

## Assignment -4

### GAS LEAKAGE MONITERING AND ALERTING SYSTEM FOR INDUSTRIES

Student Name	S.K.Saravanan
Student Register Number	821219104019

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device recent events. Upload document with wokwi share link and images of IBM cloud.

#### Code 1:

File Name : sketch.ino

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribtopic, byte* payload, unsigned int payloadLength);
#define ORG "ytvrds"
#define DEVICE_TYPE "esp32"
#define DEVICE_ID "12345"
#define TOKEN "12345678"
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribtopic[] = "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++)
{
data3 += (char)payload[i];
}
Serial.println("data: "+ data3);
data3="";
}
}

```

## Code 2:

**File Name : diagram.json**

This Meta data given in IBM Watson IoT Platform

```

{
  "version": 1,
  "author": "abdulmohamedm",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -4.67, "left": -112.87, "attrs": {} },
    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": 15.96, "left": 89.17, "attrs": {} }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [

```

```

    "esp:VIN",
    "ultrasonic1:VCC",
    "red",
    [ "h-37.16", "v-178.79", "h200", "v173.33", "h100.67" ]
  ],
  [ "esp:GND.1", "ultrasonic1:GND", "black", [ "h39.87", "v44.04", "h170" ] ],
  [ "esp:D5", "ultrasonic1:TRIG", "green", [ "h54.54", "v85.07", "h130.67" ] ],
  [ "esp:D18", "ultrasonic1:ECHO", "green", [ "h77.87", "v80.01", "h110" ] ]
]
}

```

**Wokwi Link :**

<https://wokwi.com/projects/347665080823841362>

**Output and Simulation :**

The screenshot displays the Wokwi web IDE interface. On the left, the code editor shows a sketch for an ESP32 that uses the PubSubClient library to connect to IBM Cloud IoT. It configures an ultrasonic sensor (HC-SR04) and sends an alert message to a specific topic when the measured distance is less than 100cm. On the right, the simulation window shows the physical components: an ESP32 microcontroller and an ultrasonic sensor. Below the simulation, a console window shows the output of the program, including the distance reading (96.99 cm) and the successful publication of an alert message to the IBM Cloud IoT platform.

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int payloadLength)
4 #define ORG "ytvrds"
5 #define DEVICE_TYPE "esp32"
6 #define DEVICE_ID "12345"
7 #define TOKEN "12345678"
8 String data3;
9 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
10 char publishTopic[] = "iot-2/evt/Data/fmt/json";
11 char subscribtopic[] = "iot-2/cmd/test/fmt/String";
12 char authMethod[] = "use-token-auth";
13 char token[] = TOKEN;
14 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
15 WiFiClient wificlient;
16 PubSubClient client(server, 1883, callback, wificlient);
17 const int trigPin = 5;
18 const int echoPin = 18;
19 #define SOUND_SPEED 0.034
20 long duration;
21 float distance;
22 void setup() {
23   Serial.begin(115200);
24   pinMode(trigPin, OUTPUT);
25   pinMode(echoPin, INPUT);
26   wificlient.connect();
27   mqttconnect();
28 }
29 void loop()

```

Simulation Output:

```

Publish ok
Distance (cm): 96.99
ALERT!!
Sending payload: {"Distance":96.99,"ALERT!!":"Distance less than 100cms"}
Publish ok
Distance (cm): 96.99
ALERT!!

```

**Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device recent events.**

IBM Watson IoT Platform

Device Drilldown - 12345

Recent Events

Event	Value	Format	Last Received
Data	{"Distance":96.99,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":96.99,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":96.99,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":96.99,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":96.99,"ALERT!!":"Distance less than ...	json	a few seconds ago

State

This table shows a list of data points that are reported by this device.

Showing Raw Data | No Interfaces Available

IBM Watson IoT Platform

Browse Devices

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device Simulator

Device ID	Status	Device Type	Class ID	Date Added
> 12345	Connected	esp32	Device	Nov 5, 2022 10:05 PM

Items per page 50 | 1-1 of 1 item

1 of 1 page

IBM Watson IoT Platform

Device Drilldown - 12345

Connection Information

Basic connection information about this device.

Device ID	12345
Device Type	esp32
Date Added	Nov 5, 2022 10:05 PM
Added By	manjunathvivekananthan@gmail.com
Connection Status	Connected
	Connection Time: Nov 5, 2022 10:18 PM
	Client Address: 216.246.119.62 Insecure

Recent Events

The recent events listed show the live stream of data that is coming and going from this device.

IBM Watson IoT Platform

Device Drilldown - 12345

State

This table shows a list of data points that are reported by this device.

Showing Raw Data | No Interfaces Available

Property	Value	Type	Event	Last Received
Distance	96.99	Number	Data	a few seconds ago
ALERT!!	Distance less than 100cms	String	Data	a few seconds ago

Device Information

View basic device information including location and manufacturer.

Edit Device Information

