

## Assignment-4

Date	24 October 2022
Name	Kamalakkannan A
Roll Number	620119106037
Team ID	PNT2022TMID30870
Project Name	Industry Specific Intelligent Fire Management System

### Question :

Write code and connections in wokwi for ultrasonic sensors. That 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.

### Wokwi:

<https://wokwi.com/projects/348562285697958482>

### Code:

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

WiFiClient wifiClient;

#define ORG "b7rwwm"
#define DEVICE_TYPE "Kamal"
#define DEVICE_ID "kannan"
#define TOKEN "r*(@uoImJ-flu5M_gr"
#define speed 0.034

char server[] = ORG".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/event_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="";
long duration;
float dist;
void setup()
{
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
```

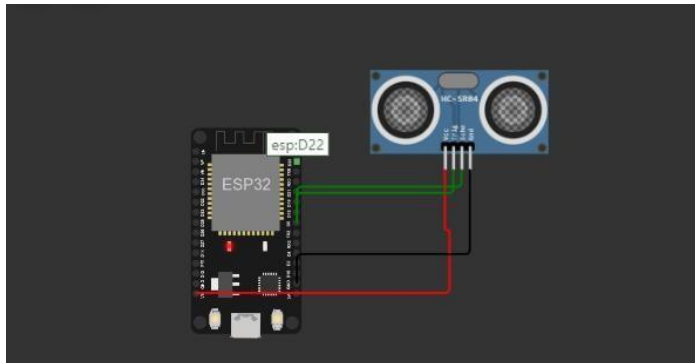
```

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(500);
}
}
initManagedDevice();
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){
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");
} else {
Serial.println("Publish FAILED");
}
}
}
}
```

**Diagram:**



### Wokwi Output:

The screenshot displays the Wokwi IDE environment. On the left, the 'sketch.ino' file is open, containing C++ code for an ESP32-based IoT project. The code includes headers for WiFi and MQTT, defines device information, and sets up an MQTT client to publish sensor data. On the right, the 'Simulation' window shows a virtual representation of the hardware: an ESP32 microcontroller board connected via I2C to an HC-SR04 ultrasonic sensor module. The simulation status bar at the bottom indicates the device is connected to WiFi with IP address 10.10.0.2 and is reconnecting the MQTT client.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3
4 WiFiClient wificlient;
5
6 #define ORG "b7rwwm"
7 #define DEVICE_TYPE "Kamal"
8 #define DEVICE_ID "kannan"
9 #define TOKEN "n*(@uoImJ-flu5M_gr"
10 #define speed 0.034
11
12
13 char server[] = ORG".messaging.internetofthings.ibmcloud.com";
14 char publishTopic[] = "iot-2/evt/event_1/fmt/json";
15 char topic[] = "iot-2/cmd/home/fmt/String";
16 char authMethod[] = "use-token-auth";
17 char token[] = TOKEN;
18 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
19 PubSubClient client(server, 1883, wificlient);
20 void publishData();
21 const int trigpin=5;
22 const int echopin=18;
23 String command;
24 String data="";
25 long duration;
26 float dist;
27 void setup()
28 {
29   Serial.begin(115200);
30   pinMode(trigpin, OUTPUT);
```

IBM Watson IoT Platform

Browse Action Device Types Interfaces

## Browse Devices

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched by criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device ID	Status	Device Type
kannan	Disconnected	Kamal

Items per page 50 | 1-1 of 1 item

Device Type: Kamal

Events 1 New event type

Event type name event\_1 Send

Schedule 20 Every Minute

Payload Specify the event payload in the editor window or by uploading a CSV file.

```
0 {
1   "randomNumber": random(0, 100)
2 }
3
```

Upload a CSV file

What functions can I apply?

### IBM cloud output:

IBM Watson IoT Platform

Identity Device Information Recent Events State Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
-------	-------	--------	---------------

Waiting for device events...

0 Simulations running