

**NAME : SAKTHIVEL K**

**REG NO : 210619205040**

**PROGRAM**

Smart Waste Management System for Metropolitan Cities

**ASSIGNMENT 4:**

Write code and connections in wokwi for ultrasonic sensors. Whenever distance is less than 100 cms send "alert" to ibm cloud and display in device recent events. Uplode document with wokwi share link

and images of ibm cloud.

CODE:

```
#include <WiFi.h>

#include <PubSubClient.h>

WiFiClient wifiClient;

String data3;

#define ORG "ztcz45"

#define DEVICE_TYPE "naveen"

#define DEVICE_ID "naveen123"

#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);

    ...

[10:32 pm, 23/10/2022] Gogul B.E CSE: }

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("IBMsubscribe 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="";
}
```

output:

The screenshot displays the Wokwi IDE on the left and the IoT Platform on the right. The Wokwi IDE shows a C++ program for an ESP8266 module that uses an ultrasonic sensor to measure distance and sends the data to a cloud platform. The IoT Platform shows the received data in a table.

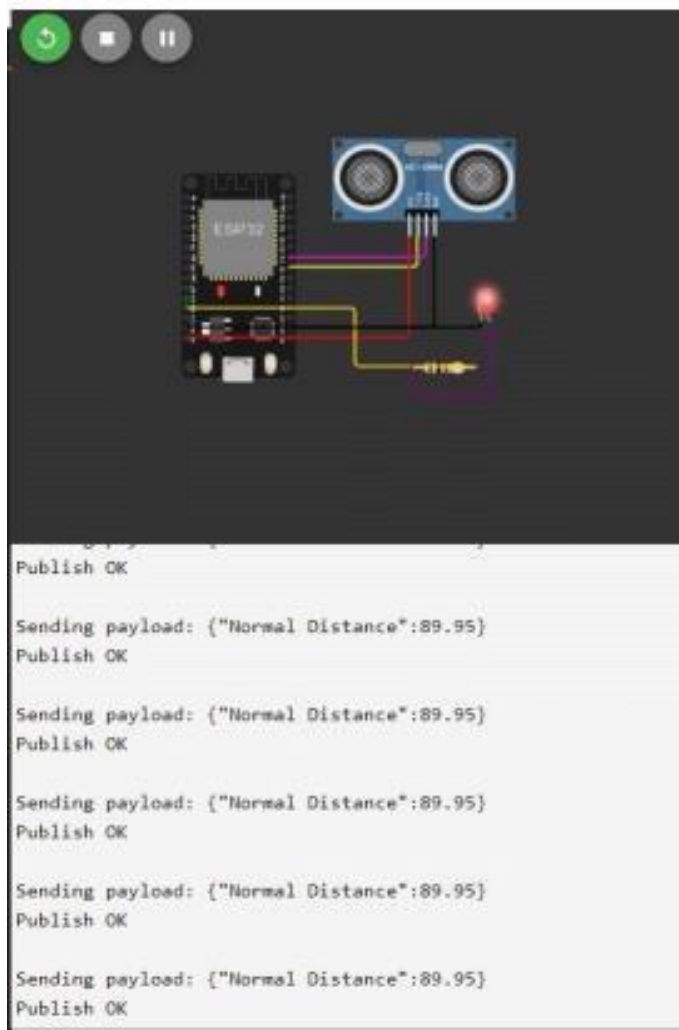
**Wokwi IDE Code:**

```
1 #include <ESP8266.h>
2 #include <WiFi.h>
3 #include <Wire.h>
4 #include <Ultrasonic.h>
5 #include <Arduino.h>
6 #include <ESP8266.h>
7 #include <WiFi.h>
8 #include <Wire.h>
9 #include <Ultrasonic.h>
10 #include <Arduino.h>
11 #include <ESP8266.h>
12 #include <WiFi.h>
13 #include <Wire.h>
14 #include <Ultrasonic.h>
15 #include <Arduino.h>
16 #include <ESP8266.h>
17 #include <WiFi.h>
18 #include <Wire.h>
19 #include <Ultrasonic.h>
20 #include <Arduino.h>
21 #include <ESP8266.h>
22 #include <WiFi.h>
23 #include <Wire.h>
24 #include <Ultrasonic.h>
25 #include <Arduino.h>
26 #include <ESP8266.h>
27 #include <WiFi.h>
28 #include <Wire.h>
29 #include <Ultrasonic.h>
30 #include <Arduino.h>
31 #include <ESP8266.h>
32 #include <WiFi.h>
33 #include <Wire.h>
34 #include <Ultrasonic.h>
35 #include <Arduino.h>
36 #include <ESP8266.h>
37 #include <WiFi.h>
38 #include <Wire.h>
39 #include <Ultrasonic.h>
40 #include <Arduino.h>
41 #include <ESP8266.h>
42 #include <WiFi.h>
43 #include <Wire.h>
44 #include <Ultrasonic.h>
45 #include <Arduino.h>
46 #include <ESP8266.h>
47 #include <WiFi.h>
48 #include <Wire.h>
49 #include <Ultrasonic.h>
50 #include <Arduino.h>
51 #include <ESP8266.h>
52 #include <WiFi.h>
53 #include <Wire.h>
54 #include <Ultrasonic.h>
55 #include <Arduino.h>
56 #include <ESP8266.h>
57 #include <WiFi.h>
58 #include <Wire.h>
59 #include <Ultrasonic.h>
60 #include <Arduino.h>
61 #include <ESP8266.h>
62 #include <WiFi.h>
63 #include <Wire.h>
64 #include <Ultrasonic.h>
65 #include <Arduino.h>
66 #include <ESP8266.h>
67 #include <WiFi.h>
68 #include <Wire.h>
69 #include <Ultrasonic.h>
70 #include <Arduino.h>
71 #include <ESP8266.h>
72 #include <WiFi.h>
73 #include <Wire.h>
74 #include <Ultrasonic.h>
75 #include <Arduino.h>
76 #include <ESP8266.h>
77 #include <WiFi.h>
78 #include <Wire.h>
79 #include <Ultrasonic.h>
80 #include <Arduino.h>
81 #include <ESP8266.h>
82 #include <WiFi.h>
83 #include <Wire.h>
84 #include <Ultrasonic.h>
85 #include <Arduino.h>
86 #include <ESP8266.h>
87 #include <WiFi.h>
88 #include <Wire.h>
89 #include <Ultrasonic.h>
90 #include <Arduino.h>
91 #include <ESP8266.h>
92 #include <WiFi.h>
93 #include <Wire.h>
94 #include <Ultrasonic.h>
95 #include <Arduino.h>
96 #include <ESP8266.h>
97 #include <WiFi.h>
98 #include <Wire.h>
99 #include <Ultrasonic.h>
100 #include <Arduino.h>
```

**IoT Platform Data Log:**

Time	Value	Format	Last Received
2023-10-10 10:10:10	[{"Alert distance":110.00}]	json	10/10/2023 10:10:10
2023-10-10 10:10:10	[{"Alert distance":110.00}]	json	10/10/2023 10:10:10
2023-10-10 10:10:10	[{"Alert distance":110.00}]	json	10/10/2023 10:10:10
2023-10-10 10:10:10	[{"Alert distance":110.00}]	json	10/10/2023 10:10:10

1. When distance under 100 cm it wil show normal distance.



2. When distance cross 100 cm it will show ALERT warning message distance

The screenshot shows the Wokwi IDE on the left and the IBM Watson IoT Platform on the right. The Wokwi IDE displays a C++ code snippet for an Arduino Uno connected to an Ultrasonic Distance Sensor. The code includes comments and logic for sending distance data to the cloud. The simulation window shows the sensor's output, with a distance of 110cm. The IBM Watson IoT Platform shows the device 'THOL123' in a 'Connected' state. The 'Recent Events' tab displays a stream of data points, including an alert message when the distance crosses 110cm.

```

1 #include <Arduino.h>
2 #include <PubSubClient.h>
3 #include <Ultrasonic.h>
4 #include <WiFi.h>
5 #include <String.h>
6 #include <EEPROM.h>
7 #include <SPI.h>
8 #include <SD.h>
9 #include <Wire.h>
10 #include <OneWire.h>
11 #include <DallasTemperature.h>
12 #include <DS18B20.h>
13 #include <Adafruit_NeoPixel.h>
14 #include <Adafruit_NeoPixel.h>
15 #include <Adafruit_NeoPixel.h>
16 #include <Adafruit_NeoPixel.h>
17 #include <Adafruit_NeoPixel.h>
18 #include <Adafruit_NeoPixel.h>
19 #include <Adafruit_NeoPixel.h>
20 #include <Adafruit_NeoPixel.h>
21 #include <Adafruit_NeoPixel.h>
22 #include <Adafruit_NeoPixel.h>
23 #include <Adafruit_NeoPixel.h>
24 #include <Adafruit_NeoPixel.h>
25 #include <Adafruit_NeoPixel.h>
26 #include <Adafruit_NeoPixel.h>
27 #include <Adafruit_NeoPixel.h>
28 #include <Adafruit_NeoPixel.h>
29 #include <Adafruit_NeoPixel.h>
30 #include <Adafruit_NeoPixel.h>
31 #include <Adafruit_NeoPixel.h>
32 #include <Adafruit_NeoPixel.h>
33 #include <Adafruit_NeoPixel.h>
34 #include <Adafruit_NeoPixel.h>
35 #include <Adafruit_NeoPixel.h>
36 #include <Adafruit_NeoPixel.h>
37 #include <Adafruit_NeoPixel.h>
38 #include <Adafruit_NeoPixel.h>
39 #include <Adafruit_NeoPixel.h>
40 #include <Adafruit_NeoPixel.h>
41 #include <Adafruit_NeoPixel.h>
42 #include <Adafruit_NeoPixel.h>
43 #include <Adafruit_NeoPixel.h>
44 #include <Adafruit_NeoPixel.h>
45 #include <Adafruit_NeoPixel.h>
46 #include <Adafruit_NeoPixel.h>
47 #include <Adafruit_NeoPixel.h>
48 #include <Adafruit_NeoPixel.h>
49 #include <Adafruit_NeoPixel.h>
50 #include <Adafruit_NeoPixel.h>
51 #include <Adafruit_NeoPixel.h>
52 #include <Adafruit_NeoPixel.h>
53 #include <Adafruit_NeoPixel.h>
54 #include <Adafruit_NeoPixel.h>
55 #include <Adafruit_NeoPixel.h>
56 #include <Adafruit_NeoPixel.h>
57 #include <Adafruit_NeoPixel.h>
58 #include <Adafruit_NeoPixel.h>
59 #include <Adafruit_NeoPixel.h>
60 #include <Adafruit_NeoPixel.h>
61 #include <Adafruit_NeoPixel.h>
62 #include <Adafruit_NeoPixel.h>
63 #include <Adafruit_NeoPixel.h>
64 #include <Adafruit_NeoPixel.h>
65 #include <Adafruit_NeoPixel.h>
66 #include <Adafruit_NeoPixel.h>
67 #include <Adafruit_NeoPixel.h>
68 #include <Adafruit_NeoPixel.h>
69 #include <Adafruit_NeoPixel.h>
70 #include <Adafruit_NeoPixel.h>
71 #include <Adafruit_NeoPixel.h>
72 #include <Adafruit_NeoPixel.h>
73 #include <Adafruit_NeoPixel.h>
74 #include <Adafruit_NeoPixel.h>
75 #include <Adafruit_NeoPixel.h>
76 #include <Adafruit_NeoPixel.h>
77 #include <Adafruit_NeoPixel.h>
78 #include <Adafruit_NeoPixel.h>
79 #include <Adafruit_NeoPixel.h>
80 #include <Adafruit_NeoPixel.h>
81 #include <Adafruit_NeoPixel.h>
82 #include <Adafruit_NeoPixel.h>
83 #include <Adafruit_NeoPixel.h>
84 #include <Adafruit_NeoPixel.h>
85 #include <Adafruit_NeoPixel.h>
86 #include <Adafruit_NeoPixel.h>
87 #include <Adafruit_NeoPixel.h>
88 #include <Adafruit_NeoPixel.h>
89 #include <Adafruit_NeoPixel.h>
90 #include <Adafruit_NeoPixel.h>
91 #include <Adafruit_NeoPixel.h>
92 #include <Adafruit_NeoPixel.h>
93 #include <Adafruit_NeoPixel.h>
94 #include <Adafruit_NeoPixel.h>
95 #include <Adafruit_NeoPixel.h>
96 #include <Adafruit_NeoPixel.h>
97 #include <Adafruit_NeoPixel.h>
98 #include <Adafruit_NeoPixel.h>
99 #include <Adafruit_NeoPixel.h>
100 #include <Adafruit_NeoPixel.h>

```

The IBM Watson IoT Platform interface shows the device 'THOL123' in a 'Connected' state. The 'Recent Events' tab displays a stream of data points, including an alert message when the distance crosses 110cm.

Event	Value	Format	Last Received
Data	{"Alert distance":110.90}	json	a few second
Data	{"Alert distance":110.90}	json	a few second
Data	{"Alert distance":110.90}	json	a few second
Data	{"Alert distance":110.90}	json	a few second
Data	{"Alert distance":110.90}	json	a few second

3. When it cross above 110 cm it today move to iff state once it

reduce to 110 it on again

Connection information:

Basic conntection information about this device.

Organization ID : ztcz45

Device Type : THOL

Device ID : THOL123

Authentication Method : use-token-auth Authentication Token : 123456789

The screenshot shows the 'Recent Events' tab of the IBM Watson IoT Platform. It displays a stream of data points, including a 'Normal Distance' event. The table below shows the details of these events.

Event	Value	Format	Last Received
Data	{"Normal Distance":89.95}	json	a few second
Data	{"Normal Distance":89.95}	json	a few second
Data	{"Normal Distance":89.95}	json	a few second
Data	{"Normal Distance":89.95}	json	a few second
Data	{"Normal Distance":89.95}	json	a few second