

## Assignment 4

Assignment Date	19 October 2022
Student Name	Mr. RAJALINGAM J
Student Roll Number	737819ECL232
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 = 19;
const int echoPin = 18;

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

#define ORG "yy3qcm" //IBM ORGANITION ID
#define DEVICE_TYPE "ibm22" //Device type mentioned in ibm watson IOTPlatform
#define DEVICE_ID "123" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678" //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, port and wificredential
```

```

void setup()// configuring 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 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());
}
}

```

OUTPUT IN WOKWI:

The image displays two screenshots of the Wokwi IDE interface, showing the code editor and the simulation window for an ESP32 project.

**Top Screenshot:**

- Code Editor (sketch.ino):** The code defines constants for an IBM IoT project, including credentials, device type, and token. It sets up a WiFi client and a PubSubClient for MQTT communication.
- Simulation Window:** Shows the ESP32 board connected to an HC-SR04 ultrasonic sensor. The output log indicates the device is connecting to the IBM IoT cloud.

**Bottom Screenshot:**

- Code Editor (sketch.ino):** The code continues with the MQTT connection logic, including a function to reconnect the client if it fails.
- Simulation Window:** Shows the same setup as the top screenshot, but the output log now shows the device successfully connecting to the IBM IoT cloud.

Wokwi link: <https://wokwi.com/projects/348296053459518036>