

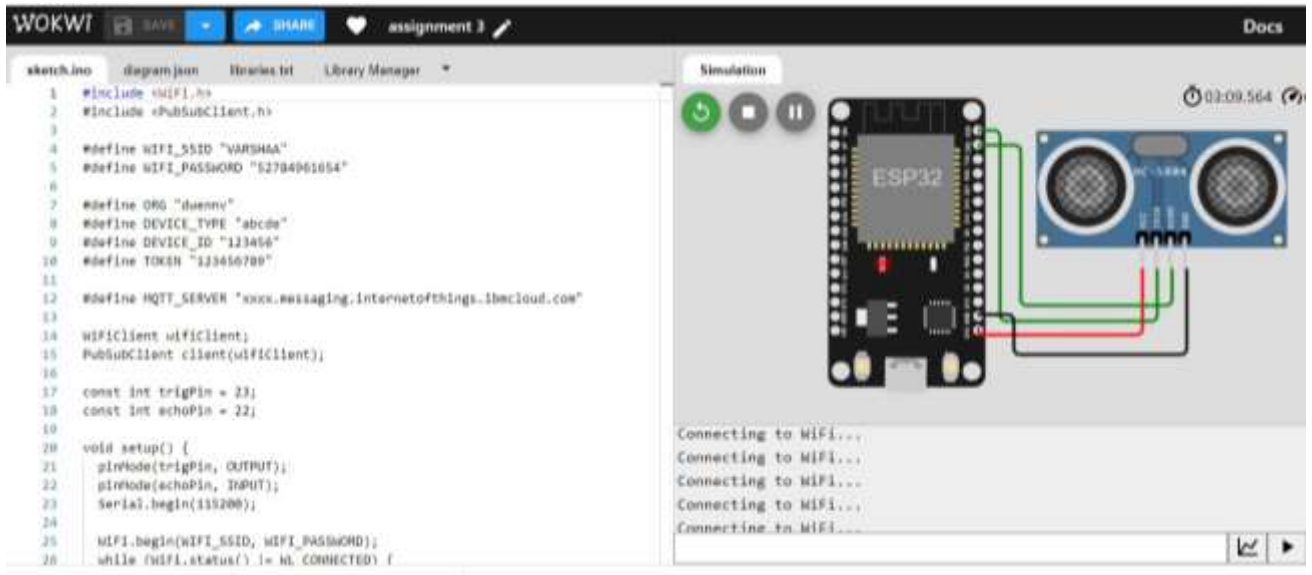
ASSIGNMENT 3

Build a wokwi product, use ultrasonic sensor and detect the Distance from the object.

Whenever distance is less than 100cms upload the value to the ibm cloud.in recent device events upload the data from wokwi

Vijayalakshmi B Group 5

CONNECTIONS :



WOKWI LINK :

<https://wokwi.com/projects/new/esp32>

PROGRAM

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

#define WIFI_SSID "VIJAYALAKSHMI"
#define WIFI_PASSWORD "52784961654"

#define ORG "dwennv"
#define DEVICE_TYPE "abcde"
#define DEVICE_ID "123456"
#define TOKEN "123456789"

#define MQTT_SERVER "xxxx.messaging.internetofthings.ibmcloud.com"

WiFiClient wifiClient;
PubSubClient client(wifiClient);

const int trigPin = 23;
const int echoPin = 22;

void setup() {
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    Serial.begin(115200);

    WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.println("Connecting to WiFi...");
    }

    client.setServer(MQTT_SERVER, 1883);
    client.setCallback(callback); }
```

```

void loop() { long duration,
    distance; digitalWrite(trigPin,
    LOW); delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);
    distance = duration * 0.034 / 2;
    Serial.print("Distance: ");
    Serial.print(distance);
    Serial.println(" cm");
    if(distance < 100) {
        publishMessage(distance);
    }
    client.loop();
    delay(500);
}

void callback(char* topic, byte* payload, unsigned int length) {
    // Handle callback function here
}

void publishMessage(long distance) {
    String payload = "{\"distance\": " + String(distance) + "}";
    char topic[100];
    sprintf(topic, "iot-2/evt/status/fmt/json");
    client.connect(DEVICE_ID, TOKEN, "");
    client.publish(topic, (char*) payload.c_str());
    client.disconnect();
}

```

IBM CLOUD

The screenshot shows the 'Browse Devices' interface in the IBM Watson IoT Platform. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various functions. The main heading is 'Browse Devices', followed by two tabs: 'All Devices' (selected) and 'Diagnose'. Below the tabs, a paragraph explains that the table shows a summary of all devices added, which can be filtered, organized, and searched. A search bar labeled 'Search by Device ID' is present. To the right, there's a link for 'Device Simulator'. The table below has columns for 'Device ID', 'Status', 'Device Type', 'Class ID', and 'Date Added'. One device is listed: 'Team_4' with status 'Disconnected' and type 'Gas_Pipeline_Monitoring'. At the bottom, it indicates 'Items per page: 50' and '1 of 1 item'.

Browse Devices

[All Devices](#) [Diagnose](#)

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

Search by Device ID Device Simulator

Device ID	Status	Device Type	Class ID	Date Added
> Team_4	Disconnected	Gas_Pipeline_Monitoring	Device	19 May 2023 16:59

Items per page: 50 | 1 - 1 of 1 item 1 of 1 page