

## Assignment 4

Assignment Date	14 November 2022
Student Name	V. THIRUMURUGAN
Maximum Marks	2 Marks

### Question-1:

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.

Solution:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
const int trigPin = 5;
const int echoPin = 18;

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

#define ORG "p2cfk6" //IBM ORGANITION ID
#define DEVICE_TYPE "ULTRA" //Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "45" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "fF3ns!LKXhrBBbe5zm" //Token

#define SOUND_SPEED 0.034

long duration;
float dist;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/command/fmt/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, wifiClient); //calling the predefined
client id by passing parameter like server id,portand wificredential

void setup() // configureing the ESP32
{
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  delay(10);
  Serial.println();
```

```

    wificonnect();
    mqttconnect();
}

void loop()// Recursive Function
{

    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);
    dist = duration * SOUND_SPEED/2;

    // Prints the distance in the Serial Monitor
    Serial.print("Distance: ");
    Serial.print(dist);
    Serial.println(" cm");
    delay(1000);

    PublishData(dist);
    delay(1000);
    if (!client.loop()) {
        mqttconnect();
    }
}
/*.....retrieving to
Cloud. .... */

void PublishData(float dist) {
    mqttconnect();//function call for connecting to ibm
    /*
        creating the String in form JSON to update the data to ibm cloud
    */
    if(dist<100)
    {
        String payload = "{\"Alert! Distance is less than 100\":";
        payload += dist;
        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");
    }
}
else{
    String payload = "{\"Distance\":";
    payload += dist;

```

```

        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);
        }
        Serial.println();
    }
}
void wificonnect() //function defination for wificonnect
{
    Serial.println();
    Serial.print("Connecting to ");

    WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials
    toestablish 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())
    ;
}

```

WOKWI LINK: <https://wokwi.com/projects/348325320874525266>

## Wokwi simulation Output:

The screenshot displays the Wokwi web interface. On the left, the 'sketch.ino' file is open, showing C++ code for an ESP32 connected to an HC-SR04 ultrasonic sensor. The code includes libraries for WiFi and MQTT, defines credentials and device information, and sets up a serial monitor. On the right, the 'Simulation' window shows a 3D model of the ESP32 and the HC-SR04 sensor connected by wires. Below the simulation, a status window shows the connection progress: 'Connecting to ...', 'WiFi connected', and 'IP address: 10.10.0.2'.

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 const int trigPin = 5;
4 const int echoPin = 18;
5
6 //-----credentials of IBM Accounts-----
7
8 #define ORG "p2cfk6" //IBM ORGANITION ID
9 #define DEVICE_TYPE "ULTRA" //Device type mentioned in ibm watson IOT Platform
10 #define DEVICE_ID "45" //Device ID mentioned in ibm watson IOT Platform
11 #define TOKEN "ff3ns!LKXhr8Bbe5zm" //Token
12
13 #define SOUND_SPEED 0.034
14
15 long duration;
16 float dist;
17
18
19 //----- Customise the above values -----
20 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
21 char publishTopic[] = "iot-2/evt/IotSensor/fmt/json"; // topic name and type of
22 char subscribeTopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT comma
23 char authMethod[] = "use-token-auth"; // authentication method
24 char token[] = TOKEN;
25 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
26
27
```

## OUTPUT IN IBM CLOUD (DEVICE RECENT EVENTS):

Identity	Device Information	Recent Events	State	Logs
The recent events listed show the live stream of data that is coming and going from this device.				
Event	Value	Format	Last Received	
Data	{"Distance":50.94900131,"message":"Alert"}	json	a few seconds ago	
Data	{"Distance":137.9720001,"message":"Normal"}	json	a few seconds ago	
Data	{"Distance":272.9349976,"message":"Normal"}	json	a few seconds ago	
Data	{"Distance":363.9360046,"message":"Normal"}	json	a few seconds ago	
Data	{"Distance":363.9700012,"message":"Normal"}	json	a few seconds ago	