Alabama Liquid Snake

University of Pretoria

Epi-Use

Botic - Privacy aware chatbot
Testing Policy

Justin Grenfell - u16028440
Peter Msimanga - u13042352
Alicia Mulder - u14283124
Kyle Gaunt - u15330967
Lesego Mabe - u15055214

Contents

1 Introduction

The development approach will be Test Driven Development as it can reduce prerelease defects densities by 90 percent compared to similar projects that do not use it, and it can improve programmer productivity[1]. It is ideal for solo programming[1], which is the programming technique we will be following.

2 Peer Review

This is done before merging a feature into dev branch on our Github repository. Automatic merges are prevented and at least one other member of team is required to review the code. Once the review is done, then only can the merge proceed.

As part of our Software Quality Assurance life cycle, we make use of verification[1] to ensure that our development activities are carried out correctly and according to the methodology we have chosen[, the Agile Unified Process]. This entails that the reviewer checks whether:

- The feature implemented was required, according to the requirements specified in the SRS document.
- The feature implemented fits in with the software design and that the software design principles were followed.
- The feature adheres to the coding standards that we have set out in our coding standards document.
- The feature implemented has unit tests and that the tests pass those unit tests (Dynamic Validation)
- The feature implemented has integration tests and that those tests pass (Dynamic Validation)

Figure 1: Peer reviews on Github

3 Automated Testing and Continuous Integration

Whenever code is merged into dev or branch, Travis CI is used for continuous integration, as well as to run automated testing once more before deployment. Travis CI will build all of our subsystems, including building the Docker images that house them, before uploading those images to our Docker repository (`https://hub.docker.com/r/alabamaliquidservices/botic`). Check our latest builds and tests here: `https://travis-ci.com/cos301-2019-se/Botic`.

3.1 Travis CI

Seperate scripts are ran to test each subsystem, and the results of those tests are pipped to Coveralls. Coveralls is a service that we use to indicate our code coverage.

3.2 Frontend Testing

Travis CI we will be running unit tests on Headless Chrome[2] since it isn't practical to do them on a browser when Travis CI runs the automated test scripts.

Karma and Jasmine are used to test the subsystem, while tslint used to check for errors, typos as well as to lint the relevant files according. As a result, the linter will be run first in preparation for the tests. Jasmine is the framework with which we are going to use for our tests and Karma will be used to run those tests.

3.2.1 TypeScript Classes in 'shared' Folder

The classes in the shared folder are those that are meant to be used across the system by various controllers, services and others. We will use the same testing framework, i.e. Jasmine, to test these. The unit tests will be developed first, according to the Test Driven Development philosophy.

These test classes, or rather unit tests, will be in the "spec" folder that is in the root "allisn/".

3.3 Message Scrubber

Pytest is the framework that will be used to test the Message Scrubber, which is coded using Python. The command to run the tests is just "pytest." We will be using Postman to run tests; this is intergrated into Travis.

3.4 Persistence Layer Testing

Each implementation file will be in the same directory as it's test file. Jasmine will be used for testing, much like the frontend. Test Driven Development will be used here as well. Testing the database queries will be done using a test database instead of a mocked mongdoDB instance.

4 References

[1] D. C. Kung, Object-Oriented Software Engineering An Agile Unified Methodology. McGraw-Hill, 2014.

[2] E. Bidelman, "Getting started with headless chrome," https://developers.google.com/web/updates/2017/04/headless-chrome, May 2019, accessed on 2019-05-24.