

## ASSIGNMENT 4

Date	24 october 2022
Team ID	PNT2022TMID35759
Project name	Gas Leakage monitoring and Alerting system for Industries
Maximum marks	4 marks

**Project Title - Gas Leakage monitoring and Alerting system for Industries**

**Team ID - PNT2022TMID35759**

**Team members**

**1 Rubak Preyan G - Team Leader**

**2 Sudharson G V - Team member**

**3 Muhilan B - Team member**

**4 Aparnaa A S - Team member**

## QUESTION:

Write code and connections in wokwi for ultrasonic sensor.

Whenever distance

is less than 100 cms send "alert" to ibm cloud and display in device recent

events.

## CODE:

```
#include <WiFi.h>
```

```
#include <PubSubClient.h>
```

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

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

```
#define ORG "Ruby5432"//IBM ORGANIZATION ID
```

```
#define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson  
IOT Platform
```

```
#define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT  
Platform
```

```
#define TOKEN "12345678" //Token
```

```
String data3;
```

```
char server[] = ORG ".messaging.internetofthings.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);

const int trigPin = 5;

const int echoPin = 18;

#define SOUND_SPEED 0.034

long duration;

float distance;

void setup() {

  Serial.begin(115200);

  pinMode(trigPin, OUTPUT);pinMode(echoPin, INPUT);

  wificonnect();

  mqttconnect();

}

void loop()

{

  digitalWrite(trigPin, LOW);

  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(trigPin, LOW);
```

```

duration = pulseIn(echoPin, HIGH);
distance = duration * SOUND_SPEED/2;

Serial.print("Distance (cm): ");

Serial.println(distance);

if(distance<100)
{
    Serial.println("ALERT!!");
    delay(1000);
    PublishData(distance);
    delay(1000);
    if (!client.loop()) {
        mqttconnect();
    }
}

delay(1000);
}

void PublishData(float dist) {
    mqttconnect();

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

    payload += dist;

    payload += ", \"ALERT!!\": \"\"Distance less than 100cms\"";

    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();

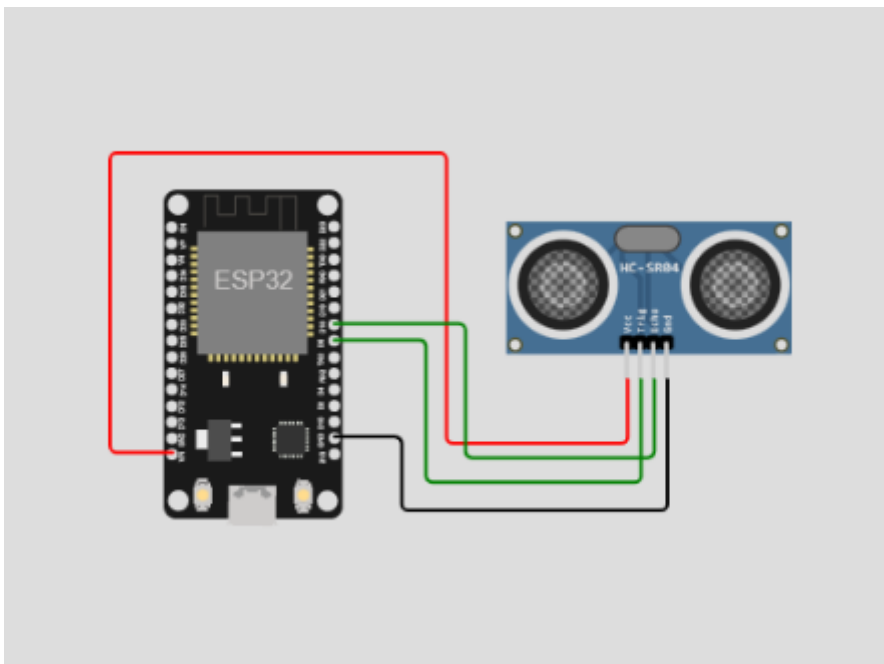
Serial.print("Connecting to ");

WiFi.begin("Wokwi-GUEST", "", 6);
```

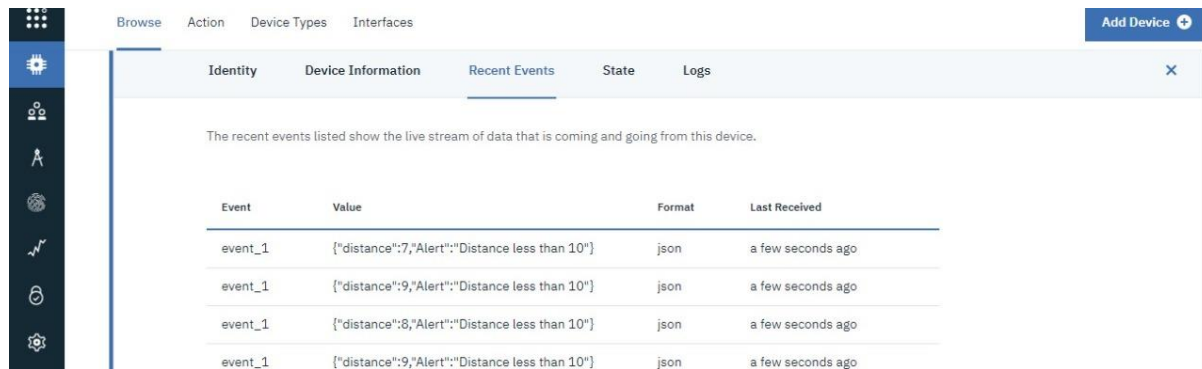
```
while (WiFi.status() != WL_CONNECTED) {  
    delay(500);  
    Serial.print(".");  
}  
  
Serial.println("");  
Serial.println("WiFi connected");  
Serial.println("IP address: ");  
Serial.println(WiFi.localIP());  
}  
  
void initManagedDevice() {  
    if (client.subscribe(subscribetopic)) {  
        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++) {
```

```
//Serial.print((char)payload[i]);  
data3 += (char)payload[i];  
}  
Serial.println("data: "+ data3);  
data3=""  
}
```

SCHEMATIC/CIRCUIT DIAGRAM:



## IBM CLOUD OUTPUT:



The screenshot shows the IBM Cloud IoT Platform console. On the left is a dark sidebar with icons for various functions. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A blue 'Add Device' button with a plus icon is on the right. Below the navigation bar, there are tabs for 'Identity', 'Device Information', 'Recent Events' (which is selected), 'State', and 'Logs'. A close button 'X' is in the top right corner of the main content area. The main content area contains a text description: 'The recent events listed show the live stream of data that is coming and going from this device.' Below this is a table with four columns: 'Event', 'Value', 'Format', and 'Last Received'. The table contains four rows of data, all showing 'event\_1' as the event name, a JSON string as the value, 'json' as the format, and 'a few seconds ago' as the last received time.

Event	Value	Format	Last Received
event_1	{"distance":7,"Alert":"Distance less than 10"}	json	a few seconds ago
event_1	{"distance":9,"Alert":"Distance less than 10"}	json	a few seconds ago
event_1	{"distance":8,"Alert":"Distance less than 10"}	json	a few seconds ago
event_1	{"distance":9,"Alert":"Distance less than 10"}	json	a few seconds ago

## WOKWI LINK:

<https://wokwi.com/projects/348774955176952404>