

ASSIGNMENT LINK IN WOKWI:

<https://wokwi.com/projects/364899125531379713>

CONNECTIONS IMAGE:



CODE:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
#include "Ultrasonic.h"
Ultrasonic ultrasonic(2, 4);
float distance;

void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

//-----credentials of IBM Accounts-----

#define ORG "hd5d3x" //IBM ORGANITION ID
#define DEVICE_TYPE "abcd" //Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678" //Token
String data3;
```

```

//float h, t;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
event perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command
type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


//-----
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the
predefined client id by passing parameter like server id,portand
wificredential
void setup()// configureing the ESP32
{
    Serial.begin(115200);

    delay(10);
    Serial.println();
    wificonnect();
    mqttconnect();
}

void loop()// Recursive Function
{
    distance = ultrasonic.read(CM);

    Serial.print("Distance in CM: ");
    Serial.println(distance);
    delay(1000);

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

```

```

/*.....retrieving to
Cloud.....*/

void PublishData(float distance) {
  mqttconnect();//function call for connecting to ibm
  /*
    creating the String in in form JSon to update the data to ibm cloud
  */
  String payload = "{\"distance\":\"";
  payload += distance;

  payload += "\"}";

  Serial.print("Sending payload: ");
  Serial.println(payload);

  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud
    then it will print publish ok in Serial monitor or else it will print publish
    failed
  } else {
    Serial.println("Publish failed");
  }
}

void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }

    initManagedDevice();
    Serial.println();
  }
}

void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
  the connection
  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++) {
        //Serial.print((char)payload[i]);
        data3 += (char)payload[i];
    }

    Serial.println("data: "+ data3);

    data3="";

}

```

SCREENSHOT :

The screenshot displays the IBM Watson IoT Platform dashboard. The top navigation bar includes links for 'Browse', 'Action', 'Device Types', and 'Interfaces', along with an 'Add Device' button. A sidebar on the left contains various icons for navigation. The main content area shows a table of devices with columns for Device ID, Status, Device Type, Class ID, and Date Added. A device with ID 1234 is highlighted, showing a status of 'Connected' and a device type of 'abcd'. Below the table, a detailed view of the selected device is shown, including its identity, device information, recent events, state, and logs. The device information section lists the Device ID (1234), Device Type (abcd), Date Added (May 16, 2023 9:02 PM), Added By (pradeep979174@gmail.com), and Connection Status (Connected). The connection status section provides additional details: Connection Time (May 16, 2023 10:10 PM) and Client Address (50.31.197.67 Insecure).

IBM Watson IoT Platform

pradeep979174@gmail.com
ID: hd5d3x

Browse Action Device Types Interfaces Add Device

Criteria: To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID Device Simulator

Device ID	Status	Device Type	Class ID	Date Added
1234	Connected	abcd	Device	May 16, 2023 9:02 PM

Identify Device Information Recent Events State Logs

Device ID 1234
Device Type abcd
Date Added May 16, 2023 9:02 PM
Added By pradeep979174@gmail.com
Connection Status Connected
Connection Time: May 16, 2023 10:10 PM
Client Address: 50.31.197.67 Insecure

Type here to search 32°C Mostly cloudy 22:11 16-05-2023