Test Plan for Quality Assessor

Authors: Marwa Hachem, Iman Saleh, Connor McDonough, Farzana Fariha, Samara Sarmiento

CIS 375 | Fall 2019

Revision Chart

Version	**Primary Author(s)**	**Description of Version**	**Date Completed**
1	Marwa Hachem, Iman Saleh, Farzana Fariha, Connor McDonough, Samara Sarmiento	Document draft prepared before peer reviews.	12/10/19
2	Marwa Hachem, Iman Saleh, Farzana Fariha, Connor McDonough, Samara Sarmiento	Document draft prepared before peer reviews.	12/12/19
3	Marwa Hachem, Iman Saleh, Farzana Fariha, Connor McDonough, Samara Sarmiento	Final Document Revisions	12/16/2019

TEST PLAN SPECIFICATION for QUALITY ASSESSOR

Marwa Hachem, Iman Saleh, Farzana Fariha, Connor McDonough, Samara Sarmiento

1.0 Introduction

This section provides an overview of the entire test document. This document describes both the test plan and the test procedure.

1.1 Goals and objectives

Overall goals and objectives of the test process are described.

We are heading towards the release of the final product of our project and we must focus on testing our software. We test out software to ensure that it is working correctly. Therefore, for a product to work correctly, it is meant to act as it has been intended to. To do this, the software is tested to achieve as close to error free as we can make it. This high quality can be achieved for our software by utilizing a detailed test plan document for the quality assessor. The test cases listed will be followed to test all of the necessary objects, data flows, limits, boundaries and constraints of the quality assessor software.

The test plan document will lay out what steps are needed to be taken in order to overcome difficulties that impact development and performance of the software. As a team, we will work to develop strategies to deal with errors, and these errors will range from uncommon to very common errors. Along with ensuring that the software is working as intended and ensuring proper performance, we are utilizing the test plan to ensure that the software meets the demands/requirements of the company.

1.2 Statement of scope

A description of the scope of software testing is developed. Functionality, features, behavior to be tested is noted. In addition, any functionality, features, behavior that is not to be tested is also noted.

Below are the different kinds of functionalities, features, and behaviors the team will be testing:

- Unit Testing:
  - Desktop Browser(s)
  - Home tab
  - Help tab
  - Pre-Questionnaire tab
  - Questionnaire tab
    - Results tab

- Integration Testing:
    - Desktop Browser(s)
    - Home tab
    - Pre-Questionnaire tab
    - Questionnaire tab
        - Results tab
- Validation Testing:
    - Desktop Browser(s)
    - Home tab
    - Help tab
    - Pre-Questionnaire tab
    - Questionnaire tab
        - Results tab
- High-Order Testing:
    - Desktop Browser(s)
    - Home tab
    - Help tab
    - Pre-Questionnaire tab
    - Questionnaire tab
        - Results tab
- Regression Testing
    - Questionnaire tab
        - Next/Back Buttons
    - Results tab

1.2.1 User Stories / Use Cases

Use Case 1: User can enter the site
Primary Actor: Website Visitor / User
Preconditions: A user must have an internet connection
Description: As a website visitor/user I want to be able to view the Home page of the Quality Assessor Tool that describes the purpose of the website. Upon landing on the website, I want to see a navigation page with all the web pages on the top of the page.
Acceptance Criteria: The Home page has been loaded up, the website description is displayed.

Use Case 2: User can start the questionnaire
Primary Actor: Website Visitor / User
Preconditions: A user must have internet connection, and either selected the assessment tab or clicked begin assessment on the home page.

Description: As a website visitor/user I want to be able to begin the Quality Assessment. I want to begin the first page of questions to begin assessing my website.
Acceptance Criteria: The assessment has begun, the first page of the questionnaire has loaded up.

Use Case 3: Going through the questionnaire
Primary Actor: Website Visitor / User
Preconditions: A user have have an internet connection, and have filled out the information to begin the questionnaire
Description: As the visitor/user begin the questionnaire they will be given questions to answer (in forms of a scale from 1-5). Each section will have 3-4 questions with a total of 9 sections. The user will be able to go back to a section using the back button and will be able to move on to the next section with the next button.
Acceptance Criteria: Answer the user enters are stored and accurate.

Use Case 4: Confirmation Page
Primary Actor: Website Visitor / User
Preconditions: The assessment must have been completed to reach this stage.
Description: The confirmation page will consist of a score on the radio button.
Acceptance Criteria: A selected radio button is displayed for each question in all sections and are editable by the user.

Use Case 5: Results Page
Primary Actor: Website Visitor / User
Preconditions: The assessment must have been completed to reach this stage.
Description: The confirmation page will consist of a numeric score based on the answers of the questionnaire.
Acceptance Criteria: A score for each section is displayed as well as an overall score.

Use Case 6: Navigation through the site
Primary Actor: Website Visitor / User
Preconditions: The user must have access to each page on the site and navigate through them with a navigation bar.
Description: As the visitor/user enters the site, they must be able to move through the pages from the navigation bar. Each link will redirect to appropriately named page.
Acceptance Criteria: The proper page has come up after a user clicks on it in the navigation bar.

1.2.2 Inputs

1. The user will select a choice between 1-5 per question
2. The user can select "N/A" as a choice
3. The user can click to go back
4. The user can click to go forward
5. The user can click to print the results
6. The user can click to restart the assessment
7. The user can click the navigation bar to go to
    a. Home, Assessment, Help pages

1.2.3 Outputs

1. Displays the user's answers (confirmation)
2. Display the user's results
3. Display a numeric score result
4. Display feedback on the user's website
    a. In the form of a grading from low quality to high
5. Display the page the user clicked on the navigation bar
6. Displays website home page

1.3 Major constraints

Any business or product line constraints that will impact the manner in which the Quality Assessor Tool is to be specified, designed, implemented or tested are noted here.

First, the Quality Assessor Tool faces many constraints that will impact every aspect of the product. The tool will be designed and implemented by a team of five members. From this, there is bound to be disagreements and at least one disappointed party with any given result that the project requires. Especially with subjective aspects such as design, a divergence in opinion is expected. In turn, this will impact the design and implementation of the tool which may negatively reflect on the product as a whole (making it a product line constraint).

Another major constraint that we face that impacts the overall product is time. With the given amount of time from the stakeholder, the team has many integral aspects of the product that are expected to be complete and working at the end of the period. The entire working product is expected on December 17, which gives the team roughly 4 weeks to compile a test plan and the product itself. To add to the tight time constraint, the team is inexperienced with web development. Of course, this will only make the time frame even tighter because everyone will need to learn the languages that come with creating websites and implement them effectively. Overall, time is a very impactful constraint that the team is aware of and will surely impact the product itself.

2.0 Test Plan

This section describes the overall testing strategy and the project management issues that are required to properly execute effective tests.

2.1 Software (SCIís) to be tested

The software to be tested is identified by name. Exclusions are noted explicitly.

2.1.1 Interfaces

Home Page:
When the Start button is clicked the user will be moved to the questionnaire tab of the site

Pre-Questionnaire Page:
The user will be asked to input a full name and website URL begin they will be allowed to use the begin button. We will test several different characters, which includes but is not limited to special characters, long strings of characters, numbers and unsupported characters. For the website URL we will test similar parameters but will test for "www." and for a domain suffix. We will test both valid and invalid inputs for the input boxes to verify the integrity of the begin button. When the begin button is successfully clicked the user will be moved on to the beginning of the questionnaire.

Questionnaire Page:
The questionnaire page(s) have multiple sections which all operate the same. Each section will have roughly 3 questions. Each with a radio button with 6 options, 1through 5 (disagree to agree) then 6 is N/A. By default the radio button is set to 6 (N/A). To test the radio buttons we will select 1-5 for our disagree/agree tests and then 6 for our N/A test. The progress bar will be tested by going through each section. When the back button is clicked the user will be moved back one section. When the next button is clicked the user will go to the next section.

Confirmation Page:

When the edit button is clicked the user will be able to edit any of the radio buttons displayed. When the submit button is clicked the user will be moved to the results page.

Results Page:
When the restart button is clicked the user will be sent to the Pre-assessment page. When the print button is clicked a printable document will be generated for the user to print/save as PDF.

Help Page:
When the 'help' tab is clicked, the user will be taken to the Help page. The user will be able to access the 'help' page tab through any other page they are on. The page will load successfully.

2.2 Testing strategy

The overall strategy for software testing is described.

2.2.1 Unit testing

The strategy for unit tested is described. This includes an indication of the components that will undergo unit tests or the criteria to be used to select components for a unit tests. Test cases are NOT included here.

As previously outlined in our Software Specification Document, each page on our Website Quality Assessor Tool will be a component. As such, the Home Page, Questionnaire Page, Results Page, and the Help Page will each be a unit or component to be tested. We will use the necessary stubs and drivers to properly test each individual component.

The Home Page will be tested for it's button "Start Questionnaire" to see if it successfully redirects the user to the appropriate page.

To further break down the Questionnaire Page, we will test each sub component of it. In other words, each section that the website assesses (content, function, structure, usability, navigability, performance, compatibility, interoperability, and security) will be individually tested as sub-components. Each section has about 3 questions, so each unit test will gauge whether these 3 questions answered will be stored and added to the overall score. These sub components can and should be tested in parallel. This will test the temporary data structure's integrity and ability to hold data--data must be able be able to flow between components.

The unit that is the Results Page is to see if the overall questionnaire is garnering the numeric score we expect. Yet, so many smaller parts build up to this page which is why it's important to test each set of questions to get this unit.

Important control paths will also be tested here such as any path from the current page to another through the navigation bar. Also, each path should be tested at least once. Since there are 6 options for each of the 27 questions, the total tests needed to test the questionnaire thoroughly would be 296,010 possibilities. In our case it's imperative to test critical units and and the components with "high cyclomatic complexity" which would be the questionnaire itself (Pressman & Maxim). So, the team will simply test the functionality of each section (as mentioned) and the ability of each button in that sub section. For sure, the team would test if the user chooses N/A for each question as a possibility. Basically, the aim is to test the error-handling paths first and foremost. More on specific tests will be highlighted in section **3.2.1 Unit Test Cases**.

The overall strategy for the unit tests on the websites is to break down the website into its smallest parts. Since the Home and Help Pages are mostly blocks of text with not much interaction, it's important to focus on the main goal of our website: the questionnaire. It must be reachable and it must garner results we expect (which will be addressed in the unit tests). We will know each test passes if they follow our expected outputs of the system.

2.2.2 Integration testing

The integration testing strategy is specified. This section includes a discussion of the order of integration by software function. Test cases are NOT included here.

The strategy for integration testing on the Website Quality Assessor is to focus on the interfacing and overall flow. Since there are multiple programmers working on this website, we need to ensure unity in our work and the overall interface. This means we need to clearly outline what is expected design wise (in our Software Design Specification) and follow these guidelines. Not only this, we need to ensure that no data is lost between the flow through the units. When it comes to data, we need to make sure that inputs and outputs are what we expect. Each input has an expected output outlined in this as well as the SDS Documents. We know if a test passes integration testing if it looks the way we expect.

In our case, we'd approach integration testing in the incremental manner. More specifically, top-down integration testing. We'd start off with the main program (home page with a nav bar) and move on depth-first into the questionnaire. Working on each component and moving on to interfacing it with the main program. Re-executing the tests will be necessary to ensure the program withstood the change that comes with interfacing.

2.2.3 Validation testing

The validation testing strategy is specified. This section includes a discussion of the order of validation by software function. Test cases are NOT included here.

Validation testing begins after the individual components have been tested. The interface errors have been corrected. You start at the validation level where the software categories do not exist anymore. Validation testing of software functions are considered successful when the results are what the customer expected. This is achieved through a series of tests that demonstrate the requirements.

The strategy for validation testing will be comparing the website to the examples and criteria that were found in some of our resources. We will use the validation testing for our quality assessor with the requirement we were given. The additional requirements and checklist for an assessor tiny tool will be found from Pressman and Maxim's Software Engineering: A Practionier's Approach. We will use these requirements to verify and validate the information our tiny tool presents is what is required.

2.2.4 High-order testing

The high-order testing strategy is specified. This section includes a discussion of the types of high order tests to be conducted, the responsibility for those tests. Test cases are NOT included here.

The teams approach to high order testing is to focus on deployment testing. To test our overall website, we need to ensure that it is operable across multiple operating systems as well as other browsers. For the Quality Assessor Tool, security testing may be implemented, but will not be a priority since the webapp doesn't store any private data. Similarly, stress testing will not be as necessary since the system doesn't do much heavy computation. Yet, it may be initiated by coupling with performance testing and attempting to overwhelm the questionnaire section. This section is the most involved with computation related tasks, and so our strategy would be to focus our efforts in terms of high order testing on its cohesion with the rest of the system.

2.2.5 Regression Testing

After each component is individually tested, the team will perform integration testing to see if the website components interface properly. The regression test will further uncover any errors due to the interfacing and will reveal any impacts the components had on each other while interfacing them. As such, all the existing features and functionality will be tested thoroughly.Any defects found in this process would be due to the interfacing of the components. Many of the previous tests will be repeated by nature to ensure no errors are found. The team will prioritize any test cases that are imperative to executing the requirements. For example, the requirement that the user can 'go through the questionnaire' is a priority since it's the basis of our entire project. The test cases related to this requirement will be tested once more to uncover any errors. In addition to this, the results page needs to function as expected after interfacing. This is also a priority and will be used in our regression tests.

2.3 Testing resources and staffing

Specialized testing resources are described and staffing is defined. The role of any ITG is also defined.

The team will be responsible for all testing. To the best of their abilities, each member will attempt to cover all the important control paths. Since the tool is a questionnaire with so many questions, there are countless paths that should be tested. For now, with the prototype, the team will cover as many strategic tests as possible. In the future, an ITG may also come into play, but for a prototype, it will not be necessary.

2.4 Test work products

The work products produced as a consequence of the testing strategy are identified.

The main test work products that will be produced by the team is a Test Plan (this document) in its completion and within it, a detailed strategy and execution goal. The Test Plan will be presented. In addition to this, we will also have a resultant time log of our work. To top it all off, we will have a prototype to show for our test strategy.

2.5 Test record keeping

Mechanisms for storing and evaluating test results are specified.

Record keeping for the testing will be stored in a local storage. As well as being backed up into a harddrive with a 1TB size. For when it comes to the evaluation, a spreadsheet will be created to compare the results at multiple times. And the comparison will be actual results vs. expected results. The spreadsheet can be accessed by all team members. This spreadsheet will be important when it comes to evaluating which results were the best.

2.6 Test metrics

A description of all test metrics to be used during the testing activity is noted here.

Throughout the entire testing phase, all testers will check for expected versus actual results per a given test case. With each failure, the tester will run each test a minimum of five (5) more times and record the results. This would be updated in a spreadsheet that will be labeled per test case. The tester will then compute the number of test cases passed over total test cases to determine percent error. The tester will create more additional tests if the minimum does not help create a solution for an error.

2.7 Testing tools and environment

A description of the test environment, including tools, simulators, specialized hardware, test files, and other resources is presented here.

The tools for testing will involve the computer resources from our own personal inventory. We will also collect resources from the server of the University of Michigan Dearborn to host our webpage. We will be using websites from random colleges to use as “mock” examples for the tiny tool.

2.8 Test schedule

A detailed schedule for unit, integration, and validation testing, as well as high order tests, are described.

- **Project Test Plan**: 12/04/2019 - 12/10/2019

  - This part is is the theory stage. No real actions will be performed. Test plan will be presented for review.
  - The Home page is complete with a functional button and navigation bar. Proper test cases initiated.
- **System Testing:** 12/06/2019 - 12/16/2019
  - Front end development began during the Project Test Plan theory stage to add in theory.
  - Remaining pages and components built and interfaced. All appropriate tests are mostly complete. Unit and Integration tests have begun
- **Generate Test Report**: 12/14/2019 - 12/17-2019
  - Generation of testing based upon completion of front/backend development
  - All tests are complete and test case tables are full of results.

3.0 Test Procedure

This section describes a detailed test procedure including test tactics and test cases for the software.

3.1 Software (SCIís) to be tested

The software to be tested is identified by name. Exclusions are noted explicitly.

Please refer to section 2.1 for Software (SCIís) to be tested.

3.2 Testing procedure

The overall procedure for software testing is described.

3.2.1 Unit test cases

The procedure for unit testing is described for each software component (that will be unit tested) is presented. This section is repeated for all components i.

3.2.1.2 Stubs and/or drivers for component i: Home Page

In order for Home Page to be tested and validated correctly, there must be a stub in place. The stub for our Home Page is the next page, which is the Questionnaire Page. In order to test if our home page is functioning correctly, we must use a “dummy” module in the place of Questionnaire page, since that page hasn’t been completed yet. Since we are using Top-Bottom approach, we are going to be using stubs and not drivers.

3.2.1.3 Test cases component i: Home Page

For the Home Page, we will be testing if the button will redirect users to the correct page once it's been clicked on. In our scenario, the correct page will be to begin the actual questionnaire assessment page.

3.2.1.4 Purpose of tests for component i: Home Page

The purpose of testing for the Home Page is to ensure that 1) the Home Page is launched correctly and 2)to ensure that the button on the Home Page was programmed to do what it is supposed to do. Since this is the first component/page in our Website Quality Assessor Tool, it's test cases are not much, but in order for everything else to work, this first page must work correctly.

3.2.1.5 Expected results for component i: Home Page

The expected results for the Home Page are to correctly launch the Questionnaire page after the user clicks the "Start Assessment Button". After this, then testing for the Questionnaire page can begin.

3.2.1.2 Stubs and/or drivers for component ii: Questionnaire Page(s)

Since the Questionnaire Page will be tested after completion of the first component, the Home Page, both a driver and a stub can be used for this testing. We have a top component already created, and we can use a bottom dummy component

3.2.1.3 Test cases component ii: Questionnaire Page(s)

For the Questionnaire Pages, we'll first be testing the "next" button to make sure that once it's clicked, it will show the correct next section as well as update the progress bar. Further into the questionnaire, we'll test that if we click the "back" button it will show the correct previous section as well as any previously answered questions. On every section page, we'll test to make sure that the rating buttons work to the correct corresponding answer.

3.2.1.4 Purpose of tests for component ii: Questionnaire Page(s)

The purpose of testing for the Questionnaire Page, is to ensure that all
1) the correct sections are shown when clicking the "next" and "back" buttons. 2) the correct rating buttons correspond to the correct answer and question. 3) that it is displaying the correct progress for the customer, and 4) the begin button directs to the start of the questionnaire. The

information we gather during the questionnaire pages is very important for Quality Assessor Tool, so it must be correct or else the results displayed at the end of the tool may be inaccurate.

3.2.1.5 Expected results for component ii: Questionnaire Page(s)

The expected results for the Questionnaire Page is all the correction sections launch when the corresponding buttons are clicked and the progress bar of the questionnaire is viewable and updates after every section.

3.2.1.2 Stubs and/or drivers for component iii: Results Page

Since the Results Page will be tested only after the completion of the questionnaire component, the questionnaire, both a driver and a stub will be used for this testing. Since we have already created a top component, we will use a bottom dummy component.

3.2.1.3 Test cases component iii: Results Page

For the Results Page, we will be testing that the results page loaded correctly and shows the correct calculations. We will also test that the restart, print, and send feedback buttons work correctly too.

3.2.1.4 Purpose of tests for component iii: Results Page

The purpose of testing the Results Page is to ensure that all the information gathered from the questionnaire is gathered correctly and shows the correct calculations to the user. This is where the user will be able to see the purpose of the Quality Assessor Tool, and to get an idea of which areas their site needs strengthening. The feedback option is important to us as it is critical to the website's development. If the Results Page doesn't work properly, this could affect the user's final results of the questionnaire and we won't be able to get the feedback to improve.

3.2.1.5 Expected results for component iii: Results Page

The Results Page loads correctly, displaying the correct results of the questions and showing the users a score out of 100 for each section. All the buttons work in order for the user to be able to restart the questionnaire, print out their results, and send in feedback of the website to us.

3.2.1.2 Stubs and/or drivers for component iv: Help Page

Since the Help Page will be made last and is for the most part a dead end that doesn’t take the user to any new parts of the site, also it doesn’t have anything functionality to test, it does not have stubs and or drivers.

3.2.1.3 Test cases component iv: Help Page

For the Help Page we will mainly only be testing if the information is displayed and the nav bar is working as designed. For the nav bar we would be testing to see if the buttons clicked go to the correct pages.

3.2.1.4 Purpose of tests for component iv: Help Page

The main purpose of testing the help page is to ensure that that information is displayed to help the user operate the website successfully. Since this is the help page if the user has any problems they will most definitely come to the help page for answers.

3.2.1.5 Expected results for component iv: Help Page

The expected results for the Help Page are to correctly display the help tips for the site and allow the user to go back to the other pages using the nav bar.

3.2.2 Integration testing

The integration testing procedure is specified.

3.2.2.1 Testing procedure for integration

After each unit test, the team plans to interface the working component to the rest of the program, this is referred to as ‘interfacing’. Only once we see that the newly interfaced item is not impacting the rest of the program in a negative way, we will move on.

The testing procedure for integration testing is to start off by interfacing our main program to the initial stage of the questionnaire. In other words, the team will attempt to add on to the Home Page a functioning button that will take them to the questionnaire. If the button successfully takes the user to the questionnaire, we move on.

From the questionnaire, the flow of data must be considered and tested thoroughly. With special attention to the flow of data, the team will test interfacing the separate pages of the questionnaire. If each page holds the data and its flows from one page to another properly, we can move on.

The flow of data is extremely important for this specific part: the Results Page. This page will tell us if the flow of data was successful or not. So, when we interface it, we must pay attention to if the data results change or not. If the results are what we expect, then we move on. (Recall: The results we 'expect' are the summation of each answer to the questionnaire. N/A is the only option without a number attached to it).

Finally, we must interface the Help Page, which can be done at any point. We only expect that the page loads up and doesn't impact any other page.

3.2.2.2 Stubs and drivers required

Since the Home Page is considered the main program, we will need a stub that is the dummy for the Questionnaire Page. The Home Page will have a button that redirects the user to the questionnaire, so to ensure this button works, we will create a placeholder page also known as a stub.

In terms of the Results Page, the driver needed is the questionnaire itself. The data flow from the questionnaire to the results is necessary. Thus, we need to create a placeholder version of the questionnaire (possibly values already set from the beginning of the program) to test the Results Page. Therefore, the driver required would be the false version of the questionnaire. It's the component that will give us an idea if the Results Page is working. So, even if we complete creating the Results Page first, we will need to use this dummy to test it.

Also, we will need a stub that is the Results Page to test if the questionnaire is working. In this case, the stub may simply show the sum of the questions as a dummy to see if each value is being assigned properly. So, upon the Questionnaire Page completion, we need a stub to test the results.

3.2.2.3 Test cases and their purpose

1. Home Button Redirect
    a. Purpose: To test if the Home Page's button will redirect the user to the Questionnaire Page. Testing the interfacing of adding the questionnaire to the main program.
2. Questionnaire Results Test

   a. Purpose: To test if the questionnaire and results are interfacing properly. Each component should behave as they would individually as they are combined to the website. Results should be a sum of all answers displayed on the Results Page. Need to ensure proper flow of data.

3. Help Page Display
   a. Purpose: To test if the interfacing of the Help page is successful and doesn't cause any errors. Accessible through navigation bar.

3.2.2.4 Expected results

1. Home Page button "Start Questionnaire" redirects user to Questionnaire Page. Interface Successful.
2. Results Page displays a score that is the sum of all the ratings in the questionnaire.
3. Help Page is loaded up properly and displays all pertinent information.

3.2.3 Validation testing

The validation testing procedure is specified.

3.2.3.1 Testing procedure for validation

We test for validation to ensure the content we provide is in fact true based on the requirements we provide. As well as, ensure the requirements in the Pressman and Maxim's Software Engineering: A Practitioner's Approach are also met.

The procedure for validation testing that we will follow is to ensure that we test various and diverse inputs to obtain a plethora of results. A mock website will be pulled up for testing and all pages will be chosen to be clicked upon. Different combinations of answers will be chosen for the survey when it comes to the questionnaire section.

3.2.3.3 Expected results

We expect there to be a wide range of results to users, based on different user answers. For example, a questionnaire that had selected "most likely" for the majority of the sections should receive a completely different score than a questionnaire that had a wide range of selected answers. In short, the results to the questionnaire should vary based on differencing user inputs/answers. Different user inputs should not receive similar answers.

We also expect that every page should link to the correct next page. For example, the Home Page should have a button to click to the next section, the Questionnaire should pop up. We do not expect the button to launch another section (for example, the Help Page).

These are all expectations that we have are supposed to mirror the requirements listed. We want the requirements to be shown in every validation testing of a software function within our software quality tool.

3.2.3.4 Pass/fail criterion for all validation tests

For our pass/fail criterion, we will check what is considered "passing" and "valid" for our assessment. The criteria for a pass in terms of system testing is that the performance of the Quality Assessor Tool is the same (or very close to it) across all tests.

<u>Testing for Different Results:</u>

Pass Criterion:

In order to know if the assessment results are valid, we must determine if different combinations of assessment answers yield different answers in the survey results. In order for this test to pass, different answers must yield different results for the websites.

Fail Criterion:

For this to fail, the results must be similar, or at least, for different completed questionnaires, the results for two different questionnaires should be in the same range.

<u>Testing for Valid Links:</u>

Pass Criterion:

In order for this test to pass, we must check for if all of the links, buttons, etc, including those on the navbar, deliver to the correct, next page. The buttons should have the correct heading and should launch the correct page based on their headings.

Fail Criterion:

In order for this test to fail, that would mean that at least one button or link did not do what it's supposed to do. That would mean, for example, if the user completed one subsection of the questionnaire and clicked on a button to take him/her to the next subsection, the questionnaire would fail and send him/her to the wrong link (could be the page before the current page s/he's on). If the buttons/links do the wrong thing, the test fails.

3.2.4 High-order testing (a.k.a. System Testing)

The high-order testing procedure is specified. For each of the high order tests specified below, the test procedure, test cases, purpose, specialized requirements and pass/fail criteria are specified. It should be noted that not all high-order test methods noted in Sections 3.2.4.n will be conducted for every project.

3.2.4.1 Deployment testing

The procedure for deployment testing is to test if the Quality Assessor Tool will be able to deploy on different environments such as platforms and operating systems. To be specific, we will test on Windows and IOS operating systems to see if the website will perform the same way. In addition to this, we will test across multiple browsers for the same. We will test on Google Chrome and Mozilla Firefox.

3.2.4.2 Performance testing

The team will test the performance of the tool in a cautious manner. It must be noted that performance of a website can oftentimes be attributed to the user's internet connection. So, to avoid such an issue, each member of the team must perform their individual test of the system and note any troublesome paths. Once a list has been complied, we must test these paths another 3 times to ensure that its performance is indeed lacking.

3.2.4.3 Alpha/beta testing

The team will reach out to outside users to alpha and beta test their Quality Assessor Tool. For the alpha tests, at least one team member will be present and note any errors in the product. This way, the team can see what a typical user would do with our tool, and how they interpret the instructions. As for the beta tests, we will entrust end users to give us a list of errors that they encounter while accessing the tool. We will take this information as a way to improve the prototype in future versions.

3.2.4.4 Pass/fail criterion for all validation tests

The criteria for a pass in terms of system testing is that the performance of the Quality Assessor Tool is the same (or very close to it) across all tests.

For example, the deployment test will be started on Google Chrome (where we build and integrate the webapp) and will be the basis of comparison throughout the tests. A pass will be that the webapp execute the same way it did in all our previous tests. A fail will be any inconsistencies from different environments.

As for the performance tests, we will each try out our own execution of the same control path. Each member will note any issues with performance. In a meeting the team will discuss these errors, and if two or more members note an area for lack of performance, we will test the problem area thrice more. If the performance is an issue each time, the test case will be a fail. Otherwise, it's a pass.

For alpha and beta testing, we will ask 2 users for each test to use the site without any guidance. So, in the alpha test, a pass would be no notable errors when watching the user go through the site. If the user misinterprets the instructions at any given time, it will be noted. Since it would just be a change in phrasing, this will count against our User Experience, but won't necessarily be a fail. For the beta test, the user will report any errors they find prevalent. For any error that has been noted by both users, then the team will have a meeting to see what is the root of the error. It will essentially be a case-by-case basis if the test will be a pass or fail in that case.

3.2.5 Regression Testing

The team's strategy for approaching regression tests is to do so with the goal of finding errors. The whole purpose of regression testing is to find any error that we may have missed in the previous tests. For this reason, a lot of the test cases are repeated: we're trying to see if any error from the previous tests to this one. As mentioned in the procedure, the team will prioritize which tests are absolutely important to fulfill the requirements that our website was built upon. In our case, it would be the navigation through the site, the going through the questionnaire, and the results page use cases. We will test the interfacing of the other components in this section, but focus on the mentioned requirements. Thus, please see previous tests for expected results and validation criteria as these have not changed.

3.3 Testing resources and staffing

Specialized testing resources are described and staffing is defined. The role of any ITG is also defined.

We will use the help from several different people. We will have the members staff record any errors found in the software and will correct them before the delivery of the software. The team will be responsible for all testing. To the best of their abilities, each member will attempt to cover all the important control paths. Since the tool is a questionnaire with so many questions, there are countless paths that should be tested. For now, with the prototype, the team will cover as many strategic tests as possible. In the future, an ITG may also come into play, but for a prototype, it will not be necessary. As mentioned in the High Order Testing section, we will ask the average user to test our website and see for ourselves what can be improved (and ask for feedback). These volunteers will not perform any extensive tests and we will limit them to about 2 people.

3.4 Test work products

The work products produced as a consequence of the testing procedure are identified.

The main test work products that will be produced by the team for the testing procedure is a Test Plan (this document) in its completion and within it, an outline of test cases and execution. The Test Plan will be presented. In addition to this, we will also have a resultant time log of our work for each test case/use case. To top it all off, we will have a prototype to show for our test procedure and carrying it out.

3.5 Test record keeping and test log

Record keeping for the testing will be stored in a local storage. As well as being backed up into a harddrive with a 1TB size. For when it comes to the evaluation, a spreadsheet will be created to compare the results at multiple times.

We will use a table created in excel to log all the tests, describe them and to record the results of the tests. We will label which test is being used and who is conducting them. The pass/fail result of the tests will be shown as well.

3.5.1 Test Case Table

All appropriate test cases from the previous sections compiled into a test case table.
Recall the use cases:
Use Case 1: User can enter the site
Primary Actor: Website Visitor / User
Preconditions: A user must have an internet connection

Description: As a website visitor/user I want to be able to view the Home page of the Quality Assessor Tool that describes the purpose of the website. Upon landing on the website, I want to see a navigation page with all the web pages on the top of the page.
Acceptance Criteria: The Home page has been loaded up, the website description is displayed.

Use Case 2: User can start the questionnaire
Primary Actor: Website Visitor / User
Preconditions: A user must have internet connection, and either selected the assessment tab or clicked begin assessment on the home page.
Description: As a website visitor/user I want to be able to begin the Quality Assessment. I want to begin the first page of questions to begin assessing my website.
Acceptance Criteria: The assessment has begun, the first page of the questionnaire has loaded up.

Use Case 3: Going through the questionnaire (next/back & answers)
Primary Actor: Website Visitor / User
Preconditions: A user have have an internet connection, and have filled out the information to begin the questionnaire
Description: As the visitor/user begin the questionnaire they will be given questions to answer (in forms of a scale from 1-5). Each section will have 3-4 questions with a total of 9 sections. The user will be able to go back to a section using the back button and will be able to move on to the next section with the next button.
Acceptance Criteria: Answer the user enters are stored and accurate.

Use Case 4: Confirmation Page
Primary Actor: Website Visitor / User
Preconditions: The assessment must have been completed to reach this stage.
Description: The confirmation page will consist of a score on the radio button.
Acceptance Criteria: A selected radio button is displayed for each question in all sections and are editable by the user.

Use Case 5: Results Page
Primary Actor: Website Visitor / User
Preconditions: The assessment must have been completed to reach this stage.
Description: The confirmation page will consist of a numeric score based on the answers of the questionnaire.
Acceptance Criteria: A score is displayed.

Use Case 6: Navigation through the site

Primary Actor: Website Visitor / User
Preconditions: The user must have access to each page on the site and navigate through them with a navigation bar.
Description: As the visitor/user enters the site, they must be able to move through the pages from the navigation bar. Each link will redirect to appropriately named page.
Acceptance Criteria: The proper page has come up after a user clicks on it in the navigation bar.

Unit Tests				
Use Case Number	**Purpose**	**Expected Output**	**Actual Output**	**Pass/ Fail**
UC 1	Home page: We will be testing if the "start" button will redirect users to the assessment page once it's been clicked on.	Redirect to Assessment page.	Redirect to Assessment page.	Pass
UC 2	Questionnaire Page: we'll first be testing the "next" button to make sure that once it's clicked, it will show the correct next section as well as update the progress bar.	Next button redirects to the next section of the assessment.	Next button redirects to the next section of the assessment.	Pass
UC 3	Questionnaire: we'll test that if we click the "back" button it will show the previous section	Back button redirects to the previous section of questionnaire.	Back button redirects to the previous section of questionnaire.	Pass
UC 3	Questionnaire: any previously answered questions are the same when user clicks the "back" button.	Answers to previous section are the exact same when "back" button is clicked.	Answers to previous section are the exact same when "back" button is clicked.	Pass
UC 3	Questionnaire: on every section page, we'll test to make sure that the rating buttons work to the correct corresponding number.	Strongly Agree = 5 Agree = 4, Neutral = 3, Disagree = 2, Strongly Disagree = 1, N/A = 0	Strongly Agree = 5 Agree = 4, Neutral = 3, Disagree = 2, Strongly Disagree = 1, N/A = 0	Pass
UC 5	Results Page: We will be testing that the results page loaded correctly and shows the correct calculations.	The calculations on the results page is generated correctly.	**In progress**	**In progress**

UC 6	Help Page: we will mainly only be testing if the information is displayed and the nav bar is working as designed.	All tips and commonly asked questions are displayed and loaded correctly. Also the nav bar has working links to the rest of the site.	All tips and commonly asked questions are displayed and loaded correctly. Also the nav bar has working links to the rest of the site.	Pass
UC 6	Help Page: For the nav bar we would be testing to see if the buttons clicked go to the correct pages.	The nav bar has working links to the rest of the site.	The nav bar has working links to the rest of the site.	Pass
Integration Tests				
Use Case Number	**Purpose**	**Expected Output**	**Actual Output**	**Pass/ Fail**
UC 1	Home page: We will be testing if the "start" button will redirect users to the assessment page once it's been clicked on.	Redirect to Assessment page.	Redirect to Assessment page.	Pass
UC 2	Questionnaire Page: we'll first be testing the "next" button to make sure that once it's clicked, it will show the correct next section as well as update the progress bar.	Next button redirects to the next section of the assessment.	Next button redirects to the next section of the assessment.	Pass
UC 3	Questionnaire: we'll test that if we click the "back" button it will show the previous section	Back button redirects to the previous section of questionnaire.	Back button redirects to the previous section of questionnaire.	Pass
UC 3	Questionnaire: any previously answered questions are the same when user clicks the "back" button.	Answers to previous section are the exact same when "back" button is clicked.	Answers to previous section are the exact same when "back" button is clicked.	Pass
UC 3	Questionnaire: on every section page, we'll test to make sure that the rating buttons work to the correct corresponding number.	Strongly Agree = 5 Agree = 4, Neutral = 3, Disagree = 2, Strongly Disagree = 1, N/A = 0	Strongly Agree = 5 Agree = 4, Neutral = 3, Disagree = 2, Strongly Disagree = 1, N/A = 0	Pass

UC 6	Help Page: we will mainly only be testing if the information is displayed and the nav bar is working as designed.	All tips and commonly asked questions are displayed and loaded correctly. Also the nav bar has working links to the rest of the site.	All tips and commonly asked questions are displayed and loaded correctly. Also the nav bar has working links to the rest of the site.	Pass
UC 6	Help Page: For the nav bar we would be testing to see if the buttons clicked go to the correct pages.	The nav bar has working links to the rest of the site.	The nav bar has working links to the rest of the site.	Pass
Validation Tests				
Use Case Number	**Purpose**	**Expected Output**	**Actual Output**	**Pass/ Fail**
UC 1	Home Page: To test if the user can enter the site and if the correct elements (which include the buttons) and descriptions is loaded.	The website has loaded and all elements of the page including it's descriptions have loaded in correctly. Buttons send user to the next page.	The website has loaded and all elements of the page including it's descriptions have loaded in correctly. Buttons send user to the next page.	Pass
UC 2	Questionnaire: To test if the user can begin the assessment that the first page (index page) has loaded up.	The assessment has begun, the first page of the questionnaire has loaded up.	The assessment has begun, the first page of the questionnaire has loaded up.	Pass
UC 3	Going through the Questionnaire: To test if the user is able to go through the questionnaire answering each of the questions with the corresponding radio button.	Answering each question with its corresponding radio button (0-5). The back button and the next button go through the questionnaire.	Answering each question with its corresponding radio button (0-5). The back button and the next button go through the questionnaire.	Pass
UC 4	Confirmation Page: To test the score being displayed on the screen of what the user selected.	The score of what the has selected is displayed on the page. The radio	The score of what the has selected is displayed on the page. The radio	Pass

		buttons are editable. The back button goes back to the last section and the confirm button goes to the results page.	buttons are editable. The back button goes back to the last section and the confirm button goes to the results page.	
UC 5	Results Page: To test that the correct results are generated for the user.	A properly generated score for each section is displayed as well as an overall score.	**In progress**	**In prog ress**
UC 6	Navigation through the site: To test that as a visitor/user enters the site, they are able to move throughout the pages from the navigation bar.	Each link/button is working as designed and no dead links are found. Navigation is smooth and natural for the user.	Each link/button is working as designed and no dead links are found. Navigation is smooth and natural for the user.	Pass
High Order Tests				
Use Case Number	**Purpose**	**Expected Output**	**Actual Output**	**Pass/ Fail**
UC 1 - 6 Mostly UC 6	Alpha test: Watch a user navigate the site to notice any errors or misunderstandings.	User navigates site with no guidance, and no errors occur.	**In progress**	**In prog ress**
UC 1 - 6 Mostly UC 6	Beta test: Ask user to navigate the site alone and report back any issues. Team will check upon these errors.	User navigates site with no issues or complaints.	**In progress**	**In prog ress**
UC 1 - 6	Deployment test 1: BASE TEST. We create site on Google Chrome, all comparisons are to this test for appearance/usability. Windows OS.	Site appears the same as implementation and previous tests. No visible errors.	Site appears the same as implementation and previous tests. No visible errors.	Pass
UC 1 - 6	Deployment test 2: Test on Mozilla Firefox to see if the website appears and functions the same way. Windows OS.	Site appears the same as it does in Google Chrome. No visible errors.	**In progress**	**In prog ress**
UC 1 - 6	Deployment test 3: Test on a different	Site appears the same	**In progress**	**In**

	operating system then implementation to see if function is the same. Test on IOS & Google Chrome.	as it does in Google Chrome on Windows. No errors.		**prog ress**
UC 1 - 6	Performance test: Each member tests the website on their own and notes any issue in function/bugs. Team meeting will be initiated to see any overlap in performance issues. Any issue will be further tested 3 more times. If an issue is still there, test is fail. Else, it's a pass.	Site encounters no errors in performance due to implementation. Any error simply because of user end issue.	**In progress**	**In prog ress**
Regression Tests				
Use Case Number	**Purpose**	**Expected Output**	**Actual Output**	**Pass/ Fail**
UC 1	Home page: We will be testing if the “start” button will redirect users to the assessment page once it’s been clicked on.	Redirect to Assessment page.	Redirect to Assessment page.	Pass
UC 2	Questionnaire Page: we’ll first be testing the “next” button to make sure that once it’s clicked, it will show the correct next section as well as update the progress bar.	Next button redirects to the next section of the assessment.	Next button redirects to the next section of the assessment.	Pass
UC 3	Questionnaire: we’ll test that if we click the “back” button it will show the previous section	Back button redirects to the previous section of questionnaire.	Back button redirects to the previous section of questionnaire.	Pass
UC 3	Questionnaire: any previously answered questions are the same when user clicks the “back” button.	Answers to previous section are the exact same when “back” button is clicked.	Answers to previous section are the exact same when “back” button is clicked.	Pass
UC 3	Questionnaire: on every section page, we’ll test to make sure that the rating buttons work to the correct corresponding number.	Strongly Agree = 5 Agree = 4, Neutral = 3, Disagree = 2, Strongly Disagree = 1, N/A = 0	Strongly Agree = 5 Agree = 4, Neutral = 3, Disagree = 2, Strongly Disagree = 1, N/A = 0	Pass
UC 5	Results Page: We will be testing that the results page loaded correctly and shows the correct calculations.	The calculations on the results page is generated correctly.	**In progress**	**In prog ress**

UC 6	Help Page: we will mainly only be testing if the information is displayed and the nav bar is working as designed.	All tips and commonly asked questions are displayed and loaded correctly. Also the nav bar has working links to the rest of the site.	All tips and commonly asked questions are displayed and loaded correctly. Also the nav bar has working links to the rest of the site.	Pass
UC 6	Help Page: For the nav bar we would be testing to see if the buttons clicked go to the correct pages.	The nav bar has working links to the rest of the site.	The nav bar has working links to the rest of the site.	Pass

Works Cited

"Integration Testing: What Is, Types, Top Down & Bottom Up Example." Guru99, https://www.guru99.com/integration-testing.html#1.

Pressman, Roger, and Bruce Maxim. "Software Engineering: A Practitioner's Approach." *McGraw-Hill*, 9 Sept. 2019, https://www.mheducation.com/highered/product/software-engineering-practitioner-s-approach-pressman-maxim/M9781259872976.html.

"Software Configuration Management in Software Engineering." Guru99, https://www.guru99.com/software-configuration-management-tutorial.html.

Testing, Software. "Functional Testing." Software Testing Fundamentals, 13 July 2018, softwaretestingfundamentals.com/functional-testing/.

"Unit Testing." *Software Testing Fundamentals*, 3 Mar. 2018, http://softwaretestingfundamentals.com/unit-testing/.