

# Sprint 4

Team ID	PNT2022TMID22874
Project Name	Personal Assistance for Seniors Who Are SelfReliant

## Code for Simulation:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQTT
#include <LiquidCrystal_I2C.h>
#include "DHT.h"// Library for dht11
#define DHTPIN 15 // what pin we're connected to
#define DHTTYPE DHT11 // 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 "64yf7x"//IBM ORGANITION ID
#define DEVICE_TYPE "b11m3edevicetype"//Device type mentioned in ibm watson
IOT Platform
#define DEVICE_ID "b11m3edeviceid"//Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "-&EMtr7l-v-Gz2G))e" //Token
String data3="";
int buzz= 13;

//----- 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
LiquidCrystal_I2C lcd(0x27,16,2);

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

```
void loop()// Recursive Function
{
    if (!client.loop())
        {mqttconnect();
        }
}
```

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

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

```
}
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("Medicine Name: "+ data3);
    if(data3 != "")
    {
        lcd.init();

        lcd.print(data3);
        digitalWrite(LED,HIGH);
        tone(buzz, 100, 1000);
        delay(2000);
        digitalWrite(LED,LOW);
        noTone(buzz);
        delay(1000);
    }
}

```

```

else
{
digitalWrite(LED,LOW);

}
data3="";
}

```

## Output:

The screenshot displays the Wokwi IDE interface for a project named "PNT2022TMD22874.ino". The code editor on the left contains the following C++ code:

```

1  #include <WiFi.h>
2  #include <PubSubClient.h>
3  #include <LiquidCrystal_I2C.h>
4  #include "DHT.h"
5  #define DHTPIN 15 // what pin we're connected to
6  #define DHTTYPE DHT11 // define type of sensor DHT 11
7  #define LED 2
8  DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and type of
9  void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
10
11
12 //-----credentials of IBM Accounts-----
13
14 #define ORG "97cp64"//IBM ORGANITION ID
15 #define DEVICE_TYPE "Healthcare"//Device type mentioned in ibm watson IOT Plat
16 #define DEVICE_ID "63829"//Device ID mentioned in ibm watson IOT Platform
17 #define TOKEN "6382987342" //Token
18 String data3="";
19 int buzz= 13;
20
21 //----- Customise the above values -----
22 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
23 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of even
24 char subscribetopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT comm
25 char authMethod[] = "use-token-auth"; // authentication method
26 char token[] = TOKEN;
27 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
28 LiquidCrystal_I2C lcd(0x27,16,2);
29
30 //-----

```

The simulation window on the right shows a visual representation of the hardware: an ESP32 microcontroller, a DHT11 temperature and humidity sensor, and a 16x2 LCD display. The terminal log at the bottom of the simulation window shows the following output:

```

WiFi connected
IP address:
10.10.0.2
Reconnecting client to 97cp64.messaging.internetofthings.ibmcloud.com
iot-2/cmd/command/fmt/String
subscribe to cmd OK

```