

**NAME : ASHOKAN K**

**REG NO : 513219104003**

**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("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="";
}
```

output:

The image shows a development environment with two main windows. The left window is the Wokwi IDE, displaying a C++ sketch for an Arduino Uno. The sketch is titled "Building Ultrasonic Distance Sensor" and includes comments in Chinese. The code defines a distance of 118cm and a delay of 100ms. It uses the `Serial` library to send data to the console. The right window is the TTP IoT Platform, showing a list of devices. The device "esp8266" is listed with a status of "Connected". Below the device list, there is a table showing the data being received from the device.

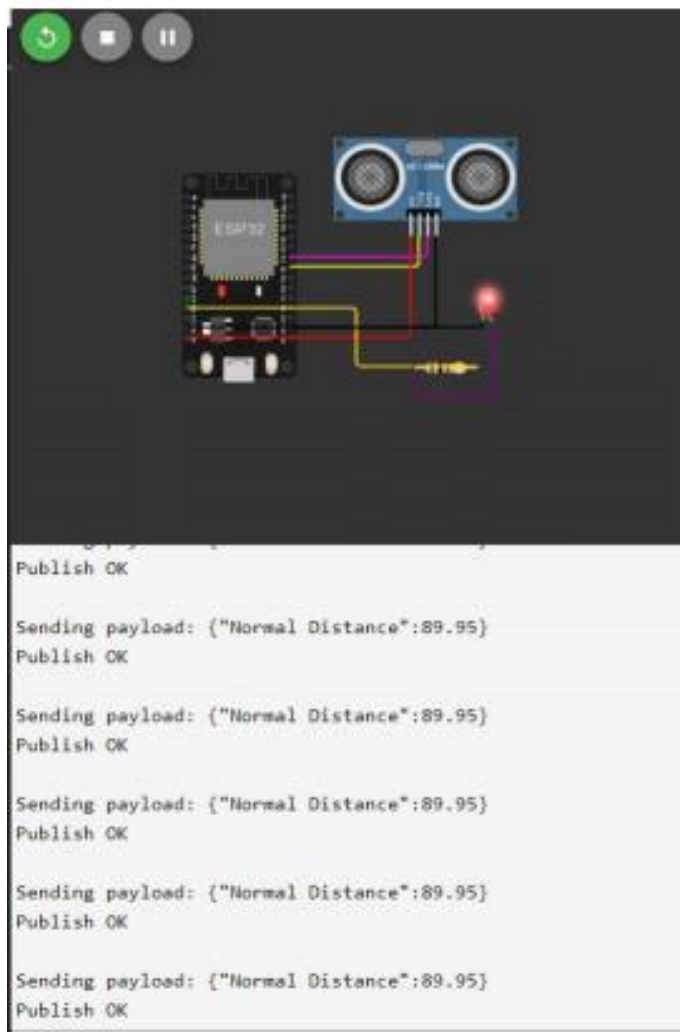
**Wokwi IDE Code:**

```
1 // 距离 118cm
2 #include <Arduino.h>
3 #include <Ultrasonic.h>
4 #include <Wire.h>
5 #define TRIG_PIN 5
6 #define ECHO_PIN 4
7 #define DISTANCE 118 // 距离
8 #define DELAY 100 // 延迟
9 #define SPEED 0.034 // 速度
10 #define LED 13
11
12 // 设置串口
13 void setup() {
14   Serial.begin(9600);
15   pinMode(LED, OUTPUT);
16   pinMode(TRIG_PIN, OUTPUT);
17   pinMode(ECHO_PIN, INPUT);
18   digitalWrite(LED, LOW);
19 }
20
21 // 发送数据
22 void sendData() {
23   Serial.print("Alert distance:");
24   Serial.print(DISTANCE);
25   Serial.println("cm");
26 }
27
28 // 主函数
29 void loop() {
30   digitalWrite(LED, HIGH);
31   delay(DELAY);
32   digitalWrite(LED, LOW);
33   sendData();
34   delay(DELAY);
35 }
```

**TTP IoT Platform Data:**

Event	Value	Format	Last Received
Data	[{"Alert distance": 118.00}]	json	10/10/2020
Data	[{"Alert distance": 118.00}]	json	10/10/2020
Data	[{"Alert distance": 118.00}]	json	10/10/2020
Data	[{"Alert distance": 118.00}]	json	10/10/2020
Data	[{"Alert distance": 118.00}]	json	10/10/2020

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 image shows two side-by-side screenshots. The left screenshot is from the Wokwi IDE, displaying a C++ code snippet for an Arduino Uno. The code includes comments and logic for an ultrasonic distance sensor. A simulation window shows a breadboard with an ultrasonic sensor module. The right screenshot is from the IBM Watson IoT Platform, showing the 'Recent Events' tab for a device named 'ranger123'. The table below lists the events received from the device.

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 connection information about this device.

Organization ID : ztcz45

Device Type : ASHOK

Device ID : ASHOK123

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

Identity	Device Information	Recent Events	State	Logs
The recent events listed show the live stream of data that is coming and going from this device.				
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	