Software Quality Assurance Plan

Submitted to:

Mr. Seidel

Submitted by:

Micheal
David
Kelvin

November 24th, 2020

Table of Contents

Acronyms & Abbreviations

HTML: Hyper Text Markup Language

CSS: Cascading Style Sheets

JS: JavaScript

SQA: Software Quality Assurance

SQA Process

SQA Records

The method of storing information will be in one centralized point, a word processing document such as Google Docs with a table of contents to jump to certain topics. This will store information such as to-do lists, to serve as an outline for what needs to be done in the project, and as storage for other notes and testing results. We will mainly use Repl.it's version history features to keep track of day to day development progress.

Outside feedback is taken with the free, reliable Google Forms. It brings all feedback together to display it in one place.

Software Documentation

We will adhere to the industry documentation standards. It involves ensuring functions are documented clearly in regards to purpose, parameters, returns, etc. In regards to specifics within programming languages, HTML, JS and CSS each have their own methods of documentation. We will utilize the documentation standards of the language we use accordingly (ex. Footers and headers with HTML).

The naming of variables will be clear and straightforward so one is able to read the code and figure without hassle what the variable represents. To achieve an outside perspective of readability, outside people will review the code (specifics below).

Graphical high level view (ex. Flowcharting) of how the program functions will be created to give the developers/reviewers a better understanding of the functionality, process and "paths" a user may take within the program.

Formal Reviews

Design Walk-throughs

This section will describe how you plan to have peers review your software (can be a different team member that does not normally work on that portion of the project as well).

We will be asking mutual friends through different social media platforms such as discord, instagram to test out our program. The way the testing will work is we will have our mutual friends go through certain aspects of our website and have them fill out a survey that will include the following things.

1. Is the UI appealing to look at
2. Is the website easy to navigate
3. Is the primary function of the website easy to use
4. Is the information easy to understand
5. Anything else the website can do to be improved that was not mentioned previously

Code Walk-throughs

- We would share our code with 3-5 people and have them look over our code. We would try to find a variety of people with different skill levels in coding. After having them review our code, we would point them to several different functions/sections and ask simple questions about the code such as:
    a. What does this function do?
    b. What are the inputs of this function?
    c. What do you think would happen if we remove this function from the code?

  If even an unskilled programmer understood the basic ideas of our function, our code is clear and our documentation worked effectively. This walkthrough will be done using a google form or in-person (socially distanced though).
- We would ask each person separately which part of the code looks/seems the most complicated. It doesn't matter if the person understood the complicated code, it just helps us determine where we can still improve on. This question helps us find the weakest link in the code.
- Whenever, the person gets confused about a question we ask/ doesn't understand a piece of code, they have the option to report it to us, so we can further improve the readability/ documentation. If testers find any overly complicated code or know a better way to do something, we will also take that information into consideration when we fix any bugs/ documentation issues.
- We will also ask them to pick any part of our code and get them to explain it to us (eg. what the different parts of the function does). This will further help us determine if our documentation was effective.

Formal Testing

Unit Tests

This section will describe how you plan to ensure that individual units (i.e. classes, methods, etc.) are functioning properly.

We will always test that all units are 100% working via assertions, logging, ect. All the testing of individual units will be under a header labeled Testing “Insert Individual Units” and under would be all the assertions logging, and/or any other testing we do.

Integration Testing

This section will describe how you plan to ensure that the different portions of code that each member of your team is working on will integrate properly without causing issues on either side.

We will test 4 times in total, the 4 times we test will be based on how finished we are based on the portion of code we decided to complete. This will help us make sure that the code we write is compatible with one another and that there are no major issues. We will also type all our variables in the code as a comment to ensure that we are not reusing the exact same variables. We will add headers and footers to organize different sections of our code along with documentation to easily integrate other things into our code. Different html files will be used so that different subpages in the website will not have to rely on each other to function properly.

Validation testing

We will have a support/ contact us section on the website. This will allow any user to quickly contact us about any issues/ problems that they encounter via email or google form. All this information will then be centralized in a google docs (SQA records) with headers (so we can quickly jump to different sections). To track our bug fixes, we can use repl.it’s built in version history feature. We will keep track of our bug fixes and record them onto our SQA records.. For example, we would add “Issue with buttons not working fixed in revision ‘November 25 4:55’” when we solve the bug/problem (the date/time can be used to track down different revisions via repl.it’s version history). Other than the regular users, we will have a specialized group of testers that will help us find issues. These people will use our design-walkthroughs, code-walkthroughs, unit testings, and integration testings to help us improve our site.