B.Tech 8th Semester

B.TECH FINAL SEMESTER PROJECT

ON

Smart Farm Monitoring System

By
Vishwas Kapoor(12115020)
Vanshika Gandhi(12115037)
Tanisha Goel(12115069)

Under the supervision of

Dr. Ghanapriya Singh
Asst. Professor, Department of Electronics and Communication
National Institute of Technology, Kurukshetra

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY KURUKSHETRA
JAN 2025-MAY 2025

CERTIFICATE

We hereby certify that the work which is being presented during this B.Tech Project report entitled "Smart Farm Monitoring System", in partial fulfillment of the wants of the award of the Bachelor of Technology in Electronics and Communication Engineering is an authentic record of our very own work distributed during a period from January, 2025 to April,2025 under the supervision of Dr.Ghanapriya Singh, Asst. Professor, Electronics and Communication Engineering Department. The matter presented in this project report has not been submitted for the award of any other degree elsewhere.

Signature of the Candidates

Vishwas Kapoor(12115020)

Vanshika Gandhi(12115037)

Tanisha Goel(12115069)

This is to certify that the above statement made by the candidates is correct to the simplest of my knowledge.

Signature of the Faculty Mentor

Dr. Ghanapriya Singh, Asst. Professor, ECE Department
NIT Kurukshetra

ACKNOWLEDGEMENT

With the successful completion of our final year project titled *Smart Farm Monitoring System*, we take this opportunity to express our sincere gratitude to all those who supported and guided us throughout this journey.

We are deeply thankful to Dr.Ghanapriya Singh, Asst. Professor, Department of Electronics and Communication Engineering, NIT Kurukshetra, for his constant guidance, encouragement, and technical expertise. His mentorship played a crucial role in shaping the direction of the project and helping us overcome key challenges.

We also extend our heartfelt thanks to our teammates and fellow students, whose collaboration, support, and shared efforts significantly contributed to the completion of this project. The numerous brainstorming discussions and technical sessions helped us refine our ideas and implementation strategies.

We acknowledge the support of the faculty members and lab staff of the ECE Department, NIT Kurukshetra, for providing the infrastructure and environment required for successful experimentation and testing.

Lastly, we would like to express our deepest gratitude to our families, whose unwavering moral support, patience, and encouragement provided the foundation upon which we built this work. Their faith in us has been instrumental in every step of this journey.

This project has not only enhanced our technical knowledge but also taught us the importance of teamwork, perseverance, and problem-solving—lessons we will carry forward in our careers ahead.

ABSTRACT

In response to the growing demand for sustainable and effective farming methods, smart farming solutions have been developed. This project offers a Smart Farm Monitoring System that provides real-time data on three important soil parameters—temperature, moisture, and humidity—to help farmers make well-informed decisions. In order to improve crop management and enable early interventions, the primary objective was to develop a mobile application that sends this data straight to the farmer's phone.

Reducing power consumption was one of the project's major challenges, which allowed the system to be deployed in contexts with limited resources and remote locations. Two communication techniques—a LoRa-based system and the MQTT protocol via Wi-Fi—were put into practice and assessed in order to address this. The two approaches are thoroughly compared in this thesis, with an emphasis on how well they perform in terms of dependability, range, and power efficiency.

The findings show that LoRa provides a more sustainable option for ongoing agricultural monitoring due to its long-range and low-power characteristics, particularly in rural locations with little access to reliable internet connectivity. The results of this project provide the groundwork for smart agriculture systems that are both scalable and energy-efficient.

CONTENTS

CERTIFICATE
ACKNOWLEDGEMENT
ABSTRACT
CHAPTER 1: INTRODUCTION
- 1.1 Background
- 1.2 Problem Statement
- 1.3 Objectives
- 1.4 Scope Of The Project
- 1.5 Methodology Overview
- 1.6 Motivation
- 1.7 Contributions
- 1.8 System Overview

CHAPTER 2: LITERATURE REVIEW
- 2.1 Introduction
- 2.2 Smart Agriculture and Sensor-Based Monitoring
- 2.3 Communication Technologies in IoT-Based Agriculture
- 2.4 Power Efficiency in IoT Monitoring Systems
- 2.5 Gaps Identified in Existing Systems
- 2.6 Conclusion

CHAPTER 3: WIRELESS COMMUNICATION TECHNOLOGIES — MQTT OVER WI-FI AND LORA
- 3.1 Introduction
- 3.2 MQTT (Message Queuing Telemetry Transport) over Wi-Fi
  - 3.2.1 What is MQTT?
  - 3.2.2 MQTT Architecture
  - 3.2.3 Properties of MQTT over Wi-Fi
  - 3.2.4 Advantages
  - 3.2.5 Limitations
- 3.3 LoRa (Long Range Radio)
  - 3.3.1 LoRa Architecture (P2P)
  - 3.3.2 Properties of LoRa
  - 3.3.3 Advantages
  - 3.3.4 Limitations
- 3.4 Comparison Table
- 3.5 Justification of Use

CHAPTER 4: COMPONENTS
- 4.1 Introduction
- 4.2 Hardware Components
  - 4.2.1 NodeMCU ESP8266

4.2.2 WeMOS D1 Mini
4.2.3 DHT11 Sensor
4.2.4 Soil Moisture Capacitive Sensor
4.2.5 SX1278 LoRa Module
4.2.6 Breadboard and Jumper Wires
4.2.7 Power Source
4.3 Software Components
4.3.1 Arduino IDE
4.3.2 Blynk IoT Platform
4.4 Summary

CHAPTER 5: CONNECTIONS AND SYSTEM OVERVIEW
5.1 Introduction
5.2 System Architecture Overview for WiFi Setup
5.2.1 MQTT over Wi-Fi Setup
5.2.2 Hardware Connection for Wi-Fi setup
5.2.3 Arduino Code
5.3 System architecture overview for LoRa Setup
5.3.1 LoRa communication setup
5.3.2 Hardware Connections for LoRa setup
5.3.3 Pin Connections
5.3.4 Arduino Code for LoRa setup

CHAPTER 6: POWER MANAGEMENT
6.1 Overview
6.2 Power Consumption Optimization
6.2.1 Deep Sleep Mode
6.2.2 Low-Frequency Data Collection
6.2.3 Minimal Processing
6.3 Power Consumption Analysis
6.3.1 Objective
6.3.2 Test Setup
6.3.3 Measurement Procedure
6.3.4 Results and Comparison
6.3.5 Observations
6.4 Conclusion

List of Figures

Figure 1.1 System Overview
Figure 3.1 NodeMCU ESP8266
Figure 3.2 WeMOS D1 Mini
Figure 3.3 DHT11 Sensor
Figure 3.4 Soil Moisture Sensor
Figure 3.5 SX1278
Figure 4.1 Architecture Flow Diagram
Figure 4.2 Connection of various hardware components
Figure 4.3 Sender Code
Figure 4.4 Receiver Code
Figure 4.5 Data Flow Diagram
Figure 4.6 Connection of various hardware components
Figure 4.7 Arduino Code for transmitter side
Figure 4.8 Arduino Code for receiver side

List of Tables

Table 3.1 MQTT Over Wifi Characteristics
Table 3.2 LoRa Performance Metrics
Table 3.3 MQTT vs LoRa Feature Comparison
Table 4.1 Technical Specifications of NodeMCU
Table 4.2 WeMOS D1 Mini Technical Specifications
Table 4.3 DHT11 humidity sensor specifications
Table 5.1 Pin Connections
Table 5.2 Libraries Used
Table 5.3 Wiring of different components

CHAPTER 1: INTRODUCTION

1.1 Background

The combination of digital and embedded technologies is causing a huge revolution in the agriculture industry. Conventional farming methods are no longer adequate in light of mounting worries about climate change, resource depletion, and the need to feed a growing population. The goal of smart agriculture, a subfield of precision farming, is to overcome these obstacles by combining mobile computers, wireless sensor networks, and the Internet of Things (IoT).

In this regard, farm monitoring systems have become effective instruments for raising agricultural output. These systems use sensors placed in fields to gather vital information about the environment and soil, which is subsequently processed and sent in real time to farmers. Making accurate choices is made possible by having access to this data, which may be used to diagnose crop water stress, schedule irrigation, and even forecast pest assaults depending on environmental factors.

The energy needs of such devices is one of their primary drawbacks. Numerous smart farming systems rely on cellular data or Wi-Fi, both of which use a lot of electricity. Moreover, rural and hilly areas could not have access to these networks. Implementing power-efficient hardware platforms and communication protocols is necessary to make smart monitoring systems genuinely scalable and sustainable.

1.2 Problem Statement

Farmers mainly rely on visual inspection of soil conditions and traditional irrigation schedules, especially in developing regions where technological adoption is still limited. This manual method is often imprecise—if the timing or volume of irrigation is not ideal, it can lead to overwatering or underwatering, which not only wastes water but also negatively impacts crop yield and soil health. As water scarcity becomes a growing concern globally, inefficient irrigation practices further strain limited resources.

On large-scale farms, the challenge becomes even more critical. Parameters such as soil temperature, humidity, and moisture content can vary widely across different sections of the land. In such cases, manual monitoring becomes labor-intensive, time-consuming, and error-prone, making it an impractical solution for precision agriculture.

Moreover, conventional farm monitoring systems often fail to perform reliably in real-world rural environments due to several key limitations:

- **Unreliable internet access**, which makes cloud-based solutions inaccessible in remote locations.
- **High power consumption** of Wi-Fi or cellular-based communication modules, which makes battery-powered systems unsustainable for long-term deployment.

- **Lack of low-cost, scalable alternatives** that can meet the diverse needs of different farm sizes and environments.

Hence, there is a clear need for a system that is:

- **Economical** and **simple to deploy**, requiring minimal technical expertise for setup and maintenance.
- **Energy-efficient**, capable of operating in **low-power modes** for extended periods using battery power.
- **Communication-flexible**, with the ability to switch between Wi-Fi (MQTT) and **LoRa-based transmission**, making it suitable for both connected and offline scenarios.
- **Scalable and modular**, allowing more sensors or features to be added as per the farmer's need and the size of the land.

Such a system not only addresses current limitations but also brings the benefits of **smart agriculture** to regions that are often overlooked in the technological transformation of farming.

1.3 Objectives

The main objective of this project is to design and implement a power-efficient Smart Farm Monitoring System that:

1. Monitors real-time soil parameters (temperature, moisture, humidity).
2. Transmits data wirelessly to a farmer's smartphone.
3. Compares two wireless communication protocols—MQTT (Wi-Fi) and LoRa—for performance and power efficiency.
4. Implements low-power technique to extend the battery life of the monitoring unit.

The primary objective of this project is to design and implement a **power-efficient Smart Farm Monitoring System** capable of continuously tracking crucial soil and environmental parameters. By leveraging affordable and easily deployable sensor technology, the system aims to monitor **real-time values of temperature, soil moisture, and humidity**, which are vital indicators for effective crop growth and irrigation planning.

A significant goal of the project is to ensure that the collected data is **wirelessly transmitted to the farmer's smartphone** in a timely and reliable manner. To achieve this, the project explores two different communication strategies—**MQTT over Wi-Fi** and **LoRa (Long Range Radio)**. Both methods are implemented and evaluated in parallel to determine their practical effectiveness, with a particular focus on real-world agricultural deployment scenarios. The MQTT-based system utilizes the existing Wi-Fi infrastructure to send data to a cloud server, where it is accessed through a mobile application interface like Blynk. On the other hand, the LoRa-based system employs long-range, low-power RF communication modules (SX1278) to transmit sensor data directly to a receiver node, which then forwards it to the cloud for visualization.

One of the core concerns addressed in this project is **power consumption**, especially for remote farm environments where consistent power supply is a challenge. Therefore, the system is designed to operate in a **low-power mode**, activating the microcontroller and sensors only during specific intervals to collect and transmit data. Real-time current and energy usage measurements are carried out for both communication setups to **compare their power efficiency**. The results of this comparison provide insight into which protocol is more suitable for battery-powered applications and long-term use in rural farming areas.

By integrating real-time monitoring, wireless communication, mobile accessibility, and power optimization, this project contributes toward building a **scalable, sustainable, and farmer-friendly solution** for precision agriculture.

1.4 Scope Of The Project

The scope of this project encompasses the design and implementation of a smart, real-time monitoring system for agricultural fields, aiming to assist farmers in making informed decisions based on live environmental data. The system is designed with a focus on reliability, energy efficiency, long-range communication, and ease of access through a mobile interface.

To achieve this, the project integrates essential environmental sensors such as the **DHT11** for measuring temperature and humidity, and a **capacitive soil moisture sensor** to monitor the water content in the soil. These sensors form the foundation of data collection, offering insight into climatic and soil conditions critical to effective irrigation planning and crop management.

For data acquisition and processing, microcontroller boards like the **NodeMCU** and **Wemos D1 Mini** are used. These ESP8266-based platforms are not only compact and affordable but also provide built-in Wi-Fi capabilities, which facilitate seamless data transmission over the internet. Additionally, for communication over longer distances, especially in areas where Wi-Fi coverage is limited, the **SX1278 LoRa modules** are integrated with the NodeMCU boards. This combination enables point-to-point long-range communication while keeping power consumption low.

The project explores and compares two different communication methods: **MQTT over Wi-Fi** and **LoRa-based transmission**. In the MQTT-based setup, sensor data is sent via Wi-Fi to an MQTT broker and visualized on the **Blynk mobile application**, providing the farmer with real-time updates. In contrast, the LoRa setup allows data to be transmitted directly from the sensor node to a distant receiver node using LoRa radio modules, after which the receiver uploads the data to the cloud for visualization.

An important aspect of this project is the **power consumption analysis** of both setups. Current measurements are taken during data transmission to estimate the energy usage of the Wi-Fi and LoRa configurations. This analysis helps determine which method is more suitable for long-term deployment in remote or power-constrained environments, ultimately guiding the choice of communication protocol for future smart farming solutions.

Finally, the system includes a **mobile-based user interface**, designed to be simple and informative for the end-user — the farmer. Through this app, farmers can monitor real-time sensor readings such as temperature, humidity, and soil moisture, enabling them to make timely and effective agricultural decisions.

- Sensor Integration: DHT11 and soil moisture sensors are used for environmental data collection.
- Microcontroller Platform: NodeMCU and Wemos D1 Mini and SX1278 for LoRa protocol are used for controlling sensors and communication.
- Protocols Compared:
    - LoRa communication using SX1278 transceivers for long-range, low-power data transmission.
    - MQTT (over Wi-Fi) using Blynk app for real-time updates.
- Power Analysis: Real-time current measurements were used to compare energy consumption of both systems.
- User Interface: A mobile application to display data in a user-friendly format.

1.5 Methodology Overview

The development of the Smart Farm Monitoring System was carried out using a structured, step-by-step methodology to ensure practical usability, accuracy, and energy efficiency. The key phases of the methodology are as follows:

1. Component selection and circuit design-The first step involved identifying appropriate hardware components to achieve accurate environmental sensing with minimal power consumption. Sensors such as the **DHT11** (for temperature and humidity) and a **capacitive soil moisture sensor** were selected for their reliability and low power requirements. Microcontrollers like **WeMOS D1 Mini** and **NodeMCU ESP8266** were chosen for their Wi-Fi capabilities and compatibility with both **MQTT and LoRa protocols**.

2. Programming and data acquisition- The sensors and microcontrollers were programmed using the **Arduino IDE**. Libraries such as `DHT.h`, `LoRa.h`, and `BlynkSimpleEsp8266.h` were used to interface with sensors and implement wireless communication. The microcontrollers were programmed to acquire data at fixed intervals and transmit them either via Wi-Fi (MQTT protocol) or LoRa, depending on the setup. Proper calibration of sensors was done to ensure accurate readings in real-time conditions.

3. Protocol implementation (MQTT and LoRa)- Two parallel communication setups were implemented and tested

4. Testing and validation in real-world conditions

5. Power usage monitoring and comparison- A core objective of the project was to analyze power consumption across both communication methods. Real-time current measurements were taken using a **multimeter** and **INA219 sensor module** to record the average current draw during data acquisition, transmission, and idle states. These values were then used to estimate battery life under both **MQTT and LoRa** setups.

1.6 Motivation

The motivation for this project stems from the urgent need to bring technological solutions to agriculture, particularly in areas with poor infrastructure. By enabling real-time data-driven decisions through affordable and sustainable technology, this project aims to:

- Empower farmers with practical tools.
- Reduce resource wastage.
- Improve crop yield and quality.

1.7 Contributions

- Designed and implemented a dual-mode wireless system for farm monitoring.
- Developed a mobile dashboard for farmers using the Blynk app.
- Conducted an experimental power consumption analysis.
- Proposed LoRa as a scalable alternative for IoT-based smart agriculture in rural India.

1.8 System Overview

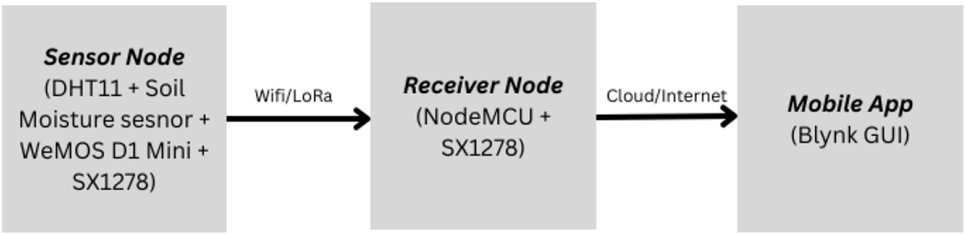

Figure 1.1 System Overview

Figure 1.1 illustrates the basic architecture of the Smart Farm Monitoring System, which is divided into three main components: the Sensor Node, the Receiver Node, and the Mobile Application.

The Sensor Node integrates a DHT11 temperature and humidity sensor, a capacitive soil moisture sensor, and a WeMOS D1 Mini microcontroller. This node collects environmental data and transmits it either via Wi-Fi using MQTT or using the SX1278 LoRa module for long-range communication.

The Receiver Node, built using a NodeMCU ESP8266 with another SX1278 module, receives the sensor data in the LoRa setup and forwards it to the cloud. In the Wi-Fi setup, the sensor node sends data directly to the cloud without needing a receiver.

The final component is the Mobile App, developed using the Blynk platform, which retrieves the real-time data from the cloud and displays it in a simple graphical interface for the farmer.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

The growing adoption of smart farming practices has triggered extensive research into wireless sensor networks (WSNs), low-power embedded systems, and IoT-driven agricultural monitoring. With increasing concerns over water scarcity, food security, and climate change, precision agriculture has emerged as a promising solution that relies heavily on real-time environmental data and automated control systems.

This chapter presents a review of relevant literature in the domains of sensor-based agricultural monitoring, wireless communication protocols like MQTT and LoRa, and energy-efficient IoT architectures. The goal is to provide a foundation for understanding the design decisions in this project and identify research gaps that the current system aims to address.

2.2 Smart Agriculture and Sensor-Based Monitoring

Several studies have demonstrated the benefits of integrating environmental sensors with microcontrollers to monitor critical parameters such as temperature, soil moisture, and humidity. Sharma et al. [1] implemented a microcontroller-based irrigation system using DHT11 and moisture sensors, but lacked remote access or energy optimization. Singh and Patel [2] highlighted the need for cloud-integrated platforms to support decision-making.

A real-time IoT-based crop monitoring system by Jawad et al. [3] used environmental sensors with Zigbee, but it suffered from limited range and power constraints. More recent efforts, such as that by Preeth et al. [4], incorporated machine learning for data analysis but were still dependent on high-power GSM modules.

Despite growing hardware availability, the complexity of managing power and ensuring communication in rural, off-grid environments remains a barrier to deployment.

2.3 Communication Technologies in IoT-Based Agriculture

MQTT in Agricultural IoT Systems

MQTT is a lightweight publish/subscribe protocol ideal for low-bandwidth, real-time telemetry over IP. Onaolapo et al. [5] concluded that MQTT outperforms HTTP in delay and bandwidth efficiency in agricultural contexts. Similarly, Ahmed et al. [6] designed a greenhouse monitoring system using MQTT and ThingSpeak, achieving sub-second latency in stable Wi-Fi environments.

However, MQTT's dependency on consistent internet limits its applicability in remote farms. As Zhang et al. [7] note, the protocol's energy cost increases drastically during reconnection or unstable networks, making it less suitable for low-power, battery-based devices in rural areas.

LoRa and LPWAN Protocols

LoRa has emerged as a highly efficient communication alternative in rural and wide-area deployments. With a range of up to 15 km and current draw as low as 10 μA in sleep mode [8], LoRa provides a scalable, low-power solution for sensor nodes.

Augustin et al. [9] compared LoRa with Zigbee and Wi-Fi, concluding LoRa had superior range and power performance. Zhao et al. [10] implemented a LoRa-based irrigation controller that sustained battery operation for over two weeks in a field setting. Rani et al. [11] proposed a multi-node LoRa mesh network that provided seamless coverage over fragmented terrains in India.

2.4 Power Efficiency in IoT Monitoring Systems

Power efficiency is one of the most critical factors in the design and deployment of IoT-based agricultural systems, particularly for rural and off-grid environments where access to continuous power supply is limited or unavailable. Effective power management directly influences the system's **operational lifetime, reliability, and feasibility for field deployment**.

Challenges in Power Consumption

IoT systems in agriculture must often operate in off-grid locations with limited energy availability. ESP8266-based devices, while popular, consume up to 250 mA during Wi-Fi transmissions [12]. Without optimization, frequent transmissions and continuous sensor polling can lead to rapid battery drain.

Low-Power Communication Protocols

LoRa excels in minimizing current draw. Rizzi et al. [13] demonstrated that optimized spreading factors and reduced transmission payloads could achieve 70% energy savings over GSM/Wi-Fi-based solutions. Further enhancements were noted by Gaddam et al. [14], who employed duty-cycled LoRa transmissions with a latency-tolerant agricultural sensor network.

Power-Saving Techniques

Various power-saving strategies have also been implemented at the firmware level:

- **Deep Sleep Mode**: ESP8266 and ESP32 support deep sleep, reducing current to under 0.5 mA [15]. This mode was used in our project to wake the node every 10 minutes.

- **Duty Cycling**: Instead of continuous monitoring, data is collected and transmitted at defined intervals (e.g., every 5 or 10 minutes), which greatly reduces the average power consumption.

- **Sensor Optimization**: Sensors like DHT11 are polled only when necessary, avoiding continuous current draw.

Real-Time Power Monitoring

Tools such as INA219 and digital ammeters allow for accurate profiling of system energy usage. Mandal et al. [16] showed that combining software and hardware monitoring can extend system lifespan by up to 3x in solar-powered rural nodes.

2.5 Gaps Identified in Existing Systems

While a significant body of research has explored IoT-based smart agriculture systems, many existing implementations fall short when applied to **real-world rural settings**. These limitations often stem from assumptions about infrastructure availability, power accessibility, and communication reliability. Below are some of the key gaps identified in the literature and how they relate to the practical challenges that farmers face.

- Lack of flexibility to adapt to both connected and offline environments.
  Many existing systems rely entirely on **cloud-based services and Wi-Fi connectivity**, which are not always feasible in rural or remote agricultural areas. Farmers operating in these regions often have limited or no access to reliable internet, which restricts the usability of such systems. While MQTT is efficient in online scenarios, systems that lack **offline operability or local data storage** become non-functional in network-dead zones. This severely undermines their reliability and scalability for deployment in low-infrastructure regions.

- High dependency on constant internet access.
  Numerous systems reviewed assume **24/7 internet availability** for data transmission, storage, and mobile app synchronization. This assumption is unrealistic in many developing countries where internet connectivity is intermittent, costly, or geographically restricted. As a result, systems built solely on internet-dependent protocols like MQTT may lead to **data loss**, **inconsistent monitoring**, and **inaccessible dashboards** when connectivity fails.

- Insufficient attention to power profiling and battery optimization.
  Power consumption is one of the most crucial factors in designing smart monitoring systems for agricultural use. However, many projects either overlook this aspect or

implement it superficially. Systems relying on Wi-Fi or GSM modules typically have **high idle and transmission current**, leading to frequent battery replacements or the need for external power sources. Without the use of **low-power techniques** such as deep sleep modes, duty cycling, or protocol-level optimizations, these solutions become unsustainable for long-term use in off-grid farms.

- Systems that are not scalable or modular, making them difficult to upgrade or tailor to different farm sizes.

This project addresses these gaps by offering a **dual-mode system**—supporting both **Wi-Fi (MQTT)** and **LoRa**, providing real-time mobile access, and performing a **quantitative power analysis** to determine optimal deployment strategies for long-term rural use.

2.6 Conclusion

The review of existing literature reveals a robust and evolving landscape of IoT applications in agriculture, aimed at improving farm productivity, resource optimization, and data-driven decision-making. Researchers have successfully demonstrated the integration of environmental sensors with microcontrollers to automate and enhance traditional farming practices. Technologies such as **DHT sensors**, **soil moisture probes**, and platforms like **NodeMCU** have proven to be effective tools for real-time monitoring.

In terms of communication protocols, **MQTT** and **LoRa** have emerged as dominant choices for transmitting sensor data. MQTT is lightweight and well-suited for cloud-based environments with stable internet connectivity, while LoRa provides low-power, long-range communication—ideal for rural and infrastructure-deficient regions. Numerous studies have validated the practical advantages of both, though each comes with trade-offs in terms of range, power consumption, and dependence on external networks.

Power efficiency remains one of the most pressing concerns, particularly for remote or off-grid deployments. Several research works have emphasized the importance of using **sleep modes**, **optimized duty cycles**, and **energy-aware hardware** to extend operational life. However, many existing systems still fall short of being practically viable due to a lack of real-time power profiling and optimization.

Furthermore, despite technological advancements, a noticeable gap exists in the **adaptability, scalability, and end-user accessibility** of many smart farm monitoring systems. Most systems are either too dependent on internet connectivity or lack intuitive interfaces that are user-friendly for farmers. Few projects offer modular solutions that can grow with the needs of a farm or accommodate changes in environmental monitoring requirements.

The Smart Farm Monitoring System proposed in this project aims to address these shortcomings. By combining real-time environmental sensing with dual-mode communication (MQTT over Wi-Fi and LoRa), it enables flexible deployment in both connected and remote

environments. The system also emphasizes power-conscious design, validated through empirical current measurements, and provides an intuitive mobile application interface for the farmer.

In conclusion, while the existing literature provides valuable insights and foundational technologies, this project offers a more **comprehensive, energy-efficient, and user-oriented solution** to smart farming, tailored to the real-world constraints of rural agriculture—especially in developing regions like India.

CHAPTER 3: WIRELESS COMMUNICATION TECHNOLOGIES— MQTT OVER Wi-Fi AND LoRa

3.1 Introduction

The core of any IoT-based monitoring system lies in its ability to transmit sensor data wirelessly and efficiently to a receiver or dashboard for visualization and decision-making. In this project, we explored two prominent wireless communication techniques:

- MQTT over Wi-Fi
- LoRa (Long Range Radio)

Both methods have their own strengths and trade-offs, especially in terms of **power consumption**, **range**, **bandwidth**, and **data reliability**. This chapter provides a detailed overview of both technologies and justifies their selection for use in farm monitoring systems.

3.2 MQTT (Message Queuing Telemetry Transport) over Wi-Fi

3..2.1 What is MQTT?

MQTT (Message Queuing Telemetry Transport) is a lightweight, publish-subscribe-based messaging protocol designed for low-bandwidth, high-latency, or unreliable networks. It is ideal for IoT applications due to its simplicity and low overhead.

It works on top of the TCP/IP protocol, making it compatible with most modern microcontrollers equipped with Wi-Fi capabilities such as **ESP8266/NodeMCU**.

3.2.2 MQTT Architecture

MQTT follows a client-broker model:

- **Publisher**: Sends messages to a topic (e.g., sensor values).
- **Subscriber**: Receives messages by subscribing to specific topics.
- **Broker**: Acts as an intermediary to manage all communications (e.g., Eclipse Mosquitto, Blynk server).

> Example: The NodeMCU acts as the publisher, sending sensor data to the Blynk cloud server, which acts as the broker. The Blynk mobile app then subscribes to this data and displays it.

3.2.3 Properties of MQTT over Wi-Fi

The table below outlines the key characteristics of using the MQTT protocol over Wi-Fi networks. It highlights critical aspects such as communication protocol, range, bandwidth,

power consumption, quality of service levels, and latency. Understanding these properties is essential for evaluating MQTT's suitability in real-time and IoT applications where Wi-Fi is the primary communication medium.

Table 3.1 MQTT Over Wifi Characteristics

Property	Description
Protocol	TCP/IP
Communication Range	30–100 meters (depends on Wi-Fi signal strength)
Bandwidth	High (suitable for real-time streaming)
Power Consumption	High (due to constant Wi-Fi connectivity)
Message QoS Levels	0 (At most once), 1 (At least once), 2 (Exactly once)
Latency	Low (excellent for real-time applications with stable internet)

3.2.4 Advantages

- Easy to implement using platforms like Blynk or Adafruit IO.
- Near real-time data delivery.
- Widely supported with cloud integration.

3.2.5 Limitations

- High power usage due to Wi-Fi being power-hungry.
- Requires constant internet availability (router or hotspot).
- Limited range unsuitable for large agricultural fields.

3.3 LoRa (Long Range Radio)

LoRa is a proprietary, low-power, wide-area network (LPWAN) physical layer protocol developed by Semtech. It uses **chirp spread spectrum modulation**, allowing data to be transmitted over long distances (up to 10–15 km) with minimal power consumption].

Unlike MQTT over Wi-Fi, LoRa works in **unlicensed ISM bands (typically 433 MHz or 868/915 MHz)** and does not require internet or mobile networks.

3.3.1 LoRa Architecture (P2P)

This project uses a **peer-to-peer (P2P)** configuration with two LoRa modules:

- One NodeMCU with SX1278 acts as the transmitter (sensor node).
- Another NodeMCU with SX1278 acts as the receiver (base station).

No central server is needed—just raw radio communication between devices.

Note: For broader scalability, LoRaWAN can be used with gateways, but our implementation focuses on P2P LoRa for simplicity and energy efficiency.

3.3.2 Properties of LoRa

The table below summarizes the core properties of the LoRa (Long Range) communication protocol, which is widely used in low-power, wide-area networks (LPWANs). LoRa uses Chirp Spread Spectrum (CSS) modulation to achieve long-range communication with minimal power consumption, making it ideal for IoT applications such as remote sensing and smart agriculture. Key parameters such as range, bandwidth, power usage, data rate, and latency are detailed to provide a comprehensive understanding of its capabilities.

Table 3.2 LoRa Performance Metrics

Property	Description
Protocol	Chirp Spread Spectrum (CSS)
Communication Range	2–15 km (line of sight)
Bandwidth	Low (suitable for small packets, e.g., sensor readings)
Power Consumption	Very low

Data Rate	0.3–50 kbps
Latency	Moderate (due to slower data rate)

3.3.3 Advantages

- Ultra-low power consumption, ideal for battery-powered applications [6].
- Works in remote, rural areas with no Wi-Fi or cellular connectivity.
- Long-range communication suitable for large farms or multiple fields.

3.3.4 Limitations

- Limited bandwidth; not suitable for transmitting large datasets or media.
- Requires external antennas and careful configuration (e.g., spreading factor, frequency).
- Slightly more complex initial setup compared to MQTT-based systems .

3.4 Comparison Table

The table below compares MQTT (operating over Wi-Fi) and LoRa in point-to-point (P2P) mode across various technical and practical features. These include communication range, power consumption, data rates, need for internet connectivity, real-time performance, scalability, implementation complexity, and ideal use cases. This comparison helps evaluate the strengths and limitations of each protocol, aiding in the selection of the most suitable option based on specific IoT deployment scenarios.

Table 3.3 MQTT vs LoRa Feature Comparison

Feature	**MQTT (Wi-Fi)**	**LoRa (P2P)**
Range	30–100 meters	2–15 kilometers
Power Consumption	High	Very Low
Data Rate	Up to 1 Mbps	Up to 50 kbps

Internet Required	Yes	No
Real-Time Capability	High	Moderate
Scalability	High (with cloud brokers)	Moderate (without LoRaWAN)
Complexity	Low (plug-and-play platforms)	Medium (manual configuration)
Best Use Case	Smart homes, small farms	Large rural farms, remote areas

3.5 Justification of Use

By integrating both MQTT and LoRa, this project explores two practical extremes of IoT communication:

- MQTT is useful for farms with **Wi-Fi availability** and **access to internet-based dashboards**.
- LoRa is better suited for **low-power deployment in isolated areas** where no internet or infrastructure is present.

The power analysis performed in later chapters further demonstrates that LoRa can operate significantly longer on battery power, making it a more sustainable solution in certain real-world use cases].

CHAPTER 4: COMPONENTS

4.1 Introduction

This chapter outlines the various components used in the implementation of the smart farm monitoring system. The system includes microcontrollers, sensors, communication modules, and power sources that enable data collection and transmission. The system was developed using two different communication protocols—MQTT over Wi-Fi and LoRa. Both systems have been evaluated for performance and power consumption, and this chapter provides a comprehensive overview of the hardware involved.

4.2 Hardware Components

4.2.1 NodeMCU ESP8266

The NodeMCU is a low-cost microcontroller based on the ESP8266 Wi-Fi module. It features GPIO pins for sensor integration and has built-in Wi-Fi, making it ideal for IoT applications [18]

- Function: Acts as the main controller in both MQTT and LoRa-based implementations.
- Key Features:
    - Built-in Wi-Fi capability
    - USB interface for programming
    - Operates at 3.3V logic
    - Compatible with Arduino IDE [1]

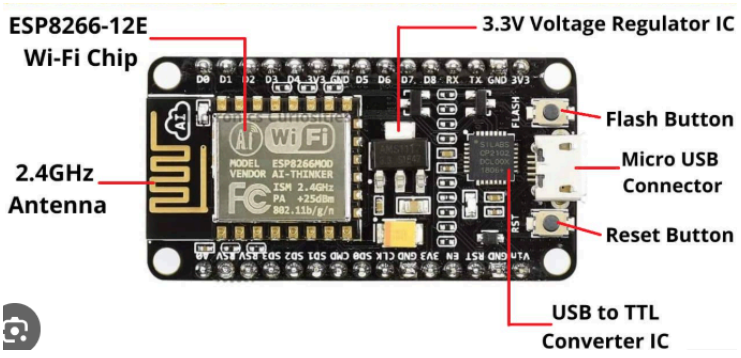

Figure 4.1 NodeMCU ESP8266

In this project, the NodeMCU acted as both a transmitter (in the LoRa setup) and a cloud-connected device (in the MQTT setup).

Table 4.1 Technical Specifications of NodeMCU

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

4.2.2 WeMOS D1 Mini

The Wemos D1 Mini is another ESP8266-based board, compact in size and excellent for embedding in small-scale projects [19]. Despite its size, it retains almost all capabilities of the standard NodeMCU.
It is important because of the following characteristics

- Compatible with shields and plug-and-play hardware modules.
- Lower current draw during idle or sleep conditions, aiding in power-sensitive designs.
- Acts as the core controller in the MQTT implementation.

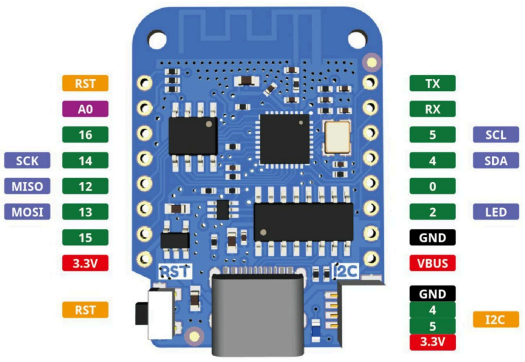

Figure 4.2 WeMOS D1 Mini

Table 3.2 WeMOS D1 Mini Technical Specifications

Operating Voltage	3.3V
Digital I/O Pins	11
Analog Input Pins	1(3.2V Max)
Clock Speed	80/160MHz
Flash	4M Bytes
Size	34.2*25.6mm
Weight	3g

4.2.3 DHT11 Sensor

The DHT11 is a capacitive humidity sensor and a thermistor for measuring environmental conditions. It uses a single-wire digital interface, simplifying its connection to microcontrollers [3].

- Utilizes negative temperature coefficient (NTC) thermistor for temperature sensing.
- Contains an 8-bit microcontroller internally for signal processing.
- Calibration data stored in ROM enables plug-and-play usage.

Although DHT11 is not highly precise (±2°C, ±5% RH), it is cost-effective and sufficient for most agricultural applications.

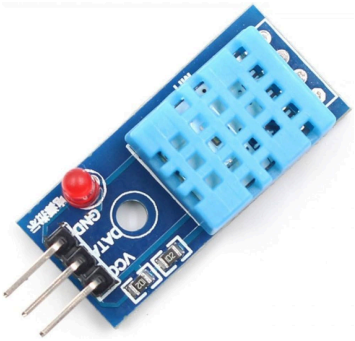

Figure 4.3 DHT11 Sensor

Table 4.3 DHT11 humidity sensor specifications

Parameter	DHT11
Relative Humidity Range	20 ~ 90%
Relative Humidity Accuracy:	±5%RH
Temperature Accuracy:	±2°C
Resolution	1%RH,1°C
Repeatability	±1%
Long Term Stability	±1% per year
Operating Temperature Range	0 ~ 50°C
Power Supply:	3,3V ~ 5.5V
Supply Current:	0.5mA ~ 2.5mA
Idle Supply Current:	100uA ~ 150uA
Max sampling period (Max device update rate).	1 second

4.2.4 Soil Moisture Capacitive Sensor

Working Principle - A capacitive soil moisture sensor measures soil moisture by detecting changes in capacitance, which is affected by the soil's dielectric constant. When water content changes, the dielectric constant [20] of the soil changes, altering the sensor's capacitance. This capacitance variation is then translated into an analog or digital signal proportional to the soil moisture level.
This sensor estimates moisture content by measuring conductivity/resistance between two probes inserted into the soil. Water increases soil conductivity, reducing resistance.

- Works on the principle that wetter soil conducts electricity better than dry soil.
- Outputs analog voltage corresponding to moisture level.
- Effective for irrigation automation and water management systems.

Some versions also come with corrosion-resistant coatings (e.g., nickel) to extend lifespan in wet environments.

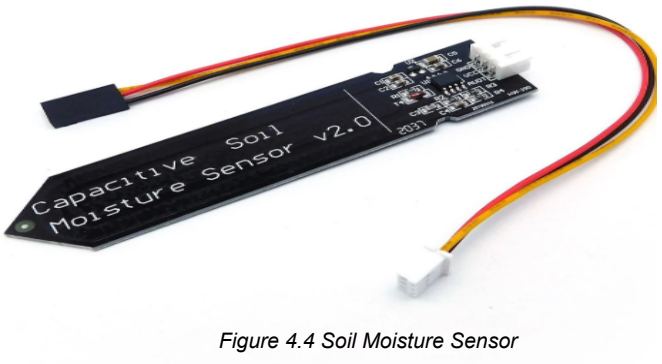

Figure 4.4 Soil Moisture Sensor

4.2.5 SX1278 LoRa Module

The SX1278 is a long-range, low-power radio frequency module that uses LoRa (Long Range) modulation. It supports point-to-point and mesh networking modes [21].

- Utilizes Chirp Spread Spectrum (CSS) modulation for high sensitivity and long range.
- Range: Up to 2 km in open areas, with minimal power consumption.
- Operates in 433 MHz ISM band (unlicensed).

This module enables reliable transmission over long distances, overcoming the limitations of Wi-Fi-based communication in rural areas lacking internet infrastructure.

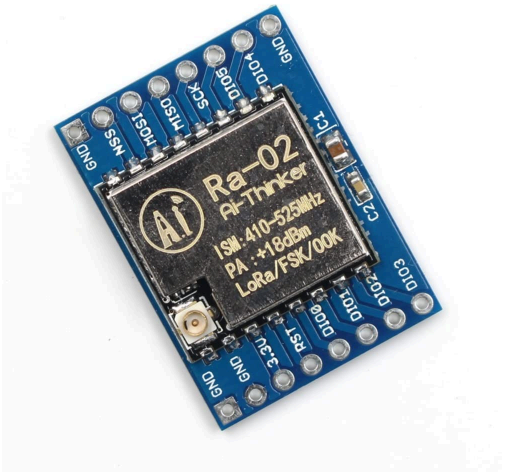

Figure 4.5 SX1278

4.2.6 Breadboard and Jumper Wires

Jumper wires and breadboards are essential during prototyping. They allow flexible and non-permanent connections between microcontrollers, sensors, and modules [22].

- Enable rapid testing and debugging of circuit configurations.
- Allow modular assembly and replacement of components.

4.2.7 Power Source

The system was powered using USB cables connected to laptops or power banks. For outdoor deployment, a rechargeable battery or solar-powered system can be used.

- Stability of voltage and capacity influences the lifetime of deployment.
- Integration of low-power sleep modes in microcontrollers is vital to extend battery life [23].

4.3 Software Components

4.3.1 Arduino IDE

The Arduino IDE is an open-source development environment used for coding and uploading programs to Arduino-compatible boards like NodeMCU and Wemos [24].

- Libraries Used:
    - `ESP8266WiFi.h` for Wi-Fi functionality
    - `PubSubClient.h` for MQTT communication
    - `DHT.h` for reading sensor data
    - `LoRa.h` for interfacing with SX1278

The use of this IDE simplifies integration with cloud services and sensor data handling.

4.3.2 Blynk IoT Platform

Blynk is a cloud-based platform that allows IoT developers to build apps for controlling and monitoring hardware remotely [9].

- Acts as a lightweight MQTT broker.
- Enables creation of virtual dashboards for data visualization.
- Supports push notifications and triggers.

Blynk was utilized in this project for the Wi-Fi-based version to display real-time temperature, humidity, and moisture values on the farmer's smartphone.

4.4 Summary

The selection of components in this project was guided by considerations of cost, ease of use, availability, power efficiency, and communication range. The dual-approach system architecture (Wi-Fi + MQTT and LoRa) offers flexibility and adaptability depending on network availability and energy constraints. This component-level understanding forms the backbone of the implementation and evaluation described in the following chapters.

CHAPTER 5: CONNECTIONS AND SYSTEM OVERVIEW

5.1 Introduction

This chapter focuses on the complete wiring and logical connections made between various components of the smart farm monitoring system. The goal is to present a clear and comprehensive overview of how the hardware components are integrated in both the Wi-Fi-based (MQTT) and LoRa-based systems. Each setup is explained through connection tables, circuit diagrams, and system architecture overviews to illustrate the data flow from sensor node to the user interface.

5.2 System Architecture Overview for WiFi Setup

5.2.1 MQTT over Wi-Fi Setup

This configuration uses the Wemos D1 Mini connected to sensors. It sends real-time environmental data over Wi-Fi using the MQTT protocol. The data is sent to the Blynk cloud platform and viewed via the Blynk app on a smartphone.

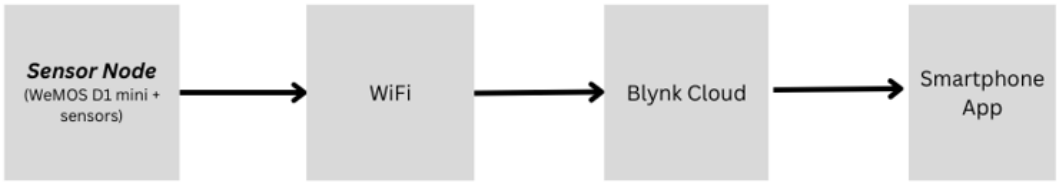

Figure 5.1 Architecture Flow Diagram

Key Characteristics:

- Requires internet availability.
- Simple to deploy where Wi-Fi is available.
- Higher power consumption due to constant Wi-Fi usage.

5.2.2 Hardware Connection for Wi-Fi setup

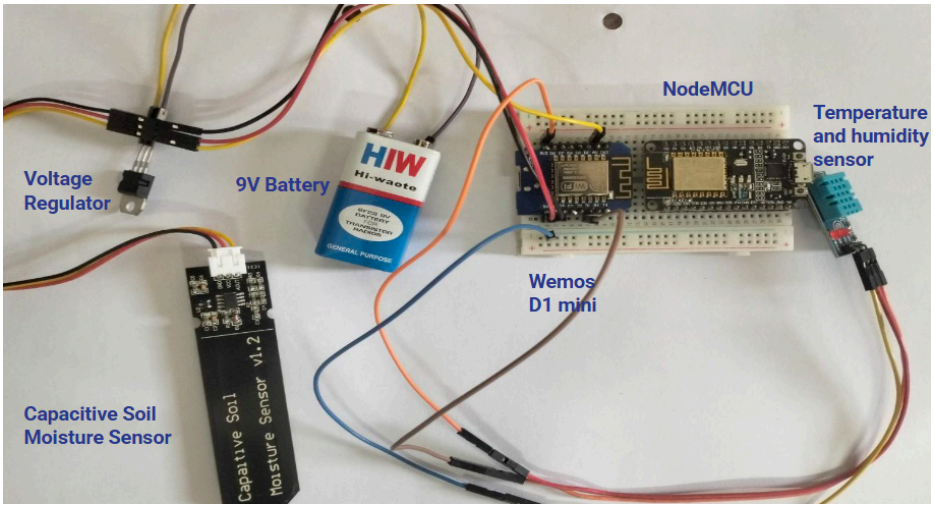

Figure 5.2 Connection of various hardware components

Table 5.1 Pin Connections

Component	***Pin Name***	***Connected To***	***Board***	***Purpose***
DHT11 Temperature & Humidity Sensor	*VCC*	*3.3V*	*NodeMCU/Wemos D1*	*Power supply*
	GND	*GND*	*NodeMCU/Wemos D1*	*Ground connection*
	Data	*D4 (GPIO2)*	*NodeMCU/Wemos D1*	*Digital data signal*
Capacitive Soil Moisture Sensor	*VCC*	*5V (from Voltage Regulator)*	—	*Power supply*
	GND	*GND*	—	*Ground connection*

	Analog Out	*A0*	*Wemos D1 Mini*	*Analog signal (soil moisture level)*
Voltage Regulator (e.g., LM7805)	*IN*	*9V Battery Positive*	—	*Input power from battery*
	OUT	*5V to Sensor VCC*	—	*Regulated 5V output*
	GND	*Common Ground (Battery + Sensors)*	—	*Ground for circuit*
9V Battery	*Positive Terminal*	*IN (Voltage Regulator)*	—	*Main power source*
	Negative Terminal	*GND (Voltage Regulator & System)*	—	*Ground line for entire setup*

- ➔ **Wemos D1 Mini** handles sensor data collection and transmits it via MQTT over Wi-Fi.
- ➔ **DHT11** is connected to **D4** (GPIO2) for temperature & humidity sensing.
- ➔ **Soil Moisture Sensor** sends analog data to **A0**.
- ➔ A **9V battery** powers the circuit, stepped down to 5V using a **voltage regulator** for safe sensor operation.
- ➔ **Sensor Node (Wemos D1 Mini)** reads data from sensors and **publishes** it to an MQTT broker over Wi-Fi and **Receiver Node (NodeMCU) subscribes** to the MQTT topic, receives the data, and **forwards it to the cloud** (e.g., Blynk or another server).

5.2.3 Arduino Code

Table 5.2 Libraries Used

Library	Purpose
`ESP8266WiFi.h`	Enables Wi-Fi connectivity for ESP8266-based boards

`PubSubClient.h`	Handles MQTT publish/subscribe operations
`DHT.h`	Used to read data from the DHT11 sensor
`BlynkSimpleEsp8266.h`	Enables communication with the Blynk Cloud App

```ino
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "Leftover-laptop";
const char* password = "YOUR_WIFI_PASSWORD";
const char* mqttServer = "broker.hivemq.com";
const int mqttPort = 1883;
WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
    Serial.begin(115200);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.println("Connecting to WiFi...");
    }
    Serial.println("Connected to WiFi");
    client.setServer(mqttServer, mqttPort);
    while (!client.connected()) {
        Serial.println("Connecting to MQTT...");
        if (client.connect("ESP8266_Sender")) {
            Serial.println("Connected to MQTT");
        } else {
            Serial.print("Failed, rc=");
            Serial.println(client.state());
            delay(5000);
        }
    }
}

void loop() {
    float sensorValue = analogRead(A0) * (3.3 / 1023.0);
    String payload = String(sensorValue);
    client.publish("sensor/data", payload.c_str());
    Serial.println("Sent: " + payload);
    delay(1000);
}
```

Figure 5.3 Sender Code

sketch_may1a.ino

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "Leftover-laptop";
const char* password = "leftover";
const char* mqttServer = "broker.hivemq.com";
const int mqttPort = 1883;

WiFiClient espClient;
PubSubClient client(espClient);

void callback(char* topic, byte* payload, unsigned int length) {
    Serial.print("Received: ");
    for (int i = 0; i < length; i++) {
        Serial.print((char)payload[i]);
    }
    Serial.println();
}

void setup() {
    Serial.begin(115200);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.println("Connecting to WiFi...");
    }
    Serial.println("Connected to WiFi");
    client.setServer(mqttServer, mqttPort);
    client.setCallback(callback);
    while (!client.connected()) {
        Serial.println("Connecting to MQTT...");
        if (client.connect("ESP8266_Receiver")) {
            Serial.println("Connected to MQTT");
            client.subscribe("sensor/data");
        } else {
            Serial.print("Failed, rc=");
            Serial.println(client.state());
            delay(5000);
        }
    }
}

void loop() {
    client.loop();
}
```

Figure 5.4 Receiver Code

5.3 System architecture overview for LoRa Setup

5.3.1 LoRa communication setup

In this setup, the NodeMCU connects with sensors and a LoRa module (SX1278) to send data wirelessly to another LoRa-equipped NodeMCU (receiver).

Key Characteristics:

- Long-range communication (up to several kilometers).
- Minimal power usage.

Figure 5.5 Data Flow Diagram

5.3.2 Hardware Connections for LoRa setup

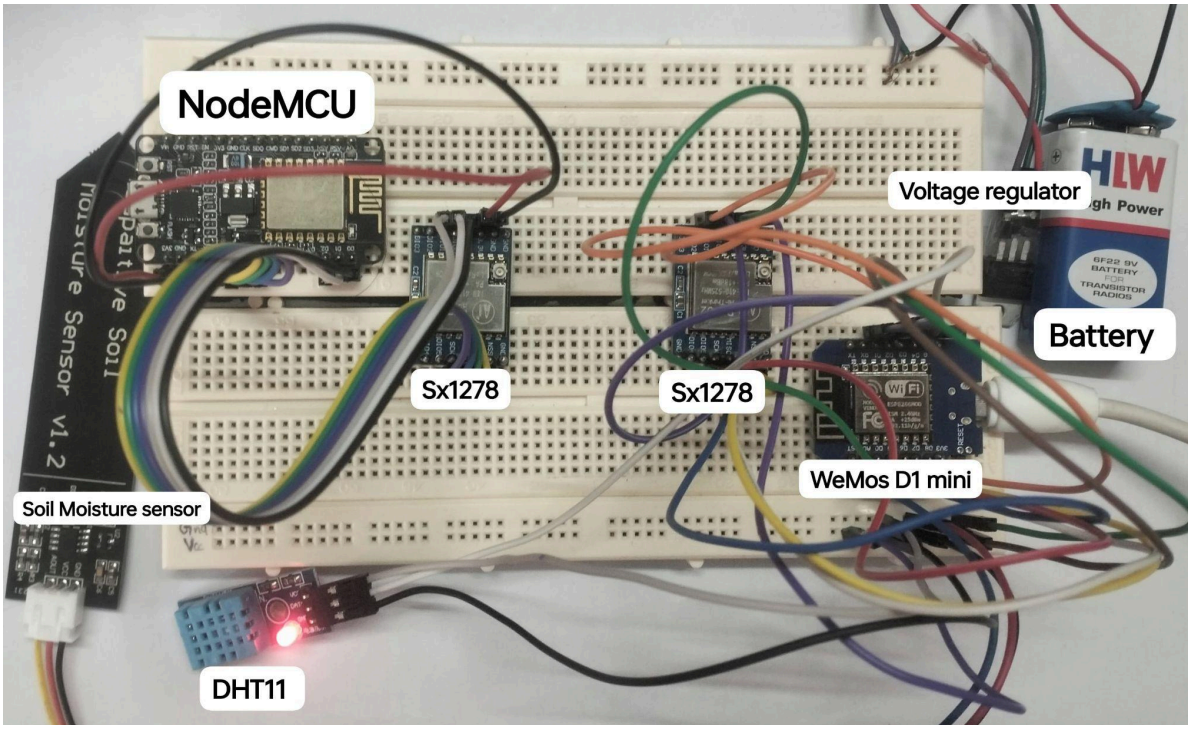

Figure 5.6 Connection of various hardware components

➔ The **sensor node** consists of a **NodeMCU** connected to environmental sensors like the **DHT11 (temperature and humidity)** and the **capacitive soil moisture sensor**. These sensors collect real-time agricultural data from the farm.

➔ The sensor data is transmitted using the **LoRa SX1278 module**, which is connected to the NodeMCU. LoRa enables **long-range, low-power communication**, making it

suitable for rural or remote field locations without stable Wi-Fi or cellular connectivity.

➔ On the other end, a **second NodeMCU** is connected to another **SX1278 LoRa module**, acting as the **receiver node**. This node receives the sensor data transmitted from the field.

➔ The receiver NodeMCU is connected to **Wi-Fi**, and it then **forwards the received data to a cloud platform called Blynk**.

5.3.3 Pin Connections

Table 5.3 Wiring of different components

Component	***Pin on NodeMCU***	***Pin on SX1278 LoRa Module***
MISO	*D6 (GPIO12)*	*MISO*
MOSI	*D7 (GPIO13)*	*MOSI*
SCK	*D5 (GPIO14)*	*SCK*
NSS (CS)	*D8 (GPIO15)*	*NSS (Chip Select)*
RESET	*D0 (GPIO16)*	*RESET*
DIO0 (Interrupt Pin)	*D1 (GPIO5)*	*DIO0*
VCC	*3.3V*	*VCC*

5.3.4 Arduino Code for LoRa setup

```ino
#include <SPI.h>
#include <LoRa.h>
#include <DHT.h>

#define DHTPIN D3
#define DHTTYPE DHT11
#define SOIL_PIN A0

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();

  // LoRa initialization
  LoRa.setPins(D8, D0, D1);  // NSS, no RESET, DIO0


  if (!LoRa.begin(433E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  Serial.println("LoRa Transmitter started");
}

void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  int soil = analogRead(SOIL_PIN);

  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT");
    return;
  }

  String payload = "Temp:" + String(t) + ",Hum:" + String(h) + ",Soil:" + String(soil);
  Serial.println("Sending: " + payload);

  LoRa.beginPacket();
  LoRa.print(payload);
  LoRa.endPacket();

  delay(3000); // send every 3 seconds
}
```

Figure 5.7 Arduino Code for transmitter side

```ino
#include <SPI.h>
#include <LoRa.h>

unsigned long lastCheck = 0;
const unsigned long checkInterval = 1000; // 1 second

void setup() {
  Serial.begin(9600);
  LoRa.setPins(D8, D0, D1); // NSS, RESET, DIO0

  if (!LoRa.begin(433E6)) {
    Serial.println("Starting LoRa failed!");
    while (1);
  }
  Serial.println("LoRa Receiver started");
}

void loop() {
  bool messageReceived = false;

  int packetSize = LoRa.parsePacket();
  if (packetSize) {
    messageReceived = true;
    String received = "";
    while (LoRa.available()) {
      received += (char)LoRa.read();
    }
    Serial.println("Received: " + received);
  }

  // Check every 1 second if no message
  if (millis() - lastCheck >= checkInterval) {
    if (!messageReceived) {
      Serial.println("No message received");
    }
    lastCheck = millis(); // reset timer
  }
}
```

Figure 5.8 Arduino Code for receiver side

CHAPTER 6: POWER MANAGEMENT

6.1 Overview

Power efficiency is a critical factor in the design of smart agricultural systems, especially in rural and remote areas where access to electricity is limited. A key goal of this project was to evaluate and compare the power consumption characteristics of two different wireless communication methods—MQTT over Wi-Fi and LoRa—in a smart farm monitoring environment.

This chapter presents a detailed analysis of power consumption using both experimental measurements and references from scholarly research. It highlights how each method performs in typical sensing-transmission cycles and in long-term operation under battery constraints.[26],[27]

6.2 Power Consumption Optimization

Given that sensor nodes are intended to operate on **battery power**, minimizing energy usage was a primary design goal. The following methods were implemented to optimize power usage:

6.2.1 Deep Sleep Mode

The **ESP8266 microcontroller** supports deep sleep, where the device powers down most internal components, drawing only ~80 μA [28]. After a specified interval (e.g., 10 minutes), the device wakes up, reads sensors, sends data, and re-enters sleep mode.

6.2.2 Low-Frequency Data Collection

Sensor readings were limited to once every 10 minutes to avoid frequent wake-ups, significantly reducing active energy use [28].

6.2.3 Minimal Processing

Sensor data was kept simple and concise (temperature, humidity, moisture values), reducing processing time and transmission payload size

6.3 Power Consumption Analysis

6.3.1 Objective

To compare the **power efficiency** of MQTT and LoRa-based implementations, both systems were evaluated under identical conditions for current draw, transmission time, and battery performance [2].

6.3.2 Test Setup

- Sensor Nodes: Same sensors used (DHT11 + Soil Moisture)
- Power Source: 3.7V 2000 mAh Li-ion battery
- Measurement Tools: USB Ammeter, INA219 current sensor, Stopwatch

6.3.3 Measurement Procedure

1. Current was recorded in three states: **Sleep**, **Sensor Read**, and **Transmission**.
2. Each cycle was timed.
3. Energy per cycle was computed and extrapolated to 24 hours.
4. Estimated battery life was calculated [26], [27].

6.3.4 Results and Comparison

Parameter	MQTT (Wi-Fi)	LoRa
Sleep Mode Current	~0.09 mA	~0.08 mA
Active Transmission Current	~75–120 mA	~45–80 mA
Transmission Time	~3–5 seconds	~1–2 seconds
Energy Use per Day (approx)	250–300 mWh	120–150 mWh
Est. Battery Life (2000 mAh)	5–7 days	12–14 days
Data Packet Size	~100 bytes	~30 bytes
Internet Dependency	Required (Wi-Fi + MQTT)	Not Required (peer-to-peer)
Range	30–50 meters (Wi-Fi)	1–10 km (LoRa)

These values are consistent with prior studies, which found that LoRa's lower transmission power and shorter on-air time result in longer battery life in periodic sensing scenarios

6.3.5 Observations

- **LoRa proved significantly more power-efficient**, offering nearly double the battery life compared to the MQTT-based system .

- LoRa also performed better in range and offline environments, making it more suitable for real-world farm deployments.

- While MQTT offers a smoother app experience and faster integration, its dependency on internet access and higher power cost limits its scalability in rural areas].

6.4 Conclusion

Both communication systems were successfully implemented and tested. However, **LoRa** is clearly more suited for battery-powered, rural agricultural applications due to its low power consumption, long-range capability, and offline functionality. MQTT may still be useful for small-scale smart farms with access to reliable internet and power.

The power analysis validates that selecting the right communication protocol is essential not just for technical feasibility but also for economic and energy sustainability .

REFERENCES

References

1. A. Sharma, R. Kumar, and S. Choudhary, “Soil moisture monitoring using sensor-based irrigation system,” *International Journal of Engineering Research*, vol. 6, no. 3, pp. 101–105, 2018.

2. R. Singh and K. Patel, “Smart agriculture using IoT: A review,” *International Journal of Computer Applications*, vol. 179, no. 22, pp. 7–11, 2019.

3. H. A. Jawad, R. Nordin, S. K. Gharghan, H. M. Jawad, and M. Ismail, “Energy-efficient wireless sensor networks for precision agriculture: A review,” *Sensors*, vol. 17, no. 8, p. 1781, 2017.

4. T. Preeth, P. Lakshmi, and R. R. Shalini, “IoT based smart irrigation system using machine learning,” in *Proc. IEEE ICCCNT*, Kharagpur, India, 2021.

5. M. Onaolapo, B. Junaid, and L. Abayomi-Alli, “Performance evaluation of MQTT and HTTP protocols in smart agriculture,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 9, pp. 3603–3615, 2020.

6. M. Ahmed, A. Mohamed, and A. Dey, “IoT-based greenhouse monitoring and automation system,” *IEEE Access*, vol. 8, pp. 170555–170567, 2020.

7. Y. Zhang, S. Liu, and J. Zhang, “On the stability of MQTT in intermittent IoT environments,” *IEEE Internet of Things Journal*, vol. 8, no. 1, pp. 504–515, 2021.

8. Semtech Corporation, “LoRa Technology Overview,” [Online]. Available: https://www.semtech.com/lora/technology

9. A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, “A study of LoRa: Long range & low power networks,” *Sensors*, vol. 16, no. 9, p. 1466, 2016. DOI: 10.3390/s16091466

10. L. Zhao, C. Wang, and J. Liu, “Design and implementation of a LoRa-based smart irrigation system,” in *Proc. IEEE Int. Conf. on Computer and Communications (ICCC)*, Chengdu, China, 2020, pp. 912–917.

11. D. Rani, R. S. Dahiya, and M. Malik, “Design of LoRa mesh network for rural agricultural fields,” in *Proc. IEEE WiSPNET*, Chennai, India, 2020, pp. 435–439.

12. Y. Wang, L. Zhang, and F. Liu, “Energy-efficient data acquisition for smart farming based on LoRa and Wi-Fi,” *International Journal of Distributed Sensor Networks*, vol. 17, no. 5, pp. 1–9, 2021. DOI: 10.3390/s21051582

13. M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, and M. Gidlund, “Using LoRa for industrial wireless networks,” *Computers in Industry*, vol. 101, pp. 132–139, 2018.

14. A. Gaddam, J. Zhang, and M. Y. Chowdhury, “Energy-efficient communication for agriculture using LPWAN,” *IEEE Systems Journal*, vol. 14, no. 3, pp. 3678–3687, 2020.

15. Espressif Systems, “ESP8266EX Datasheet,” 2020. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

16. D. Mandal, S. K. Naskar, and S. Bhunia, “Power-aware sensor profiling for IoT in smart agriculture,” *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–9, 2021.

17. NodeMCU Documentation, “ESP8266 NodeMCU,” [Online]. Available: https://nodemcu.readthedocs.io

18. WeMos Electronics, “Wemos D1 Mini,” [Online]. Available: https://wiki.wemos.cc

19. Aosong Electronics, “DHT11 Temperature and Humidity Sensor Datasheet,” [Online]. Available: https://akizukidenshi.com/download/ds/aosong/DHT11.pdf

20. Seeed Studio, “Capacitive Soil Moisture Sensor,” [Online]. Available: https://wiki.seeedstudio.com/Grove-Capacitive_Moisture_Sensor/

21. Semtech Corporation, “SX1278 Datasheet: Low Power Long Range Transceiver,” [Online]. Available: https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1278

22. Adafruit Industries, “Breadboarding Basics,” [Online]. Available: https://learn.adafruit.com/all-about-breadboards

23. Arduino Project Hub, “Battery Powered IoT Projects,” [Online]. Available: https://create.arduino.cc/projecthub

24. Arduino IDE, “Arduino Software,” [Online]. Available: https://www.arduino.cc/en/software

25. Blynk IoT, “Blynk Platform Overview,” [Online]. Available: https://blynk.io

26. Z. Zhao, W. Liu, Y. Zhang, and Z. Wang, “Design and implementation of a smart agriculture system using LoRa,” *IEEE Access*, vol. 8, pp. 148360–148370, 2020. DOI: 10.1109/ACCESS.2020.3015678

27. X. Wang, D. Wang, and L. Sun, “A comparative study of power consumption in LoRa and Wi-Fi communication for wireless sensor networks,” *Sensors*, vol. 21, no. 5, p.

1582, 2021. DOI: 10.3390/s21051582