SMART GARDENING SYSTEM

Mini Project – IT 23A31- Internet of Things

Submitted By

Siva Bharathi K(230701317)

Sowbarnigaa Sridharan(230701327)

May 2025

Department of Computer Science and Engineering

Rajalakshmi Engineering College, Thandalam

BONAFIDE CERTIFICATE

This is to certify that the Mini project work titled “**Smart Gardening System**” done by **“Siva Bharathi K(230701317) and Sowbarnigaa Sridharan(230701327)"**, is a record of bonafide work carried out by him/her under my supervision as a part of Mini project for the Course IT23A31- Internet of Things , Department of Computer Science and Engineering, REC.

SUPERVISOR   Head/CSE

Date:

TABLE OF CONTENT

ABSTRACT

This project presents the design and development of a Smart Irrigation System capable of adapting to different soil types using an Arduino UNO, soil moisture sensor, relay module, DC water pump, and a 5V power supply. The system intelligently automates the irrigation process by continuously monitoring soil moisture levels and dynamically adjusting its behavior based on the identified soil type. Unlike conventional systems with static thresholds, this smart solution utilizes sensor data to classify the soil type (e.g., sandy, loamy, or clay) and determine the optimal moisture threshold for irrigation. When the moisture content falls below the appropriate level for the detected soil type, the Arduino activates the water pump via the relay module to irrigate the soil. The pump is automatically deactivated once the ideal moisture range is restored, ensuring precise and efficient water usage. This approach minimizes water waste, enhances plant health, and reduces manual intervention. The system's adaptability and low power requirements make it ideal for use in home gardens, farms, and greenhouse environments. By combining real-time sensing, soil-type intelligence, and automation, the Smart Irrigation System offers a scalable and sustainable solution to modern agricultural and gardening challenges.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

Efficient and timely irrigation is essential for sustaining healthy plant growth in home gardens, agricultural fields, and controlled indoor environments. Manual watering methods are often labor-intensive, inconsistent, and inefficient, leading to issues such as over-watering or under-watering. These problems become more pronounced during periods when users are unavailable to tend to their plants, resulting in potential plant stress or damage.

Traditional irrigation systems typically operate on fixed schedules or static moisture thresholds, without accounting for the unique water retention characteristics of different soil types such as sandy, loamy, or clay soils. This lack of adaptability leads to suboptimal irrigation, water wastage, and poor crop health. Moreover, such systems require frequent user monitoring and adjustment, which defeats the purpose of automation.

To address these challenges, this project proposes the development of a Smart Irrigation System that automatically detects soil moisture levels and dynamically adjusts its irrigation logic based on the identified soil type. The system uses an Arduino microcontroller, soil moisture sensor, relay module, and water pump to automate the irrigation process. By calibrating optimal moisture thresholds for different soil types, the system ensures that water is supplied precisely when and where needed. This not only promotes plant health and conserves water but also significantly reduces human effort.

The solution is ideal for gardeners, farmers, and environmentally conscious users seeking a more intelligent and sustainable approach to irrigation.

1.2 Objective of the project

The objective of this project is to design an Automatic Irrigation System that automates the watering of plants based on soil moisture levels. The system aims to:

1. Automate irrigation to eliminate manual effort.

2. Optimize water usage by watering only when necessary.

3. Maintain optimal soil moisture for healthy plant growth.

4. Reduce human intervention, especially during absences.

5. Provide a customizable and energy-efficient solution adaptable for home gardens and agricultural fields.

The project addresses modern irrigation challenges by promoting efficiency, sustainability, and ease of use.

CHAPTER 2

SYSTEM DESIGN

2.1 System Architecture / Circuit Diagram

The Automatic Irrigation System is composed of the following key components working together to automate the watering process for plants:

Soil Moisture Sensor:

Role: The soil moisture sensor is used to measure the moisture level in the soil. It provides real-time data on the water content of the soil.

Working: The sensor detects soil conductivity, which varies based on the moisture content. It sends analog or digital signals to the Arduino to indicate whether the soil is dry or wet.

Arduino UNO:

Role: The Arduino UNO acts as the central controller for the system. It processes data from the soil moisture sensor and controls the relay module to manage the water pump.

Working: Based on the moisture readings, the Arduino decides whether to turn the water pump ON or OFF by sending control signals to the relay module.

Relay Module:

Role: The relay module serves as a switch to control the operation of the DC water pump.

Working: When the Arduino sends a signal, the relay module activates or deactivates the pump to supply water to the soil.

Water Pump:

Role: The water pump is responsible for delivering water to the plants when the soil moisture is low.

Working: The pump draws water from a reservoir and irrigates the soil whenever the relay module activates it.

Power Supply (5V Battery):

Role: The 5V battery powers the Arduino, soil moisture sensor, and relay module, providing a portable and energy-efficient solution for the system.

Flow of Operation:

1. The soil moisture sensor continuously monitors the moisture level in the soil in real-time.
2. The system analyzes the sensor data to determine the soil type (e.g., sandy, loamy, or clay) based on its moisture retention behavior.
3. For the detected soil type, a specific moisture threshold is set dynamically to ensure optimal irrigation.
4. If the soil moisture level falls below the threshold defined for that soil type, the Arduino processes the data and activates the relay module.
5. The relay module then turns on the water pump, initiating irrigation.
6. As the soil absorbs water, the moisture level rises. Once it reaches the optimal threshold for that particular soil type, the Arduino deactivates the relay, turning off the water pump.

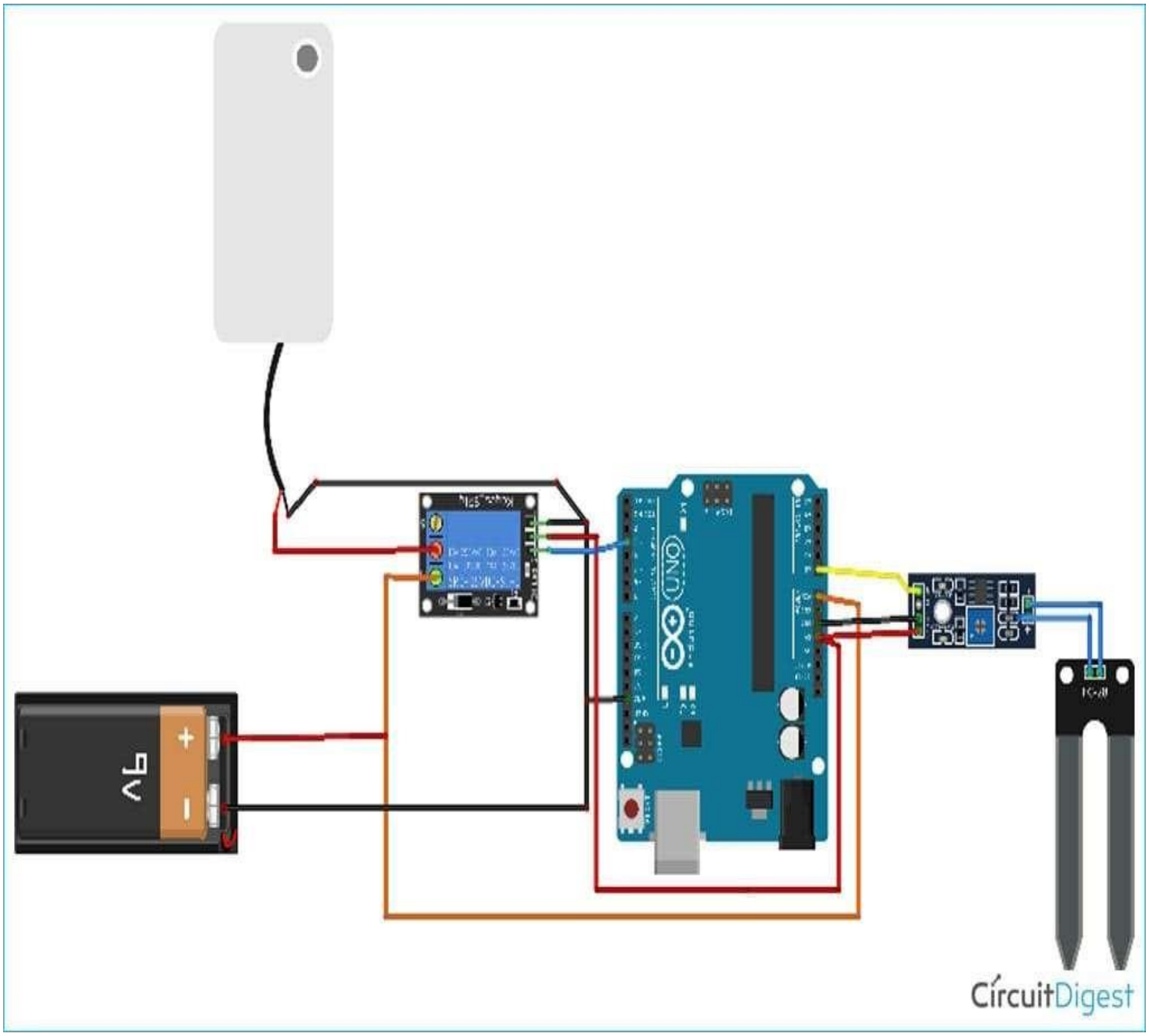
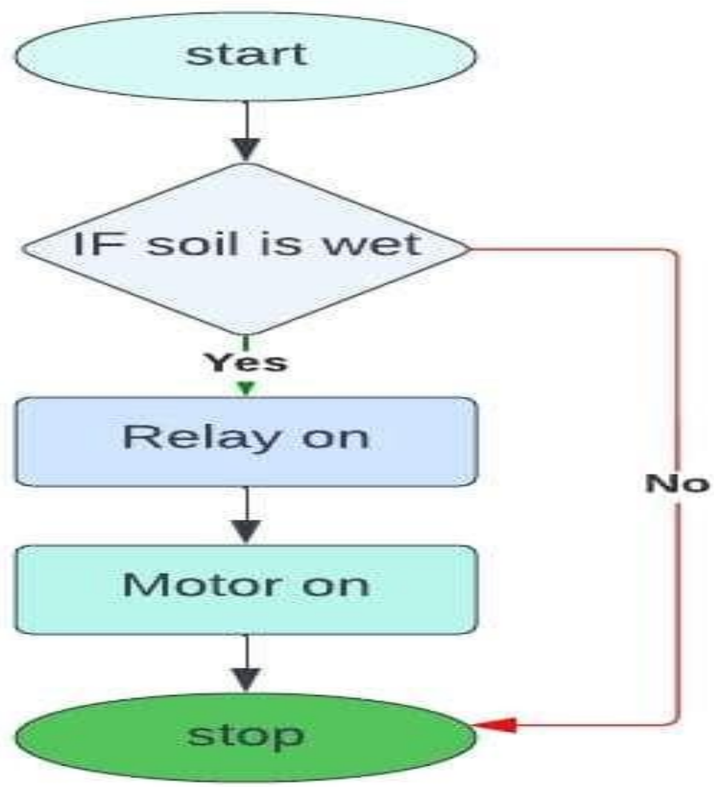


Figure 2.1 Circuit diagram

2.2 UML Modeling

2.2.1 Use Case Diagram



Flowchart

2.3 System Specification

2.3.1. Software Requirements

1. Arduino IDE (Integrated Development Environment)

Version: 1.8.16

Role: The Arduino IDE is used to write, compile, and upload the code to the Arduino board. It provides an interface for programming the microcontroller and interacting with various hardware components such as soil moisture sensors, water pumps, and relay switches.

2. Programming Language

C/C++: The Arduino code is primarily written in C/C++. This language allows for efficient control of hardware components by enabling simple logic to read sensor data and control the irrigation system's activation (e.g., water pumps). The web dashboard is run using Python (Flask).

3. Libraries

Soil Moisture Sensor Library:

Version: Standard library available in Arduino IDE or custom libraries based on the sensor model.

Role: This library facilitates interaction with the soil moisture sensor to read moisture levels from the soil and determine when irrigation is needed.

Relay Library:

Version: Standard version available in Arduino IDE or custom relay control libraries.

Role: Used to control the relay that switches the water pump on and off based on sensor readings.

4. Arduino Board Drivers

Role: Drivers are necessary for communication between the Arduino board and the computer to upload the code. These drivers are typically installed automatically during the IDE setup but may require manual installation in certain cases.

5. Serial Monitor (for Debugging)

Role: The Serial Monitor helps in debugging and provides real-time feedback on the system's performance. It is used to display sensor data, such as soil moisture levels, and confirm that the irrigation system is working correctly by showing when the pump is activated.

2.3.2. Hardware requirements

1. Arduino UNO:

The Arduino UNO serves as the central processing unit, responsible for controlling the sensor readings, processing data, and triggering alarm signals.

Specifications:

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Digital I/O Pins: 14 (6 PWM outputs)
- Power: Can be powered by USB or 9V battery

Quantity: 1

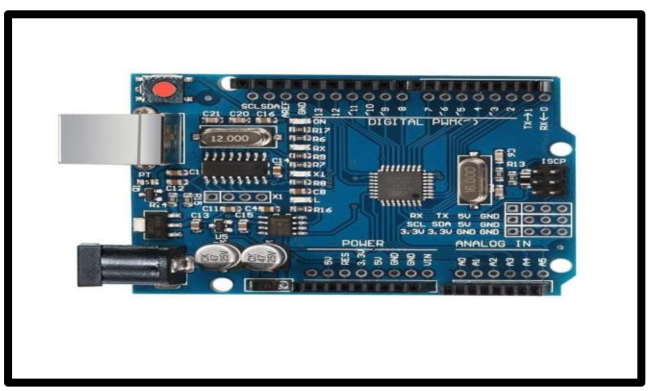


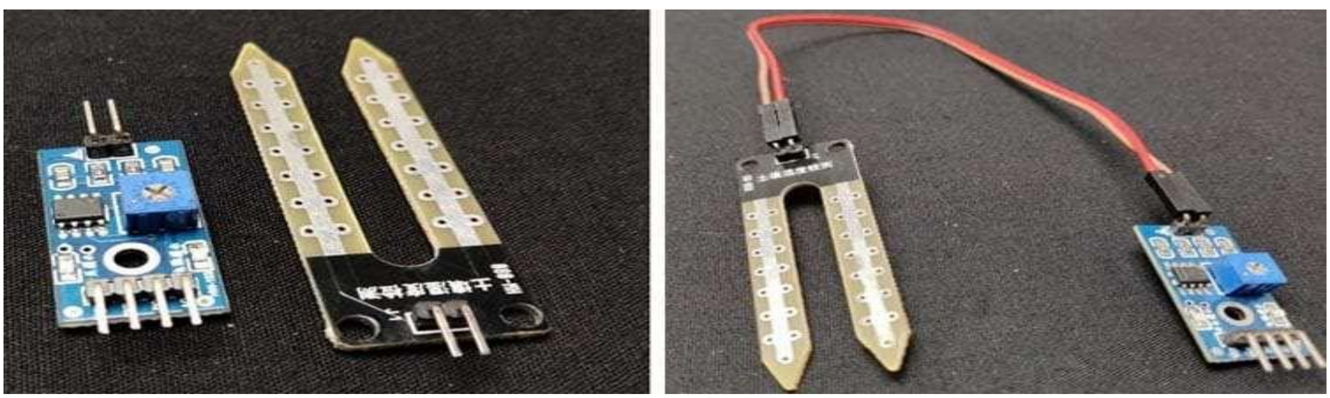
Figure 2.3.2(a) Arduino Uno

2.Soil Moisture Sensor

The soil moisture sensor plays a crucial role in monitoring the water content of the soil in real-time. In this smart irrigation system, the sensor not only detects whether the soil is dry or sufficiently moist but also provides continuous data used to identify the soil type based on its moisture retention characteristics. This information enables the system to dynamically adjust moisture thresholds according to the specific needs of different soil types (e.g., sandy, loamy, or clay). By accurately determining when irrigation is necessary, the soil moisture sensor ensures precise water delivery, promoting healthy plant growth while minimizing water waste.

Specifications:

- Operating Voltage: 3.3V to 5V
- Measurement Range: 0% to 100% (corresponding to dry to wet soil)
- Output: Analog signal (varies with soil moisture level)
- Response Time: Typically 1-2 seconds



- Quantity: 1

Figure 2.3.2(b) Soil Moisture Sensor

3. Relay Module 5V

The 5V relay module is used to control high-voltage devices such as water pumps in the irrigation system. It acts as a switch that can be triggered by the microcontroller to turn the pump on or off based on the soil moisture sensor readings.

Specifications:

- Operating Voltage: 5V
- Trigger Voltage: 3.3V to 5V (logic level trigger)
- Load Capacity: Up to 250V AC (10A) or 30V DC (10A)
- Output: Normally Open (NO) and Normally Closed (NC) contacts
- Indicator: Built-in LED to indicate relay activation

Quantity: 1

Figure 2.3.2(c)Relay Module

4.Mini Water Pump 6V

The 6V mini water pump is used to supply water in the irrigation system. It operates efficiently in low-power setups and is controlled by the relay module based on soil moisture sensor readings, ensuring precise watering.

Specifications:

- Operating Voltage: 6V DC
- Power Consumption: Approximately 2W to 5W
- Flow Rate: 80 to 100 litres per hour (L/h)
- Lift Height: Up to 1.5 meters
- Input/Output: Plastic inlet and outlet nozzles for tubing connections

Quantity: 1

Figure 2.3.2(d) Mini Water Pump

5. Power Supply (9V Battery with Battery Clip)

The power supply provides the necessary voltage to the Arduino and other components. A 9V battery is commonly used to power the Arduino and the water pump.

Specifications:

Voltage: 9V

Battery Type: Standard 9V battery

Quantity: 1

Figure 2.3.2(e) Power Supply(9V Battery)

6. Jumper Wires

Jumper wires are used to make the necessary connections between the Arduino, ultrasonic sensor, servo motor, and power supply.

Specifications:

Male-to-male jumper wires for connecting to the Arduino pins and components.

Male-to-female jumper wires for connecting to the Arduino pins and components.

Quantity: Several (as required)

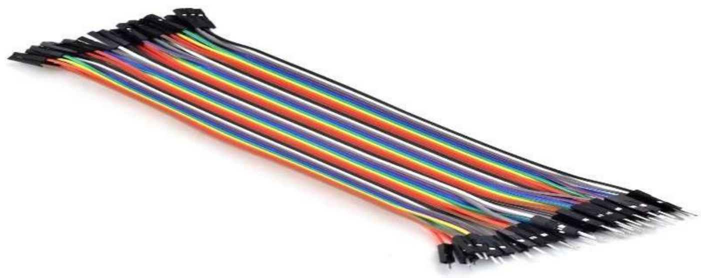


Figure 2.3.2(f) Male to Female jumper wires

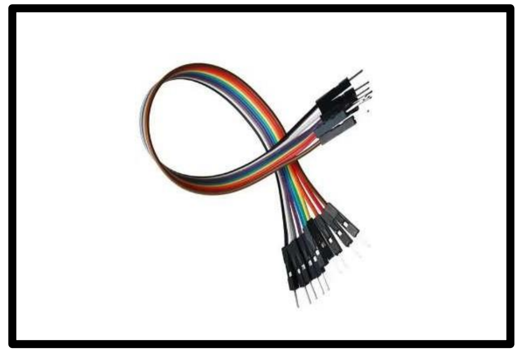


Figure 2.3.2(g)Male to Male jumper wires

2.4. Tools / Platforms:

Arduino Integrated Development Environment (IDE):

The Arduino IDE is essential for writing, compiling, and uploading code (sketches) to the Arduino board. It features a user-friendly interface that supports code editing, debugging, and managing libraries.

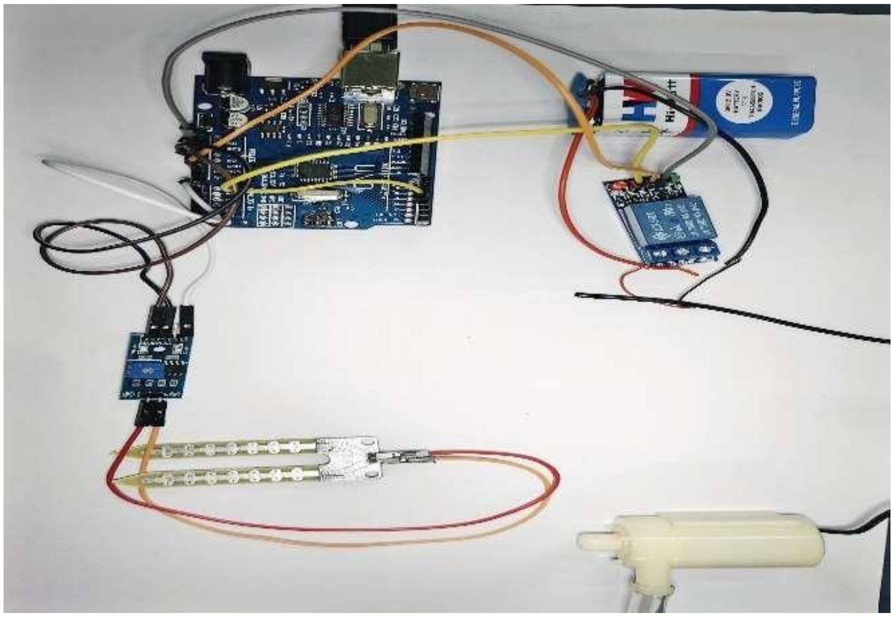
Documentation Tools:

Platforms like Google Docs, Microsoft Word are used to document the project details and to prepare the project report.

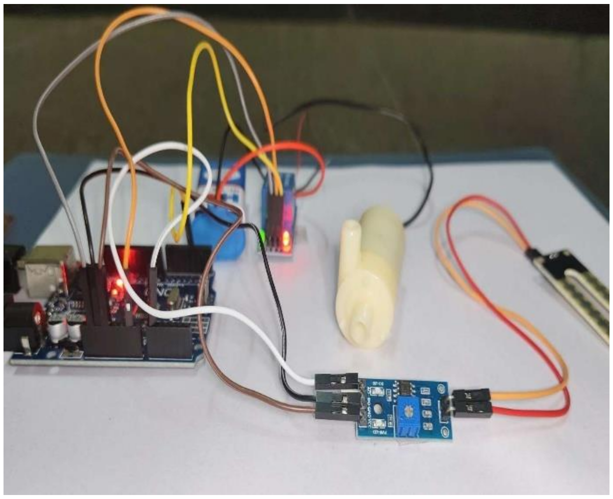
CHAPTER 3

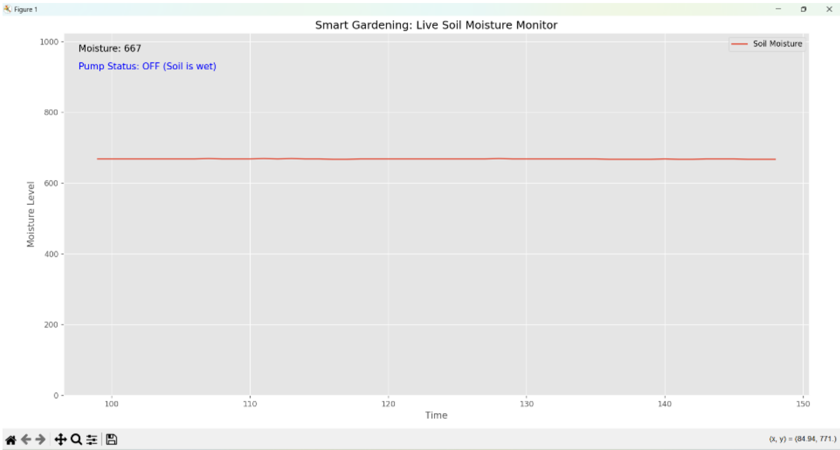
IMPLEMENTATION

3.1. Input Image:



3.3 Output images:





CHAPTER 4

CONCLUSION

A smart irrigation system that adapts to different soil types is a significant advancement in modern plant care and sustainable agriculture. By automating the watering process based on real-time soil moisture data and intelligently adjusting to various soil characteristics, the system ensures that plants receive the precise amount of water they need. This minimizes the risks of overwatering and underwatering, leading to healthier plants, optimized water usage, and reduced manual intervention.

Such systems are not only ideal for home gardens and indoor plants but also scalable for larger agricultural applications. The integration of adaptive features like soil-type detection and dynamic thresholding enhances the system's versatility, making it suitable for a wide range of environments and plant species. Moreover, by promoting water conservation and reducing human error, smart irrigation systems contribute to eco-friendly and sustainable gardening practices.

Overall, this solution combines efficiency, intelligence, and environmental responsibility—empowering both casual gardeners and professional farmers to maintain optimal plant health with minimal effort.

FUTURE SCOPE:

The smart irrigation system, enhanced with soil-type adaptability, holds immense potential for innovation and widespread application in agriculture, horticulture, and urban green spaces. Its integration with modern technologies can pave the way for more intelligent, efficient, and sustainable plant care solutions.

Here's an overview of its future scope:

1. Smart Agriculture

Precision Farming: By incorporating IoT-based soil sensors, weather data, and AI algorithms, the system can deliver highly accurate irrigation tailored to soil type, crop needs, and environmental conditions—enhancing yield and reducing resource waste.

Sustainable Farming Practices: As water scarcity becomes a critical global issue, soil-adaptive irrigation systems offer a reliable method to minimize water usage without compromising crop health, promoting sustainability on a large scale.

Crop- and Soil-Specific Customization: Future systems can be trained using machine learning to fine-tune irrigation strategies for specific crops and soil compositions, ensuring optimal growth and resource efficiency.

2. Urban Gardening and Smart Cities

Smart Green Infrastructure: The system can be deployed in rooftop gardens, vertical farms, community gardens, and indoor plant setups where space and water management are crucial. Its ability to adjust to different soil media makes it ideal for varied urban applications.

Smart Home Integration: With the integration of Wi-Fi or Bluetooth modules, these systems can connect to smart home ecosystems, enabling real-time monitoring and control via mobile apps, dashboards, or voice assistants like Alexa and Google Assistant.

Data Analytics and Predictive Maintenance: Future versions may include data logging and analytics features, allowing users to track soil health trends, predict irrigation needs, and receive maintenance alerts for better long-term performance.

In essence, the future of smart, soil-adaptive irrigation systems lies in their ability to combine environmental intelligence with automation, offering scalable solutions for both individual and commercial use while supporting global efforts toward sustainability and food security.

REFERENCES:

1. Y. Zhao, J. Zhang, J. Guan and W. Yin, "Study on precision water-saving irrigation automatic control system by plant physiology," 2009 4th IEEE Conference on Industrial Electronics and Applications, Xi'an, China, 2009, pp. 1296-1300, doi: 10.1109/ICIEA.2009.5138411. keywords: {Irrigation;Automatic control;Physiology;Communication system control;Data communication;Control systems;Biomedical monitoring;Soil;Research and development;Computerized monitoring;soil water;water-saving irrigation;automatic control}

2. M. Mayuree, P. Aishwarya and A. Bagubali, "Automatic Plant Watering System," 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 2019, pp. 1-3, doi: 10.1109/ViTECoN.2019.8899452. keywords: {Moisture;Pins;Water conservation;Switches;Soil moisture;Irrigation;Automatic plant watering system;Arduino UNO;Moisture sensor;Water level sensor;GSM module}

3. S. Bhardwaj, S. Dhir and M. Hooda, "Automatic Plant Watering System using IoT," 2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT), Bangalore, India, 2018, pp. 659-663, doi: 10.1109/ICGCIoT.2018.8753100. keywords: {arduino;automatic;soil moisture sensor;water pump;plant}

4. D. Oksana, L. Alexander, L. Nikolay and S. Sherali, "Increasing the small hydropower plants efficiency in Ukraine through the application of automatic control system modes," 2017 IEEE International Young Scientists Forum on Applied Physics and Engineering (YSF), Lviv, Ukraine, 2017, pp. 367-370, doi: 10.1109/YSF.2017.8126649. keywords: {Hydroelectric power generation;Power systems;Physics;Economics;Control systems;automatic system;operation mode;hydroelectric power plant;power;control action}

5. D. A. Aziz, R. Asgarnezhad and S. N. Mahmood, "The Recent Advances In IoT Based Smart Plant Irrigation Systems: A Brief Review," 2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 2021, pp. 97-104, doi: 10.1109/ISMSIT52890.2021.9604699. keywords: {Irrigation;Costs;Microcontrollers;Plants (biology);Crops;Sensor systems;Manufacturing;Smart Irrigation System;Automatic Plant Watering System;IoT based Monitoring System}

6. J. -H. Xu and A. -L. Luo, "Research on Water Resources Automatic Monitoring and Management System," 2012 Fourth International Conference on Computational and Information Sciences, Chongqing, China, 2012, pp. 1135-1138, doi: 10.1109/ICCIS.2012.265. keywords: {Monitoring;Water resources;Databases;Graphics;Real time systems;Fingerprint recognition;Business;water quality monitoring;hydrological monitoring;real-time communication;data management}

7. S. Prasad, K. Umme Haniya, S. Zeenathunnisa, M. Tahir and M. Sulaiman, "Deep learning based Automatic Indoor Plant Management System using Arduino," 2023 4th IEEE Global Conference for Advancement in Technology (GCAT), Bangalore, India, 2023, pp. 1-5, doi: 10.1109/GCAT59970.2023.10353399. keywords: {Automation;Plants (biology);Moisture;Real-time systems;Sensor systems;Water pumps;Monitoring;Smart Agriculture;Plant Management System;Arduino;Deep learning}

8. R. Parepalli, S. Sreejith, A. Talari, M. Divya, G. Mohan and C. Meghana, "IoT and AI Based Smart Plant Watering System," 2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2024, pp. 1-6, doi: 10.1109/ACCAI61061.2024.10601880. keywords: {Temperature sensors;Temperature measurement;Machine learning algorithms;Soil moisture;Water conservation;Valves;Resource management;AI;Machine learning;IoT;Smart plant watering;Kodular;Python;Sensors;Hardware;Sustainable}

9. W. Wongthai, S. Chanmee and S. Lohawet, "An Enhancement of an Automatic Plant Watering System," 2018 22nd International Computer Science and Engineering Conference (ICSEC), Chiang Mai, Thailand, 2018, pp. 1-4, doi: 10.1109/ICSEC.2018.8712791. keywords: {Moisture;Sensors;Internet of Things;Cloud computing;Soil moisture;Relays;Cloud Computing;Watering System;Monitoring system;Microcontroller;IoT applications},

APPENDIX:

ARDUINO CODE:

```
const int moistureSensorPin = A0;
const int relayPin = 7;
int sandyThreshold = 500;
int loamyThreshold = 400;
int clayeyThreshold = 300;
int selectedThreshold = 400;
bool pumpOn = false;
String soilType = "Unknown";
void setup() {
  Serial.begin(9600);
  pinMode(moistureSensorPin, INPUT);
  pinMode(relayPin, OUTPUT);
  digitalWrite(relayPin, HIGH);
  delay(2000);
  int initialMoisture = analogRead(moistureSensorPin);
  Serial.print("Initial Soil Moisture: ");
  Serial.println(initialMoisture);
```

```
if (initialMoisture < 350) {
    soilType = "Clayey";
    selectedThreshold = clayeyThreshold;
  } else if (initialMoisture >= 350 && initialMoisture <= 450) {
    soilType = "Loamy";
    selectedThreshold = loamyThreshold;
  } else {
    soilType = "Sandy";
    selectedThreshold = sandyThreshold;
  }
  Serial.print("Detected Soil Type: ");
  Serial.println(soilType);
  Serial.print("Using Threshold: ");
  Serial.println(selectedThreshold);
}
void loop() {
  int moisture = analogRead(moistureSensorPin);
  Serial.print("Moisture: ");
  Serial.println(moisture);
  if (moisture < selectedThreshold && !pumpOn) {
    digitalWrite(relayPin, LOW);
    pumpOn = true;
    Serial.println("Pump ON - Soil is dry");
  }
```

```
  else if (moisture >= selectedThreshold && pumpOn) {
    digitalWrite(relayPin, HIGH);
    pumpOn = false;
    Serial.println("Pump OFF - Soil is wet");
  }
  delay(1000);
}
```

PYTHON(FLASK) CODE:

```
import serial
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
SERIAL_PORT = 'COM5'
BAUD_RATE = 9600
ser = serial.Serial(SERIAL_PORT, BAUD_RATE)
moisture_values = []
time_stamps = []
plt.style.use('ggplot')
fig, ax = plt.subplots()
x_data, y_data = [], []
pump_status = ""
threshold = 500
def animate(i):
    global pump_status
    if ser.in_waiting:
```

```
    try:
        line = ser.readline().decode('utf-8').strip()
        value = int(line)
        moisture_values.append(value)
        time_stamps.append(len(moisture_values))
        x_data.append(len(x_data))
        y_data.append(value)
        # Determine pump status
        pump_status = "ON (Soil is dry)" if value < threshold else "OFF (Soil is wet)"
        ax.clear()
        ax.plot(x_data[-50:], y_data[-50:], label='Soil Moisture')
        ax.set_ylim(0, 1023)
        ax.set_xlabel("Time")
        ax.set_ylabel("Moisture Level")
        ax.set_title("Smart Gardening: Live Soil Moisture Monitor")
        ax.legend()
        ax.text(0.02, 0.95, f"Moisture: {value}", transform=ax.transAxes, fontsize=12)
        ax.text(0.02, 0.90, f"Pump Status: {pump_status}", transform=ax.transAxes, fontsize=12,
color='blue')
    except:
        pass
ani = FuncAnimation(fig, animate, interval=1000)
plt.tight_layout()
plt.show()
```