

## SPRINT 1

Date	15 November 2022
Team ID	PNT2022TMID16188
Project Name	Project – Signs with Smart Connectivity for Better Road Safety

SPRINT	FUNCTIONAL REQUIREMENT (EPIC)	USER STORY/TASK	STORY POINTS	PRIORITY	TEAM MEMBERS
Sprint-1	Hardware utilization	Integrate the hardware to be able to access the cloud functions	2	Medium	SOWMYA B, SHAMYUKDHA PV, VIJAYALAKSHMI C V, VAISHNAVI N

### STEP 1:

### WOKWI SIMULATION

The code is simulated in WOKWI platform.

The screenshot displays the WOKWI simulation interface. On the left, the 'sketch.ino' file contains the following code:

```

31 WiFiClient wificlient; // creating the instance for wificlient
32 PubSubClient client(server, 1883, callback ,wificlient); //calling the predefi
33
34
35 void setup()// configuring the ESP32
36 {
37   Serial.begin(115200);
38   dht.begin();
39   pinMode(33, INPUT); //North
40   pinMode(25, INPUT); // South
41   pinMode(26, INPUT); // East
42   pinMode(27, INPUT); // West
43   delay(10);
44   Serial.println();
45   wificlient.connect();
46   mqttconnect();
47 }
48
49 int n, s, e, w;
50
51 void loop()// Recursive Function
52 {
53
54   h = dht.readHumidity();
55   t = dht.readTemperature();
56   Serial.print("temp:");
57   Serial.println(t);
58   Serial.print("humidity:");

```

On the right, the 'Simulation' window shows a visual representation of the ESP32 microcontroller connected to a DHT22 sensor. The sensor's output is displayed as follows:

```

{"temp":37.40,"humidity":86.00,"North":0,"South":0,"East":0,"West":0}
Publish ok
temp:37.40
humidity:86.00
Sending payload:
{"temp":37.40,"humidity":86.00,"North":0,"South":0,"East":0,"West":0}
Publish ok

```

## WOKWI CODE:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQTT
#include "DHT.h"// Library for dht11
#define DHTPIN 5      // what pin we're connected to
#define DHTTYPE DHT22  // define type of sensor DHT 11

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 "d5zx56"//IBM ORGANITION ID
#define DEVICE_TYPE "SMARTBOARD"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "SMARTCONNECTIVITY"//Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "957846465"      //Token
String data3;
float h, 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(33, INPUT); //North
    pinMode(25, INPUT); // South
    pinMode(26, INPUT); // East
    pinMode(27, INPUT); // West
    delay(10);
    Serial.println();
    wificonnect();
    mqttconnect();
}

int n, s, e, w;

void loop()// Recursive Function
{
    h = dht.readHumidity();
    t = dht.readTemperature();
    Serial.print("temp:");
    Serial.println(t);
    Serial.print("humidity:");
    Serial.println(h);

    n = digitalRead(33);
    s = digitalRead(25);
    e = digitalRead(26);
    w = digitalRead(27);

    PublishData(t, h, n, s, e, w);
    delay(1000);
    if (!client.loop()) {
        mqttconnect();
    }
}

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

void PublishData(float temp, float humid, int n, int s, int e, int w) {
    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 += humid;
    payload += "," " \"North\":";

```

```

payload += n;
payload += "," "\"South\":";
payload += s;
payload += "," "\"East\":";
payload += e;
payload += "," "\"West\":";
payload += w;
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");
    }
}

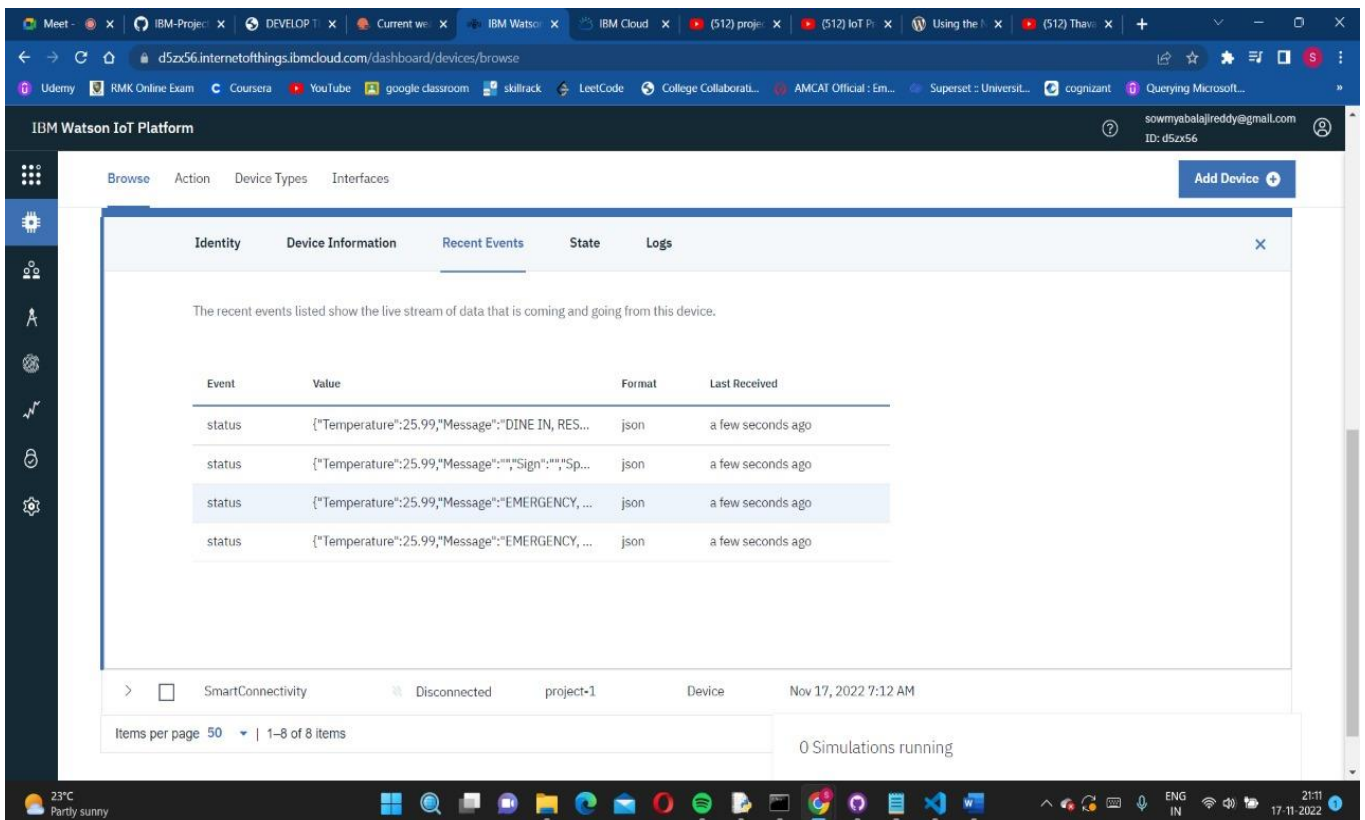
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{
    Serial.print("callback invoked for topic: ");
    Serial.println(subscribetopic);
    for (int i = 0; i < payloadLength; i++) {
        //Serial.print((char)payload[i]);
        data3 += (char)payload[i];
    }
    Serial.println("data: "+ data3);
    // if(data3=="lighton")
    // {
    // Serial.println(data3);
    // digitalWrite(LED,HIGH);
    // }
    // else
    // {
    // Serial.println(data3);
    // digitalWrite(LED,LOW);
    // }
    // data3="";
}

```

## STEP 2:

### IOT DEVICE- IOT PLATFORM

By simulating in wokwi platform,the data is published in IBM cloud.



The screenshot displays the IBM Watson IoT Platform dashboard. The top navigation bar includes tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various IoT functions. The main content area shows a modal window with the 'Recent Events' tab selected. This tab displays a table of events, including status updates and emergency messages, along with their values, formats, and last received times. Below the table, there is a status bar indicating the device is 'Disconnected' and '0 Simulations running'.

Event	Value	Format	Last Received
status	{"Temperature":25.99,"Message":"DINE IN, RES..."}	json	a few seconds ago
status	{"Temperature":25.99,"Message":"","Sign":"","Sp..."}	json	a few seconds ago
status	{"Temperature":25.99,"Message":"EMERGENCY, ..."	json	a few seconds ago
status	{"Temperature":25.99,"Message":"EMERGENCY, ..."	json	a few seconds ago

Items per page 50 | 1-8 of 8 items

0 Simulations running

THANK YOU