Software Requirements Specification (SRS)

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this SRS is to describe the requirements for the e-commerce application built using React for the front-end and Spring Boot for the back-end. This document will outline the functional and non-functional requirements, as well as provide an overview of the system architecture and design constraints.

1.2 Scope

This e-commerce application will allow users to browse products, add them to a shopping cart, and purchase them. Sellers can add their products and update the status of the product. It will also have administrative features to analyze the overall growth and sales in the products.

1.3 Definitions, Acronyms, and Abbreviations

• React: A JavaScript library for building user interfaces.
• Spring Boot: A Java-based framework used to create stand-alone, production-grade Spring-based applications.
• REST: Representational State Transfer, an architectural style for designing networked applications.
• API: Application Programming Interface, a set of routines, protocols, and tools for building software applications.

1.4 References

• React Documentation: https://reactjs.org/docs/getting-started.html
• Spring Boot Documentation: https://spring.io/projects/spring-boot

1.5 Overview

This document is structured to provide a comprehensive description of the e-commerce application's requirements. It covers the overall system description, detailed requirements, and the appendices.

2. Overall Description

2.1 Product Perspective

The e-commerce application is a standalone system that will interact with external payment gateways and possibly other third-party services for shipping and customer reviews.

2.2 Product Functions

• User Registration and Authentication
• Product Browsing and Searching
• Shopping Cart Management
• Order Placement and Checkout

• Order History and Tracking
• Seller Panel for Product and Category Management
• Admin Panel for Analysis

2.3 User Classes and Characteristics

• End Users: Customers who will browse and purchase products.
• Seller Users: Seller who will manage products and categories.
• Admin User: Seller who will manage accounts and analyze .

2.4 Operating Environment

• Front-End: Developed using React and deployed on a web server.
• Back-End: Developed using Spring Boot and deployed on an application server.
• Database: MySQL for data storage.
• Payment Gateway: Integration with external payment services like Razorpay.

2.5 Design and Implementation Constraints

• The system must be compatible with major web browsers (Chrome, Firefox, Safari, Edge).
• The back-end must adhere to RESTful principles.
• Security must be implemented for user data and transactions.

2.6 Assumptions and Dependencies

• Users have access to a modern web browser.
• Internet connectivity is available.
• Third-party services (e.g., payment gateways) are reliable and available.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 User Registration and Authentication

• Users shall be able to register by providing an email, password, and other necessary details.
• Users shall be able to log in using their email and password.
• Passwords shall be stored securely using encryption.

3.1.2 Product Browsing and Searching

• Users shall be able to browse products by categories.
• Users shall be able to search for products using keywords.

3.1.3 Shopping Cart Management

• Users shall be able to add products to their shopping cart.
• Users shall be able to view, edit, and remove items from their shopping cart.

3.1.4 Order Placement and Checkout
• Users shall be able to proceed to checkout from their shopping cart.
• Users shall be able to provide shipping and payment information.
• Users shall receive a confirmation email upon successful order placement.

3.1.5 Order History and Tracking
• Users shall be able to view their past orders and track current orders.

3.1.6 Seller Panel
• Sellers shall be able to add, edit, and delete products.
• Sellers shall be able to manage product categories.

3.1.7 Seller Panel
• Admin shall be able to manage accounts of sellers.
• He will be able to analyze data of sales and look after reviews of users on particular sellers.

3.2 External Interface Requirements

3.2.1 User Interfaces
• The front-end will be built using React, providing a responsive and interactive user experience.

3.2.2 Hardware Interfaces
• The system will run on standard web servers and will not require specialized hardware.

3.2.3 Software Interfaces
• Integration with external payment gateways such as PayPal or Stripe.
• RESTful APIs for front-end and back-end communication.

3.2.4 Communication Interfaces
• HTTPS will be used for secure communication between the client and the server.

3.3 System Features
• Product Recommendations: The system may provide product recommendations based on user behavior.
• Wishlist: Users can save products to a wishlist for future purchase.

3.4 Non-Functional Requirements

3.4.1 Performance Requirements
• The system shall handle up to 1,000 concurrent users.
• The system shall have a response time of less than 2 seconds for most operations.

3.4.2 Security Requirements

- User data shall be encrypted in transit and at rest.
- The system shall implement role-based access control.

3.4.3 Usability Requirements

- The user interface shall be intuitive and easy to navigate.
- The system shall provide help and support documentation.

3.4.4 Reliability Requirements

- The system shall have an uptime of 99.9%.
- The system shall provide error handling and recovery mechanisms.

3.5 Other Requirements

- The system shall comply with relevant data protection regulations (e.g., GDPR).