

Senior project documentation

IOT system

By Fatemeh Rahimi

Under supervision of : Dr.Ali hamze

Contents

Project description	3
Data base	4
Software	4
Webserver	4
Sign up	4
Sign in	5
Edit system information	6
Add new data	7
Retrieve temp	8
Retrieve humidity	9
Retrieve data	10
api.php	11
Website	14
Back-end	14
Front-end	19
Hardware	23

Project description

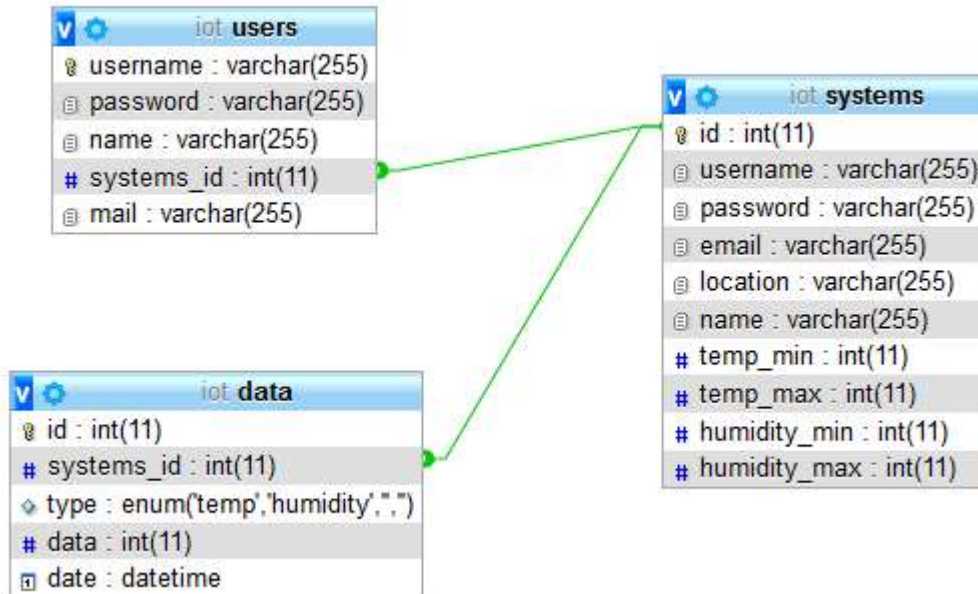
This project is designed to give a hand to people who work in the kitchen such as, people working in the kitchens of grand hotels, where the matter of temperature and heat turns out to be essential. They have giant refrigerators, freezers, huge kitchen, and a couple of storage to be taken care of.

To make sure that each section is being in an appropriate situation and the systems are functioning correctly, this project is designed to monitor a system to prevent any damage that might happen to these workers.

In this system, there exists a temperature and humidity sensor, the information which is receiving from the environment will be sent out through a Wi-Fi module to a web server, the web server will store the data. The user can access theses information from a website. There is also the feature of setting temperature of humidity limit so that the website can give the user some information about the history of the status and also email the user if any change accrued that went below the limit.

Data base

Here is the ERD design of this system:



Software

Webserver

Webserver of this project is developed using Laravel PHP framework.

There were 2 controllers in this project, in this directory:

senior project\webserver\iot\app\Http\Controllers

<< webserver > iot > app > Http > Controllers				Search Controllers	
Name	Date modified	Type	Size		
Auth	2018-02-27 2:29 AM	File folder			
Controller.php	2018-02-07 12:15 ...	PHP File	1 KB		
DataController.php	2018-03-04 1:54 AM	PHP File	7 KB		
SystemController.php	2018-02-16 4:12 PM	PHP File	10 KB		

System controller contains following functions:

Sign up

Method Name:	Signup
--------------	--------

Type:	POST
Address:	/api/v1/signup
Parameters:	username(required, unique) password(required) email(required,should be valid email,unique) name(optional, maximum 30 chars)
Response if params validation fails: Response status code:400	<pre>{ "success": "false", "messages": { Error messages } }</pre> <p>Sample:</p> <pre>{ "success": "false", "messages": { "username": ["The username field is required."], "password": ["The password field is required."], "email": ["The email field is required."] } }</pre>
Success response: Response status code:200	<pre>{ "success": "true", "message": "Signup successful", "token": token }</pre>

Sign in

Method Name:	Signin
Type:	POST
Address:	/api/v1/signin
Parameters:	username(required) password(required)

Response if params validation fails: Response status code:400	<pre>{ "success": "false", "messages": { Error messages } }</pre> <p>Sample:</p> <pre>{ "success": "false", "messages": { "username": ["The username field is required."], "password": ["The password field is required."] } }</pre>
Success response: Response status code:200	<pre>{ "success": "true", "message": "Login successful", "token": token, "system_info": { "id": id, "username": username , "email": email, "location": location, "name": name, "temp_min": minimum temperature, "temp_max": maximum temperature, "humidity_min": minimum humidity, "humidity_max": maximum humidity } }</pre>

Edit system information

Method Name:	edit_system_information
Type:	POST
Address:	/api/v1/editSystemInformation
Parameters:	token(required) name(optional) email(required, should be valid email) min_temp(required) max_temp(required)

	min_humidity(required) max_humidity(required)
Response if params validation fails: Response status code:400	<pre>{ "success": "false", "messages": { Error messages } }</pre> <p>Sample:</p> <pre>{ "success": "false", "messages": { "token": ["The token field is required."], "email": ["The email field is required."], "min_temp": ["The min temp field is required."], "max_temp": ["The max temp field is required."], "min_humidity": ["The min humidity field is required."], "max_humidity": ["The max humidity field is required."] } }</pre>
Success response: Response status code:200	<pre>{ "success": "true", "message": "system info updated" }</pre>

Data controller contains following function:

[Add new data](#)

Method Name:	new_data
Type:	POST
Address:	/api/v1/ new_data

Parameters:	token(required) data(required) type(required)
Response if params validation fails: Response status code:400	<pre>{ "success": "false", "messages": { Error messages } }</pre> <p>Sample:</p> <pre>{ "success": "false", "messages": { "token": ["The token field is required."], "type": ["The type field is required."], "data": ["The data field is required."] } }</pre>
Success response: Response status code:200	<pre>{ "success": "true", "message": "adding data was successful." }</pre>

Retrieve temp

Method Name:	retrieve_temp
Type:	POST
Address:	/api/v1/ retrieve_temp
Parameters:	token(required)
Response if params validation fails: Response status code:400	<pre>{ "success": "false", "messages": { Error messages } }</pre> <p>Sample:</p> <pre>{</pre>

	<pre> "success": "false", "messages": { "token": ["The token field is required."] } </pre>
<p>Success response:</p> <p>Response status code:200</p>	<pre> { "success": "true", "temps": [{ "id": 1, "systems_id": 4, "type": "temp", "data": "23", "date": "2018-02-16 17:46:16" }, { "id": 60, "systems_id": 4, "type": "temp", "data": "27.70", "date": "2018-02-27 00:17:19" }, ] } </pre>

Retrieve humidity

Method Name:	retrieve_humidity
Type:	POST
Address:	/api/v1/ retrieve_humidity
Parameters:	token(required)
<p>Response if params validation fails:</p> <p>Response status code:400</p>	<pre> { "success": "false", "messages": { Error messages } } </pre> <p>Sample:</p> <pre> { "success": "false", "messages": { "token": [</pre>

	<pre> "The token field is required."] } } </pre>
Success response: Response status code:200	<pre> { "success": "true", "humidity": [{ "id": 3, "systems_id": 4, "type": "humidity", "data": "45", "date": "2018-02-16 17:46:16" }, { "id": 4, "systems_id": 4, "type": "humidity", "data": "23", "date": "2018-02-16 17:46:16" }, ...] } </pre>

Retrieve data

Method Name:	retrieve_data
Type:	POST
Address:	/api/v1/ retrieve_data
Parameters:	token(required)
Response if params validation fails: Response status code:400	<pre> { "success": "false", "messages": { Error messages } } </pre> <p>Sample:</p> <pre> { "success": "false", "messages": { "token": ["The token field is required."] } } </pre>

	<pre>] } }</pre>
Success response: Response status code:200	<pre> { "success": "true", "data": [{ "id": 1, "systems_id": 4, "type": "temp", "data": "23", "date": "2018-02-16 17:46:16" }, { "id": 3, "systems_id": 4, "type": "humidity", "data": "45", "date": "2018-02-16 17:46:16" }, { "id": 4, "systems_id": 4, "type": "humidity", "data": "23", "date": "2018-02-16 17:46:16" }, ...] }</pre>

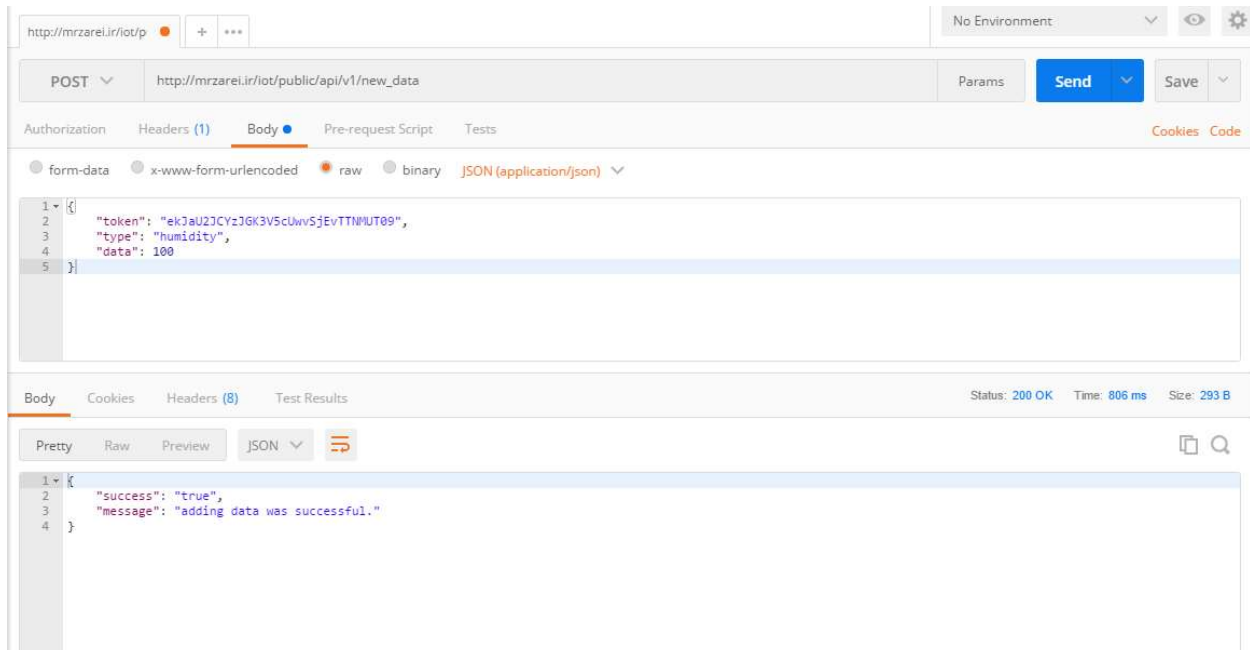
[api.php](#)

this file which is located at in routes/**api.php** contains set of mapping routes between the method we call and the actual method name that has been used in the code itself. Here is the api.php for this project.

```
api.php x
14 */
15
16 Route::middleware('auth:api')->get('/user', function (Request $request) {
17     return $request->user();
18 });
19
20 Route::group(['prefix'=>'v1'], function(){
21     Route::post('signup', 'SystemController@signup');
22     Route::post('signin', 'SystemController@signin');
23     Route::post('setMinTemp', 'SystemController@set_min_temp');
24     Route::post('setMaxTemp', 'SystemController@set_max_temp');
25     Route::post('setMaxHumidity', 'SystemController@set_max_humidity');
26     Route::post('setMinHumidity', 'SystemController@set_min_humidity');
27     Route::post('retrieve_temp', 'DataController@retrieve_temp');
28     Route::post('retrieve_humidity', 'DataController@retrieve_humidity');
29     Route::post('retrieve_data', 'DataController@retrieve_data');
30     Route::post('new_data', 'DataController@new_data');
31     Route::post('editSystemInformation', 'SystemController@edit_system_information');
32 });
```

This webserver is now on an actual server with the domain name of: <http://mrzareei.ir/> you can send your requests with postman to check these functions using Postman¹.

Here is an example of testing the new_data function.



¹ **Postman** is a powerful HTTP client for testing web services. Created by Abhinav Asthana, a programmer and designer based in Bangalore, India, **Postman** makes it easy to test, develop and document APIs by allowing users to quickly put together both simple and complex HTTP requests.[1]

Headers of this request was set like this:



Now that everything is tested and is working alright on the webserver, we start coding the website.

- Min and max of humidity and temperature

As it was mentioned in the project description this system can have some features about controlling it.

Four limits can be set in this project:

- Minimum temperature
- Maximum temperature
- Minimum humidity
- Maximum humidity

Each time a new data is added, it is checked whether it is in the given range or not. If it was out of range the system will automatically email the user about this change. Here is the code of this feature:

```
47     if ($data->save()) {
48         if($type == "temp"){//-----check-min-max-temp-----
49             if($request->data < $temp_min || $temp_max < $request->data){
50                 //email the System admin
51
52                 $data = 'ERROR: Temperature out of range!';
53                 $email = Mail::send('emails.mail', ['data' => $data], function($message) use($email){
54                     $message->to($email)->subject('WARNING!');
55                 });
56             }
57         }
58         if($type == "humidity"){//-----check-min-max-humidy-----
59             if($request->data < $humidity_min || $humidity_max < $request->data){
60                 //email the System admin
61
62                 $data = 'ERROR: Humidity out of range!';
63                 $email = Mail::send('emails.mail', ['data' => $data], function($message) use($email){
64                     $message->to($email)->subject('WARNING!');
65                 });
66             }
67         }
68         return response()->json(['success'=>'true','message'=>'adding data was successful.'],200);
69     }
70     else{
71         $response = ['success' => 'false',
72                     'message' => 'adding data was unsuccessfull.Please try again.'];
73         return response()->json($response, 500);
74     }
75 }
```


An email is created for this purpose: iot.system.mail@gmail.com

Here is the email sample that this system sends:

WARNING!

Inbox x



 **IOT system** <iot.system.mail@gmail.com>
to me ▾

Feb 27 (11 days ago) ☆



 This message may not have been sent by: iot.system.mail@gmail.com [Learn more](#) [Report phishing](#)

ERROR: Temperature out of range!

Website

The website of this System is developed using CodeIgniter PHP framework.

For connecting with the webserver, guzzle² was installed on CodeIgniter. Guzzle was installed and used with the instruction that were given in the Lynda course which is titled CONSUMING RESTFUL APIS IN PHP WITH GUZZLE.

```
$ mkdir restful
$ cd restful
$ composer require guzzlehttp/guzzle:~6.0
./composer.json has been created
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 4 installs, 0 updates, 0 removals
  - Installing guzzlehttp/promises (v1.3.1) Downloading: 100%
  - Installing psr/http-message (1.0.1) Downloading: 100%
  - Installing guzzlehttp/psr7 (1.3.1) Downloading: 100%
  - Installing guzzlehttp/guzzle (6.2.2) Downloading: 100%
Writing lock file
Generating autoload files
$ █
```

This is how to install this library. This photo is from the tutorial that I just mentioned.

Back-end

The controller of CodeIgniter is located at \application\controllers.

Here are this system controllers:

- Login
- Dashboard
- EditInfo

In this project I finished the back-end and front-end with just html, after everything worked fine I worked on front-end to make it look better.

In Login Controller a form validation exists to control the inputs of this form, and give the proper error if anything went wrong.

² **Guzzle** is a PHP HTTP client that makes it easy to send HTTP requests and trivial to integrate with web services. ... Middleware system allows you to augment and compose client behavior.[2]

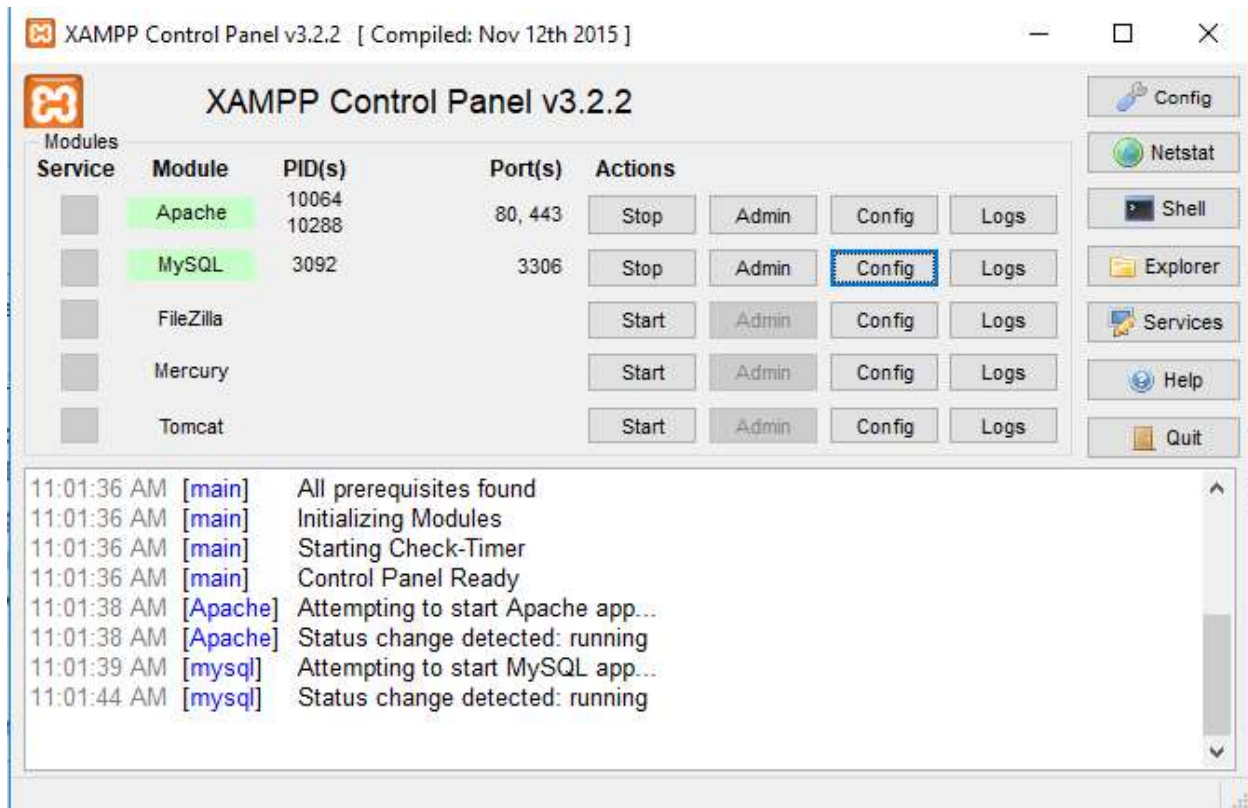
```

Login.php x
1  <?php
2  defined('BASEPATH') OR exit('No direct script access allowed');
3  require 'vendor/autoload.php';
4  use GuzzleHttp\Client;
5
6  class Login extends CI_Controller {
7
8      public function __construct()
9      {
10         parent::__construct();
11         $this->load->library('session');
12         $this->load->helper(array('form', 'url'));
13         $this->load->library('form_validation');
14         // Your own constructor code
15     }
16     public function index()
17     {
18         // url for this function is: http://localhost/iot_website/index.php/login/
19         $this->form_validation->set_rules('username', 'نام کاربری', 'trim|required' ,
20             array('required' => 'لطفا نام کاربری خود را وارد کنید.'));
21         $this->form_validation->set_rules('password', 'رمز عبور', 'trim|required',
22             array('required' => 'لطفا رمز عبور خود را وارد کنید.'));
23
24
25         if ($this->form_validation->run() == FALSE) //age ghalat bod
26         {
27             $this->form_validation->set_error_delimiters('<div dir="rtl" class="alert-box error"><span>e
28             $this->load->view('login_view');
29         }
30         else //age doros bod
31         {
32             $data["username"] = $this->input->post("username", 'true');
33             // echo 'hi:'. $data["username"];

```

The website can work on localhost and It works when Xamp³ is working:

³ XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes.[3]



The project would work if it is placed in the htdocs directory of xamp.

Here how I requested to login to server in this controller:


```

Login.php x
30     else //age doros bod
31     {
32         $data["username"] = $this->input->post("username", 'true');
33         $data["password"] = $this->input->post("password", 'true');
34         $client = new \GuzzleHttp\Client([
35             //Base URI is used with relative requests
36             'base_uri' => 'http://mrzareei.ir/iot/public/api/v1/'
37         ]);
38         //define what the client will be doing
39         try {
40             $response = $client->request(
41                 'POST',
42                 'signin',
43                 [
44                     'json' => [
45                         'username' => $data["username"],
46                         'password' => $data["password"],
47                     ]
48                 ]
49             );
50             $json = (string) $response->getBody();
51             $array = json_decode($json, true);
52             $user_data = array(
53                 'username' => $data["username"],
54                 'token' => $array['token'],
55                 'email' => $array['system_info']['email'],
56                 'name' => $array['system_info']['name'],
57                 'min_temp' => $array['system_info']['temp_min'],
58                 'max_temp' => $array['system_info']['temp_max'],
59                 'min_humidity' => $array['system_info']['humidity_min'],
60                 'max_humidity' => $array['system_info']['humidity_max'],
61             );
62             $this->session->set_userdata($user_data);
63             redirect('Dashboard', 'refresh');
64         }
65         //catch exception
66         catch(Exception $e) {
67             echo 'Message: ' . $e->getMessage();
68             $wrong["login_wrong"] = "Username or password was incorrect!";
69             $this->load->view('login_view', $wrong);
70         }
71
72         // if correct username and password: 200
73         //if any thing goes wrong: 400
74
75     }

```

As it was explained earlier I used guzzle library for making request to the API. Username and password that was received from the input is being send to the webserver. And if it responds, a session is created. And then the user is redirecting to Dashboard controller.

Here is dashboard controller's functions:

```

Dashboard.php x
1  |<?php
2  |defined('BASEPATH') OR exit('No direct script access allowed');
3  |require 'vendor/autoload.php';
4  |use GuzzleHttp\Client;
5  |
6  |class Dashboard extends CI_Controller {
7  |    public function __construct(){
8  |        parent::__construct();
9  |        $this->load->library('session');
10 |        $this->load->helper(array('form', 'url'));
11 |        $this->load->library('form_validation');
12 |        // Your own constructor code
13 |    }
14 |    public function index(){...
52 |    }
53 |    public function temerature_chart(){...
88 |    }
89 |    public function temerature_table(){...
124 |    }
125 |    public function humidity_chart(){...
161 |    }
162 |    public function humidity_table(){...
197 |    }
198 |    public function logout(){...
202 |    }
203 |}

```

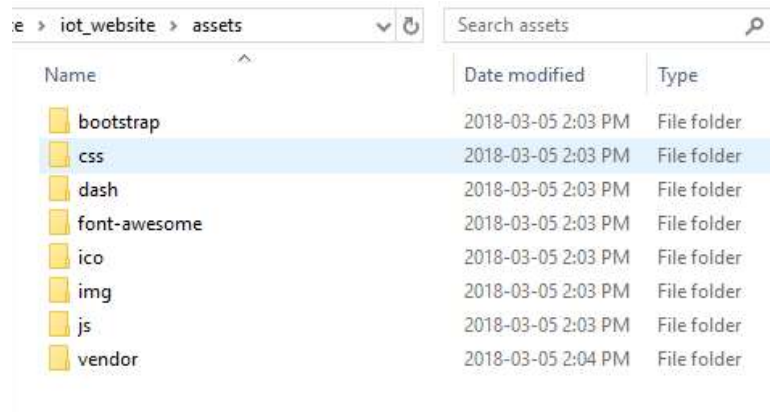
- Index function
this function shows the system information and it is also a notification center. If any temperature of humidity were out of the range that system was limited to. A notification is shown in this section. This data is shown in dashboard_view.
- temerature_chart
This function shows the temperatures of this system in a chart form.
- temerature_table
This function shows the temperature of this system in a table form.
- humidity_chart
This function shows the humidity of this system in a chart form.
- humidity_table
This function shows the humidity of this system in a humidity table.
- Logout
This function clears the user session and redirect the user to Login controller.

Front-end

The front-end of this system is mainly designed with bootstrap⁴. login and dashboard template was directly downloaded from the internet from the [5],[6] resources.

I changed these templates to use in my own project.

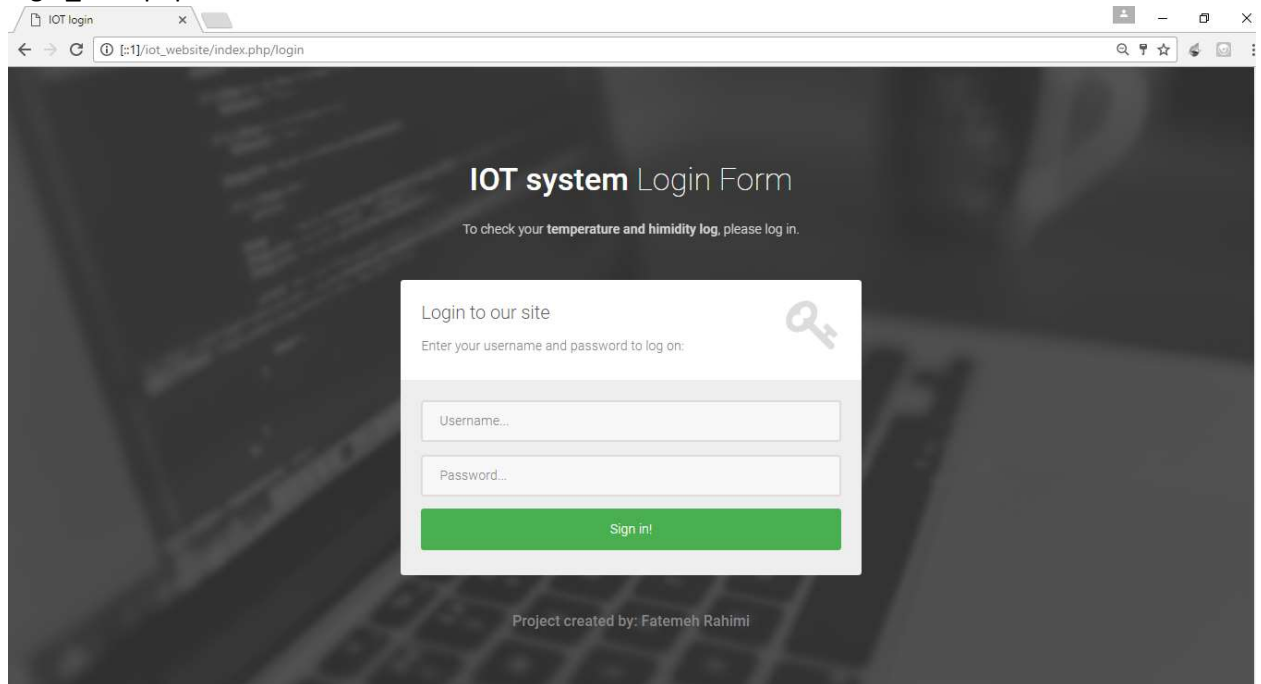
Every front end contain CSS, JavaScript, jQuery, ... files. These files of the project is located at: /assets



Name	Date modified	Type
bootstrap	2018-03-05 2:03 PM	File folder
css	2018-03-05 2:03 PM	File folder
dash	2018-03-05 2:03 PM	File folder
font-awesome	2018-03-05 2:03 PM	File folder
ico	2018-03-05 2:03 PM	File folder
img	2018-03-05 2:03 PM	File folder
js	2018-03-05 2:03 PM	File folder
vendor	2018-03-05 2:04 PM	File folder

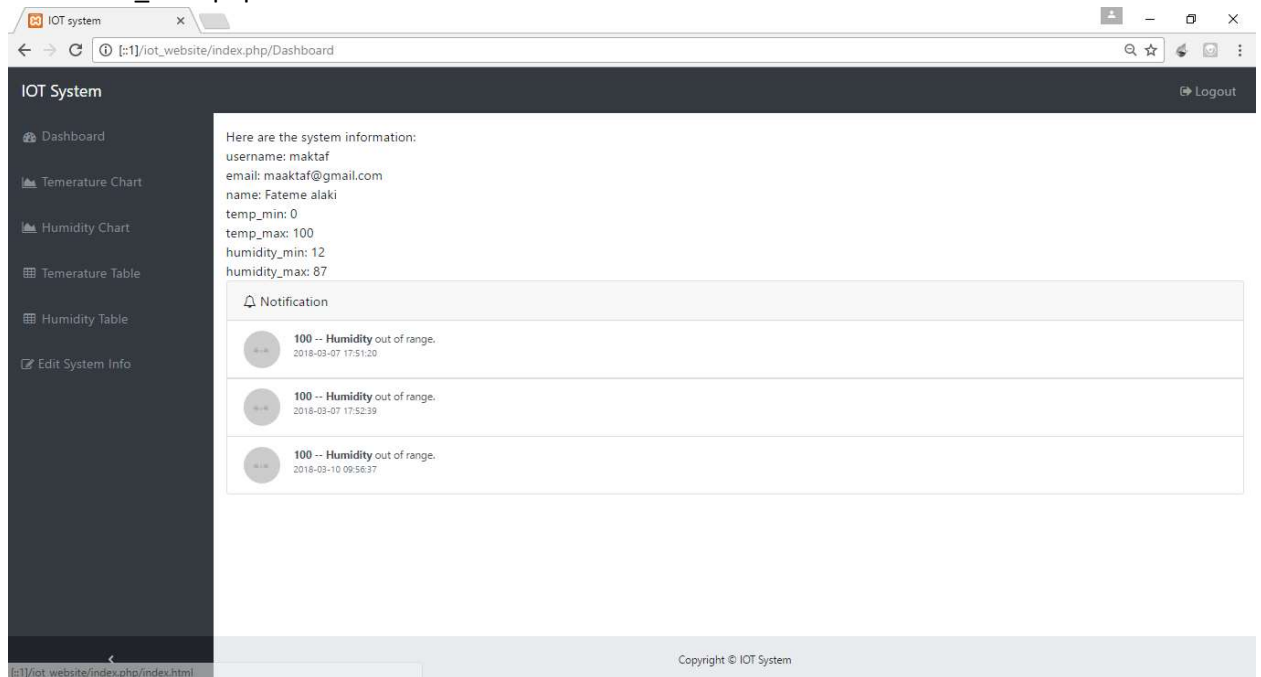
This project includes following views:

❖ login_view.php

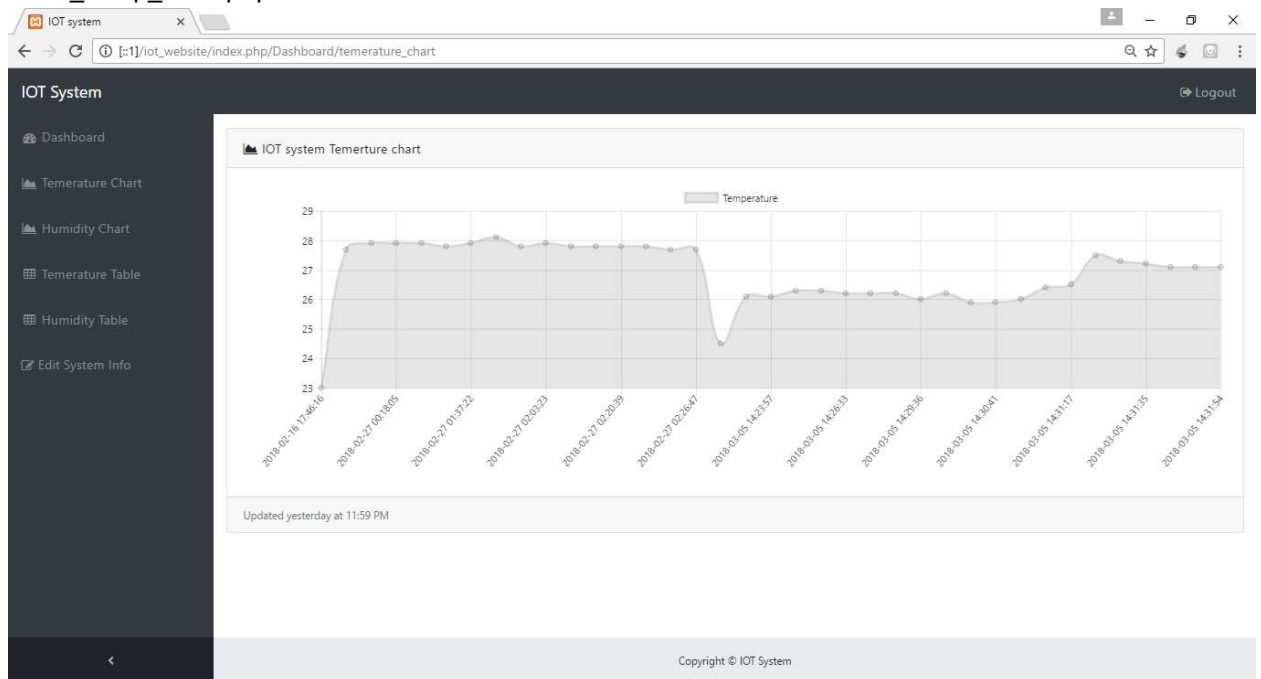


⁴ Bootstrap is a free and open-source front-end library for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other ...[4]

❖ dashboard_view.php



❖ chart_temp_view.php



❖ table_temp.php

IOT system

Dashboard

Temperature Chart

Humidity Chart

Temperature Table

Humidity Table

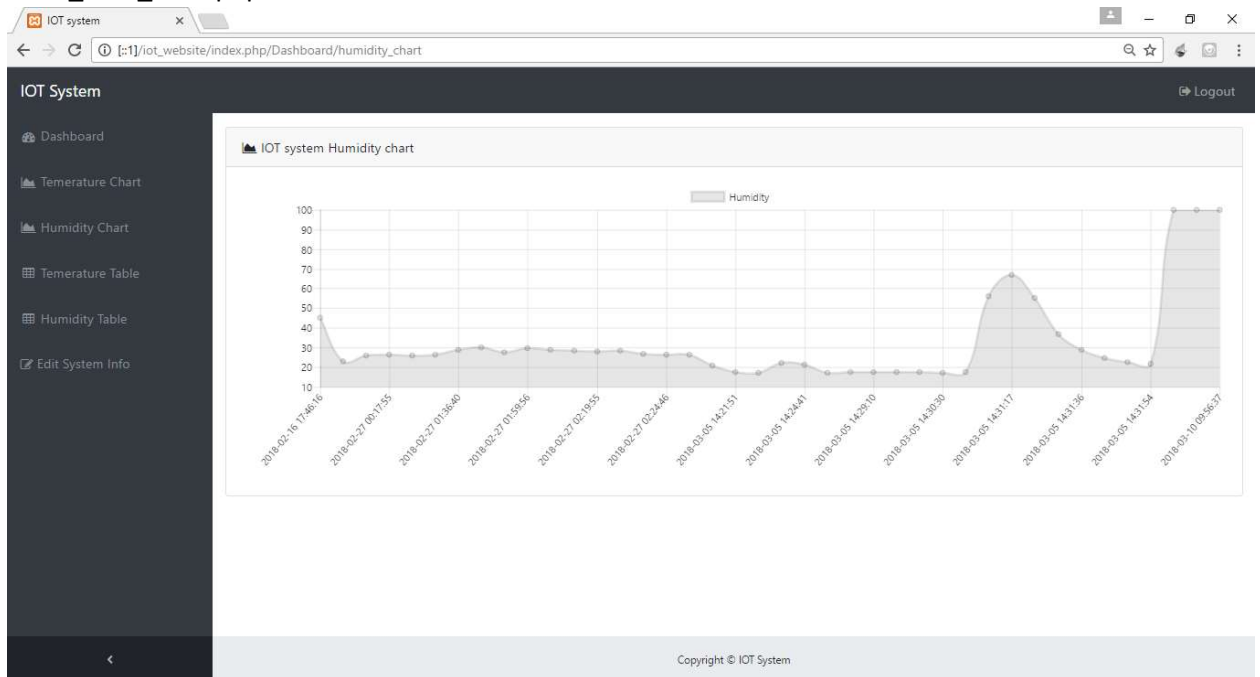
Edit System Info

Logout

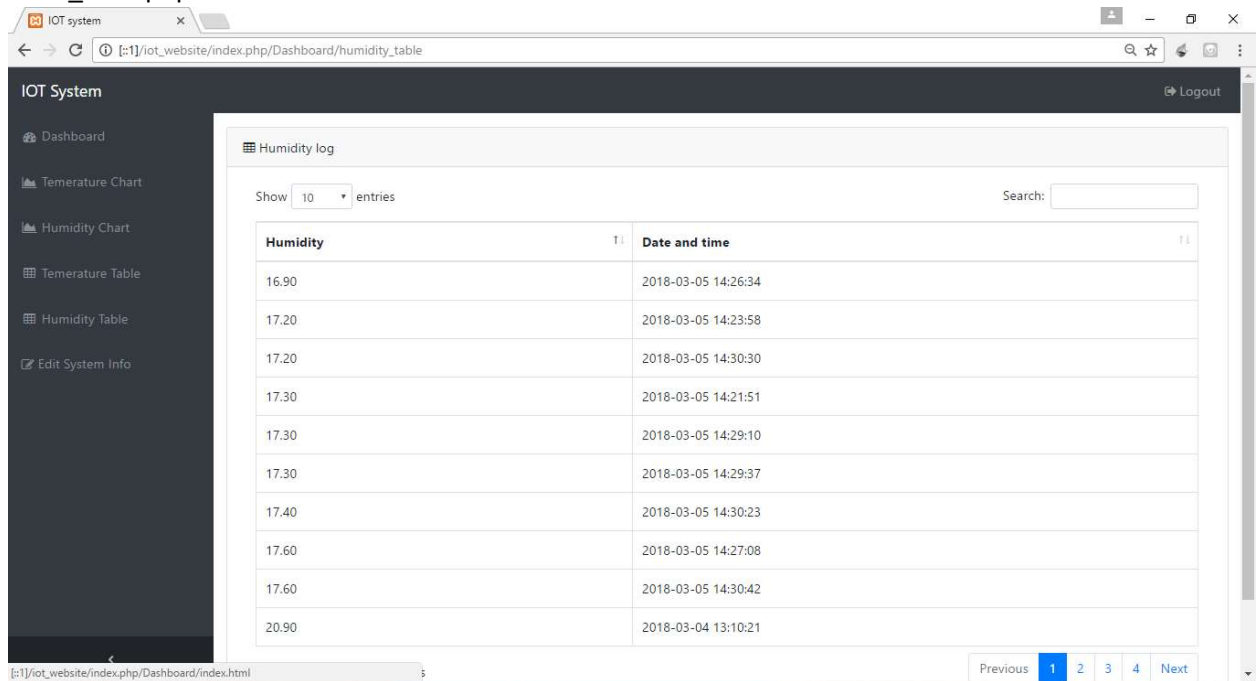
Temperature log

Temperature	Date and time
23	2018-02-16 17:46:16
27.70	2018-02-27 00:17:19
27.90	2018-02-27 00:17:55
27.90	2018-02-27 00:18:05
27.90	2018-02-27 00:18:49
27.80	2018-02-27 01:36:40
27.90	2018-02-27 01:37:22
28.10	2018-02-27 01:53:31
27.80	2018-02-27 01:59:55
27.90	2018-02-27 02:03:23
27.80	2018-02-27 02:14:04
27.80	2018-02-27 02:19:55

❖ chart_hum_view.php



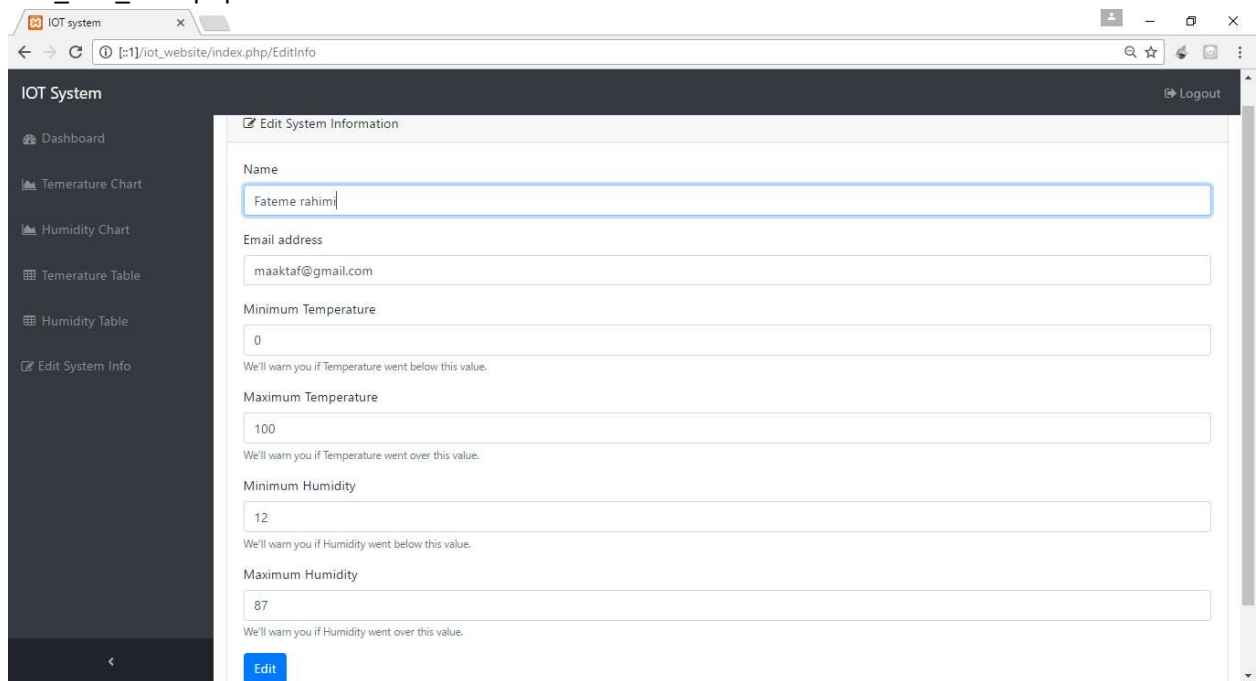
❖ table_hum.php



The screenshot shows the 'Humidity log' section of the IOT System dashboard. It features a table with two columns: 'Humidity' and 'Date and time'. The table displays 10 entries, with a search bar and pagination controls at the bottom.

Humidity	Date and time
16.90	2018-03-05 14:26:34
17.20	2018-03-05 14:23:58
17.20	2018-03-05 14:30:30
17.30	2018-03-05 14:21:51
17.30	2018-03-05 14:29:10
17.30	2018-03-05 14:29:37
17.40	2018-03-05 14:30:23
17.60	2018-03-05 14:27:08
17.60	2018-03-05 14:30:42
20.90	2018-03-04 13:10:21

❖ edit_info_view.php



The screenshot shows the 'Edit System Information' form in the IOT System dashboard. The form contains fields for Name, Email address, Minimum Temperature, Maximum Temperature, Minimum Humidity, and Maximum Humidity. Each temperature and humidity field has a warning message below it: 'We'll warn you if Temperature went below this value.' and 'We'll warn you if Temperature went over this value.' respectively.

Name
Fateme rahim

Email address
maaktat@gmail.com

Minimum Temperature
0
We'll warn you if Temperature went below this value.

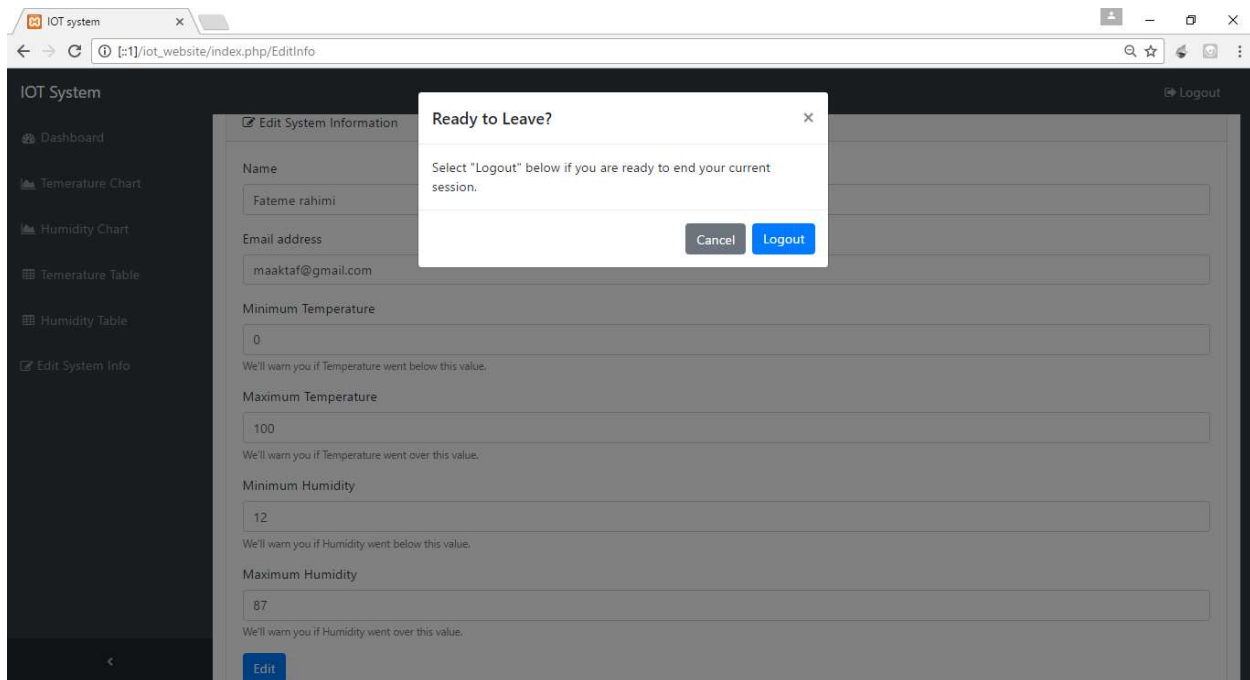
Maximum Temperature
100
We'll warn you if Temperature went over this value.

Minimum Humidity
12
We'll warn you if Humidity went below this value.

Maximum Humidity
87
We'll warn you if Humidity went over this value.

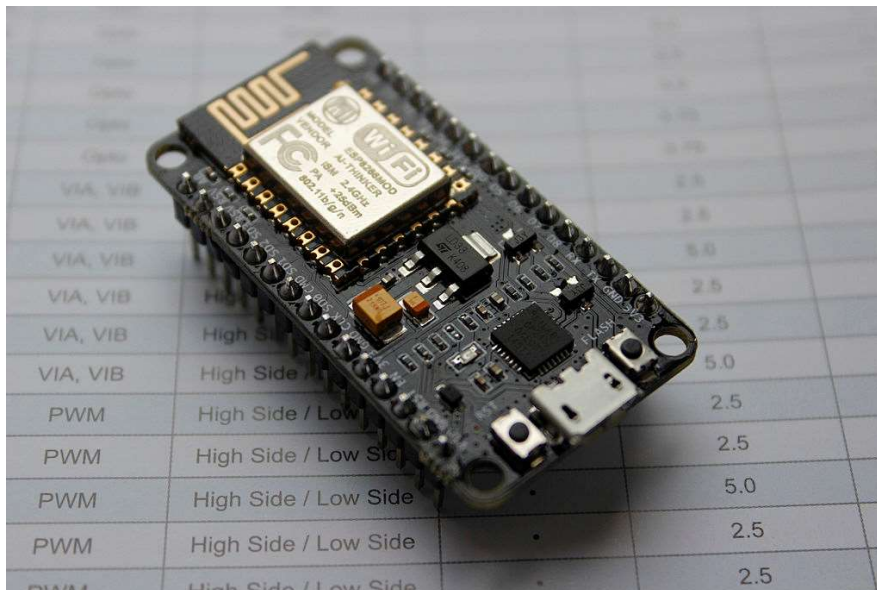
Edit

Logging out did not have its own view, but in every page of the dashboard in the right corner in the upper side, user have this ability to logout, and after clicking this will be shown to the user:



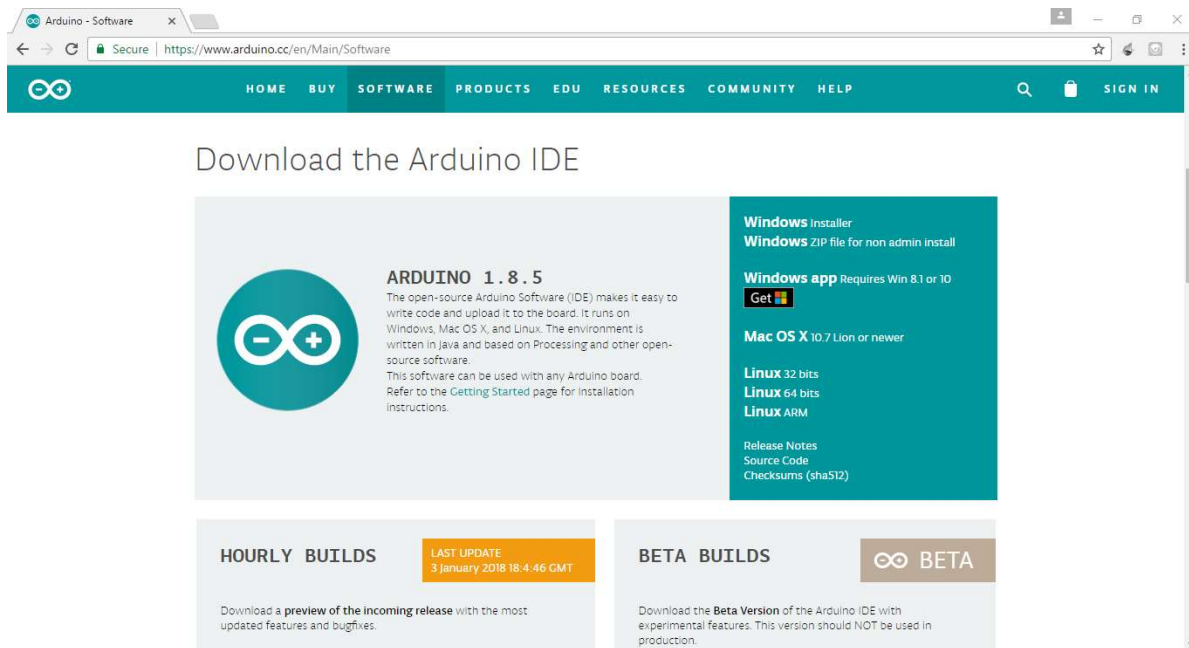
Hardware

The hardware of this system is nodemcu⁵. Here is the image[7] of this board:



⁵ **NodeMCU** is an open source IoT platform. It includes firmware which runs on the **ESP8266** Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.[7]

To code this board we first need to have the arduino IDE.



The screenshot shows the Arduino Software download page in a web browser. The browser's address bar displays "https://www.arduino.cc/en/Main/Software". The page features a teal navigation bar with links for HOME, BUY, SOFTWARE, PRODUCTS, EDU, RESOURCES, COMMUNITY, and HELP. The main heading is "Download the Arduino IDE". Below this, a large section for "ARDUINO 1.8.5" includes a description of the IDE and a list of download links for Windows (Installer, ZIP file, app), Mac OS X, and Linux (32 bits, 64 bits, ARM). At the bottom, there are sections for "HOURLY BUILDS" and "BETA BUILDS".

ARDUINO 1.8.5

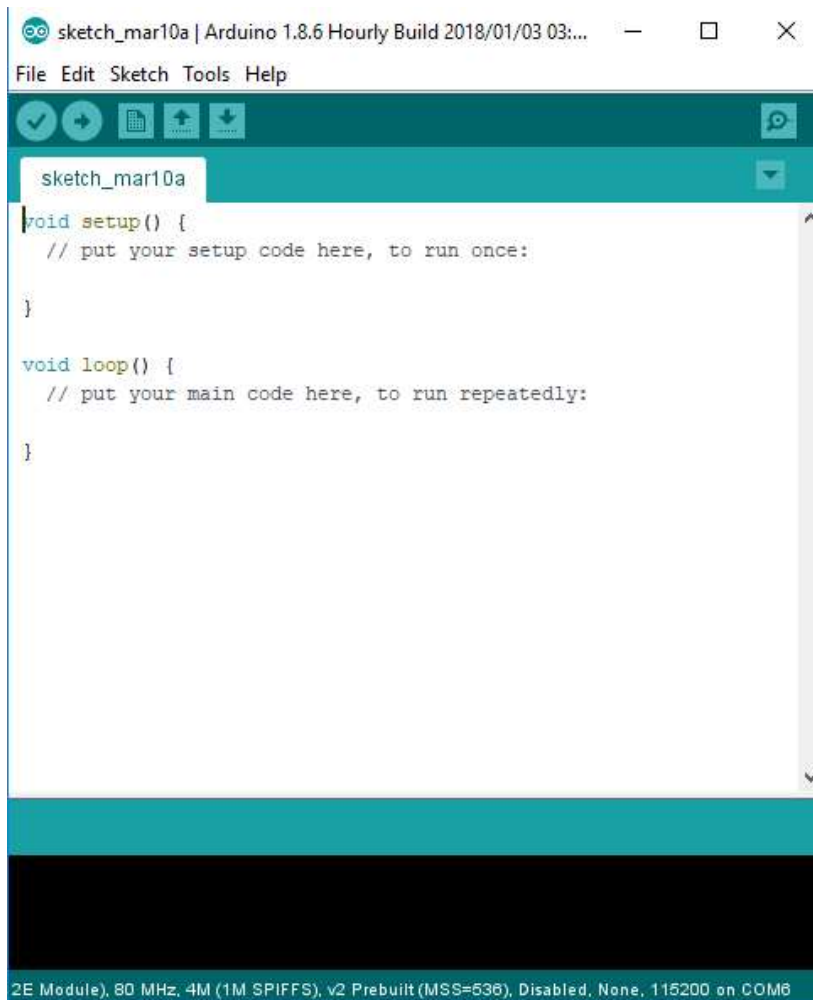
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. Refer to the [Getting Started](#) page for installation instructions.

Windows Installer
Windows ZIP file for non admin install
Windows app Requires Win 8.1 or 10
Get
Mac OS X 10.7 Lion or newer
Linux 32 bits
Linux 64 bits
Linux ARM
[Release Notes](#)
[Source Code](#)
[Checksums \(sha512\)](#)

HOURLY BUILDS **LAST UPDATE** 3 January 2018 18:46 GMT
Download a **preview of the incoming release** with the most updated features and bugfixes.

BETA BUILDS **BETA**
Download the **Beta Version** of the Arduino IDE with experimental features. This version should NOT be used in production.

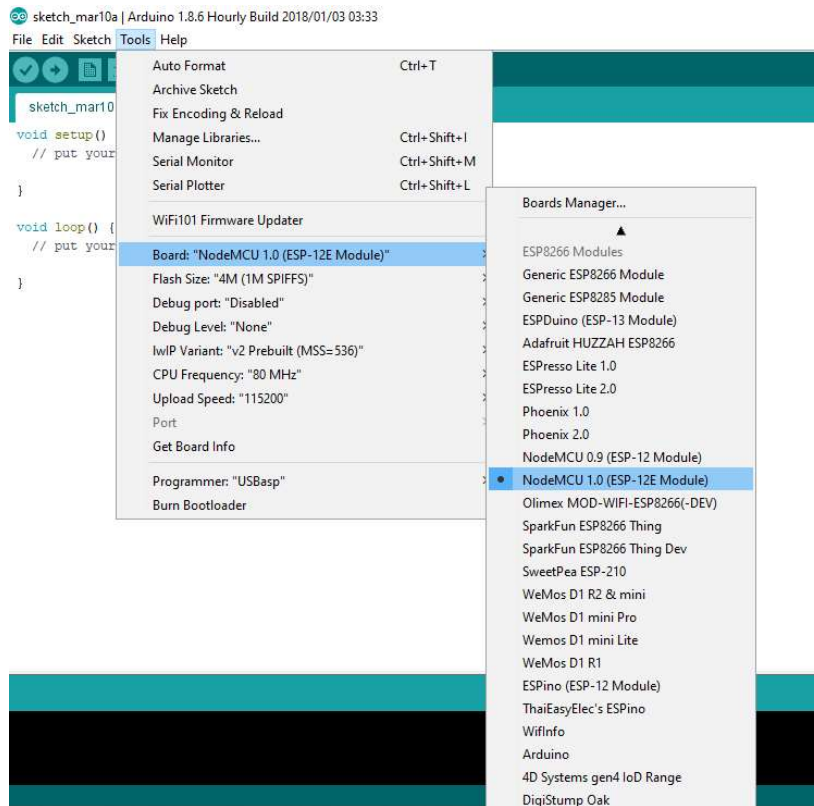
Here is how this IDE looks like when you open it:



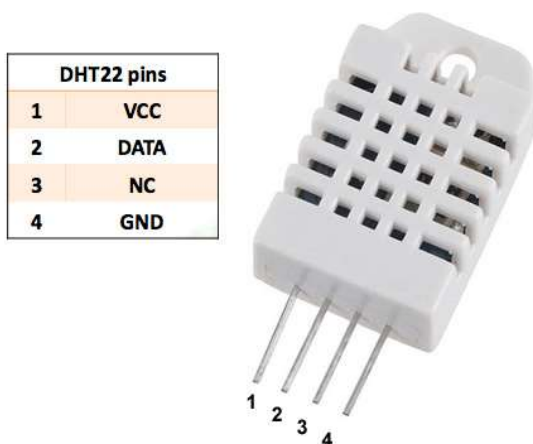
When we first download this IDE nodemcu is not in the board lists to choose. So we need to install it ourselves.

This is the github link to Arduino core for ESP8266 WiFi chip: <https://github.com/esp8266/Arduino>

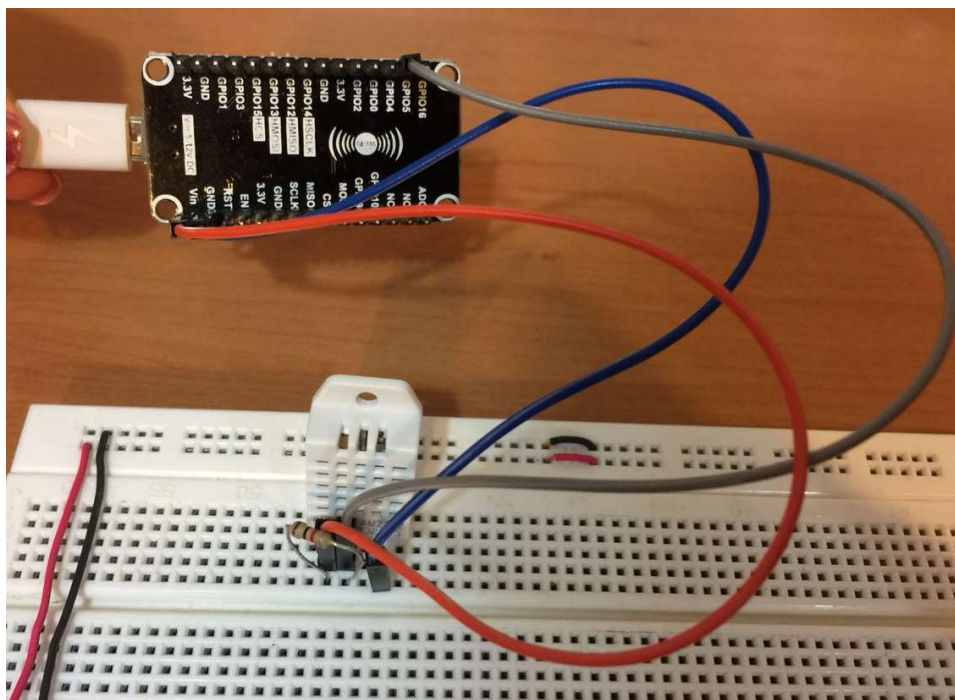
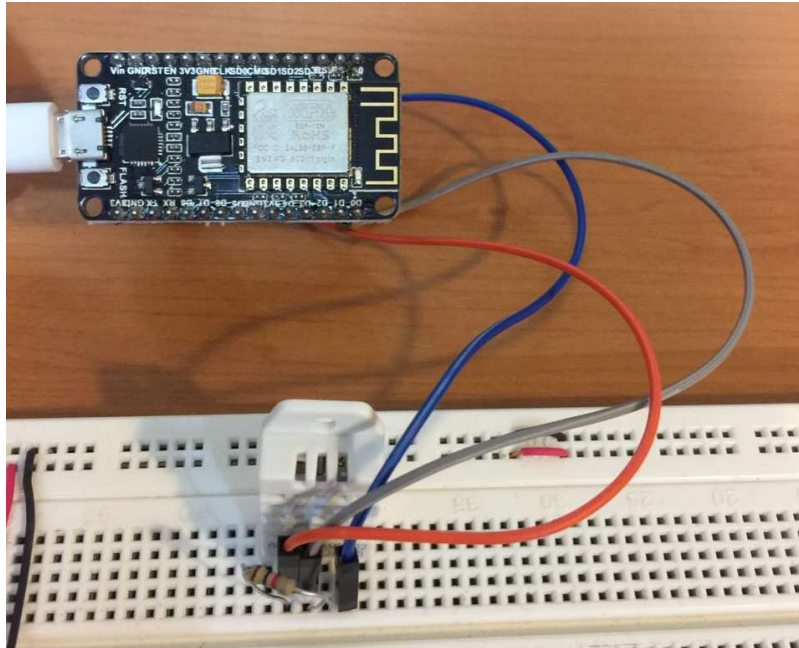
With following the instruction that was given in this link esp8266 was installed. Now we can choose this board from tools>board.



The sensor that we use in this project is DHT22. Here is a photo[8] of it:



With the use of a breadboard, we can design our hardware. Here is the photo of this IOT system:



Orange wire connects the vcc of DHT22 to the vcc of nodemcu.

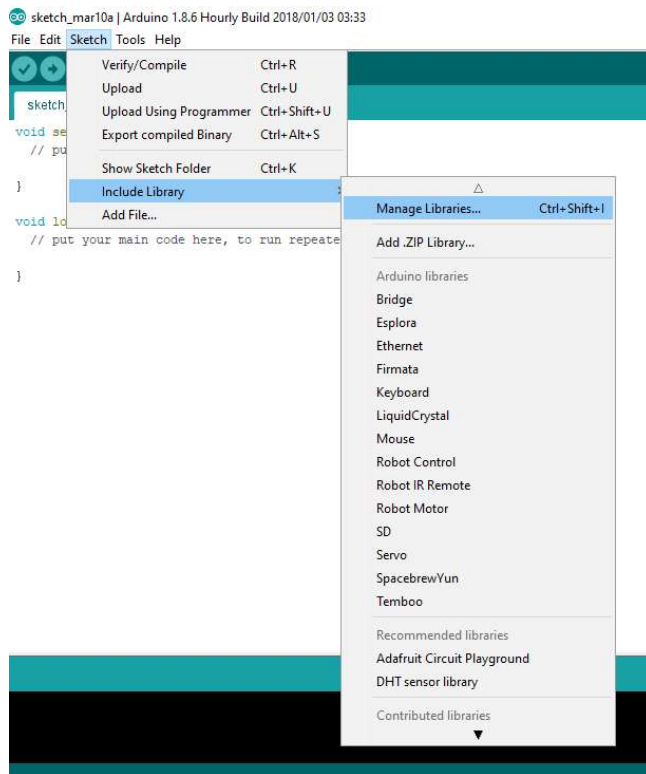
Blue wire connects the ground of DHT22 to the ground of nodemcu.

Grey wire connects the data of DHT22 to to the GPIO 01 of nodemcu.

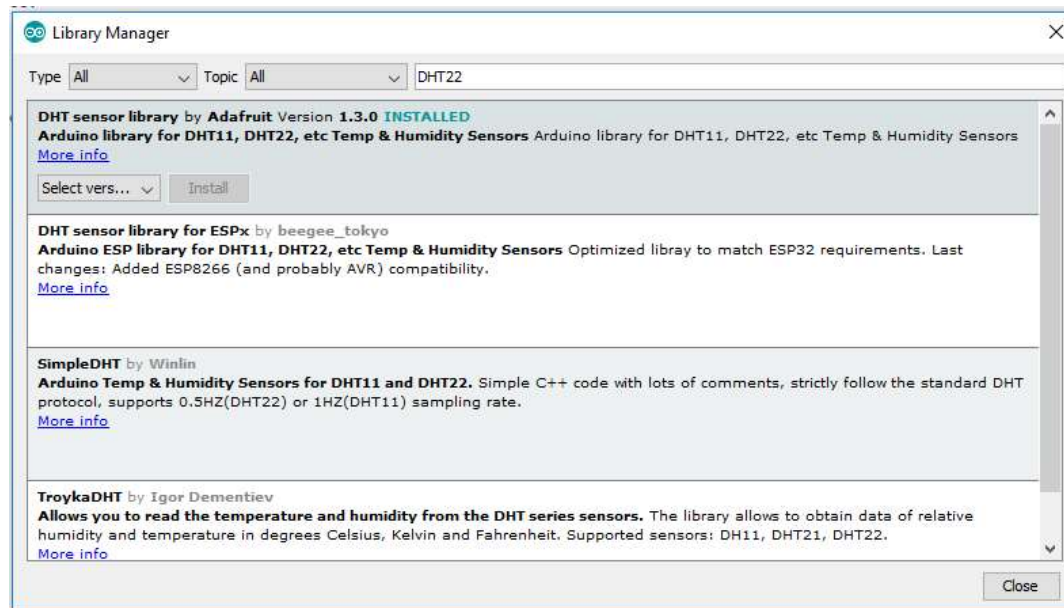
10k resistor is connecting the data and VCC of DHT22.

In order to read the data from DHT22 we need to add its library to the Arduino IDE.

In Arduino IDE> sketch > include library > manage library



This window open:



Search for DHT22, and as you can see I have already installed it.

In the setup part of the code:

```

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);           //Serial connection
  WiFi.begin(ssid, password);    //WiFi connection
  |
  while (WiFi.status() != WL_CONNECTED) { //Wait for the WiFi connection completion
    delay(500);
    Serial.println("Waiting for connection");
  }
  Serial.println("WiFi connected");
  dht.begin();
}

```

We create to a WiFi connection and connect to a WiFi that has internet.

In the loop part, we read data from the DHT and send it to the webserver.

```

void loop() {
  if(WiFi.status()== WL_CONNECTED){ //Check WiFi connection status
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
    float h = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();
    // Read temperature as Fahrenheit (isFahrenheit = true)
    float f = dht.readTemperature(true);
    // Check if any reads failed and exit early (to try again).
    if (isnan(h) || isnan(t) || isnan(f)) {
      Serial.println("Failed to read from DHT sensor!");
      strcpy(celsiusTemp, "Failed");
      strcpy(fahrenheitTemp, "Failed");
      strcpy(humidityTemp, "Failed");
    }
    else{
      char hh[100] = "";
      char tt[100] = "";
      sprintf(hh, "token=ekJaU2JCYzJGK3V5cUwvSjEvTINMUT09&type=humidity&data=%.2f", h);
      sprintf(tt, "token=ekJaU2JCYzJGK3V5cUwvSjEvTINMUT09&type=temp&data=%.2f", t);

      // Computes temperature values in Celsius + Fahrenheit and Humidity
      float hic = dht.computeHeatIndex(t, h, false);
      dtostrf(hic, 6, 2, celsiusTemp);
      float hif = dht.computeHeatIndex(f, h);
      dtostrf(hif, 6, 2, fahrenheitTemp);
      dtostrf(h, 6, 2, humidityTemp);
      // You can delete the following Serial.print's, it's just for debugging purposes
      Serial.print("Humidity: ");
    }
  }
}

```



```

    Serial.print(h);
    Serial.print(" %\t Temperature: ");
    Serial.print(t);
    Serial.print(" ^C \n");

    HTTPClient http;    //Declare object of class HTTPClient
    http.begin("http://mrzareei.ir/iot/public/api/v1/new_data");    //Specify request destination
    http.addHeader("Content-Type", "application/x-www-form-urlencoded");
    int httpCode = http.POST(tt);    //Send the request
    String payload = http.getString();    //Get the response payload
    Serial.println(httpCode);    //Print HTTP return code
    Serial.println(payload);    //Print request response payload
    http.end();    //Close connection

    HTTPClient http2;    //Declare object of class HTTPClient
    http2.begin("http://mrzareei.ir/iot/public/api/v1/new_data");    //Specify request destination
    http2.addHeader("Content-Type", "application/x-www-form-urlencoded");
    int httpCode2 = http2.POST(hh);    //Send the request
    String payload2 = http2.getString();    //Get the response payload
    Serial.println(httpCode2);    //Print HTTP return code
    Serial.println(payload2);    //Print request response payload
    http2.end();    //Close connection
}
}
else{
    Serial.println("Error in WiFi connection");
}
delay(5000);    //Send a request every 2 minutes
}

```

After reading the data from DHT. We send it to webserver with a post request.

This delay at the end of the code indicates how often this data should be send to the server.

Resources:

- [1] <https://www.programmableweb.com/news/review-postman-client-makes-restful-api-exploration-breeze/brief/2014/01/27>
- [2] <http://docs.guzzlephp.org/en/stable/>
- [3] <https://en.wikipedia.org/wiki/XAMPP>
- [4] [https://en.wikipedia.org/wiki/Bootstrap_\(front-end_framework\)](https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework))
- [5] <https://startbootstrap.com/template-overviews/sb-admin/>
- [6] <http://azmind.com/bootstrap-login-forms/>
- [7] <https://en.wikipedia.org/wiki/NodeMCU>
- [8] <http://www.santy.cz/senzory-c24/dht22-alone-i103/>
- [9] <http://randomnerdtutorials.com/esp8266-dht11dht22-temperature-and-humidity-web-server-with-arduino-ide/>
- [10] <http://www.instructables.com/id/Quick-Start-to-Nodemcu-ESP8266-on-Arduino-IDE/>
- [11] <http://www.instructables.com/id/Interface-DHT11-Humidity-Sensor-Using-NodeMCU/>