Week 2 Assignment—E-Commerce Website Test Plan

Shaun Hoadley

CST313: Software Testing

Michael Hayden

06/14/2021

Software Requirements Specification

for

E-Commerce Website Test Plan

Version 1.0 Release 0.7 Waiting for Approval

Prepared by Shaun Hoadley

CSC313: Software Testing

06/14/2021

Table of Contents

Table of Contents iii

Revision History iii

1. Introduction 4

1.1 Purpose **4**

1.2 Document Conventions **4**

1.3 Intended Audience and Reading Suggestions **4**

1.4 Product Scope **4**

2. Overall Description 4

2.1 Product Perspective 4

2.2 Product Functions 5

2.3 User Classes and Characteristics 5

2.4 Operating Environment 5

2.5 Design and Implementation Constraints 5

2.6 User Documentation 5

2.7 Assumptions and Dependencies 5

3. External Interface Requirements 6

3.1 User Interfaces 6

3.2 Hardware Interfaces 6

3.3 Software Interfaces 6

3.4 Communications Interfaces 6

4. System Features 6

4.1 System Feature 1 6

4.2 System Feature 2 (and so on) 7

5. Other Nonfunctional Requirements 7

5.1 Performance Requirements 7

5.2 Safety Requirements 7

5.3 Security Requirements 7

5.4 Software Quality Attributes 8

5.5 Business Rules 8

6. Other Requirements 8

Appendix A: Glossary 8

Appendix B: Analysis Models 8

Appendix C: To Be Determined List 8

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this SRS (software requirements specification) is to outline the requirements of software build version 1.0, document release 0.6, of the e-commerce website. The e-commerce website has the expressed purpose of allowing customers to quickly locate and purchase items from a range of options. Release 0.6 of this document refers to the e-commerce website version 1.0.

## Document Conventions

This document will be broken into 5 sections. These sections will have headers for Introduction, Overall Description, External Interface Requirements, System Features, and Other Non-Functional Requirements. Sub-headers will provide the details of the specifics for each of those sections.

## Intended Audience and Reading Suggestions

This SRS has an intended audience of software developers, architects, designers, project managers, software testers, and any other stakeholders involved this project and its execution. To gain full understanding for the purpose and requirements for the e-commerce website, it is suggested that all audience members read the document read the document in its entirety.

## Product Scope

The e-commerce website allows users to find the products they want quickly and easily. The user will be able to search for an item, select from a list of search suggestions, narrow the search by selecting different filters items, view the details of the items, add items to a cart, continue shopping for more items, and checkout when they are ready. The development of this e-commerce website should increase growth and profitability, which is the objective of the business.

# Overall Description

## Product Perspective

This is the build for a new and self-contained E-commerce website.

## Product Functions

The user will be able to enter an item search, add filters to narrow the search, view the item’s details, add items to a cart, and continue shopping and/or checkout. The filters will allow items to be narrowed by type and category, and other cascading filters specific to the item.

## User Classes and Characteristics

The e-commerce website will contain user classes for administrators, vendors, and customers. The administrators will have access to all of the website functionality. Vendors will be able to add, remove, and update the items they supply. Customers will be able to search for and select items for purchase.

## Operating Environment

The e-commerce website needs to be agnostically coded to support a variety of operating systems, web browsers, and mobile platforms. These include, but are not limited to, MacOS, Windows, Linux, IOS, Android, Edge, Chrome, Firefox, and Safari.

## Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>

## User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

**References**

***Please note:*** *None of the following references were directly cited. Wiegers is the template I used for the overall document layout. Kumar has the problem statement to be designed. Lastly, Spillner provided the context behind the SRS documentation and the importance it has in software testing.*

Kumar, J. (2015). [The e-commerce problem statement](https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781783981847/1/ch01lvl1sec12/the-e-commerce-problem-statement). In *Apache Solr patterns*. Packt Publishing. https://bit.ly/2Rn8inE

Spillner, A., Linz, T., & Schaefer, H. (2014). [*Software testing foundations: A study guide for the certified tester exam* (4th ed.)](https://ashford.instructure.com/courses/86077/modules/items/4350686). Rocky Nook.

Wiegers, K. E. (1999). [*Software requirements specifications for <project>*](https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc). [Template]. <https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc>