

SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFETY
ASSIGNMENT-4

| | |
|--------------|--|
| NAME | E. ANANDHI |
| DATE | 25 th October 2022 |
| TEAM ID | PNT2022TMID38378 |
| PROJECT NAME | Signs with Smart Connectivity for Better Road Safety |

ASSIGNMENT-4

Write code and connections in wokwi for ultrasonic sensors. Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device recent events. Upload document with wokwi share link and images of IBM cloud.

CODE:

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

WiFiClient wifiClient; #define ORG
#define ORG "snzz1b"
#define DEVICE_TYPE "raspberrypi"
#define DEVICE_ID "12345"
#define TOKEN "12345678"
#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/raspberrypi_1/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="";
String lat="14.167589";
String lon="80.248510";
String name="point2";
String icon="";
```

```

long duration;
int dist;

void setup()
{
    Serial.begin(115200);
    pinMode(trigpin, OUTPUT);
    pinMode(echopin, INPUT);
    wifiConnect();
    mqttConnect();
}

void loop() {

    publishData();
    delay(500);

    if (!client.loop()) {
        mqttConnect();
    }
}

void wifiConnect() {
    Serial.print("Connecting to "); Serial.print("Wifi");
    WiFi.begin("Wokwi-GUEST", "", 6);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.print("WiFi connected, IP address: ");
    Serial.println(WiFi.localIP());
}

void mqttConnect() {
    if (!client.connected()) {
        Serial.print("Reconnecting MQTT client to "); Serial.println(server);
        while (!client.connect(clientId, authMethod, token)) {
            Serial.print(".");
            delay(1000);
        }
        initManagedDevice();
        Serial.println();
    }
}

void initManagedDevice() {
    if (client.subscribe(topic)) {
        Serial.println(client.subscribe(topic));
    }
}

```

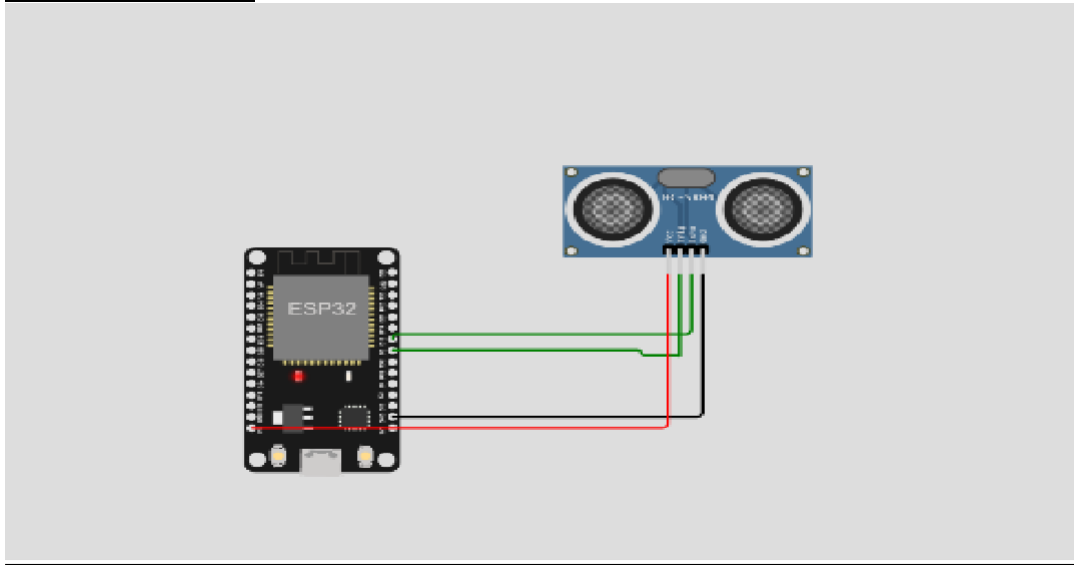
```

        Serial.println("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){
        dist=100-dist;
        icon="fa-trash";
    }else{
        dist=0;
        icon="fa-trash-o";
    }
    DynamicJsonDocument doc(1024);
    String payload;
    doc["Name"]=name;
    doc["Latitude"]=lat;
    doc["Longitude"]=lon;
    doc["Icon"]=icon;
    doc["FillPercent"]=dist;
    serializeJson(doc, payload);
    delay(3000);
    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);
    if (client.publish(publishTopic, (char*) payload.c_str())) {
        Serial.println("Publish OK");
    } else {
        Serial.println("Publish FAILED");
    }
}
}

```

CONNECTIONS:



WOKWI LINK:

<https://wokwi.com/projects/346487528804581971>

OUTPUT:

A screenshot of the Wokwi web interface. On the left, the 'sketch.ino' file is open, showing C++ code for an ESP32 connected to an MQTT broker. The code includes headers for WiFi, PubSubClient, and ArduinoJson, and defines constants for the MQTT server, topic, and device ID. The main function sets up the WiFi and MQTT client, and publishes data to the topic. On the right, the 'Simulation' window shows the same circuit diagram as above. Below the diagram, the output console displays the following text: 'Connecting to Wifi...Wifi connected, IP address: 10.10.0.2', 'Reconnecting MQTT client to snzz1b.messaging.internetofthings.ibmcloud.com', and '....'. The simulation is running at 84% speed.

A screenshot of the IBM Watson IoT Platform dashboard. The dashboard shows a list of devices, with one device selected. The selected device is named '12345', has a status of 'Disconnected', and is of type 'raspberrypi'. The 'Recent Events' tab is selected, showing a list of events. The events are as follows:

| Event | Value | Format | Last Received |
|---------|---------------------|--------|-------------------|
| event_1 | ["Alert Number":11] | json | a few seconds ago |
| event_1 | ["Alert Number":68] | json | a few seconds ago |
| event_1 | ["Alert Number":64] | json | a minute ago |
| event_1 | ["Alert Number":71] | json | 2 minutes ago |
| event_1 | ["Alert Number":64] | json | 2 minutes ago |

