PATUAKHALI SCIENCE AND TECHNOLOGY UNIVERSITY

COURSE CODE: EEE-212
Course Title: Electrical Technology Sessional

SUBMITTED TO:

Md. Naimur Rahman
Professor
Department of Electrical and Electronics Engineering
Faculty of Computer Science and Engineering

SUBMITTED BY:

Shaid Ibna Sobhan
ID: **2102057**
Registration No: **10184**
Faculty of Computer Science and Engineering

Date of submission: 29-10-2024
Project title: Automatic Plant Watering System

Automatic Plant Watering System

Introduction

The Automatic Plant Watering System is an innovative solution designed to automate plant watering based on soil moisture, environmental temperature, and weather conditions. This system is particularly useful for individuals with busy lifestyles, ensuring plants receive adequate water only when necessary. Equipped with sensors and controlled by an Arduino microcontroller, the system efficiently manages water supply to maintain optimal soil moisture, conserve water, and minimize human intervention. By considering temperature, rain, and soil moisture levels, this system promotes healthy plant growth in a sustainable way.

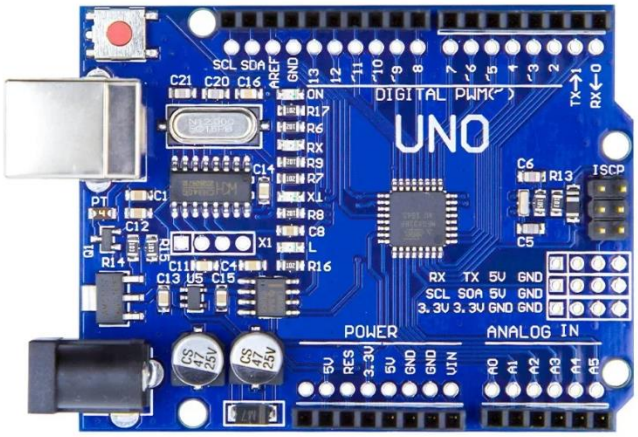
Objectives

The primary objectives of this project are:

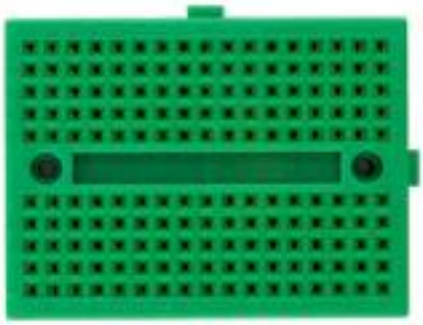
1. To automate plant watering based on soil moisture and environmental factors.
2. To initiate watering based on high temperature conditions, supporting plants in hot climates.
3. To prevent over-watering by disabling the water pump during rain.
4. To create an efficient and user-friendly solution for home gardening and small-scale agricultural applications.

Components Used: Description and Role in Project

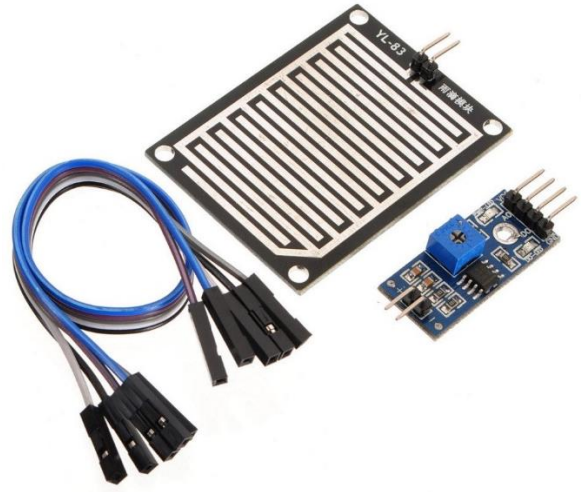
1. **Arduino Uno R3**
    - **Description:** The Arduino Uno R3 is a microcontroller board based on the ATmega328P. It is widely used for prototyping and electronic projects due to its ease of programming and versatility.
    - **Role:** Acts as the central controller for the system, receiving data from the sensors and making decisions about when to activate or deactivate the water pump based on predefined conditions.



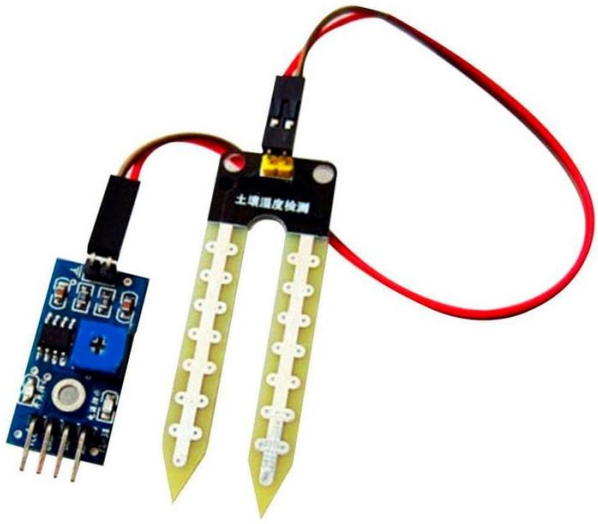
2. **Mini Breadboard (170 Tie Point)**
    - **Description:** A mini breadboard is used to create temporary circuit connections without the need for soldering.
    - **Role:** Provides a platform to connect components, sensors, and jumper wires, allowing for quick adjustments and testing in the circuit.



3. **Rain Sensor Module**
    - **Description:** The rain sensor module consists of a water-detecting board and a control board. It senses the presence of water and sends a signal to the Arduino.
    - **Role:** Prevents the water pump from operating during rainy conditions, helping to avoid over-watering and conserve water.



4. **Soil Moisture Sensor**
    - **Description:** A sensor that measures the volumetric water content of the soil. It has two probes that go into the soil to detect moisture levels.
    - **Role:** Measures the moisture level in the soil. If the soil is dry (below a certain threshold), the Arduino triggers the water pump to supply water.



5. **DHT11 Humidity/Temperature Sensor Module**
    - **Description:** The DHT11 sensor is a low-cost digital sensor that provides temperature and humidity readings.

- **Role:** Monitors the environmental temperature, which is used to activate the water pump if temperatures exceed a set limit, thereby ensuring plants are hydrated in hot weather.

6. **Jumper Wires (Male-to-Male, Male-to-Female, Female-to-Female)**
   - **Description:** Flexible wires used for making connections between the breadboard, Arduino, and other components.
   - **Role:** Connects all the components to the Arduino and the breadboard, allowing signal transfer between sensors and the controller.

7. **LCD 1602 with I2C Module**
   - **Description:** A 16x2 LCD display with an I2C module for simpler connections and reduced wiring. It displays data in a readable format.

- **Role:** Shows real-time data for soil moisture, temperature, and humidity, allowing the user to monitor the plant's environmental conditions directly.

8. 5V Water Pump with Pipe

- **Description:** A small DC-powered water pump that operates on 5V and can transport water through a connected pipe.
- **Role:** Supplies water to the plant when activated by the relay, based on moisture or temperature readings from the sensors.

9. 5V 1 Channel Relay Module

- **Description:** A relay module that acts as a switch, allowing the Arduino to control high-power devices like the water pump using a low-power signal.
- **Role:** Acts as the switch for the water pump, enabling or disabling water flow to the plant based on signals from the Arduino.

10. 5 mm PVC Sheet (1 sq ft)

- **Description:** A piece of polyvinyl chloride sheet used as a base for mounting and organizing components.
- **Role:** Provides a stable and insulated platform for the system components, ensuring they remain secure and organized.
-

11. **Double-Side Foam Tape**

- **Description:** Adhesive tape with foam in the middle, used for mounting components to surfaces without screws.
- **Role:** Holds the sensors, Arduino, and other components in place on the PVC sheet, ensuring a compact and stable setup for the system.

Components Cost

Component	Quantity	Price (৳)
Arduino Uno R3	1	**720.00**
Mini Breadboard (170 Tie Point)	1	**40.00**
Rain Sensor Module	1	**85.00**
Soil Moisture Sensor	1	**75.00**
DHT11 Humidity/Temperature Sensor Module	1	**135.00**
Male to Male/Male to Female/Female to Female Jumper Wires	20	**40.00**
LCD 1602 with I2C Module	1	**320.00**

5V Water Pump with Pipe	1	**100.00**
5V 1 Channel Relay Module	1	**70.00**
5 mm PVC (1 sq ft)	1	**65.00**
Double-Side Foam Tape	1	**60.00**
Total		**1,710.00**

System Design and Working

System Design

The system is built around an Arduino Uno R3, which collects data from a soil moisture sensor, temperature sensor (DHT11), and rain sensor. These sensors determine if and when watering is necessary based on:

1. Soil moisture levels (to determine if the soil is dry).
2. Temperature levels (to activate the pump when temperatures are high).
3. Rain detection (to prevent watering during rain).

When conditions indicate the need for water, the Arduino triggers a relay to start the water pump, and stops it once the conditions are met or if rain is detected.

Working Principle

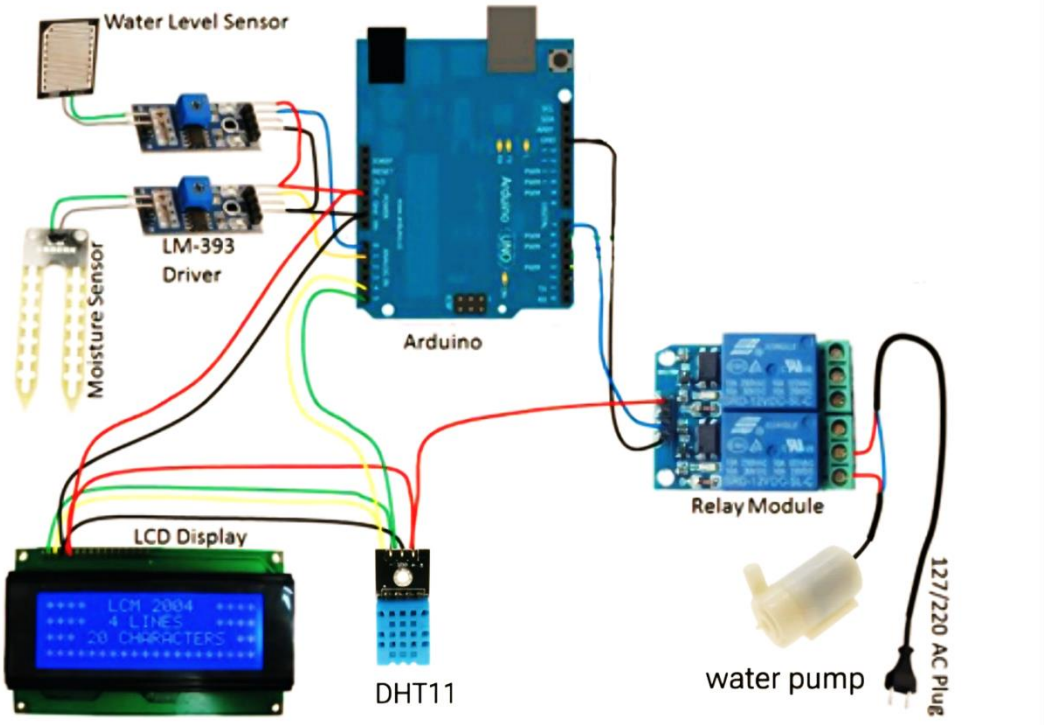
1. **Soil Moisture Detection**: The soil moisture sensor continuously monitors soil moisture. If the moisture level is below a set threshold, the Arduino will consider activating the pump.
2. **Temperature Detection**: The DHT11 sensor monitors the surrounding temperature. When the temperature exceeds a set high threshold, the system activates the water pump, helping to cool and support plants under heat stress.
3. **Rain Detection**: The rain sensor detects precipitation, preventing the system from watering during rain to avoid over-watering.

4. **Water Pump Activation**: If either the temperature is high or the soil moisture is low, and there is no rain detected, the relay module triggers the pump, allowing water to flow to the plant until the desired conditions are restored.
5. **User Display**: An LCD display shows real-time data on temperature, humidity, and soil moisture, enabling easy monitoring of the plant's environment.

Circuit Diagram

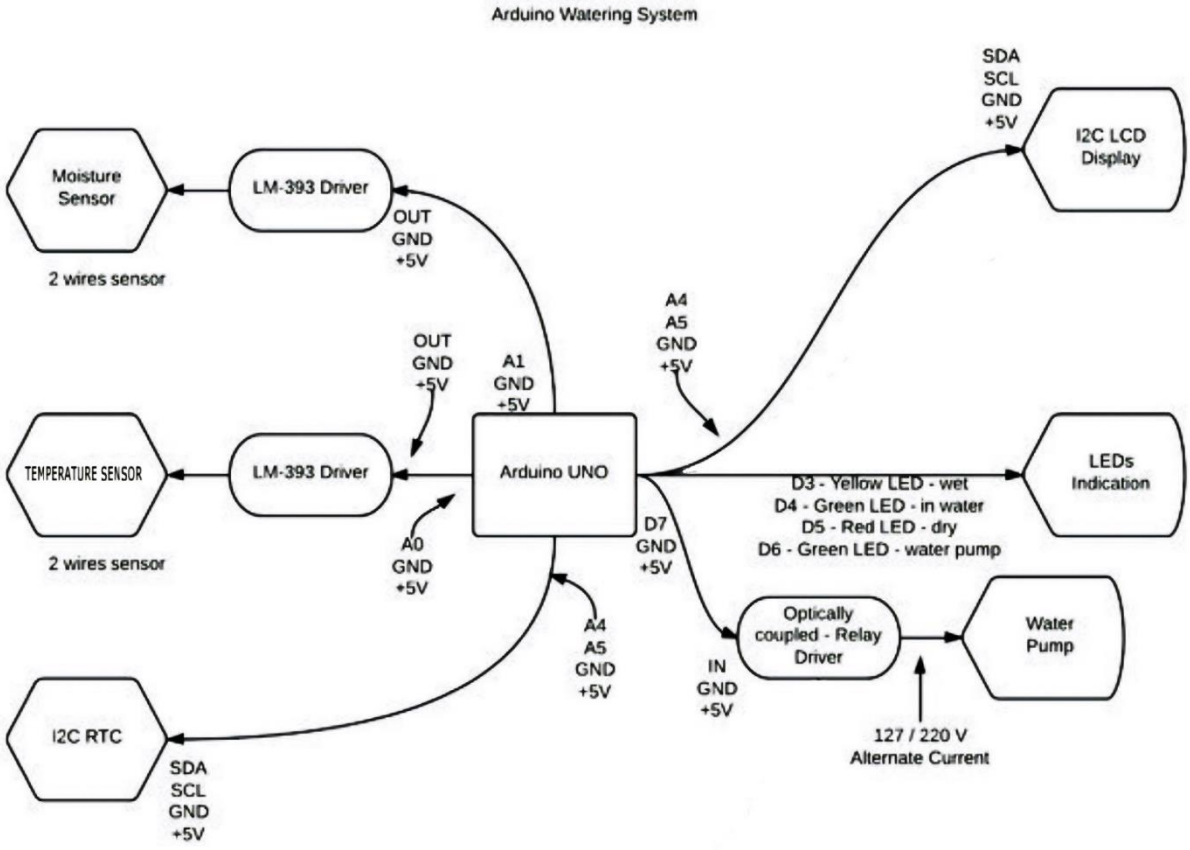
The circuit includes:

- **Sensors**: Connected to the Arduino for monitoring soil, temperature, and rain.
- **Relay and Pump**: Controlled by the Arduino based on sensor input.
- **LCD Module**: Displays current temperature, humidity, and soil moisture levels.



Data Flow Diagram

An automatic plant watering system DFD illustrates the flow of data from a soil moisture sensor to a controller, which processes the data and, if needed, activates a water pump to irrigate the plant. The sensor continuously monitors soil moisture, creating a feedback loop to maintain ideal moisture levels.



Program Code

The code for an automatic plant watering system on an Arduino Uno reads soil moisture levels from a sensor. If the soil is dry, the Arduino activates a water pump to water the plant. This process repeats, with the Arduino monitoring and adjusting the water flow to maintain optimal moisture. Here the code:

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

#include <DHT.h>  // Include the DHT sensor library


// LCD Setup

LiquidCrystal_I2C lcd(0x27, 16, 2);  // Initialize LCD with I2C address 0x27


// DHT11 Sensor Setup

#define DHTPIN 2       // DHT11 Data Pin connected to Arduino Pin 2

#define DHTTYPE DHT11  // DHT11 sensor type

DHT dht(DHTPIN, DHTTYPE);  // Initialize DHT sensor


// Relay and Moisture Sensor Setup

int relayPin = 8;                 // Relay module for water pump

int moistureSensorPin = A0;          // Soil Sensor input at Analog PIN A0

int outputValue = 0;                // Stores soil moisture level

int dryThreshold = 30;              // Moisture level below which pump turns on (percentage)
```

```
int wetThreshold = 70;              // Moisture level above which pump turns off (percentage)
bool isPumping = false;             // Track the pump status (ON or OFF)

// Raindrop Sensor Setup
int rainSensorPin = A3;             // Raindrop sensor pin
int rainThreshold = 300;            // Value below which rain is detected
void setup() {
  Serial.begin(9600);
  pinMode(relayPin, OUTPUT);
  digitalWrite(relayPin, HIGH);   // Ensure the relay is off initially (active LOW)
  // Initialize the LCD
  lcd.begin(16, 2);  // Initialize LCD for 16x2 characters
  lcd.backlight();    // Turn on LCD backlight
  lcd.setCursor(0, 0);
  lcd.print("Initializing...");
  // Initialize the DHT11 sensor
  dht.begin();
  Serial.println("Initializing system...");
  delay(2000);  // Initial delay for system readiness
  lcd.clear();  // Clear LCD after initialization message
}

// Function to read and map soil moisture sensor values
```

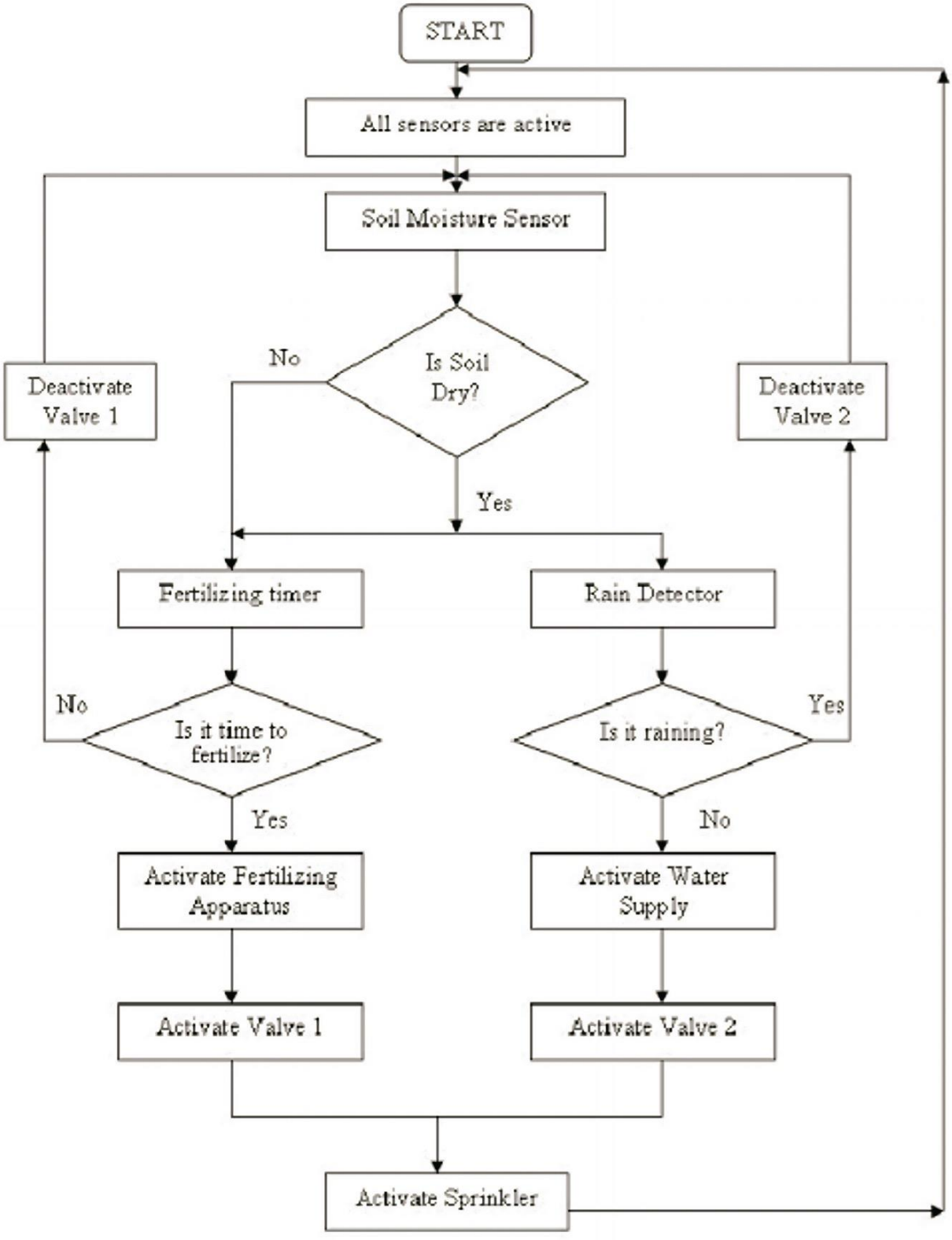
```
int readMoistureLevel() {
  int sensorValue = analogRead(moistureSensorPin);
  // Map the sensor value from 550 (dry) to 10 (wet) into a percentage (0 to 100)
  int moistureLevel = map(sensorValue, 550, 10, 0, 100);
  // Ensure the moisture level is clamped within the range 0 to 100%
  moistureLevel = constrain(moistureLevel, 0, 100);
  return moistureLevel;
}
// Function to control the water pump based on moisture level
void controlPump(int moistureLevel) {
  if (moistureLevel < dryThreshold) {
    if (!isPumping) {  // Only turn on if not already pumping
      digitalWrite(relayPin, LOW);  // Turn ON the pump (active LOW relay)
      isPumping = true;
      Serial.println("Watering the plant (Pump ON)...");
      lcd.setCursor(0, 1);
      lcd.print("Pump ON");
    }
  }
  else if (moistureLevel > wetThreshold) {
    if (isPumping) {  // Only turn off if it's currently on
      digitalWrite(relayPin, HIGH);  // Turn OFF the pump
      isPumping = false;
```

```
      Serial.println("Pump OFF (Soil moist)");
      lcd.setCursor(0, 1);
      lcd.print("Pump OFF");
    }
  }
}
// Function to update the LCD
void updateLCD(int moistureLevel, float temperature, float humidity, bool isRaining) {
  // Display moisture level on the first line
  lcd.setCursor(0, 0);
  lcd.print("Moist:");
  lcd.print(moistureLevel);
  lcd.print("%");  // Add spaces to clear previous content

  if (isRaining) {
    lcd.print("|Rain ");
  } else {
    lcd.print("NoRain");
  }
  // Display temperature and rain status on the second line
  lcd.setCursor(0, 1);
  lcd.print("T: ");
  lcd.print(int(temperature));
  lcd.print("C ");
```

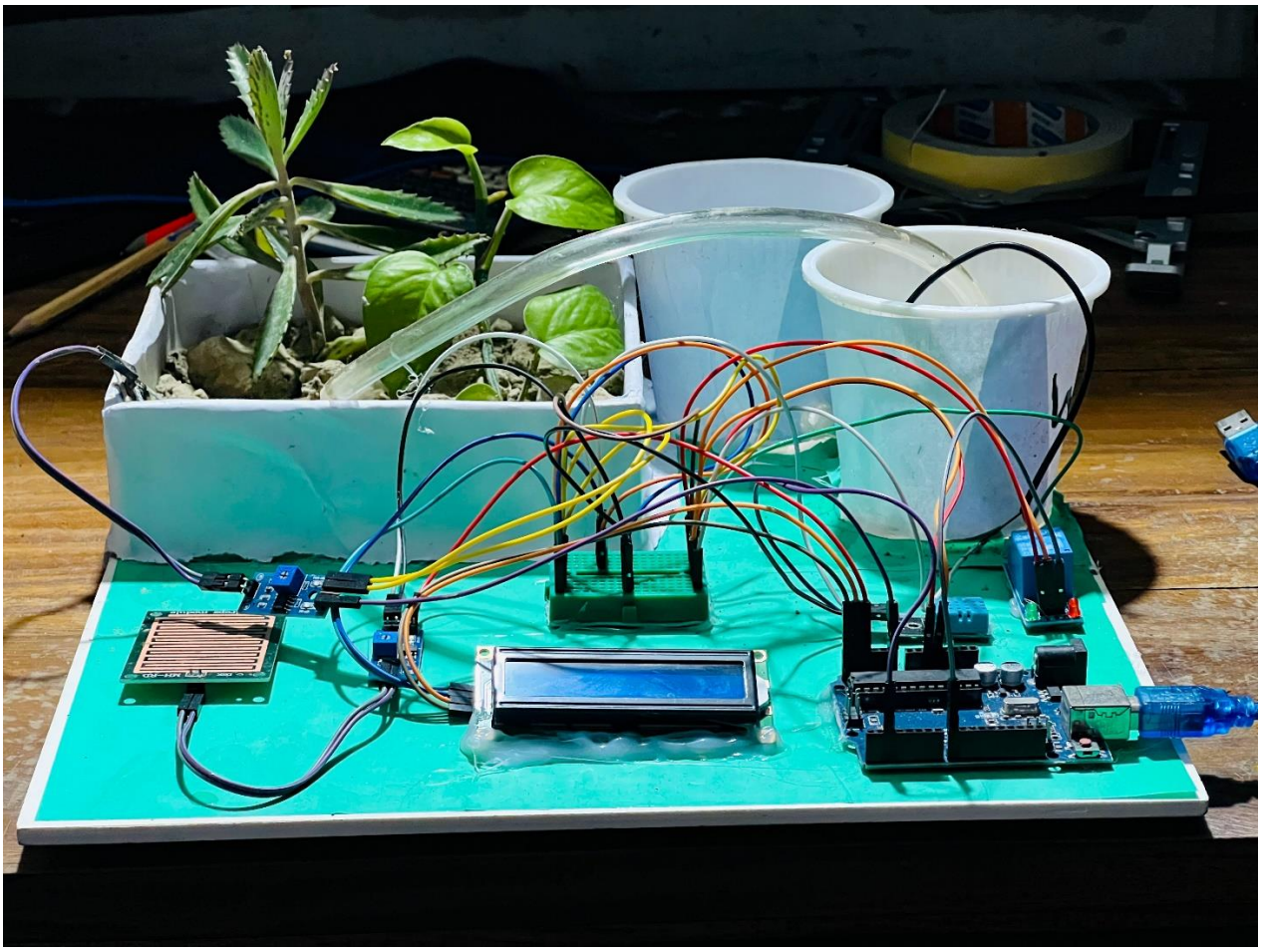
```
}
void loop() {
  // Read current moisture level
  outputValue = readMoistureLevel();

  // Read DHT11 sensor values
  float temperature = dht.readTemperature();  // Read temperature in Celsius
  float humidity = dht.readHumidity();        // Read humidity in percentage

  // Check if the readings are valid
  if (isnan(temperature) || isnan(humidity)) {
    Serial.println("Failed to read from DHT sensor!");
    lcd.setCursor(0, 1);
    lcd.print("Sensor error!    ");
    return;
  }

  // Check rain status
  int rainValue = analogRead(rainSensorPin);
  bool isRaining = (rainValue < rainThreshold);

  // Print the moisture level to the Serial Monitor
  Serial.print("Moisture level: ");
```

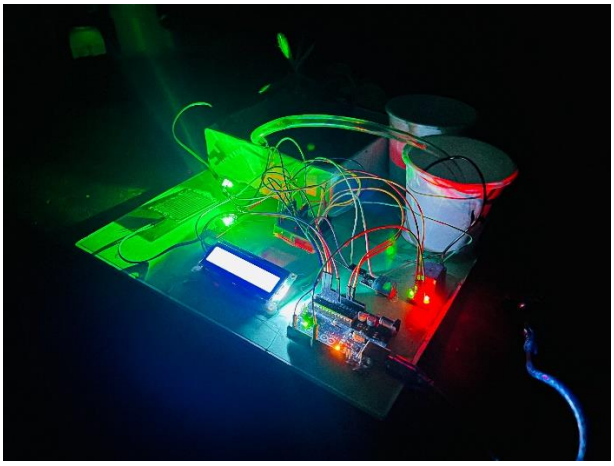
```
  Serial.print(outputValue);
  Serial.println("%");

  // Print temperature, humidity, and rain status to Serial Monitor
  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.print(" C, Humidity: ");
  Serial.print(humidity);
  Serial.print("%, Rain detected: ");
  Serial.println(isRaining ? "Yes" : "No");

  // Update the LCD with current values
  updateLCD(outputValue, temperature, humidity, isRaining);

  // Control the water pump based on current moisture level
  controlPump(outputValue);

  delay(5000);  // Check every 5 seconds to avoid excessive sensor polling
}
```

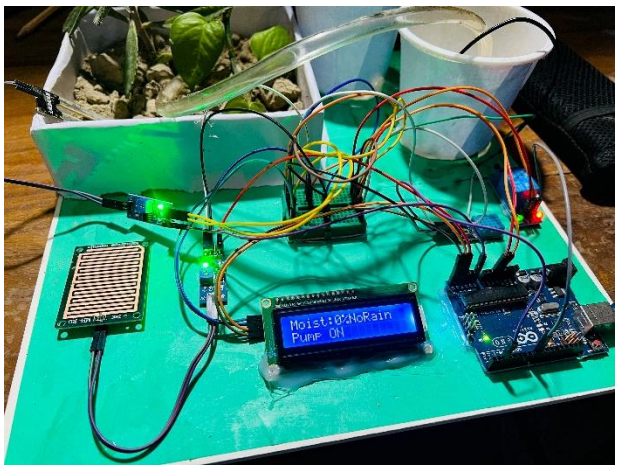
Flowchart


START
All sensors are active
Soil Moisture Sensor
Is Soil Dry?
No
Yes
Deactivate Valve 1
Deactivate Valve 2
Fertilizing timer
Rain Detector
Is it time to fertilize?
No
Yes
Is it raining?
Yes
No
Activate Fertilizing Apparatus
Activate Water Supply
Activate Valve 1
Activate Valve 2
Activate Sprinkler

Main Illustration







Results and Observations

The Automatic Plant Watering System demonstrated the following results:

1. **Optimized Water Usage**: Water was dispensed only when necessary, based on moisture levels and temperature, conserving water effectively.
2. **High Temperature Activation**: The system successfully activated the water pump under high-temperature conditions, providing relief to the plants during hot weather.
3. **Rain Prevention**: The rain sensor accurately detected rain, preventing the pump from operating during rainfall, thus preventing over-watering.
4. **Real-Time Monitoring**: The LCD displayed accurate real-time data on temperature, humidity, and soil moisture, allowing easy monitoring of environmental conditions.

Observations

- The system operated reliably under different conditions, responding effectively to changes in soil moisture and temperature.
- The rain sensor performed well, accurately disabling the water pump during rain.
- The temperature sensor provided an additional layer of control, aiding plant health during high-temperature periods.

Future Improvements

1. **Mobile App Integration**: Enabling remote monitoring and control via a smartphone app.
2. **Wireless Communication**: Adding Wi-Fi or Bluetooth to allow remote access and control of the system.

3. **Solar Power Addition**: Integrating solar power to make the system more sustainable and energy efficient.
4. **Advanced Sensor Network**: Adding sensors for multiple plants or for monitoring other environmental factors such as soil pH and light exposure.

Conclusion

The Automatic Plant Watering System is a versatile and reliable solution for maintaining plant health. By using real-time data from soil moisture, temperature, and rain sensors, the system efficiently manages water supply, minimizing waste and supporting plant growth. Its temperature-based watering feature is especially beneficial in hot climates, ensuring plants receive adequate hydration even in extreme weather. Future enhancements could further extend its functionality and usability, making it an ideal solution for sustainable gardening and small-scale agriculture.

Alhamdulillah, this project effectively demonstrates the potential of automation in plant care, offering a low-cost and practical method to maintain optimal plant health with minimal human intervention.