

Name: Chia Ei Ji

Project: Back End Assessment

## Sample Output

### Rest API (Postman)

#### Register user

<http://localhost:8000/api/register>

#### Headers

Key	Value
Content-Type	application/x-www-form-urlencoded
Accept	Accept

#### Body and Output:

The screenshot displays a Postman interface for a REST client. At the top, a list of requests is shown, with the current one being a POST request to `http://localhost:8000/api/register`. The main panel shows the request details for this endpoint. The method is POST, and the URL is `http://localhost:8000/api/register`. The 'Body' tab is selected, showing form data with the following fields:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	lawrence Chia 55	
<input checked="" type="checkbox"/> email	admin@lawrencechia55.com	
<input checked="" type="checkbox"/> password	password	
<input checked="" type="checkbox"/> confirm-password	password	
Key	Value	Description

Below the form data, the 'Body' tab shows the response in JSON format:

```
1 {
2   "name": "lawrence Chia 55",
3   "email": "admin@lawrencechia55.com",
4   "updated_at": "2021-06-29T13:56:16.900000Z",
5   "created_at": "2021-06-29T13:56:16.900000Z",
6   "id": 13
7 }
```

The status bar at the bottom indicates a successful response with status 200 OK, time 1560 ms, and size 464 B.

## Login user

<http://localhost:8000/api/login>

### Body and Output:

[illegible]

```
## The access token will be used to access the CRUD functions
```

### Show the details of certain user

<http://localhost:8000/api/user/13/detail>

**13 is userId**

## Headers

Key	Value
Content-Type	application/x-www-form-urlencoded
Authorization	Bearer XXXX

XXXX = Access Token

Output:

http://localhost:8000/api/user/13/detail

Save

✎

🗨

GET

http://localhost:8000/api/user/13/detail

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Headers

7 hidden

Cookies

KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets
<input checked="" type="checkbox"/> Content-Type	application/x-www-form-urlencoded				
<input checked="" type="checkbox"/> Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiU9.eyJhdWQiOiJ1IiwianRpIjo				
Key	Value	Description			

## Show the user list

<http://localhost:8000/api/users>

## Headers

Key	Value
Accept	application/json
Authorization	Bearer XXXX

XXXX = Access Token

## Output:

The screenshot shows a REST client interface with a GET request to `http://localhost:8000/api/users`. The request headers are set to `Accept: application/json` and `Authorization: Bearer eyJ0eXAIOWKV1QILCJhbGciOiJIUzI1NiIsInR5cCI6IkpXZWxianRpIjo`. The response status is 200 OK, and the body contains a JSON array of five user objects.

```
1 {
2   {
3     "id": 1,
4     "name": "Lawrence Chia",
5     "email": "lwchia1998@gmail.com",
6     "email_verified_at": null,
7     "created_at": "2021-06-29T10:23:34.000000Z",
8     "updated_at": "2021-06-29T10:23:34.000000Z"
9   },
10  {
11    "id": 2,
12    "name": "ABC 123",
13    "email": "abc123@gmail.com",
14    "email_verified_at": null,
15    "created_at": "2021-06-29T10:34:04.000000Z",
16    "updated_at": "2021-06-29T10:34:04.000000Z"
17  },
18  {
19    "id": 4,
20    "name": "Finally v2 123",
21    "email": "admin@lawrencechia123.com",
22    "email_verified_at": null,
23    "created_at": "2021-06-29T11:58:48.000000Z",
24    "updated_at": "2021-06-29T12:44:06.000000Z"
25  },
26  {
27    "id": 5,
```

## Add user

<http://localhost:8000/api/user/13/addUser>

**13 is userId**

## Headers

Key	Value
Accept	application/json
Authorization	Bearer XXXX
Content-Type	application/x-www-form-urlencoded

XXXX = Access Token

## Body and Output:

The screenshot shows a REST client interface with a POST request to `http://localhost:8000/api/user/13/addUser`. The request body is set to `x-www-form-urlencoded` and contains the following data:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	example 1234	
<input checked="" type="checkbox"/> email	example1234@example.com	
<input checked="" type="checkbox"/> password	password1234	
<input checked="" type="checkbox"/> confirm-password	password1234	
Key	Value	Description

The response is a JSON object with the following structure:

```
1 {
2   "name": "example 1234",
3   "email": "example1234@example.com",
4   "updated_at": "2021-06-29T13:59:13.000000Z",
5   "created_at": "2021-06-29T13:59:13.000000Z",
6   "id": 15
7 }
```

The status is 201 Created, Time: 937 ms, Size: 464 B.

## Update user Details

<http://localhost:8000/api/user/14/updateDetail>

**14 is userId**

### Headers

Key	Value
Accept	application/json
Authorization	Bearer XXXX
Content-Type	application/x-www-form-urlencoded

XXXX = Access Token

### Body and Output:

The screenshot displays a REST client interface with a PUT request to `http://localhost:8000/api/user/14/updateDetail`. The request body is a JSON object with the following fields: `id` (14), `name` (Hello1234), `email` (hello123@hello.com), `email_verified_at` (null), `created_at` (2021-06-29T13:58:51.000000Z), and `updated_at` (2021-06-29T13:59:39.000000Z). The response status is 200 OK, with a time of 951 ms and a size of 476 B. The response body is a JSON object with the same fields as the request body.

**Request Headers:**

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	Hello1234	
<input checked="" type="checkbox"/> email	hello123@hello.com	
<input checked="" type="checkbox"/> password	password	
<input checked="" type="checkbox"/> confirm-password	password	
Key	Value	Description

**Response Body:**

```
1 {
2   "id": 14,
3   "name": "Hello1234",
4   "email": "hello123@hello.com",
5   "email_verified_at": null,
6   "created_at": "2021-06-29T13:58:51.000000Z",
7   "updated_at": "2021-06-29T13:59:39.000000Z"
8 }
```

# Delete User

<http://localhost:8000/api/user/15>

15 is userId

## Headers and Output:

POST http://localhost:8000/api/user/15

DELETE http://localhost:8000/api/user/15

PUT http://localhost:8000/api/user/15

POST http://localhost:8000/api/user/15

+

No Environment

Save

DELETEDelete User

Send

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Cookies

Headers

7 hidden

KEY	VALUE	DESCRIPTION		Bulk Edit	Presets
<input checked="" type="checkbox"/> Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1Ni99eyJhdWQiOiJ1b3Rpb2o				
<input checked="" type="checkbox"/> Accept	application/json				
Key	Value	Description			

Body

Cookies (2)

Headers (10)

Test Results

Status: 200 OK

Time: 1009 ms

Size: 319 B

Save Response

Pretty

Raw

Preview

Visualize

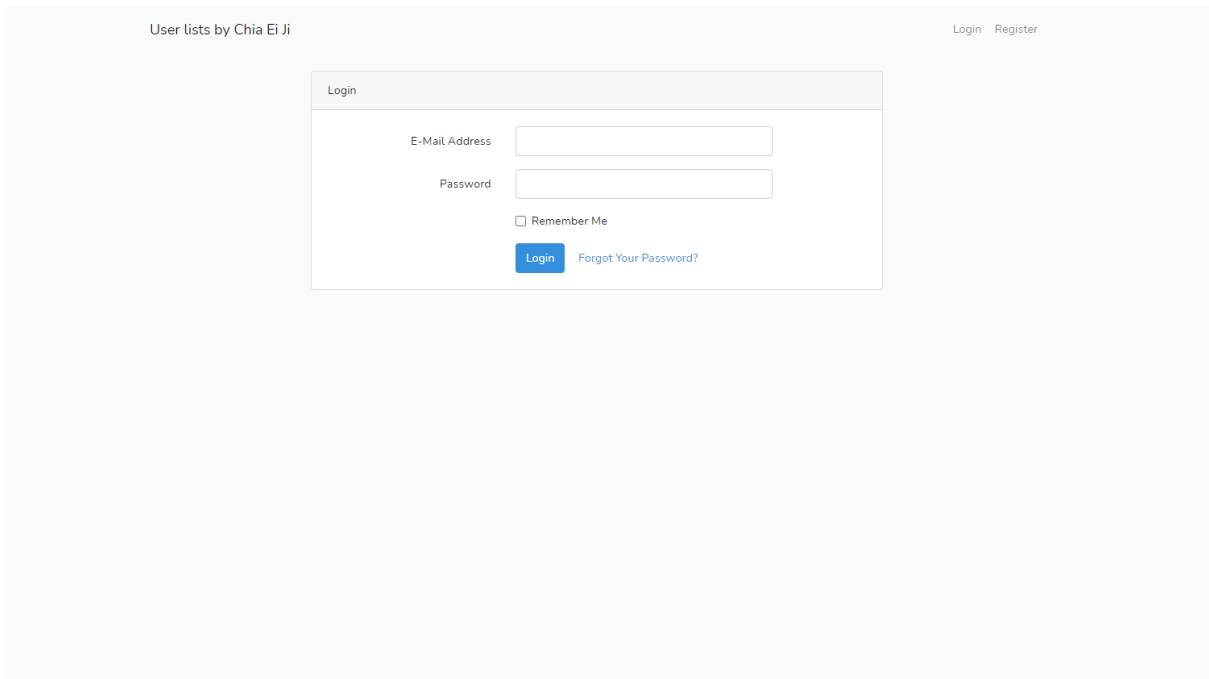
HTML

1

284

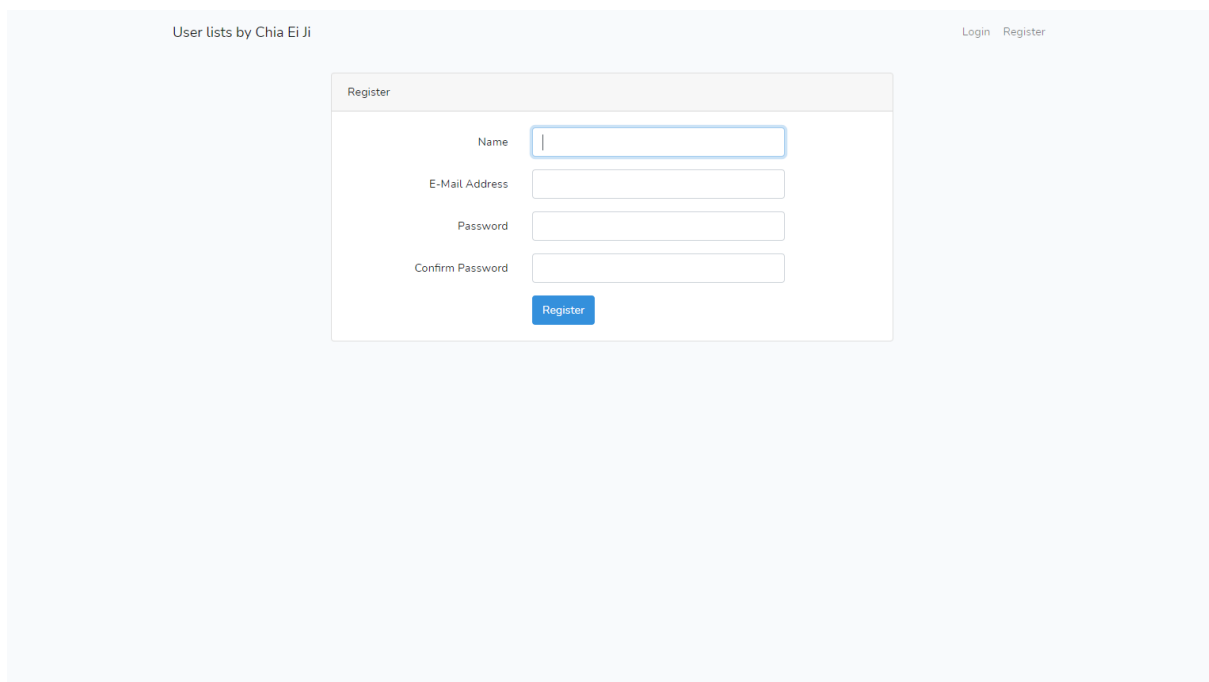
## Rest API (Web)

The login page is set as default web route:



The screenshot shows a web application interface with a light blue background. At the top left, the text "User lists by Chia Ei Ji" is displayed. At the top right, there are links for "Login" and "Register". In the center, there is a "Login" form with a title bar. The form contains two input fields: "E-Mail Address" and "Password". Below these fields is a checkbox labeled "Remember Me". At the bottom of the form, there is a blue "Login" button and a link that says "Forgot Your Password?".

For the first time user, you can register as a user as well:



The screenshot shows a web application interface with a light blue background. At the top left, the text "User lists by Chia Ei Ji" is displayed. At the top right, there are links for "Login" and "Register". In the center, there is a "Register" form with a title bar. The form contains four input fields: "Name", "E-Mail Address", "Password", and "Confirm Password". At the bottom of the form, there is a blue "Register" button.



The main page after logged in:

User lists by Chia Ei Ji

Manage UsersABC 123

Users Management

Create New User

No	Name	Email	Action
1	Hello1234	hello123@hello.com	ShowEditDelete
2	lawrence Chia 55	admin@lawrencechia55.com	ShowEditDelete
3	hellohello	hellohello@example.com	ShowEditDelete
4	hello1234	hello1234@example.com	ShowEditDelete
5	hello123	hello123@example.com	ShowEditDelete

<

1

2

>

Create new user:

User lists by Chia Ei Ji

Manage UsersABC 123

Create New User

Back

Name:

Name

Email:

Email

Password:

Password

Confirm Password:

Confirm Password

Submit

Show the certain user details:

User lists by Chia Ei Ji

Manage Users ABC 123 ▾

Show User

Back

ID: 14

**Name:** Hello1234

**Email:** hello123@hello.com

**Created At:** 2021-06-29 13:58:51

**Updated At:** 2021-06-29 13:59:39

Edit and update the certain user:

User lists by Chia Ei Ji

Manage Users ABC 123 ▾

Edit New User

Back

**Name:**

Hello1234

**Email:**

hello123@hello.com

**New Password:**

Password

**Confirm Password:**

Confirm Password

Submit