# 3.Smart Garden

## Project Description: -

The **Smart Garden System** is an automated solution designed to optimize plant care by monitoring and controlling essential environmental factors. This project integrates multiple sensors and actuators to **maintain ideal soil moisture, track light intensity, and regulate water levels**, ensuring efficient plant growth with minimal human intervention.

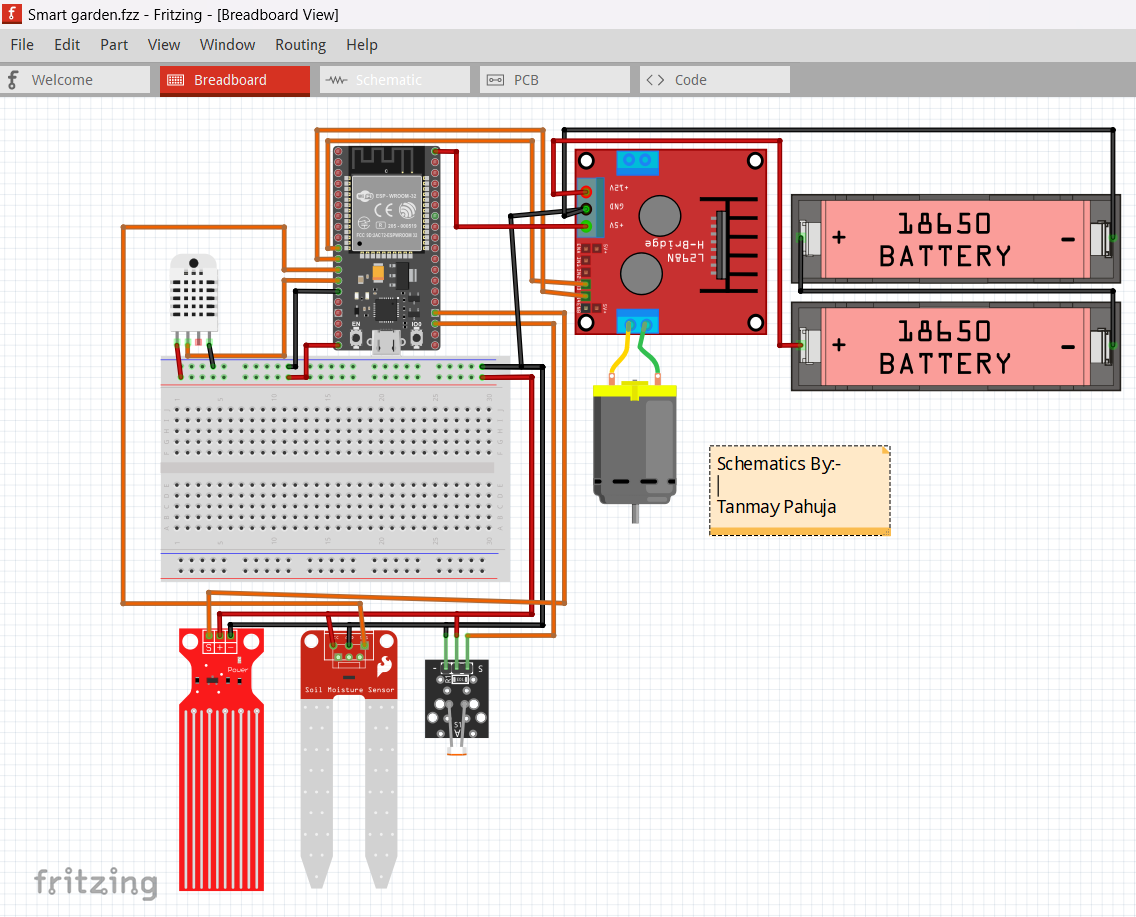
The system uses an **Arduino** as the central controller, which collects real-time data from a **soil moisture sensor, DHT22 temperature & humidity sensor, LDR module (light detection), and a water tank level sensor**. Based on the sensor readings, the **L298N motor driver** controls water pumps for irrigation.

This project enhances **resource efficiency**, reduces water wastage, and automates plant care, making gardening more convenient and sustainable. The system ensures that plants receive water only when needed, preventing overwatering or underwatering. Additionally, the LDR module helps monitor sunlight exposure, allowing for optimal placement of plants or potential integration with a shading mechanism. The water tank level sensor ensures a steady supply of water, preventing pump damage due to dry running. With its modular design, the system can be easily expanded for larger gardens or customized for specific plant needs.

## Apparatus: -

| **S. No.** | **Component** | **Specification** | **Quantity** |
| --- | --- | --- | --- |
| 1 | ESP-32 | Bluetooth, WIFI enabled board | 1 |
| 2 | L298N Motor Driver | Dual H-Bridge for pump/motor control | 1 |
| 3 | DHT22 Sensor | Temperature & Humidity Sensor | 1 |
| 4 | Soil Moisture Sensor | Analog moisture detection | 1 |
| 5 | LDR Module | Light intensity detection | 1 |
| 6 | Water Tank Level Sensor | Float or conductivity-based | 1 |
| 7 | Water Pump | 5V/12V pump for irrigation | 1 |
| 8 | Relay Module (Optional) | For pump control (if needed) | 1 |
| 9 | Jumper Wires | Male-to-male, male-to-female | As needed |
| 10 | Power Supply | 5V/12V based on pump & Arduino needs | 1 |
| 11 | Breadboard | For prototyping connections | 1 |
| 12 | Tubing & Connectors | For water flow system | As needed |

## Connection Diagram: -



**1. ESP32 to L298N Motor Driver (Pump Control)**

| **ESP32 Pin** | **L298N Pin** | **Function** |
| --- | --- | --- |
| 26 | IN1 | Motor/Pump Control 1 |
| 27 | IN2 | Motor/Pump Control 2 |
| GND | GND | Common Ground |
| 5V | VCC | Power Supply |
| 12V (External) | Motor VCC | Power for Pump |

**2. ESP32 to DHT22 Sensor (Temperature & Humidity Monitoring)**

| **ESP32 Pin** | **DHT22 Pin** | **Function** |
| --- | --- | --- |
| 12 | Data | Reads temperature & humidity |
| 3.3V | VCC | Power Supply |
| GND | GND | Common Ground |

**3. ESP32 to Soil Moisture Sensor**

| **ESP32 Pin** | **Soil Sensor Pin** | **Function** |
| --- | --- | --- |
| 14 | Analog Out | Reads soil moisture level |
| 3.3V | VCC | Power Supply |
| GND | GND | Common Ground |

**4. ESP32 to LDR Module (Light Detection)**

| **ESP32 Pin** | **LDR Pin** | **Function** |
| --- | --- | --- |
| 8 | Analog Out | Reads light intensity |
| 3.3V | VCC | Power Supply |
| GND | GND | Common Ground |

**5. ESP32 to Water Tank Level Sensor**

| **ESP32 Pin** | **Water Sensor Pin** | **Function** |
| --- | --- | --- |
| 15 | Analog Out | Detects water level |
| 3.3V | VCC | Power Supply |
| GND | GND | Common Ground |

## Code: -

#include <DHT.h>

#include <Wire.h>

#include <BluetoothSerial.h> // Include Bluetooth Serial library

// Pin Definitions

#define DHT\_PIN 12          // Pin connected to DHT22 sensor

#define DHT\_TYPE DHT22      // Specify the DHT sensor type (DHT22)

#define SOIL\_MOISTURE\_PIN 14 // Analog pin for Soil Moisture Sensor

#define LDR\_PIN 27           // Analog pin for LDR (Light Dependent Resistor) Module

#define WATER\_LEVEL\_PIN 15   // Analog pin for Water Tank Level Sensor

#define L298N\_IN1 26         // IN1 pin for L298N Motor Driver

#define L298N\_IN2 25         // IN2 pin for L298N Motor Driver

// Initialize DHT Sensor

DHT dht(DHT\_PIN, DHT\_TYPE);

// Initialize Bluetooth Serial

BluetoothSerial SerialBT;

// Thresholds for Sensor Readings

const int soilMoistureDry = 4000;   // Above this value: Dry soil

const int soilMoistureGood = 3800;  // Below this value: Good moisture

const int sunlightLow = 80;         // Above this value: Low sunlight

const int sunlightHigh = 20;        // Below this value: Excessive sunlight

const int waterLevelLow = 40;       // Below this value: Low water level

const int pumpDelay = 5000;         // Duration (ms) to run the water pump when activated

void setup() {

  // Initialize Serial Monitor and Bluetooth communication

  Serial.begin(115200);             // For debugging via Serial Monitor

  SerialBT.begin("SmartGarden");    // Start Bluetooth communication with the name "SmartGarden"

  // Initialize DHT Sensor

  dht.begin();

  // Set pin modes for L298N motor driver

  pinMode(L298N\_IN1, OUTPUT);

  pinMode(L298N\_IN2, OUTPUT);

  pinMode(L298N\_ENA, OUTPUT);

  // Ensure pump is OFF initially

  digitalWrite(L298N\_IN1, LOW);

  digitalWrite(L298N\_IN2, LOW);

  analogWrite(L298N\_ENA, 0);

  // Notify user that the setup is complete

  Serial.println("Smart Garden Bluetooth Ready");

  SerialBT.println("Smart Garden Bluetooth Ready");

}

void loop() {

  // Read temperature and humidity from DHT22 sensor

  float temperature = dht.readTemperature();

  float humidity = dht.readHumidity();

  // Read raw data from soil moisture, LDR, and water level sensors

  int rawSoilMoisture = analogRead(SOIL\_MOISTURE\_PIN);

  int rawSunlight = analogRead(LDR\_PIN);

  int rawWaterLevel = analogRead(WATER\_LEVEL\_PIN);

  // Determine soil moisture status based on updated range

  String soilMoistureStatus;

  if (rawSoilMoisture > soilMoistureDry) {

    soilMoistureStatus = "Dry Soil";  // Very low moisture

  } else if (rawSoilMoisture > soilMoistureGood) {

    soilMoistureStatus = "Moderate Moisture"; // Medium moisture

  } else {

    soilMoistureStatus = "Good Moisture"; // Adequate moisture

  }

  // Determine sunlight status

  String sunlightStatus;

  if (rawSunlight > sunlightLow) {

    sunlightStatus = "Low Sunlight"; // Too little sunlight

  } else if (rawSunlight < sunlightHigh) {

    sunlightStatus = "Excessive Sunlight"; // Too much sunlight

  } else {

    sunlightStatus = "Good Sunlight"; // Optimal sunlight

  }

  // Determine water level status

  String waterLevelStatus = (rawWaterLevel < waterLevelLow) ? "Low Water Level" : "Normal Water Level";

  // Prepare data to send via Bluetooth

  String data = "Temperature: " + String(temperature, 1) + "C, "

                + "Humidity: " + String(humidity, 1) + "%, "

                + "Soil: " + soilMoistureStatus + " (" + String(rawSoilMoisture) + "), "

                + "Sunlight: " + sunlightStatus + " (" + String(rawSunlight) + "), "

                + "Water: " + waterLevelStatus + " (" + String(rawWaterLevel) + ")";

  // Send data over Bluetooth and print to Serial Monitor

  SerialBT.println(data);

  Serial.println(data);

  delay(2000); // Wait for 2 seconds before sending data again

  // Check soil moisture and water level to decide pump activation

  if (rawSoilMoisture > soilMoistureDry && rawWaterLevel > waterLevelLow) {

    // If soil is too dry and there's enough water in the tank

    SerialBT.println("Activating Water Pump: Watering Plants");

    Serial.println("Activating Water Pump: Watering Plants");

    // Turn on pump via L298N

    digitalWrite(L298N\_IN1, HIGH);

    digitalWrite(L298N\_IN2, LOW);

    analogWrite(L298N\_ENA, 255); // Set motor speed to maximum

    delay(pumpDelay);            // Keep the pump running for the specified duration

    digitalWrite(L298N\_IN1, LOW);

    digitalWrite(L298N\_IN2, LOW);

    analogWrite(L298N\_ENA, 0);  // Turn off pump

  } else if (rawWaterLevel <= waterLevelLow) {

    // If water level in the tank is low

    SerialBT.println("Warning: Low Water Level");

    Serial.println("Warning: Low Water Level");

    // Ensure pump stays off when water level is low

    digitalWrite(L298N\_IN1, LOW);

    digitalWrite(L298N\_IN2, LOW);

    analogWrite(L298N\_ENA, 0);

  }

}

## Project Outcome: -

The **Smart Garden System** effectively automates plant care by integrating **sensors and actuators** to monitor and control environmental conditions. By using an ESP32, the system enables real-time data collection and decision-making, ensuring optimal plant health with minimal user intervention. This automation makes gardening more efficient, reducing manual effort and improving plant growth conditions.

One of the key achievements of the system is **automated irrigation**. The soil moisture sensor continuously monitors the moisture level in the soil, and when it drops below a predefined threshold, the system **activates the water pump** to hydrate the plants. This ensures that plants receive adequate water without overwatering, which can cause root rot, or underwatering, which can lead to dehydration.

Additionally, the **DHT22 sensor monitors temperature and humidity**, helping users understand the microclimate surrounding their plants. This data can be used to make informed decisions about shading, ventilation, or additional watering requirements. The **LDR module detects light intensity**, ensuring plants receive the necessary sunlight for photosynthesis. If light levels are consistently low, the gardener may need to reposition the plants or use artificial lighting to support growth.

The system also incorporates a **water tank level sensor** to prevent the pump from running dry. This feature ensures **continuous operation without damage to the pump**, reducing the need for frequent manual checks. The use of the **L298N motor driver** efficiently controls the water pump, allowing smooth and reliable operation based on sensor readings.

Overall, the **Smart Garden System improves resource efficiency** by optimizing water usage, reducing waste, and enhancing plant health. The modular design allows for **scalability**, making it adaptable to different garden sizes, greenhouses, or even hydroponic setups. With its **real-time monitoring and automation**, this system serves as a practical and sustainable solution for both hobbyist gardeners and agricultural applications.