WOKWI

Sketch.ino:

```
#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
#define BUZZER 4
#define SPRKLR 12

#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C LCD = LiquidCrystal_I2C(0x27, 20, 4);

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 "1s2adz"//IBM ORGANITION ID
#define DEVICE_TYPE "ardiuno"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "0910"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float humid, temp, gas, spr=0;

//-------- 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);
  pinMode(BUZZER,OUTPUT);
  digitalWrite(LED,LOW);
  digitalWrite(BUZZER,LOW);
  pinMode(SPRKLR,OUTPUT);
  digitalWrite(SPRKLR,LOW);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
  LCD.init();
  LCD.backlight();
  LCD.setCursor(0, 0);
  LCD.print("Connecting to ");
  LCD.setCursor(0, 1);
  LCD.print("WiFi ");
  delay(1000);
  LCD.clear();
}

void loop()// Recursive Function
{
    LCD.setCursor(0,2);
    LCD.print("Gas: ");
    LCD.setCursor(0, 0);
    LCD.print("Temp: ");
    LCD.setCursor(14, 0);
    LCD.print("C");
    LCD.setCursor(0, 1);
    LCD.print("Humid: ");
    LCD.setCursor(14, 1);
    LCD.print("%");
  humid = dht.readHumidity();
  temp = dht.readTemperature();
  gas = random(0,900);

  if (gas>400)
  {
    Serial.print("Gas: ");
    Serial.println("Detected");
    digitalWrite(BUZZER,HIGH);
```

```
  LCD.setCursor(7, 2);
  LCD.print("YES");
  LCD.setCursor(0, 3);
  LCD.print("WARNING!FIREACCIDENT");
  digitalWrite(SPRKLR,HIGH);
  digitalWrite(SPRKLR,HIGH);
}
else{
    Serial.print("Gas: ");
    Serial.println("Not Detected");
    digitalWrite(BUZZER,LOW);
    LCD.setCursor(7, 2);
    LCD.print(" NO");
    LCD.setCursor(0, 3);
    LCD.print("                    ");
    digitalWrite(SPRKLR,LOW);
    spr=0;
  }
if(temp>60)
{
    Serial.print("FIRE: ");
    Serial.println("Detected");
    spr=1;
    digitalWrite(SPRKLR,HIGH);
    spr=1;
}
else
{
    Serial.print("FIRE: ");
    Serial.println("nOT Detected");
    spr=0;
    digitalWrite(SPRKLR,LOW);
    spr=0;
}
Serial.print("temp:");
Serial.println(temp);
LCD.setCursor(7, 0);
LCD.print(temp);
Serial.print("humid:");
Serial.println(humid);
LCD.setCursor(7, 1);
LCD.print(humid);

PublishData(temp, humid, gas);
delay(1000);
if (!client.loop()) {
  mqttconnect();
}
```

```
}
/*.....................................retrieving to
Cloud.............................*/

void PublishData(float temp, float humid, float gas) {
  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 += temp;
  payload += "," "\"humid\":";
  payload += humid;
  payload += "," "\"gas\":";
  payload += gas;
  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");
  LCD.setCursor(0, 0);
  LCD.print("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=="sprinkleron")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);
  }
data3="";
}
```

Diagram.json:

```
{
  "version": 1,
  "author": "sudhakar",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -10.12, "left":
19.21, "attrs": {} },
    {
      "type": "wokwi-led",
      "id": "led1",
      "top": -57.53,
      "left": 151.58,
      "attrs": { "color": "red" }
    },
    {
      "type": "wokwi-resistor",
      "id": "r1",
      "top": 66.62,
      "left": 158.95,
      "rotate": 90,
      "attrs": {}
    },
    {
      "type": "wokwi-dht22",
      "id": "dht1",
      "top": -135.61,
      "left": 213.07,
      "attrs": { "humidity": "90", "temperature": "47.4" }
    },
    {
      "type": "wokwi-lcd2004",
      "id": "lcd2",
      "top": -15.95,
      "left": 308.35,
      "attrs": { "pins": "i2c" }
    },
    {
      "type": "wokwi-buzzer",
      "id": "bz1",
      "top": -177.38,
      "left": 104.94,
      "attrs": { "volume": "0.1" }
    },
    {
      "type": "wokwi-led",
      "id": "led4",
```

```
      "top": -52.83,
      "left": -72.43,
      "attrs": { "color": "red" }
    },
    {
      "type": "wokwi-resistor",
      "id": "r4",
      "top": 79.31,
      "left": -63.95,
      "rotate": 90,
      "attrs": {}
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "esp:GND.1", "led1:C", "black", [ "h0" ] ],
    [ "led1:A", "r1:1", "green", [ "v0" ] ],
    [ "r1:2", "esp:D2", "green", [ "h0", "v38" ] ],
    [ "dht1:VCC", "esp:3V3", "red", [ "v0" ] ],
    [ "dht1:GND", "esp:GND.1", "black", [ "v0" ] ],
    [ "dht1:SDA", "esp:D15", "green", [ "v0" ] ],
    [ "lcd2:GND", "esp:GND.1", "black", [ "h-15.76", "v129.04" ] ],
    [ "lcd2:VCC", "esp:3V3", "red", [ "h-8.7", "v132.79" ] ],
    [ "lcd2:SDA", "esp:D21", "green", [ "h-19.3", "v19.04" ] ],
    [ "lcd2:SCL", "esp:D22", "green", [ "h-28.14", "v-15.2" ] ],
    [ "bz1:1", "esp:GND.1", "green", [ "v133.92", "h-2.55" ] ],
    [ "bz1:2", "esp:D4", "green", [ "v0" ] ],
    [ "led4:A", "r4:1", "green", [ "v0" ] ],
    [ "led4:C", "esp:GND.1", "black", [ "v0" ] ],
    [ "esp:D12", "r4:2", "green", [ "h0" ] ]
  ]
}
```

LINK: https://wokwi.com/projects/348835706528858708