

Assignment - 4

Assignment Date	25 October 2022
Student Name	Ms. KANNAGI N
Student Roll Number	912019106007
Maximum Marks	2 Marks

Question-1:

Write a code and make a connection in wokwi for ultrasonic sensor. Whenever distance is less than 100 send 'alert' to ibm cloud and display in device recent events.

Solution:

Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient wifiClient;
String data3;
#define ORG "4yi0vc"
#define DEVICE_TYPE "nodeMcu"
#define DEVICE_ID "Assignment4"
#define TOKEN "123456789"
#define speed 0.034
#define led 14
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="";
```

```
long duration;
float dist;
```

```
void setup()
{
  Serial.begin(115200);
  pinMode(led, OUTPUT);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
```

```

    wifiConnect();
    mqttConnect();
}

void loop() {
    bool isNearby = dist < 100;
    digitalWrite(led, isNearby);

    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(500);
        }
        initManagedDevice();
        Serial.println();
    }
}

void initManagedDevice() {
    if (client.subscribe(topic)) {
        // Serial.println(client.subscribe(topic));
        Serial.println("IBM 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){

```

```

String payload = "{\"Normal Distance\":";
payload += dist;
payload += "}";

Serial.print("\n");
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish OK");
}

}

if(dist>101 && dist<111){
String payload = "{\"Alert distance\":";
payload += dist;
payload += "}";

Serial.print("\n");
Serial.print("Sending payload: ");
Serial.println(payload);
if(client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Warning crosses 110cm -- it automaticaly of the loop");
    digitalWrite(led,HIGH);
}else {
    Serial.println("Publish 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++){
    dist += (char)payload[i];
}
Serial.println("data:" + data3);
if(data3=="lighton"){
    Serial.println(data3);
    digitalWrite(led,HIGH);
}
data3="";
}

```

WOKWI CODE:

<https://wokwi.com/projects/346937989346099796>

OUTPUT

Distance is greater than 100

The screenshot shows the Wokwi IoT simulator interface. On the left, the sketch code is displayed, which includes headers for WiFi and PubSubClient, defines for device ID, token, and speed, and a loop function that publishes data to an IBM Cloud IoT topic. On the right, the simulation output shows the device connecting to WiFi and the MQTT client, and then publishing data to the IBM Cloud IoT topic. The LED is shown as off.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 WiFiClient wificlient;
4 String data3;
5 #define ORG "ks8pti"
6 #define DEVICE_TYPE "ESP32"
7 #define DEVICE_ID "143143"
8 #define TOKEN "123456789"
9 #define speed 0.034
10 #define led 14
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Kannan/fmt/json";
13 char topic[] = "iot-2/cmd/home/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 PubSubClient client(server, 1883, wificlient);
18 void publishData();
19
20
21 const int trigpin=5;
22 const int echopin=10;
23 String command;
24 String data="";
25
26 long duration;
```

Simulation output:

```
Connecting to Wifi..Wifi connected, IP address: 10.10.0.2
Reconnecting MQTT client to ks8pti.messaging.internetofthings.ibmcloud.com
IBM subscribe to cmd OK
```

IBM cloud is connected and LED is off state

Distance is less than 100

The screenshot shows the Wokwi IoT simulator interface. On the left, the sketch code is displayed, which includes headers for WiFi and PubSubClient, defines for device ID, token, and speed, and a loop function that publishes data to an IBM Cloud IoT topic. On the right, the simulation output shows the device connecting to WiFi and the MQTT client, and then publishing data to the IBM Cloud IoT topic. The LED is shown as on.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 WiFiClient wificlient;
4 String data3;
5 #define ORG "ks8pti"
6 #define DEVICE_TYPE "ESP32"
7 #define DEVICE_ID "143143"
8 #define TOKEN "123456789"
9 #define speed 0.034
10 #define led 14
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Kannan/fmt/json";
13 char topic[] = "iot-2/cmd/home/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 PubSubClient client(server, 1883, wificlient);
18 void publishData();
19
20
21 const int trigpin=5;
22 const int echopin=10;
23 String command;
24 String data="";
25
26 long duration;
```

Simulation output:

```
Warning crosses 110cm -- it automaticaly of the loop
Sending payload: {"Alert Distance":47.99}
Warning crosses 110cm -- it automaticaly of the loop
Sending payload: {"Alert Distance":47.96}
Warning crosses 110cm -- it automaticaly of the loop
```

LED is on state

IBM Cloud foundry connection

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes the platform name, a user profile icon, and the email address 'kannankaruppaiah07052002@gmail.com' with ID 'ks8pti'. The main content area is titled 'Browse' and contains a table of events. The table has four columns: 'Event', 'Value', 'Format', and 'Last Received'. It lists five identical entries for a device named 'Kannan', each with a JSON value containing an alert distance of 47.96. The interface also features a sidebar with various icons and a bottom section with pagination controls.

Event	Value	Format	Last Received
Kannan	{"Alert Distance":47.96}	json	a few seconds ago
Kannan	{"Alert Distance":47.96}	json	a few seconds ago
Kannan	{"Alert Distance":47.96}	json	a few seconds ago
Kannan	{"Alert Distance":47.96}	json	a few seconds ago
Kannan	{"Alert Distance":47.96}	json	a few seconds ago

Items per page 50 | 1-2 of 2 items 1 of 1 page < 1 >

Getting alert message from wokwi