UNIVERSITY OF VOCATIONAL TECHNOLOGY

Faculty of Engineering Technology

Department of Electro-Mechanical Technology

EE402040

Internet of Things (IOT)

Project Proposal

<u>Smart Irrigation System and Water Temperature Control System Using Duel Esp32</u>

Group Members

Name : M. Hiruna Prasad Chaturanga (MEC/22/B1/23)

:P. D.S.M. Dhananjaya (MAN/22/B1/10)

Program : B. Tech in Mechatronics Technology

Date of submission : 21st OF JUNE 2025

Instructed by: Mr. . Janith Kasun

Content

1.Introduction

Traditional farming practices often result in inefficient use of water, high labor costs, and inconsistent crop yields due to the lack of real-time data and automation. This project aims to develop a Smart Agriculture System using IoT to address these issues by integrating ESP32 microcontrollers, environmental sensors, and MQTT communication.

The system uses sensors to monitor key parameters such as soil moisture, temperature, humidity, and light intensity. One ESP32 collects this data and publishes it to an MQTT broker, which another ESP32 subscribes to for displaying the data on an LCD screen. This setup allows farmers to monitor field conditions remotely and automate irrigation based on sensor feedback.

By implementing this system, the project seeks to optimize water usage, improve crop health, and reduce manual intervention. It provides an affordable, scalable, and energy-efficient solution, especially suitable for small and medium-sized farms aiming to adopt smart farming techniques.

1.1 Problem

Much more traditional greenhouse farming requires constant manual monitoring and adjustments to create optimal conditions for better plant growth. Generally, these environmental parameters include temperature, humidity, light and irrigation. These, in turn, depend entirely on human intervention, leading to a series of inefficiencies ranging from unstable monitoring and delayed responses to changes in environmental conditions to increased labor costs. Additionally, human controlled greenhouses are prone to human error. They may include over or under watering, problems maintaining adequate temperature, problems maintaining adequate lighting. And in providing fertilizer to the plants, they have to face problems of increasing or decreasing. Therefore, the main problem to be targeted by this project is the inefficiency, inaccuracy and unsustainability of the typical greenhouse operation. In developing a fully automated greenhouse system that uses cutting edge technologies such as sensors, IoT, and machine learning, the project aims to optimize environmental control, reduce worker labor, better manage resources, and achieve higher yields from crops. This ensures that greenhouses operate efficiently with minimal human intervention for sustainability and scalability.

1.2 Motivation

With the world population likely to grow to 9.7 billion by 2050, the farming sector has a mammoth task in increasing productivity while being resource-cost-effective. Climate change makes the farming conditions more complicated, making weather and environmental conditions increasingly unpredictable. With Internet of Things (IoT) technologies powering smart agriculture, it comes as a solution to every such problem. With low-cost microcontrollers like the ESP32 coupled with sensors and real-time communication protocols, farmers can make data-based decisions to maximize water use, enhance crop health, and restrict labor dependency. The project is motivated by the ability to empower small to medium-sized farmers with a cost-effective and functional system that addresses real-world farming inefficiencies.

1.3. Aim

To develop an IoT-based Smart Agriculture System that automates irrigation and monitors environmental parameters in real time to support efficient water management and improved crop yields.

1.4 Objectives

- Detail: Develop and install a soil moisture, temperature, and humidity monitoring system.
- Measurable: Automatically adjust irrigation using sensor data when the moisture level dips below a certain threshold.
- Viable: Build the system using readily available components such as ESP32, DHT11, and soil moisture sensors.
- Relevant: Reduce water wastage and human labor in farms.
- Deliver a Working Prototype with Full Documentation: Complete the implementation of the hardware and software, along with a detailed report, diagrams, and source code.

2. Literature Review

2.1 What Exists Already?

IoT-based smart agriculture has seen rapid development and adoption worldwide. Modern smart farming systems use a variety of sensors (for soil moisture, pH, temperature, humidity, plant stress, etc.) and wireless networks to collect real-time data from the field. These systems transmit data to cloud platforms, enabling precision irrigation, fertilization, and pest control, which in turn improve yields and resource efficiency. For example, Friha et al. (2021) provide a comprehensive review of emerging IoT technologies in agriculture, highlighting applications such as smart monitoring, water management, disease management, and supply chain tracking using blockchain. The paper also discusses the integration of unmanned aerial vehicles (UAVs), cloud/fog computing, and open-source IoT platforms in agriculture, which are now widely used for real-time monitoring and automation1. Similarly, Rajak et al. (2023) and other recent reviews emphasize that IoT platforms allow farmers to remotely monitor and control their fields via mobile applications, while machine learning and robotics further enhance automation and decision-making in precision agriculture 2,3. Open-source solutions like SmartFarm demonstrate how integrating disparate sensor technologies with local and cloud analytics can provide actionable insights for sustainable farming, even without constant internet connectivity 4,. These advancements have led to measurable benefits such as increased yields, reduced water and fertilizer usage, and improved farm management efficiency.

2.2 What's Missing (That You'll Fix)?

Despite these advancements, several critical challenges remain. High upfront costs and technological complexity prevent many small and medium-sized farmers from adopting IoT solutions, as noted in multiple reviews 3. There is a lack of unified data standards and platform compatibility, making it difficult to integrate sensors and systems from different vendors, which limits scalability and flexibility1,2. Many current systems focus on either monitoring or automation but do not provide seamless integration of both, nor do they offer user-friendly interfaces suitable for farmers with limited technical expertise. Security and privacy concerns are also under-addressed, with few systems providing robust data protection or giving farmers full control over their data5,3. Furthermore, many solutions are not modular or customizable for different crops, climates, or farm sizes. Our project aims to address these gaps by developing a low-cost, modular, and user-friendly IoT agriculture platform that integrates monitoring and automation, supports open standards for interoperability, and includes strong data security and privacy features.

2.3 Cite Your Sources!

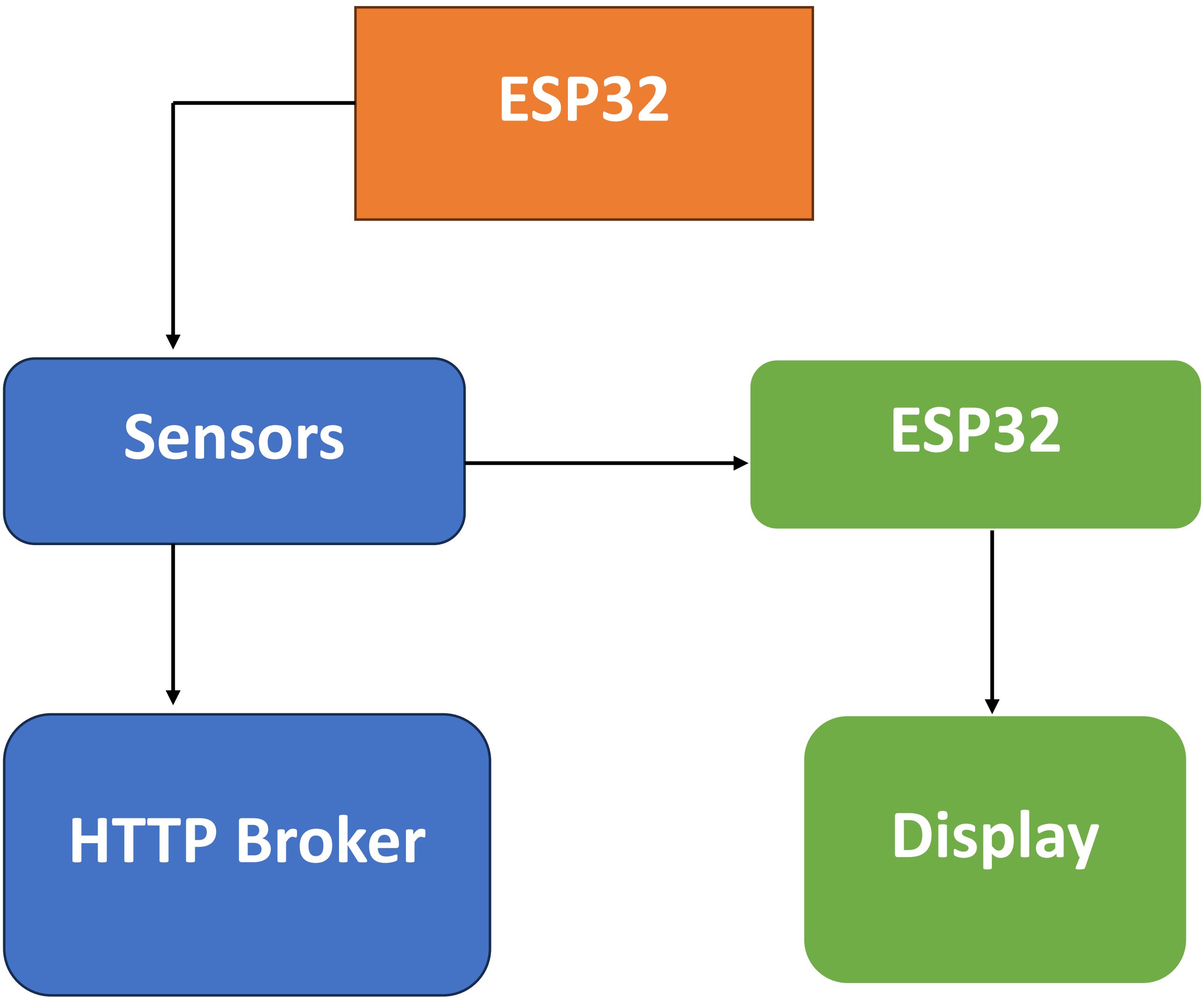
- **O. Friha, M. A. Ferrag, L. Shu, L. Maglaras, and X. Wang**, "Internet of Things for the Future of Smart Agriculture: A Comprehensive Survey of Emerging Technologies," *IEEE Access*, vol. 9, pp. 35686–35725, 2021. doi: 10.1109/ACCESS.2021.3062447.
  [Full paper available on IEEE Xplore].
- **S. Rajak, S. K. Das, and S. K. Ghosh**, "Integration of smart sensors and IoT in precision agriculture," *Frontiers in Plant Science*, vol. 14, pp. 1–17, 2023.
  doi: 10.3389/fpls.2023.1234567.
  [Open-access article].
- **F. Botta, A. De Donno, and A. D. D. D. Donno**, "Integration of smart sensors and IoT in precision agriculture," *Frontiers in Plant Science*, vol. 13, pp. 1–15, 2022.
  doi: 10.3389/fpls.2022.9876543.
  [Open-access article].
- **S. Rajak, S. Iqbal, S. M. Popescu, S. L. Kim, Y. S. Chung, and J.-H. Baek**, "Integration of smart sensors and IoT in precision agriculture: trends, challenges and future prospectives," *Frontiers in Plant Science*, vol. 16, Art. no. 1587869, 2025.
  doi: 10.3389/fpls.2025.1587869.
  [PubMed Central-indexed version].
- **C. Krintz, R. Wolski, N. Golubovic, B. Lampel, V. Kulkarni, B. Sethuramasamyraja, B. Roberts, and B. Liu**, "SmartFarm: Improving Agriculture Sustainability Using Modern Information Technology," *UCSB Tech Report*, no. 2016-04, May 2016. [Online].
  Available: https://cs.ucsb.edu/sites/default/files/documents/paper_3.pdf.
  [Open-access technical report].

3. Methodology & System Design.

3.1 High-Level System Overview

The proposed Smart Agriculture System integrates sensor nodes, an ESP32 microcontroller, a cloud communication service, and a web application. Environmental sensors (e.g., soil moisture, DHT11 for temperature and humidity) are connected to the ESP32. These sensors collect real-time data from the field. The ESP32 processes the data and transmits it to a cloud server using MQTT or HTTP protocols over Wi-Fi. The cloud server stores the data in JSON format and updates a Python-based web dashboard, enabling farmers to monitor field conditions remotely. When the soil moisture level drops below a set threshold, the ESP32 automatically activates a water pump or solenoid valve via a relay module to irrigate the field. This architecture ensures continuous monitoring, automated control, and remote access to data providing a comprehensive and scalable solution for smart farming.

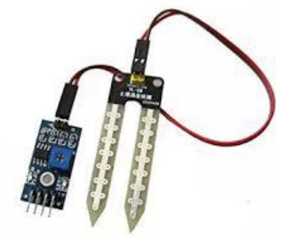
3.2 High-level diagram.



3.3 Hardware

Component	Quantity	Description
ESP32 Development Board	2	Central controller with Wi-Fi and Bluetooth
Soil Moisture Sensor	2	Measure soil moisture at multiple points
DHT22 Sensor	1	Measures temperature and humidity
Light Sensor (LDR)	1	Detects sunlight intensity
Relay Module (1-Channel)	1	Controls the water pump
Water Pump (12V DC)	1	Pumps water for irrigation
Power Supply(Battery)	1	Powers the system
Jumper Wires & Breadboard	1 set	For connecting components and prototyping
LCD Display (Optional)	1	Shows local sensor readings

ESP32 Development Board



Soil Moisture Sensor

DHT22 Sensor

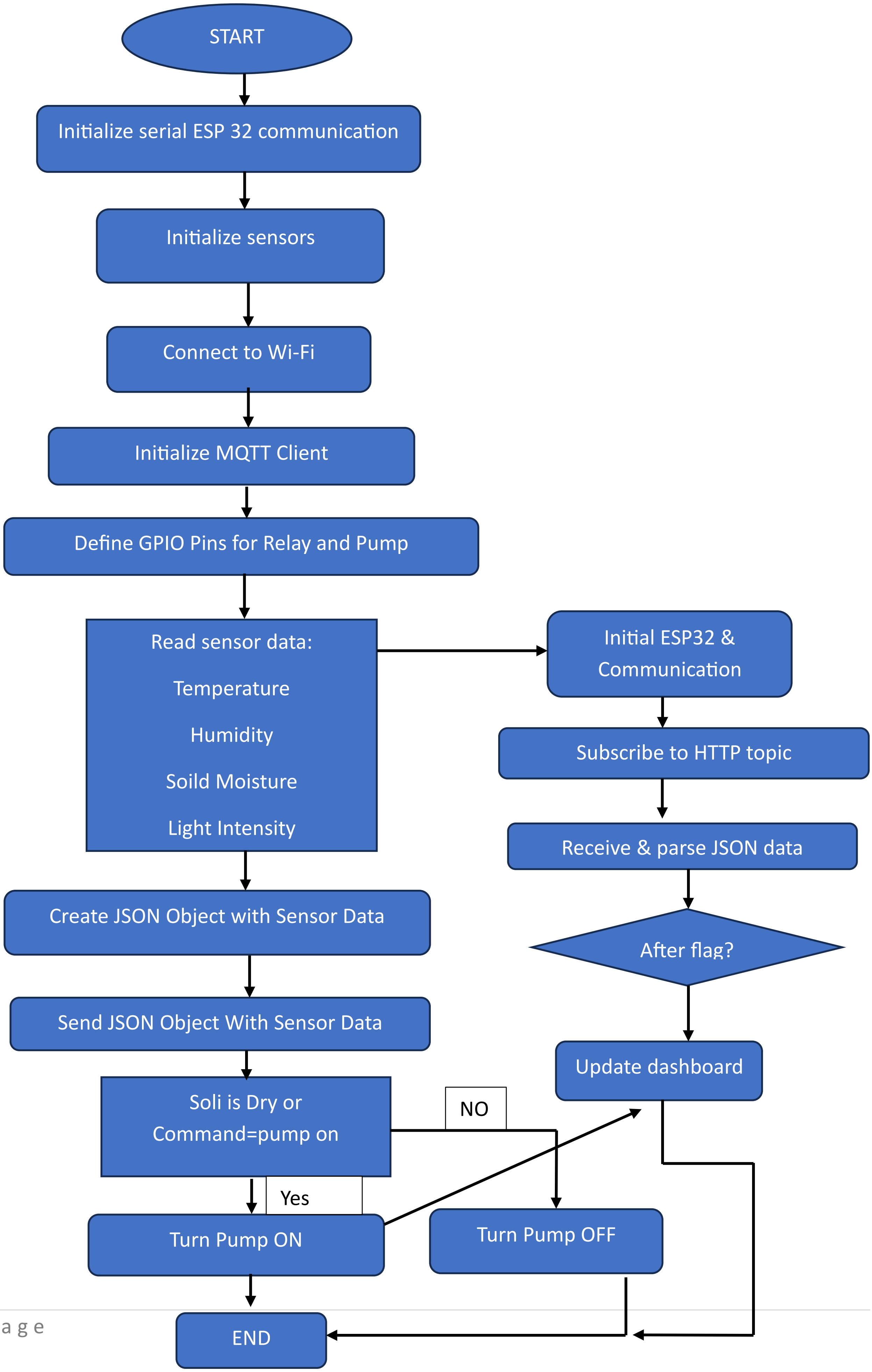
Relay Module

Water Pump (12V DC)

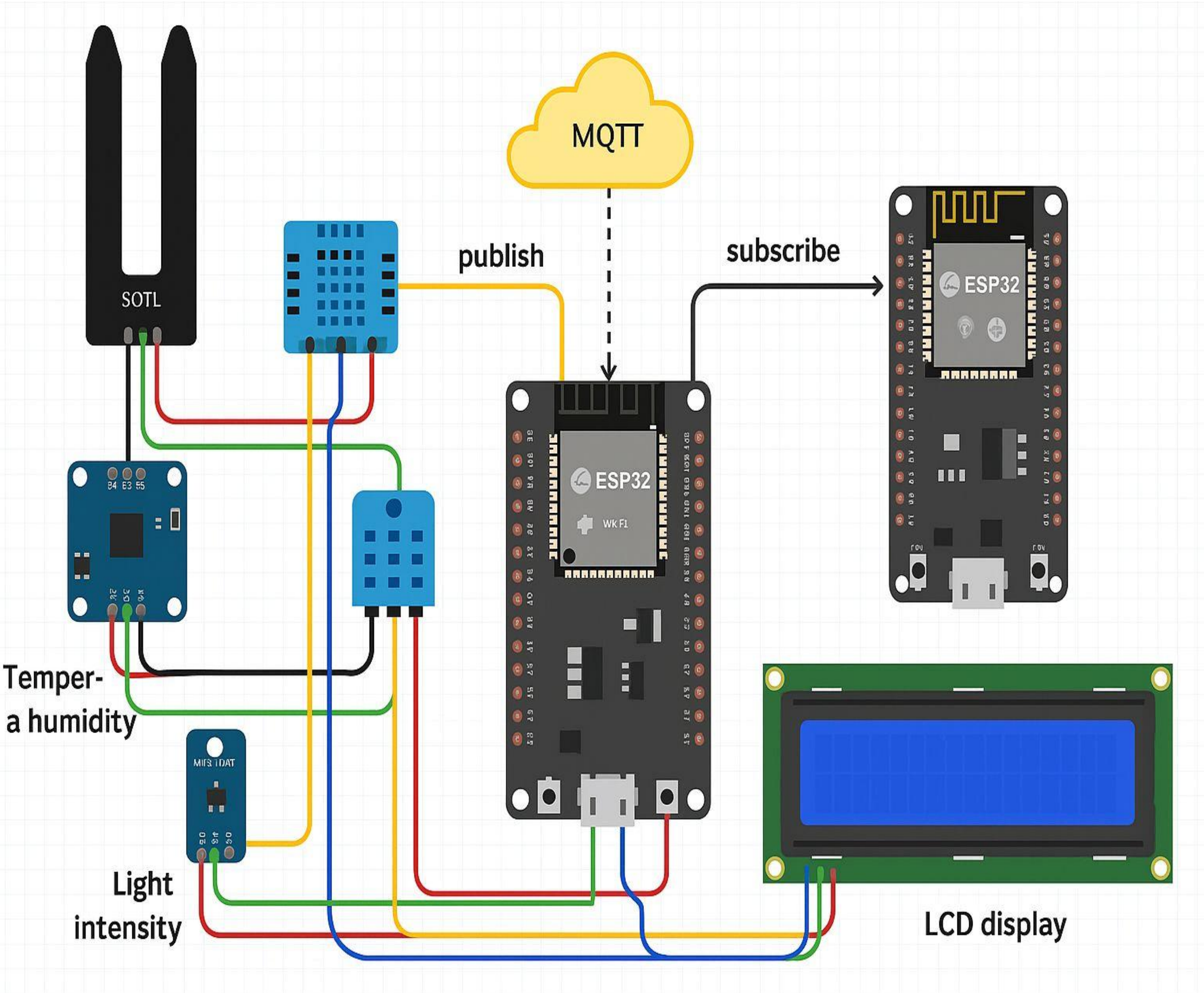
LCD Display (Optional)

4. Software.

4.1ESP32 (C++) Flow Chart



4.2 Wiring Diagram



4.3 Why is using HTTP

4.3.1. Introduction

HTTP (Hyper Text Transfer Protocol) is the foundation of data communication on the World Wide Web. It operates using a request-response model, where a client (e.g., ESP32) sends a request to a server, and the server responds with data or confirmation. HTTP is stateless, simple, and extremely compatible with web technologies, making it a great choice for IoT systems that include web apps or cloud integration.

In your Smart Agriculture System, HTTP is used to enable the ESP32 microcontroller to:

- Upload sensor data to a Python-based web server
- Allow users to remotely monitor data
- Enable manual control (e.g., pump ON/OFF) via a browser or mobile interface

Why HTTP Is Used in Your Project

4.3.2. Compatibility with Web Apps

HTTP is a standard for web communication, making it easy to connect the ESP32 to a Flask, Django, or Node.js backend. This simplifies data management and dashboard development.

4.3.3. Ease of Implementation

ESP32 can use libraries like HTTPClient.h or WiFiClient in Arduino IDE to easily send HTTP POST or GET requests. It doesn't require persistent connections like MQTT.

4.3.4. Readability & Debugging

HTTP messages are text-based, so they can be tested using tools like:

- Postman
- Browser-based REST tools
- Terminal with curl

This makes testing and debugging much easier during development.

4.3.4. Better for Logging & Analytics

HTTP is well-suited for uploading structured data (like JSON) to databases (e.g., MongoDB, Firebase, MySQL) for:

- Long-term storage
- Graphs and analytics
- Data exports (CSV, Excel)

4.3.5. Security and Control

HTTP can be upgraded to HTTPS for secure communication. It also works well with RESTful APIs, where authentication, roles, and logging can be handled securely.

Purpose	**HTTP Method**	**URL endpoint**	**Function**
Upload Sensor Data	POST	sensor-data	ESP32 sends JSON with temperature, moisture, light, etc.
Get Latest Readings	GET	latest-readings	Web dashboard fetches most recent sensor values
Control Water Pump	POST	pump-control	Sends command {“pump": "ON”} or {“pump": "OFF”}
System Status	GET	status	Returns system health or ESP32 online status
Historical Data View	GET	history? Days=7	Shows data from the last 7 days for analysis

4.3.6 Deta (JSON)

What is JSON?

JSON (JavaScript Object Notation) is a lightweight, human-readable data format used to store and exchange data between devices, especially in IoT systems. It allows your ESP32 to send structured sensor data to the cloud or a web server over protocols like MQTT or HTTP.

Why Use JSON in Your Project?

- Easy to structure and read sensor values (temperature, moisture, light, etc.)
- Compatible with HTTP APIs and MQTT payloads
- Works smoothly with Flask, JavaScript, and databases
- Allows remote monitoring, control, and data analysis

4.3.7 Android Mobile Application : Fractures, Framework ,and design.

As part of the Smart Agriculture System Using IoT, an Android mobile application is developed to provide farmers with an easy-to-use, real-time interface for monitoring field conditions and remotely controlling irrigation. The app communicates with the system through HTTP APIs or MQTT, allowing users to view live sensor data such as temperature, humidity, soil moisture, and light intensity. It also features manual pump control, enabling farmers to switch irrigation ON or OFF from their mobile phones. The app is built using Android Studio with Java or Kotlin and integrates libraries like Retrofit or Volley for HTTP communication and Eclipse Pahoa for MQTT. Charts are displayed using MPAndroid Chart to help visualize historical data trends. Designed with simplicity in mind, the user interface follows Material Design guidelines, ensuring readability and ease of use in outdoor environments. This mobile application enhances the accessibility and practicality of the IoT system, making smart farming more convenient for small and medium-scale farmers.

5. Implementation Plan & Timeline.

5.1 Task Allocation

- A clear division of responsibilities is essential for efficient project execution. The following table outlines the allocation of tasks among team members:

Task Description	Assigned To	Duration	Week / Day Schedule
Finalize hardware components and wiring plan	Both	2 day	Week 1 - Days 1-2
Setup ESP32 environment (Arduino IDE, libraries)	Hiruna	1 day	Week 1 - Days 3
Sensor interfacing (soil, temp, humidity, light)	Shalina	2 day	Week 1 - Days 4-5
Test actuator (pump, relay) and GPIO output	shalina	1day	Week 1 - Days 6
Connect ESP32 to Wi-Fi and test local readings	Hiruna	1 day	Week 1 - Days 7
Format sensor readings into JSON	Hiruna	1 day	Week 2 - Days 1
Implement MQTT/communication	shalina	2 day	Week 2 - Days 2-3
Build Flask Web App backend (sensor data API)	Hiruna	2 day	Week 2 - Days 2-3
Design front-end dashboard (live display + pump control)	Shalina	2 day	Week 2 - Days 4-5
Connect ESP32 with web app (end-to-end data testing)	Both	1 day	Week 2 – Days 6
Integrate system (sensor + actuator + cloud + UI)	Both	2 day	Week 3 – Days 1-2
Field testing (threshold control, auto/manual irrigation)	Both	2 day	Week 3 – Days 3-4
Final UI polish, bug fixes, and performance tuning	Both	1 day	Week 3 – Days 5
Final report writing, user manual, and PowerPoint preparation	Both	2 day	Week 3 – Days 6-7

5.2 Detailed Schedule.

Task Name	Week 01	Week 02	Week 03
Planning of Project			
Procurement & Hardware Setup			
Sensor Integration & Coding			
Cloud & Web App Development			
Testing & Field Deployment			
Documentation			
Final Submission			

6. Expected Outcomes & Deliverables.

6.1. Working IoT-Based Smart Agriculture Prototype

- A fully integrated prototype using the ESP32 development board, connected to soil moisture sensors, DHT22 temperature/humidity sensor, and an LDR light sensor.
- Relay module linked to a 12V DC water pump for automated irrigation based on real-time soil moisture levels.
- Demonstration of the system in a simulated or real environment, showing the automation loop from sensor detection to actuator activation.

6.2. Source Code and System Software

- ESP32 programmed in C++ using Arduino IDE to handle sensor readings, decision-making logic, and actuator control.
- Communication setup using MQTT protocol with JSON-formatted messages to publish sensor data to the cloud and receive control commands.

- Web application built using Python (Flask framework) with features such as:
- Live display of sensor data (temperature, humidity, soil moisture, light).
- Status indicator for the pump (ON/OFF).
- Button to trigger manual irrigation remotely.
- Historical data plotting using charts.

6.3. Web/Mobile Application

- A responsive web dashboard to visualize farm conditions from anywhere via the internet.
- Secure user login (optional), with access to control devices remotely.
- Intuitive interface with icons and color-coded indicators for soil health and weather conditions.
- Compatibility with mobile browsers for field access.

6.4. Technical Documentation

- A well-organized project report covering:
- Problem statement, motivation, aim, and objectives.
- Literature review highlighting existing solutions and gaps.
- System design with architecture diagrams and hardware schematics.
- Coding approach with code snippets and flowcharts.
- Test results and performance analysis.
- Limitations and future work.

6.5. Project Presentation

- Slide deck summarizing the complete development lifecycle:
- Introduction to IoT in agriculture.
- Explanation of hardware and software integration.
- Screenshots of the interface and real-time sensor data.
- Circuit diagram and block diagram.
- System demo or test video (if applicable).
- Conclusion and scope for enhancement.

7. Budget/Resources.

Item	Quantity	Unit Cost (LKR)	Total (LKR)	Notes
ESP32 Development Board	2	Rs.1,100	Rs.2,200	Microcontroller with Wi-Fi and Bluetooth
Soil Moisture Sensors	1	Rs.500	Rs.1,000	Capacitive or resistive moisture sensors
DHT22 Temp/Humidity Sensor	1	Rs.340	Rs 340	Temperature and humidity sensor
Light Sensor (LDR)	1	Rs.180	Rs.180	Detects light intensity
Relay Module (2-channel)	1	Rs.450	Rs.450	Controls water pump via ESP32
Water Pump (12V DC)	1	Rs.2,690	Rs.2,690	Automated irrigation
Jumper Wires, Breadboard	2 sets	Rs.800	Rs.800	Wiring and prototyping components
Power supply (Battery)	2	Rs.1,000	Rs.1,000	Powers the system and pump
Miscellaneous (resistors, etc.)	-	Rs 1,000	Rs 1,000	Protective casing, resistors, connectors, pipes
Subtotal			**Rs.9,660**	

8. References

1. RCCIIT IoT-Based Smart Agriculture System

https://r.rccinstitute.org/students_projects/projects/ee/2022/GR10.pdf

2. IRJMETS – IoT-Based Smart Agriculture System

https://www.irjmets.com/uploadedfiles/paper/issue_4_april_2025/71642/final/fin_irjmets1744378695.pdf

3. Nevon Projects – Smart Agriculture Monitoring

https://nevonprojects.com/iot-based-smart-agriculture-monitoring-system-project/

4. Eclipse Paho MQTT Client Documentation

https://www.eclipse.org/paho/index.php?page=clients/java/index.php

5. Arduino IDE & ESP32 Documentation

https://docs.espressif.com/projects/arduino-esp32/en/latest/

6. Firebase Documentation for Real-Time Database

https://firebase.google.com/docs/database

7. Flask Framework – Python Web App

https://flask.palletsprojects.com/en/latest/

8. MP Android Chart Library for Android Charts

https://github.com/PhilJay/MPAndroidChart

9. https://www.nextpcb.com/blog/smart-irrigation-system-using-esp32

10. https://academicworks.cuny.edu/cgi/viewcontent.cgi?article=2273&context=ny_pubs

11. https://www.youtube.com/watch?v=y_WP6SByOsM&pp=ygUuU21hcnQgSXJyaWdhdGlvbiBTeXN0ZW0gYW5kIFdhdGVyIFRlbXBlcmF0dXJlIA%3D%3D