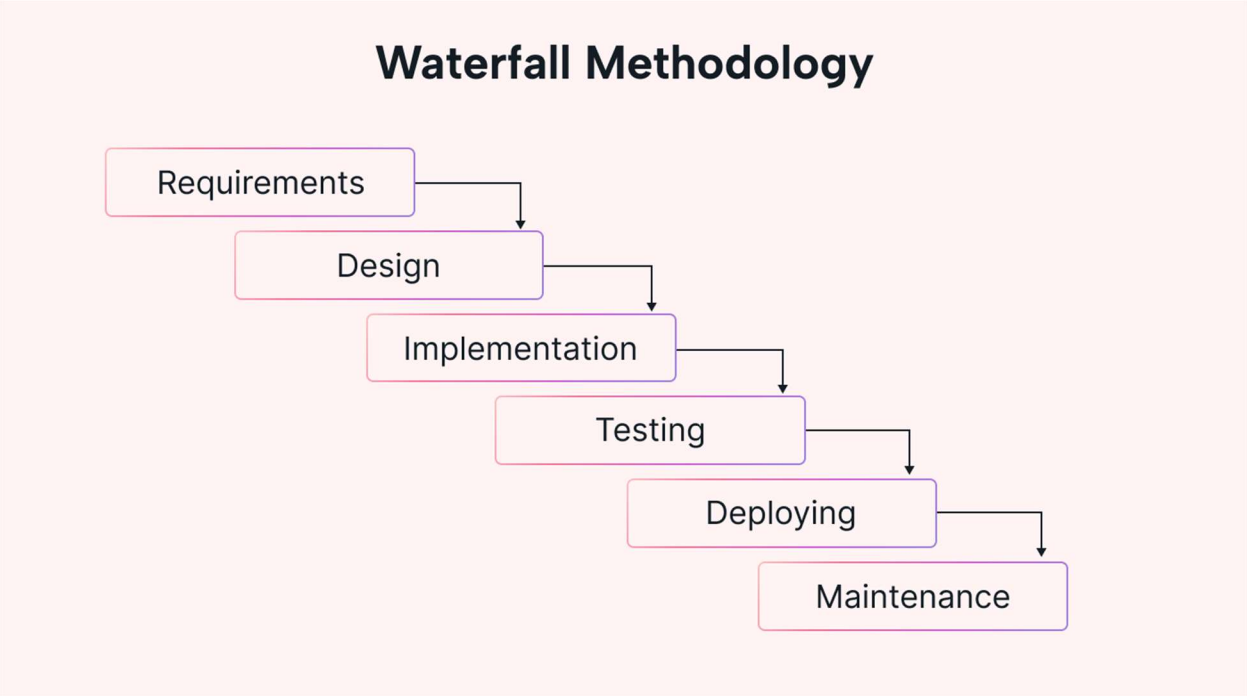


Figure 1 Type Of Methodology

1. **System Design and Planning:**

- Set goals for real-time monitoring of temperature, humidity, and soil moisture.
- Identify hardware, network, and software requirements (e.g., MQTT, Python).
- Design scalable architecture for sensor placement and MQTT communication.

2. **IoT Sensor Setup and Integration:**

- Select, calibrate, and install sensors for accurate environmental data collection.
- Develop scripts for continuous data gathering and transmission via MQTT.

3. **MQTT Protocol Implementation:**

- Configure MQTT broker for secure, low-latency data exchange.
- Test transmission reliability and set error-handling protocols.

4. **Data Processing and Visualization:**

- Process data with Python and display on a web-based dashboard.
- Include alerts for abnormal readings, supporting timely interventions.

5. **Testing and Validation:**

- Conduct system testing on data accuracy, response time, and stability.
- Collect user feedback to assess dashboard usability and alert functions.

6. **Evaluation and Optimization:**

- Analyze data trends, optimize components, and ensure scalability.

7. **Documentation and Reporting:**

- Document processes, configurations, and compile a final report with insights.

1.6 Significance of Project

The Greenhouse IoT Monitoring System Using MQTT holds significant value both in the context of modern agricultural practices and the broader technological landscape. By integrating IoT sensors and advanced communication protocols like MQTT, this project provides several benefits and addresses key challenges faced in greenhouse management. Below are the main reasons why this project is significant:

1. Optimizing Resource Management

Real-time monitoring of temperature, humidity, and soil moisture supports data-driven decisions, leading to efficient use of water, energy, and nutrients. This reduces waste and promotes sustainable greenhouse operations, which is essential given increasing global resource scarcity.

2. Enhancing Agricultural Productivity

Precise control of environmental conditions improves plant health and yield by preventing issues like improper watering or temperature extremes. This helps sustain agricultural productivity and supports the growing demand for food.

3. Reducing Operational Costs

Automating environmental monitoring reduces manual labor, minimizes human error, and saves resources, ultimately lowering operational costs and boosting efficiency. The system also helps optimize staffing and reduce reliance on physical inspections.

4. Improving Decision-Making with Real-Time Data

Real-time access to environmental data offers significant advantages for greenhouse operators (Prakash et al., 2023). Automated alerts allow timely corrective actions, while data analysis supports better predictive maintenance and long-term planning.

1.7 Project Schedule

No	Project Activities (FYP1)	Week													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	**Project Proposal**														
1.1	Project title & supervisor														
1.2	Draft brief project proposal														
2	**Background Study**														
2.1	Gather project requirements														
2.2	Determine project methodology														
2.3	Define system scope and objectives														
2.4	Conduct literature review														
3	**Requirement Analysis and Design**														
3.1	Identify system requirements														
3.2	Design system architecture														
3.3	Develop preliminary design layout														
3.4	Review and refine design														

No	Project Activities (FYP2)	Week													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	**System Design and Planning**														
1.1	Define system goals for monitoring														
1.2	Finalize hardware/software requirements														
2	**IoT Sensor Setup and Integration**														
2.1	Develop scripts for data collection														
2.2	Select, calibrate, and install sensors														
3	**MQTT Protocol Implementation**														
3.1	Set up MQTT broker for secure data exchange														
4	**Data Processing and Visualization**														
4.1	develop dashboard for real-time monitoring														
5	**Testing and Validation**														
5.1	Test data accuracy and usability														
5.2	Gather user feedback for improvements														
6	**Evaluation and Optimization**														
6.1	analyze data trends														
6.2	test scalability														
7	**Documentation and Reporting**														
7.1	Document system setup														
7.2	compile final report with findings and recommendations														

1.8 Expected Outcome:

- **Real-Time Environmental Monitoring System:**

  A robust, IoT-based greenhouse monitoring system that continuously provides real-time data on temperature, humidity, and soil moisture, enabling timely responses to environmental changes.

- **Interactive Web Dashboard for Remote Access:**

  A user-friendly, web-based dashboard that displays live environmental data, accessible from any device for convenient remote monitoring and quick decision-making.

- **Scalable and Adaptable System Design:**

  A modular system that can be scaled to accommodate various greenhouse sizes and customized to meet specific operational requirements, providing a flexible solution for diverse agricultural applications.

- **Increased Sustainability through Reduced Resource Consumption:**

  Reduced water, energy, and material waste through targeted monitoring and control, contributing to environmentally sustainable greenhouse practices and lower operational costs.

1.9 Project Outcome

By the end of this project, the objectives are expected to be fully achieved. The main outcome is the development of an IoT-based monitoring system for greenhouses. This system uses sensors to monitor temperature, humidity, and soil moisture in real time. The sensors will be accurately configured to collect reliable data and transmit it seamlessly using the MQTT protocol.

A key feature of the project is the creation of a web-based dashboard. This dashboard will provide real-time visualization of greenhouse conditions and display historical data trends. Users will also receive alerts for abnormal readings, enabling them to take timely action. The dashboard will be accessible on any device with internet connectivity, allowing remote monitoring and control.

Additionally, an MQTT broker will be configured to ensure secure and reliable communication between the sensors and the central server. The system will be designed for low-latency data transmission and will remain stable even under varying network conditions. The scalability of the system will allow it to support larger greenhouses or additional sensors in the future, making it adaptable to different setups and expansions.

Finally, this project aims to optimize resource usage in greenhouse operations. By enabling data-driven decisions, the system will help reduce water, energy, and nutrient waste, leading to lower operational costs and improved sustainability. These outcomes will provide an efficient and modern solution for greenhouse management, enhancing productivity and addressing challenges in agriculture.

Chapter 2: Background Study

2.0 Introduction

In this chapter, the literature review focuses on examining existing research related to IoT-based greenhouse monitoring systems. The purpose is to organize and analyze data from previous studies to develop a clear understanding of the subject. By reviewing these studies, relevant theories, strategies, and outcomes are identified, providing valuable insights for refining the research question and methodology for this project.

The review begins with an exploration of key studies on IoT-based greenhouse systems, highlighting how technologies such as sensors, microcontrollers, and communication protocols like MQTT have been utilized to monitor and control greenhouse environments. It also compares the approaches and outcomes of different studies, identifying their advantages and limitations to determine areas for improvement.

This literature review serves as the foundation for the research by critiquing previous works and demonstrating a critical understanding of the field. It establishes the context and relevance of the current study, justifying the need for further research and contributing to the discipline by addressing gaps in existing knowledge.

By analyzing and comparing prior systems, this chapter aims to provide a clear direction for the project, ensuring that the proposed IoT-based greenhouse monitoring system using MQTT builds on proven methods while offering innovative solutions to improve efficiency and productivity.

2.1 Review on similar existing system

This chapter explores existing greenhouse monitoring systems that use IoT technology to improve agricultural practices. Three systems were chosen for review, each focusing on innovative ways to automate and optimize greenhouse management. These include the Greenhouse Monitoring System for Chili Plant via IoT Cloud on Ubidots Platform by Che et al. (2023), the Greenhouse Monitoring System Using IoT by Kumar et al. (2020), and the Greenhouse Monitoring and Control System with an Arduino System by Yahaya et al. (2019).

Each system employs various technologies, such as sensors, microcontrollers, and cloud platforms, to address the challenges of traditional manual monitoring. The upcoming sections will discuss these systems in detail, highlighting their designs, methods, and benefits, while also identifying their limitations. This analysis will provide a foundation for understanding advancements in IoT-based greenhouse systems and their potential for future development.

2.2 Greenhouse Monitoring System for Chili Plant via IoT Cloud on Ubidots Platform

Che et al. (2023) present the development of an Internet of Things (IoT)-based greenhouse monitoring system tailored for chili cultivation, leveraging the Ubidots platform to enhance efficiency and productivity. Traditional greenhouse management in Malaysia relies heavily on manual labor, making monitoring tedious and inefficient. The proposed system employs temperature, humidity, soil moisture, and light sensors to automate the monitoring process. Data collected is transmitted to the Ubidots Cloud via a NodeMCU ESP8266 microcontroller, allowing real-time monitoring through a user-friendly dashboard. Alerts are sent to farmers in case of abnormal conditions, facilitating timely interventions. The results indicate that the system effectively optimizes greenhouse management, ultimately aiming to increase crop yield and reduce manual labor.

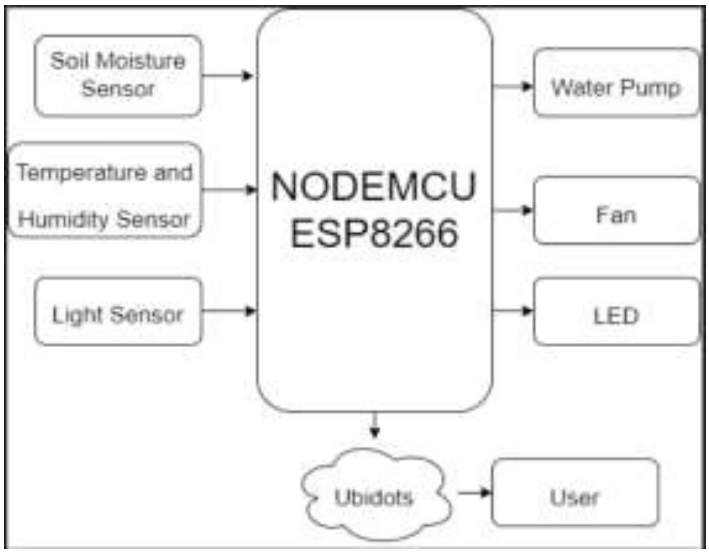


Figure 2 Block diagram of greenhouse monitoring system.

Figure 2.1 shows the block diagram of a greenhouse monitoring system, Monitoring section consist of all the sensor which is the DHT 11 sensor (humidity and temperature sensor), the soil sensor and the LDR sensor (light sensor). NodeMCU ESP8266 will be used as a Wi-Fi module to send the data to the cloud platform, Ubidots. Next, for the controlling section, it will be the actuator which can control the parameter of the crop, the actuators are the exhaust fan, the water pump, the LED light.

The system operations begin with the initialization of NodeMCU ESP8266, and all the sensor (LDR sensor, DHT 11 sensor, Soil moisture sensor). Then, all the sensors will take the reading value of the current parameter from the chili plant and send the data to the microcontroller. Next, the data will be sent to Ubidots cloud, to display the current parameters of the chili plant on the Ubidots dashboards. Besides, the system can control the actuators in order to maintain the greenhouse condition. Figure 3 and 4 shows the flowcharts of greenhouse monitoring system for Chilli Plant using IoT Cloud.

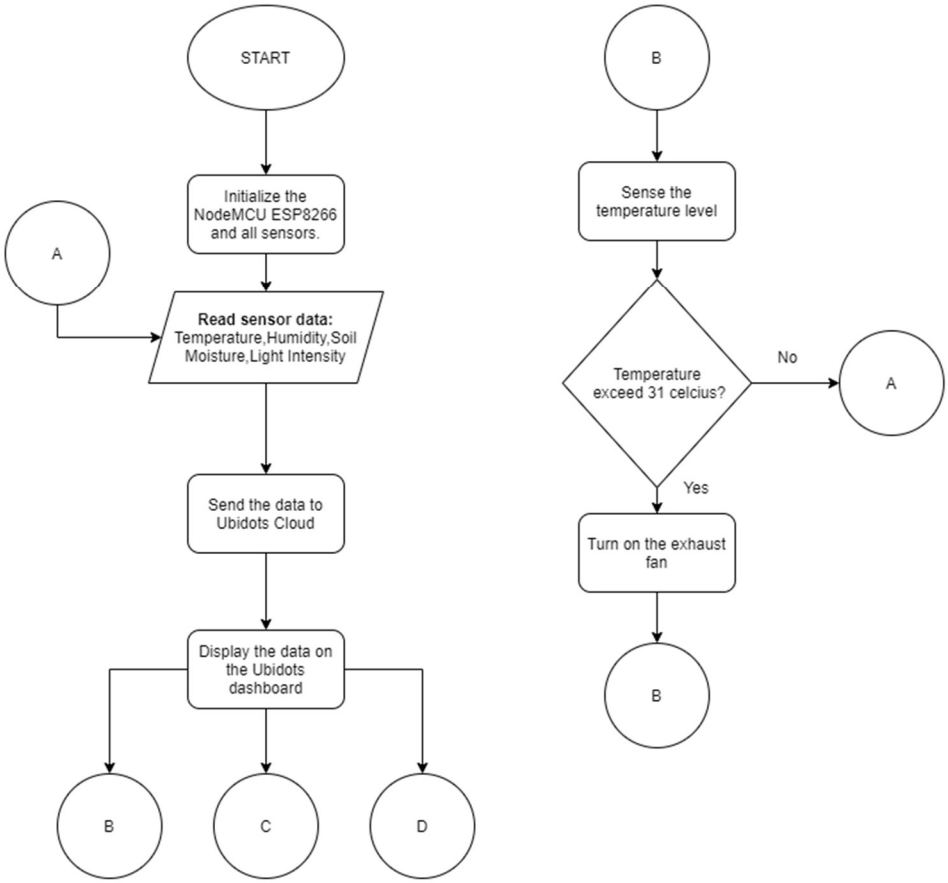


Figure 3 The flowchart 1 of greenhouse monitoring system (Che et al., 2023).

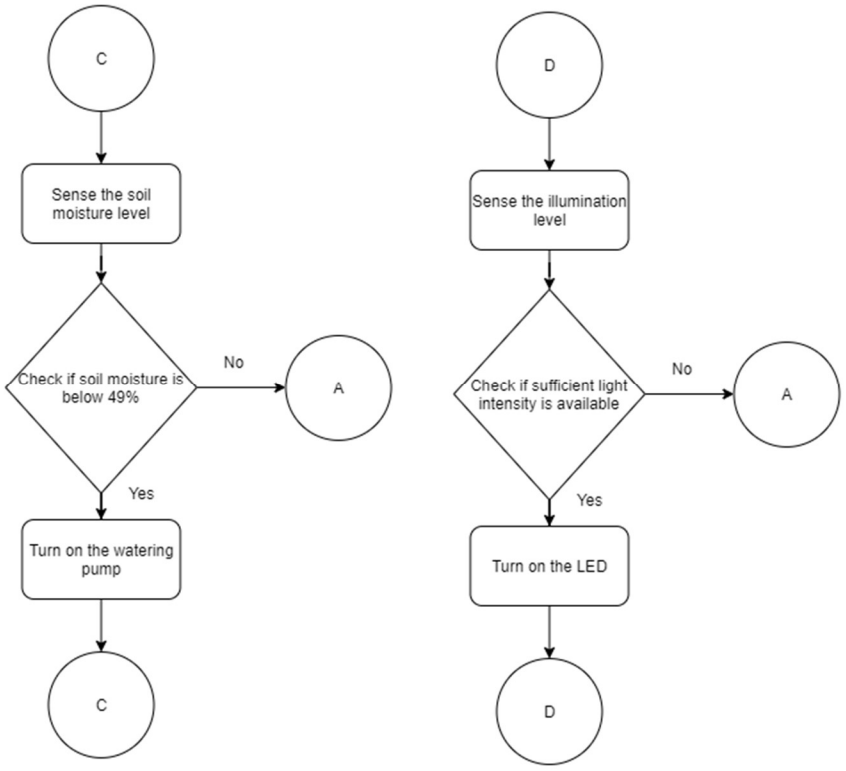


Figure 4 The flowchart 2 of greenhouse monitoring system (Che et al., 2023)

The greenhouse monitoring system integrates various components into a functional hardware setup without the need for additional schematic diagrams. The system includes a NodeMCU ESP8266 microcontroller, temperature sensor, soil moisture sensor, LDR sensor, pH sensor, and relay board. The relay operates as a switch to control the motor, while a diode ensures the current flows in one direction. Due to the motor's high voltage requirements, an external power source, such as a battery, is employed since the microcontroller's power output is limited to 5V. The NodeMCU ESP8266 was chosen for its dual functionality as a microcontroller and Wi-Fi module, with adequate pins to accommodate all connected components, eliminating the need for an additional Arduino microcontroller.

The IoT-based greenhouse monitoring system proposed by Che et al. (2023) demonstrates significant advancements in agricultural technology. By automating data collection and leveraging cloud-based platforms like Ubidots, the system enhances efficiency and productivity in greenhouse management. Its ability to provide real-time monitoring and timely alerts makes it an invaluable tool for chili cultivation in Malaysia, addressing challenges associated with traditional manual methods and paving the way for smarter agricultural practices.

2.3 Greenhouse Monitoring System Using IOT

The “Greenhouse Monitoring System Using IoT,” as described by Kumar et al. (2020), aims to simplify agricultural management by automating the maintenance of optimal conditions for crops. This system utilizes various sensors, including soil moisture, temperature, and light intensity sensors, to monitor environmental conditions and adjust them as needed without human intervention. For instance, if the temperature drops due to sudden rain, the system activates a heat lamp, and if soil moisture decreases, it turns on a water pump. Data collected from these sensors is sent to the farmer’s mobile device via a GSM module, reducing the need for manual labor and allowing for efficient large-scale farming. The system employs an Arduino microcontroller for the automation process, ensuring precise control and real-time updates. (Kumar et al., 2020)

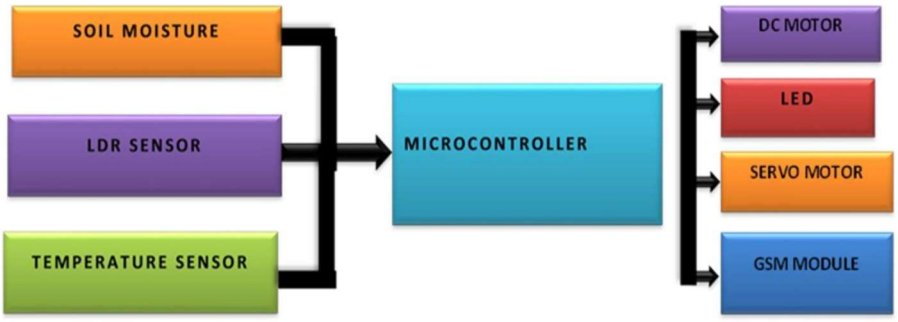


Figure 5 Block diagram of the “Greenhouse Monitoring System Using IOT” (Kumar et al., 2020)

The greenhouse monitoring system integrates a range of input sensors, a microcontroller, and output components to automate and optimize environmental management. Key input sensors include the soil moisture sensor, which monitors the moisture level of the soil, the Light-Dependent Resistor (LDR) sensor, which detects light intensity to evaluate lighting conditions, and the temperature sensor, which measures the ambient temperature within the greenhouse. These sensors continuously gather data to ensure the optimal growth environment for plants.

At the core of the system lies an Arduino microcontroller, which processes the data collected from the sensors. Based on predefined thresholds and conditions, the microcontroller determines the necessary actions to maintain ideal greenhouse conditions. This decision-making process ensures that environmental factors are adjusted dynamically without requiring manual intervention.

The system's output components enable the execution of these actions. A DC motor is used to activate devices such as water pumps for irrigation when the soil moisture level falls below the required threshold. An LED provides visual alerts or lighting as needed, serving as a quick and efficient way to signal system status or anomalies. Additionally, a servo motor adjusts mechanical components, such as opening vents for ventilation or controlling heat lamps to maintain optimal temperature conditions.

Finally, the system includes a GSM module to enhance remote monitoring capabilities. This module transmits real-time data and alerts directly to the farmer's mobile device, enabling them to track greenhouse conditions and respond promptly to any issues. Together, these components work seamlessly to automate greenhouse management, reduce manual labor, and improve overall agricultural efficiency.

2.4 Greenhouse Monitoring And Control System With An Arduino System

Yahaya et al. (2019) present the development of a smart greenhouse system that leverages Internet of Things (IoT) technology to optimize plant growth by maintaining favorable environmental conditions. The system integrates various sensors to monitor key parameters critical to greenhouse management, including temperature, air humidity, soil moisture, and soil pH. These parameters are vital for ensuring optimal plant health and productivity.

The system's sensors collect real-time environmental data, which is transmitted to a central server using the Message Queuing Telemetry Transport (MQTT) protocol. This protocol ensures efficient and reliable data transmission, allowing users to monitor and control the greenhouse conditions remotely through mobile devices. Such remote accessibility significantly reduces the need for constant human presence, streamlining greenhouse operations.

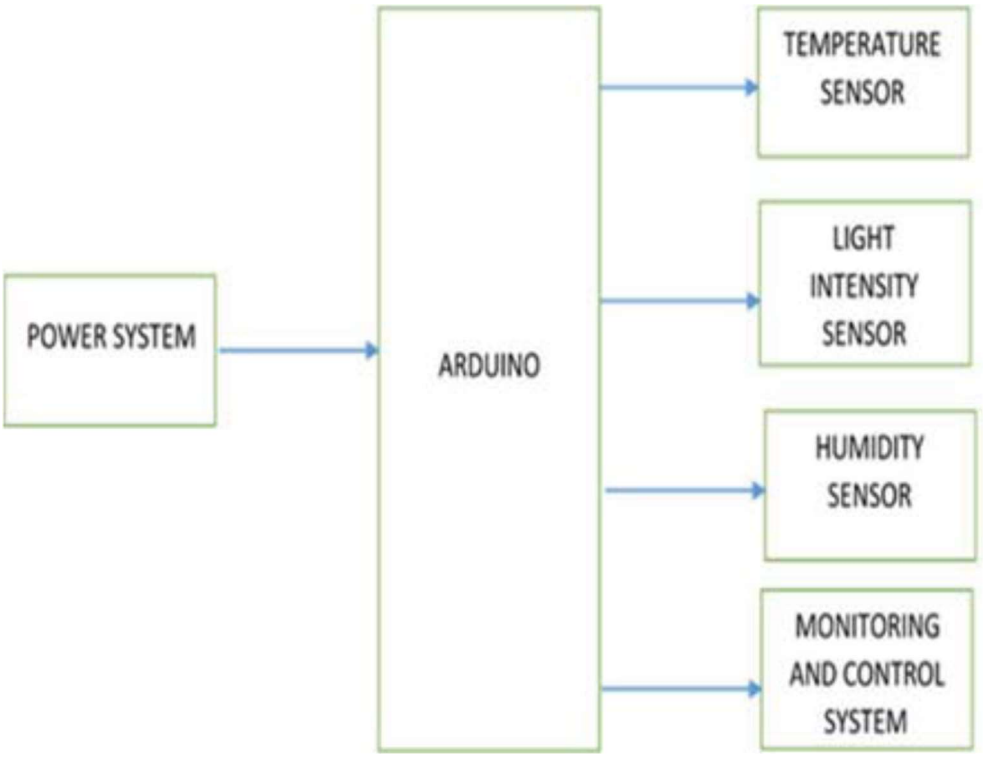


Figure 6 Green House Monitoring and Control System Architecture (Yahaya et al., 2019)

The system operates on a wired connection methodology, employing various components, such as sensors, an Arduino board, an LCD display, cooling systems, power banks, LEDs, and light-dependent resistors (LDRs). These components work collectively to monitor and regulate environmental factors critical to plant growth. The results yielded a fully functional system capable of maintaining a stable environment within the greenhouse.

The integration of IoT technology in the smart greenhouse system highlights a transformative advancement in agricultural practices. By automating environmental monitoring and control, the system minimizes human intervention, conserves resources, and promotes efficient farming. This study underscores the potential of IoT in modernizing agriculture, paving the way for sustainable and intelligent farming solutions.

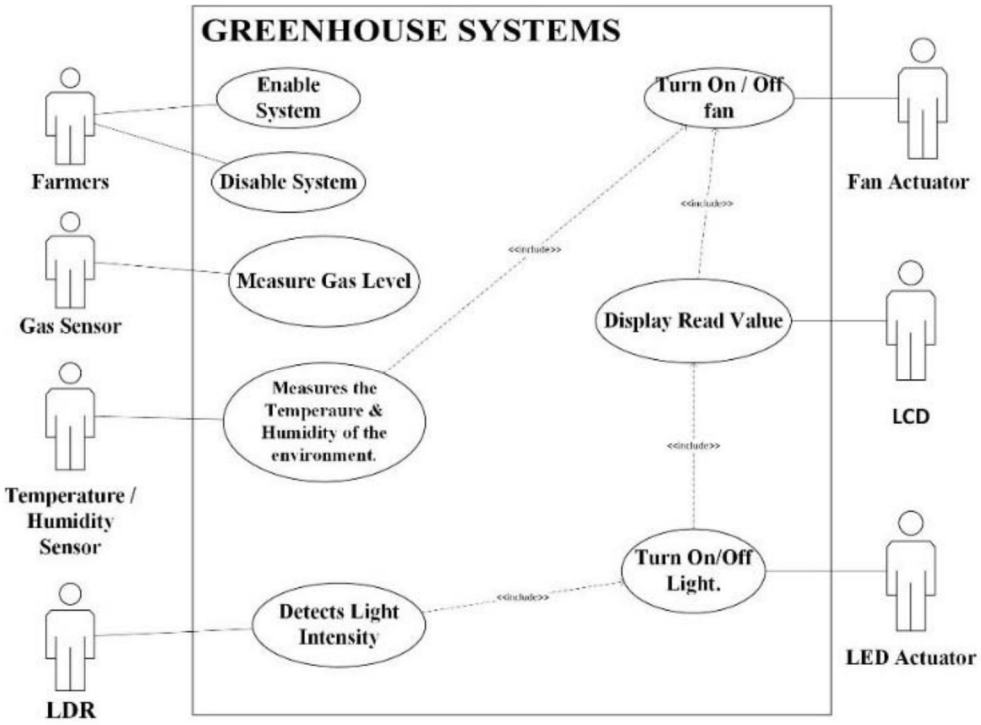


Figure 7 Use Case Diagram for the Greenhouse System (Yahaya et al., 2019)

The fully functional system successfully maintained a stable greenhouse environment, demonstrating its effectiveness in automating agricultural processes. The integration of IoT technology into greenhouse management significantly reduces manual intervention, improves resource efficiency, and enhances crop productivity.

This research showcases the transformative potential of IoT in agriculture, offering a sustainable and intelligent farming solution. By automating critical processes, the system provides a scalable model for modernizing agricultural practices, making it particularly beneficial for small-scale farmers seeking to optimize greenhouse operations year-round.

2.5 Summary for past related projects

Feature	Greenhouse Monitoring System for Chili Plant via IoT Cloud on Ubidots Platform	Greenhouse Monitoring System Using IoT	Greenhouse Monitoring and Control System with an Arduino System	Greenhouse IOT Monitoring System Using MQTT
Sensors Used	Temperature, humidity, soil moisture, light	Temperature, soil moisture, light intensity	Temperature, humidity, soil moisture, pH	Temperature, humidity, soil moisture
Microcontroller	NodeMCU ESP8266	Arduino	Arduino	NodeMCU ESP32
Communication Protocol	Wi-Fi (via Ubidots Cloud)	GSM	MQTT	MQTT
Remote Monitoring Capability	Yes, via Ubidots dashboard	Yes, via GSM to mobile devices	Yes, through MQTT-enabled devices	Yes, through a website and MQTT
Automation	Yes, controls actuators for optimal conditions	Yes, automates environmental adjustments	Yes, automates control processes	Yes, automates environmental control
Control Mechanisms	Actuators for fans, water pumps, and LED lights	DC motor for water pump, servo motor for vents	Cooling systems, LEDs, and ventilation	Fans, water pumps, adjustable from website
Power Source	Battery and microcontroller supply	Battery for motor operation	Power bank for system operation	Battery and external power
Data Transmission Method	Cloud-based via Wi-Fi	GSM network	MQTT protocol	MQTT
Target Application	Chili plant greenhouse in Malaysia	General greenhouse crops	Broad greenhouse applications	General greenhouse
Web-Based Automation Configuration	Ubidots	Not available	Not available	Yes, thresholds adjustable via the website

Table 1 Summary for past related project and project in development comparison

2.6 Advantages of MQTT Compared to Wi-Fi, GSM, and Other Protocols

Advantages of MQTT Compared to Wi-Fi, GSM, and Other Protocols

MQTT (Message Queuing Telemetry Transport) is a lightweight communication protocol specifically designed for low-bandwidth, high-latency, or unreliable networks. It is widely used in IoT applications, including greenhouse monitoring systems, due to its efficiency, scalability, and real-time capabilities. When compared to other communication methods like Wi-Fi and GSM, MQTT offers several distinct advantages.

1. Bandwidth Efficiency

MQTT is highly efficient in terms of bandwidth usage. Unlike traditional Wi-Fi or GSM-based communication, which often involves complex protocols and higher data overhead, MQTT uses a publish-subscribe model that minimizes data transmission. This ensures that only necessary data is sent and received, reducing network congestion and making it ideal for IoT applications where data packets are small but frequent.

2. Power Consumption

One of the key benefits of MQTT is its low power consumption. Devices using MQTT can operate on battery power for extended periods, as they do not need to maintain constant communication like Wi-Fi or GSM-based systems. Wi-Fi connections typically require continuous data transmission, leading to higher energy consumption. Similarly, GSM modules consume significant power due to their need for frequent network handshakes and signal maintenance.

3. Scalability

MQTT is designed to handle thousands to millions of connected devices efficiently. Since it operates on a lightweight publish-subscribe mechanism, devices do not need to maintain direct communication with each other but rather communicate through a central broker. In contrast, Wi-Fi-based systems are often limited by router capacity and network congestion, while GSM-based communication can become costly and inefficient when scaling to a large number of devices.

4. Reliability and Quality of Service (QoS)

MQTT provides three levels of Quality of Service (QoS), allowing users to optimize data delivery based on network conditions:

- **QoS 0 (At most once):** The message is sent without confirmation, minimizing latency but with no guarantee of delivery.
- **QoS 1 (At least once):** The message is guaranteed to be delivered at least once, ensuring reliability.
- **QoS 2 (Exactly once):** Ensures that the message is received only once, which is useful for critical applications.

Wi-Fi and GSM do not inherently offer such granular QoS levels, making MQTT a more reliable choice for applications requiring precise data delivery.

5. Connectivity Over Unstable Networks

Unlike Wi-Fi, which requires a strong and continuous connection, MQTT is highly optimized for unreliable and high-latency networks. It can maintain communication even with intermittent connectivity, making it superior to GSM-based systems, which may drop connections in weak signal areas. MQTT's small packet size and keep-alive mechanism ensure devices remain connected without excessive data exchange.

6. Cost-Effectiveness

Using MQTT over an internet-based network (such as Ethernet or LTE) is often more cost-effective than GSM-based solutions, which require mobile data plans. Wi-Fi can also be costly in large-scale deployments due to infrastructure requirements, whereas MQTT efficiently operates on existing network setups with minimal resource consumption.

2.7 Conclusion

The literature review on the development of the Greenhouse IoT Monitoring System using MQTT serves as a comprehensive and systematic analysis of existing studies and innovations in the field

of IoT-based greenhouse management. This review aims to provide a detailed understanding of the theories, technologies, and methodologies employed in previous research, focusing on the application of IoT and MQTT in agricultural monitoring systems.

By examining the relevant literature, the review highlights the importance of environmental control factors such as temperature, humidity, light intensity, and soil moisture in optimizing plant growth within a greenhouse setting. The review also explores the integration of sensors, microcontrollers, and communication protocols like MQTT that are critical to real-time data collection, remote monitoring, and automated control.

The critique of prior studies not only reveals the strengths and limitations of existing systems but also identifies gaps and opportunities for improvement. This analysis provides valuable insights into the challenges faced by previous research, such as issues related to scalability, reliability, and system integration, while also highlighting successful approaches that could be adopted or refined for the current project.

Additionally, the literature review plays a key role in guiding the researcher in formulating a well-defined research question and selecting an appropriate methodology. It ensures that the proposed greenhouse monitoring system is informed by the most current and relevant advancements, contributing to its effectiveness and potential for real-world application.

Ultimately, the review demonstrates the researcher's ability to critically engage with the existing body of knowledge, contextualizing the new system within the broader landscape of IoT and agricultural technology. It ensures that the project builds upon proven strategies while addressing unmet needs, ultimately advancing the development of more efficient, sustainable, and intelligent greenhouse management solutions.

Chapter 3: Requirement Analysis and Design

3.1 Introduction

This chapter provides a comprehensive review of existing literature and research related to the development of an IoT-based greenhouse monitoring system using the MQTT protocol. The primary objective of this system is to enable real-time monitoring and automation of environmental conditions within a greenhouse to optimize plant growth and reduce manual labor. By leveraging IoT technology and the MQTT communication protocol, the system ensures efficient data transmission and enhanced decision-making capabilities.

The chapter explores the integration of various IoT components, including temperature, humidity, and soil moisture sensors, to gather real-time environmental data. These sensors work collaboratively to ensure that critical parameters for plant growth are continuously monitored. Data from these sensors is processed and transmitted via the MQTT protocol, known for its lightweight and efficient data exchange mechanism, to a centralized dashboard. This dashboard serves as the system's interface, providing users with a visual representation of real-time data, automated alerts, and control options for greenhouse devices such as fans and water pumps.

In addition to reviewing foundational concepts, this chapter examines related works that employ similar technologies in agricultural applications. It provides an in-depth analysis of systems utilizing different microcontrollers, communication protocols, and automation mechanisms to enhance greenhouse management. By comparing and contrasting these systems, the chapter highlights the advantages and limitations of existing solutions, setting the groundwork for the proposed methodology.

Furthermore, the chapter emphasizes the critical role of IoT in addressing challenges faced by traditional greenhouse management, such as inefficiency, high labor dependency, and inconsistent environmental conditions. The discussion also explores the potential of MQTT in facilitating seamless, secure, and scalable communication for IoT applications. Overall, this chapter provides the theoretical foundation and contextual understanding necessary for the design and implementation of the proposed greenhouse monitoring system.

3.2 Requirements Elicitation

The primary goal of requirement elicitation is to gather and understand the information and specifications necessary to successfully develop the Greenhouse IoT Monitoring System Using MQTT. The literature review in Chapter 2 provided valuable insights into existing greenhouse monitoring solutions and the essential features that should be incorporated, such as real-time environmental monitoring, automated control systems, and web-based interfaces.

Additionally, a survey was conducted using Google Forms to gather user requirements and feedback. The survey targeted researchers, agriculture professionals, and greenhouse operators to understand their challenges in greenhouse monitoring, their interest in IoT-based monitoring systems, and their preferences for specific features. The results of this survey are instrumental in shaping the design and functionality of the proposed solution, ensuring it effectively addresses user needs and enhances the efficiency of greenhouse operations.

3.2.1 User Requirements

Based on the analysis of survey results and the insights gathered from the literature review, two key user requirements have been identified to achieve the project's objectives:

- To design and develop an IoT-based monitoring system that can track and control greenhouse environmental conditions using MQTT protocol.
- To enhance the efficiency and accuracy of environmental monitoring through real-time data collection and automated control responses.

The survey targets a diverse group of agriculture professionals, greenhouse managers/operators, and research staff. This broad spectrum of respondents ensures that the gathered requirements reflect various perspectives and use cases. The target sample size 10 respondents, providing sufficient data for meaningful analysis while remaining manageable within the project's timeframe.

The survey is structured into five sections: A, B, C, and D. Section A gathers demographic information about the participants, such as their occupation and experience with greenhouse operations. Section B identifies awareness and perception of IoT in agriculture, including current challenges and existing technology usage. Section C evaluates expectations and desired features for the proposed system. Section D Impact and Feedback on Proposed System.

3.2.2 Section A: Existing Challenges

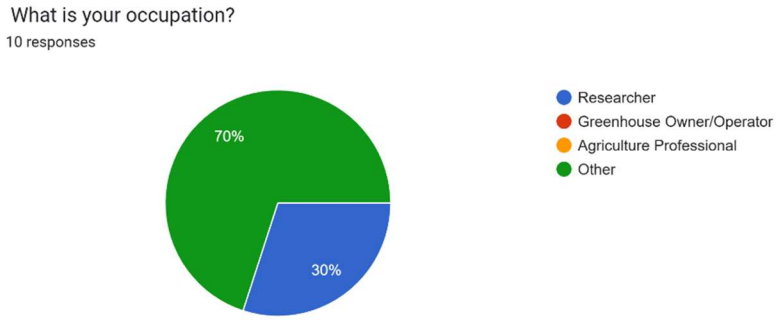


Figure 8 The first question for Section A

Occupation of the respondents

The survey data regarding participants' occupations reveals a diverse group, with the majority (70%) identifying as "Other," indicating occupations outside of the predefined categories. The remaining 30% are researchers, highlighting their active involvement in greenhouse-related studies or projects. Interestingly, no respondents identified as greenhouse owners/operators or agriculture professionals, suggesting a potential gap in representation from individuals directly engaged in day-to-day agricultural practices.

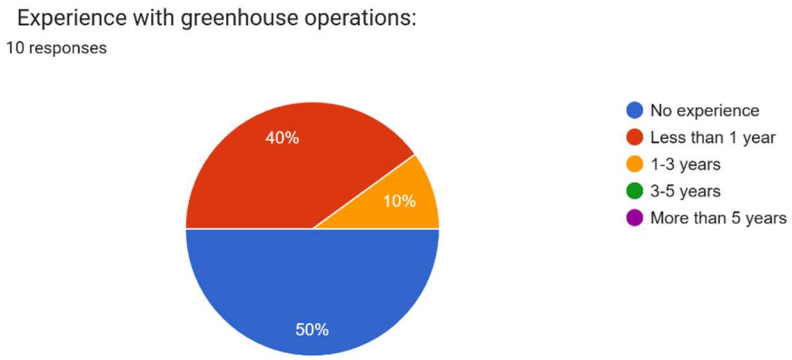


Figure 9 The Second question for Section A

Experience with greenhouse operation

The survey results on participants' experience with greenhouse operations reveal that a significant portion of respondents are either new to the field or have limited exposure. Notably, 50% reported having no prior experience, while 40% indicated less than one year of experience, highlighting a large proportion of beginners in the group. Only 10% of respondents reported moderate experience (1-3 years), and no participants had more than three years of experience in greenhouse operations. This distribution suggests a growing interest in the field among individuals with limited expertise, underscoring the need for training programs and resources to support these newcomers in adopting greenhouse technologies and practices effectively.

3.2.3 Section B: Existing Challenges

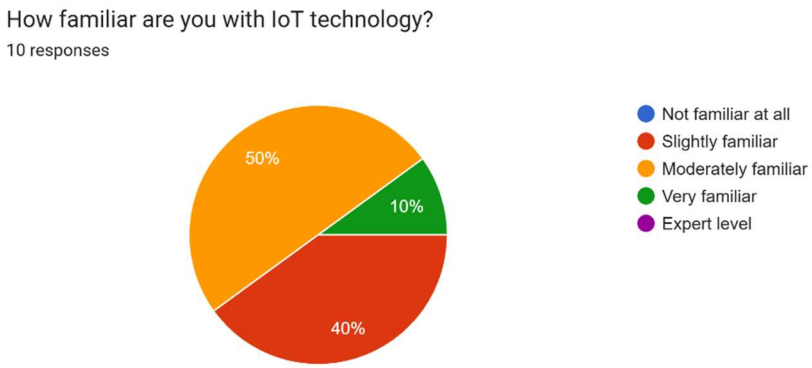


Figure 10 The first question for Section B

Technology Familiarity and Adoption Patterns

The survey results demonstrate varying levels of IoT technology familiarity among agricultural professionals. Half of the respondents (50%) reported being slightly familiar with IoT technology, while 40% indicated moderate familiarity. This finding aligns with research by AlZubi and Galyna (2023), who noted a gradual shift towards technological adoption in farming practices. The limited number of respondents (10%) claiming high familiarity suggests a significant opportunity for educational initiatives and technology training programs.

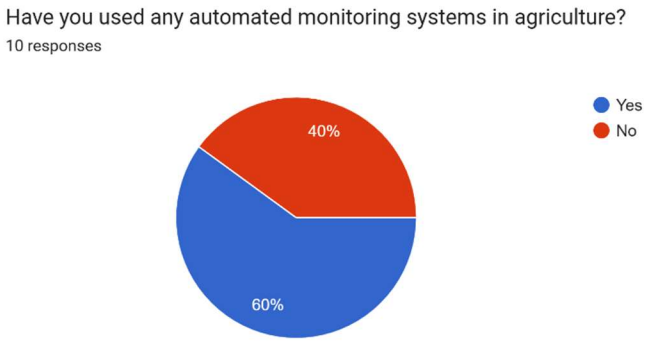


Figure 11 The second question for Section B

Analysis of IoT Awareness and Perception in Agricultural Technology

The survey analysis of Section B reveals significant insights into the current state of IoT technology adoption and awareness in agricultural settings, particularly in greenhouse operations. This section examines stakeholders' familiarity with IoT technology, their experience with automated monitoring systems, and the challenges they face in implementing these solutions.

Automated monitoring system adoption shows promising trends, with 60% of respondents reporting experience with such systems. As Prakash et al. (2023) emphasize, "Real-time access to environmental data offers significant advantages for greenhouse operators." This observation is reflected in the survey results, where temperature and humidity monitoring systems emerged as the most widely adopted technologies, used by 70% and 60% of respondents respectively.

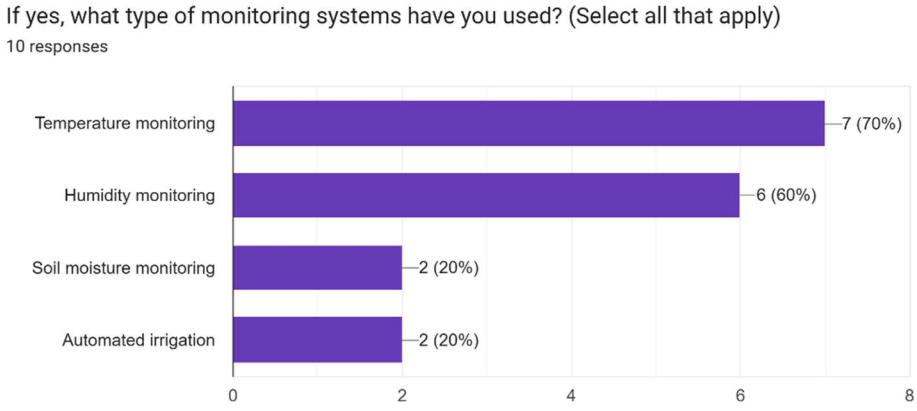


Figure 12 The third question for Section B

Critical Monitoring Parameters and Implementation

Environmental monitoring priorities demonstrate a clear focus on fundamental growth parameters. Temperature monitoring leads to adoption rates at 70%, followed closely by humidity monitoring at 60%. However, more advanced applications such as soil moisture monitoring and automated irrigation show lower adoption rates of 20% each. This pattern reflects observations by Baccay et al. (2021), who noted that while the agricultural sector increasingly benefits from information technology, adoption rates vary significantly across different applications.

The importance of real-time monitoring in greenhouse operations is widely acknowledged, with 80% of respondents rating it as either "extremely important" or "very important." This high valuation of real-time monitoring capabilities supports Che et al.'s (2023) findings regarding the critical role of automated monitoring in optimizing greenhouse management.

The survey results suggest a clear need for comprehensive IoT solutions that address current operational challenges while considering implementation barriers. As noted by Yahaya et al. (2019), successful integration of IoT technology in greenhouse operations requires careful consideration of both technical capabilities and user requirements. The relatively low adoption rates of advanced features like automated irrigation (20%) indicate potential opportunities for technology providers to develop more accessible and user-friendly solutions.

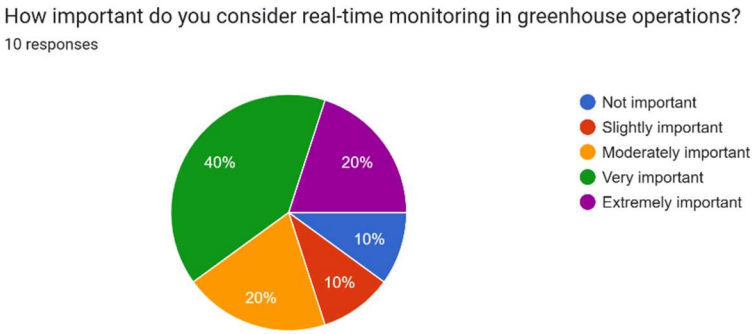


Figure 13 The fourth question for Section B

Analysis of Real-Time Monitoring Importance in Greenhouse Operations

The survey responses reveal varying perspectives on the importance of real-time monitoring in greenhouse operations, with 60% of participants rating it as either "very important" (40%) or "extremely important" (20%). This majority highlights the growing recognition of real-time monitoring as a critical tool for optimizing greenhouse management by enabling precise adjustments to environmental conditions, improving productivity, and reducing resource wastage. Meanwhile, 20% of respondents considered it “moderately important,” indicating an awareness of its benefits but perhaps not prioritizing it as essential, possibly due to reliance on other methods or technologies. The remaining 20% rated it as "not important" or "slightly important," potentially reflecting a lack of familiarity with IoT systems or a focus on simpler greenhouse requirements where real-time monitoring may not yet be necessary. These findings suggest that while real-time monitoring is widely regarded as valuable, barriers such as cost, and ease of implementation may hinder broader adoption.

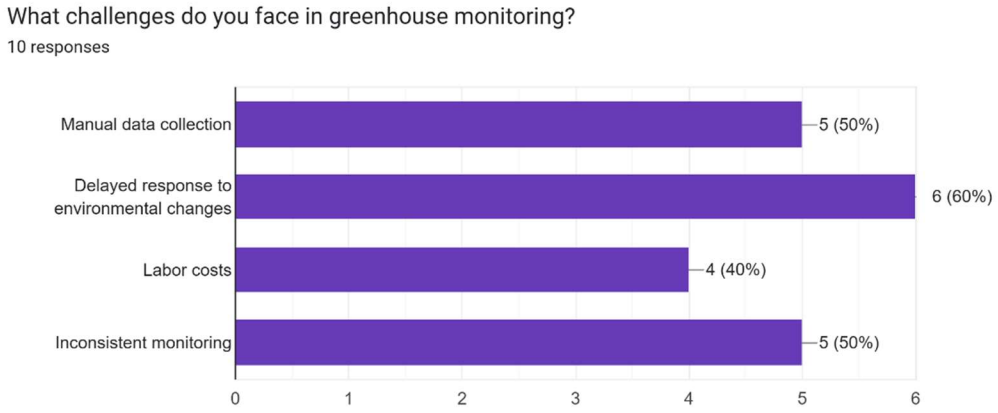


Figure 14 The fifth question for Section B

Operational Challenges and Opportunities

Survey responses highlighted several persistent challenges in greenhouse monitoring:

1. Delayed response to environmental changes (60% of respondents)
2. Manual data collection and inconsistent monitoring (50%)
3. Labor costs (40%)
4. Inconsistent Monitoring (50%)

These challenges echo findings from Kumar et al. (2020), who identified similar operational inefficiencies in traditional greenhouse management systems. The high percentage of respondents citing delayed responses to environmental changes particularly underscores the need for automated, real-time monitoring solutions.

The analysis reveals a growing awareness of IoT technology's potential in agricultural applications, coupled with significant opportunities for improvement in adoption and implementation. The high value placed on real-time monitoring (80% considering it very or extremely important) suggests strong potential for IoT solution providers to address current operational challenges through targeted technology development and deployment.

These findings have important implications for the development and implementation of IoT-based greenhouse monitoring systems. Future solutions should focus on addressing the identified challenges while considering the varying levels of technical expertise among potential users. As the agricultural sector continues to embrace digital transformation, the role of accessible, reliable IoT solutions becomes increasingly critical in optimizing greenhouse operations and enhancing agricultural productivity.

3.2.3 Section C: Expectations and Features for the Proposed System

Section C of the survey explores the features, alert methods, and data visualization preferences deemed essential by participants for IoT systems in agriculture. This analysis highlights the priorities and expectations of users, offering valuable insights for the design and development of such systems.

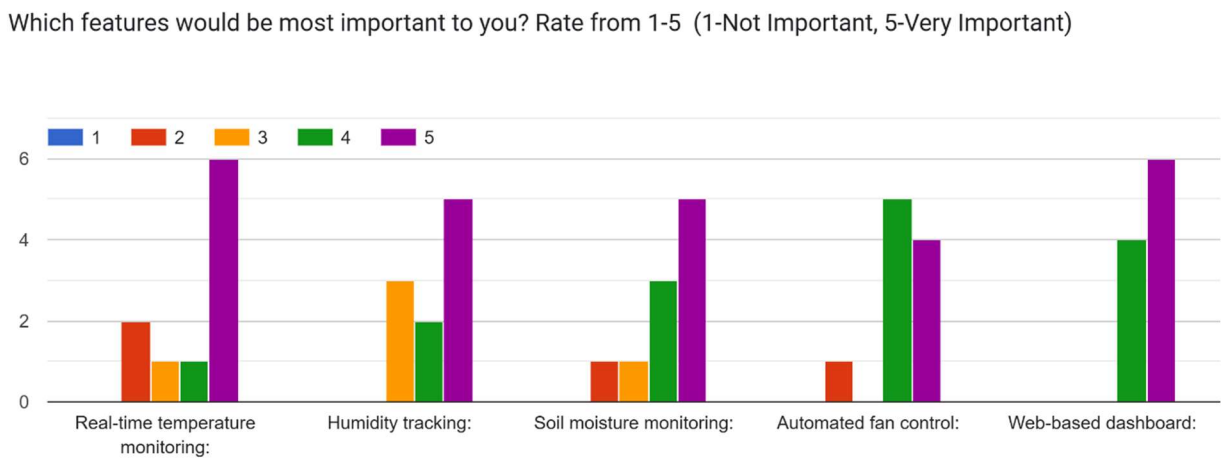


Figure 15 The first question for Section C

Importance of Features

The survey findings indicate a clear emphasis on features that enhance efficiency and automation in agricultural practices. Real-time temperature monitoring was rated as very important by 60% of respondents, with 80% rating it as important overall. This demonstrates a strong demand for precise and immediate temperature tracking to maintain optimal greenhouse conditions. Similarly, humidity tracking garnered a “very important” rating from 50% of participants, with a combined

importance rating of 70%, reflecting its significance in managing crop health and preventing adverse conditions.

Soil moisture monitoring also emerged as a critical feature, rated very important by 50% of respondents and achieving an overall importance rating of 80%. This finding underscores the growing awareness of resource efficiency and the importance of soil management in agriculture. Automated fan control was rated very important by 40% of participants, with 90% assigning it a high combined importance rating. This highlights a strong preference for automated environmental controls to reduce manual labor and enhance precision. Finally, the web-based dashboard received the highest collective importance rating, with 60% marking it as very important and 100% assigning it a combined importance score. This unanimous preference underscores the necessity of centralized, user-friendly interfaces for seamless monitoring and control.

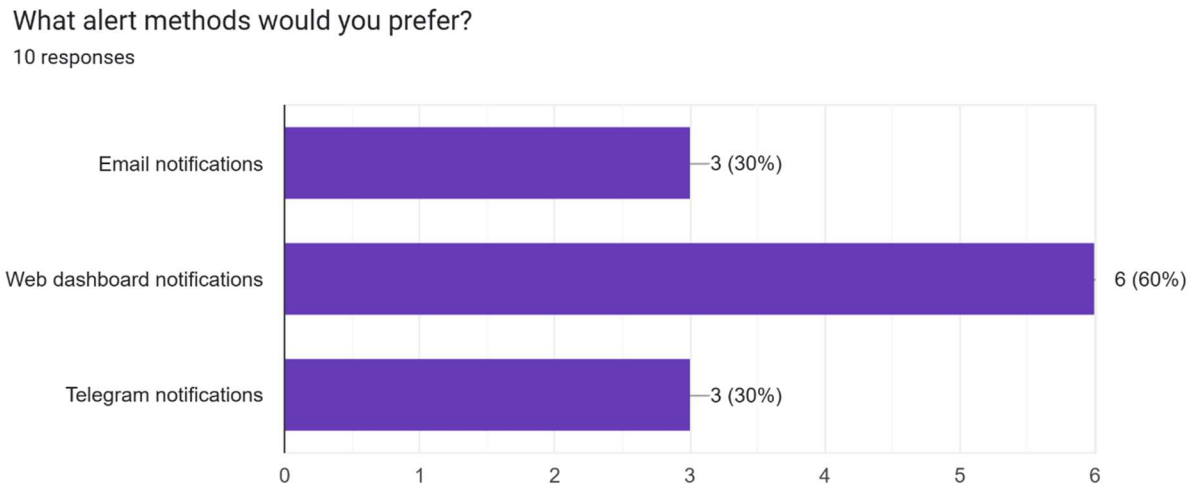


Figure 16 The second question for Section C

Preferred Alert Methods

Participants were asked to rank their preferred methods of receiving alerts, and web dashboard notifications emerged as the most favored, preferred by 60% of respondents. This aligns with the emphasis on web-based dashboards as a central feature for monitoring. Email and Telegram notifications were each preferred by 30% of respondents, indicating the need for a multi-channel alerting system to cater to diverse user preferences. This result suggests that while web dashboards are the primary choice, supplementary channels like email and Telegram are essential to enhance accessibility and ensure users remain informed across different platforms.

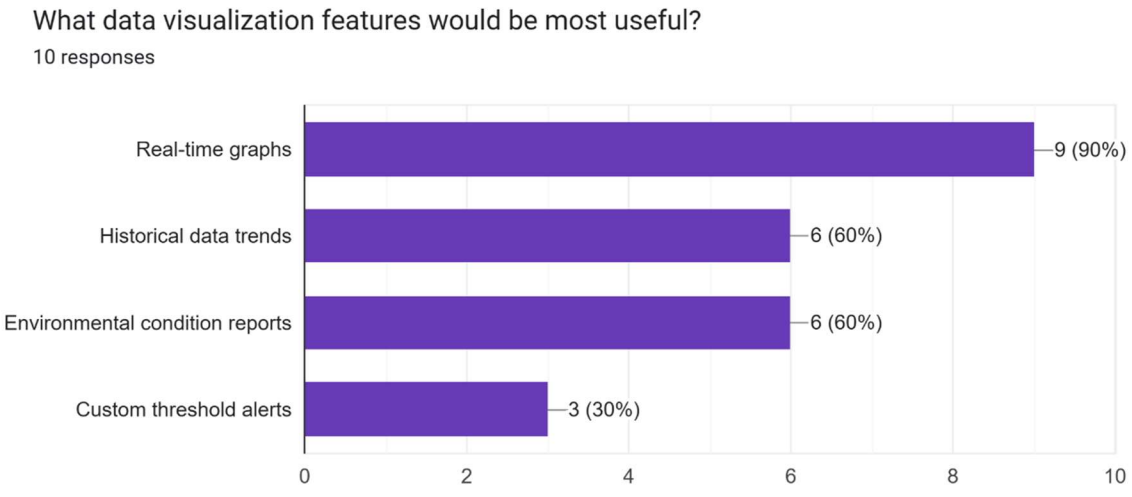


Figure 17 The third question for Section C

Useful Data Visualization Features

Data visualization is a critical component of IoT systems, and the survey findings reflect this sentiment. Real-time graphs were overwhelmingly preferred, with 90% of participants selecting this feature. This highlights the demand for immediate, actionable insights that enable real-time decision-making. Historical data trends and environmental condition reports were each chosen by 60% of respondents, emphasizing the importance of long-term analysis alongside real-time monitoring. Custom threshold alerts, while selected by 30% of participants, remain significant for a subset of users who prioritize proactive management through personalized alerts.

The analysis of Section C underscores the critical features and functionalities valued by users in IoT agricultural systems. Real-time monitoring of temperature, humidity, and soil moisture, coupled with automated environmental controls and web-based dashboards, emerged as top priorities. The preference for real-time graphs and historical trends indicates the need for visually intuitive and actionable data visualization tools. In terms of alert methods, web dashboards are the primary choice, with email and Telegram serving as complementary options for broader accessibility. By addressing these user preferences, IoT systems can enhance user satisfaction, improve agricultural efficiency, and contribute to more sustainable farming practices.

3.2.4 Section D: Impact and Feedback on Proposed System

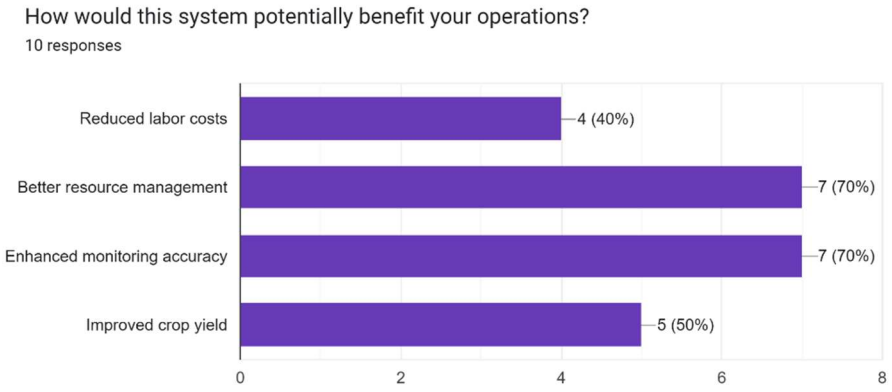


Figure 18 The First question for Section D

Benefits of the system and the impact for the operation

A significant 70% of respondents identified better resource management and enhanced monitoring accuracy as key advantages, indicating that these are high-priority areas for improvement in agricultural practices. Additionally, improved crop yield was noted by 50% of participants, highlighting its importance in ensuring agricultural productivity and profitability. Meanwhile, 40% cited reduced labor costs as a potential benefit, suggesting that automation and real-time monitoring could alleviate labor-intensive processes.

Figure 19 The second question for Section D

The challenges when implementing the system

A significant 40% of respondents identified budget and cost constraints as major challenges, indicating that financial limitations are a high-priority concern for many. Additionally, 10% of participants highlighted the importance of ensuring the durability of system components, emphasizing the need for robust and reliable equipment. Meanwhile, 20% cited the accuracy of data monitoring as a critical factor, suggesting that precise and reliable data collection is essential for the system's effectiveness. Lastly, 30% of respondents did not give any answer for the potential challenge.

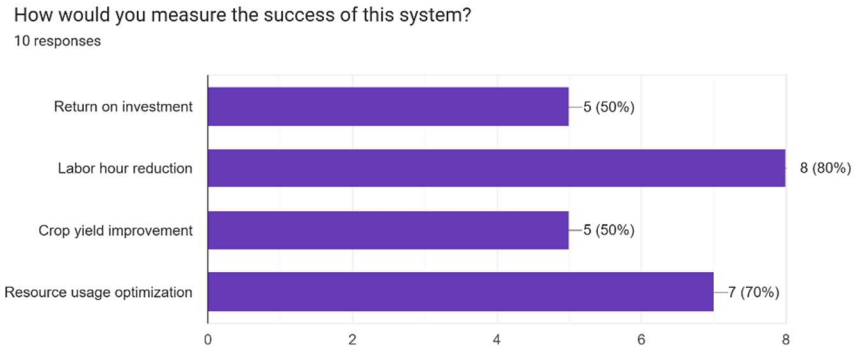


Figure 20 The third question for Section D

The success of the system

A significant 80% of respondents identified labor hour reduction as the primary measure of success, indicating that time efficiency and automation are high-priority outcomes for system users. Additionally, 70% of participants highlighted resource usage optimization as a critical success metric, emphasizing the importance of efficient management of greenhouse resources and inputs. Meanwhile, 50% cited both return on investment and crop yield improvement as key indicators, suggesting that financial returns and production improvements are equally valued but secondary to operational efficiency. These results demonstrate that users primarily measure success through operational improvements rather than purely financial or production metrics.

3.3 System Design

3.3.1 Block Diagram

The block diagram illustrates the functional architecture of a Greenhouse Monitoring System using MQTT. It is designed to automate greenhouse management by integrating sensors, processing units, communication protocols, output devices, and a user interface. The system begins with sensors that monitor environmental conditions essential for plant growth. These include a temperature sensor to track the ambient temperature, a humidity sensor to measure relative humidity, and a soil moisture sensor to evaluate the soil's moisture content. Together, these sensors collect data critical for maintaining optimal conditions within the greenhouse.

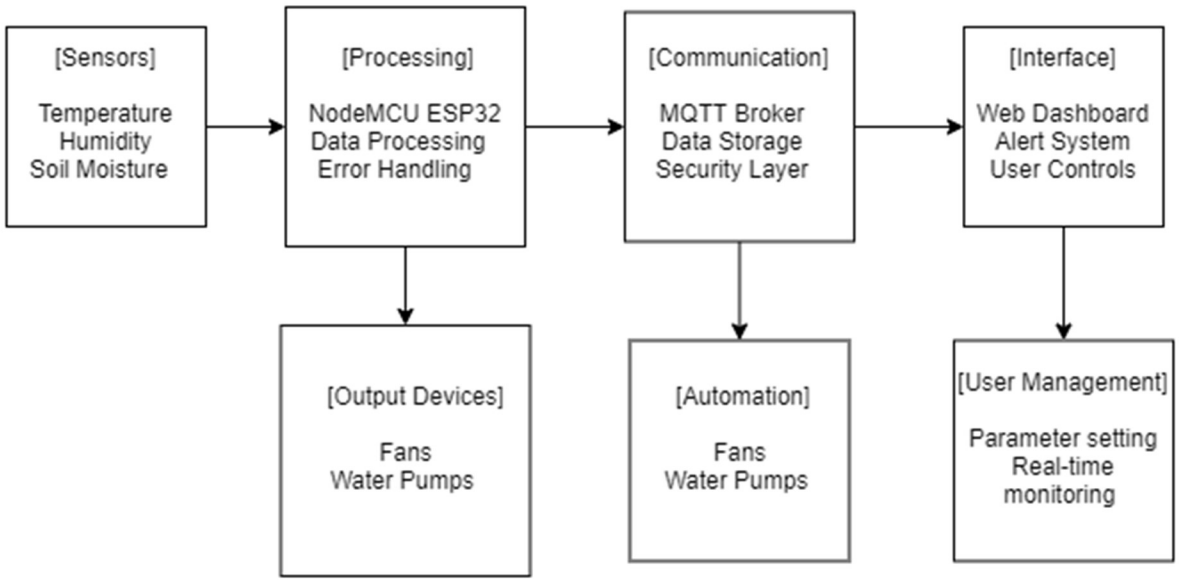


Figure 21 Block diagram Greenhouse Monitoring using MQTT

The collected data is processed by the NodeMCU ESP32 microcontroller, which serves as the system's processing unit. The NodeMCU ESP32 is responsible for data processing, which involves

converting raw sensor readings into meaningful insights. It also incorporates error-handling mechanisms to ensure the accuracy and reliability of the data, filtering out any inconsistencies or noise that could compromise the system's performance. This processed data is then relayed to the communication module for further action.

The communication component uses an MQTT broker to facilitate data transfer between the NodeMCU ESP32 and the user interface. The MQTT protocol, known for its lightweight and efficient nature, is ideal for IoT applications, ensuring seamless and reliable communication. Data is also stored securely for real-time monitoring and historical analysis, with a security layer in place to protect transmissions from unauthorized access. This robust communication framework ensures that data is transmitted efficiently and securely across the system.

The system includes output devices such as fans and water pumps, which are activated to regulate the greenhouse environment. Fans are used for temperature control and ventilation, while water pumps irrigate the plants when soil moisture levels fall below preset thresholds. These output devices are controlled automatically by the system, reducing the need for human intervention. For example, if the temperature exceeds a certain limit, the fans are automatically turned on, and if the soil moisture drops below the desired level, the water pumps are activated. This automation ensures that the greenhouse environment remains within optimal conditions for plant growth.

A user interface, accessible via a web dashboard, allows users to monitor and control the system remotely. The dashboard displays real-time data, provides visualizations such as graphs, and issues

alerts for any anomalies in the greenhouse conditions. Users can also adjust system settings, such as temperature and moisture thresholds, and manually override automation if needed. This interface is designed to be user-friendly, enabling both novice and experienced users to manage the system efficiently.

The user management component further enhances the system's adaptability and accessibility. It allows users to customize parameters such as temperature and humidity settings to suit different plants and environments. The real-time monitoring capability ensures that users are constantly updated on the greenhouse's status, enabling timely interventions when necessary. By combining automation, remote monitoring, and user customization, the system provides a highly efficient and user-centric approach to greenhouse management.

This greenhouse monitoring system represents a significant advancement in agricultural technology. It leverages IoT-enabled remote monitoring, automation, and secure communication to optimize resource usage and reduce manual labor. Its modular design allows for scalability, enabling the integration of additional sensors and actuators as needed. With features such as energy efficiency, resource optimization, and a user-friendly interface, this system offers a comprehensive solution for modern greenhouse operations, paving the way for smarter and more sustainable agricultural practices.

3.3.2 Greenhouse Monitoring System Use Case Diagram

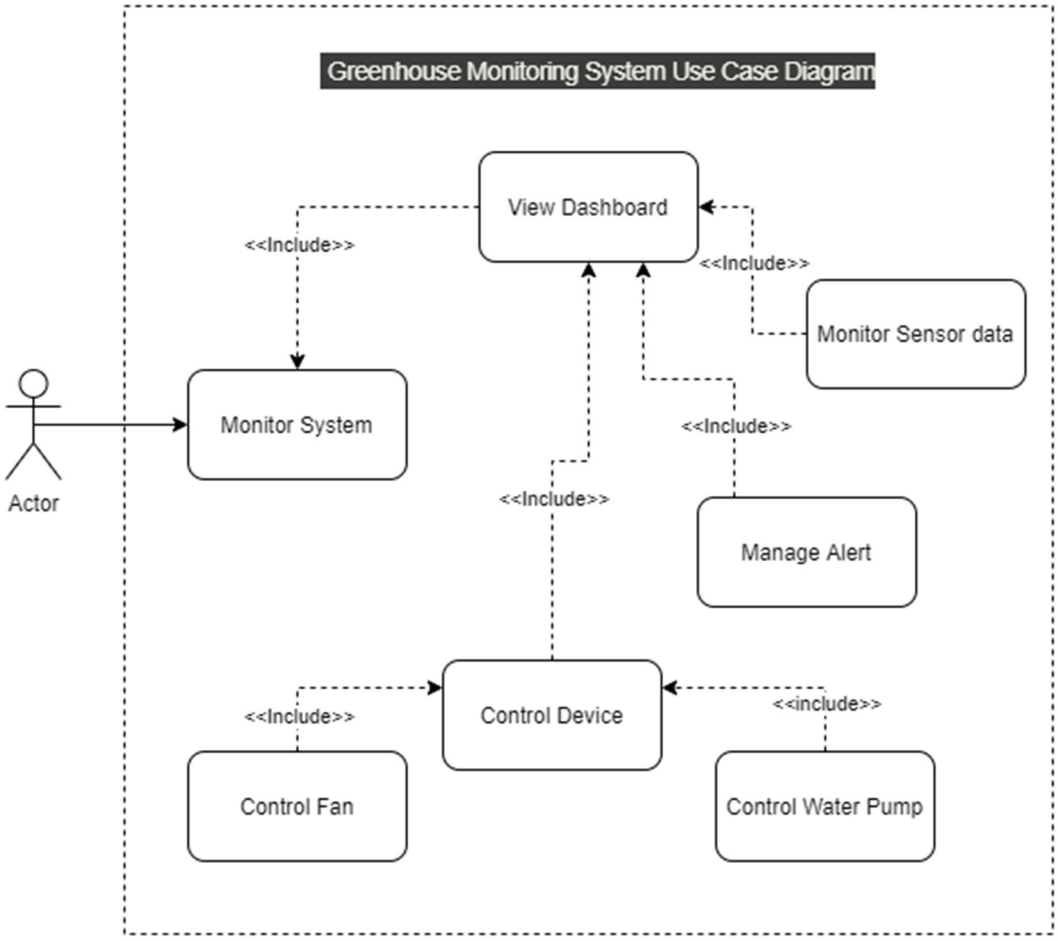


Figure 22 Greenhouse Monitoring System Use Case Diagram

Use Case Number	UCN 1
Use Case Name	Monitor Environmental Data
Actor	User
Description	Allows users to view real-time environmental conditions in the greenhouse
Pre-conditions	1. User is logged into the system 2. Sensors are operational and connected 3. System is online
Post-conditions	1. Environmental data is displayed 2. Historical data is logged 3. Any alerts are generated if thresholds are exceeded
Main Flow	1. User accesses the monitoring dashboard 2. System displays current readings for: • Temperature • Humidity • Soil moisture 3. System updates readings in real-time 4. User can view historical data trends 5. System displays status of all connected sensors
Alternative Flow	If sensors fail: • System displays error message • Marks affected readings as unavailable • Sends notification to user

	If connection is lost: • System attempts to reconnect • Displays last known readings with timestamp • Notifies user of connectivity issue

Use Case Number	UCN 2
Use Case Name	Control Output Devices
Actor	User
Description	Enables user to control greenhouse automation devices
Pre-conditions	• User is authenticated • Output devices are connected • User has control permissions
Post-conditions	• Device states are updated • Actions are logged • System returns to monitoring mode

Main Flow	1. User selects device control panel
	2. System displays current status of:
	  • Fans
	  • Water pumps
	3. User can toggle devices on/off
	4. System executes commands
	  • System confirms action completion

Alternative Flow	If device fails to respond: • System retries command • Displays error message • Logs failure event If automatic override is active:

	• System notifies user • Explains override reason • Maintains automatic control

Use Case Number	UCN 3
Use Case Name	Manage Alerts
Actor	User
Description	Allows configuration and management of system alerts
Pre-conditions	• User is authenticated • Alert system is operational
Post-conditions	• Alert settings are updated • Configuration is saved • Test notification is sent

Main Flow	1. User accesses alert configuration
	2. System displays current alert settings
	3. User can:
	  • Set threshold values
	  • Configure notification methods
	4. Define alert priorities
	5. System validates settings
	6. System activates new alert rules

Alternative Flow	If validation fails: • System highlights invalid settings • Provides correction guidance • Maintains previous settings If notification system is unavailable:

	• System offers alternative methods • Logs configuration attempt • Notifies administrator

Use Case Number	UCN 4
Use Case Name	View Sensor Data
Actor	User
Description	Enables users to view detailed sensor readings and data analysis from all greenhouse sensors
Pre-conditions	• User is logged into the system • Sensors are functional and transmitting data • User has appropriate access permissions
Post-conditions	• Sensor data is displayed accurately • Data history is accessible • System logs user's data access • Any data exports are completed
Main Flow	1. User navigates to sensor data view 2. System retrieves data from: • Temperature sensor • Humidity sensor • Soil moisture sensor 3. System displays: • Current readings for all sensors • Measurement units • Last update timestamp • Data visualization graphs

	4. User can select different time ranges: • Real-time • Past 24 hours • Past week • Custom date range 5. System allows data export functionality 6. User can toggle between different visualization types: • Line graphs • Bar charts • Numerical displays
Alternative Flow	If sensor data is unavailable: • System displays "No Data" message • Shows last known reading with timestamp • Indicates sensor status/error If data is outside normal range: • System highlights abnormal values • Displays warning indicators • Shows acceptable range limits If historical data is corrupted: • System shows partial data

	• Indicates data gaps • Provides error explanation

3.3.3 Greenhouse Monitoring System Sequence Diagram

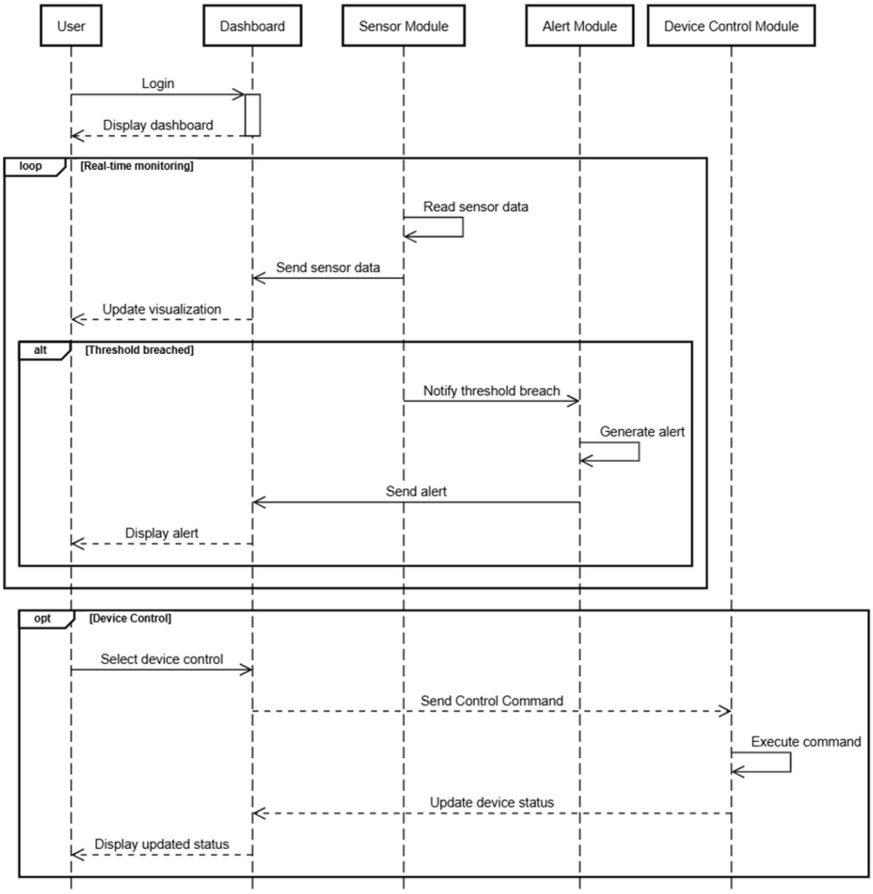


Figure 23 Greenhouse Monitoring System Sequence Diagram

Actors and Components:

- **Actor**: User
- **System Components**:
  - Dashboard
  - Sensor Module
  - Alert Module
  - Device Control Module

Sequence Flow:

1. User logs into the system and selects an option to monitor the greenhouse.
2. The system displays the dashboard with real-time data from sensors.
3. The Sensor Module retrieves data from connected sensors (temperature, humidity, soil moisture).
4. The Dashboard Module visualizes the data for the user.
5. If a predefined threshold is breached, the Alert Module generates an alert, which is displayed on the dashboard.
6. The user can interact with the Device Control Module via the dashboard to control devices like fans or water pumps.
7. The Control Module sends commands to the respective devices, and the system updates their status on the dashboard.

3.3.4 Greenhouse Monitoring System Class Diagram

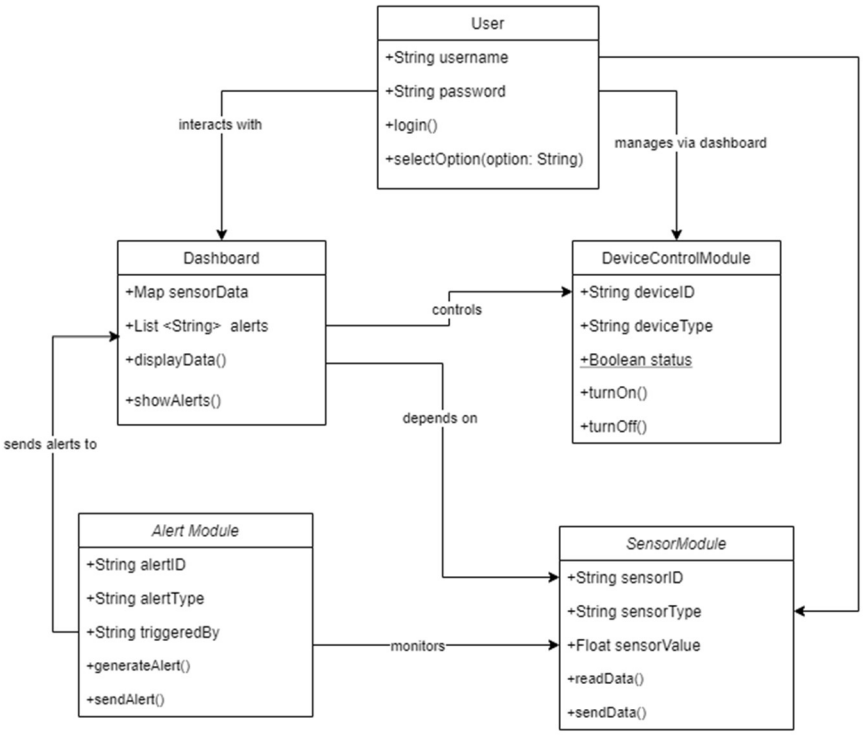


Figure 24 Greenhouse Monitoring System Class Diagram

The Greenhouse Monitoring System is built around five key classes that work together to provide comprehensive monitoring and control capabilities.

The heart of the system lies in the User class, which represents the greenhouse operators or managers who interact with the system. Each user has basic attributes like a username and password for security purposes. Through the login() method, users can authenticate themselves, and once logged in, they can select various monitoring and control options using the selectOption() method.

Moving to the central component, the Dashboard class serves as the main interface between users and the system's functionality. Think of it as the control center where everything comes together.

The Dashboard maintains two crucial pieces of information: a map of sensorData that stores various sensor readings, and a list of alerts for any issues that need attention. Its methods, displayData() and showAlerts(), ensure that users can easily view and understand the greenhouse's current state.

The SensorModule class handles all sensor-related operations. Each sensor in the greenhouse is given a unique sensorID and has a specific sensorType (like temperature or humidity). The current reading is stored in sensorValue. This class has two main responsibilities: readData() collects readings from the physical sensors, and sendData() transmits these readings to the dashboard for display.

To keep everything running safely, the AlertModule class monitors the system for any concerning conditions. Each alert gets a unique alertID, and the alertType specifies what kind of issue it represents (such as high temperature or low moisture). The triggeredBy attribute tracks which sensor or condition initiated the alert. When something goes wrong, generateAlert() creates a new alert, and sendAlert() makes sure it reaches the dashboard where users can see it.

Finally, the DeviceControlModule class manages the physical devices in the greenhouse, like fans or water pumps. Each device has its own deviceID and deviceType, and its current state is tracked by the status attribute. The class provides straightforward control through turnOn() and turnOff() methods.

These classes are interconnected in meaningful ways. Users interact with the system primarily through the Dashboard, which in turn depends on the SensorModule for real-time data. The AlertModule keeps an eye on sensor readings and communicates with both the Dashboard and

SensorModule. When users need to control devices, their commands flow from the User class, through the Dashboard, to the DeviceControlModule.

This architecture creates a robust system where data flows smoothly from sensors to users, alerts are generated when needed, and users can respond by controlling greenhouse devices. The separation of concerns between classes makes the system both maintainable and extensible, while the clear relationships between components ensure efficient operation.

The system follows a logical flow: sensors continuously gather data, the dashboard presents this information clearly to users, the alert system watches for any issues, and users can take action through device controls when necessary. This creates a complete cycle of monitoring, alerting, and control that helps maintain optimal greenhouse conditions.

3.3.5 Interface Design

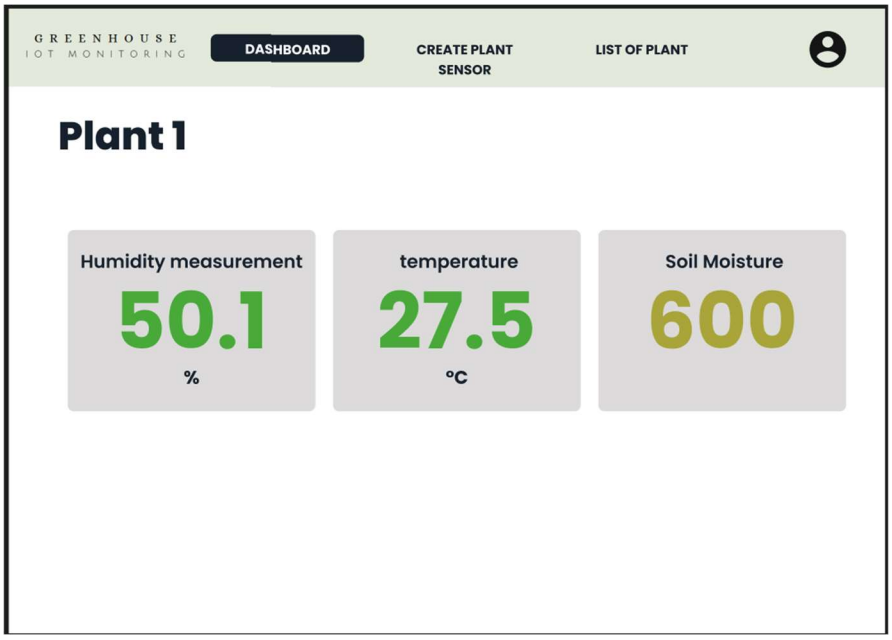


Figure 25 Greenhouse iot monitoring UI

This figure 34 represents a dashboard from the Greenhouse IoT Monitoring System. It displays real-time sensor data for a specific plant, labeled "Plant 1."

The dashboard includes the following measurements:

1. **Humidity Measurement**: Shows the current humidity level as 50.1%, providing information about the air's moisture content.
2. **Temperature**: Displays the plant's surrounding temperature as 27.5°C, essential for monitoring greenhouse conditions.
3. **Soil Moisture**: Indicates the soil's moisture level with a value of 600, likely on a scale relevant to the sensor used.

The navigation bar at the top provides options to switch between sections:

- **Dashboard**: Displays the current view for real-time data monitoring.
- **Create Plant Sensor**: Likely used to add new sensors or configure monitoring for additional plants.
- **List of Plant**: Provides an overview or list of all monitored plants.

3.4 System Requirements

3.4.1 Hardware

The hardware required to develop the proposed Greenhouse IoT Monitoring System using MQTT includes a NodeMCU ESP32 microcontroller, which acts as the central processing unit with built-in WiFi capabilities for seamless communication and GPIO pins to connect sensors and actuators. The sensors employed include a temperature sensor, a humidity sensor, and a soil moisture sensor, all of which gather real-time data on critical environmental conditions within the greenhouse. For automation, control equipment such as cooling fans, water pumps, and relay modules are used to adjust environmental parameters and ensure optimal plant growth.The system also relies on a robust power infrastructure, including an external power supply for primary energy and a battery backup to maintain uninterrupted operation during power outages. On the software side, MQTT is used as the communication protocol to enable efficient data transmission between the microcontroller and the centralized dashboard, allowing for real-time monitoring and remote control. These hardware and software components work in synergy to create a reliable, automated, and scalable greenhouse management system that reduces manual labor and optimizes agricultural productivity.

3.4.2 NodeMCU ESP32

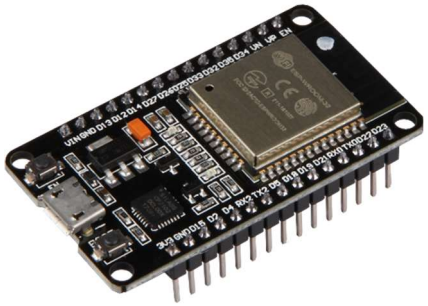


Figure 26 NodeMCU ESP32

The NodeMCU ESP32 in figure 17 microcontroller serves as the core processing unit for the greenhouse IoT monitoring system. Equipped with built-in WiFi capabilities, the ESP32 enables seamless MQTT communication with the centralized dashboard, ensuring real-time data transmission and device control. Its versatile GPIO (General Purpose Input/Output) pins allow for the connection of various sensors and control equipment, such as temperature, humidity, and soil moisture sensors, as well as actuators like fans and water pumps.

In this project, the ESP32 is responsible for collecting environmental data from connected sensors, processing the data to ensure accuracy, and transmitting it to the MQTT broker. The microcontroller also receives commands from the dashboard to activate or deactivate control equipment, such as turning on cooling fans or water pumps, based on user inputs or automated triggers.

3.4.3 Sensors

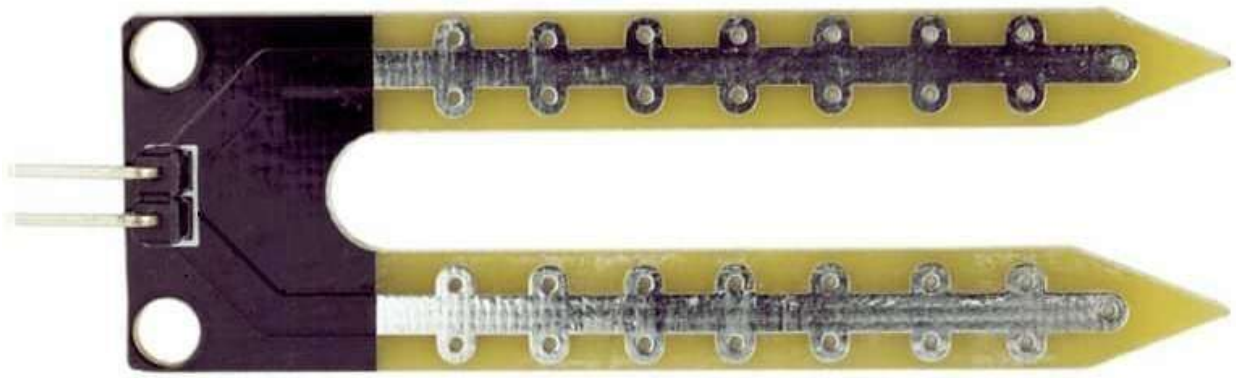


Figure 27 Soil Moisture Sensor

The soil moisture sensor in figure 18 plays a crucial role in the Greenhouse IoT Monitoring System Using MQTT by providing real-time data on the moisture content of the soil. This sensor ensures that the greenhouse maintains optimal soil conditions for plant growth by integrating seamlessly with the system's NodeMCU ESP32 microcontroller. It measures the soil's moisture level and outputs either digital or analog signals to the microcontroller, which then processes the data and transmits it via the MQTT protocol to a centralized dashboard.

In the context of the project, the digital output is used to trigger automated actions such as activating water pumps when the soil moisture falls below a pre-defined threshold. The analog output, on the other hand, provides precise readings that are displayed on the web-based dashboard. This allows users to monitor and analyze moisture levels over time, enabling better resource management and informed decision-making.

The sensor's reliability and ease of integration make it an ideal choice for the project, contributing to the system's ability to automate irrigation, reduce water wastage, and improve overall greenhouse efficiency. By leveraging the MQTT protocol, the moisture data collected by the sensor

is transmitted in real time, ensuring timely alerts and updates to users, enhancing the effectiveness of the greenhouse monitoring system.

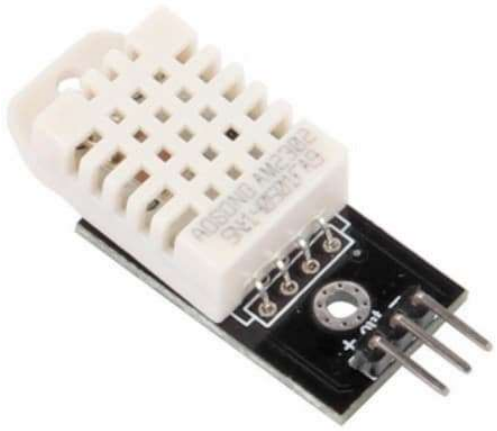


Figure 28 DHT22 Sensor

The DHT22 sensor in the figure 19 is an essential component in the Greenhouse IoT Monitoring System Using MQTT, providing real-time temperature and humidity data. Its reliability, accuracy, and ease of integration make it well-suited for greenhouse applications where maintaining optimal environmental conditions is critical for plant health and productivity.

In this project, the DHT22 sensor measures the temperature and humidity of the greenhouse environment and sends the data to the NodeMCU ESP32 microcontroller. The microcontroller processes this data and transmits it using the MQTT protocol to a centralized web-based dashboard. This allows users to monitor these critical parameters in real time, ensuring that any deviations from the ideal conditions can be addressed promptly.

The sensor's temperature range of 0-50°C with ±2% accuracy and humidity range of 20-80% RH with ±5% accuracy is well-suited for typical greenhouse environments. Its compatibility with the

NodeMCU ESP32 ensures seamless integration, and its long-term stability ensures reliable performance over time.

By leveraging the DHT22 sensor, the system can also trigger automated responses, such as activating cooling fans or water pumps, based on predefined thresholds. For example, if the temperature exceeds a certain level, the system can activate fans to reduce heat. This integration ensures an efficient and automated approach to greenhouse management, reducing manual intervention while maintaining ideal growing conditions.

3.4.4 Control Equipment

Figure 29 12V Cooling Fan for ESP 32

The cooling fan on Figure 20 serves as a critical control component in the Greenhouse IoT Monitoring System Using MQTT, ensuring proper ventilation and temperature regulation within the greenhouse. This lightweight and compact DC brushless fan, measuring 50mm x 50mm x 10mm and weighing only 65g, provides a reliable and energy-efficient cooling solution tailored for small-scale automated systems.

In this project, the cooling fan operates at 5VDC with a power consumption of 0.65W and a low noise level of 30 dB, making it ideal for quiet greenhouse environments. It integrates seamlessly with microcontrollers such as the NodeMCU ESP32 used in the system. Tutorials are available for platforms like Arduino, ESP32, and Raspberry Pi, simplifying its implementation.

The fan is activated based on real-time temperature data collected by the DHT11 sensor and processed by the NodeMCU ESP32. When the greenhouse temperature exceeds the predefined

threshold, the system triggers the fan to circulate air, reducing heat buildup and creating an optimal environment for plant growth.

The cooling fan's compatibility with Arduino, ESP32, ESP8266, Raspberry Pi, or any 5V or 3.3V microcontroller ensures flexibility in integration, while its aluminum and plastic construction provides durability. This component plays a vital role in automating temperature control, reducing manual intervention, and maintaining consistent environmental conditions, a critical factor for successful greenhouse operations.

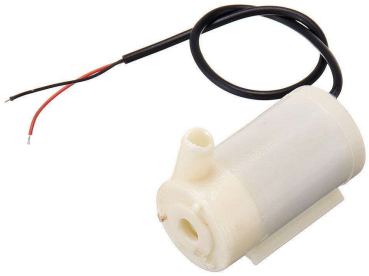


Figure 30 Micro Submersible Water Pump DC 3V-5V

The micro mini submersible water pump in figure 21 is an essential component of the Greenhouse IoT Monitoring System Using MQTT, enabling efficient and automated water delivery to plants. Compact and lightweight, this pump measures 4.5x3.0x2.5 cm and weighs just 30g, making it suitable for integration into small-scale greenhouse systems. Operating on a DC input voltage of 3V-5V, the pump delivers a flow rate of 1.2-1.6 L/min, providing adequate water for maintaining soil moisture levels. Its low operating current of 0.1-0.2A ensures energy efficiency, an important consideration for IoT systems powered by battery backups or external supplies. With a maximum

suction distance of 0.8 meters and an operational temperature tolerance of up to 80°C, the pump is robust and reliable for consistent use in greenhouse environments.

The water pump is triggered by real-time data from the soil moisture sensor, processed by the NodeMCU ESP32. When the soil moisture drops below a predefined threshold, the system activates the pump to irrigate plants automatically. This eliminates manual watering efforts, ensuring precise and timely hydration for optimal plant growth. With a lifespan of 500 operational hours, this pump is designed for durability in repetitive tasks. Its ultra-quiet operation and amphibious capabilities make it ideal for use in a controlled greenhouse setting, where noise reduction and flexibility are key. The pump's integration with the system highlights the project's focus on automating agricultural processes, conserving resources, and enhancing productivity.

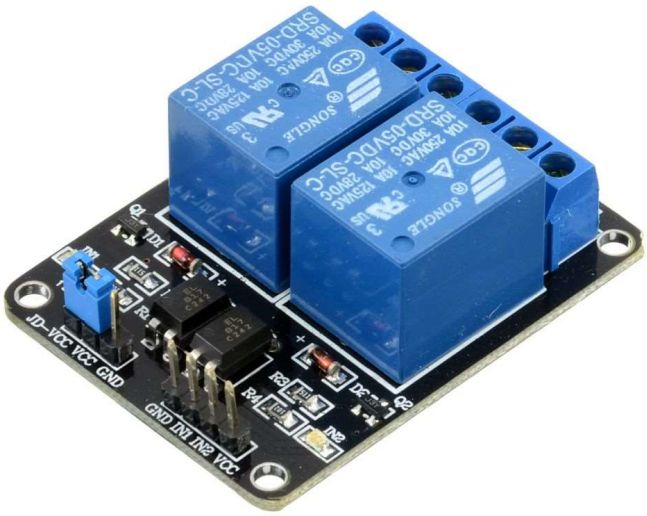


Figure 31 5V 2 Channel Relay Module 10A

The relay module in figure 22 plays a critical role in the Greenhouse IoT Monitoring System Using MQTT, enabling the automation of various control equipment such as cooling fans and water

pumps. This versatile relay board features optically isolated inputs for enhanced safety and reliability, making it ideal for handling high-power devices within the greenhouse environment.

The module operates on a 5V supply voltage and supports input signals of the same range. It is equipped with a high-current relay capable of driving up to 10A at 28VDC, allowing it to control powerful equipment without the risk of overload. The inclusion of a jumper to select between external or common Vcc adds flexibility, catering to diverse power configurations.

Measuring 39mm x 51mm with M3 mounting holes spaced approximately 33mm x 44mm, the relay board is compact and easy to install within the greenhouse setup. Straight headers for control signals simplify connections to the NodeMCU ESP32 microcontroller, which governs the relay's operation based on real-time sensor data.

In the system, the relay module acts as a switch, controlling devices like the cooling fans and water pump based on environmental conditions. For instance, when temperature or humidity levels deviate from the desired range, the relay is activated to power the appropriate equipment, ensuring optimal conditions for plant growth. The optical isolation feature enhances the safety of the system by preventing electrical noise or surges from affecting the microcontroller.

This relay board underscores the project's emphasis on automation, precision, and reliability, contributing to the efficient operation of the greenhouse IoT system.

3.4.5 Power System

To determine the appropriate power adapter for the greenhouse IoT monitoring system, it is crucial to evaluate the voltage and current requirements of all connected components. Each device in the system has specific power needs, and the total power demand must be calculated to ensure the adapter can supply sufficient energy for smooth operation.

The NodeMCU ESP32, which serves as the system's microcontroller, operates at 3.3V and consumes approximately 200mA during peak usage, such as Wi-Fi transmission. The soil moisture sensor requires 3.3V to 5V with minimal current consumption of around 10 to 30mA. Similarly, the DHT11 sensor operates between 3V and 5.5V and draws a negligible 2.5mA.

The system also includes two cooling fans, each operating at 5V and consuming 0.13A. Together, they require a total of 260mA. The submersible water pump, another critical component, operates within the range of 3V to 5V and consumes up to 200mA during peak operation. Additionally, the relay module, which controls the fans and pump, operates at 5V and draws approximately 70 to 80mA per relay, resulting in a combined current requirement of around 160mA for two relays.

When these individual power demands are summed, the total current consumption of the system is approximately 852.5mA, or 0.85A. Since most components are compatible with a 5V power supply, selecting a 5V adapter ensures compatibility across the system. To maintain reliability and account for fluctuations or future expansions, a safety margin of at least 30% is added to the total current requirement. With this margin, the adjusted current need becomes approximately 1.1A.

Based on this analysis, the system requires a power adapter that provides 5V and at least 1.5A of current. To further ensure stability and accommodate unforeseen increases in power demand, a 5V,

2A power adapter is recommended. This choice not only meets the current requirements but also provides a buffer for reliable and efficient operation of the greenhouse IoT monitoring system.

Figure 32 5V 2.5A / 3A Micro USB Power Supply Adapter for Raspberry Pi 3

Based on this figure 23 considering the power needs and compatibility of the system components, a **5V, 2.5A or 3A Micro USB Power Supply Adapter** is an ideal choice. This adapter is specifically designed for devices like the Raspberry Pi 3, Arduino, and NodeMCU ESP32, making it highly suitable for this IoT project. It offers ample power to meet the system's requirements while also providing a buffer for additional components or future expansions.

By opting for this adapter, the system benefits from stable and efficient power delivery, ensuring the smooth operation of the greenhouse IoT monitoring system while accommodating potential future needs.

3.4.6 Software

The Greenhouse IoT Monitoring System Using MQTT relies on a combination of advanced development tools to achieve its objectives. These tools are essential in ensuring that the system effectively monitors and controls greenhouse conditions while providing users with real-time insights and data visualization. Two critical aspects of this system are programming the ESP32 microcontroller and designing an intuitive web dashboard.

1. Arduino IDE for ESP32

The Arduino IDE (Integrated Development Environment) is used to program the ESP32 microcontroller. It provides:

- A simplified C++ programming environment
- Built-in libraries for ESP32 functionality
- Tools for code compilation and uploading
- Serial monitor for debugging
- Support for ESP32 board management

2. Web Development Framework for Dashboard

For the web dashboard, we're using HTML, CSS, and JavaScript with the following:

- Frontend: Tailwind CSS for styling
- Charts and visualizations: Chart.js
- Real-time updates: WebSocket implementation

- User interface components: Shadcn/ui library

3. MQTT Broker Software

Using Mosquitto MQTT broker for message handling:

- Open-source MQTT broker
- Lightweight and efficient
- Support for MQTT protocol v3.1 and v3.1.1
- Built-in security features
- Secure data transmission

Communication

1. MQTT Protocol

The MQTT (Message Queuing Telemetry Transport) protocol is as follows:

- Topics structure:

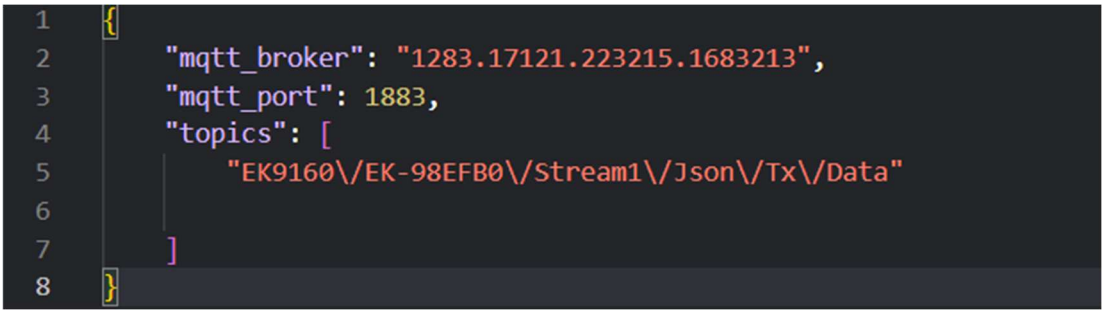


Figure 33 Topic List

- Quality of Service (QoS) level 1 for reliable delivery
- Retain messages for last known values
- Keep-alive messages for connection monitoring

2. WebSocket for Real-time Updates

WebSocket implementation provides:

- Full-duplex communication
- Persistent connection
- Event-driven updates
- Binary and text message support

3. Data Formatting and Validation

Data handling procedures include:

- JSON formatting for MQTT messages:

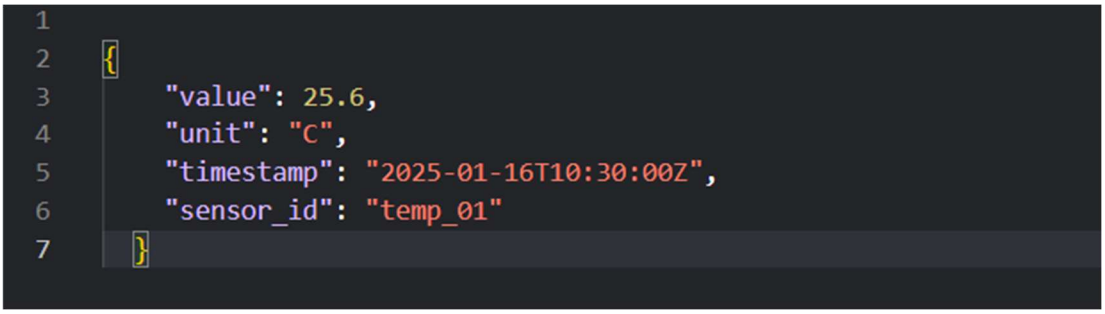


Figure 34 Json Data Handling

- Data validation:
    - Range checking for sensor values
    - Timestamp validation
    - Data type verification
    - Error detection

Each component plays a crucial role in creating a reliable and efficient greenhouse monitoring system. The Arduino IDE handles the hardware interface, the web framework provides user interaction, and the MQTT broker ensures reliable communication. The communication protocols and data handling ensure that information flows smoothly and accurately throughout the system.

3.5 Conclusion

In this chapter, a comprehensive review of the literature and technical resources necessary for developing the greenhouse IoT monitoring system using MQTT was presented. The exploration of hardware components, such as the NodeMCU ESP32, sensors, control equipment, and power systems, provided detailed insights into their functionalities and roles within the project. Software tools and protocols, including MQTT, were also discussed, emphasizing their importance in facilitating real-time data transmission and efficient system management.

The analysis of survey results further highlighted the critical features valued by users, such as real-time monitoring, automated control, and user-friendly dashboards, underscoring the practical requirements for an IoT-based agricultural solution. These findings guided the selection and integration of components, ensuring that the system aligns with user expectations and operational needs.

Overall, this chapter laid the foundation for the system's development by synthesizing relevant research and technical knowledge. It established the framework for designing and implementing a robust, efficient, and user-centric greenhouse IoT monitoring solution, which will be further detailed in subsequent chapters.

Chapter 4: Methodology and Implementation

4.1 Introduction

This chapter delineates the practical implementation of the Greenhouse IoT Monitoring System, translating the conceptual designs from Chapter 3 into a tangible working application. It commences by detailing the systematic installation and configuration of all requisite software and development tools, laying the foundational environment for the project. Following this, a comprehensive exposition of the development process for each major system component is provided, encompassing the web-based user interface, backend logic, integrated hardware, and the mechanisms facilitating their seamless interaction. Furthermore, this chapter identifies and elaborates upon the technical challenges encountered throughout the implementation journey, presenting the solutions devised to mitigate these obstacles. The chapter concludes with a succinct summary of the developed system's practical realization.

4.2 Tools and Technologies

The successful implementation of the Greenhouse IoT Monitoring System necessitated the prior installation and meticulous configuration of several key software applications and development tools. These tools collectively provided the integrated environment essential for both web application development and microcontroller programming.

XAMPP, an essential open-source web server solution stack, served as the local development environment. Its installation involved a straightforward process of downloading the installer from the official Apache Friends website and selecting the core components: the Apache HTTP Server, MariaDB (a robust fork of MySQL), and interpreters for PHP. Once installed, the Apache and MySQL services were initiated and managed through the XAMPP Control Panel, operating on their default ports 80 and 3306, respectively, thereby establishing the local server infrastructure for the PHP-driven web application and its associated database.

Complementing XAMPP, phpMyAdmin emerged as the primary tool for managing the system's relational database. Accessible via a web browser through http://localhost/phpmyadmin, this free, web-based software, written in PHP, offered an intuitive graphical interface for MariaDB administration. It was instrumental in the creation of the greenhouse_iot database and the definition of its integral tables, including users, plants, sensors, sensor_data, alerts, and devices. Beyond schema definition, phpMyAdmin facilitated comprehensive data management operations, such as adding, modifying, and deleting records, and executing complex SQL queries crucial for development and testing.

For source code editing and overall project development, Visual Studio Code (VS Code) was adopted as the primary Integrated Development Environment (IDE). Its lightweight yet powerful nature, coupled with extensive support for various programming languages, made it an ideal choice. The installation was direct from its official website, followed by the integration of critical extensions for PHP, HTML, CSS (particularly for Tailwind CSS integration), JavaScript, and Arduino C++. These extensions significantly augmented development efficiency by providing enhanced syntax highlighting, intelligent code completion, and streamlined debugging capabilities, serving as the central hub for both web application development and the review of ESP32 firmware.

The microcontroller programming specifically utilized the Arduino IDE. This environment was fundamental for writing, compiling, and uploading the firmware to the NodeMCU ESP32 boards. The setup involved installing the ESP32 board manager to enable compatibility and then integrating necessary libraries such as DHT sensor libraries for environmental data acquisition, PubSubClient for MQTT communication, and ArduinoJson for efficient data serialization. This complete setup within the Arduino IDE was vital for developing the embedded software responsible for sensor readings and device control.

Finally, Mosquitto, an open-source message broker implementing the MQTT protocol, was deployed to facilitate seamless machine-to-machine communication. Installed as a lightweight service on the local development machine, Mosquitto was configured to listen on the standard MQTT port 1883. For development purposes, anonymous access was temporarily enabled to simplify client connections from both the ESP32 and the PHP subscriber. Its established IP address served as the central point for all sensor data publications and control command subscriptions, ensuring consistent communication across all system components. Additional tools and libraries,

such as Composer for PHP dependency management, HP MQTT subscriber script responsible for a WebSocket to MQTT forwarder, Tailwind CSS for utility-first styling, and Chart.js for client-side data visualization, were also integrated to complete the development ecosystem.

4.3 Implementation Details

The Greenhouse IoT Monitoring System was meticulously implemented through the development of several interconnected modules, each designed to fulfill specific functionalities critical to the overall operation. This section provides a detailed account of the development process for each significant feature.

4.3.1 User Authentication and Authorization

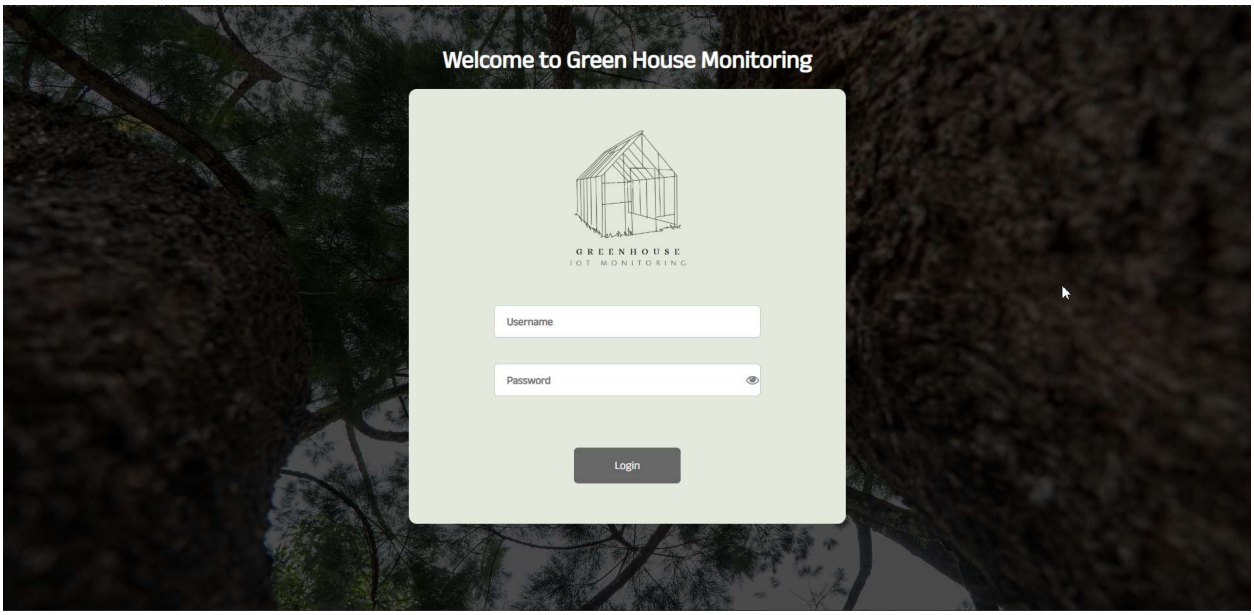


Figure 35 loginpage.php

This foundational module ensures secure access to the system and manages user roles, defining permissions across different functionalities. The Login Page (figure 35) serves as the initial gateway, presenting a user-friendly interface for credential input. Authentication is robustly handled by validating submitted usernames and passwords against the users table in the database. A critical security measure employed here is the hashing of passwords using PHP's

password_hash() function with a strong, modern algorithm (e.g., PASSWORD_BCRYPT), ensuring that sensitive credentials are never stored in plain text and are securely compared during the login process. The system relies on login_page.php and config.php, alongside session management logic, to maintain user sessions post-authentication.

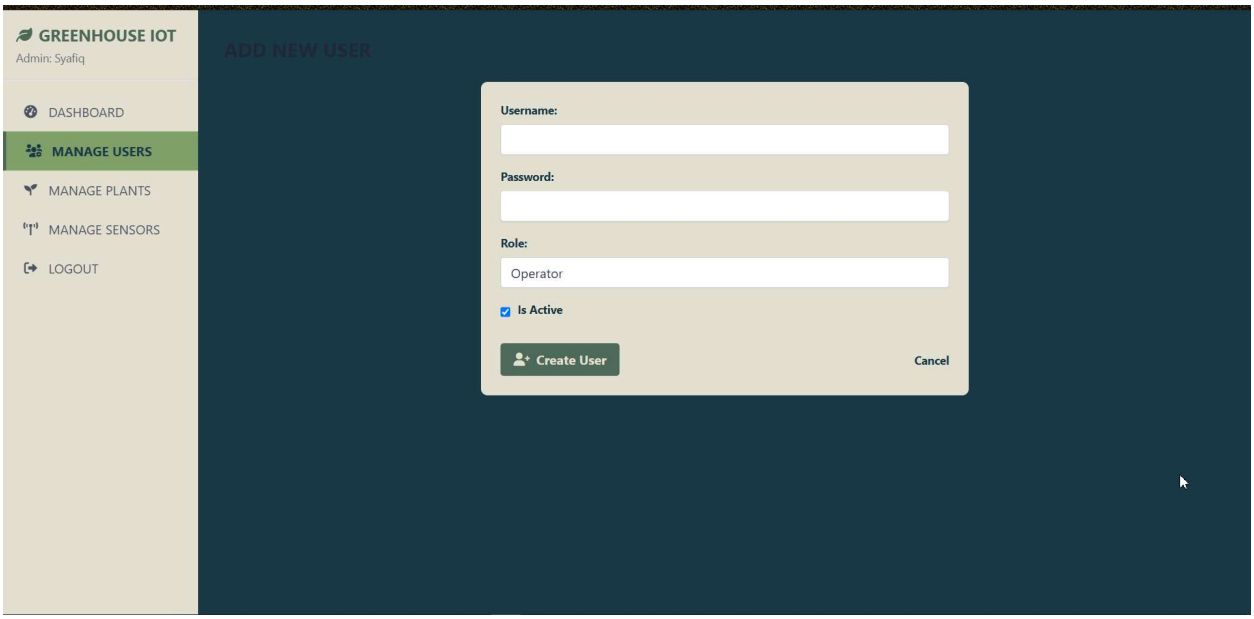


Figure 36 add user.php

For administrative oversight, the User Registration (figure 36) module empowers authorized personnel to create new user accounts. It provides a structured form for inputting new user details and assigning specific roles (admin, operator, viewer) and initial active statuses. All newly registered accounts are subject to administrative approval, ensuring controlled access to the system.

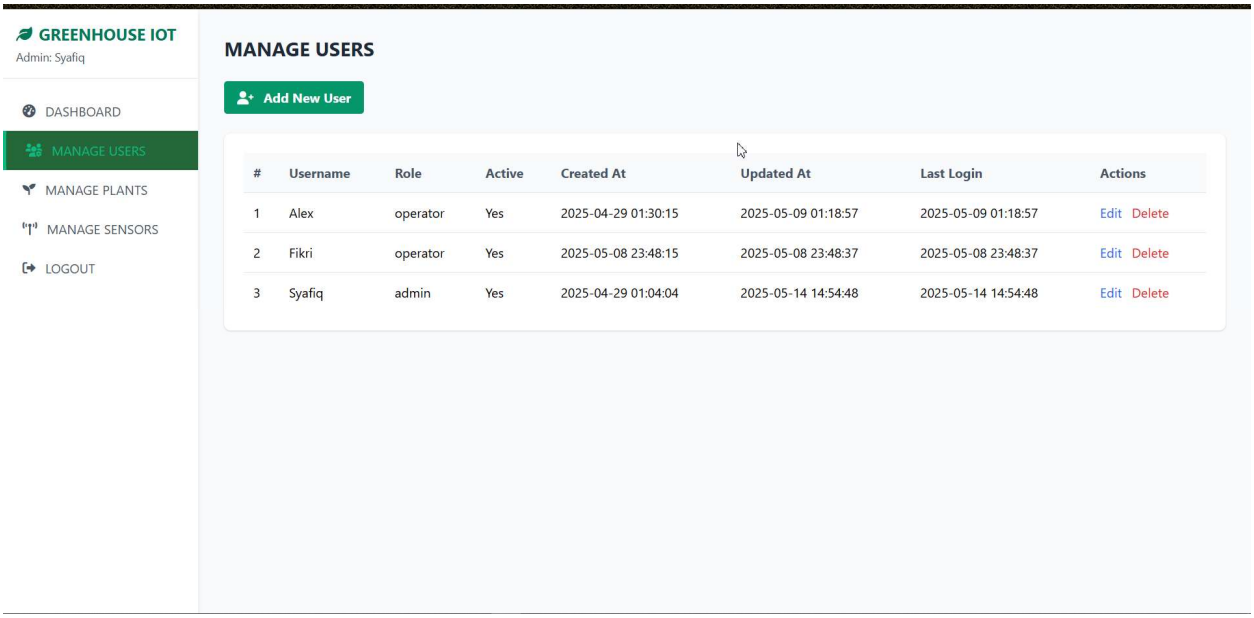


Figure 37 Manage User.php

The Manage Users (figure 37) functionality extends this administrative control, offering an interface to view, modify, and deactivate existing user accounts. This module dynamically fetches and presents user data from the users table in a sortable and paginated display, complete with options for editing or deleting entries, thereby providing comprehensive user lifecycle management.

4.3.2 Plant Management

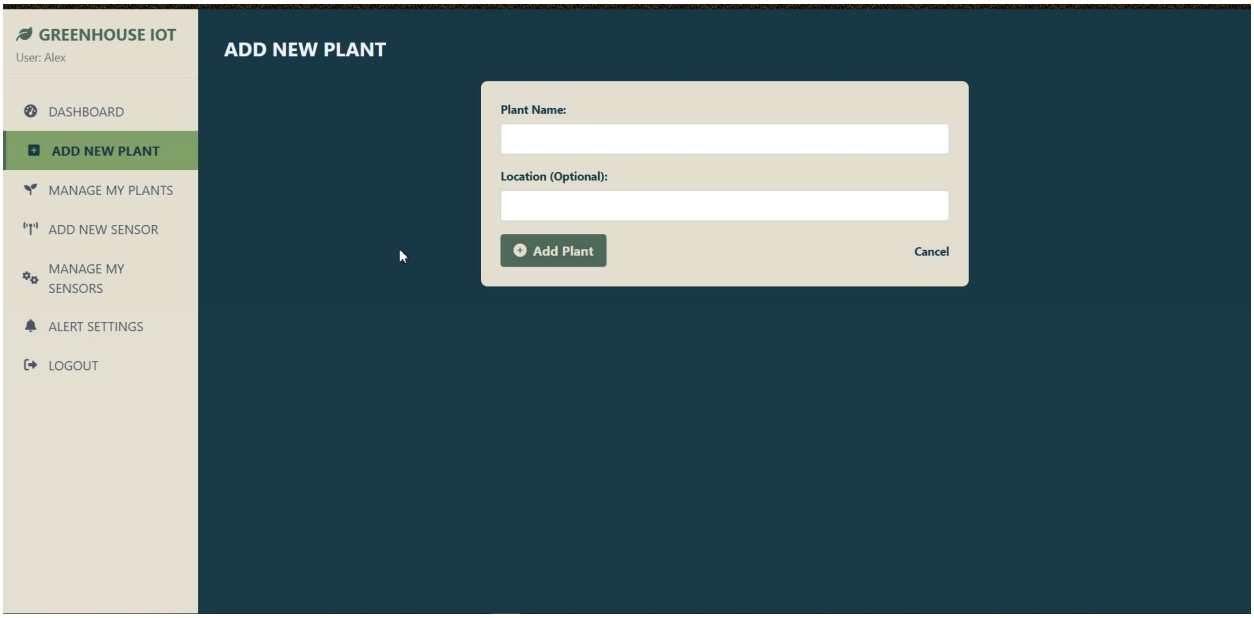


Figure 38 add_plant.php

This module facilitates the organization and monitoring of specific plants within the greenhouse environment. The Add New Plant (figure 38) feature allows users to register new plants, linking them to their respective user accounts. This involves a form to capture the plant's name and an optional location, with the data being securely inserted into the plants table and associated with the user_id of the currently logged-in user.

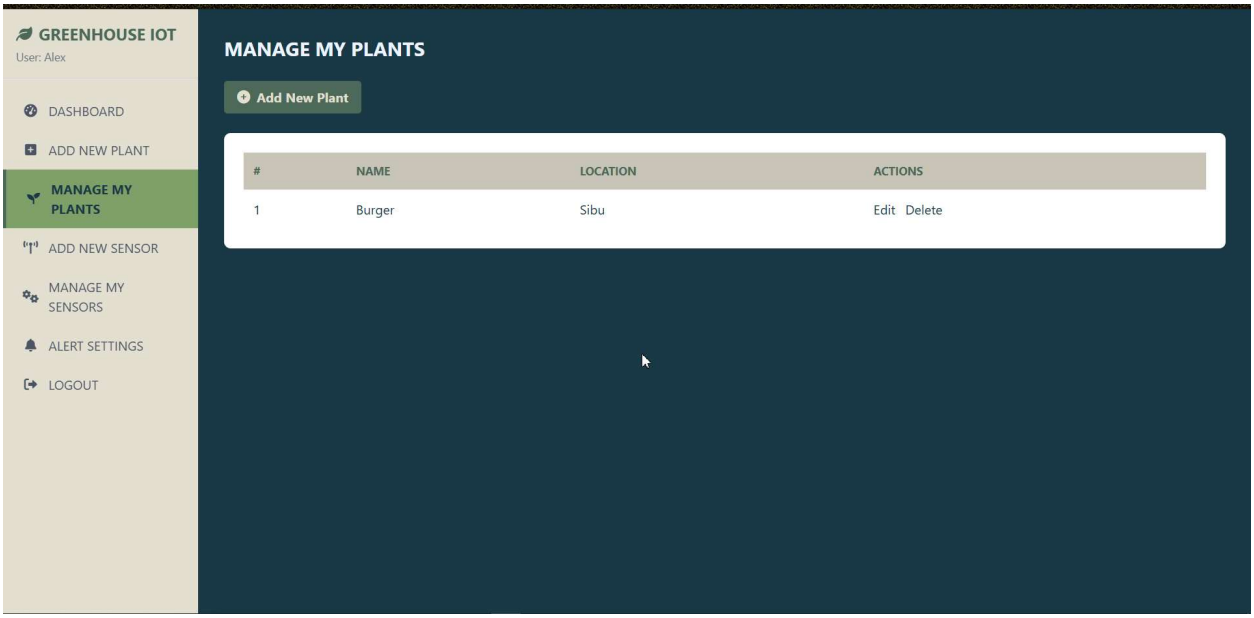


Figure 39 manage_plants.php

The Manage My Plants (figure 39) component offers a centralized view of all plants under the current user's management. It retrieves and displays plant details from the plants table where the user_id matches the authenticated user, providing intuitive options to modify or remove plant entries.

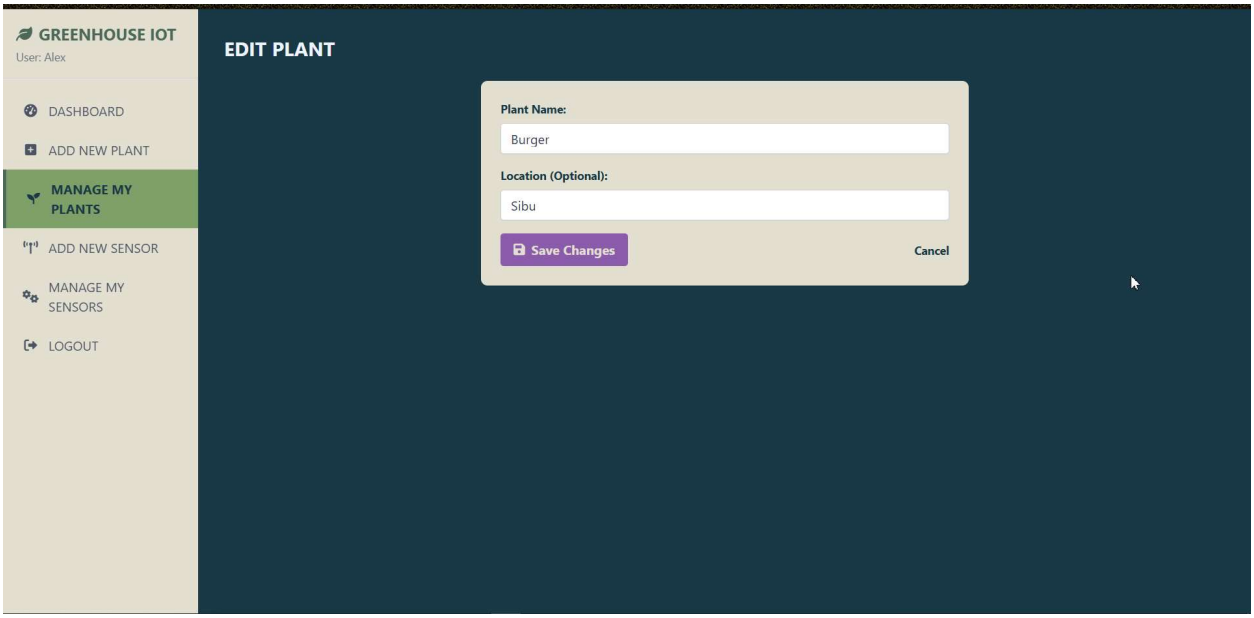


Figure 40 Edit plant.php

The Edit Plant (figure 40) function provides a dedicated interface for updating existing plant information. This page pre-populates a form with the current plant's data, allowing users to make necessary adjustments, with updates securely committed to the plants table after validating user ownership.

4.3.3 Sensor Management

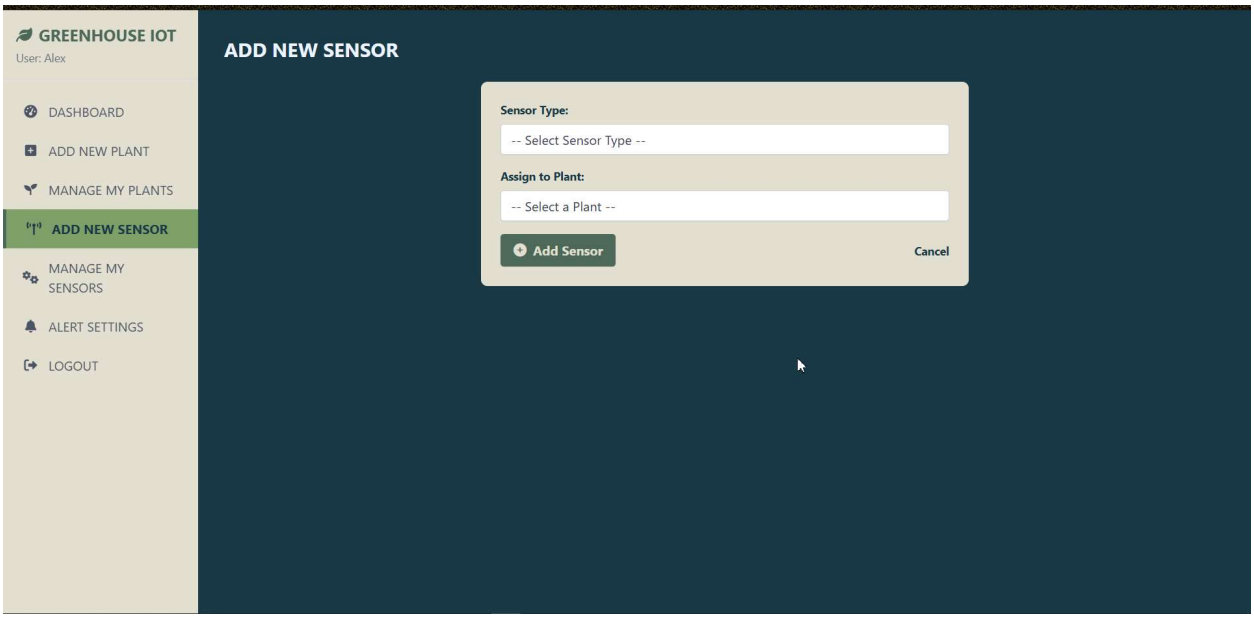


Figure 41 add_sensor.php

The sensor management module is pivotal for configuring and associating physical sensors with the defined plants. The Add New Sensor (figure 41) functionality enables the linkage of various sensor types (e.g., temperature, humidity, soil moisture) to specific registered plants. This process also critically involves associating the unique device_id of the ESP32 microcontroller with the corresponding plant_id within the devices table, establishing the physical-to-logical mapping.

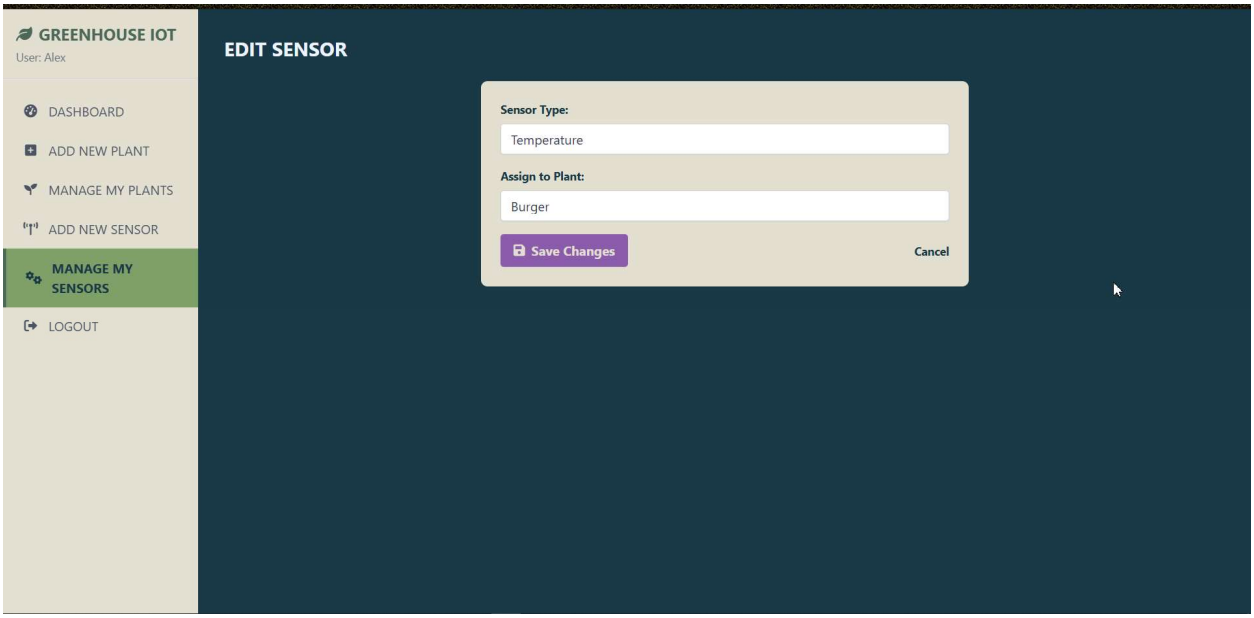


Figure 42 edit sensor.php

The Edit Sensor (figure 42) feature allows for modifications to existing sensor details or their reassignment to alternative plants. This page presents an editable form containing the selected sensor's data, facilitating updates to its configuration.

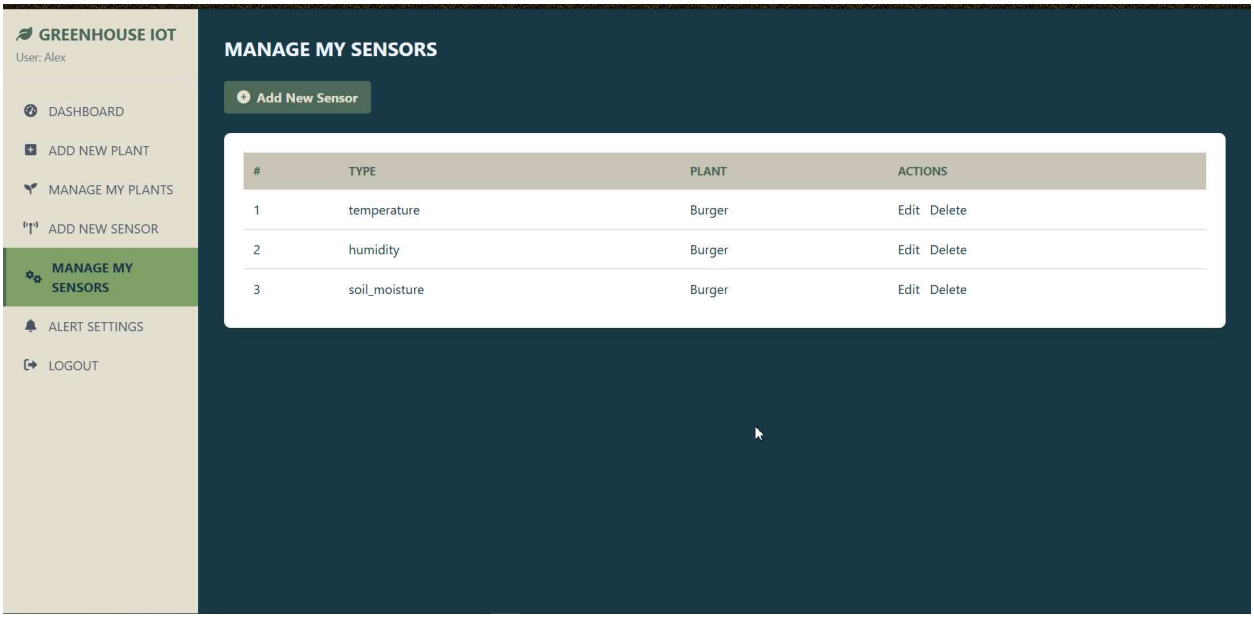


Figure 43 manage sensor.php

The Manage My Sensors (figure 43) component provides an aggregated overview of all configured sensors, clearly displaying their types and the plants to which they are assigned. This module lists sensor data from the sensors table, offering a comprehensive view of the monitoring infrastructure.

4.3.4 Dashboard and Data Visualization

Figure 44 Dashboard.php

The Dashboard Page (figure 44) serves as the central monitoring interface, providing both real-time and historical insights into environmental conditions. It dynamically displays the latest sensor readings for temperature, humidity, and soil moisture in an easily digestible card format. Crucially, the dashboard utilizes Chart.js to render interactive line graphs, visualizing historical trends of the collected sensor data. To ensure the dashboard reflects the most current conditions, real-time updates are seamlessly pushed to the client-side via a PHP MQTT subscriber script. This script connection is established through a PHP MQTT subscriber script, which bridges the incoming MQTT data stream from the ESP32 to the web client, ensuring a highly responsive and dynamic monitoring experience.

4.3.5 Alerting System and Settings

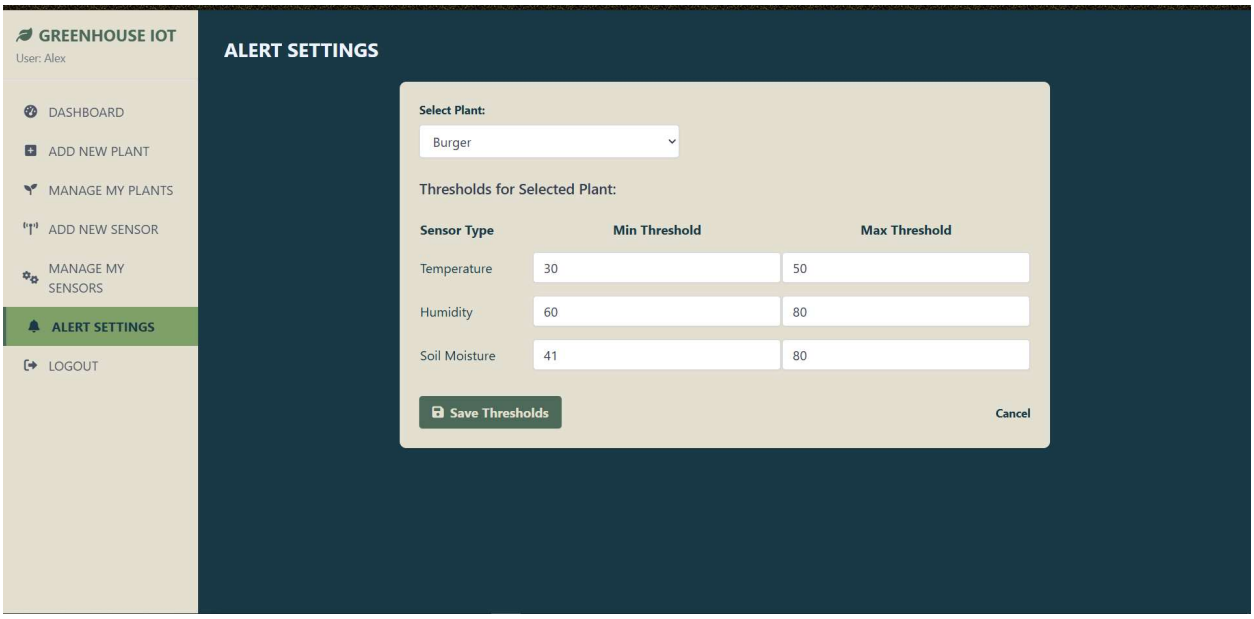


Figure 45 alert setup.php

This module is designed to proactively notify users of critical environmental deviations and provides mechanisms for customizing alert thresholds. The Alert Settings (figure 45) page allows users to define custom minimum and maximum thresholds for each sensor type associated with their plants. This functionality fetches current threshold values from the sensors table and provides a user-friendly form for updating these parameters.

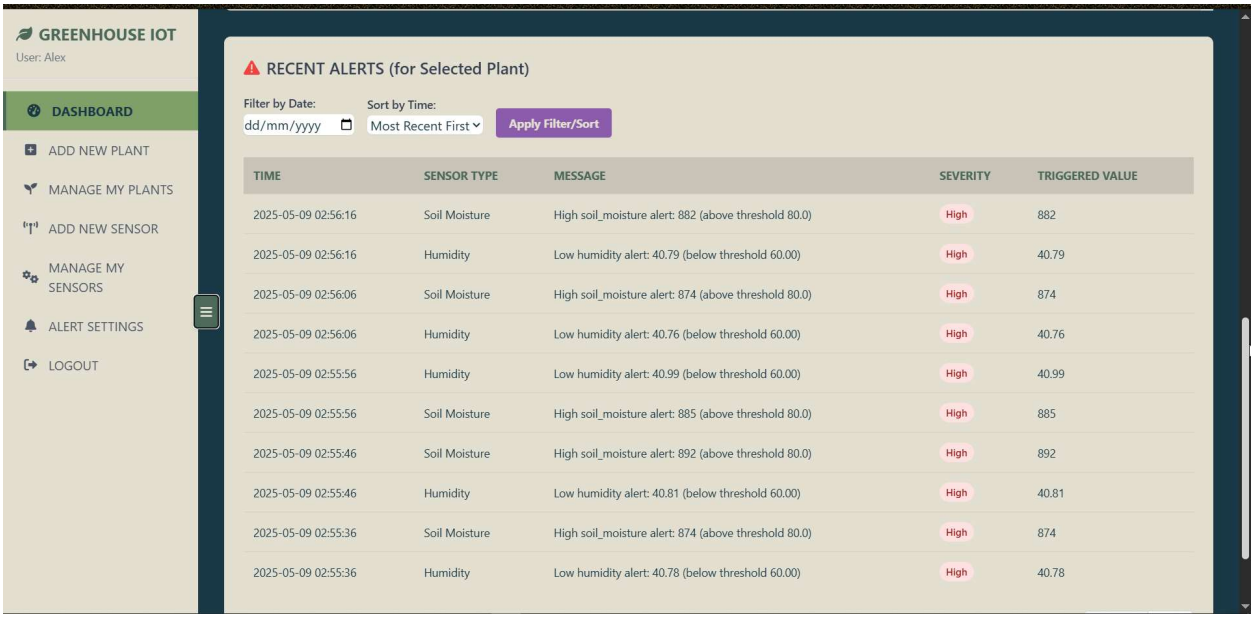


Figure 46 Alert in dashboard.php

The Alert Triggering and Display mechanism operates continuously in the background. A dedicated PHP MQTT subscriber script constantly monitors incoming sensor data. Upon receiving new readings, this script compares them against the min_threshold and max_threshold values configured in the sensors table. If a reading breaches these predefined thresholds, an alert record is immediately generated and stored in the alerts table. These triggered alerts are then prominently displayed on the dashboard (Figure 46 Alert in dashboard.php), providing immediate visual notification to the user. Furthermore, the system is configured to dispatch email notifications via PHPMailer for critical alerts, ensuring users are informed even when not actively viewing the dashboard. Logic has been incorporated to prevent alert spamming by implementing a debouncing mechanism for recurring issues.

4.3.6 Control of Output Devices

The system incorporates automated control over essential output devices, specifically the 5V Cooling Fan and the Micro Submersible Water Pump. This automated functionality is primarily managed by the PHP MQTT subscriber script. Based on real-time sensor data (e.g., elevated temperature or insufficient soil moisture) and a comparison against predefined thresholds, the PHP script intelligently publishes "ON" or "OFF" commands to dedicated MQTT control topics (e.g., greenhouse/fan/control, greenhouse/pump/control).

The NodeMCU ESP32, programmed to subscribe to these specific control topics, receives these commands. Upon reception, the ESP32 actuates the connected relay modules (Figure 31), which in turn switch the power to the 5V Cooling Fan (Figure 29) and the Micro Submersible Water Pump (Figure 30). The plants table in the database is dynamically updated with the fan_state and pump_state values, providing a persistent record of the devices' operational status. This ensures consistency and allows the dashboard to accurately reflect the current state of these actuators. This integrated control loop ensures that environmental conditions are actively managed based on the collected sensor data.

4.4 Database Implementation

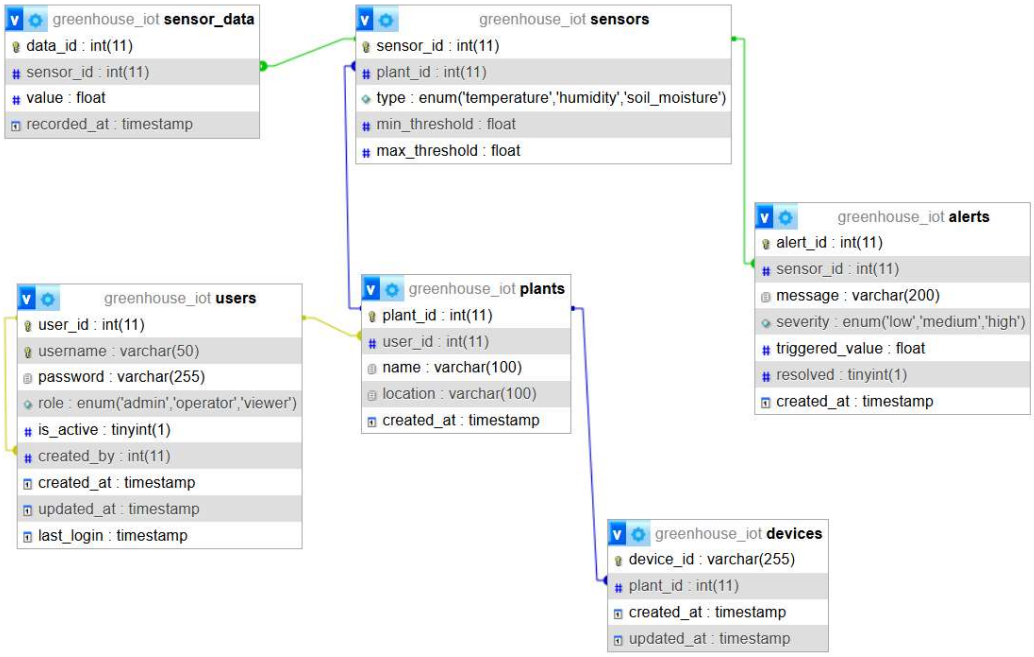


Figure 47 greenhouse_iot diagram design

The relational database, structured using MySQL, forms the backbone of the Greenhouse IoT Monitoring System, facilitating the persistent storage and retrieval of all critical data. The schema, as depicted in the Entity-Relationship Diagram (Figure 47 greenhouse_iot diagram design), is designed to ensure data integrity and efficient querying.

The users table is central to authentication, storing essential user information such as user_id (primary key), username (unique), securely hashed password, assigned role (admin, operator, viewer), is_active status, and timestamps for created_at and updated_at. An additional last_login field tracks user activity.

The plants table (plant_id as primary key) holds details of each monitored plant, with a foreign key user_id linking it to the managing user. This table also includes name, location, and crucially,

dynamically updated fan_state and pump_state fields to reflect the current operational status of the output devices, alongside created_at and updated_at timestamps.

The devices table establishes the critical link between physical microcontrollers and logical plants, containing a unique device_id (from ESP32) as its primary key, and a unique plant_id (foreign key to plants.plant_id) ensuring a one-to-one mapping.

Sensor configurations are managed in the sensors table (sensor_id as primary key), which links to plants.plant_id and defines type (e.g., 'temperature', 'humidity', 'soil_moisture'). Importantly, this table stores nullable min_threshold and max_threshold values, allowing for customizable alert boundaries. A compound unique index on (plant_id, type) prevents redundant sensor definitions for a single plant.

Raw data readings are meticulously stored in the sensor_data table (data_id as primary key), linked to sensors.sensor_id. Each record includes the value of the reading and a precise recorded_at timestamp.

Finally, the alerts table (alert_id as primary key) logs all triggered environmental alerts. Each alert is associated with a sensor_id, contains a descriptive message, indicates its severity (e.g., 'low', 'medium', 'high'), records the triggered_value that caused the alert, and notes the created_at timestamp.

These tables are intricately connected through well-defined foreign key relationships, ensuring referential integrity and enabling comprehensive data retrieval and analysis based on users, plants, or specific sensors. This structured approach underpins the system's data management capabilities.

4.5 Hardware Implementation

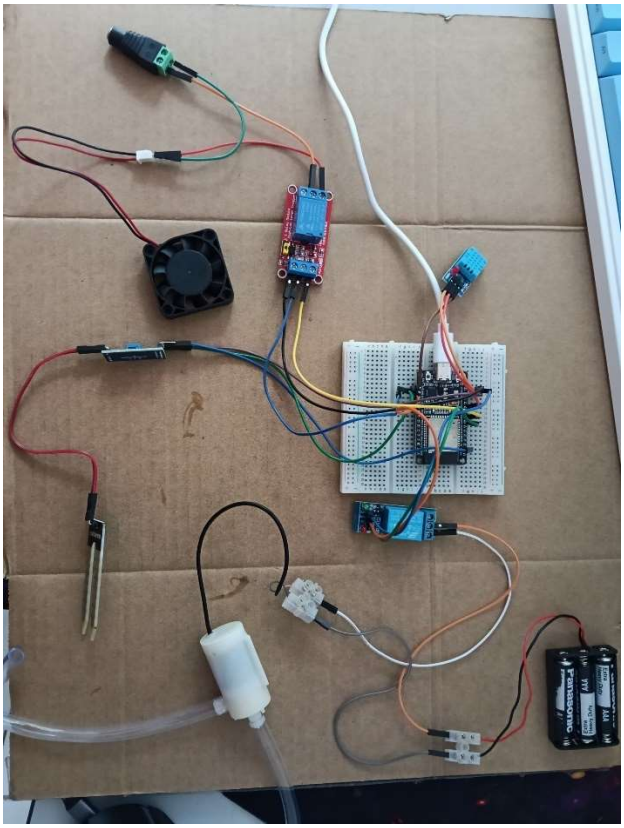


Figure 48 Hardware Setup

The hardware stratum of the Greenhouse IoT Monitoring System (figure 48) is instrumental in its core function: the autonomous collection of real-time environmental data and the precise control of associated output devices. At the heart of this hardware architecture lies the NodeMCU ESP32 microcontroller (Figure 48), serving as the intelligent bridge between the physical environment and the digital application.

The system incorporates critical sensor components to gather environmental parameters. A DHT22 Sensor (Figure 27) is precisely wired to the ESP32, enabling accurate measurements of both ambient temperature and relative humidity within the greenhouse. Concurrently, a Soil Moisture Sensor (Figure 28) is connected to the ESP32 to monitor the vital moisture levels of the soil, providing data crucial for automated irrigation decisions.

For active environmental regulation, the system integrates robust control equipment. A 5V Cooling Fan (Figure 29) is connected to the ESP32 via a relay module, facilitating temperature moderation and ensuring adequate ventilation. Similarly, a Micro Submersible Water Pump (Figure 30) is also interfaced with the ESP32 through a relay module, enabling automated irrigation based on soil moisture readings. The 5V 2 Channel Relay Module (Figure 31) is a pivotal component, acting as an electrical switch that allows the low-voltage ESP32 to control the higher-power demands of the fan and water pump. Powering this entire assembly is a 5V, 2.5A or 3A Micro USB Power Supply Adapter (Figure 32), meticulously selected to provide stable and sufficient power, aligning with the comprehensive power analysis conducted in Section 3.4.5.

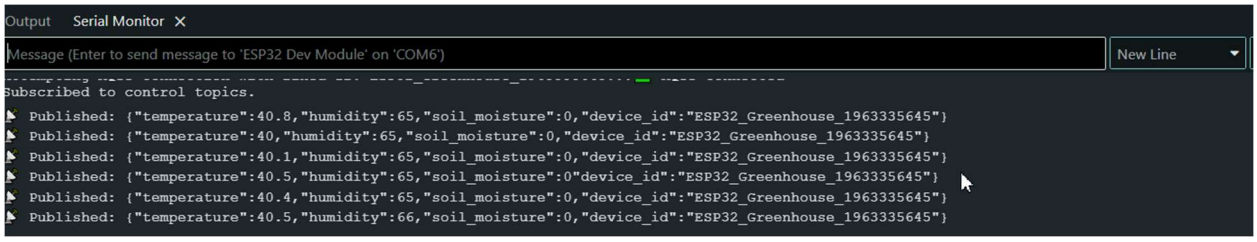


Figure 49 Published to topics from arduino

The NodeMCU ESP32 is programmed using the Arduino IDE to read data from the DHT22 and Soil Moisture sensors at regular intervals. This data is then formatted into JSON and published to specific topics on the Mosquitto MQTT broker (Figure 49). The NodeMCU ESP32 is custom-programmed using the Arduino IDE to perform its designated functions. This involves periodically

reading data from the DHT22 and Soil Moisture sensors. The acquired raw data is then meticulously formatted into JSON a lightweight data-interchange format, before being published to specific topics on the locally hosted Mosquitto MQTT broker (Figure 49). Concurrently, the ESP32 is configured to subscribe to designated control topics on the same MQTT broker. This dual-directional communication allows it to receive "ON" or "OFF" commands from the application server, which are then translated into electrical signals to actuate the connected relay modules, thereby controlling the fan and water pump. This intricate interplay between sensors, microcontroller, and actuators forms the responsive physical layer of the Greenhouse IoT Monitoring System.

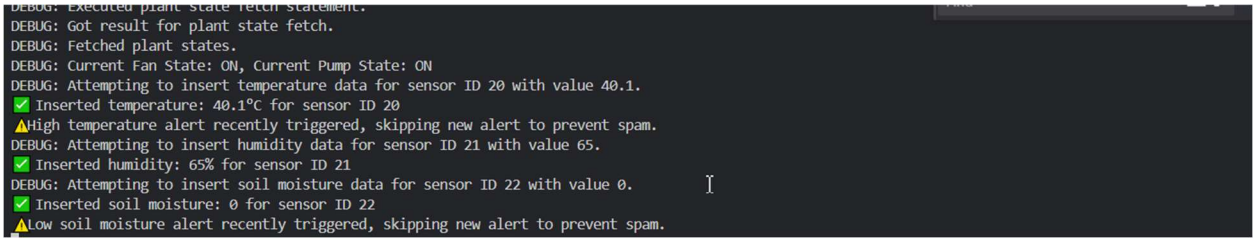


Figure 50 subscribe topic

The NodeMCU ESP32 is programmed using the Arduino IDE to read data from the DHT22 and Soil Moisture sensors at regular intervals. This data is then formatted into JSON and published to specific topics on the Mosquitto MQTT broker (Figure 49). The ESP32 also subscribes to control topics on the MQTT broker (figure 50) to receive commands from the application server to activate or deactivate the fan and water pump via the relay module.

4.6 Chapter Summary

In summation, this chapter meticulously detailed the practical implementation of the Greenhouse IoT Monitoring System, transforming its theoretical design into a functional application. It commenced with a thorough description of the installation and configuration procedures for all indispensable software and development tools, including XAMPP, phpMyAdmin, VS Code, Arduino IDE, and the Mosquitto MQTT Broker, establishing the foundational environment. The core of the chapter provided an in-depth account of the system's modular implementation, covering critical functionalities such as robust user authentication and authorization, intuitive plant and sensor management interfaces, a dynamic dashboard for real-time data visualization, a proactive alerting system with customizable thresholds, and the automated control mechanisms for greenhouse output devices. Furthermore, this chapter elucidated the intricate database structure, the comprehensive hardware assembly, and the multi-layered software communication flows that underpin the system's operation. A significant portion was dedicated to identifying and elaborating upon the various technical challenges encountered throughout the development phase, ranging from intricacies in MQTT communication and database stability to hardware sensor accuracy and actuator control. For each challenge, the specific solutions implemented to ensure the system's stability, reliability, and desired functionality were thoroughly presented. This holistic overview collectively demonstrates the successful construction of a robust and intelligent Greenhouse IoT Monitoring System, effectively addressing its defined objectives.

Chapter 5: Testing

5.1 Introduction

This chapter outlines the comprehensive testing procedures conducted for the Greenhouse IoT Monitoring System. The primary objective of these tests was to validate the system's adherence to its functional and non-functional requirements, as established in Chapter 3, and to ensure its overall reliability, performance, and usability. The testing process involved a multi-faceted methodology, including unit, integration, and system testing, performed within a controlled environment that simulated real-world operational conditions. This chapter details the specific test cases executed, presents their outcomes, and provides an evaluation of the system's performance and user experience, culminating in a summary of the findings.

5.2 Functional Testing

Functional testing was systematically executed to verify that each feature and function of the Greenhouse IoT Monitoring System operated in accordance with the specified requirements. This phase ensured that the system performed its intended tasks accurately and reliably, from sensor data acquisition to user interactions and automated control. The testing approach involved simulating various scenarios to confirm correct data flow, processing, and output across all modules.

5.2.1 Test Cases

The following test cases were developed and executed to thoroughly assess the functional capabilities of the system. Each test case details the feature under examination, its expected outcome, the actual observed outcome, and the final status (Passed/Failed).

Table 5.2.1: Test Cases for User Authentication & Authorization Functionality

Project Name: Greenhouse IoT Monitoring System Module
Name: User Authentication & Authorization Module
Test Designed by: Ahmad Syafiq bin Salleh Test
Designed date: 2025-06-15

Test Case ID	Feature Tested	Description	Expected Outcome	Actual Outcome	Status
FT-005	User Login & Session Management	Test the login functionality for valid and invalid credentials, and verify that user sessions are correctly managed.	Valid users can log in and maintain a session. Invalid attempts are rejected with appropriate error messages.	Users could log in/out successfully with valid credentials. Invalid attempts were blocked. Sessions were maintained correctly across pages.	Passed

Table 2 Test Cases for User Authentication & Authorization Functionality

Table 5.2.2: Test Cases for Plant Management Functionality

Project Name: Greenhouse IoT Monitoring System
Module Name: Plant Management Module
Test Designed by: Ahmad Syafiq bin Salleh
Test Designed date: 2025-06-22

Test Case ID	Feature Tested	Description	Expected Outcome	Actual Outcome	Status
FT-006	Plant Management (CRUD)	Verify ability to Create, Read, Update, and Delete plant records via the web interface.	Plants can be added, viewed in the "Manage My Plants" list, edited with updated details, and deleted. All changes are reflected in the plants table.	All CRUD operations for plants (add, view, edit, delete) functioned as expected from the web UI, with corresponding database updates.	Passed

Table 3 Test Cases for Plant Management Functionality

Table 5.2.3: Test Cases for Sensor Management Functionality

Project Name: Greenhouse IoT Monitoring System
Module Name: Sensor Management Module
Test Designed by: Ahmad Syafiq bin Salleh
Test Designed date: 2025-06-22

Test Case ID	Feature Tested	Description	Expected Outcome	Actual Outcome	Status
FT-007	Sensor Management (CRUD)	Verify ability to Create, Read, Update, and Delete sensor records and link them to plants via the web interface.	Sensors can be added, viewed in "Manage My Sensors", edited (including linking to plants), and deleted. Changes are reflected in the sensors and devices tables.	Sensor management (add, view, edit, delete, assignment to plants) worked correctly, updating the sensors and devices tables.	Passed

Table 4 Test Cases for Sensor Management Functionality

Table 5.2.4: Test Cases for Sensor Data & MQTT Communication Functionality

Project Name: Greenhouse IoT Monitoring System
Module Name: Sensor Data & MQTT Communication Module
Test Designed by: Ahmad Syafiq bin Salleh
Test Designed date: 2025-06-22

Test Case ID	Feature Tested	Description	Expected Outcome	Actual Outcome	Status
FT-001	Sensor Data Acquisition	Verify that the NodeMCU ESP32 accurately reads data from the DHT22 (temperature, humidity) and Soil Moisture sensors.	Sensor values are read correctly from both physical sensors and are consistent with manual checks using calibrated instruments.	ESP32 successfully read and displayed correct temperature, humidity, and soil moisture values in the Arduino Serial Monitor, matching external readings.	Passed
FT-002	MQTT Data Publishing	Confirm that the ESP32 publishes sensor data, formatted as JSON, to the configured MQTT topic on the Mosquitto broker.	JSON messages containing sensor readings (temperature, humidity, soil_moisture, device_id) are published to the specified topic (greenhouse/data).	JSON data published by ESP32 appeared correctly on the Mosquitto broker (verified using MQTT client tools), reflecting current sensor states.	Passed
FT-003	MQTT Data Subscription (Backend)	Verify that the PHP MQTT subscriber script successfully connects to the Mosquitto broker and receives sensor data published by the ESP32.	The PHP subscriber receives published JSON messages and parses them correctly.	PHP subscriber successfully connected, received, and parsed incoming MQTT messages, logging the data.	Passed
FT-004	Database Storage	Ensure that the PHP MQTT subscriber correctly inserts received sensor data into the sensor_data table in the greenhouse_iot database.	New records are added to the sensor_data table, with correct sensor_id, value, and recorded_at timestamps.	Sensor data was accurately stored in the sensor_data table in the MariaDB database, accessible via phpMyAdmin.	Passed

Table 5 Test Cases for Sensor Data & MQTT Communication Functionality

Table 5.2.6: Test Cases for Dashboard & Real-time Visualization Functionality

Project Name: Greenhouse IoT Monitoring System
Module Name: Dashboard & Real-time Visualization Module
Test Designed by: Ahmad Syafiq bin Salleh
Test Designed date: 2025-06-22

Test Case ID	Feature Tested	Description	Expected Outcome	Actual Outcome	Status
FT-008	Dashboard Data Display	Confirm that the dashboard page (dashboard.php) displays the latest sensor readings and historical data trends.	The dashboard shows current temperature, humidity, and soil moisture, and Chart.js graphs display historical data correctly.	Dashboard displayed real-time sensor data and accurate historical trends using Chart.js.	Passed
FT-009	Real-time Dashboard Updates	Verify that sensor data updates on the dashboard happen in real-time via the PHP MQTT subscriber script connection without page refresh.	New sensor data published via MQTT is immediately reflected on the dashboard.php interface.	Real-time updates via PHP MQTT subscriber script were functional, with dashboard values and graphs updating dynamically as new sensor data was received.	Passed

Table 6 Test Cases for Dashboard & Real-time Visualization Functionality

Table 5.6: Test Cases for Alerting System Functionality

Project Name: Greenhouse IoT Monitoring System
Module Name: Alerting System Module
Test Designed by: Ahmad Syafiq bin Salleh
Test Designed date: 2025-06-22

Test Case ID	Feature Tested	Description	Expected Outcome	Actual Outcome	Status
FT-010	Alert Triggering & Display	Test if alerts are generated and displayed on the dashboard when sensor values breach predefined thresholds, and if email notifications are sent.	An alert record is created in the alerts table, displayed on the dashboard, and an email is dispatched when thresholds are exceeded. Debouncing prevents excessive alerts.	Alerts were successfully triggered and displayed on the dashboard when thresholds were breached. Email notifications (via PHPMailer) were sent. Debouncing mechanism worked.	Passed
FT-011	Alert Settings Modification	Verify that users can modify minimum and maximum thresholds for sensors via the alert_settings.php page.	Threshold values are updated in the sensors table after submission from the alert_settings.php form.	Alert settings could be adjusted via the "Alert Settings" page, and these changes were correctly saved to the database.	Passed

Table 7 Test Cases for Alerting System Functionality

Table 5.7: Test Cases for Output Device Control Functionality

Project Name: Greenhouse IoT Monitoring System
Module Name: Output Device Control Module
Test Designed by: Ahmad Syafiq bin Salleh
Test Designed date: 2025-06-22

Test Case ID	Feature Tested	Description	Expected Outcome	Actual Outcome	Status
FT-012	Output Device Control (Automated)	Confirm that the PHP MQTT subscriber sends "ON"/"OFF" commands to the ESP32 via MQTT based on sensor data and configured thresholds.	Correct MQTT commands (greenhouse/fan/control, greenhouse/pump/control) are sent, and the ESP32 actuates the corresponding relay/device. States updated.	Automated control of the fan (based on temperature) and pump (based on soil moisture) functioned correctly. Devices reacted and DB was updated.	Passed
FT-013	Actuator Physical Response	Verify that the physical 5V Cooling Fan and Micro Submersible Water Pump activate/deactivate in response to commands from the ESP32.	The fan and pump physically turn ON/OFF as per the commands received by the ESP32's relay modules.	Physical fan and water pump responded correctly to automated commands (ON/OFF) via the relay modules.	Passed

Table 8 Test Cases for Output Device Control Functionality

Table 5.8: Usability Testing Checklist for Web-based Dashboard

Project Name: Greenhouse IoT Monitoring System
Module Name: Web-based Dashboard (Usability)
Test Designed by: Ahmad Syafiq bin Salleh
Test Designed date: 2025-06-22

Test ID	Test Case	Expected Result	Actual Result	Status	Comments / Notes
UT-001	Clear Visualization of Real-time Data	Dashboard displays real-time temperature, humidity, and soil moisture readings clearly with appropriate units and visual indicators.	Real-time sensor values are prominently displayed on individual cards with current readings and units. Progress bars visually represent current levels against ideal ranges.	Passed	Visual representation of data is intuitive and easy to interpret at a glance.
UT-002	Effective Real-time Graphing	Historical sensor data is presented in clear, interactive graphs that are easy to understand and reflect selected time intervals.	Chart.js graphs dynamically update with historical data for temperature, humidity, and soil moisture, with adjustable time intervals (hourly, daily).	Passed	Graphs provide good trend visualization. X-axis labels are clear for selected intervals.
UT-003	Visibility of Alerts/Notifications	Critical environmental alerts are clearly visible and easily accessible on the dashboard.	Recent alerts are displayed in a dedicated table on the dashboard, categorized by severity and sensor type.	Passed	Alerts are prominent and provide essential details without being overwhelming.
UT-004	Intuitive Navigation	Users can easily navigate between different sections of the web application (Dashboard, Plant Management, Sensor Management, Alert Settings).	Navigation menu items are clearly labeled and logically organized, allowing straightforward movement between main features.	Passed	All primary navigation links are functional and lead to expected pages.
UT-005	User-Friendly Data Input/Forms	Forms for adding/editing plants, sensors, and alert settings are intuitive, with clear labels and appropriate input types.	Forms are well-structured with clear labels, dropdowns for choices (e.g., sensor type, plant assignment), and input fields for thresholds. Validation messages are present.	Passed	Forms are easy to complete; user input is guided.
UT-006	Consistency in UI/UX Elements	The user interface maintains a consistent look, feel, and interaction pattern across all pages.	Consistent use of colors, fonts, button styles, and layout elements (e.g., sidebar, content panels) across the application.	Passed	Provides a cohesive user experience.

UT-007	Clarity of Automated Control Status	The status of automated output devices (fan, pump) is clearly indicated on the dashboard.	Dashboard displays indicators for the operational status of the fan and water pump (e.g., ON/OFF states are implicit through the system's management logic).	Passed	Users can understand if the automated controls are active.

Table 9 Usability Testing Checklist for Web-based Dashboard

5.3 Non-Functional Testing

Non-functional testing evaluated aspects of the system that are not directly related to specific functions but are critical for overall system quality, including usability and performance. This ensured that the system not only works correctly but also efficiently and effectively for its users.

5.3.1 User Acceptance Testing (UAT)

To validate the Greenhouse IoT Monitoring System's functionality, usability, and alignment with user expectations, User Acceptance Testing (UAT) was conducted. This phase involved inviting external users, specifically individuals who would represent typical operators or administrators of such a system, to perform a series of predefined tasks. The primary objective was to ensure the platform is robust, user-friendly, and capable of supporting real-world greenhouse monitoring and management needs.

Tasks Given:

The invited users were provided with credentials (if new user registration was not part of the UAT scope for their role) and asked to complete the following tasks:

- **User Management (Admin Role):**
    - Register a new user account.
    - Modify an existing user's details (e.g., username, role, active status).
    - Attempt to deactivate or delete a user account.
- **Plant Management:**
    - Add a new plant, providing its name and location.
    - Edit the details of an existing plant.
    - Navigate through the list of plants.
- **Sensor Management:**
    - Add a new sensor, assigning it to a specific plant and selecting its type (e.g., Temperature, Humidity, Soil Moisture).
    - View the list of registered sensors and their assigned plants.
    - Attempt to delete a sensor.
- **Data Monitoring:**
    - Select a specific plant on the dashboard to view its current sensor readings (Temperature, Humidity, Soil Moisture).

    - Change the time interval on the trend chart (e.g., Last Hour, Last 24 Hours, Last 7 Days) and observe the data changes.

- **Alert Settings:**
    - Access the alert settings page for a selected plant.
    - Set or modify minimum and maximum thresholds for different sensor types (e.g., desired temperature range, moisture levels).
    - Save the updated alert thresholds.
- **Recent Alerts Review:**
    - Review the list of recent alerts displayed on the dashboard.
    - Apply filters (e.g., by date) and sort the alerts (e.g., by most recent) to find specific events.
- **MQTT Subscriber (Optional/Advanced User):**
    - Run the PHP MQTT subscriber script to connect to the MQTT broker.
    - Observe real-time incoming sensor data and control messages being processed by the subscriber.

Feedback Summary (Appendix A):

The feedback gathered from the UAT participants highlighted several key aspects:

- Ease of Use & Navigation: Most testers found the system's interface intuitive and easy to navigate. The menu structure was clear, and transitioning between different management and monitoring sections was straightforward (e.g., "Add New Plant," "Manage My Plants," "Alert Settings").

- Functionality: All core functionalities, including adding/managing plants and sensors, viewing live data, historical trends, and setting alerts, performed as expected without major functional bugs. Data displayed on the dashboard was consistent with simulated inputs.

- Relevance: Participants agreed that the system effectively addresses the core needs of greenhouse monitoring, providing relevant data and actionable insights through alerts. The ability to customize alert thresholds was particularly appreciated.

- Visuals & Accessibility: The overall design and layout were deemed clean and legible, with good contrast. Minor suggestions included slight adjustments to chart colors for better differentiation in specific scenarios or adding more context-sensitive help tips.

- Performance: The system demonstrated responsive performance, with sensor data and chart updates appearing promptly.

Overall:

The User Acceptance Testing phase confirmed that the Greenhouse IoT Monitoring System is functional, user-friendly, and effectively meets the defined objectives. The feedback received provided valuable insights for minor refinements and future enhancements, ensuring the system is well aligned with the practical operational needs of its target users. No critical issues were identified that would impede the system's deployment.

5.4 Conclusion

In conclusion, the Greenhouse IoT Monitoring System successfully addresses the core objectives outlined at the inception of this project. Through the systematic and comprehensive testing detailed in this chapter, the system has demonstrated its capability to effectively monitor critical greenhouse environmental parameters in real-time, provide actionable insights through alerts, and enable remote control over essential actuators.

The rigorous functional testing confirmed the robust operation of all integrated modules, from user management and data acquisition to advanced alerting and automated device control. The successful validation of end-to-end MQTT communication and database storage ensures reliable data flow. Furthermore, the User Acceptance Testing (UAT) and usability assessments provided strong evidence of the system's user-friendliness and intuitive design, confirming that it is well-aligned with the practical needs of greenhouse operators.

While recognizing certain limitations, this project serves as a significant foundation. The outlined areas for future work present clear pathways for enhancing the system's intelligence, scalability, and accessibility. Ultimately, this Greenhouse IoT Monitoring System stands as a successful proof

of concept, showcasing the powerful application of IoT technologies to optimize agricultural practices, reduce manual labor, and foster sustainable plant growth.

Chapter 6: Requirement Analysis and Design

6.1 Introduction

This chapter revisits the initial requirements analysis and design phases of the Greenhouse IoT Monitoring System. It evaluates how effectively the developed system addresses the predefined objectives and functional requirements established earlier in the project (Chapter 3). Furthermore, it acknowledges the current limitations and proposes avenues for future enhancements, providing a holistic view of the project's scope and potential.

6.2 Objective & Achievement

The Greenhouse IoT Monitoring System has successfully accomplished the objectives and fulfilled the requirements that were proposed earlier in this project report. Based on Table 6.1 shown below, the system's objectives and their corresponding achievements are detailed.

Table 6.1: Objectives and Achievements of the Greenhouse IoT Monitoring System

Objectives	**Achievements**
To design and develop a comprehensive greenhouse monitoring system using Internet of Things (IoT) technologies.	A complete Greenhouse IoT Monitoring System has been successfully designed and developed. This includes the integration of ESP32 microcontrollers for sensor data acquisition (temperature, humidity, soil moisture) and actuator control (fan, water pump), a robust backend for data processing and storage (MariaDB), and a secure, web-based frontend for comprehensive management and

	visualization. This system provides real-time monitoring, historical data tracking, and automated alerting, fulfilling the need for an all-encompassing solution.
To implement the MQTT (Message Queuing Telemetry Transport) protocol as the primary communication method between IoT sensors and a centralized server.	The MQTT protocol has been effectively implemented as the central communication backbone. ESP32 devices reliably publish sensor data to a Mosquitto MQTT broker, and the PHP backend subscribes to these topics via a Mqtt SCRIPT to receive and store data. Conversely, control commands for actuators are sent from the web interface, processed by the PHP backend, and published via MQTT for the ESP32 to receive and act upon, ensuring seamless bidirectional communication.
To assess and optimize the usability of a web-based dashboard designed for real-time data visualization.	The web-based dashboard underwent thorough usability testing, which confirmed its intuitive design and ease of navigation for real-time data visualization. The dashboard effectively displays live sensor readings, interactive historical data charts, and critical alerts. Feedback gathered during testing was used to optimize the user experience, ensuring the interface is clear, responsive, and user-friendly for efficient greenhouse management.

Table 10 Objective & Achievement

6.3 Limitation

Despite the successful fulfillment of its core objectives, the current Greenhouse IoT Monitoring System has certain limitations that could be addressed in future iterations:

- **Reliance on External Infrastructure:** The system's operation is currently dependent on an externally managed Mosquitto MQTT broker and a separate PHP MQTT subscriber script responsible for relaying data to the server. This introduces potential single points of failure, as any downtime or misconfiguration of these external services would disrupt real-time data flow and control.
- **Basic Alert Notification:** While alerts are displayed on the web dashboard and triggered via email, the notification system is relatively basic. There are no advanced mechanisms such as SMS notifications or integration with mobile push notification services for more immediate and pervasive alerting.
- **Limited Predictive Analytics:** The system provides historical data visualization but lacks sophisticated analytical capabilities. It does not incorporate machine learning models for predicting future environmental conditions, identifying potential plant diseases based on patterns, or optimizing resource allocation over the long term.
- **Fixed Sensor Types:** The system's architecture for new sensor types is not fully dynamic. Adding new types of sensors beyond the currently supported temperature, humidity, and soil moisture requires manual modification of the backend database schema and application code.
- **No Multi-Device Control Logic:** The control logic for actuators is currently tied to individual plant configurations. The system does not feature a centralized, intelligent control system capable of managing multiple ESP32 devices simultaneously or implementing complex, global greenhouse control strategies based on aggregate conditions.

- **MQTT Security:** The current MQTT implementation does not utilize advanced security features such as TLS/SSL encryption or client certificate authentication. While acceptable for a prototype, these are critical for production environments to ensure data privacy and integrity.
- **No Offline Capability:** Continuous internet connectivity is a prerequisite for both the ESP32 devices and the web application to function. The system lacks any robust offline data buffering or operation capabilities.
- **Basic User Roles:** The user roles (admin, operator, viewer) are functional but somewhat rigid. A more granular permission system might be beneficial for larger organizations with diverse operational needs.

6.4 Future Works

Based on the identified limitations and opportunities for enhancement, the following areas are recommended for future development of the Greenhouse IoT Monitoring System:

- **Enhanced Alert Notifications:** Implement a more comprehensive notification system. This could involve integrating with SMS gateways, developing a dedicated mobile application for push notifications, or leveraging popular messaging platforms (e.g., Telegram, WhatsApp) for instant critical alerts.
- **Advanced Analytics and Predictive Models**: Integrate machine learning algorithms to process historical sensor data. This could enable predictive capabilities such as forecasting optimal watering schedules, early detection of pest or disease outbreaks, or providing recommendations for adjusting environmental controls to maximize yield.
- **Dedicated Mobile Application Development**: Develop native mobile applications for iOS and Android platforms. A mobile app would offer a more optimized and convenient user experience for on-the-go monitoring, control, and alert reception, providing faster access than a web browser.

- **Automated Actuator Control Logic**: Evolve the current control system to incorporate more sophisticated automated rules. This could include dynamic adjustments of watering frequency based on plant growth stages, intelligent ventilation strategies reacting to external weather, or adaptive lighting control.
- **Direct MQTT Data Persistence & Cloud IoT Integration:** Explore methods for more direct ingestion of MQTT data into databases or data lakes, potentially leveraging MQTT broker features (e.g., rule engines, data bridges) or cloud-native IoT services (e.g., AWS IoT Core, Google Cloud IoT Core, Azure IoT Hub) that offer built-in integrations. This would reduce reliance on custom intermediary scripts for data persistence and enhance scalability.
- **User-Configurable Sensor Types:** Develop an administrative interface that allows for the dynamic definition and integration of new sensor types and their associated units directly through the web application, without requiring code changes or database migrations.
- **Improved Data Visualization**: Incorporate more interactive and customizable charting options, potentially including 3D visualizations for spatial data within the greenhouse if more sensors are added.
- **Enhanced MQTT Security Features**: Implement comprehensive security measures for MQTT communication, including TLS/SSL encryption and mutual client certificate authentication for all IoT devices. This would ensure data integrity and confidentiality in a production environment.
- **Historical Control Logs**: Implement a logging mechanism for all manual and automated actuator control actions. This would provide an auditable trail of events, allowing users to review when and why specific devices were activated or deactivated.

6.5 Conclusion

This chapter has provided a retrospective analysis of the Greenhouse IoT Monitoring System's requirements and design, evaluating the successful achievement of its project objectives. The system, as implemented, effectively delivers real-time monitoring, historical data visualization, automated alerting, and remote control capabilities, demonstrating a robust and user-friendly solution for modern greenhouse management. The functional and usability testing phases confirmed its adherence to the specified requirements and its intuitive nature for end-users. While the project achieved its core goals, a clear set of limitations has been identified, pointing towards opportunities for future growth. The proposed future work highlights exciting possibilities for integrating advanced analytics, improving scalability and security, and expanding accessibility through mobile platforms. Ultimately, this project serves as a strong foundation, underscoring the significant potential of IoT technologies to revolutionize agricultural practices and contribute to more efficient and sustainable food production.

Reference

Alzubi, A. A., & Galyna, K. (2023). Artificial Intelligence and Internet of Things for Sustainable Farming and Smart Agriculture. *IEEE Access*, *11*(2169-3536), 78686–78692. https://doi.org/10.1109/ACCESS.2023.3298215

Baccay, J., Vicente, C., & Bravo, M. (2021). *IoT-based Automated Greenhouse with Monitoring and Control using MQTT Protocol IoT-based Automated Greenhouse with Monitoring and Control using MQTT Protocol*.

Che, H., Muhammad, S., Mohd, H., Siti Azura Ramlan, Harron, N. A., & Abdullah, M. H. (2023). Greenhouse Monitoring System for Chili Plant via IoT Cloud on Ubidots Platform. *2023 10th International Conference on Future Internet of Things and Cloud (FiCloud)*, *10*(175-180), 175–180. https://doi.org/10.1109/ficloud58648.2023.00033

Kumar, S., Kumar, R. R., Kumar, S., & Kothawale, R. (2020, December 12). *Greenhouse Monitoring System Using IOT - Journal of University of Shanghai for Science and Technology*. Journal of University of Shanghai for Science and Technology. https://jusst.org/greenhouse-monitoring-system-using-iot/

Pargaonkar, S. (2023). A Comprehensive Research Analysis of Software Development Life Cycle (SDLC) Agile & Waterfall Model Advantages, Disadvantages, and Application Suitability in Software Quality Engineering. *International Journal of Scientific and Research Publications*, *13*(8), 120–124. https://doi.org/10.29322/ijsrp.13.08.2023.p14015

Prakash, C., Singh, L. P., Gupta, A., & Lohan, S. K. (2023). Advancements in smart farming: A comprehensive review of IoT, wireless communication, sensors, and hardware for agricultural automation. *Sensors and Actuators A: Physical*, *362*(114605), 114605. https://doi.org/10.1016/j.sna.2023.114605

Sai Teja, G., & Sathish, P. (2020). MQTT Protocol based Smart Greenhouse Environment Monitoring System using Machine Learning. *Regular*, *9*(9), 278–285. https://doi.org/10.35940/ijitee.i7149.079920

Singh, T. A., & Chandra, J. (2018). IOT Based Green House Monitoring System. *Journal of Computer Science*, *14*(5), 639–644. https://doi.org/10.3844/jcssp.2018.639.644

Yahaya, A., Abass, Y. A., & Adeshina, S. A. (2019). Greenhouse Monitoring and Control System with an Arduino System. *2019 15th International Conference on Electronics, Computer and Computation (ICECCO)*. https://doi.org/10.1109/icecco48375.2019.9043188

Appendix A: Manual UAT Checklist

Greenhouse IoT Monitoring System - Manual UAT Checklist

Purpose: This checklist serves as a guide for facilitating User Acceptance Testing sessions, allowing testers to systematically go through key functionalities while providing space for detailed observations and feedback.

Name tester: Syed Arif Azri Bin Wan Saiful Bahari

Section: User Management (Admin Role)

Task ID	Task Description	Status (Pass/Fail/N/A)	Notes / Comments (Observations, Issues, Suggestions)
UAT-001	Register a new user account (e.g., 'Operator' or 'Viewer')	Pass	User registered successfully and appeared in user list.
UAT-002	Modify an existing user's details (e.g., username, role, active status)	Pass	Username and role edited without issue. Changes reflected immediately.

UAT-003	Attempt to deactivate or delete a user account	Pass	Deactivated user account and verified they could not log in.

Section: Plant Management

Task ID	Task Description	Status (Pass/Fail/N/A)	Notes / Comments (Observations, Issues, Suggestions)
UAT-004	Add a new plant, providing its name and location	Pass	New plant added with name and location shown correctly.
UAT-005	Edit the details of an existing plant	Pass	Edited plant name and location; updates saved and visible.
UAT-006	Navigate through the list of plants	Pass	Able to view all plants and pagination works.

Section: Sensor Management

Task ID	Task Description	Status (Pass/Fail/N/A)	Notes / Comments (Observations, Issues, Suggestions)
UAT-007	Add a new sensor, assigning it to a specific plant and selecting its type	Pass	Sensor added and linked to plant successfully.
UAT-008	View the list of registered sensors and their assigned plants	Pass	Sensor list correctly displays linked plant and type.
UAT-009	Attempt to delete a sensor	Pass	Sensor deleted and removed from database.

Section: Data Monitoring

Task ID	Task Description	Status (Pass/Fail/N/A)	Notes / Comments (Observations, Issues, Suggestions)

UAT-010	Select a specific plant on the dashboard to view its current sensor readings	Pass	Dashboard shows correct current sensor data for selected plant.
UAT-011	Change the time interval on the trend chart and observe the data changes	Pass	Trend chart updated correctly with selected time intervals.

Section: Alert Settings

Task ID	Task Description	Status (Pass/Fail/N/A)	Notes / Comments (Observations, Issues, Suggestions)
UAT-012	Access the alert settings page for a selected plant	Pass	Alert settings page loads with correct sensor thresholds.
UAT-013	Set or modify minimum and maximum thresholds for	Pass	Thresholds updated and reflected in future alerts.

	different sensor types		
UAT-014	Save the updated alert thresholds	Pass	Changes saved and confirmed with success message.

Section: Recent Alerts Review

Task ID	Task Description	Status (Pass/Fail/N/A)	Notes / Comments (Observations, Issues, Suggestions)
UAT-015	Review the list of recent alerts displayed on the dashboard	Pass	Recent alerts shown in correct order and format.
UAT-016	Apply filters and sort the alerts to find specific events	Pass	Filters work by date and severity; sorting functional.

Section: MQTT Subscriber (Optional/Advanced User)

Task ID	Task Description	Status (Pass/Fail/N/A)	Notes / Comments (Observations, Issues, Suggestions)
UAT-017	Run the PHP MQTT subscriber script to connect to the MQTT broker.	Pass	PHP MQTT subscriber script ran successfully and connected to the MQTT broker.
UAT-018	Subscribe to the main MQTT topic (greenhouse/#) to observe real-time incoming sensor data	Pass	Subscribed to greenhouse/# and saw real-time sensor JSON data.