Content Aggregator

Test Plan

April 4, 2020

Version 1.0
Team Galvatron

David Guo	57734162
Pengwei Zhou	73569758
Peter Han	14912166
Tony Kong	47078150
Tyler Nee	22705157
Vickie Yen	13358156
Yuting Wen	44625168

Revision History

Date	Version	Status	Prepared by	Comments
Mar 12, 2020	0.1	Start	Team Galvatron	Initial Test Plan Document
Apr 4, 2020	1.0	Final	Team Galvatron	Final submission

Summary

Our overall testing approach consists of automated SAT tests written using Jasmine 3.50 and supertest 4.0.2, and manual UAT tests. The developer will be responsible for automating SAT tests, while manual testers will be responsible for running the manual test scripts. We will be applying a risk based approach to testing. There are an estimated 3 test cycles for SAT, and our accepted criterion to move forward is that all happy path test cases are passing and there are no high priority bugs.

Test Environment Setup

Our test environment will be located at http://galvatron.dream.sh:3000. Our test environment is set up on an AWS EC2 t2.micro instance. We will be running our automated tests in this environment with our test data generated using the scripts listed in the appendix. Our deployment scripts are also listed in the appendix as A.3.

Functional Test Plan

We have each of our components in a table format, and included our testing technique for each category as well as the number of anticipated scripts. We do not include all the different categories, UAT, SAT, Unit, for every component as they are not always appropriate. We will not have any unit tests listed because our UAT and SAT tests fully cover all the cases that unit tests would be applicable for.

1. Users can login via Google OAuth	
SAT	Technique: black box and automated Jasmine tests
UAT	Technique: black box and manual
Number of Anticipated Scripts	2 UAT tests, and 2 SAT tests to test valid and invalid cases.

2. Users can submit posts	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual

Number of anticipated scripts	2 UAT tests, and 2 SAT tests to test valid and invalid cases.

3. Users can submit blog streams	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	2 UAT tests, and 2 SAT tests to test valid and invalid cases.

4. Users can view published posts	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	4 UAT tests, and 1 SAT test to test valid and invalid cases.

5. Users can search and filter posts	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	2 UAT tests, and 3 SAT tests to test valid and invalid cases.

6. Categories can be managed by the admin	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	4 UAT tests, and 5 SAT tests to test valid and invalid cases.

7. Companies can be managed by the admin	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	4 UAT tests, and 3 SAT tests to test valid and invalid cases.

8. Approvers can approve posts/blogs	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	2 UAT tests, and 1 SAT tests to test valid and invalid cases.

9. Published posts can be managed by admins	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	3 UAT tests, and 4 SAT tests to test valid and invalid cases.

10. Unpublished posts can be managed by admins	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	3 UAT tests, and 4 SAT tests to test valid and invalid cases.

11. Users can view posts view RSS	
UAT	Technique: black box and manual
Number of anticipated scripts	1 UAT test to verify user can view RSS

12. Users will receive daily/weekly notifications	
SAT	Technique: white box and automated Jasmine tests
Number of anticipated scripts	0 UAT tests, and 1 SAT tests to test valid and invalid cases.

13. Admins can modify system settings	
SAT	Technique: white box and automated Jasmine tests
UAT	Technique: black box and manual
Number of anticipated scripts	1 UAT tests, and 4 SAT tests to test valid and invalid cases.

14. Blog streams will be read on a hourly schedule	
SAT	Technique: white box and automated Jasmine tests
Number of anticipated scripts	1 UAT tests, and 1 SAT tests to test valid and invalid cases.

Non Functional Testing

We will not be testing concurrent users for our application, as we are deployed on a free tier AWS EC2 instance, and do not anticipate being able to properly test concurrent users until the application is deployed to a proper server that can properly handle over 100 users. For the components we are testing we will only be having manual UAT tests.

1. Limit response times for searches and filters be less than 4 seconds.	
UAT	Technique: black box manual tests
Number of anticipated scripts	4 scripts that test the loading times for searching and filtering.

2. Multi-browser support (Chrome, Microsoft Edge, Safari, Firefox).	
UAT	Technique: black box manual tests
Number of anticipated scripts	3 scripts to test browser support for the latest versions of Edge, Safari, Firefox

Test Scripts

Our approach to writing the test scripts is to have enough coverage to test both the happy path as well as edge cases for all our listed components, while minimizing repetition. We have listed our test scripts in our appendix as A1. We have a functional test plan as well as a non functional test plan further divided into UAT/SAT. Our UAT tests are all manual tests and they describe the step by step process to get to our expected result. While our SAT tests are automated and describe what it's testing and the expected result.

Security Testing and Test Data Approach

We will not be conducting security testing as authentication is handled with OAuth 2.0 with our identity provider being Google. However we do not use the standard OAuth 2.0 workflow as we rely on authenticating with the google token returned from the standard workflow, and then

using a JWT to authenticate future API requests. We have included authorization based testing by testing that users without a proper role are not able to navigate to unauthorized endpoints. We are using Sequelize as our MySql object-relational mapping and library will automatically handle sanitizing our user inputs to prevent SQL injection attacks.

Our project contains data regarding a user's email and company. However, we do not contain any other data regarding user profiles that could be potentially harmful to them, such as passwords. The test data for posts are from public blogs. Data masking will not be required as we do not store any sensitive information.

We will have test data of 500 posts, 500 users, 20 companies, and 20 categories. We will use a mixture of data from sources provided by the industry sponsor, as well as generated through python scripts. Our data will cover all possible data types. For instance, there will be users, admins, and approvers. There will be blog posts and blog streams as well, where some are unpublished and some are published. Please refer to Appendix A.2 for test data.

Risks

Some risks considered include SQL injection, sensitive data exposure, broken authentication management, and broken access control. SQL injections are prevented as we are using Sequelize, which automatically sanitizes our user inputs. Sensitive data exposure is prevented, as we are using Google OAuth for authentication and don't need to store user passwords. To prevent broken access control, such as an attacker accessing a resource by just knowing the URL, we have run tests for every endpoint to ensure that only the correct roles have access to each endpoint.

Regression Testing Needs

Regression Testing pack would include our automated SAT tests. These tests should be run on every build and will run quickly. We will not include UAT tests in our regression pack as they would be too costly to frequently run.

Appendix

A.1 Manual Test Script:
https://docs.google.com/spreadsheets/d/1krXjpl4Uc9pvpEQOagxldiYUvdriTbeCwK4-UG2GRpA/edit?usp=sharing

A.2 Mock Data Script:
https://drive.google.com/file/d/1D_rFeI_-l66-UNeyByuyNOnRmYr10pgA/view?usp=sharing

A.3 Deployment script:

https://drive.google.com/open?id=1ffiyVuE_QSDTMuZRFww8iO0Rb3wOX2un