**BÀI 4: Thiết kế giao diện web điều khiển led hiển thị giá trị từ cảm biến,vẽ biểu đồ.**

## 1.Mục tiêu

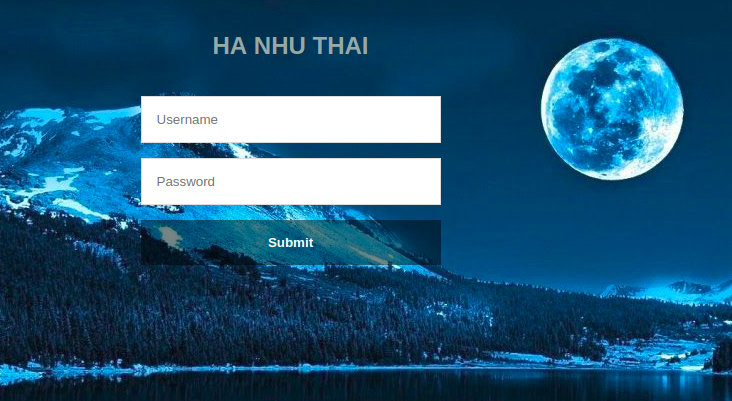
- Sử dụng arduno để đọc dữ liệu cảm biến và gửi qua MQTT.

- Sử dụng nodejs đưa dữ liệu vào database và đưa lên web.

- Sử dụng bootstrap để có giao diện html đẹp hơn.

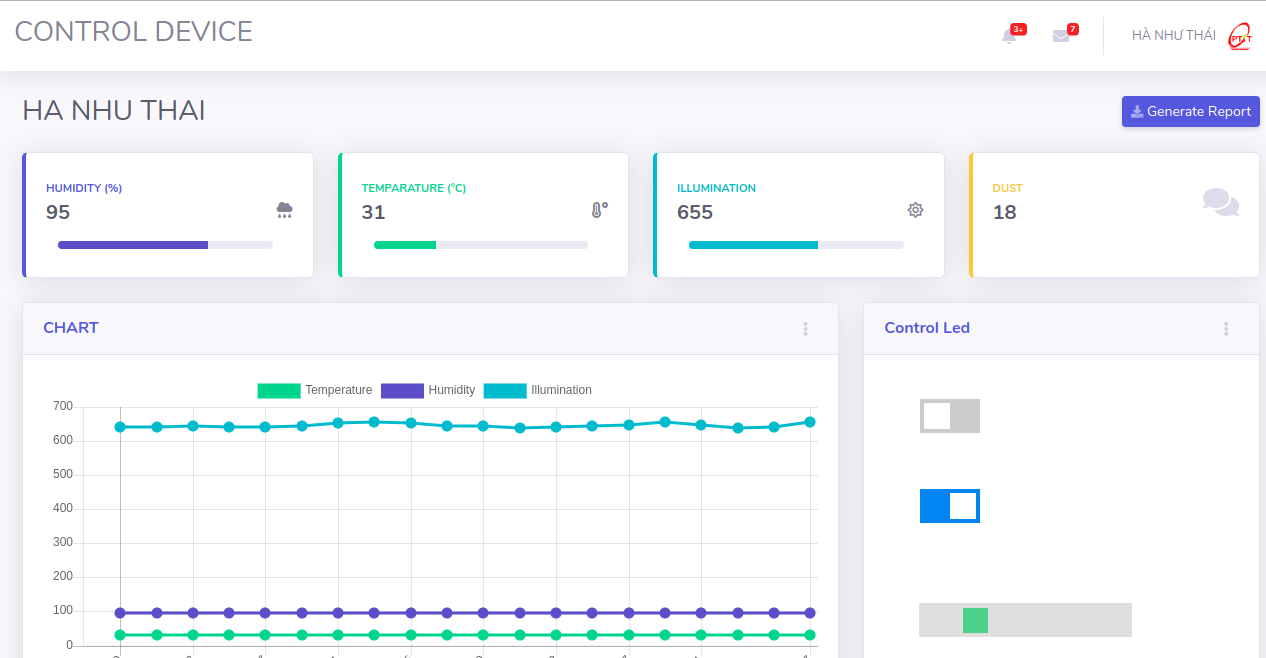
- Tạo được giao diện web với đầy đủ chức năng:

**Login**:



### 

**Home:**



## 2.Nội dung

### 2.1 Arduno:đọc dữ liệu từ cảm biến và gửi giữ liệu bằng MQTT.

-Tạo file mqtt2\_esp.ino để thực hiện code arduno:

**B1:Thêm thư viện,khai báo chân sử dụng:**

(ở đây mình sử dụng esp 8266 nodemcu v3,cảm biến ánh sáng LM393,cảm biến nhiệt độ độ ẩm dht11)

-Thực hiện nối chân : +)D6 với led1.

+)D5 với led2.

+)A0 với chân analog của cảm biến LM393.

+)D4 với chân ao của DHT11.

|  |
| --- |
| 1. #include "DHT.h" 2. #include <ESP8266WiFi.h> 3. #include <PubSubClient.h> 4. #define DHTPIN 2 //d4 5. #define DHTTYPE DHT11 6. DHT dht(DHTPIN, DHTTYPE); 7. #define led1 D6 8. #define led2 D5 9. const char\* ssid = "ThuvienPTIT"; //tên wifi đang kết nối 10. const char\* password = "12345678"; 11. const char\* mqtt\_server = "192.168.0.103"; //ip wifi đang kết nối 12. WiFiClient espClient; 13. PubSubClient client(espClient); 14. int sensor = A0; 15. long lastMsg = 0; 16. char msg[50]; 17. int value = 0; |

-Trong đó ssid,password,mqtt\_server lần lượt là tên wifi,mật khẩu,địa chỉ ip wifi đang kết nối (lấy bằng lệnh ifconfig)

**B2:Thực hiện hàm callback để điều khiển:**

|  |
| --- |
| void callback(char\* topic, byte\* message, unsigned int length) {  Serial.print("Message arrived on topic: ");  Serial.print(topic);  Serial.print(". Message: ");  String messageTemp;    for (int i = 0; i < length; i++) {  Serial.print((char)message[i]);  messageTemp += (char)message[i];  }  Serial.println();  if (String(topic) == "esp32/output1") {  Serial.print("setup led1 ");  if (strcmp(messageTemp.c\_str(),"true")==0)  //strcmp:hàm so sánh chuỗi  digitalWrite(led1,1);  else  digitalWrite(led1,0);  Serial.println(messageTemp);  }  if (String(topic) == "esp32/output2") {  Serial.print("setup led2 ");  if (strcmp(messageTemp.c\_str(),"true")==0)  digitalWrite(led2,1);  else  digitalWrite(led2,0);  Serial.println(messageTemp);  }  if (String(topic) == "esp32/output3") {  Serial.print("setup led1 pwm ");  Serial.println(messageTemp.toInt());  analogWrite(led2,messageTemp.toInt());  }  } |

-Khi có message gửi thực hiện đếm chuỗi.

-c\_str() chuyển string thành char nếu =true thì cho led 1 sáng,ngược lại sẽ ở mức 0(tắt).

**B3:Thực hiện hàm reconnect để gọi nếu kết nối không thành công:**

|  |
| --- |
| void reconnect() {  // Lặp cho đến khi đã kết nối lại  while (!client.connected()) {  Serial.print("Đang cố gắng kết nối mqtt...");  // Attempt to connect  if (client.connect("ESP8266Client","hanhuthai","nhuthai97")) {  Serial.println("connected");  // Subscribe  client.subscribe("esp32/output1");  client.subscribe("esp32/output2");  client.subscribe("esp32/output3");  } else {  Serial.print("failed, rc=");  Serial.print(client.state());  Serial.println(" try again in 5 seconds");  // Wait 5 seconds before retrying  delay(1000);  }  }  } |

**B4:Thực hiện hàm setup**

|  |
| --- |
| void setup() {  Serial.begin(115200);  Serial.println("DHTxx test!");  setup\_wifi();  client.setServer(mqtt\_server, 1883);  client.setCallback(callback);  pinMode(led1,OUTPUT);  pinMode(led2,OUTPUT);  pinMode(sensor,INPUT);  dht.begin();  } |

**B5:Hàm loop:**

|  |
| --- |
| client.loop();  char tempString[8];  sprintf(tempString, "%d",temperature);  Serial.print(" Temperature: ");  Serial.print(tempString);  client.publish("home/sensors/temperature", tempString);    char humString[8];  sprintf(humString, "%d", humidity);  Serial.print(" Humidity: ");  Serial.println(humString);  client.publish("home/sensors/humidity", humString);  char Illumination[8];  sprintf(Illumination, "%d", value);  Serial.print("Illumination: ");  Serial.println(Illumination);  client.publish("home/sensors/illumination", Illumination);    delay(2000);  } |

-Thực hiện lấy giá trị đọc được từ cảm biến và publish giữ liệu qua mqtt sử dụng

3 topic.

**2.2 Nodejs**

**B1:Tạo file index.js,khởi đầu sẽ được cấu hình như sau:**

|  |
| --- |
| var mysql = require('mysql');  var mqtt = require('mqtt');  var express = require('express');  var session = require('express-session');  var bodyParser = require('body-parser');  var path = require('path');  var app = express();  app.set('view engine', 'html');  app.set('views', \_\_dirname )  app.engine('html', require('ejs').renderFile);  app.use(express.static('css')); |

Để subcribe và public dữ liệu ta khai báo các topic:

|  |
| --- |
| var count = 0;  var client = mqtt.connect("mqtt://localhost:1883",{clientId:"mqttjs01",username:"hanhuthai",password:"nhuthai97"});  var topic1 = "esp32/output1";  var topic2 = "esp32/output2";  var topic3 = "esp32/output3";  var message="test message";  var topic\_list=["home/sensors/temperature","home/sensors/humidity","home/sensors/illumination"];  console.log("connected flag " + client.connected); |

**B2:Tạo connection để kết nối với database:**

|  |
| --- |
| var connection = mysql.createConnection({  host : 'localhost',  user : 'root',  password : 'nhuthai97',  database : 'thcsb4'  }); |

-password:mật khẩu của máy để truy cập vào database

-database:lựa chọn database để làm việc trong đó.

**B3:Kết nối với sql và thực hiện tạo bảng:**

|  |
| --- |
| connection.connect(function(err) {  if (err)  throw err;  console.log("mysql connected");  var sql ="DROP TABLE IF EXISTS sensors";  connection.query(sql, function(err, result){  if (err)  throw err;  console.log("drop tables sensors ok");  });  sql = "CREATE TABLE sensors( id INT(10) PRIMARY KEY auto\_increment , Sensor\_ID varchar(10) not null, Date\_and\_Time datetime not null, Temperature int(3) not null,Humidity int(3) not null,Illumination int(3) not null)"  connection.query(sql, function(err, result){  if (err)  throw err;  console.log("create tables sensors ok");  });  }); |

-Thực hiện drop bảng cũ và tạo bảng sensors mới nếu kết nối thành công.

-Câu lệnh tạo bảng sql đầy đủ:

sql = "CREATE TABLE sensors( id INT(10) PRIMARY KEY auto\_increment , Sensor\_ID varchar(10) not null, Date\_and\_Time datetime not null, Temperature int(3) not null,Humidity int(3) not null,Illumination int(3) not null)".

**B4:Thực hiện chức năng đăng nhập:**

-Khởi tạo sever:

|  |
| --- |
| var server = app.listen(3000, () => { //Start the server, listening on port 4000.  console.log("Listening to requests on port 4000...");  }) |
|  |

-Hàm kiểm tra username,password:

|  |
| --- |
| app.post('/auth', function(request, response) {  var username = request.body.username;  var password = request.body.password;  if (username && password) {  connection.query('SELECT \* FROM accounts WHERE username = ? AND password = ?', [username, password], function(error, results, fields) {  if (results.length > 0) {  request.session.loggedin = true;  request.session.username = username;  response.redirect('/home');  } else {  response.send('Incorrect Username and/or Password!');  }  response.end();  });  } else {  response.send('Please enter Username and Password!');  response.end();  }  }); |

-Để link tới trang đăng nhập ta thực hiện:

|  |
| --- |
| app.get('/', function(request, response) {  response.sendFile(path.join(\_\_dirname + '/login.html'));  }); |

Ta sẽ không cho link luôn vào trang home nếu chưa đăng nhập,vì vậy ta cần kiểm tra đã đăng nhập chưa bằng cách sử dụng session để lưu tạm thời:

|  |
| --- |
| app.get('/home', (request, response) => {  if (request.session.loggedin) {  connection.query("SELECT \* FROM sensors", function (err, result, fields) {  if (err) throw err;  response.sendFile(path.join(\_\_dirname + '/index.html'));    });  }else {  response.send('Please login to view this page!!');  response.end();  }  }) |

Nếu chưa login ta sẽ dẫn đến 1 đoạn text:please login to view this page.

**B5:Chức năng điều khiển thiết bị:**

|  |
| --- |
| io.on('connection', (socket) => {  console.log("Someone connected."); //show a log as a new client connects.  var today = new Date();  connection.query("SELECT \* FROM sensors", function (err, result, fields) {  if (err) throw err;  result.forEach(function(value) {  var m\_time = value.Date\_and\_Time.toString().slice(4,24);  console.log(m\_time);  io.sockets.emit('temp', {date: today.getDate()+"-"+today.getMonth()+1+"-"+today.getFullYear(), time:m\_time , temp:value.Temperature,hum:value.Humidity,illu:value.Illumination});  });    });  socket.on('led\_status1',(data) =>{  console.log("led status change: led1 %d",data.led);  publish(topic1,data.led.toString(),options);  });  socket.on('led\_status2',(data) =>{  console.log("led status change: led2 %d",data.led);  publish(topic2,data.led.toString(),options);  });  socket.on('led\_pwm',(data) =>{  console.log("led pwm change: led1 %d ",data.led1);  publish(topic3,data.led1.toString(),options);  });  }) |

**B6:Lưu giá trị cảm biến vào csdl:**

**-**Trước tiên ta cần kiểm tra đã gửi đầy đủ chưa và lưu giữ liệu vào các biến:

-Lưu vào Temp nếu nó là topic\_list[0],tương tự với hum,dust.

|  |
| --- |
| var Temp = 12;  var Hum = 120;  var Illumination = 123;  var cnt\_check = 0;  client.on('message',function(topic, message, packet){  console.log("message is "+ message);  console.log("topic is "+ topic);  if( topic == topic\_list[0]){  cnt\_check ++;  Temp = message;  }  else if( topic == topic\_list[1]){  cnt\_check ++;  Hum = message;  }  else if( topic == topic\_list[2]){  cnt\_check ++;  Illumination = message;  }  if( cnt\_check == 3 ){  cnt\_check = 0;  console.log(Temp,Hum,Illumination);  console.log("ready to save");  var first\_name = "DHT-11";  var Date\_and\_Time = new Date().toMysqlFormat();  let query = "INSERT INTO `sensors` (Sensor\_ID,Date\_and\_Time,Temperature,Humidity,Illumination) VALUES ('" +  first\_name + "', '" + Date\_and\_Time + "', '" + Temp + "', '" + Hum + "', '"+ Illumination + "')";  connection.query(query, (err, result) => {  if (err) {  throw err;  }  });  var today = new Date();  io.sockets.emit(first\_name, {date: today.getDate()+"-"+today.getMonth()+1+"-"+today.getFullYear(), time:Date\_and\_Time , temp:Temp,hum:Hum});    connection.query("SELECT \* FROM sensors ORDER BY id DESC LIMIT 1", function (err, result, fields) {  if (err) throw err;  result.forEach(function(value) {  var m\_time = value.Date\_and\_Time.toString().slice(4,24);  console.log('temp', {date: today.getDate()+"-"+today.getMonth()+1+"-"+today.getFullYear(), time:m\_time , temp:value.Temperature,hum:value.Humidity,illu:value.Illumination});  io.sockets.emit('temp', {date: today.getDate()+"-"+today.getMonth()+1+"-"+today.getFullYear(), time:m\_time , temp:value.Temperature,hum:value.Humidity,illu:value.Illumination});  });    });  }  }); |

-Nếu btncheck =3 nghĩa là đã đủ giữ liệu ta thực hiện lưu vào csdl:

**2.3 Xây dựng trang Login (html)**

**B1:** Vào database đang sử dụng,tạo table accounts với các thuộc tính:

|  |
| --- |
|  |

**B2** : Insert dữ liệu vào bảng,username và password chính là tài khoản và mật khẩu khi bạn đăng nhập trên web.

**B3 :**-Tạo file login.html đặt ngang hàng với file index.js

-Sử dụng phương thức POST để gửi dữ liệu có tính bảo mật như password.

-html bắt buộc phải có để login có thể gửi giữ liệu:

|  |
| --- |
| <form action="auth" method="POST">  <input type="text" name="username" placeholder="Username" required>  <input type="password" name="password" placeholder="Password" required>  <input type="submit">  </form> |

-Bạn có thể thêm css hoặc giao diện bootstrap có sẵn để trông đẹp hơn(sử dụng từ khóa “template login free”).

**2.4 Home (html)**

**a)Đổ dữ liệu temp,hum,illu lên web.**

-Đây là giao diện người dùng,đuôi html.

**B1:Tạo file index.html ngang hàng với file index.js**

-Bạn có thể sự dụng template có sẵn bằng cách search “**template control device**”

Link một số template đẹp free mình đã chọn lọc:

<https://uicookies.com/free-responsive-bootstrap-admin-templates/>

Cách sử dụng:Trong gói template đã tải về bản sẽ sử dụng file index.html có sẵn và mình chỉ chỉnh sửa nhưng tuyệt đối không thay đối cấu trúc thư mục của họ (sẽ gây ra vỡ vì mất css).

**B2:Thêm 3 thẻ văn bản html này vào chỗ bạn muốn hiển thị nhiệt độ,độ ẩm,ánh sáng.**

|  |
| --- |
| <div id="hum" class="rectangle">Hum  <center><p id='humidity'>0</p></center>  </div>  <div id="temp" class="rectangle">Temp  <center><p id='temperature'>0</p></center>  </div>  <div id="illu" class="rectangle">Illu  <center><p id='illumination'>0</p></center>  </div> |

**B3:Thêm đoạn script sau để đọc dữ liệu từ server và hiển thị lên vùng đã tạo.**

-Trước tiên để đổ được dữ liệu lên html ta cần kết nối với server bằng cách thêm dòng script sau:

var socket = io.connect('[http://localhost:3000](http://localhost:3000/)');

3000:là chỉ số port cần trùng với

var server = app.listen(3000, () bên server

-Để đổ dữ liệu,ta thêm đoạn script:

|  |
| --- |
| socket.on('temp', function(data) { //As a temp data is received  console.log(data.temp);  // nhiet do , do am , anh sang  var my\_data = [data.temp , data.hum, data.illu ];  console.log(my\_data);  var index = 0;  document.getElementById('temperature').innerHTML = data.temp; //update the date  document.getElementById('humidity').innerHTML = data.hum; //update the date  document.getElementById('illumination').innerHTML = data.illu; //update the date |

**b)Vẽ đồ thị:**

**B1:Khai báo thư viện:**

|  |
| --- |
| <link href="https://fonts.googleapis.com/css?family=Lato" rel="stylesheet">  <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.4.0/Chart.min.js">  </script>  <script src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/2.0.4/socket.io.js">  </script> |

**B2:Tạo vùng mà bạn muốn hiển thị trong html:**

|  |
| --- |
| <div class="chart-container" style="position: relative; width:60%;margin:auto; ">  <canvas id="myChart"></canvas>  </div> |

**B3:Viết đoạn script để đọc và vẽ đồ thị:**

|  |
| --- |
| <script>  var socket = io.connect('http://localhost:4000');  //connect to server  var ctx = document.getElementById('myChart').getContext('2d');  var chart = new Chart(ctx, {  // The type of chart we want to create  type: 'line',  // The data for our dataset  data: {  labels: [],  datasets: [{  label: "Temperature",  borderColor: "blue",  data: [],  fill: false,  pointStyle: 'circle',  backgroundColor: 'blue',  pointRadius: 5,  pointHoverRadius: 7,  lineTension: 0,  },  {  label: "Humidity",  borderColor: "#FF0000",  data: [],  fill: false,  pointStyle: 'circle',  backgroundColor: '#FF0000',  pointRadius: 5,  pointHoverRadius: 7,  lineTension: 0,  },  {  label: "Illumination",  borderColor: "green",  data: [],  fill: false,  pointStyle: 'circle',  backgroundColor: 'green',  pointRadius: 5,  pointHoverRadius: 7,  lineTension: 0,  }  ]  },  // Configuration options go here  options: {}    }); |

-Vẽ nhiệt độ,có thể thay đồi màu border,màu nền…

-Tương tự ta vẽ được độ ẩm và ánh sáng:

-Đổ dữ liệu vào:

|  |
| --- |
| socket.on('temp', function(data) { //As a temp data is received  console.log(data.temp);  // nhiet do , do am , anh sang  var my\_data = [data.temp , data.hum, data.illu ];  console.log(my\_data);  var index = 0;  document.getElementById('temperature').innerHTML = data.temp;  //update the date  document.getElementById('humidity').innerHTML = data.hum;  document.getElementById('illumination').innerHTML = data.illu;  if(chart.data.labels.length != 20) {  chart.data.labels.push(data.time); //Add time in x-asix  chart.data.datasets.forEach((dataset) => {  dataset.data.push(my\_data[index]); //Add temp in y-axis  console.log(index);  index ++;  });  }  else { //If there are already 15 data points in the graph.  chart.data.labels.shift(); //Remove first time data  chart.data.labels.push(data.time); //Insert latest time data  chart.data.datasets.forEach((dataset) => {  dataset.data.shift(); //Remove first temp data  dataset.data.push(my\_data[index]); //Insert latest temp data  index ++;  });  }  chart.update(); //Update the graph.  }); |

**c) Điều khiển thiết bị**

**B1:****Tạo 2 button và 1 range sliders trong html:**

|  |
| --- |
| <label class="switch">  <input id='led1' type="checkbox" onclick="calc\_led1()">  <span class="slider"></span>    </label>  <label class="switch" >  <input id='led2' type="checkbox" onclick="calc\_led2()">  <span class="slider"></span>  </label>    <div class="slidecontainer">  LED1 PWM  <input type="range" min="0" max="255" value="50" class="slider\_s" id="myRange" onchange="updateSlider(this.value)">  </div> |

**B2:Đọc sự kiện button và kiểm tra đã check.**

|  |
| --- |
| function calc\_led1()  {  socket.emit("led\_status1",{led:document.getElementById('led1').checked});  console.log("led1\_status change");  }  function calc\_led2() {  socket.emit("led\_status2",{led:document.getElementById('led2').checked});  console.log("led2\_status change");  }  function updateSlider(slideAmount) {  socket.emit("led\_pwm",{led1:slideAmount});  }  </script> |

-Và cuối cùng:truy cập tới địa chỉ: <http://localhost:4000> thực hiện login và thu được kết quả.

|  |
| --- |
| *Designed by HaNhuThai* |