

# SPRINT 1

TEAM ID	PNT2022TMID36752
PROJECT NAME	REAL-TIME RIVER WATER QUALITY MONITORING AND CONTROL SYSTEM
TEAM LEADER	DHARANIKUMAR.B
TEAM MEMBER 1	HARI.N
TEAM MEMBER 2	JAYACHANDRAN.R
TEAM MEMBER 3	JOHN YABAZ.S
TEAM MEMBER 4	SURESH.S

The screenshot displays the Wokwi IDE interface. On the left, the sketch.ino file is open, showing a C++ program. The code includes libraries for WiFi, MQTT, and DHT. It defines a DHT22 sensor on pin 15 and an LED on pin 2. The program sets up a WiFi connection and an MQTT client to send data to IBM Watson IoT Platform. The simulation window on the right shows a virtual ESP32 board connected to a DHT22 sensor and a red LED. The console output shows the device connecting to WiFi and sending data to IBM Watson IoT Platform.

```

1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #include "DHT.h" // Library for dht11
4 #define DHTPIN 15 // what pin we're connected to
5 #define DHTTYPE DHT22 // define type of sensor DHT 11
6 #define LED 2
7 DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and type
8
9 void callback(char* topic, byte* payload, unsigned int length) {
10
11 //-----credentials of IBM Accounts-----
12
13 #define ORG "910vsj" //IBM ORGANIZATION ID
14 #define DEVICE_TYPE "demo123" //Device type mentioned in ibm watson IOT Platform
15 #define DEVICE_ID "demo1234" //Device ID mentioned in ibm watson IOT Platform
16 #define TOKEN "demo1234" //Token
17 String data;
18 float t;
19
20 //----- Customise the above values -----
21
22 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server address
23 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of data
24 char subscribTopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENTATION
25 char authMethod[] = "use-token-auth"; // authentication method
26 char token[] = TOKEN;
  
```

Simulation window output:

```

Connecting to ..
WiFi connected
IP address:
10.10.0.2
Reconnecting client to
910vsj.messaging.internetofthings.ibmcloud.com
...
  
```

## PROGRAM

```

#include <WiFi.h> //library for wifi

#include <PubSubClient.h> //library for MQTT

#include "DHT.h" // Library for dht11

#define DHTPIN 15 // what pin we're connected to

#define DHTTYPE DHT22 // define type of sensor DHT 11

#define LED 2
  
```

**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 "910vsj"//IBM ORGANITION ID**

**#define DEVICE\_TYPE "dem0123"//Device type mentioned in ibm watson IOT Platform**

**#define DEVICE\_ID "demo1234"//Device ID mentioned in ibm watson IOT Platform**

**#define TOKEN "demo1234" //Token**

**String data3;**

**float t;**

**//----- 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();  
pinMode(LED,OUTPUT);  
delay(10);  
Serial.println();  
wificonnect();  
mqttconnect();  
}
```

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

```
    t = dht.readTemperature();  
    Serial.print("temperature:");  
    Serial.println(t);
```

```
    PublishData(t);  
    delay(1000);  
    if (!client.loop()) {  
        mqttconnect();  
    }  
}
```

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

```
void PublishData(float temp) {  
    mqttconnect();//function call for connecting to ibm  
    /*
```

**creating the String in in form JSon to update the data to ibm cloud**

**\*/**

**String payload = "{\"temperature\":\"";**

**payload += temp;**

**payload += "\"";**

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

**Serial.println(payload);**

**if (client.publish(publishTopic, (char\*) payload.c\_str())) {**

**Serial.println("Publish ok");// if it sucessfully 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 defination 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];  
}  
  
Serial.println("data: "+ data3);  
if(data3=="lighton")  
{  
Serial.println(data3);  
digitalWrite(LED,HIGH);  
  
}  
  
else  
{  
Serial.println(data3);  
digitalWrite(LED,LOW);  
  
}  
data3="";  
  
}
```

W sketch.ino - Wokwi Arduino and | W sketch.ino - Wokwi Arduino and | IBM Watson IoT Platform

910vsj.internetofthings.ibmcloud.com/dashboard/devices/browse

Google IBM Watson IoT Pla... IBM IBM ii IBM Cloud Node-RED : node-r... Application Details...

IBM Watson IoT Platform

kumardharani202@gmail.com  
ID: 910vsj

Browse

Action

Device Types

Interfaces

Add Device

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
event_1	{"randomNumber":1}	json	a few seconds ago
event_1	{"randomNumber":63}	json	a few seconds ago
event_1	{"randomNumber":20}	json	a few seconds ago
event_1	{"randomNumber":82}	json	a few seconds ago

2 Simulations running

Type here to search

ENG 07:14 PM 16-11-2022