

SPRINT 4

Date	7/11/2022
Team ID	PNT2022TMID23222
Project Name	Personal Assistance for Seniors Who Are Self-Reliant.

CODE:

```
#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()// configuring 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 != "")
    {
        cd.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. On the left, the code for 'ESP32 NTP Example.ino' is shown, which includes libraries for WiFi, PubSubClient, LiquidCrystal_I2C, and DHT. It defines pins for DHT (DHTPIN 15, DHTTYPE DHT11) and an LED (LED 2). The code sets up an ESP32 module with an NTP client, connecting to an IBM Watson IoT Platform. It includes credentials for an IBM account and defines a publish topic for sending data. The code also initializes a DHT sensor and an LCD display (LiquidCrystal_I2C lcd(0x27,16,2)).

On the right, the 'Simulation' window shows a virtual representation of the ESP32 module connected to an LCD display and a buzzer. The ESP32 module is connected to the LCD display via I2C (SCL to pin 15, SDA to pin 4) and to the buzzer via a digital pin (pin 13). The simulation is running, as indicated by the green play button icon.