

ASSIGNMENT – 4

Date	22 nd October 2022
Name	Kampalle vineetha
Project name	Hazardous Area Monitoring for Industrial Plant powered by IoT
Maximum marks	2 marks

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic,byte* payload, unsigned int payloadLength);
#define ORG "7jhiq2"
#define DEVICE_TYPE "iot"
#define DEVICE_ID "2002"
#define TOKEN "91132798"
String data3;

char server[]= ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[]="iot-2/evt/distance/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 2
#define TRIG_PIN 4
#define led 5

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(2000);
}

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="";
}
}

```

Wokwi project link:

<https://wokwi.com/projects/347231695817146963>

Alert case:

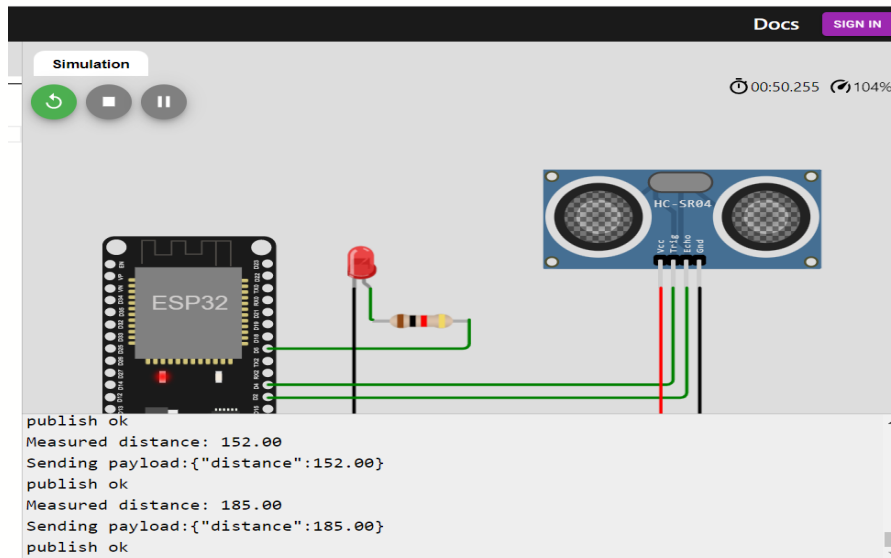
The screenshot displays the Wokwi IDE interface. On the left, the sketch code is visible, featuring an ESP32 setup with an LED (pin 5) and an ultrasonic sensor (HC-SR04). The code includes a callback function that publishes distance data to a topic. On the right, the simulation window shows the physical components connected: the ESP32, a red LED, and the HC-SR04 sensor. The simulation output log at the bottom shows the sensor measuring a distance of 148.00 cm, which is then published as a JSON payload. When the distance changes to 22.00 cm, an 'ALERT' message is published.

```

publish ok
Measured distance: 148.00
Sending payload:{"distance":148.00}
publish ok
Measured distance: 22.00
Sending payload:{"ALERT":22.00}
publish ok

```

Normal case:



Cloud storage:

IBM Watson IoT Platform

2019504534@smartinternz.com
ID: 7jhq2

← Back

Device Drilldown - 2002

	Event	Value	Format	Last Received
Recent Events	distance	{"distance":118}	json	a few seconds ago
	distance	{"ALERT":48}	json	a few seconds ago
	distance	{"distance":108}	json	a few seconds ago

State

This table shows a list of data points that are reported by this device.

Showing Raw Data | No Interfaces Available

Property	Value	Type	Event	Last Received
distance	118	Number	distance	a few seconds ago