ASSIGNMENT 4

Ultrasonic sensor simulation in Wokwi

TEAM ID:PNT2022TMID36623

Degree & Branch: B.E.-FINAL YEAR-COMPUTER SCIENCE AND ENGINEERING
College: T.J.S ENGINEERING COLLEGE

STUDENT NAME: KUMAR A

RED.No:112819104018

Subject: Professional Readiness for Innovation, Employability & Entrepreneurship (Nalaiya Thiran)

Question :

Write a code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100cms send an “Alert” to IBM cloud and display in the device recent events.

Solution:

Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-------credentials of IBM Accounts------
#define ORG "9lxobn"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "ESP32PROJECT" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/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);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
void setup() {
Serial.begin(115200);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
wificonnect();
mqttconnect();
}
void loop()
{
```

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = duration * SOUND_SPEED/2;
Serial.print("Distance (cm): ");
Serial.println(distance);
if(distance<100)
{
Serial.println("ALERT!!");
delay(1000);
PublishData(distance);
delay(1000);
if (!client.loop()) {
mqttconnect();
}
}
delay(1000);
}
void PublishData(float dist) {
mqttconnect();
String payload = "{\"Distance\":";
payload += dist;
payload += ",\"ALERT!!\":""\"Distance less than 100cms\"";
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 client to ");
Serial.println(server);
while (!!!client.connect(clientId, authMethod, token)) {
Serial.print(".");
delay(500);
}
initManagedDevice();
Serial.println();
}
}
```

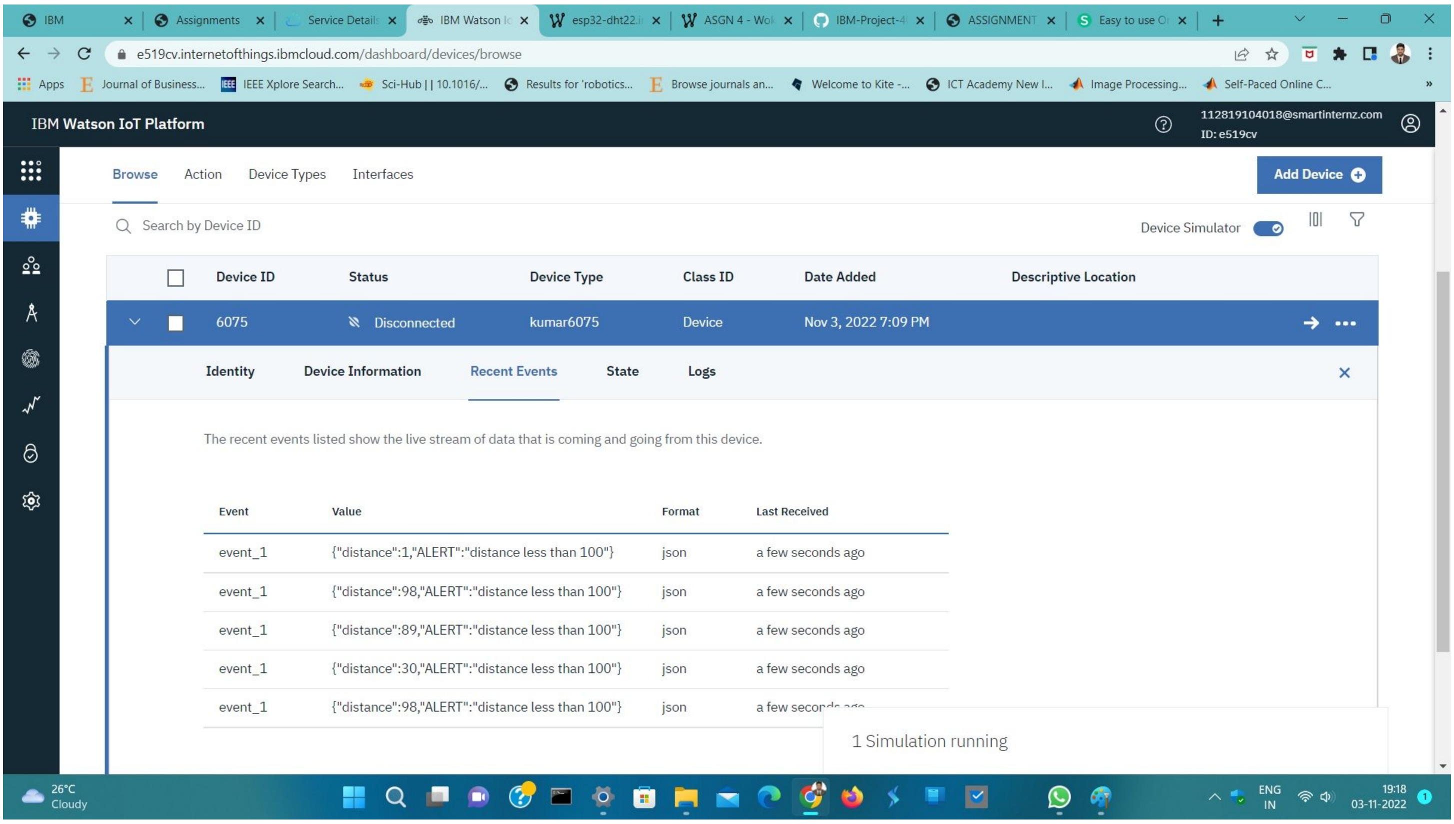
```
void wificonnect()
{
Serial.println();
Serial.print("Connecting to ");
WiFi.begin("Wokwi-GUEST", "", 6);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}
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++) {
//Serial.print((char)payload[i]);
data3 += (char)payload[i];
}
Serial.println("data: "+ data3);
data3="";
}
```

Diagram.json:

```
{
  "version": 1,
  "author": "KUMAR A",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -87.68, "left": -233.71, "attrs": {} },
    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -150.05, "left": -4.82, "attrs": {} }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [
        "esp:VIN",
        "ultrasonic1:VCC",
        "red",
        [ "h-37.16", "v-178.79", "h200", "v173.33", "h100.67"],
      [ "esp:GND.1", "ultrasonic1:GND", "black", [ "h39.87", "v44.04", "h170" ] ],
      [ "esp:D5", "ultrasonic1:TRIG", "green", [ "h54.54", "v85.07", "h130.67" ] ],
      [ "esp:D18", "ultrasonic1:ECHO", "green", [ "h77.87", "v80.01", "h110" ] ]
    ]
  }
```

Output of Program:

Wokwi Project Link: https://wokwi.com/projects/322410731508073042



Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
6075	Disconnected	kumar6075	Device	Nov 3, 2022 7:09 PM	

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
event_1	{"distance":1,"ALERT":"distance less than 100"}	json	a few seconds ago
event_1	{"distance":98,"ALERT":"distance less than 100"}	json	a few seconds ago
event_1	{"distance":89,"ALERT":"distance less than 100"}	json	a few seconds ago
event_1	{"distance":30,"ALERT":"distance less than 100"}	json	a few seconds ago
event_1	{"distance":98,"ALERT":"distance less than 100"}	json	a few seconds ago

Explanation of Program:

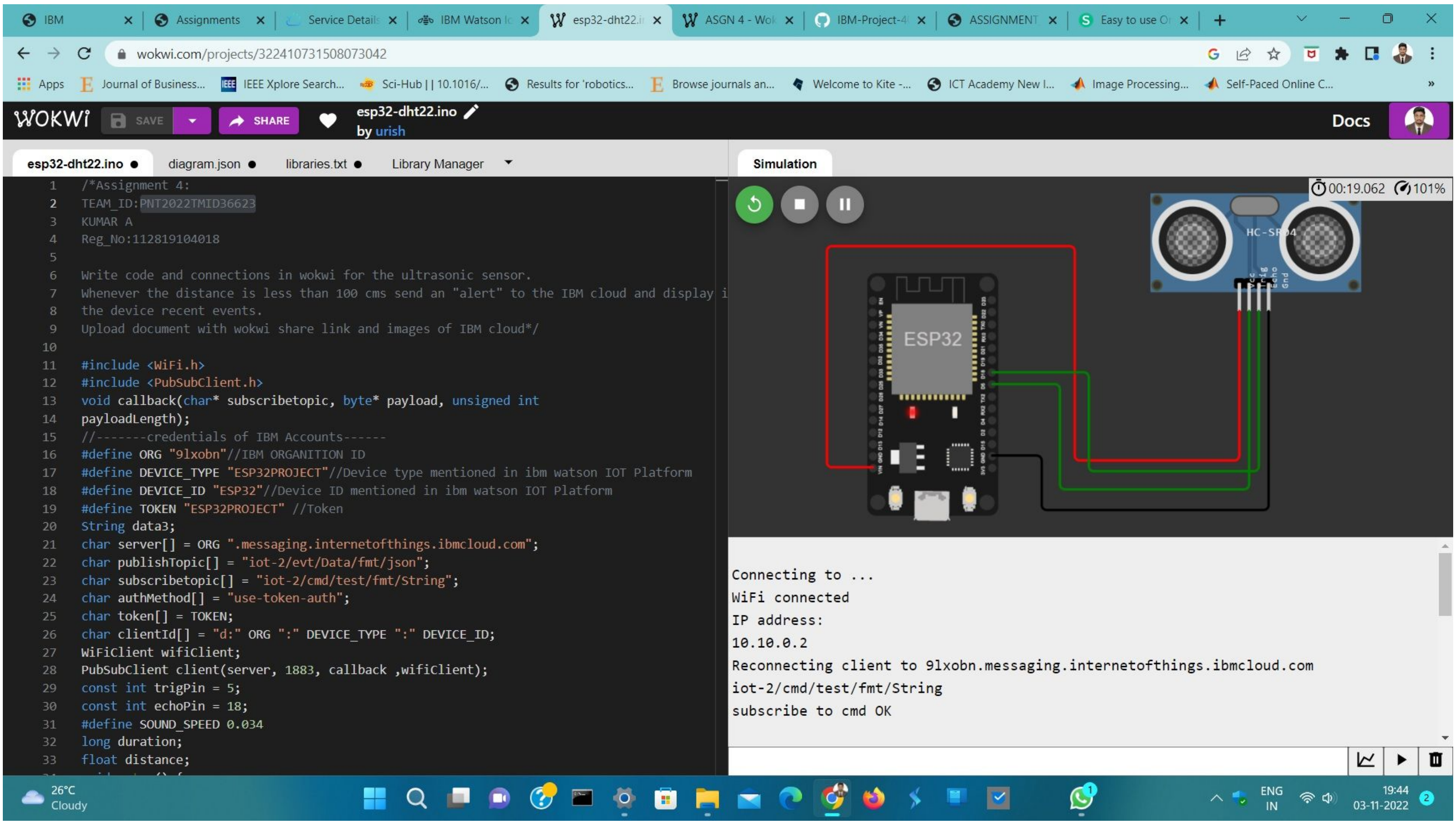
Initially, we have imported the <Wifi.h> and <PubSubClient.h> header files as they are needed to connect wifi and MQTT Protocol. Then, Define Trigger pin and Echo pin values where the ultrasonic sensor is connected with ESP32 module. Then, Define the IBM Account Credentials such as ORG, Device_Type, Device_ID and Token. Also define the server, publishTopic, SubscribeTopic, authMethod, Token and ClientID. Create Object for WifiClient and PubSubClient.

Then, Start the void Setup() Function, Begin the Serial Monitor and set PinMode of Trigger Pin as Output and Echo Pin as Input and call wificonnect() and mqttconnect() to initialize wifi and mqtt Connection and Define their methods to make the Connection.

Then, Begin the void loop() function, digitalWrite HIGH to Trigger Pin and create a delay of 10 microseconds and write back LOW. Then, use pulseIn() function with Echo Pin to calculate the Duration and calculate the distance. If the Distance is less than 100cm call PublishData() Function to publish the Data to IoT Watson Device.

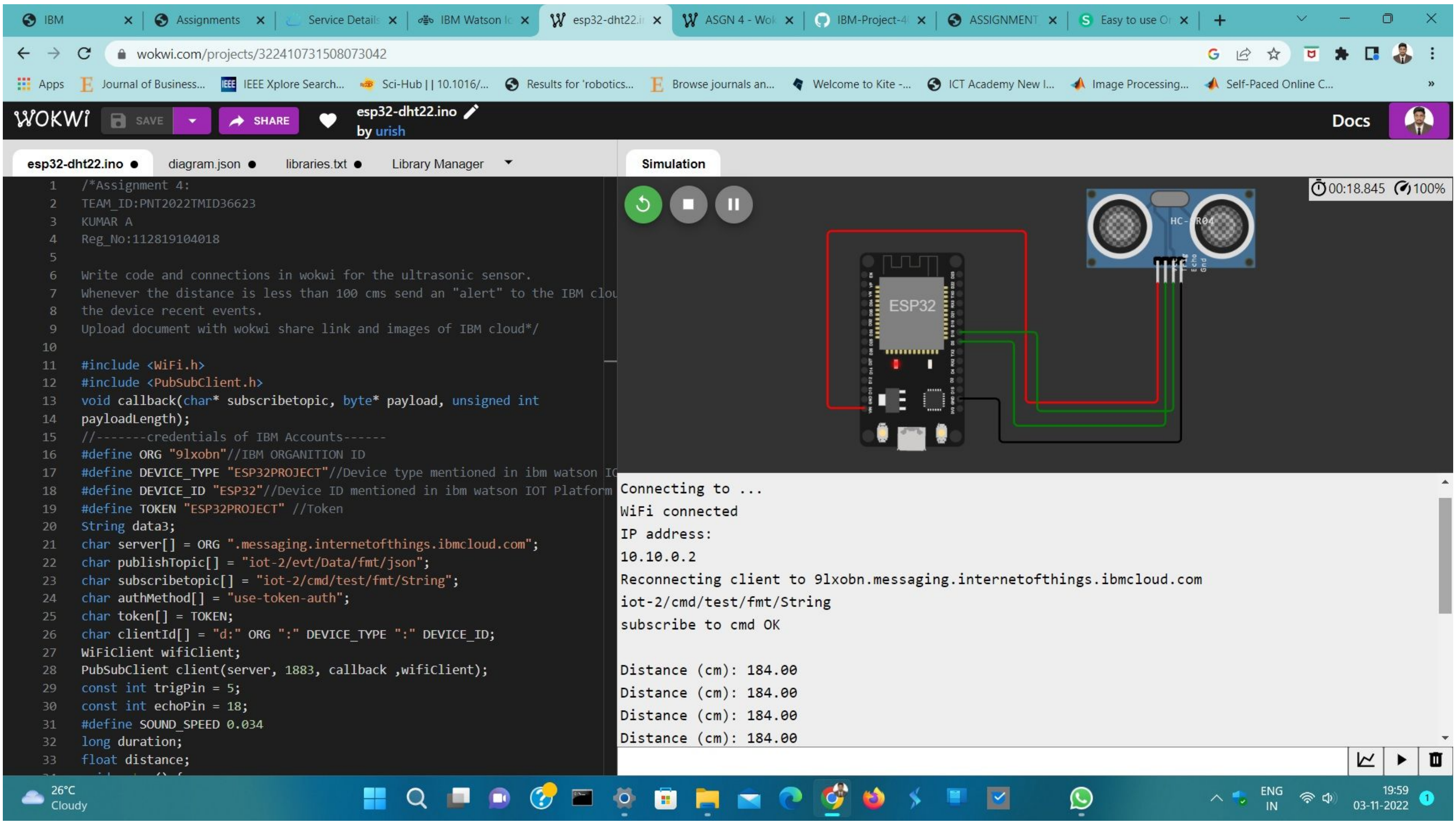
Finally, Define the PublishData() function with message as parameter. Then Define string that contains the payload with the message to be sent in the Json Format. Call Client.publish() function with publishTopic and payload as parameter. Also define the wificonnect() and mqttconnect() to make intial connection with wifi and mqtt connection.

WOKWI OUTPUT:



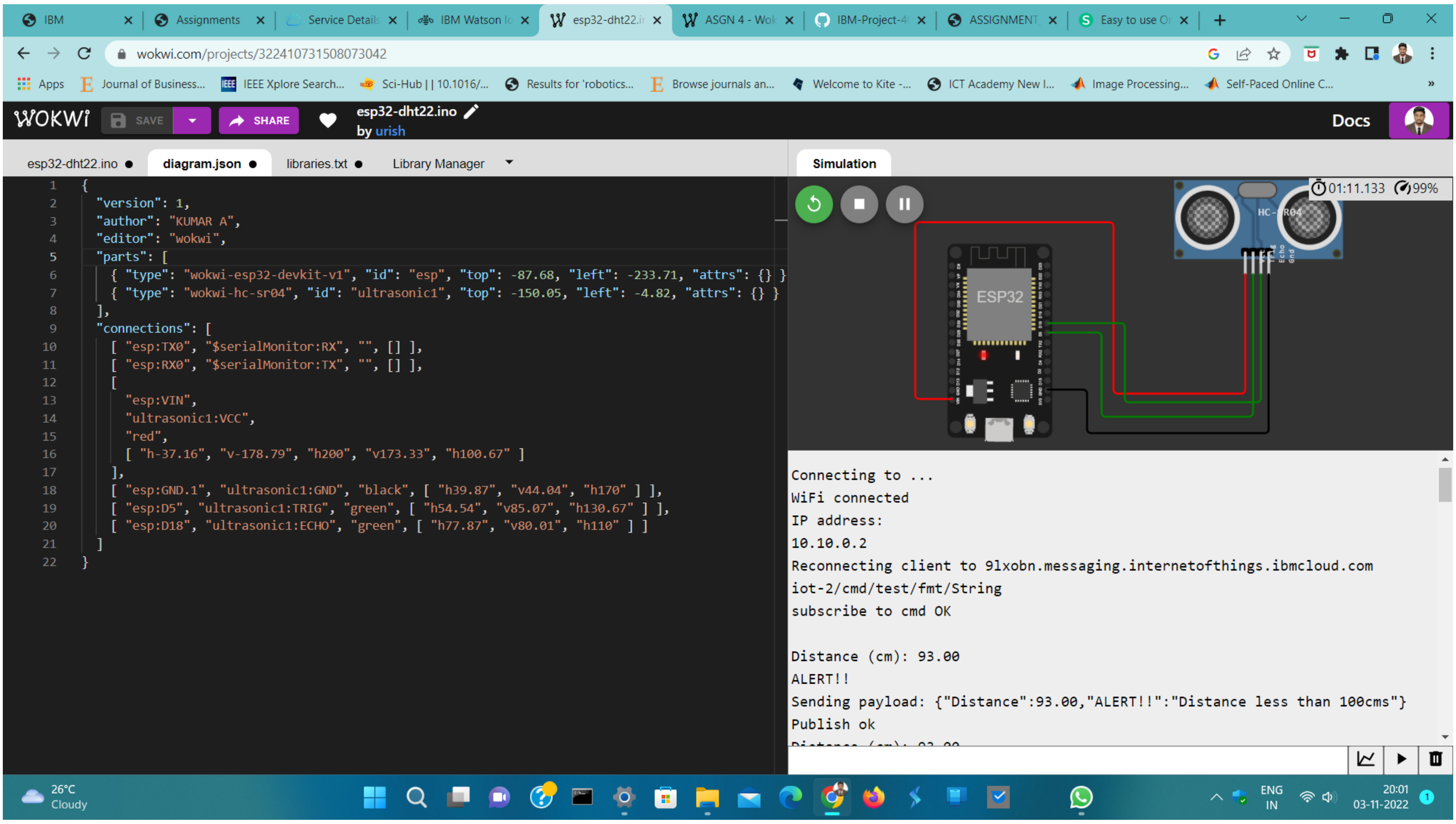
```
/*Assignment 4:
TEAM_ID:PNT2022TMID36623
KUMAR A
Reg_No:112819104018

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 i
the device recent events.
Upload document with wokwi share link and images of IBM cloud*/

#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-------credentials of IBM Accounts------
#define ORG "9lxobn"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "ESP32PROJECT" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/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);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
```

```
Connecting to ...
WiFi connected
IP address:
10.10.0.2
Reconnecting client to 9lxobn.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK
```



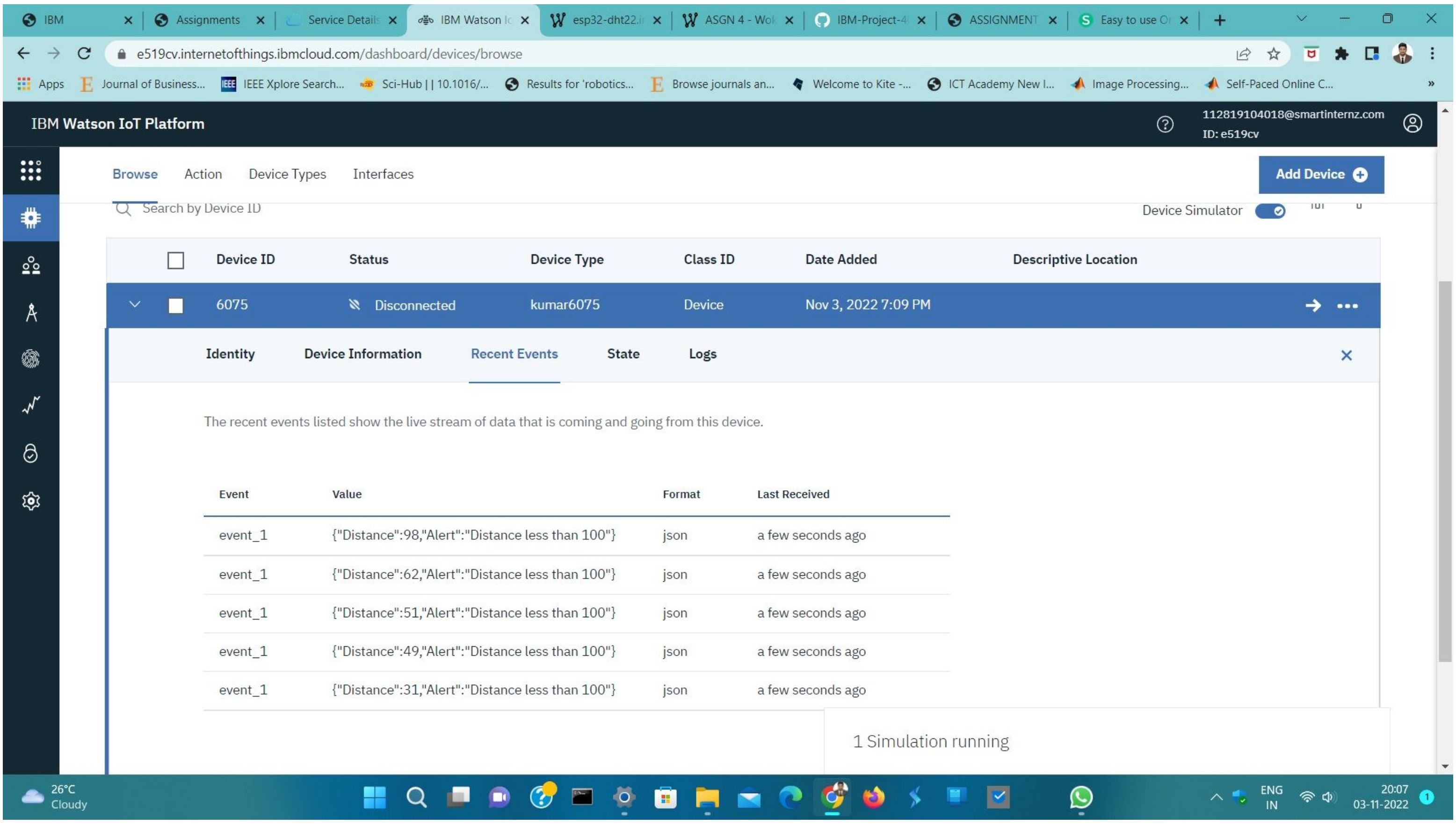
```
/*Assignment 4:
TEAM_ID:PNT2022TMID36623
KUMAR A
Reg_No:112819104018

Write code and connections in wokwi for the ultrasonic sensor.
Whenever the distance is less than 100 cms send an "alert" to the IBM clou
the device recent events.
Upload document with wokwi share link and images of IBM cloud*/

#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-------credentials of IBM Accounts------
#define ORG "9lxobn"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IO
#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "ESP32PROJECT" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/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);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
```

```
Connecting to ...
WiFi connected
IP address:
10.10.0.2
Reconnecting client to 9lxobn.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK

Distance (cm): 184.00
Distance (cm): 184.00
Distance (cm): 184.00
Distance (cm): 184.00
```



```
{
  "version": 1,
  "author": "KUMAR A",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -87.68, "left": -233.71, "attrs": {} }
    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -150.05, "left": -4.82, "attrs": {} }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [
      "esp:VIN",
      "ultrasonic1:VCC",
      "red",
      [ "h-37.16", "v-178.79", "h200", "v173.33", "h100.67" ]
    ],
    [ "esp:GND.1", "ultrasonic1:GND", "black", [ "h39.87", "v44.04", "h170" ] ],
    [ "esp:D5", "ultrasonic1:TRIG", "green", [ "h54.54", "v85.07", "h130.67" ] ],
    [ "esp:D18", "ultrasonic1:ECHO", "green", [ "h77.87", "v80.01", "h110" ] ]
  ]
}
```

```
Connecting to ...
WiFi connected
IP address:
10.10.0.2
Reconnecting client to 9lxobn.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK

Distance (cm): 93.00
ALERT!!
Sending payload: {"Distance":93.00,"ALERT!!":"Distance less than 100cms"}
Publish ok
```



	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
□	6075	Disconnected	kumar6075	Device	Nov 3, 2022 7:09 PM	

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
event_1	{"Distance":98,"Alert":"Distance less than 100"}	json	a few seconds ago
event_1	{"Distance":62,"Alert":"Distance less than 100"}	json	a few seconds ago
event_1	{"Distance":51,"Alert":"Distance less than 100"}	json	a few seconds ago
event_1	{"Distance":49,"Alert":"Distance less than 100"}	json	a few seconds ago
event_1	{"Distance":31,"Alert":"Distance less than 100"}	json	a few seconds ago