CSCI-4830 Web App 2

**Budgeting Tool**

System Test Plan

**Members:** Alec Bokowski, Christian Woodle, Gabriel Perez, James Triplett, Mark Turpen, Thomas Cordwin

March 7th, 2019

**System Test Plan**

Introduction

This document will outline the testing methods that we will use for our budgeting application. Our testing methods will be performed by both developers and our quality assurance lead. Developers will test their own code through the use of unit tests, and the user interface developer and backend developer will integrate their code and test it to ensure compatibility. Once integration has been completed successfully, quality assurance will write automated test cases to test our application. They will also determine what manual test cases are needed, if any, and perform them.

Unit Testing

Developers will be required to write unit tests for each segment of their code. By ensuring that modules of code work correctly, we can ensure that they will work correctly when put together. Writing unit tests for small pieces of code makes it much easier for developers to troubleshoot issues and determine where any errors are occuring. We strive to have a minimum of 95% of our code covered throughout all of our unit tests. Once developers have accomplished this code coverage and all units are passed, they can move onto integration testing. Developers will use JUnit 5 to write unit test cases.

Integration Testing

Backend, front-end, and database developers are will integrate their code to ensure that they all work together. Testing all of their code together helps them to see if any errors occur when all of the code works together. Integration testing will be performed in a development environment that will mirror our production environment. Once integration testing is complete and the developers are satisfied, the application can be moved to QA testing.

Quality Assurance Testing

Automated Test Cases

In order to automate our testing we will be using Selenium. Selenium is an automated web testing tool. Since our program consists of input and set expected output, we will be able to use Selenium to go through all the test cases. Selenium will be able to verify that every web page loads correctly by checking known html elements and confirming their presence. This way we know that every page is loading correctly, and we can focus mostly on the input and output of the program code itself.

Step 1: Creating an account / Logging in and out

1. Selenium will load the web page and verify the elements are correctly loaded.
2. Selenium will then click on the “Create an Account” button and create a testing account.
3. Selenium will then go through and try to inset bad input in the account income and expense tabs.
4. Selenium will then go through and fill out the income and expense tabs with valid input, in order to finish setting up the account. It will also store these variables for later verification.
5. Selenium will then log out and log back in to confirm the logging functionality, and that the account was properly created.
6. Any errors throughout the process will be output to an error file to be read after the script is done running.

Step 2: Verifying Data Integrity

1. After logging in, Selenium will navigate through the income and expense tabs to confirm that the data has not changed.
2. Any incorrect data will be output to an error file containing the location of the invalid data.

Step 3: User Settings

1. Selenium will click on the “User Settings” button to start the user settings testing.
2. Selenium will test the change username function.
3. Selenium will test the change password function.
4. Selenium will then log out and log back in to confirm these changes worked.

Step 4: Deleting Account

1. Selenium will go back into the “User Settings” and delete the account used for testing.
2. Selenium will try to log in again with the credentials to ensure the account is deleted.

Manual Test Cases

There will be some manual testing that needs to be done due to the complexity of verifying such functions through Selenium. This will be done by the quality assurance lead.

Step 1: Error Verification

1. A tester will need to initiate the Selenium test script and make sure all the variables are set up correctly for the current release version.
2. The tester will need to verify that the script is running properly.
3. The tester will then need to see if an error log was output by the automation script, and then proceed to step 2.

Step 2: Error Reporting

1. The tester will need to send all errors over to the developers with comments explaining the errors found, and a general description of what it should be doing instead.
2. If no errors were found from the automation script, the tester should let the project manager know that quality assurance testing is done, and let the developers know that their code executed properly, and that the software is ready for deployment.