

ASSIGNMENT - 4

DATE	17 OCTOBER 2022
TEAM ID	PNT2022TMID06972
NAME	SHOWMINI R
STUDENT ROLL NUMBER	GCTC1918L16
MAXIMUM MARKS	2 MARKS

QUESTION:

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

WOKWI CODE AND IMPLEMENTATION LINK:

<https://wokwi.com/projects/346688722332287572>

CODE:

```
esp32-dht22.ino  diagram.json  libraries.txt  Library Manager
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 "pet212" //IBM ORGANIZATION ID
9  #define DEVICE_TYPE "Fire_Device" //Device type mentioned in ibm watson IoT Platform
10 #define DEVICE_ID "Fire_123" //Device ID mentioned in ibm watson IoT Platform
11 #define TOKEN "kidHlae9gcclgvijs" //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
```

```
esp32-dht22.ino  diagram.json  libraries.txt  Library Manager
28 int LED = 4;
29
30 int trig = 5;
31
32 int echo = 18;
33
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   wificlient();
44   mqttconnect();
45
46 }
47
48 void loop() // Recursive Function
49 {
50
51   delayMicroseconds(10);
52   digitalWrite(trig, LOW);
53   digitalWrite(trig, LOW);
54   digitalWrite(trig, HIGH);
55 }
```

W esp32-dht22.ino copy - Wokwi

wokwi.com/projects/346687281544823380

Gmail YouTube Maps Implementing HT...

WOKWI SAVE SHARE Docs

esp32-dht22.ino diagram.json libraries.txt Library Manager

```
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   mqttconnect();
67 }
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 }
```

Type here to search

27°C Mostly cloudy

ENG 21:31 27-10-2022

W esp32-dht22.ino copy - Wokwi

wokwi.com/projects/346687281544823380

Gmail YouTube Maps Implementing HT...

WOKWI SAVE SHARE Docs

esp32-dht22.ino diagram.json libraries.txt Library Manager

```
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   }
```

Type here to search

27°C Mostly cloudy

ENG 21:31 27-10-2022

W esp32-dht22.ino copy - Wokwi X +

wokwi.com/projects/346687281544823380

Gmail YouTube Maps Implementing HTTP...

WOKWI SAVE SHARE Docs

esp32-dht22.ino diagram.json libraries.txt Library Manager

```
111 Serial.println(server);
112 while (!client.connect(clientId, authMethod, token)) {
113   Serial.print(".");
114   delay(500);
115 }
116
117 initManagedDevice();
118 Serial.println();
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() {
```

Type here to search

27°C Mostly cloudy ENG 21:32 27-10-2022

W esp32-dht22.ino copy - Wokwi X +

wokwi.com/projects/346687281544823380

Gmail YouTube Maps Implementing HTTP...

WOKWI SAVE SHARE Docs

esp32-dht22.ino diagram.json libraries.txt Library Manager

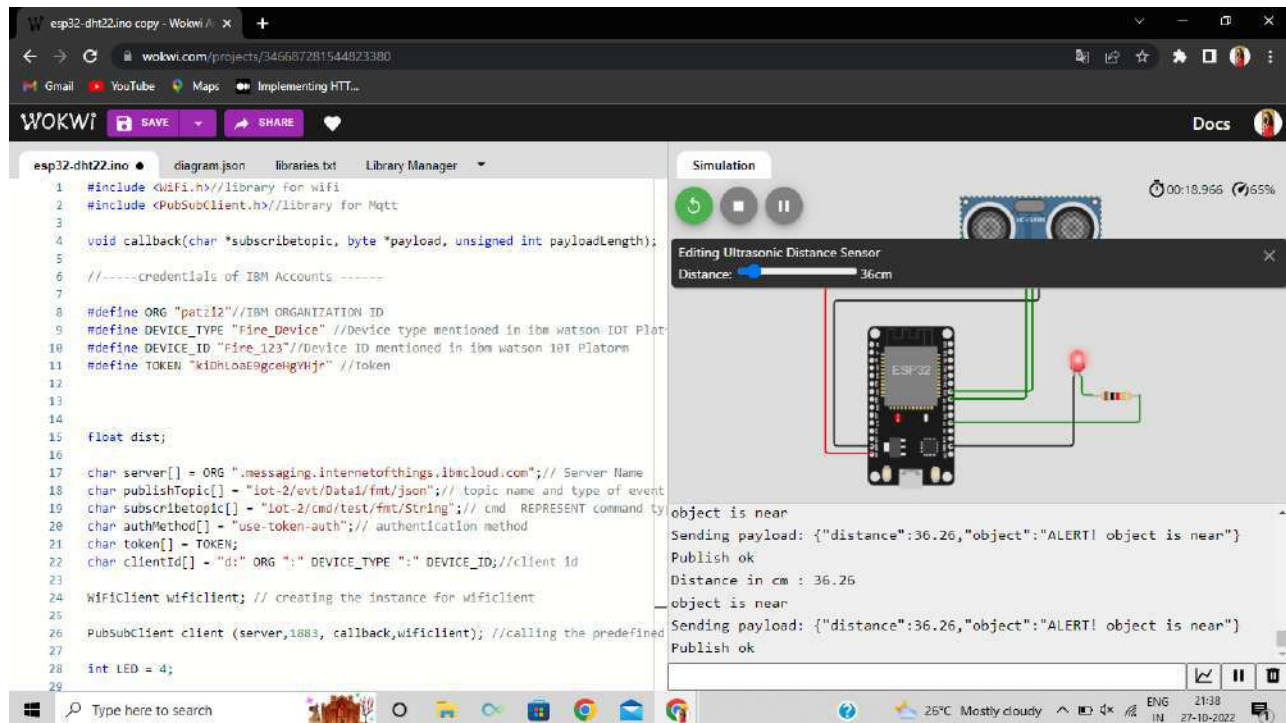
```
139 if (client.subscribe(subscribetopic)) {
140   Serial.println((subscribetopic));
141   Serial.println("subscribe to cmd OK");
142 }
143 else {
144   Serial.println("subscribe to cmd FAILED");
145 }
146 }
147 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
148 {
149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

Type here to search

26°C Mostly cloudy ENG 21:34 27-10-2022

OUTPUT:

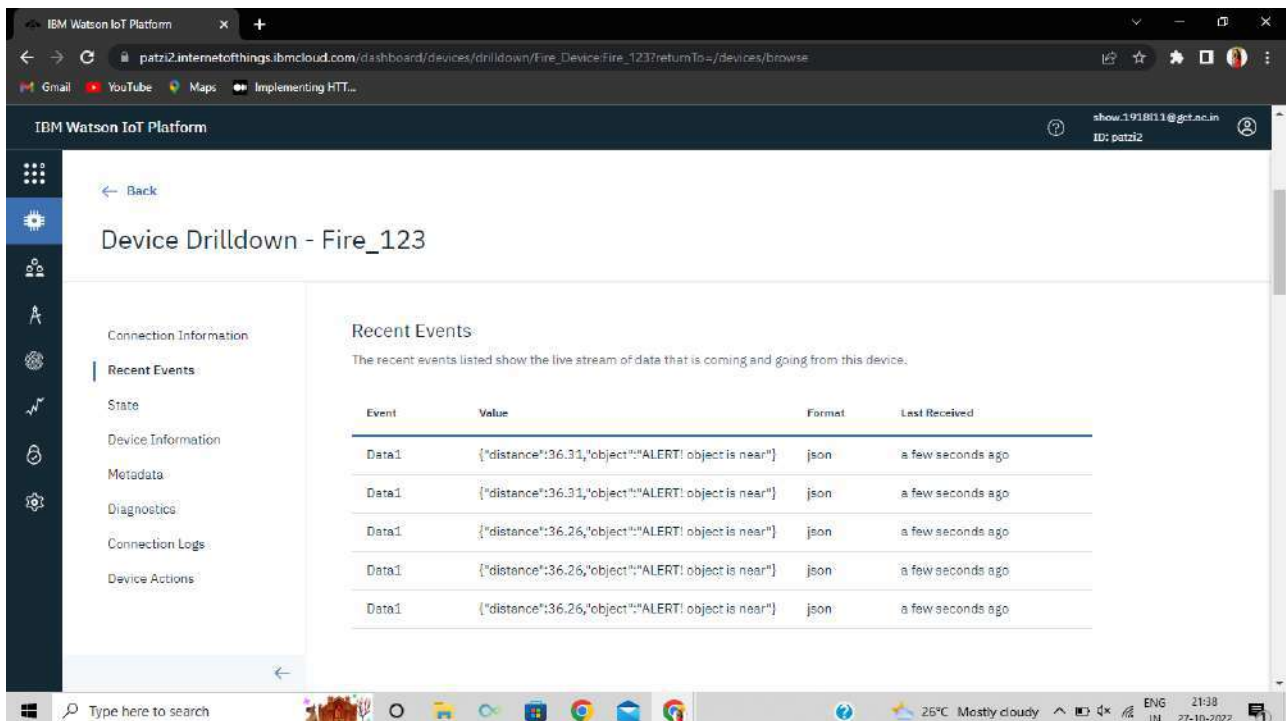
When the distance is less than 100 cms, send an “alert” message to IBM Watson IOT Platform.



The screenshot displays the Wokwi IDE interface. On the left, the Arduino code for an ESP32 is shown, which includes libraries for WiFi and MQTT, defines IBM IoT credentials, and sets up a callback function to send an alert message when the distance is less than 100 cm. On the right, a simulation of the ESP32 board is shown with an ultrasonic sensor. A dialog box indicates the distance is 36 cm. Below the simulation, the console output shows the alert message being sent: {"distance":36.26,"object":"ALERT! object is near"}.

```
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 "pat212" //IBM ORGANIZATION ID
9 #define DEVICE_TYPE "Fire_Device" //Device type mentioned in ibm watson IoT Plat
10 #define DEVICE_ID "Fire_123" //Device ID mentioned in ibm watson IoT Platform
11 #define TOKEN "kidH0aE9gcagvUjR" //token
12
13
14 float dist;
15
16 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
17 char publishTopic[] = "iot-2/evt/Data1/fmt/json"; // topic name and type of event
18 char subscribTopic[] = "iot-2/cmd/test/fmt/String"; // cmd REPRESENT command ty
19 char authMethod[] = "use-token-auth"; // authentication method
20 char token[] = TOKEN;
21 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
22
23 WiFiClient wificlient; // creating the instance for wificlient
24 PubSubClient client (server,1883, callback,wificlient); //calling the predefined
25
26 int LED = 4;
```

IBM CLOUD IMAGE



The screenshot shows the IBM Watson IoT Platform dashboard. The main heading is "Device Drilldown - Fire_123". The dashboard is divided into two main sections: "Connection Information" and "Recent Events". The "Recent Events" section displays a table of recent events, showing the event name, value, format, and last received time.

Event	Value	Format	Last Received
Data1	{"distance":36.31,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":36.31,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":36.26,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":36.26,"object":"ALERT! object is near"}	json	a few seconds ago
Data1	{"distance":36.26,"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 code for `esp32-dht22.ino` is displayed. It includes libraries for WiFi and PubSubClient, and defines constants for an IBM Watson IoT device. The code sets up a WiFi client and a PubSubClient, and includes a callback function for receiving sensor data. On the right, a simulation window titled "Editing Ultrasonic Distance Sensor" shows a virtual sensor connected to an ESP32 board. The distance is set to 217cm. Below the simulation, a console window shows the output of the program, indicating that the sensor has detected "no object found" and the corresponding JSON payload is being sent to the IBM Watson IoT platform.

```
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 "patz12" //IBM ORGANIZATION ID
9 #define DEVICE_TYPE "Fire_Device" //Device type mentioned in ibm watson IoT
10 #define DEVICE_ID "Fire_123" //Device ID mentioned in ibm watson IoT Platform
11 #define TOKEN "kidiLoe9gc0hgVhja" //token
12
13
14 float dist;
15
16 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
17 char publishTopic[] = "iot-2/evt/Data1/fmt/json"; // topic name and type of
18 char subscribeTopic[] = "iot-2/cmd/test/fmt/String"; // cmd REPRESENT command
19 char authMethod[] = "use-token-auth"; // authentication method
20 char token[] = TOKEN;
21 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
22
23 WiFiClient wifiClient; // creating the instance for wifiClient
24 PubSubClient client(server, 1883, callback, wifiClient); //calling the predefine
25
26 int LED = 4;
27
28
29
30
```

no object found
Sending payload: {"distance":218.85,"object":"No object found"}
Publish ok
Distance in cm : 218.85
no object found
Sending payload: {"distance":218.85,"object":"No object found"}
Publish ok

IBM CLOUD IMAGE

The screenshot shows the IBM Watson IoT Platform dashboard. The main heading is "Device Drilldown - Fire_123". On the left, there is a sidebar with navigation options: Connection Information, Recent Events, State, Device Information, Metadata, Diagnostics, Connection Logs, and Device Actions. The "Recent Events" section is selected, showing a list of events. The events are all of type "Data1" and contain the same JSON payload: {"distance":218.85,"object":"No object found"}. The events are listed in a table with columns for Event, Value, Format, and Last Received.

Event	Value	Format	Last Received
Data1	{"distance":218.85,"object":"No object found"}	json	a few seconds ago
Data1	{"distance":218.85,"object":"No object found"}	json	a few seconds ago
Data1	{"distance":218.85,"object":"No object found"}	json	a few seconds ago
Data1	{"distance":218.85,"object":"No object found"}	json	a few seconds ago
Data1	{"distance":218.85,"object":"No object found"}	json	a few seconds ago