

SPRINT-4

DATE	18 NOVEMBER 2022
TEAM ID	PNT2022TMID40922
PROJECT NAME	HAZARDOUS AREA MONITORING FOR INDUSTRIAL PLANT POWERED BY IoT

WOWKI CODE :

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>
#include "DHT.h"// Library for dht11
#define DHTPIN 15 // what pin we're connected to
#define DHTTYPE DHT22

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht
connected

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

//-----credentials of IBM Accounts-----

#define ORG "aqudbz"//IBM ORGANITION ID
#define DEVICE_TYPE "NodeMCU"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "EON8Q6-UN@GTJ&zH-Q" //Token
String data3;
float Humidity, Temp;

//----- Customise the above values -----
```

```
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform
and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command
type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
```

```
//-----
```

```
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by
passing parameter like server id,portand wificredential
```

```
void setup()// configureing the ESP32
```

```
{
  Serial.begin(115200);
  dht.begin();
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}
```

```
void loop()// Recursive Function
```

```
{

  Humidity = dht.readHumidity();
```

```
Temp = dht.readTemperature();  
Serial.print("Temp:");  
Serial.println(Temp);  
Serial.print("Humidity:");  
Serial.println(Humidity);
```

```
PublishData(Temp,Humidity);  
delay(1000);  
if (!client.loop()) {  
    mqttconnect();  
}  
}
```

/.....retrieving to Cloud...../

```
void PublishData(float Temp, float Humidity) {  
    mqttconnect();//function call for connecting to ibm  
    /*  
        creating the String in in form JSon to update the data to ibm cloud  
    */  
    String payload = "{\"Temp\":";  
    payload += Temp;  
    payload += "," \"Humidity\":";  
    payload += Humidity;  
    payload += "}";
```

```
Serial.print("Sending payload: ");
```

```
Serial.println(payload);
```

```
if (client.publish(publishTopic, (char*) payload.c_str())) {
```

```
    Serial.println("Publish ok");// if it successfully upload data on the cloud then it will print  
    publish ok in Serial monitor or else it will print publish failed
```

```
} 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() //function definition for wificonnect
```

```
{
```

```
    Serial.println();
```

```
Serial.print("Connecting to ");
```

```
WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection
```

```
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];
}
}

```

WOWKI OUTPUT :

The screenshot displays the Wokwi web IDE interface. On the left, the code editor shows a C++ program for an ESP32. The code includes a setup for WiFi, a loop to connect to a network, and a callback function to process incoming data. The right side of the interface shows a simulation of the ESP32 board connected to a DHT22 temperature and humidity sensor. Below the simulation, the serial output window shows the following text:

```

Connecting to .....
WiFi connected
IP address:
10.10.0.2
Reconnecting client to aqudbz.messaging.internetofthings.ibmcloud.com

```

LINK : <https://wokwi.com/projects/348655340794937938>

IBM WATSON PLATFORM :

The screenshot shows the IBM Watson IoT Platform interface. The browser address bar displays the URL: `https://aqudbz.internetofthings.ibmcloud.com/dashboard/devices/drilldown/NodeMCU:12345?returnTo=/devices/browse`. The page title is "Device Drilldown - 12345". On the left sidebar, the "Recent Events" menu item is selected. The main content area is titled "Recent Events" and includes a description: "The recent events listed show the live stream of data that is coming and going from this device." Below this is a table with the following data:

Event	Value	Format	Last Received
status	{"temperature":85,"humidity":25}	json	a few seconds ago
event_1	{"randomNumber":25}	json	a few seconds ago
status	{"temperature":1,"humidity":73}	json	a few seconds ago
status	{"temperature":16,"humidity":14}	json	a few seconds ago
status	{"temperature":41,"humidity":79}	json	a few seconds ago

At the bottom of the page, a status bar indicates "1 Simulation running".

The screenshot shows the IBM Watson IoT Platform dashboard. The browser address bar displays the URL: `https://aqudbz.internetofthings.ibmcloud.com/dashboard/boards/4cd98019-5601-4777-b769-42d7e7c309bb`. The page title is "HAM". On the left sidebar, the "HAM" menu item is selected. The main content area displays two line charts. The left chart is titled "Line chart" and shows "temperature" data. The right chart is titled "Line chart" and shows "humidity" data. Both charts have a y-axis ranging from 0 to 100 and an x-axis showing time from 21:30:30 to 21:31. Below each chart is a "1 minute" time range selector and a "now" button. At the bottom of the page, a status bar indicates "1 Simulation running".