TABLE OF CONTENTS

Abstract……………………………………………………………………………….03

Chapter 1-Introduction…………………………………………………...….(05-6)

1.1 **Motivation……………………………………………………...……05**
1.2 **Problem Statement…………………………………………...…...05**
1.3 **Objectives…………………………………………………………05**
1.4 **Summary………………………………………………………….6**

Chapter 2-Literature Survey……………………………………………...…(08-09)

Chapter 3-System Requirements…………………………………………………11

Chapter 4-System Design and Development………………………………….…(13-20)

Chapter 5-Implementation ……………………………………………….……(21-25)

5.1 User Authentication and Management Module ………………….(22-23)

5.2 Sensor Data Management……………………………………......…(23-24)

5.3 Motor Control ………………………..……………………………....24

5.4 Real-Time Data Display ……………………………………….….…..25

5.5 Motor Control Interface ……………………………………….……25

Chapter 6- Result …………………………………………………...…..(27-31)

Chapter 7- Conclusion…………………………………………………..……33

Reference---34

List of Figures

Chapter No	Figure No	Description	Page NO
4	4.4	Hardware Components	17
	4.7.1	Deployment	20
	4.7.2	Testing	20
6	6.1	Login Page	27
	6.2	Registeration page	28
	6.3	Smart Irrigation Dashboard	29
	6.4.1	Hardware Connection	30
	6.4.2	ESP8266 Sensor Data	30
	6.4.3	Cloud data	30
	6.4.4	Dashboard	31

ABSTRACT

The smart irrigation system backend is an innovative platform built with Node.js and Express, designed to modernize agricultural practices through IoT and automation. It acts as the central hub for data management, user interaction, and device control, with advanced security measures such as bcrypt for password hashing and JSON Web Tokens (JWT) for secure session management. The backend integrates seamlessly with an ESP32 microcontroller, which collects crucial environmental data like soil moisture, temperature, and humidity. This data is stored in a MongoDB database for real-time monitoring and historical analysis, enabling efficient water management and supporting sustainable farming practices. By leveraging this integration, the system minimizes water waste while optimizing irrigation processes to promote environmental conservation.

A key feature of the backend is its dual-mode motor control functionality, which allows users to toggle between manual and automatic irrigation modes. Manual mode provides direct control over the motor, while automatic mode uses environmental data to determine optimal watering schedules, ensuring precision and efficiency. The backend also tracks the ESP32's connectivity status, ensuring reliable operation and alerting users to potential issues. With a secure API, authenticated users can access sensor data and control irrigation mechanisms remotely, enhancing usability while safeguarding sensitive information. Designed for scalability and adaptability, this backend supports farms of all sizes, offering actionable insights derived from real-time data. By integrating technology with sustainability, the smart irrigation system improves productivity, reduces resource waste, and addresses key challenges in modern agriculture.

CHAPTER 1

CHAPTER 1

INTRODUCTION

The Smart Irrigation System is an innovative approach to modernizing agricultural practices through automation and the Internet of Things (IoT). By integrating environmental sensors with real-time data monitoring and control capabilities, the system optimizes water usage, enhancing crop yield while conserving valuable resources. This report explores the design, development, and implementation of the backend architecture for such a system, which facilitates seamless interaction between users, devices, and databases.

1.1 Motivation

Agriculture is one of the most water-intensive industries, and with the growing concerns over water scarcity, efficient water management has become crucial. Traditional irrigation systems often result in overuse of water, leading to environmental degradation. The motivation behind developing a Smart Irrigation System is to address these issues by creating a solution that can automate irrigation based on real-time environmental data, ensuring that water is used optimally. This system aims to improve crop yields while reducing waste and promoting sustainability.

1.2 Problem Statement

The primary challenge in traditional irrigation systems is inefficient water usage, which leads to the depletion of water resources and increased costs for farmers. Moreover, manual irrigation requires constant monitoring, which can be labor-intensive and prone to human error. The problem is further exacerbated by unpredictable weather conditions and varying soil moisture levels, which make it difficult to determine the ideal times for irrigation. A smart system that automates the process based on accurate data can significantly enhance water efficiency and reduce manual labor.

1.3 Objective

The main objective of this project is to develop a Smart Irrigation System that uses IoT devices to collect real-time environmental data, which is then processed to automate irrigation decisions. The backend system, built using Node.js and Express, will handle user authentication, data storage, and control of irrigation motors. By providing accurate, real-time data and enabling remote control, the system will empower farmers to make informed decisions, optimize water usage, and reduce operational costs. The goal is to create a scalable solution that can be adapted to various agricultural settings, from small farms to large-scale operations.

1.4 Summary

This report outlines the design and development of the Smart Irrigation System backend, focusing on the integration of IoT sensors, user management, and motor control functionality. It describes how real-time environmental data, such as soil moisture, temperature, and humidity, is collected and stored in a database for easy access. The backend system ensures secure user interaction through authentication and allows farmers to control the irrigation system remotely. Through automation and data-driven decision-making, the system aims to optimize water usage, reduce waste, and enhance agricultural productivity.

CHAPTER 2

CHAPTER 2

LITERATURE SURVEY

The literature survey on IoT-based smart irrigation systems reveals the significant role of IoT technologies in revolutionizing agricultural practices. It highlights the integration of sensors for real-time data monitoring, remote control, and water conservation. Papers discussed explore the automation of irrigation, optimization of water usage, and the application of wireless sensor networks (WSNs). Challenges such as initial setup costs, scalability, and infrastructure maintenance are also mentioned. These studies collectively emphasize the potential of IoT to improve sustainability and efficiency in agriculture.

2.1 "Advancing IoT-Based Smart Irrigation" explores how IoT technologies are evolving in irrigation systems. It highlights automation, remote monitoring, and water conservation in agriculture through smart irrigation systems. The integration of IoT sensors and cloud platforms for real-time data analysis is a key focus.

- **Pros**: Highlights the automation, remote monitoring, and real-time data analysis.
- **Cons**: May face challenges in scalability and initial implementation costs.

Reference: https://ieeexplore.ieee.org/document/10683134

2.2 "Smart Irrigation System Using IoT" focuses on a prototype that uses sensors and Wi-Fi to provide real-time soil moisture data for automated irrigation. The system optimizes water use and improves crop growth while reducing human intervention.

- **Pros**: Optimizes water usage and reduces human intervention.
- **Cons**: Relies on stable Wi-Fi, which may be an issue in rural areas.

Reference: https://ieeexplore.ieee.org/document/10722739

2.3 "Design and Implementation of IoT-based Smart Irrigation System" discusses the design, development, and deployment of IoT systems in agriculture. It introduces the use of wireless sensor networks (WSNs) to monitor environmental factors like soil moisture, temperature, and humidity.

- **Pros**: Uses wireless sensor networks (WSNs) to monitor various environmental factors.
- **Cons**: Requires continuous maintenance of sensors and infrastructure.

Reference: https://ieeexplore.ieee.org/document/10456291

2.4 "A Survey of Smart Irrigation Systems for Agriculture" provides a comprehensive review of various smart irrigation approaches and technologies used in the agriculture sector. It compares systems based on IoT, sensors, and machine learning.

- **Pros**: Offers a comprehensive comparison of smart irrigation technologies.
- **Cons**: May lack in-depth details on practical implementation.

Reference: https://ieeexplore.ieee.org/document/10299321

2.5 "IoT-based Smart Agriculture: A Survey" reviews how IoT devices and technologies are revolutionizing agriculture, particularly through smart irrigation. The paper covers different sensor types and their integration with cloud platforms.

- **Pros:** Covers a wide range of IoT sensor types and cloud integration.
- **Cons:** Lacks focus on specific challenges faced during system deployment.

Reference: https://ieeexplore.ieee.org/document/10248213

CHAPTER 3

CHAPTER 3

SYSTEM REQUIREMENTS

Hardware Requirements:

1. **Microcontroller (ESP32/ESP8266)**: Handles sensor data collection and communicates with the server via Wi-Fi.
2. **Soil Moisture Sensor**: Measures soil moisture to determine irrigation needs.
3. **Temperature and Humidity Sensor (DHT11/DHT22)**: Monitors ambient temperature and humidity for optimal irrigation decisions.
4. **Relay Module**: Controls the water pump or motor.
5. **Power Supply**: Provides necessary power to all components.

Software Requirements:

1. **Operating System**: Linux or Windows for the server; Android/iOS for client devices.
2. **Backend Technologies**: Node.js, Express.js, MongoDB for handling sensor data, user management, and motor control.
3. **Frontend Technologies**: Web interface (HTML, CSS, JavaScript
4. **Communication Protocols**: HTTP/HTTPS and WebSocket for real-time data exchange; optionally MQTT for efficient messaging.
5. **Cloud Hosting/Local Server**: AWS, Google Cloud, or a local server for hosting the backend.
6. **Security**: JWT for authentication, SSL/TLS for secure communication.
7. **Version Control**: Git/GitHub for code management.

CHAPTER 4

CHAPTER 4

SYSTEM DESIGN AND DEVELOPMENT

The "IoT-Based Smart Irrigation System" represents an innovative leap toward addressing critical challenges in agricultural water management. Traditional irrigation methods often lead to water wastage and require significant manual intervention. This system seeks to modernize the process by leveraging Internet of Things (IoT) technology, enabling precise and automated irrigation control. The system integrates sensors to monitor soil moisture, temperature, and humidity in real-time, alongside cloud platforms for data storage and decision-making. A web-based user interface allows farmers to monitor conditions, receive notifications, and manually override automated actions if necessary. This combination of automation, analytics, and user engagement results in a system that not only saves water but also enhances crop health and reduces the farmer's workload. The following sections delve deeper into the system's design components, methodologies, and operational framework.

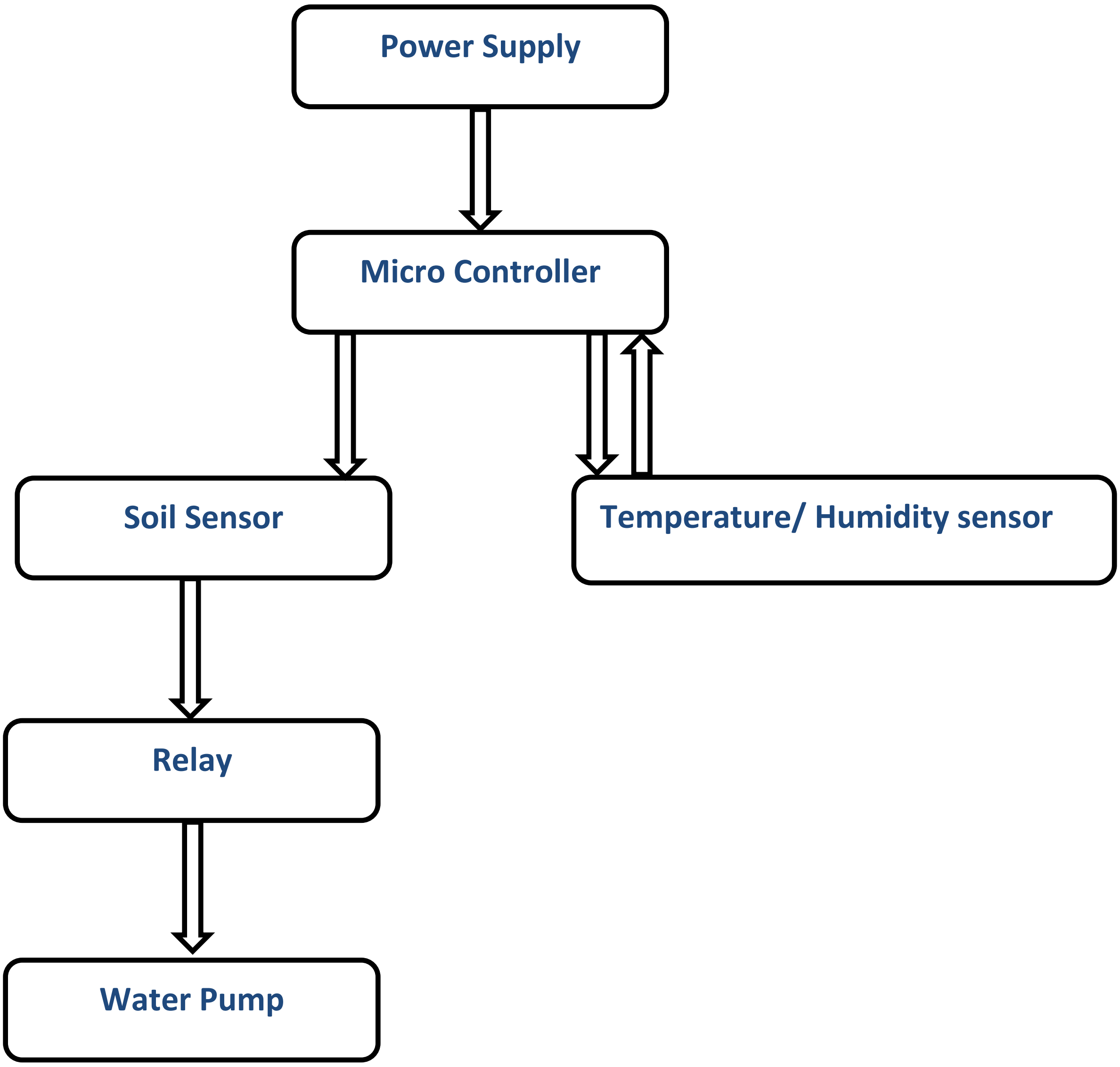
4.1 Block Diagrams

Block diagrams are a simplified representation of the system's architecture. They depict the main components and their interactions.

Explanation

The system consists of key components:

1. **Sensors**: Soil moisture sensors determine the water content in the soil, while temperature and humidity sensors provide environmental context.
2. **Microcontroller**: Acts as the central processing unit, interpreting sensor data and communicating with other components.
3. **Cloud Platform**: Handles data storage and advanced analytics, sending commands back to the microcontroller.
4. **Motor Driver and Pump**: Responsible for activating the irrigation system based on decisions made by the microcontroller or cloud analytics.
5. **User Dashboard**: Displays sensor data, system status, and manual control options.



4.2 Use Case Diagrams

Use Case Diagrams provide insights into the interactions between users and the system, highlighting functional requirements.

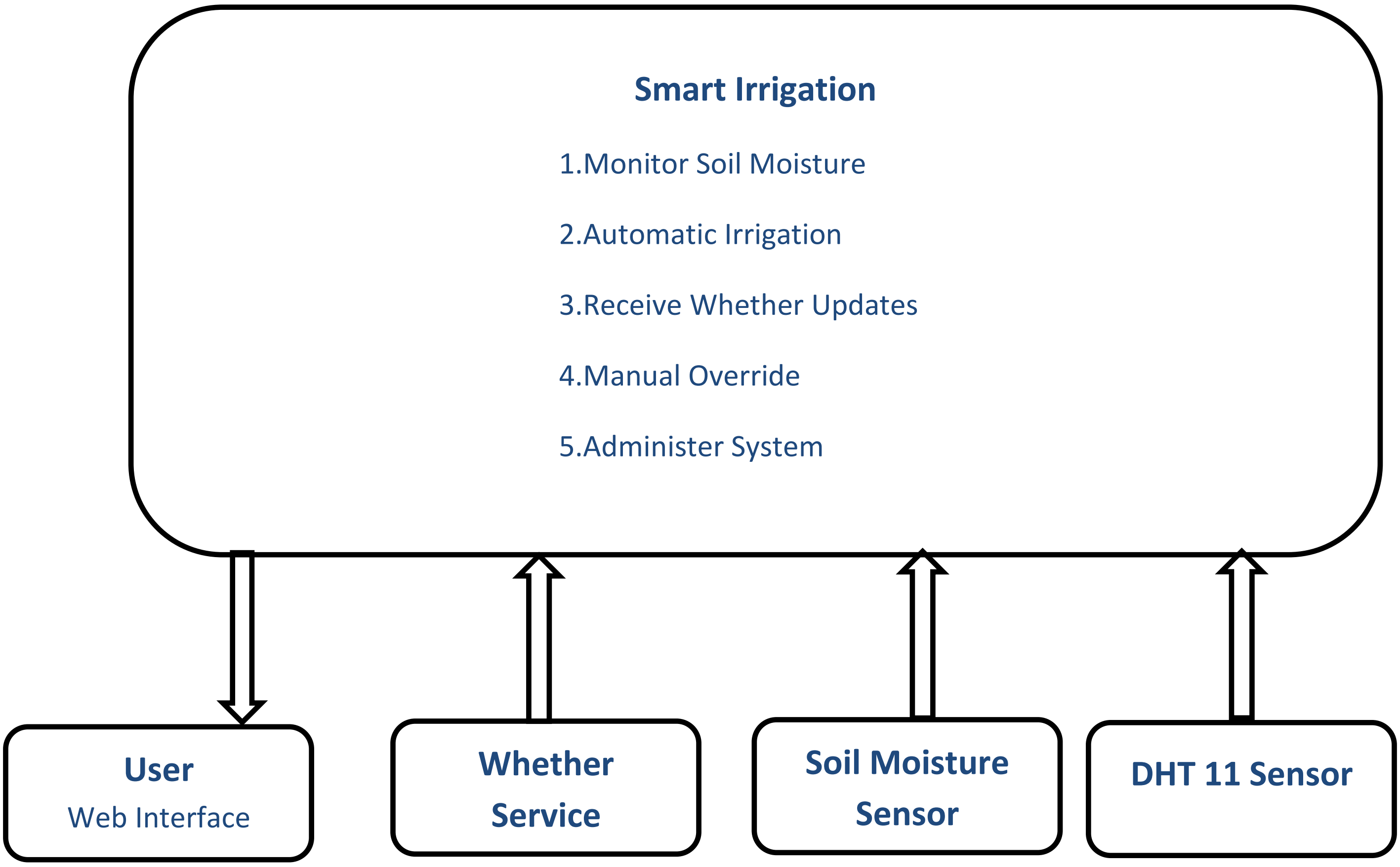
Explanation

The diagram identifies the key actors:

- **Farmer:** Uses the dashboard to monitor environmental conditions, control irrigation schedules, and receive alerts.
- **IoT System:** Operates autonomously, gathering data, making decisions, and executing irrigation.

The interactions outlined include:

- Monitoring soil moisture and weather conditions via the dashboard.
- Automatic irrigation triggered by predefined thresholds.



4.3 Data Flow Diagrams (DFD)

Data Flow Diagrams offer a detailed view of how information is processed within the system.

Level 0 DFD

- **Inputs:** Real-time sensor data from the field.
- **Processes:** Data logging, cloud analytics, and irrigation control decisions.
- **Outputs:** Commands to the motor driver and status updates to the user dashboard.

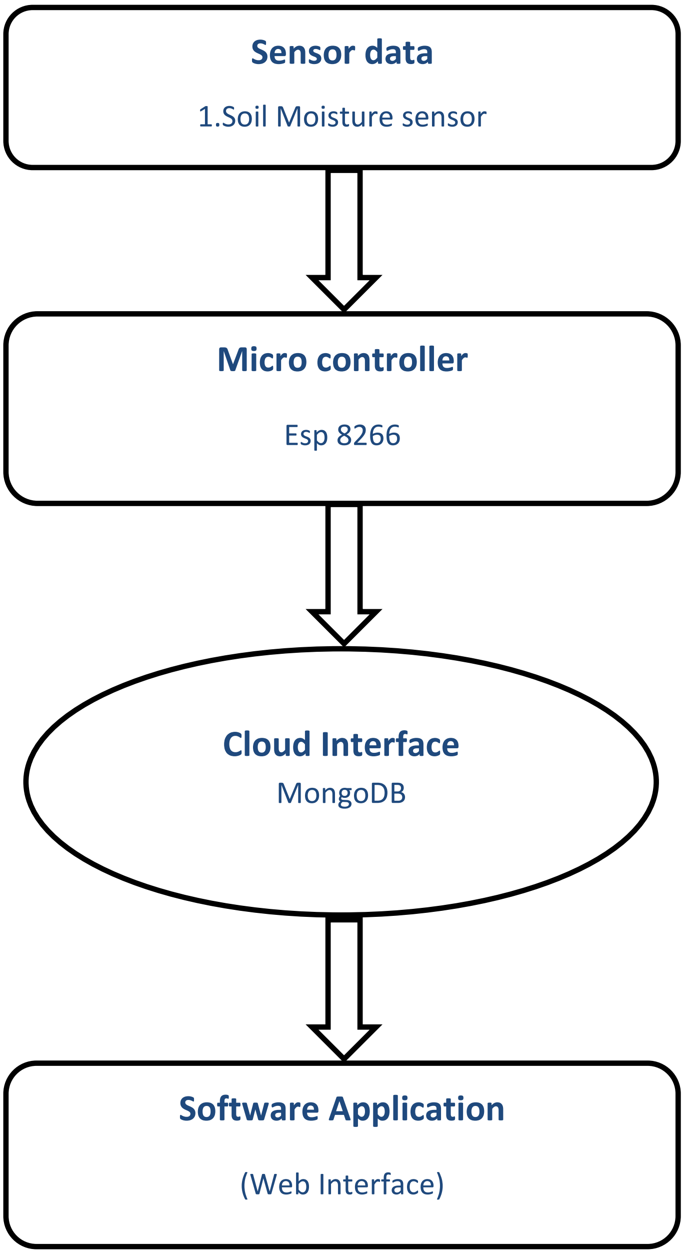
Level 1 DFD

- Expands on processes, detailing tasks such as:

    1. **Sensor Data Collection:** Capturing soil and weather metrics.

    2. **Cloud Synchronization:** Uploading data for centralized processing.

    3. **Dashboard Updates:** Providing actionable insights to the farmer.



4.4 Hardware Components

- **Sensors:** Soil moisture sensors determine the irrigation need, while environmental sensors provide context to optimize water usage, DHT11 Sensor to know the temperature and humidity.
- **Microcontroller (e.g., ESP8266):** Processes sensor inputs, communicates with the cloud, and executes irrigation commands.
- **Motor Driver and Relay Circuit:** Activates the water pump based on system decisions.
- **Power Supply:** Ensures uninterrupted operation even in remote agricultural fields.

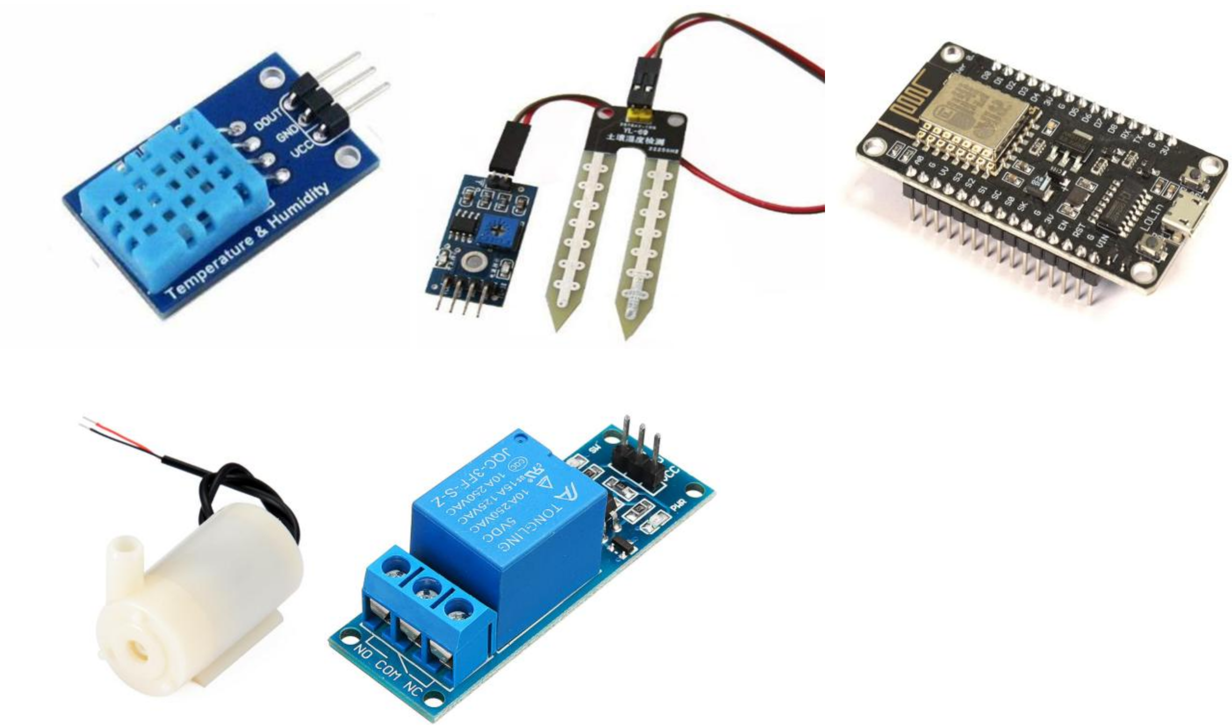


Fig 4.4:Hardware Components

4.5 Software Architecture

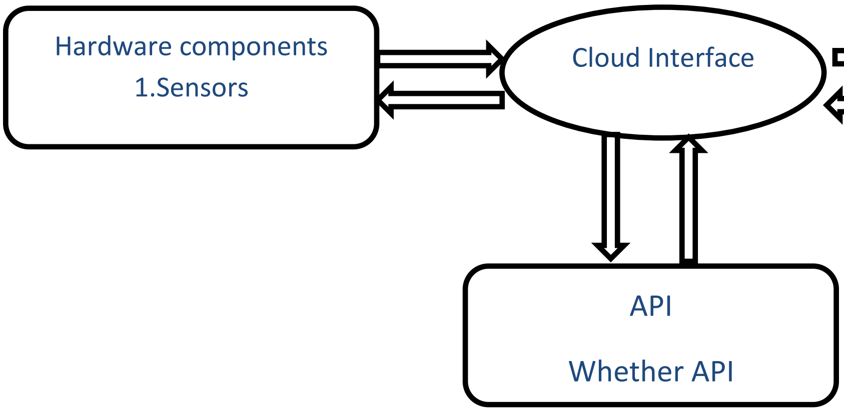
The software serves as the system's backbone, facilitating communication and decision-making.

Key modules include:

1. **Sensor Integration:** Reads real-time data and converts it into actionable metrics.
2. Cloud Services: Provides scalable storage and advanced analytics.
3. **User Dashboard:** Offers a user-friendly interface for monitoring and control.

4. **APIs:** Enable seamless interaction between hardware and cloud platforms.

This layered architecture ensures robust and scalable performance.

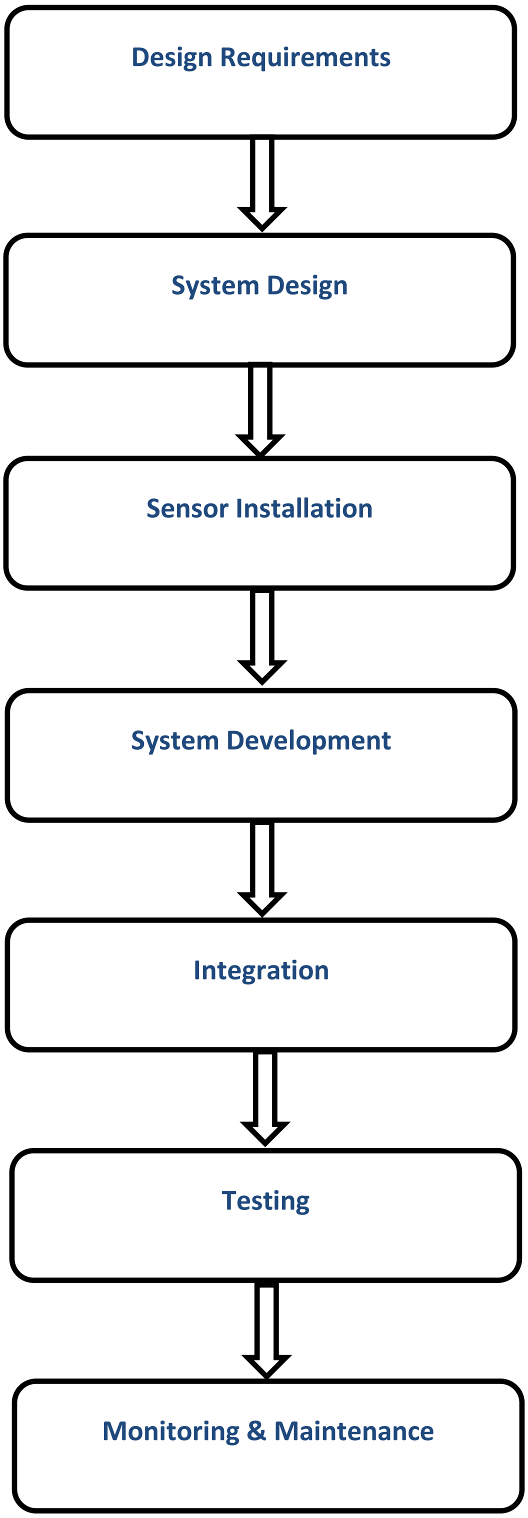


4.6 Methodology

The development methodology involves a systematic approach:

1. **Requirement Gathering:** Understanding user needs and agricultural challenges.
2. **System Design:** Creating detailed diagrams and selecting hardware/software components.
3. **Implementation:** Coding and integrating hardware with cloud services.
4. **Testing:** Ensuring reliability and accuracy in simulated and real-world conditions.
5. **Deployment:** Installing the system in fields and monitoring its performance.

This structured process ensures the system meets its intended objectives.



4.7 Deployment and Testing

Deployment and testing focus on ensuring the system’s reliability and effectiveness.

Deployment

- Sensors are installed in the field to capture real-time data.
- The microcontroller and cloud platform are configured for seamless communication.

The system is tested for usability through the user dashboard.(**Fig 4.7.1:Deployment)**

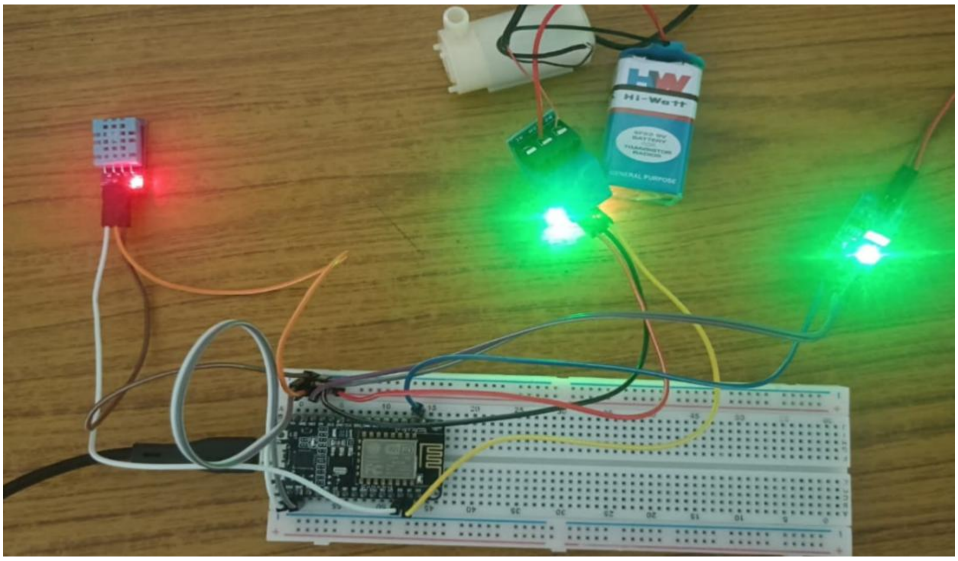


Fig 4.7.1:Deployment

Testing

- Sensor readings are validated against manual measurements.
- Cloud synchronization is tested for minimal latency.

The irrigation system is tested under varying environmental conditions.(**Fig 4.7.2:Testing)**

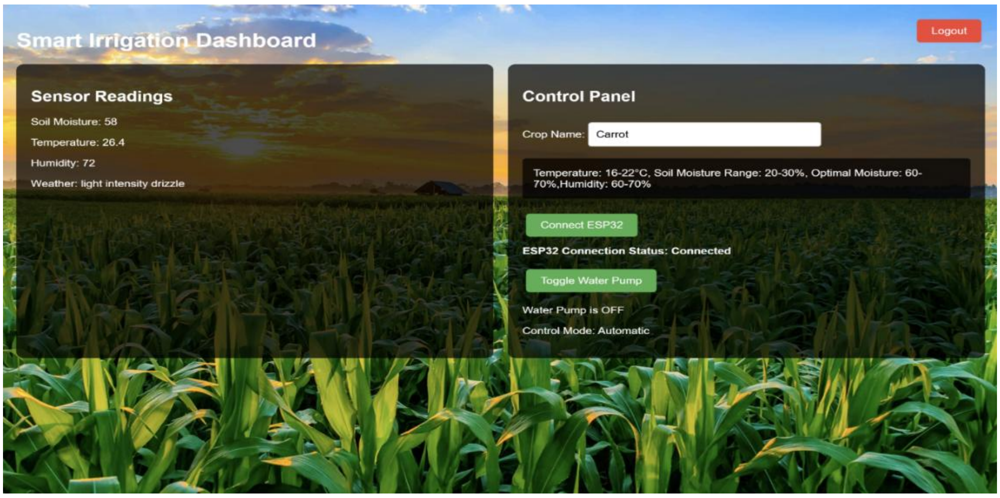


Fig 4.7.2:Testing

CHAPTER 5

CHAPTER 5

IMPLEMENTATION

The smart irrigation system implementation involves both backend and frontend development. The backend is built using Node.js and Express to handle user authentication, manage sensor data, and control the irrigation system. The frontend is a simple web interface built using HTML and JavaScript to allow users to monitor the system and control it in real-time. The system also integrates MongoDB for data storage and uses JWT (JSON Web Tokens) for secure user authentication.

List of Modules:

5.1 User Authentication and Management (Backend):

The user authentication module is responsible for managing user registration and login. It ensures that the username chosen during registration is unique and stores passwords securely using bcrypt for hashing. When users attempt to log in, the system checks their credentials and generates a JWT token to authenticate their future requests, ensuring secure access to the system's features.

Registeration

```
Function register(username, password):
    Check if username already exists in the database
    If exists, return error message
    Encrypt password using bcrypt
    Save username and encrypted password to the database
    Return success message
```

Fig 5.1.1: User Authentication and Management Registeration

Login

```
Function login(username, password):
    Find user by username in the database
    If not found, return error message
    Compare provided password with stored encrypted password using
bcrypt
    If they match, generate JWT token with user ID and expiration
time
    Return JWT token
```

Fig 5.1.2: User Authentication and Management Login

5.2 Sensor Data Management (Backend):

This module is designed to handle the reception and storage of sensor data sent from the ESP32 device. It collects real-time data on soil moisture, temperature, and humidity, storing each record with a timestamp in a MongoDB database. This enables easy tracking of environmental conditions over time and helps maintain accurate data for future use, such as triggering irrigation actions.

Receive Sensor Data

```
Function receiveSensorData(soilMoisture, temperature, humidity):
    Capture current time
    Save sensor data to MongoDB (soilMoisture, temperature,
humidity, timestamp)
    Update last connection time for ESP32
    Return success message
```

Fig 5.2: Sensor Data Management

Real time Sensor Data

```
Function fetchSensorData():
    Verify JWT token from request header
    If invalid, return error message
    Check ESP32 connection status based on last connection time
    Fetch the latest sensor data from the database
    If data exists, return soilMoisture, temperature, humidity,
and ESP32 connection status
    If no data, return error message
```

Fig 5.2: Real Time Sensor Data Management

5.3 Motor Control (Backend):

The motor control module provides the functionality to control the irrigation system's motor. It allows toggling the motor’s state (on/off) and supports switching between manual and automatic control modes. This flexibility ensures that the irrigation system can be managed either by user input (manual mode) or based on predefined conditions (automatic mode), enhancing user convenience.

Motor Control

```
Function toggleMotor(state, mode):
    Set motor state (on/off) based on input
    Set mode to manual or automatic
    Return the current motor state and mode
```

Fig 5.3: Motor Control

5.4 Real-Time Data Display (Frontend):

The real-time data display module on the frontend fetches the latest sensor data from the backend and visually presents it to the user. It displays key metrics such as soil moisture, temperature, and humidity, allowing users to monitor the conditions of their irrigation system in real time. This module enhances user experience by providing an intuitive interface for system monitoring.

Fetch Sensor Data (Front End)

```
Function fetchSensorData():
    Send GET request to backend with Authorization token
    Parse JSON response and update HTML elements with soil
moisture, temperature, and humidity
```

Fig 5.4: Real-Time Data Display

5.5 Motor Control Interface (Frontend):

This module on the frontend enables users to interact with the irrigation system by controlling the motor. It provides buttons to toggle the motor on/off and to switch between manual and automatic modes. The motor control interface is designed to be simple and intuitive, allowing users to control the system easily from a web-based interface.

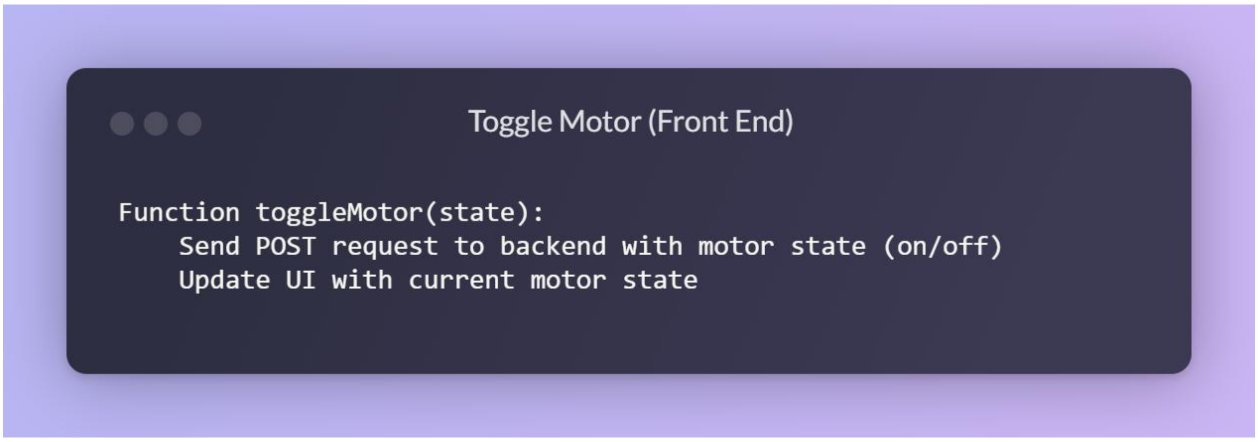


Fig 5.5:Real-Time Data Display

CHAPTER 6

CHAPTER 6

RESULTS

The IoT-based smart irrigation system demonstrated remarkable efficiency and automation in managing agricultural water usage. By integrating sensors, Wi-Fi modules, and cloud platforms, the system optimized water distribution based on real-time soil moisture data. This approach significantly reduced water wastage while enhancing crop health by ensuring optimal soil conditions. The automation minimized manual intervention, reducing labor efforts and errors in irrigation management. The inclusion of remote monitoring capabilities via a user-friendly dashboard allowed farmers to track environmental parameters like temperature, humidity, and soil moisture seamlessly. Additionally, the ESP32 connection tracking ensured uninterrupted communication and real-time data updates, enhancing system reliability. Overall, the system effectively demonstrated its potential to improve agricultural productivity and sustainability.

6.1 Login Image

Explanation:

- The login page is the first point of interaction for users in the system.
- Users need to enter their username and password to authenticate and gain access to the system.
- This page typically includes two input fields (username, password) and a button to submit the credentials.

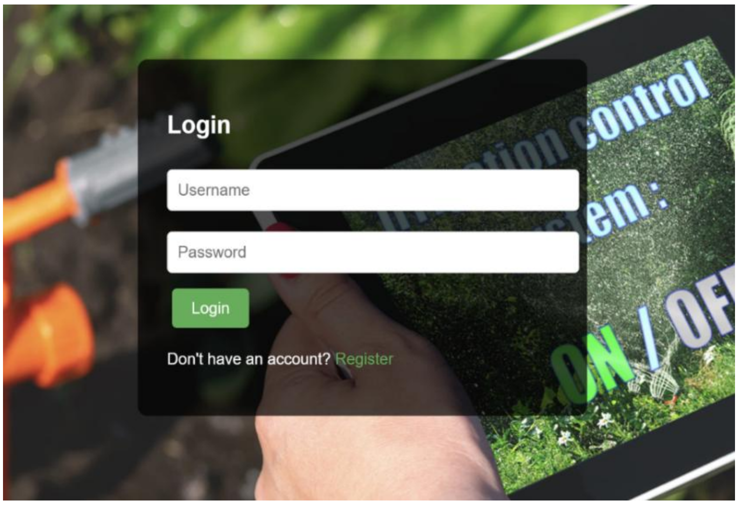


Fig 6.1: Login Page

6.2 Register Image

Explanation:

The register page allows new users to sign up and create an account.

- The user needs to provide a unique username, password, and confirm their password for registration.
- The registration page typically includes a form with these fields and a button to submit the form .(**Fig 6.2:Registeratio page)**

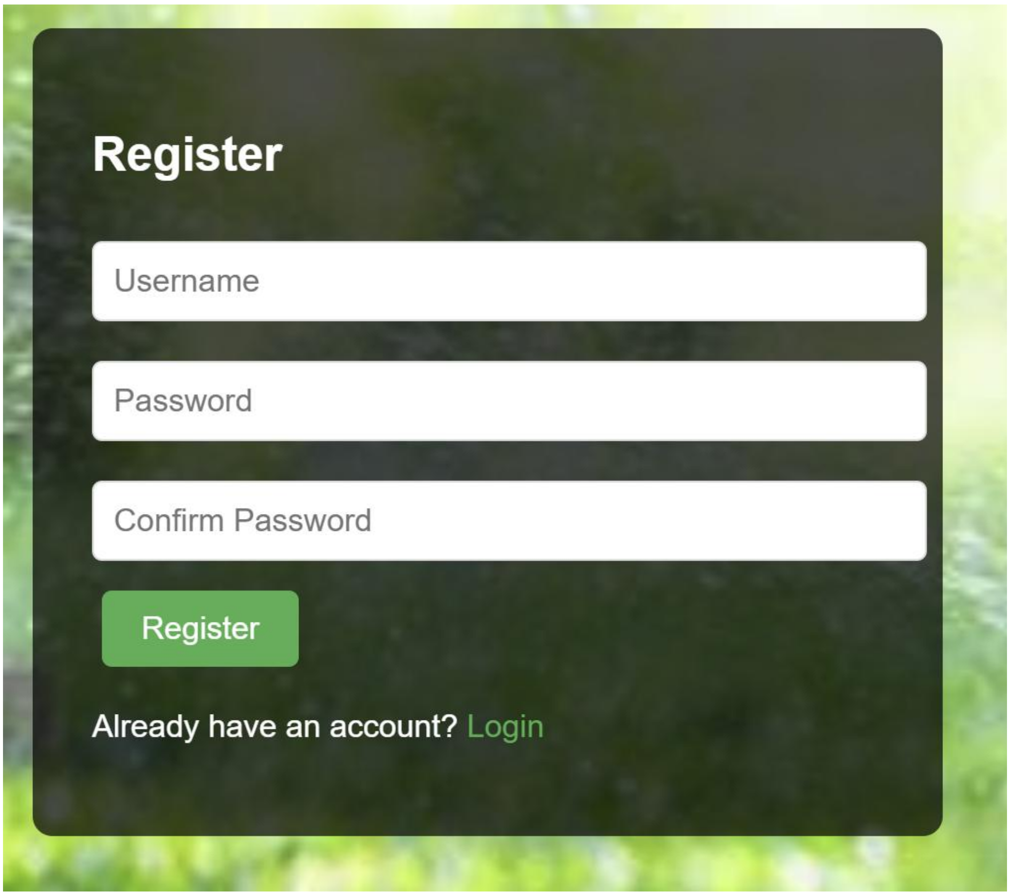


Fig 6.2:Registeration page

6.3 Sensor Data Image

- The sensor data page displays real-time data sent by the ESP32 sensor.
- This data typically includes soil moisture, temperature, and humidity values.

The page also includes status information, such as whether the ESP32 is connected and receiving data.(**Fig 6.3:Smart Irrigation Dashboard)**

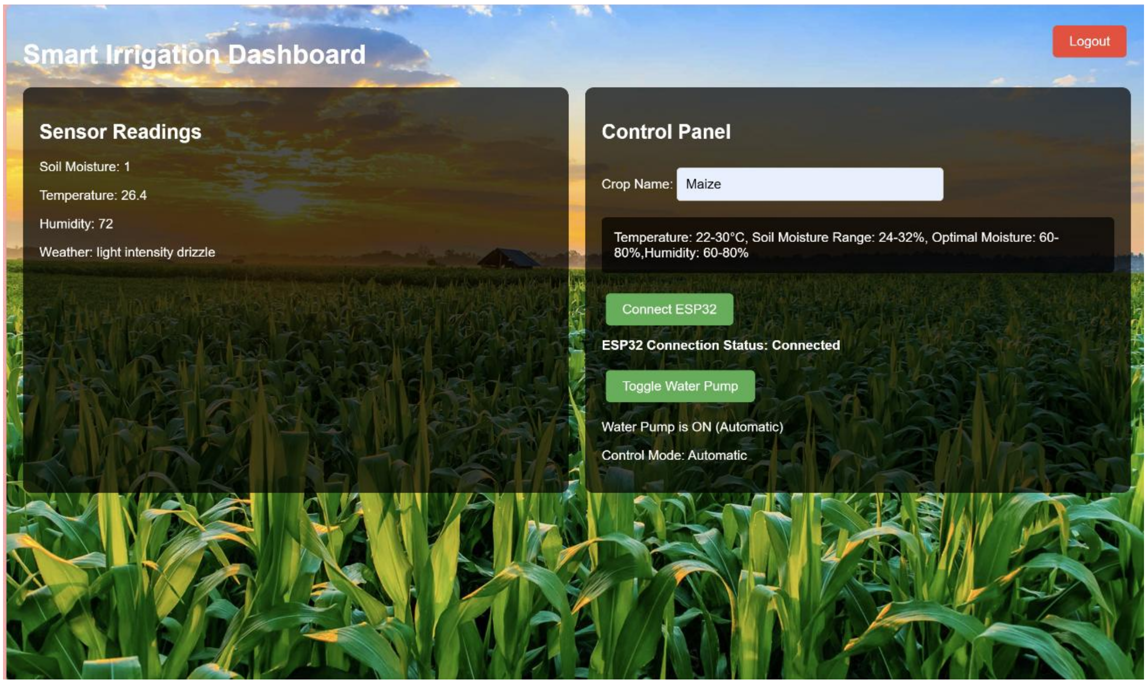


Fig 6.3:Smart Irrigation Dashboard

6.4 Complete Working Connection Image

- The system architecture image explains how all components of the system are connected and communicate with each other.
- **Components include:**
    - ESP32: Collects sensor data.(Fig 6.4.1 Hardware Connection)
    - Node.js Server: Handles HTTP requests, processes data, and interacts with the database.(Fig 6.4.2 Hardware Connection)
    - MongoDB Database: Stores sensor data.(Fig 6.4.3 Cloud Data)

- User Interface (UI): Displays real-time sensor data and allows motor control. (Fig 6.4.4 dash Board)

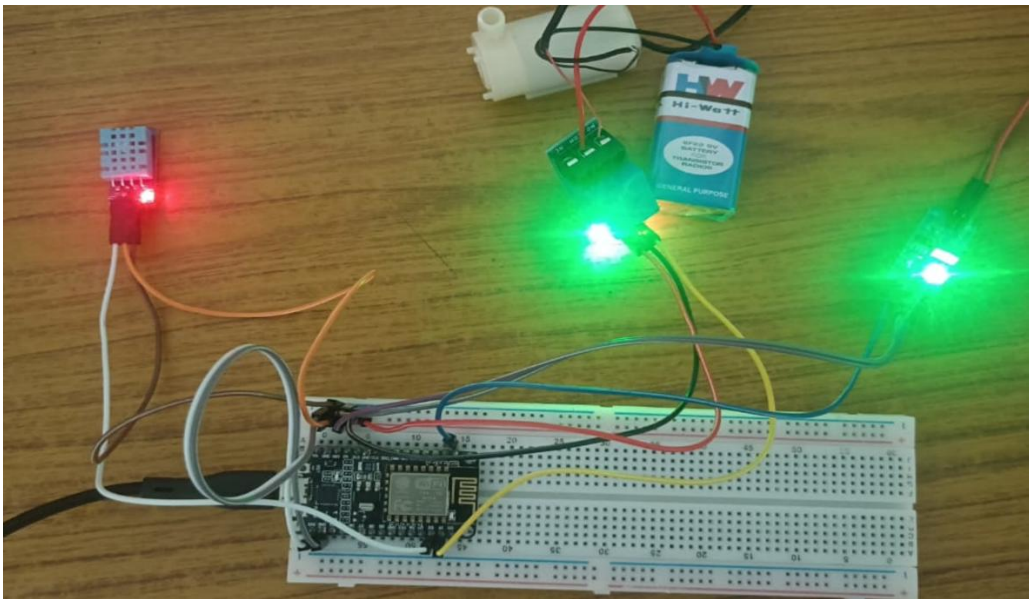


Fig 6.4.1:Hardware Connection

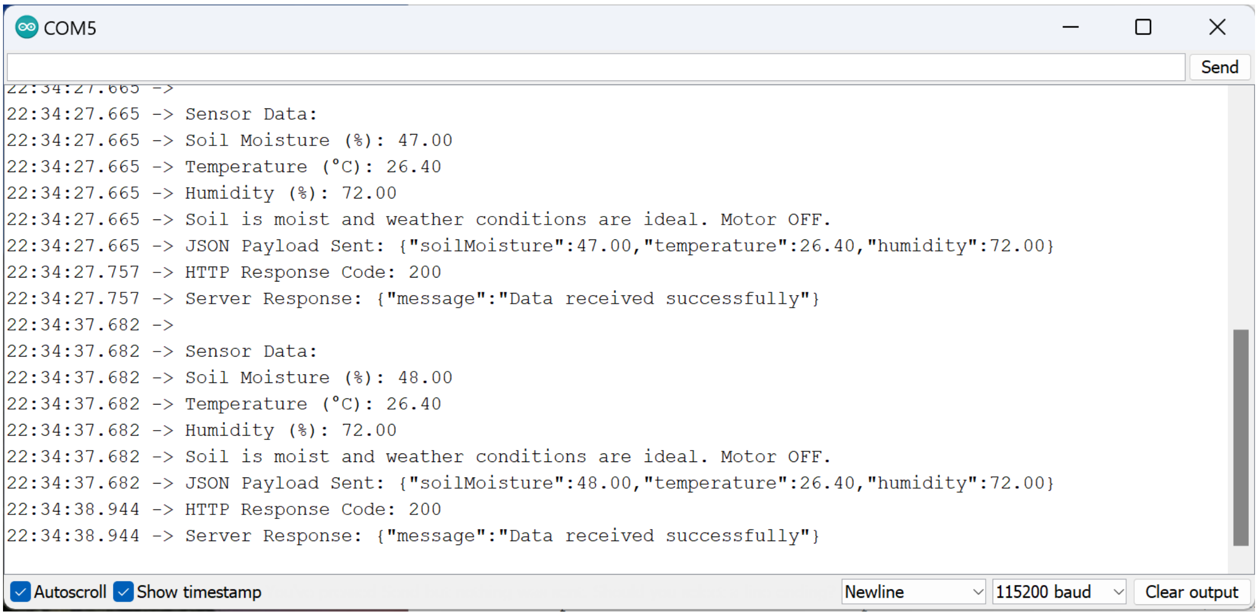


Fig 6.4.2:ESP8266 Sensor Data

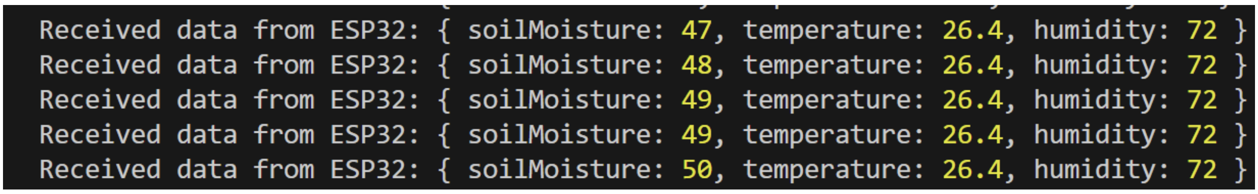


Fig 6.4.3:Cloud data

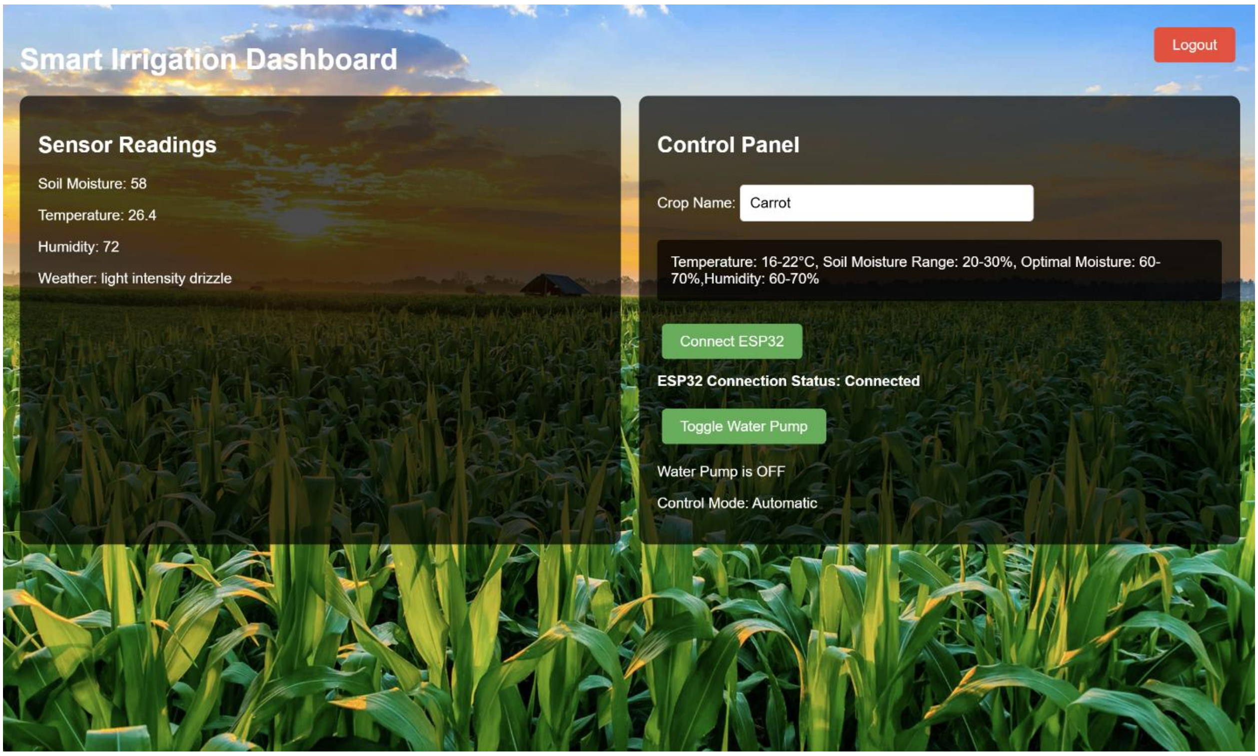


Fig 6.4.4:Dashboard

CHAPTER 7

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

The IoT-based smart irrigation system bridges the gap between traditional agricultural methods and modern technological advancements. By utilizing sensors to monitor critical parameters such as soil moisture, temperature, and humidity, the system offers precise, data-driven irrigation management. This results in substantial water conservation, as water is delivered only when and where it is needed, reducing waste and ensuring sustainability.

Additionally, the system enhances crop health by maintaining optimal growth conditions, which leads to better yield quality and quantity. Automation reduces human intervention, allowing farmers to focus on other essential activities while ensuring consistency in irrigation. This approach not only saves time and effort but also minimizes human error, which is crucial for large-scale farming operations.

The project demonstrates how emerging technologies such as IoT and cloud computing can effectively address agricultural challenges, ensuring resource optimization and environmental conservation. Its real-time monitoring and decision-making capabilities are a testament to the transformative impact of technology in agriculture, paving the way for smarter, more efficient farming practices.

Future Enhancement:

- **Machine Learning Integration:** Implement predictive algorithms to optimize irrigation schedules based on historical data and environmental factors.
- **Extended Sensor Networks:** Increase the number of sensors to cover larger agricultural fields for more granular monitoring.
- **Mobile Application:** Develop a user-friendly mobile app to provide real-time updates and remote control of irrigation systems.
- **Renewable Energy Sources:** Incorporate solar-powered systems to ensure sustainability and reduce dependency on conventional energy.
- **Enhanced Security:** Strengthen data encryption and authentication mechanisms to prevent unauthorized access and ensure system reliability.

REFERENCES

- "Advancing IoT-Based Smart Irrigation"
  This paper explores the evolution of IoT technologies in irrigation systems, emphasizing automation, remote monitoring, and efficient water usage.
  Reference: https://ieeexplore.ieee.org/document/10683134
- "Smart Irrigation System Using IoT"
  A prototype-focused paper discussing how sensors and Wi-Fi enable real-time soil moisture monitoring for automated irrigation.
  Reference: https://ieeexplore.ieee.org/document/10722739
- "Design and Implementation of IoT-Based Smart Irrigation System"
  This paper covers the integration of wireless sensor networks (WSNs) for monitoring environmental conditions and automating irrigation processes.
  Reference: https://ieeexplore.ieee.org/document/10456291
- "A Survey of Smart Irrigation Systems for Agriculture"
  A comprehensive review of various technologies in smart irrigation, comparing systems that leverage IoT, sensors, and machine learning.
  Reference: https://ieeexplore.ieee.org/document/10299321
- "IoT-Based Smart Agriculture: A Survey"
  Focuses on the impact of IoT devices in agriculture, particularly in irrigation, with discussions on sensors and cloud platform integration.
  Reference: https://ieeexplore.ieee.org/document/10248213