

Assignment - 4

Assignment Date	17 October 2022
Student Name	Harsihraaj S
Student Roll Number	737819CSR057
Maximum Marks	2 Marks

Question-1:

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device recent events. Upload document with wokwi share link and images of IBM cloud

CODE 1 :

```
#include <WiFi.h> #include
<PubSubClient.h>
void callback(char* subscribtopic, byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "cbseji"//IBM ORGANITION ID
#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json"; char
subscribtopic[] = "iot-2/cmd/test/fmt/String"; char authMethod[]
= "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
const int trigPin = 5; const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration; float distance;
void          setup()          {
Serial.begin(115200);
pinMode(trigPin,  OUTPUT);
pinMode(echoPin, INPUT);
wificonnect(); mqttconnect();
}
void loop()
{ digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW); duration =
pulseIn(echoPin, HIGH); distance =
duration * SOUND_SPEED/2;
Serial.print("Distance (cm): ");
Serial.println(distance); if(distance<100)
{
```

```

Serial.println("ALERT!!");
delay(1000);
PublishData(distance);
delay(1000); if
(!client.loop()) {
mqttconnect();
} }
delay(1000);
}
void PublishData(float dist) { mqttconnect();
String payload = "{\"Distance\":\""; payload
+= dist;
payload += "\",\"ALERT!!\":\"\"Distance less than 100cms\"";
payload += "\"}";
Serial.print("Sending payload: ");
Serial.println(payload);

if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Publish ok");
} else {
Serial.println("Publish failed");
} } void mqttconnect() {
if (!client.connected()) {
Serial.print("Reconnecting client to ");
Serial.println(server);
while (!client.connect(clientId, authMethod, token)) {
Serial.print(".");
delay(500); }
initManagedDevice();
Serial.println();
} }
void wificonnect() {
Serial.println();
Serial.print("Connecting          to          ");
WiFi.begin("Wokwi-GUEST", "", 6); while
(WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}
void initManagedDevice() { if
(client.subscribe(subscribetopic)) {
Serial.println((subscribetopic));
Serial.println("subscribe to cmd
OK");
} else {
Serial.println("subscribe to cmd FAILED");
} }
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{

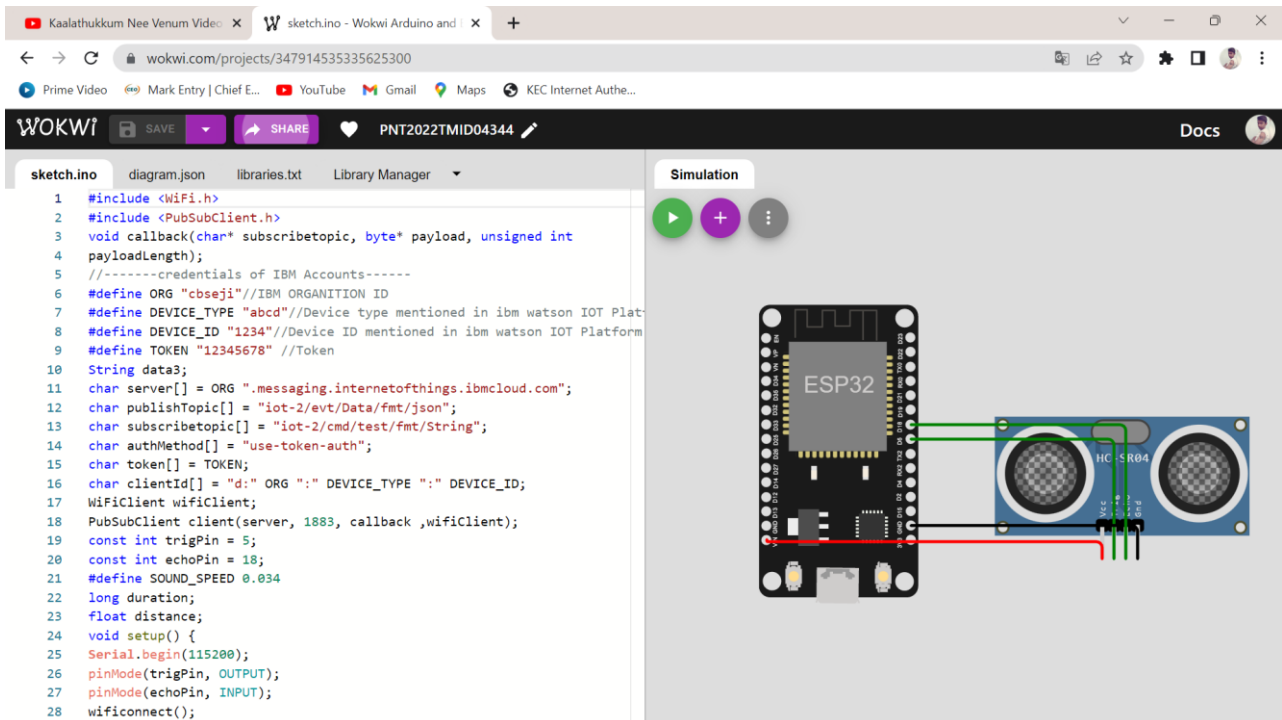
```

```
Serial.print("callback invoked for topic: ");  
Serial.println(subscribetopic); for (int i = 0;  
i < payloadLength; i++) {  
//Serial.print((char)payload[i]); data3 +=  
(char)payload[i];  
}  
Serial.println("data: "+ data3); data3="";  
}
```

Wokwi Link :

<https://wokwi.com/projects/347914535335625300>

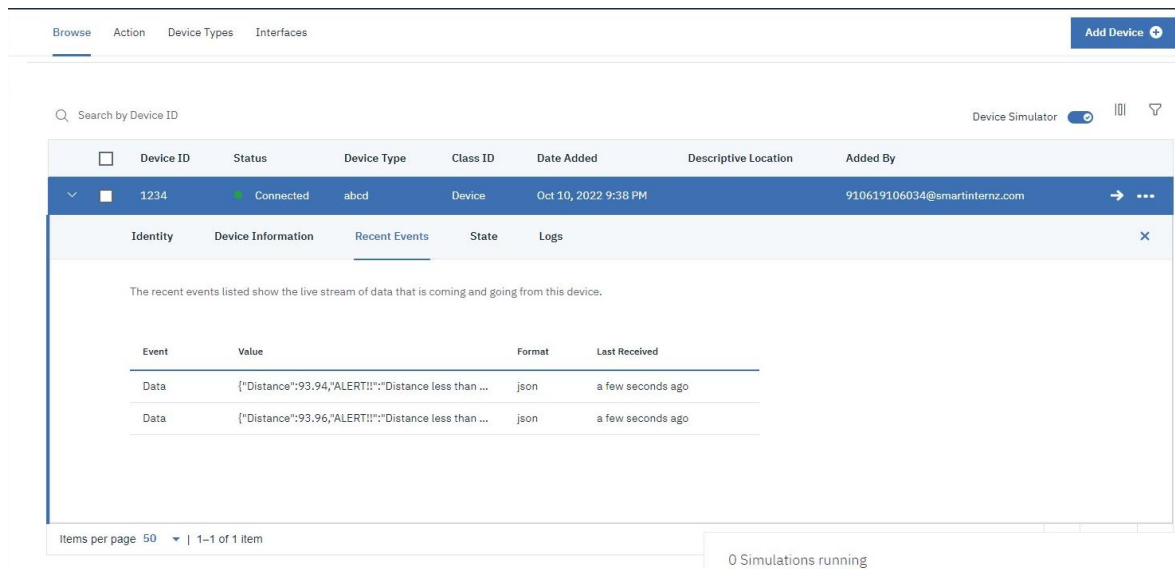
Output and Simulation :



The screenshot displays the Wokwi web interface. On the left, the 'sketch.ino' file is open, showing an Arduino sketch that includes the `WiFi.h` and `PubSubClient.h` libraries. The sketch defines constants for an IBM Watson IoT device, including the organization ID, device type, device ID, and token. It sets up a WiFi client and a PubSubClient, and defines pins for a trigPin (5) and an echoPin (18). The `setup()` function initializes the serial port and pin modes. The `loop()` function (partially visible) would handle the distance measurement and data transmission.

On the right, the 'Simulation' tab is active, showing a 3D model of an ESP32 microcontroller board connected to an HC-SR04 ultrasonic sensor module. The sensor is connected to the ESP32 via a breadboard with jumper wires.

Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device recent events.



The screenshot shows the Wokwi device simulator interface. At the top, there are tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A search bar is present, and a 'Device Simulator' toggle is set to 'On'. Below the tabs, a table lists the devices. The first device has ID 1234, is 'Connected', and is of type 'abcd'. The 'Recent Events' tab is selected, showing a table of events.

Event	Value	Format	Last Received
Data	{"Distance":93.94,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":93.96,"ALERT!!":"Distance less than ...	json	a few seconds ago

At the bottom, it indicates 'Items per page 50' and '1-1 of 1 item'. A status bar at the very bottom shows '0 Simulations running'.