Dashing Through the Snow

System Verification Plan

MCDA 5510(Software Development in Business Environment)

M.Sc. in Computing and Data Analytics

Saint Mary’s University

Halifax, NS B3H 3C3, Canada

Date – 26th Nov 2022

Table of Contents

[1. Introduction 3](#_Toc7163744)

[1.1 Purpose 1](#_Toc7163745)

[1.2 Reference Document 1](#_Toc7163745)

[1.3 Abbreviations and Acronyms 1](#_Toc7163745)

[2. Testing Approach](#_Toc7163746)

[2.1 Component Testing 2](#_Toc7163747)

[2.2 Integration Testing](#_Toc7163748) 2

[2.3 Interface Testing ----](#_Toc7163748)3

[2.4 Recovery Testing 3](#_Toc7163748)

[2.5 Performance Testing 3](#_Toc7163748)

[2.6 Regression Testing 4](#_Toc7163748)

[2.7 Acceptance Testing 4](#_Toc7163748)

[3. Test Description 5](#_Toc7163753)

[3.1 Test Description 5](#_Toc7163754)

[3.1.1 Functions 5](#_Toc7163755)

[3.1.2 Test Data 6](#_Toc7163756)

[3.1.3 Test Procedure 6](#_Toc7163757)

Appendix A: Record of Changes 11

Appendix B: Approvals 11

**List of Tables**

[*Table 1 - Functions* 6](#_Toc120484451)

[*Table 2 - Test Deliveries* 7](#_Toc120484452)

[*Table 3 - User Interface test procedure* 8](#_Toc120484453)

[*Table 4 - Function test procedure* 9](#_Toc120484454)

[*Table 5 - Data test procedure* 10](#_Toc120484455)

[*Table 6 - Record of Changes* 11](#_Toc120484457)

[*Table 7 - Approvals* 11](#_Toc120484458)

## 

# Introduction

System Verification Plan (SVP) provides information about defining the process of testing an E-commerce platform.

The E-commerce platform is intended to allow the users of the platform to search, cart and buy Christmas themed items.

### Purpose

A System Verification Plan is **essential to ensure that requirements for the e-commerce website are developed, and that the system and its components are tested and meet the requirements.** Successful verification of all system requirements initiates functional acceptance of a system.

### Reference Document

System Requirement Document (SRS)

System Design Document (SDD)

### Abbreviations and Acronyms

**Abbreviation Description**

ECW E-Commerce Website

SVP System Verification Plan

SDD System Design Document

SRS System Requirements Specification

MVC Model View Controller

SQL Structured Query Language

NFR Non-Functional Requirements

FR Functional Requirements

IR Interface Requirement

DB Database

Frontend Client-Side User Interface

Backend Server

## Testing Approach

This section describes the Test Approach principles and strategies to be used as guidelines when testing E-Commerce System.

### Component Testing

Testing components separately, without integrating them with other components, is known as component testing. The process is also known as module testing when viewed from an architectural perspective. Component Testing is also referred to as Unit Testing, Program Testing or Module Testing.

When it comes to validation of an ECW, the components can be tested as below

* Whether a user can successfully register/sign into the website
* An item can be searched for and ordered on the website.
* An item can be placed into a separate cart for every user.
* User can access their own profile information.

### Integration Testing

Integration Testing is a type of software testing in which the different units, modules or components of a software application are tested as a combined entity.

It is testing conducted on a complete, integrated system to evaluate the system’s compliance with its specified requirement.

In this testing, it is performed to verify the interactions between the modules of the ECW. It involves verifying the high-level and low-level software requirements specified in the Software Requirements Specification/Data and the Software Design Document. Integration testing in ECW would include testing the following interactions: -

* Whether the different modules/ interfaces of the website are compatible with one another.
* Whether after integration, the system does not crash and functions as per its functionalities.

### Interface Testing

An application's visual elements are tested to determine whether they meet the expected performance and functionality.

A connection that integrates two components is called interface.

By testing the GUI, testers can validate that UI functions are free from defects. Unit Testing: It tests the functionality of an individual part of the ECW. It makes sure every part of the programs runs well and is properly aligned with the software.

Our E-commerce website will be tested for the following components:

* Visual Design
* Compliance
* Functionality
* Usability
* Performance

### Recovery Testing

Recovery testing is the forced failure of the software in a variety of ways to verify that recovery is properly performed.

Recovery testing is basically done to check how fast and better the application can recover against any type of crash or hardware failure etc. Recovery testing is simulating failure modes or causing failures in a controlled environment.

* We must create a test bed close to actual conditions to ensure that testing is done on the correct set of software and hardware.
* Testing our backup cloud systems to ensure they are working fine after a disaster.
* Restart the system while a browser has a definite number of sessions open and check whether the browser can recover all of them or not.

### Performance Testing

Performance Testing tests the speed, response time, stability, reliability, scalability, and resource usage of a software application under particular workload. The main purpose of performance testing is to identify and eliminate the performance bottlenecks in the software application.

The focus of Performance Testing for E-commerce Website includes below conditions.

Speed – Checks to see if the maximum user load can be handles by the software application.

Scalability – Checks for the maximum user load that the software can handle.

Stability – Checks to see if the application is stable under varying workloads.

### Regression Testing

Regression testing is a software testing practice that ensures an application still functions as expected after any code changes, updates, or improvements.

Regression testing is responsible for the overall stability and functionality of the existing features. Whenever a new modification is added to the code, regression testing is applied to guarantee that after each update, the system stays sustainable under continuous improvements.

Typically, regression testing is applied under these circumstances:

* A new requirement is added to an existing feature
* A new feature or functionality is added
* The codebase is fixed to solve defects
* The source code is optimized to improve performance
* Patch fixes are added
* Changes in configuration

Therefore, Regression Testing becomes necessary. Regression Testing can be carried out from testing functions in ECW such as login, registration, add to cart, Selecting/Searching for an item etc.

### Acceptance Testing

Acceptance testing is aquality assurance (QA) process that determines to what degree an application meets end users' approval.

Acceptance tests are conducted to ensure the software or app matches business requirements and end-user needs. An acceptance test returns either a pass or fail result. A fail suggests that there is a flaw present, and the software should not go into production.

Acceptance Test will be covered in Test Procedure.

## Test Description

### Functions

Functional testing will be performed to check the functions of the E-commerce System. We carry out functional testing by giving an input to the application and verifying the output of the same.

The below table describes the overall test. Details of tests will be described in section 3.3. Note: The scope can be changed when the project start

|  |  |  |  |
| --- | --- | --- | --- |
| **User** | **Scenarios** | **Sub Levels** | **Complexity** |
| Users | Login Page | Login | Medium |
|  |  | Validate user credentials | Complex |
| Users | Register | Add personal details | Medium |
| Users | Search for Item | Search by item | Complex |
|  |  | Search by category | Complex |
|  |  | Sort by Category | Complex |
| Users | Add to cart | New session for every user | Complex |
|  |  | Separate cart for every user | Complex |
|  |  | Add to cart | Medium |
|  |  | Delete from cart | Medium |
| Users | Payment | Save payment information | Medium |
|  |  | Payment gateway | Complex |
|  |  | View Summary | Medium |
| Users | Order History | View History | Medium |

*Table 1 - Functions*

Testers: Testing Team

Test Deliveries

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Deliverable Name** | **Author** | **Reviewer** |
| 1 | Test Plan | Test Lead | Project Manager/ Team Lead |
| 2 | Functional Test Cases | Test Team | Business Analyst’s Sign off |
| 3 | Logging Defects in Testing Environments | Test Team | Test Lead/ Programming Lead |
| 4 | Daily/weekly status report | Test Team/ Test Lead | Test Lead/ Project Manager |
| 5 | Test Closure report | Test Lead | Project Manager |

*Table 2 - Test Deliveries*

Test Lead: Shiney Prabhakar

Test Team: Piyush Priyam, Rashad Ahmed Imtiaz

Project Manager: Neil Williams, Dan Penny

Business Analyst: GBenro

### Test Data

Testing team will make use of dummy data to test the system during the time of execution.

Sample inventory data present in our MS SQL will be used by our e-commerce application to test all functionalities. From accepting data in the backend through our MVC to integration with our client, server, generator services and payment layer, data will be transferred and tested.

### Test Procedure

This process involves understanding every requirement and building a separate document for each.

In this phase, we will be following all the testing steps described in section 2 of this document.

Testers will maintain a spreadsheet detailing the events that occur for every test scenario.

The same will be periodically shared with the rest of the team for review.

**3.3.1 User Interface test procedure**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step** | **Test Scenario** | **Testing method** | **Expected Outcome** |
| 1 | Check if each subpage on the menu is loading | Interface Testing | Pages should load with all content |
| 2 | Should be able to identify menu icons/pages | Interface Testing | Icons are identifiable |
| 3 | Check drop-down lists | Interface Testing | Contents should be displayed in the list |
| 4 | User can navigate from one page to another | Interface Testing | Navigation is easy enough for users to understand |
| 5 | Check filter and sort options to search items | Interface Testing | Sort and filter icons need to be easily understandable. |

*Table 3 - User Interface test procedure*

**3.3.2 Function test procedure**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step** | **Test Scenario** | **Testing method** | **Expected Outcome** |
| 1 | To login/register into the system | Component Testing | Can login/register successfully |
| 2 | User searches for item in home page | Component Testing | Can find item or related items or receive appropriate ‘item not found’ message |
| 3 | User searches based on category | Component Testing | Should display all items in that category or appropriate ‘category not found’ message. |
| 4 | Check Load time for the search result in search menu | Performance Testing | Load Time should be fast enough |
| 5 | User adds/deletes item to/from a cart | Component Testing | Should be able to add/delete an item from the cart |
| 6 | User pays for cart items | Component Testing | Should successfully connect to payment gateway |
| 7 | User views order history | Component Testing | Order history needs to be displayed |
| 8 | Check data integrity on the page of result view | Integration Test | Selected result should match with the relevant data |

*Table 4 - Function test procedure*

**3.3.3 Data test procedure**

|  |  |  |  |
| --- | --- | --- | --- |
| **Step** | **Test Scenario** | **Testing method** | **Expected Outcome** |
| 1 | Check column constraints | Component Testing | Not null etc.. |
| 2 | Check for acid properties. | Component Testing | Data should be compliant with the acid properties |
| 3 | Check logical components of each tables | Component Testing | e.g. Customer, cart, orders etc |
| 5 | Check the primary key column in table | Component Testing | Tables should have a primary key column and a primary key should not be null |
| 6 | Verify that data inserted from UI is reflecting properly in the appropriate table. | Integration Test | The data should be correctly matched in both database and web page |

*Table 5 - Data test procedure*

Appendix A: Record of Changes

Table 6 - Record of Changes

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Date** | **Author/A#** | **Description of Change** |
| 1 | 11/26/2022 | Shiney Prabhakar/A00466455 | First version of System verification plan. |
|  |  |  | The document review was done by the team members and all the comments are incorporated. |

*Table 6 - Record of Changes*

**Appendix B: Approvals**

The undersigned acknowledge that they have reviewed the System Validation Plan and agree with the information presented within this document. Changes to this SVP will be coordinated with and approved by the undersigned, or their designated representatives.

**Table 7 – Approvals**

|  |  |
| --- | --- |
| **Document Approved By** | **Date Approved** |
| Piyush Priyam, Team Member, A00464893 | 11/26/2022 |
| Rashad Ahmed Imtiaz, Team Member, A00466845 | 11/26/2022 |

*Table 7 - Approvals*