Unit Test Document

## Installation and run

1. Environment and Library

- Mocha: Testing Environment

Mocha is a javascript framework for Node.js which allows Asynchronous testing. Let's say it provides the environment in which we can use our favorite assertion libraries to test the code.

Mocha comes with tons of great features, the website shows a long list but here are the ones I like the most:

* simple async support, including promises.
* async test timeout support.
* before, after, before each, after each hooks (very useful to clean the environment where each test!).
* use any assertion library you want, Chai in our tutorial.

- Chai: Assertion Library

Chai shines on the freedom of choosing the interface we prefer: "should", "expect", "assert" they are all available. I personally use should but you are free to check it out the [API](http://chaijs.com/api/) and switch to the others two. Lastly [Chai HTTP](https://github.com/chaijs/chai-http) addon allows Chai library to easily use assertions on HTTP requests which suits our needs.

1. How to use?

update scripts in package.json

"scripts": {

"test": "mocha --compilers js:babel-core/register",

},

1. How to run?

Npm test

## Unit Test Descriptions

- Test Basic Auth

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Mothod | Name | Description |
| 1 | /POST | it should valid username and password | Test login user with valid username and password API URL : /api/v1/auth/login  Username : admin  Password : admin |
| 2 | /POST | it should invalid username and password | Test login user with invalid username and password API URL : /api/v1/auth/login  Username : admin1  Password : admin1 |
| 3 | /POST | it should required username and password | Test login user with null username and password API URL : /api/v1/auth/login  Username : null  Password : null |

- Test Basic Patient

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Method | Name | Decription |
| 1 | /GET | it should GET all the patients but not login | Get all the patient, but user not login  API URL: /api/v1/patient |
| 2 | it should GET all the patients user login | Get all the patient, user have been login API URL: /api/v1/patient |
| 3 | /POST | it should POST a Patient not login | Create new Patient, but user not login API URL: /api/v1/patient |
| 4 | it should POST a Patient but patientId field null | Create new Patient, user have been login, but param patientId is null API URL: /api/v1/patient |
| 5 | it should POST a Patient but patientName  field null | Create new Patient, user have been login, but param patientName is null API URL: /api/v1/patient |
| 6 | it should POST a Patient valid | Create new Patient, user have been login. All param valid API URL: /api/v1/patient |
| 7 | it should POST a Patient duplicate patientId | Create new Patient, user have been login. But duplicate patientId  API URL: /api/v1/patient |