```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include <MQ131.h>
#define LED 24

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

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

#define ORG "sxai7i"//IBM ORGANITION ID
#define DEVICE_TYPE "gasleakage"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "gasid"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "fE!Ha?HaL3y3mg4SW-"     //Token
#define METHOD "use-token-auth"
String data1;
float g;


//-------- 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/test/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



int gasSensor=A1;
int buzzer=13;
int led=12;
void setup()
{
  pinMode(A1, INPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(led, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  int sensorValue=analogRead(gasSensor);
  Serial.print("GAS LEVEL:");
  Serial.print(sensorValue);
  delay(10);
  if (sensorValue>250)
  {
    digitalWrite(buzzer,HIGH);
    digitalWrite(led,HIGH);
```

```
 }
 else
 {
   digitalWrite(buzzer,LOW);
   digitalWrite(led,LOW);
 }

 PublishData(g);
 delay(1000);
 if (client.loop())
 {
   mqttconnect();
 }
}

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

void PublishData(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 = "{\"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");

  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]);
    data1 += (char)payload[i];
  }

  Serial.println("data: "+ data1);
  if(data1=="lighton")
  {
Serial.println(data1);
digitalWrite(LED,HIGH);

  }

  else
  {
Serial.println(data1);
digitalWrite(LED,LOW);


  }
data1="";
}
```