Internal Testing Documentation

COMPSCI 4ZP6B

Darren Tu - 400017919
Ji Who Choi - 400023747
Sujan Kandeepan - 400079697
Immanuel Odisho - 400074199

How to run the back end tests

Our back-end tests are run through making API calls to the endpoints. We use the *Jest* testing framework and the *Supertest* library to run these tests. To run the tests yourself, please do the following steps:

1. Install MongoDB locally on your machine.
2. Create a directory called ‘erukuDatabase’
3. Run mongod --dbpath=/path/to/erukuDatabase (this assumes you have mongod in your path)
4. Within the cloned repository, create a file named .env and put it into the backend folder.
5. Within the .env file, specify the mongo URI. If the mongo port is 27017 (the default), then have MONGODB_URI=mongodb://localhost:27017/ in the .env file.
6. Run npm install within the backend directory. Ensure that the package.json contains *Jest* and *Supertest*.
7. Run npm run test.

*If some of the test cases are failing, that is probably because you have pre-existing info inside of the MongoDB database. Please ensure those are deleted before you run the tests.

To do this, run mongo and then run show dbs. It should list the databases. Assuming you just recently installed mongoDB, there should only be admin and local. If there are databases such as Events, then you need to drop them before you run the tests. Check this (https://docs.mongodb.com/manual/reference/method/db.dropDatabase/) to drop a database.

How to run the front end tests

Simply run npm install followed by npm test or jest --verbose --runInBand tests from the project root directory to run all tests written for the front end.

These tests have also been written using the *Jest* testing framework and require no additional dependencies. The option to use a test renderer library had been explored but due to the complexity and number of bugs with the libraries involved in the setup, this was not possible. Instead, the front-end tests verify the behaviour of the UI components by testing their underlying function implementations (refactored into separate files) with mock state and HTTP requests.

Current test case failures are due to some features being not yet fully implemented and some data still hardcoded, as the process to integrate the database with all front-end functions is still ongoing. We will follow a test-driven approach to integrate the back end into the front-end application until all test cases pass with little to no modification required. Some UI components have still not yet been implemented, and test cases for those will also have to be written later.

How to run the stress tests

The backend server must be running on the default port in order to perform the stress tests. Python 3 also needs to be installed in order to run the tests. Simply run npm install -g artillery to install the artillery stress testing toolkit. Once that is completed run the following command python3 stress.py inside the eruku-mesi/stress directory. The command will run multiple stress tests and load tests for all of the endpoints in the backend. The stress test will output a file named report.txt . The report can then be reviewed and give details on how the backend performed. Status codes of 404 are not expected in the report.

Configuring the stress tests

config.yaml

The configuration file can be modified in order to change the number of requests and duration of stress testing. Inside the configuration file, simply change the duration to the number of seconds you would like the stress test to take and change the arrivalRate to the number of virtual users that will be created per second to execute the test. For example, in the default configuration, the arrivalRate is set to 1, and the duration is set to 10. This means each yaml file will create 1 virtual user to execute the scenario defined in that yaml per for 10 seconds.

Acceptance tests

The above tests also serve as acceptance tests and are based on our functional requirements.
SRS - https://gitlab.cas.mcmaster.ca/tut/eruku-mesi/-/blob/master/doc/srs.md