**Software Requirements Specification**

For E-Commerce Product Project

Version 1.0 approved

Prepared by Group 11

Intro to SE

1/30/24

# **1.** **Introduction**

This document is the Software Specification Requirement (SRS) for the online shopping system Books R Us, a user-friendly system to both consumers and vendors alike.

The product will allow the consumers to easily search and purchase desired books and vendors to list and sell books.

## **1.1** **Purpose**

The purpose of this document is to display and examine varied ideas to provide a detailed definition of the system in regards to users. In addition, outlines of potential features, ideas, and hindrances will be presented, structured and discarded as the system’s development requires. These outlines will be used to further our understanding and by extension the quality of the product.

Overall, this document describes a complete overview of the system, guidelines, and objectives, definingthe software and hardware specifications, user interface, and target clients.

## **1.2** **Document Conventions**

The use of the pound symbol (#) signifies a comment that is not inherently part of the SRS. These comments are intended to provide helpful insight, highlight a potential problem that will need to be addressed, or for simple ease of use by any of the users of this document.

The product features (Section 2 & 3) and requirements (Section 4 & 5) are not designated to showcase different priority levels through fonts, highlighting, or order. Priority level will be assigned through declaration or magnitude of detailed requirements.

## **1.3** **Intended Audience and Reading**

This document is intended for use by the product developers, product overseers, and users. The product developers reference this document to develop the system and amend the SRS as necessary. The product overseers examine this document to determine if the developer team has correctly defined the product, features, and requirements without omitting crucial information. The users grasp a greater understanding of the features given to them, hardware and software requirements necessary to utilize the product, and a comprehension of how the system side of the product works with the client side.

## **1.4** **Product Scope**

The software application is an online web system that provides basic functions to allow a web customer to make book purchases and a web vendor to provide books for purchase. The objective of this interaction is to provide an online foundation to allow consumers to purchase books they seek and allow vendors to sell books for profit. This interactional relationship in turn allows the product provider to gain profit and marketing for expansion.

## **1.5** **References**

Intro to SE SRS document template by Dr. Charan Gudla- <https://canvas.msstate.edu/courses/121638/files/9824905?wrap=1>

# **2.** **Overall Description**

## **2.1** **Product Perspective**

The "Books R Us" ecommerce app is a standalone product designed by a team of four students using C++ for the backend logic and SQL for database management. It is a new, self-contained product that aims to provide a platform for users to buy and sell books. The system interacts with a database to manage book inventory, user information, and transaction records.

*Diagram:*

[User Interface] <---> [C++ Backend] <---> [SQL Database]

## **2.2** **Product Functions**

* User Registration
* Browse Books
* Add to Cart
* Purchase
* Inventory Management
* Profile Management  
    
  A screenshot of a computer screen

  Description automatically generated A screenshot of a computer program

  Description automatically generated

## **2.3** **User Classes and Characteristics**

Buyers(Users)

Able to search, add to cart, and checkout simply and easily.  
Sellers(System Admins)

Able to update the inventory, Manage user accounts, using the app instead of having to make changes to the SQL database or code.

## **2.4** **Operating Environment**

## Hardware Platform: Compatible with standard PCs, laptops, and mobile devices.

## Operating System: Compatible with Windows, MacOS, Linux operating systems.

## Software Components: Requires C++ runtime environment and SQL database support.

## **2.5** **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).>

# **3.** **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.>

## **3.1** **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:

## **3.2** **System Feature 2 (and so on)**

# **4.** **Other Nonfunctional Requirements**

## **4.1** **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.>

## **4.2** **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.>

## **4.3** **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.>

## **4.4** **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.>

# **5.** **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.>