單元名稱 (LAB I2605)

Name : 劉泰佑
Student ID : C100E030

1.1 Objective

To be able to setup the raspberry pi to run server, database CRUD (create, read, update, and delete), and setup ESP8266 to be able to send temperature and humidity data using DHT22 sensor also control LED color either manually from the website or automatically as temperature indicator.

1.2 Working theory

By running MySQL in raspberry pi we can utilize it as a server so we can do data transfer from the website and the esp8266. In this case we use it to send temperature and humidity data manually from the website into database and show it in the website, next we use it to control led color from the website and then the esp8266 will catch the data from the database and use it to change the led color, and last one is we use the esp8266 to capture temperature and humidity data using dht22 and send it to database and the website will show the data and decide the led color to indicate the temperature status. To make all these work we need to create API to work as a bridge so that we can connect our interfaces seamlessly.

1.3 Experimental device and components)

- Raspberry Pi
- ESP8266
- DHT22
- RGB LED
- Access Point
- Putty
- WinSCP
- Visual Studio Code
- Thonny

1.4 Procedure

Step 1

First, we need to create new database, in this case I name my database as C100E030.

```
MariaDB [(none)]> create database C100E030;
Query OK, 1 row affected (0.001 sec)
```

Fig. 1.4-1

Step 2

And then we need to use the database.

```
MariaDB [(none)]> use C100E030
Database changed
```

Fig. 1.4-2

Step 3

After that, we can continue to make LEDCOLOR table.

```
MariaDB [C100E030]> create table LEDCOLOR (RedLED char(1), GreenLED char(1), Blu
eLED char(1), Time timestamp default current_timestamp);
Query OK, 0 rows affected (0.071 sec)
```

Fig. 1.4-3

Step 4

Next up is we also need to create DHT table so that we can store our captured temperature and humidity data.

```
MariaDB [C100E030]> create table DHT (ID integer auto_increment primary key, Tem
p char(10), Humidity char(10), Time timestamp default current_timestamp);
Query OK, 0 rows affected (0.079 sec)
```

Fig. 1.4-4

Step 5

Next we need to make the interface for our website so that we can easily monitor and control the systems, in this case I use visual studio code to design the interface and after that we open the file in the browser.

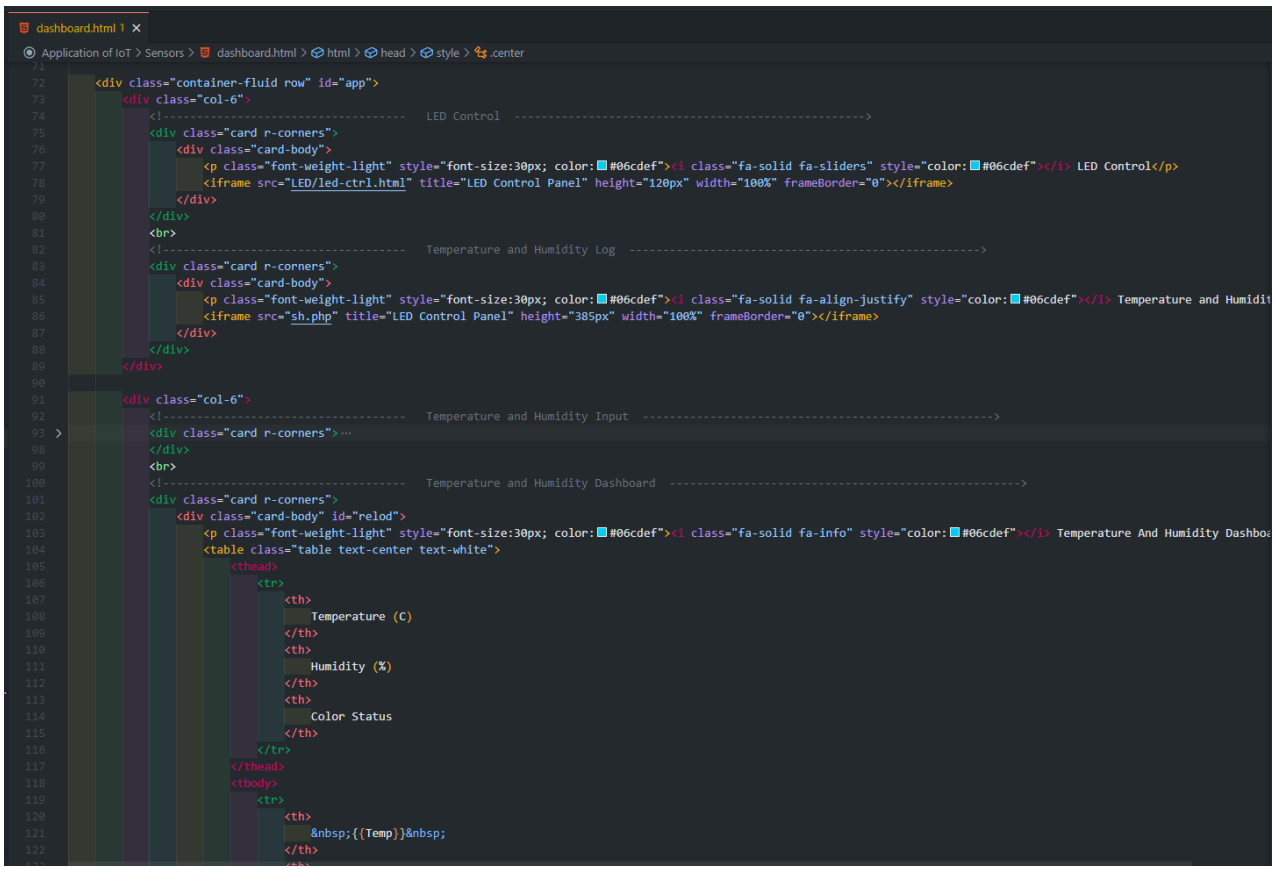


Fig. 1.4-5.1

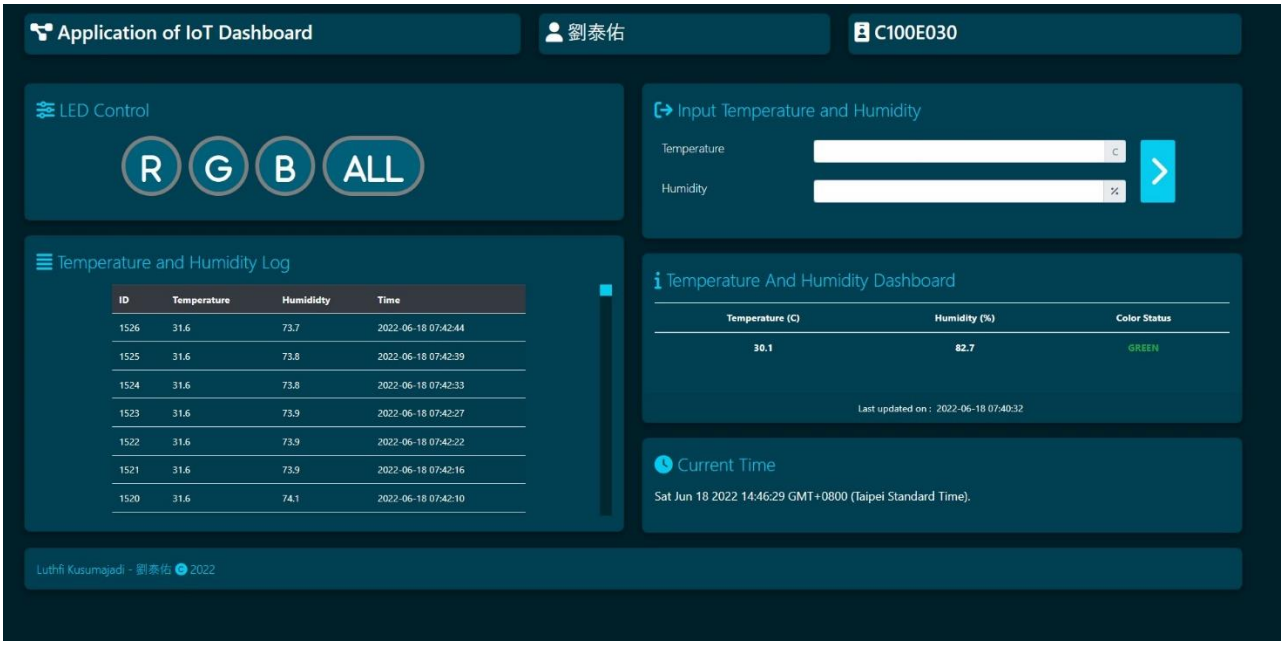


Fig. 1.4-5.2

Step 6

Next we need to create APIs so that we can get and post data to the database, this picture is just an example, but every API structure are pretty much the same as the picture.

```php
<?php
    header("Content-Type:text/html; charset=utf-8");
    date_default_timezone_set("asia/taipei");
    $dns="mysql:host=127.0.0.1;dbname=C100E030";
    try{
        $db=new PDO($dns,'C100E030','raspberrypi');
        $db->exec("set names utf8");
        $sql = "select RedLED,GreenLED,BlueLED from LEDCOLOR ORDER BY Time ASC LIMIT 1;";
        $myArray = array();
        $result=$db->query($sql);

        if($row=$result->fetch(PDO::FETCH_ASSOC))
        {
            $myArray[] = $row;
        }
        //echo  "\n";
        echo json_encode($myArray);
        //echo "\n";
        $db=null;

        // print json_encode($data);
    }catch(PDOException $e){
        printf("DATABASE_ERRMESG:%s",$e->getMessage());
    }
?>
```

Fig. 1.4-6

Step 7

After we finish with the website side, we move to the esp8266 side, where we need to make a program so that the esp8266 will get the led color from the database so that it can change the led color in Thonny.

```
No 2.py
        pass
print('Network connected!')
print(sta_if.ifconfig())
rpi_headers = {"Content-type": "application/x-www-form-urlencoded","Accept": "text/plain"}
res = requests.get(url= ip +'/C100E030/LED/GetLED.php', headers=rpi_headers)
response=ujson.loads(res.content)
RedLED = int(response[0]['RedLED'])
GreenLED = int(response[0]['GreenLED'])
BlueLED = int(response[0]['BlueLED'])
LEDSW(RedLED, GreenLED, BlueLED)

while True:
  if sta_if.isconnected():

    #********************************************************************************
    #HTTP GET request
    rpi_headers = {"Content-type": "application/x-www-form-urlencoded","Accept": "text/plain"}
    res = requests.get(url= ip +'/sensors/C100E030/LED/GetLED.php', headers=rpi_headers)
    print(res.content.decode())
    print(ujson.loads(res.content))
    response=ujson.loads(res.content)
    print(type(response))
    print("LED RED = {}, LED GREEN = {}, LED BLUE = {}".format(response[0]['RedLED'], response[0]['GreenLED'], response[0]['BlueLED']))
    RedLED = int(response[0]['RedLED'])
    GreenLED = int(response[0]['GreenLED'])
    BlueLED = int(response[0]['BlueLED'])
    print(RedLED)8
    print(GreenLED)
    print(BlueLED)
    LEDSW(RedLED, GreenLED, BlueLED)

    time.sleep(0.5)
```

Fig. 1.4-7

Step 8

Next we need to make a program so that the esp8266 can capture data from DHT22 and send it into database and receive led color data from the database in Thonny.

```
No 2.py ×  No 3.py ×
64    # 等一下它連接
65    while not sta_if.isconnected():
66        pass
67 print('Network connected!')
68 print(sta_if.ifconfig())
69
70 while True:
71     if sta_if.isconnected():
72         rpi_headers = {"Content-type": "application/x-www-form-urlencoded","Accept": "text/plain"}
73         d22.measure()
74         temp=d22.temperature()
75         humidity=d22.humidity()
76         print('Temperature = {:.1f}°C'.format(temp)) # eg. 23 (°C)
77         print('Humidity = {:.1f}%'.format(humidity))   # eg. 41 (% RH)
78         querydata="Temp={:.1f}&Humidity={:.1f}".format(temp, humidity)
79         #print(querydata)
80         response = requests.post(url= ip +'/sensors/C100E030/AddData.php?' + querydata, headers=rpi_headers)
81         response.close()
82
83         res = requests.get(url= ip +'/sensors/C100E030/LED/GetLED.php', headers=rpi_headers)
84         print(res.content.decode())
85         print(ujson.loads(res.content))
86         response=ujson.loads(res.content)
87         print(type(response))
88         print("LED RED = {}, LED GREEN = {}, LED BLUE = {}".format(response[0]['RedLED'], response[0]['GreenLED'], response[0]['BlueLED']))
89         RedLED = int(response[0]['RedLED'])
90         GreenLED = int(response[0]['GreenLED'])
91         BlueLED = int(response[0]['BlueLED'])
92         print(RedLED)
93         print(GreenLED)
94         print(BlueLED)
95         LEDSW(RedLED, GreenLED, BlueLED)
96
97         time.sleep(1)
98
```

Fig. 1.4-8

Step 9

Open WinSCP so that we can transfer all the files that we made into the raspberry pi “/var/www/html/” directory.

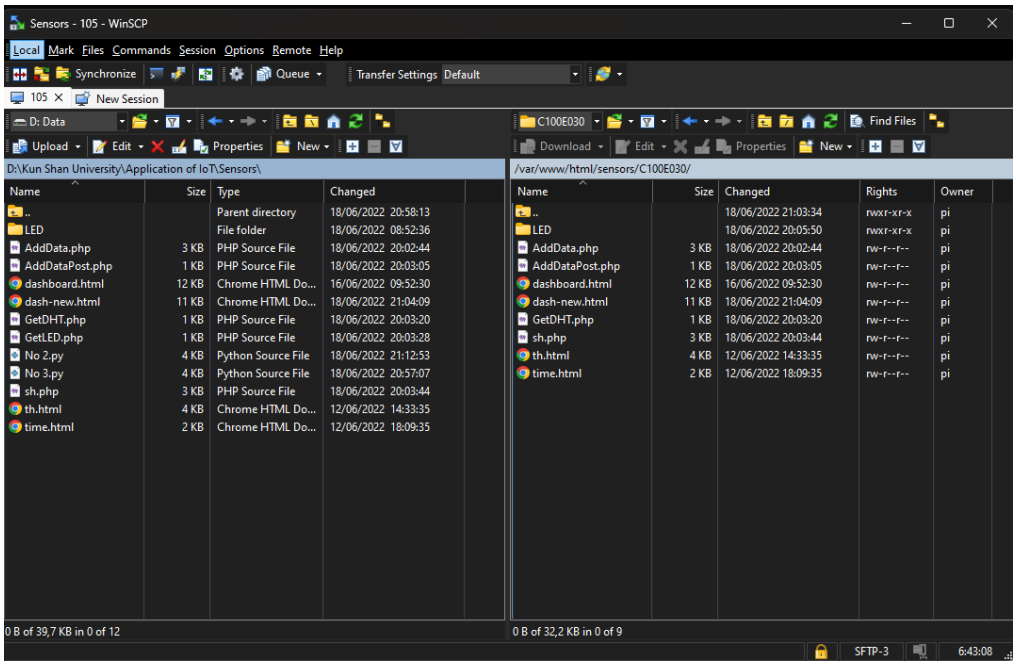


Fig. 1.4-9

Step 10

After that we can try access our raspberry pi ip address to check if it can run our program in the browser.

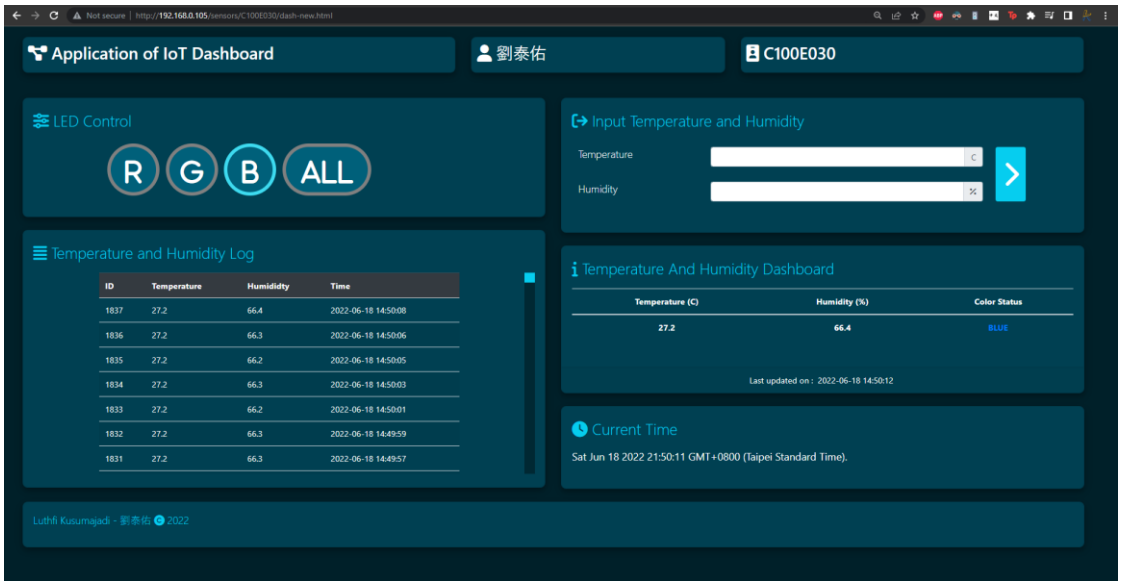


Fig. 1.4-10

1.5 Results

Finally we can try if our work is working or not, since we have three objective here, we have to try out all of our three programs.

1. First one is to manually input temperature and humidity from the website, input the value in the top right box.

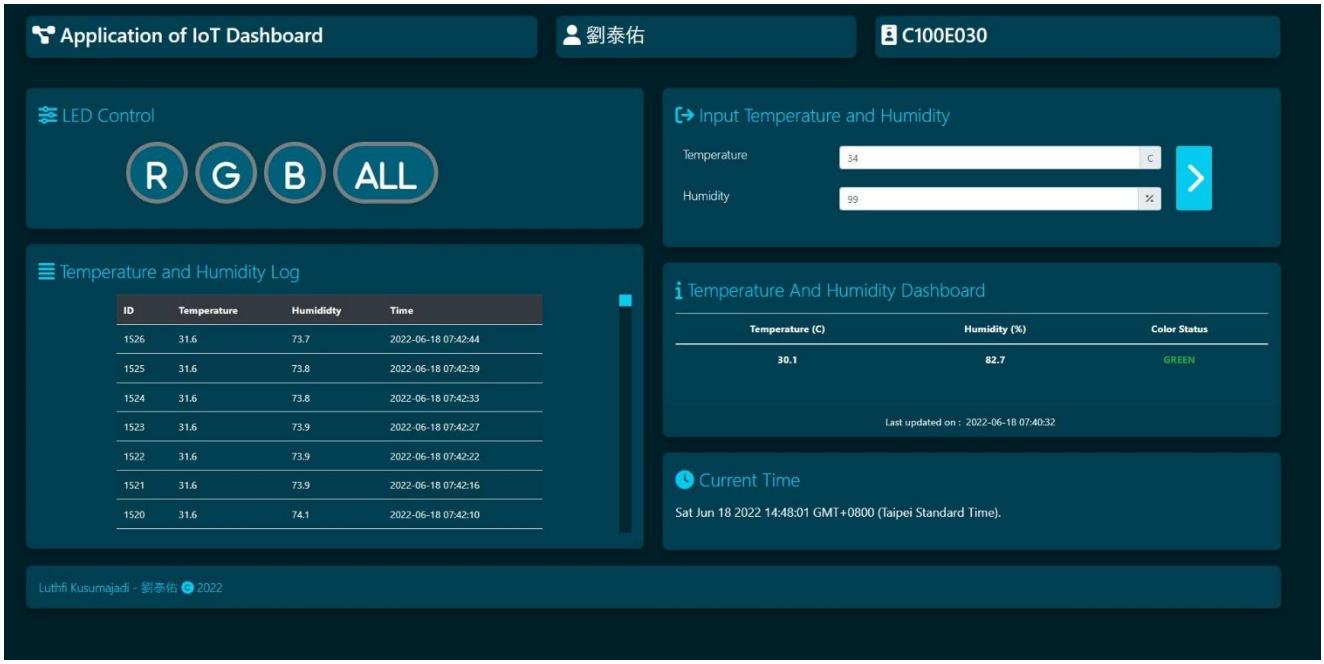


Fig. 1.5-1

2. And press the blue button if success it will show a success text.

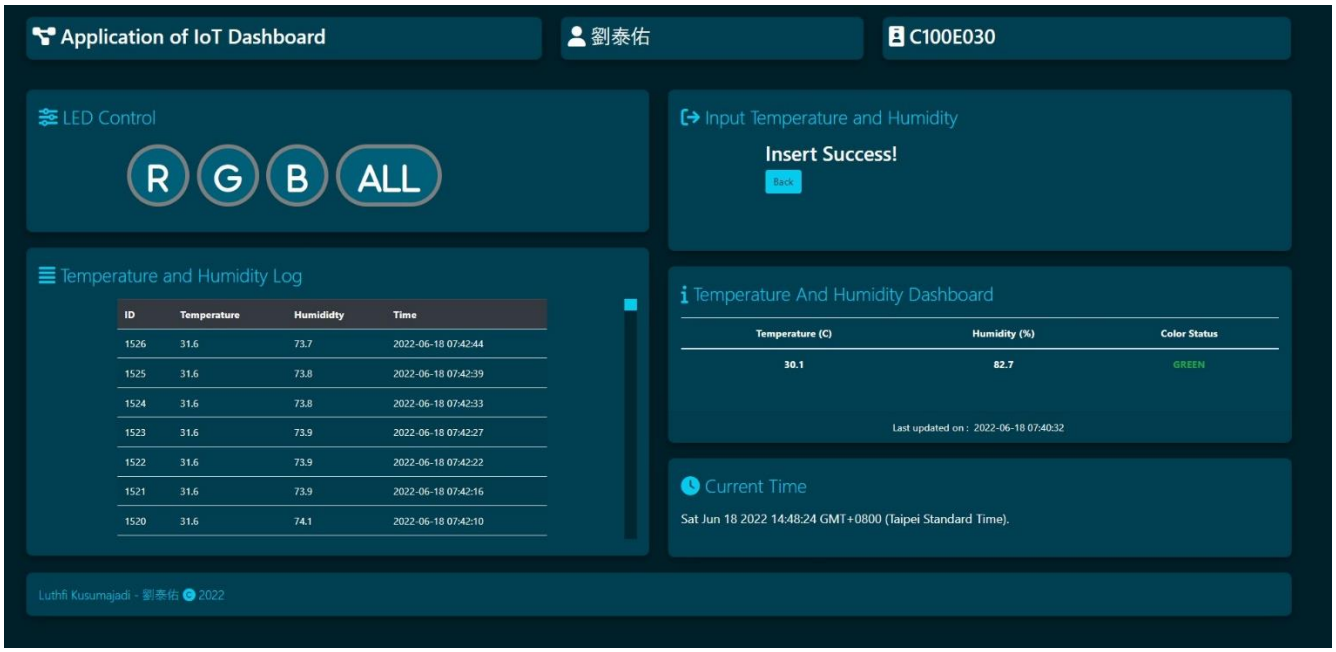


Fig. 1.5-2

3. Also the data will be shown in the table on the bottom left side.

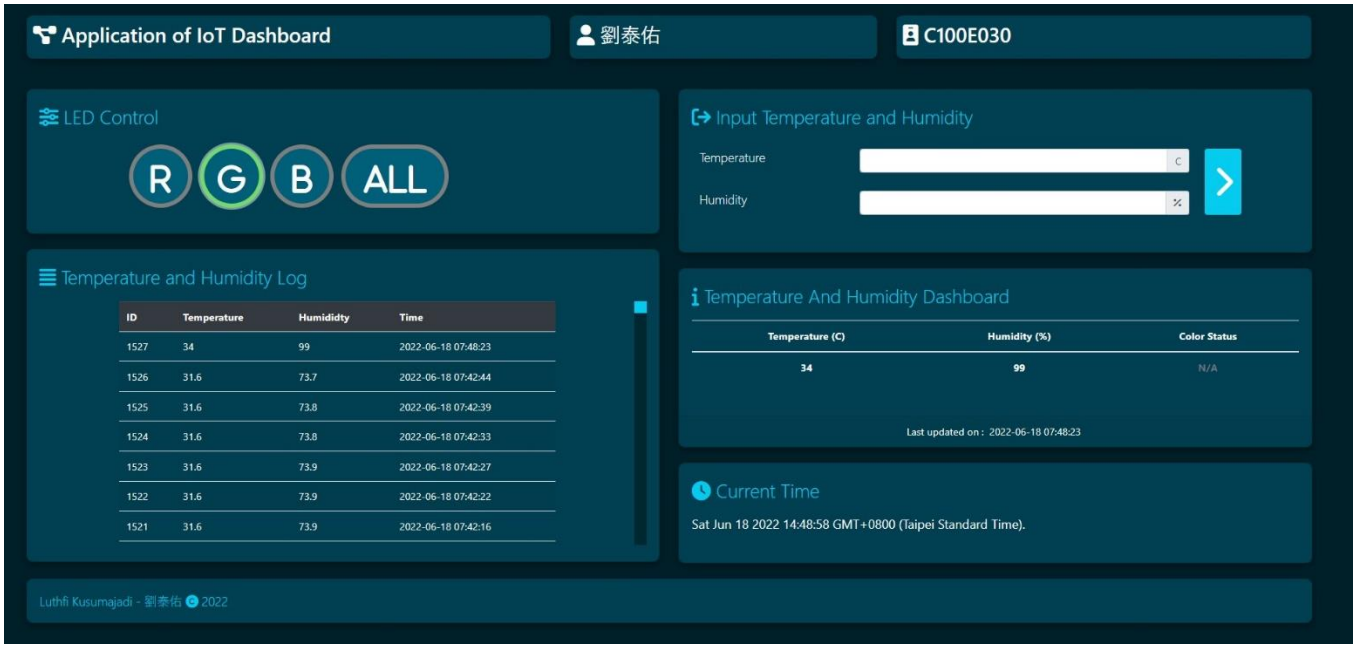
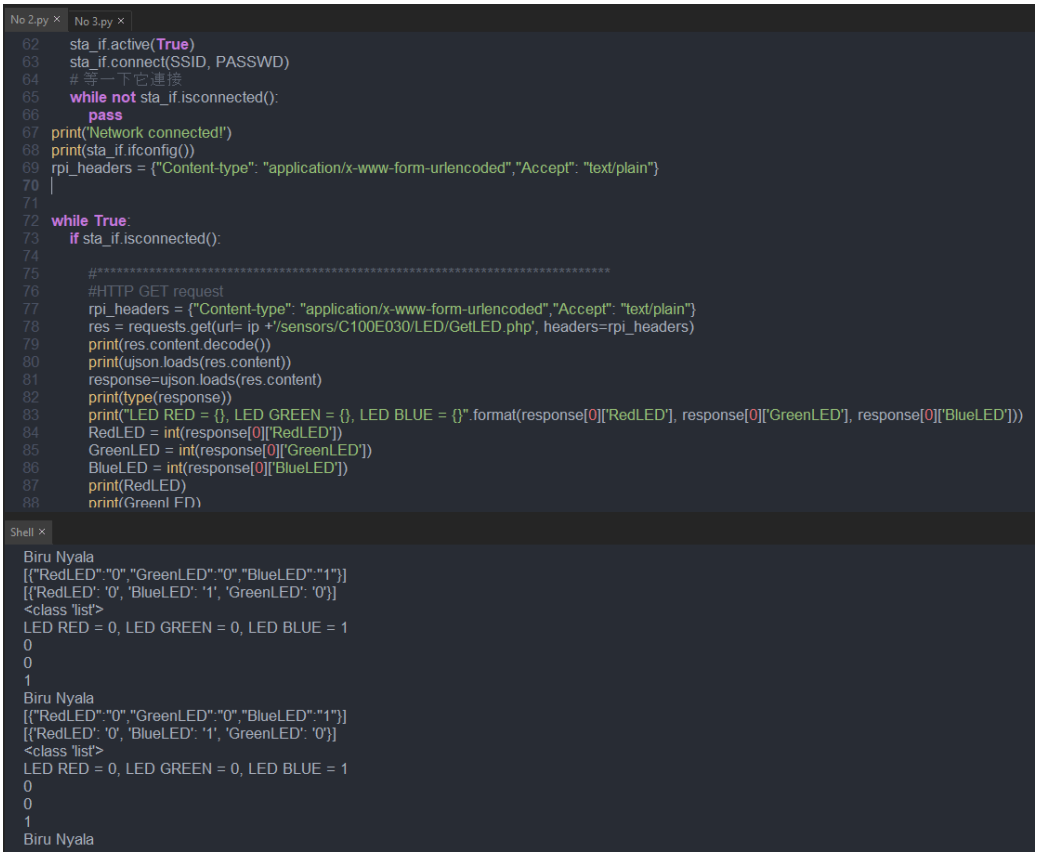


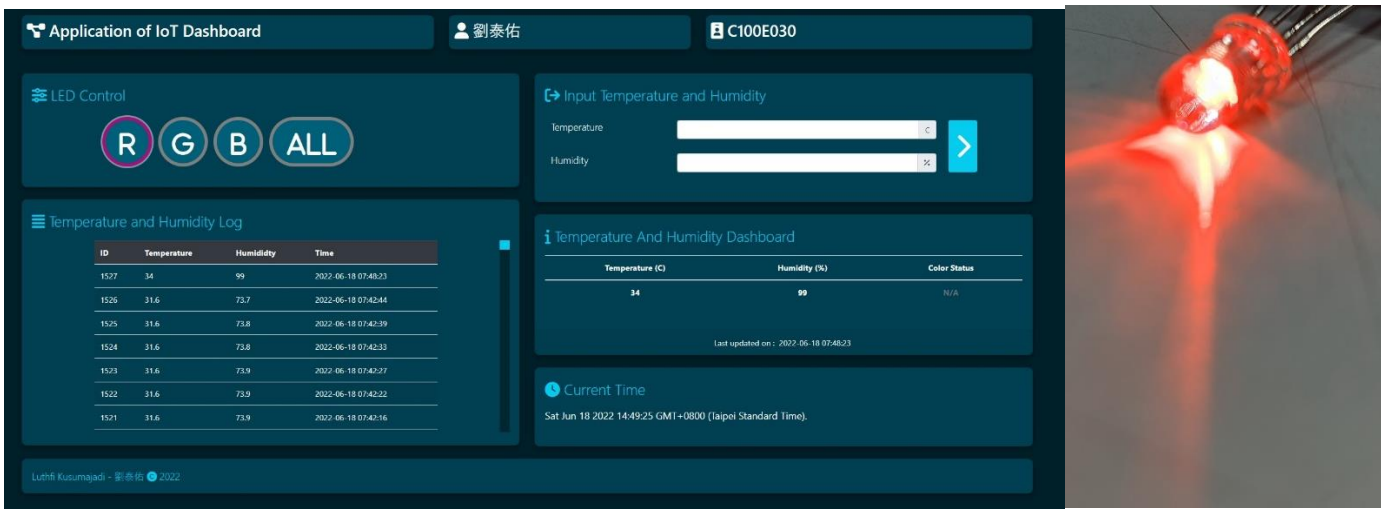
Fig. 1.5-3

4. The first program is a success, next we want to try the led control program, first run the No.2 thonny program, and try to control the color using the buttons in the top left side of the website.



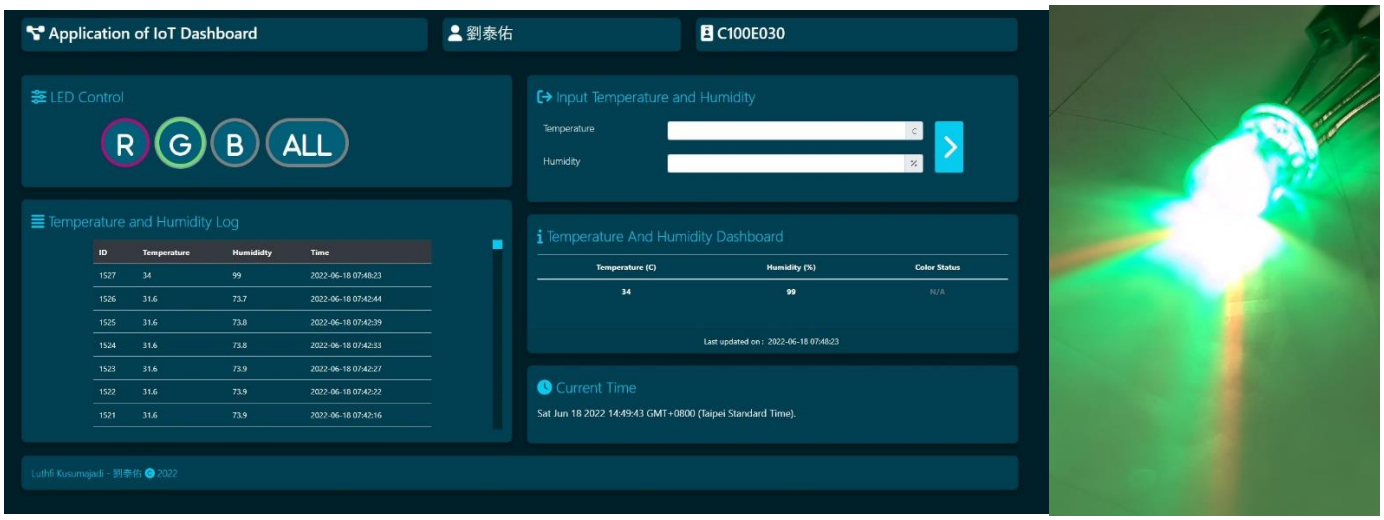
```
No 2.py ×  No 3.py ×
    sta_if.active(True)
    sta_if.connect(SSID, PASSWD)
    # 等一下它連接
    while not sta_if.isconnected():
        pass
print('Network connected!')
print(sta_if.ifconfig())
rpi_headers = {"Content-type": "application/x-www-form-urlencoded","Accept": "text/plain"}

while True:
    if sta_if.isconnected():

        #********************************************************************************
        #HTTP GET request
        rpi_headers = {"Content-type": "application/x-www-form-urlencoded","Accept": "text/plain"}
        res = requests.get(url= ip +'/sensors/C100E030/LED/GetLED.php', headers=rpi_headers)
        print(res.content.decode())
        print(ujson.loads(res.content))
        response=ujson.loads(res.content)
        print(type(response))
        print("LED RED = {}, LED GREEN = {}, LED BLUE = {}".format(response[0]['RedLED'], response[0]['GreenLED'], response[0]['BlueLED']))
        RedLED = int(response[0]['RedLED'])
        GreenLED = int(response[0]['GreenLED'])
        BlueLED = int(response[0]['BlueLED'])
        print(RedLED)
        print(GreenLED)
Shell ×
Biru Nyala
[{"RedLED":"0","GreenLED":"0","BlueLED":"1"}]
[{'RedLED': '0', 'BlueLED': '1', 'GreenLED': '0'}]
<class 'list'>
LED RED = 0, LED GREEN = 0, LED BLUE = 1
0
0
1
Biru Nyala
[{"RedLED":"0","GreenLED":"0","BlueLED":"1"}]
[{'RedLED': '0', 'BlueLED': '1', 'GreenLED': '0'}]
<class 'list'>
LED RED = 0, LED GREEN = 0, LED BLUE = 1
0
0
1
Biru Nyala
```

Fig. 1.5-4.1



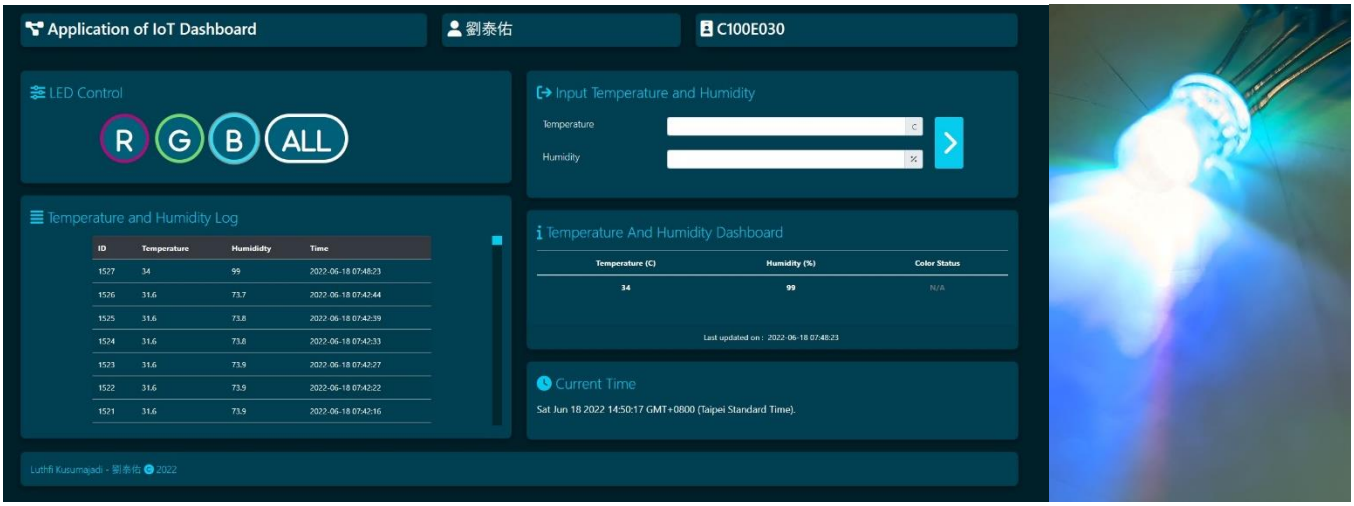

```
MariaDB [C100E030]> select * from LEDCOLOR;
+--------+----------+---------+---------------------+
| RedLED | GreenLED | BlueLED | Time                |
+--------+----------+---------+---------------------+
| 1      | 0        | 0       | 2022-06-18 13:06:44 |
+--------+----------+---------+---------------------+
1 row in set (0.001 sec)
```

Fig. 1.5-4.2



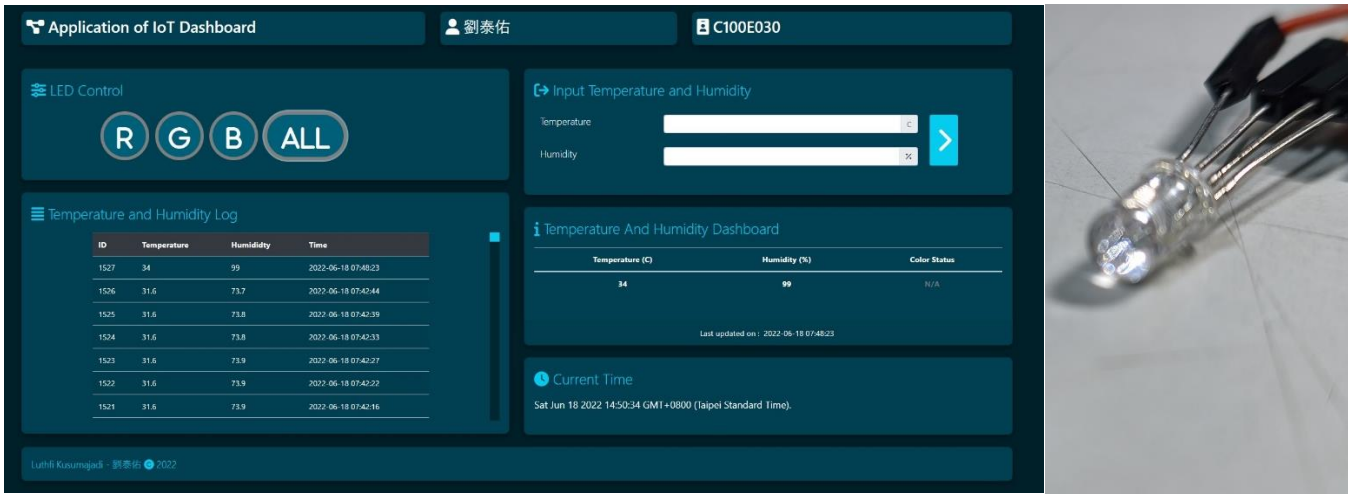

```
MariaDB [C100E030]> select * from LEDCOLOR;
+--------+----------+---------+---------------------+
| RedLED | GreenLED | BlueLED | Time                |
+--------+----------+---------+---------------------+
| 1      | 1        | 0       | 2022-06-18 13:06:44 |
+--------+----------+---------+---------------------+
1 row in set (0.001 sec)
```

Fig. 1.5-4.3




```
MariaDB [C100E030]> select * from LEDCOLOR;
+--------+----------+---------+---------------------+
| RedLED | GreenLED | BlueLED | Time                |
+--------+----------+---------+---------------------+
| 1      | 1        | 1       | 2022-06-18 13:06:44 |
+--------+----------+---------+---------------------+
1 row in set (0.001 sec)
```

Fig. 1.5-4.4




```
MariaDB [C100E030]> select * from LEDCOLOR;
+--------+----------+---------+---------------------+
| RedLED | GreenLED | BlueLED | Time                |
+--------+----------+---------+---------------------+
| 0      | 0        | 0       | 2022-06-18 13:06:44 |
+--------+----------+---------+---------------------+
1 row in set (0.001 sec)
```

Fig. 1.5-4.5

5. The second program is also a success, then we can move to the next project, which is where the esp8266 capture and send the temperature and humidity data into the database, then the website decide what led color is to indicate the temperature. First we need to run the esp8266 program in thonny, then we can continue to monitor from the website.

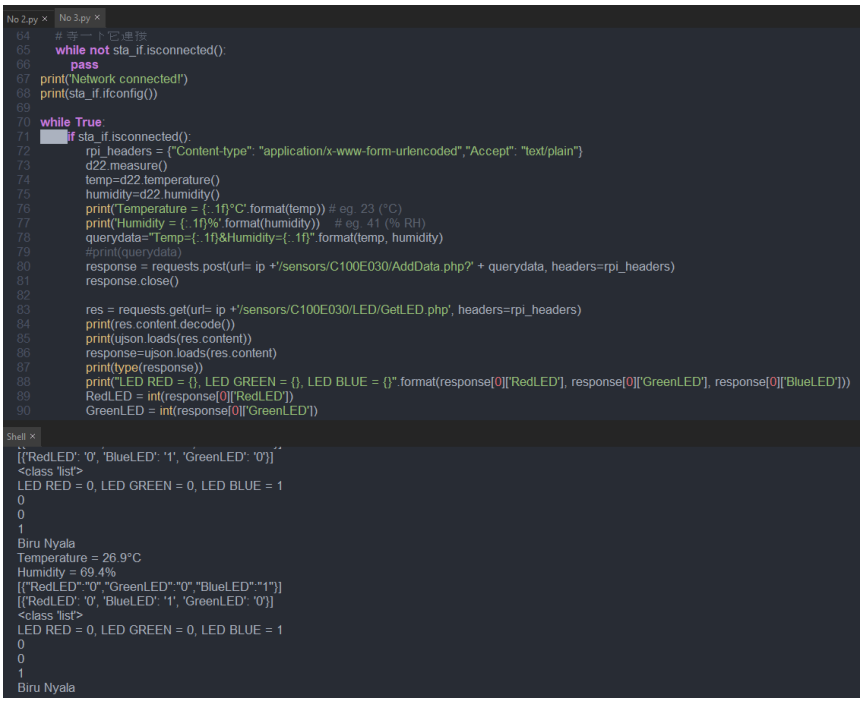


Fig. 1.5-6.1

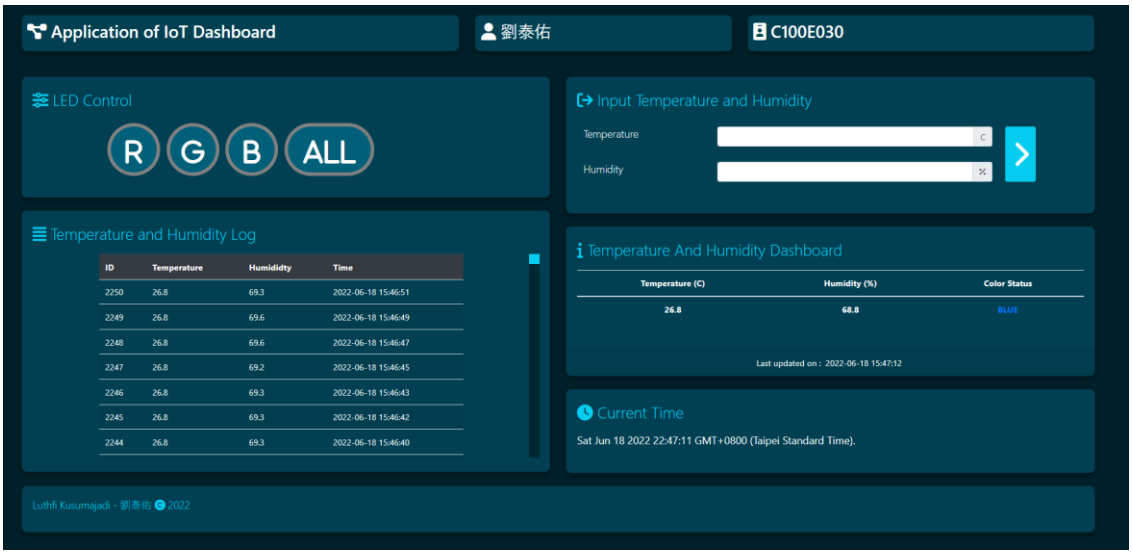


Fig. 1.5-6.2

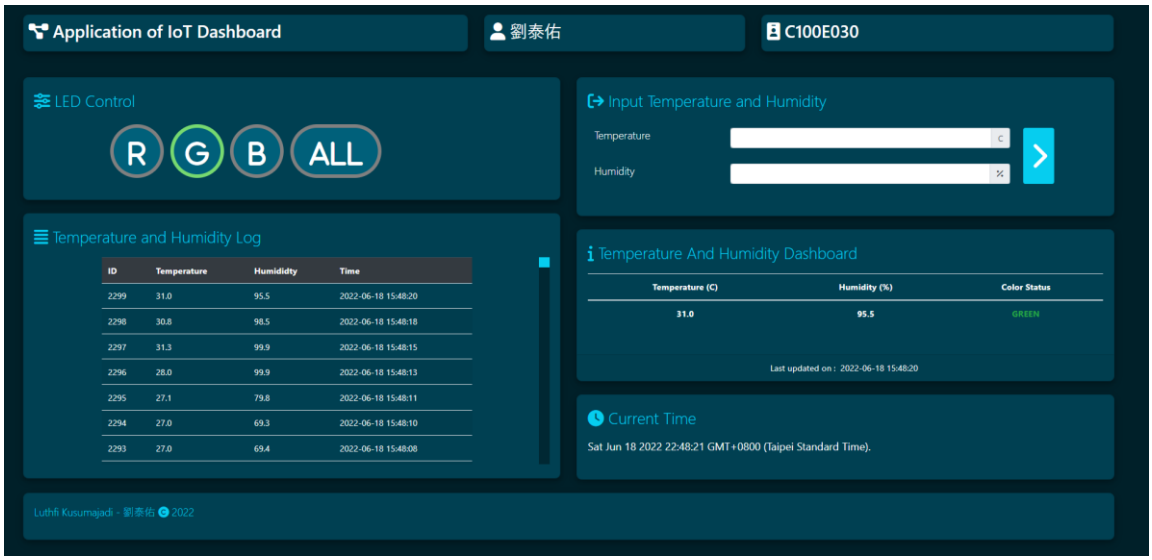


Fig. 1.5-6.3

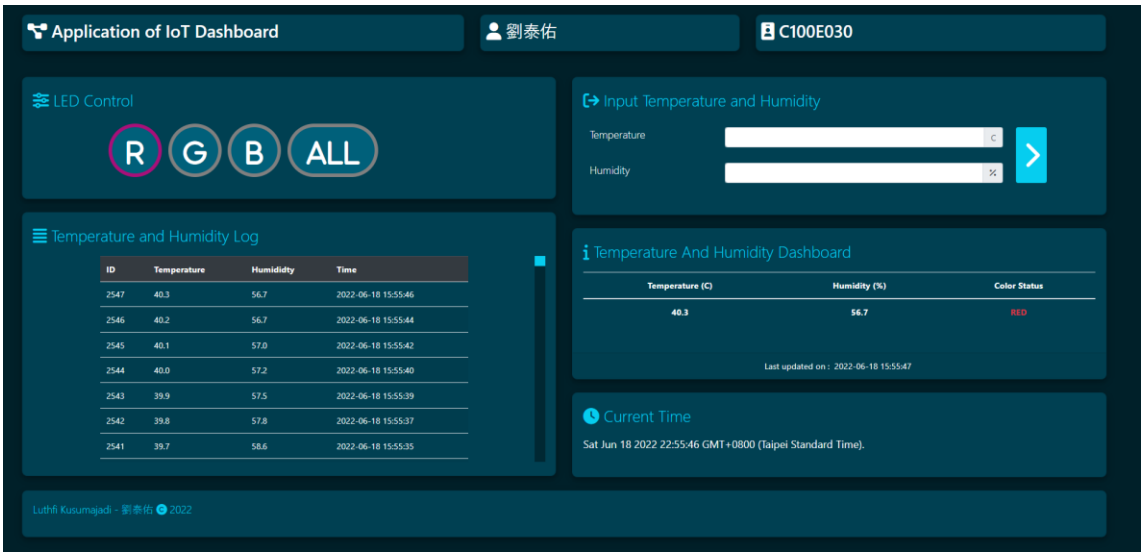


Fig. 1.5-6.4

1.6 Conclusion and Discussion

In conclusion, setting up the raspberry pi as a server is pretty simple, the only programming part is only the part where it gets very tricky since you have to find the program that fits the requirement. But other than that doing this project is a fun experience.