ASSIGNMENT 3

COURSE: Naan Mudhalvan (Internet of Things)

Team ID : NM2023TMID15707

Team Leader : SRISABARI S

Team member : SHARUKESHAN R

Team member : VEDANAYAGAM L

Team member : SRUTHIE N

Question:

Build wowki product, use ultrasonic sensor and detect the distance from the object. Whenever distance is less than 100cms upload the value to the ibm cloud.in recent device events upload the data from wokwi.

Example: distance is 20 cms. Upload the 20 value to the ibm cloud in recent event in the ibm iot platform device.

Requirement:

- Wokwi Platform
- Ibm cloud

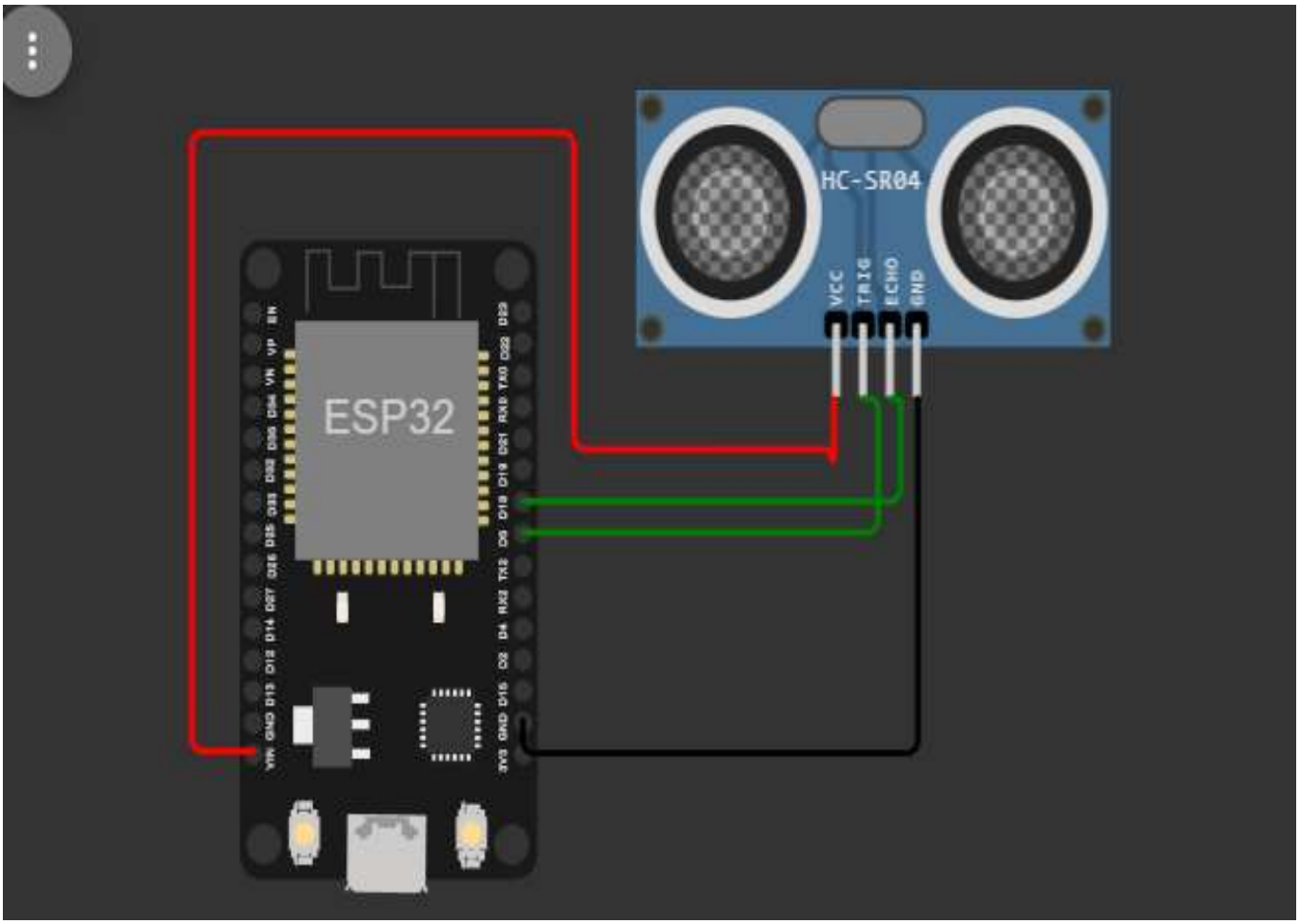
Wokwi Project Link:

https://wokwi.com/projects/364169802473364481

Components Required:

- ESP 32
- Ultrasonic sensor

Connection:



PROGRAM:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned intpayloadLength);
//-------credentials of IBM Accounts- - - -
#define ORG "z0daqf"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson IOTPlatform
#define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOTPlatform
#define TOKEN "kUbT7UdS(+g75i(qcB" //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()
```

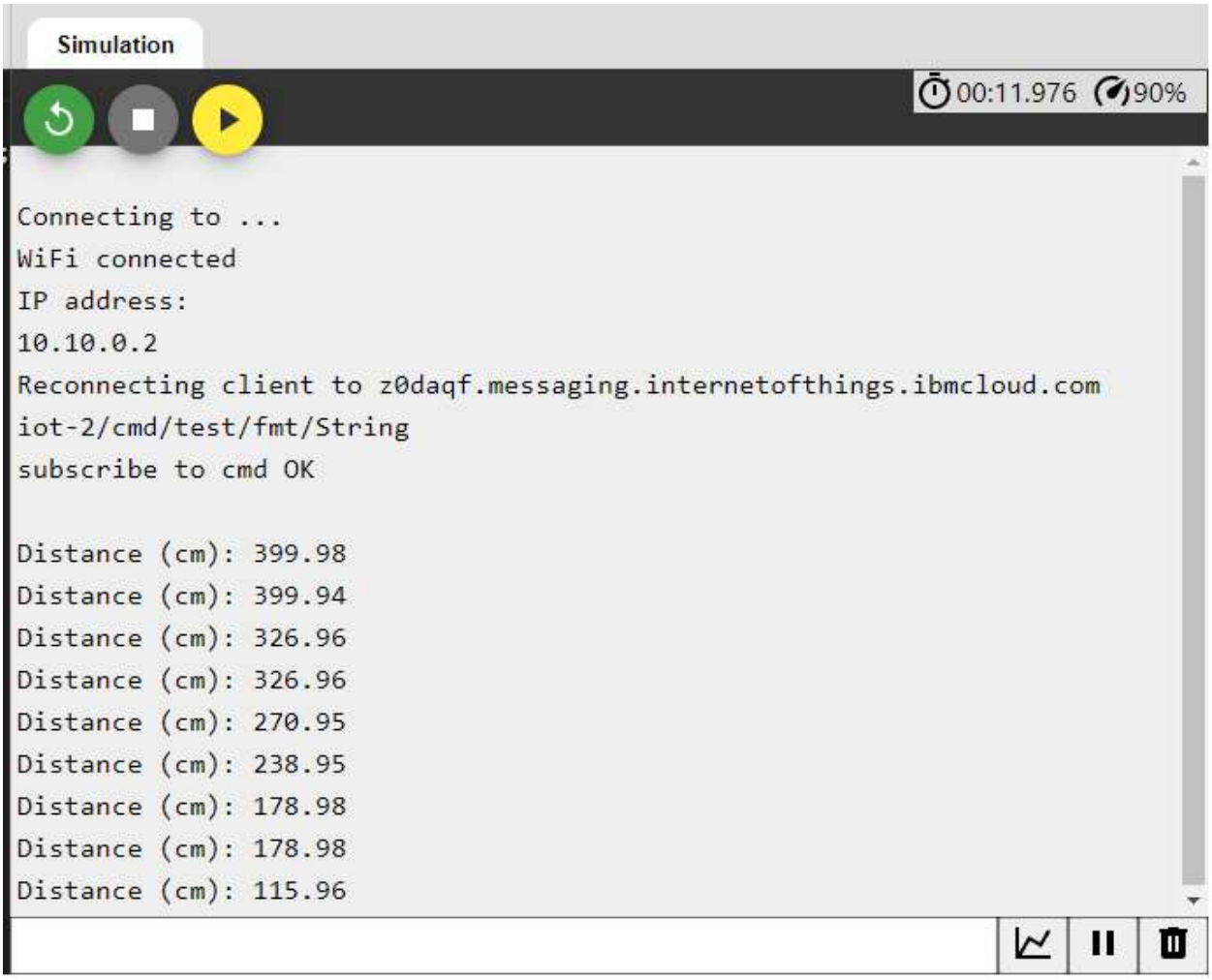
```
 {
   if (client.subscribe(subscribetopic))
    {
     Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
    }
     else {
       Serial.println("subscribe to cmd FAILED");
       }
       }
void callback(char* subscribetopic, byte* payload, unsigned intpayloadLength)
{
 Serial.print("callback invoked for topic: ");
 Serial.println(subscribetopic);
 for(int i = 0; i < intpayloadLength; i++)
 {
  Serial.print((char)payload[i]);
  data3 += (char)payload[i];
  }
  Serial.println("data: "+ data3);
  data3="";
}
```

Diagram.json:

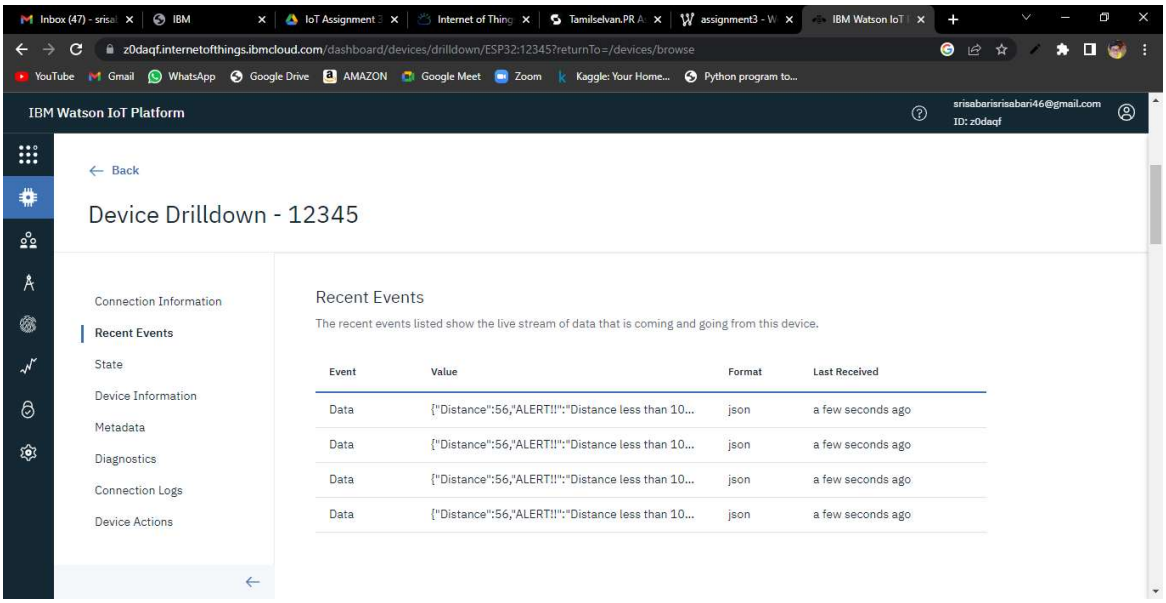

```
{
  "version": 1,
  "author": "SRISABARI S",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 27.31, "left": -103.79,
"attrs": {} },
    {
      "type": "wokwi-hc-sr04",
      "id": "ultrasonic1",
      "top": -17.03,
      "left": 37.63,
      "attrs": { "distance": "116" }
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "esp:D18", "ultrasonic1:ECHO", "green", [ "h135.2", "v-44.63" ] ],
    [ "esp:D5", "ultrasonic1:TRIG", "green", [ "h127.2", "v-51.56" ] ],
    [ "ultrasonic1:GND", "esp:GND.1", "black", [ "v108.93", "h-145.1" ] ],
    [
      "ultrasonic1:VCC",
      "esp:VIN",
```

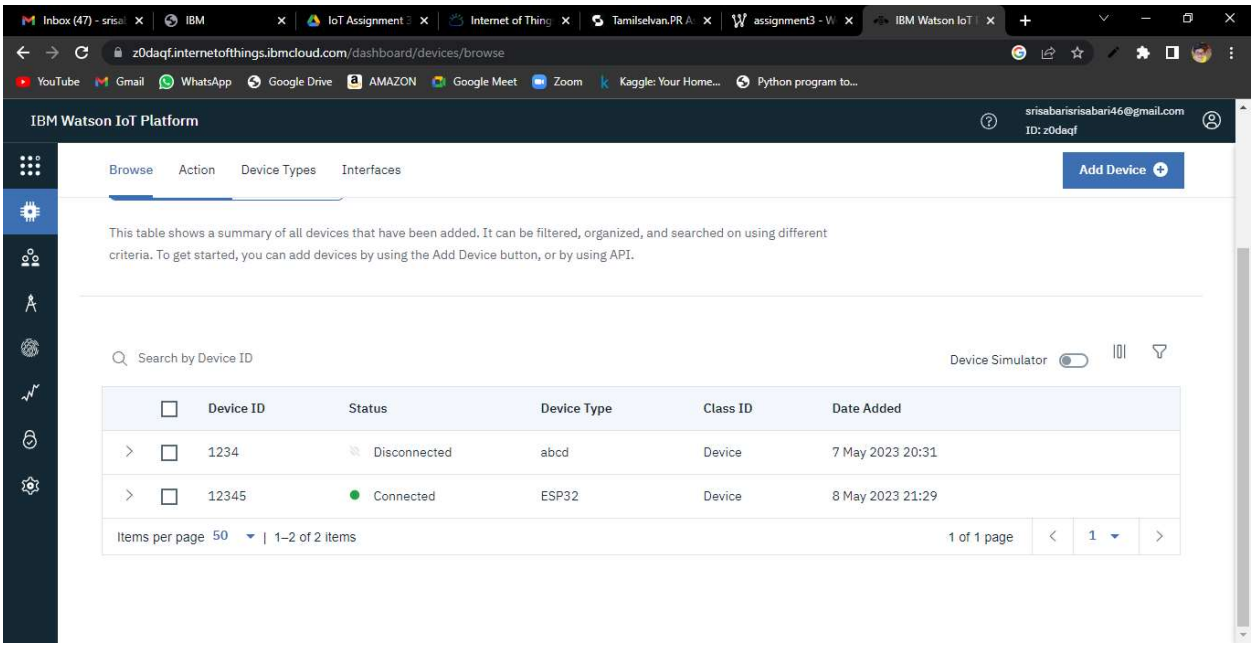
```
    "red",
    [ "v18.93", "h-1.1", "v-3.33", "h-92", "v-97.33", "h-136.67", "v201.33" ]
   ]
  ],
  "dependencies": {}
}
```

Simulation output:



Ibm cloud :


z0daqf.internetofthings.ibmcloud.com/dashboard/devices/drilldown/ESP32:12345?returnTo=/devices/browse
IBM Watson IoT Platform
srisabarisrisabari46@gmail.com
ID: z0daqf
Back
Device Drilldown - 12345
Connection Information
Recent Events
State
Device Information
Metadata
Diagnostics
Connection Logs
Device Actions
Recent Events
The recent events listed show the live stream of data that is coming and going from this device.
Event
Value
Format
Last Received
Data
{"Distance":56,"ALERT!!":"Distance less than 10...
json
a few seconds ago


z0daqf.internetofthings.ibmcloud.com/dashboard/devices/browse
IBM Watson IoT Platform
srisabarisrisabari46@gmail.com
ID: z0daqf
Browse
Action
Device Types
Interfaces
Add Device
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.
Search by Device ID
Device Simulator
Device ID
Status
Device Type
Class ID
Date Added
1234
Disconnected
abcd
Device
7 May 2023 20:31
12345
Connected
ESP32
Device
8 May 2023 21:29
Items per page 50
1–2 of 2 items
1 of 1 page