Automated Irrigation control system based on environmental sensing

TEAM MEMBERS:

Hilary Royson CB

Nitish Kumar J

Allan christ B

Ahash J

AIM:

To design and develop an automated irrigation system using ESP8266 that intelligently controls watering by continuously monitoring soil moisture, temperature, humidity, rainfall, water level, and flow rate. This ensures optimal water usage and helps in conserving resources while maintaining healthy plant growth..

COMPONENTS REQUIRED:

- ESP-8266 (Microcontroller)

SENSORS:

- Soil Moisture sensor
- DHT22
- Rain sensor
- Water level sensor
- Water flow sensor (YFS201)

ACTUATORS:

- Solenoid Valve
- Relay module

ESP-8266:

The ESP8266 is a low-cost, energy-efficient microcontroller with built-in Wi-Fi capabilities. It is widely used in Internet of Things (IoT) projects, enabling remote control and monitoring. In this system, it reads sensor data and triggers watering actions based on real-time environmental conditions.

Soil Moisture Sensor:

This sensor measures the water content in the soil by detecting its conductivity or resistance. It helps the system decide when watering is necessary to maintain optimal soil conditions for plant growth. It prevents both overwatering and underwatering, ensuring efficient water usage.

DHT22 (Temperature & Humidity Sensor):

The DHT22 sensor provides accurate readings of temperature and humidity in the environment. These factors influence plant health and watering schedules, so integrating this sensor helps the system adjust irrigation patterns according to weather conditions.

Rain Sensor:

The rain sensor detects the presence of rainfall by sensing water droplets on its surface. By identifying rainy conditions, the system can stop irrigation, conserving water and avoiding unnecessary watering during wet weather.

Water Level Sensor:

This sensor monitors the level of water in the tank or reservoir, ensuring that there is enough supply for irrigation. It helps prevent the system from operating when water is insufficient, protecting pumps and valves from damage.

Water Flow Sensor (YFS201):

The water flow sensor measures how much water is passing through the pipes by counting pulses generated by the sensor’s internal turbine. It ensures accurate water delivery and helps monitor usage for better resource management.

Solenoid Valve:

A solenoid valve controls the flow of water by opening or closing when triggered by an electrical signal. It is a crucial actuator that allows the system to start or stop watering automatically without human intervention.

Relay Module:

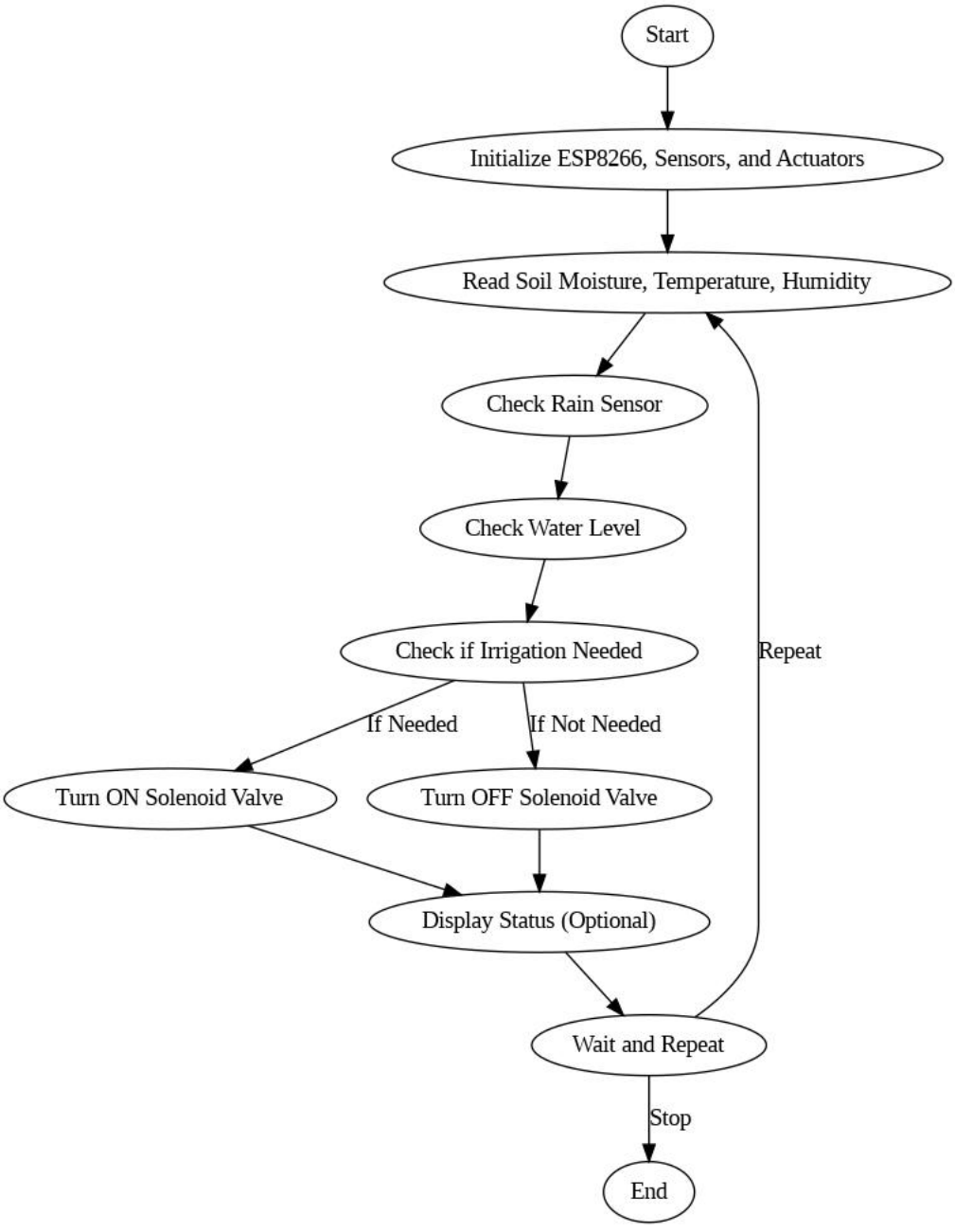
The relay module acts as a safe switch between the low-power ESP8266 and the high-power devices like the pump or valve. It protects the microcontroller from electrical spikes and ensures reliable control over irrigation hardware.

Pin Table:

Component	Pin on Component	ESP8266 Pin	Description
Soil Moisture Sensor	VCC	3.3V or 5V	Power supply
Soil Moisture Sensor	GND	GND	Ground connection
Soil Moisture Sensor	Analog Output	A0 (ADC)	Reads moisture data
DHT22 Sensor	VCC	3.3V or 5V	Power supply
DHT22 Sensor	GND	GND	Ground connection
DHT22 Sensor	DATA	GPIO 4	Reads temperature/humidity
Rain Sensor	VCC	3.3V or 5V	Power supply
Rain Sensor	GND	GND	Ground connection

Circuit Connection:

Flowchart:


Start
Initialize ESP8266, Sensors, and Actuators
Read Soil Moisture, Temperature, Humidity
Check Rain Sensor
Check Water Level
Check if Irrigation Needed
Repeat
If Needed
If Not Needed
Turn ON Solenoid Valve
Turn OFF Solenoid Valve
Display Status (Optional)
Wait and Repeat
Stop
End

Coding:

```
#include <DHT.h>

// Define Pins
#define SOIL_MOISTURE_PIN A0
#define DHT_PIN 4
#define RAIN_SENSOR_PIN 5
#define WATER_LEVEL_PIN 14
#define WATER_FLOW_PIN 12
#define RELAY_PIN 13

// Define Constants
#define SOIL_THRESHOLD 500    // Adjust based on soil calibration
#define WATER_LEVEL_THRESHOLD 300  // Adjust based on tank level

// Initialize DHT22
#define DHTTYPE DHT22
DHT dht(DHT_PIN, DHTTYPE);

void setup() {
  Serial.begin(115200);

  // Initialize pins
  pinMode(SOIL_MOISTURE_PIN, INPUT);
  pinMode(RAIN_SENSOR_PIN, INPUT);
  pinMode(WATER_LEVEL_PIN, INPUT);
  pinMode(WATER_FLOW_PIN, INPUT);
  pinMode(RELAY_PIN, OUTPUT);
```

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

  // Initially turn off relay
  digitalWrite(RELAY_PIN, LOW);

  Serial.println("Automated Irrigation System Initialized");
}

void loop() {
  // Read Soil Moisture
  int soilMoisture = analogRead(SOIL_MOISTURE_PIN);

  // Read Temperature and Humidity
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();

  // Read Rain Sensor
  int rain = digitalRead(RAIN_SENSOR_PIN);

  // Read Water Level
  int waterLevel = analogRead(WATER_LEVEL_PIN);

  // Read Water Flow Sensor (for monitoring only)
  int waterFlow = pulseIn(WATER_FLOW_PIN, HIGH);

  // Display readings
```

```
  Serial.print("Soil Moisture: "); Serial.println(soilMoisture);
  Serial.print("Temperature: "); Serial.println(temperature);
  Serial.print("Humidity: "); Serial.println(humidity);
  Serial.print("Rain Detected: "); Serial.println(rain ? "Yes" : "No");
  Serial.print("Water Level: "); Serial.println(waterLevel);
  Serial.print("Water Flow (pulse duration): "); Serial.println(waterFlow);

  // Check conditions
  bool soilDry = soilMoisture > SOIL_THRESHOLD;
  bool enoughWater = waterLevel > WATER_LEVEL_THRESHOLD;
  bool raining = rain == HIGH;

  if (soilDry && enoughWater && !raining) {
    Serial.println("Irrigation Started");
    digitalWrite(RELAY_PIN, HIGH); // Turn ON solenoid valve
  } else {
    Serial.println("Irrigation Stopped");
    digitalWrite(RELAY_PIN, LOW); // Turn OFF solenoid valve
  }

  // Wait before next reading
  delay(5000);
}
```

Execution: