

ASSIGNMENT-4

WOKWI PROGRAM

CODE

```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient wifiClient;
String data3;
#define ORG "9putym"
#define DEVICE_TYPE "Vijayadharshini"
#define DEVICE_ID "Assignment4"
#define TOKEN "1234567890"
#define speed 0.034
#define led 14
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Vijayadharshini/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);
    pinMode(echopin, INPUT);
    wifiConnect();
    mqttConnect();
}

void loop() {
    bool isNearby = dist < 100;
    digitalWrite(led, isNearby);

    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(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 ) {
        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)
{

```

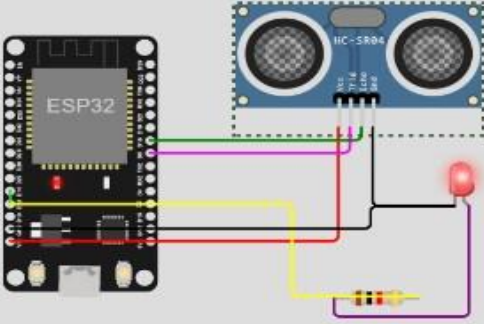
```
Serial.print("callback invoked for topic:");
Serial.println(subscribeTopic);
for (int i = 0; i < payloadLength; i++) {
    dist += (char)payload[i];
}
Serial.println("data:" + data3);
if (data3 == "lighton") {
    Serial.println(data3);
    digitalWrite(led, HIGH);
}
data3 = "";
}
```

OUTPUT

Simulation

01:52.742 93%

Editing Ultrasonic Distance Sensor
Distance: 69cm



Publish OK

Sending payload: {"Normal Distance":68.95}
Publish OK

Sending payload: {"Normal Distance":68.95}
Publish OK

Event	Value	Format	Last Received
Vijayadharshini	{"Normal Distance":68.95}	json	a few seconds ago
Vijayadharshini	{"Normal Distance":68.95}	json	a few seconds ago
Vijayadharshini	{"Normal Distance":68.95}	json	a few seconds ago
Vijayadharshini	{"Normal Distance":68.95}	json	a few seconds ago
Vijayadharshini	{"Normal Distance":68.95}	json	a few seconds ago

when distance under 100m It will show normal distance

Simulation

01:44.378
87%

Editing Ultrasonic Distance Sensor
Distance: 115cm

Sending payload: {"Alert distance":114.94}
Warning crosses 110cm -- it automaticaly of the loop

Sending payload: {"Alert distance":114.94}
Warning crosses 110cm -- it automaticaly of the loop

Sending payload: {"Alert distance":114.94}

Event	Value	Format	Last Received
Vijayadharshini	{"Alert distance":114.95}	json	a few seconds ago
Vijayadharshini	{"Alert distance":114.94}	json	a few seconds ago
Vijayadharshini	{"Alert distance":114.94}	json	a few seconds ago
Vijayadharshini	{"Alert distance":114.94}	json	a few seconds ago
Vijayadharshini	{"Alert distance":114.94}	json	a few seconds ago

when distance cross 100cm it will show ALERT with warning message distance