SMART IRRIGATION SYSTEM

A

MAJOR PROJECT-II REPORT

Submitted in partial fulfillment of the requirements

for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

By

GROUP NO.21

Lucky Soni	**0187CS211091**
Muskan Khaira	**0187CS211104**
Krishnpal Rajput	**0187CS211087**
Manish Ahirwar	**0187CS211095**
Jayprakash	**0187CS211079**

Under the guidance of

Dr. Bhavana Gupta

(Associate Professor)

Department of Computer Science & Engineering

Sagar Institute of Science &Technology (SISTec), Bhopal (M.P)

Approved by AICTE, New Delhi & Govt. of M.P.

Affiliated to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.)

June -2025

Sagar Institute of Science & Technology (SISTec), Bhopal (M.P)

Department of Computer Science & Engineering

CERTIFICATE

We hereby certify that the work which is being presented in the B.Tech. Major Project-II Report entitled **SMART IRRIGATION SYSTEM,** in partial fulfillment of the requirements for the award of the degree of ***Bachelor of Technology,*** submitted to the Department of **Computer Science & Engineering**, Sagar Institute of Science & Technology (SISTec)**,** Bhopal (M.P.) is an authentic record of our own work carried out during the period from Jan-2025 to Jun-2025 under the supervision of **Dr. Bhavana Gupta.**

The content presented in this project has not been submitted by me for the award of any other degree elsewhere.

Lucky Soni	**Muskan Khaira**	**Krishnpal Rajput**	**Manish Ahirwar**	**Jayprakash**
0187CS211091	**0187CS211104**	**0187CS211087**	**0187CS211095**	**0187CS211079**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Dr. Bhavana Gupta	**Dr. Amit Kumar Mishra**	**Dr. D.K. Rajoriya**
Project Guide	**HOD, CSE**	**Principal**

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. D. K. Rajoriya**, **Principal, SISTec and Dr. Swati Saxena, Vice Principal SISTec** Gandhi Nagar, Bhopal for giving us an opportunity to undertake this project.

We also take this opportunity to express a deep sense of gratitude to **Dr. Amit Kumar Mishra, HOD, Department of Computer Science & Engineering** for his kindhearted support

We extend our sincere and heartfelt thanks to our guide, **Dr. Bhavana Gupta,** for providing us with the right guidance and advice at the crucial junctures and for showing us the right way.

I am thankful to the **Project Coordinator, Prof. Deepti Jain** who devoted her precious time in giving us the information about various aspects and gave support and guidance at every point of time.

I would like to thank all those people who helped me directly or indirectly to complete my project whenever I found myself in any issue.

TABLE OF CONTENTS

ABSTRACT

Efficient water management in agriculture is essential to address challenges such as water scarcity, crop stress, and declining yields. Traditional irrigation methods often lack precision, leading to excessive water usage or poor crop health due to over- or under-watering.

To overcome these inefficiencies, this project introduces an IoT-based Crop-Specific Smart Irrigation System, an upgraded version of the previous generic smart irrigation system. This new approach integrates ESP8266 microcontroller, soil moisture sensors, temperature sensors, and humidity sensors to monitor real-time environmental conditions. A predefined database of crop-specific irrigation requirements ensures that each plant receives the right amount of water based on its optimal soil moisture, temperature, and humidity needs.

Additionally, the system is connected to the Blynk app, allowing farmers to monitor irrigation remotely and manually control water distribution if needed. This enhances water efficiency, improves crop yield, and promotes sustainable agricultural practices by ensuring precise irrigation tailored to different crop requirements.

LIST OF ABBREVIATIONS

ACRONYM	FULL FORM
SDLC	Software Development Life Cycle
AN	Arduino Nano
PS	Power Supply
VCC	Voltage Common Collector
5V	5 Volts
BO	Battery Operated
DHT11	Digital Humidity and Temperature sensor
GND	Ground

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

CHAPTER 1
INTRODUCTION

A Crop-Specific Smart Irrigation System is an advanced agricultural solution that leverages IoT technology and sensor-based automation to optimize water distribution for different crops. Unlike traditional irrigation methods, which apply a one-size-fits-all approach, this system considers crop-specific water needs based on real-time data from soil moisture, temperature, and humidity sensors.

By integrating the ESP8266 microcontroller, Blynk app, and a database of predefined optimal conditions for different crops, the system automatically regulates irrigation, ensuring efficient water management and improved crop yield. Additionally, remote monitoring and manual control through a mobile application enhance user convenience and scalability.

1.1 ABOUT PROJECT

This project introduces an IoT-based Smart Irrigation System designed to meet the specific water requirements of various crops. Traditional irrigation methods often fail to account for the unique needs of different crops, leading to water wastage and reduced productivity. To address this challenge, the system combines soil moisture sensors, automation, and Internet of Things (IoT) capabilities to deliver precise irrigation, ensuring crops receive the right amount of water at the right time.

The core components of the system include:

- Soil moisture sensors to detect real-time soil moisture levels
- ESP32 microcontroller for processing sensor data and managing system automation
- Blynk mobile application for remote monitoring and control

The system continuously collects data on soil moisture, temperature, and humidity, and based on pre-defined thresholds, automatically activates irrigation only when required. This ensures optimized water usage, improved crop health, and enhanced agricultural productivity.

Farmers can monitor the system remotely using the Blynk app, which provides real-time insights into environmental conditions and system performance. They can also manually control the irrigation system or make adjustments to thresholds as needed from any location, offering greater flexibility and convenience. This is especially beneficial for those managing multiple fields or

dealing with unpredictable weather.

To ensure affordability and accessibility, the system uses low-cost components like the ESP32 and basic moisture sensors. Despite being budget-friendly, the solution does not compromise on reliability or scalability. The ESP32's built-in Wi-Fi removes the need for extra modules, simplifying the design and reducing the overall cost.

A key feature of the system is its data logging capability. It stores historical data on moisture levels, temperature, humidity, and irrigation cycles—either on a local SD card or a cloud platform depending on the configuration. This data allows farmers to identify trends, optimize irrigation schedules, and plan crop rotations more efficiently.

The system also includes real-time alerts and notifications. For instance, if moisture levels drop significantly or the system malfunctions, an instant notification is sent via the Blynk app, enabling the farmer to take timely action and prevent crop damage.

To enhance decision-making further, the system can be integrated with weather APIs, enabling it to adjust irrigation based on weather forecasts. If rain is predicted, the system can delay irrigation, saving water and preventing over-irrigation.

1.2 PROJECT OBJECTIVES

1. **Automate the Irrigation Process**
   Design an automated system that uses real-time soil moisture data to determine the optimal time for irrigation. The system triggers irrigation only when necessary, ensuring adequate water supply without wastage.
2. **Minimize Water Wastage**
   Optimize water usage by controlling irrigation based on precise data, helping conserve water resources, especially in regions facing water scarcity.
3. **Remote Monitoring and Control**
   Enable farmers to monitor soil moisture, temperature, and irrigation status through the Blynk mobile app. This feature allows adjustments to irrigation schedules remotely, ensuring timely and efficient operation.
4. **Develop a Cost-Effective Solution**
   Use affordable components like the ESP32 microcontroller and basic sensors to create a reliable yet economical system, making it accessible to farmers in low-income regions.

5. **Increase Crop Yield**

   Provide optimal irrigation conditions to improve crop health and productivity, avoiding the risks of over- or under-watering.

6. **Promote Sustainable Farming**

   Encourage sustainable practices by optimizing water usage, reducing dependency on external water sources, and minimizing the environmental impact of farming.

7. **Improve Efficiency and Reduce Labor Costs**

   Automate irrigation to reduce manual intervention, saving time and labor costs. This allows farmers to focus on other critical farming activities, improving overall efficiency.

Chapter 2
Software
&
Hardware
Requirements

CHAPTER 2
SOFTWARE & HARDWARE REQUIREMENTS

SOFTWARE REQUIREMENTS

Software Requirements (Developer):

- Arduino IDE
- Blynk App
- WiFi Library
- DHT Library

Software Requirements (Client):

- Blynk App
- Internet Connection

HARDWARE REQUIREMENTS

Hardware Requirements (Developer):

- ESP8266 Board
- Capacitive Soil Moisture Sensor
- DHT11 Sensor
- Water Pump
- Relay Module
- 3.7V Li-ion Battery

Hardware Requirements (Client):

- Smartphone or Tablet
- Wi-Fi Network

Chapter 3
Problem Description

CHAPTER 3
PROBLEM DESCRIPTION

In many parts of the world, traditional irrigation systems are still widely used for watering crops. While these methods have been effective in the past, they are becoming increasingly inefficient and unsustainable—especially in areas facing water shortages and unpredictable climate patterns. With the growing need for efficient water usage in agriculture, it's clear that traditional practices must evolve. This section outlines the key problems in current irrigation systems, such as water wastage, excessive labor dependence, and the lack of real-time data. These challenges can be addressed effectively through automation and IoT-based solutions like the ESP32 and Blynk-powered Smart Irrigation System.

Water Wastage

Water wastage remains one of the most pressing issues in agriculture today. Traditional irrigation methods like flood irrigation apply water uniformly across large fields without considering the specific needs of different soil types or crops. As a result, large quantities of water are lost through surface runoff, evaporation, and deep percolation. In drought-prone regions, this not only wastes a vital resource but also poses a threat to food security.

Over-irrigation leads to waterlogging, depletes essential nutrients, and reduces soil fertility. In the long term, this affects both crop yield and soil health. Moreover, water runoff from excessive irrigation often carries harmful fertilizers and chemicals into nearby rivers and lakes, damaging ecosystems and polluting drinking water sources. These problems highlight the urgent need for intelligent systems that monitor and optimize water usage.

Labor Inefficiencies

Conventional irrigation is highly dependent on human labor. Farmers must manually monitor soil conditions, check weather updates, and determine when and how much to irrigate. This process is time-consuming and prone to error—especially on larger farms or during the peak growing season.

Managing irrigation for multiple types of crops increases complexity and often requires frequent manual adjustments. In areas with limited access to trained labor or modern equipment, these inefficiencies become even more pronounced. The lack of automation leads to inconsistent watering, which can result in over- or under-irrigated crops and, consequently, lower yields. A system that reduces the need for manual effort while maintaining precision in irrigation can be a game-changer.

Lack of Real-Time Data for Decision-Making

Another limitation of traditional systems is the absence of real-time data. Farmers often rely on static forecasts, delayed weather updates, or guesswork based on observation. Without accurate, real-time information about soil moisture, temperature, or environmental conditions, it's difficult to make the best decisions about irrigation timing and quantity.

This can result in either excessive water use or crop stress due to under-irrigation. Without data-driven insights, farmers cannot adapt quickly to changing conditions, which can lead to reduced efficiency and resource wastage.

Environmental Impact and Resource Management

Poor irrigation practices damage the environment, degrade soil quality, and contribute to the depletion of natural water sources. Over-irrigation can lead to soil salinization—where minerals accumulate and make the soil toxic to plants. Such conditions are difficult to reverse and harm long-term agricultural productivity.

Moreover, high water demand in agriculture creates pressure on already limited freshwater resources. Competing demands for water between agriculture, industry, and domestic use can create tension in water-stressed regions. This makes it critical to adopt systems that maximize water use efficiency and promote sustainability.

The Need for Automation

To address these challenges, automation through IoT offers an effective solution. Smart irrigation systems powered by ESP32 microcontrollers and connected to applications like Blynk enable farmers to automate the watering process based on real-time sensor data.

These systems use soil moisture sensors, temperature/humidity sensors, and weather APIs to make intelligent decisions about when and how much to irrigate. They help avoid water wastage by activating the system only when needed and reduce labor requirements by removing the need for constant manual intervention.

With real-time monitoring and mobile-based controls, farmers can manage their fields remotely, receive instant alerts, and adjust parameters on the fly. This not only conserves water and reduces costs but also improves crop health, enhances yields, and supports long-term environmental sustainability.

Chapter 4
Literature Survey

CHAPTER 4
LITERATURE SURVEY

Efficient water management is a critical global challenge, particularly in agriculture, where water use constitutes a significant portion of overall consumption. The Smart Irrigation System addresses this issue by leveraging IoT technology to optimize water use, ensuring sustainable agricultural practices and conserving resources. Agriculture accounts for approximately 70% of global water consumption, and inefficient irrigation practices often lead to water wastage and reduced crop yields. Factors like climate variability, soil conditions, and crop water requirements necessitate a dynamic and intelligent approach to irrigation management. This system improves water use efficiency by monitoring soil moisture levels, temperature, humidity, and weather conditions in real time and automating irrigation accordingly.

Water scarcity is a growing concern worldwide, with studies estimating that nearly 1.8 billion people will face absolute water scarcity by 2025, making efficient water use in agriculture a necessity [1]. Traditional irrigation systems often rely on fixed schedules or manual operation, which may not account for real-time environmental changes. Smart irrigation systems address these limitations by integrating sensor-based technologies and automation for precision water management. These systems not only save water but also enhance crop health and yield by ensuring timely and adequate irrigation [2].

Developing an effective smart irrigation system requires addressing several challenges. First, soil moisture sensors need precise calibration to account for varying soil types and environmental conditions, as incorrect thresholds can result in over- or under-irrigation. Second, real-time adjustments must consider factors such as sudden weather changes, which could lead to water wastage if not properly accounted for. Third, high initial costs for IoT devices and sensors can limit adoption, especially for small-scale farmers. Finally, agricultural regions lack stable power supply and internet connectivity, making system reliability a concern [3].

The proposed smart irrigation system integrates IoT devices, wireless communication, and cloud computing to enable real-time monitoring, automated irrigation, data analysis, and remote control. Sensors continuously measure soil moisture, temperature, and weather conditions to determine irrigation needs. Automated irrigation valves are triggered based on sensor readings, ensuring optimal water delivery without manual intervention. Collected data is analyzed to create patterns and predict irrigation schedules, further improving efficiency. Additionally, farmers can

monitor and control the system remotely via a smartphone application or web interface [4].

The benefits of smart irrigation are significant. Optimized irrigation reduces water waste by supplying only the required amount of water, which conserves water resources and lowers water bills. Adequate irrigation also improves crop health and productivity, enhancing yield. Furthermore, these systems promote sustainability by preventing over-irrigation and soil erosion, making them a critical tool for environmentally friendly farming practices [5]. Smart irrigation systems are applicable to various agricultural settings, including farms with diverse crop types, greenhouses requiring precise environmental control, and urban landscaping for water-efficient maintenance of parks and gardens [2].

Chapter 5

Software Requirements Specification

CHAPTER 5
SOFTWARE REQUIREMENTS SPECIFICATION

For the successful development of an IoT-based Smart Irrigation System using ESP32 and Blynk, defining the software requirements is essential. These requirements ensure that the system operates efficiently, securely, and reliably. The software components enable real-time monitoring, remote control of irrigation, and seamless interaction between hardware and the user through the mobile application.

5.1 FUNCTIONAL REQUIREMENTS

The functional requirements define the primary operations of the system and how users interact with it via the Blynk app for monitoring and control.

Real-Time Monitoring: The ESP32 must continuously collect data from the soil moisture sensor and environmental sensors (e.g., temperature and humidity). These readings should be updated on the Blynk app in real-time or at regular intervals to allow users to view the current field conditions.

Automatic Irrigation Control: When the soil moisture level falls below a specified threshold, the system must automatically activate the water pump, ensuring water is only used when necessary.

Remote Control and Monitoring via Blynk App: The system should allow users to monitor real-time sensor data and control the water pump manually through the Blynk mobile app from any location with internet access.

Data Logging: The software should record historical data related to soil moisture, temperature, and humidity. This data must be viewable in the app for analyzing trends and making informed irrigation decisions.

Configurable Thresholds: Users should be able to set and modify the moisture, temperature, and humidity thresholds via the app, adapting the system to various crop types and conditions.

Alert Notifications: When sensor readings cross critical thresholds or system maintenance is

required, the app must send alerts or push notifications to the user to prompt timely action.

5.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements focus on the quality attributes of the software, including reliability, scalability, cost-efficiency, and responsiveness. These elements ensure the system remains usable, stable, and adaptable to the needs of various users.

Reliability: The software should operate consistently under various conditions, with minimal errors or downtime. Reliability is critical in agriculture, as irrigation failures can lead to crop losses. All sensors and actuators should work seamlessly, providing accurate readings and actions when required.

Scalability: The system should be designed to support scalability, enabling additional sensors, pumps, or zones to be incorporated as required. This allows farmers to expand the system to larger fields or diverse crop areas without significant modification to the software architecture.

Efficiency and Cost-Effectiveness: The software must prioritize resource efficiency by minimizing data processing time and battery consumption. Efficient data processing helps conserve power, essential for field deployment where power sources may be limited.

Interoperability: The software should be compatible with standard IoT and mobile platforms, allowing integration with other agriculture-related tools and applications, such as weather forecasting software or farm management systems**.**

5.3 PERFORMANCE

Performance requirements focus on response times, data processing speeds, and network reliability to ensure the system meets real-time operational needs.

Data Collection Frequency: The software should collect and process data from sensors at a frequency of at least once every minute. This ensures that the system has up-to-date information for irrigation decisions.

Network Reliability and Range: Since the system relies on wireless data transfer, the software must handle intermittent network issues gracefully. The ESP8266 or similar microcontroller should maintain a stable connection within a range of 50-300 meters, depending on the field setup.

System Responsiveness: The irrigation activation process should trigger within 2–3 seconds of a threshold breach. Rapid responsiveness is necessary to prevent under-watering or over-watering, especially in areas with fluctuating soil conditions.

Data Storage and Access: The system must store historical data without significant latency. Data access should be optimized for mobile devices, allowing users to retrieve past records within 2–3 seconds.

5.4 SECURITY

Security measures protect both the data and control functions from unauthorized access, ensuring only authenticated users can operate the system and view data.

Data Encryption: All data exchanged between the sensors, microcontroller, and Blynk app should be encrypted to prevent unauthorized interception and ensure data privacy.

Access Control: Only authenticated users with verified credentials should be allowed to access the irrigation system’s settings, sensor readings, and control functionalities.

Firmware Security: The ESP8266 microcontroller should be configured to allow secure bootloading, ensuring that only authorized firmware updates are installed. This prevents malicious code from being executed on the hardware**.**

5.5 USABILITY

Usability requirements focus on the software's user interface, accessibility, and ease of use, making it suitable for non-technical users such as small-scale farmers.

- **User-Friendly Interface**: The Blynk app should have an intuitive interface, displaying key data such as soil moisture, temperature, and irrigation status in a visually accessible way. The app should use icons and simple layouts to guide users through the monitoring and control process.

- **Ease of Setup**: The software should feature a guided setup process, allowing users to connect the hardware, calibrate sensors, and configure settings with minimal technical knowledge.

- **Language Accessibility**: To enhance accessibility, the software should support multiple languages relevant to the user base, such as local languages for small-scale farmers.

- **Compatibility with Mobile Devices**: The Blynk app should be compatible with both Android and iOS devices, ensuring wide accessibility and convenience for farmers.

Chapter 6
Software and Hardware Design

CHAPTER 6
SOFTWARE AND HARDWARE DESIGN

6.1 USE CASE DIAGRAM

The Use Case Diagram illustrates the interactions between the user (driver) and the system. It shows what tasks the system performs and how the user interacts with these features.

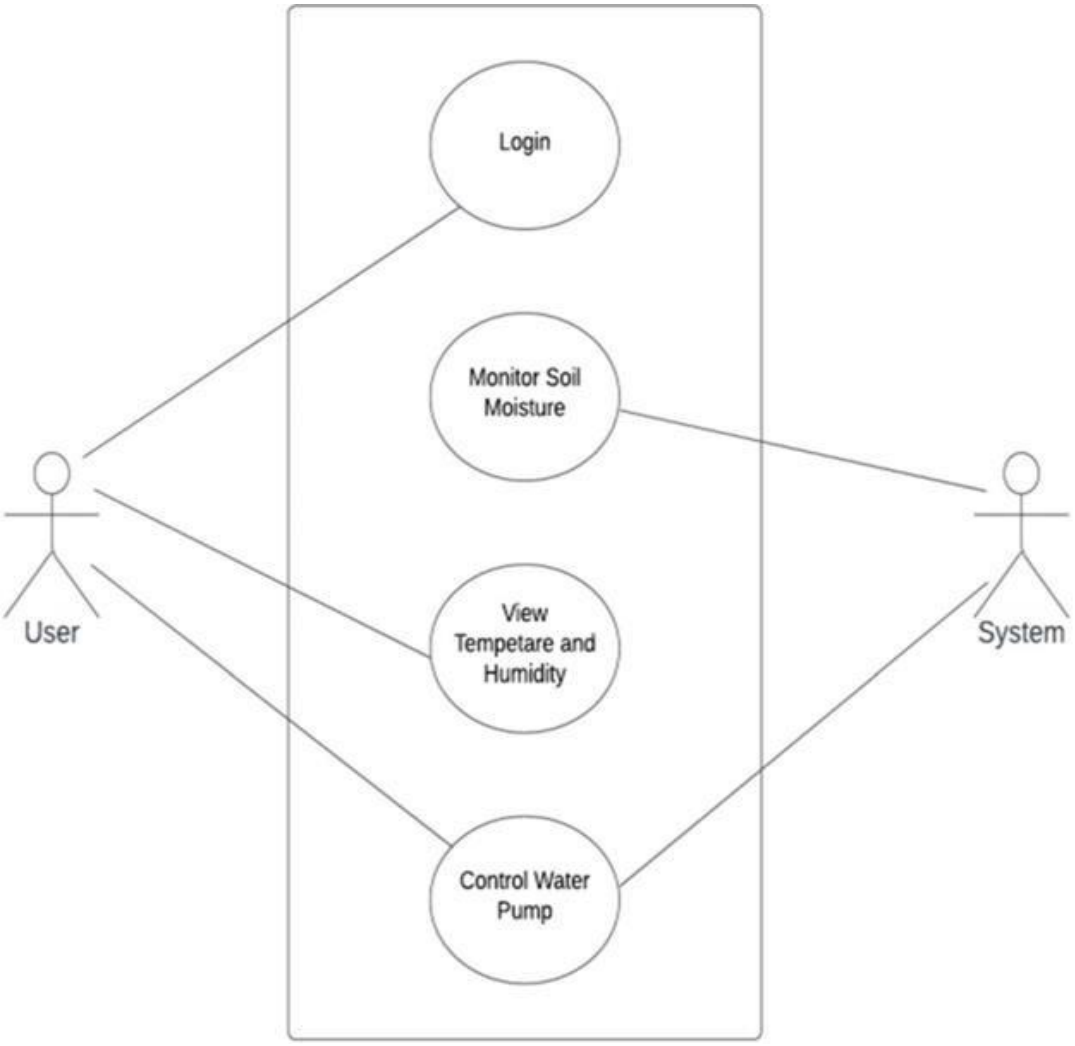

Figure 6.1: Use Case Diagram

6.2 ARCHITECTURE

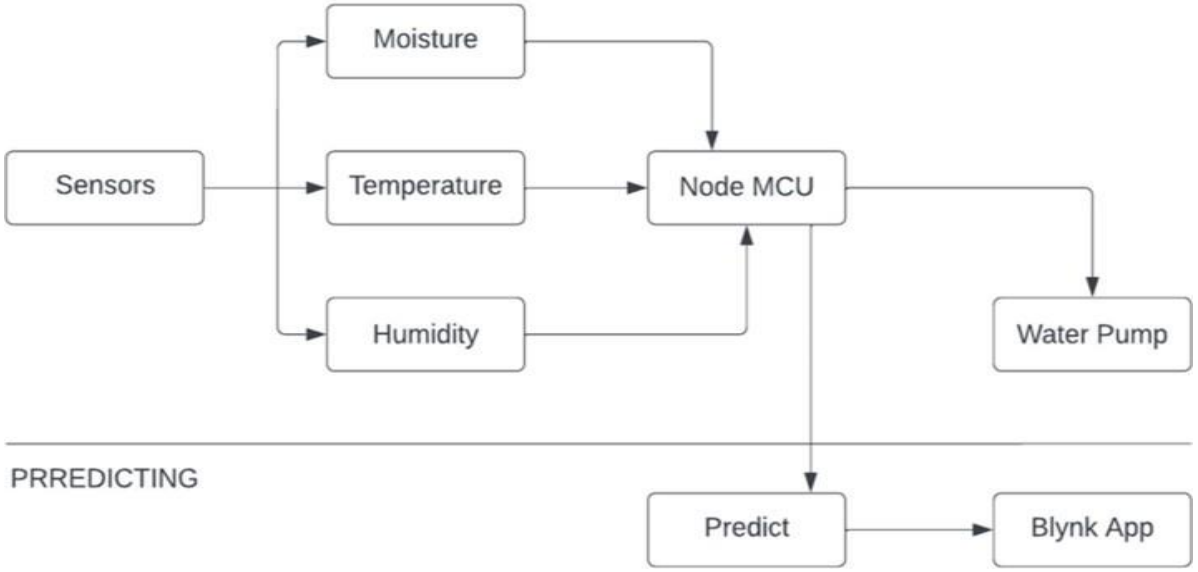

Figure 6.2: System Architecture

6.3 FLOW CHART DIAGRAM

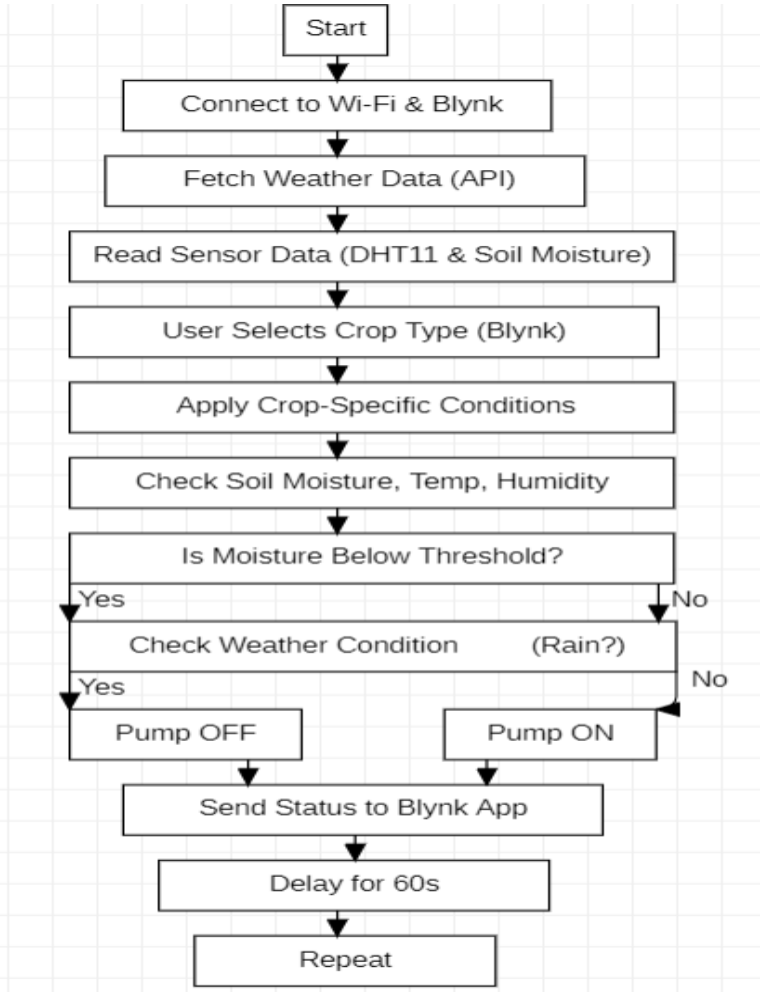

Figure 6.3: Flow Chart Diagram

6.4 CIRCUIT DIAGRAM

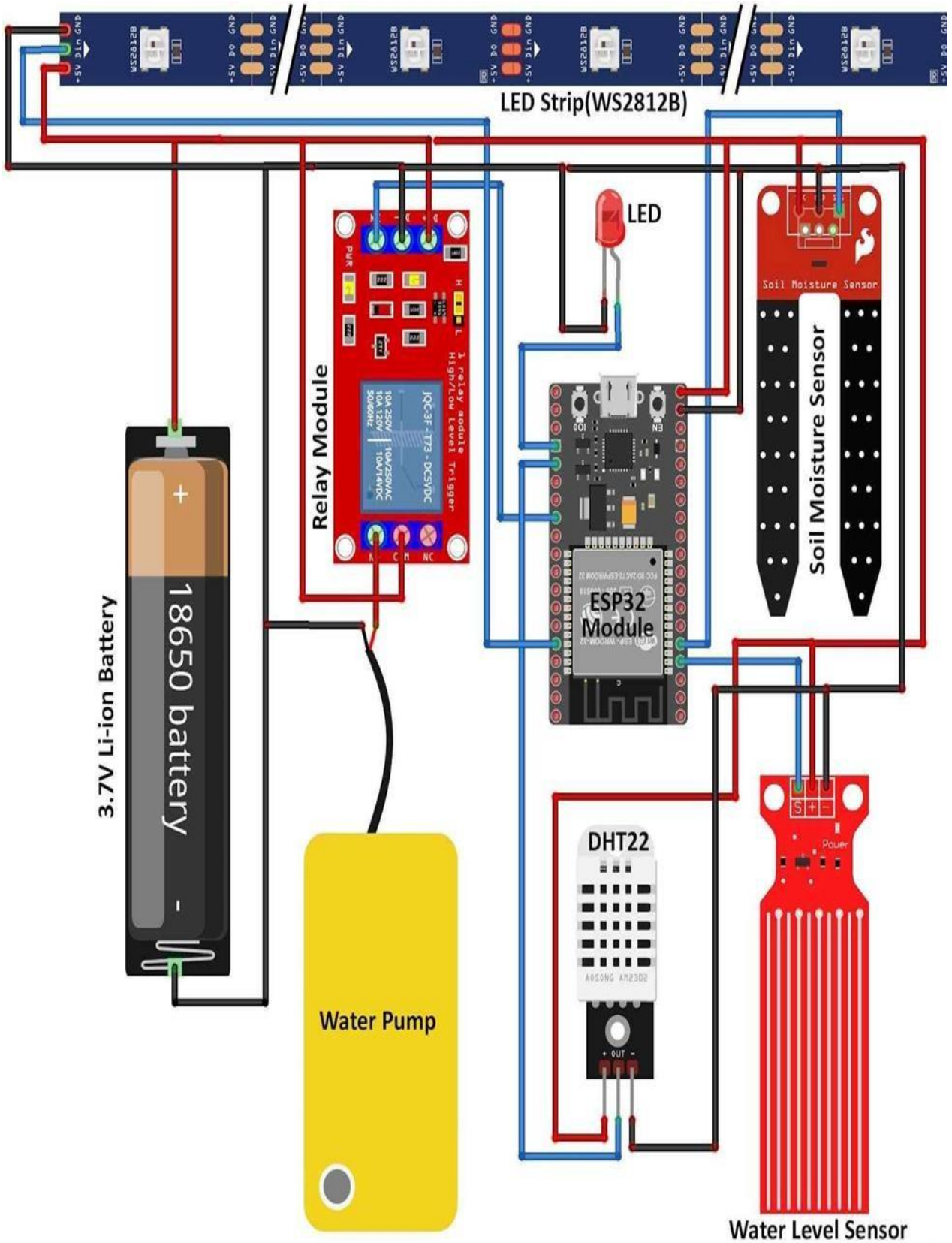

Figure 6.4: Circuit Diagram

6.5 PIN DIAGRAM

The Pin Diagram provides a detailed layout of the microcontroller or other key components used in the system. It shows the purpose of each pin (e.g., input, output, power) and how it connects to other parts of the system.

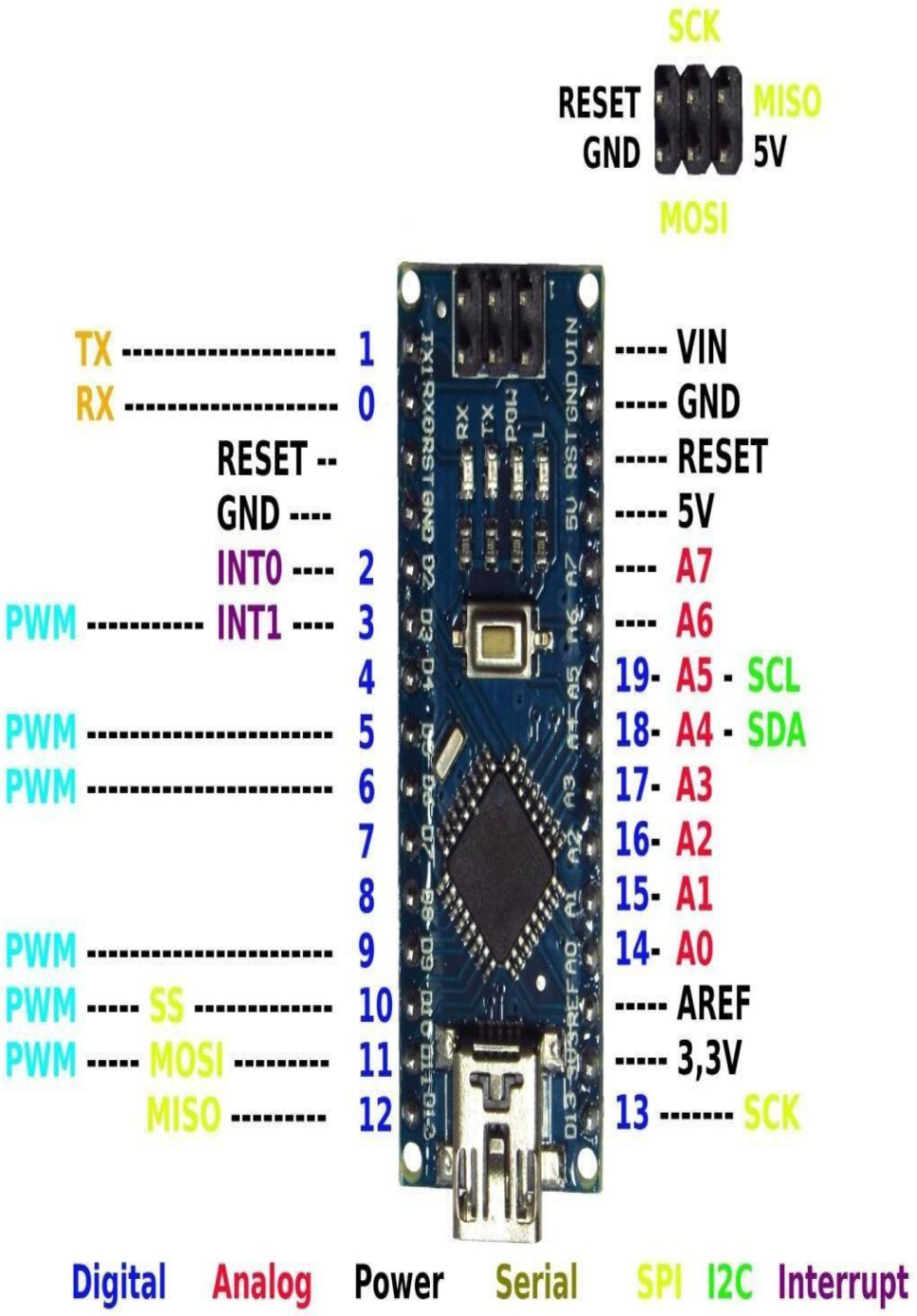

Figure 6.5: Pin Diagram

Chapter 7
IoT Module

CHAPTER 7
IOT MODULE

7.1 PRE-PROCESSING

Preparing the sensors and components to work smoothly with the Arduino Nano is a critical step building the **Smart Irrigation System**. First, the soil moisture sensor is set up to monitor the water content in the soil, providing real-time data on dryness or saturation levels. Similarly, the temperature and humidity sensors are configured to measure environmental conditions, ensuring accurate readings that influence irrigation decisions. All components, including the water pump (controlled via a relay module) and the 5V power supply, are properly connected and calibrated. This setup ensures that each sensor delivers clean and reliable data to the Arduino Nano, enabling the system to automate irrigation efficiently and respond quickly to changes in soil and weather conditions.

7.2 SIGNALS-PROCESSING

Signal processing is a critical component in the **Smart Irrigation System**, enabling the system to accurately monitor environmental and soil conditions. The primary function of signal processing in this system is to interpret sensor data—specifically, the readings from soil moisture, temperature, and humidity sensors. The system must differentiate between normal environmental variations and conditions that necessitate irrigation, such as dry soil or high temperature. Proper signal processing ensures that the irrigation system activates only when needed, avoiding false triggers that could lead to water waste.

The first stage of signal processing involves acquiring raw data from the sensors. The soil moisture sensor, for instance, provides continuous signals that represent the soil's water content. These signals are often noisy due to environmental disturbances, such as fluctuations in electrical interference or sensor placement variations. Signal filtering techniques, like low- pass filters or moving average algorithms, are applied to smooth the raw data and ensure it accurately reflects soil moisture levels. This step is essential to ensure reliable input for irrigation decisions.

Next, the system applies thresholding techniques to the filtered data to determine when irrigation is necessary. Thresholding involves setting specific values for parameters like soil moisture levels, temperature, or humidity. For example, if the soil moisture falls below a predefined level or the temperature exceeds a certain threshold the system triggers the irrigation process. Time-based thresholds may also be applied to prevent the system from overreacting to short-term fluctuations. This process ensures that irrigation is activated only under genuine

need, optimizing water usage.

In addition to soil moisture detection, the system may incorporate weather forecasting data or rain sensors to avoid unnecessary irrigation. For instance, if rain is expected, the system delays watering to conserve resources. Signal processing algorithms analyze data from multiple sources, integrating soil and environmental conditions to create a comprehensive profile of the field’s water requirements This multi-sensor approach enhances the system's efficiency and accuracy, ensuring crops receive the optimal amount of water.

Finally, signal processing involves the integration of all sensor data to make real-time decisions. After processing soil, temperature, and humidity signals, the system continuously evaluates conditions and adjusts irrigation schedules. If the system detects dry soil or high temperature, it activates the water pump. Conversely, if moisture levels are sufficient, the system pauses irrigation to avoid overwatering. This dynamic decision-making process ensures that the system responds appropriately and efficiently, contributing to sustainable water management and improved crop health.

7.3 IOT MODEL DESCRIPTION

The **Smart Irrigation System** uses an IoT model to monitor soil and environmental conditions, improving water management and crop health. Sensors such as soil moisture, temperature, and humidity detectors collect real-time data on the field's condition. This data is sent to an Arduino Nano, which analyzes the inputs and activates a water pump when irrigation is necessary.

The system is powered by a reliable source, such as a solar panel or a standard electrical supply, and includes a switch to manually turn it on or off as needed. It is compact, easy to install, and, operates in real-time without requiring an internet connection, making it an efficient and cost-effective solution for sustainable farming.

7.3.1 Capacitive Soil Moisture Sensor

Figure 7.3.1: Capacitive Soil Moisture Sensor

This sensor is an analog capacitive soil moisture sensor that measures soil moisture levels through capacitive sensing. The sensor adjusts its capacitance based on the amount of water in the soil. You can translate this capacitance variation into a voltage level, ranging from a minimum of 1.2V to a maximum of 3.0V. One of the benefits of the Capacitive Soil Moisture Sensor is its construction from corrosion-resistant materials, ensuring a prolonged service life.

7.3.2 DHT11

Figure 7.3.2: DHT11

The DHT11 is a basic and very cheap sensor that measures both temperature and humidity. It uses a special sensor to find out how much humidity is in the air and a temperature sensor to measure the temperature. The sensor gives us this information in a digital way through a single wire. It's easy to use, but we need to be patient because it takes about 2 seconds to get new information. In our project, we'll use this sensor to tell us how hot and humid the air is.

7.3.3 Microcontroller

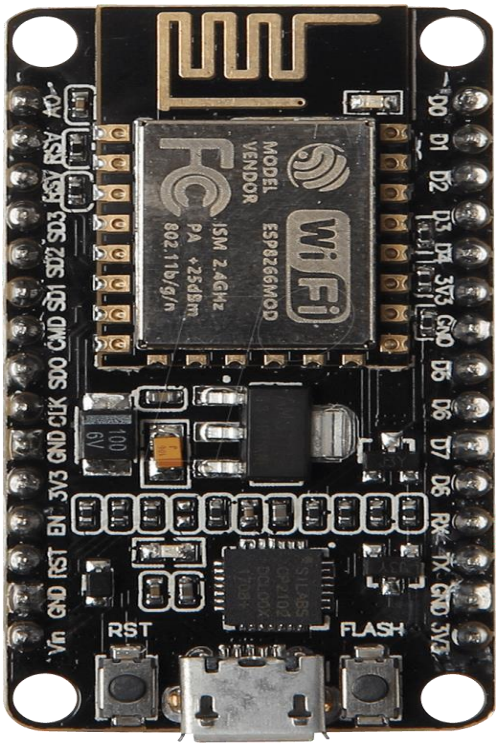

Figure 7.3.3: ESP8266

Acts as the brain of the system, processing data from sensors and controlling the water pump. Microcontrollers like the Arduino Nano are easy to program and integrate with various components, while ESP8266 or ESP32 adds Wi-Fi capabilities for remote monitoring.

7.3.4 Relay module

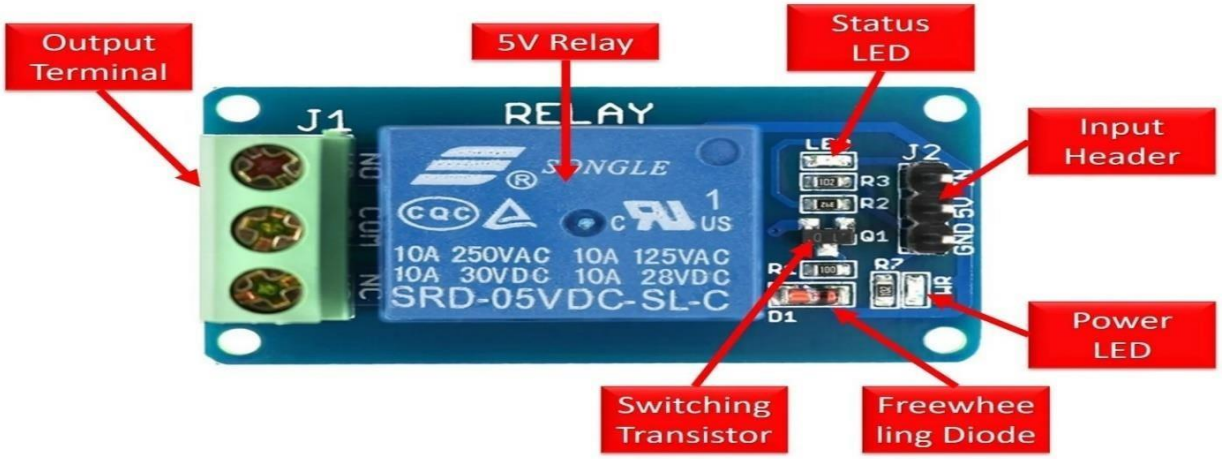

Figure 7.3.4: Relay module

Serves as a switch to control the water pump. The relay module ensures the pump operates only when triggered by the microcontroller based on sensor data.

7.3.5 Water pump

Figure 7.3.5: Water pump

Responsible for delivering water to the plants. It is controlled by the microcontroller and is activated when the soil moisture drops below the defined threshold.

7.3.6 Power Supply

Figure 7.3.6: Power Supply

Provides power to the system. This can be a direct electrical supply, a battery, or a solar panel for off-grid installations. Voltage regulators may be used to ensure stable power for the component.

7.4 RESULT ANALYSIS

The implementation and testing of the Smart Irrigation System yield promising results in terms of efficiency, water conservation, and crop health. Below is a detailed analysis of the system's performance based on various parameters:

1. **Accuracy of Sensor Readings**

The soil moisture sensor effectively detects real-time soil conditions, providing precise data on water levels. By activating the water pump only when soil moisture falls below a pre-set threshold, the system prevents overwatering and wastage. The integration of environmental data, such as temperature and rainfall, optimizes water usage further. environmental conditions, enabling dynamic adjustments to irrigation schedules. Rain sensors (if included) accurately detect rainfall, preventing unnecessary watering during wet conditions.

2. **Crop Yield and Health**

Crops irrigated with the smart system showed improved growth due to consistent and optimal watering. Overwatering-related issues, such as root rot and nutrient leaching, were minimized.

Chapter 8
Coding

CHAPTER 8
CODING

8.1 SOURCE CODE

```
#define BLYNK_TEMPLATE_ID "TMPL3yXgFsrqq" // Add your Blynk Template ID
#define BLYNK_TEMPLATE_NAME "Smart Irrigation System" // Your Blynk Template Name #define
BLYNK_AUTH_TOKEN "j-PNWSyHRwMDFRXCLOQmjZ3tVUlJ7a8A" // Your Blynk Auth Token

#include <ESP8266WiFi.h> #include <BlynkSimpleEsp8266.h> #include <DHT.h>

// Replace with your network credentials char ssid[] = "Wifi name"; char

pass[] = "Wifi Password";

// Define DHT11 sensor pin and type
#define DHTPIN 2  // DHT11 connected to GPIO2 (D4) #define DHTTYPE DHT11   // Define DHT type
as DHT11 DHT dht(DHTPIN, DHTTYPE);

// Define Soil Moisture pin
#define SOIL_MOISTURE_PIN A0 // Soil Moisture Sensor on A0 #define RELAY_PIN 5      //      Relay
connected to GPIO5 (D1)

BlynkTimer timer; // Blynk timer for sensor updates

// Function to read sensor data and send to Blynk void sendSensorData() {
// Read temperature and humidity from DHT11 sensor float humidity = dht.readHumidity(); float
temperature = dht.readTemperature();

// Read soil moisture value from the analog pin A0
int soilMoistureValue = analogRead(SOIL_MOISTURE_PIN);
```

```
// If DHT11 fails, display error

if (isnan(humidity) || isnan(temperature)) { Serial.println("Failed to read from DHT sensor!"); return;
}

// Send temperature and humidity data to Blynk (Virtual Pin V1 and V2) Blynk.virtualWrite(V1,
temperature); // V1 for temperature Blynk.virtualWrite(V2, humidity);
      // V2 for humidity

// Send soil moisture value to Blynk (Virtual Pin V3) Blynk.virtualWrite(V3, soilMoistureValue);
 // V3 for soil moisture

// Control relay based on soil moisture level (automatically) if (soilMoistureValue > 400)
{ // If soil is dry digitalWrite(RELAY_PIN, LOW); // Turn pump ON Serial.println("Pump
ON - Soil Dry");
} else {
digitalWrite(RELAY_PIN, HIGH);

// Turn pump OFF  Serial.println("Pump

OFF - Soil Wet");
}
// Debugging output on

Serial Monitor Serial.print("Temperature: "); Serial.print(temperature);

 Serial.print("°C, Humidity: "); Serial.print(humidity); Serial.print("%,

 Soil Moisture: ");

Serial.println(soilMoistureValue);
}

   // Manual control of the pump using the Blynk button
```

```
 BLYNK_WRITE(V4) {
int pumpState = param.asInt();
// Read button value (0 or 1) if (pumpState == 1) {

digitalWrite(RELAY_PIN, LOW); // Turn pump ON Serial.println("Pump ON - Button Pressed");

} else {

digitalWrite(RELAY_PIN, HIGH); // Turn pump OFF Serial.println("Pump OFF - Button Released");
}
}
void setup() {

// Initialize serial communication Serial.begin(115200);

// Initialize DHT sensor dht.begin();

// Set Relay pin as output and keep it OFF initially

pinMode(RELAY_PIN, OUTPUT);

digitalWrite(RELAY_PIN, HIGH); // Relay is active LOW

// Connect to WiFi and Blynk Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);

// Setup a timer to send sensor data to Blynk every 2 seconds

timer.setInterval(2000L, sendSensorData);

}

void loop() {
Blynk.run(); // Run Blynk timer.run(); // Run the timer
  }
```

Chapter 9
Result and Output Screens

CHAPTER 9
RESULT AND OUTPUT SCREENS

9.1 DESCRIPTION OF OUTPUTS

Water Pump Activation: When the system detects that the soil moisture level falls below the predefined threshold, it automatically activates the water pump to irrigate the field. Once the desired soil moisture level is reached, the pump stops, ensuring optimal water usage without manual intervention.

Visual Indicators: If the system includes an LED or display, it provides real-time information on irrigation status. For instance, messages such as "Soil Drying Detected," "Irrigating Now," or "Moisture Level Optimal" may be displayed. Blinking lights or other visual cues can indicate operational status or alert for sensor malfunction.

9.2 OUTPUT SCREENS

A screen (or mobile app interface) can provide a detailed output such as:

- "Current Soil Moisture Level: 20% (Irrigating…)"
- "Weather Alert: Rain Predicted, Irrigation Paused."

"Pump Status: Off (Optimal Moisture Level Achieved)."

These outputs allow farmers to monitor system performance and make adjustments as necessary.

9.3 PICTURES OF HARDWARE IN ACTION

System in the Car: Include photos showing the placement of sensors in the soil, the water pump connected to the irrigation lines, and the control unit (e.g., Arduino, Raspberry Pi) housed in a protective enclosure. Visuals of the system in operation, such as water being distributed when the pump is activated, can illustrate its functionality.

Driver Interaction: Show images where the driver is engaging with the system, such as the alarm going off when the system detects dro3wsiness (e.g., the eyes are closed for too long).

9.3.1 DRIVER INTERACTION

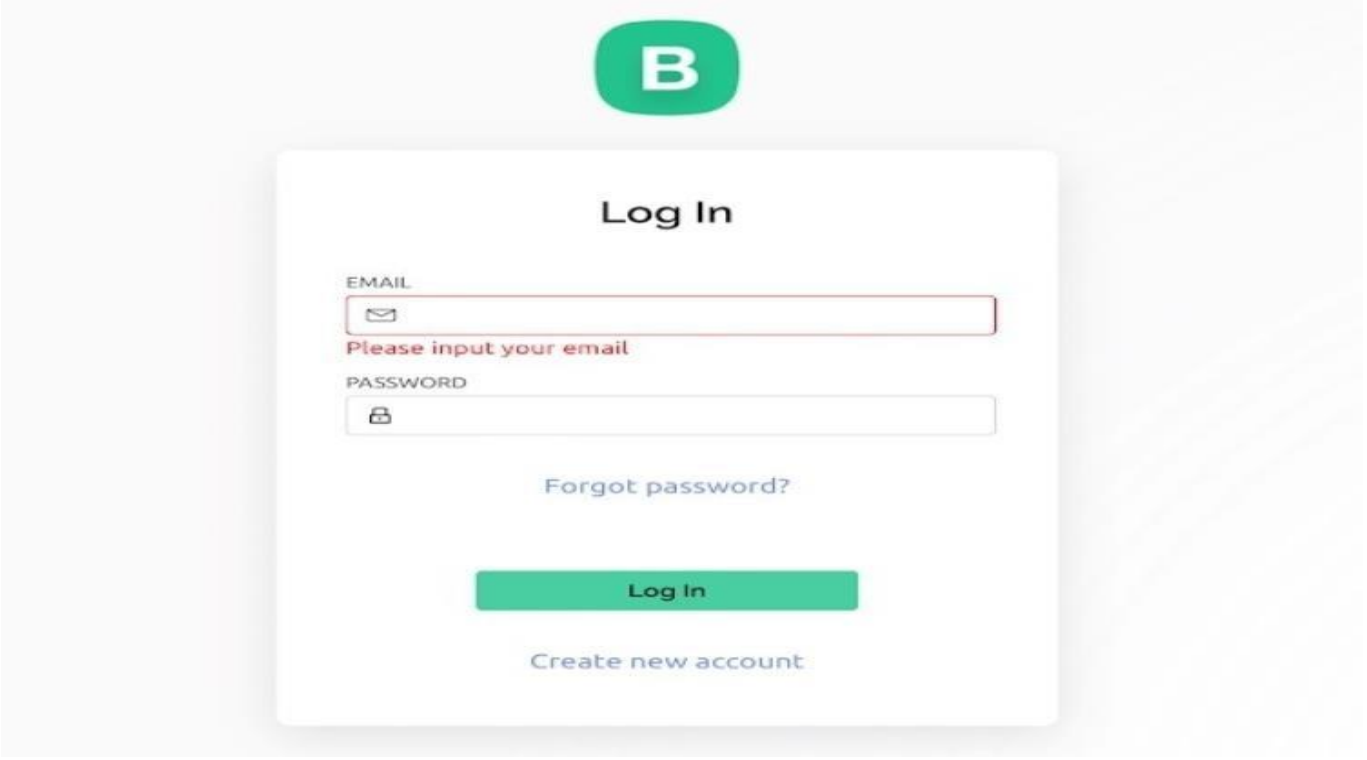

Figure 9.3.1: Driver Interaction

9.4 TEST SCENARIOS AND RESULTS

Test Case 1: The soil moisture level drops below the predefined threshold (e.g., 30%). The system detects the change and automatically activates the water pump.

Test Case 2: Soil moisture is within the optimal range, and no irrigation is needed .

Test Case 3: Rain is detected through weather data integration or an external rain sensor.

9.5 OBSERVATIONS

Response Time: The system activates the water pump almost instantly (within 1–3 seconds) after detecting low soil moisture levels. Similarly, irrigation stops promptly when the target moisture level is reached, ensuring efficient water use.

Battery Life: The system is designed for efficient power usage, lasting through long trips without overconsuming energy. The switch for powering the system allows easy activation and deactivation.

9.5.1 RESULT VIEW

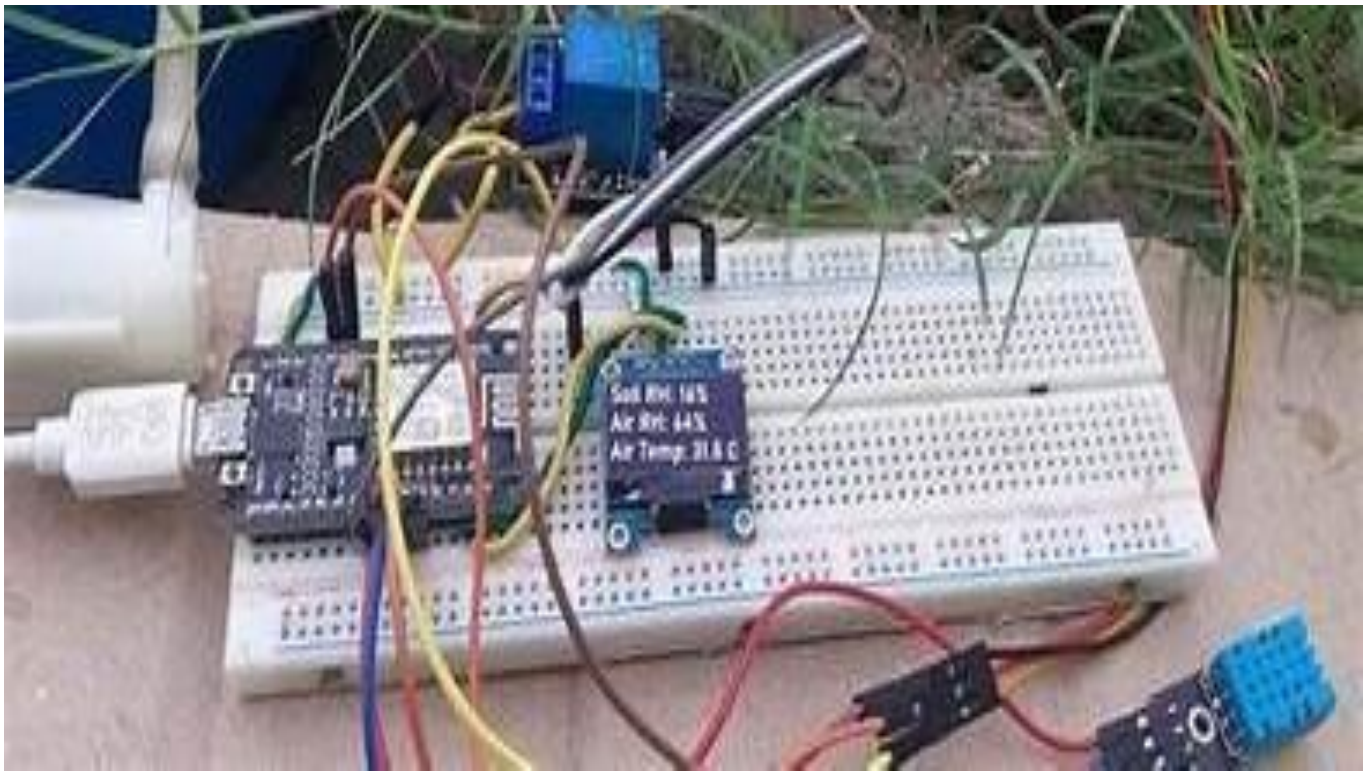

Figure 9.5.1: Result View

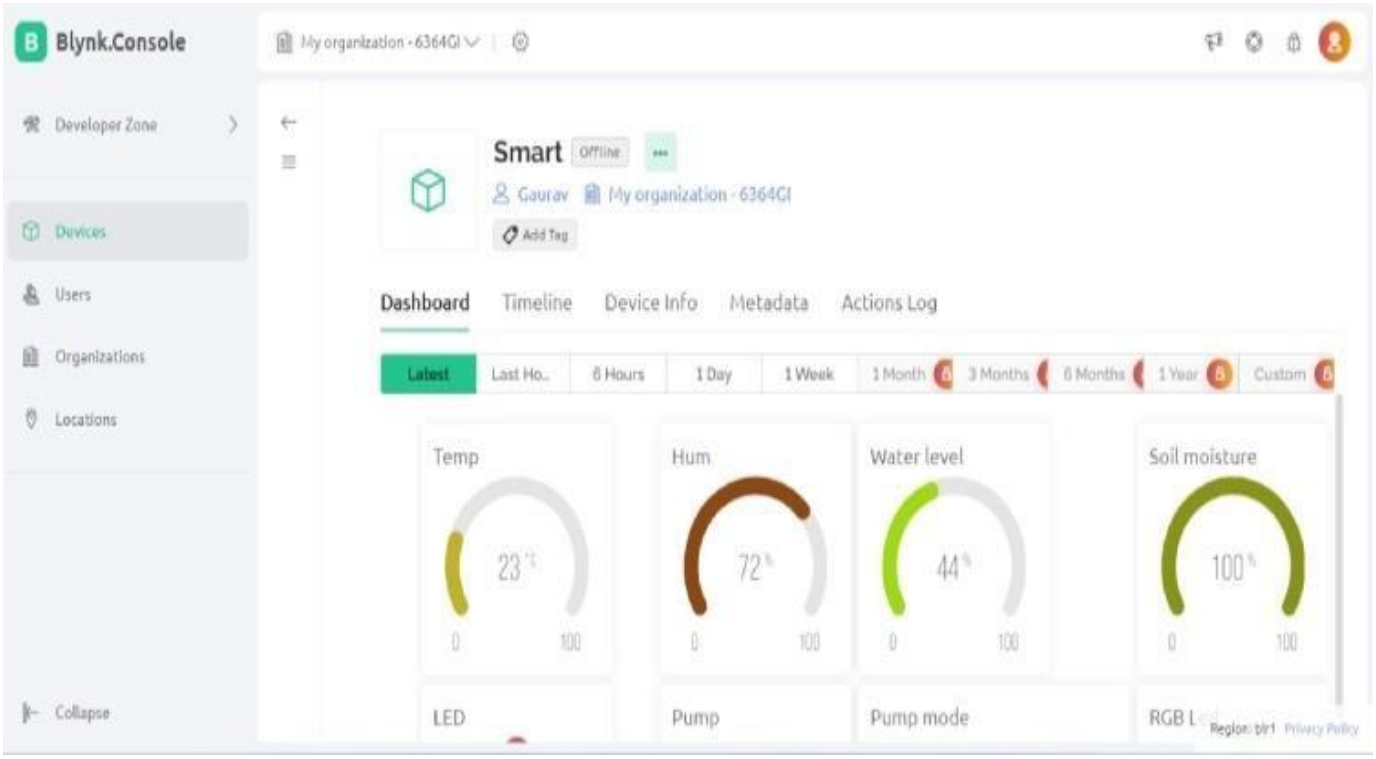

Figure 9.5.2: Result View

Chapter 10
Conclusion and Future work

CHAPTER 10
CONCLUSION AND FUTURE WORK

10.1 CONCLUSION

The developed IoT-based Smart Irrigation System using ESP32 and the Blynk app provides an efficient, affordable, and user-friendly solution for automated irrigation. By monitoring soil moisture in real-time and automating water supply, it conserves water, reduces manual effort, and ensures healthier crop growth. The system's remote control capability, via a mobile app, empowers farmers to manage irrigation from anywhere. This project promotes sustainable farming and is especially beneficial for small-scale farmers in water-scarce regions. With further improvements, it holds great potential to revolutionize traditional irrigation methods.

10.2 FUTURE WORK

The future development of the Smart Irrigation System will focus on enhancing precision, scalability, and integration with advanced technologies to further optimize water usage and improve agricultural efficiency. Below are key areas of improvement and potential advancements.

Advanced Sensor Integration:

Expanded Sensor Capabilities: Future systems could incorporate additional sensors, such as pH sensors to monitor soil acidity or salinity sensors to track salt levels in irrigated areas.

Multi-layer Soil Sensors: Using sensors to measure moisture and temperature at different soil depths would provide a more comprehensive understanding of water distribution in the root zone. Weather Prediction Sensors: Incorporating real-time weather sensors or data feeds could allow the system to preemptively adjust irrigation schedules based on upcoming rainfall, temperature spikes, or drought conditions**.**

Machine Learning and Adaptive Systems

Predictive Analytics: Implementing machine learning algorithms to analyze historical sensor data and predict future irrigation needs based on weather patterns, crop type, and soil conditions.

Personalization for Crop Types: Customizable settings for different crops, allowing the system to adapt to specific irrigation requirements based on plant species and growth stages.

Mobile and Cloud Connectivity

IoT and Cloud Integration: Expanding connectivity through IoT platforms for remote monitoring and control via mobile apps. Farmers could receive real-time alerts about soil conditions or system malfunctions.

Data Analytics and Insights: Uploading irrigation data to the cloud for analysis over time, providing actionable insights into water usage efficiency and crop performance.

Community Sharing: Enabling farmers to share anonymized data, creating collective intelligence for region-specific farming practices and water conservation strategies.

Smart Farm Integration

Cross-System Communication: Enabling the irrigation system to integrate with other smart farming tools, such as pest management systems, fertilizer dispensers, and crop health monitors.

Standardized Protocols: Using communication protocols (e.g., MQTT or LoRaWAN) to ensure interoperability across devices and platforms.

Multimodal Irrigation Control

Flexible Water Delivery Systems: Future iterations could support multiple irrigation techniques, such as drip, sprinkler, and flood irrigation, allowing dynamic selection based on crop and soil type.

Precision Irrigation with Robotics: Integrating robotic systems or autonomous vehicles to deliver targeted irrigation to specific plant rows or areas needing water.

REFERENCES

JOURNALS / RESEARCH PAPERS

1. Author surname, initials. (Year) Title of article, Journal name, Volume number (Issue or part number), first and last page numbers.

2. Jagtap, S., Babar, P., and Pawar, A. (2022) 'IoT-Based Soil Moisture Monitoring for Smart Irrigation', International Journal of Agricultural Technology, Vol. 45, pp. 98–110.

3. Zhang, M., and Smith, J. (2020) 'Real-Time Water Management in Precision Agriculture', Journal of Environmental Monitoring, Vol. 18, pp. 205–214..

BOOKS

4. Author surname, initials, Book Title, Publisher, Edition, Year of Publication.

5. Patel, R., IoT Solutions for Smart Agriculture, GreenTech Publishers, 1st Edition, 2021.

6. Kumar, S., and Verma, P., Sustainable Irrigation Techniques Using IoT, Springer, 1st Edition, 2020..

WEBSITES (with exact URL up to page)

7. https://www.ijraset.com/resea/Smart Irrigation Using IoT: A Review

8. https://www.ttjlawfirm.com/blog/2020/11/The Role of Technology in Water Conservation

9. https:/ Precision Agriculture and Smart Irrigation Systems

PROJECT SUMMARY

About Project

Title of the project	Smart Irrigation System
Semester	8th
Members	5
Team Leader	Lucky Soni
Describe role of every member in the project	Software and coding : Muskan khaira and Krishnpal Rajput Hardware and connectivity : Manish Ahirwar and Jayprakash Responsibilities: Oversee the project's progress, assign tasks, set deadlines, and ensure communication among team members. Coordinate testing phases, manage resources: Lucky Soni
What is the motivation for selecting this project?	The motivation for selecting the Smart Irrigation System project is to address water scarcity and promote sustainable agricultural practices by creating a system that efficiently manages water usage. This system aims to optimize irrigation monitoring environmental factors like soil moisture, temperature, and weather conditions, thereby ensuring crops receive adequate water while minimizing waste.
Project Type **(Desktop Application, Web Application, Mobile App, Web)**	IoT based project

Tools &Technologies

Programming language used	C, C++
Compiler used **(with version)**	Arduino
IDE used **(with version)**	Arduino IDE 2.3.3
Front End Technologies **(with version, wherever Applicable)**	NA
Back End Technologies **(with version, wherever applicable)**	NA
Database used **(with version)**	NA

Software Design& Coding

Is prototype of the software developed?	Yes
SDLC model followed (Waterfall, Agile, Spiral etc.)	Waterfall
Why above SDLC model is followed?	The Waterfall SDLC model was followed for the Smart Irrigation System project because the requirements were clearly defined and unlikely to change. This approach allowed for a structured, development process with minimal need for iteration or adjustments.
Justify that the SDLC model mentioned above is followed in the project.	The Waterfall SDLC model was followed in the Smart Irrigation System project due to its well-defined and stable requirements, allowing for a linear development process. Each phase was completed sequentially, ensuring minimal changes and a structured approach to project execution.
Software Design approach followed (Functional or Object Oriented)	The Object-Oriented Design (OOD) approach was followed in the Smart Irrigation System project. This approach allowed for better organization, modularity, and reusability of code by using classes and objects to model real-world entities and their interactions.
Name the diagrams developed (According to the Design approach followed)	
In case Object Oriented approach is followed, which of the OOPS principles are covered in design?	The design of the Smart Irrigation System project follows OOPS principles such as Encapsulation, Abstraction, Inheritance, and Polymorphism to ensure modularity, reusability, and flexibility in the system.
No. of Tiers (example 3-tier)	2-tier architecture
Total no. of front-end pages	-
Total no. of tables in database	-
Database in which Normal Form?	-
Are the entries in database encrypted?	-
Front end validations applied (Yes / No)	No
Session management done (in case of web applications)	-

Is application browser compatible **(in case of web applications)**	No
Exception handling done **(Yes / No)**	No
Commenting done in code **(Yes / No)**	Yes
Naming convention followed **(Yes / No)**	Yes
What difficulties faced during deployment of project?	Adjusting the soil moisture sensor and setting correct detection thresholds to avoid false readings or inaccurate water supply levels can be challenging. Ensuring the sensors are calibrated correctly to account for varying soil types, environmental conditions, and sensor drift is crucial to maintain accurate and reliable irrigation control.
Total no. of Use-cases	1
Give titles of Use-cases	

Project Requirements

MVC architecture followed **(Yes / No)**	No
If yes, write the name of MVC architecture followed **(MVC-1, MVC-2)**	-
Design Pattern used **(Yes / No)**	-5
If yes, write the name of Design Pattern used	-
Interface type **(CLI / GUI)**	GUI
No. of Actors	2
Name of Actors	User, System
Total no. of Functional Requirements	6
List few important non-Functional Requirements	Response Time, Data Refresh Rate, Power Efficiency

Testing

Which testing is performed? **(Manual or Automation)**	Manual

Is Beta testing done for this project?	No

Write project narrative covering above mentioned points

The Smart Irrigation System project, developed by a team of five in the 8th semester, aims to promote sustainable water management and optimize agricultural productivity. The IoT-based system continuously monitors environmental factors such as soil moisture, temperature, and weather conditions to determine the optimal irrigation schedule. This project is motivated by the pressing need to address water scarcity and improve irrigation efficiency in agriculture.

Lucky soni	0187CS211091
Muskan khaira	0187CS211104
Manish Ahirwar	0187CS211095
Krishnpal Rajput	0187CS211087
Jayprakash	0187CS211079

Guide Signature
Dr. Bhavna Gupta

APPENDIX-1 GLOSSARY OF TERMS

(In alphabetical order)

A

Arduino Nano — A microcontroller used in prototyping and building IoT systems, such as an anti-sleep alarm system, for driver fatigue detection.

B

Battery — A battery serves as the primary or backup power source for a system, ensuring continuous operation during power outages or in remote areas where direct power supply is unavailable. In a smart irrigation system, the battery powers components like sensors, controllers, and water pumps, enabling the system to function autonomously.

G

Ground — A well-grounded system also reduces the risk of electrical shocks and interference, especially in outdoor environments where environmental conditions can impact performance.

I

IoT (Internet of Things) — A network of physical devices (such as the ESP32, sensors, and relays) that connect to the internet to collect, exchange, and manage data.

R

Relay — An electrical component that allows the ESP32 to control high voltage devices (such as the fan and LED) by switching them on or off through a low- voltage signal.

V

VCC — A **Voltage Common Collector (VCC)** refers to the primary power supply point in an electronic circuit. It provides the necessary voltage to power all active components, such as sensors, controllers, and actuators in the system.