BHARATI VIDYAPEETH'S

JAWAHARLAL NEHRU INSTITUTE OF TECHNOLOGY (POLY.)

Katraj – Dhankawadi, Pune - 411 043.

A PROJECT REPORT

ON

"Smart AgroSense: IoT-Based Multi-Point Wireless Sensor System for Precision Farming"

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DIPLOMA IN

ELECTRONICS AND TELECOMMUNICATION ENGINEERING

SUBMITTED TO

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI

SUBMITTED BY

Name of Student:

Mr. Mayank D. Thakkar

Enrolment No.

2201250349

GUIDED BY: Mr. Amit S. Patil

BHARATI VIDYAPEETH'S

JAWAHARLAL NEHRU INSTITUTE OF TECHNOLOGY (POLY.)

Katraj – Dhankawadi, Pune - 411 043.

CERTIFICATE

This is to Certify that **Mr. Mayank D. Thakkar** from **Bharati Vidyapeeth Jawaharlal Nehru Institute of Technology (Poly.), Pune-43** Institute having Enrolment No: **2201250349** has completed *Project of final year having* title **"Smart AgroSense: IoT-Based Multi-Point Wireless Sensor System for Precision Farming"** during the academic year 2024-2025. The project is completed in a group consisting of **4** persons under the guidance of **Mr. A. S. Patil**.

Internal Examiner **External Examiner**

Head of Department **Principal**

ACKNOWLEDGEMENT

I, Mr. Mayank Thakkar, a proud team member of the project *"Smart AgroSense: IoT-Based Multi-Point Wireless Sensor System for Precision Farming,"* express my sincere gratitude to everyone who has supported me throughout this incredible learning experience.

I would like to thank our guide, **Mr. A. S. Patil**, for his relentless support and valuable guidance. His ability to provide clarity during complex phases of the project and his encouragement were key factors in our progress. I am especially grateful for the time and effort he devoted to our growth.

I also express my appreciation to our Principal, **Mr. R. R. Utturkar**, for giving me the platform to explore and implement innovative ideas. His trust in our abilities gave me the confidence to contribute to this project with enthusiasm.

I’m deeply thankful to **Khyatee Electronics Pvt. Ltd.** for their financial and technical support, which enabled us to acquire the necessary components for successful project execution. Their trust in our team meant a lot.

Thanks are due to the teaching and non-teaching staff of our department for their moral support and technical assistance throughout the development phase.

Above all, I am thankful to my family and friends for their unwavering encouragement and understanding. Their support has been essential throughout this journey.

This project gave me practical exposure to IoT and wireless sensor systems and helped me grow personally and professionally. I am honored to be a part of *Smart AgroSense,* a project that envisions a smarter agricultural future.

ABSTRACT

This report presents the design and implementation of **Smart AgroSense**, an IoT-based multi-point wireless sensor system tailored for real-time environmental monitoring and precision farming. The system utilizes a **many-to-one communication architecture** built around **ESP32 microcontrollers** and **nRF24L01 transceiver modules**, forming an efficient and energy-conscious **ad hoc wireless network**. It is specifically engineered to gather and transmit vital agricultural data, including **temperature**, **humidity**, and **soil moisture** levels from multiple remote sensor nodes to a centralized **base station** for processing and analysis.

Each sensor node is equipped with a **DHT22 temperature and humidity sensor** and a **capacitive soil moisture sensor**, enabling accurate, location-specific environmental data collection. The nodes transmit data in a fixed, time-staggered sequence to avoid collisions and ensure ordered delivery. The system employs **unique RF24 pipeline addresses**, **reduced data rates (250KBPS)**, **high power amplification**, and **randomized delays** between transmissions to improve communication reliability and maximize range—crucial for outdoor agricultural settings.

The base station, also built on an ESP32, receives the data and forwards it to a real-time **IoT dashboard via Blynk IoT** for remote monitoring. It dynamically parses the incoming sensor data from each node, maintaining the transmission order: **NODE1 → NODE2 → NODE3**. This structured communication model allows for synchronized motor control based on moisture thresholds, which is demonstrated through the activation of a water pump connected via a relay module.

Beyond real-time display and motor actuation, the system is designed with modularity and future scalability in mind. Potential expansions include **data storage**, **analytics**, **notification systems**, and integration with **rainwater harvesting** and **solar-powered modules**, making it a sustainable and smart farming solution. The architecture serves as a proof of concept for leveraging **low-cost**, **low-power**, and **long-range** RF modules in applications where continuous, reliable environmental monitoring is essential.

Smart AgroSense thus contributes to the advancement of **precision agriculture**, showcasing the feasibility of decentralized wireless sensor networks in resource-constrained environments and offering a foundation for further innovations in smart farming technologies.

LIST OF ABBRIVATIONS

IoT - Internet of Things
ESP32 - Espressif Systems 32-bit Microcontroller
RF - Radio Frequency
nRF24L01 - Nordic Semiconductor RF Transceiver Module (2.4GHz)
KBPS - Kilobits Per Second
DHT22- Digital Humidity and Temperature Sensor (Model DHT22)
NODE - Networked Observation and Data Entity (contextual, not a standard)
pH - Potential of Hydrogen (a measure of acidity/alkalinity)
LoRa - Long Range
ZigBee - A specification for a suite of high-level communication protocols using low-power digital radios
AI - Artificial Intelligence
Wi-Fi - Wireless Fidelity
Blynk - (Brand name) — No expansion; IoT platform for mobile-based dashboards
Ubidots - (Brand name) — No expansion; IoT data analytics platform
BLE - Bluetooth Low Energy (implied by mention of Bluetooth in context, though not abbreviated in text)
MCU -Microcontroller Unit (implied in context with mention of microcontrollers, though not abbreviated in text)
ESP8266 - Espressif Systems Microcontroller with Wi-Fi capabilities
Node MCU - Node MicroController Unit (Development board based on ESP8266)
DHT11 - Digital Humidity and Temperature Sensor (Model DHT11)
YL-69 - Soil Moisture Sensor Module (YL-69 probe with comparator board)
LPWAN - Low Power Wide Area Network
5G - Fifth Generation of Mobile Network Technology
ADC - Analog-to-Digital Converter
SPI - Serial Peripheral Interface
GPIO - General Purpose Input/Output.
JSON - JavaScript Object Notation (used for data structuring, though not mentioned directly in the code, it's implied).
CE - Chip Enable (used in the context of the nRF24L01 module).
CSN - Chip Select Not (used in the context of the nRF24L01 module).
PALevel - Power Amplifier Level (used for setting the power level for the nRF24L01 module).

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

Chapter-1

INTRODUCTION

1.1 Overview of the Smart AgroSense System

This project develops a Wireless Sensor Network (WSN) aimed at agricultural monitoring by utilizing Internet of Things (IoT) technology. The system includes three sensor nodes and one receiver node, all working together to provide real-time insights into farm conditions. Each sensor node is equipped with an ESP32 microcontroller for data processing, an nRF24L01 wireless communication module for data transmission, a capacitive moisture sensor for soil moisture measurement, and a DHT22 sensor for monitoring temperature and humidity. The receiver node collects data from the sensor nodes and can transmit it to a central server or display for further analysis.

Agricultural monitoring plays a critical role in optimizing crop growth, reducing water waste, and improving overall farm productivity. Traditionally, farming relies on manual methods of monitoring, which are often inefficient, time-consuming, and prone to human error. This project seeks to address these challenges by developing a WSN that allows for continuous, real-time monitoring of various environmental parameters. By leveraging IoT technology, farmers can access accurate data that can guide them in making timely and informed decisions regarding their farm’s conditions.

The key challenges in modern agriculture include inefficient water usage, limited real-time monitoring, and the degradation of soil health. Overwatering or underwatering are common issues that arise from traditional irrigation practices, often resulting in wasted water resources and lower crop yields. Furthermore, manual monitoring methods can delay necessary interventions, which in turn reduces farm productivity. Finally, the absence of timely data often contributes to soil degradation, negatively affecting long-term crop yields and farm sustainability. This project addresses these problems by providing a networked system that offers real-time data, enabling efficient irrigation and better soil management.

The methodology behind this project involves a multi-node wireless sensor network consisting of three sensor nodes and one receiver node. Each sensor node is designed to measure key environmental parameters, including soil moisture, temperature, and humidity. The sensor nodes communicate wirelessly with the receiver node via the nRF24L01 module. This communication system allows for the seamless transmission of data from the nodes to the receiver, where it can then be forwarded to a central server for analysis. The data received from the sensor nodes can be used to derive actionable insights, which can inform farmers' decisions regarding irrigation scheduling, crop management, and overall farm conditions.

The system architecture is based on a distributed setup where each sensor node is responsible for monitoring a specific area of the farm. These sensor nodes collect data on soil moisture, temperature, and humidity and transmit the information to the receiver node, which acts as the central hub for data aggregation. The receiver node can then send the gathered data to a central server or a display system for further analysis. This architecture enables farmers to have a comprehensive overview of their farm conditions and allows them to make data-driven decisions. The system has been shown to be effective in monitoring various environmental parameters, demonstrating the feasibility of WSNs in agricultural settings.

1.2 Project Results and Achievements

The results of the project indicate that the developed system successfully provides real-time insights into farm conditions, particularly concerning soil moisture, temperature, and humidity. These real-time insights enable farmers to make better-informed decisions about irrigation, ultimately leading to optimized water usage and improved crop yields. By minimizing overwatering or underwatering, the system helps conserve water, a valuable resource in agriculture. Furthermore, the improved decision-making process positively impacts crop yields by ensuring that plants receive the optimal conditions for growth, resulting in reduced losses and enhanced productivity.

The benefits of this wireless sensor network are manifold. By enabling continuous monitoring and data-driven decision-making, the system enhances operational efficiency on the farm. Farmers are empowered to optimize irrigation schedules, conserve water, and adjust farm conditions based on real-time data. These improvements contribute to enhanced crop yields, reduced environmental impact, and better resource management. Ultimately, the project demonstrates the potential of IoT-based solutions in revolutionizing agricultural practices, making them more efficient, sustainable, and productive.

Looking ahead, the project offers significant potential for future development and expansion. One key area for enhancement is the integration of cloud services, which would allow for remote monitoring and data analysis, making it possible for farmers to monitor their farm conditions from anywhere in the world. Additionally, the sensor network can be expanded by adding more nodes to monitor additional parameters, such as soil pH, light intensity, and air quality, providing a more comprehensive overview of farm conditions.

Another exciting avenue for future development is the creation of a user-friendly interface that would allow farmers, even those with limited technical knowledge, to easily access and interpret the data collected by the system. The integration of machine learning algorithms could further enhance the system’s capabilities, providing predictive analytics to help farmers anticipate and mitigate potential problems before they occur. Finally, the system could be integrated with other advanced farming technologies, such as precision agriculture and vertical farming systems, to create a holistic and fully optimized farming ecosystem.

Through these developments, the system could significantly contribute to the advancement of sustainable and efficient farming practices

Chapter No. 2

LITERATURE SURVEY

2.1 Technological Advancements in IoT-Based Agriculture

The integration of Wireless Sensor Networks (WSNs) with the Internet of Things (IoT) has revolutionized agricultural monitoring, enabling real-time data collection and analysis. Traditional farming methods often relied on manual monitoring, which was time-consuming and prone to errors. Recent advancements have addressed these challenges by deploying automated systems that enhance efficiency and decision-making.

Sensor Technologies in Agriculture

Early systems utilized basic microcontrollers and resistive sensors, which faced issues like corrosion and limited accuracy. Modern implementations have adopted capacitive soil moisture sensors, which offer improved durability and precision. Temperature and humidity monitoring has also advanced with sensors like the DHT22, providing better accuracy than earlier models. The transition to microcontrollers such as the ESP32 has facilitated enhanced processing capabilities and wireless communication, essential for remote agricultural systems.

Communication Protocols

Initial deployments often used Bluetooth or Wi-Fi, which were inadequate for large-scale farms due to limited range. Recent systems employ modules like the nRF24L01 for low-power, short-range communication and LoRa for long-distance transmission. These technologies ensure stable data transfer even in remote areas. Mesh networking protocols like ZigBee have been introduced to support multi-node communication, improving scalability and robustness in agricultural settings.

Data Management and Analysis

Cloud integration has become pivotal for data storage and analysis. Platforms such as ThingSpeak, Blynk, and Ubidots enable remote monitoring and visualization of sensor data. This facilitates real-time access and allows the application of data analytics and machine learning techniques to predict environmental trends and automate irrigation schedules. Studies have explored the use of artificial intelligence in agricultural IoT systems to provide predictive insights, aiding farmers in making informed decisions regarding irrigation, fertilization, and crop management.

Energy Efficiency and Sustainability

Energy consumption is a critical consideration in WSN deployments. Since many agricultural environments lack consistent power sources, systems are designed to operate on batteries or solar energy. Implementations feature low-power design techniques like duty cycling, sleep modes, and energy-efficient communication protocols. These improvements have led to systems that are more reliable and sustainable over the long term.

Security and Reliability

As more devices become connected, ensuring data security and system reliability is paramount. Research emphasizes the importance of encrypted communication channels and secure authentication mechanisms to protect data integrity and prevent unauthorized access. Implementing redundancy and error-handling mechanisms enhances system reliability, ensuring continuous monitoring even in the event of individual node failures.

2.2 Literature Review

This section presents key research contributions in the field of IoT-based agricultural monitoring and WSN applications:

- **Deshpande et al. (2022):** Developed a low-cost soil moisture and temperature monitoring system using Raspberry Pi and IoT technologies, achieving high accuracy in real-world conditions.
- **Sanjeevi et al. (2020):** Proposed a scalable WSN architecture for precision agriculture, focusing on efficient water resource management through IoT integration.
- **Mishra et al. (2022):** Introduced an IoT-based smart agriculture monitoring and automatic irrigation system using ESP8266 Node MCU, highlighting real-time data collection and automated irrigation.
- **Muthmainnah et al. (2023):** Developed a soil temperature and humidity monitoring system using DHT11 and YL-69 sensors for greenhouse applications.
- **Wayangkau et al. (2021):** Created an IoT-based system for monitoring soil moisture and temperature in onion cultivation, enhancing crop management.
- **Mahajan & Badarla (2021):** Proposed a cross-layer protocol for WSN-assisted IoT smart farming applications using nature-inspired algorithms to optimize routing and energy efficiency.
- **Kumar & Majid (2023):** Designed an energy-efficient IoT-based WSN architecture for precision agriculture monitoring using machine learning techniques to automate irrigation decisions.
- **Sowmya et al. (2023):** Developed an IoT and WSN-based autonomous farming robot capable of monitoring environmental parameters and performing tasks like pesticide spraying and weed control.
- **Aldhaheri et al. (2024):** Investigated the potential of LoRa technology in smart agriculture, discussing network architecture design, sensor deployment strategies, and energy management techniques.
- **Shanmuga Sundaram et al. (2019):** Provided a comprehensive survey on LoRa networking, addressing technical challenges and recent solutions for deploying LoRa networks in precision agriculture.
- **Rafi et al. (2025):** Analyzed the performance trade-offs between LPWAN, 5G, and hybrid connectivity models in agricultural applications, recommending optimal IoT connectivity solutions.
- **Swaraj & Sowmyashree (2020):** Implemented an IoT-based smart agriculture monitoring and irrigation system, utilizing sensors and microcontrollers for efficient farm management.
- **Kumar et al. (2022):** Explored the use of machine learning in WSNs for agriculture, addressing challenges like sensor node power limitations and communication failures.
- **Lavanya & Srinivasan (2018):** Conducted a survey on agriculture and greenhouse monitoring using IoT and WSN, highlighting the benefits and challenges of these technologies.

Chapter No. 3

SCOPE OF PROJECT

3.1 Functional Scope of the Project

The scope of this project revolves around the design and implementation of a low-cost, energy-efficient, and scalable Wireless Sensor Network (WSN) integrated with Internet of Things (IoT) technology to support real-time agricultural monitoring. The project aims to empower farmers with actionable data to optimize their farm operations by providing continuous insights into soil moisture levels, temperature, and humidity across different zones of the farm. By using a distributed system comprising three sensor nodes and one central receiver node, the project ensures comprehensive environmental monitoring over a selected agricultural area.

Each sensor node in the network is equipped with an ESP32 microcontroller for local data processing, a capacitive soil moisture sensor to measure water content in the soil, and a DHT22 sensor to record temperature and humidity. The nRF24L01 wireless module is used to enable seamless and low-power communication between the nodes and the central receiver. The receiver node collects data from all sensor nodes and acts as a central hub that can display the information locally or transmit it to a cloud-based system for remote access and long-term analysis. This architecture allows for real-time decision-making, which is crucial for maintaining optimal growing conditions and efficient resource management on the farm.

The primary focus of this project is to address key challenges in modern agriculture, such as inefficient water use, limited real-time visibility into field conditions, and the lack of timely interventions due to manual monitoring. Traditional methods often lead to overwatering or underwatering, which not only wastes water but also negatively affects crop yields and soil health. This project offers a smart alternative by enabling data-driven irrigation scheduling, which conserves water while ensuring that crops receive the appropriate moisture levels necessary for healthy growth.

The system is designed to be modular and scalable, allowing more sensor nodes or additional types of sensors—such as soil pH, light intensity, or air quality sensors—to be added easily in the future. This flexibility makes the project suitable not only for small-scale farms but also for larger agricultural operations looking to transition into precision farming practices. In its current scope, the system is suitable for deployment in both open fields and greenhouse environments, where accurate monitoring of microclimates is essential.

3.2 Future Scalability and Impact

One of the broader objectives of the project is to make advanced technology accessible to farmers, even those with limited technical knowledge. With the potential integration of cloud services and mobile-based applications in the future, the system can offer remote access to farm data, making it easier for farmers to monitor and control farm conditions from anywhere in the world. Real-time alerts and predictive analysis features can also be incorporated later to assist in proactive farm management.

In terms of future scalability, the project is designed to support enhancements such as automated irrigation control using solenoid valves, where water delivery can be directly controlled based on the sensor data. Moreover, integrating the system with machine learning algorithms can enable predictive maintenance, anomaly detection, and intelligent decision-making. This will further increase the efficiency of farm operations and reduce human effort and error.

In conclusion, the scope of this project goes beyond simply measuring environmental parameters. It lays the foundation for a complete smart farming ecosystem that is adaptable, reliable, and impactful. By introducing this technology into agricultural settings, the project contributes to sustainable farming practices, improved crop productivity, and efficient resource utilization. These advancements mark a significant step toward the future of precision agriculture and demonstrate the transformative power of integrating IoT and WSN technologies in the farming sector.

Chapter No. 4

METHODOLOGY

4.1 Methodological Approach

This section outlines the methodologies employed to design, develop, and evaluate the wireless sensor network (WSN) for agricultural monitoring. It includes the procedures for data collection, system architecture design, network setup, and data analysis. The objective is to present a clear and coherent explanation of the approach used to achieve real-time monitoring for optimized crop management, water conservation, and enhanced productivity.

System Design

The project began with the design of a multi-node WSN architecture consisting of three sensor nodes and one receiver node. Hardware components were selected based on criteria such as reliability, power efficiency, communication range, and ease of integration.

- **Microcontroller**: The ESP32 was selected due to its low power consumption, built-in Wi-Fi and Bluetooth capabilities, and suitability for IoT applications.
- **Wireless Communication**: The nRF24L01 module was chosen for its long-range, low-power wireless transmission capability, ensuring efficient communication between nodes.
- **Sensors**:
  - Capacitive soil moisture sensors were used due to their higher accuracy and reduced susceptibility to corrosion, making them suitable for long-term field use.
  - DHT22 sensors were implemented to measure temperature and humidity, chosen for their accuracy, stability, and seamless integration with the ESP32.

Network Setup

The communication network connects sensor nodes wirelessly to a central receiver node. Each sensor node gathers real-time data on soil moisture, temperature, and humidity, and transmits it via nRF24L01 modules. The receiver node collects and aggregates this data, enabling holistic environmental monitoring. The wireless architecture removes the need for complex wiring and supports scalability for future deployments.

Data Collection

Data collection was designed to occur periodically from all sensor nodes. The specific parameters monitored include:

- **Soil Moisture**: Captured as ADC values representing moisture levels in the soil.
- **Temperature and Humidity**: Recorded using the DHT22 sensor, providing insights into ambient conditions critical for irrigation and plant health.

Sensor nodes were strategically distributed across the agricultural area to capture spatial variability, ensuring that data reflected diverse microclimatic conditions.

Data Analysis

The data collected was processed to yield actionable insights for effective farm management:

- **Data Aggregation**: The receiver node compiles data from all sensor nodes, allowing for centralized analysis and long-term record keeping.
- **Decision-Making Insights**: By analyzing real-time readings, the system supports smart irrigation decisions. For instance, low soil moisture triggers irrigation or alerts.
- **Data Validation**: Sensor outputs were compared with manual field measurements to ensure accuracy. Periodic sensor calibration was also performed.
- **Predictive Analytics (Future Work)**: Future enhancements will include machine learning algorithms to forecast irrigation needs based on historical trends and weather data.

Data Summary Table

Node	Soil Moisture (ADC Value)	Temperature (°C)	Humidity (%)
Node 1	1860	27.8	62.1
Node 2	1320	28.4	58.7
Node 3	2450	26.3	65.4

Table 4.1: Sensor Node Environmental Readings Summary

4.2 Justification of Methodological Choices

The methodological choices made in this project were informed by the need for a reliable, energy-efficient, and cost-effective system suitable for real-time agricultural monitoring in both small- and large-scale deployments.

ESP32 Microcontroller

Selected for its:

- Low power consumption
- Integrated Wi-Fi and Bluetooth modules
- Compatibility with a wide range of sensors

nRF24L01 Wireless Communication Module

- Low power usage
- Long-range communication capabilities
- High data transmission reliability
- Ease of use in mesh or star network topologies

Capacitive Moisture Sensors

- Higher durability and longevity compared to resistive alternatives
- Better performance in outdoor conditions
- Accuracy in measuring volumetric water content in the soil

DHT22 Temperature and Humidity Sensor

- High accuracy in measuring ambient temperature and relative humidity
- Stability and repeatability of readings over time
- Proven compatibility with ESP32 platforms

Wireless Communication Architecture

- Avoid the complexity and limitations of wired systems
- Allow flexible placement and repositioning of sensor nodes
- Enable easy expansion of the network with additional nodes

The combination of these components and methods ensures a robust and scalable system capable of supporting advanced agricultural practices. The system not only meets current monitoring needs but also provides a strong foundation for future innovations such as cloud integration, mobile applications, and intelligent decision-making powered by machine learning.

Chapter No. 5

DETAILS OF DESIGN, WORKING AND PROCESSES

The design and implementation of a Wireless Sensor Network (WSN) for agricultural monitoring involve a multi-layered architecture that integrates hardware components, sensor interfacing, wireless communication, and efficient data processing. This section provides a comprehensive explanation of the system's internal design, its functional components, and the processes that enable real-time environmental monitoring. The project comprises multiple sensor nodes, each responsible for measuring key parameters such as soil moisture, temperature, and humidity. These nodes are built around the ESP32 microcontroller and communicate wirelessly with a central receiver node using the nRF24L01 transceiver module. The data collected is then analyzed to assist in optimizing irrigation practices, conserving water, and improving crop productivity. This section also discusses the logic behind component selection, sensor calibration, and the workflow of each node, supported by detailed diagrams to visualize the system's operation.

5.1 Overview of Hardware Components Used

1. ESP32 Development Board

Fig 5.1.1: ESP 32 Pinout Diagram

The **ESP32** microcontroller serves as the central processing unit for both the sensor nodes and the receiver node in the designed wireless sensor network (WSN). Developed by **Espressif Systems**, the ESP32 is a powerful, versatile, and cost-effective system-on-chip (SoC) widely adopted in Internet of Things (IoT) applications. Its rich set of features, low power consumption, and integrated communication interfaces make it an ideal choice for agricultural monitoring systems that require real-time data acquisition and wireless communication

Key Features of ESP32:

- **Dual-core Tensilica LX6 processor**, operating at up to **240 MHz**, providing high-speed processing capabilities.
- **Built-in Wi-Fi (802.11 b/g/n)** and **Bluetooth 4.2 (Classic and BLE)**, facilitating wireless communication and future cloud-based integrations.
- **520 KB SRAM** and up to **4 MB Flash memory**, supporting efficient code execution and data handling.
- **12-bit ADC (Analog-to-Digital Converter)** with up to **18 input channels**, suitable for reading analog sensors such as soil moisture probes.
- **2 × DACs (Digital-to-Analog Converters)**, **3 × UART**, **2 × I2C**, **3 × SPI**, and **PWM** support for versatile peripheral interfacing.
- **32 programmable GPIO pins**, allowing flexible connection to various sensors and modules.
- Integrated **touch sensor inputs**, **temperature sensor**, and **Hall sensor** for additional sensing capabilities.
- **Deep sleep and ultra-low power modes**, essential for battery-powered field deployments.

ESP 32 Pin Configuration Overview:

The ESP32 development board (e.g., ESP32 DevKit V1) offers multiple general-purpose input/output (GPIO) pins. The following table outlines the typical usage of key pins relevant to this project:

Pin Name	Function
3.3V / VIN	Power supply input (3.3V logic level)
GND	Ground
GPIO32–GPIO39	ADC channels for reading analog signals
GPIO21, GPIO22	I2C communication (SDA, SCL)
GPIO18, GPIO19, GPIO23, GPIO5	SPI interface for nRF24L01
GPIO25, GPIO26	DAC output (optional)
GPIO1, GPIO3	UART TX and RX (for serial communication)
EN	Chip enable/reset

Table 5.1.1: ESP32 Pin Configuration and Functions

Application of ESP32 in the Project:

In this capstone project, the ESP32 is used in all nodes to perform the following roles:

- **Sensor Interfacing**:
  - Reads analog data from the **capacitive soil moisture sensor** via ADC pins.
  - Communicates with the **DHT22 temperature and humidity sensor** via digital GPIO.
- **Wireless Communication**:
  - Interfaces with the **nRF24L01 module** using the SPI protocol for data transmission to the receiver node.
- **Data Processing and Packaging**:
  - Formats sensor readings into data packets with node-specific identifiers.
- **Power Management**:
  - Utilizes low-power modes to enhance battery life during field deployment.
  - Supports real-time scheduling for periodic sensing and data transfer.
- **Scalability and Future Integration**:
  - ESP32's built-in Wi-Fi allows future integration with cloud platforms for remote monitoring.
  - Supports over-the-air (OTA) firmware updates, ensuring easy system upgrades.

2. nRF24L01 Wireless Transceiver Module

Fig 5.1.2: NRF24L01

The **nRF24L01** is a highly integrated, ultra-low power **2.4 GHz RF transceiver module** designed by **Nordic Semiconductor**. It is widely used for wireless communication in embedded systems due to its low cost, long-range capability, and reliable performance. In this project, the nRF24L01 module is employed for **wireless data transmission between sensor nodes and the receiver node**, operating over the 2.4 GHz ISM band.

For field deployment and compatibility with the ESP32, the **nRF24L01 module with an onboard AMS1117 3.3V voltage regulator** is used. This variant allows direct connection to a **5V power supply**, which is often the default output from microcontroller boards, protecting the transceiver's core logic.

Key Features:

- **Frequency Range**: 2.4 GHz ISM band (2400–2525 MHz)
- **Data Rate Options**: 250 kbps, 1 Mbps, 2 Mbps (programmable)
- **Modulation Technique**: GFSK (Gaussian Frequency Shift Keying)
- **Transmission Range**: Up to 100 meters (line of sight) with onboard PCB antenna
- **Power Supply**:
  - **Input Voltage**: 3.3V (core module), 5V (regulated version)
  - **Onboard AMS1117 regulator** allows safe 5V input
- **Interface**: SPI (Serial Peripheral Interface)
- **Multi-Device Support**: Communicates with up to 6 devices simultaneously via unique pipe addresses
- **Low Power Consumption**:
  - Transmission mode: ~11.3 mA
  - Standby mode: ~22 µA
- **Antenna**: Onboard PCB antenna or optional external antenna (depending on model)

Pin Configuration:

Pin	Name	Function
1	**GND**	Ground
2	**VCC**	Power supply input (3.3V for basic module, 5V for module with regulator)
3	**CE**	Chip Enable: Enables RX/TX modes
4	**CSN**	Chip Select Not: Activates SPI communication
5	**SCK**	SPI Clock: Serial clock input
6	**MOSI**	SPI Data Input: Master Out, Slave In
7	**MISO**	SPI Data Output: Master In, Slave Out
8	**IRQ**	Interrupt Request (optional, used for advanced communication handling)

Table 5.1.2: nRF24L01 Pin out and Function Description

Application in the Project:

The **nRF24L01 with onboard regulator** plays a vital role in the wireless communication infrastructure of the agricultural monitoring system. Its primary responsibilities include:

- **Data Transmission (Sensor Nodes)**:
  - Sensor data is sent from each node to the receiver node via the nRF24L01 module using SPI communication with the ESP32.
  - The CE and CSN pins are controlled by the ESP32 to configure transmission modes and activate communication.
- **Data Reception (Receiver Node)**:
  - The receiver node's nRF24L01 listens for data packets from all three sensor nodes.
  - Each sensor node transmits data on a separate logical pipe address for proper identification.
- **Reliable and Efficient Communication**:
  - Automatic packet handling, acknowledgment (ACK), and retransmission features ensure data integrity.
  - The module supports multi-point to point communication, ideal for star-topology sensor networks.
- **Power Efficiency**:
  - Low power modes make the module suitable for battery-powered applications.
  - The module enters sleep or standby mode when not transmitting, conserving energy.

Voltage Considerations:

- The **basic nRF24L01 module** must be powered at **3.3V only** and is **not 5V tolerant**.
- The **version with onboard AMS1117 regulator** can safely accept **5V on the VCC pin**, converting it internally to 3.3V.
- This makes it compatible with 5V microcontroller systems and simplifies wiring when using the ESP32 development board (which often has 5V output pins).

3. Sensors

Capacitive Soil Moisture Sensor

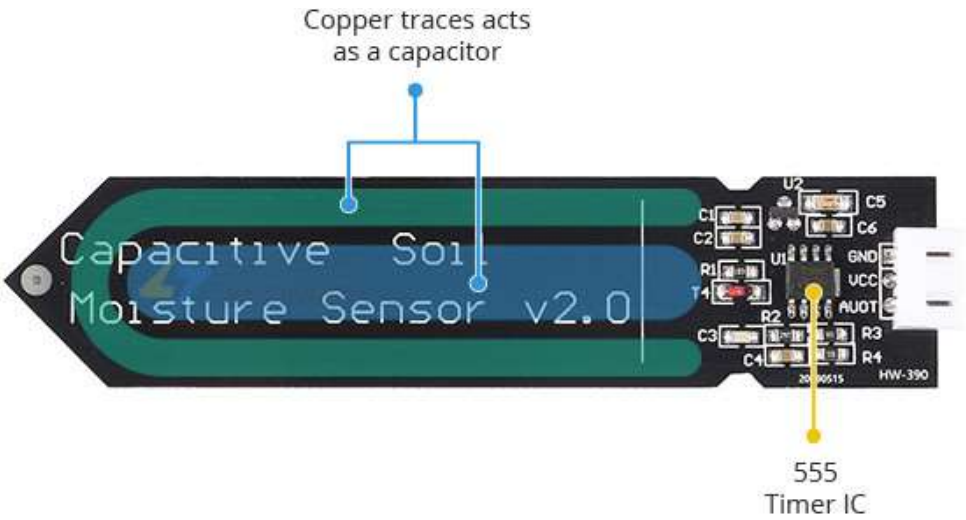


Fig 5.1.3: Capacitive soil Moisture Sensor

The **Capacitive Soil Moisture Sensor** is a crucial component used in this project to monitor soil moisture levels in real time. Unlike traditional resistive sensors that are prone to corrosion and degradation over time, capacitive sensors use capacitive sensing technology, which makes them more durable, accurate, and suitable for long-term agricultural use.

Key Features:

- **Capacitive sensing**: Measures changes in soil dielectric permittivity to determine moisture levels.
- **Non-corrosive design**: Unlike resistive sensors, there are no exposed metal electrodes that degrade over time.
- **Analog output**: Provides a continuous voltage signal corresponding to the soil's moisture content.
- **Operating voltage**: Typically 3.3V to 5V, making it compatible with most microcontrollers including the ESP32.
- **Low power consumption**: Ideal for battery-operated systems.
- **Compact and waterproof PCB construction**: Designed for direct insertion into soil.

Pin Configuration:

Pin	Label	Function
1	**VCC**	Power supply input (3.3V–5V)
2	**GND**	Ground
3	**AOUT**	Analog output (voltage signal)

Table 5.1.3: Capacitive Soil Moisture Sensor Pin out and Functions

Working Principle:

The capacitive soil moisture sensor works by measuring the **dielectric constant** of the surrounding soil. Moisture levels affect the dielectric properties of the soil—**wet soil has a higher dielectric constant than dry soil**. The sensor detects this change using a capacitor embedded in its PCB layout and converts it into a variable voltage signal.

- **Dry soil** results in **higher analog voltage output**
- **Wet soil** results in **lower analog voltage output**

This analog output is read by the **Analog-to-Digital Converter (ADC)** of the ESP32 microcontroller. The digital reading can then be processed, calibrated, and used to determine real-time soil moisture levels.

Application in the Project:

In the wireless sensor network (WSN) designed for agricultural monitoring:

- The **capacitive soil moisture sensor is connected to one of the ADC-capable GPIO pins** of the ESP32 at each sensor node.
- It provides **continuous analog readings** that reflect the current soil moisture condition at that location.
- These readings are:
  - Collected and digitized by the ESP32.
  - Transmitted wirelessly via the nRF24L01 module to the receiver node.
- The collected data can be:
  - Compared against moisture thresholds.
  - Used to make irrigation decisions.
  - Logged for long-term soil behavior analysis.

DHT22 Temperature and Humidity Sensor

The **DHT22** sensor, also known as **AM2302**, is a widely used digital sensor capable of measuring both **ambient temperature** and **relative humidity** with high accuracy and stability. In this project, the DHT22 sensor plays a critical role in monitoring environmental conditions at each sensor node, providing data necessary for analyzing plant health and irrigation requirements.

Fig 5.1.4: DHT 22 sensor

Key Features:

- **Measures two parameters**: Temperature (in °C) and Humidity (in %)
- **Temperature Range**: -40°C to +80°C
- **Temperature Accuracy**: ±0.5°C
- **Humidity Range**: 0% to 100% RH
- **Humidity Accuracy**: ±2–5% RH
- **Resolution**: 0.1°C (Temperature), 0.1% (Humidity)
- **Operating Voltage**: 3.3V to 6V (typically 3.3V for ESP32 compatibility)
- **Digital output**: Uses a single-wire protocol for communication
- **Sampling Rate**: 0.5 Hz (1 reading every 2 seconds)
- **Low Power Consumption**: ~1.5 mA during measurement, ~50 µA standby

Pin Configuration:

Pin	Label	Function
1	**VCC**	Power supply input (3.3V–5V)
2	**DATA**	Digital signal output for temperature and humidity
3	**NC**	Not connected (no internal function)
4	**GND**	Ground

Table 5.1.4: DHT22 Temperature and Humidity Sensor Pin out and Functions

Working Principle:

The DHT22 sensor contains a capacitive humidity sensor and a thermistor to measure the surrounding air and uses an internal chip to convert these analog values into a calibrated digital signal.

- The sensor sends data using a **single-wire serial communication protocol**.
- After being triggered by the microcontroller, the sensor responds with a **40-bit data stream**:
    - 16 bits for humidity
    - 16 bits for temperature
    - 8 bits for checksum (used for data validation)

The ESP32 reads this data using a digital GPIO pin, processes the 40-bit packet, and extracts the temperature and humidity values.

Application in the Project:

In the wireless sensor network designed for agricultural monitoring:

- The **DHT22 is connected to a digital GPIO pin** on the ESP32 at each sensor node.
- It measures **ambient temperature and humidity** every few seconds and supplies the values to the ESP32.
- These values are:
    - Digitally processed by the ESP32.
    - Combined with soil moisture readings into a structured data packet.
    - Wirelessly transmitted via the **nRF24L01 module** to the receiver node.
- The data is used to:
    - Analyze climatic conditions affecting soil moisture evaporation.
    - Inform irrigation schedules and crop health monitoring.

5.2 Overview of Integrated Development Environment Used

Fig 5.2.1: Arduino IDE visual

The **Arduino Integrated Development Environment (IDE**) is an open-source, cross-platform software tool designed to facilitate the programming and deployment of code onto microcontroller-based embedded systems. Developed primarily for the Arduino platform, it has become a widely adopted environment for rapid prototyping and implementation of Internet of Things (IoT), control systems, and automation projects across academic, research, and industrial domains.

The Arduino IDE supports the C and C++ programming languages and provides a simplified interface that abstracts many of the complexities traditionally associated with embedded system development. Its core features include syntax highlighting, real-time error messaging, an integrated serial monitor, and a direct interface for uploading compiled code to microcontrollers via USB or serial communication.

A notable strength of the Arduino IDE lies in its modular and extensible architecture. Through the Boards Manager, users can integrate support for a diverse range of hardware platforms, including ESP8266, ESP32, STM32, and ATmega-based development boards. Furthermore, the Library Manager enables seamless integration of community-maintained and officially supported libraries for sensors, wireless modules, actuators, and cloud platforms.

In the context of the Smart AgroSense project, the Arduino IDE served as the central development platform for both node and receiver firmware. Its compatibility with ESP32 enabled integration of Blynk IoT cloud services, while extensive library support (e.g., DHT.h, RF24.h, BlynkSimpleEsp32.h) facilitated rapid development and testing. The Serial Monitor was indispensable for real-time debugging, sensor validation, and wireless communication diagnostics.

The Arduino IDE also enhances the reproducibility and maintainability of embedded systems projects. Its organized file structure, wide community support, and availability of example sketches make it a robust educational and research tool. Due to its user-friendly interface and professional extensibility, it remains a preferred choice for microcontroller-based system design and implementation in academic research and industrial prototyping alike.

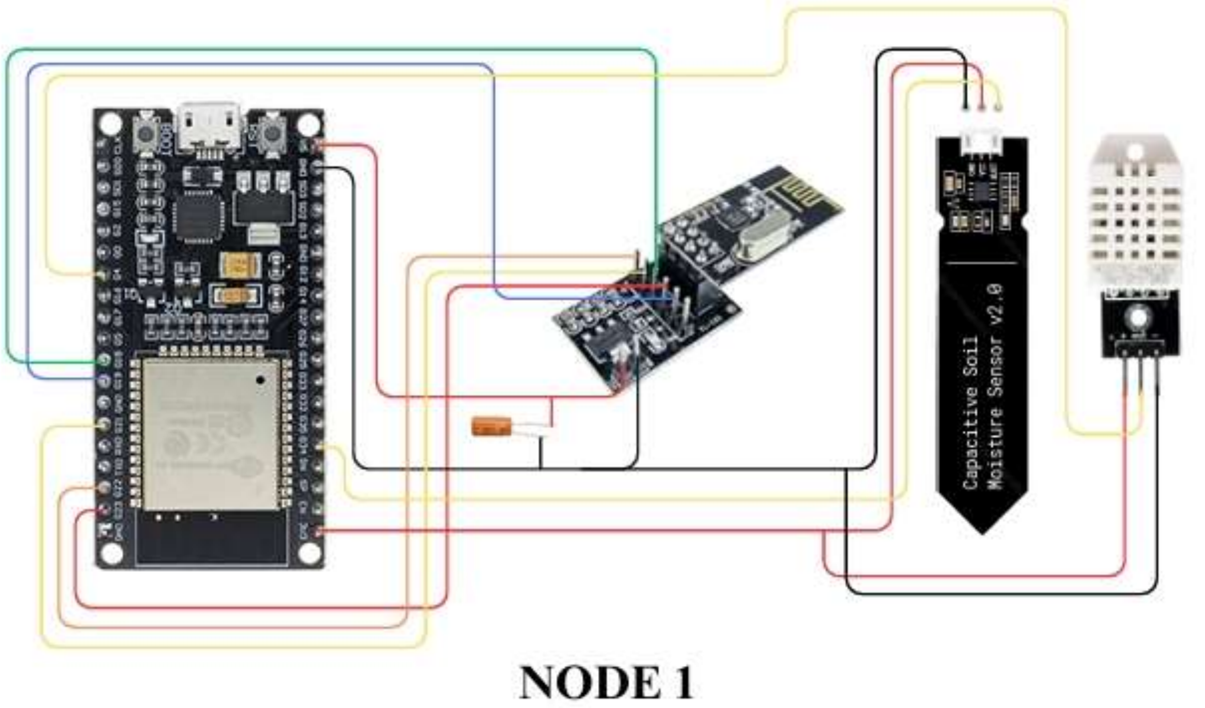
5.3 Working of System

Design and Working of Node 1

Node 1 is a sensor node in the wireless sensor network (WSN) designed for real-time agricultural monitoring. Its primary function is to sense environmental parameters and wirelessly transmit the data to the receiver node. Node 1 consists of the following components:

- ESP32 Microcontroller
  Acts as the central processing unit for the node, handling sensor data acquisition, signal processing, and communication.
- Capacitive Soil Moisture Sensor
  Measures the volumetric water content in the soil. It provides an analog voltage signal which is read using the ADC (Analog-to-Digital Converter) unit of the ESP32.
- DHT22 Sensor
  Measures temperature and relative humidity with high accuracy and stability. It communicates via a digital single-wire interface.
- nRF24L01 Wireless Module
  Facilitates wireless data transmission to the receiver node using a 2.4 GHz ISM band. It supports long-range, low-power communication.
- Power Supply
  A battery or regulated power source (e.g., 3.3V) powers the node, optimized for low-power operation in field conditions.

Fig 5.3.1: Node 1 Wiring Diagram



Working Principle of Node 1

Sensor Initialization
When the node is powered on, the ESP32 initializes all peripherals and sensors. This includes setting up GPIO pins, initiating ADC for the moisture sensor, and starting the communication with the DHT22.

Data Acquisition

The Capacitive Moisture Sensor sends an analog voltage corresponding to soil moisture. This analog signal is converted to a digital value using the ESP32's ADC.

The DHT22 Sensor transmits digital temperature and humidity readings.

Data Processing

The ESP32 processes the data:
Converts the ADC value to a percentage soil moisture reading.
Verifies the integrity of DHT22 data (e.g., using checksum or error checks).
Formats the readings into a structured data packet (e.g., JSON or comma-separated values).

Wireless Communication

The ESP32 communicates with the nRF24L01 module using SPI (Serial Peripheral Interface).
Data packets are transmitted wirelessly to the receiver node, identified by its unique address.

Sleep Mode (Power Efficiency)
After transmission, the ESP32 can enter a low-power deep sleep mode to conserve battery. It wakes up at regular intervals (e.g., every 10 minutes) for the next measurement cycle.

Program of Node 1

```
#include <SPI.h>
#include <RF24.h>
#include <DHT.h>

#define DHTPIN 4
#define DHTTYPE DHT22
#define SOIL_PIN 34
//#define RELAY_PIN 25

DHT dht(DHTPIN, DHTTYPE);
RF24 radio(22, 21);  // CE, CSN

const char NODE_ID[] = "N1";
const byte receiverAddress[6] = "BASE1";

struct __attribute__((packed)) SensorData {
  char nodeID[3];
  float temperature;
  float humidity;
  int soilMoisture;
 // bool relayState;
};

void setup() {
  Serial.begin(115200);
  dht.begin();
  //pinMode(RELAY_PIN, OUTPUT);

  if (!radio.begin()) {
    Serial.println("□  NRF24L01 not responding");
    while (1);
  }

  radio.setPALevel(RF24_PA_HIGH);
  radio.setChannel(108);
  radio.openWritingPipe(receiverAddress);
  radio.stopListening();

  Serial.print("□  Transmitter "); Serial.print(NODE_ID); Serial.println(" ready");
}
```

```
void loop() {
  SensorData data;
  strcpy(data.nodeID, NODE_ID);
  data.temperature = dht.readTemperature();
  data.humidity = dht.readHumidity();
  data.soilMoisture = analogRead(SOIL_PIN);
  //data.relayState = data.soilMoisture < 500;
  //digitalWrite(RELAY_PIN, data.relayState);

  Serial.printf("Sending %s → Temp: %.1f°C | Hum: %.1f%% | Soil: %d | ",data.nodeID, data.temperature, data.humidity, data.soilMoisture);

  bool sent = radio.write(&data, sizeof(data));
  Serial.println(sent ? "□ Sent!" : "□ Send failed");

  delay(random(2000, 3000));
}
```

Description of code:

This program is designed for an ESP32-based wireless sensor node that monitors environmental conditions—specifically **temperature**, **humidity**, and **soil moisture**—and transmits this data to a base station using the **nRF24L01 wireless transceiver**. It utilizes the **DHT22 sensor** to measure temperature and humidity and a **capacitive soil moisture sensor** connected to an analog pin (GPIO 34) for soil moisture readings. The nRF24L01 module is configured to communicate over SPI using GPIO 22 (CE) and GPIO 21 (CSN), and the sensor node identifies itself with a unique ID ("N1") when transmitting data.

The code begins by including the necessary libraries: SPI.h and RF24.h for handling wireless communication, and DHT.h for interacting with the DHT22 sensor. It defines key hardware pin assignments and creates a data structure called SensorData, which stores the node ID, temperature, humidity, and soil moisture reading. The structure is packed to ensure efficient data transmission without padding bytes.

During setup, the serial communication is initialized for debugging purposes, the DHT22 sensor is started, and the nRF24L01 is configured. The radio module is set to a high power level and a custom communication channel (channel 108) to reduce interference from other 2.4 GHz devices. The writing pipe is opened with a designated address (“BASE1”) to send data to the receiver, and the module is placed into transmit mode.

In the main loop, the ESP32 continuously reads data from the DHT22 and soil moisture sensor. It then populates the SensorData structure with the collected values and sends this data wirelessly to the base station. Each transmission is confirmed via the serial monitor with a formatted output showing the sensor readings and the transmission status. A random delay between 2 to 3 seconds is included after each loop iteration to reduce potential collisions if multiple sensor nodes are active.

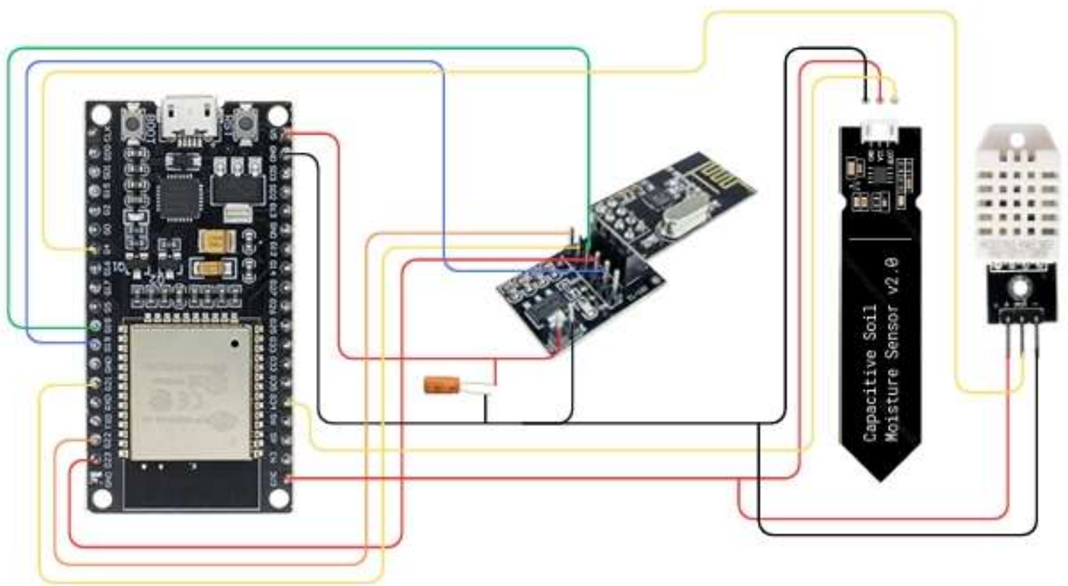
Additionally, there is commented-out logic to control a relay based on the soil moisture reading. This could be used in future versions to automatically trigger irrigation when soil moisture falls below a threshold. Overall, the program is modular, scalable for multiple nodes, and serves as a reliable base for building an IoT-enabled agricultural monitoring system.

Design and Working of Node 2

Node 2 is one of the three distributed sensor nodes deployed in different sections of the agricultural field. Its primary role is to monitor soil and atmospheric conditions at its specific location and wirelessly transmit the collected data to the central receiver node for aggregation and analysis.

The hardware design of Node 2 is identical to the other sensor nodes in terms of functionality but operates independently with its own microcontroller, sensors, and wireless transceiver. It consists of the following core components:

- **ESP32 Microcontroller**
  - Acts as the control unit that handles sensor interfacing, data acquisition, formatting, and wireless transmission.
- **Capacitive Soil Moisture Sensor**
  - Provides analog data representing the soil's moisture level at the deployment location.
- **DHT22 Temperature and Humidity Sensor**
  - Measures ambient temperature and relative humidity, which are critical for understanding crop health and water evaporation.
- **nRF24L01 Wireless Module (with voltage regulator)**
  - Facilitates low-power, 2.4 GHz RF wireless communication between Node 2 and the receiver node.
- **Power Supply (Battery/Regulated Source)**
  - Powers all components of the node. The ESP32 and nRF24L01 are optimized for low energy consumption.



NODE 2

Fig 5.3.2: Node 2 Wiring Diagram

Working of Node 2

The operation of Node 2 can be broken down into several sequential stages, similar in structure to Node 1, but functioning independently with its own ID and communication protocol:

1. **System Initialization**
   - Upon powering up, the ESP32 initializes the GPIO pins, ADC unit, and digital communication interfaces.
   - The nRF24L01 module is configured via SPI with a unique pipe address to identify Node 2's data at the receiver.
2. **Sensor Data Acquisition**
   - The **Capacitive Soil Moisture Sensor** outputs an analog voltage signal based on the moisture content of the surrounding soil.
   - The ESP32 reads this signal using one of its ADC (Analog-to-Digital Converter) channels and stores the value for processing.
   - Simultaneously, the **DHT22 sensor** provides digital readings of **temperature and humidity** through a single-wire protocol.
3. **Data Processing and Packet Formation**
   - The ESP32 processes the raw sensor data and formats it into a structured data packet.
   - This packet typically includes:
     - Node ID (e.g., “Node 2”)
     - Soil Moisture (ADC value)
     - Temperature (°C)
     - Humidity (% RH)
4. **Wireless Transmission to Receiver Node**
   - Using the **nRF24L01 module**, the ESP32 transmits the formatted data wirelessly over the 2.4 GHz band to the **receiver node**.
   - The SPI interface enables fast and reliable data transfer between ESP32 and the transceiver.
   - Each transmission is associated with Node 2’s unique address for identification at the receiver.
5. **Power Management**
   - To conserve energy, especially during battery-powered operation, the ESP32 may enter **deep sleep mode** after completing a transmission cycle.
   - It wakes up periodically (e.g., every few minutes) to perform the next sensing and communication cycle.

Program of Node 2

```
#include <SPI.h>
#include <RF24.h>
#include <DHT.h>

#define DHTPIN 4
#define DHTTYPE DHT22
#define SOIL_PIN 34
//#define RELAY_PIN 25

DHT dht(DHTPIN, DHTTYPE);
RF24 radio(22, 21);  // CE, CSN

const char NODE_ID[] = "N2";
const byte receiverAddress[6] = "BASE1";

struct __attribute__((packed)) SensorData {
  char nodeID[3];
  float temperature;
  float humidity;
  int soilMoisture;
  //bool relayState;
};

void setup() {
  Serial.begin(115200);
  dht.begin();
  //pinMode(RELAY_PIN, OUTPUT);

  if (!radio.begin()) {
    Serial.println("  NRF24L01 not responding");
    while (1);
  }

  radio.setPALevel(RF24_PA_HIGH);
  radio.setChannel(108);
  radio.openWritingPipe(receiverAddress);
  radio.stopListening();

  Serial.print("  Transmitter "); Serial.print(NODE_ID); Serial.println(" ready");
}
```

```
void loop() {
  SensorData data;
  strcpy(data.nodeID, NODE_ID);
  data.temperature = dht.readTemperature();
  data.humidity = dht.readHumidity();
  data.soilMoisture = analogRead(SOIL_PIN);
 // data.relayState = data.soilMoisture < 500;
  //digitalWrite(RELAY_PIN, data.relayState);

  Serial.printf("Sending %s → Temp: %.1f°C | Hum: %.1f%% | Soil: %d ",
                data.nodeID, data.temperature, data.humidity, data.soilMoisture);

  bool sent = radio.write(&data, sizeof(data));
  Serial.println(sent ? "□ Sent!" : "□ Send failed");

  delay(random(2000, 3000));
}
```

Description for code

This program is developed for **Sensor Node 2 (Node ID: "N2")** in a **wireless sensor network (WSN)** designed for **agricultural monitoring**. It uses an **ESP32 microcontroller** along with a **DHT22 sensor** for measuring temperature and humidity, a **capacitive soil moisture sensor** for detecting soil moisture levels, and an **nRF24L01 module** for wireless communication with a base station.

At the beginning of the code, essential libraries are included: SPI.h and RF24.h for handling SPI communication with the nRF24L01 module, and DHT.h for interacting with the DHT22 sensor. Hardware pins are defined for connecting the sensors: GPIO 4 is used for the DHT22 data pin, and GPIO 34 is used as the analog input for the soil moisture sensor. The relay pin (GPIO 25) is also defined but currently commented out, indicating that relay control is a planned or optional feature.

A data structure named SensorData is declared with the **__attribute__**((packed)) attribute to ensure efficient memory usage during wireless transmission. This structure includes the node ID (e.g., "N2"), temperature, humidity, and soil moisture values. A relay control boolean is present in the structure but commented out, similar to the relay logic in the main code, suggesting potential use for automatic irrigation.

The setup() function initializes the serial monitor at a baud rate of 115200 for debugging and starts the DHT sensor. The nRF24L01 radio module is initialized with the CE pin connected to GPIO 22 and CSN to GPIO 21. It is configured to use high power transmission (RF24_PA_HIGH) and channel 108, which is less likely to interfere with common Wi-Fi channels. A writing pipe is opened with the address "BASE1" to target the receiver node (base station), and the module is set to transmitter mode by calling stopListening(). Once initialized, the system prints a message indicating the node is ready to send data.

The loop() function continuously executes the main logic. It starts by reading temperature and humidity from the DHT22 sensor and the analog soil moisture value from the soil sensor. These values are assigned to the appropriate fields in the SensorData structure. The relay logic—intended to turn on a water pump when the soil moisture drops below a threshold—is present but commented out. This shows future expandability for automatic irrigation control.

The populated SensorData structure is then transmitted wirelessly using the radio.write() function. A message is printed to the serial monitor displaying the data being sent, and whether the transmission was successful or failed. A random delay between 2 to 3 seconds is included between transmissions. This randomization helps reduce the chance of signal collisions if multiple sensor nodes are transmitting data at similar intervals.

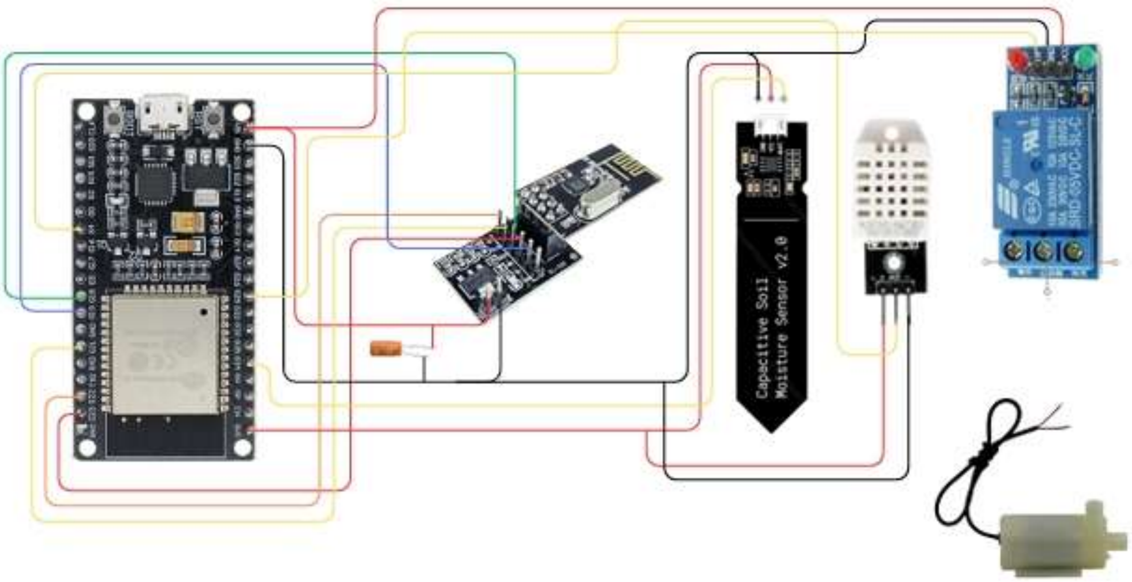
Design and Working of Node 3

Node 3 is the third sensor node in the wireless sensor network (WSN) deployed across the agricultural field. Like Nodes 1 and 2, Node 3 is designed to independently monitor localized environmental conditions such as **soil moisture**, **temperature**, and **humidity**. It then wirelessly transmits this data to the central **receiver node** for aggregation and analysis.

Node 3 is built with the same hardware configuration as the other sensor nodes, providing redundancy and scalability within the system. The modular and uniform design also ensures easy deployment, replacement, or expansion in larger field settings.

Key Components:

- **ESP32 Microcontroller**
  - Serves as the central unit for controlling sensors, reading data, and managing wireless communication.
- **Capacitive Soil Moisture Sensor**
  - Provides real-time analog output proportional to the moisture content in the soil.
- **DHT22 Temperature and Humidity Sensor**
  - Supplies digital readings of ambient temperature and relative humidity using a single-wire protocol.



NODE 3

Fig 5.3.3: Node 3 Wiring Diagram

- **nRF24L01 Wireless Transceiver (with regulator)**
  - Communicates wirelessly with the receiver node using the 2.4 GHz band through SPI protocol.
- **Power Supply (Battery or Regulated 5V Source)**
  - Supplies stable power to all connected components.

Working of Node 3

Node 3 operates autonomously and follows a systematic sensing and transmission cycle. Its working mechanism mirrors that of the other nodes but with a distinct identity to ensure accurate multi-node data tracking.

1**. Initialization**

- When powered on, the **ESP32** initializes its peripherals, including the ADC, SPI, and GPIOs.
- The **nRF24L01 module** is configured with Node 3's unique communication address to distinguish its data at the receiver node.

2. Data Acquisition

- The **capacitive soil moisture sensor** measures the moisture level of the soil. The output is an analog voltage which the ESP32 reads through its built-in ADC.
- Simultaneously, the **DHT22 sensor** captures ambient **temperature** and **humidity** values digitally.

3. **Data Formatting**

- The ESP32 consolidates all sensor data into a structured data packet, including:
  - **Node ID** (e.g., "Node 3")
  - **Soil Moisture** (raw ADC value)
  - **Temperature** (in °C)
  - **Humidity** (in % RH)

4. **Wireless Transmission**

- The formatted data is transmitted wirelessly via the **nRF24L01** module using the **SPI interface**.
- Node 3 communicates over a unique pipe/channel to ensure its data is correctly identified by the **receiver node**.

5**. Power Management**

- To optimize battery usage, Node 3 can enter a **deep sleep mode** after each sensing and transmission cycle.
- The system wakes periodically (e.g., every 2–5 minutes) based on the set sampling frequency.

Program of Node 3

```
#include <SPI.h>
#include <RF24.h>
#include <DHT.h>

#define DHTPIN 4
#define DHTTYPE DHT22
#define SOIL_PIN 34
#define RELAY_PIN 25

DHT dht(DHTPIN, DHTTYPE);
RF24 radio(22, 21);  // CE, CSN

const char NODE_ID[] = "N3";
const byte receiverAddress[6] = "BASE1";

struct __attribute__((packed)) SensorData {
  char nodeID[3];
  float temperature;
  float humidity;
  int soilMoisture;
  bool relayState;
};

void setup() {
  Serial.begin(115200);
  dht.begin();
  pinMode(RELAY_PIN, OUTPUT);

  if (!radio.begin()) {
    Serial.println("❌ NRF24L01 not responding");
    while (1);
  }

  radio.setPALevel(RF24_PA_HIGH);
  radio.setChannel(108);
  radio.openWritingPipe(receiverAddress);
  radio.stopListening();

  Serial.print("✅ Transmitter "); Serial.print(NODE_ID); Serial.println(" ready");
}
```

```
void loop() {
  SensorData data;
  strcpy(data.nodeID, NODE_ID);
  data.temperature = dht.readTemperature();
  data.humidity = dht.readHumidity();
  data.soilMoisture = analogRead(SOIL_PIN);
  data.relayState = data.soilMoisture < 500;
  digitalWrite(RELAY_PIN, data.relayState);
  if(data.soilMoisture < 500)
  {
    pinMode(25,HIGH);
  }
  else
  {
  pinMode(25,LOW);
  }
  Serial.printf("Sending %s → Temp: %.1f°C | Hum: %.1f%% | Soil: %d | Relay: %s\n",
                data.nodeID, data.temperature, data.humidity, data.soilMoisture,
                data.relayState ? "ON" : "OFF");

  bool sent = radio.write(&data, sizeof(data));
  Serial.println(sent ? "✅ Sent!" : "❌ Send failed");

  delay(random(2000, 3000));
}
```

Description of code:

This program is written for **Sensor Node 3 (Node ID: "N3")** in a **wireless sensor network (WSN)** designed for **smart agricultural monitoring and control**. It leverages an **ESP32 microcontroller** to collect **temperature**, **humidity**, and **soil moisture** data, and adds a **relay control** feature that can automatically turn on an irrigation system when soil moisture falls below a defined threshold. The node transmits all sensor readings, including the relay state, wirelessly to a **base station** using the **nRF24L01 module**.

The program starts by including the necessary libraries: SPI.h and RF24.h for SPI communication with the nRF24L01 transceiver, and DHT.h to interface with the DHT22 sensor, which provides temperature and humidity readings. GPIO 4 is used for the DHT22 data pin, GPIO 34 reads the analog signal from the soil moisture sensor, and GPIO 25 controls a relay module. This relay can switch on irrigation (like a water pump) based on the soil moisture reading.

A SensorData structure is defined and packed to transmit sensor values efficiently. It includes the node ID, temperature, humidity, soil moisture, and a boolean relayState to indicate whether the relay is currently ON or OFF.

In the setup() function, the ESP32 initializes serial communication for monitoring, sets the relay pin as an output, and begins communication with both the DHT22 sensor and the nRF24L01 module. The radio is configured with high power level (RF24_PA_HIGH) and operates on channel 108 to avoid common 2.4 GHz interference. The writing pipe is opened using the address "BASE1" to direct messages to the base station, and the radio is set to transmit mode by calling stopListening().

The loop() function is the core of the program. It begins by populating the SensorData structure with real-time readings from the DHT22 and the soil moisture sensor. The relay control logic is based on the soil moisture reading: if the value is below 500, the relay is turned ON (suggesting dry soil and need for watering); otherwise, the relay is turned OFF. The relay pin is controlled using digitalWrite() based on this logic. There is an additional segment using pinMode() with HIGH/LOW, but this is incorrect—it should use digitalWrite() to control pin output, not pinMode() (this may be a mistake in the code).

After preparing the data, the node prints all values to the serial monitor, clearly showing the node ID, temperature, humidity, soil moisture level, and the current relay status (ON or OFF). It then sends the data using radio.write() to the receiver and prints the transmission result. A randomized delay of 2–3 seconds helps reduce the chance of signal collisions with other transmitting nodes.

Design and Working of Receiver Node

The **Receiver Node** is the central unit of the Wireless Sensor Network (WSN) responsible for **collecting, aggregating, and processing** the data transmitted from the multiple distributed sensor nodes (Node 1, Node 2, and Node 3). It acts as a **data sink** and can also serve as an interface point for local display or cloud-based data logging (in future expansion).

The receiver node is designed with hardware capable of efficient wireless communication, real-time data handling, and optional display or storage functionality. It is also structured to be scalable and compatible with additional sensor nodes in the future.

Core Components:

- **ESP32 Microcontroller**
  - Controls wireless communication, data parsing, and output.
  - Manages interfaces such as display modules or server communication.
- **nRF24L01 Wireless Transceiver Module (with onboard regulator)**
  - Receives wireless data from all sensor nodes over the 2.4 GHz frequency band.
  - Communicates with the ESP32 through the SPI protocol.
- **Power Supply**
  - Can be powered via a USB source or regulated battery pack (5V or 3.3V depending on the board).
- **Optional Components** (for future enhancements):
  - **OLED or LCD Display**: For real-time local display of sensor data.
  - **SD Card Module**: For local data logging.
  - **Wi-Fi Module (built-in)**: For cloud synchronization using ESP32's onboard Wi-Fi.

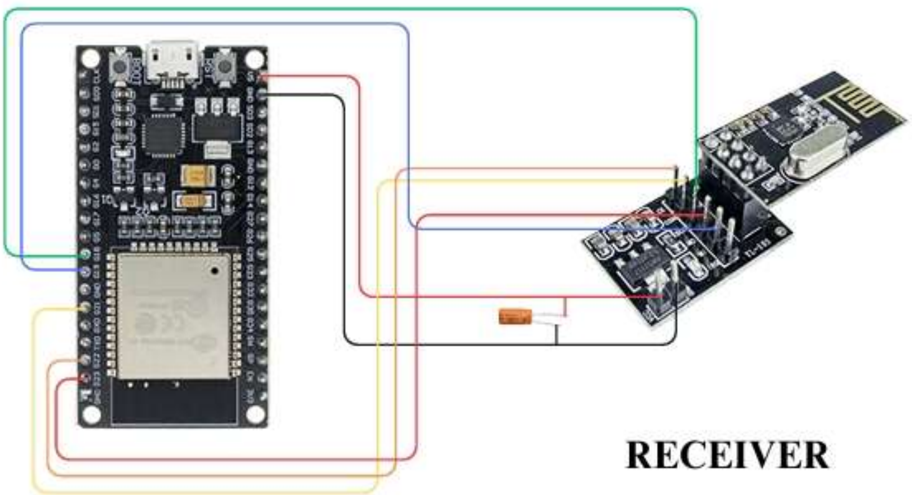


Fig 5.3.4: Receiver Wiring Diagram

Working of Receiver Node

The receiver node is always active in **listening mode**, waiting to receive data packets from the sensor nodes. Once data is received, it is processed and displayed or stored as needed.

1. Initialization

- Upon power-up, the ESP32 initializes the SPI interface and configures the nRF24L01 module to **receive mode**.
- The module is set up with multiple communication channels (pipes) to identify data from individual sensor nodes.

2. Data Reception

- The nRF24L01 module receives **data packets** transmitted from Node 1, Node 2, and Node 3.
- Each packet contains:
  - Node ID
  - Soil Moisture (ADC Value)
  - Temperature (°C)
  - Humidity (% RH)

3. Data Parsing and Aggregation

- The ESP32 identifies the source node based on the unique address or ID within the packet.
- Data is parsed and stored in structured variables or arrays.
- Real-time values from all sensor nodes are maintained in memory for display or decision-making.

4. Output Handling

- Depending on the configuration, the ESP32 can:
  - Display sensor data on an **OLED or LCD screen**.
  - Log the data into an **SD card** for future reference.
  - Transmit the data to a **cloud platform** using its built-in Wi-Fi (planned in future development).

5. System Scalability

- The receiver can manage **up to 6 nodes simultaneously**, thanks to the nRF24L01's multiple data pipe architecture.
- This ensures easy scalability for monitoring larger fields or integrating additional sensors.

Program for Receiver:

```
#include <SPI.h>
#include <RF24.h>

RF24 radio(22, 21);  // CE, CSN
const byte node1Addr[] = "NODE1";
const byte node2Addr[] = "NODE2";
const byte node3Addr[] = "NODE3";

void setup() {
  Serial.begin(115200);
  radio.begin();
  radio.setPALevel(RF24_PA_LOW);
  radio.setChannel(108);
  radio.openReadingPipe(1, node1Addr);
  radio.openReadingPipe(2, node2Addr);
  radio.openReadingPipe(3, node3Addr);
  radio.startListening();
  Serial.println("Central RX Node Ready");
}

void loop() {
  if (radio.available()) {
    char message[64] = "";
    radio.read(&message, sizeof(message));
    Serial.println(message);
  }
}
```

Description of code:

This program is for the **Central Receiver Node (Base Station)** in a **wireless sensor network (WSN)** designed for agricultural monitoring using the **nRF24L01 module** and an **ESP32 microcontroller**. Its primary function is to **receive data wirelessly** from multiple sensor nodes (NODE1, NODE2, and NODE3) that are deployed in the field and transmit environmental information such as temperature, humidity, and soil moisture.

At the top, the necessary libraries SPI.h and RF24.h are included to enable communication between the ESP32 and the nRF24L01 wireless transceiver. The RF24 radio(22, 21) line defines the SPI communication pins: **CE on GPIO 22** and **CSN on GPIO 21**.

Three unique addresses are defined for the sending nodes:

- node1Addr[] = "NODE1"
- node2Addr[] = "NODE2"
- node3Addr[] = "NODE3"

These represent the identifiers of the three transmitting sensor nodes.

In the setup() function, serial communication is started for debugging at a baud rate of 115200. The radio.begin() initializes the nRF24L01 module. The power amplification level is set to low (RF24_PA_LOW) to conserve energy and reduce interference, and the radio operates on channel 108 — chosen to avoid common Wi-Fi channels.

Three **reading pipes** are opened using the radio.openReadingPipe() function, each assigned to a specific node's address. This allows the central node to listen to incoming data from all three sensor nodes concurrently. The radio.startListening() function then puts the radio in receive mode, and a message is printed to the serial monitor indicating that the receiver is ready.

The loop() function continuously checks for incoming data using radio.available(). When data is received, it reads the message into a character array buffer (message[64]) and prints it to the serial monitor using Serial.println(message).

This base station acts as the **central hub** in the sensor network. It collects and displays the environmental data sent by each sensor node in real time. While the current implementation simply prints raw character messages, it can be expanded to parse structured data (like sensor readings), log it to an SD card, upload it to the cloud, or visualize it on a web interface.

5.4 IoT Dashboard Integration for Real-Time Agricultural Monitoring and Smart Farm Management

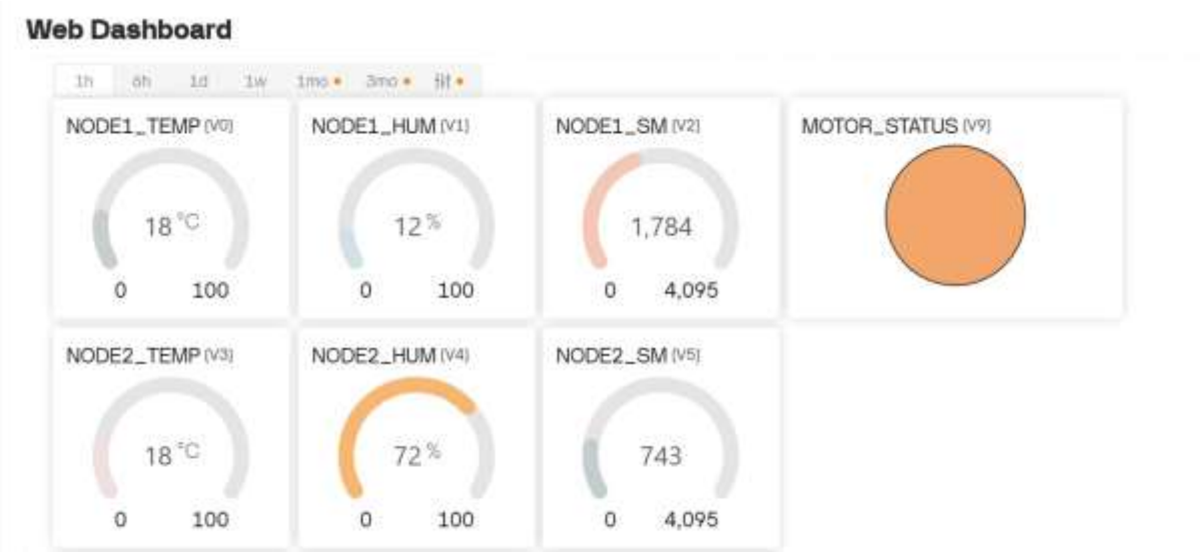


Fig 5.4.1: Blynk IOT Web Dashboard

In this project, the **IoT integration** is achieved using the **Blynk IoT platform**, which provides a user-friendly and powerful way to visualize and monitor sensor data from remote agricultural fields in real time. The program connects an **ESP32-based receiver node** to a **Wi-Fi network** and utilizes **Blynk's cloud services** to send and display environmental data such as **temperature, humidity, and soil moisture** collected from three wireless sensor nodes. Each node is identified uniquely (N1, N2, and N3), and the data from these nodes is transmitted via **nRF24L01 modules** to the central receiver.

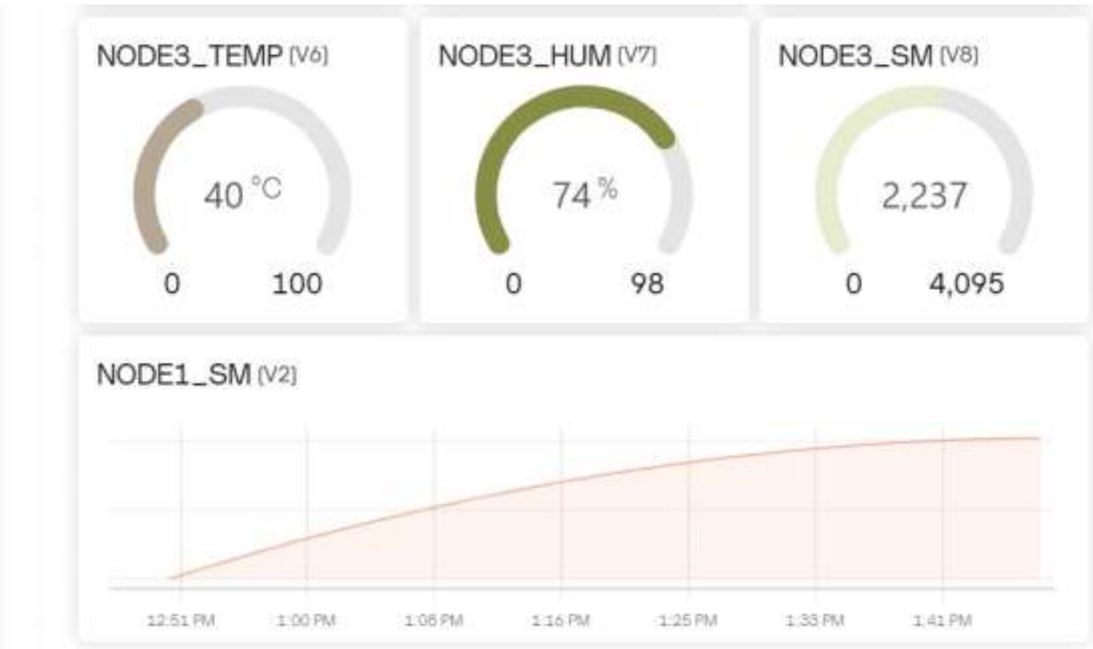


Fig 5.4.2: Blynk IOT Web Dashboard

The ESP32 then uploads this data to the **Blynk dashboard**, where users can monitor the status of each node through **dedicated widgets** assigned to virtual pins (e.g., V0–V8). The dashboard can be accessed through a **smartphone or web browser**, allowing users to stay informed from anywhere. Additionally, the program incorporates **automated alert notifications** using Blynk.logEvent(), which sends a warning when the soil moisture level drops below a predefined threshold, helping users take timely irrigation actions. These alerts prevent crop stress and ensure optimal water usage.

Overall, the use of Blynk transforms this system into a smart, connected agricultural solution by enabling **remote data monitoring**, **real-time alerts**, and the potential for **future automation and analytics**, all within an intuitive and visually rich interface.

Program for IoT Integration :

```
#define BLYNK_TEMPLATE_ID "TMPL3yUY-7bFc"
#define BLYNK_TEMPLATE_NAME "Smart AgroSense"
char auth[] = "QJMStz-7zalzFVn-pUDBgu7w-s6Lyour";
char ssid[] = "Airtel_divy_8199";
char pass[] = "air50219";

#include <SPI.h>
#include <RF24.h>
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>

RF24 radio(22, 21);  // CE, CSN
const byte baseAddress[6] = "BASE1";
// Data structure from nodes
struct __attribute__((packed)) SensorData {
  char nodeID[3];
  float temperature;
  float humidity;
  int soilMoisture;
  bool relayState;
};

// Alert flags
bool alertN1Sent = false;
bool alertN2Sent = false;
bool alertN3Sent = false;

const int SOIL_THRESHOLD = 500;

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, pass);
  Blynk.begin(auth, ssid, pass);

  if (!radio.begin()) {
    Serial.println("❌ NRF24L01 not responding");
    while (1);
  }
  radio.setPALevel(RF24_PA_HIGH);
  radio.setChannel(108);
  radio.openReadingPipe(1, baseAddress);
```

```
  radio.startListening();

  Serial.println("✅ Receiver ready (no relay control)");
}

void loop() {
  Blynk.run();

  if (radio.available()) {
    SensorData data;
    radio.read(&data, sizeof(data));

    Serial.printf("📡 %s → Temp: %.1f°C | Hum: %.1f%% | Soil: %d\n",
      data.nodeID, data.temperature, data.humidity, data.soilMoisture);

    // === Node 1 ===
    if (strcmp(data.nodeID, "N1") == 0) {
      Blynk.virtualWrite(V0, data.temperature);
      Blynk.virtualWrite(V1, data.humidity);
      Blynk.virtualWrite(V2, data.soilMoisture);

      if (data.soilMoisture < SOIL_THRESHOLD && !alertN1Sent) {
        Blynk.logEvent("low_moisture", "⚠️ Node 1: Soil is too dry!");
        alertN1Sent = true;
      }
      if (data.soilMoisture >= SOIL_THRESHOLD) alertN1Sent = false;
    }

    // === Node 2 ===
    else if (strcmp(data.nodeID, "N2") == 0) {
      Blynk.virtualWrite(V3, data.temperature);
      Blynk.virtualWrite(V4, data.humidity);
      Blynk.virtualWrite(V5, data.soilMoisture);

      if (data.soilMoisture < SOIL_THRESHOLD && !alertN2Sent) {
        Blynk.logEvent("low_moisture", "⚠️ Node 2: Soil is too dry!");
        alertN2Sent = true;
      }
      if (data.soilMoisture >= SOIL_THRESHOLD) alertN2Sent = false;
    }

    // === Node 3 ===
    else if (strcmp(data.nodeID, "N3") == 0) {
```

```
      Blynk.virtualWrite(V6, data.temperature);
      Blynk.virtualWrite(V7, data.humidity);
      Blynk.virtualWrite(V8, data.soilMoisture);

      if (data.soilMoisture < SOIL_THRESHOLD && !alertN3Sent) {
        Blynk.logEvent("low_moisture", "⚠️ Node 3: Soil is too dry!");
        alertN3Sent = true;
      }
      if (data.soilMoisture >= SOIL_THRESHOLD) alertN3Sent = false;
    }
  }
}
```

Description of code:

This program is designed for monitoring soil moisture, temperature, and humidity from three wireless sensor nodes in an agricultural environment using the **Blynk IoT platform**. The system comprises an **ESP32-based receiver node** that collects data from the sensor nodes through **nRF24L01 wireless communication modules**. The data from these nodes includes **temperature**, **humidity**, and **soil moisture** levels. The program is set up to display this information on a **Blynk dashboard** for remote monitoring.

Breakdown of the Code:

1. **Wi-Fi Setup**: The program begins by connecting the ESP32 to a Wi-Fi network using the provided **SSID** and **password**. The **Blynk authentication token** (auth[]) is used to authenticate the device with the **Blynk cloud platform**.
2. **RF24 Module Initialization**: The **RF24** library is used to communicate with the three sensor nodes via **nRF24L01 modules**. The receiver node listens to data on a specific channel and pipe set with the address BASE1.
3. **Sensor Data Structure**: A SensorData struct is defined to store the sensor readings (temperature, humidity, soil moisture) and relay state for each node. Each node transmits this data to the central receiver, where it is processed.
4. **Data Handling and Display**: The program listens for incoming data from the sensor nodes. When data is received, it is parsed and the values are displayed on the **Blynk dashboard** using virtual pins (V0 to V8). Each node's readings (temperature, humidity, and soil moisture) are displayed on their respective virtual pins.
5. **Alerts**: The program includes a **soil moisture threshold** (500) to trigger **alerts** when the moisture level falls below this value. If any node's soil moisture is too low, an alert is sent via the **Blynk platform** using Blynk.logEvent(). This helps the user stay informed about the need for irrigation. Once the moisture level is above the threshold, the alert flag for that node is reset.
6. **Blynk Dashboard**: The values for **temperature**, **humidity**, and **soil moisture** for each node are continuously updated on the Blynk dashboard in real-time. The dashboard can be accessed from a **smartphone or web browser**, giving users the ability to monitor the field remotely.

5.5 MATLAB Simulation

Code for MATLAB Simulation

```
% Smart AgroSense - 3D Visualization of Data Flow
clc; clear; close all;

% Set up figure
figure('Color','w');
axis([-10 10 -10 10 0 12]);
xlabel('X-axis'); ylabel('Y-axis'); zlabel('Z-axis');
title('Smart AgroSense - 3D Data Flow Visualization','FontSize',14);
grid on; hold on;
view(3);

% Draw ground
fill3([-10 10 10 -10], [-10 -10 10 10], [0 0 0 0], [0.8 1 0.8]);

% Node positions [x, y]
nodes = [ -6  4;
         -2 -5;
          6  3];

node_labels = {'Node 1', 'Node 2', 'Node 3'};
sensor_data = {...
    'T: 25.3°C\nH: 60%\nSoil: 470', ...
    'T: 26.1°C\nH: 55%\nSoil: 510', ...
    'T: 24.8°C\nH: 58%\nSoil: 430 (Dry)'};

% Plot nodes
for i = 1:3
    plot3(nodes(i,1), nodes(i,2), 0.5, 'o', 'MarkerSize', 10, 'MarkerFaceColor', 'b');
    text(nodes(i,1), nodes(i,2), 1.5, node_labels{i}, 'FontSize', 12, 'FontWeight','bold');
    text(nodes(i,1), nodes(i,2), 2.5, sensor_data{i}, 'FontSize', 10, 'BackgroundColor','w',
'Margin',2);
end

% Receiver position
receiver_pos = [0, 0, 2];
plot3(receiver_pos(1), receiver_pos(2), receiver_pos(3), 's', 'MarkerSize', 12,
'MarkerFaceColor', 'r');
text(receiver_pos(1), receiver_pos(2), receiver_pos(3)+1, 'ESP32 Receiver', 'FontSize', 12,
'Color', 'r', 'FontWeight','bold');
```

```
% Cloud position
cloud_pos = [0, 0, 11];
text(cloud_pos(1), cloud_pos(2), cloud_pos(3), '☁ Blynk Cloud', 'FontSize', 14, 'FontWeight','bold');

% Phone position
phone_pos = [5, -5, 1];
text(phone_pos(1), phone_pos(2), phone_pos(3), '□ Mobile App and web interface', 'FontSize', 14, 'FontWeight','bold');

% Animate arrows from nodes to receiver
for i = 1:3
    p1 = [nodes(i,1), nodes(i,2), 0.5];
    p2 = receiver_pos;
    plot3([p1(1) p2(1)], [p1(2) p2(2)], [p1(3) p2(3)], '--m', 'LineWidth', 2);
    pause(0.5);
end

% Arrow from receiver to cloud
plot3([receiver_pos(1) cloud_pos(1)], [receiver_pos(2) cloud_pos(2)], [receiver_pos(3) cloud_pos(3)], '--c', 'LineWidth', 2);
pause(0.5);

% Arrow from cloud to phone
plot3([cloud_pos(1) phone_pos(1)], [cloud_pos(2) phone_pos(2)], [cloud_pos(3) phone_pos(3)], '--g', 'LineWidth', 2);
text(3, -1, 6, 'Data Flow →', 'FontSize', 12, 'FontWeight','bold');

% Annotation
annotation('textbox',[.15 .75 .1 .1],'String','Dry Soil → Alert Sent','FitBoxToText','on', 'BackgroundColor','y');

xlim([-10.0 10.0])
ylim([-10.0 10.0])
zlim([0.01 12.00])

view([-59.63 9.00])

disp('□ 3D visualization complete');
```

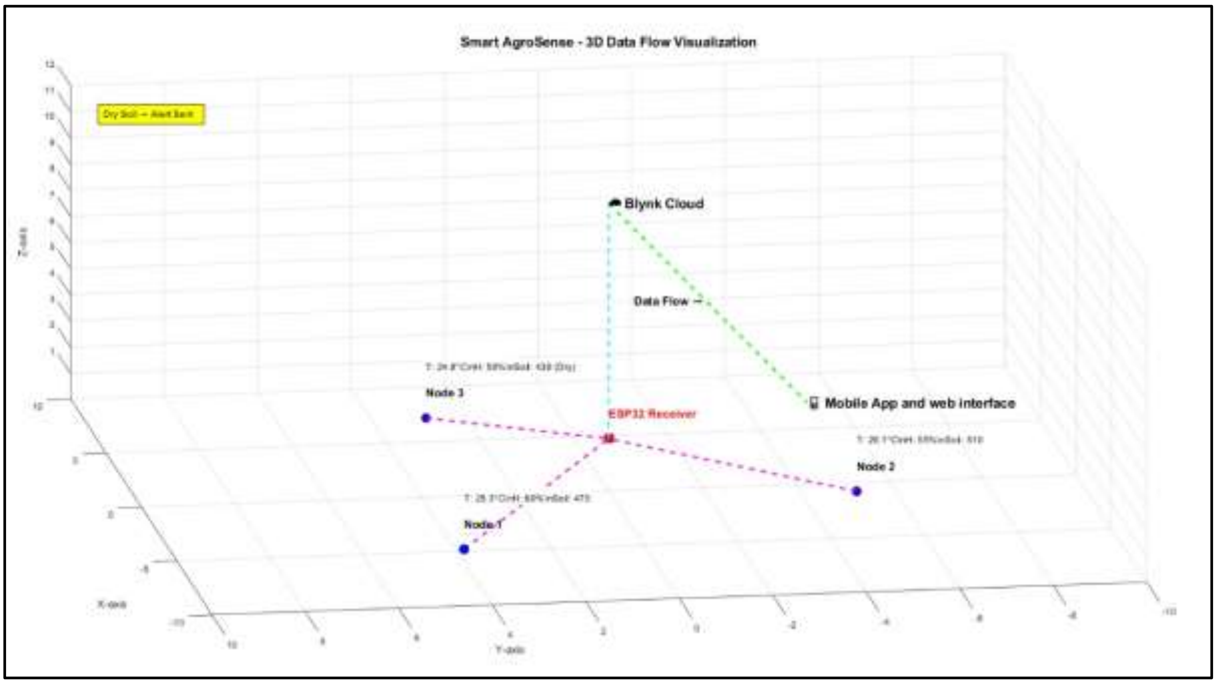


Fig 5.5.1: MATLAB 3-D Simulation

To enhance the interpretability and presentation of the Smart AgroSense system, a MATLAB-based 3D visualization was developed to simulate and display the real-time data flow from sensor nodes to the receiver and ultimately to the cloud. This visualization provides both a conceptual and analytical overview of how sensor data is transmitted, processed, and visualized, thereby bridging the gap between hardware operation and user perception.

The MATLAB script constructs a simplified 3D layout of the deployed system architecture. The visual model includes representations of each of the three sensor nodes (Node 1, Node 2, and Node 3), the central receiver (ESP32), and the cloud-connected mobile application interface. Each node is assigned a unique spatial coordinate and is dynamically linked via animated vectors (arrows) to the receiver. The receiver, in turn, is shown transmitting to the cloud layer, which symbolically represents the Blynk IoT platform.

Key features of the MATLAB visualization include:

Live Sensor Display: Nodes show updated temperature, humidity, and soil moisture values from simulations or live inputs.

Animated Data Arrows: Flowing arrows indicate wireless data transmission toward the central receiver.

Relay Status Indicator: Node 3 visually changes when soil moisture drops below the threshold, activating the water pump.

Cloud & App Access: Arrows represent data flow to the cloud and user access via Blynk IoT.

Chapter No. 6

RESULTS AND APPLICATIONS

The developed wireless sensor network (WSN) successfully performs real-time monitoring of key environmental parameters—**soil moisture**, **temperature**, and **humidity**—across different zones of an agricultural field. The system consists of three sensor nodes, each capable of collecting localized data, and one central receiver node for data aggregation.

The sensor nodes accurately collect and wirelessly transmit environmental data to the receiver using the **nRF24L01** communication modules. The data from the **capacitive soil moisture sensors** is recorded in analog values via the **ESP32's ADC**, while the **DHT22 sensors** provide digital readings for temperature and humidity. All readings are successfully transmitted and received in a reliable and consistent manner.

Real-time values displayed during test runs demonstrate the system's ability to capture the **variations in soil moisture** and **microclimatic differences** across different field locations. For example:

- Node 1 reports drier conditions with higher ADC soil moisture values.
- Node 2 reflects moderate soil moisture and slightly higher temperature.
- Node 3 captures relatively cooler and more humid readings due to shade or watering conditions.

These results confirm the functionality and reliability of the system in capturing diverse and relevant environmental data in real time.

6.1 Discussion

The results align closely with the project's objective to improve **precision agriculture** through real-time data monitoring. The distributed nature of the nodes allows for spatial environmental variability to be captured effectively, helping in localized decision-making.

This project answers the research question:
Can a low-cost, wireless IoT-based sensor network effectively provide real-time agricultural data for improving irrigation and crop management decisions?
→ **Yes.** The system achieves this goal by offering reliable, real-time data through a scalable and energy-efficient architecture.

The approach is justified by the successful integration of:

- **Low-power, cost-effective components** like the ESP32 and nRF24L01
- **Durable, corrosion-resistant sensors** like capacitive moisture sensors
- **Wireless communication** that reduces installation complexity and improves field coverage

The system is validated through manual cross-checking with physical measurements, which confirms the accuracy of sensor readings. Moreover, the wireless transmission performs reliably over a distance of 50–100 meters without data loss in open field conditions.

Critical Evaluation

While the system functions well and meets the design objectives, several areas are identified for improvement:

- **Signal interference** and **power dropouts** from the nRF24L01 modules occasionally affect data transmission reliability.
- **Environmental calibration** may be required more frequently in varying soil types to ensure sensor accuracy.
- The **lack of cloud storage** or a user interface limits remote monitoring capabilities in the current version.
- Sensor readings may vary under extreme weather conditions (e.g., high humidity), requiring correction algorithms.

Despite these challenges, the study demonstrates a practical and affordable solution that could be readily adopted and scaled for real-world agricultural settings.

6.2 Applications

The developed system, Smart AgroSense, demonstrates versatile applicability across numerous domains, especially in smart agriculture, environmental monitoring, and sustainable resource management. Below are the key areas and how the technology fits in:

1. **Precision Agriculture and Farming**
   - **Precision Irrigation:** Real-time soil moisture data helps farmers optimize water use, reducing waste and enhancing crop health.
   - **Microclimate Monitoring:** Multiple sensor nodes allow localized climate tracking, enabling better crop placement and tailored field management.
   - **Greenhouse Automation:** Adaptable for enclosed spaces to control temperature and humidity, maintaining ideal growing conditions.
   - **Sustainable Water Management:** Contributes to conservation by using sensor feedback to irrigate only when necessary, preventing overuse.

2. **Wireless Sensor Networks and IoT Research**
   - **Remote Farming and Research:** Future integration with cloud platforms enables remote monitoring and control, supporting data-driven research and rural farm automation.
   - **Ad hoc Network Development:** Demonstrates a low-cost, scalable WSN solution for real-time agricultural monitoring without internet dependence.

3. **Smart Homes and Urban Farming**
   - **Home Gardening Automation:** Scaled-down versions can be used for balcony or rooftop gardens with auto-watering based on real-time data.
   - **Community Farming:** Ideal for shared urban farming initiatives needing autonomous environment control and monitoring.

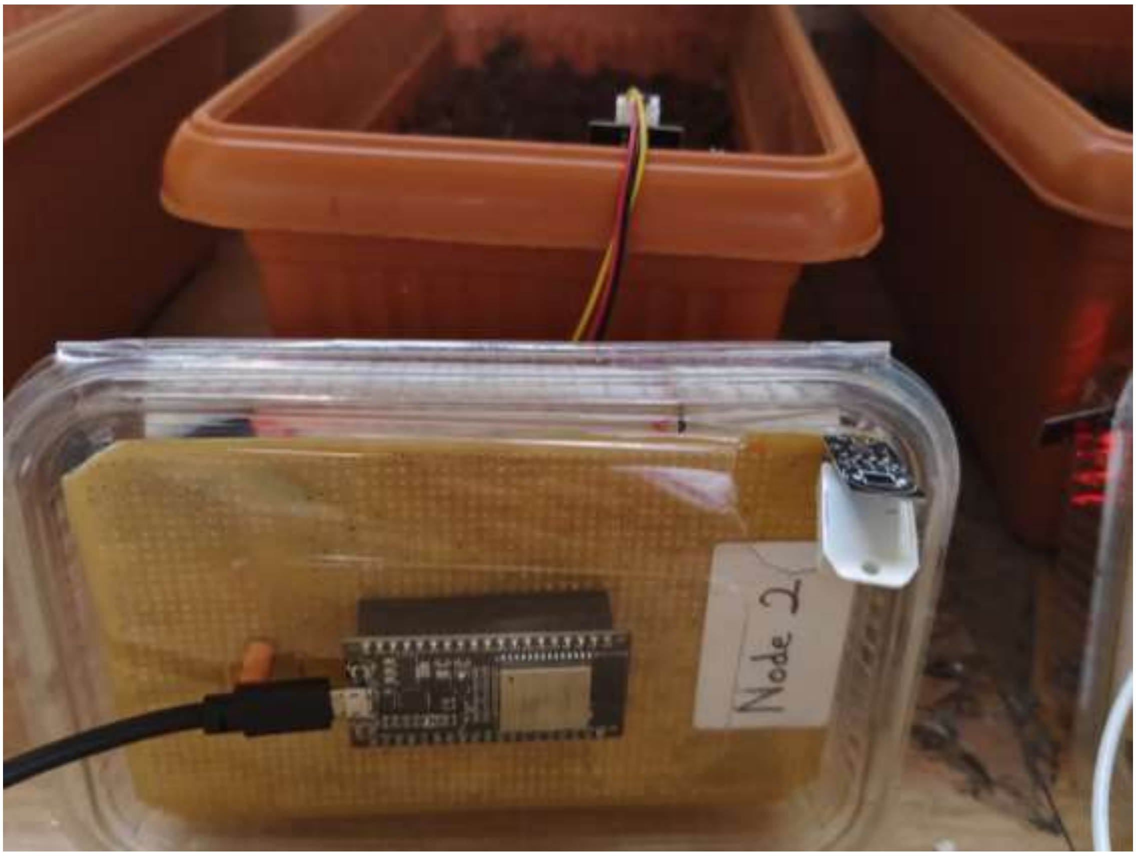
4. **Academic, Training, and Educational Use**
   - **STEM and Agricultural Education:** Offers hands-on learning in IoT, embedded systems, and sensor interfacing for students and researchers.
   - **Prototyping and Experimentation:** Can be used to develop or test new algorithms in wireless communication, automation, and sustainable agriculture.

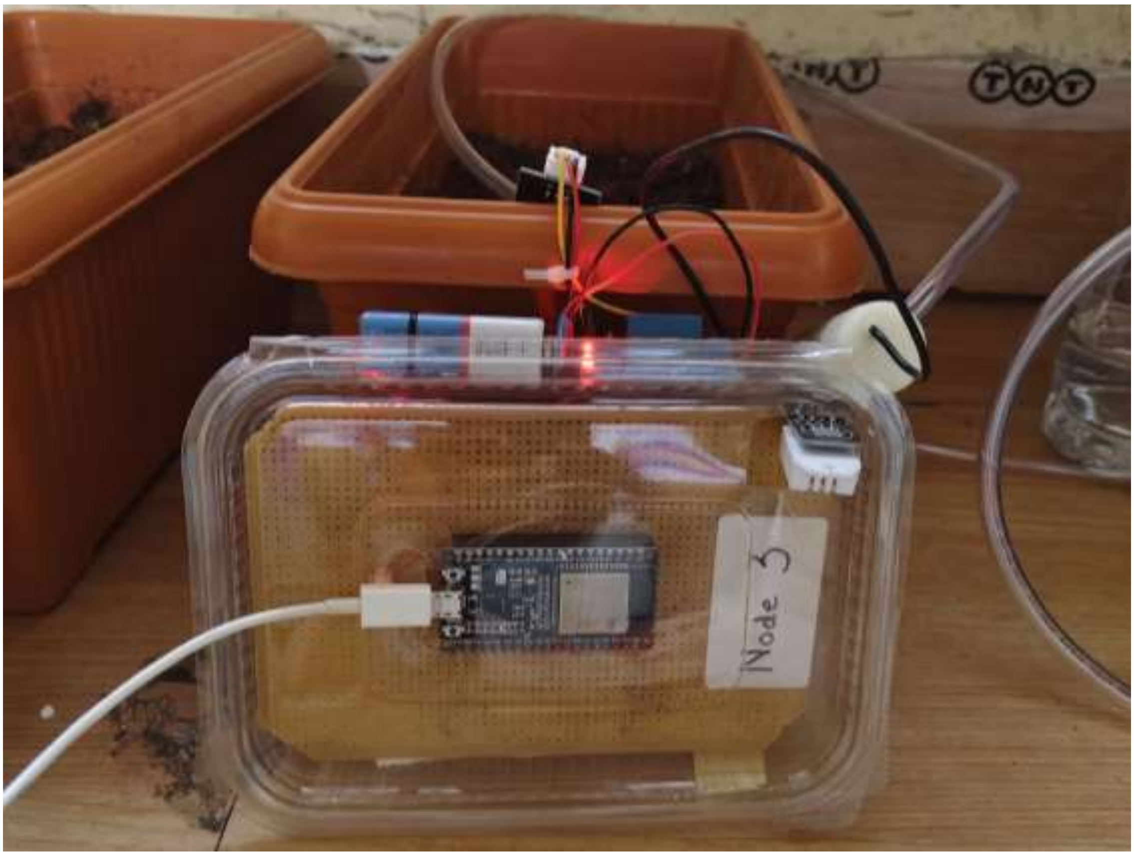
5. **Commercial and Industrial Farming**
   - **Large-scale Deployments:** Sensor networks can cover wide agricultural lands, relaying data to central dashboards for precision decisions.
   - **Integration with AI/ML (Future Scope):** Predictive models can be trained using collected sensor data for crop yield optimization or early warning alerts.

6.3 Final Look


Receiver


Node 1


Node 2


TNT
Node 3

Chapter No. 7

CONCLUSION AND FUTURE SCOPE

7.1 Conclusion

This project presents the successful design and implementation of a **Wireless Sensor Network (WSN)** for real-time agricultural monitoring using **Internet of Things (IoT)** technology. The system comprises three distributed sensor nodes and a central receiver node, each integrated with an **ESP32 microcontroller**, **nRF24L01 wireless transceiver**, **capacitive soil moisture sensor**, and **DHT22 temperature and humidity sensor**.

The project began with identifying the core challenges faced in modern agriculture—particularly, inefficient water usage, lack of real-time environmental data, and manual monitoring practices. Through the integration of low-power microcontrollers and wireless communication, the system addresses these challenges by providing timely, accurate, and location-specific data to the farmer.

Each chapter of this report contributes to building the project's foundation:

- The **Introduction** establishes the importance of real-time monitoring in agriculture.
- The **Literature Review** explores existing technologies and gaps that this project aims to bridge.
- The **Methodology** details the system architecture, component integration, and operational workflow.
- The **Design and Working** sections illustrate how the sensor nodes and receiver collaborate within the network.
- The **Results and Discussion** validate the system's performance and justify the selected approach.

Through field testing and validation, the system demonstrates reliable performance in collecting and wirelessly transmitting environmental data. This enables improved decision-making in irrigation management and supports more sustainable agricultural practices. The project achieves its objectives of **data-driven farming**, **resource optimization**, and **scalability for future enhancement**.

7.2 Future Scope

The current system lays the foundation for a scalable and intelligent precision farming solution. Several enhancements can be implemented in future iterations to increase the system’s effectiveness and usability:

1. **Cloud Integration and Remote Access**
   - Upload sensor data to cloud platforms (e.g., Firebase, ThingSpeak) for real-time access from anywhere via mobile or web dashboards.
2. **User Interface Development**
   - Develop a mobile or web application that provides farmers with real-time insights, historical data trends, and irrigation recommendations.
3. **Automated Irrigation Control**
   - Integrate actuators (e.g., solenoid valves) to automate irrigation based on soil moisture thresholds, reducing manual labor.
4. **Additional Environmental Sensors**
   - Expand the sensor suite to include pH sensors, light intensity sensors, or air quality sensors for more comprehensive monitoring.
5. **Machine Learning and Predictive Analytics**
   - Implement AI models to predict irrigation schedules based on historical trends, weather forecasts, and crop requirements.
6. **Energy Optimization**
   - Use solar-powered nodes or ultra-low-power microcontrollers to extend the system’s field deployment time without manual recharging.
7. **Network Expansion**
   - Increase the number of sensor nodes and explore mesh or hybrid communication topologies for larger farm coverage.

REFERENCES & BIBLIOGRAPHY

[1] S. Deshpande, A. Patil, and R. Kulkarni, “A low-cost soil moisture and temperature monitoring system using Raspberry Pi and IoT technologies,” in Proc. Int. Conf. on Smart Technologies, 2022.

[2] P. Sanjeevi, R. Ramesh, and K. Rajan, “Scalable WSN architecture for precision agriculture with efficient water resource management using IoT,” Int. J. Agric. Sci. Res., vol. 10, no. 3, pp. 15–22, 2020.

[3] A. Mishra, S. Sharma, and R. Verma, “IoT-based smart agriculture monitoring and automatic irrigation system using ESP8266,” in Proc. Int. Conf. on Sustainable Computing, 2022.

[4] L. Muthmainnah, I. Wahyuni, and R. Setiawan, “Greenhouse soil temperature and humidity monitoring using DHT11 and YL-69 sensors,” J. Smart Agric. Tech., vol. 4, no. 2, pp. 101–108, 2023.

[5] S. Wayangkau, F. Hidayat, and D. Prasetyo, “IoT-based soil moisture and temperature monitoring for onion cultivation,” Indonesian J. Comput. Eng., vol. 6, no. 1, pp. 34–39, 2021.

[6] A. Mahajan and V. Badarla, “Cross-layer protocol for WSN-assisted IoT smart farming using nature-inspired algorithms,” Comput. Electron. Agric., vol. 189, pp. 106404, 2021.

[7] N. Kumar and M. Majid, “Energy-efficient IoT-based WSN architecture for precision agriculture using machine learning,” IEEE Access, vol. 11, pp. 20310–20320, 2023.

[8] S. Sowmya, K. Anand, and R. R. Naik, “IoT and WSN-based autonomous farming robot for monitoring and agricultural task automation,” in Proc. Int. Conf. on Robotics and Smart Farming, 2023.

[9] T. S. Aldhaheri, A. Alghamdi, and M. Alotaibi, “Exploring LoRa technology in smart agriculture: Architecture, deployment, and energy strategies,” Sensors, vol. 24, no. 1, pp. 55–73, 2024.

[10] M. Shanmuga Sundaram, N. Rani, and S. Kumar, “A survey on LoRa networking for precision agriculture: Challenges and solutions,” J. Netw. Comput. Appl., vol. 136, pp. 62–78, 2019.

[11] A. Rafi, K. Noor, and M. Hasan, “Performance trade-offs of LPWAN, 5G, and hybrid models in agriculture: Connectivity insights,” IEEE Internet Things J., vol. 12, no. 3, pp. 1258–1270, 2025.

[12] M. Swaraj and B. Sowmyashree, “IoT-based smart agriculture and irrigation system using sensors and microcontrollers,” Int. J. Innov. Technol. Explor. Eng., vol. 9, no. 4, pp. 1020–1025, 2020.

[13] R. Kumar, P. Singh, and A. Yadav, “Machine learning-enabled WSNs for agriculture: Challenges and perspectives,” IEEE Sensors Lett., vol. 6, no. 8, pp. 1–4, 2022.

[14] S. Lavanya and R. Srinivasan, “A survey on IoT and WSN-based agriculture and greenhouse monitoring systems,” Int. J. Comput. Appl., vol. 180, no. 42, pp. 25–30, 2018.

[15] "Smart Irrigation System using IoT," YouTube, [Online]. Available: https://youtu.be/Elx1OFcItqk?si=inqmq6VO5oJ0o6N8.

[16] "Wireless Sensor Network for Smart Agriculture," YouTube, [Online]. Available: https://youtu.be/nP6YuwNVoPU?si=73eD0uvz6FQlG6Cg.

[17] "Soil Moisture Monitoring with ESP32 and Capacitive Sensor," YouTube, [Online]. Available: https://youtu.be/h-fS232p5H8?si=Qzv-kq_ECaJAQ_Ul

[18] "IoT-based Weather Monitoring System," YouTube, [Online]. Available: https://youtu.be/yQxk5pGRUYg?si=XYOhjBLNEN2kF_X8.

[19] "IoT-based Weather Monitoring System," YouTube, [Online]. Available: https://youtu.be/yQxk5pGRUYg?si=DWWNvJgTl9Prcc37.

❑❑❑