Software Requirements Specification

for

ShoeBae

**Version 1.0 approved**

**Gage Fulwood - Team Leader**

**Citlali Hernandez Lopez**

**John Mark Taylor**

**Yusuf Sarigul**

**Easton Greenwood**

**Mississippi State University**

**Jan 1, 2023**

# Introduction

## Purpose

The purpose of this document is to provide an informative description of this E-commerce system. While providing our customers with the goals we hope to achieve, the document will also provide information regarding software specifications used in the development of **Shoebae 1.0**. This document will benefit each stakeholder of **Shoebae 1.0** by providing a better understanding of concepts that have been used, such as: product management, seller accounts, customer accounts, order management, payment processing, and security measures taken.

## Document Conventions

This document employs a hierarchical numbering system to organize sections and subsections. Sections are presented in bold, size eighteen font, using the format 'X.SectionName,' where 'X' is the section number. Subsections follow a similar convention, displayed in bold, size fourteen font as 'X.Y.SubsectionName,' where 'Y' represents the subsection number. Subsubsections are presented in bold, size 12 font as ‘X.Y.Z.SubsubsectionName’, where ‘Z’ represents the subsubsection number. Additionally, the sections themselves will be written in regular, size 12 font.

For the referencing of the Shoe Bae 1.0 SRS document, our team will be using MLA formatting, specifically the MLA 9th edition format. This format will allow for in-text citations and a works cited section (1.5 references).

## Intended Audience and Reading

This Software Requirements Specification (SRS) document is crafted to cater to a diverse set of stakeholders involved in the development, implementation, and maintenance of Shoebae 1.0. The primary audience includes:

**Developers:** To understand the technical specifications, coding requirements, and implementation details.

**Project Managers:** To gain insights into the project scope, timelines, and resource requirements.

**Marketing Staff:** To comprehend the features and functionalities that can be leveraged for promotional activities.

**Users:** To understand the capabilities and functionalities of the E-commerce system.

**Testers:** To identify testing scenarios and requirements.

**Documentation Writers:** To assist in the creation of user manuals and system documentation.

Readers are advised to follow a sequence that aligns with their specific interests and responsibilities:

**Overview Sections:**

Begin with sections providing an overview of the E-commerce system (e.g., Purpose, Product Scope).

Understand the goals, benefits, and objectives of Shoebae 1.0.

**Product Scope:**

Delve into the short description of Shoebae 1.0, its purpose, and how it aligns with corporate goals or business strategies.

**Overall Description:**

Explore the product perspective, understanding the context, and origin of Shoebae 1.0.

Identify major components through simple diagrams and understand the system's interactions.

**System Features:**

Examine the major functions and features the E-commerce system must perform.

Follow the functional requirements organized by system features.

**User Classes and Characteristics:**

Learn about different user classes (Unregistered, Registered, Buyer, Seller, Admin) and their respective privileges.

**Operating Environment:**

Understand the hardware platform, operating systems, and other software components necessary for Shoebae 1.0.

**Design and Implementation Constraints:**

Identify any limitations or constraints that may impact development options.

**Other Requirements:**

Check for any additional requirements not covered in other sections.

**Nonfunctional Requirements:**

Review performance, safety, and security requirements.

**Appendices:**

Refer to the Glossary for definitions of key terms.

Explore Analysis Models (if included) for visual representations.

Check the To Be Determined List for pending items.

## Product Scope

The web application ShoeBae provides consumers with a hub that other users use to buy, sell, and trade goods, primarily shoes. The purpose of this application is to offer users a database of goods sold by other consumers, providing a convenient way to shop and a platform to sell goods at a price of their choosing. Keeping up with modern standards, this platform allows users to filter through content using various forms of characterization, such as type, color, and size. It also provides the opportunity for people to sell their goods on our platform, thanks to our user-friendly design that makes setting up listings easy and simple.

This design ensures that the software and its features are as simple as an average consumer scrolling through, finding an item, and adding it to a cart or adding a listing while waiting for an offer. Our platform is a place for goods to be widely viewed by a large audience, facilitating purchases. In more detail, as a user, you will sequentially create an account with a username and password, and then log in. You can search through our platform with tools to assist your search, such as filtering through items by type, color, or size. If you find an item, you can easily add it to your cart and complete the process by checking out.

For sellers, the process is similarly simple, with a few additional features to make listing their item hassle-free. The process prompts users with questions about their product to seamlessly categorize it according to their desired specifications within our system. Once these questions are answered, they can list the item and wait for offers or trades to arrive in their inbox for consideration.

Should users decide to sell their goods, the domain owners will take a small commission. This fee is intended to provide continuous support, keeping the web application operational and ensuring it doesn't discourage users from selling their goods. The fee is kept relatively low to maintain user satisfaction and encourage continued engagement on the platform. To further support the application, ads relevant to users' searches will be implemented, alongside the continuous support of developers and investors, all contributing to the website's quality and reliability.

## References

1. *Judy, Tirian, et al. “Software Requirements Specification for Y.” Canvas, Mississippi State University, 10 Sept. 2023,* [*https://canvas.msstate.edu/courses/121638/files/10040003*](https://canvas.msstate.edu/courses/121638/files/10040003)
2. *Pressman, Roger S., and Bruce R. Maxim. Software Engineering: A Practioner’s Approach. 9th ed., McGraw-Hill, 2020.*

# Overall Description

## Product Perspective

*<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>*

## Product Functions

*<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>*

## User Classes and Characteristics

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

## Operating Environment

*<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>*

## 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).>*

# 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.>*

# 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.>*