

# ASSIGNMENT-4

Date	21 october 2022
Student Name	M.Jothika
Student Roll Number	110519106005
Maximum Marks	2 Marks

## CODEING:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic,byte* payload, unsigned int
payloadLength);
#define ORG "jfe9dc"
#define DEVICE_TYPE "jothika77"
#define DEVICE_ID "6005"
#define TOKEN "0hx)_c(*RoUyNZfqST"
String data3;

char server[]= ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[]="iot-2/evt/jothika77/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);

#define ECHO_PIN 12
#define TRIG_PIN 13
#define led 14

void setup() {
    // put your setup code here, to run once:
    Serial.begin(115200);
    pinMode(led, OUTPUT);
    pinMode(TRIG_PIN, OUTPUT);
    pinMode(ECHO_PIN, INPUT);
    wificonnect();
    mqttconnect();
}
float readDistanceCM() {
    digitalWrite(TRIG_PIN, LOW);
    delayMicroseconds(2);
    digitalWrite(TRIG_PIN, HIGH);
```

```

    delayMicroseconds(10);
    digitalWrite(TRIG_PIN, LOW);
    int duration=random(1,200);
    //Serial.println(duration);
    //duration = pulseIn(ECHO_PIN, HIGH);
    return duration ;
    //Serial.println(duration);
}

void loop() {
    float distance = readDistanceCM();
    //Serial.println(distance);

    bool isNearby = distance < 100;
    digitalWrite(led, isNearby);

    Serial.print("Measured distance: ");
    Serial.println(distance);
    if(distance<100){
        PublishData2(distance);
    }
    else{
        PublishData1(distance);

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

    //delay(200);
}
void PublishData1(float dist){
    mqttconnect();
    String payload= "{\"distance\":\"";
    payload += dist;
    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 PublishData2(float dist){
    mqttconnect();
    String payload= "{\"ALERT\":";
    payload += dist;
    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 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");
    }
}

```

```

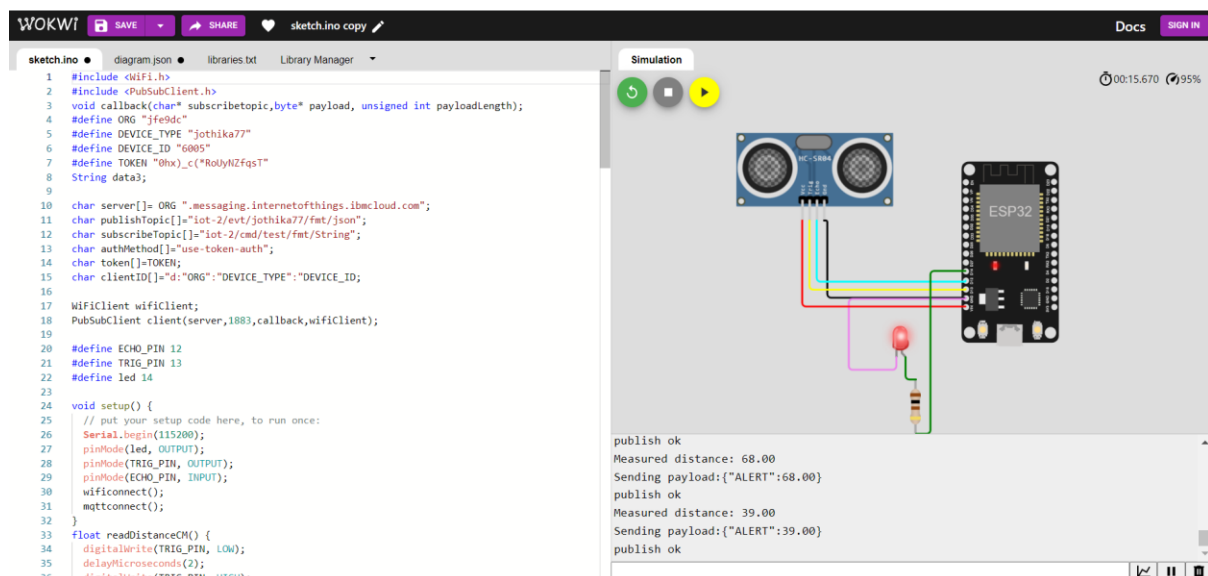
    }
}

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];
    }
    Serial.println("data:" + data3);
    if(data3=="lighton"){
        Serial.println(data3);
        digitalWrite(led,HIGH);
    }else{
        Serial.println(data3);
        digitalWrite(led,LOW);
    }
    data3="";
}
}

```

## OUTPUT:

### Less than 100cm-LED Glow:



The screenshot displays the Wokwi IoT simulator interface. On the left, the sketch code is shown, which includes MQTT client setup, topic subscriptions, and a callback function that triggers an LED (pin 14) to turn on when the measured distance is less than 100cm. The right side shows the simulation of the hardware, including an ESP32, an HC-SR04 sensor, and an LED. The output console at the bottom shows the sequence of events: a publish to 'iot-2/cwd/test/fmt/led', a distance measurement of 68.00cm, and the LED turning on.

```

publish ok
Measured distance: 68.00
Sending payload:{"ALERT":68.00}
publish ok
Measured distance: 39.00
Sending payload:{"ALERT":39.00}
publish ok

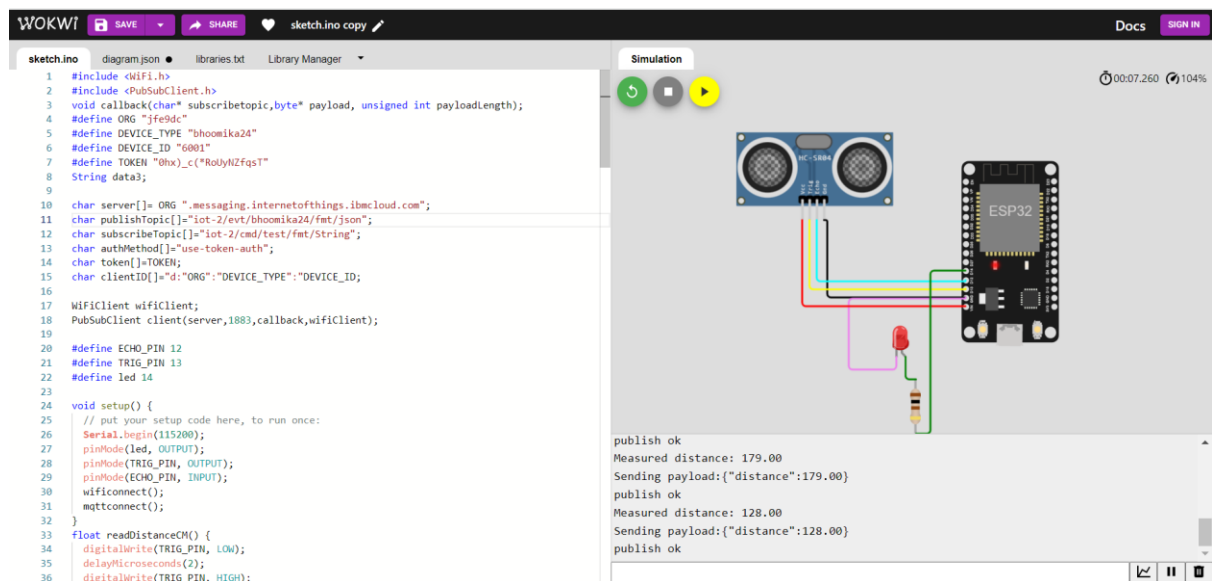
```

```

Measured distance: 68.00
Sending payload:{"ALERT":68.00}
publish ok
Measured distance: 39.00
Sending payload:{"distance":39.00}
publish ok

```

## More than 100cm-LED Doesn't Glow:



```
Measured distance: 179.00
Sending payload:{"ALERT":179.00}
publish ok
Measured distance: 128.00
Sending payload:{"distance":128.00}
publish ok
```