

NALAIYA THIRAN

ASSIGNMENT-4

**USER CASE: PERSONAL ASSISTANCE FOR SENIORS WHO ARE SELF RELIANT
BY TEAM MEMBER 3: KAMAL RAJ.R**

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

CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient wifiClient;
String data3;
#define ORG "3eqctu"
#define DEVICE_TYPE "ESP32"
#define DEVICE_ID "0000"
#define TOKEN "123456789"
#define speed 0.034
#define led 14
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/shreedharen/fmt/json";
char topic[] = "iot-2/cmd/led/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
const int trigpin=5;
const int echopin=18;
String command;
String data="";
long duration;
float dist;
void setup()
```

```

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

void loop() {
bool isNearby = dist < 100;
digitalWrite(led, isNearby);
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(500);
}
initManagedDevice();
Serial.println();
}
}
void initManagedDevice() {
if (client.subscribe(topic))
{
Serial.println("IBM 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){
String payload = "{\"Alert Distance\":\"";
payload +=
dist;payload +=
"}";
Serial.print("\n");

```

```
Serial.print("Sending payload:");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str()))
{Serial.println("Publish OK");
}
}
if(dist>100){
String payload =
"{\"Distance\":";payload += dist;
payload += "}";
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");
}
}}
```

OUTPUT:

LINK: <https://wokwi.com/projects/348038026125902419>

The screenshot shows the Wokwi IDE interface. On the left, the sketch.ino file is open, displaying the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 WiFiClient wificlient;
4 String data3;
5 #define ORG "3eqctu"
6 #define DEVICE_TYPE "ESP32"
7 #define DEVICE_ID "0000"
8 #define TOKEN "123456789"
9 #define speed 0.034
10 #define led 14
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/shreedharen/fmt/json";
13 char topic[] = "iot-2/cmd/led/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 PubSubClient client(server, 1883, wificlient);
18 const int trigpin=5;
19 const int echopin=18;
20 String command;
21 String data="";
22 long duration;
23 float dist;
24 void setup()
25 {
26   Serial.begin(115200);
27   pinMode(led, OUTPUT);
28   pinMode(trigpin,
29   OUTPUT);
30   pinMode(echopin, INPUT);
31   wifiConnect();
32   mqttConnect();
33 }
34 void loop() {
35   bool isNearby = dist < 100;
```

On the right, the simulation window shows a virtual circuit with an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sensor is connected to the ESP32's pins 5 (VCC), 18 (GND), and 4 (Trig). The ESP32 is also connected to an LED on pin 14. The simulation is running, and the console shows the following output:

```
Publish OK
Sending payload:{"Distance":399.94}
Publish OK
Sending payload:{"Distance":399.96}
Publish OK
```

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes links for "Browse", "Action", "Device Types", and "Interfaces". The main content area displays a table of devices, with the following columns: "Device ID", "Status", "Device Type", "Class ID", "Date Added", and "Descriptive Location". The table shows a single device with ID "0000", status "Connected", and type "ESP32".

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
0000	Connected	ESP32	Device	Nov 12, 2022 6:40 PM	

Below the table, there is a section for "Recent Events" showing a live stream of data. The events are listed in a table with columns: "Event", "Value", "Format", and "Last Received".

Event	Value	Format	Last Received
shreedharen	{"Distance":399.96}	json	a few seconds ago
shreedharen	{"Distance":399.96}	json	a few seconds ago
shreedharen	{"Distance":399.94}	json	a few seconds ago
shreedharen	{"Distance":399.94}	json	a few seconds ago
shreedharen	{"Distance":399.96}	json	a few seconds ago