Testing Policy Document

COS 301: REDIRECTION

2018

Team Members

Stephen Teichert - u16254661

Kyle Wood - u16087993

Russell Dutton - u16016612

Jeffrey Russell - u16010648

Justin Grenfell - u16028440

Byron Antak - u16039689

1 Testing Process

Included in our tests are linters which are configured to check whether the files in the branch macth with the coding standards as stipulated in the coding standards document. These linters are run along with the tests. If the files fail the linting checks (indicating violating the coding standards), the tests will fail.

We ensured that the dev branch cannot be accessed directly by merge or push requests, this has been done in order to prevent any team member from changing the branch and potentially messing with the system as a whole. When a new feature needs to be added, an issue is created on GitHub and a corresponding branch will be created with the following structure <branch_name>-<issue_number> and all scripts are added and changed within the branch to in order to solve the problem or issue at hand. The dev branch can only be changed upon a merge request thus once a branch has been finalized, a merge request is made to the dev branch. Consequently the member who creates the pull/merge request assigns 2 code reviewers to look over the code. The code reviews using github then comment on the code or request changes (blocking the request from being finalized). Also note that that if automated tests fail Github is probihited from completing the merge request, further ensuring that features are tested properly before becoming apart of our codebase. (The same applies to both dev and master) Once the reviewers approve the changes, the branch our team leader (Jeffrey Russell) who will finalize the merge into the dev or master branch.

2 Testing Tools

Since our code is fundamentally split into front-end and back-end, we have decided to implement Karma (https://karma-runner.github.io/2.0/index.html) as our front-end test environment to test the angular side of development and we have decided to use Jasmine https://jasmine.github.io/2.0/introduction.html as our test environment for the express back-end API server.

We are using automated Testing implemented by Travis (https://travis-ci.org/) integrated with GitHub such that when a push is made to any branch, all the tests are run. Travis will then proceed to email the person who pushed lasting indicating if the tests had failed. Hence if there is a code standard violation or a test fails, the member who pushed that branch will be notified.

3 Testing Cases

Since we are using an angular framework, each component (essentially a web page) has its own tests and hence we cannot take you to all the test cases. For the back-end:

- Code Generation (https://github.com/301-Redirection/301-Project/tree/master/tests/testLuaCodeMana
- API Test (https://github.com/301-Redirection/301-Project/tree/master/tests/testBotAPISpec.js)
- MySQL Test (https://github.com/301-Redirection/301-Project/tree/master/tests/testMySqlSpec.js)
- Data scraping Test (https://github.com/301-Redirection/301-Project/tree/master/tests/testStaticSpec.js

4 Testing History

The tests history can be viewed at Travis (https://travis-ci.org/301-Redirection/301-Project/builds) (Please note that these are the most recent and more can be viewed if you click the "show more" button). Included below is a snapshot of the test history for convenience.