**Draft Test Plan**

**Project description:**

The Public Portal is one of PageFreezer's tools. The public portal feature allows our customer to publish content from our archives and make it available.

**Objective:**

Assessing one of our tools and checking how it's working! Test this portal (at least its most basic features). Will not tackle all the use cases, but focus on the most important.

Manually complete exploratory testing to determine the data needed for locating web-elements .

This discovered data will be used for automated tests

The same web portal will also be subjected to automated testing

**Scope:**

Only basic usability testing will be completed

Three test cases derived from the requirements will be automated and run through UI test automation.

The rest of testing and testcases are currently out of scope.

**Environment and Resources:**

Windows based environment with Java sdk 1.8 installed

Test Public Portal URL: https://social.pagefreezer.com/openrecords/cityofsacramento

The portal is expected to have some relevant test data just like real data

All testing, manual-exploratory and automated, is completed using the current version of Chrome browser

**Tools:**

Current Selenium Web driver with Java language binding

Current TestNG version as the test harness and for wiring up Selenium tests

**Test cases:**

**All testing will start by opening the Public Portal home page at:**

<https://social.pagefreezer.com/openrecords/cityofsacramento>

**Test case 1: Requirement 1:**

- Users can input search words in the big search box and obtain results from the website platform and the Social Media platform;

**Objective**: To verify that

1. User is able to type in search words in the big search box
2. User is able to click the Search button
3. The results from valid steps 1 and 2 is a set of search results in the results area at the bottom of the page

**Steps to Reproduce for exploratory and then for Selenium automated test steps:**

1. On the portal home page click in the big search box and enter a search string that will result in zero results. Example: noSearchResultsWithThisStringAtAll
2. Click the search button
3. Wait for the page to get into steady state so that the page appears to have rendered the result of the search
4. Verify that the results area is empty and shows no results
5. Repeat step 1 with a search word that is sure to have archives. Example: sacramento
6. Wait for the page to get into steady state so that the page appears to have rendered the result of the search
7. Verify requirement 1

**Expected Results:** The results area should be populated with some results so that this is different from the no results baseline

**Actual Result:** The results area is populated with some results so that this is different from the no results baseline

**Test case 2: Requirement 2:**

- Users can access specific archives by clicking on them in the bottom part of the page;

**Objective**: To verify that archives listed by default at the bottom of the page are accessible and linked to archived content

**Steps to Reproduce for exploratory and then for Selenium automated test steps:**

1. On the portal home page examine the bottom of the page and locate a tab with the following tab label “Websites”
2. Click on the tab label to make it active
3. This action will result in showing the archived web links on the tab content section
4. Click on the first link
5. This will result in a new tab opening to display the archived content for the clicked link and gaining focus as the current tab
6. Verify requirement 2

**Expected Results:** The newly opened tab should display archived content

**Actual Result:** The newly opened tab displays archived content

**Test case 3: Requirement 3:**

- Users can select where they want to perform their search (All archives, Only social media, Only websites, Select specific websites/social media). They can do this using the dropdown to the left of the search box;

**Objective**: To verify that the portal allows user to select a setting to target their search performed from the main Search box

**Steps to Reproduce for exploratory and then for Selenium automated test steps:**

1. On the portal home page examine the label for the drop-down prepended(left) to the big search box
2. Verify that the default label for the drop-down is “Selected All”
3. Click on the drop-down and verify that you can see three options as radio buttons with “Selected All” as the Default selection:
   1. Select All
   2. All Social Media
   3. All Websites
4. Click on the “All websites” radio button
5. Notice that the label for the drop-down has changed to reflect the selection

**Expected Results:** The current label of the drop-down should be “All Websites”

**Actual Result:** The current label of the drop-down is “All Websites”

**Test Execution Results;**

One out of the four tests failed due to test cleanup failure

When the failed test was run separately, it passed, which confirms the tainted environment theory.

Console out put from the test execution is given below:

**First execution run of all tests as a suite:**

PASSED: a\_userCanDoBasicSearch

PASSED: c\_canSelectOnlyWebsitesToSearch

PASSED: d\_canUsePortalHomePage

FAILED: b\_canAccessDefaultWebsiteArchives

===============================================

Default test

Tests run: 4, Failures: 1, Skips: 0

===============================================

**Second execution of separate run:**

PASSED: b\_canAccessDefaultWebsiteArchives

===============================================

Default test

Tests run: 1, Failures: 0, Skips: 0

**Conclusion:**

All tests passed in verification of product functionality to verify the requirements; 1, 2 and 3

The one test failure has been verified as not being a product defect.

It has been logged a framework code bug and will be fixed as such.