

ASSIGNEMENT 4

Team ID: PNT2022TMID15932

- 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.**

Code:

```
#include <WiFi.h>

#include <PubSubClient.h>

#include <ArduinoJson.h>

WiFiClient wifiClient;

#define ORG "noeto5"

#define DEVICE_TYPE "devicetype"

#define DEVICE_ID "deviceid"

#define TOKEN "token"

#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char publishTopic[] = "iot-2/evt/Data/fmt/json";

char topic[] = "iot-2/cmd/home/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

PubSubClient client(server, 1883, wifiClient);

void publishData();

const int trigpin=5;

const int echopin=18;

String command;

String data="";

String icon="";
```

```
long duration;
int dist;

void setup()
{
    Serial.begin(115200);
    pinMode(trigpin, OUTPUT);
    pinMode(echopin, INPUT);
    wifiConnect();
    mqttConnect();
}

void loop() {

    publishData();
    delay(500);

    if (!client.loop()) {
        mqttConnect();
    }
}

void wifiConnect() {
    Serial.print("Connecting to "); Serial.print("Wifi");
    WiFi.begin("Wokwi-GUEST", "", 6);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.print("WiFi connected, IP address: "); Serial.println(WiFi.localIP());
}
```

```

void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(1000);
    }
    initManagedDevice();
    Serial.println();
  }
}

```

```

void initManagedDevice() {
  if (client.subscribe(topic)) {
    Serial.println(client.subscribe(topic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}

```

```

void publishData()
{
  digitalWrite(trigpin,LOW);
  digitalWrite(trigpin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin,LOW);
  duration=pulseIn(echopin,HIGH);
  dist=duration*speed/2;

  if(dist<100){
    icon="fa-trash";

```

```
DynamicJsonDocument doc(1024);  
String payload;  
doc["Alert: Distance is less than 100cm"]=dist;  
serializeJson(doc, payload);  
delay(3000);  
Serial.print("\n");  
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{  
    icon="fa-trash-o";  
}  
  
}
```

Wokwi Link:

<https://wokwi.com/projects/346424392358560338>

Output:

The screenshot shows the Wokwi simulation environment. On the left, the sketch code is displayed in the `sketch.ino` file. The code includes libraries for WiFi, PubSubClient, and ArduinoJson. It defines constants for the device ID, token, and speed. The main logic involves connecting to an MQTT broker, publishing data, and triggering an alert when the distance is less than 100cm.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 #include <ArduinoJson.h>
4
5 WiFiClient wifiClient;
6
7 #define ORG "noeto5"
8 #define DEVICE_TYPE "RaspberryPi"
9 #define DEVICE_ID "12345"
10 #define TOKEN "12345678"
11 #define speed 0.034
12
13 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
14 char publishTopic[] = "iot-2/evt/Data/fmt/json";
15 char topic[] = "iot-2/cmd/home/fmt/String";
16 char authMethod[] = "use-token-auth";
17 char token[] = TOKEN;
18 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
19 PubSubClient client(server, 1883, wifiClient);
20 void publishData();
21
22 const int trigpin=5;
23 const int echopin=18;
24 String command;
25 String data="";
26 String icon="";
27
28 long duration;
29 int dist;
```

On the right, the simulation window shows the ESP32 microcontroller and the HC-SR04 ultrasonic sensor. The distance is currently 48cm. The output console shows the following messages:

```
Publish OK
Sending payload: {"Alert: Distance is less than 100cm":1}
Publish OK
Sending payload: {"Alert: Distance is less than 100cm":48}
Publish OK
Go to Settings to activate Windows.
```

The screenshot shows the IBM Watson IoT Platform dashboard. The device is named "12345" and is connected. The dashboard displays the "Recent Events" tab, which shows a list of events received from the device. The events are in JSON format and contain alert messages about the distance.

Event	Value	Format	Last Received
Data	{"Alert: Distance is less than 100cm":48}	json	a few seconds ago
Data	{"Alert: Distance is less than 100cm":1}	json	a few seconds ago
Data	{"Alert: Distance is less than 100cm":1}	json	a few seconds ago
Data	{"Alert: Distance is less than 100cm":96}	json	a few seconds ago
Data	{"Alert: Distance is less than 100cm":96}	json	a few seconds ago

At the bottom of the dashboard, it indicates "0 Simulations running".