

Assignment - 4

Wowki & IBM Cloud

Assignment Date	31 October 2022
Student Name	Mohamed Shaik Wasim M
Student Roll Number	921319106127
Maximum Marks	2 Marks

Question-1:

Write code and connections in wowki for the ultrasonic sensor. Whenever the distance is less than 100cms sent "alert" to IBM cloud and display in device recent events.

Code:

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

WiFiClient wifiClient;

#define ORG "oa3490"
#define DEVICE_TYPE "TestDeviceType"
#define DEVICE_ID "12345"
#define TOKEN "-A)0raS44f)fdjYBVS"
#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/abcd_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="";
String lat="14.167589";
String lon="80.248510";
String name="point2";
String icon="";

long duration;
int 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(1000)
;
}
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){
    dist=100- dist; icon="fa-
trash";
}else{ dist=0;
    icon="fa-trash-
o";
}

DynamicJsonDocument doc(1024) ;
String payload; doc["Name"]=
name; doc["Latitude"]= lat;
doc["Longitude"]= lon;
doc["Icon"]= icon;
doc["FillPercent"]= dist;
serializeJson(doc, payload);
delay(3000) ;
Serial.print("\n") ;

Serial.print("Sending payload: ") ; Serial.println(
payload);

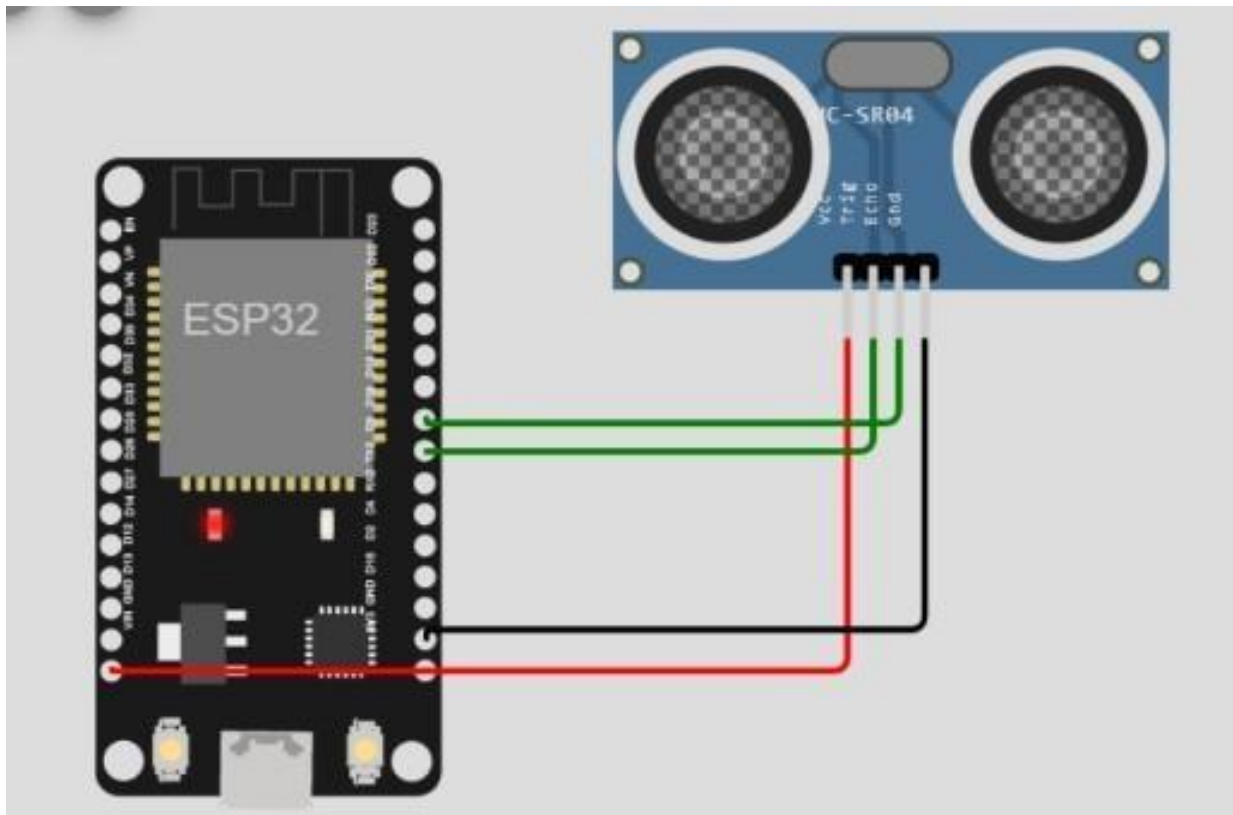
```

```

if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish OK") ;
} else {
    Serial.println("Publish FAILED") ;
}
}

```

Connections:



Output:

The screenshot shows the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, displaying an Arduino sketch. The sketch includes libraries for WiFi, PubSubClient, and ArduinoJson. It defines constants for the server, topic, token, and device ID. The main loop publishes data to the server every 5 seconds. The data payload is a JSON object containing the device name, latitude, longitude, icon, and fill percent. The simulation on the right shows an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sensor's distance is displayed as 94cm. The console output shows the following:

```
trash", "FillPercent": 6}
Publish OK

Sending payload:
{"Name": "point2", "Latitude": "14.167589", "Longitude": "80.248510", "Icon": "fa-trash", "FillPercent": 6}
Publish OK
```

Output :(IBM Cloud)

The screenshot shows the IBM Cloud IoT Platform dashboard. The device with ID 12345 is listed as 'Disconnected'. The 'Recent Events' tab is selected, showing a list of events. The events are as follows:

Event	Value	Format	Last Received
event_1	{\"Alert Distance\":8}	json	a few seconds ago
event_1	{\"Alert Distance\":81}	json	a few seconds ago
event_1	{\"Alert Distance\":56}	json	a few seconds ago
event_1	{\"Alert Distance\":98}	json	a few seconds ago
event_1	{\"Alert Distance\":72}	json	a few seconds ago

The dashboard also shows a status bar at the bottom indicating '1 Simulation running'.