**FINAL DELIVERABLES - FINAL CODE**

|  |  |
| --- | --- |
| **TEAM ID** | **PNT2022TMID43096** |
| **PROJECT NAME** | **GAS LEAKAGE MONITORING AND ALERTING SYSTEM FOR INDUSTRIES** |

**WOKWI CODE:**

Including Required Header Files

#include <WiFi.h>

#include <PubSubClient.h>

#include <DHTesp.h>

#include <Stepper.h>

#include <ESP32Servo.h>

/\*

 NOTE:

//

 As Gas Sensor is not available in Wokwi platform.

 Slide Potentiometer is used instead of Gas Sensor, to variably set level of gas leakage.

\*/

// Defining Constants

#define DHTPIN 15

#define GAS\_LEVER 34  // Slide Potentiometer

#define buzzer 13

#define LED 5

const int servoPin = 12;

Servo valve;

DHTesp dhtsensor;

Stepper stepper(1000, 19,21,22,23);

void callback(char\* subscribetopic, byte\* payload, unsigned int payloadLength);

#define ORG "wpf8mr"

#define DEVICE\_TYPE "ESP32"

#define DEVICE\_ID "GAS\_LEAKAGE\_MONITOR"

#define TOKEN "0123456789"

String data3;

float h, t, g;

int pos=0;

boolean valve\_open=true;

//-------- Customise the above values --------

char server[] = ORG ".[messaging.internetofthingsx0.ibmcloud.com](http://messaging.internetofthingsx0.ibmcloud.com/)";

char publishTopic[] = "iot-2/evt/Data/fmt/json";

char subscribetopic[] = "iot-2/cmd/test/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;

//-----------------------------------------

WiFiClient wifiClient;

PubSubClient client(server, 1883, callback ,wifiClient);

void setup()

{

**Serial**.begin(115200);

  dhtsensor.setup(DHTPIN,DHTesp::DHT22);

  stepper.setSpeed(100);

  valve.attach(servoPin);

  pinMode(GAS\_LEVER, INPUT);

  pinMode(buzzer,OUTPUT);

  delay(10);

**Serial**.println();

  wificonnect();

  mqttconnect();

  valve.write(90);

}

void loop()

{

  TempAndHumidity data=dhtsensor.getTempAndHumidity();

  t=data.temperature;

  h=data.humidity;

  g=map(int(analogRead(GAS\_LEVER)), 0, 4095, 200, 2000);

**Serial**.print("temperature:");

**Serial**.println(t);

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

**Serial**.println(h);

**Serial**.print("Gas Level:");

**Serial**.println(g);

  if(g>500){

    tone(buzzer, 1000);

    stepper.step(1000);

    valve.write(180);

  }

  else{

    valve.write(90);

    noTone(buzzer);

  }

  PublishData(t, h, g);

  delay(1000);

  if (!client.loop()) {

    mqttconnect();

  }

}

/\*.....................................retrieving to Cloud...............................\*/

void PublishData(float temp, float humid, float gas\_level) {

  mqttconnect();

  String payload = "{\"temperature\":";

  payload += temp;

  payload += "," "\"humidity\":";

  payload += humid;

  payload += "," "\"gas\_level\":";

  payload += gas\_level;

  payload += "}";

**Serial**.print("Sending payload: ");

**Serial**.println(payload);

  if (client.publish(publishTopic, (char\*) payload.c\_str())) {

**Serial**.println("Publish ok");

  } else {

**Serial**.println("Publish failed");

  }

}

void mqttconnect() {

  if (!client.connected()) {

**Serial**.print("Reconnecting client to ");

**Serial**.println(server);

    while (!!!client.connect(clientId, authMethod, token)) {

**Serial**.print(".");

      delay(500);

    }

     initManagedDevice();

**Serial**.println();

  }

}

void wificonnect()

{

**Serial**.println();

**Serial**.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);

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

    delay(500);

**Serial**.print(".");

  }

**Serial**.println("");

**Serial**.println("WiFi connected");

**Serial**.println("IP address: ");

**Serial**.println(WiFi.localIP());

}

void initManagedDevice() {

  if (client.subscribe(subscribetopic)) {

**Serial**.println((subscribetopic));

**Serial**.println("subscribe to cmd OK");

  } else {

**Serial**.println("subscribe to cmd FAILED");

  }

}

void callback(char\* subscribetopic, byte\* payload, unsigned int payloadLength)

{

**Serial**.print("callback invoked for topic: ");

**Serial**.println(subscribetopic);

  for (int i = 0; i < payloadLength; i++) {

    data3 += (char)payload[i];

  }

**Serial**.println("data: "+ data3);

  data3="";

  }