

IBM ASSIGNMENT -04

Assignment Date	07 November 2022
Student Name	Chandhni C
Student Roll Number	710019106009
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 "eokk9e"//IBM ORGANITION ID

#define DEVICE_TYPE "Chandhnidevice"//Device type mentioned in ibm watson
IOT Platform

#define DEVICE_ID "2804"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "Chandhni82" //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();
    }
  }
}

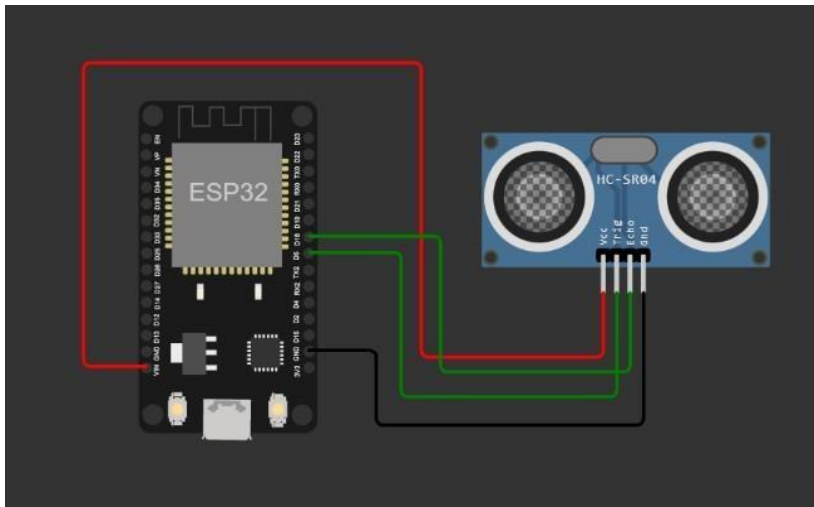
```

```

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:

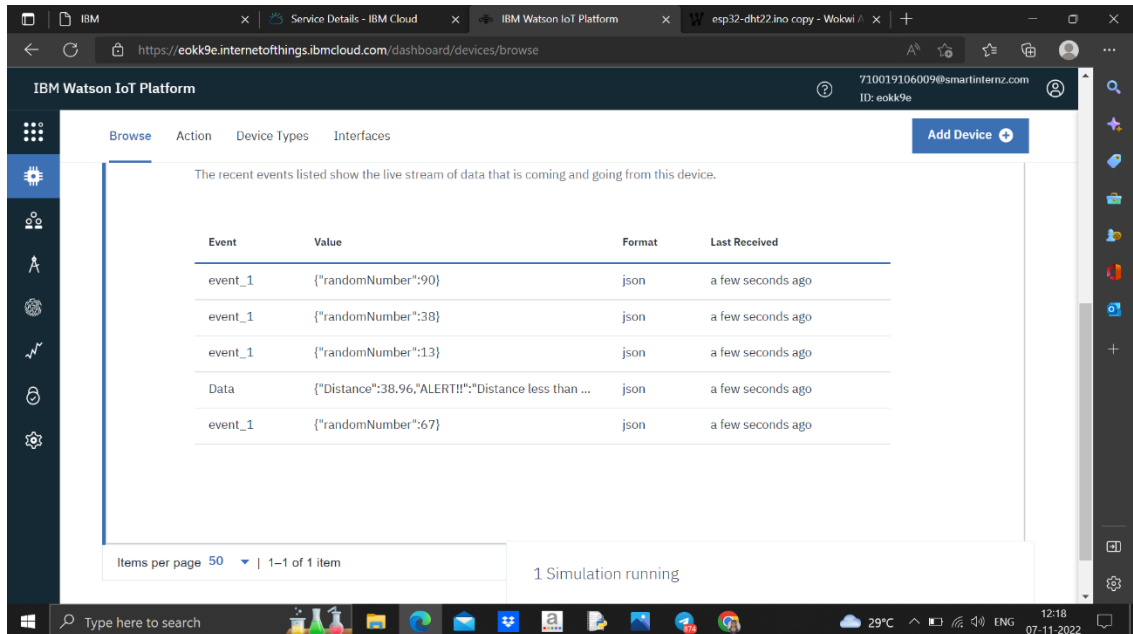
WOKWI interface showing the simulation of the ESP32-dht22.ino copy project. The code is displayed on the left, and the simulation results are shown on the right.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribtopic, byte* payload, unsigned int
4 payloadLength);
5 //-----credentials of IBM Accounts-----
6 #define ORG "eokk9e"//IBM ORGANITION ID
7 #define DEVICE_TYPE "Chandhndidevice"//Device type mentioned in ibm wa
8 #define DEVICE_ID "2804"//Device ID mentioned in ibm watson IOT Platf
9 #define TOKEN "Chandhni82" //Token
10 String data3;
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Data/fmt/json";
13 char subscribtopic[] = "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);
27   pinMode(echoPin, INPUT);
28   wificonnect();
```

Simulation results:

Distance (cm): 85.97
ALERT!!
Sending payload: {"Distance":85.97,"ALERT!!":"Distance less than 100cms"}
Publish ok
Distance (cm): 155.94
Distance (cm): 169.98

IBM CLOUD OUTPUT:



The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes tabs for 'Service Details - IBM Cloud', 'IBM Watson IoT Platform', and 'esp32-dht22.ino copy - Wokwi'. 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 listed are 'event_1' with values like '{"randomNumber":90}', '{"randomNumber":38}', and '{"randomNumber":13}', and a 'Data' event with a value '{"Distance":38.96,"ALERT!";"Distance less than ...}'. The bottom status bar indicates '1 Simulation running'.

Event	Value	Format	Last Received
event_1	{"randomNumber":90}	json	a few seconds ago
event_1	{"randomNumber":38}	json	a few seconds ago
event_1	{"randomNumber":13}	json	a few seconds ago
Data	{"Distance":38.96,"ALERT!";"Distance less than ...	json	a few seconds ago
event_1	{"randomNumber":67}	json	a few seconds ago

WOKWI LINK:

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