#include <ESP8266WiFi.h>

#include <ESP8266WebServer.h>

#define PIN\_ENA D5 // The ESP8266 pin connected to the EN1 pin L298N for Motor 1

#define PIN\_IN1 D6 // The ESP8266 pin connected to the IN1 pin L298N for Motor 1

#define PIN\_IN2 D7 // The ESP8266 pin connected to the IN2 pin L298N for Motor 1

#define PIN\_ENB D1 // The ESP8266 pin connected to the EN2 pin L298N for Motor 2

#define PIN\_IN3 D2 // The ESP8266 pin connected to the IN3 pin L298N for Motor 2

#define PIN\_IN4 D3 // The ESP8266 pin connected to the IN4 pin L298N for Motor 2

#define RAIN\_SENSOR\_PIN A0

#define RAIN\_THRESHOLD 500

// WiFi credentials

const char\* ssid = "OPPO";

const char\* password = "gwapako123";

ESP8266WebServer server(80);

bool automationEnabled = false; // Flag to control the rain sensor automation

bool isRaining() {

int rainValue = analogRead(RAIN\_SENSOR\_PIN);

return rainValue > RAIN\_THRESHOLD;

}

void setup() {

Serial.begin(9600);

// Connect to WiFi

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting to WiFi...");

}

Serial.println("Connected to WiFi");

Serial.print("IP Address: ");

Serial.println(WiFi.localIP());

// Initialize digital pins as outputs for Motor 1

pinMode(PIN\_IN1, OUTPUT);

pinMode(PIN\_IN2, OUTPUT);

pinMode(PIN\_ENA, OUTPUT);

// Initialize digital pins as outputs for Motor 2

pinMode(PIN\_IN3, OUTPUT);

pinMode(PIN\_IN4, OUTPUT);

pinMode(PIN\_ENB, OUTPUT);

// Set default motor direction and speed for Motor 1

digitalWrite(PIN\_IN1, HIGH); // Forward direction

digitalWrite(PIN\_IN2, LOW);

analogWrite(PIN\_ENA, 0); // Initially stopped

// Set default motor direction and speed for Motor 2

digitalWrite(PIN\_IN3, HIGH); // Forward direction

digitalWrite(PIN\_IN4, LOW);

analogWrite(PIN\_ENB, 0); // Initially stopped

// Define server routes

server.on("/", handleRoot);

server.on("/forward", handleForward);

server.on("/backward", handleBackward);

server.on("/stop", handleStop);

server.on("/toggle-automation", handleToggleAutomation);

server.begin();

Serial.println("HTTP server started.");

}

void loop() {

server.handleClient();

// Check rain status every 5 seconds if automation is enabled

if (automationEnabled) {

static unsigned long lastRainCheckTime = 0;

static bool lastRainStatus = false; // To track previous rain status

if (millis() - lastRainCheckTime >= 5000) {

lastRainCheckTime = millis();

bool currentRainStatus = isRaining();

// If rain status has changed

if (currentRainStatus != lastRainStatus) {

if (currentRainStatus) {

// If it starts raining, move forward (close roof)

handleBackward();

delay(5000); // Move forward for 5 seconds

handleStop();

} else {

// If it stops raining, move backward (open roof)

handleForward();

delay(5000); // Move backward for 5 seconds

handleStop();

}

// Update last rain status

lastRainStatus = currentRainStatus;

}

}

}

}

void handleRoot() {

String rainStatus = isRaining() ? "Raining" : "Not Raining";

String automationStatus = automationEnabled ? "Enabled" : "Disabled";

// Break the HTML content into smaller parts and concatenate them

String page = "<!DOCTYPE html><html lang=\"en\"><head><meta charset=\"UTF-8\"><meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0\">";

page += "<title>Smart Roof System</title><style>body {font-family: Arial, sans-serif; display: flex; justify-content: center; align-items: center; height: 100vh; margin: 0; background-color: #f5f5f5;}";

page += ".box {padding: 2em; border: 1px solid #ccc; border-radius: 5px; background-color: #87CEEB; box-shadow: 0 0 10px rgba(0, 0, 0, 0.1); text-align: center;}";

page += "h1 {margin: 0 0 1em 0;} button {font-size: 1.2em; padding: 0.5em 1em; margin: 0.5em; cursor: pointer; border: none; border-radius: 5px; color: white;}";

page += "#forward {background-color: yellow;} #stop {background-color: red;} #backward {background-color: gray;} #automation {background-color: blue;}";

page += "button:hover {background-color: #45a049;} #display {margin: 1em auto; font-size: 1.2em; width: 330px; padding: 10px; border: 1px solid #ccc; border-radius: 5px; background-color: #00FF00;}</style></head>";

page += "<body><div class=\"box\"><h1>Smart Roof System</h1>";

page += "<button id=\"forward\">Close Roof</button><button id=\"stop\">Stop</button><button id=\"backward\">Open Roof</button>";

page += "<button id=\"automation\">" + String(automationEnabled ? "Disable" : "Enable") + " Automation</button>";

page += "<div id=\"display\">Smart Roof System: Inactive<br>Rain Sensor Status: " + rainStatus + "<br>Automation: " + automationStatus + "</div></div>";

page += "<script>const forwardButton = document.getElementById('forward');const stopButton = document.getElementById('stop');const backwardButton = document.getElementById('backward');const automationButton = document.getElementById('automation');";

page += "forwardButton.addEventListener('click', () => { fetch('/forward'); });stopButton.addEventListener('click', () => { fetch('/stop'); });";

page += "backwardButton.addEventListener('click', () => { fetch('/backward'); });automationButton.addEventListener('click', () => { fetch('/toggle-automation'); location.reload(); });";

page += "</script></body></html>";

server.send(200, "text/html", page);

}

void handleForward() {

digitalWrite(PIN\_IN1, HIGH); // Forward direction

digitalWrite(PIN\_IN2, LOW);

analogWrite(PIN\_ENA, 255); // Full speed for Motor 1

digitalWrite(PIN\_IN3, HIGH); // Forward direction

digitalWrite(PIN\_IN4, LOW);

analogWrite(PIN\_ENB, 255); // Full speed for Motor 2

server.send(200, "text/html", "<html><body><h1>Roof Closing</h1><p><a href='/'>Back to Home</a></p></body></html>");

}

void handleBackward() {

digitalWrite(PIN\_IN1, LOW); // Backward direction

digitalWrite(PIN\_IN2, HIGH);

analogWrite(PIN\_ENA, 255); // Full speed for Motor 1

digitalWrite(PIN\_IN3, LOW); // Backward direction

digitalWrite(PIN\_IN4, HIGH);

analogWrite(PIN\_ENB, 255); // Full speed for Motor 2

server.send(200, "text/html", "<html><body><h1>Roof Opening</h1><p><a href='/'>Back to Home</a></p></body></html>");

}

void handleStop() {

analogWrite(PIN\_ENA, 0); // Stop Motor 1

analogWrite(PIN\_ENB, 0); // Stop Motor 2

server.send(200, "text/html", "<html><body><h1>Roof Stopped</h1><p><a href='/'>Back to Home</a></p></body></html>");

}

void handleToggleAutomation() {

automationEnabled = !automationEnabled; // Toggle the automation flag

server.send(200, "text/html", "<html><body><h1>Automation " + String(automationEnabled ? "Enabled" : "Disabled") + "</h1><p><a href='/'>Back to Home</a></p></body></html>");

}