**Assignment -4**

|  |  |
| --- | --- |
| Assignment Date | 03 November 2022 |
| Student Name | Madhubala T |
| Student Roll Number | 73771913139 |
| Maximum Marks | 2 Marks |

**Question-1:**

Write code and connections in wokwi for the ultrasonic sensor. 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

|  |
| --- |
| **Solution:** |

**PROGRAM CODE:**

**SKETCHCODE.ino:**

#include <WiFi.h>

#include <PubSubClient.h>

WiFiClient wifiClient;

String data3;

#define ORG "3yngbh"

#define DEVICE\_TYPE "ESP32"

#define DEVICE\_ID "1"

#define TOKEN "234567890"

#define speed 0.034

#define led 14

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char publishTopic[] = "iot-2/evt/display/fmt/json";

char topic[] = "iot-2/cmd/led/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;

PubSubClient client(server, 1883, wifiClient);

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 = "{\"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("Publish OK");

    }

  }

    if(dist>100){

    String payload = "{\"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");

    }else {

**Serial**.println("Publish FAILED");

    }

  }

  }

**Diagram.json:**

{

  "version": 1,

  "author": "Anonymous maker",

  "editor": "wokwi",

  "parts": [

    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 51.33, "left": -80.67, "attrs": {} },

    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -40.1, "left": 35.88, "attrs": {} }

  ],

  "connections": [

    [ "esp:TX0", "$serialMonitor:RX", "", [] ],

    [ "esp:RX0", "$serialMonitor:TX", "", [] ],

    [ "ultrasonic1:VCC", "esp:VIN", "red", [ "v148.58", "h0.31", "v7.33" ] ],

    [ "ultrasonic1:ECHO", "esp:D18", "green", [ "v79.24", "h1.19" ] ],

    [ "ultrasonic1:TRIG", "esp:D5", "green", [ "v85.91", "h0.42" ] ],

    [ "ultrasonic1:GND", "esp:GND.1", "black", [ "v141.91", "h4.64" ] ]

  ]

}

**libraries.txt:**

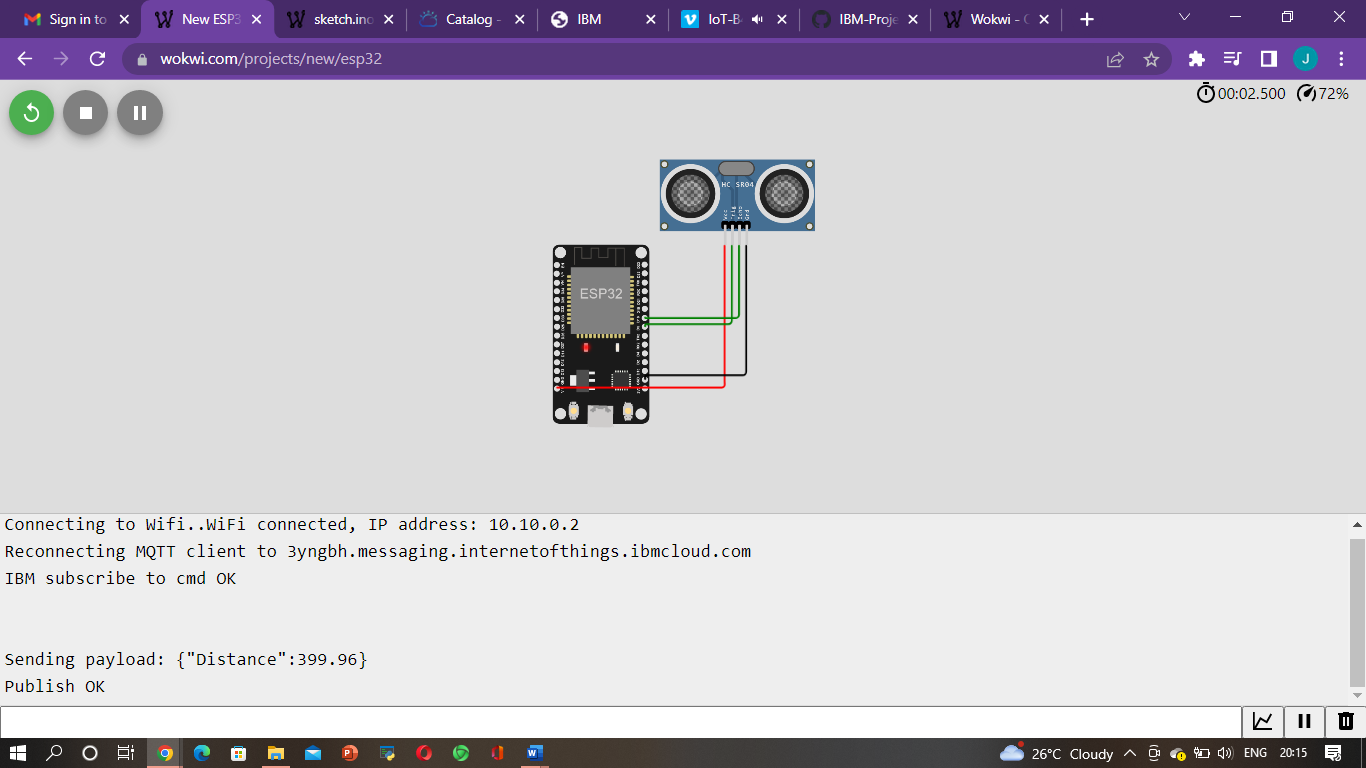
# Wokwi Library List

# See https://docs.wokwi.com/guides/libraries

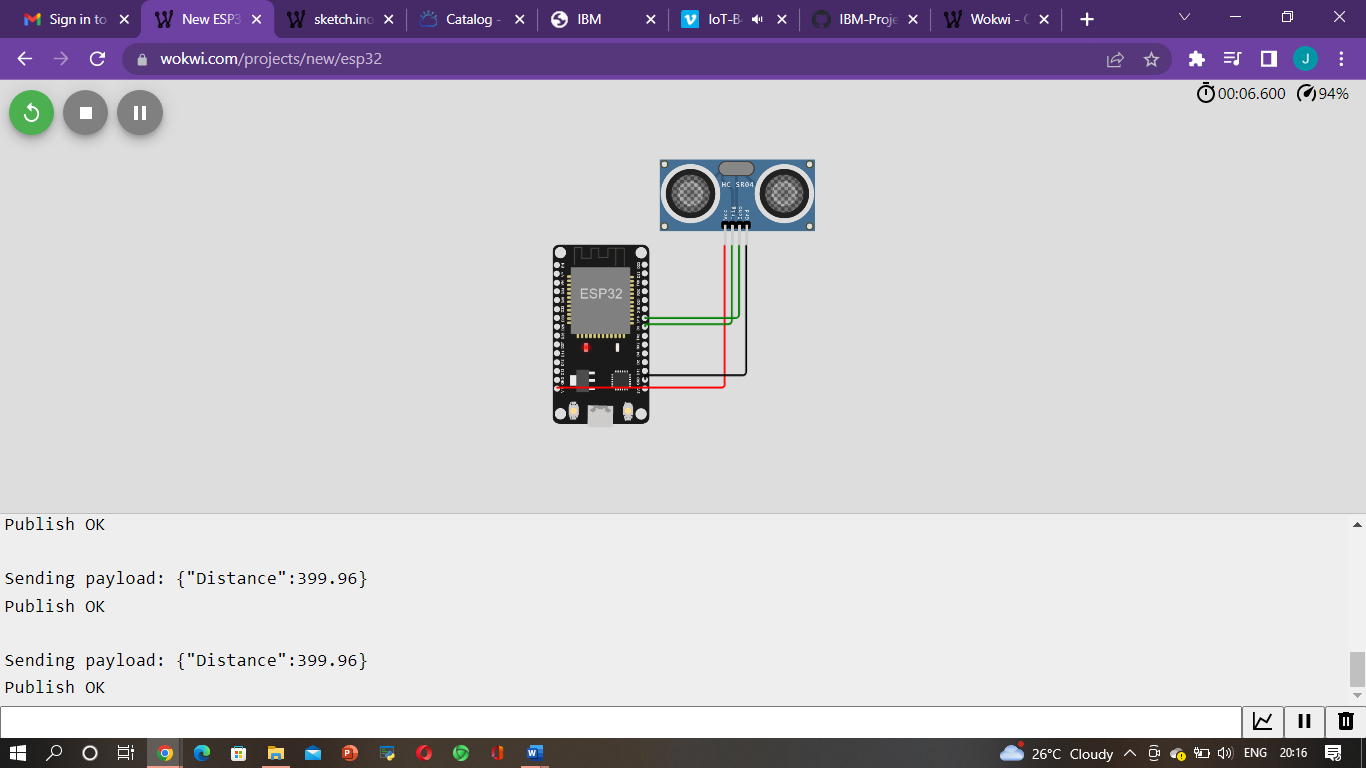
ArduinoJson

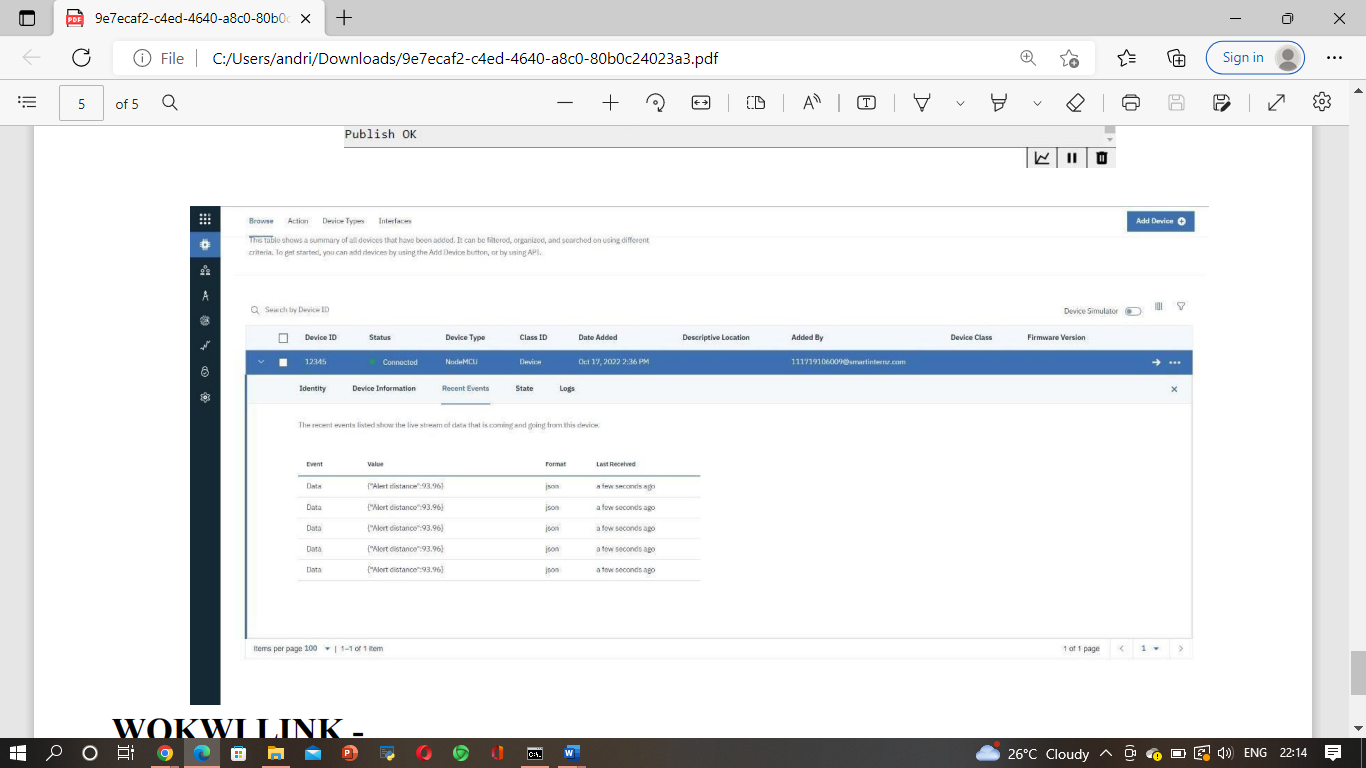
PubSubClient

DHT sensor library for ESPx

****

**OUTPUT:**

****

****

**WOKWI LINK:**

<https://wokwi.com/projects/347283925182186066>