#define BLYNK\_TEMPLATE\_ID "TMPL6kWN92xgM"

#define BLYNK\_TEMPLATE\_NAME "Automated Air purifier"

#define BLYNK\_AUTH\_TOKEN "29-TfEOHXuD37x\_ERtbiYVxHfZMiodqj"

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <Adafruit\_Sensor.h>

#include <DHT.h>

#include <BlynkSimpleEsp32.h>

#include <WiFi.h>

// Define the pins for the DHT22 sensor

#define DHTPIN 2  // Replace with the actual pin connected to DHT22

#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);

LiquidCrystal\_I2C lcd(0x27, 16, 2);  // 0x27 is the I2C address of the LCD

const int potPin = 34;  // Replace with the actual pin connected to the potentiometer

const int ledPin = 4;   // Replace with the actual pin connected to the LED

char ssid[] = "Wokwi-GUEST";

char pass[] = "";

BlynkTimer timer;

void sendData() {

  // Read temperature and humidity from the DHT22 sensor

  float temperature = dht.readTemperature();

  float humidity = dht.readHumidity();

  // Read gas value from the potentiometer

  int gasValue = analogRead(potPin);

  // Send data to Blynk

  Blynk.virtualWrite(V1, temperature);

  Blynk.virtualWrite(V2, humidity);

  Blynk.virtualWrite(V3, gasValue);

}

void displayMessage(String line1, String line2, int delayTime = 2000) {

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print(line1);

  lcd.setCursor(0, 1);

  lcd.print(line2);

  delay(delayTime);

}

void setup() {

  // Initialize the LCD

  lcd.init();

  lcd.backlight();

  // Initialize DHT sensor

  dht.begin();

  // Initialize the LED pin

  pinMode(ledPin, OUTPUT);

  // Connect to Wi-Fi

  WiFi.begin(ssid, pass);

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

    delay(250);

  }

  // Initialize Blynk

  Blynk.begin(BLYNK\_AUTH\_TOKEN, ssid, pass);

  // Map virtual pins

  Blynk.virtualWrite(V1, 0); // Initialize with 0

  Blynk.virtualWrite(V2, 0); // Initialize with 0

  Blynk.virtualWrite(V3, 0); // Initialize with 0

  // Setup a function to be called every 5 seconds

  timer.setInterval(3000L, sendData);

}

void loop() {

  Blynk.run();

  timer.run();

  // Read temperature and humidity from the DHT22 sensor

  float temperature = dht.readTemperature();

  float humidity = dht.readHumidity();

  // Read gas value from the potentiometer

  int gasValue = analogRead(potPin);

  // Determine air level based on the specified conditions

  String airLevel;

  // Check temperature and humidity conditions

  if ((temperature >= 22 && temperature <= 30) && (humidity > 30 && humidity < 60)) {

    airLevel = "Good";

  } else if ((temperature >= 30 && temperature <= 40) && (humidity >= 60 && humidity <= 70)) {

    airLevel = "Normal";

  } else {

    airLevel = "Bad";

  }

  // Determine gas level based on the criteria

  String gasLevel;

  if (gasValue >= 0 && gasValue <= 1364) {

    gasLevel = "Good";

  } else if (gasValue >= 1365 && gasValue <= 2730) {

    gasLevel = "Normal";

  } else {

    gasLevel = "Bad";

  }

  // Determine air quality based on the criteria

  String airQuality;

  if ((airLevel == "Good" || airLevel == "Normal") && (gasLevel == "Good" || gasLevel == "Normal")) {

    airQuality = "Good Air Quality";

  } else {

    airQuality = "Bad Air Quality";

  }

  // Display temperature and humidity on the LCD

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("Temp: " + String(temperature) + " C");

  lcd.setCursor(0, 1);

  lcd.print("Humidity: " + String(humidity) + " %");

  delay(2000); // Display temperature and humidity for 2 seconds

  // Display air level on the LCD

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("Air Level: " + airLevel);

  delay(2000); // Display air level for 2 seconds

  // Display gas level and gas value on the LCD

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("Gas Level: " + gasLevel);

  lcd.setCursor(0, 1);

  lcd.print("Gas Value: " + String(gasValue));

  delay(2000); // Display gas level and value for 2 seconds

  // Display air quality on the LCD

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("Air Quality: ");

  lcd.setCursor(0, 1);

  lcd.print(airQuality);

  delay(2000); // Display air quality for 2 seconds

  // Control the LED based on air quality

  if (airQuality == "Bad Air Quality") {

    digitalWrite(ledPin, HIGH); // Turn on the LED

  } else {

    digitalWrite(ledPin, LOW); // Turn off the LED

  }

}





