SPRINT - I

Date	07 November 2022
Team ID	PNT2022TMID06928
Project Title	Project - Industry Specific Intelligent Fire Management System

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 2

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht connected

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

//-------credentials of IBM Accounts------

#define ORG "gh6uoi"//IBM ORGANITION ID
#define DEVICE_TYPE "trial1"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234abcd"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float h, t;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format
in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND
COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


//----------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by passing
parameter like server id,portand wificredential
```

```
void setup()// configureing the ESP32
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED,OUTPUT);
  delay(10);
  Serial.println();
  pinMode(mq2, INPUT);
  pinMode (flame_sensor_pin , INPUT ); // declaring sensor pin as input pin for Arduino
  pinMode(BUZZER_PIN, OUTPUT);
  wificonnect();
  mqttconnect();
}


void loop()// Recursive Function
{


  t = dht.readTemperature();
  Serial.print("temp:");
  Serial.println(t);

 if(t > 60)
 {
 Serial.println("Alert");
 digitalWrite(BUZZER_PIN, HIGH); // turn on
 }
else
 {
 digitalWrite(BUZZER_PIN, LOW); // turn on
 }

  int gassensorAnalogmq2 = analogRead(mq2);
 Serial.print("mq2 Gas Sensor: ");
 Serial.print(gassensorAnalogmq2);


 if (gassensorAnalogmq2 > 1500)
 {
 Serial.println("mq2Gas");
 Serial.println("Alert");
 }
else
 {
 Serial.println("No mq2Gas");
 }

 flame_pin = digitalRead ( flame_sensor_pin ) ; // reading from the
sensor if (flame_pin == LOW ) // applying condition
{
Serial.println ( " ALERT: FLAME DETECTED" ) ;
digitalWrite ( BUZZER_PIN , HIGH ) ;// if state is high, then turn high the
BUZZER }

else
{
Serial.println ( " NO FLAME DETECTED " ) ;
digitalWrite ( BUZZER_PIN , LOW ) ; // otherwise turn it low
}


  PublishData(t, gassensorAnalogmq2, flame_pin);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}



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

void PublishData(float t, float gassensorAnalogmq2 , int flame_pin ) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */
  String payload = "{\"temp\":";
  payload += t;
  payload += "," "\"gasvalue\":";
  payload += gassensorAnalogmq2;
  payload += "," "\"flame\":";
  payload += flame_pin;
  payload += "}";


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


  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it will print publish ok
in Serial monitor or else it will print publish failed
  } 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() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection
  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++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }
  Serial.println("data: "+ data3);
  if(data3=="lighton")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);
  }
data3="";
}
```

NOTE: As we are unable to find gas and flame sensors in WOKWI simulating platform we are not able to attach a circuit diagram for this program**.**