



Test Plan

Date Submitted: 10/27/2021

Team Hobby:

Colin Creasman

Daniel Bribiesca

Team Lead: Jacob Delgado

Long Nguyen

Rifat Hasan

Document History

Date	Version	Changes
10/14/2021	0.1	Created Doc with Initial Outline
10/15/2021	0.2	Drafted Initial Test Scenario with Specific Test Case
10/15/2021	0.3	Removed Specific Test Cases; Added Test Scenarios
10/19/2021	0.4	Removed Redundant Testing; Added Timeline

Table of Contents

Introduction.....	3
Scope.....	3
Project Goals.....	3
Testing	5
Test Scenarios.....	6

1 - Introduction

The website will be region locked to the United States of America and will be supporting American English only. The website will only support the latest Chrome version as of 05/20/22 and will not support beta versions. Our website is based on the tools a user has, so the website will display all measurements in metric units, but will also offer imperial conversions.

1.1 - In Scope

Here we need to list out all the system components that will be tested. In order to figure this out we need to know functional requirements, project's budget, and project specification. Typically this means all the functions of the external interface of the website.

1.2 - Out of Scope

Here we need to list out all the system components that will not be tested. Typically this means the nonfunctional requirements (performance, etc) are out of scope

1.3 - Quality Objective

Our website will not:

- Have any monetization or product recommendations.
- Support Software/coding related projects or skill sets.
- Support languages other than American English.
- Support the ASCII-Extended table, only the original 127 characters.
- Support professional level projects that require licensing to complete.
- Have an email system to recommend projects or give notifications to our users.
- Be supporting direct messaging between users.
- Responsive to other devices such as mobile

1.3 - Roles and Responsibilities

Our website will not:

- Have any monetization or product recommendations.

- Support Software/coding related projects or skill sets.
- Support languages other than American English.
- Support the ASCII-Extended table, only the original 127 characters.
- Support professional level projects that require licensing to complete.
- Have an email system to recommend projects or give notifications to our users.
- Be supporting direct messaging between users.
- Responsive to other devices such as mobile

2 - Testing Strategy

2.1 - Methodology Overview

We will be using an **Agile** testing methodology

2.2 - Test levels

Here we need to have a bulleted list to define all the test levels that will be executed for the Application Unit Test (AUT). Testing levels depend on scope, time, and budget. These will most likely be

- **Integration** Testing (Individual software modules are combined and tested as a group)
- **System** Testing: Conducted on a **complete, integrated** system to evaluate the system's compliance with its specified requirements
- **API testing:** Test all the APIs create for the software under tested
- **Unit Testing:**

3 - Test Scenarios

3.1 - Login

Verification

The functionality of the login feature is verified from the application homepage. Part of the application homepage should be centered around a large UI component. For login, it displays empty text fields for the user to input their username and password. ``

Validation

The functionality of the login feature is validated using the following test scenarios:

- **Valid Login** - The user provides valid credentials and is successfully logged in to the private homepage.
- **Invalid Login** - The user provides invalid credentials and is not logged in.

Priority

High - Aside from the public search and discovery page, all other application features are only available once a user is logged in, thus it is a **core component** of the application. This means that the login functionality is of high priority as many other features' functionalities are dependent on it.

3.1.1 - System (End-to-End) Level Testing

System-level testing will be required for both valid and invalid test scenarios because they will involve end-to-end communication across various system components such as the third party API, the UI components of the login feature, backend business logic, and the internal database.

Valid Login

- **Pass:**
 - The login service authorizes the user's credentials using matching credentials in the server's internal database.
 - The validated credentials are used to authenticate any additional requests made by the user while logged in.
 - The user is successfully logged in and taken to the application private homepage.
 - All login-specific features and UI components are now available to the user
- **Fail:**

- The system cannot authorize the correct credentials and fails to log in the user
- The validated credentials cannot be used to authenticate any additional requests.
- The system is unable to verify matching credentials for the user in the system's internal database.

Invalid Login

- **Pass:**
 - The user is not logged in and is redirected back to the login page to try again.
 - An error message is displayed indicating the specific login error that occurred (incorrect password/username, etc).
 - A link to reset username/password is displayed to the user
- **Fail:**
 - No communication occurs with the system
 - The user is not redirected back to the login page to try again
 - No error message is displayed
 - No username/password reset link is displayed to the user

3.1.2 - Timeline

Because the login functionality is a core component of the system, it must undergo exhaustive testing before any of the other features that depend on it can be tested. This means that all testing for this functionality must be completed as early in the development lifecycle as possible.

Since the login feature is dependent on the user having an account to log in, in the first place, testing for this functionality cannot begin until the account creation and user questionnaire features have been fully tested first. Development and testing for these preliminary features will be completed during **Sprint 5** (01/20/22 - 02/03/22).

Testing for the login functionality will begin in **Sprint 6 (2/4/22 - 2/18/22)** and be completed before the start of Sprint.

3.2 - Logout

Verification

The functionality of the logout feature is verified after logging out of the user's account. Users should not be able to access any of the following login-specific features:

- Publish and delete project their own project posts
- Receive notifications
- Receive customized recommendations for project posts and tools
- Comment text and/or pictures on project posts
- Rate project posts

Logged out users will be able to view the same content as a non-registered user.

Validation

The functionality of the logout feature is validated using the following test scenarios:

- **Valid Logout**- The user is brought back to the homepage of the site and will be able to view the site as any other non-registered user.
- **Invalid Logout** - The user encounters an error when logging out and will still be logged in.

Priority

Low - The need to log out of an account is not as demanding as logging in. Most users would not need to use this feature unless they had multiple accounts, want to see the website as a non-registered user, or for personal reasons. Since users will not gain anything from logging out, this feature is not as essential as the other functional features.

3.2.1 - Integration Level Testing

Test Cases

- User intends to log out of account
 - Pass:
 - System must be able to successfully log out user out of account
 - System brings user back to homepage after logging out is successful
 - Fail:
 - User is not able to log out
 - System does not redirect user to homepage
- User exits out of application

- Pass:
 - User is still logged in regardless of inactivity/termination of website
- Fail:
 - User is automatically logged out with the termination of the website
- User is brought back to homepage after logging out
 - Pass:
 - All previously recommended items are hidden
 - User is now able to view the website as a non-registered user
 - All registered user perks are revoked from logged out user
 - Fail:
 - User is still able to comment, post, and use website as a registered user
 - Homepage contains personalized projects and/or project recommendations from user questionnaire

3.3 - Create Account

3.3.1 - System (End-to-End) Level Testing

System-level testing will be used to present the creation of accounts in order to view both the expected and failed cases that could potentially occur.

Invalid Creation of Account

Test Cases

- **Pass**
 - User does not follow the set guideline for a complete password
 - Username must match password
 - At least 1 special character
 - At least one uppercase letter
 - At least one 1 number
 - Character length of 8-16
 -
- **Fail**
 - No error message is displayed to the user when error is made
 -

3.4 - Questionnaire

Verification

After clicking the UI component to register on the New Account page, the user should be directed to a page to fill out the questionnaire. The functionality of the questionnaire feature is verified when the user sees the four UI components for each part of the questionnaire displayed on this page. Each of the four UI components should display a series of options with checkboxes that the user can interact with.

Validation

The functionality of the user questionnaire feature is validated using the following test scenarios:

- **Valid Selection** - The user fills out the questionnaire by selecting at least one checkbox in each of the four UI components. Then the user is able to click “done” to finish registering for their account.
- **Invalid Selection:** The user fills out the questionnaire without at least one checkbox marked in each of the four UI components.

Priority

High - Aside from the public search and discovery page, all other application features are only available after a user has created their account and completed the questionnaire, thus it is a **core component** of the application. This means that the questionnaire feature is of high priority as many other features’ functionalities are dependent on it.

3.4.1 - Integration Testing

Integration testing will be required for the user questionnaire because it involves communication between the UI components that fill out the questionnaire and the system’s internal database that will store the information as the user fills it out.

Valid Selection

- **Pass:**
 - The UI component for each checkbox successfully updates after being selected by the user.
 - The user is able to click the “done” UI component at the bottom of the page to finish registering for their account.
- **Fail:**
 - The UI component for each checkbox does not update after being selected by the user.
 - Users click “done” and nothing happens.

Invalid Selection

- **Pass:**
 - Users are unable to finish registering for their account after clicking “done”.
 - After attempting to finish their registration, the user is redirected back to the questionnaire page to try again.
 - An error message is displayed to the user to notify them that the questionnaire was filled out incorrectly.
 - The UI components of the questionnaire page with the invalid selections are highlighted to notify them of the errors.
- **Fail:**
 - Users click “done” and the system allows the user to register an account.
 - The user is not redirected back to the questionnaire page to try again.
 - No error message is displayed to the user.
 - The UI components with the invalid selection are not highlighted.

3.4.2 - Timeline

Because the questionnaire functionality is a core component of the system, it must undergo exhaustive testing before any of the other features that depend on it can be tested. This means that all testing for this functionality must be completed in the beginning of the development lifecycle.

Testing for the questionnaire feature will begin in **Sprint 5 (1/20/22 - 2/3/22)** and must be completed before the start of Sprint 6.

3.5 - Search

3.5.1 - Integration Level Testing

Valid Search

- **Pass:**
 - System directs user to a page that contains the keywords user used in search bar
 -
- **Fail:**
 -

Invalid Search

- **Pass:**
 -
- **Fail:**
 -

3.6 - Apply Search Filter

3.6.1 - Integration Level Testing

Valid Selection

- **Pass:**
 - User clicks on the “filter by” button and the drop-down menu appears.
 - User selects any of the offered filters and the projects become rearranged accordingly.
- **Fail:**
 - User clicks on the “filter by” button and nothing happens.
 - User selects any of the offered filters and nothing happens or they are redirected to the wrong page.

3.7 - Apply Discovery Filter

3.7.1 - Integration Level Testing

Valid choices

Test Cases

- **Pass:**
 - User clicks on the “filter by” button and the drop-down menu appears.
 - User selects any of the offered filters and the projects become rearranged accordingly.
- **Fail:**
 - User clicks on the “filter by” button and nothing happens.
 - User selects any of the offered filters and nothing happens or they are redirected to the wrong page

3.8 - Public UI Controls

3.8.1 - Integration Level Testing

Valid choices

Test Cases

- **Pass:**
 - User clicks on any publicly accessible button and it redirects to the correct page.
- **Fail:**
 - User clicks on any publicly accessible button and it does nothing or redirects to the wrong page.

3.9 - Private UI Controls

3.9.1 - Integration Level Testing

Valid choices

Test Cases

- **Pass:**
 - User clicks on any privately accessible button and it redirects to the correct page.
- **Fail:**
 - User clicks on any privately accessible button and it does nothing or redirects to the wrong page.

3.10 - Notification Alerts

3.10.1 - Integration Level Testing

Valid choices

Test Cases

- **Pass:**
 - User is properly alerted about a new comment.
 - User is properly alert about a new recommendation.
 - Users are properly alerted about their project's first upvote.
- **Fail:**
 - User isn't alerted at all.

3.11 - Notification Page

3.11.1 - Integration Level Testing

Valid choices

Test Cases

- **Pass:**
 - User is able to click on a notification and is able to fully expand it.
 - User has all of the correct notifications show up on the page.
- **Fail:**
 - User clicks on a notification and nothing happens.
 - User notifications are missing or not showing up at all.

3.12 - Post Project

3.12 - Upload a File to a Project

3.12.1 - Integration Level Testing

Valid choices

Test Cases

- **Pass:**
 - User is able to access their local storage to add a file
 - File is successfully embedded into website server to be published
 - Uploading a project file outside of accepted bounds will display error message
- **Fail:**
 - Adding file option does not return anything

- When clicking a project from local storage, no file is sent to server
- No error message is shown when user choose a file that does not fit within website guidelines

3.13 - Save Project Draft

3.13.1 - Integration Level Testing

Valid choices

Test Cases

- **Pass:**
 - Project is saved on server but is not viewable to other users
 - User can go back to previously drafted projects
- **Fail:**
 - Users cannot retrieve drafted projects
 - Project is not drafted and instead is published to the public
 - Project is not drafted and lost

3.14 - Edit a Posted Project

3.15 - Edit a Saved Project Draft

3.16 - Delete a Posted Project

3.17 - Delete a Project Draft

3.16 - Rating a Project

3.17 - Edit a Project Rating

3.17 - Posting a Comment

3.18 - Uploading a File to a Comment

3.19 - Rating a comment

3.20 - Edit a Comment Rating

3.21 - Filter Comments

3.20 - Receive a Project Recommendation

3.21 - Receive a Tool Recommendation

3.23 - Collect Project from External Source

3.24 - Display Collected Projects

3.25 - Archiving(what are we storing?)

3.26 - Logging (Errors)

3.27 - Data Store (Ex: Projects, Ratings, Comments)