

# Assignment-4

Name	S.Snekha
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* subscribtopic,byte* payload, unsigned int payloadLength);
#define ORG "zikscr"
#define DEVICE_TYPE "iot"
#define DEVICE_ID "321"
#define TOKEN "87654321"
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="";
}

```

# Output:

## Normal case:

WOKWI sketch.ino copy

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int payloadLength);
4 #define ORG "ziksco"
5 #define DEVICE_TYPE "iot"
6 #define DEVICE_ID "321"
7 #define TOKEN "87654321"
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

publish ok  
Measured distance: 157.00  
Sending payload:{"distance":157.00}  
publish ok  
Measured distance: 164.00  
Sending payload:{"distance":164.00}  
publish ok

## Alter Case:

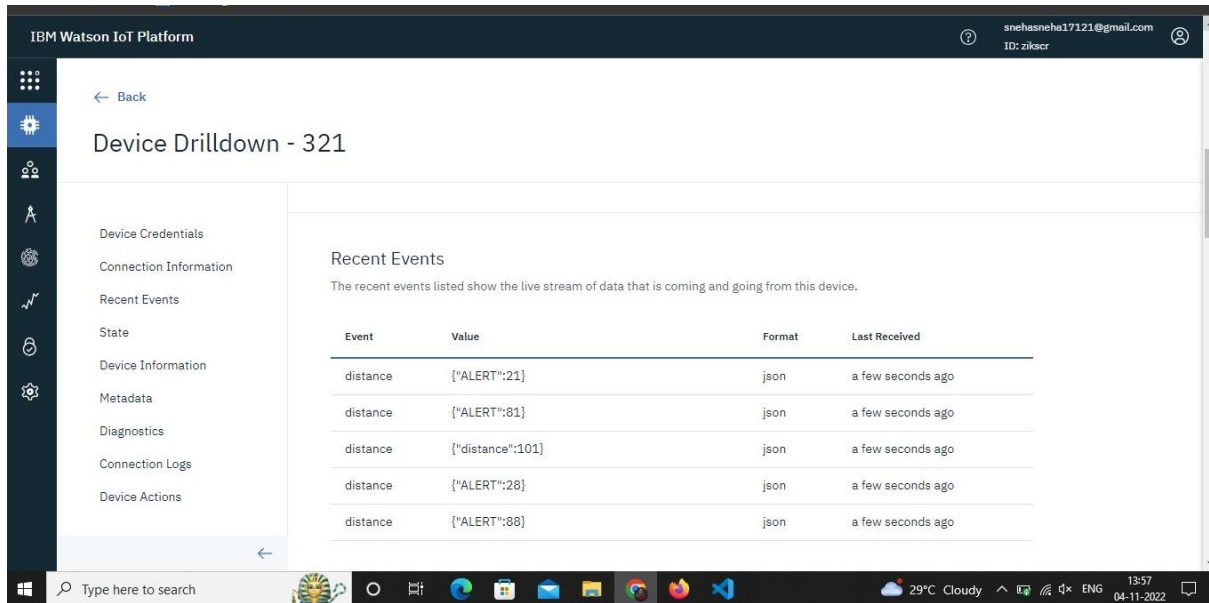
WOKWI sketch.ino copy

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int payloadLength);
4 #define ORG "ziksco"
5 #define DEVICE_TYPE "iot"
6 #define DEVICE_ID "321"
7 #define TOKEN "87654321"
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

publish ok  
Measured distance: 112.00  
Sending payload:{"distance":112.00}  
publish ok  
Measured distance: 81.00  
Sending payload:{"ALERT":81.00}  
publish ok

# Cloud storage:



The screenshot shows the IBM Watson IoT Platform interface. The top header displays the platform name and user information. The left sidebar contains navigation icons and labels. The main content area is titled 'Device Drilldown - 321' and features a 'Recent Events' section with a table of data.

IBM Watson IoT Platform

snehasneha17121@gmail.com  
ID: zikscr

← Back

## Device Drilldown - 321

Device Credentials  
Connection Information  
Recent Events  
State  
Device Information  
Metadata  
Diagnostics  
Connection Logs  
Device Actions

### Recent Events

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
distance	{"ALERT":21}	json	a few seconds ago
distance	{"ALERT":81}	json	a few seconds ago
distance	{"distance":101}	json	a few seconds ago
distance	{"ALERT":28}	json	a few seconds ago
distance	{"ALERT":88}	json	a few seconds ago

Type here to search

29°C Cloudy 13:57 04-11-2022