#include <ESP8266WiFi.h>

#include <WiFiUdp.h>

#include <coap-simple.h>

#define LED\_PIN D0      // Built-in LED for ESP8266 (GPIO2 on NodeMCU)

#define PIR\_PIN D1     // PIR sensor pin (GPIO5 on NodeMCU), adjust as needed

// Wi-Fi credentials

const char\* ssid = "Cs";          // Replace with your Wi-Fi SSID

const char\* password = "249242727"; // Replace with your Wi-Fi password

// UDP and CoAP class

WiFiUDP udp;

Coap coap(udp);

// LED state

bool ledState = false;

// Observer: Ubuntu machine's IP & port (replace with your actual IP and port)

IPAddress observerIP(192, 168, 183, 129); // e.g., your Ubuntu VM IP

int observerPort = 5683;                  // CoAP port

// ------------------- LED Resource Callback ------------------- //

void callback\_led(CoapPacket &packet, IPAddress ip, int port) {

  Serial.println("[LED] Received a CoAP request");

  // Log the packet code (GET or PUT)

  Serial.print("Packet Code: ");

  if (packet.code == COAP\_GET) {

    Serial.println("GET");

  } else if (packet.code == COAP\_PUT) {

    Serial.println("PUT");

  } else {

    Serial.println("UNKNOWN");

  }

  // Log the sender's IP and port

  Serial.print("Sender IP: ");

  Serial.println(ip);

  Serial.print("Sender Port: ");

  Serial.println(port);

  // If PUT, handle payload

  if (packet.code == COAP\_PUT) {

    char payload[packet.payloadlen + 1];

    memcpy(payload, packet.payload, packet.payloadlen);

    payload[packet.payloadlen] = '\0'; // Null-terminate the payload

    Serial.print("Payload received: ");

    Serial.println(payload);

    String message(payload);

    if (message == "1") {

      digitalWrite(LED\_PIN, HIGH);

      ledState = true;

      coap.sendResponse(ip, port, packet.messageid, "LED is ON");

    } else if (message == "0") {

      digitalWrite(LED\_PIN, LOW);

      ledState = false;

      coap.sendResponse(ip, port, packet.messageid, "LED is OFF");

    } else {

      coap.sendResponse(ip, port, packet.messageid, "Invalid command");

    }

    Serial.println("Processed PUT request with payload: " + message);

  }

  // If GET, handle response

  if (packet.code == COAP\_GET) {

    String response = ledState ? "LED is ON" : "LED is OFF";

    coap.sendResponse(ip, port, packet.messageid, response.c\_str());

    Serial.println("Sent LED status: " + response);

  }

}

// ------------------- PIR Resource Callback ------------------- //

// Only handling GET here. If you want to handle PUT or other methods, modify as needed.

void callback\_pir(CoapPacket &packet, IPAddress ip, int port) {

  Serial.println("[PIR] Received a CoAP request");

  if (packet.code == COAP\_GET) {

    // Read the PIR sensor

    int pirVal = digitalRead(PIR\_PIN);  // 1 if motion detected, 0 otherwise

    // Convert to "1" or "0"

    String response = pirVal == HIGH ? "1" : "0";

    // Send the response

    coap.sendResponse(ip, port, packet.messageid, response.c\_str());

    Serial.print("Sent PIR state: ");

    Serial.println(response);

  } else {

    // Method not allowed for anything except GET

    coap.sendResponse(ip, port, packet.messageid, "Method Not Allowed");

  }

}

void setup() {

  Serial.begin(115200);

  // Initialize the LED (built-in)

  pinMode(LED\_PIN, OUTPUT);

  digitalWrite(LED\_PIN, LOW);

  // Initialize the PIR sensor pin

  pinMode(PIR\_PIN, INPUT);

  // Connect to Wi-Fi

  Serial.print("Connecting to Wi-Fi");

  WiFi.begin(ssid, password);

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

    delay(500);

    Serial.print(".");

  }

  Serial.println("\nWiFi connected");

  Serial.print("ESP IP address: ");

  Serial.println(WiFi.localIP());

  // Register the CoAP endpoint for /led

  coap.server(callback\_led, "led");

  // Register the CoAP endpoint for /pir

  coap.server(callback\_pir, "pir");

  // Start the CoAP server

  coap.start();

  Serial.println("CoAP server started");

  // On startup, send the initial LED state to the observer

  String response = ledState ? "1" : "0";

  coap.put(observerIP, observerPort, "led", response.c\_str());

  Serial.println("Sent initial LED state to observer: " + response);

  // Optionally send an initial PIR state to the observer

  String pirState = digitalRead(PIR\_PIN) ? "1" : "0";

  coap.put(observerIP, observerPort, "pir", pirState.c\_str());

  Serial.println("Sent initial PIR state to observer: " + pirState);

}

void loop() {

  // Keep the CoAP server running

  coap.loop();

  // Example: send periodic PUT requests to the CoAP server with LED & PIR states

  static unsigned long lastSendTime = 0;

  if (millis() - lastSendTime > 5000) { // Every 5 seconds

    lastSendTime = millis();

    // Send LED state ("1" or "0") to the observer

    String payload = ledState ? "1" : "0";

    coap.put(observerIP, observerPort, "led", payload.c\_str());

    Serial.print("Sent LED status to CoAP server: ");

    Serial.println(payload);

    // Send PIR state ("1" or "0") to the observer

    String pirPayload = digitalRead(PIR\_PIN) ? "1" : "0";

    coap.put(observerIP, observerPort, "pir", pirPayload.c\_str());

    Serial.print("Sent PIR status to CoAP server: ");

    Serial.println(pirPayload);

  }

}