

#### Assignment-4

##### Distance Detection Using Ultrasonic Sensor

Assignment Date	29 October 2022
Student Name	Shyam kumar
Team ID	PNT2022TMID24432
Maximum Marks	2 Marks

#### Question-1:

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100cms send "alert" to ibm cloud and display in device recent events.

WOKWI LINK : <https://wokwi.com/projects/346518948162830932>

#### CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

#define ORG "f59trs"
#define DEVICE_TYPE "ultrasonicsensor"
#define DEVICE_ID "distancedetection"
#define TOKEN "AlGMGaaF01nawa1QA3"
String data3;
float dist;

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

int LED = 4;
int trig = 5;
int echo = 18;
void setup()
{
  Serial.begin(115200);
  pinMode(trig,OUTPUT);
```

```

pinMode(echo, INPUT);
pinMode(LED, OUTPUT);
delay(10);
wificonnect();
mqttconnect();
}
void loop()
{
    digitalWrite(trig, LOW);
    digitalWrite(trig, HIGH);
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    float dur = pulseIn(echo, HIGH);
    float dist = (dur * 0.0343)/2;
    Serial.print ("Distance in cm :");
    Serial.println(dist);

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

void PublishData(float dist) {
    mqttconnect();
    String object;
    if (dist < 100)

```

```

if (dist < 100)
{
    digitalWrite(LED, HIGH);
    Serial.println("object is near");
    object = "Near";
}
else
{
    digitalWrite(LED, LOW);
    Serial.println("no object found");
    object = "No";
}

String payload = "{\"distance\": ";
payload += dist;
payload += ", \"object\": ";
payload += object;
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);
        }
        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");
    }
}

```

```

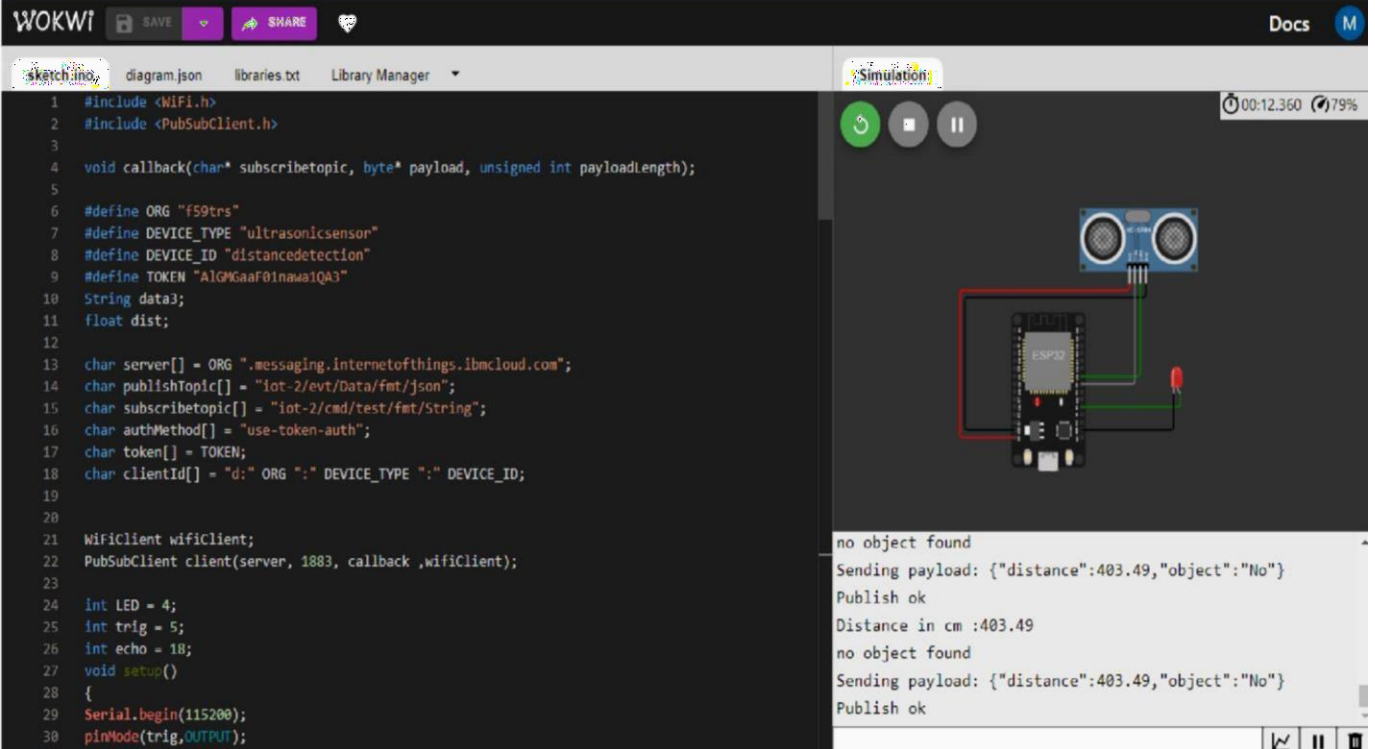
        Serial.println("subscribe to cmd FAILED");
    }
}

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];
    }
    data3="";
}

```

## OUTPUT:

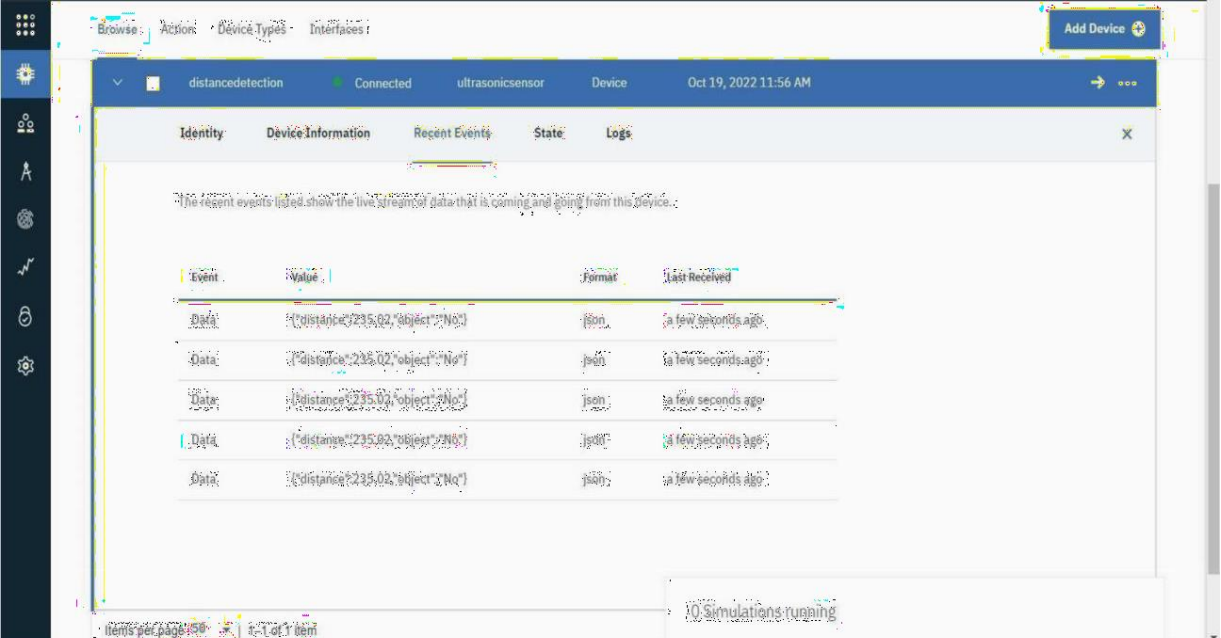
### When object is not near to the ultrasonic sensor



The screenshot shows the Wokwi IoT simulator interface. On the left, a code editor displays a sketch for an ESP8266. The sketch includes headers for `WiFi.h` and `PubSubClient.h`, and defines constants for the MQTT server, topic, device ID, and token. It sets up a `WiFiClient` and a `PubSubClient` instance. The `setup` function initializes the serial port and the ultrasonic sensor pin. The `loop` function (not fully visible) would typically read the sensor and publish data. On the right, the simulation window shows a 3D model of the ESP8266 module connected to an ultrasonic sensor. Below the model, the console output shows the following sequence of events:

```
no object found
Sending payload: {"distance":403.49,"object":"No"}
Publish ok
Distance in cm :403.49
no object found
Sending payload: {"distance":403.49,"object":"No"}
Publish ok
```

### Data sent to the IBM cloud device when the object is far

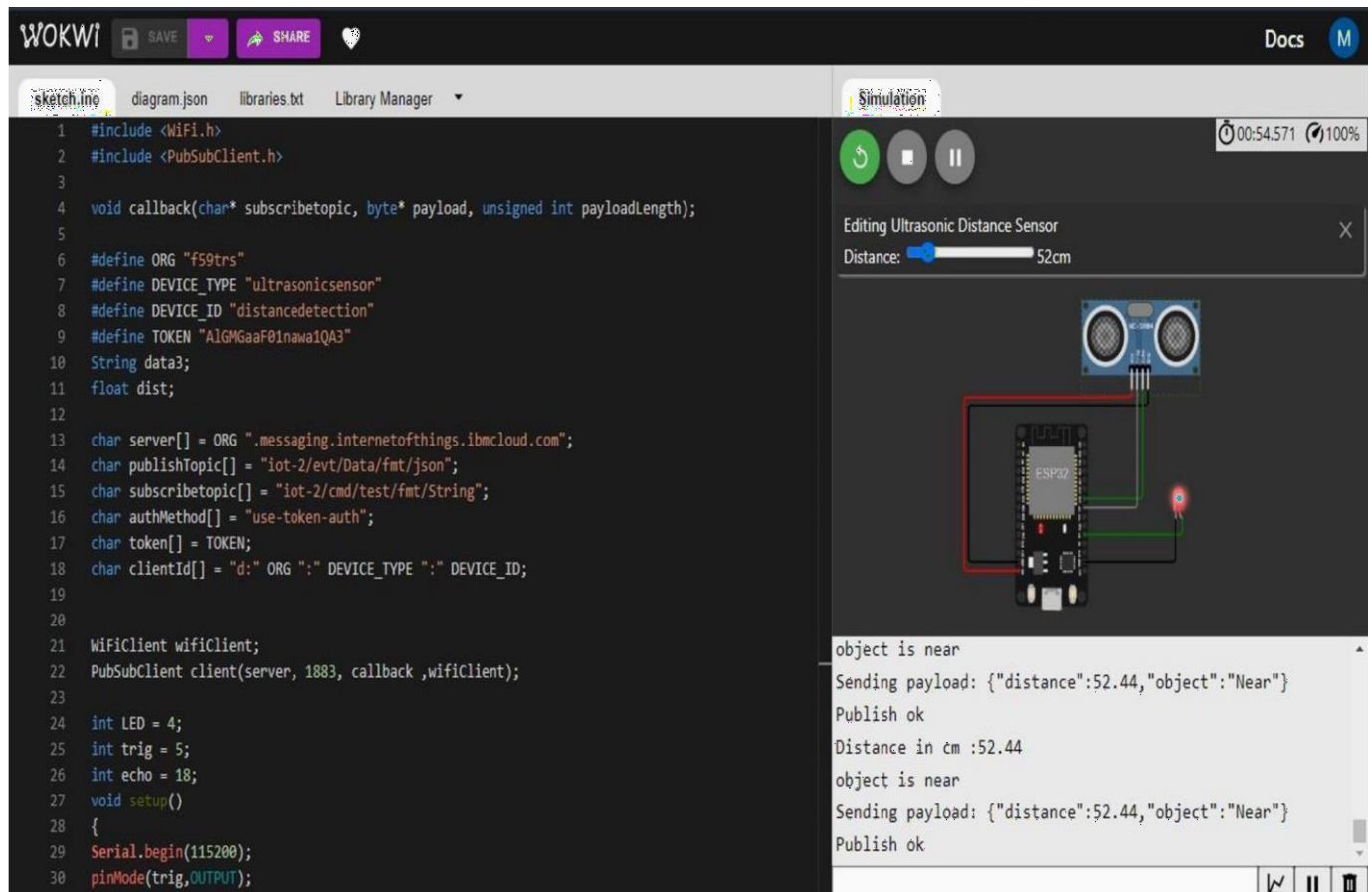


The screenshot shows the IBM Watson IoT Platform interface. The top navigation bar includes 'Browse', 'Actions', 'Device Types', and 'Interfaces'. The main content area displays a table of device data for a device named 'distancedetection'. The table has columns for 'Identity', 'Device Information', 'Recent Events', 'State', and 'Logs'. The 'Recent Events' column shows a list of events with their values, formats, and last received times.

Event	Value	Format	Last Received
Data	["distance":235.02,"object":"No"]	json	a few seconds ago
Data	["distance":235.02,"object":"No"]	json	a few seconds ago
Data	["distance":235.02,"object":"No"]	json	a few seconds ago
Data	["distance":235.02,"object":"No"]	json	a few seconds ago
Data	["distance":235.02,"object":"No"]	json	a few seconds ago



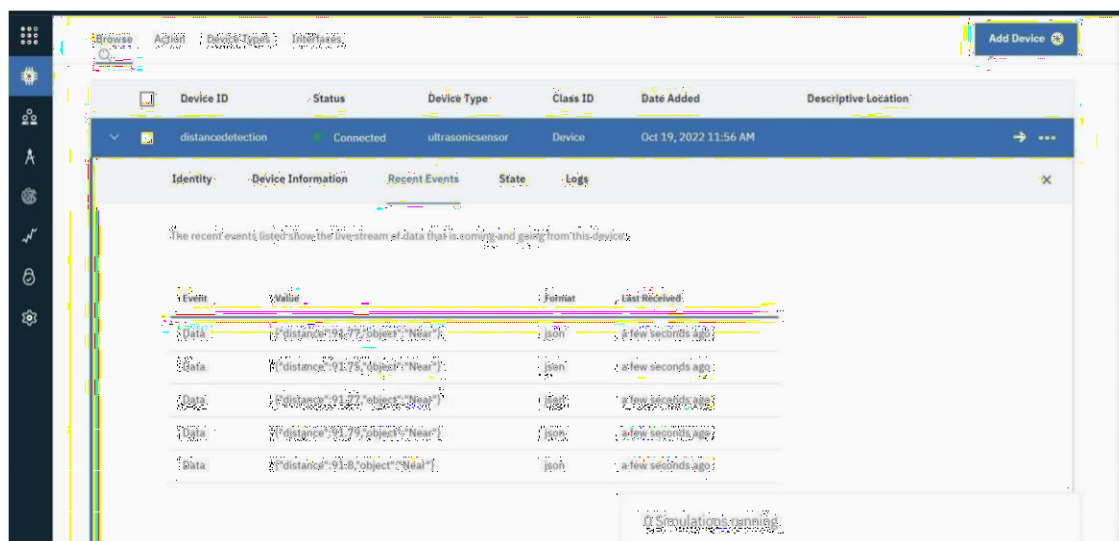
## When object is nearer to the ultrasonic sensor



```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3
4 void callback(char* subscribtopic, byte* payload, unsigned int payloadLength);
5
6 #define ORG "f59trs"
7 #define DEVICE_TYPE "ultrasonicsensor"
8 #define DEVICE_ID "distancedetection"
9 #define TOKEN "AlGMGaaF0Inawa1QA3"
10 String data3;
11 float dist;
12
13 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
14 char publishTopic[] = "iot-2/evt/Data/fmt/json";
15 char subscribtopic[] = "iot-2/cmd/test/fmt/String";
16 char authMethod[] = "use-token-auth";
17 char token[] = TOKEN;
18 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
19
20
21 WiFiClient wifiClient;
22 PubSubClient client(server, 1883, callback ,wifiClient);
23
24 int LED = 4;
25 int trig = 5;
26 int echo = 18;
27 void setup()
28 {
29   Serial.begin(115200);
30   pinMode(trig,OUTPUT);
```

object is near  
Sending payload: {"distance":52.44,"object":"Near"}  
Publish ok  
Distance in cm :52.44  
object is near  
Sending payload: {"distance":52.44,"object":"Near"}  
Publish ok

## Data sent to the IBM cloud device when the object is near



Event	Value	Format	Last Received
data	{"distance":52.44,"object":"Near"}	json	a few seconds ago
data	{"distance":52.44,"object":"Near"}	json	a few seconds ago
data	{"distance":52.44,"object":"Near"}	json	a few seconds ago
data	{"distance":52.44,"object":"Near"}	json	a few seconds ago

<https://wokwi.com/projects/346518948162830932>