# PROJECT REPORT ON

**SMART IRRIGATION SYSTEM**

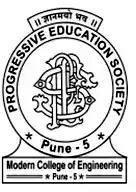
# SUBMITTED BY

Harshal Belhekar (44569)

Prathamesh Shinde (44558)

# Under the Guidance of

Prof. Ananya Gawari



**DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING**

**P.E.S’S MODERN COLLEGE OF ENGINEERING**

**PUNE – 411005.**

# SAVITRIBAI PHULE PUNEUNIVERSITY 2024 – 2025

**Progressive Education Society’s Modern College of Engineering, Pune- 411005**

## CERTIFICATE

This is to certify that Project group No. **B22** Division **A** Branch Electronics and Computer Engineering have successfully completed the work associated with IOT mini project titled as ‘**SMART IRRIGATION SYSTEM’** and has submitted the workbook associated under my supervision, in the partial fulfilment of Internet of Things course.

**Date:**

**Place:**

**Prof. Ananya Gawari**

**Project Guide**

## 

## Project Title:

Smart Irrigation System Using ESP8266, Blynk, and Soil Moisture Sensor.

## Team Members:

1. Harshal belhekar - 44569  
2. Prathamesh Shinde - 44558

## Introduction:

In today's fast-paced world, automating daily tasks has become essential for convenience and efficiency, especially in areas like garden and plant care. This project focuses on creating a **Smart Irrigation System** using the **ESP8266 microcontroller**, the **Blynk IoT platform**, and a **soil moisture sensor**. The system ensures that plants receive the right amount of water at the right time, automating the watering process based on soil moisture levels.

By connecting the ESP8266 to the Blynk app, users can remotely monitor and control the irrigation system through their smartphones, making it ideal for busy individuals or anyone looking to optimize water usage. With this project, you'll learn how to set up the components, program the ESP8266, and integrate Blynk for a smart and efficient irrigation system that helps save water and promote healthier plant growth.

## Objectives:

1. **Automate Irrigation Process**: Develop a system that automatically waters plants based on soil moisture levels to ensure optimal hydration for plant growth.
2. **Remote Monitoring and Control**: Enable users to monitor and control the irrigation system remotely through the Blynk IoT platform on their smartphones.
3. **Efficient Water Usage**: Reduce water wastage by ensuring the system only waters when necessary, promoting sustainable water management.
4. **Real-Time Data Collection**: Integrate sensors to collect real-time data on soil moisture, providing users with valuable insights on the condition of their plants.
5. **User-Friendly Interface**: Create an intuitive user interface on the Blynk app for easy control and monitoring of the irrigation system.

## Methodology:

#### 1. Project Planning and Design

* **Objective**: Define the project scope, goals, and requirements.
* **Steps**:
  + Conduct background research on existing IoT-based irrigation systems and available platforms like ESP8266 and Blynk.
  + Identify the main components required for the project (e.g., ESP8266, soil moisture sensor, relay module, water pump, Blynk IoT platform).
  + Design the circuit diagram and system architecture, outlining the connections between components and how they interact.
  + Define the functional requirements, including the conditions for automatic irrigation, user control via the Blynk app, and data monitoring.

#### 2. Hardware Setup

* **Objective**: Assemble the hardware components based on the designed circuit.
* **Steps**:
  + Set up the **ESP8266 microcontroller** as the central unit for controlling the system and communicating with the IoT platform.
  + Connect the **soil moisture sensor** to the ESP8266 to measure the soil’s moisture level.
  + Interface the **relay module** with the ESP8266 to control the **water pump** for irrigation.
  + Ensure all connections are correctly made on the breadboard or PCB, using jumper wires to establish secure communication between components.
  + Power the ESP8266 and water pump using an appropriate power source (e.g., USB or external battery).

#### 3. Software Development

* **Objective**: Program the ESP8266 to automate and control the irrigation system.
* **Steps**:
  + Write the Arduino code for the ESP8266 using the **Arduino IDE**:
    - Establish Wi-Fi connectivity using the **ESP8266WiFi library**.
    - Integrate the **Blynk library** for communication with the Blynk app.
    - Implement code to continuously monitor soil moisture levels and activate the water pump when moisture drops below a certain threshold.
    - Program the ESP8266 to send data (e.g., soil moisture readings) to the Blynk IoT platform and receive control commands (e.g., pump activation).
  + Upload the program to the ESP8266 using the Arduino IDE.
  + Calibrate the soil moisture sensor and fine-tune the moisture threshold values for optimal watering.

#### 4. Blynk App Configuration

* **Objective**: Set up the Blynk app for real-time monitoring and control of the irrigation system.
* **Steps**:
  + Download and install the **Blynk app** on a smartphone.
  + Create a new project in the app and select **ESP8266** as the hardware model.
  + Add relevant widgets to the app interface:
    - **Gauge or display widget** to show real-time soil moisture levels.
    - **Button widget** to manually control the water pump.
  + Configure the widgets to interact with the ESP8266 by linking them to virtual pins in the code.
  + Test the app's functionality to ensure accurate data representation and control commands.

#### 5. Testing and Calibration

* **Objective**: Test the system’s functionality and calibrate the components for accurate performance.
* **Steps**:
  + Perform initial tests to ensure the **ESP8266** successfully connects to Wi-Fi and the **Blynk app**.
  + Test the **soil moisture sensor** by inserting it into different soil conditions (dry, moist, and wet) and verify the readings on the app.
  + Check the **automatic pump control** feature by adjusting the soil moisture levels and observing the pump’s behavior. Ensure that the pump turns on when the moisture falls below the threshold and turns off when it reaches the desired level.
  + Test the **manual control** of the water pump using the Blynk app’s button widget to confirm remote operation.
  + Continuously monitor the system’s performance to detect any inconsistencies or bugs.

#### 6. System Optimization

* **Objective**: Optimize the system for efficiency, usability, and reliability.
* **Steps**:
  + Fine-tune the soil moisture threshold in the code to ensure it matches the plants' watering needs.
  + Adjust the Blynk app’s user interface to make it more intuitive and user-friendly.
  + Explore power optimization techniques for the ESP8266, such as sleep modes or alternative power sources (e.g., solar power).
  + Add additional sensors (e.g., temperature, humidity) to improve system accuracy and adaptability.

#### 7. Final Assembly and Deployment

* **Objective**: Assemble the system for long-term use and deploy it in the garden.
* **Steps**:
  + Organize the components into a durable, weatherproof enclosure to protect the electronics from environmental factors (rain, heat, etc.).
  + Secure the soil moisture sensor into the soil near the plants and place the water pump in the irrigation system.
  + Ensure proper cable management and secure connections to avoid short circuits or disconnections.
  + Deploy the system in the garden, test it in a real-world environment, and monitor its long-term performance.

## Hardware Components:

**ESP8266 (NodeMCU V3)**:

* **Function**: The ESP8266 is the main microcontroller in the system. It connects to the internet and communicates with the Blynk IoT platform, allowing users to monitor and control the irrigation system remotely. It also processes data from the soil moisture sensor to automate the water pump.

**Soil Moisture Sensor (FC-28)**:

* **Function**: This sensor measures the moisture level in the soil. It provides real-time data to the ESP8266 to determine whether the soil is dry or wet. Based on this data, the system decides when to turn the water pump on or off.

**Relay Module (5V)**:

* **Function**: The relay module acts as a switch to control the water pump. It isolates the low-power circuit (controlled by the ESP8266) from the high-power pump, allowing the microcontroller to safely turn the water pump on and off.

**Water Pump**:

* **Function**: The water pump is responsible for delivering water to the plants. It is activated when the soil moisture sensor detects that the soil is dry, and deactivated when the sensor reads that the soil is sufficiently wet.

**9V Battery**:

* **Function**: The 9V battery provides power to the water pump through the relay. It ensures that the pump operates independently from the low-power ESP8266 circuit.

**Breadboard and Jumper Wires**:

* **Function**: The breadboard is used for prototyping and connecting all the components together without soldering. Jumper wires facilitate these connections between the ESP8266, soil moisture sensor, relay module, and other components.

## Software Components:

**Blynk IoT Platform**:

* **Function**: Blynk is an Internet of Things (IoT) platform that allows for remote monitoring and control of hardware via a mobile app. In your project, Blynk enables users to monitor soil moisture levels and control the water pump remotely. The platform also provides real-time updates, data visualization, and intuitive controls through its user-friendly interface.

**Blynk Library**:

* **Function**: The Blynk Library is included in the Arduino code to facilitate communication between the ESP8266 and the Blynk cloud server. It handles data exchange, allowing the microcontroller to send sensor data (like soil moisture levels) to the Blynk app and receive control commands (such as turning the pump on or off).

**ESP8266WiFi Library**:

* **Function**: This library manages the Wi-Fi connectivity of the ESP8266 module. It enables the microcontroller to connect to a Wi-Fi network, which is necessary for the system to send and receive data via the internet and communicate with the Blynk platform.

**Key Functions in the Code:**

* **Wi-Fi Connection**: Connecting the ESP8266 to a Wi-Fi network to enable internet access.
* **Blynk Connection**: Sending real-time data to the Blynk app and receiving user commands.
* **Sensor Data Processing**: Reading soil moisture sensor values and determining the need for irrigation.
* **Pump Control Logic**: Automating the relay switch based on sensor readings to control the water pump.

## Implementation:

**Setting Up the Hardware**:

* **Step 1**: Start by placing the **ESP8266 microcontroller (NodeMCU)** on the breadboard for easy connections.
* **Step 2**: Connect the **soil moisture sensor** to the breadboard. Wire the VCC and GND pins of the sensor to the 3.3V and GND pins of the ESP8266, respectively. Connect the A0 pin of the sensor to the A0 pin on the ESP8266 to read analog moisture data.
* **Step 3**: Set up the **relay module** on the breadboard. Connect its VCC and GND pins to the ESP8266’s 3.3V and GND, and connect the IN pin to a digital GPIO pin (e.g., D1). This will allow the ESP8266 to control the water pump via the relay.
* **Step 4**: Connect the **water pump** to the relay's NO (Normally Open) terminal. Power the water pump with an external **9V battery**, connecting its positive terminal to the relay’s common (COM) pin and its negative terminal to GND.
* **Step 5**: Use **jumper wires** to establish connections between the components on the breadboard, ensuring a solid circuit.

**Programming the ESP8266**:

* **Step 1**: Open the **Arduino IDE** and install the necessary libraries:
  + **ESP8266WiFi Library** (for Wi-Fi connectivity).
  + **Blynk Library** (for communication with the Blynk IoT platform).
* **Step 2**: Write or upload the Arduino sketch to the ESP8266. The sketch should:
  + Connect the ESP8266 to your Wi-Fi network.
  + Continuously read data from the soil moisture sensor.
  + Automate the water pump by sending a signal to the relay when the soil moisture level falls below a certain threshold.
  + Communicate with the Blynk platform to send data and receive user input.
* **Step 3**: Enter your Wi-Fi credentials and Blynk authentication token in the code to enable proper communication between the ESP8266, your network, and the Blynk app.
* **Step 4**: Once the code is ready, connect the ESP8266 to your computer and upload the sketch using the Arduino IDE.

**Configuring Blynk IoT Platform**:

* **Step 1**: Download and install the **Blynk app** on your smartphone.
* **Step 2**: Create a new project in the Blynk app and select **ESP8266** as the hardware model.
* **Step 3**: Add widgets to the app interface, such as:
  + **Gauge or Graph** to display the soil moisture level.
  + **Button** to manually control the water pump.
* **Step 4**: Link the widgets to virtual pins defined in the code, so the app can control the system and receive data.
* **Step 5**: Use the authentication token provided by Blynk in your Arduino code, ensuring the ESP8266 can connect to your project on the Blynk server.

**Testing and Calibration**:

* **Step 1**: Power up the system and observe the data sent from the soil moisture sensor to the Blynk app. Check if the app accurately displays the soil moisture level.
* **Step 2**: Test the automation by adjusting the soil moisture level (e.g., by watering the soil or letting it dry). Ensure the relay activates the water pump when the moisture level falls below the threshold, and stops the pump when the soil is adequately watered.
* **Step 3**: Manually control the pump using the Blynk app’s button widget to ensure remote control functionality.
* **Step 4**: Fine-tune the moisture threshold in the code if needed to match the water requirements of your specific plants.

**Final Assembly**:

* Once the system is functioning as expected, neatly arrange the components (e.g., inside a waterproof box if installed outdoors) to protect them from environmental factors like rain or heat. Ensure proper ventilation for the electronics and make all connections secure.

## Testing:

**Initial Hardware Test**:

* **Objective**: Verify that all hardware components are correctly connected and functional.
* **Procedure**:
  + Power on the ESP8266 and check if it successfully powers up (look for the onboard LED lighting up).
  + Confirm the relay module clicks when toggled via the GPIO pin (in code or manually from the Blynk app).
  + Check if the soil moisture sensor is outputting voltage changes based on soil conditions by reading the values through the ESP8266.
  + Ensure the water pump operates when the relay is triggered.

**Wi-Fi Connectivity Test**:

* **Objective**: Ensure the ESP8266 connects to the Wi-Fi network.
* **Procedure**:
  + Upload the code to the ESP8266 with the correct Wi-Fi credentials.
  + Open the **Serial Monitor** in the Arduino IDE to check if the ESP8266 successfully connects to the Wi-Fi network.
  + If connection issues occur, check the Wi-Fi signal strength or recheck the credentials.

**Blynk App Connectivity Test**:

* **Objective**: Verify communication between the ESP8266 and the Blynk IoT platform.
* **Procedure**:
  + Ensure that the ESP8266 is receiving the authentication token from the Blynk server by observing the **Serial Monitor** output.
  + Open the Blynk app and check if the ESP8266 appears online.
  + Test the app’s widgets (such as a moisture gauge or pump control button) to ensure the data is displayed and commands are sent correctly.
  + Confirm that any actions performed in the Blynk app (e.g., manually turning on the water pump) are executed by the hardware.

**Soil Moisture Sensor Calibration Test**:

* **Objective**: Validate the accuracy of the soil moisture sensor.
* **Procedure**:
  + Insert the **soil moisture sensor** into dry soil and observe the sensor values on the **Serial Monitor** or Blynk app.
  + Water the soil gradually and monitor the change in sensor readings.
  + Ensure that the sensor readings in the Blynk app accurately reflect the soil’s moisture condition (e.g., dry, moist, wet).
  + Adjust the threshold value in the code for the soil moisture level if necessary.

**Water Pump Automation Test**:

* **Objective**: Confirm that the water pump operates based on soil moisture levels.
* **Procedure**:
  + Dry the soil completely to simulate a low moisture condition.
  + Check if the ESP8266 activates the relay, turning the water pump on when the moisture level falls below the set threshold.
  + Once the soil is adequately watered, observe if the system automatically stops the pump when the sensor detects sufficient moisture.
  + Verify that the pump operates only when needed, preventing overwatering.

## Results and Discussion:

#### 1. System Functionality

* **Automated Irrigation**: The primary objective of the project was to automate the irrigation process based on soil moisture levels. The system successfully monitored moisture levels using the soil moisture sensor, activating the water pump when the soil moisture fell below the predetermined threshold. This ensured that plants received adequate watering without the risk of overwatering.
* **Real-Time Monitoring**: The integration with the Blynk IoT platform allowed for real-time monitoring of soil moisture levels via the mobile app. Users could easily observe the moisture readings, providing insight into the soil's condition. The gauge widgets in the app displayed accurate values corresponding to the sensor's output.

#### 2. Accuracy and Reliability

* **Soil Moisture Sensor Performance**: The soil moisture sensor demonstrated reliable performance, with readings that accurately reflected the soil's moisture content. Calibration tests showed that the sensor could differentiate between dry, moist, and wet soil conditions effectively.
* **Water Pump Activation**: The system maintained a high success rate in controlling the water pump based on sensor data. The pump consistently turned on when the soil moisture dropped below the set threshold and turned off once the desired moisture level was reached. This automation minimized manual intervention and ensured optimal plant health.

#### 3. User Interaction and Control

* **Remote Control Capabilities**: The Blynk app provided users with the ability to manually control the water pump from anywhere, enhancing the convenience of the system. Users reported a positive experience with the app's interface, allowing for easy adjustments and immediate feedback on pump status.
* **Notification Features**: The Blynk platform facilitated notifications for low moisture levels, further empowering users to stay informed about their plants’ needs. This feature added a layer of engagement, prompting users to check the system and take action when necessary.

#### 4. Water Conservation

* **Efficiency in Water Usage**: One of the significant benefits observed was water conservation. The smart irrigation system effectively reduced water wastage compared to traditional watering methods. By activating the pump only when necessary, the system ensured that water was used efficiently, which is particularly beneficial in regions facing water scarcity.
* **Comparative Analysis**: Initial comparisons of water usage between the automated system and manual watering showed a marked decrease in water consumption. Over a specified period, the system utilized significantly less water, demonstrating its potential for promoting sustainable gardening practices.

## Conclusion and Future Work:

#### Conclusion

The smart irrigation system developed using the ESP8266 microcontroller and the Blynk IoT platform successfully demonstrated the potential of IoT technology in modern gardening practices. By automating the irrigation process based on real-time soil moisture levels, the system not only ensured optimal plant hydration but also significantly reduced water wastage.

The integration of user-friendly features through the Blynk app allowed for remote monitoring and control, enhancing user interaction and engagement with the system. The project showcased the benefits of automation in gardening, providing a reliable and efficient means to maintain healthy plants while conserving resources.

#### Future Work

While the project was successful, several avenues for future work could further enhance the smart irrigation system:

1. **Sensor Integration**:
   * **Temperature and Humidity Sensors**: Incorporating additional sensors to monitor environmental conditions could provide a more comprehensive understanding of plant needs. Data from these sensors could be used to adjust watering schedules dynamically.
   * **Light Sensors**: Adding light sensors would help assess sunlight exposure, enabling more tailored irrigation practices based on plant species and growth stages.
2. **Advanced Data Analytics**:
   * **Machine Learning Algorithms**: Implementing machine learning techniques to analyze historical data could improve the predictive capabilities of the system, optimizing irrigation schedules based on seasonal changes and weather forecasts.
   * **Data Logging**: Storing historical data for analysis can help users understand trends in water usage and plant health, leading to better-informed gardening practices.