CS160 Assignment #4 - Testing

Deadline: 11/20, Tuesday, 11:59:59pm
Submission: All submissions will be via Canvas. Each team only needs to submit one .zip file under "Assignment -> Assignment 4" (see below for details).

Assignment Description

In this assignment, you will work on testing your system based on your implementation in Assignment 3. You are required to produce a working system with executable test cases and a testing document. Note that, since your implementation (Assignment 3) and testing (Assignment 4) are overlapping, you may choose to start working on this assignment immediately, or at some point after you feel your implementation becomes sufficiently stable.

You should test the "core" requirements. Remember that you will likely need to write multiple test cases for the same requirement in order to test it thoroughly. During testing, you will probably find bugs in your implementation. You should fix the identified bugs, i.e., you should change your implementation as a result. Your testing document should include those changes, and the reason for the changes. Please carefully document each change and turn in your "fixed" working system along with your testing document. We will grade this assignment based on (1) your test cases and the rationale; (2) the robustness of your "fixed" system on some randomly selected "core" requirements. This is further clarified below.

Instructions

You should use at least one white-box testing tool. Optionally, you can have one black-box testing tool (extra credit) of your choice to test your system. Some possibilities are following. If you are using external services/platforms that have built-in testing tools, you can use them as well, e.g. Firebase Test Lab.

Deliverable

You should submit a zip file containing the following items. We will test your "fixed" working system based on the test cases we create.

1. Your "fixed" frontend project (client-side).
2. Your "fixed" server project (server-side).
3. A README file to explain how to run your app if there are any extra steps.

4. Your testing document, which should include the information specified below.

Your testing document must have the following sections.

1. Project Title and Authors
    - Your team name
    - A list of all team members (names and SJSU ID numbers)
2. Preface
3. Instruction
    a. The instruction for executing your test cases
4. Black-box Tests
    - **You should have at least 15 executable black-box test cases.***
    - Each test case must have the following information
        i. The location of the test case (e.g. in which folder, which file, what is thename of the test case).
        ii. Description of what the test case does and how to execute the test case.
        iii. Description of the rationale behind the test case (e.g., Why do you choose the specific inputs? How do you generate the test case?).
        iv. The result of executing the test case (e.g., screenshot with description).
        v. If the test case helps you uncover a bug, document the details (e.g., What is the bug? How did you fix it?). Remember to do regression testing after fixing the bug, to make sure you did not break anything that was working previously.
5. White-box Tests
    - **You should have at least 15 executable white-box testcases.***
    - You should clearly identify the coverage criteria used in your white-box testing: What coverage objective(s) did you opt for and why? How did you ensure you are meeting your coverage criterion?
    - Each test case must have the following information
        i. The location of the test case (e.g. in which folder, which file, what is the name of the test case).
        ii. Description of what does the test case do and how to execute the test case.
        iii. Description of the rationale behind the test case (e.g., Why do you choose the specific inputs? How do you generate the test case?).
        iv. Result of the test case (e.g., screenshot with description).
        v. If the test case helps you uncover a bug, document the details (e.g., What is the bug? How did you fix it?). Remember to do regression testing after fixing the bug, to make sure you did not break anything that was working previously.

* There is nothing magical about the number 15. In fact, you will likely find that you need (many) more than 15 test cases to properly test your system.