**Software Requirements Specification Document**

**Table of Contents**

1. Introduction  
   1.1. Purpose of Document  
   1.2. Intended Audience  
   1.3. Abbreviations  
   1.4. Document Convention
2. Overall System Description  
   2.1. Project Background  
   2.2. Project Scope  
   2.3. Not In Scope  
   2.4. Project Objectives  
   2.5. Stakeholders  
   2.6. Operating Environment  
   2.7. System Constraints  
   2.8. Assumptions & Dependencies
3. External Interface Requirements  
   3.1. Hardware Interfaces  
   3.2. Software Interfaces  
   3.3. Communications Interfaces
4. Functional Requirements  
   4.1. Functional Hierarchy  
   4.2. Use Cases  
   4.2.1. User Registration  
   4.2.2. User Login  
   4.2.3. Add Product to Cart  
   4.2.4. Checkout Process  
   4.2.5. Admin Add Product
5. Non-Functional Requirements  
   5.1. Performance Requirements  
   5.2. Safety Requirements  
   5.3. Security Requirements  
   5.4. User Documentation
6. References
7. Appendices

**1. Introduction**

**1.1. Purpose of Document**

This Software Requirements Specification (SRS) document outlines the requirements for developing an e-commerce web application. It defines the functional and non-functional requirements, system interfaces, and use cases to guide the development team in building a robust, user-friendly, and secure platform. The document ensures that all stakeholders have a shared understanding of the system's features and constraints, serving as a foundation for design, development, and testing phases.

**1.2. Intended Audience**

This document is intended for the following audience:

* **Project Managers**: To oversee project progress and ensure alignment with business goals.
* **Development Team**: Including software engineers, UI/UX designers, and database administrators to implement the system as per requirements.
* **Quality Assurance Team**: To develop test cases and validate the system against the specified requirements.
* **Stakeholders and Clients**: To review and approve the system requirements and ensure they meet business needs.
* **End Users (optional)**: To understand the system's capabilities and features.

**1.3. Abbreviations**

|  |  |
| --- | --- |
| Abbreviation | Description |
| SRS | Software Requirements Specification |
| UI | User Interface |
| API | Application Programming Interface |
| HTTPS | HyperText Transfer Protocol Secure |
| SQL | Structured Query Language |
| XSS | Cross-Site Scripting |
| CSRF | Cross-Site Request Forgery |

**1.4. Document Convention**

This document uses the following conventions:

* **Font**: Aptos
* **Font Size**: 12pt for body text, 14pt for headings, 16pt for main titles
* **Formatting**: Bold for headings, italics for emphasis, and tables for structured data.
* **Use Case Format**: As per the provided template, with numbered steps and sections for actors, preconditions, scenarios, alternate scenarios, postconditions, and cross-references.

**2. Overall System Description**

**2.1. Project Background**

The e-commerce web application is being developed to address the growing demand for online shopping platforms. The business identified an opportunity to create a user-friendly, scalable platform that caters to both customers and administrators. The current market lacks a seamless, secure, and feature-rich e-commerce solution that supports small to medium-sized businesses in managing their online stores efficiently. This project aims to fill that gap by providing a comprehensive solution for product browsing, purchasing, and administration.

**2.2. Project Scope**

The e-commerce web application will include the following main functionalities:

* **User Panel**:
  + User registration and login.
  + Product browsing, searching, and filtering.
  + Shopping cart management (add, update, remove items).
  + Checkout and payment processing.
  + Order tracking and user profile management.
* **Admin Panel**:
  + Product management (add, update, delete products).
  + Order management (view order details).
* **Non-functional Features**:
  + Performance optimization for high traffic.
  + Security measures to protect user data and transactions.  
    The system will be a web-based application accessible via modern browsers on desktops, tablets, and mobile devices.

**2.3. Not In Scope**

* Integration with external logistics or shipping APIs for real-time tracking.
* Support for cryptocurrency payments.
* Multi-language and multi-currency support beyond English and USD.
* Hardware-level testing or optimization for specific devices.

**2.4. Project Objectives**

The project aims to:

* Develop a scalable e-commerce platform to support online shopping for small to medium-sized businesses.
* Provide seamless user experience for browsing, purchasing, and managing orders.
* Ensure secure transactions and protect user data from common vulnerabilities.
* Enable administrators to efficiently manage products and orders.
* Deliver a high-performance system capable of handling peak traffic with minimal latency.  
  The end result will be a fully functional e-commerce platform that increases customer satisfaction and business revenue.

**2.5. Stakeholders**

|  |  |
| --- | --- |
| Stakeholder | Role/Description |
| Business Owner | Oversee the project, defines business goals. |
| End Users (Customers) | Shoppers use the platform to browse and purchase products. |
| Admin Users | Manage products and orders via the admin panel. |
| Project Manager | Coordinates the project, ensures timely delivery. |
| Development Team | Designs, develops, and tests the application. |
| QA Team | Validates the system against requirements. |
| Payment Gateway Provider | Provides payment processing services (e.g., PayPal). |

**2.6. Operating Environment**

* **Hardware Platform**: Cloud-based servers with scalability for traffic surges.
* **Operating Systems**: Cross-platform compatibility (Windows, macOS, Linux for servers; client-side OS-agnostic).
* **Browsers**: Chrome, Firefox, Safari, Edge (latest versions).
* **Network Environment**: Internet access via Wi-Fi, 4G, or 3G.
* **Coexisting Software**: The system will interact with a relational database (e.g., MySQL) and payment gateway APIs.

**2.7. System Constraints**

* **Software Constraints**: Must use open-source frameworks (e.g., React for frontend, Node.js for backend) to minimize costs.
* **Hardware Constraints**: No specific hardware requirements for end users beyond standard devices (PC, tablet, smartphone).
* **Cultural Constraints**: The system will support English only in the initial phase.
* **Environmental Constraints**: The system must function in varying network conditions (e.g., low bandwidth).
* **User Constraints**: The UI must be intuitive for users with minimal technical expertise, including graphical elements for navigation.
* **Third-Party Components**: Payment gateways may impose transaction limits or require specific integration protocols.

**2.8. Assumptions & Dependencies**

* **Assumptions**:
  + Users have access to modern web browsers and an internet connection.
  + Payment gateways will be available and functional during operation.
  + The cloud infrastructure will support scalability needs.
* **Dependencies**:
  + Availability of third-party APIs (e.g., PayPal for payments).
  + Stable internet connectivity for real-time transactions.
  + Access to a relational database system for storing user and product data.

**3. External Interface Requirements**

**3.1. Hardware Interfaces**

The system will operate on cloud-based servers and interact with end-user devices (PCs, tablets, smartphones) via web browsers. No direct hardware control is required, but the system must support:

* **Device Types**: Desktops, laptops, tablets, and smartphones with standard input/output capabilities (keyboard, touch, mouse).
* **Data Interactions**: HTTP/HTTPS requests for data exchange between client devices and servers.

**3.2. Software Interfaces**

|  |  |  |  |
| --- | --- | --- | --- |
| Component | Name/Version | Purpose | Data Exchanged |
| Database | MongoDB | Store user, product, and order data | User profiles, product details, orders |
| Payment Gateway | PayPal | Process payments | Payment requests, transaction status |
| Web Framework | React | Frontend UI rendering | UI components, user inputs |
| Backend Server | Node.js | Handle business logic and API requests | API responses, user data |

**4. Functional Requirements**

**4.1. Functional Hierarchy**

* **User Panel**
  + User Management
    - Register
    - Login
    - Update Profile
  + Product Browsing
    - View Product Catalog
    - Search Products
    - Filter/Sort Products
  + Shopping Cart
    - Add to Cart
    - Update Cart
    - Remove from Cart
  + Checkout
    - Enter Shipping Details
    - Process Payment
  + Order Management
    - View Order History
    - View Order Details
* **Admin Panel**
  + Product Management
    - Add Product
    - Update Product
    - Delete Product
  + Order Management
    - View Orders
    - View Order Details

**4.2. Use Cases**

**4.2.1. User Registration**

**Use Case ID: UC\_001**  
**Actors**: Customer (initiates the use case), System  
**Feature**: User Management  
**Pre-condition**:

* The user must have a valid email address.
* The registration page must be accessible via a web browser.

**Scenarios**

|  |  |  |
| --- | --- | --- |
| Step# | Action | Software Reaction |
| 1 | Customer navigates to the registration page | The system displays the registration form with fields for email, password, and name |
| 2 | Customer enters email, password, and name | The system validates the input fields (e.g., email format, password strength) |
| 3 | Customer submits the form | System checks if email is unique; if yes, creates a new user account |
| 4 | None | System redirects the user to the login page with a success message |

**Alternate Scenarios**  
1a: If the email format is invalid

* System displays an error: "Invalid email format."  
  2a: If the email already exists
* System displays an error: "Email already registered."  
  2b: If the password does not meet criteria (e.g., <6 characters)
* The system displays an error: "Password must be at least 6 characters."

**Post Conditions**

|  |  |
| --- | --- |
| **Step#** | **Description** |
| 1 | A new user account has been created in the database. |
| 2 | User receives a confirmation email (if enabled). |

**Use Case Cross Referenced**: UC\_002 (User Login)

**4.2.2. User Login**

**Use Case ID: UC\_002**  
**Actors**: Customer (initiates the use case), System  
**Feature**: User Management  
**Pre-condition**:

* The user must have a registered account.
* The login page must be accessible via a web browser.

**Scenarios**

|  |  |  |
| --- | --- | --- |
| Step# | Action | Software Reaction |
| 1 | Customer navigates to the login page | The system displays the login form with fields for email and password |
| 2 | Customer enters email and password | System validates the credentials against the database |
| 3 | Customer submits the form | System authenticates the user and creates a session |
| 4 | None | System redirects the user to the homepage with a welcome message |

**Alternate Scenarios**  
2a: If the email or password is incorrect

* System displays an error: "Invalid email or password."  
  2b: If the account is locked (e.g., after multiple failed attempts)
* System displays an error: "Account locked. Please contact support."

**Post Conditions**

|  |  |
| --- | --- |
| **Step#** | **Description** |
| 1 | Users are logged in and a session is created. |
| 2 | User is redirected to the homepage. |

**Use Case Cross Referenced**: UC\_001 (User Registration), UC\_003 (Add Product to Cart)

**4.2.3. Add Product to Cart**

**Use Case ID: UC\_003**  
**Actors**: Customer (initiates the use case), System  
**Feature**: Shopping Cart  
**Pre-condition**:

* The user must be logged in (or guest checkout enabled).
* The product must be in stock and available in the catalog.

**Scenarios**

|  |  |  |
| --- | --- | --- |
| Step# | Action | Software Reaction |
| 1 | Customer navigates to a product page | System displays product details (name, price, stock status) |
| 2 | Customer selects quantity (default: 1) | System validates the quantity against stock availability |
| 3 | Customer clicks "Add to Cart" | System adds the product to the user’s cart |
| 4 | None | System updates the cart icon with the new item count |

**Alternate Scenarios**  
2a: If the selected quantity exceeds stock

* System displays an error: "Requested quantity not available."  
  3a: If the product is out of stock
* System disables the "Add to Cart" button and shows "Out of Stock."

**Post Conditions**

|  |  |
| --- | --- |
| **Step#** | **Description** |
| 1 | Products are added to the user’s cart. |
| 2 | Cart item count is updated in the UI. |

**Use Case Cross Referenced**: UC\_002 (User Login), UC\_004 (Checkout Process)

**4.2.4. Checkout Process**

**Use Case ID: UC\_004**  
**Actors**: Customer (initiates the use case), System, Payment Gateway  
**Feature**: Checkout  
**Pre-condition**:

* The user must have items in the cart.
* The user must be logged in (or guest checkout enabled).

**Scenarios**

|  |  |  |
| --- | --- | --- |
| Step# | Action | Software Reaction |
| 1 | Customer clicks "Proceed to Checkout" | System displays the checkout page with cart summary |
| 2 | Customer enters shipping address | System validates the address format |
| 3 | Customer selects payment method (e.g., PayPal) | System redirects to the payment gateway |
| 4 | Customer completes payment | Payment Gateway processes the transaction |
| 5 | None | System confirms the order and sends a confirmation email |

**Alternate Scenarios**  
2a: If the address is invalid

* System displays an error: "Invalid address format."  
  4a: If the payment fails (e.g., insufficient funds)
* System displays an error: "Payment failed. Please try again."

**Post Conditions**

|  |  |
| --- | --- |
| **Step#** | **Description** |
| 1 | Order is created and stored in the database. |
| 2 | Cart is cleared for the user. |
| 3 | User receives an order confirmation email. |

**Use Case Cross Referenced**: UC\_003 (Add Product to Cart)

**4.2.5. Admin Add Product**

**Use Case ID: UC\_005**  
**Actors**: Admin (initiates the use case), System  
**Feature**: Product Management  
**Pre-condition**:

* The user must be logged in as an admin.
* The admin panel must be accessible via a web browser.

**Scenarios**

|  |  |  |
| --- | --- | --- |
| Step# | Action | Software Reaction |
| 1 | Admin navigates to the "Products" section | The system displays the product management interface |
| 2 | Admin clicks "Add New Product" | The system displays a form for product details (name, price, stock, etc.) |
| 3 | Admin enters product details | System validates the input (e.g., price must be positive) |
| 4 | Admin submits the form | System adds the product to the catalog and database |
| 5 | None | The system displays a success message: "Product added successfully." |

**Alternate Scenarios**  
3a: If the price is negative or invalid

* The system displays an error: "Invalid price. Must be a positive number."  
  3b: If required fields are missing
* System displays an error: "All required fields must be filled."

**Post Conditions**

|  |  |
| --- | --- |
| **Step#** | **Description** |
| 1 | A new product has been added to the catalog. |
| 2 | Products are available for customers to view and purchase. |

**Use Case Cross Referenced**: None

**5. Non-Functional Requirements**

**5.1. Performance Requirements**

* The system must load pages within 3 seconds under normal traffic conditions.
* The system must handle 1000 concurrent users with a response time of less than 5 seconds.
* Checkout and payment processing must be completed within 10 seconds per transaction.
* The search functionality must return results within 2 seconds for up to 10,000 products.

**5.2. Safety Requirements**

* The system must prevent accidental data loss by implementing session timeouts after 15 minutes of inactivity.
* All user inputs must be validated to prevent harmful actions.
* The system must comply with PCI DSS standards to ensure safe payment processing.
* Error messages must not expose sensitive system information (e.g., stack traces).

**5.3. Security Requirements**

* User passwords must be hashed using bcrypt before storage.
* All communications must use HTTPS with SSL/TLS encryption.
* The system must protect against common vulnerabilities such as SQL injections, XSS, and CSRF.
* Session tokens must be securely generated and expired after logout or timeout.
* Admin access must require two-factor authentication (2FA).

**5.4. User Documentation**

* **User Manual**: A PDF document detailing how to register, browse products, manage the cart, checkout, and track orders.
* **Admin Guide**: A PDF guide for admins on managing products and orders.
* **Online Help**: A help section within the application with FAQs and step-by-step guides.
* **Context-Sensitive Help**: Tooltips and inline help for complex forms (e.g., checkout, product addition).

**6. References**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Document Title | Report Number | Date | Publishing Organization | Source |
| PayPal API Documentation | v2.0 | 2024-01-15 | PayPal | https://developer.paypal.com |

**7. Appendices**

* **Appendix A**: Glossary of Terms
  + **Cart**: A temporary storage for products selected by the user before checkout.
  + **Catalog**: A collection of products available for purchase.
* **Appendix B**: System Architecture Diagram
  + A high-level diagram showing the interaction between the frontend (React), backend (Node.js), database (MySQL), and external APIs (PayPal).  
    (Note: Diagram to be added during the design phase.)