

Assignment-4

Name	G.Gowri
Topic	Smart Solutions for Railways
Team id	PNT2022TMID38248

Write code and connections in wokwi for ultrasonic sensor

Whenever distance is less than 100 cms send “alert” to ibm cloud and display in device recent events.upload wokwi share link and images of ibm cloud

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic,byte* payload, unsigned int payloadLength);
#define ORG "eugl0y"
#define DEVICE_TYPE "iot"
#define DEVICE_ID "456"
#define TOKEN "12345678"
String data3;

char server[]= ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[]="iot-2/evt/distance/fmt/json";
char subscribeTopic[]="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);

#define ECHO_PIN 2
#define TRIG_PIN 4
#define led 5

void setup() {
  // put your setup code here, to run once:
```

```

    Serial.begin(115200);
    pinMode(led, OUTPUT);
    pinMode(TRIG_PIN, OUTPUT);
    pinMode(ECHO_PIN, INPUT);
    wificonnect();
    mqttconnect();
}
float readDistanceCM() {
    digitalWrite(TRIG_PIN, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG_PIN, HIGH);
    delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);
    int duration=random(1,200);
    //Serial.println(duration);
    //duration = pulseIn(ECHO_PIN, HIGH);
    return duration ;
    //Serial.println(duration);
}

void loop() {
    float distance = readDistanceCM();
    //Serial.println(distance);

    bool isNearby = distance < 100;
    digitalWrite(led, isNearby);

    Serial.print("Measured distance: ");
    Serial.println(distance);
    if(distance<100){
        PublishData2(distance);

    }else{
        PublishData1(distance);

    }
    //PublishData(distance);
    delay(1000);
    if(!client.loop()){
        mqttconnect();
    }

    //delay(2000);
}
void PublishData1(float dist){
    mqttconnect();
    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 PublishData2(float dist){
    mqttconnect();
    String payload= "{\\"ALERT\\":";
    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 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);
    if(data3=="lighton"){
        Serial.println(data3);
        digitalWrite(led,HIGH);
    }else{
        Serial.println(data3);
        digitalWrite(led,LOW);
    }
    data3="";
}

```

LINK: <https://wokwi.com/projects/347492831664800339>

Output:

Normal Case:

WOKWI **SAVE** **SHARE** sketch.ino copy **Docs** **SIGN UP**

sketch.ino • diagram.json libraries.txt Library Manager

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribtopic,byte* payload,unsigned int payloadLength);
4 #define ORG "eugloy"
5 #define DEVICE_TYPE "iot"
6 #define DEVICE_ID "456"
7 #define TOKEN "12345678"
8 String data3;
9
10 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
11 char publishTopic[] = "iot-2/evt/distance/fmt/json";
12 char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
13 char authMethod[] = "use-token-auth";
14 char token[] = TOKEN;
15 char clientId[] = "d:" + ORG + ":" + DEVICE_TYPE + ":" + DEVICE_ID;
16
17 WiFiClient wifiClient;
18 PubSubClient client(server,1883,callback,wifiClient);
19
20 #define ECHO_PIN 2
21 #define TRIG_PIN 4
22 #define led 5
23
24 void setup() {
25   // put your setup code here, to run once:
26   Serial.begin(115200);
27   pinMode(led, OUTPUT);
28   pinMode(TRIG_PIN, OUTPUT);
29   pinMode(ECHO_PIN, INPUT);
30 }
```

Simulation 00:27.833 52%

publish ok
Measured distance: 86.00
Sending payload:{"ALERT":86.00}
publish ok
Measured distance: 128.00
Sending payload:{"distance":128.00}
publish ok

Alter Case:

WOKWI **SAVE** **SHARE** sketch.ino copy **Docs** **SIGN UP**

sketch.ino • diagram.json libraries.txt Library Manager

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribtopic,byte* payload,unsigned int payloadLength);
4 #define ORG "eugloy"
5 #define DEVICE_TYPE "iot"
6 #define DEVICE_ID "456"
7 #define TOKEN "12345678"
8 String data3;
9
10 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
11 char publishTopic[] = "iot-2/evt/distance/fmt/json";
12 char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
13 char authMethod[] = "use-token-auth";
14 char token[] = TOKEN;
15 char clientId[] = "d:" + ORG + ":" + DEVICE_TYPE + ":" + DEVICE_ID;
16
17 WiFiClient wifiClient;
18 PubSubClient client(server,1883,callback,wifiClient);
19
20 #define ECHO_PIN 2
21 #define TRIG_PIN 4
22 #define led 5
23
24 void setup() {
25   // put your setup code here, to run once:
26   Serial.begin(115200);
27   pinMode(led, OUTPUT);
28   pinMode(TRIG_PIN, OUTPUT);
29   pinMode(ECHO_PIN, INPUT);
30 }
```

Simulation 00:14.608 84%

publish ok
Measured distance: 4.00
Sending payload:{"ALERT":4.00}
publish ok
Measured distance: 15.00
Sending payload:{"ALERT":15.00}
publish ok

IBM CLOUD:

The screenshot displays the IBM Watson IoT Platform web interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A user profile is visible in the top right corner with the email 'idletalker15@gmail.com' and ID 'eugl0y'. The main content area shows details for a device with ID '456', which is 'Connected'. The 'Recent Events' tab is selected, displaying a table of live data events.

Event	Value	Format	Last Received
distance	{"ALERT":34}	json	a few seconds ago
distance	{"ALERT":74}	json	a few seconds ago
distance	{"distance":106}	json	a few seconds ago
distance	{"ALERT":40}	json	a few seconds ago
distance	{"ALERT":1}	json	a few seconds ago

The bottom of the image shows a Windows taskbar with various application icons, a search bar, and system status information including temperature (28°C), time (18:30), and date (05-11-2022).