

ASSIGNMENT – 4

Name	:	GAYATHRI S
Date	:	06-11-2022
Team ID	:	PNT2022TMID42243
Register Number	:	710019106012
Project Title	:	Smart Solutions for Railway

PROGRAM CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "bmpy78"//IBM ORGANITION ID
#define DEVICE_TYPE "Quaddevice"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "Quadrays" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "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/347576318929404498>

WOWKI OUTPUT:

The screenshot shows the Wokwi web-based simulation environment. On the left, the code editor displays the `esp32-dht22.ino` sketch. The sketch initializes an MQTT client to connect to an IBM Watson IoT Platform organization. It configures pins for a HC-SR04 ultrasonic sensor and sets up a callback function to handle distance measurements. The circuit diagram on the right shows the physical connections between the ESP32 board and the HC-SR04 module. The terminal window on the right shows the simulation results, including an alert message and the JSON payload sent to the cloud.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
4 //-----credentials of IBM Accounts-----
5 #define ORG "bmpy78"/IBM ORGANIZATION ID
6 #define DEVICE_TYPE "Quaddevice"/Device type mentioned in ibm watson IOT Platform
7 #define DEVICE_ID "12345"/Device ID mentioned in ibm watson IOT Platform
8 #define TOKEN "Quadrays" //Token
9
10 String data;
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Data/fmt/json";
13 char subscribetopic[] = "iot-2/cmd/test/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 WiFiClient wifiClient;
18 PubSubClient client(server, 1883, callback, wifiClient);
19 const int trigPin = 5;
20 const int echoPin = 18;
21 #define SOUND_SPEED 0.034
22 long duration;
23 float distance;
24 void setup() {
25   Serial.begin(115200);
26   pinMode(trigPin, OUTPUT);
27   pinMode(echoPin, INPUT);
28   wifiConnect();
29   mqttConnect();
30 }
31 void loop()
32 {
33   digitalWrite(trigPin, LOW);
34   delayMicroseconds(2);
35   digitalWrite(trigPin, HIGH);
36 }
```

ALERT!!
Sending payload: {"Distance":98.94,"ALERT!!":"Distance less than 100cms"}
Publish ok
Distance (cm): 98.94
ALERT!!
Sending payload: {"Distance":98.94,"ALERT!!":"Distance less than 100cms"}
Publish ok

IBM CLOUD OUTPUT :

The screenshot shows the IBM Watson IoT Platform dashboard for a specific device. The left sidebar provides navigation and monitoring tools. The main area displays the device's identity and recent events. The event log table shows five entries of distance measurements and alerts, each timestamped as "a few seconds ago" or "a minute ago". The bottom of the page includes pagination controls and a note about simulations running.

Event	Value	Format	Last Received
Data	{"Distance":98.94,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":98.94,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":98.94,"ALERT!!":"Distance less than ...	json	a minute ago
Data	{"Distance":98.94,"ALERT!!":"Distance less than ...	json	a minute ago
Data	{"Distance":98.94,"ALERT!!":"Distance less than ...	json	a minute ago

Items per page 50 | 1-1 of 1 item

1 of 1 page < 1 >

0 Simulations running