

IBM ASSIGNMENT -04

Assignment Date	06 November 2022
Student Name	Sowmiya N
Student Roll Number	710019106044
Team ID	PNT2022TMID42278

Write code and connections in Wokwi for ultrasonic sensor.
Whenever distance is less than 100 cms send "alert" to IBM cloud and display in device recent events.

CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "7v14xw"//IBM ORGANITION ID
#define DEVICE_TYPE "Sowmi"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "Sowmi@12" //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);

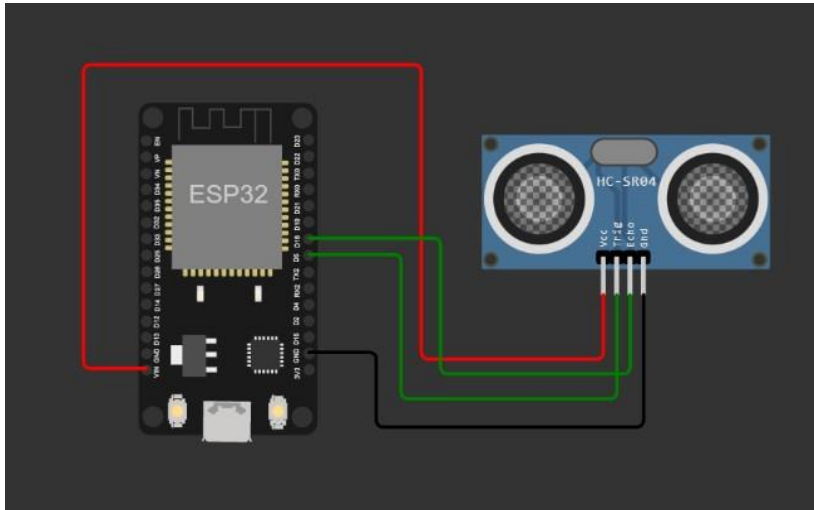
  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");
  } else {
    Serial.println("Publish failed");
  }
}

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="";
}
```

Schematic Diagram:



WOKWI OUTPUT:

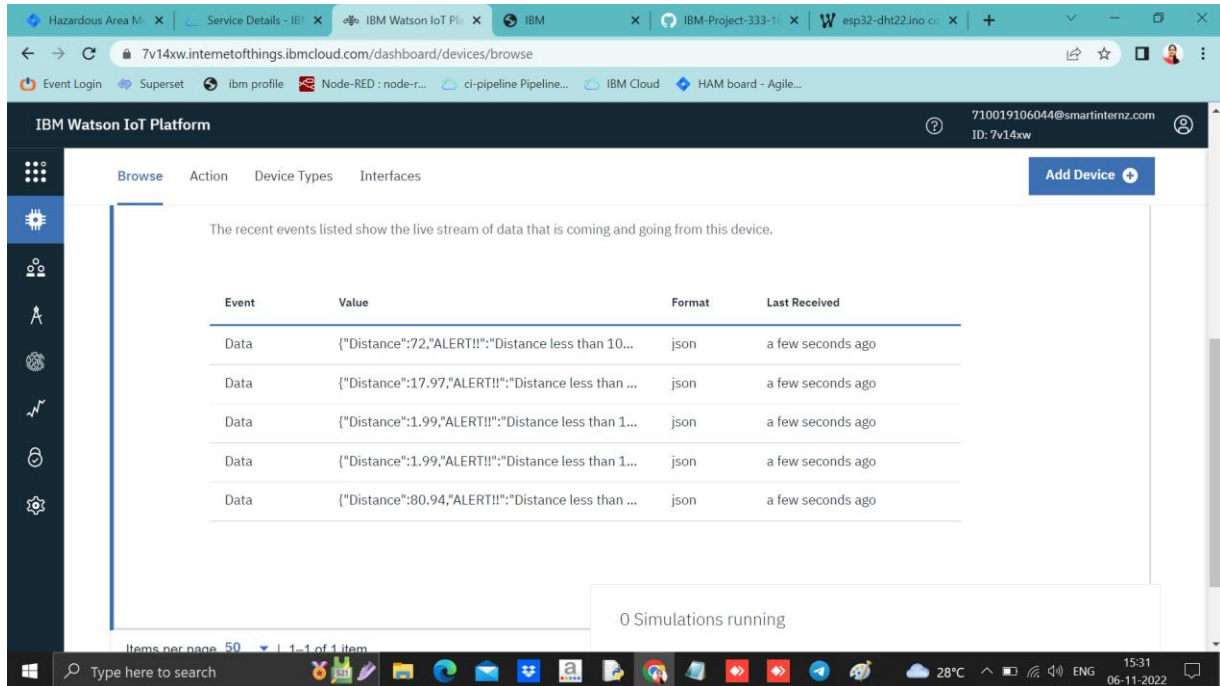
WOKWI interface showing the simulation of the ESP32 and HC-SR04 sensor. The code in the left pane is as follows:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int
4   payloadLength);
5 //-----credentials of IBM Accounts-----
6 #define ORG "7v14xw"//IBM ORGANIZATION ID
7 #define DEVICE_TYPE "Sowmi"//Device type mentioned in ibm watson IOT Pla
8 #define DEVICE_ID "1"//Device ID mentioned in ibm watson IOT Platform
9 #define TOKEN "Sowmi@12" //Token
10 String data3;
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Data/fmt/json";
13 char subscribetopic[] = "iot-2/cmd/test/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 WiFiClient wificlient;
18 PubSubClient client(server, 1883, callback ,wificlient);
19 const int trigPin = 5;
20 const int echoPin = 18;
21 #define SOUND_SPEED 0.034
22 long duration;
23 float distance;
24 void setup() {
25   Serial.begin(115200);
26   pinMode(trigPin, OUTPUT);
```

The right pane shows the simulation results, including a distance sensor output window displaying the distance in cm:

Distance (cm)
110.96
110.94
110.96
110.96
110.96
110.96
110.96

IBM CLOUD OUTPUT:



The screenshot displays the IBM Watson IoT Platform dashboard. The browser address bar shows the URL: `7v14xw.internetofthings.ibmcloud.com/dashboard/devices/browse`. The dashboard header includes the IBM Watson IoT Platform logo and a user profile section with the email `710019106044@smartinternz.com` and ID `7v14xw`. The main content area is titled "Browse" and shows a table of recent events. The table has four columns: "Event", "Value", "Format", and "Last Received". The events are listed as "Data" with JSON values containing distance and alert information, all in "json" format, and received "a few seconds ago". A status bar at the bottom indicates "0 Simulations running".

Event	Value	Format	Last Received
Data	{"Distance":72,"ALERT!!":"Distance less than 10..."}	json	a few seconds ago
Data	{"Distance":17.97,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":1.99,"ALERT!!":"Distance less than 1...	json	a few seconds ago
Data	{"Distance":1.99,"ALERT!!":"Distance less than 1...	json	a few seconds ago
Data	{"Distance":80.94,"ALERT!!":"Distance less than ...	json	a few seconds ago

WOKWI LINK:

<https://wokwi.com/projects/347571375433581140>