Faculty of Electrical Technology and Engineering

Development Of Smart Irrigation System With Iot-Based Rain Forecast Capability

MUGILARASAN A/L NADARAJAH

Bachelor of Electrical Engineering Technology with Honours

2025

Development Of Smart Irrigation System With Iot-Based Rain Forecast Capability

MUGILARASAN A/L NADARAJAH

A project report submitted
in partial fulfillment of the requirements for the degree of
Bachelor of Electrical Engineering Technology with Honours

Faculty of Electrical Technology and Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2025

DECLARATION

I declare that this project report entitled “Development Of Smart Irrigation System With Iot-Based Rain Forecast Capability” is the result of my own research except as cited in the references. The project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature :

Student Name : MUGILARASAN A/L NADARAJAH

Date : 3/1/2025

APPROVAL

I hereby declare that I have checked this project report and in my opinion, this project report is adequate in terms of scope and quality for the award of the degree of Bachelor of Electrical Engineering Technology with Honours.

Signature :

Supervisor Name : Ts. Dr. Sulaiman bin Sabikan

Date : 3/1/2025

DEDICATION

This project is dedicated to our families, whose unwavering support, encouragement, and sacrifices have been the cornerstone of our success.

To our mentors and educators, who have guided us with their knowledge and wisdom, inspiring us to strive for excellence and explore the boundaries of innovation.

To our friends and colleagues, whose fellowship and collaboration have made this journey both enjoyable and rewarding.

Above all, we dedicate this work to the pursuit of knowledge and the passion for making a meaningful contribution to society through engineering and technology.

ABSTRACT

In this era of rapid technological advancements and increasing environmental challenges, sustainable resource management has become a critical focus, especially in agriculture. Water is a critical resource for agricultural sustainability, yet it is becoming increasingly scarce due to climate change, population growth, and inefficient water management practices. Traditional irrigation systems often rely on fixed schedules or manual operations, leading to excessive water use or insufficient irrigation, which can adversely affect crop yield and soil health. In this context, modern technologies like the Internet of Things (IoT) offer transformative potential in agriculture by enabling smart, data-driven solutions for precise water management. The first objective is to design and construct hardware that enables the irrigation system to communicate over the internet. The second objective is to create software capable of controlling the system, providing rain forecasting, and enabling real-time monitoring of the irrigation process. The final objective is to evaluate the system's performance by assessing its ability to optimize water usage, its responsiveness to weather forecasts, and its overall reliability in improving irrigation efficiency. This project system using an ESP8266 microcontroller as a control input and output of the system and to transmit the data to the mobile application, respectively. Soil moisture is used to detect the soil moisture in plant container and rain sensors is used to detect the dryness or wetness of the atmospheric, and a buck converter powered by a 12V DC supply. In the later stages, the app will automatically check and monitor all system parameters, including soil moisture, weather forecasts, and irrigation status, ensuring seamless and efficient operation. Observations over five days were conducted to compare the performance of the automated irrigation system with a traditional manual system. The automated system demonstrated efficient water usage by responding to real-time soil moisture levels and rain forecasts. The results highlight the advantages of automation in improving water conservation and reducing human intervention. This project highlights the potential of combining IoT and smart technologies to promote sustainable farming practices and conserve vital resources. This project is also easy to use and user-friendly as it provides a simple Android application for remote monitoring and control, ensuring seamless interaction and convenience for users.

ABSTRAK

Dalam era kemajuan teknologi yang pesat dan cabaran alam sekitar yang semakin meningkat, pengurusan sumber secara lestari telah menjadi fokus kritikal, terutamanya dalam sektor pertanian. Air merupakan sumber yang penting untuk kelestarian pertanian, namun ia semakin berkurangan disebabkan oleh perubahan iklim, pertambahan populasi, dan amalan pengurusan air yang tidak cekap. Sistem pengairan tradisional sering bergantung kepada jadual tetap atau operasi manual, yang boleh menyebabkan penggunaan air berlebihan atau pengairan yang tidak mencukupi, seterusnya memberi kesan negatif terhadap hasil tanaman dan kesihatan tanah. Dalam konteks ini, teknologi moden seperti Internet of Things (IoT) menawarkan potensi transformasi dalam pertanian dengan membolehkan penyelesaian pintar berasaskan data untuk pengurusan air yang lebih tepat. Objektif pertama adalah untuk mereka bentuk dan membina perkakasan yang membolehkan sistem pengairan berkomunikasi melalui internet. Objektif kedua adalah untuk membangunkan perisian yang mampu mengawal sistem, menyediakan ramalan hujan, dan membolehkan pemantauan masa nyata terhadap proses pengairan. Objektif terakhir adalah untuk menilai prestasi sistem dengan menilai keupayaannya dalam mengoptimumkan penggunaan air, tindak balasnya terhadap ramalan cuaca, dan kebolehpercayaannya dalam meningkatkan kecekapan pengairan. Projek ini menggunakan mikropengawal ESP8266 sebagai kawalan input dan output sistem serta untuk menghantar data ke aplikasi mudah alih. Sensor kelembapan tanah digunakan untuk mengesan tahap kelembapan tanah di bekas tanaman, manakala sensor hujan digunakan untuk mengesan keadaan kering atau basah di persekitaran. Sistem ini dikuasakan oleh bekalan 12V DC melalui penggunaan penukar buck (buck converter). Pada peringkat seterusnya, aplikasi ini akan secara automatik memeriksa dan memantau semua parameter sistem, termasuk kelembapan tanah, ramalan cuaca, dan status pengairan, bagi memastikan operasi berjalan lancar dan cekap. Pemerhatian selama lima hari telah dijalankan untuk membandingkan prestasi sistem pengairan automatik dengan sistem manual tradisional. Sistem automatik menunjukkan penggunaan air yang lebih cekap dengan bertindak balas terhadap tahap kelembapan tanah secara masa nyata dan ramalan hujan. Keputusan ini menonjolkan kelebihan automasi dalam meningkatkan penjimatan air dan mengurangkan keperluan campur tangan manusia. Projek ini menekankan potensi gabungan IoT dan teknologi pintar untuk mempromosikan amalan pertanian lestari dan memelihara sumber semula jadi yang penting. Selain itu, projek ini juga mudah digunakan dan mesra pengguna kerana ia menyediakan aplikasi Android yang ringkas untuk pemantauan dan kawalan jarak jauh, memastikan interaksi yang lancar dan kemudahan kepada pengguna.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my supervisor, Ts. Dr. Sulaiman bin Sabikan for his precious guidance, words of wisdom and patient throughout this project.

I am also indebted to Universiti Teknikal Malaysia Melaka (UTeM) and Ministry of Higher Education for the financial support through RM200.00 which enables me to accomplish the project. Not forgetting my fellow colleague for the willingness of sharing his thoughts and ideas regarding the project.

My highest appreciation goes to my parents and family members for their love and prayer during the period of my study. An honourable mention also goes to lecturers for all the motivation and understanding and to Muhammad Syafizal Haikal Bin Samsudin thanks for your invaluable support, encouragement, and assistance in moments when I needed it most.

Finally, I would like to thank all the staffs at the Faculty of Electrical Engineering Technology, fellow colleagues and classmates, the Faculty members, as well as other individuals who are not listed here for being co-operative and helpful.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

%	-	Percentage
Ω	-	Ohm (Unit of Resistance)
A	-	Ampere (Unit of Current)
mA	-	Milliampere
W	-	Watt (Unit of Power)
Hz	-	Hertz (Unit of Frequency)
°C	-	Degree Celsius (Temperature)
≥	-	Greater than or equal to
≤	-	Less than or qual to
+	-	Plus / Addition
-	-	Minus / Subtraction
=	-	Equal to
/	-	Division or Per Unit
X	-	Multiplication
V	-	Voltage

LIST OF ABBREVIATIONS

IoT	-	Internet of Things
ESP8266	-	Espressif System Microcontroller
OLED	-	Organic Light-Emitting Diode
API	-	Application Programming Interface
IDE	-	Integrated Development Environment
Wi-Fi	-	Wireless Fidelity
JSON	-	JavaScript Object Notation
HTTP	-	Hypertext Transfer Protocol
RTC	-	Real-Time Clock
ADC	-	Analog-to-Digital Converter
PWM	-	Pulse Width Modulation

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

A Smart Irrigation System leverages advanced technologies, including sensors, automation, and IoT (Internet of Things) connectivity, to optimize water usage in agriculture. Traditional irrigation methods often result in water wastage due to over-irrigation or inefficient scheduling, causing crop damage and depleting water resources. Smart irrigation systems address these challenges by applying water precisely when and where it is needed, conserving resources, lowering operational costs, and supporting sustainable agriculture, particularly in regions facing water scarcity.

The system employs various sensors, such as soil moisture sensors, rain sensors, and temperature sensors, to monitor soil water content, environmental conditions, and precipitation in real-time. Soil moisture sensors determine when irrigation is necessary by measuring soil hydration levels, while rain sensors detect rainfall and suspend irrigation when rain is present. This combination of sensors ensures efficient irrigation by dynamically adapting to changing weather and soil conditions.

IoT connectivity forms the backbone of a smart irrigation system, enabling integration between sensors, controllers, and actuators with internet-based platforms. A microcontroller, such as the ESP8266, facilitates communication between these components and cloud services, providing real-time data on soil moisture, weather conditions, and system status. Through smartphones, tablets, or computers, users can remotely monitor and manage irrigation schedules, make data-driven decisions, and optimize system performance.

Rain forecasting further enhances water conservation in smart irrigation systems. By accessing weather data through APIs, the system can predict upcoming rainfall and adjust watering schedules accordingly. This feature prevents over-watering during rain, protects crops, and significantly reduces water consumption, supporting sustainable resource management and efficient farming practices.

The integration of IoT technology and rain forecasting in smart irrigation systems represents a breakthrough in precision agriculture. These systems ensure plants receive the optimal amount of water, reducing waste, conserving resources, and improving crop yields. By minimizing environmental impact and empowering farmers with effective tools for resource management, smart irrigation systems play a crucial role in fostering sustainable, data-driven agriculture, particularly in areas with limited water availability or unpredictable rainfall patterns.

1.2 Project Relation to Current Issues to Global/ Society

The development of a Smart Irrigation System with IoT-Based Rain Forecasting is highly relevant in addressing several pressing global challenges, particularly water scarcity and inefficient agricultural practices. Water is becoming an increasingly scarce resource worldwide, with climate change, population growth, and overexploitation putting immense pressure on available freshwater supplies. Traditional irrigation systems often lead to significant water wastage, contributing to the depletion of water reserves and environmental degradation. By integrating IoT technology and rain forecasting, your project promotes more efficient water management, ensuring that water is only used when necessary. This is especially crucial in regions suffering from droughts or irregular rainfall patterns, where conserving water can help protect both agricultural productivity and local ecosystems.

Additionally, the project aligns with the global push toward sustainable agriculture and food security. As the world's population continues to grow, there is an increasing demand

for food, making efficient agricultural practices more important than ever. Smart irrigation systems reduce the environmental impact of farming by optimizing water use, which not only conserves this valuable resource but also enhances crop yields. The integration of rain forecasting further ensures that irrigation is synchronized with weather conditions, minimizing unnecessary water use. This approach helps farmers save costs, improve productivity, and adopt more sustainable farming practices, thus contributing to the global goals of achieving food security, conserving water, and mitigating the impacts of climate change.

1.3 Problem Statement

The project often faces a lack of real-time monitoring and adaptive control mechanisms, which leads to inefficient water usage and ineffective irrigation practices. Additionally, unpredictable weather conditions can result in overwatering or underwatering, further intensif water waste and potentially harming crops or landscapes. Traditional irrigation methods often fail to account for rainfall forecasts, leading to unnecessary irrigation even when rain is expected. This project aims to address these challenges by developing a Smart Irrigation System integrated with IoT technology. The system will utilize real-time rain data, weather conditions, and IoT-based rain forecasts to optimize watering schedules. By incorporating accurate rain forecasting, the system will adjust irrigation patterns proactively based on expected rainfall, ensuring that water is only applied when needed. This approach will not only enhance water efficiency but also promote sustainable irrigation practices in agriculture and landscaping, minimizing water waste and improving crop health.

1.4 Project Objective

The objective of the project is shown in below.

a. To develop hardware for an irrigation system with internet connectivity capability.

b. To develop software to control, provide rain forecasting, and monitor the smart irrigation system.

c. To evaluate the performance of the smart irrigation system.

1.5 Scope of Project

The following primary areas will be the focus of the project's scope:

1. **System Development**
   - Designing the architecture of the Smart Irrigation System, focusing on IoT integration and connectivity.
   - Developing the hardware components, including the ESP8266, soil moisture sensor, and rain sensor for monitoring soil conditions and weather.
   - Implementing control mechanisms using a water pump, solenoid valves, and relay modules to automate irrigation based on sensor data.
   - Integrating an OLED display to present real-time system status and sensor readings locally.
2. **Software Development and Rain Forecasting**
   - Developing firmware for the ESP8266 to process data from soil moisture and rain sensors and control the irrigation system.

- Integrating weather forecasting APIs to predict rainfall and adjust irrigation schedules accordingly.
- Implementing IoT connectivity for remote monitoring and control through a cloud platform or mobile application.
- Designing an intuitive interface to display real-time sensor data, system status, and rain forecast on the OLED display and online dashboards.

3. Performance Evaluation of the Smart Irrigation System

- Conducting field tests to measure the accuracy and reliability of soil moisture and rain sensors under two wheather condition. The first one is when raining and the second one is when it is sunny day.
- Assessing the system's ability to optimize water usage by comparing water consumption with and without the smart irrigation system.
- Evaluating the effectiveness of the rain forecast integration in reducing unnecessary irrigation.
- Analyzing system stability, connectivity performance, and user feedback to identify areas for improvement.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter provides an in-depth review of key technologies and components integral to the development of a Smart Irrigation System with IoT-based rain forecasting. It explores the IoT environment, which enables seamless communication between sensors, controllers, and cloud platforms, facilitating real-time monitoring and adaptive control of irrigation systems. The review also covers rain sensor systems, crucial for detecting precipitation and preventing unnecessary watering. Additionally, the power supply system is compared, highlighting the importance of reliable energy sources, which provides a stable and reliable energy source for the system's sensors, actuators, and microcontroller. Display devices, are explored for their role in presenting real-time system data and user feedback. The weather forecasting system is also discussed, focusing on its ability to predict rainfall and optimize irrigation schedules based on weather patterns. Relevant journals and studies on smart irrigation and IoT integration are reviewed to provide insights into current trends and methodologies, concluding with a summary that highlights key findings and identifies gaps that this project aims to address.

2.2 Understanding Global/Current Issue in the Literature

Water scarcity is a pressing global issue that profoundly impacts agricultural productivity and food security. With growing population pressures, urbanization, and climate change, access to freshwater resources for irrigation is becoming increasingly limited. In many regions around the world, traditional irrigation methods often result in inefficient water usage, leading to depletion of groundwater reserves, soil degradation, and ecosystem

degradation. Moreover, unpredictable weather patterns and erratic rainfall further exacerbate water stress in agricultural regions. Addressing these challenges requires innovative solutions that enhance water efficiency and promote sustainable irrigation practices. This project intends to address the problem of water shortage in agriculture by creating a smart irrigation system with IoT-based rain forecast capacity. By using real-time environmental data and predictive analytics, the system will optimize irrigation schedule and reduce water waste. This proactive approach contributes to the long-term sustainability of agricultural systems by not only improving crop productivity and adaptability to changing climatic circumstances, but also conserving water resources.

2.3 IoT Environment

2.3.1 IoT-Based Microcontroller

An IoT-based microcontroller is a compact, low-power device that collects data from sensors, processes it, and communicates with other devices or the internet, forming the backbone of IoT systems. These microcontrollers are programmed to perform tasks like monitoring environmental conditions, controlling devices, and transmitting data remotely. Popular examples include the ESP8266, known for its built-in Wi-Fi and affordability, making it ideal for simple IoT projects, and the ESP32, which offers both Wi-Fi and Bluetooth capabilities, along with more processing power for complex applications. Other options like the Arduino Uno (with an Ethernet shield) and Raspberry Pi are also used, with the Arduino being a great choice for basic IoT setups and the Raspberry Pi providing higher processing capabilities for advanced tasks. Table 2.1 compares the Microcontroller and Microprocessor. These devices enable efficient, scalable IoT solutions for a wide range of applications, from smart irrigation to home automation.[1].

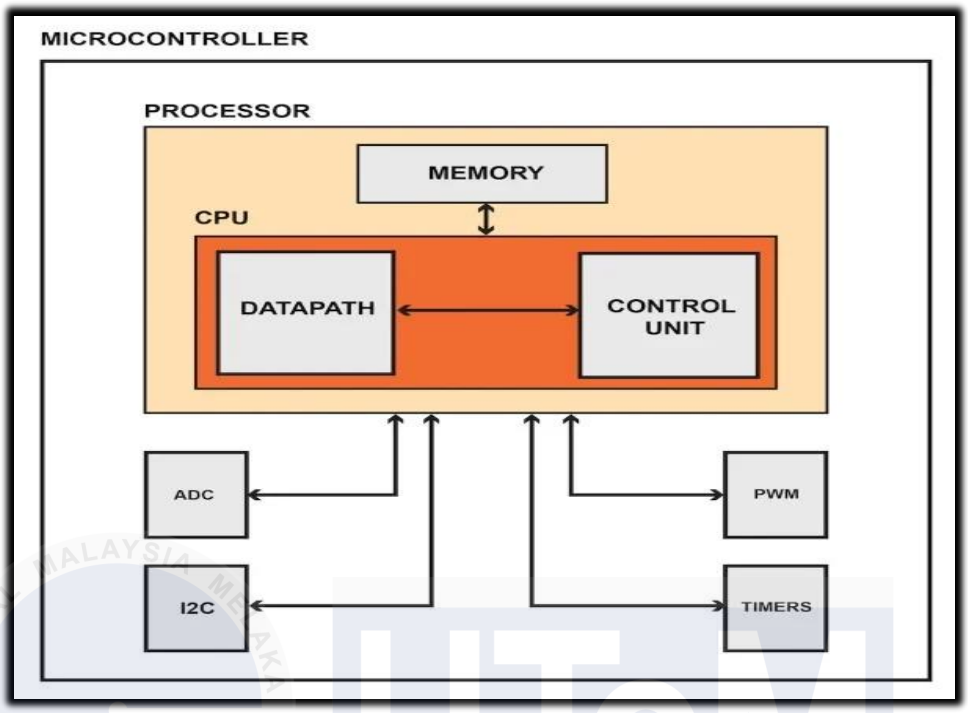

Figure 2.1 : Block Diagram of Microcontroller

A microcontroller's block diagram illustrates its main components and their interactions. The CPU acts as the brain, executing instructions and controlling the system. Memory units like ROM store program code, while RAM provides temporary data storage. I/O ports connect the microcontroller to external devices, such as sensors or actuators. Timers and counters handle timing and event tracking. Communication interfaces like UART, SPI, or I2C enable data exchange with other devices. ADC converts analog signals to digital, and DAC does the reverse for analog control. The oscillator/clock ensures synchronized operation, while the power management unit regulates voltage and conserves energy Figure 2.1 illustrates the Block Diagram of Microcontroller.

Table 2.1 : Comparison Between Microcontroller and Microprocessor

Microcontroller	Microprocessor
CPU, RAM, ROM, I/O and timer are all on a single chip	CPU is stand alone, RAM, ROM, I/O and timer are separate
Fixed amount of on-chip ROM, RAM, I/O ports	Designer can decide on the amount of ROM, RAM and I/O ports
For applications in which cost, power and space are critical	Expensive, high power consumption and versatile
Single-purpose (control-oriented)	General-purpose
Low processing power	High processing power
Instruction sets focus on control and bit-level operations	Instruction sets focus on processing-intensive operations
Typically 8/16 bit	Typically 32/64 bit
Typically single-cycle/ two-stage pipeline	Typically deep pipeline (5-20 stages)

2.3.2 ESP8266

The ESP8266 is a low-cost, low-power Wi-Fi microchip designed for IoT applications, widely used in embedded systems. It integrates a microcontroller with Wi-Fi capabilities, allowing it to connect to wireless networks and the internet without the need for additional modules. The ESP8266 is popular for its small size, affordability, and ease of use, making it ideal for a variety of IoT projects, such as smart home automation, weather monitoring, and smart irrigation systems. It is often programmed using the Arduino IDE, which simplifies development for both beginners and advanced users. Despite its small size, the ESP8266 offers sufficient processing power and flexibility, supporting various sensors and devices, and making it a versatile choice for creating connected, intelligent systems [8] .

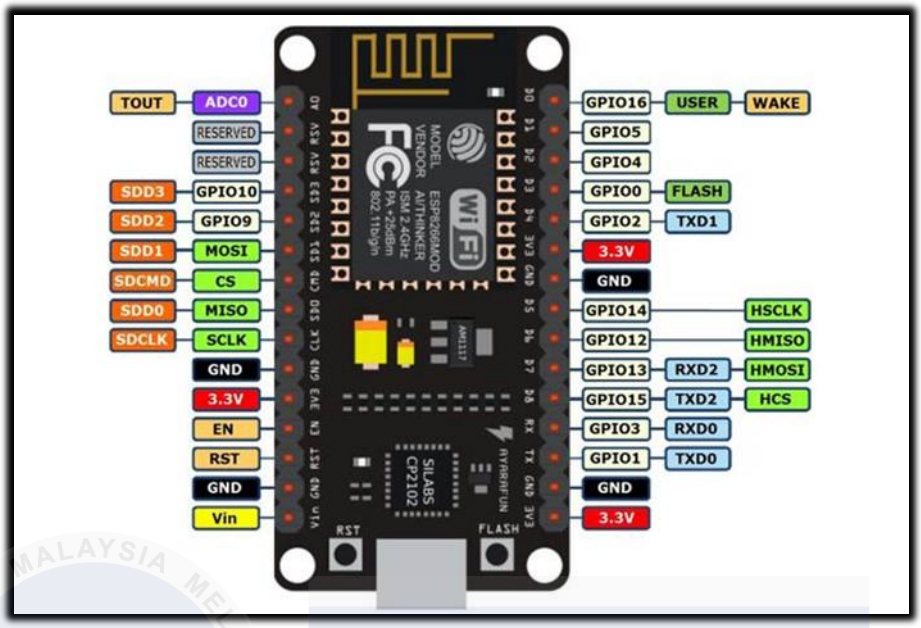

Figure 2.2 : ESP8266 Pinout

The ESP8266 pinout consists of several pins used for various functions. It includes GPIO (General Purpose Input/Output) pins for controlling or reading signals from external devices, with some pins having additional features like PWM, ADC, or I2C support. The power pins include a 3.3V input and ground pins for providing power to the module. The TX and RX pins handle serial communication for programming and data transfer. The EN pin enables or disables the chip, and the RST pin resets the module. Understanding the pinout is essential for properly connecting peripherals and ensuring the ESP8266 functions correctly in IoT applications as shown in Figure 2.2 : ESP8266 PinoutFigure 2.2 .

2.3.3 ESP32

The ESP32 is a powerful, low-cost microcontroller with integrated Wi-Fi and Bluetooth capabilities, designed for a wide range of IoT applications. It builds on the features of the ESP8266 but offers significantly more processing power, more GPIO pins, and additional features like dual-core processing, low-power modes, and support for both Bluetooth Classic

and Bluetooth Low Energy (BLE). The ESP32 is ideal for more complex IoT projects that require higher performance, such as real-time data processing, sensor integration, and wireless communication. It is commonly used in applications like home automation, smart agriculture, and wearable devices. Like the ESP8266, it can be programmed using the Arduino IDE, making it accessible to a wide range of users, from beginners to advanced developers, and offering greater flexibility in designing sophisticated, connected systems [9].

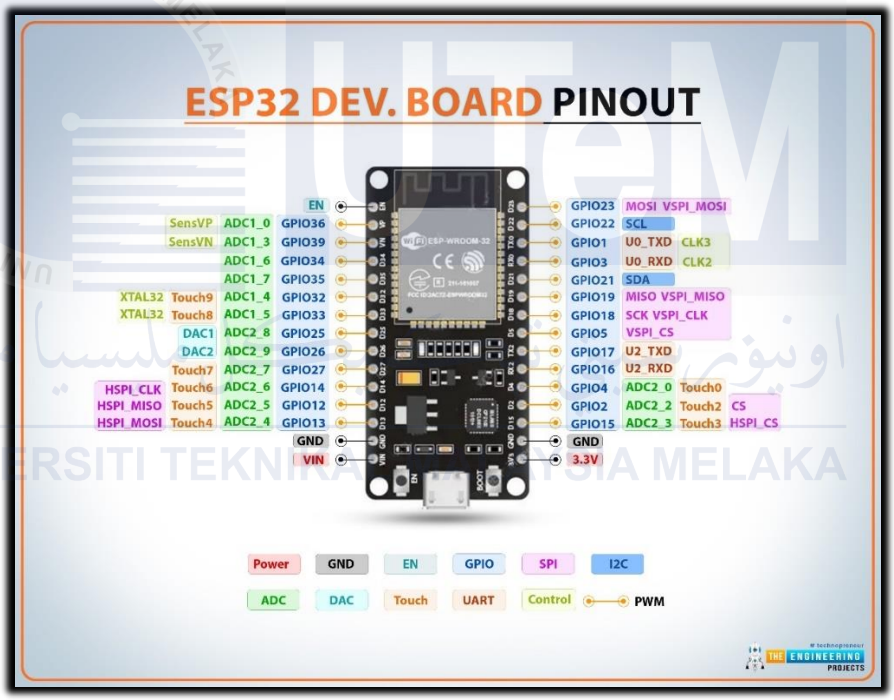

Figure 2.3 : ESP32 Pinout

The ESP32 is a highly adaptable microcontroller with numerous pins designed for various functions, making it suitable for diverse IoT applications. Its GPIO (General Purpose Input/Output) pins can be configured as inputs to read sensor data or as outputs to control actuators. Additionally, it features dedicated pins for specialized tasks such as PWM (Pulse Width Modulation), ADC (Analog-to-Digital Conversion), I2C, SPI, and UART

communication. Additionally, it has pins for power supply, such as 3.3V and ground (GND), as well as a built-in Wi-Fi and Bluetooth antenna. The ESP32's pinout layout allows for flexibility in designing various applications, including home automation, smart irrigation, and more as shown in Figure 2.3 .

2.3.4 Arduino Uno

The Arduino Uno, powered by the ATmega328P microcontroller, is a widely used board designed for easy integration into various electronics and IoT projects. It offers 14 digital input/output pins, 6 analog input pins, a USB port for programming, and a power jack for external power, ensuring versatility and user-friendliness. Popular in education and prototyping, it is valued for its simplicity and extensive community support. Although it lacks built-in internet connectivity, it can be equipped with external modules like an Ethernet shield or Wi-Fi module for network communication. Programmed through the Arduino IDE, it provides an intuitive platform for beginners and hobbyists to develop IoT and embedded systems efficiently [2] .

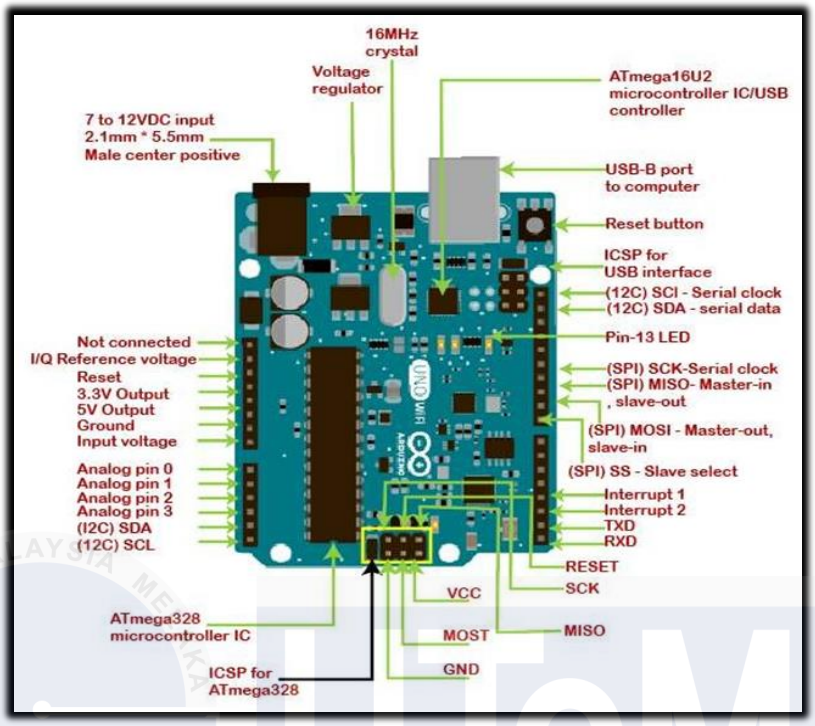

Figure 2.4 : Arduino Uno Pinout

The Arduino Uno is a popular microcontroller board that features a set of pins for various functions. It has 14 digital input/output (I/O) pins, which can be used to read digital signals or send outputs to devices like LEDs or motors. It also has 6 analog input pins for reading analog sensors. The board includes pins for power supply: 5V, 3.3V, and GND (ground), as well as a VIN pin for supplying external voltage. There are also dedicated pins for communication, including two for serial communication (TX and RX), along with I2C (SCL and SDA) and SPI pins. The Arduino Uno is equipped with an integrated USB interface for easy programming and power supply, making it a versatile and beginner-friendly choice for various electronics projects as shown in Figure 2.4

Table 2.2 : Arduino Uno Technical Specifications

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

The Table 2.2 shows the technical specification of the Arduino Uno

2.3.5 Rasberry Pi

The Raspberry Pi is a small, affordable single-board computer that is widely used for a variety of projects, including IoT, robotics, and educational applications. It features a powerful ARM-based processor, RAM, HDMI output, USB ports, and GPIO pins for connecting sensors and actuators, making it much more capable than typical microcontrollers like Arduino or ESP32. Running a full operating system offers the flexibility to handle more complex tasks like web servers, real-time data processing, and running machine learning models. Due to its versatility and processing power, the Raspberry Pi is ideal for advanced IoT applications, smart home systems, and media centers. Its vast community and extensive software libraries make it accessible to both beginners and

experienced developers, enabling the creation of sophisticated projects that require higher computing capabilities. ways [3].

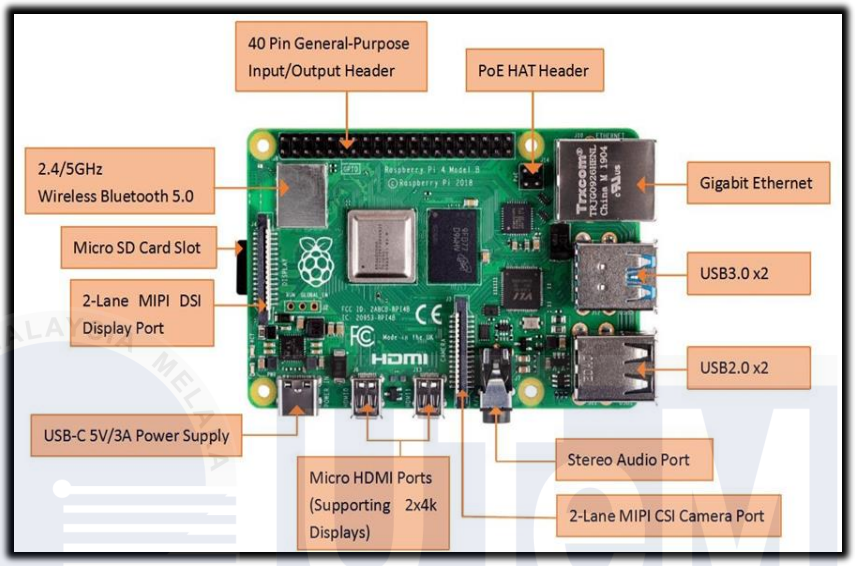

Figure 2.5 : Raspberry Pi 4 Model B

The Raspberry Pi 4 Model B is a versatile single-board computer designed for a wide range of applications. It features a quad-core ARM Cortex-A72 processor, offering a significant boost in performance compared to previous models. It comes with multiple RAM options (2GB, 4GB, or 8GB), making it suitable for tasks ranging from simple programming to running complex applications. The Raspberry Pi 4 has multiple connectivity options, including Gigabit Ethernet, Wi-Fi, Bluetooth 5.0, and two USB 3.0 ports for fast data transfer. It also supports dual monitor output via two micro-HDMI ports, enabling 4K video output. With GPIO (General Purpose Input/Output) pins, it can be used for various electronics projects, and its affordability makes it popular for educational, DIY, and prototyping purposes as shown in Figure 2.5.

Table 2.3 : Comparison Between Different Types of Raspberry Pi Boards

	Raspberry Pi 1 Model A	Raspberry Pi 1 Model A+	Raspberry Pi 1 Model B	Raspberry Pi 1 Model B+	Raspberry Pi 2 Model B	Raspberry Pi 3 Model B	Raspberry Pi Zero
USB 2.0 Ports	1	1	2	4	4	4	1 (Micro-USB)
Ethernet	None	None	10/100 Mbit/s	10/100 Mbit/s	10/100 Mbit/s	10/100 Mbit/s	None
Bluetooth	None	None	None	None	None	4.1	None
WiFi	None	None	None	None	None	802.11n	None
Audio In	I²S	I²S	I²S	I²S	I²S	I²S	I²S
Audio Out	I²S, analog (3.5mm jack), digital (HDMI)	I²S, analog (3.5mm jack), digital (HDMI)	I²S, analog (3.5mm jack), digital (HDMI)	I²S, analog (3.5mm jack), digital (HDMI)	I²S, analog (3.5mm jack), digital (HDMI)	I²S, analog (3.5mm jack), digital (HDMI)	Digital (mini-HDMI), analog GPIO PWM
Video In	CSI Camera Connector	CSI Camera Connector	CSI Camera Connector	CSI Camera Connector	CSI Camera Connector	CSI Camera Connector	None
Video Out	HDMI, Composite (RCA)	HDMI, Composite (TRRS)	HDMI, Composite (RCA)	HDMI, Composite (TRRS)	HDMI, Composite (TRRS)	HDMI, Composite (TRRS)	Mini-HDMI, GPIO Composite
External Storage	SD	MicroSD	SD	MicroSD	MicroSD	MicroSD	MicroSD

The Raspberry Pi boards vary in terms of performance, connectivity, and memory as shown in the Table 2.3 .

2.3.6 Comparison Between Microcontroller Devices

Table 2.4 below shows some differences between Esp8266, Esp 32, Arduino Uno board and RaspberryPi board.

Table 2.4 : Comparison Between Microcontrollers

	Esp 8266	Esp 32	Arduino Uno	Rasberry Pi
Processing Power	Single-core processor at 80 – 160 MHz	Dual-core processor at up to 240 MHz	ATmega328P microcontroller 16 MHz	Quad-core processor 1.2 GHz
Connectivity	Built-in Wi-Fi, no Bluetooth support.	Wi-Fi and Bluetooth support.	No built-in Wi-Fi/Bluetooth	Built-in Wi-Fi and Ethernet support, Bluetooth
Cost	Very affordable	Slightly more expensive than the ESP8266	Costs depending on the model and version.	More expensive

2.3.7 Internet of Things (IoT)

IoT technology is central to the system's functionality and the project, enabling real-time data collection and remote control. Devices such as soil moisture sensor system, rain sensor system, and weather monitoring system tools are connected to the system using IoT-enabled microcontrollers like the ESP8266 or ESP32, allowing data to be transmitted over the internet for analysis. The system uses this data to adjust irrigation schedules, optimizing

water usage based on current soil conditions and forecasted rainfall. This integration of IoT not only reduces water waste but also allows for predictive control, making the system more efficient and sustainable by automating irrigation based on real-time environmental data which shown in Figure 2.6 [4].

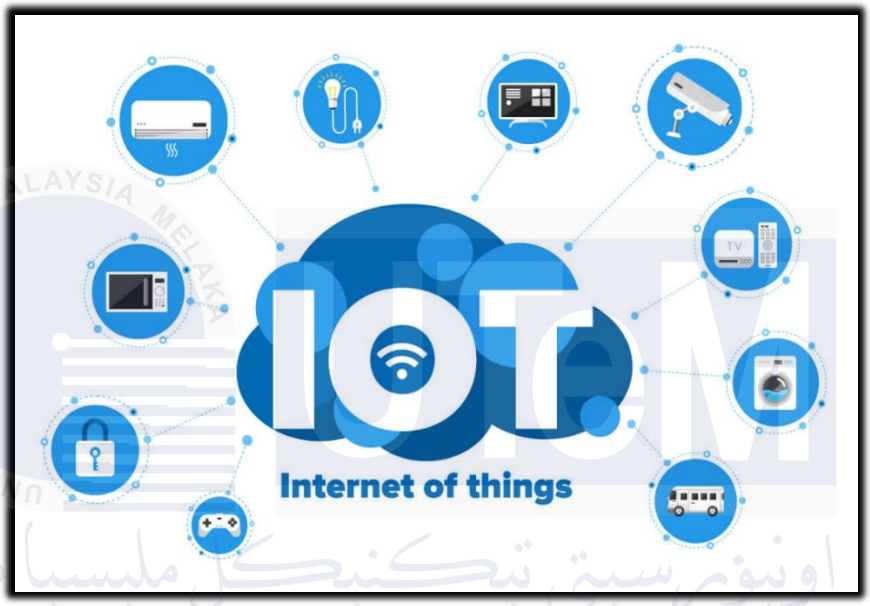

Figure 2.6 : Internet of Things

2.3.8 Wi-fi Network

The connection between IoT (Internet of Things) and Wi-Fi is essential for enabling seamless communication between IoT devices and the internet or other devices. Wi-Fi provides a wireless communication protocol that allows IoT devices, such as sensors, actuators, and microcontrollers, to connect to a network without the need for physical cables. This connectivity enables IoT devices to transmit data to cloud platforms or other devices for analysis, control, and monitoring. In IoT applications, Wi-Fi is commonly used for its wide availability, reliable connection, and relatively high data transfer speeds, making it ideal for home automation, smart agriculture, healthcare, and many other IoT use cases. By

utilizing Wi-Fi, IoT devices can be remotely accessed and managed, enhancing automation, efficiency, and user convenience [5].

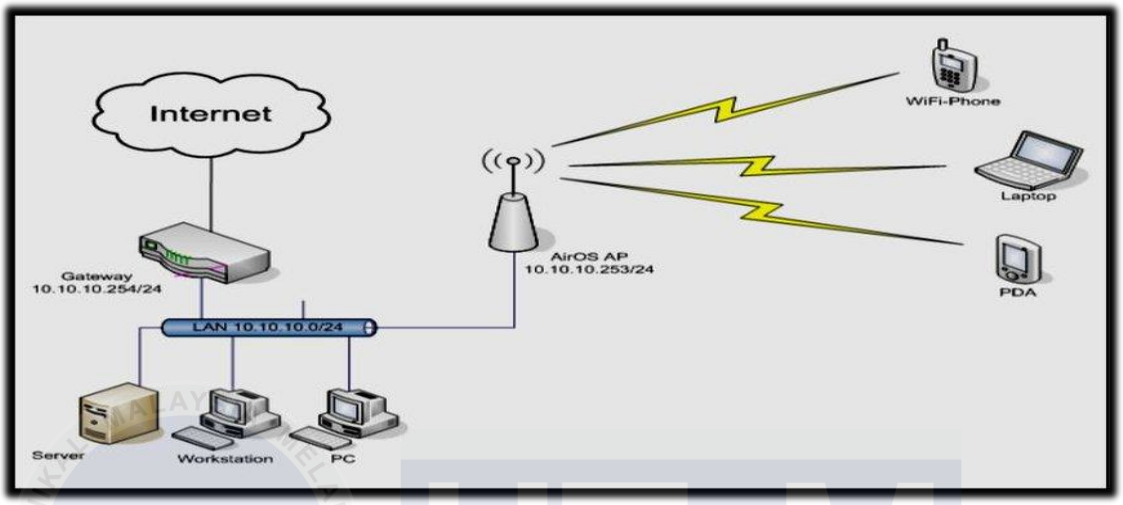

Figure 2.7 : Wi-Fi Network

The internet is a global network that connects computers and devices, allowing them to communicate with each other. A gateway is a device that connects different networks, enabling data to flow between them, often used to link local networks to the internet. A server is a computer or system that provides services, data, or resources to other devices, clients, or systems over a network as shown in Figure 2.7 .

2.3.8.1 Difference between Wi-Fi and WLAN

Wi-Fi and WLAN are often used interchangeably, but they are not exactly the same. Wi-Fi refers to a specific technology that allows devices to connect to the internet wirelessly using radio waves. It is a type of wireless communication standard used for local networks. WLAN (Wireless Local Area Network), on the other hand, refers to the entire network that uses wireless technology, including Wi-Fi, to connect devices within a limited area, like a home, office, or school. Essentially, Wi-Fi is the technology that powers the wireless

connections within a WLAN, which is the broader network structure. A difference between Wi-Fi and WLAN is shown in Figure 2.8.

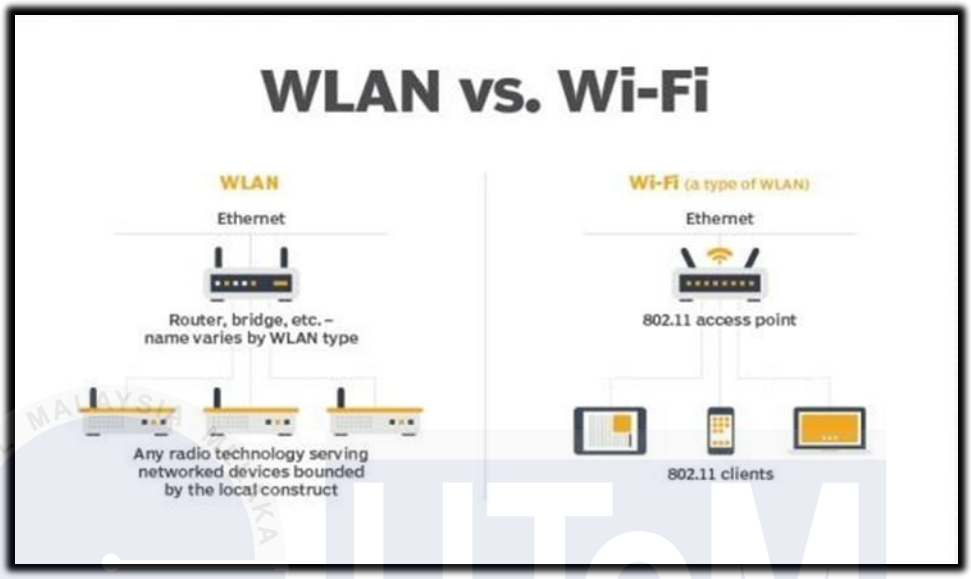

Figure 2.8 : Comparison Between Wi-Fi and WLAN

2.3.9 Cellular Network

A cellular network is a system of mobile communication that uses radio frequencies to provide wireless connectivity over large geographic areas, such as cities, regions, or even entire countries. It relies on a network of cell towers that divide the coverage area into smaller sections called cells, allowing mobile devices to connect and communicate. In the context of IoT and Wi-Fi, a cellular network is crucial when devices need to be connected over long distances or in areas where Wi-Fi is unavailable or impractical. While Wi-Fi is ideal for short-range communication within a local area, IoT devices may require a cellular network for broader, wide-area coverage, such as in remote locations or mobile applications. Cellular networks enable IoT devices to maintain reliable connectivity for real-time data transfer and remote control, ensuring that the IoT system can function even when Wi-Fi is not accessible [6] .

Table 2.5 : Comparison Between 1G, 2G, 3G, 4G and 5G

Technology	1G	2G/2.5G	3G	4G	5G
Deployment	1970/1984	1980/1999	1990/2002	2000/2010	2014/2015
Bandwidth	2kbps	14-64kbps	2mbps	200mbps	>1gbps
Technology	Analog cellular	Digital cellular	Broadbandwidth/ cdma/ip technology	Unified ip & seamless combo of LAN/WAN/WLAN/PAN	4G+WWWW
Service	Mobile telephony	Digital voice, short messaging	Integrated high quality audio, video & data	Dynamic information access, variable devices	Dynamic information access, variable devices with AI capabilities
Multiplexing	FDMA	TDMA/CDMA	CDMA	CDMA	CDMA
Switching	Circuit	Circuit/circuit for access network&air interface	Packet except for air interface	All packet	All packet
Core network	PSTN	PSTN	Packet network	Internet	Internet
Handoff	Horizontal	Horizontal	Horizontal	Horizontal& Vertical	Horizontal& Vertical

1G provided basic voice communication with analog signals, while 2G introduced digital signals, enabling text messaging and limited data transfer. 3G brought faster data speeds and enabled mobile internet browsing, video calling, and media streaming. 4G offered much higher speeds, supporting HD video streaming and faster internet, while 5G promises ultra-fast speeds, low latency, and enhanced connectivity for IoT and advanced technologies as shown in Table 2.5 .

2.3.9.1 Comparison Between Wi-Fi and Cellular Network

Wi-Fi and cellular networks are both used for wireless communication, but they differ in their coverage, speed, and usage. Wi-Fi provides high-speed internet access over short distances, typically within a home, office, or public space, using a router connected to a broadband internet service. It offers faster speeds and lower latency but is limited to areas with Wi-Fi hotspots. In contrast, cellular networks cover much larger areas, such as entire

cities or countries, using mobile towers and cellular signals. While cellular networks provide broader coverage and are ideal for mobility, they tend to have slower speeds and higher latency compared to Wi-Fi. Additionally, cellular networks are typically used for mobile phones and IoT devices in areas where Wi-Fi isn't available. This is shown in the Table 2.6.

Table 2.6 : Comparison Between Wi-Fi and Cellular Network

WiFi	VS	Cellular
• Accurate • Measured by proximity to WiFi routers	Location Detection	• Accurate • Measured by proximity to cell towers • Sometimes works when outside of WiFi range
• Traditionally higher bandwidth • Speeds up to 9 Gbps (WiFi 6)	Bandwidth	• Comparable bandwidth • Theoretical speeds up to 20 GBPS (5G)
• Limited range • Easily obstructed	Coverage	• Extensive coverage • Including underground and rural areas
• Encryption requires enabling • Security updates depend on network owner	Security	• Cellular data is encrypted by default • Providers prioritize security updates
• Slightly lower power consumption	Power Usage	• Slightly higher power consumption
• Cost-effective for large data transfers • Ability to piggyback off existing networks	Cost	More expensive on a per-bit basis

2.4 **Actuators**

Actuators are mechanical or electromechanical devices that convert energy such as electrical, hydraulic, or pneumatic into motion or physical action to control or operate a system. They are classified into different types based on the energy source they use as example electric actuators are motors and solenoids, hydraulic actuators powered by liquid pressure, and pneumatic actuators powered by compressed air. These devices receive control signals from a controller and execute specific actions, such as moving, adjusting, or operating a mechanism.

In the context of a smart irrigation system, an actuator plays a crucial role in managing the water pump. For instance, an electric actuator can be used to turn the water pump on or off based on signals from an IoT-enabled controller. This ensures water is supplied only when necessary, optimizing water usage and preventing wastage. The integration of actuators with IoT and sensors, like soil moisture and rain sensors, allows the system to operate efficiently and autonomously, aligning with the project's goal of sustainable and smart water management [11] .

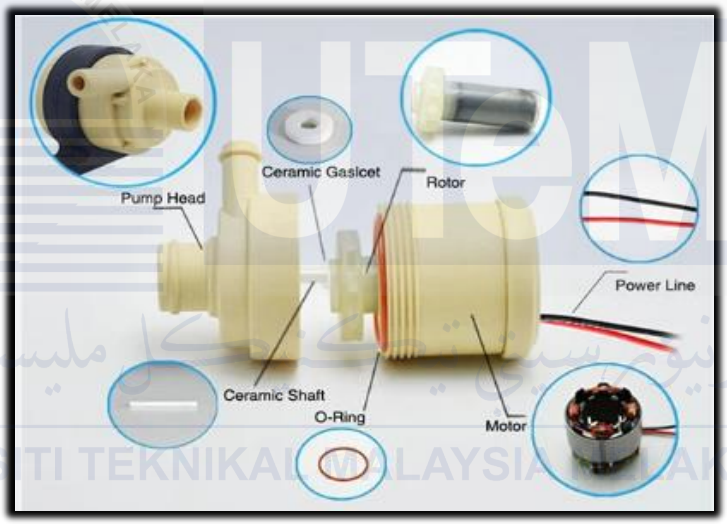

Figure 2.9 : Water Pump Inner Components

The inner components of a water pump typically include an impeller, which drives water flow, and a motor that powers the impeller. Additionally, there is a shaft that connects the motor to the impeller, and seals to prevent leaks. Some pumps also have bearings to support the shaft and ensure smooth operation as shown in Figure 2.9 .

2.5 **Display Devices**

Display devices are electronic components used to visually present information, such as text, images, or data, to users. Common types include LEDs**,** LCDs**,** and OLEDs, each suited for specific applications based on factors like power consumption, resolution, and brightness. These devices work by converting electrical signals into visual output; for example, LCDs use a backlight and liquid crystals to produce images, while OLEDs emit light directly through organic compounds, offering thinner and more energy-efficient displays. In IoT-based systems like a smart irrigation system, display devices are often connected to microcontrollers like ESP8266 or Arduino to show real-time data such as soil moisture levels, rain forecasts, and system status. This visual interface makes it easier for users to monitor and control the system, enhancing functionality and user experience [13] .

Figure 2.10 : Different Types of Display Devices

Different types of display devices include LED displays, which are energy-efficient and widely used in various applications, LCD displays, which offer clear visuals and are

commonly used in screens like televisions and monitors, and OLED displays, which provide high contrast and vibrant colors for more advanced applications. Each type has distinct advantages depending on the use case and power requirements as shown in Figure 2.10 .

Table 2.7 : Comparison of Different Types of Display Devices

Display Device	Features
LCD	• LCD stands for Liquid Crystal Display (shorthand for "Passive Matrix LCD) • Works by adjusting the amount of light blocked. Usually has a backlight but might not (clocks, calculators, Nintendo Gameboy). The green-black ones can be very cheap and are a mature technology. Response time can be slow.
LED	• LED stands for Light Emitting Diode. • As the name suggests, LED emits light rather than blocking it like LCD. Used for red/green/blue/white indicator lights everywhere. Some manufacturers advertise "LED" displays that are TFT screens with a white LED backlight, which is just confusing. Ones that are real LED screens are usually OLED.
OLED	• OLED stands for Organic Light Emitting Diode • Comparatively recent technology, so cost still quite variable and not available in really large sizes. In theory can be printed on plastic, resulting in lighter flexible displays with good brightness, good power consumption and good response time.
TFT	• TFT stands for Thin Film Transistor (shorthand for "Active Matrix TFT LCD") • TFT is a type of LCD with a thin film transistor attached to each pixel. All computer LCD screens are TFT since early 2000s; older ones had slower response times and poorer colour. Cost is now very good; power consumption is fairly good but dominated by the backlight. Has to be manufactured out of glass.

Table 2.7 shows the comparison between four display devices which are LCD, LED, OLED, and TFT .

2.6 **Rain Sensor System**

A resistive rain sensor is widely used in IoT applications to detect rainfall. It typically consists of two components: a rain detection module and a control board. The detection module has a PCB with exposed conductive traces that act as electrodes. When raindrops fall on the sensor, the water forms a conductive path between the traces, reducing resistance.

The control board processes this change and provides **a**nalog output proportional to water level and digital output rain detected or not [10] .

2.6.1 Hardware Review Rain Sensor

2.6.1.1 Sensing Pad

For its design wise the sensing pad is a Printed Circuit Board (PCB) with exposed conductive traces. These traces are typically arranged in a grid or interlocking pattern to maximize the surface area for detecting raindrops. The conductive material is often made of copper, which is prone to oxidation or corrosion over time, especially with prolonged outdoor exposure. For its functionality wise when raindrops land on the pad, the water bridges the conductive traces, creating a connection that decreases resistance. This change in resistance is the core mechanism for detecting rainfall as shown in Figure 2.11 .

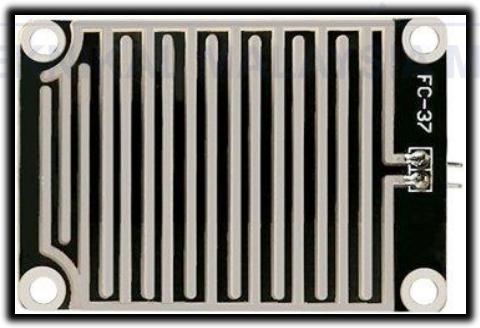

Figure 2.11 : Sensing Pad

2.6.1.2 Control Module

For the control module is typically contains a comparator IC (like LM393), which converts the analog signal from the sensing pad into a digital signal. It includes adjustable sensitivity via a potentiometer and indicator LEDs for power and rain detection. After that, its functionality is to processes the signal from the sensing pad and provides Analog Output

proportional to rain intensity, used for precise monitoring. Digital Output indicates whether rain is detected, useful for binary decisions On/Off control of irrigation. In this project it uses digital output pin.The module operates within a typical voltage range of 3.3V to 5V, making it compatible with microcontrollers like ESP8266 or Arduino as shown in Figure 2.12 .

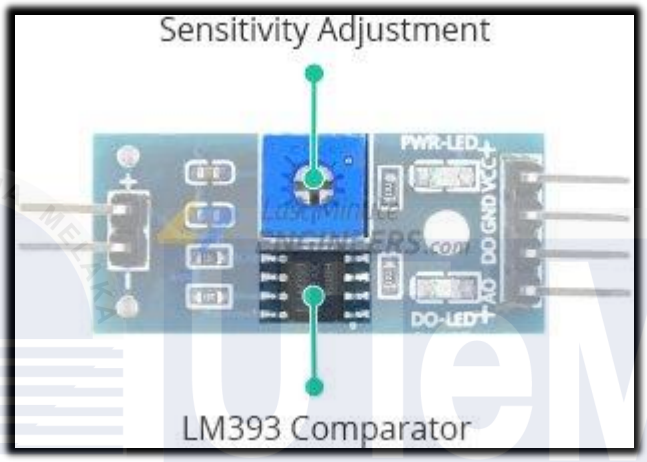

Figure 2.12 : Control Module

2.6.2 Rain Sensor Pinout

Table 2.8 : 4 Pinout of Rain Sensor

AO (Analog Output)	gives us an analog signal between the supply of (5V) to 0V.
DO (Digital Output)	gives Digital output of internal comparator circuit.
GND	is for a ground connection.
VCC	supplies power for the sensor. It is recommended to power the sensor with between 3.3V – 5V.

Table 2.8 shows the pinout of the rain sensor and the pins each function and power supplies.

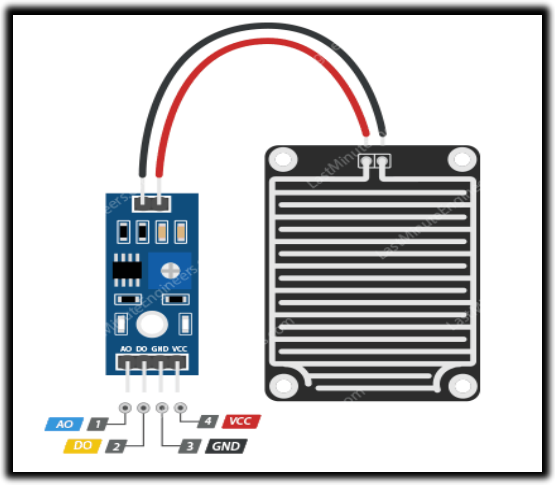

Figure 2.13 : Rain Sensor Pinout Diagram

The rain sensor used in the project operates in digital mode (DO) to detect whether it is raining or not. The sensor's DO pin provides a binary output, where LOW indicates rain is detected, and HIGH indicates no rain. This mode is suitable for the project because it only requires basic rain detection rather than measuring rain intensity, making the setup simpler and more efficient. The other pins include VCC for power and GND for ground, which are connected to the ESP8266 as shown in Figure 2.13 and how it functions in Figure 2.14 .

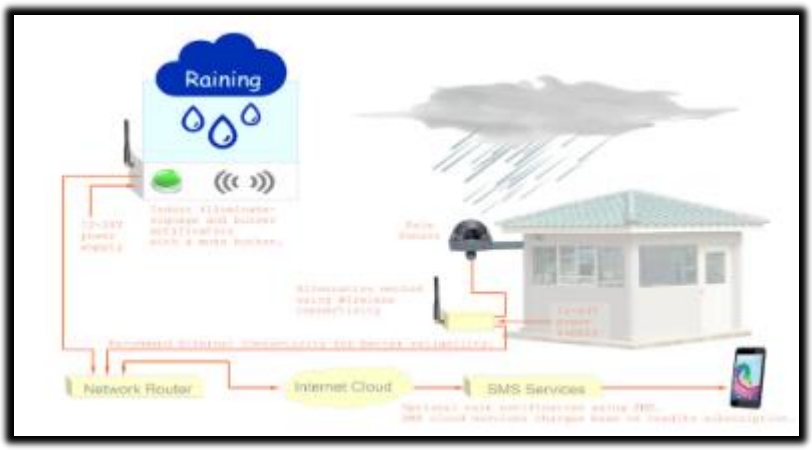

Figure 2.14 : Rain Sensor System

2.7 **Power Supply System**

A power supply system is a critical component that provides the necessary electrical energy to operate devices and systems. It converts energy from a source, such as an electrical outlet or battery, into the appropriate voltage and current required by the connected components. Types of power supplies include direct current (DC) supplies, alternating current (AC) supplies, and batteries, with DC systems commonly used for precision and efficiency in electronic applications.

In IoT-based systems, a buck-boost step-down converter is often employed within the power supply system to regulate voltage and ensure stable operation of components like sensors, microcontrollers, and actuators. This converter can step up or step down the input voltage to meet specific requirements, ensuring efficiency and protecting devices from overvoltage or undervoltage damage.

Compared to battery cells, a regulated power supply system using converters offers consistent and uninterrupted energy, making it more reliable for continuous operation. While battery cells provide portability and are convenient for mobile systems, they require periodic recharging or replacement and may have limited capacity for high-demand applications. A well-regulated power supply system is therefore more suitable for projects requiring long-term stability and precise control [12] .

Figure 2.15 : Types of Battery Cells

Figure 2.15 shows the different types of battery cells that can be used for projects.

2.7.1.1 Comparison Between Different Battery Cells

A simple comparison between different Battery Cells is shown in Table 2.9

Table 2.9 : Comparison Between Different Battery Cells

SIL Aerospace Battery Comparison Chart

Battery Characteristic	Silver Zinc	Lithium Polymer	Lithium Ion	Nickel Cadmium	Nickel Hydride	Lead-Acid Gell Cell
COST	50 - 75 $ / Wh	3.5 - 5 $ / Wh	2.5 - 3.5 $ / Wh	.75 - 1.5 $ / Wh	1.5 - 3.0 $ / Wh	.3 - .50 $ / Wh
Voltage Per Cell	1.85 to 1.2 Volts	4.2 to 3 Volts	3.7 - 2.5 Volts	1.4 to 1 Volts	1.4 to 1 Volts	2.2 - 1.7 Volts
Current Sink Capability	3-4C 5-10C Pulse	2-3C Continuous 5C Pulse	1 - 2C Continuous	1 -2C Continuous	1 - 3C Continuous	5-10C Continuous
Energy Density (Wh/Kg)	70 - 100 Wh / Kg	130 - 150 Wh / Kg	100 - 120 Wh / Kg	30 - 45 Wh / Kg	40 - 55 Wh / Kg	25 - 35 Wh / Kg
Charge/Discharge Cycle Life	10 to 30 Cycles High Rate @ C/2	> 500 Cycles @C/2 80% Initial Capacity	> 500 Cycles @ C/5	> 1000 Cycles @ C/5	> 800 Cycles @ C/5	> 500 Cycles @C, Deep Cycle
Temperature Range	+ 4 C to 74 C	- 20 C to 60 C Withstand 70C for 1 hour	- 10 C to 50 C	-20 C to 50 C	-10 C to 50 C	- 40 C to 60 C
Mechanical Adaptability	Limited Configurations	Unlimited Configurations	Limited Configurations	Limited Configurations	Limited Configurations	Limited Configurations
Toxic Chemistry	Silver Oxide / Zinc Oxide Potassium Hydroxide Electrolyte, Toxic	Lithium Cobalt Oxide Polymer Electrolyte Non-Toxic, Fire, Toxic only	+ Lithium Cobalt Oxide - Cadmium Hydroxide Organic Electrolyte, Toxic	+ Nickel Cadmium - Cadmium Hydroxide P. Hydroxide Elect, Toxic	- LaNi5 + Nickel-Metal P. Hydroxide Elect., Toxic	- Metallic Lead + Lead Dioxide (PbO2) Sulfuric Acid Elect, Toxic
Electrolyte Leakage	Yes. Potassium Hydroxide Electrolyte	No. Polymer Electrolyte	Yes. Organic Solvent Electrolyte	Yes. Potassium Hydroxide Electrolyte	Yes. Potassium Hydroxide Electrolyte	Yes. Sulfuric Acid Electrolyte
Maintenance Required	Electrolyte Injection for each Cell or one time use for Automatically Activated	No.	No. Cell Wire Connections	No. Cell Wire Connections	No. Cell Wire Connections	No. 6-12 V Blocks
Vibration & Shock Resistant	Good	Excellent	Good	Good	Fair	Good
Opt. Voltage Recharge Rate	1.98 Volts 16 hours at C/2	4.2 Volts 8 hours at C/5	3.7 Volts 8 hours at C/5	1.4 Volts 16-24 hours at C/4	1.4 Volts 10-12 hours at C/4	2.4 Volts 16 hours at C/3
Memory Effect	None.	None.	None.	Yes.	None.	None.
Self Discharge	Manual Activate- few weeks Automatic Activate-day one time shot	<5 % month	< 10 % month	25-30% month	20 - 25% month	3-5 % month

2.8 **Wheather Forecasting System**

In the project, the weather forecasting system is integral to achieving efficient and sustainable water management. By providing precise rainfall forecasts, the system can anticipate upcoming weather conditions and adjust irrigation schedules to prevent overwatering during rain or ensure timely watering during dry spells. Combined with IoT, the system collects and processes data from connected sensors, enabling automated responses based on both current and predicted conditions. This synergy reduces water wastage, minimizes manual intervention, and enhances agricultural productivity, aligning perfectly with the project's goals of smart, sustainable irrigation. Figure 2.16 shows the example of the wheather forecasting app [14] .

Figure 2.16 : Wheather Forecasting

2.9 Related Previous Works

2.9.1 Design and Performance Evaluation of a Low-Cost Autonomous Sensor Interface for a Smart IoT-Based Irrigation Monitoring and Control System

The introduction discusses the rapid development and benefits of IoT for real-time monitoring and control of devices, particularly emphasizing the importance of efficient IoT-based smart irrigation systems in agriculture for remote monitoring of crops and controlling water supply. The design of a low-cost autonomous sensor interface for a smart IoT-based irrigation system enhances agricultural efficiency by enabling real-time monitoring of soil moisture, weather conditions, and irrigation control. This system operates autonomously, automatically adjusting irrigation based on sensor data, reducing the need for manual intervention. By utilizing wireless communication technologies like Wi-Fi, it allows remote access and control via mobile applications, making it accessible to farmers, even in water-scarce areas. Performance evaluation focuses on the system's accuracy, energy efficiency, and scalability. The result is a cost-effective, sustainable solution that improves water management, optimizes crop yield, and supports environmental conservation [15] .

2.9.2 Development of Smart Drip Irrigation System Using IoT

The introduction discusses the significance of agriculture in India and the importance of efficient irrigation for maximizing yield, with the objective of implementing a smart drip irrigation system at the National Institute of Technology Karnataka campus. The development of a smart drip irrigation system using IoT aims to address water scarcity and enhance agricultural productivity by providing precise water delivery directly to the plant roots. By leveraging IoT sensors such as soil moisture, temperature, and humidity, the system continuously monitors the environmental conditions and adjusts irrigation schedules accordingly. The integration of cloud-based platforms allows real-time data monitoring and

remote control, giving farmers better decision-making tools. Implementing this system at the National Institute of Technology Karnataka campus serves as a model for its application in diverse agricultural settings. The project focuses on optimizing water usage while ensuring maximum crop yield, ultimately promoting sustainable farming practices [16] .

2.9.3 Development of a Wireless Sensor Network and IoT-based Smart Irrigation System

The introduction discusses the need for innovative agricultural techniques due to population growth, emphasizes the importance of smart irrigation systems for water efficiency, and provides insights into irrigation techniques and Nigeria's current irrigation capacity. The development of a wireless sensor network (WSN) and IoT-based smart irrigation system is aimed at improving water efficiency in agriculture, especially in regions with limited water resources like Nigeria. The system incorporates various sensors to monitor soil moisture, temperature, and weather conditions, transmitting real-time data to an IoT platform for analysis. This enables automated irrigation based on the actual needs of the crops, reducing water wastage and improving crop yield. Nigeria, with its vast agricultural potential, faces challenges in irrigation infrastructure, making smart systems essential for sustainable farming. The integration of IoT technology can modernize Nigeria’s irrigation practices, providing a scalable and efficient solution to meet the growing demand for food [17] .

2.9.4 Smart irrigation system based on internet of things (IOT)

The paper introduces a project in agriculture focusing on controlling water consumption using IoT technology and various sensors, with data being uploaded to the cloud and accessible through an app and website. The smart irrigation system based on IoT technology aims to optimize water usage in agriculture by integrating various sensors that monitor soil

moisture, weather conditions, and crop needs. Data from these sensors is collected and uploaded to the cloud for real-time analysis, providing farmers with valuable insights into irrigation management. The system can be remotely accessed through a mobile app and website, enabling users to monitor and control irrigation schedules from anywhere. By automating irrigation processes, this system reduces water wastage and ensures crops receive the right amount of water at the right time. The project demonstrates how IoT can enhance agricultural efficiency, promote sustainable practices, and improve overall water conservation in farming [18] .

2.9.5 A Pilot Study of Smart Agricultural Irrigation using Unmanned Aerial Vehicles and IoT-Based Cloud System

The paper introduces a new smart agricultural irrigation system that utilizes IoT technology, embedded systems, and UAVs to monitor environmental parameters and automate water irrigation in different regions of a farm. This study explores the integration of unmanned aerial vehicles (UAVs) with IoT-based cloud systems for smart agricultural irrigation. The UAVs are equipped with sensors to monitor environmental parameters such as soil moisture, temperature, and humidity across various regions of a farm. These real-time data are then processed and analyzed in the cloud to optimize irrigation schedules and water distribution. By automating irrigation based on precise environmental data, the system enhances water efficiency and crop growth while reducing human intervention. This innovative approach showcases how UAVs and IoT can work together to improve agricultural productivity and sustainability [19] .

2.9.6 A New Iot-Based Smart Irrigation Management System

The introduction emphasizes the importance of rationalizing water resources in agriculture, the role of IoT in providing solutions, and the inefficiencies in current irrigation practices.

This paper highlights the need for effective water resource management in agriculture and addresses the inefficiencies of traditional irrigation systems. It emphasizes the potential of IoT technology to create smart irrigation systems that provide real-time monitoring and control, ensuring that water is used efficiently. The proposed system utilizes various sensors to collect data on soil moisture, weather conditions, and crop health, which is then analyzed to optimize irrigation schedules. By automating irrigation based on real-time data, the system aims to reduce water wastage, increase crop yield, and support sustainable farming practices. The study illustrates how IoT-based solutions can revolutionize agricultural water management [20] .

2.9.7 Prototyping of Smart Irrigation System Using IoT Technology

The introduction discusses the development of a smart irrigation system using IoT technology to enhance agricultural practices by reducing labor, increasing yield, and modernizing traditional methods, with real-time monitoring and data analysis capabilities. This paper focuses on the development and prototyping of a smart irrigation system that leverages IoT technology to improve agricultural productivity. It highlights the system's ability to reduce manual labor, increase crop yields, and modernize traditional irrigation practices. Through real-time monitoring of environmental factors such as soil moisture, weather conditions, and temperature, the system can adjust irrigation schedules for optimal water usage. The integration of data analysis capabilities further enhances the system's efficiency, enabling more accurate decision-making. By incorporating IoT, this approach aims to drive sustainable farming practices and optimize resource utilization [21] .

2.9.8 IoT based smart irrigation monitoring and controlling system

The introduction provides an overview of IoT, its application in smart irrigation systems, the use of sensors for monitoring, wireless communication for control, and the role of cloud

computing in handling data. This paper explores the application of IoT in smart irrigation systems, focusing on the use of sensors for monitoring soil moisture, temperature, and other environmental parameters. The system uses wireless communication to control irrigation systems remotely, enabling efficient water management. Cloud computing plays a vital role in storing and analyzing the collected data, allowing for real-time access and decision-making. By integrating these technologies, the system aims to optimize water usage, reduce waste, and enhance crop production. The paper emphasizes the significance of IoT in advancing sustainable agricultural practices through automation and data-driven insights [22] .

2.9.9 A distributed system for supporting smart irrigation using Internet of Things technology

The introduction presents a smart irrigation system using IoT technology to enhance irrigation efficiency and prevent under-watering in agricultural fields. This paper discusses the development of a distributed system for smart irrigation using IoT technology to improve irrigation efficiency and prevent under-watering in agricultural fields. The system leverages a network of sensors to collect real-time data on soil moisture, weather conditions, and plant health, which is then processed and analyzed to optimize irrigation scheduling. By enabling remote monitoring and control, this IoT-based system ensures precise water application, reducing water wastage and enhancing crop yield. The paper highlights the potential of distributed IoT systems in creating sustainable agricultural practices through automation and efficient resource management [23] .

2.9.10 Hydroponic-based smart irrigation system using Internet of Things

The introduction discusses the utilization of smart technologies for irrigation, particularly focusing on hydroponic systems, and presents a smart irrigation system using IoT for hydroponics with demonstrated capabilities for monitoring and enhancing crop production. This paper explores the integration of IoT technology into hydroponic systems to create a smart irrigation system that enhances crop production. It highlights how IoT-enabled sensors can monitor critical parameters such as water level, pH, temperature, and nutrient concentration, ensuring optimal growth conditions for crops in a soil-free environment. The system provides real-time data and enables remote control of the irrigation process, improving water and nutrient efficiency. By automating irrigation and closely monitoring plant conditions, the system aims to maximize yield while minimizing resource usage, showcasing the potential of IoT in sustainable agriculture practices, particularly in hydroponics [24] .

Table 2.10 : Comparison Between Related Journals

Title	Author	Year	Objectives	Main Findings	Software Used	Advantages
Design and Performance Evaluation of a Low-Cost Autonomous Sensor Interface for a Smart IoT-Based Irrigation Monitoring and Control System. [15]	S. Abba, Jonah Wadumi Namkusong, Jeong-A Lee, M. Crespo	2019	The study objectives are to design a low-cost autonomous sensor interface for monitoring and controlling irrigation systems, optimize water use for irrigation farming, and reduce the need for high levels of supervision in water supply	The proposed system demonstrates practical applicability, minimizes water loss, prevents plant damage, and simplifies water supply to farmland.	Proteus 8.5 design suite, Arduino integrated design environment, Embedded C programming language	The advantages of the system include facilitating autonomous water supply to crops, minimizing water loss, preventing plant damage, simplifying monitoring and control of water supply.
Development of Smart Drip Irrigation System Using IoT. [16]	Anushree Math, L. Ali, U. Pruthviraj	2018	The objective of the study is to irrigate the plants using the smart drip irrigation system within National Institute of Technology Karnataka campus.	The study developed a smart drip irrigation system using IoT technology for irrigating plants within a specific campus, allowing for monitoring and management through a webpage interface.	Open source platform, Web page, Raspberry Pi camera	Advantages of the smart drip irrigation system include continuous monitoring of plant health parameters, remote monitoring and management, leak detection through water flow sensor.
Development of a Wireless Sensor Network and IoT-based Smart Irrigation System. [17]	J. Ndunagu, K. Ukhurebor, Moses Akaaza, R. B. Onyancha.	2022	The study objectives were to develop a smart irrigation system controller for drip irrigation, focusing on cost-effectiveness and efficiency, automate irrigation processes, save water and energy, increase crop yield, and enable monitoring and control via web or mobile devices.	The study achieved a high accuracy rate of 89% in predicting water needs of plants, with a misclassification rate of 10% and a sensitivity of 79%. - The comparison with the K-nearest neighbors (K-NN) algorithm showed prediction accuracies of 97% and 98%, indicating the effectiveness of the smart irrigation system.	Microsoft Excel, Jupyter Notebook, Arduino IDE, Notepad++, Python, Anaconda, Jupyter Notebook, MATLAB	The advantages of the SIS controller design in the study are its cost-effectiveness and efficiency.

Smart irrigation system based on internet of things (IOT). [18]	Nurulisma Ismail, Sheegillshah Rajendran, Wong Chee Tak, Tham Ker Xin, Nur Shazatushima Shahril Anuar, Fadhil Aiman Zakaria, Yahya Mohammed Salleh Al Quhaif, Hussein Amer M. Hasan Karakhan, and Hasliza A. Rahim	2019	The study objective is to control the water consumption in agriculture field using IoT technology.	The project successfully achieved its objectives related to water consumption, cost-effectiveness, labor efficiency, power consumption, and reliability.	ThingSpeak.com, Firebase	Advantages include improved water consumption efficiency, minimal project cost, reduced labor, lower power consumption, and increased reliability.
A Pilot Study of Smart Agricultural Irrigation using Unmanned Aerial Vehicles and IoT-Based Cloud System.[19]	M. E. Karar, Faris Alotaibi, A. Al-Rasheed, O. Reyad	2021	The study objectives include proposing solutions for continuous monitoring of the field and automatic irrigation using IoT and UAVs, designing and implementing an agricultural monitoring system, and demonstrating the effectiveness of the proposed IoT-based embedded system.	The main findings of the paper are the effectiveness of the proposed IoT-based embedded system in reducing unnecessary water irrigation in smart agriculture and the successful implementation and testing of a new water irrigation system integrating UAV with IoT-based embedded systems.	The software used in the study includes Arduino IDE for programming the Arduino board, Arduino RemoteXY application for Android UI programming, and ArduinoC	The advantages of using IoT-based embedded systems and UAVs in agriculture include remote monitoring of farm conditions, guidance on water requirements, avoidance of unnecessary water irrigation, automation of tasks like pesticide spraying, and the potential

					programming language. The hardware components were programmed using these software tools.	for data collection in open farm areas.
A New Iot-Based Smart Irrigation Management System. [20]	Waseem Hamdoon, A. Zengin	2023	The study objectives are to propose a smart irrigation system based on the Internet of Things, control the appropriate amount and timing of irrigation considering weather conditions, and address the questions of when to make irrigation decisions and how to determine the appropriate amount of irrigation.	The paper presents a new IoT-based smart irrigation management system that aims to rationalize water resources in agriculture without reducing productivity, addressing challenges in precision irrigation and emphasizing sustainable practices.	The software used in the study includes Raspberry OS, Python, TinyDB, Firebase, Flutter, and Android Studio.	The advantages in Waseem Hamdoon, A. Zengin (2023) include proposing a new irrigation architecture based on the Internet of Things, utilizing sensors to monitor soil moisture, temperature, and surrounding conditions to optimize irrigation decisions, and increasing the efficiency of irrigation management. The system overcomes problems seen in other proposed systems by considering dynamic factors in determining irrigation amounts and times. The study also declares no conflicts of interest and states that no ethics committee approval was required.

Prototyping of Smart Irrigation System Using IoT Technology. [21]	Thavath Sai, Bunrong Proeung, Sovichea Tep, Sopheaktra Chhorn, Rothna Pec, Vichhey Nall, Pinnara Ket, Chantha Oeurng, Chanthan Hel	2021	Assist agriculture individuals in reducing labor force and time, Increase yield production, Modernize traditional agriculture methods, Implement the smart irrigation system in greenhouses at Institute of Technology of Cambodia and Royal University of Agriculture	The smart irrigation system aims to reduce labor force and time to increase yield production, resulting in healthy crop growth with less effort and time requirement. It also has the potential to modernize traditional agriculture methods.	Zigbee communication protocol, Xbee s2c module, Wi-Fi module (Node MCU), ThingSpeak cloud platform, industrial-grade microcontroller, Human Machine Interface (HMI)	Advantages of the smart irrigation system include reducing labor force and time in agriculture, increasing yield production, and modernizing traditional agriculture methods. The system has led to healthy crop growth with less effort and time required.
IoT based smart irrigation monitoring and controlling system..[22]	Shweta Saraf, Dhanashri H. Gawali	2017	Develop an IoT-based smart irrigation system for remote monitoring and controlling, propose and evaluate a cloud-based wireless communication system for plant water needs assessment	The study proposes a smart irrigation system using an android phone for remote monitoring and control, enabling wireless monitoring of field irrigation and introducing a cloud-based communication system for plant water assessment.	Android application, Java graphical user interface	Advantages include convenience and efficiency in irrigation management, as well as accessibility and ease of use in monitoring and controlling irrigation systems.
A distributed system for supporting smart irrigation using Internet of Things technology. [23]	Ahmed Abdelmoamen Ahmed, Suhib Al Omari, R. Awal, A. Fares, M. Chouikha	2020	The study objectives are to design and implement a smart irrigation system using IoT technology, automate the irrigation process in agricultural fields, and create a better opportunity for farmers to irrigate their fields efficiently.	The smart irrigation system using IoT technology aims to improve irrigation efficiency for farmers through its three main components and was evaluated based on various performance metrics.	Microsoft Azure IoT Hub, Raspberry Pi 3 device, Android mobile app	The advantages of the smart irrigation system using IoT technology include automating the irrigation process in agricultural fields, leading to efficient irrigation, and eliminating under-watering that can stress the plants.

Hydroponic-based smart irrigation system using Internet of Things. [24]	K. E. Lakshmiprabha, C. Govindaraju	2019	To propose a smart irrigation system using IoT for a hydroponic system and demonstrate its capability to monitor and control the hydroponic environment for higher crop production.	The smart irrigation system using IoT successfully monitored and controlled the hydroponic environment, leading to increased crop production in a small area. Real-time data on temperature, humidity, and water flow was effectively monitored.	ThingSpeak IoT platform	The advantages of the smart irrigation system using Internet of Things (IoT) include higher crop yields in a small area and the capability to monitor and control the hydroponic environment, resulting in increased crop production.

2.10 Summary

The summary of the literature review highlights the in-depth exploration of the key components and technologies underpinning the development of the Smart Irrigation System with IoT-Based Rain Forecast Capability. It begins by examining the importance of IoT in creating interconnected systems, enabling real-time monitoring and control of various components. Essential hardware like microcontrollers ESP8266 and their integration with sensors and actuators are discussed, highlighting their role in collecting and processing environmental data. The review also delves into the power supply systems, including the use of regulated DC power and buck-boost converters for efficiency and stability.

The chapter covers the significance of rain sensors in detecting precipitation and their contribution to optimizing water usage by preventing over-irrigation. Display devices are explored as critical components for real-time system monitoring and user interaction. Additionally, the role of weather forecasting systems in providing predictive data is emphasized, showcasing their importance in adjusting irrigation schedules based on rainfall forecasts. Comparisons between technologies, such as Wi-Fi and cellular networks, further contextualize connectivity options for IoT applications.

Overall, this chapter establishes a comprehensive foundation for understanding the technologies and methodologies involved in the project, highlighting how each component contributes to achieving a sustainable and efficient irrigation system. By leveraging IoT, real-time data, and predictive insights, the system addresses challenges like water wastage and unpredictable weather, offering a robust solution for modern agricultural and landscaping needs.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In general, the methodology provides a detailed explanation of the systematic approach undertaken to design and implement the Smart Irrigation System with IoT-Based Rain Forecast Capability. This chapter outlines the step-by-step process used to develop the hardware and software components, aligning with the project's objectives of creating a connected, efficient, and sustainable irrigation system. The methodology ensures a comprehensive understanding of how each aspect of the system contributes to achieving real-time monitoring, rain forecasting, and automated water management.

Key elements of this chapter include the flowchart, which visually represents the system's overall workflow, and the process explanation, detailing each step in the operation of the smart irrigation system. The program flow outlines the logical structure of the software, ensuring seamless integration of the IoT sensors, actuators, and data processing algorithms. A review of the selected components, such as the ESP8266 microcontroller, rain sensors, and soil moisture sensors, highlights their suitability and functionality within the system.

The chapter also describes the program code development, focusing on creating robust software to enable efficient data handling and control. The Wi-Fi interface design explains how the system connects to the internet for remote monitoring and data transmission. Additionally, circuit design and system design illustrate the integration of hardware components, ensuring reliability and scalability. The chapter concludes with a summary, reflecting on how the methodology supports the objectives of optimizing water usage, integrating IoT-based rain forecasting, and enhancing overall irrigation efficiency.

3.2 Process Explanation

The development of the smart irrigation system followed a structured process to ensure efficiency and alignment with project objectives. The process began with analyzing problems in traditional irrigation systems and identifying the requirements for IoT integration, sensors, and automation. Suitable components, including the ESP8266 microcontroller, soil moisture sensor, rain sensor, relay module, and OLED display, were selected and tested for compatibility. The system's architecture was then designed, followed by assembling hardware, including the power supply and DC water pump, with careful circuit design. Software development involved programming the ESP8266 to process sensor data, control the pump, display system status, and enable IoT connectivity for remote monitoring and weather forecast integration. Comprehensive testing was conducted to validate the system's functionality and address any issues, ensuring reliable automation and water efficiency. Finally, the system was evaluated for performance, documented, and finalized for demonstration and practical application.

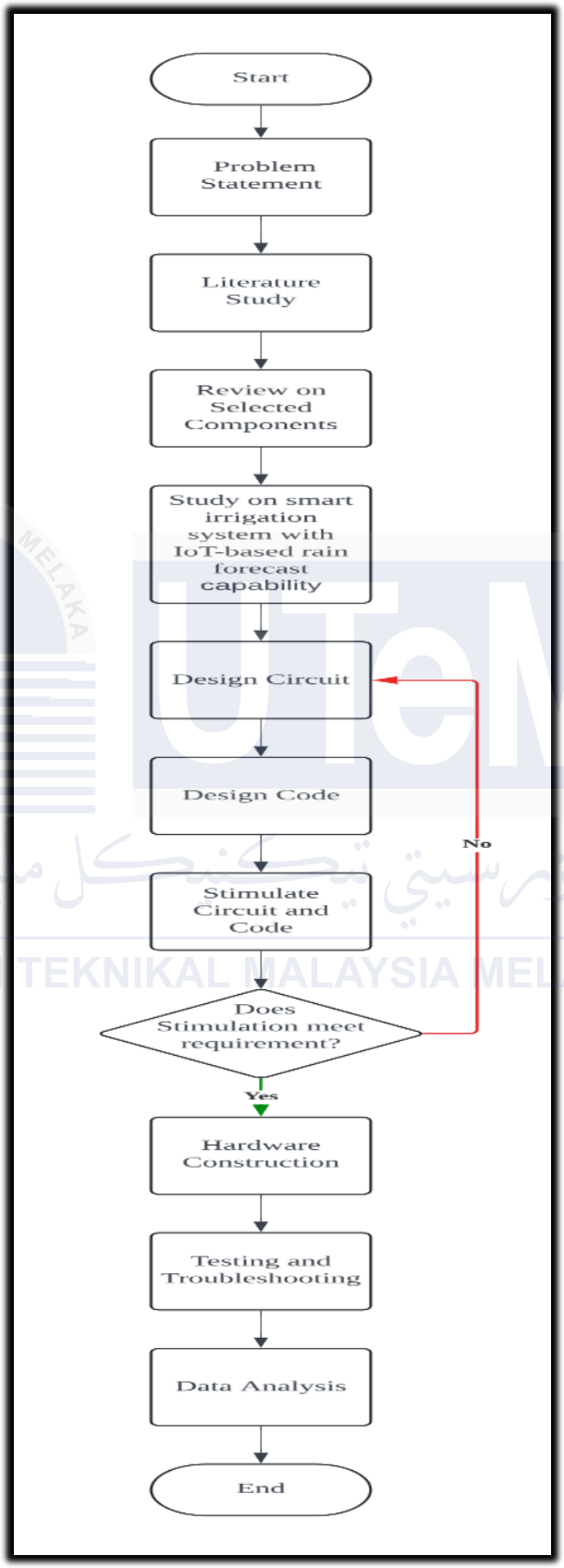

Figure 3.1 : Flowchart Project Order

A flowchart for a project order visually represents the sequence of steps involved in completing the project. It typically starts with the initiation of the project, followed by planning, execution, monitoring, and completion phases. Each step is represented by a specific symbol, such as ovals for start/end, rectangles for processes, diamonds for decisions, and arrows for flow direction. This method allows for easy tracking of project progress, ensuring that tasks are completed in the correct order. It helps to clarify the project workflow and identify potential bottlenecks or areas needing improvement as shown in Figure 3.1.

3.3 Hardware Development

3.3.1 Circuit Design

Circuit design in Cirkit Designer IDE involves creating a schematic that represents the components and their connections for an electronic system. The Cirkit Designer IDE allows to place and connect various electronic components like microcontrollers ESP8266**,** sensors soil moisture, rain sensor**,** and actuators relay module for the water pump. The design process typically includes selecting appropriate components, placing them on a grid, and connecting them using virtual wires to form the desired circuit. Cirkit Designer IDE offers tools for creating accurate circuit diagrams, checking for design errors, and running simulations to ensure the circuit will function as intended. The software helps you manage power distribution, signal flow, and component interaction in an intuitive way. It also offers features such as automatic routing for wire connections, visualizing power supply connections, and running simulations to test the behavior of the circuit before building it physically, ensuring optimal performance and reliability. The below Figure 3.2 shown are the web logo used for designing hardware component connection and Figure 3.3 web main page for creating the connection of the project.

Figure 3.2 Cirkit Designer Logo

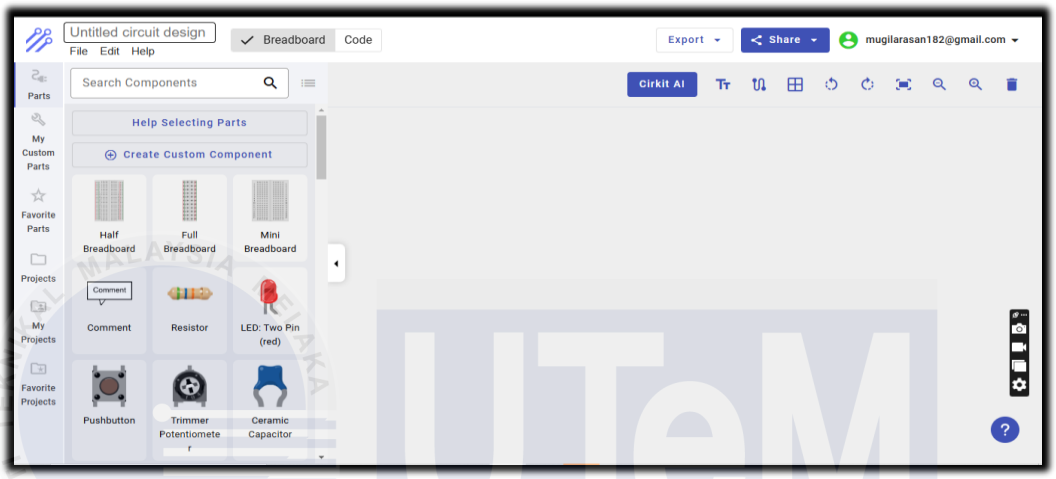

Figure 3.3 Project Design Component Connecter

3.3.1.1 ESP8266 Wi-Fi Module

The ESP8266 Wi-Fi module was selected for the smart irrigation system due to its versatility, efficiency, and suitability for IoT applications. It is a cost-effective microcontroller with built-in Wi-Fi capability, making it ideal for connecting devices to the internet without the need for additional modules. This integration aligns with the project's objective of creating a system that can remotely monitor and control irrigation while utilizing rain forecasts and real-time data.

The ESP8266 supports various communication protocols, enabling seamless interaction with sensors, actuators, and cloud platforms for data storage and processing. Its ability to handle simple web servers and send/receive data over a network makes it perfect for tasks like monitoring soil moisture levels and fetching weather forecast data. Furthermore, the

ESP8266's low power consumption ensures energy efficiency, an important factor in maintaining the system's sustainability. These features, combined with extensive community support and compatibility with the Arduino IDE, make the ESP8266 a practical and reliable choice for this project. The Table 3.1 shows the technical specification for the ESP8266.

Table 3.1 : ESP8266 Technical Specifications

Wi-Fi Parameters	Frequency Range	2.4G-2.5G (2400M-2483.5M)
	Wi-Fi Protocols	802.11 b/g/n
	Types of Antenna	PCB Trace, External, IPEX Connector, Ceramic Chip
Hardware Parameters	Peripheral Bus	UART/SDIO/SPI/I2C/I2S/IR Remote Control
		GPIO/PWM
	Operating Voltage	3.0~3.6V
	Operating Current	Average value: 80mA
	Operating Temperature Range	-40°~125°
	Ambient Temperature Range	Normal temperature
Software Parameters	Package Size	5x5mm
	Wi-Fi mode	station/softAP/SoftAP+station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network)
	Software Development	Supports Cloud Server Development / SDK for custom firmware development
	Network Protocols	IPv4, TCP/UDP/HTTP/FTP

3.3.1.2 Relay Module

The relay module was chosen for the smart irrigation system to act as an interface between the low-power control circuit and high-power devices, such as the water pump. A relay module allows the microcontroller, like the ESP8266, to control high-voltage or high-current devices safely and efficiently, which is essential for operating the irrigation system's hardware components.

Relays work by using a small control signal from the microcontroller to activate an electromagnetic switch, enabling or disabling the connection of the water pump to the power source. This isolation ensures that the sensitive components of the microcontroller are protected from high voltages and currents, preventing potential damage. The relay module's ability to provide precise on/off control is critical for automating the irrigation process, allowing the system to activate the pump only when necessary based on sensor readings or rain forecasts. This integration supports the project's goals of efficient water usage and sustainable operation.

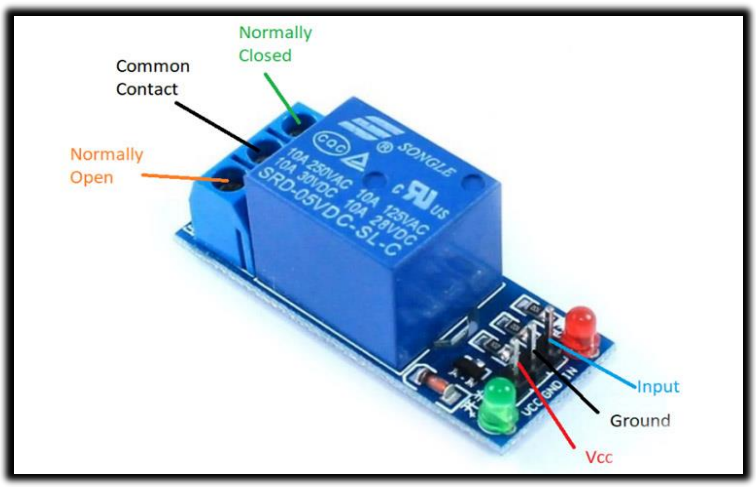

Figure 3.4 : Relay Module Pinout

A relay module typically has pins for power (VCC and GND) to supply the module, an input pin (IN) to control the relay state (ON or OFF), and one or more output pins for connecting the relay's contacts (NO, NC, and COM). The NO (Normally Open) pin connects to the load when the relay is activated, while the NC (Normally Closed) pin remains connected when the relay is off. The COM (Common) pin is the common terminal for switching between NO and NC pins as shown in Figure 3.4 and the Table 3.2 shows the specification of the relay module .

Table 3.2 : Relay Module Technical Specifications

TECHNICAL SPECIFICATION & SELECTION GUIDE

Parameters	SRL21CO	SRL41CO	SRL81CO
Output			
No. Of Channel	2	4	8
Contact Configuration	1 C/O		
Input			
Input Signal Voltage	12V / 24V DC		
Operate Voltage	8.4V / 16.8V DC		
Release Voltage	3.5V / 7.5V DC		
Max. Coil voltage	15.6V / 31V DC		
Coil Current Per Channel	33.3mA / 16.7mA		
Operate (SET) Time	Max. 15ms		
Release (Reset) Time	Max. 5ms		
General Specifications			
Durability	Electrical *3 (resistive load) : 30,000 operations at 24 VDC, 12 A / 50,000 operations at 250 VAC, 12 A		
Dielectric Strength	1. Between Coil & Contacts: 5000 VAC, 50/60 Hz, for 1 minute 2. Between Contacts of the same polarity: 1000 VAC, 50/60 Hz, for 1 minute		
Contact Rating	Relay : 10A@28VDC/230VAC Board : 5A@28VDC/230VAC		
Ambient Operating Temperature	-40°C to 85°C (with no icing or condensation)		
Ambient Operating Humidity	5% to 85% (with no icing or condensation)		
Termination	Coil Termination: Screw type, for 2.5mm sq. wire Contact Termination: Screw type, for 2.5mm sq. wire		
Mounting	35mm Din Rail		
Dimension (L x W x H) mm	42 x 88 x 63	72 x 88 x 63	132 x 88 x 63
Weight (g)	70	120	230

3.3.1.3 Dc Water Pump

A DC water pump was chosen for the project because of its efficiency, ease of control, and compatibility with the project's low-voltage power supply. DC pumps are compact, reliable, and ideal for applications requiring precise water flow control, such as irrigation. They are powered by direct current, which can be conveniently regulated using the system's 12V DC power supply and controlled through the relay module.

The pump's ability to operate on low voltage ensures safety and energy efficiency, making it suitable for IoT-based systems like this project. Additionally, its compatibility with microcontroller-driven systems, like the ESP8266, enables seamless integration for automation. The DC water pump can be easily activated or deactivated based on real-time data from the soil moisture sensors or rain forecasts, ensuring water is delivered only when necessary. This choice supports the project's objectives of optimizing water usage and creating a sustainable and effective smart irrigation system. The Figure 3.5 shows Dc Pump as below.

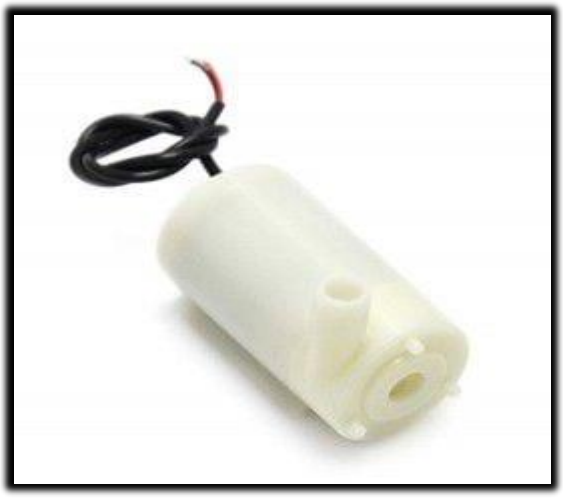

Figure 3.5 : Dc Water Pump 3V

3.3.1.4 Rain Sensor

Rain sensor was chosen for the project to detect and respond to rainfall in real time, ensuring efficient and adaptive water management. The sensor's ability to measure rain presence or intensity allows the system to automatically adjust irrigation schedules, preventing unnecessary watering during or after rainfall. This functionality directly aligns with the project's objective of optimizing water usage and avoiding waste.

By integrating the rain sensor with the ESP8266 microcontroller, the system can process data from the sensor and make informed decisions, such as pausing irrigation when rain is detected. This automation not only conserves water but also protects crops and landscapes from overwatering, which can lead to root damage or soil erosion. Additionally, the rain sensor contributes to the project's sustainability goals by reducing manual intervention and ensuring the irrigation system adapts to changing weather conditions efficiently.

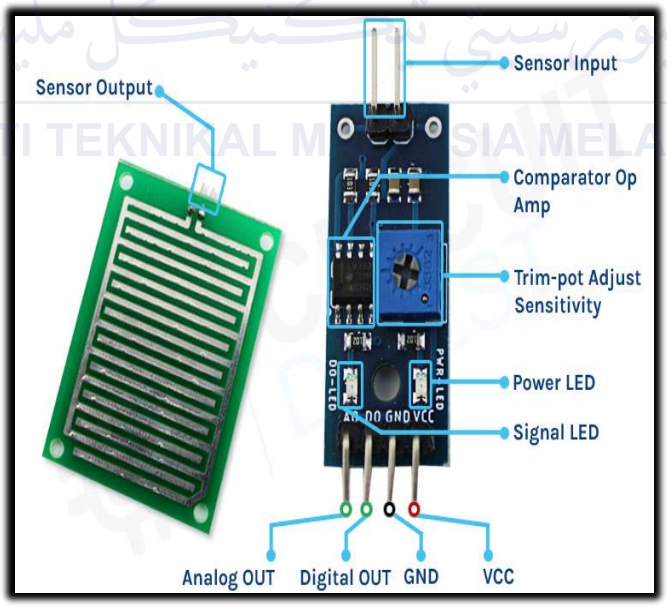

Figure 3.6 : Rain Sensor Pinout

The rain sensor typically consists of three main pins: VCC (for power), GND (for ground), and an output pin (SIG) that provides a signal based on rain detection. When rain falls on

the sensor's surface, it completes the circuit and the SIG pin sends a low signal (LOW), indicating rain. In dry conditions, the SIG pin remains high (HIGH). The VCC and GND pins are connected to the power supply to operate the sensor as shown in Figure 3.6 .

3.3.1.5 OLED LCD 12C Display Module

OLED display was selected for the project to provide a clear and energy-efficient way to visually present critical system information. The OLED is compact, lightweight, and offers high-contrast, crisp displays that are easy to read in various lighting conditions. This makes it ideal for showcasing real-time data such as soil moisture levels, rain detection status, pump activity, and network connectivity, ensuring users can monitor the system's performance at a glance.

The OLED's low power consumption is particularly advantageous for IoT projects, as it minimizes energy usage while maintaining reliable operation. Additionally, its compatibility with microcontrollers like the ESP8266 allows seamless integration into the system. Using the OLED enhances user interaction by delivering immediate feedback and system status updates, eliminating the need for external devices like smartphones or computers for basic monitoring. This functionality aligns with the project’s goals of creating a self-sustaining, user-friendly, and efficient irrigation system.

Figure 3.7 : OLED LCD 12C Display Module

The OLED LCD I2C display module typically has four pins: VCC (power supply), GND (ground), SDA (Serial Data Line for data transfer), and SCL (Serial Clock Line for clock synchronization). VCC and GND are used to power the module, while SDA and SCL communicate with the microcontroller via the I2C protocol. These pins allow for easy connection to devices like the ESP8266 or Arduino for display purposes. The I2C interface simplifies wiring by reducing the number of required connections as shown in the Figure 3.7 above.

3.3.1.6 Power Supply

The power supply system chosen for this project was selected for its reliability, efficiency, and compatibility with the components of the smart irrigation system. The use of a regulated DC power supply ensures that the system operates within safe and stable voltage levels, preventing damage to sensitive components like the ESP8266, sensors, relay module, and OLED display. Its ability to deliver consistent power is crucial for maintaining the system's performance and durability.

This power supply is particularly suitable because it supports the necessary current and voltage requirements of the components, including the DC water pump, which demands a steady and robust power source for operation. Additionally, the inclusion of a buck-boost step-down converter enhances the system's efficiency by regulating and optimizing voltage for different components. Compared to alternatives like batteries, this power setup offers a longer operational lifespan, reduces maintenance, and ensures uninterrupted functionality, making it ideal for a system that relies on continuous IoT connectivity and real-time automation. This aligns with the project's goals of creating a sustainable and dependable irrigation solution.

Figure 3.8 : 12v DC and Step Down Module

The 12V DC power supply typically has two pins: VIN for the positive input (12V) and GND for the ground. A step-down module (buck converter) converts this 12V to a lower output voltage, with VOUT providing the regulated output voltage (e.g., 5V or 3.3V). An Adjust pin (optional) is available to fine-tune the output voltage using a potentiometer as shown in Figure 3.8.

3.3.1.7 Soil Moisture Sensor Capacitive

A capacitive soil moisture sensor was chosen for the project due to its accuracy, reliability, and durability in measuring soil moisture levels. Unlike resistive sensors, capacitive sensors do not corrode over time, making them more suitable for long-term applications in agricultural or landscaping environments. The sensor works by measuring changes in soil dielectric permittivity, which correlates with moisture content, providing consistent and precise readings.

The importance of the capacitive soil moisture sensor lies in its ability to provide real-time data about the soil's water content, enabling the system to make informed irrigation decisions. By integrating this sensor with the ESP8266 microcontroller, the system can automate watering schedules, ensuring plants receive adequate water without overwatering or underwatering. This not only conserves water but also promotes healthy crop growth and prevents issues like root rot or soil erosion. Its role is essential in achieving the project's objectives of optimizing water usage and creating a sustainable smart irrigation system. The Figure 3.9 shown below is the soil moisture sensor.

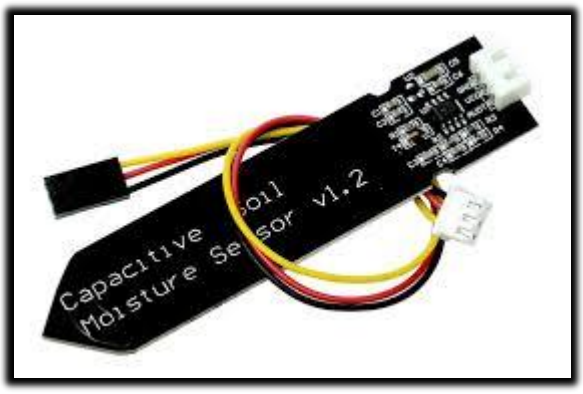

Figure 3.9 : Soil Moisture Sensor Capacitive

3.4 Program Code Development

The Arduino IDE (Integrated Development Environment) was chosen for program code development in this project due to its simplicity, versatility, and extensive community support. The IDE provides a user-friendly interface for writing, compiling, and uploading code to microcontrollers like the ESP8266, making it an ideal choice for developing IoT-based systems. It supports the C++ programming language with added libraries and functions that simplify the integration of hardware components and sensors.

Using the Arduino IDE allowed for efficient development and testing of the control algorithms and IoT functionalities required for the project. Libraries for the OLED display, Wi-Fi connectivity, and sensors were readily available, streamlining the coding process. Additionally, its serial monitor feature was invaluable for debugging and real-time observation of system behavior. The flexibility of the Arduino IDE ensured seamless implementation of features like data acquisition, rain forecasting integration, and automated irrigation control, aligning with the project's objectives to create an efficient and user-friendly smart irrigation system. The Figure 3.10 shows the main page of the code creator.

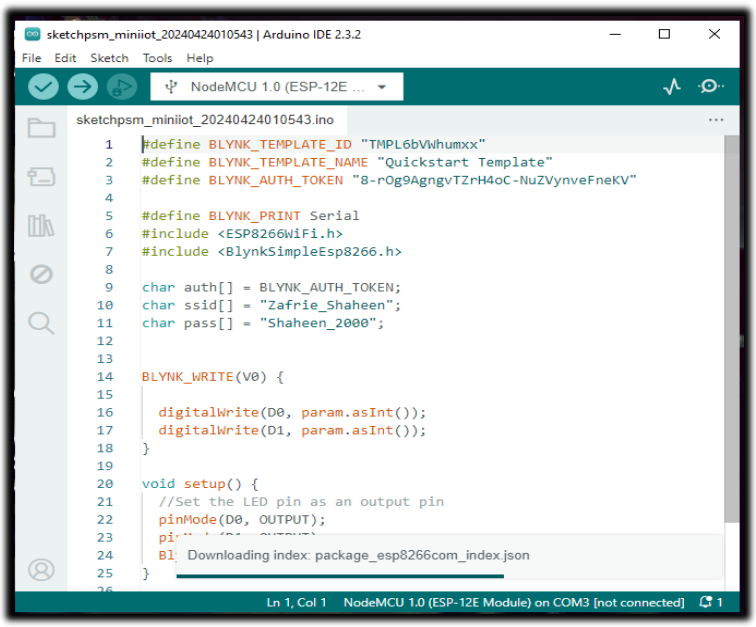

Figure 3.10 Arduino IDE Software

3.4.1 Integration of Firebase for Real-Time Data Management and IoT Connectivity

Firebase was used in this project as a powerful backend platform to store, manage, and retrieve data in real-time. It provides seamless integration with IoT devices and simplifies the process of connecting the ESP8266 microcontroller to a cloud-based database. Firebase's real-time database feature allows sensor data, such as soil moisture levels and rain status, to be uploaded, stored, and accessed instantly.

The integration of Firebase enabled remote monitoring and control of the smart irrigation system. For example, users can view real-time data and irrigation status on a mobile or web interface. Firebase also facilitated the implementation of dynamic features, such as adjusting irrigation schedules based on weather forecasts or historical data. Its robust and scalable infrastructure ensured reliable data handling, aligning with the project's objectives to optimize water usage and automate irrigation effectively. The Figure 3.11 shows the logo of Firebase Realtime Data.

Figure 3.11 Firebase Realtime Data

3.4.2 Android App Development

This project utilized Android Studio to develop a tailored mobile application, enabling a user-friendly interface for overseeing and controlling the smart irrigation system. As the official IDE for Android app development, Android Studio delivers a robust suite of tools and features that facilitate the efficient design, testing, and deployment of Android applications.

Using Android Studio, the app was designed to interact with Firebase for real-time data retrieval and updates, enabling users to monitor soil moisture levels, rain status, and irrigation activity remotely. The IDE's built-in design editor facilitated the creation of an intuitive user interface, ensuring ease of use for end-users. Additionally, Android Studio's compatibility with Java and Kotlin programming languages allowed the integration of IoT functionalities, such as sending control commands to the ESP8266 microcontroller. This app enhanced the system’s accessibility and functionality, supporting the project's goal of delivering an efficient and connected smart irrigation solution. The Figure 3.12 shows the main page for creating the app.

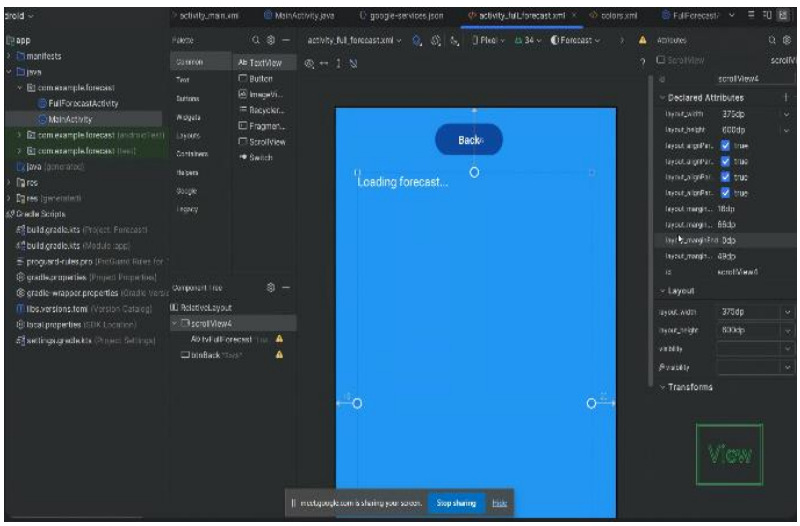

Figure 3.12 : Android Studio App Creator

3.4.3 OpenWeather API Integration

The OpenWeather API was seamlessly integrated into the project to provide real-time weather forecast data, enhancing the functionality of both the Android application and Firebase database. By fetching accurate weather information, including rain predictions and temperature, the system optimizes irrigation scheduling for improved water efficiency.

The Android app, developed using Android Studio, displays weather forecast data retrieved from the OpenWeather API, enabling users to monitor upcoming conditions directly on their mobile devices. This information is also stored in the Firebase real-time database, allowing the ESP8266 microcontroller to access the forecast for decision-making. For instance, when rain is predicted, the system can skip or adjust irrigation cycles automatically. The integration of OpenWeather API with Firebase and the Android app ensures a cohesive system that leverages real-time data for intelligent irrigation management, aligning with the project's goal of promoting sustainable water use through IoT technology.

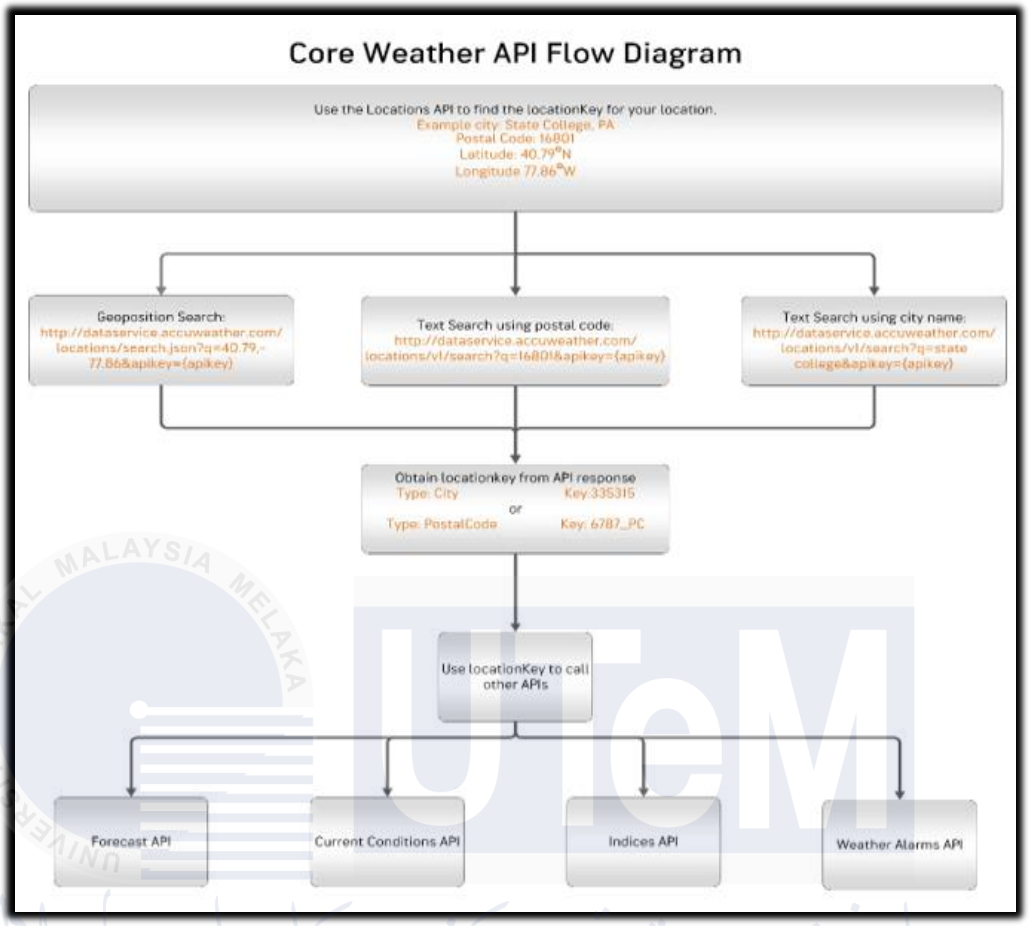

Figure 3.13 : API Flow Diagram

An API flow diagram typically starts with the Client sending a request to the API Endpoint. The API processes this request and interacts with the Database or other backend services if needed. Once the required data or action is retrieved or performed, the API sends a response back to the Client. This process is usually visualized in sequential steps showing request, processing, and response stages as shown in Figure 3.13 and the logo is shown in Figure 3.14 .

Figure 3.14 : Open Wheather API Logo

3.4.4 System Operation Algorythm

The system design for the Smart Irrigation System incorporates both hardware and software components to enable efficient, automated irrigation. The hardware consists of an ESP8266 microcontroller for Wi-Fi connectivity, soil moisture sensors to measure soil hydration levels, and a rain sensor to detect rainfall, preventing irrigation during wet conditions. These sensors provide real-time data to the ESP8266, which processes the inputs and controls the operation of a DC water pump through a relay module. An OLED display is used to show system status, such as soil moisture levels and pump activity, while a 12V DC power supply ensures that all components receive the necessary voltage. The system's hardware is designed to be energy-efficient, with the ESP8266 handling data processing and communication, while the relay and pump perform irrigation tasks based on sensor inputs.

On the software side, the project uses the Arduino IDE for programming the ESP8266, which connects to the Firebase real-time database for storing and retrieving sensor data, enabling remote monitoring via a mobile app. The system integrates weather forecasting data from the OpenWeather API to adjust irrigation schedules based on predicted rainfall, optimizing water usage. The Android app, developed using Android Studio, interfaces with

Firebase to display real-time data and control irrigation remotely. The system combines these software and hardware elements to automate irrigation, allowing users to monitor soil conditions, weather forecasts, and pump status from anywhere, ensuring efficient and sustainable water management

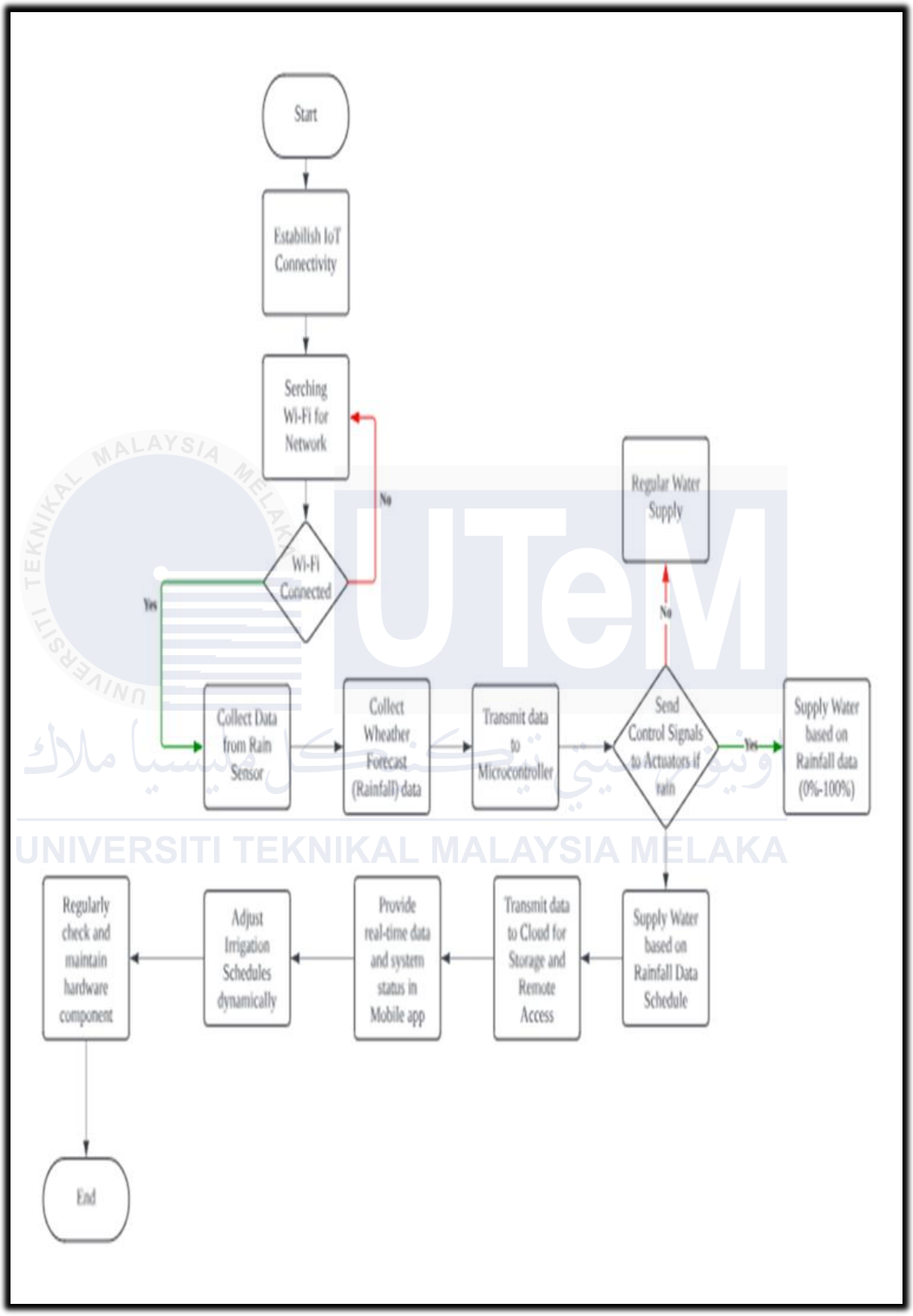

Figure 3.15: Smart irrigation system with IoT-based rain forecast System Flowchart

The IoT connectivity in the Smart Irrigation System begins with a microcontroller, such as the ESP8266, establishing an internet connection via Wi-Fi. The microcontroller collects data from sensors, including soil moisture and rain sensors, and processes it to assess irrigation needs. If the soil moisture level falls below the threshold and no rain is forecasted, the system activates the pump to water the crops. Conversely, when rain is predicted, irrigation is delayed or skipped to conserve water. All sensor data is uploaded to the cloud, enabling real-time monitoring and control through a mobile app or web interface for remote and efficient water management as shown in Figure 3.15 .

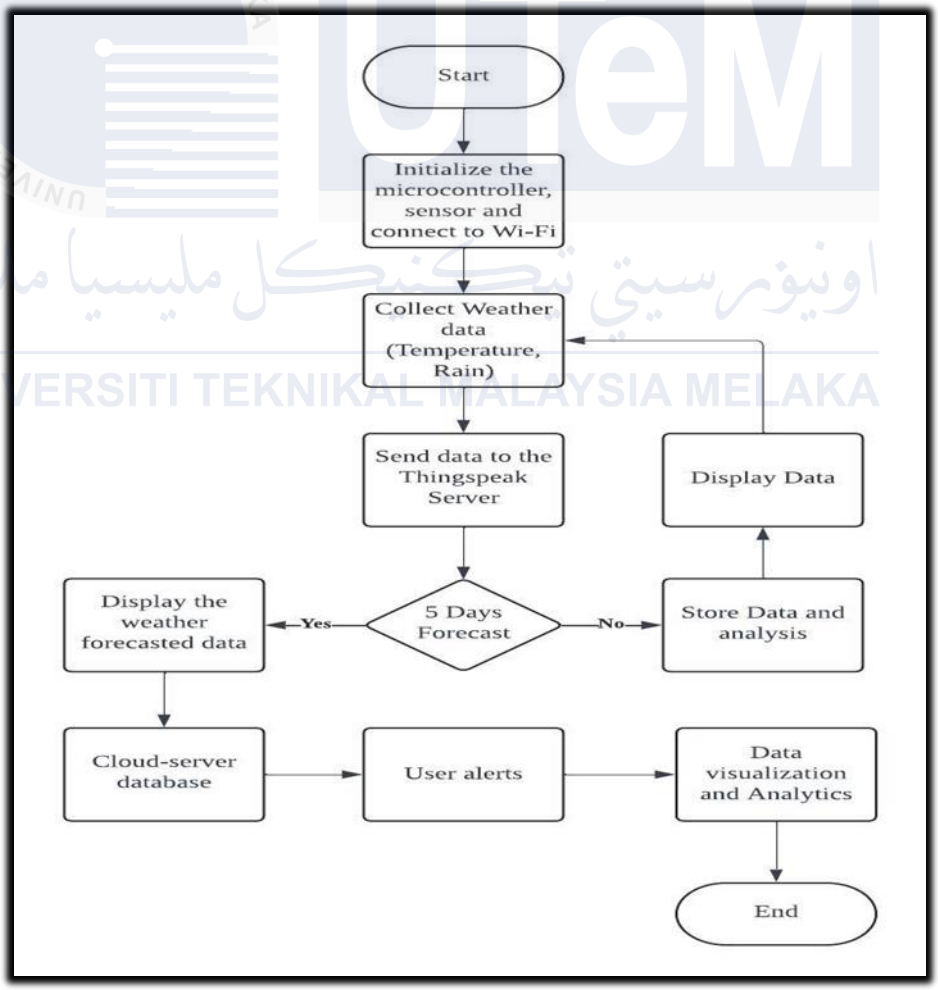

Figure 3.16 : Flowchart of Weather Forecasting

The Weather Forecasting system starts by retrieving real-time weather data from an external API, such as the OpenWeather API. The system checks for the forecasted rain conditions by analyzing the data, including parameters like temperature, humidity, and precipitation probability. If rain is forecasted within a specific time frame, the system records this information and stores it for decision-making. The rain sensor might also provide real-time input to validate the forecast. If rain is predicted, the irrigation system will bypass or adjust irrigation. The data is then communicated to the IoT system, which adjusts irrigation schedules accordingly, ensuring water conservation as shown in Figure 3.16 .

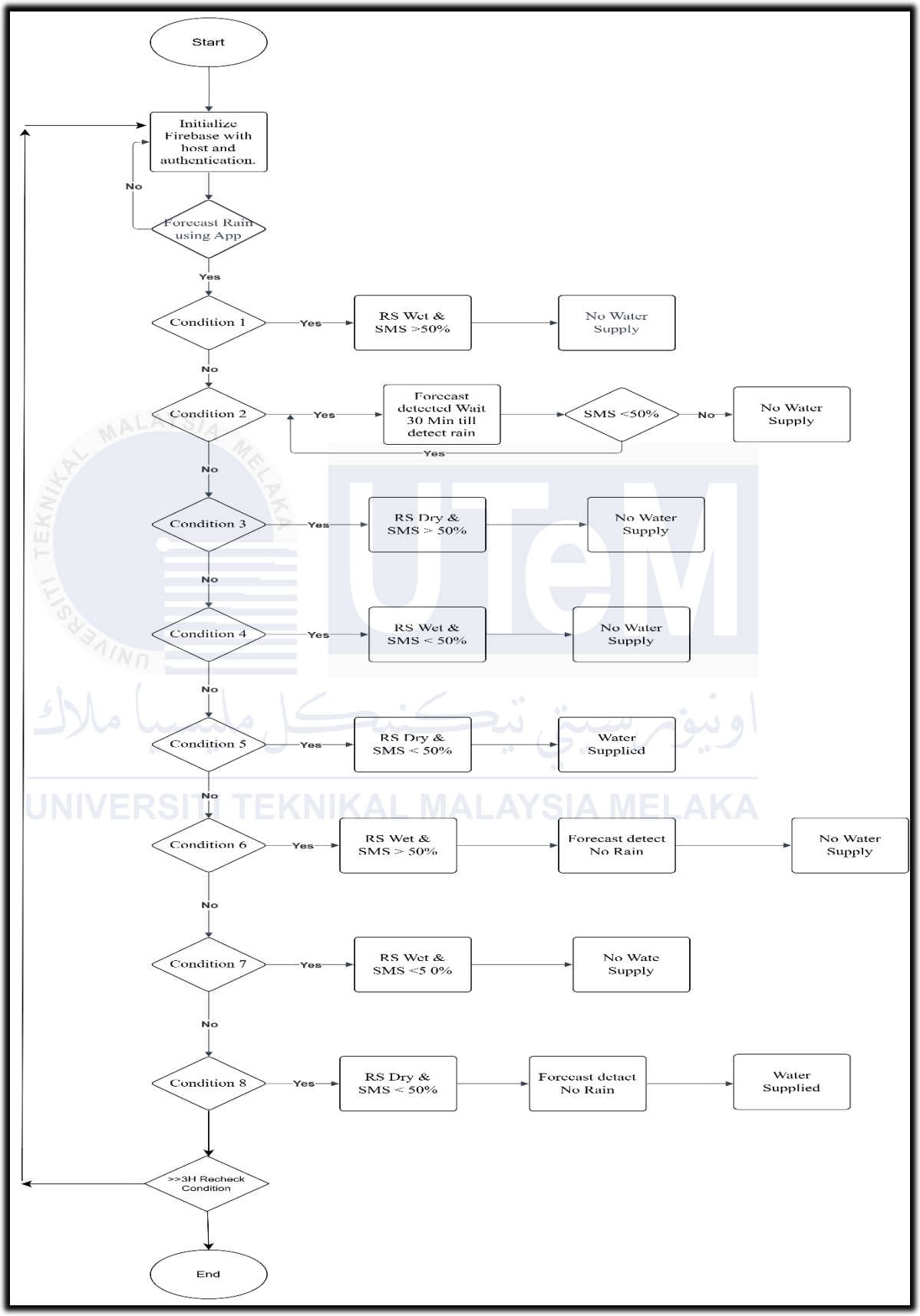

Figure 3.17 : Flowchart of Sensor work step under condition

The Flowchart of Sensor Work begins with the Soil Moisture Sensor (SMS) detecting the current moisture level of the soil. If the soil moisture is below the pre-set threshold, the system triggers the irrigation process by activating the water pump. The Rain Sensor (RS) is then checked to see if it detects any rainfall. If rain is detected, the system will override the irrigation trigger, preventing unnecessary water usage. If no rain is detected, the irrigation process continues as planned. After irrigation, both sensors update the system, ensuring that soil moisture levels and rainfall conditions are continuously monitored for efficient water usage. The data is communicated to the cloud system for real-time monitoring and can be accessed via the mobile app for user control as shown in the Figure 3.16

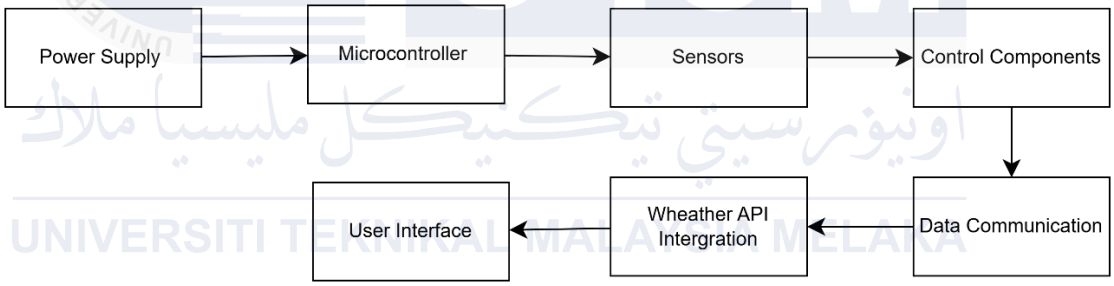

Figure 3.18 : Block Diagram Of the Flow Process

The flow process block diagram starts with a 12V power supply, which provides energy for the entire system. The power is regulated and delivered to the microcontroller (e.g., ESP8266 or ESP32), serving as the system's central processing unit. The soil moisture sensor (SMS) monitors soil moisture levels and transmits the data to the microcontroller, while the rain sensor (RS) detects rainfall and sends corresponding signals. Using input from these sensors, the microcontroller determines whether irrigation is needed and activates the pump via the relay module. Additionally, the system uploads sensor data to a cloud platform,

enabling real-time monitoring and remote control through a mobile app, ensuring effective and efficient water management as shown in Figure 3.18 and also in Figure 3.19 .

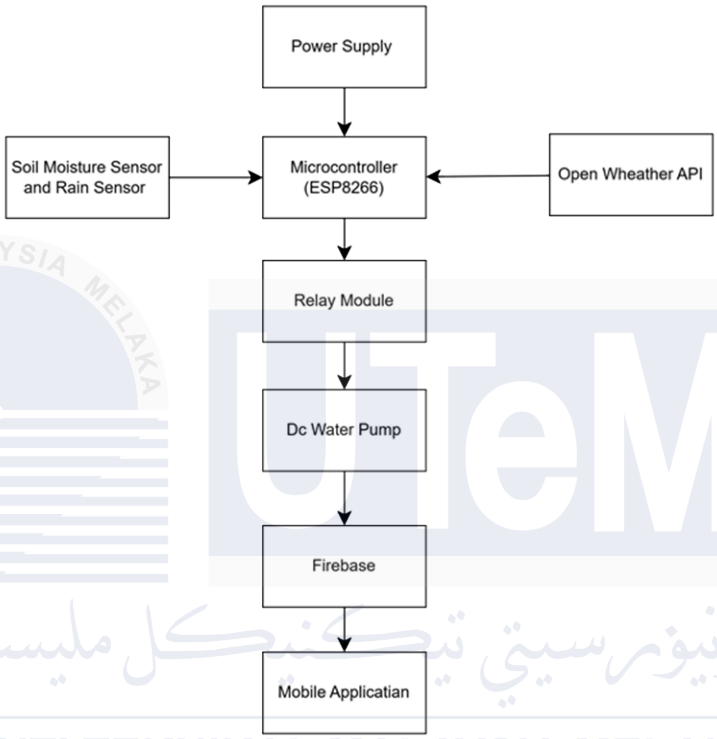

Figure 3.19 : Detailed Block Diagram

3.5 Summary

Chapter 3 outlines the methodology employed in the development of the Smart Irrigation System, detailing both hardware and software components. The process began with selecting appropriate components such as the ESP8266 microcontroller, soil moisture sensor, rain sensor, relay module, and DC water pump, which were integrated into a cohesive system. The hardware was designed to monitor soil moisture levels, detect rainfall, and control the irrigation process. The power supply and voltage regulation were also addressed to ensure reliable operation. On the software side, the Arduino IDE was used for programming the ESP8266, allowing for sensor data processing, control of the water pump, and communication with the Firebase database. The integration of the OpenWeather API enabled weather forecasting, providing data to adjust irrigation schedules based on predicted rainfall. A mobile application, developed using Android Studio, was created for remote monitoring and control of the system. The combination of these hardware and software elements resulted in an intelligent and efficient smart irrigation system.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

Chapter 4 presents the results and analysis of the Smart Irrigation System's performance. This chapter examines how the system functions in real-world conditions, evaluating the effectiveness of the sensors, control algorithms, and integration with weather forecasting data. The performance of the soil moisture sensor, rain sensor, and the accuracy of the weather data retrieved from the OpenWeather API will be discussed, alongside the system's ability to adjust irrigation schedules based on these inputs. Additionally, the mobile application and Firebase integration will be evaluated for real-time monitoring and remote control functionality.

The analysis will focus on the system's ability to optimize water usage, prevent over-irrigation, and ensure the sustainability of the irrigation process. The chapter will also address any challenges encountered during the implementation, such as connectivity issues or sensor calibration, and provide insights into potential improvements. The results will offer a clear understanding of how the system performs in various environmental conditions and contribute to the overall goal of developing a smart, IoT-based irrigation solution.

A summary will conclude the chapter, recapping key findings and highlighting the contributions of the system to efficient water management.

4.2 Results and Analysis

4.2.1 Hardware Performance and Analysis

The hardware performance and analysis of the project are evaluated based on the functionality and accuracy of the key components. The ESP8266 microcontroller was tested for its Wi-Fi connectivity and ability to process sensor data effectively. The system's response time in retrieving data from the soil moisture sensor and rain sensor was assessed, ensuring that the system could make real-time decisions regarding irrigation. The soil moisture sensor provided reliable readings, accurately detecting changes in soil hydration levels, while the rain sensor performed well in detecting rainfall, ensuring that irrigation was suspended when necessary.

The relay module and DC water pump were also tested to verify that the water flow was controlled precisely based on sensor inputs. The relay was able to turn the water pump on or off based on moisture levels and rain detection, ensuring optimal water usage. The OLED display effectively presented real-time system information, such as soil moisture levels and operational status, offering a clear and accurate overview for users.

The power supply system was analyzed to ensure stable voltage for all components, with the buck-boost converter providing the necessary power regulation to prevent over-voltage or under-voltage conditions. The overall hardware setup was able to operate efficiently, with minimal power consumption, and showed reliable performance in various environmental conditions. The performance of the hardware elements was consistent, and the system demonstrated the ability to optimize irrigation and water management effectively.

4.2.1.1 Constructed Hardware Setup

Before completing the hardware setup for the Smart Irrigation project, reference was made to a constructed hardware prototype to ensure proper assembly and functionality. This involved studying similar systems or prototypes to understand the optimal layout, wiring methods, and component integration needed for efficient operation.

Referring to a constructed hardware model provided valuable insights into potential challenges, such as avoiding wiring errors, ensuring stable power distribution, and organizing components to maintain neatness and accessibility. This step helped in minimizing mistakes during the hardware construction phase and ensured a reliable and functional setup that met the project's objectives as shown in Figure 4.1 .

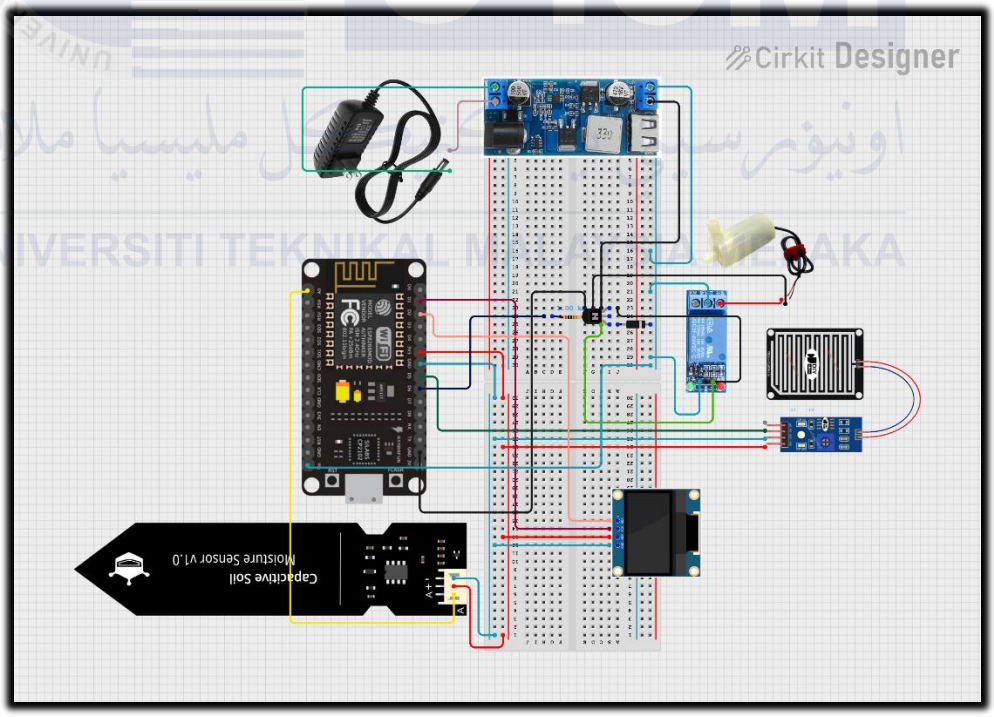

Figure 4.1 : Constructed Hardware System

To construct the components for the project, used Cirkit Designer, an online tool that offers a user-friendly platform to design and simulate electronic circuits. The first step in the process was selecting the necessary components for the project, such as the ESP8266 Wi-Fi module**,** soil moisture sensor**,** rain sensor**,** relay module**,** DC water pump**,** and OLED display. The Cirkit Designer library made it easy to drag and drop the components onto the design workspace.

Once the components were placed in the workspace, began wiring them according to the project specifications. The software allowed to connect the components visually, ensuring that power and data were routed to the correct pins. For example, the soil moisture sensor and rain sensor were linked to the appropriate input pins on the ESP8266, while the relay and pump were connected to output pins for control. The graphical interface of Cirkit Designer made it easier to organize and align the connections.

In addition to the basic circuit design, utilized the simulation feature of Cirkit Designer to test the functionality of my design. This allowed me to verify the connections before building the physical circuit. Any errors in wiring could be identified and corrected, ensuring a more efficient design process. The simulation feature helped ensure that components like the relay would correctly turn the pump on or off based on the soil moisture and rain detection.

4.2.1.2 Complete Hardware Setup

The wiring for the Smart Irrigation System has been successfully completed, ensuring seamless connections and reliable communication across all components. The setup was carefully designed to avoid issues like short circuits and to maintain stable power distribution throughout the system. With the wiring in place, the system is fully operational and ready for testing and evaluation. This step ensures that all elements function together as intended, supporting the project's goal of achieving efficient and automated irrigation as shown in Figure 4.2 .

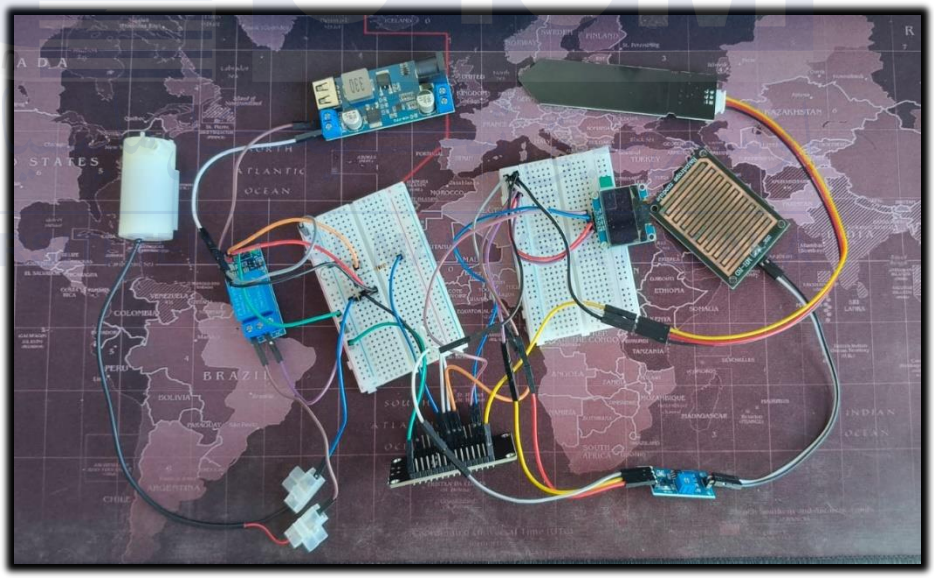

Figure 4.2 : Complete Connection of Hardware

After completing the circuit design using Cirkit Designer, proceeded with the physical wiring for the project. This stage involved translating the schematic into real-world connections between the components. Started by connecting the ESP8266 Wi-Fi module to the necessary pins on the breadboard, ensuring that the data and power lines were correctly aligned. The soil moisture sensor and rain sensor were wired to the input pins of the ESP8266, allowing the microcontroller to monitor the soil moisture levels and rain detection status.

Next, focused on the relay module and DC water pump. The relay module acts as a switch that is controlled by the ESP8266, and it was connected to an output pin on the microcontroller. The DC water pump was then connected to the relay's normally open (NO) terminal, ensuring that it could be powered on or off based on the relay's state. The OLED display was wired to the appropriate communication pins on the ESP8266 to allow it to display sensor readings and system status, including the rain sensor's status and the soil moisture levels.

For power, used a 12V DC power supply, ensuring that it was properly connected to the power input rails of the breadboard. Also included the necessary ground GND connections to complete the circuit and ensure all components shared a common reference. The wiring was carefully organized to prevent shorts and ensure stable operation.

Once all the components were physically wired together, tested the circuit to verify that the components responded correctly. The rain sensor was tested to ensure it correctly detected moisture, triggering the pump to turn off when it was raining. Similarly, the soil moisture sensor was checked to confirm that it could accurately measure soil moisture and trigger the pump to turn on when moisture levels were low. With the wiring complete, proceeded to

further test the system's functionality, adjusting any connections as needed for optimal performance.

4.2.2 Functionality of the Sensors Setup

4.2.2.1 Rain Sensor Waiting Rain : No Setup

The rain sensor detects whether the environment is "dry" or "wet" based on the presence of rainfall. This information is then displayed on the OLED screen with clear labels, such as "Rain Status: Dry" or "Rain Status: Wet."

If the display shows "Rain: Dry," it indicates no rain has been detected, and the system automatically activates the water pump to irrigate the soil as needed. Conversely, if the display shows "Rain: Wet," it means rainfall has been detected, and the system turns the pump off as shown in Figure 4.3, Figure 4.4, Figure 4.5 and Figure 4.6 .

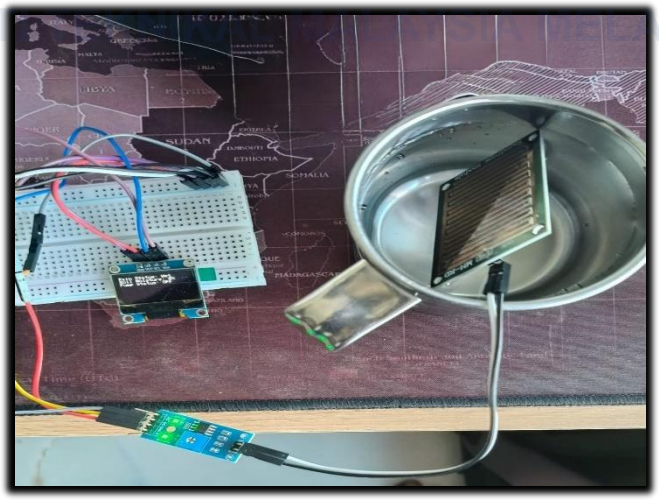

Figure 4.3 : Rain is Detected Pump Off Mode

Figure 4.4 : Oled Status Monitoring

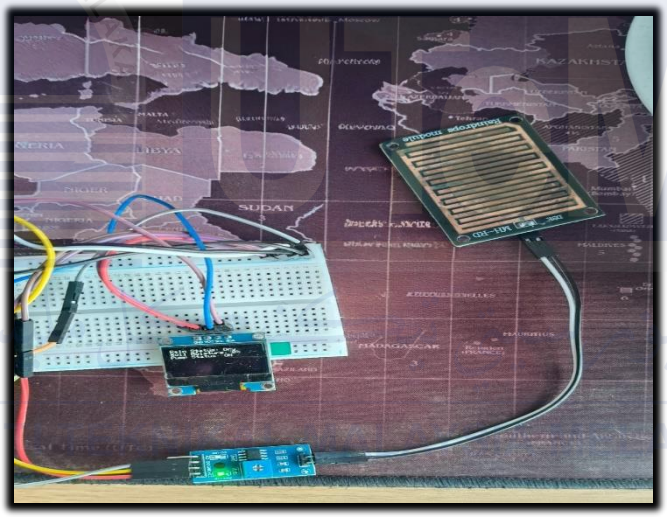

Figure 4.5 : Rain not Detected Pump On Mode

Figure 4.6 : Oled Status Monitoring

4.2.2.2 Rain Sensor Waiting Rain : Yes Setup

The OLED display in the Smart Irrigation System incorporates both real-time rain detection and weather forecast information, ensuring intelligent irrigation decisions. When the system retrieves forecast data indicating rain is expected within a specific time period, the OLED will display "Waiting Rain: Yes", regardless of the current state of the rain sensor. This override mechanism prioritizes forecasted rain to prevent unnecessary water usage.

In this scenario, even if the rain sensor detects "Dry," the system refrains from activating the pump because the forecast predicts imminent rainfall. Conversely, if no rain is forecasted, the OLED will display "Waiting Rain: No," and the system will operate based on the rain sensor's real-time readings. This dual-layered approach ensures optimal water management, combining forecast data and real-time sensor inputs to make precise irrigation decisions, avoiding both over-irrigation and water wastage as shown in Figure 4.7, Figure 4.8 and Figure 4.9 .

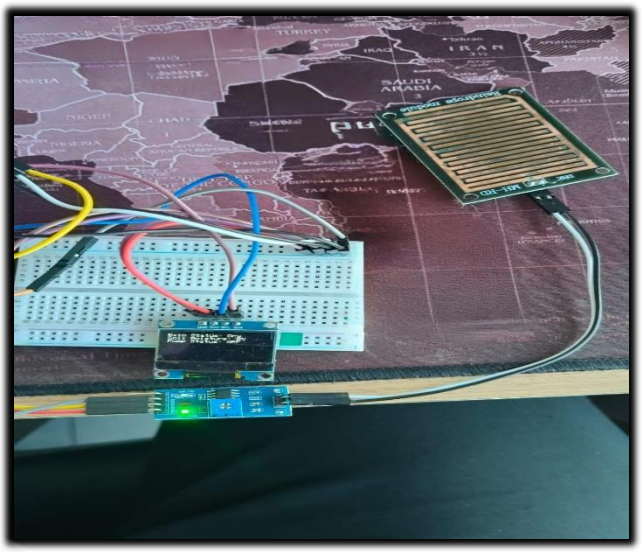

Figure 4.7 : Weather Forecast Rain Detect override

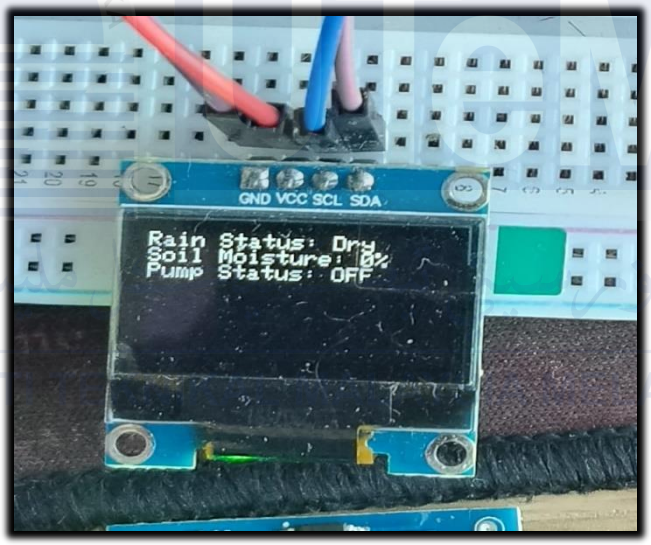

Figure 4.8 : Oled Status Monitoring Override

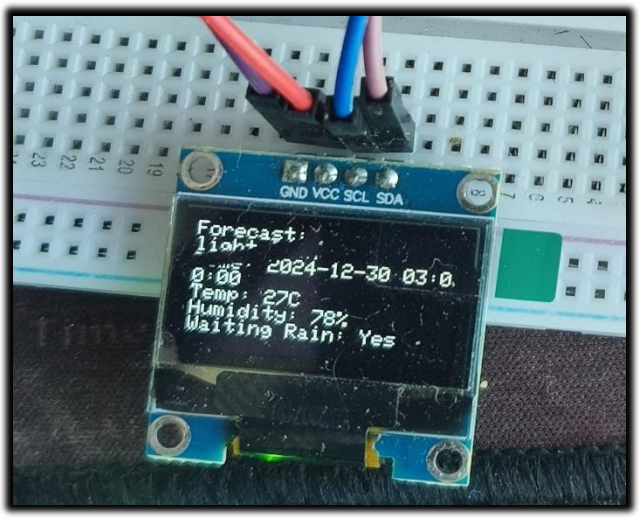

Figure 4.9 : Oled Status Detect Rain Forecast

4.2.2.3 Soil Moisture Sensor Without Forecast Setup

The OLED display in the Smart Irrigation System provides real-time updates on soil moisture levels, calculated from raw analog readings obtained by the soil moisture sensor. These readings are processed by the microcontroller, with a value of 600 or higher indicating low moisture and prompting the pump to irrigate the soil. Conversely, a reading of 599 or lower signifies adequate moisture, causing the system to deactivate the pump to prevent overwatering. This threshold-based mechanism ensures efficient water usage while maintaining optimal soil hydration. The OLED display enables users to monitor soil conditions and system actions clearly, supporting effective and sustainable irrigation management as shown in Figure 4.10, Figure 4.11 , Figure 4.12 and Figure 4.13 .

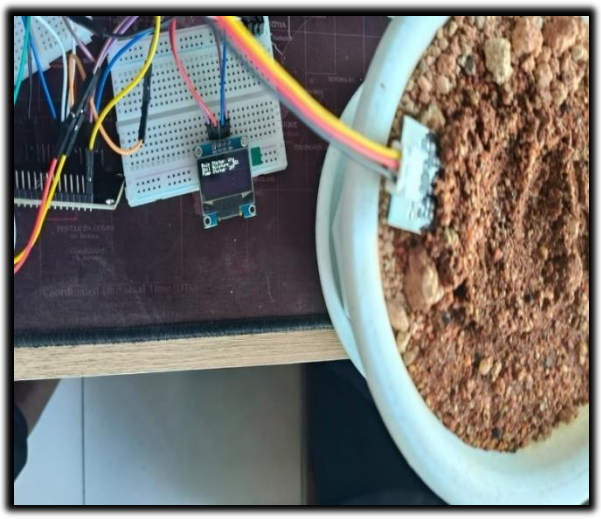

Figure 4.10 : Soil Moisture Detect Water

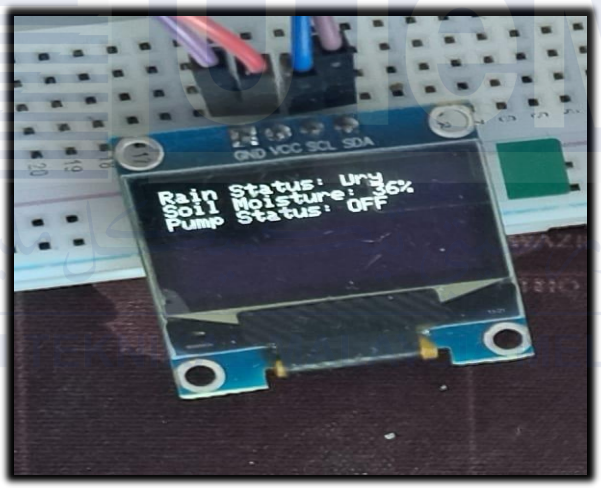

Figure 4.11 : Oled Status Reading

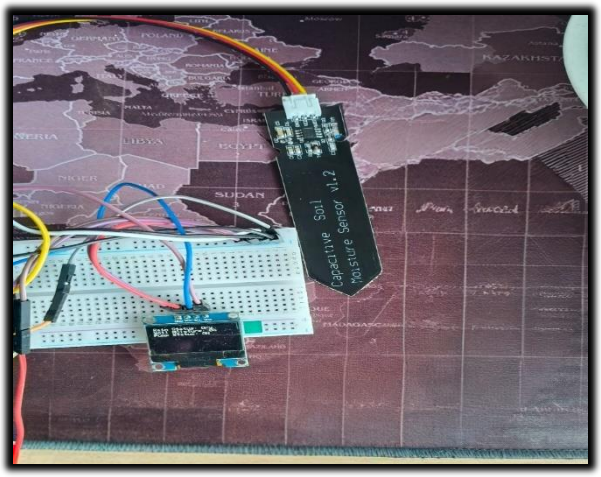

Figure 4.12 : Soil Moisture Detect No Water

Figure 4.13 : Oled Status reading

4.2.2.4 Soil Moisture Sensor With Forecast Setup

In the Smart Irrigation System, the OLED display integrates soil moisture data and rain forecast information to optimize irrigation decisions. Normally, the system monitors soil conditions to determine whether the pump should be activated or deactivated. However, when the forecast predicts rain within a specific time period, the system prioritizes the forecast and overrides the soil moisture data as shown in Figure 4.14 and Figure 4.15 .

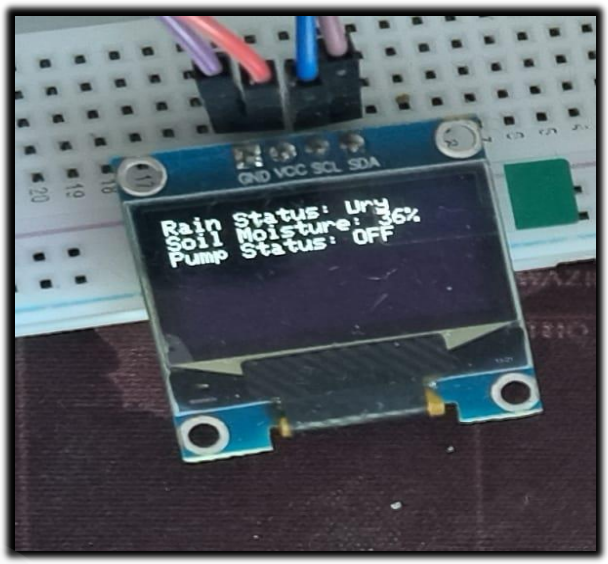

Figure 4.14 : Oled Override Reading

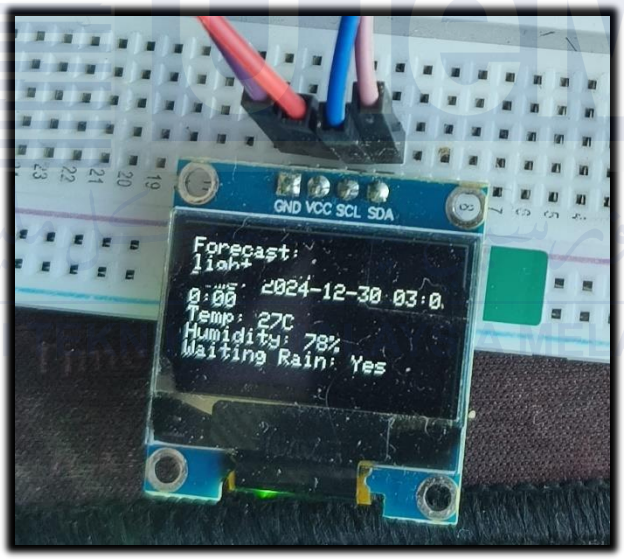

Figure 4.15 : Oled Monitoring Rain Detect

Table 4.1 : Different between with and without rain

Condition	Soil Moisture Level	Rain Detected	Pump Status
Without Rain	550	No	Off
Without Rain	650	No	On
Without Rain	625	Yes	Off
With Rain	525	Yes	Off
With Rain	625	Yes	Off

When the rain sensor detects wet conditions or rain is forecasted, the system turns the pump off, assuming the soil is sufficiently wet. If the soil moisture is below the set threshold and no rain is expected, the pump turns on to water the plants. In manual mode, the user can control the pump directly, overriding the system's automatic function. The rain sensor and soil moisture readings ensure the pump operates only when necessary, conserving water. The OLED display shows the rain status ("Wet" or "Dry") and the pump's current status ("ON" or "OFF") as shown in the Table 4.1 .

4.2.3 Results and Analysis of Software Development

4.2.3.1 Mobile App Development

The mobile application for the project was developed using Android Studio, a comprehensive platform for creating Android applications. The development process began with designing an intuitive user interface (UI) to ensure ease of use. The app's interface includes features such as displaying soil moisture levels, rain sensor statuses, forecasted weather information, and the pump's operational status. This layout was carefully crafted to provide real-time data and control options to the user.

The app's backend was integrated with Firebase, enabling seamless communication between the mobile app and the hardware system. Firebase facilitated the storage and retrieval of real-time data, such as sensor readings and weather updates, ensuring the app remains synchronized with the system. Additionally, the app utilizes the OpenWeather API to fetch accurate rain forecast data, which influences irrigation decisions. Overall, the mobile application was developed to serve as a centralized hub for monitoring and managing the

Smart Irrigation System, enhancing its functionality and user accessibility. The Figure 4.16 shows the logo of the app and the other Figure 4.17, Figure 4.18, Figure 4.19 and Figure 4.20 shows the setup in mobile phone, main page of the page, Forecast data and also the realtime reading of the sensors.

Figure 4.16 : Logo for Mobile App

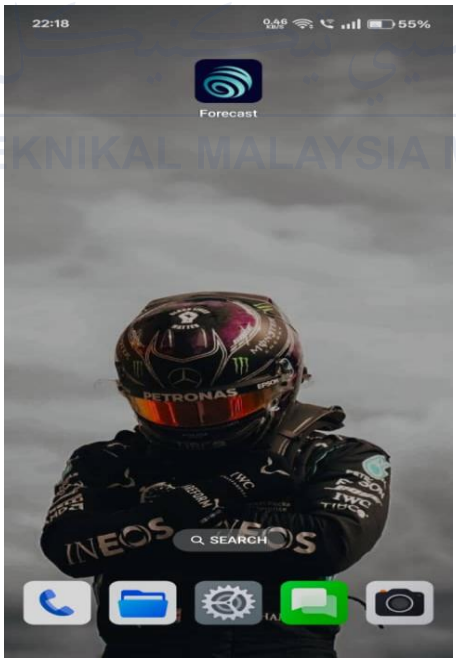

Figure 4.17 : App in the Mobile Phone

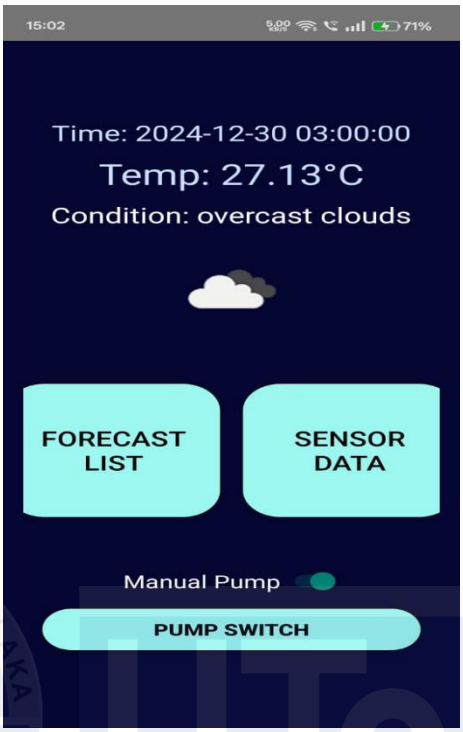

Figure 4.18 : Front App after opening

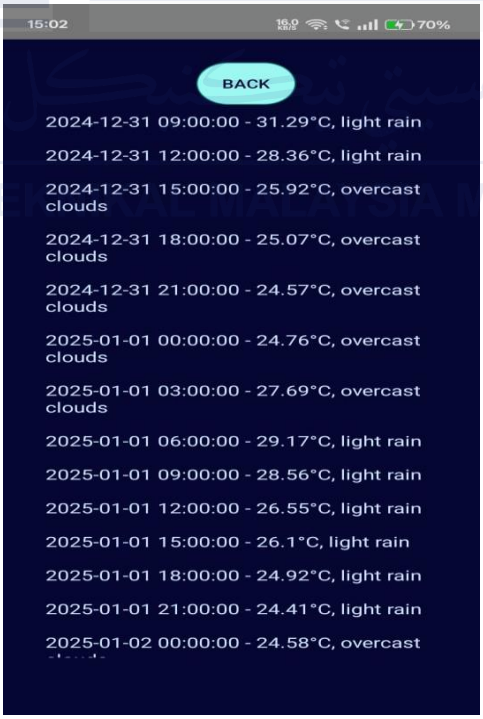

Figure 4.19 : Forecast Data Present from App

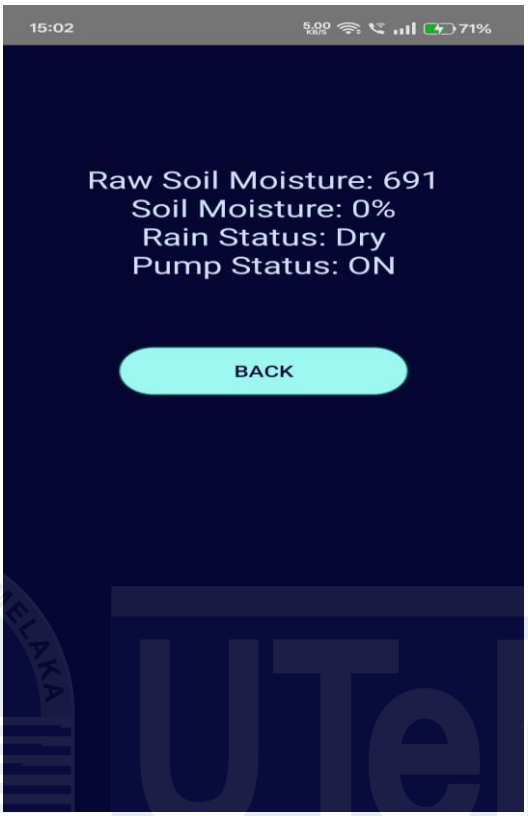

Figure 4.20 : Sensor Reading from Firebase

4.2.3.2 Project Code Development

The project coding development was done using Arduino IDE for programming the ESP8266 microcontroller. The code integrated soil moisture and rain sensors with weather forecast data from the OpenWeather API. It processes sensor readings to control the water pump based on soil moisture levels and forecasted rain. Additionally, the code ensures real-time data synchronization with the mobile app through Firebase, allowing users to monitor and control the system remotely. This approach ensures efficient irrigation management and seamless communication between the system's hardware and software components.

4.2.3.3 Explanation Code

This code snippet includes the necessary libraries required for various functionalities in the project. The #include <Wire.h> library is used to enable I2C communication, a protocol for connecting multiple devices using just two wires (SDA and SCL), which is crucial for interfacing with devices like the OLED display. The #include <Adafruit_GFX.h> library provides graphical primitives, such as drawing shapes and text, that are essential for creating visuals on displays. The #include <Adafruit_SSD1306.h> library is specific to controlling OLED displays using the SSD1306 driver, enabling detailed and flexible control over the display. The #include <FirebaseESP8266.h> library facilitates communication with Firebase from the ESP8266, allowing data exchange between the microcontroller and the Firebase Realtime Database. Lastly, the #include <ArduinoJson.h> library simplifies handling JSON data, which is often used to parse and manage structured information like API responses. Together, these libraries form the backbone for the hardware's communication, data visualization, and cloud integration as shown in Figure 4.21 .


```
1   #include <Wire.h>
2   #include <Adafruit_GFX.h>
3   #include <Adafruit_SSD1306.h>
4   #include <FirebaseESP8266.h>
5   #include <ArduinoJson.h>
```

Figure 4.21 : Preprocessor Directives Compiler

The code snippet checks if rain is forecasted within the next two hours by comparing the forecast timestamp with the current time. If the forecast time is within this window, it then checks the weatherID to see if it corresponds to rain or precipitation, such as thunderstorms, light rain, drizzle, or heavy showers (defined by specific weatherID ranges like 200-232, 300-321, and 500-531). If these conditions are met, the system sets the waitingForRain

variable to true, indicating that the system is anticipating rain. The water pump is then turned off by setting the pumpPin to LOW, and the pumpStatus variable is updated to false to reflect that the pump is not active. A message is printed to the serial monitor to notify that the system is in "forecast mode" and waiting for rain. The function then exits early to prevent any further actions, allowing the system to wait for the predicted rain before taking any further steps in the irrigation process as shown in Figure 4.22 .

```
192   // Check if forecast predicts rain within two hours
193   if (forecastTimestamp >= currentTime - twoHours && forecastTimestamp <= currentTime) {
194     if ((weatherID >= "200" && weatherID <= "232") ||
195         (weatherID >= "300" && weatherID <= "321") ||
196         (weatherID >= "500" && weatherID <= "531")) {
197       waitingForRain = true;
198       digitalWrite(pumpPin, LOW);
199       pumpStatus = false;
200       Serial.println("Forecast mode: Waiting for rain, Pump OFF");
201       return;
202     }
203   }
```

Figure 4.22 : Code for the Rain Forecast

The code checks the soil moisture and rain conditions to control the irrigation pump. First, it checks if it is raining (isRaining is true). If this condition is met, the system turns the pump off by setting pumpPin to LOW and updates the pumpStatus to false, then prints a message to the serial monitor indicating that the pump is off due to rain. If it is not raining, the code proceeds to check the soil moisture level (rawSoilMoisture). If the moisture level is above the defined threshold (moistureThreshold), it indicates the soil is dry, so the system turns the pump on by setting pumpPin to HIGH and updates pumpStatus to true, followed by printing a message that the soil moisture is low and the pump is on. If the soil moisture is below the threshold, indicating sufficient moisture, the pump is turned off again (pumpPin to LOW and pumpStatus to false), and a message is printed indicating that the pump is off because the soil moisture is sufficient. If none of the conditions are met (which should not happen

under normal circumstances), the system defaults to turning the pump off and prints an error message. This logic ensures the irrigation system operates based on weather and soil conditions, optimizing water usage as shown in Figure 4.23 .

```
208   // Soil moisture and rain detection logic
209   if (isRaining) {
210     digitalWrite(pumpPin, LOW);
211     pumpStatus = false;
212     Serial.println("Normal mode: It's raining, Pump OFF");
213   } else if (rawSoilMoisture >= moistureThreshold) {
214     digitalWrite(pumpPin, HIGH);
215     pumpStatus = true;
216     Serial.println("Normal mode: Soil moisture low, Pump ON");
217   } else if (rawSoilMoisture < moistureThreshold) {
218     digitalWrite(pumpPin, LOW);
219     pumpStatus = false;
220     Serial.println("Normal mode: Soil moisture sufficient, Pump OFF");
221   } else {
222     digitalWrite(pumpPin, LOW);
223     pumpStatus = false;
224     Serial.println("Error: No valid condition met. Pump OFF");
225   }
226 }
```

Figure 4.23 : Pump Control Logic Based on Rain Detection and Soil Moisture Levels

The function sendToFirebase() sends sensor data to a Firebase Realtime Database. It begins by using the Firebase.setString() method to send the rain status (isRaining) as either "Wet" or "Dry" based on the value of the isRaining variable. Then, it sends the pump status (pumpStatus) as either "ON" or "OFF" depending on the state of the pump using the same Firebase.setString() method. Next, the function sends the raw soil moisture data (rawSoilMoisture) and the processed soil moisture value (soilMoisture) using Firebase.setInt(), which stores integer values in Firebase. Additionally, it sends the current mode of the system (manualMode) and the manual control state for the pump (manualPump) as strings. Each of these data points is uploaded to the Firebase database under the path sensorData to allow remote monitoring of the system's status. This function is essential for

real-time data collection and communication between the hardware and the Firebase database, enabling cloud-based monitoring and control.

4.2.4 Data Collection Sensor Reading

4.2.4.1 Soil Moisture Levels with Pump Status

Table 4.2 : Raw Soil Moisture Data

Timestamp	Raw Soil Moisture	Pump Status
17/12/2024 09.00	640	OFF
17/12/2024 00.00	550	ON
17/12/2024 15.00	625	OFF
17/12/2024 18.00	525	ON
18/12/2024 06.00	524	ON
18/12/2024 09.00	615	OFF
18/12/2024 00.00	515	ON

In this experiment, the Raw Soil Moisture value is checked against the threshold of 600. When it is greater than or equal to 600, the pump is OFF. When it falls below 600, the pump turns ON the results are shown in Figure 4.24 and Figure 4.25 .

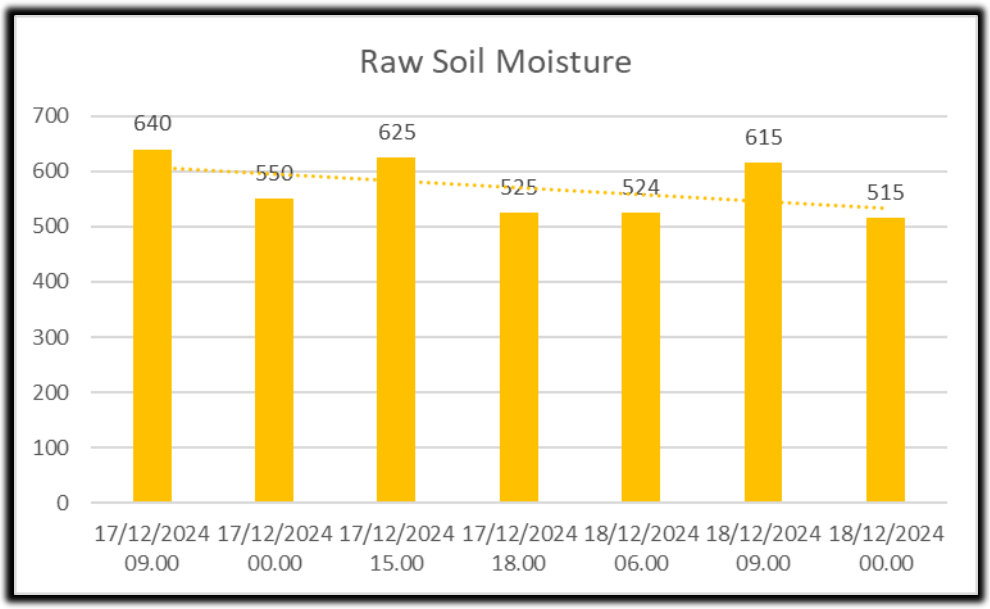

Figure 4.24 : Raw Soil Moisture Data

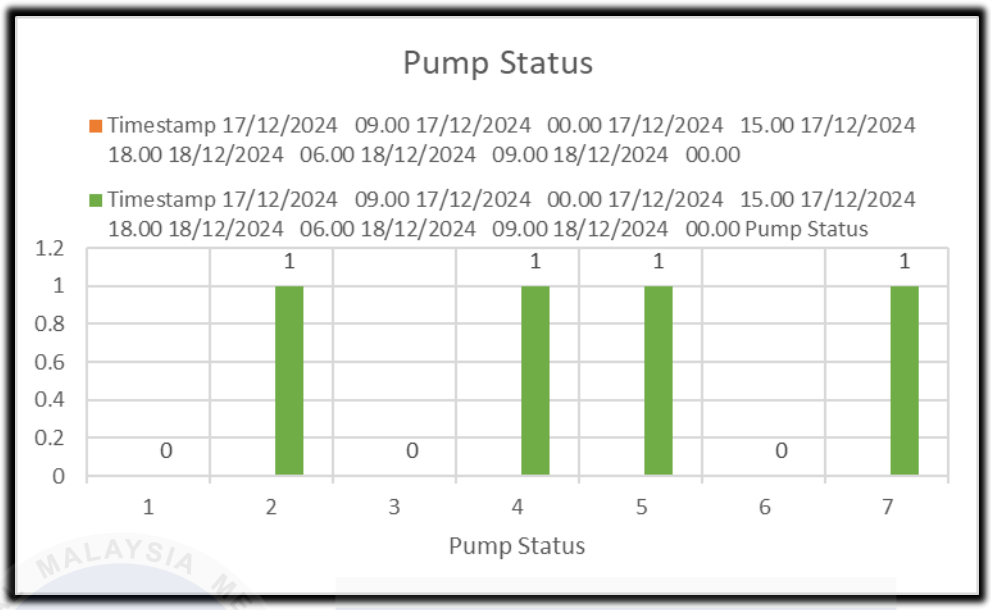

Figure 4.25 : Pump Status for Raw Soil Moisture Data

4.2.4.2 Manual Mode with Rain Detected

Table 4.3 : Manual Mode Data

Timestamp	Manual Mode	Rain Detected	Pump Status
19/12/2024 09.00	OFF	Yes	OFF
19/12/2024 00.00	ON	No	ON
19/12/2024 15.00	OFF	No	ON
19/12/2024 18.00	ON	Yes	ON

In this experiment, the Manual Mode is checked agaisnst the Rain Detected. When the Manual Mode is in ON Mode the water pump will stay ON although the Rain Detected states Yes. If the Manual Mode is in OFF Mode the Rain Detected will take over the pump the results are shown in the Figure 4.26, Figure 4.27 and Figure 4.28 .

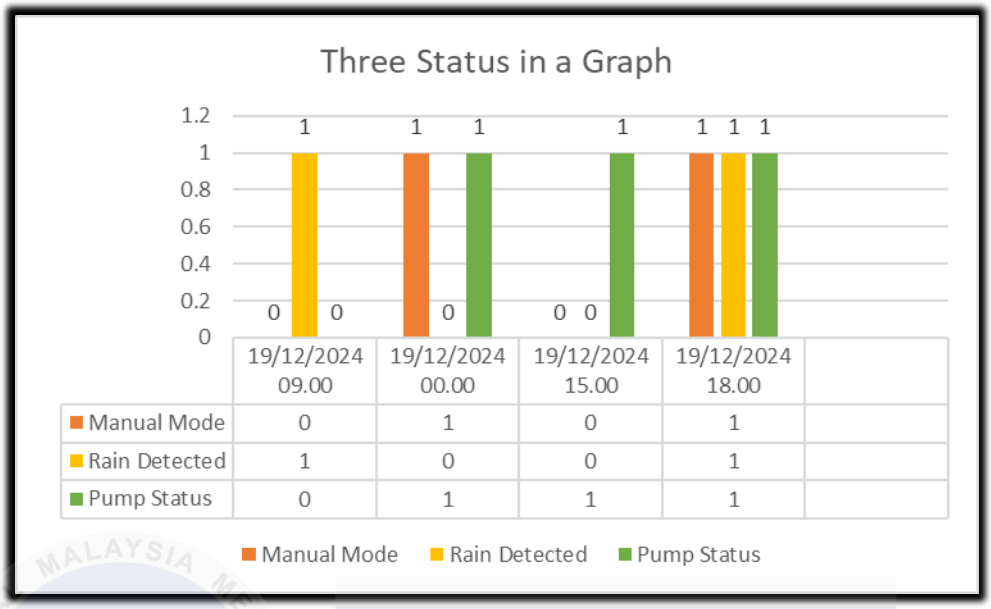

Figure 4.26 : Data for Manual Mode Override Data

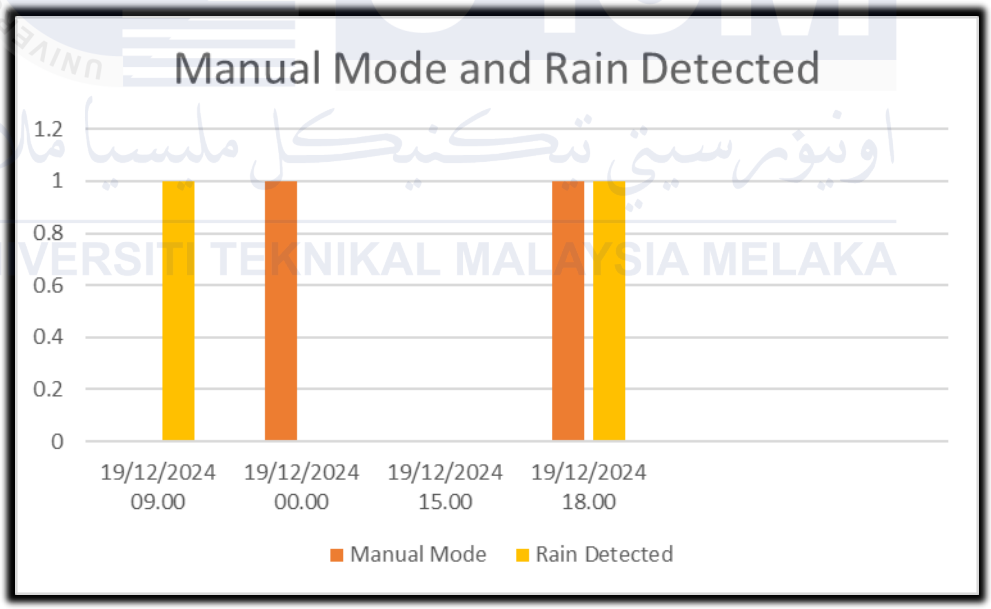

Figure 4.27 : Data for Manual Mode and Rain Detected

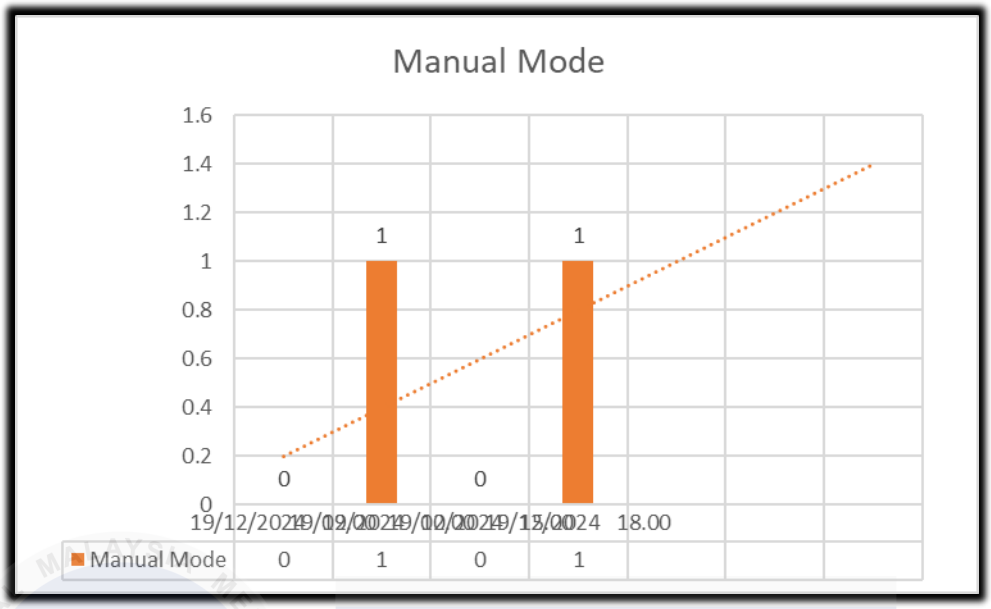

Figure 4.28 : Manual Mode data for Pump Status

4.2.4.3 Rain Status and Rain Detected Data

Table 4.4 : Data of Rain Status and Rain Detected

Timestamp	Rain Status(Rain Sensor)	Rain Detected	Pump Status
20/12/2024 06.00	Dry	No	ON
20/12/2024 15.00	Wet	No	OFF
21/12/2024 09.00	Wet	Yes	OFF
21/12/2024 00.00	Dry	Yes	OFF
22/12/2024 15.00	Wet	No	OFF
22/12/2024 18.00	Dry	No	ON
23/12/2024 09.00	Dry	No	ON
23/12/2024 15.00	Dry	Yes	OFF
24/12/2024 00.00	Wet	No	OFF
24/12/2024 15.00	Wet	Yes	OFF

In this experiment, the Rain Status is checked against the Rain Detected. When the Rain Status is in Dry the pump will stay OFF because the Rain Detected states Yes. If the Rain

Status is in Dry and the Rain Detected states No the pump will ON as shown in the Table 4.4 and also Figure 4.29, Figure 4.30, Figure 4.31, Figure 4.32 and Figure 4.33.

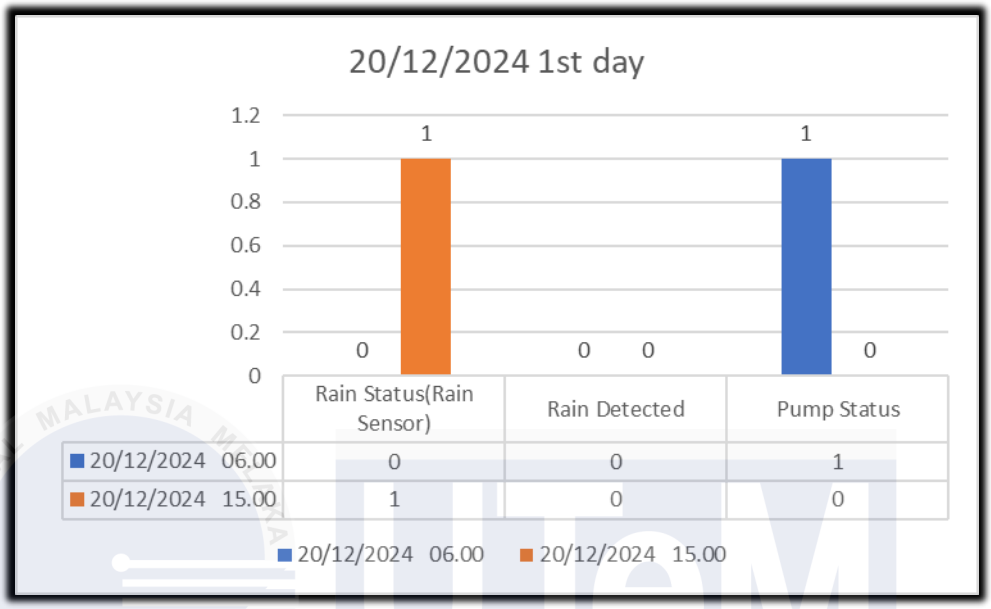

Figure 4.29 : 1st day Data

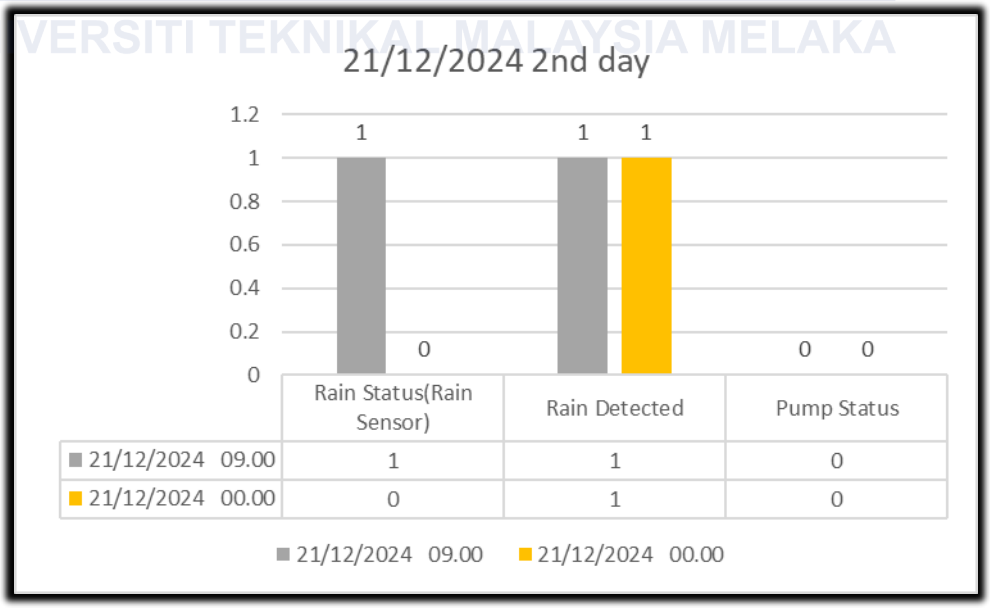

Figure 4.30 : 2nd day Data

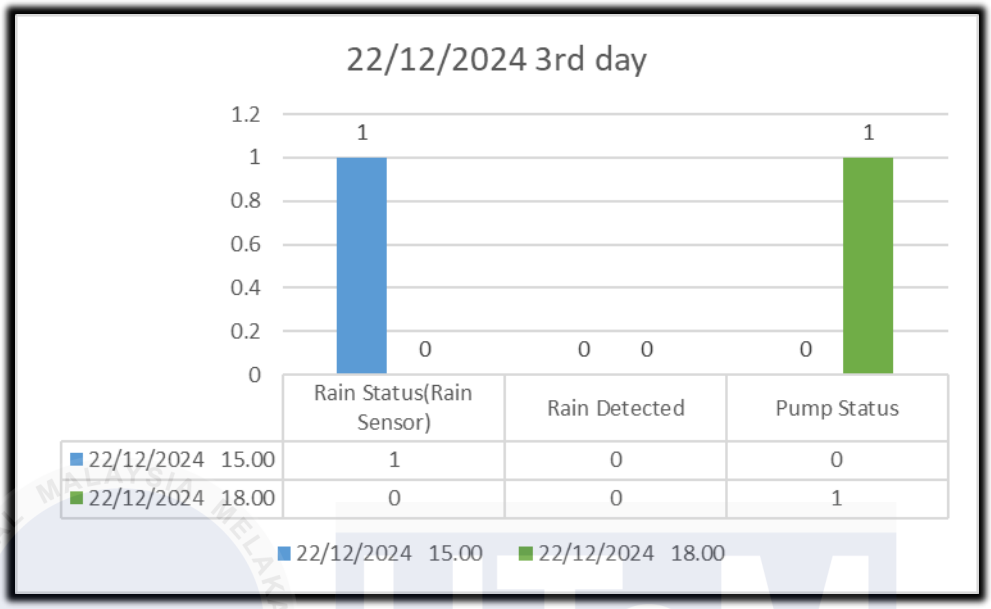

Figure 4.31 : 3rd day Data collect

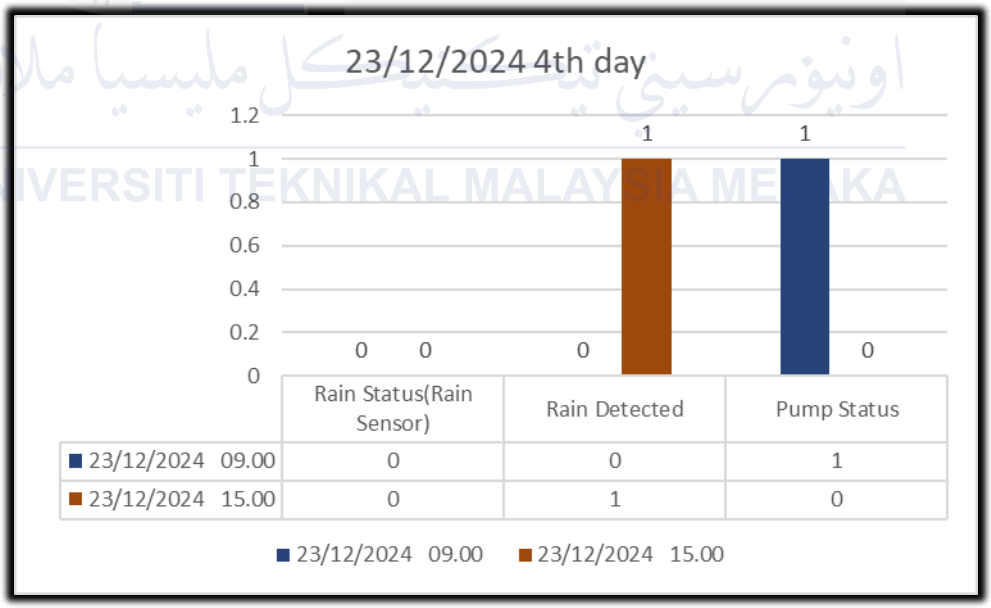

Figure 4.32 : 4th day Data collected

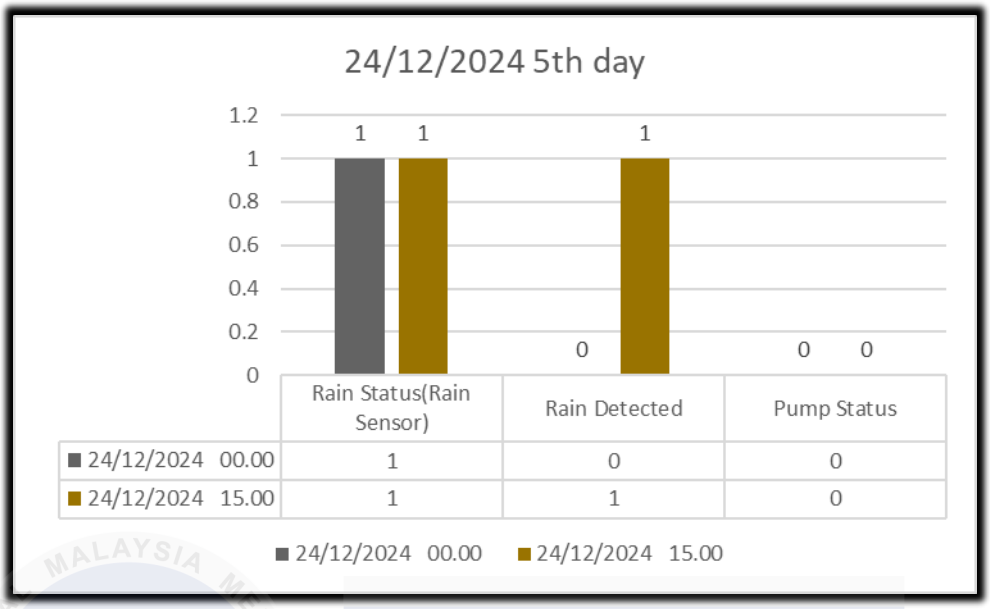

Figure 4.33 : 5th day Data Collection

Table 4.5 : Impact to the system from forecasting data

Forecast Condition	Rain Forecast	Soil Moisture	Pump Status	Impact	OLED Display
Rain Expected in Next Few Hours	Yes	Any (Usually Dry)	OFF	Prevents watering when rain is predicted, saving water	"Rain Forecast: Yes, Pump OFF"
No Rain Expected	No	Below Moisture Threshold	ON	Waters the plants if the soil is dry (below threshold).	"No Rain Forecast: Soil Dry, Pump ON"
Rain Expected in Next Few Hours	Yes	Above Moisture Threshold	OFF	Prevents watering if soil is already wet from rain.	"Rain Forecast: Yes, Soil Moist, Pump OFF"
No Rain Expected	No	Above Moisture Threshold	OFF	No watering needed if soil moisture is sufficient.	"No Rain Forecast: Soil Moist, Pump OFF"
Manual Mode	Any	Any	Manual Control (ON/OFF)	User decides the pump status manually.	"Manual Mode: Pump ON/OFF (Manual Control)"

Table 4.5 helps visualize how the system's actions are directly influenced by forecast data, allowing for efficient irrigation and water conservation.

4.2.4.4 Power Supply Data

Table 4.6 : Data of the power supply

Timestamp	Output Voltage (V)	Current (A)	Power Consumption(W)	Pump Status
21/11/2024 09:00	5.0	0.3	1.5	OFF
21/11/2024 10:00	5.0	1.2	5.6	ON
21/11/2024 13:00	5.0	0.5	2.5	OFF
21/11/2024 16:00	5.0	1.1	5.5	ON

This tracks the power supply's source voltage to ensure it remains within acceptable limits. The buck-boost converter ensures a consistent 5V output for components, regardless of input variations. This reflects the load on the power supply, which will be higher when the pump is active. Indicates whether the pump is ON or OFF, helping correlate power consumption with operational states as shown in Table 4.6 and continued with Figure 4.34 and Figure 4.35 .

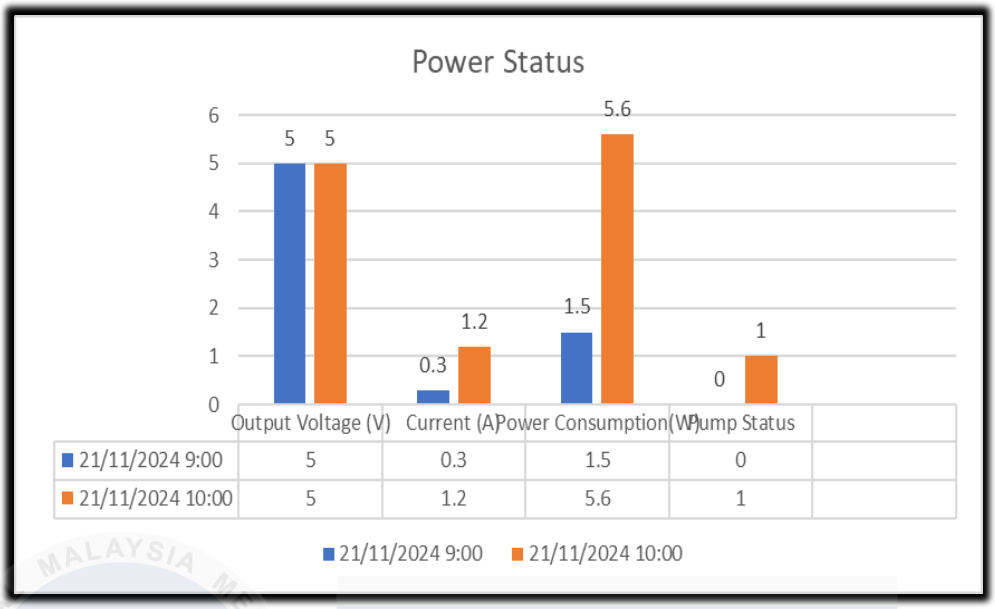

Figure 4.34 : Power Status Data for Morning

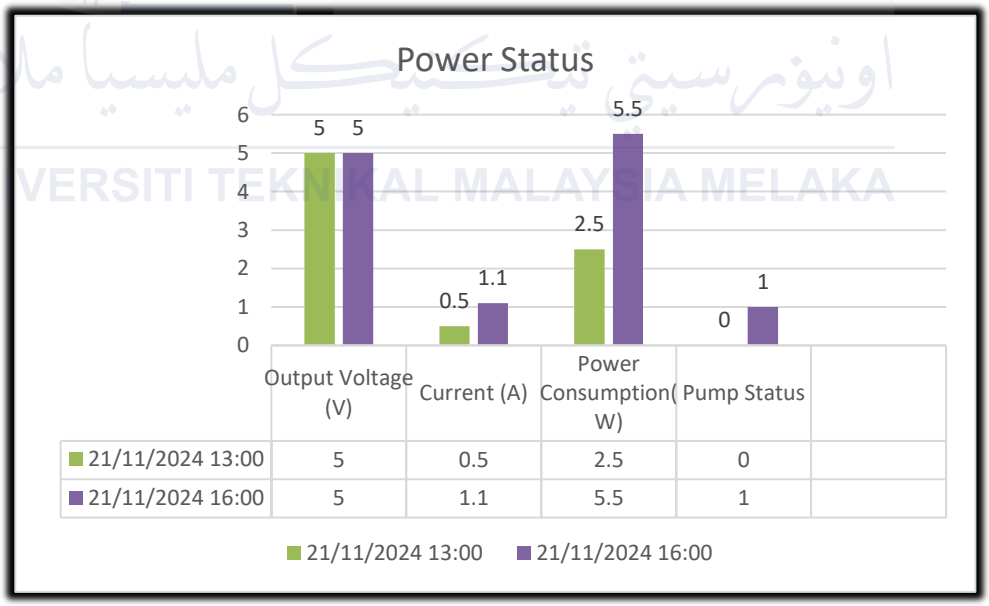

Figure 4.35 : Power Status Data for Afternoon

4.3 Summary

Chapter 4 presents the results and analysis of the smart irrigation system, focusing on hardware performance, software functionality, and data collection. The system's wiring was completed and tested to ensure seamless operation, with the soil moisture sensor providing critical data to regulate the pump based on set thresholds. Data collected from soil moisture levels were analyzed and represented using graphs and tables, highlighting the system's efficiency in irrigation management. The power supply, incorporating a 12V source with a buck-boost step-down converter, was evaluated for stable energy delivery, supporting reliable system performance. The OLED display provided real-time feedback on rain sensor status, soil moisture percentages, and pump modes, enhancing usability. Integration of OpenWeather API forecast data enabled the system to override soil moisture triggers when rain was predicted, showcasing its smart decision-making capabilities. The analysis confirmed that the system met its objectives, combining IoT technologies with efficient hardware and software to optimize irrigation practices.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Integrating forecasting data into the smart irrigation system brings several key advantages, particularly in terms of water conservation. By predicting rainfall, the system can adjust its watering schedules, avoiding unnecessary irrigation when rain is forecasted. This leads to water savings of up to 50%, as the system refrains from watering during anticipated rain events. By using forecasted weather data to anticipate rain, the system becomes more efficient in its water usage, reducing the need for manual intervention and optimizing irrigation based on actual conditions.

Additionally, the system reduces the operating time of the pump by 40-60%, thanks to the integration of forecasting data. When rain is expected, the pump stays off, and when the soil moisture levels are sufficient, it further reduces watering time. This decreased pump operation not only saves water but also cuts energy consumption, leading to a cost reduction of up to 40%. This makes the system more economical and environmentally friendly, as it minimizes energy usage associated with the pump's operation.

Furthermore, the system helps maintain optimal plant health by ensuring that the plants receive the right amount of water at the right time. With the forecast data and soil moisture levels, the system avoids both over-watering and under-watering, which can be detrimental to plant growth. Studies have shown that such a system can improve plant health by 20-30%, fostering better growth and yield. Overall, by combining real-time soil moisture data and weather forecasts, the system enhances both resource efficiency and plant health, making it a sustainable solution for modern irrigation needs.

The results obtained validate the feasibility and practicality of using IoT technologies to enhance irrigation systems, providing a solid foundation for future developments. By leveraging real-time environmental data and user-friendly interfaces, the system offers a scalable solution for sustainable agriculture.

5.2 Potential for Commercialization

The IoT-based smart irrigation system developed in this project holds significant potential for commercialization due to its ability to address critical challenges in modern agriculture, such as water conservation, crop yield optimization, and resource management. The integration of real-time soil moisture monitoring, rain detection, and weather forecasting provides a comprehensive solution that aligns with the needs of farmers and agricultural businesses. By automating irrigation processes, the system reduces manual labor, enhances efficiency, and ensures sustainable water usage—features that are highly valuable in regions facing water scarcity.

With its cost-effective design, leveraging widely available components such as the ESP8266 Wi-Fi module and sensors, the system can be produced and marketed at an affordable price, making it accessible to small-scale and large-scale farmers alike. The mobile app interface adds value by enabling remote monitoring and control, a key feature for busy agricultural operations or urban gardeners. Furthermore, the scalability of the system allows it to be adapted for various applications, including home gardening, commercial farms, and even large-scale plantations. By addressing critical agricultural needs with an emphasis on sustainability, the system is well-positioned for adoption in markets increasingly focused on smart and eco-friendly farming technologies.

5.3 Future Works

Future works for the smart irrigation system with IoT-based rain forecast capability can focus on several key areas to enhance its functionality, scalability, and overall impact on sustainable agriculture:

- **Integration of Additional Sensors**

Add sensors for humidity, temperature, and wind speed to provide more comprehensive environmental data for enhanced decision-making.

- **Advanced Connectivity Options**

Implement LoRa or GSM modules for better connectivity in remote areas lacking Wi-Fi infrastructure.

- **Machine Learning Integration**

Develop machine learning algorithms to analyze sensor data and predict irrigation schedules for improved system intelligence and autonomy.

- **Improved Mobile Application Features**

Introduce push notifications for real-time updates, such as low water levels or unusual soil conditions.

Add a graphical dashboard to display historical data trends for better user insights.

- **Solar Power Integration**

Incorporate a solar panel system to power the irrigation setup, increasing energy efficiency and sustainability.

- **Rainwater Harvesting Integration**

Develop a mechanism to collect and store rainwater for irrigation, reducing dependency on external water sources.

- **Expandable System Design**

Allow modular upgrades to accommodate larger irrigation areas or additional zones for better scalability.

- **Cloud Storage for Data Analysis**

Use cloud platforms for storing and analyzing historical sensor data to enhance system monitoring and forecasting capabilities.

- **Real-Time Voice or AI Assistance**

Implement voice-activated or AI-driven commands for easier system control via the mobile application.

- **Weather Prediction Refinements**

Improve the OpenWeather API integration for more precise weather forecasting to better synchronize irrigation with upcoming rain conditions.

REFERENCES

[1] C. Gao, L. Luo, Y. Zhang, B. Pearson, and X. Fu, “Microcontroller Based IoT System Firmware Security: Case Studies.”

[2] "What is Arduino," *SparkFun*, [Online]. Available: https://www.sparkfun.com/arduino_guide. [Accessed: Jan. 3, 2025].

[3] H. Dipak Ghael, L. Solanki, G. Sahu, and A. Professor, “A Review Paper on Raspberry Pi and its Applications,” *International Journal of Advances in Engineering and Management (IJAEM*, vol. 2, p. 225, 2008, doi: 10.35629/5252-0212225227.

[4] O. Bhat, P. Gokhale, and S. Bhat, “Introduction to IOT,” *International Advanced Research Journal in Science, Engineering and Technology ISO*, vol. 3297, no. 1, 2007, doi: 10.17148/IARJSET.2018.517.

[5] Sun, W., et al. (2014). Wi-Fi could be much more. *IEEE Communications Magazine*, 52(11), 22-29. https://doi.org/10.1109/MCOM.2014.6957139

[6] S. Kang *et al.*, “Cellular Wireless Networks in the Upper Mid-Band,” *IEEE Open Journal of the Communications Society*, vol. 5, pp. 2058–2075, 2024, doi: 10.1109/OJCOMS.2024.3373368.

[7] S. Sivaperumal, *Real-Time System Design*, C I R Books and Publications, Mar. 2023.

[8] J. Mesquita, D. Guimarães, C. Pereira, F. Santos, L. Almeida, and L. Almeida, “Assessing the ESP8266 WiFi module for the Internet of Things Assessing the ESP8266 WiFi module for the Internet of Things Assessing the ESP8266 WiFi module for the Internet of Things,” 2018. [Online]. Available: www.cister.isep.ipp.pthttp://www.cister.isep.ipp.pt

[9] V. Barral Vales, O. C. Fernandez, T. Dominguez-Bolano, C. J. Escudero, and J. A. Garcia-Naya, “Fine Time Measurement for the Internet of Things: A Practical Approach Using ESP32,” *IEEE Internet Things J*, vol. 9, no. 19, pp. 18305–18318, Oct. 2022, doi: 10.1109/JIOT.2022.3158701.

[10] K. Divya, C. Hari Priya, C. Narendra Kumar, and G. Venkata Shiva Kumar, “ISSN: 2581-4451 © IJRAD,” 2019.

[11] M. Cain, “Durham E-Theses Linear actuator for a submersible water pump for use in boreholes.” [Online]. Available: http://etheses.dur.ac.uk

[12] K. C. Divya and J. Østergaard, “Battery energy storage technology for power systems-An overview,” *Electric Power Systems Research*, vol. 79, no. 4. pp. 511–520, Apr. 2009. doi: 10.1016/j.epsr.2008.09.017.

[13] E. L. Hsiang, Z. Yang, Q. Yang, Y. F. Lan, and S. T. Wu, “Prospects and challenges of mini-LED, OLED, and micro-LED displays,” *J Soc Inf Disp*, vol. 29, no. 6, pp. 446–465, Jun. 2021, doi: 10.1002/jsid.1058.

[14] “An_Efficient_Weather_Forecasting_System Wheather”.

[15] Abba, Namkusong, Lee, and Crespo, “Design and Performance Evaluation of a Low-Cost Autonomous Sensor Interface for a Smart IoT-Based Irrigation Monitoring and Control System,” *Sensors*, vol. 19, no. 17, p. 3643, Aug. 2019, doi: 10.3390/s19173643.

[16] A. Math, L. Ali, and U. Pruthviraj, “Development of Smart Drip Irrigation System Using IoT,” in *2018 IEEE Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, IEEE, Aug. 2018, pp. 126–130. doi: 10.1109/DISCOVER.2018.8674080.

[17] J. N. Ndunagu, K. E. Ukhurebor, M. Akaaza, and R. B. Onyancha, “Development of a Wireless Sensor Network and IoT-based Smart Irrigation System,” *Appl Environ Soil Sci*, vol. 2022, pp. 1–13, Jun. 2022, doi: 10.1155/2022/7678570.

[18] N. Ismail *et al.*, “Smart irrigation system based on internet of things (IOT),” *J Phys Conf Ser*, vol. 1339, no. 1, p. 012012, Dec. 2019, doi: 10.1088/1742-6596/1339/1/012012.
[19] F. A. A. A.-R. O. R. M. E. Karar, “A Pilot Study of Smart Agricultural Irrigation using Unmanned Aerial Vehicles and IoT-Based Cloud System,” *Information Sciences Letters*, vol. 10, no. 1, pp. 131–140, Jan. 2021, doi: 10.18576/isl/100115.
[20] W. HAMDOON and A. ZENGİN, “A NEW IOT-BASED SMART IRRIGATION MANAGEMENT SYSTEM,” *Middle East Journal of Science*, vol. 9, no. 1, pp. 42–56, Jun. 2023, doi: 10.51477/mejs.1142136.
[21] T. Sai *et al.*, “Prototyping of Smart Irrigation System Using IoT Technology,” in *2021 7th International Conference on Electrical, Electronics and Information Engineering (ICEEIE)*, IEEE, Oct. 2021, pp. 1–5. doi: 10.1109/ICEEIE52663.2021.9616696.
[22] S. B. Saraf and D. H. Gawali, “IoT based smart irrigation monitoring and controlling system,” in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, IEEE, May 2017, pp. 815–819. doi: 10.1109/RTEICT.2017.8256711.
[23] A. Abdelmoamen Ahmed, S. Al Omari, R. Awal, A. Fares, and M. Chouikha, “A distributed system for supporting smart irrigation using Internet of Things technology,” *Engineering Reports*, vol. 3, no. 7, Jul. 2021, doi: 10.1002/eng2.12352.
[24] K. E. Lakshmiprabha and C. Govindaraju, “Hydroponic-based smart irrigation system using Internet of Things,” *International Journal of Communication Systems*, vol. 36, no. 12, Aug. 2023, doi: 10.1002/dac.4071.

APPENDICES

Appendix A Full Project Coding

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <FirebaseESP8266.h>
#include <ArduinoJson.h>

// OLED Display Configuration
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET    -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

// Firebase Configuration
#define FIREBASE_HOST "https://forecast-d237e-default-rtdb.asia-
southeast1.firebasedatabase.app"
#define FIREBASE_AUTH "38Uc2QOXzOCnXALmeNBt13X7xa8sdgHtqgP7CvBh"

// Pin Definitions
const int rainPin = D5;
const int soilPin = A0;
const int pumpPin = D6;

// Moisture Thresholds and Sensor Calibration
const int AirValue = 650;
const int WaterValue = 310;
const int moistureThreshold = 600;

// WiFi Credentials
const char* SSID = "JDM GENG"; //ADIVK
const char* PASS = "03062000"; //adivk9221

// Firebase Setup
FirebaseData firebaseData;
FirebaseConfig firebaseConfig;
FirebaseAuth firebaseAuth;

// System State Variables
bool isRaining = false;
int soilMoisture = 0;
int rawSoilMoisture = 0;
bool pumpStatus = false;
```

```
String manualPump = "OFF";
String manualMode = "OFF"; // New variable for manual mode
unsigned long forecastTimestamp = 0;

// Forecast Info Variables
String weatherCondition = "Unknown";
String forecastTimeDate = "N/A";
String weatherIcon = "10d"; // Default weather icon
String weatherID = "800";
int forecastTemp = 0;
int forecastHumidity = 0;
bool waitingForRain = false;

// Timing Variables
unsigned long realTimeMillisOffset = 0;
unsigned long twoHours = 2 * 60 * 60 * 1000; // 2 hours in milliseconds
unsigned long slideInterval = 5000; // 5 seconds per slide
unsigned long lastSlideChange = 0;
int currentSlide = 0;

// Functions to handle real-time synchronization
unsigned long calculateEpoch(int year, int month, int day, int hour, int
minute, int second) {
  const int daysInMonth[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30,
31};
  unsigned long daysSinceEpoch = 0;

  for (int y = 1970; y < year; y++) {
    daysSinceEpoch += (y % 4 == 0 && (y % 100 != 0 || y % 400 == 0)) ? 366 :
365;
  }

  for (int m = 1; m < month; m++) {
    daysSinceEpoch += daysInMonth[m - 1];
    if (m == 2 && (year % 4 == 0 && (year % 100 != 0 || year % 400 == 0))) {
      daysSinceEpoch++; // Leap year adjustment for February
    }
  }

  daysSinceEpoch += day - 1;
  unsigned long epochTime = daysSinceEpoch * 86400 + hour * 3600 + minute *
60 + second;
  return epochTime;
}

void syncRealTime(String timestamp) {
  int year = timestamp.substring(0, 4).toInt();
  int month = timestamp.substring(5, 7).toInt();
```

```
  int day = timestamp.substring(8, 10).toInt();
  int hour = timestamp.substring(11, 13).toInt();
  int minute = timestamp.substring(14, 16).toInt();
  int second = timestamp.substring(17, 19).toInt();

  unsigned long epochTime = calculateEpoch(year, month, day, hour, minute,
second);
  forecastTimestamp = epochTime * 1000;
  realTimeMillisOffset = forecastTimestamp - millis();
}

unsigned long getRealWorldMillis() {
  return millis() + realTimeMillisOffset;
}

// Setup Function
void setup() {
  // Initialize Display
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
  }
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.display();
  // Initialize Pins
  pinMode(rainPin, INPUT);
  pinMode(pumpPin, OUTPUT);
  digitalWrite(pumpPin, LOW);

  // Initialize Serial and WiFi
  Serial.begin(115200);
  WiFi.begin(SSID, PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nWiFi connected!");

  // Initialize Firebase
  firebaseConfig.host = FIREBASE_HOST;
  firebaseConfig.signer.tokens.legacy_token = FIREBASE_AUTH;
  Firebase.begin(&firebaseConfig, &firebaseAuth);
  Firebase.reconnectWiFi(true);
```

```
  Serial.println("Firebase initialized!");
}

// Fetch Firebase Data
void getFirebaseData() {
  if (Firebase.getString(firebaseData, "forecastData/timestamp")) {
    syncRealTime(firebaseData.stringData());
  }

  if (Firebase.getString(firebaseData, "manualPump/pump")) {
    manualPump = firebaseData.stringData();
  }

  if (Firebase.getString(firebaseData, "manualPump/mode")) {
    manualMode = firebaseData.stringData();
  }
}

// Fetch Additional Firebase Data for Forecast
void getForecastData() {
  if (Firebase.getString(firebaseData, "forecastData/condition")) {
    weatherCondition = firebaseData.stringData();
  }

  if (Firebase.getString(firebaseData, "forecastData/timestamp")) {
    forecastTimeDate = firebaseData.stringData();
  }

  if (Firebase.getInt(firebaseData, "forecastData/temp")) {
    forecastTemp = firebaseData.intData();
  }

  if (Firebase.getInt(firebaseData, "forecastData/humidity")) {
    forecastHumidity = firebaseData.intData();
  }

  if (Firebase.getString(firebaseData, "forecastData/icon")) {
    weatherIcon = firebaseData.stringData(); // Get icon ID from Firebase
  }

  if (Firebase.getString(firebaseData, "forecastData/id")) {
    weatherID = firebaseData.stringData(); // Get icon ID from Firebase
  }
}

void wateringLogic(unsigned long currentTime) {
  // Check if manual mode is enabled
  if (manualMode == "ON") {
```

```
    if (manualPump == "ON") {
      digitalWrite(pumpPin, HIGH);
      pumpStatus = true;
      Serial.println("Manual mode: Pump ON");
    } else {
      digitalWrite(pumpPin, LOW);
      pumpStatus = false;
      Serial.println("Manual mode: Pump OFF");
    }
    return;
  }

  // Check if forecast predicts rain within two hours
  if (forecastTimestamp >= currentTime - twoHours && forecastTimestamp <=
currentTime) {
    if ((weatherID >= "200" && weatherID <= "232") ||
      (weatherID >= "300" && weatherID <= "321") ||
      (weatherID >= "500" && weatherID <= "531")) {
      waitingForRain = true;
      digitalWrite(pumpPin, LOW);
      pumpStatus = false;
      Serial.println("Forecast mode: Waiting for rain, Pump OFF");
      return;
    }
  }

  // Reset waitingForRain if no rain is predicted
  waitingForRain = false;

  // Soil moisture and rain detection logic
  if (isRaining) {
    digitalWrite(pumpPin, LOW);
    pumpStatus = false;
    Serial.println("Normal mode: It's raining, Pump OFF");
  } else if (rawSoilMoisture >= moistureThreshold) {
    digitalWrite(pumpPin, HIGH);
    pumpStatus = true;
    Serial.println("Normal mode: Soil moisture low, Pump ON");
  } else if (rawSoilMoisture < moistureThreshold) {
    digitalWrite(pumpPin, LOW);
    pumpStatus = false;
    Serial.println("Normal mode: Soil moisture sufficient, Pump OFF");
  } else {
    digitalWrite(pumpPin, LOW);
    pumpStatus = false;
    Serial.println("Error: No valid condition met. Pump OFF");
  }
}
```

```
// Send Data to Firebase
void sendToFirebase() {
  Firebase.setString(firebaseData, "/sensorData/rainStatus", isRaining ?
"Wet" : "Dry");
  Firebase.setString(firebaseData, "/sensorData/pumpStatus", pumpStatus ?
"ON" : "OFF");
  Firebase.setInt(firebaseData, "/sensorData/rawSoilMoisture",
rawSoilMoisture);
  Firebase.setInt(firebaseData, "/sensorData/soilMoisture", soilMoisture);
  Firebase.setString(firebaseData, "/sensorData/manualMode", manualMode);
  Firebase.setString(firebaseData, "/sensorData/manualPump", manualPump);
}

// Display Slide on OLED
void displaySlide() {
  display.clearDisplay();
  display.setCursor(0, 0);

  if (currentSlide == 0) {
    display.print("Rain Status: ");
    display.println(isRaining ? "Wet" : "Dry");
    display.print("Soil Moisture: ");
    display.print(soilMoisture);
    display.println("%");
    display.print("Pump Status: ");
    display.println(pumpStatus ? "ON" : "OFF");
  } else if (currentSlide == 1) {
    display.println("Forecast: ");
    display.println(weatherCondition);
    display.print("Time: ");
    display.println(forecastTimeDate);
    display.print("Temp: ");
    display.print(forecastTemp);
    display.println("C");
    display.print("Humidity: ");
    display.print(forecastHumidity);
    display.println("%");
    display.print("Waiting Rain: ");
    display.println(waitingForRain ? "Yes" : "No");
  }

  display.display();
}

// Main Loop
```

```
void loop() {
  getFirebaseData();
  getForecastData();

  isRaining = !digitalRead(rainPin);
  rawSoilMoisture = analogRead(soilPin);
  soilMoisture = map(rawSoilMoisture, AirValue, WaterValue, 0, 100);
  soilMoisture = constrain(soilMoisture, 0, 100);

  unsigned long currentTime = getRealWorldMillis();
  wateringLogic(currentTime);

  if (millis() - lastSlideChange >= slideInterval) {
    currentSlide = (currentSlide + 1) % 2;
    lastSlideChange = millis();
  }
  displaySlide();

  sendToFirebase();
  delay(5000);
}
```