**Quality Assurance Sample Testing Plan**

Site: https://www.williams-sonoma.com

Version: 1.0

Created: 7/17/2019

Updated: 7/17/2019

Document Contact: Marc Legris (mlegris@iu.edu)

Contents

[1. Introduction 3](#_Toc14271516)

[1.1 Scope of Testing 3](#_Toc14271517)

[1.2 Testing Tools 3](#_Toc14271518)

[2. Test Methodology 4](#_Toc14271519)

[2.1 Test Reporting 4](#_Toc14271520)

[2.2 Bug Reporting 5](#_Toc14271521)

[2.3 Bug Resolution 5](#_Toc14271522)

[3. Core Functional Testing Documentation 5](#_Toc14271523)

[4. Final Notes 11](#_Toc14271524)

## 1. Introduction

This test plan covers the overall framework of verifying the functionality of the live Williams Sonoma (WS) e-commerce site. The testing plan includes reporting of testing results and logging of any issues that may negatively impact a user’s experience on the WS site.

# 1.1 Scope of Testing

The WS site is accessible through both mobile and native browsers on multiple operating systems. The WS site should be tested to the best of the QA team’s ability to provide adequate coverage with respect to the current market share of operating systems and browsers. Currently1, Windows, Android, and iOS/OS X have the largest operating system share and Chrome, Safari, IE, Firefox and Opera are the most used web browsers. Operating system and browser combinations can be testing on native systems or with the help of Virtual Machines (VM) such as VMWare or VirtualBox. Mobile testing can be completed with in-house physical unit testing, VMs or through third-party mobile testing providers. For the simple QA testing plan, only one OS/Browser (Windows 10/Chrome) combination will be considered.

[1] <https://www.w3counter.com/globalstats.php>

# 1.2 Testing Tools

The focus of testing will be the automation of all functional and unit tests if possible. In cases where automation is not possible, manual tests will be conducted. Selenium (version 3.141.59) has been selected to implement all automated testing using Python (version 3.14.0) as the client driver and Chrome (version 75.0.3770.100) as the testing browser.

# 2. Test Methodology

Each test will consist of documenting its type (Unit, Functional, UI, etc), purpose, start/end conditions and priority. The tests will be categorized by its type, version and given a unique identifier for cataloging. Once the test has been documented, it will be implemented in Selenium and stored in the test repository directory on Github. Selenium tests can be updated, replaced or removed if the supporting documentation is updated to the current status.

Example test documentation:

Type: Functional

Version: 1.0

ID: F.1

Purpose: Contact form – Sending test

Start conditions: Contact page fully loaded

Webpage: https://www.williams-sonoma.com/customer-service/email-us/?cm\_type=fnav

End conditions: Thank you page loaded

Webpage: https://www.williams-sonoma.com/customer-service/email-us/thanks-for-email.html

Text: Thank you for contacting us by email. We will try to respond within one business day.

Priority: Medium/low

Test Repository: [PATH]

Notes: None

# 2.1 Test Reporting

The completion of each test in Selenium will output the status of the test, such as Pass/Fail or partial completion. The overall test results will be grouped by category for easy viewing of overall test completion using the same categories as the test documentation. Detailed test results of each category will be available to drill down to individual tests. Failures of individual tests will be rerun as an automated test or by manual testing for verification. Once a failed test has been verified, a bug will be opened in the currently used bug tracking tool.

# 2.2 Bug Reporting

Each verified failed test will be entered in the bug tracking tool with appropriate details. The bug report must contain easy to follow steps for bug reproduction. The report must also include bug description, expected result, user’s operating system version, browser version, browser plugins, screenshots (if necessary), and any additional relevant details. Priority on the bug report should be set to the same level as in the test documentation and assignment set to the appropriate party.

# 2.3 Bug Resolution

When a bug has been accepted and a fix pushed to production, regression testing must take place to verify the bug no longer exists and the test can be completed with the expected outcome. The bug report can then be closed and marked as no longer outstanding. If the bug persists after regression testing, the bug report must be reopened and updated to the appropriate status.

# 3. Core Functional Testing Documentation

In this section, core use cases will be documented in the manner described in the test methodology section.

**Core Use Case 1:**

Type: Functional

Version: 1.0

ID: F.1

Purpose: Shopping cart – add/remove item

Start conditions: Shopping cart webpage, empty cart

Webpage: https://www.williams-sonoma.com/shoppingcart/?cm\_type=gnav

End conditions: Shopping cart webpage, empty cart

Webpage: https://www.williams-sonoma.com/shoppingcart/?cm\_type=gnav

Priority: High

Test Repository:

https://github.com/Legris-buoy/Skills/Challenge/blob/master/Core\_tests.py

Notes:

1. Verify empty cart
2. Add item to cart
3. Verify item in cart
4. Remove item in cart
5. Verify empty cart

**Core Use Case 2:**

Type: Functional

Version: 1.0

ID: F.2

Purpose: Purchase item – shopping cart

Start conditions: Shopping cart webpage, items in cart

Webpage: https://www.williams-sonoma.com/shoppingcart/?cm\_type=gnav

End conditions: Landed on payment page

Webpage: https://www.williams-sonoma.com/checkout/payment.html

Priority: High

Test Repository:

https://github.com/Legris-buoy/Skills/Challenge/blob/master/Core\_tests.py

Notes:

1. Click Checkout from shopping cart
2. Click “Guest Checkout”
3. Enter Address
4. Click “Use this as my billing address”
5. Click “Continue”
6. Verify on “Payment & Review” page
7. End test (no test CC available for full purchase)

**Core Use Case 3:**

Type: Functional

Version: 1.0

ID: F.3

Purpose: Login to W&S account

Start conditions: W&S homepage

Webpage: https://www.williams-sonoma.com

End conditions: Successful login to account page

Webpage: https://www.williams-sonoma.com/

Priority: High

Test Repository:

https://github.com/Legris-buoy/Skills/Challenge/blob/master/Core\_tests.py

Notes:

1. Click “My Account”
2. Click “Sign in” on dropdown
3. Enter “buoy@mail-group.net” in email field
4. Enter “1EmYouBuoyMe1” in password field
5. Click “Sign in”
6. Click “Continue”
7. Verify on account page

**Core Use Case 4:**

Type: Functional

Version: 1.0

ID: F.4

Purpose: Track Order

Start conditions: Tracking page

Webpage:

https://www.williams-sonoma.com/customer-service/order-shipment-tracking/

End conditions: Order details page displayed

Webpage:

https://www.williams-sonoma.com/customer-service/order-shipment-tracking/results.html?orderNumber=043289560862&postalCode=02144

Priority: High

Test Repository:

https://github.com/Legris-buoy/Skills/Challenge/blob/master/Core\_tests.py

Notes:

1. Enter tacking number “043289560862” in tracking field
2. Enter “02144” in zip code field
3. Click “Track Order”
4. Verify Order details

**Core Use Case 5:**

Type: Functional

Version: 1.0

ID: F.5

Purpose: Create account

Start conditions: Main webpage

Webpage: https://www.williams-sonoma.com

End conditions: Account page

Webpage: https://www.williams-sonoma.com/account/

Priority: High

Test Repository:

https://github.com/Legris-buoy/Skills/Challenge/blob/master/Core\_tests.py

Notes:

1. Click “My Account” from main page
2. Enter Name, email and password
3. Verify welcome page

# 4. Final Notes

Implementation of overall documentation would optimally occur in a wiki rather then in a Word document for built in version control and easier collaboration between QA team members. Individual test documentation would be auto generated and updated by pulling comments from the test scripts themselves. This would allow for the test scripts and documentation to be kept in sync and be more efficient then having to manually update the test documentation whenever an individual test was updated. Performance testing was not considered in this simple test plan but would be a useful detail to include to test results. Implemented tests in Selenium were kicked off sequentially but could have easily been run in parallel.