[Unity QA Tast_QA_Search Test Strategy]

Test Strategy

Revision History

Date	Version	Author	Description
2020-11-10	0.1	Chen	

Table of Contents

1. Introduction and Scope

1.1 Introduction

The purpose of the test strategy for Unity QA Task Search Function (Search Function) is to:

Provide a central artifact to govern the strategic approach of the test effort; it defines the general approach to be employed when testing the functions of Search Function and when evaluating the results of the testing.

- Who will review the document?

  PO/Dev Manager/QA Manager

- Who will approve this document?

  PO

- Testing activities carried out with timelines
- Functional Test before 2020-11-12 need to be done submit to QA manager

1.2 Scope

2. Test Approach

The test approach defines the scope and general direction of the test effort. It is a high-level description of the important issues needing to be covered in the test plan and test scripts. Will cover basic Search function test. Since there is no user acceptance criteria documents, Being given the tight deadline and severity of testing, I used the exploratory approach of testing. My goal was to test basic Search features and to find violations of compatibility requirements.

Testing Approach

-Make use of heuristics to guide testing.

-Test cases execution and test case creation go hand in hand.

-Test cases keep on evolving based on tester observation and learning.

-Different testing techniques like Boundary value analysis, equivalence testing etc.

-Testers can branch out their ideas but never stray from your mission.

Benefits

-Benefits of this testing include():

-Promote real-time thinking and helps in uncovering more defects.

-Promote use cases and scenario-based testing.

-Minimal documentation, maximum testing.

-Emphasis is more on learning and broadening the horizon of a tester.

-Avoid duplicate work.

-Useful when you want to audit other tester’s work.

Demerits：

-Demerits are enlisted below:

-Testing depends on tester experience, skill, and knowledge.

-Require time to learn the application. Tester is more likely to miss if they know less about the application.

-Not appropriate for projects with long execution time.

For each testing phase, a detailed test plan shall be developed that identifies the testing requirements specific to that phase.

2.1 Test Items

Only low-level wireframe Search Function functional feature

2.3 Test Execution Procedures

2.4 Roles and responsibilities of each team member

2.5 Test Deliverables

Documents, including test strategy and test plan sheet
Repo
Readme txt

2.6 Test Data Management

2.7 Test Schedule

2.8 Defects Management

High Priority defects, re-testing, Defect triage, Regression Testing and test sign off

3. Test Environment and Tools

- 3.1 Testing Environment

  https://frontend.nopcommerce.com/

- 3.2 Testing Technology Stack

  Java + Maven + TestNG + Selenium WebDriver +Git

- ## 3.3 Testing Tools

  IntelliJIDEA+Git

- ## 3.4 Documents

  Define backup of test data and restore strategy
  Google Sheet
  Test Plan

	Basic search function Test Scenarios	**Status**	**Comment**
1	Basic Search Positive Elements validate Test	Passed	
2	Basic Search Positive Exact Match Test	Passed	
3	Basic Search Similar Match Positive Test	Passed	
4	Partly Search Test	Passed	
5	Logic Operator Search Test	Failed	Not support, Need PO review
6	Invalid Character Search Negative Test	Passed	
7	Search Enter KeyTest : User should be able to search when he enters the keyword and hits ‘Enter’ button on keyboard	Passed	
8	SearchButton Test	Passed	
9	Enter Empty Input Search Test	Passed	
10	Invalid Characters Input Search validate	Failed	Defects if users enter invalid characters , supposed to popup a alert or error hint message
11	Solid Search Test with super long input or max integer not show error or broken	Passed	
12	One Character Show Warning Message	Passed	
13	Search Hint Text after enter partly text validate	Passed	
14	Search not match Error Validate	Passed	
15	Validate if Case Sensitive Search still works	Passed	
16	Wildcard Character And Case Sensitive Validate	Passed	

17	Space Compatibility Validate	Failed	Defect, need to deal with the space trim for the input search content
18	Basic Search Cover All Category Validate	Passed	
19	After Search Be Able To Go Back To HomePage Validate	Passed	
20	Minimum Length Warning Search	Passed	
21	Destroy Search Test	Failed	Defect,if users enter super long search content, system supposed to popup a alert windows instead of jump to a error page

4. Release Control

- Release management plan with appropriate version history that will make sure test execution for all modification in that release

5. Risk Analysis

- List all risks that you can estimate
- Give a clear plan to mitigate the risks also a contingency plan

6. Review and Approvals

- All these activities are reviewed and sign off by the business team, project management, development team, etc.
- Summary of review changes should be traced at the beginning of the document along with the approved date, name, and comment

Attachment

Readme

1. This is a maven project, Including ui test under test folder, please run maven clean install get all the library.
2. Need to make sure your chromedriver in the correct path (please check the Const chromeDriverPath variable, make the correct path and the download the chromedriver)
   Or you may include the ChromeDriver location in your system PATH environment variable
   (If not specified WebDriver will search your system PATH environment variable for locating the chromedriver)
3. Run test suite?
   Run maven test.
   (1) In the Maven tool window, under the Lifecycle node, right-click run, start to run
   (2) In command line: mvn test

   Run testng.xml (Which is in /resources/testng.xml), select testng.xml file, mouse right click run.

4. How to run single test?

(1)In the Maven tool window, under the Lifecycle node, right-click the test goal.

From the context menu, select Create 'name of the module/project and name of a goal'.

(2)In command line: mvn -Dtestname test

5. Run test cases via testng.bat

Example

```
set projectLocation=C:\ %projectLocation%set
classpath=%projectLocation%\bin;%projectLocation%\lib\*java org.testng.TestNG
%projectLocation%\testng.xmlpause
```

And in build step give the name of your TestNG.bat in Jenkin, add test description.

6. You may also run test suite at Jenkins or sourceLab.

7. You may change testng.xml config to control your regression test or unit test.

   Add suite name and run ui and api test cases separately

8. This project will use chromedriver , if you want to use different Browser, you may change at BaseTest, there is the instruction.

Note: To use a different WebDriver you need to specify the driver path in the file system and then instantiate it. For example, if you want to use IE then here is what you'd need to do: System.setProperty("webdriver.ie.driver", "path/to/IEDriver");

WebDriver driver = new InternetExplorerDriver();

9.Check the test results

Could be check via your IDE control window