

## **IBM Assignment - 4**

***Done by Sanjay Karthick R***

### **QUESTION :**

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. Upload document with wokwi share link and images of IBM cloud.

**CODE:**

```

#include
<WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>

WiFiClient wifiClient;

#define ORG
"1bklkq"
#define DEVICE_TYPE "abcd"
#define DEVICE_ID "rasp"
#define TOKEN "12345678"
#define speed
0.034

char server[] = ORG
".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-
2/evt/abcd_1/fmt/json"; char topic[] = "iot-
2/cmd/home/fmt/String"; char authMethod[] =
"use-token-auth"; char token[] = TOKEN; char
clientId[] = "d:" ORG ":" DEVICE_TYPE ":"
DEVICE_ID; PubSubClient client(server, 1883,
wifiClient); void publishData();
const int
trigpin=5;
const int
echopin=18;
String command;
String data="";
String lat="14.167589";
String lon="80.248510";
String name="point2";
String icon="";

long duration;
int dist;
void
setup()
{

```

```

    Serial.begin(115200);
    pinMode(trigpin,
    OUTPUT);
    pinMode(echopin,
    INPUT); wifiConnect();
    mqttConnect();
}
void loop()
{

    publishData();
    delay(500);
    if (!client.loop())
    {
        mqttConnect();
    }
}
void
wifiConnect()
{
    Serial.print("Connecting      to      ");
    Serial.print("Wifi");
    WiFi.begin("Wokwi-GUEST", "", 6); while
    (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.print("WiFi connected, IP address: ");
    Serial.println(WiFi.localIP());
}

void mqttConnect() {
    if (!client.connected()) {
        Serial.print("Reconnecting MQTT client to ");
        Serial.println(server); while
        (!client.connect(clientId, authMethod, token))
        { Serial.print("."); delay(1000);
        }
        initManagedDevic
        e();
    }
}

```

```

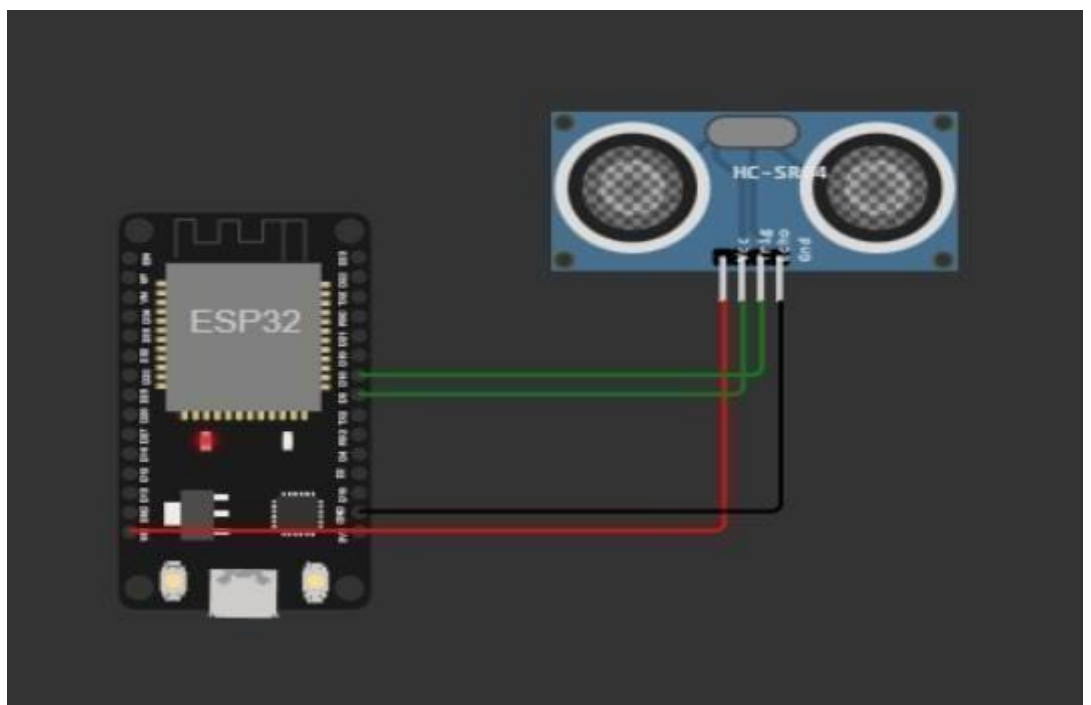
        Serial.println();
    }
}
void
initManagedDevice()
{
    if (client.subscribe(topic)) {
        Serial.println(client.subscribe(topic));
        Serial.println("subscribe to cmd OK");
    } else {
        Serial.println("subscribe to cmd FAILED");
    } } void
publishData()
{ digitalWrite(trigpin,LOW);
  digitalWrite(trigpin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin,LOW);
  duration=pulseIn(echopin,HIGH);
  dist=duration*speed/2;

  if(dist<100){
    dist=100-dist;
    icon="fa-trash";
  }else{ dist=0;
    icon="fa-trash-o";
  }
  DynamicJsonDocument
  doc(1024); String
  payload;
  doc["Name"]=name;
  ++++doc["Latitude"]=lat;
  doc["Longitude"]=lon;
  doc["Icon"]=icon;
  doc["FillPercent"]=dist;

```

```
serializeJson(doc,  
payload); delay(3000);  
Serial.print("\n");  
Serial.print("Sending payload: ");  
Serial.println(payload);  
if (client.publish(publishTopic, (char*)  
    payload.c_str())) { Serial.println("Publish OK");  
} else {  
    Serial.println("Publish FAILED");  
}  
}
```

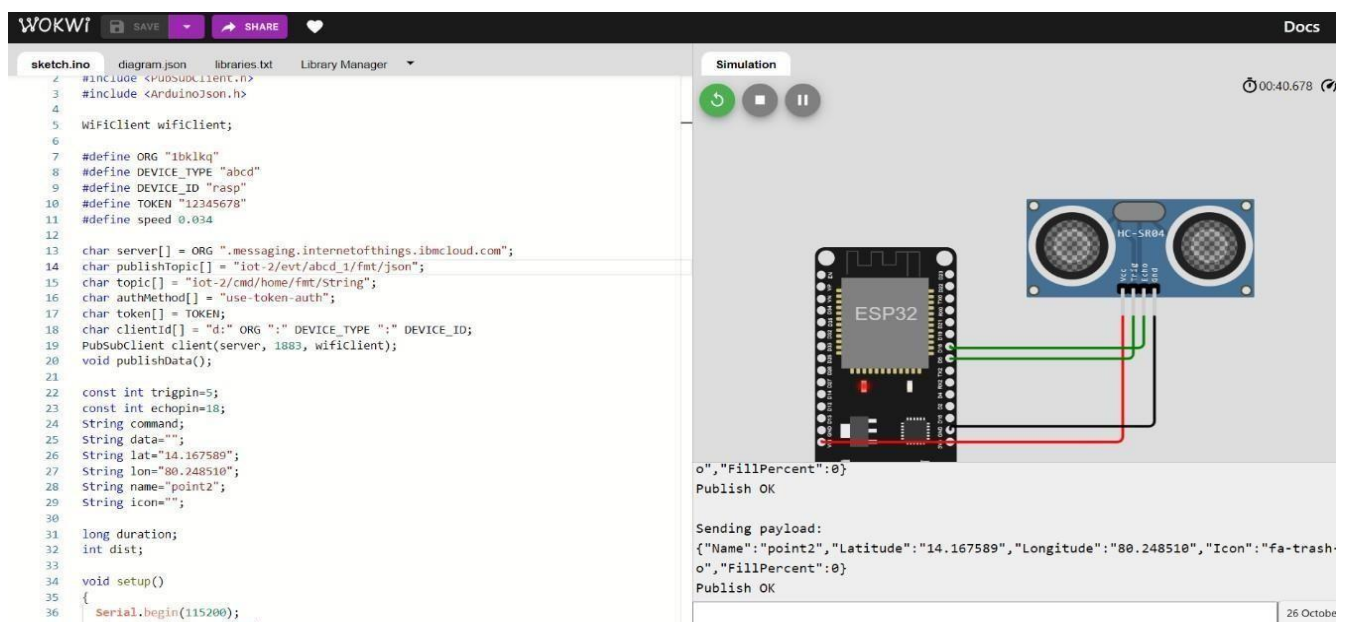
## Connections:



## WOKWI LINK:

<https://wokwi.com/projects/346587874175484499>

## OUTPUT:



The screenshot displays the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, showing the following code:

```
1 #include <PubSubClient.h>
2 #include <ArduinoJson.h>
3
4 WiFiClient wifiClient;
5
6 #define ORG "Ibklkq"
7 #define DEVICE_TYPE "abcd"
8 #define DEVICE_ID "rasp"
9 #define TOKEN "12345678"
10 #define speed 0.034
11
12 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
13 char publishTopic[] = "iot-2/evt/abcd_1/fmt/json";
14 char topic[] = "iot-2/cmd/home/fmt/String";
15 char authMethod[] = "use-token-auth";
16 char token[] = TOKEN;
17 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
18 PubSubClient client(server, 1883, wifiClient);
19 void publishData();
20
21 const int trigpin=5;
22 const int echopin=18;
23 String command;
24 String data="";
25 String lat="14.167589";
26 String lon="80.248510";
27 String name="point2";
28 String icon="";
29
30 long duration;
31 int dist;
32
33 void setup()
34 {
35   Serial.begin(115200);
36 }
```

On the right, the 'Simulation' window shows a visual representation of the hardware. An ESP32 microcontroller is connected to an HC-SR04 ultrasonic sensor. The sensor's VCC pin is connected to the ESP32's 5V pin, and its GND pin is connected to the ESP32's GND pin. The sensor's TRIG pin is connected to the ESP32's pin 5, and its ECHO pin is connected to the ESP32's pin 18. Below the simulation, the output console shows the following messages:

```
o","FillPercent":0}
Publish OK

Sending payload:
{"Name":"point2","Latitude":"14.167589","Longitude":"80.248510","Icon":"fa-trash-o","FillPercent":0}
Publish OK
```

The bottom right corner of the interface shows the date '26 October'.

Identity    Device Information    **Recent Events**    State    Logs

Event	Value	Format	Last Received
event_1	{"Alert Distance":83}	json	a few seconds ago
event_1	{"Alert Distance":59}	json	a few seconds ago
event_1	{"Alert Distance":7}	json	a few seconds ago
event_1	{"Alert Distance":30}	json	a few seconds ago
event_1	{"Alert Distance":51}	json	a few seconds ago