

ASSIGNMENT – 4

Date	11 November 2022
Team ID	PNT2022TMID30897
Project Name	GAS LEAKAGE MONITORING AND ALERTING SYSTEM FOR INDUSTRIES

QUESTION:

Write code and connection in wokwi for ultrasonic sensors. That whenever distance is less than 100cms send “alert” to IBM Cloud and display in device recent events.

Upload document with wokwi share link and images.

Wokwi :

<https://wokwi.com/projects/348204126867292755>

CODE:

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

WiFiClient wifiClient;

#define ORG "w1p5bv"
#define DEVICE_TYPE "nsp5"
#define DEVICE_ID "nsp_1"
#define TOKEN "EWGwG15F6EKUtFh5W_"
#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);
    }
  }
}

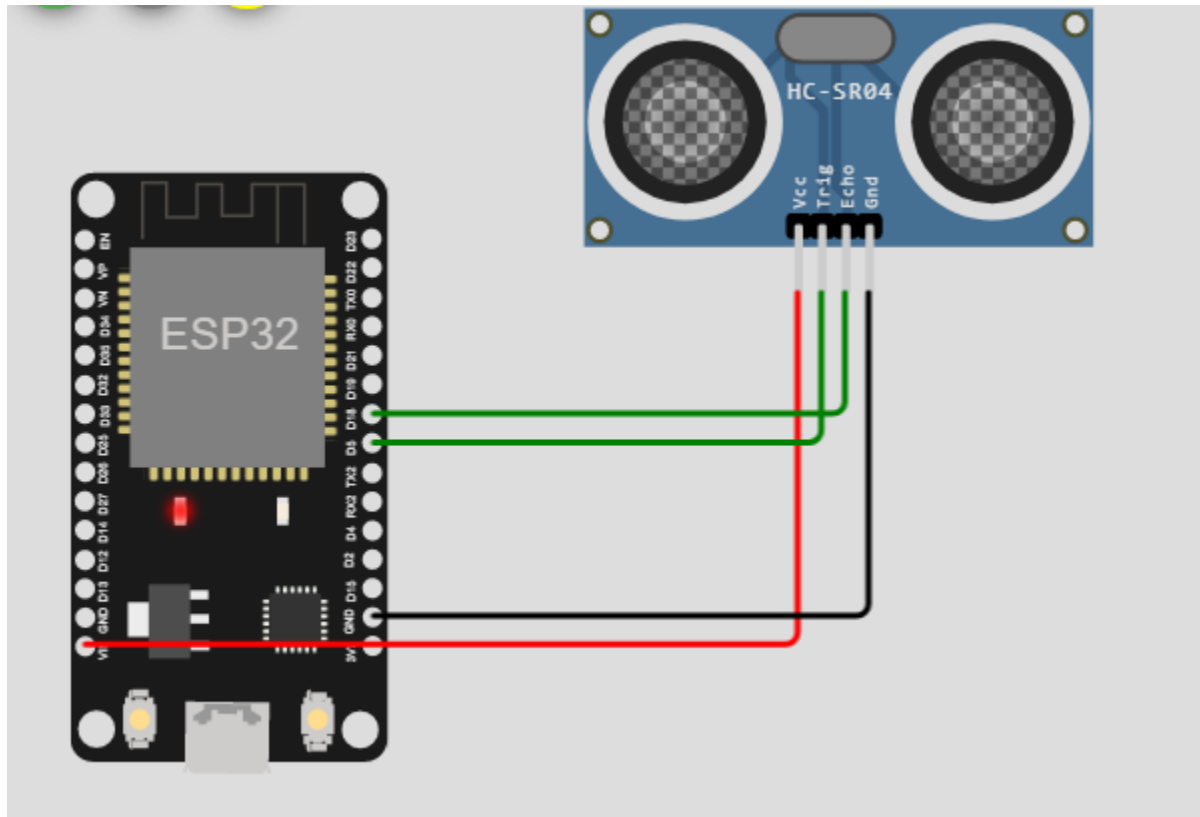
```

```

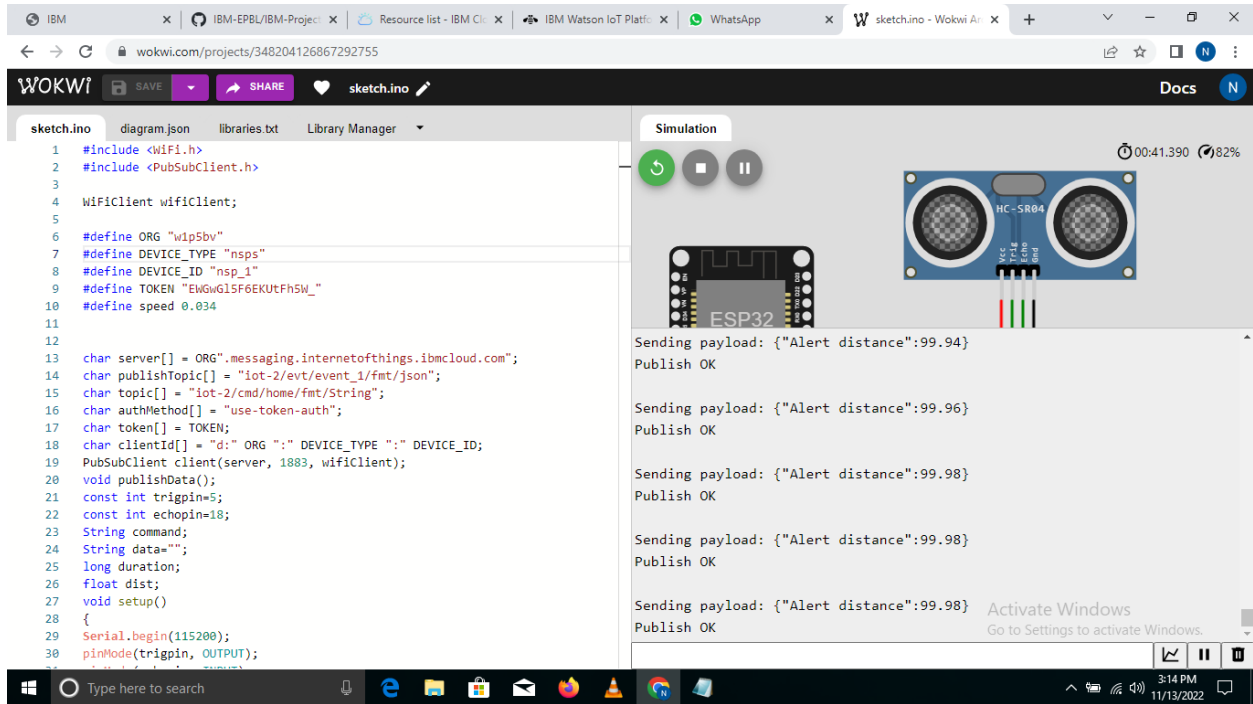
}
initManagedDevice();
Serial.println();
}
}
void initManagedDevice() {
if (client.subscribe(topic)) {
// Serial.println(client.subscribe(topic));
Serial.println("subscribe to cmd OK");
}
else {
Serial.println("subscribe to cmd FAILED");
}
}
void publishData()
{
digitalWrite(trigpin, LOW);
digitalWrite(trigpin, 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 interface with a C++ sketch for an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sketch configures the sensor and publishes distance data to an IBM Watson IoT Platform topic. The simulation window shows the sensor's output, and the serial monitor displays the published payloads.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3
4 WiFiClient wificlient;
5
6 #define ORG "wlp5bv"
7 #define DEVICE_TYPE "nsp5"
8 #define DEVICE_ID "nsp_1"
9 #define TOKEN "EWGwG15F6EKUTFh5W_"
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);
```

Simulation

00:41.390 82%

Sending payload: {"Alert distance":99.94}
Publish OK

Sending payload: {"Alert distance":99.96}
Publish OK

Sending payload: {"Alert distance":99.98}
Publish OK

Sending payload: {"Alert distance":99.98}
Publish OK

Sending payload: {"Alert distance":99.98}
Publish OK

Activate Windows
Go to Settings to activate Windows.

The screenshot displays a Windows 10 desktop environment. Two web browsers are open. The left browser, Google Chrome, shows the IBM Watson IoT Platform dashboard at the URL `w1p5bv.internetofthings.ibmcloud.com/dashboard/devices/browse`. The dashboard has a dark theme and a navigation bar with tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A 'Add Device' button is visible. Below the navigation bar, a message states: 'The recent events listed show the live stream of data that is coming and going from this device'. A table of recent events is displayed with the following data:

Event	Value	Format
event_1	{"Alert distance":99.99}	json
event_1	{"Alert distance":99.98}	json
event_1	{"Alert distance":99.98}	json
event_1	{"Alert distance":99.98}	json
event_1	{"Alert distance":99.98}	json

Below the table, a status message indicates '2 Simulations running'. The right browser, also Google Chrome, is open to the sketch.ino website. It shows a 'Simulation' window with a play button and a progress bar at 81%. The simulation log displays the following messages:

```

Sending payload: {"Alert distance":99.98}
Publish OK

Sending payload: {"Alert distance":99.98}
Publish OK

Sending payload: {"Alert distance":99.98}
Publish OK

Sending payload: {"Alert distance":99.98}
Publish OK

Sending payload: {"Alert distance":99.98}
Publish OK
  
```

The desktop background is a blue abstract image. The taskbar at the bottom contains icons for 'This PC', 'Microsoft Edge', 'Pre Development', 'ELAKKIYA', 'Emulated Turbo C...', 'Microsoft 365', and several application icons. The system clock in the bottom right corner shows '3:18 PM' and '11/13/2022'.

The screenshot shows the IBM Watson IoT Platform interface. The top navigation bar includes the IBM logo and the text 'IBM Watson IoT Platform'. Below this, a dark blue header contains the user's email 'sriramnathiya@gmail.com' and the device ID 'w1p5bv'. The main content area is titled 'w1p5bv.internetofthings.ibmcloud.com/dashboard/devices/browse'. The left sidebar contains icons for various functions. The top navigation bar has tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. The 'Add Device' button is visible in the top right. The 'Recent Events' tab is selected, showing a table of events. The table has columns for 'Event', 'Value', 'Format', and 'Last Received'. All events are labeled 'event_1' and have a value of '{\"Alert distance\":99.98}' in 'json' format, received 'a few seconds ago'. A notification at the bottom right states '2 Simulations running'.

Event	Value	Format	Last Received
event_1	{\"Alert distance\":99.99}	json	a few seconds ago
event_1	{\"Alert distance\":99.98}	json	a few seconds ago
event_1	{\"Alert distance\":99.98}	json	a few seconds ago
event_1	{\"Alert distance\":99.98}	json	a few seconds ago
event_1	{\"Alert distance\":99.98}	json	a few seconds ago

