

Assignment -4
WOWKI SIMULATION

Project name	SmartFarmer - IoT Enabled Smart Farming Application
Student Name	JosephinflorenceY
Student Roll Number	950919106010
Maximum Marks	2 Marks

Question-1:

Write a code and make a connection in WOKWI for ultrasonic sensor. Whenever distance is less than 100 , send “alert” to IBM cloud and display in device recent events.

PROGRAM

```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient;
String data3;
#define ORG "b4hkg6"
#define DEVICE_TYPE "b11m3edevicetype" #define
DEVICE_ID "b1m3edeviceid"
#define TOKEN "Ao?yFfGVDA7dcv-KyQ"
#define speed 0.034 #define led 14 char server[] = ORG
".messaging.internetofthings.ibmcloud.com"; char publishTopic[] =
"iot-2/evt/Arya/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(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:

WOKWI SIMULATION

The screenshot shows the Wokwi web 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 "dqohie"
6 #define DEVICE_TYPE "PNT2022TMID34083"
7 #define DEVICE_ID "PNTID34083"
8 #define TOKEN "X5ZfHwFVJ5rwDmqS3v"
9 #define speed 0.034
10 #define led 14
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Aiswarya/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
19
20 const int trigpin=5;
21 const int echopin=18;
22 String command;
23 String data="";
24
25 long duration;
```

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. The simulation is running, as indicated by the 'Publish OK' and 'Sending payload: {"Alert Distance":93.96}' messages in the output window.

When distance<100:

This screenshot shows the same Wokwi simulation setup as the previous one. The ESP32 is connected to the HC-SR04 sensor. The output window shows the following messages:

```
Publish OK

Sending payload: {"Alert Distance":93.96}
Publish OK

Sending payload: {"Alert Distance":93.96}
Publish OK
```

The simulation is running, and the distance value is consistently 93.96.

When distance>100:

Simulation

02:27.394 83%

ESP32

HC-SR04

Publish OK

Sending payload: {"Distance":223.96}

Publish OK

Sending payload: {"Distance":223.96}

Publish OK

IBM CLOUD OUTPUT

IBM Watson IoT Platform

Browse

Action

Device Types

Interfaces

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
event_1	{"Temperature":71,"Humidity":59}	json	a few seconds ago
event_1	{"Temperature":18,"Humidity":86}	json	a few seconds ago
event_1	{"Temperature":22,"Humidity":21}	json	a few seconds ago
event_1	{"Temperature":35,"Humidity":85}	json	a few seconds ago
event_1	{"Temperature":48,"Humidity":72}		

1 Simulation running