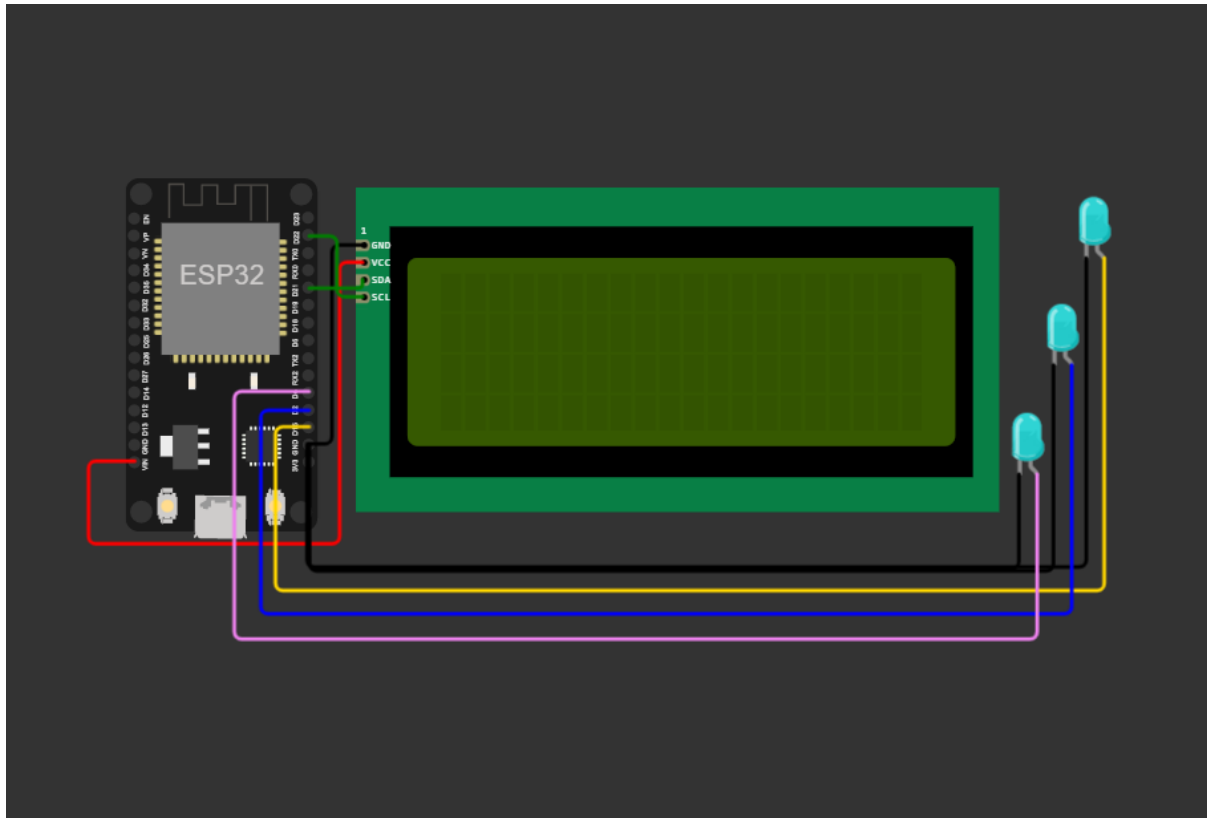


SPRINT - 3

Team ID	PNT2022TMID27886
Project Name	Project - Personal Assistance for Senior Citizens Who are self reliant

Wokwi Setup:-

Circuit:-



Code:-

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

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

#include<ESP32Servo.h>

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 20, 4);

#define LED 2

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

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

#define ORG "0sha8g"//IBM ORGANITION ID

#define DEVICE_TYPE "1234"//Device type mentioned in ibm watson IOT
Platform

#define DEVICE_ID "1919"//Device ID mentioned in ibm watson IOT
Platform

#define TOKEN "12345678" //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
```

```
char compartment_med[3][20];

char compartment_count[3][20], alertArr[20];

String username=" ", devicedetails, alert;

int f=0;

void setup() // configureing the ESP32
{
    Serial.begin(115200);

    pinMode(LED, OUTPUT);

    pinMode(4, OUTPUT);

    pinMode(2, OUTPUT);

    pinMode(15, OUTPUT);

    delay(10);

    Serial.println();

    wificonnect();

    mqttconnect();

    //PublishData("getdetail");

    //PublishData(compartment);

    PublishData("getdetail");

    lcd.begin(20, 4);

    lcd.init();

    lcd.backlight();

    lcd.setCursor(5, 1);

    lcd.print("GetMeds");

    lcd.setCursor(6, 2);

    lcd.print("Welcome!");

    delay(2000);

    lcd.clear();

}

void loop() // Recursive Function
```

```
{  
  
    //PublishData(compartment);  
  
    PublishData("getdetail");  
  
    lcd.clear();  
  
    lcd.setCursor(0,0);  
  
    lcd.print("Username:"+username);  
  
    //lcd.setCursor(0,1);  
  
    //lcd.print("Medicine : Count");  
  
    //lcd.setCursor(0,2);  
  
    for(int i =0;i<2;i++)  
    {  
  
        /*lcd.print(compartment_med[i]);  
  
        lcd.print(":");  
  
        lcd.print(compartment_count[i]);  
  
        lcd.print(" ");*/  
  
        devicedetails += compartment_med[i];  
  
        devicedetails += "$";  
  
        devicedetails += compartment_count[i];  
  
        devicedetails += "$";  
  
    }  
  
    if(f==0)  
    {  
  
        digitalWrite(4, LOW);  
  
        digitalWrite(2, LOW);  
  
        digitalWrite(15, LOW);  
  
        lcd.setCursor(0,1);  
  
        lcd.print(compartment_med[0]);  
  
        lcd.print(":");  
  
        lcd.print(compartment_count[0]);  
  
    }
```

```
lcd.setCursor(0,2);

lcd.print(compartment_med[1]);

lcd.print(":");

lcd.print(compartment_count[1]);

lcd.setCursor(0,3);

lcd.print(compartment_med[2]);

lcd.print(":");

lcd.print(compartment_count[2]);

}

else if(f==1)

{

    lcd.setCursor(0,1);

    lcd.print("It's time to take ");

    lcd.setCursor(0,2);

    lcd.print("medicine ,");

    lcd.setCursor(0,3);

    lcd.print(alert);

    for(int j=0;j<alert.length();j++)

    {

        alertArr[j]=alert[j];

    }

    Serial.println(alertArr);

    for(int i=0;i<3;i++)

    {

        if(strcmp(compartment_med[i],alertArr)==0)

        {

            if(i==0){

                digitalWrite(15, HIGH);

            }

        }

    }

}
```

```

        else if(i==1){

            digitalWrite(2, HIGH);

        }

        else if(i==2){

            digitalWrite(4, HIGH);

        }

    }

}

//Serial.println(devicedetails);

devicedetails = "";

delay(1000);

if (!client.loop()) {

    mqttconnect();

}

}

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

void PublishData(String comp) {

    mqttconnect();//function call for connecting to ibm

    /*

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

    */

    String payload="";

    if(comp=="getdetail")

    {

        payload="{\"Command\":\"";

        payload+="\"";

        payload+=comp;

    }

}

```

```

    payload+="\"";

    payload += " ,";

    payload += "\"user\":";

    payload += "\"";

    payload += username;

    payload += "\"";

    payload += " }";

}

else {

payload = "{\"Medicine\":";

payload += "\"";

payload += comp ;

payload += "\"";

payload += " ,";

payload += "\"User\":";

payload += "\"";

payload += username;

payload += "\",";

payload += "\"Command\":";

payload += "\"senddetail\"";

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(data3);

    char arr[6][20];

    int k=0;

    for(int i=0 ;i<6;i++)

    {

        if(k<=data3.length()){

            for(int j=0 ;j<20;j++)

            {

                if(data3[k]!='$'){

                    arr[i][j] = data3[k++];

                }

            }

            else {


```

```
        arr[i][j]='\0';

        k++;

        break;

    }

}

}

else

break;

}

Serial.println(arr[0]);

if(strcmp(arr[0],"clearAlert")==0)

{

    f=0;

}

else if(strcmp(arr[0],"ALERT")==0){

    f=1;

    alert=arr[1];

}

else {

    k=0;

    for(int i=0;i<3;i++)

    {

        strcpy(compartment_med[i] , arr[k++]);

        strcpy(compartment_count[i] , arr[k++]);

    }

}

data3="";

}
```