Costco.com website UI automation

Prepared for

Ty – DemandDrivenTech.com – Take Home Test Interview

Prepared by

**Jiayue Hu (raymondhu6188@gmail.com)**

Revision and Signoff Sheet

Change Record

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Version | Change Reference |
| 2/12 | Jiayue Hu | 1.0 | Initial draft for review/discussion |
| 2/13 | Jiayue Hu | 1,1 | Add test cases, Dependences |
| 2/18 | Jiayue Hu | 1.8 | Final version |

Reviewers

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Version Approved | Position | Date |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1 Test Plan Summary 4](#_Toc506760325)

[2 Test Environment Specification 5](#_Toc506760326)

[2.1 The Test Team 5](#_Toc506760327)

[2.2 Test Planning 5](#_Toc506760328)

[2.2.1 Scope and Objectives 5](#_Toc506760329)

[2.2.2 Schedule 5](#_Toc506760330)

[2.2.3 Risks 6](#_Toc506760331)

[2.3 Documentation Requirements 6](#_Toc506760332)

[3 Test Subject and Resource Requirements 7](#_Toc506760333)

[3.1 Features and Functionality to Test 7](#_Toc506760334)

[Functionality test cases: 8](#_Toc506760335)

[3.2 Expected Results of Tests 12](#_Toc506760336)

[3.3 Test Resource Requirements 12](#_Toc506760337)

[3.3.1 Environmental Needs 12](#_Toc506760338)

[3.3.2 Test Tool Requirements 12](#_Toc506760339)

[4 Testing Walkthrough 14](#_Toc506760340)

[5 Reporting 15](#_Toc506760341)

[5.1 Tracking and Reporting Status 15](#_Toc506760342)

[5.2 Bug Reporting Tools and Methods 15](#_Toc506760343)

[6 Risks and Dependencies 16](#_Toc506760344)

[7 Open Issues 17](#_Toc506760345)

[8 Appendix A—Test Case Specification Example 18](#_Toc506760346)

1. Test Plan Summary

This test plan is used for Costco.com website UI automation, focus on some major functionalities.

1. Test Environment Specification
   1. The Test Team

SDET: Jiayue Hu

1. Setup and maintain test solution
2. Setup CI test environment

Reviewer: Ty Norton, Frank Zhang, Eric Falsken

* 1. Test Planning
     1. Scope and Objectives

Covered:

Functional

Not covered:

Localization other than English (in v2 with resource files)

Globalization

Performance

Security

Objective:

Test Suite: <http://www.costoc.com>

* + 1. Schedule

2/13 Test plan design / Test cases writing / Dependencies collection

2/14 Solution and architecture design and implement / example

2/15 – 16 Implement solution based on scenarios

2/17 Review and delivery

* + 1. Risks

1. External: Network / Website maintenance
2. Time limitation
   1. Documentation Requirements

Selenium(3.9) Docs: <http://www.seleniumhq.org/docs/>

Nunit(3.x) Docs: <http://nunit.org/documentation/>

Nunit-Console: <https://github.com/nunit/nunit-console/releases/tag/3.8>

1. Test Subject and Resource Requirements
   1. Features and Functionality to Test

Features:

Here’s the major features which are going to do functional testing in this test plan on the test suite.

1. User Scenario
   1. New user can register successfully and redirect back to homepage
   2. Existing user can sign in/out
   3. After sign in, “My Account” can show user menu
   4. “Account Detail” page show user information
2. Search
   1. Search component is location at header
   2. Groceries dropdown list is enabled and selectable
   3. Search leads to catalog search page or specific result page, or friendly-nothing page
3. Catalog Page
   1. Based on different catalog, the page shows related products, filters, results, crumb
   2. Products are shown as tile-set and sorted, paged
   3. Products are clickable to related product detail page
4. Product Detail
   1. Show all information about the select product
   2. List clear price and deal info
   3. User is enable to add count to shopping cart
5. Shopping cart
   1. Page shows correct products count, price and all other information in list
   2. Cart Icon is on header which is always shown in page
   3. User is enable to edit products count, or remove in the page
6. Find Warehouse
   1. The function is on header
   2. User can search warehouse by address, city, zip
   3. The result warehouses are shown in list, while a Bing map is shown too

## Functionality test cases:

* **User Scenarios**

|  |  |
| --- | --- |
| Testcase Name | UserScenarios\_Register |
| Goal | Register function |
| Requirement | Generate new email randomly |
| Steps | 1. Open test suite homepage 2. Click “Sign In/Register” button 3. Click “Create new account” button in log on page 4. Submit the form to create new account |
| Verification | 1. The UI layout during actions 2. Register successfully 3. After registration, should redirect to homepage with signed in |

|  |  |
| --- | --- |
| Testcase Name | UserScenarios\_Sign In/Out |
| Goal | Sign in / out function |
| Requirement | Existing account |
| Steps | 1. Open test suite homepage 2. Click “Sign In/Register” button 3. Sign in with existing account 4. Mouse over “My Account” button 5. Click sign out button |
| Verification | 1. The UI layout during actions 2. Sign in / out successfully 3. After signed in, should redirect to homepage |

|  |  |
| --- | --- |
| Testcase Name | UserScenarios\_AccountPage |
| Goal | User Menu after signed-in |
| Requirement | Existing account |
| Steps | 1. Sign in with existing account 2. Mouse over “My Account” button 3. Click the first “Account Detail” button |
| Verification | 1. The UI layout during actions 2. User menu’s functionality 3. Menu items in User Menu’s functionality |

* **Search Scenarios**

|  |  |
| --- | --- |
| Testcase Name | SearchScenarios\_Basic |
| Goal | Go through search flow |
| Requirement | One keyword which causes normal search. Such as: sofa |
| Steps | 1. Search the keyword from homepage header 2. Click the first item in the catalog search page |
| Verification | 1. The UI layout during actions 2. Search result is acceptable 3. Product tiles are ordered, paged, and clickable |

|  |  |
| --- | --- |
| Testcase Name | SearchScenarios\_NoResult |
| Goal | Friendly no result page |
| Requirement | One keyword which causes no result in search. Such as: asdasd |
| Steps | 1. Search the keyword from homepage header |
| Verification | 1. The UI layout during actions 2. Friendly no result page |

* **Find Warehouse Scenarios**

|  |  |
| --- | --- |
| Testcase Name | FinWarehouseScenarios\_Basic |
| Goal | Go through find flow |
| Requirement | One keyword which causes normal in search. Such as: 98052 |
| Steps | 1. Search the keyword from homepage header |
| Verification | 1. The UI layout during actions 2. Search result is acceptable 3. Warehouses are listed by distances, and also maps |

|  |  |
| --- | --- |
| Testcase Name | FinWarehouseScenarios\_NoResult |
| Goal | no result page |
| Requirement | One keyword which causes no result in search. Such as: asdasd |
| Steps | 1. Search the keyword from homepage header |
| Verification | 1. The UI layout during actions 2. Friendly no result |

* **Shopping Cart Scenarios**

|  |  |
| --- | --- |
| Testcase Name | ShoppingCartScenarios\_WithItems |
| Goal | Go through normal flow |
| Requirement |  |
| Steps | 1. Search the keyword from homepage header 2. Select one item to open product detail page 3. Add to cart 4. Open cart page 5. Open another item detail page 6. Add to cart 7. Open cart page again |
| Verification | 1. The UI layout during actions 2. After adding the first item, cart page shows correct related item 3. After adding the second item, cart pages shows two items with correct count 4. The price, total price show correctly |

|  |  |
| --- | --- |
| Testcase Name | ShoppingCartScenarios\_NoItem |
| Goal | With no item, the cart page shows correctly |
| Requirement |  |
| Steps | 1. In home page, click “Cart” icon directly |
| Verification | 1. The UI layout during actions 2. Friendly no item display |

* **Item Detail Scenarios**

|  |  |
| --- | --- |
| Testcase Name | ItemDetailScenarios\_Basic |
| Goal | Information verification |
| Requirement |  |
| Steps | 1. Open one product detail page 2. Update count, then click “Add to cart” 3. Continue to shopping 4. Go another product detail page 5. Add to cart 6. Go to cart |
| Verification | 1. The UI layout during actions 2. All information and image should be displayed correctly 3. The correct count items are added to cart successfully |

* 1. Expected Results of Tests

Test Run Summary

Overall result: Passed

Test Count: 10, Passed: 10, Failed: 0, Warnings: 0, Inconclusive: 0, Skipped: 0

* 1. Test Resource Requirements
     1. Environmental Needs

Windows 10

Google Chrome 64.0.3282.167

Recommend hardware settings of test machine:

CPU: xxx

Ram: xxx gb

Disk: xxx gb

* + 1. Test Tool Requirements

VS2017

Selenium webdriver 3.9

Selenium Chrome Driver 2.35.0

NUnit 3.x

Github

Travis CI

NUnit-console.exe

1. Testing Walkthrough

Preparation:

1. Install the latest Chrome browser
2. Install NUnit-console
3. Clone the project from <https://github.com/raymondhu6188/SeleniumDemo.git>

In windows, download git bash for windows

Open git bash and go to the folder you want to clone

Run the command: Git clone <https://github.com/raymondhu6188/SeleniumDemo.git>

1. Open the solution in Visual Studio
2. In Nuget Package Management Console, run the following commands to install packages
   1. Nuget Install packages – In “Package Management Console”
   2. Install-Package nunit
   3. Install-Package NUnit3TestAdapter -projectname Raymond.Costco.Web.AutoTest
   4. Install-Package Selenium.Chrome.WebDriver
   5. Install-Package Selenium.WebDriver
   6. Install-Package Selenium.Support
3. Build the solution

Run the test

1. Using NUnit-console. Here one example.
2. D:\>"c:\Program Files (x86)\NUnit.org\nunit-console\nunit3-console.exe" “D:\Sources\Github\SeleniumDemo\Raymond.Costco.Web.AutoTest\bin\Debug \Raymond.Costco.Web.AutoTest.dll”

<http://nunit.org/docs/2.5/consoleCommandLine.html>

1. Using Test Explorer in VS
2. Open Test Explorer in vs
3. Select the cases, and click Run
4. Reporting
   1. Tracking and Reporting Status

NUnit-console can generate test result xml file, which can use as report.

* 1. Bug Reporting Tools and Methods

By now no bug tracking is required

Can use Issues on github.com to track bugs

<https://github.com/raymondhu6188/SeleniumDemo/issues>

1. Risks and Dependencies

Risks:

1. Travis CI – first 100 builds free only
2. The solution architecture may need to be refined due to complex scenarios
3. May meet different setting based on browser types

Dependencies:

1. Selenium framework
2. NUnit framework and unite-console app
3. Open Issues
4. Currently, only Chrome browser is supported. Should add more browser types, and TDD in config file
5. Consider switching test data I/O in xml or db
6. Consider handling localization
7. CI script is not completed yet
8. Appendix A—Test Case Specification Example

|  |  |
| --- | --- |
| Testcase Name | <Scenarios Name>-<Testcase Name> |
| Goal | <list the major goal of test case> |
| Requirement | <list all requirement, setting, condition> |
| Steps | # the steps only in the test case   1. Step1 2. Step2 3. Step3 4. … |
| Verification | # all validation points   1. The UI layout during actions 2. Validation point 1 3. Validation point 2 4. … |