Testing Policy

Team Hackermen

TicketSalad

Team Members:
Thato Mothusi
Jarryd Baillie
Brandon Texeira
Thomas Honiball
Tristan Joseph

Client:
Tribus Digita

1 Testing Processes

Since we are adopting scrum methodology and it does not say much about the testing process, we decided to adopt a waterfall approach within our sprint.Our sprints usually last 2 weeks and within those two weeks we develop certain features of the system and once those features are completed, we then write tests for those features we developed. So we are essentially using a white-box testing method whereby we first write code for the system and then write the test cases once we have knowledge of the source code.

1.1 Functional Requirements Tested

- A user is able to login
- A user is able to claim on an event
- A user is able to view their profile
- A user is able to add a credit card
- A user is able to buy credits

1.2 Non-Functional Requirements Tested

- The users password is encrypted
- The system is able to handle a minimum of 3000 function calls per second
- The system prevents users from entering invalid credit card data
- The system prevents proceeding with an incorrect username and/or password
- The system database listens on a non-default port
- The MongoDB HTTP status interface is not visible on port 28017
- The system is using up to date versions of its software
- The system uses TLS/SSL encryption for database communication

- The system is protected from the following MongoDB vulnerabilities:
  - CVE-2015-7882
  - CVE-2015-2705
  - CVE-2014-8964
  - CVE-2015-1609
  - CVE-2014-3971
  - CVE-2014-2917
  - CVE-2013-4650
  - CVE-2013-3969
  - CVE-2012-6619
  - CVE-2013-1892
  - CVE-2013-2132

2 Testing Tools

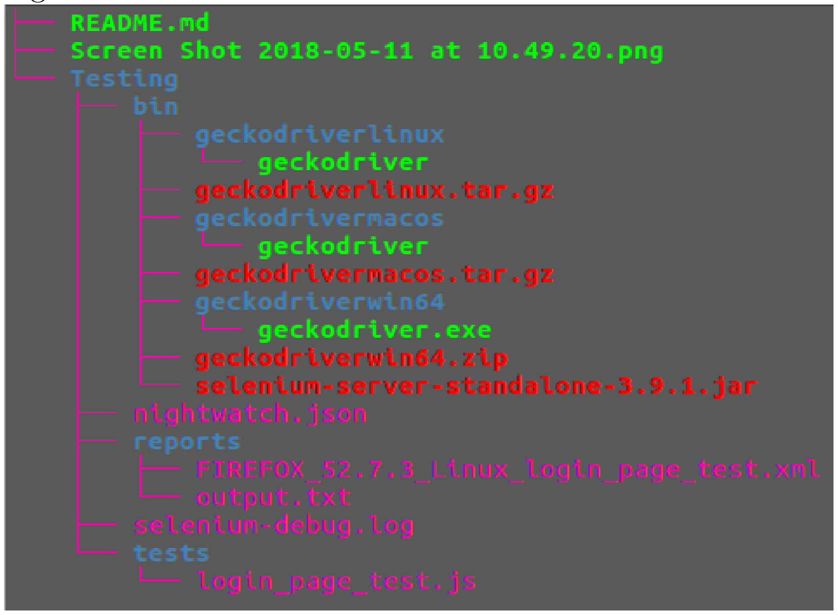
Nightwatch is an automated testing framework for web applications and websites, written in Node.js and using the W3C Web-driver API (formerly Selenium Web-driver).Nightwatch relies on ”nightwatch.json” as the configuration file to run test files, this json file is placed on the project's root directory. The json file allows for the specification of configuration settings like test environments, test file paths, and selenium specific settings.The reason we chose to use nightwatch is beacuse it has a simple but powerful syntax which enables us to write tests very quickly, using languages like javascript and CSS or Xpath selectors. It also has a built in command-line test runner which can run tests either sequentially or in parallel. Lastly nightwatch allows for flexibility, what this means is that there is a flexible command and assertion framework which makes it easy to extend to implement our application specific commands and assertions.

The Grinder is an application which allows us to write scripts that will test websocket and functional response of the server under a heavy load. We chose to use this application as it was not only able to interface with Meteor functions specifically, but also due to the versatility offered by its code based testing method.

Mongoaudit is an application which runs automated testing to determine vulnerabilities present within our database server. The system provides good insight into what vulnerabilities might be present within our system so that we may take precautions to prevent various attacks such as data injections and side channel attacks targeted at our server.

3 Test Cases

Our test cases are located on the master branch in a folder called Testing.Below is a tree structure of where our test cases are located on git.



```
├── README.md
├── Screen Shot 2018-05-11 at 10.49.20.png
└── Testing
    ├── bin
    │   ├── geckodriverlinux
    │   │   └── geckodriver
    │   ├── geckodriverlinux.tar.gz
    │   ├── geckodrivermacos
    │   │   └── geckodriver
    │   ├── geckodrivermacos.tar.gz
    │   ├── geckodriverwin64
    │   │   └── geckodriver.exe
    │   ├── geckodriverwin64.zip
    │   └── selenium-server-standalone-3.9.1.jar
    ├── nightwatch.json
    ├── reports
    │   ├── FIREFOX_52.7.3_Linux_login_page_test.xml
    │   └── output.txt
    ├── selenium-debug.log
    └── tests
        └── login_page_test.js
```

4 History