

ASSIGNMENT - 4

DATE	17 October 2022
TEAM ID	PNT2022TMID07000
NAME	DHARSHINI B
STUDENT ROLL NUMBER	GCTC1918109
MAXIMUM MARKS	2 Marks

QUESTION :

Write code and connections in wokwi for ultrasonic sensor.

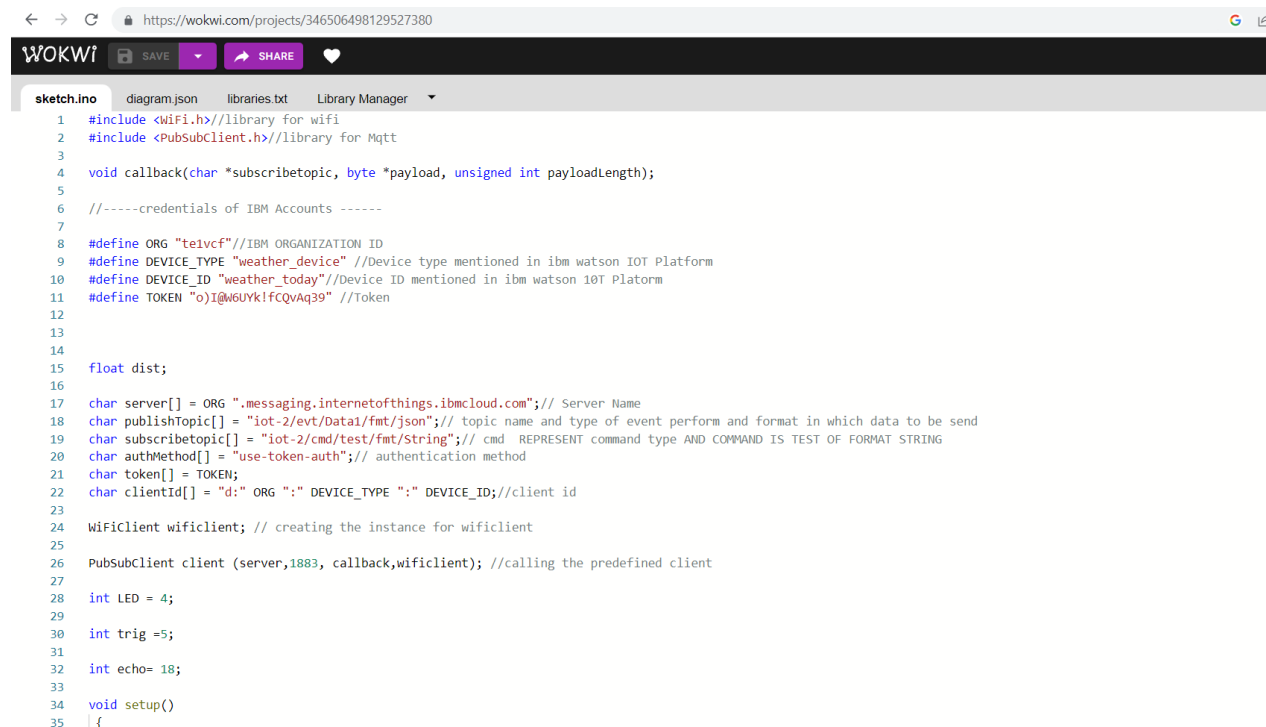
Whenever distance is less than 100 cms send "alert" to ibm cloud and display in device recent events.

Upload document with wokwi share link and images of ibm cloud

WOKWI CODE AND IMPLEMENTATION LINK:

<https://wokwi.com/projects/346506498129527380>

CODE:



```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for Mqtt
3
4 void callback(char *topic, byte *payload, unsigned int payloadLength);
5
6 //-----credentials of IBM Accounts -----
7
8 #define ORG "te1vcf" //IBM ORGANIZATION ID
9 #define DEVICE_TYPE "weather_device" //Device type mentioned in ibm watson IOT Platform
10 #define DEVICE_ID "weather_today" //Device ID mentioned in ibm watson IOT Platform
11 #define TOKEN "oI@W6UYk!fCQvAq39" //Token
12
13
14
15 float dist;
16
17 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
18 char publishTopic[] = "iot-2/evt/Data1/fmt/json"; // topic name and type of event perform and format in which data to be send
19 char subscribetopic[] = "iot-2/cmd/test/fmt/String"; // cmd REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING
20 char authMethod[] = "use-token-auth"; // authentication method
21 char token[] = TOKEN;
22 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
23
24 WiFiClient wificlient; // creating the instance for wificlient
25
26 PubSubClient client (server,1883, callback,wificlient); //calling the predefined client
27
28 int LED = 4;
29
30 int trig =5;
31
32 int echo= 18;
33
34 void setup()
35 {
```

```
34 void setup()
35 {
36
37   Serial.begin(115200);
38   pinMode(trig, OUTPUT);
39   pinMode(echo, INPUT);
40   pinMode(LED, OUTPUT);
41   delay(10);
42
43   wificonnect();
44
45   mqttconnect();
46
47 }
48
49 void loop()// Recursive Function
50 {
51
52   delayMicroseconds(10);
53   digitalWrite(trig, LOW);
54   digitalWrite(trig, LOW);
55   digitalWrite(trig,HIGH);
56   float dur= pulseIn(echo,HIGH);
57   float dist = (dur* 0.0343)/2;
58   Serial.print ("Distance in cm : ");
59   Serial.println(dist);
60
61   PublishData(dist);
62
63   delay(1000);
64
65   if (!client.loop()) {
66
67     mqttconnect();
68   }
```

```
67   mqttconnect();
68 }
69 }
70
71 void PublishData(float dist) {
72   mqttconnect();
73
74   String object;
75
76   if (dist<100)
77   {
78     digitalWrite(LED, HIGH);
79     Serial.println("object is near");
80     object = "ALERT! object is near";
81   }
82
83   else
84   {
85     digitalWrite(LED,LOW);
86     Serial.println("no object found");
87     object ="No object found";
88   }
89
90   String payload="{\"distance\":";
91   payload += dist;
92   payload += ", \"object\":\";";
93   payload += object;
94   payload += "\";";
95
96   Serial.print("Sending payload: ");
97   Serial.println(payload);
98
99   if (client.publish(publishTopic, (char*) payload.c_str()))
100   {
101     Serial.println("Publish ok"); // if it sucessfully upload
102   }
```

```
103   else {
104     Serial.println("Publish failed");
105   }
106 }
107
108 void mqttconnect() {
109   if (!client.connected()) {
110     Serial.print("Reconnecting client to ");
111     Serial.println(server);
112     while (!client.connect(clientId, authMethod, token)) {
113       Serial.print(".");
114       delay(500);
115     }
116     initManagedDevice();
117     Serial.println();
118   }
119 }
120
121
122
123 void wificonnect() //function definition for wificonnect
124 {
125   Serial.println();
126   Serial.print("Connecting to ");
127
128   WiFi.begin("Wokwi-GUEST", "", 6); //passing the wifi credentials to establish the connection
129   while (WiFi.status() != WL_CONNECTED) {
130     delay(500);
131     Serial.print(".");
132   }
133   Serial.println("");
134   Serial.println("WiFi connected");
135   Serial.println("IP address: ");
136   Serial.println(WiFi.localIP());
137 }
```

```
138
139 void initManagedDevice() {
140
141   if (client.subscribe(subscribetopic)) {
142     Serial.println(subscribetopic);
143     Serial.println("subscribe to cmd OK");
144   }
145   else {
146     Serial.println("subscribe to cmd FAILED");
147   }
148 }
149
150 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
151 {
152   Serial.print("callback invoked for topic: ");
153   Serial.println(subscribetopic);
154   for (int i = 0; i < payloadLength; i++) {
155     //Serial.print((char)payload[i]);
156     // data3 += (char)payload[i];
157   }
158
159   // Serial.println("data: " + data3);
160   //if(data3=="lighton")
161   {
162     //Serial.println(data3);
163     digitalWrite(LED,HIGH);
164
165   }
166
167   //else
168   {
169     //Serial.println(data3);
170     digitalWrite(LED,LOW);
171   }
172 }
```

OUTPUT:

When the distance is less than 100 cms, send an “alert” message to IBM Watson IoT Platform.

The screenshot shows the Wokwi IDE interface. On the left, the sketch code is displayed, which includes the following key sections:

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for Mqtt
3
4 void callback(char *topic, byte *payload, unsigned int payloadLength);
5
6 //-----credentials of IBM Accounts -----
7
8 #define ORG "te1vcf" //IBM ORGANIZATION ID
9 #define DEVICE_TYPE "weather_device" //Device type mentioned in ibm watson IOT Platform
10 #define DEVICE_ID "weather_today" //Device ID mentioned in ibm watson IOT Platform
11 #define TOKEN "oI@6UYk1fcQvAq39" //Token
12
13
14
15 float dist;
16
17 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
18 char publishTopic[] = "iot-2/evt/Data1/fmt/json"; // topic name and type of event perform
19 char subscribTopic[] = "iot-2/cmd/test/fmt/string"; // cmd REPRESENT command type AND CO
20 char authMethod[] = "use-token-auth"; // authentication method
21 char token[] = TOKEN;
22 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
23
24 WiFiClient wificlient; // creating the instance for wificlient
25
26 PubSubClient client (server,1883, callback,wificlient); //calling the predefined client
27
28 int LED = 4;
29
30 int trig =5;
31
32 int echo= 18;
33
34 void setup()
35 {
```

On the right, the simulation window shows an ESP32 board connected to an Ultrasonic Distance Sensor. The sensor's distance is set to 42cm. Below the simulation, the console output shows the following messages:

```
object is near
Sending payload: {"distance":42.34,"object":"ALERT! object is near"}
Publish ok
Distance in cm : 42.33
object is near
Sending payload: {"distance":42.33,"object":"ALERT! object is near"}
Publish ok
```

IBM CLOUD IMAGE

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes the following tabs: Identity, Device Information, Recent Events, State, and Logs. The 'Recent Events' tab is selected, displaying a table of recent events.

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data1	{"distance":42.34,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":42.34,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":42.33,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":42.33,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":42.34,"object":"ALERT! object is near"}	json	a few seconds ago

When the object is far(greater than 100 cms) , send “ no object found” to the IBM Watson IOT Platform.

The screenshot shows the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, displaying an Arduino sketch that configures an ESP32 to connect to the IBM Watson IoT Platform and send sensor data. The sketch includes comments for credentials and defines the device type as 'weather_device'. On the right, a simulation window shows an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sensor's distance is currently set to 141cm. Below the simulation, a console window shows the output of the program, including the message 'no object found' and the JSON payload being sent to the IoT platform: `{"distance":142.19,"object":"No object found"}`.

IBM CLOUD IMAGE

The screenshot displays the IBM Watson IoT Platform dashboard. At the top, a table lists the device 'weather_today', which is 'Connected'. Below this, the 'Recent Events' tab is selected, showing a stream of data events. The events are listed in a table with columns for Event, Value, Format, and Last Received.

Event	Value	Format	Last Received
Data1	<code>{"distance":142.19,"object":"No object found"}</code>	json	a few seconds ago
Data1	<code>{"distance":142.19,"object":"No object found"}</code>	json	a few seconds ago
Data1	<code>{"distance":142.22,"object":"No object found"}</code>	json	a few seconds ago