ABSTRACT

In today's fast-paced world, efficient resource management is vital to ensuring sustainability and improving agricultural productivity. This project introduces an IoT-based Smart Irrigation System utilizing Arduino, aimed at providing an efficient and user-friendly solution for real-time water management. The system leverages soil moisture sensors to accurately monitor soil conditions and optimizes irrigation by automatically delivering water when needed. Integrated with a mobile application, it sends instant notifications if the soil moisture falls below or exceeds predefined thresholds, alerting users to take corrective action.

Unlike traditional irrigation methods, which often lack precision and waste resources, our system addresses this gap with dedicated sensors and intelligent data processing. The mobile app enhances usability by maintaining a detailed history of soil moisture and irrigation activity, enabling users and agricultural professionals to assess trends and make data-driven decisions. This feature is particularly beneficial for farmers, gardeners, or greenhouse operators, where efficient water usage is critical.

The proposed system is compact, cost-effective, and scalable, making it suitable for small-scale gardens as well as large agricultural fields. By combining IoT technology with proactive water management, this project aspires to provide a significant contribution to sustainable agriculture, promoting resource conservation and improved crop yield.

CONTENTS

INTRODUCTION

In recent years, advancements in the Internet of Things (IoT) have enabled significant progress in agriculture and resource management, offering solutions that enhance efficiency and sustainability. Water is a critical resource in agriculture, and its efficient use is essential for maximizing crop yield while minimizing waste. Traditional irrigation methods often lead to overwatering or underwatering, which can harm plants and waste valuable resources. This project presents an IoT-based Smart Irrigation System that addresses these limitations by providing a reliable, real-time solution for water management.

The proposed system utilizes Arduino as the core processing unit and integrates high-precision soil moisture sensors to monitor soil conditions continuously. The data is transmitted wirelessly to a mobile application, ensuring seamless connectivity and user convenience. One of the system's distinctive features is its ability to automate watering based on real-time soil moisture levels and predefined thresholds. Notifications are generated when the soil requires watering or if abnormal conditions are detected, allowing users to take timely action.

Unlike traditional irrigation systems that operate on fixed schedules regardless of environmental conditions, this system offers enhanced accuracy and adaptability. The mobile application also serves as a repository for historical soil moisture and irrigation data, enabling users and agricultural professionals to analyze trends and optimize irrigation strategies over time.

The project emphasizes cost-effectiveness, scalability, and ease of use, making it a viable solution for farmers, gardeners, and greenhouse operators. By combining IoT technology with real-time environmental monitoring, this system has the potential to revolutionize irrigation practices, empowering individuals and communities to conserve water and improve agricultural productivity proactively.

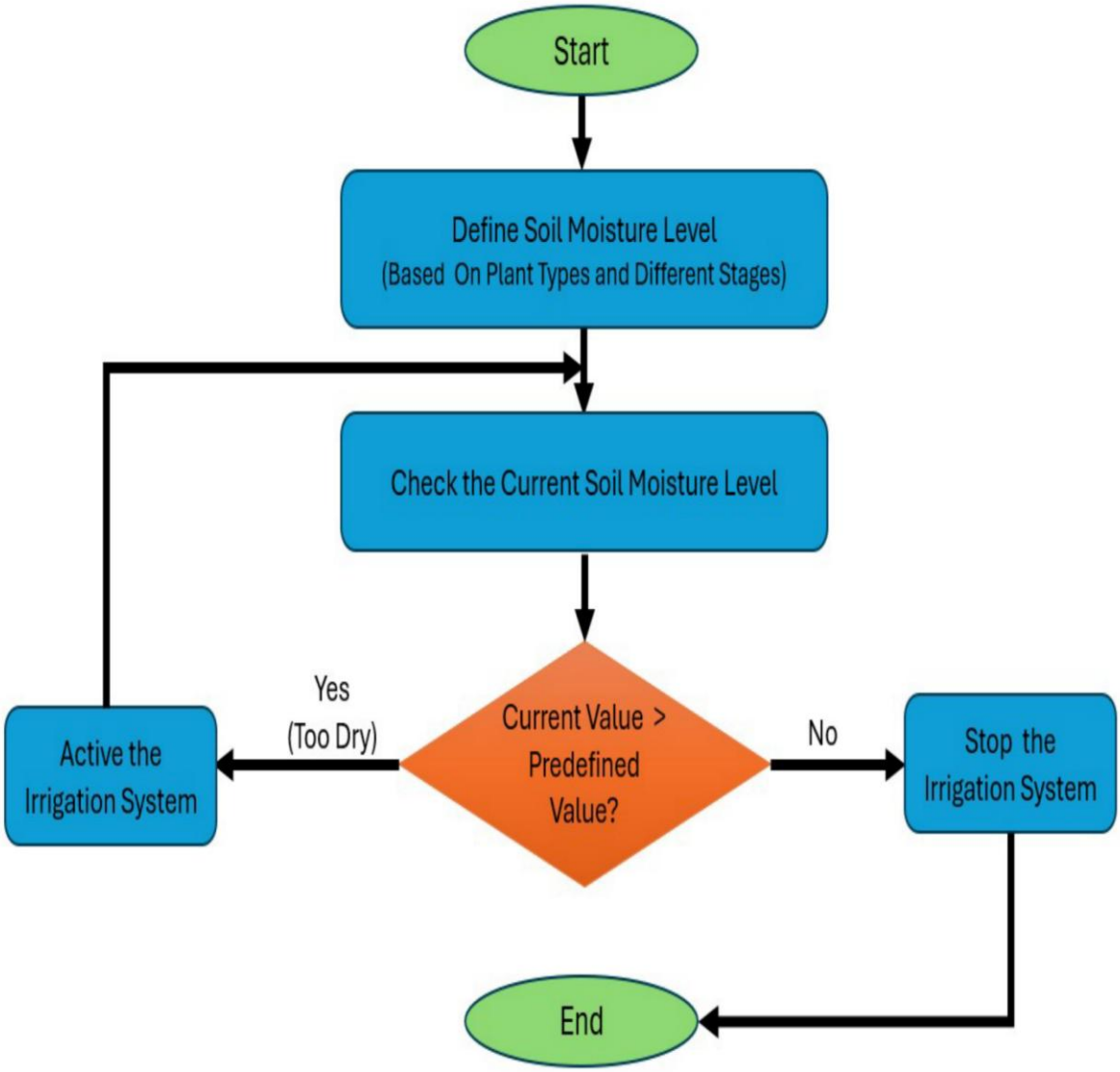


Fig 1: Working flow of Smart Irrigation System

THEORETICAL STUDY

Theoretical Framework for an IoT-Based Smart Irrigation System

Our project integrates core principles from sensor technology, IoT systems, embedded systems, communication protocols, and data processing to create an efficient and reliable irrigation monitoring and control system. Below is a breakdown of the theories applied:

1. Environmental Sensing Theory

- **Soil Moisture Sensing:** The system uses a soil moisture sensor to measure the water content in the soil. These sensors work based on changes in electrical conductivity or resistance with moisture levels.The sensor's analog signal represents real-time soil moisture data, crucial for efficient irrigation.temperature variations accurately within a predefined range.
- **Signal Conversion:** The analog output from the soil moisture sensor is converted into digital data using the ADC in the **NodeMCU ESP32**, ensuring compatibility and precision.

2. Embedded Systems Theory

- **Microcontroller-Based System:** The **NodeMCU ESP32** serves as the system's core, processing input from the sensors in real-time. It evaluates soil and environmental data against thresholds to determine whether irrigation is needed.
- **Real-Time Processing:** The ESP32 ensures timely responses to changing conditions, triggering the pump or sending alerts without significant delays.

3. Internet of Things (IoT) Theory

- **IoT Architecture:** The system follows an IoT framework, where devices (sensors, ESP32, and actuators) are connected to the internet for remote monitoring and control. The collected data is transmitted to a mobile application for analysis and system interaction.
- **MQTT Protocol:** Lightweight communication protocols like MQTT is employed to ensure efficient and reliable data transfer between the system and the app over the internet.

4. Communication Theory

- **Wireless Communication:** The ESP32 uses Wi-Fi for wireless data transmission between the microcontroller and the mobile application. It leverages radio frequency communication principles to ensure real-time data exchange.

- **Mobile Notifications:** Notifications are sent using HTTP or push notification services, alerting users about irrigation activities or abnormal conditions such as prolonged dryness.

5. Data Processing Theory

- **Data Validation and Filtering:** To ensure accuracy, the system applies filtering techniques like moving averages to smooth sensor data and reduce noise.This process ensures reliable decision-making for irrigation control.
- **Threshold-Based Decision Making:** The processed data is compared with pre-set thresholds (e.g., soil moisture below 30%). If conditions meet the criteria, the system activates the pump or sends alerts.

6. Mobile Application and Notification Theory

- **Application Design:** The mobile app follows client-server architecture principles, displaying real-time soil and environmental data in a user-friendly interface.
- **Push Notification System:** Notifications, powered by services like Firebase Cloud Messaging (FCM), alert users about irrigation status or anomalies.

7. Data Security Theory

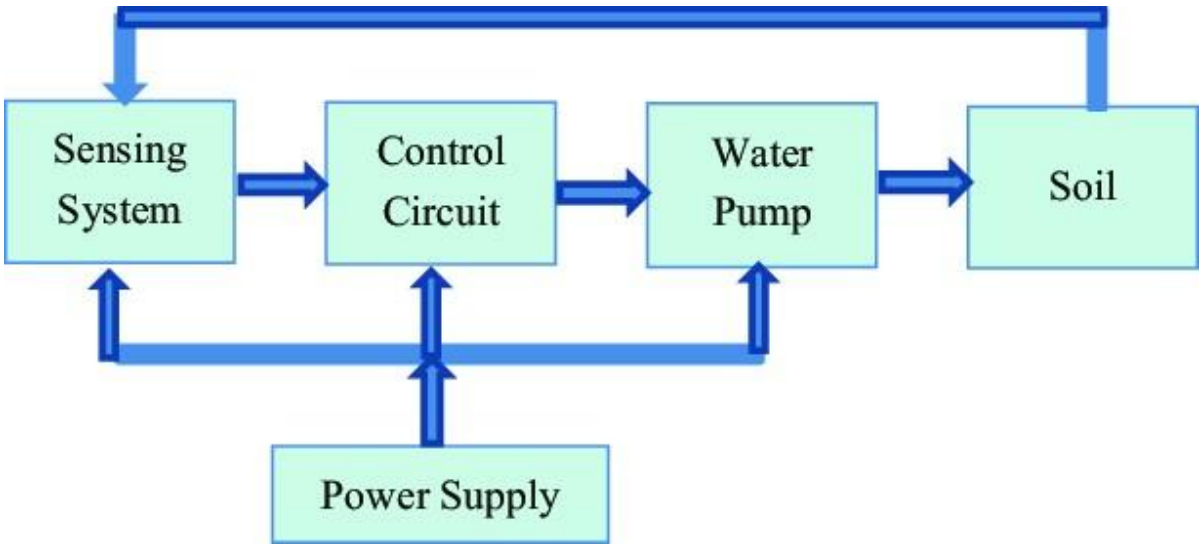
- **Encryption and Authentication:** Secure communication between the ESP32, cloud storage, and the mobile app is ensured using encryption protocols like AES or SSL/TLS. This protects sensitive data from unauthorized access.

8. Cloud Storage and Analytics

- **Cloud Computing Principles:** Soil moisture, temperature, and irrigation logs are stored in a cloud database. This data can be analyzed to optimize irrigation schedules and conserve resources.
- **Remote Monitoring:** Users can monitor their system and control the pump remotely through the mobile app, providing convenience and flexibility.

By combining these theories, the project creates a robust, scalable solution for smart irrigation, improving agricultural productivity, conserving water, and addressing inefficiencies in traditional methods.

PROPOSED METHODOLOGY



SENSORS AND COMPONENTS USED

1. **NodeMCU ESP32**
    - **Purpose:** The NodeMCU ESP32 is a powerful microcontroller used for processing sensor data and enabling IoT functionalities.
    - **Features:**
        - Dual-core processor with integrated Wi-Fi and Bluetooth capabilities.
        - Multiple GPIO pins for interfacing with sensors and actuators.
        - Supports communication protocols like I2C and UART.
    - **Application:** Processes temperature data from the MLX90614 sensor and transmits it to the Blynk IoT platform.

Fig. NodeMCU ESP32

2. **Humidity and Temperature sensor**
    - **Purpose:** The DHT22 sensor is designed to measure **relative humidity** and **temperature** accurately, making it ideal for environmental monitoring and control systems.
    - **Features:**
        - **High Precision**:
            1. Humidity measurement range: **0% to 100% RH** with ±2% accuracy.
            2. Temperature measurement range: **-40°C to +80°C** with ±0.5°C accuracy.
    - **Application:**

- **Environmental Monitoring**: Tracks indoor and outdoor humidity and temperature.
- **Greenhouses**: Ensures optimal growing conditions by monitoring and controlling humidity and temperature.

Fig. DHT22 (Humidity and Temperature Sensor)

3. **Relay Module:**
   - **Purpose:** A relay module is an electronic switch that allows low-power control signals (from a microcontroller) to operate high-power electrical devices, such as motors, lights, and home appliances.
   - **Features:**
     - Uses an electromagnetic coil to separate the control circuit from the high-power circuit, ensuring safety.
     - control devices operating at **AC 220V** or **DC 30V**, with current ratings typically up to **10A**.
   - **Application:** Controls appliances like lights, fans, and heaters.

Fig. Relay Module

4. **Soil Moisture Sensor**
    - **Purpose:** The soil moisture sensor is used to measure the water content in soil, helping to determine whether the soil needs watering, making it ideal for agricultural and gardening applications.
    - **Features:**
        - Provides an analog signal indicating moisture level and a digital output for threshold-based actions.
        - Suitable for battery-operated or solar-powered setups.
    - **Application:**
        - Automates watering by detecting dry soil conditions.
        - Helps maintain optimal water levels for plants in pots or gardens.

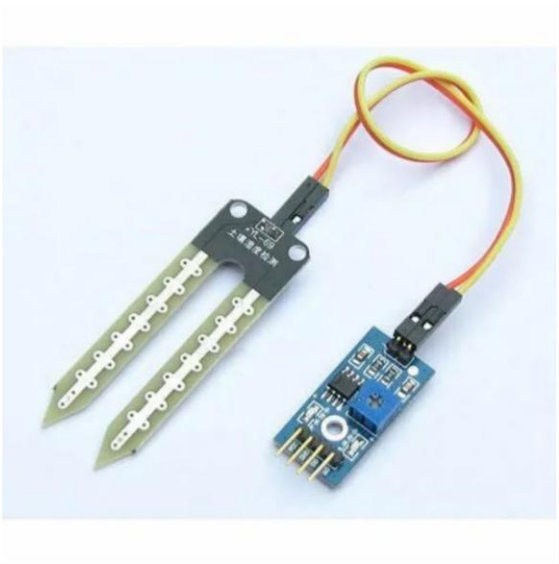


Fig. Soil Moisture Sensor

5. **Pump Motor**
    - **Purpose:** A pump motor is used to transfer or circulate liquids such as water, making it essential in automated systems like irrigation, water circulation, and dispensing.
    - **Features:**
        - Typically operates on **3V to 12V DC**, suitable for microcontroller-based systems.
        - Designed for easy connection to hoses for liquid transfer.
    - **Application:** Automates water delivery in gardens or agriculture.

Fig. Pump Motor

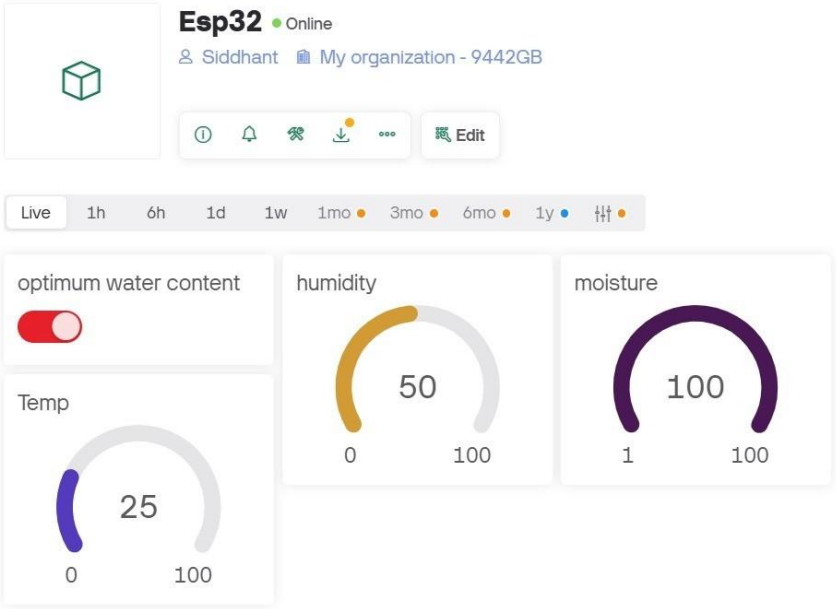
6. **Battery**
    - **Purpose:** A 9-volt battery is a compact and portable power source designed for small electronic devices, making it a staple for prototyping and embedded systems.
    - **Features:**
        - Delivers a consistent **9V** output, ideal for powering sensors, microcontrollers, and small circuits.
        - Designed for reliable performance in everyday applications.
    - **Application**: Used to run microcontroller-based projects like Arduino and ESP32 , etc.

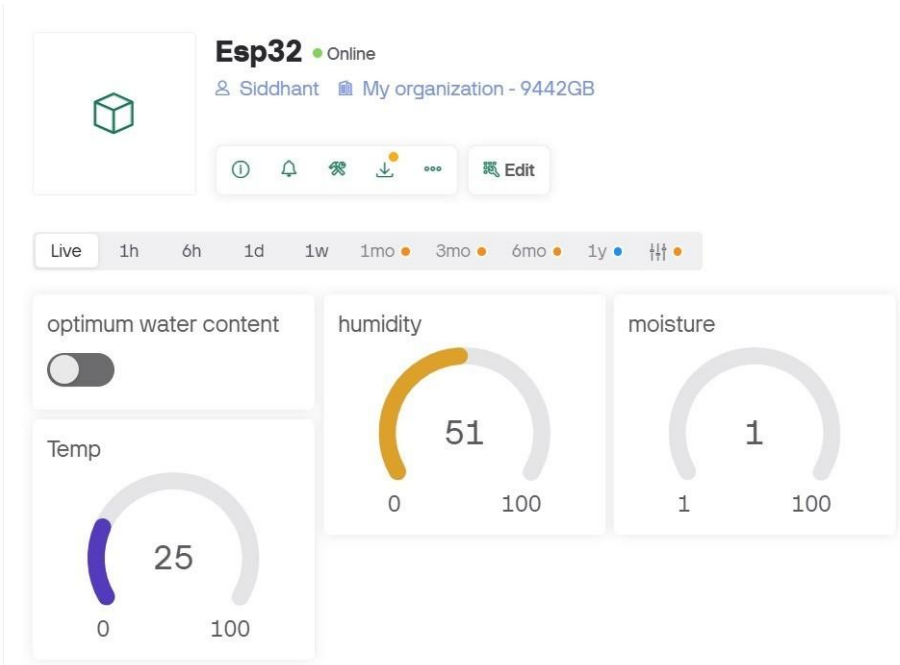
Fig. Battery (9 Volt)

7. **Breadboard:**
    - **Purpose:** A breadboard is a reusable platform designed for building and testing electronic circuits without soldering, making it ideal for prototyping and experimentation.
    - **Features:**
        - Includes a grid of interconnected holes for easy placement of components and wires.
        - Allows components to be inserted and removed multiple times without damage
    - **Application:** Used for teaching electronics and circuit design to students and beginners.

Fig. Breadboard

8. **Blynk Application:**
    - **Purpose:** A cloud-based IoT platform for visualizing data and managing alerts.
    - **Features:**
        - Customizable mobile app dashboard for real-time data display.
        - Push notification system for sending alerts directly to the user.
        - Data logging and analysis capabilities for tracking historical temperature trends.
    - **Application:** Provides a user-friendly interface for remote monitoring and notifications.




Esp32
Online
Siddhant
My organization - 9442GB
Edit
Live
1h
6h
1d
1w
1mo
3mo
6mo
1y
optimum water content
humidity
51
0
100
moisture
1
1
100
Temp
25
0
100

WORK DONE

The development of the IoT-Based Smart Irrigation System involved the integration of various sensors, microcontrollers, communication modules, and IoT platforms to optimize irrigation processes based on real-time environmental and soil conditions. The methodology for the project is outlined below:

1. **Design and Setup**

- **Component Selection:** Identified and procured essential hardware components, including the **Soil Moisture Sensor**, **DHT22 Temperature and Humidity Sensor**, **Relay Module**, **Pump Motor**, **NodeMCU ESP32 microcontroller**, and **Battery** for power supply.
- **System Design:** Designed a circuit that connects the **Soil Moisture Sensor** and **Temperature and Humidity Sensor** to the **NodeMCU ESP32** microcontroller. Configured the **Relay Module** to control the **Pump Motor**, which provides irrigation when needed.

2. **Temperature Measurement System**

- **Sensor Configuration:** Calibrated the **Soil Moisture Sensor** to accurately measure moisture levels in the soil. Configured the **DHT22 Sensor** for real-time temperature and humidity data collection from the environment.
- **Microcontroller Integration:** Programmed the **NodeMCU ESP32** to collect data from both the **Soil Moisture Sensor** and the **Temperature and Humidity Sensor**. The microcontroller analyzes this data and determines whether irrigation is necessary based on predefined moisture thresholds and environmental conditions.

3. **Data Logging and IoT Integration**

- **Mobile Application Integration:** Configured the **NodeMCU ESP32** to connect to an IoT platform (such as Blynk or ThingSpeak) via Wi-Fi. The system sends real-time sensor data (soil moisture, temperature, humidity) to the app, allowing remote monitoring and control.
- **Data Logging**: Enabled the system to log data for further analysis, including tracking soil moisture levels, environmental temperature, and pump activity. Historical data is stored on a cloud server for later review.

4. **Irrigation Control Mechanism**

- **Relay Module Integration:** Connected the **Relay Module** to the **NodeMCU ESP32**, enabling control over the **Pump Motor**. The relay switches the pump on or off based on the moisture levels detected by the sensor.
- **Pump Motor Activation:** When soil moisture levels fall below a predefined threshold, the **NodeMCU ESP32** sends a signal to the relay, which activates the **Pump Motor** to irrigate the soil. Once the moisture level is sufficient, the relay turns the pump off.

5. **Remote Monitoring and Control**

- **Wi-Fi Setup:** Connected the **NodeMCU ESP32** to a local Wi-Fi network to ensure internet access for remote data transmission and monitoring via the mobile app.
- **Push Notifications:** Configured the mobile app to send push notifications when irrigation is triggered, or if sensor readings fall outside the predefined range (e.g., very low moisture or abnormal temperature).

6. **System Testing and Optimization**

- **System Calibration:** Conducted iterative testing to calibrate the **Soil Moisture Sensor** for accurate readings and adjusted the **NodeMCU ESP32** program for optimal real-time decision-making.
- **Performance Evaluation:** Tested the system in various environmental conditions to ensure reliable performance, effective irrigation control, and seamless data transmission. Optimized the system to ensure low power consumption and efficient operation.

This methodology led to the successful development of a smart, IoT-based irrigation system that efficiently manages water usage based on real-time environmental conditions and soil moisture levels, providing an automated and sustainable solution for irrigation management.

CIRCUIT DESCRIPTION

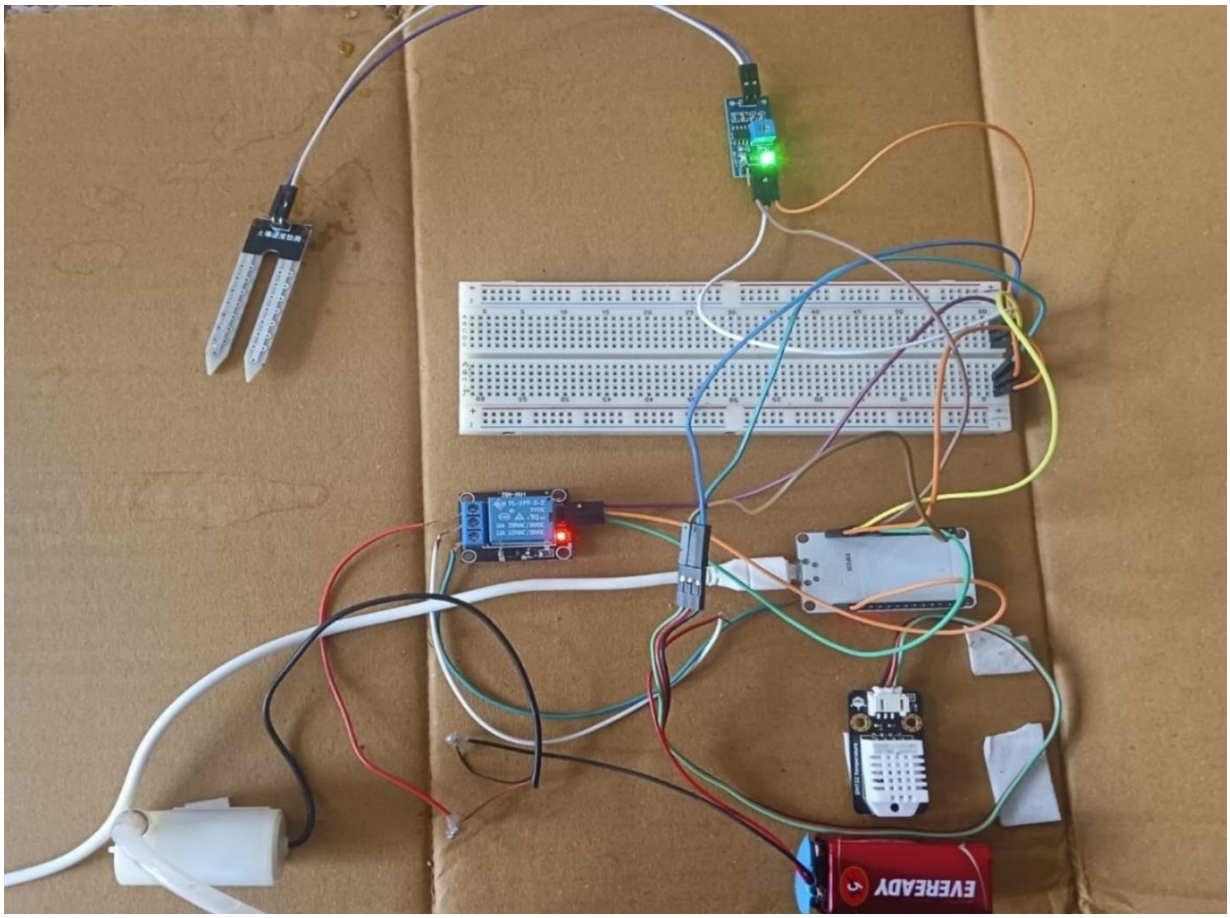


Fig. Circuit For Smart Irrigation System

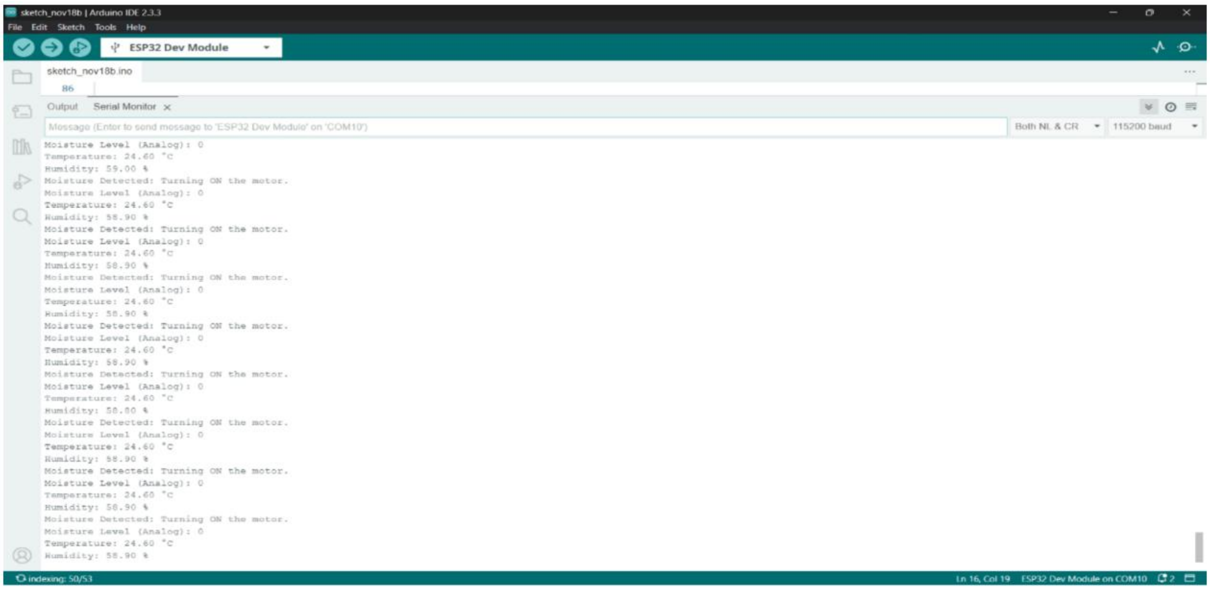


Fig. Arduino Terminal

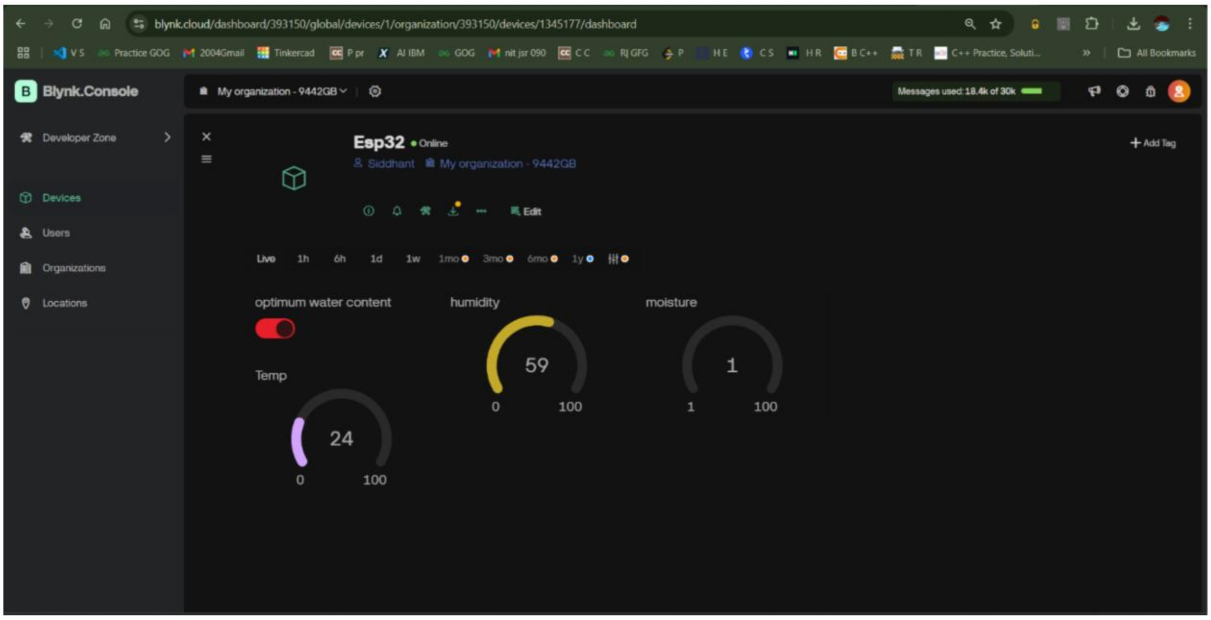
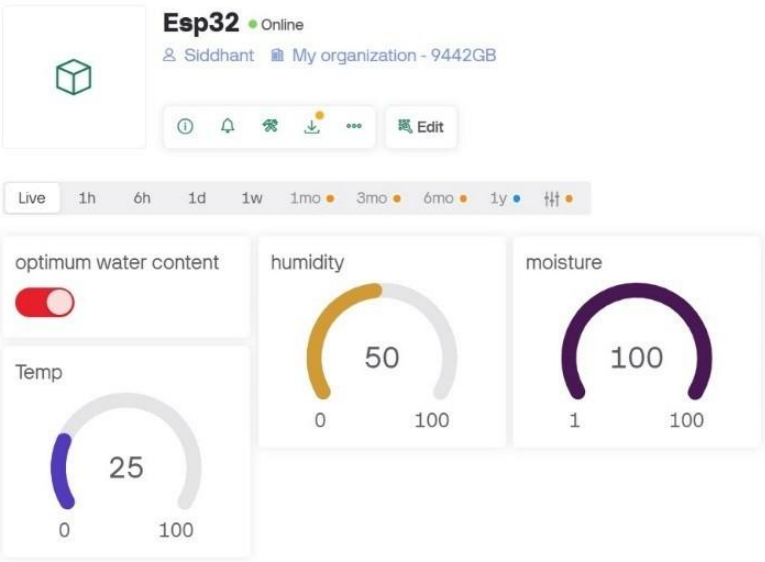
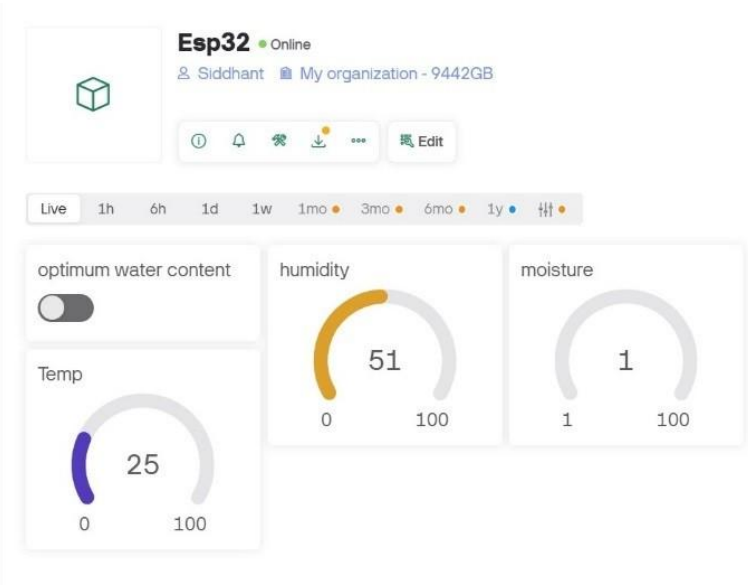


Fig. Browser Dashboard for Smart Irrigation System


Esp32
Online
Siddhant
My organization - 9442GB
Edit
Live
1h
6h
1d
1w
1mo
3mo
6mo
1y
optimum water content
humidity
50
0
100
moisture
100
1
100
Temp
25
0
100


Esp32
Online
Siddhant
My organization - 9442GB
Edit
Live
1h
6h
1d
1w
1mo
3mo
6mo
1y
optimum water content
humidity
51
0
100
moisture
1
1
100
Temp
25
0
100

t

ARDUINO CODE

```
#define BLYNK_TEMPLATE_ID "TMPL3mAqq8Zkt"   // Replace with your Blynk Template ID
#define BLYNK_TEMPLATE_NAME "Iot Project"    // Replace with your Blynk Template Name
#define BLYNK_DEVICE_NAME "esp32"           // Replace with your Device Name

#include <WiFi.h>              // Wi-Fi library for ESP32
#include <BlynkSimpleEsp32.h>  // Blynk library for ESP32
#include "DHT.h"               // DHT library for temperature and humidity sensor

// Wi-Fi credentials
const char* ssid = "Galaxy A14 5G EF76";        // Replace with your Wi-Fi network name
const char* password = "siddrail131";          // Replace with your Wi-Fi password

// Blynk Auth Token
char auth[] = "LPbxpD2NBdSy4axpP1TWMxetO61hsKT5";          // Replace with your Blynk Auth Token

// Pin Definitions
const int relayPin = 4;       // GPIO4 connected to the relay module
const int waterPin = 2;       // GPIO2 corresponds to D2 on ESP32
const int dhtPin = 5;         // GPIO5 connected to the DHT22 data pin

// DHT sensor type
#define DHTTYPE DHT22
```

```
// Variables

int waterSta e;

DHT dht(dhtPin, DHTTYPE);     // Create a DHT object

// Virtual Pins for Blynk App
#define VIRTUAL_TEMP V1       // Virtual pin for temperature (Gauge)
#define VIRTUAL_HUMID V2      // Virtual pin for humidity (Gauge)
#define VIRTUAL_RELAY V3      // Virtual pin to control relay
#define VIRTUAL_WATER V4      // Virtual pin to display water sensor state (Gauge)

void setup() {
  // Initialize Serial Monitor
  Serial.begin(115200);
  delay(10);

  // Configure pins
  pinMode(relayPin, OUTPUT);       // Relay pin as output
  pinMode(waterPin, INPUT_PULLUP); // Use an internal pull-up resistor for the water
sensor pin

  // Ensure the motor is OFF initially
  digitalWrite(relayPin, HIGH);    // Turn OFF relay (motor OFF)

  // Start DHT sensor
  dht.begin();
```

```
t  // Connect to Wi-Fi and Blynk
   Serial.println("Connecting to Wi-Fi and Blynk...");
   Blynk.begin(auth, ssid, password);
```

```
        t    Serial.prin ln("Connected to Wi-Fi and Blynk!");
           }

void loop() {
  // Keep Blynk connected
  Blynk.run();

  // Read water sensor input
  waterState = digitalRead(waterPin);

  // Read temperature and humidity from DHT22
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();

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

  // Send temperature and humidity data to Blynk gauges
  Blynk.virtualWrite(VIRTUAL_TEMP, temperature);    // Update temperature gauge
  Blynk.virtualWrite(VIRTUAL_HUMID, humidity);      // Update humidity gauge

  // Determine water sensor state and send to Blynk
  String waterStatus = (waterState == LOW) ? "Water Detected" : "No Water";
```

```
t
  // Send water status as a gauge (you can display this on a gauge as 0 or 100 or show a
custom message)
  if (waterState == LOW) {
    Blynk.virtualWrite(VIRTUAL_WATER, 100); // Water detected
  } else {
    Blynk.virtualWrite(VIRTUAL_WATER, 0);   // No water detected
  }


  // Relay control logic based on water sensor
  if (waterState == LOW) { // Water detected
    Serial.println("Water Detected: Turning off the Motor");
    digitalWrite(relayPin, LOW); // Turn ON the relay (motor ON)
    Blynk.virtualWrite(VIRTUAL_RELAY, 1); // Update relay state on Blynk
  } else { // No water detected
    Serial.println("No Water Detected: Turning ON the Motor");
    digitalWrite(relayPin, HIGH); // Turn OFF the relay (motor OFF)
    Blynk.virtualWrite(VIRTUAL_RELAY, 0); // Update relay state on Blynk
  }


  // Debugging: Show the sensor values in the Serial Monitor
  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" °C");
  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.println(" %");
  Serial.print("Water sensor state: ");
  Serial.println(waterStatus);
```

```
  delay(2000); // Delay for stabilization
}
```

RESULTS

The developed **Smart Irrigation System** successfully integrates the **Soil Moisture Sensor**, **DHT22 Temperature and Humidity Sensor**, **NodeMCU ESP32 microcontroller**, **Relay Module**, **Pump Motor**, and an IoT platform to provide automated and efficient irrigation management. Key results include: Accurate Temperature Monitoring:

1. **Real-Time Soil Moisture Monitoring:**
   The system accurately measures soil moisture levels using the **Soil Moisture Sensor**, with real-time data displayed on the connected IoT platform or mobile app.

2. **Threshold-Based Irrigation Control:**
   The system activates the **Pump Motor** through the **Relay Module** when soil moisture levels fall below predefined thresholds. Irrigation stops automatically once the moisture level is adequate, ensuring optimal water usage.

3. **Environmental Monitoring:**
   The **DHT22 Temperature and Humidity Sensor** provides real-time environmental data, enabling comprehensive monitoring of the surrounding conditions to enhance the irrigation process.

4. **Mobile Notification System:**
   Notifications are sent to the user's smartphone via the IoT platform, informing them of irrigation events or abnormal conditions like extremely dry soil or extreme environmental temperatures.

5. **Automated Pump Operation:**
   The **Pump Motor** operates efficiently and automatically based on sensor readings, minimizing manual intervention and ensuring consistent irrigation.

6. **Energy Efficiency:**

   The system is powered by a **Battery**, making it energy-efficient and suitable for remote areas with limited access to electricity.

7. **Usefulness for Sustainable Farming:**

   This system is highly beneficial for farmers, reducing water wastage and labor costs while promoting sustainable agricultural practices through automated irrigation.

This project demonstrates a robust and efficient solution for automated irrigation management, combining IoT technology with user-friendly features to address modern agricultural challenges and promote water conservation.

FUTURE WORK

The implementation of an **IoT-Based Smart Irrigation System** using a **Soil Moisture Sensor**, **DHT22 Temperature and Humidity Sensor**, **NodeMCU ESP32**, **Relay Module**, and **Pump Motor** provides a solid foundation for future advancements. Potential future developments include: **Additional Sensors:** Add humidity, air quality, or motion sensors for comprehensive health and environmental monitoring.

1. **Integration of Additional Sensors:** Include sensors for rainfall detection, light intensity (LDR), and water flow monitoring to optimize irrigation decisions further.
2. **AI-Based Irrigation Optimization:** Implement machine learning algorithms to analyze historical soil moisture, temperature, and weather data to predict and automate irrigation needs effectively.
3. **Multi-Zone Control:** Expand the system to manage irrigation for multiple fields or zones, each with customized settings and sensors.
4. **Customizable Alerts:** Enable user-defined thresholds for soil moisture and environmental parameters, with notifications sent via SMS, email, or app.
5. **Offline Functionality:** Incorporate local data storage to ensure uninterrupted operation during internet outages, with automatic synchronization once connectivity is restored.
6. **Renewable Power Sources:** Add solar panels to power the system, ensuring sustainability and operation in remote locations without access to electricity.
7. **Remote Pump Control:** Provide manual override functionality via a mobile app for users to start or stop irrigation remotely when needed.
8. **Enhanced Visualization Tools:** Develop advanced dashboards with graphs and heatmaps for trend analysis, helping users understand water usage and crop conditions.
9. **Weather Forecast Integration:** Integrate real-time weather forecast data to delay or advance irrigation based on expected rainfall or temperature changes.
10. **Geofencing Features:** Automatically activate or deactivate the system based on the user's location, ensuring irrigation only occurs when necessary.
11. **Smart Crop Suggestions:** Use data insights to recommend crop types suitable for current soil and environmental conditions.

These upgrades can enhance the system's usability, scalability, and sustainability, making it a versatile solution for modern agriculture and water management.

CONCLUSION

In conclusion, our **IoT-Based Smart Irrigation System** utilizing the **Soil Moisture Sensor**, **DHT22 Temperature and Humidity Sensor**, **NodeMCU ESP32**, **Relay Module**, **Pump Motor**, and **Battery** offers an innovative and efficient solution for automated irrigation management. By integrating accurate soil moisture detection, environmental monitoring, wireless communication, and a user-friendly IoT platform, the system ensures optimal water usage while reducing manual intervention.

The **Blynk IoT application** enhances user experience by enabling real-time remote monitoring and control of irrigation activities, allowing users to access critical data and manage the system from anywhere. Notifications for low soil moisture or irrigation activation ensure timely and efficient responses, contributing to water conservation and crop health.

This project demonstrates the transformative potential of IoT technology in agriculture, providing an affordable, scalable, and sustainable solution for modern farming practices. Future enhancements, including additional sensors, AI-based analytics, and renewable power sources, will further improve its reliability and usability, making it a versatile tool for diverse agricultural applications.

REFERENCES

1. **Kumar, S., & Verma, A. (2020). IoT-Based Smart Irrigation System Using ESP32.** *International Journal of Advanced Research in Computer Science and Electronics Engineering.* Retrieved from https://www.ijarcsee.org/
2. **DHT22 Temperature and Humidity Sensor Datasheet. (n.d.).** Adafruit Learning System. Retrieved from https://learn.adafruit.com/
3. **Soil Moisture Sensor: Working Principle and Applications. (n.d.).** SparkFun Electronics. Retrieved from https://www.sparkfun.com/
4. **NodeMCU ESP32 Datasheet and Applications. (n.d.).** Espressif Systems. Retrieved from https://www.espressif.com/en/products/socs/esp32/resources
5. **Relay Module Datasheet and Specifications. (2018).** Electronics Hub. Retrieved from https://www.electronicshub.org/
6. **Blynk IoT Platform Documentation. (n.d.).** Blynk Inc. Retrieved from https://docs.blynk.io/

These references provide detailed insights into the components and technologies utilized in the development of the **IoT-Based Smart Irrigation System**, offering valuable information on design, implementation, and applications.