

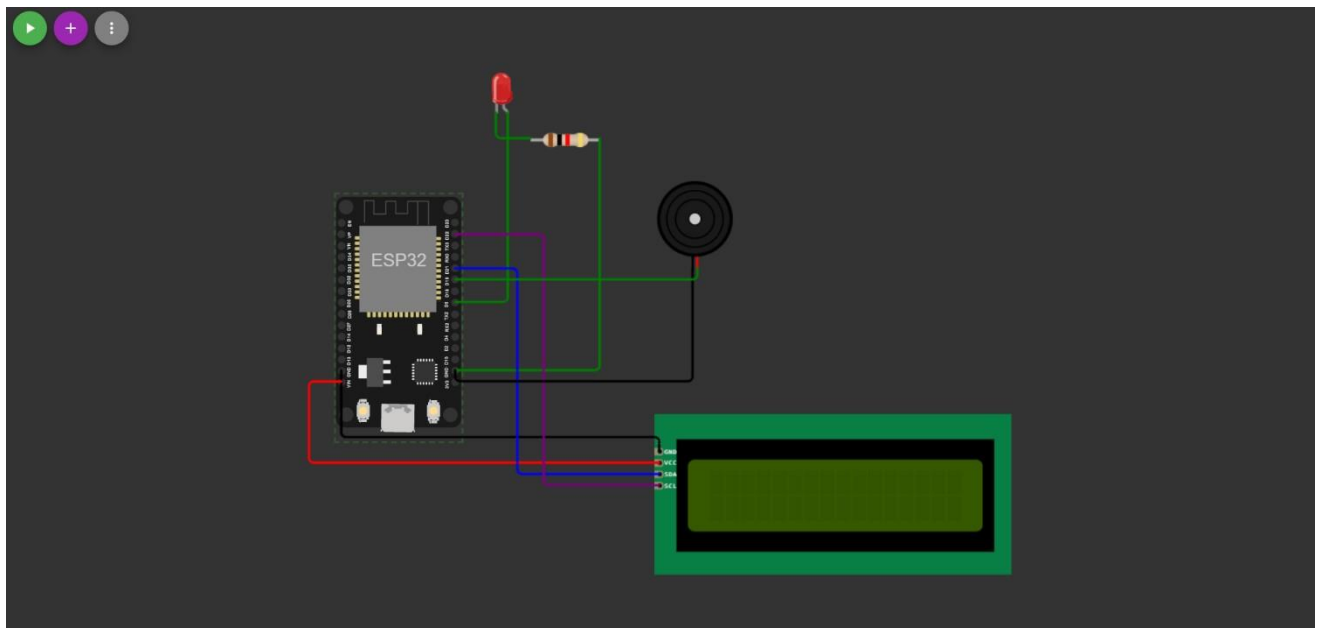
# PROJECT DEVELOPMENT PHASE-SPRINT 3

Team ID	PNT2022TMID00130
Project Name	Personal assistance for Seniors who are self-reliant

**SPRINT 3** -*Creating a IOT device using esp32 and notify user when the medicine time arrives.*

## 1.DEVICE SETUP

The device consists of ESP32 which is used for connecting with the ibm cloud r to publish and subscribe data. The LED glows when the medicine time arrives. The LCD displays the medicine name and the buzzer rings.



### CODE:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
#define LED 5
#include <LiquidCrystal_I2C.h>
```

```

LiquidCrystal_I2C lcd(0x27,16,2);
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "by18wl"//IBM ORGANITION ID
#define DEVICE_TYPE "IOT_DEVICE"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "123456789"
//Token String data3,light; float h,
t;

#define BUZZER_PIN 19 // ESP32 GIOP21 pin connected to Buzzer's pin
//----- 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/test/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()// configuring the ESP32
{
    Serial.begin(115200);
    Serial.begin(9600); //
dht.begin();
pinMode(LED,OUTPUT);
pinMode(BUZZER_PIN, OUTPUT);
delay(10); lcd.init();
lcd.clear();
lcd.backlight();
Serial.println();
wificonnect();
mqttconnect();
} void loop()// Recursive
Function
{

```



```

    digitalWrite(BUZZER_PIN,
HIGH);    delay(1000);    if
(!client.loop()) {
mqttconnect();
    }
}

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);
light=(char)payload[0];    for (int i = 1; i <
payloadLength; i++) {

        Serial.print((char)payload[i]);
data3 += (char)payload[i];
    }

        // Make sure backlight is on


    Serial.println("data: "+ data3);
if(light=="n")
{
    digitalWrite(BUZZER_PIN,
HIGH);

Serial.println(data3); digitalWrite(LED,HIGH);
    // Print a message on both lines of the LCD.
lcd.setCursor(2,0);    //Set cursor to character 2 on line 0
lcd.print("It's time for");
    lcd.setCursor(2,1);    //Move cursor to character 2 on
line 1    lcd.print(data3);    delay(3000);
digitalWrite(BUZZER_PIN, LOW);    digitalWrite(LED,LOW);
lcd.clear();

}
else
{
    digitalWrite(BUZZER_PIN, LOW);
Serial.println(data3);
digitalWrite(LED,LOW);
lcd.clear();

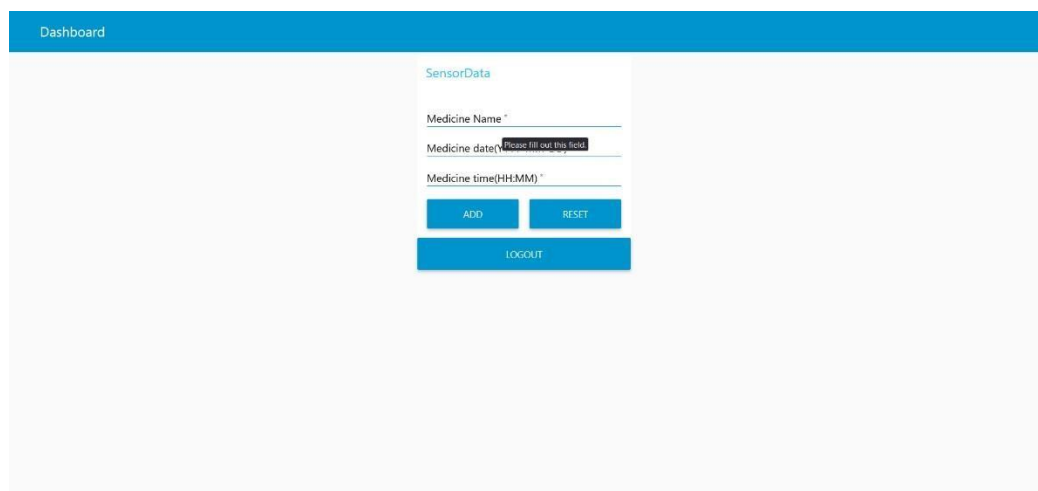
}
data3="";

}

```

## 2. WEB DASHBOARD

The user can enter the medicine name, date and time when it has to be consumed which will be stored in the cloudant database and node-red checks in the cloudant database if any medicine has to be taken every minute.



The screenshot shows a web dashboard with a blue header bar labeled 'Dashboard'. In the center, there is a white box titled 'SensorData'. Inside this box, there are three input fields: 'Medicine Name \*', 'Medicine date \*', and 'Medicine time(HH:MM) \*'. The 'Medicine date \*' field has a red error message 'Please fill out this field'. Below the input fields are two blue buttons labeled 'ADD' and 'RESET'. At the bottom of the white box is a blue button labeled 'LOGOUT'.

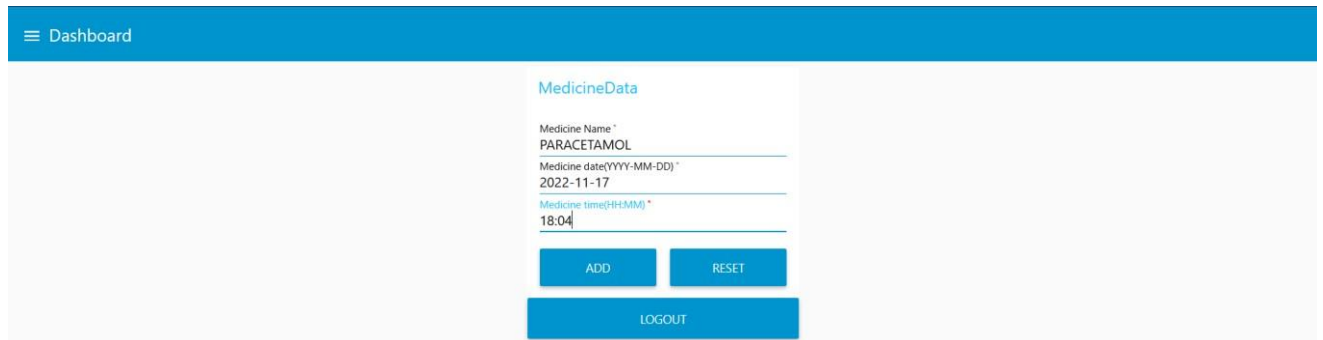
## 3. NODE-RED FLOW

When user adds a medicine into the database the node red flow keeps checking the database every minute and if a medicine has to be taken, it issues a command to the IOT device through IBM IOT Watson platform.



When the medicine time arrives the node-red flow sends DEVICE COMMAND to the IBM IOT platform. ESP32 which has subscribed to the IBM IOT platform turns the LED and displays the medicine name on the LCD display on receiving the command.

- ADDING MEDICINE

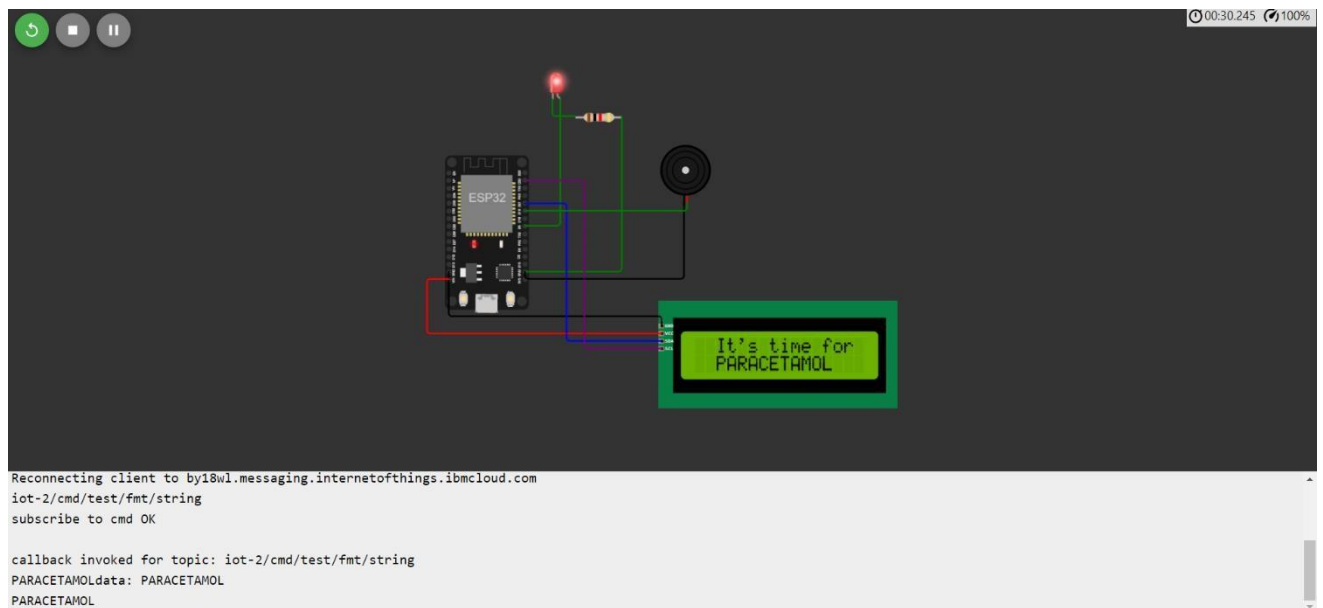


The screenshot shows a web application titled "MedicineData" with a "Dashboard" header. The main content area contains a form with the following fields:

- Medicine Name\*: PARACETAMOL
- Medicine date(YYYY-MM-DD)\*: 2022-11-17
- Medicine time(HHMM)\*: 18:04

Below the form are three buttons: "ADD", "RESET", and "LOGOUT".

- When the medicine time arrives the LED in the device glows, LCD displays the medicine name and buzzer rings.



*DEVELOPMENT IN SPRINT 4: In Sprint 4, text to speech service will be implemented in the IOT device and Mobile app to monitor and control the medicine schedule will be implemented.*