

Question1 :

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.

WOKWI LINK :

<https://wokwi.com/projects/305566932847821378>

CODE :

```
1 //===== setup() ===== for mqtt
2 #include <PubSubClient.h> //library for mqtt
3
4
5 void callback(char* topic, byte* payload, unsigned int payloadLength);
6
7 // ===== constants of IBM Watson =====
8
9 #define IBM_HOST "ibm" //IBM IP address
10 #define DEVICE_TYPE "ultrasonic" //Device type mentioned in the Watson IoT Platform
11 #define DEVICE_ID "DISTANCE TEST" //Device ID mentioned in the Watson IoT Platform
12 #define TOKEN "a000a7942ba9585a" //token
13 String data;
14 float dist;
15
16
17 // ===== Options of the client =====
18 char server[] = IBM_HOST; //messaging.internetofthings.ibmcloud.com // Server Name
19 char publishTopic[] = "iot-2/rest/data/v1/devices" // topic name and type of event (prefix and format to which data to be sent)
20 char subscribeTopic[] = "iot-2/monitoring/data/v1" // can subscribe command type and response (a text or control action)
21 char authMethod[] = "api-token-auth" // authentication method
22 char token[] = TOKEN;
23 char clientId[] = "i:" IBM_HOST "/" DEVICE_TYPE "/" DEVICE_ID; // client ID
24
25
26 // =====
27 #if defined(WIFI) // creating the instance for wifi client
28 #include <WiFiClient.h> // creating the instance for wifi client
29 #endif
30 #if defined(ESP8266) // creating the instance for esp8266 client
31 #include <ESP8266WiFiClient.h> // creating the instance for esp8266 client
32 #endif
33 #if defined(ESP32) // creating the instance for esp32 client
34 #include <WiFiClient.h> // creating the instance for esp32 client
35 #endif
36
37 // =====
38 int led = 4;
39 int trig = 5;
40 int echo = 18;
41 void setup()
42 {
43   Serial.begin(115200);
```

```

36 pinMode(trig,OUTPUT);
37 pinMode(echo,INPUT);
38 pinMode(LED, OUTPUT);
39 delay(10);
40 wificonnect();
41 mqttconnect();
42 }
43 void loop()// Recursive function
44 {
45
46     digitalWrite(trig,LOW);
47     digitalWrite(trig,HIGH);
48     delayMicroseconds(10);
49     digitalWrite(trig,LOW);
50     float dur = pulseIn(echo,HIGH);
51     float dist = (dur * 0.0343)/2;
52     Serial.print ("Distance in cm");
53     Serial.println(dist);
54
55
56     PublishData(dist);
57     delay(1000);
58     if (!client.loop()) {
59         mqttconnect();
60     }
61 }
62
63
64
65 /*.....retrieving to cloud.....*/
66
67 void PublishData(float dist) {
68     mqttconnect();//function call for connecting to ihs
69     /*
70     | creating the String in in form json to update the data to ihs cloud

```

```

70 // creating the string in in form json to update the data to the cloud
71 */
72 String object;
73 if (dist < 100)
74 {
75     digitalWrite(LED, HIGH);
76     Serial.println("object is near");
77     object = "near";
78 }
79 else
80 {
81     digitalWrite(LED, LOW);
82     Serial.println("no object found");
83     object = "no";
84 }
85
86 String payload = "{\"distance\": ";
87 payload += dist;
88 payload += ", \"object\": \"";
89 payload += object;
90 payload += "\"}";
91
92
93 Serial.print("sending payload: ");
94 Serial.println(payload);
95
96
97
98

```

explains re • SuperHero • Arduino • Project, Arduino •

```

99
100 if (client.publish(topic, (char*) payload_str.c_str())) {
101     Serial.println("Publish ok");// if it successfully send data to the cloud then it will print publish ok in serial monitor or else it will print publish failed
102 } else {
103     Serial.println("Publish failed");
104 }
105
106 }
107
108 void mqttConnect() {
109     if (!client.connected()) {
110         Serial.print("reconnecting client to ");
111         Serial.print(server);
112         while (!client.connect(clientId, username, password)) {
113             Serial.print(".");
114             delay(100);
115         }
116
117         lastMessageReceived();
118         Serial.println();
119     }
120 }
121
122 void mqttConnect() //function definition for mqttConnect
123 {
124     Serial.println();
125     Serial.print("connecting to ");
126
127     WiFi.begin("mbed-SSL", "", 0); //passing the wifi credentials to initialize the connection
128     while (WiFi.status() != WL_CONNECTED) {
129         delay(100);
130         Serial.print(".");
131     }
132     Serial.println("");
133     Serial.println("wifi connected");
134     Serial.println("IP address: ");
135
136 }

```

```

124 WiFi.begin("Wotwi-GUEST", "", 0); //passing the wifi credentials to establish the connection
125 while (WiFi.status() != WL_CONNECTED) {
126     delay(500);
127     Serial.print(".");
128 }
129 Serial.println("");
130 Serial.println("WiFi connected");
131 Serial.println("IP address: ");
132 Serial.println(WiFi.localIP());
133 }
134
135 void InitManagedDevice() {
136     if (client.subscribe(subscribetopic)) {
137         Serial.println((subscribetopic));
138         Serial.println("subscribe to cmd OK");
139     } else {
140         Serial.println("subscribe to cmd FAILED");
141     }
142 }
143
144 void callback(char* subscribetopic, byte* payload, unsigned int payloadlength)
145 {
146
147     Serial.print("callback invoked for topic: ");
148     Serial.println(subscribetopic);
149     for (int i = 0; i < payloadlength; i++) {
150         //Serial.print((char)payload[i]);
151         data3 += (char)payload[i];
152     }
153
154     // Serial.println("data: " + data3);
155     // if(data3=="Near")
156     // {
157     // Serial.println(data3);
158     // }
159 }

```

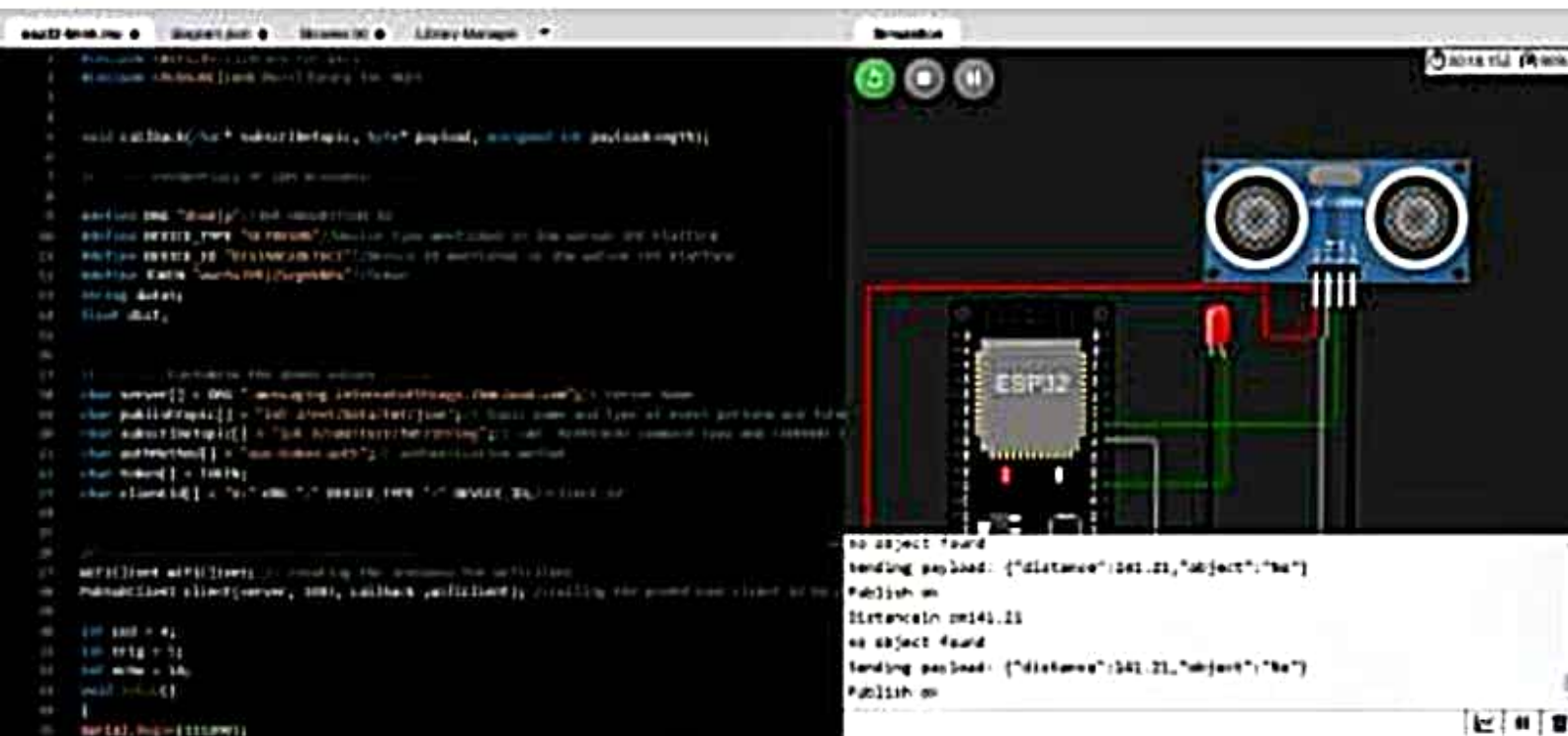


```

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

```

OUTPUT:



The screenshot displays the Arduino IDE interface. The left pane shows the code for an ESP8266, which includes a callback function for an MQTT topic. The right pane shows a breadboard circuit with an ESP12 module and an ultrasonic sensor (HC-SR04) connected to it. The bottom pane shows the serial monitor output, which includes the MQTT topic and the payload data.

Serial Monitor Output:

```

no object found
sending payload: {"distance":241.21,"object":"No"}
Publish on
distancein cm141.21
no object found
sending payload: {"distance":241.21,"object":"No"}
Publish on

```

IBM Watson IoT Platform

Groups Actions Device Types Interfaces

Add Device

DEVMONITOR | Recommended | STEADON | Status | Oct 20, 2022 9:45 AM

Identity Device Information Recent Events State Logs

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

Event	Value	Format	Last Received
Data	("distance":341.21,"object":"Near")	json	a few seconds ago
Data	("distance":341.21,"object":"Near")	json	a few seconds ago
Data	("distance":341.21,"object":"Near")	json	a few seconds ago
Data	("distance":341.21,"object":"Near")	json	a few seconds ago
Data	("distance":341.21,"object":"Near")	json	a few seconds ago

Items per page 50 | 1-2 of 2 items

1 of 1 page

when object is near to the ultrasonic sensor

acikol.com | esp32-arduino-ide

Simulation

```

object is near
sending payload: {"distance":97.82,"object":"Near"}
publish on
Distancein cm97.82
Object is near
sending payload: {"distance":97.82,"object":"Near"}
publish on
  
```

Data sent to the IBM Cloud Device when the object is near

