**GreenSense IoT Environmental Monitoring System - Documentation**

**Overview**

**This document outlines the architecture and functioning of the GreenSense IoT environmental monitoring system. The system uses an ESP32S board to monitor various environmental parameters, including temperature, humidity, soil moisture, and light intensity, while also integrating external weather data from the OpenWeatherMap API. The data is sent to the Blynk app for visualization and interaction. This system is designed for remote plant health monitoring, offering smart insights on plant growth conditions.**

**Components Used**

* **ESP32S Development Board**
* **DHT11 Temperature and Humidity Sensor**
* **LDR (Light Dependent Resistor) for Light Intensity Measurement**
* **Soil Moisture Sensor**
* **OpenWeatherMap API for external weather data**
* **Blynk Mobile App for data visualization and interaction**

**Circuit Connections**

**A. DHT11 Sensor (Temperature and Humidity)**

* **Pin 1 (VCC): Connected to 3.3V on ESP32**
* **Pin 2 (Data): Connected to GPIO 4 (DHTPIN)**
* **Pin 3 (NC): Not connected**
* **Pin 4 (GND): Connected to ESP32 GND**

**B. LDR (Light Intensity)**

* **One leg connected to GPIO 35 (LDR pin)**
* **Other leg connected to GND with a pull-down resistor (optional, depending on LDR type)**

**C. Soil Moisture Sensor**

* **Signal Pin: Connected to GPIO 34 (SensorPin)**
* **Power Pin (VCC): Connected to 3.3V on ESP32**
* **Ground Pin (GND): Connected to ESP32 GND**

**D. External Weather Data**

* **API call is made to OpenWeatherMap using HTTPClient to fetch weather data.**

**ESP32 Code Snippet**

**#define BLYNK\_TEMPLATE\_ID "TMPL39MdTsO6M"**

**#define BLYNK\_TEMPLATE\_NAME "temp and moisture"**

**#define BLYNK\_AUTH\_TOKEN "1wC0NR6K-5t6gWeepgxvJYwEJG2e3b3z"**

**#define BLYNK\_PRINT Serial**

**#include <WiFi.h>**

**#include <HTTPClient.h>**

**#include <BlynkSimpleEsp32.h>**

**#include <DHT.h>**

**#include <ArduinoJson.h>**

**char auth[] = BLYNK\_AUTH\_TOKEN;**

**char ssid[] = "lop"; // your WiFi name**

**char pass[] = "12345678"; // your WiFi password**

**BlynkTimer timer;**

**#define DHTPIN 4**

**#define DHTTYPE DHT11**

**DHT dht(DHTPIN, DHTTYPE);**

**#define LIGHT\_SENSOR\_PIN 35 // LDR connected to GPIO35**

**#define SensorPin 34 // Soil moisture sensor pin**

**const int dryValue = 2910;**

**const int wetValue = 1465;**

**const String apiKey = "your\_api\_key"; // <-- Replace this**

**const String city = "Bangalore";**

**const String country = "IN";**

**String weatherURL = "http://api.openweathermap.org/data/2.5/weather?q=" + city + "," + country + "&appid=" + apiKey + "&units=metric";**

**// Variables for calculating daily light average**

**unsigned long lightSum = 0;**

**unsigned int lightSamples = 0;**

**void sendSensor() {**

**int moistureRaw = analogRead(SensorPin);**

**int soilMoisturePercent = map(moistureRaw, dryValue, wetValue, 0, 100);**

**float h = dht.readHumidity();**

**float t = dht.readTemperature();**

**int lightRaw = analogRead(LIGHT\_SENSOR\_PIN);**

**int lightPercent = map(lightRaw, 0, 4095, 0, 100); // Assuming 0-3.3V range**

**lightSum += lightPercent;**

**lightSamples++;**

**if (isnan(h) || isnan(t)) {**

**Serial.println("Failed to read from DHT sensor!");**

**return;**

**}**

**Blynk.virtualWrite(V0, soilMoisturePercent);**

**Blynk.virtualWrite(V1, t);**

**Blynk.virtualWrite(V2, h);**

**Blynk.virtualWrite(V6, lightPercent); // LDR data**

**Serial.print("Soil Moisture: "); Serial.print(soilMoisturePercent);**

**Serial.print(" Temp: "); Serial.print(t);**

**Serial.print(" Humidity: "); Serial.print(h);**

**Serial.print(" Light: "); Serial.println(lightPercent);**

**}**

**void getWeatherData() {**

**if (WiFi.status() == WL\_CONNECTED) {**

**HTTPClient http;**

**http.begin(weatherURL);**

**int httpCode = http.GET();**

**if (httpCode > 0) {**

**String payload = http.getString();**

**StaticJsonDocument<1024> doc;**

**DeserializationError error = deserializeJson(doc, payload);**

**if (!error) {**

**float temp = doc["main"]["temp"];**

**float humidity = doc["main"]["humidity"];**

**const char\* description = doc["weather"][0]["description"];**

**Blynk.virtualWrite(V3, description);**

**Serial.println("External Weather Data:");**

**Serial.print("Temperature: "); Serial.println(temp);**

**Serial.print("Humidity: "); Serial.println(humidity);**

**Serial.print("Condition: "); Serial.println(description);**

**} else {**

**Serial.println("JSON parse error");**

**}**

**} else {**

**Serial.println("HTTP request failed");**

**}**

**http.end();**

**}**

**}**

**void sendDailyLightAverage() {**

**if (lightSamples > 0) {**

**int avgLight = lightSum / lightSamples;**

**Blynk.virtualWrite(V7, avgLight); // Daily light average**

**Serial.print("Daily Average Light: ");**

**Serial.println(avgLight);**

**}**

**// Reset for the next day**

**lightSum = 0;**

**lightSamples = 0;**

**}**

**void setup() {**

**Serial.begin(115200);**

**Blynk.begin(auth, ssid, pass);**

**dht.begin();**

**timer.setInterval(15000L, sendSensor); // Send sensor data every 15 seconds**

**timer.setInterval(60000L, getWeatherData); // Get weather data every minute**

**timer.setInterval(86400000L, sendDailyLightAverage); // Send daily light average every 24 hours (86400000 ms)**

**}**

**void loop() {**

**Blynk.run();**

**timer.run();**

**}**

**Optional Smart Features**

1. **Low Battery Warning:**
   * **Alert when battery voltage falls below a certain threshold (e.g., 3.3V) via Blynk or LED indicator.**
2. **Daylight Detection (Solar Active):**
   * **Detect sunlight presence when solar panel voltage exceeds a threshold (e.g., 3.5V), indicating that solar energy is available for charging.**
3. **Data Transmission Optimization:**
   * **Transmit sensor data only when solar power is available, conserving battery for low-light conditions.**
4. **Power Saving Modes:**
   * **Enter deep sleep mode if battery voltage is low and solar charging is not available, to extend the device’s operation time.**
5. **Long-Term Logging:**
   * **Store sensor data (e.g., temperature, humidity, moisture levels, light) on an SD card or send it to the cloud for future analysis.**

**Notes**

* **Ensure the ESP32 has a stable WiFi connection for fetching weather data from the OpenWeatherMap API.**
* **The LDR and soil moisture sensors may need calibration depending on the environment.**
* **To extend battery life, consider using solar power and/or a low-power mode during inactivity.**

**This design offers real-time plant monitoring via Blynk while integrating environmental conditions from OpenWeatherMap, providing useful insights into plant health and ensuring energy efficiency.**