

Assignment -4

Team Id	PNT2022TMID52316
Student Name	BHAVITHRA A
Student Roll Number	963519106015

Write code and connection in Wowki for ultrasonic sensor.

Whenever distance is less than 100 cm send “Alert” to IBM cloud and display

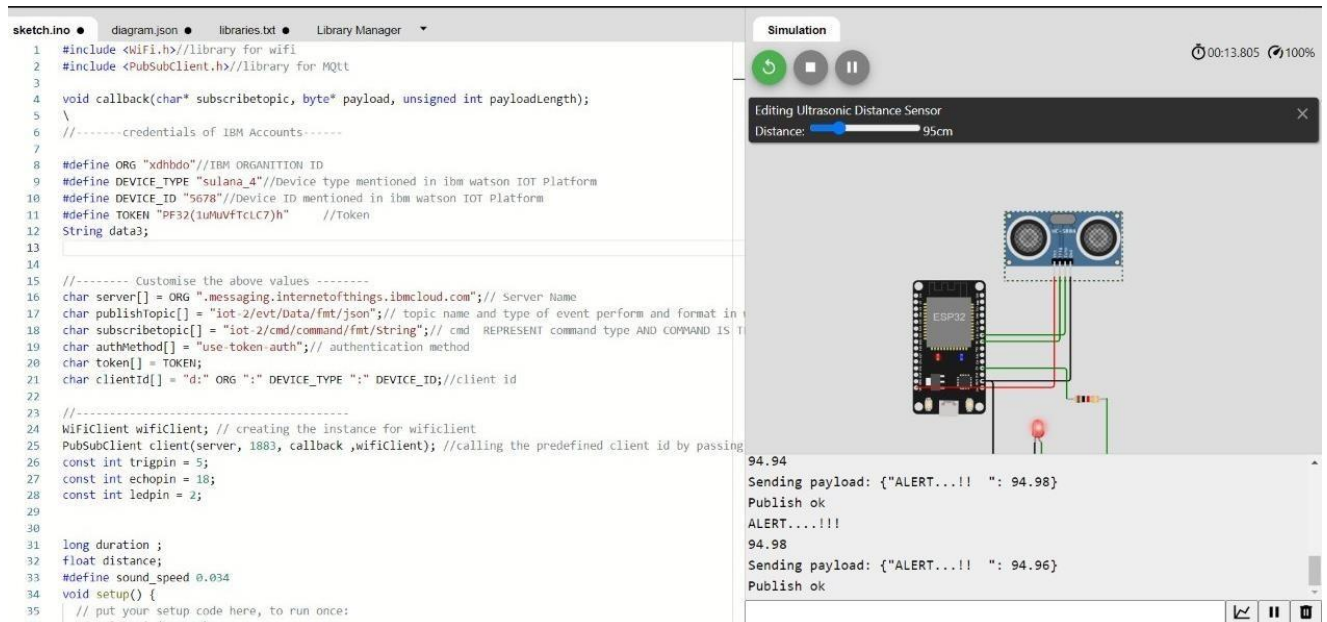
Step 1 : Add new device in IBM cloud

The screenshot displays the 'Browse Devices' interface in the IBM Watson IoT Platform. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A blue 'Add Device' button is located in the top right corner. Below the navigation bar, the 'Browse Devices' section features a search bar and a 'Diagnose' button. A descriptive text states: 'This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.' The main table lists three devices:

	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
>	1234	Disconnected	abcd	Device	Oct 12, 2022 6:29 PM	
>	123456	Disconnected	aaaa	Device	Oct 12, 2022 6:43 PM	
>	5678	Connected	sulana_4	Device	Oct 24, 2022 12:08 PM	

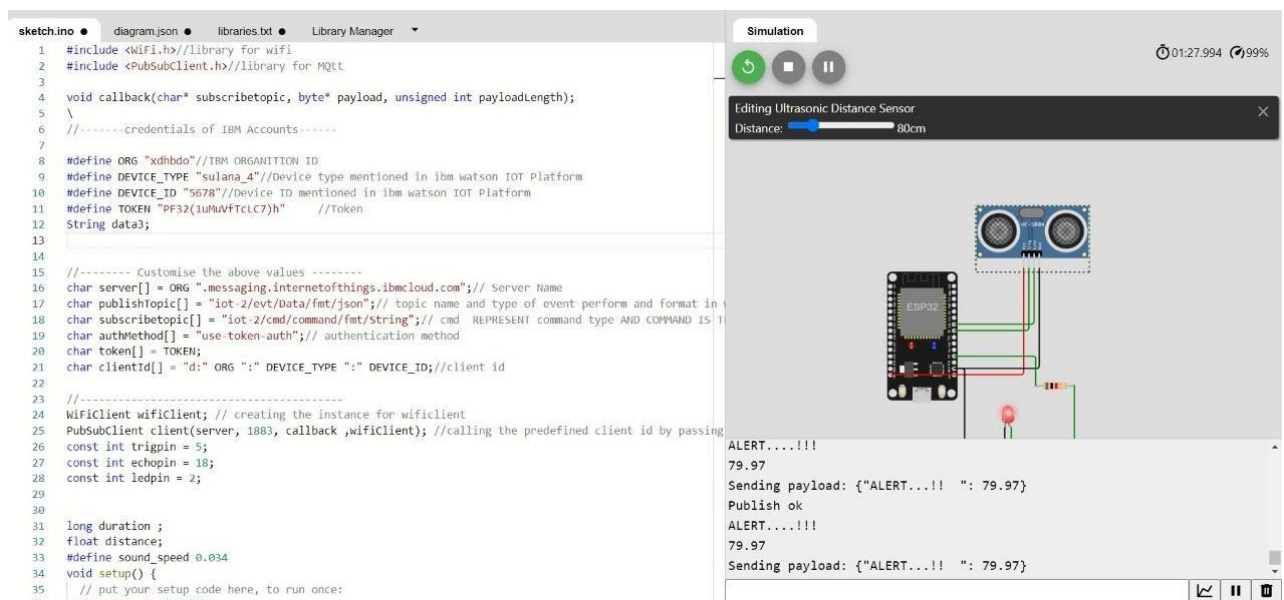
At the bottom of the table, there is a pagination control showing 'Items per page 50' and '1-3 of 3 items'. On the right side of the table, there is a 'Device Simulator' toggle switch and a filter icon.

Step 2: Complete the Circuit and run the program

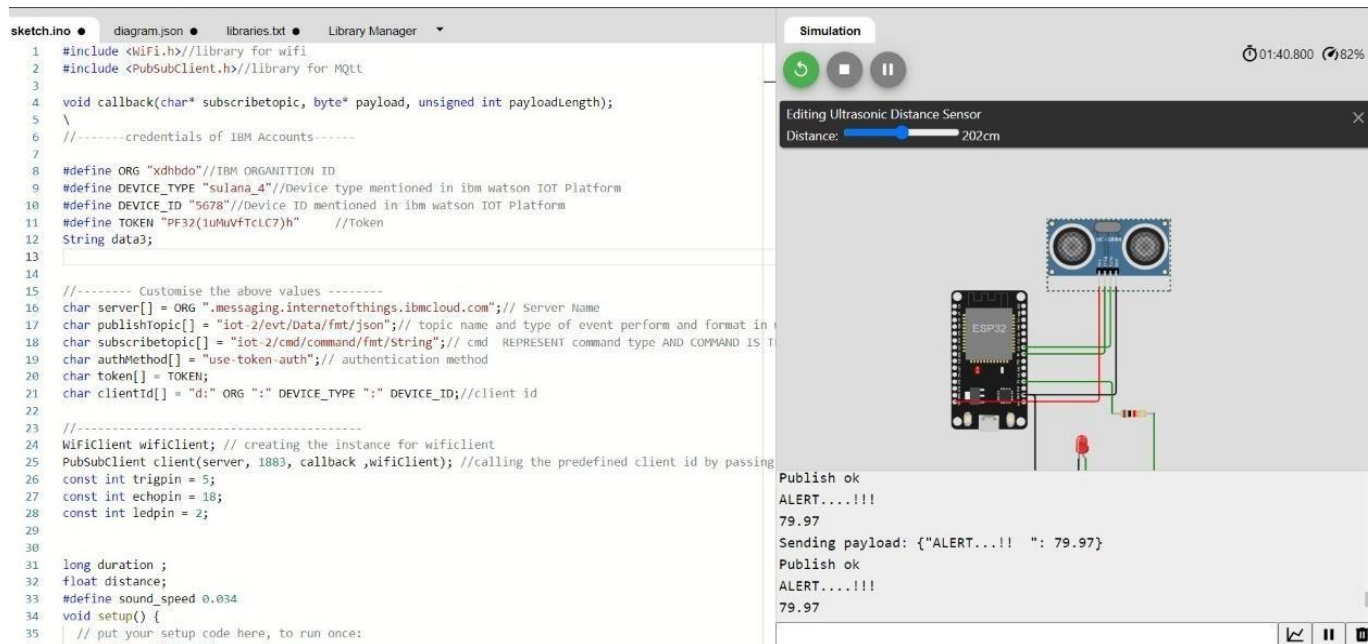


Output in WOWKI

a) when the distance is below 100 cms

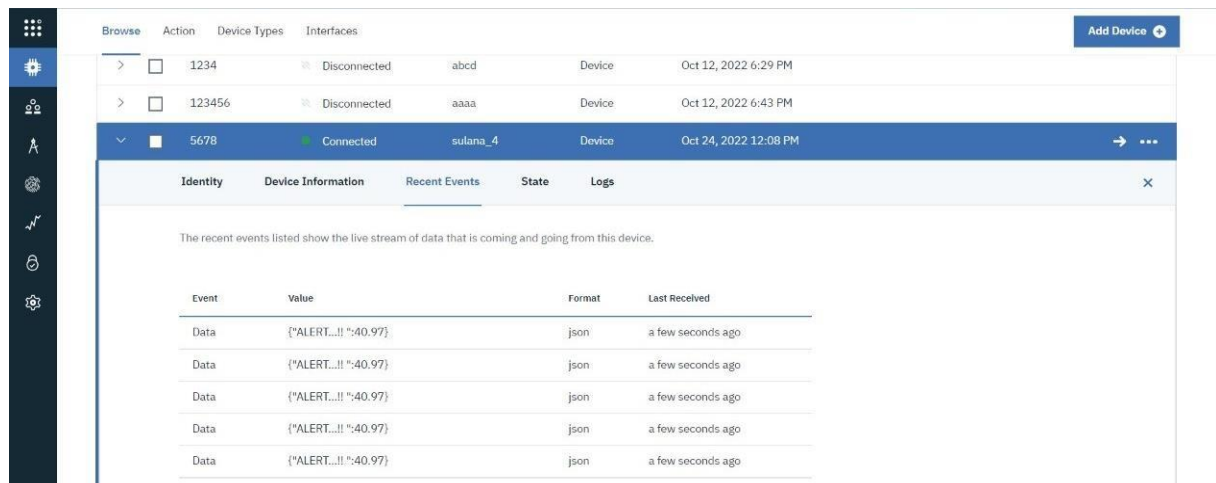


b) when the distance is above 100 cms ,(no alert message is displayed here for 202 cm)



Output in IBM CLOUD (Watson Platform)

Displayed in device recent events



Program Code:

```

#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQTT
void callback(char* subscribtopic, byte* payload, unsigned int payloadLength);
\
//-----credentials of IBM Accounts-----
#define ORG "xdhbdo"//IBM ORGANITION ID
#define DEVICE_TYPE "sulana_4"//Device type mentioned in ibm watson IOT Platform#define
DEVICE_ID "5678"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "PF32(1uMuVfTcLC7)h" //Token
String data3;
//----- 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 andformat in
which data to be send
char subscribtopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT commandtype AND
COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication methodchar 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 bypassing
parameter like server id,portand wificredential
const int trigpin = 5;
const int echopin = 18;
const int ledpin = 2;

long duration ;
float distance;
#define sound_speed 0.034
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, OUTPUT);
  pinMode(ledpin, OUTPUT);
  wificonnect(); mqttconnect();
}

void loop() {
  digitalWrite(trigpin, LOW);
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin, LOW);

  duration= pulseIn(echopin,HIGH);

```



```

distance = duration * sound_speed / 2;
if(distance <= 100){
PublishData(distance);
delay(1000);
if (!client.loop()) {
    mqttconnect();
}
    digitalWrite(ledpin, HIGH);
    Serial.println("ALERT
    ..... !!!")
    ;
    Serial.println(distance);
}
else
{
    digitalWrite(ledpin, LOW);
}
// put your main code here, to run repeatedly:
delay(10); // this speeds up the simulation
}
/*.....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 cloudString payload
        = "{\"ALERT...!! \": ";
        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 printpublish 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 definition 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);
    if(data3=="lighton")
    {
        Serial.println(data3);
    }
    else
    {
        Serial.println(data3);
    }
    data3="";
}

```

