**SRS for Bookish e-commerce web application**

**Scope:**

This software system will allow users to buy different books via the Internet. The system will be run on a central server with each user having a remote user interface through a web browser to interact with it. The Bookish E-Commerce System will allow any user to create an account to become a customer. Through the process of account creation, the customer will have the option to become a seller of the site.

**Description:**

Bookish is an online platform dedicated to book enthusiasts, providing a seamless experience for buying and selling books. The platform aims to connect readers with their favorite titles while empowering independent sellers to showcase their collections. Whether users are looking for the latest bestsellers, rare collectibles, or academic textbooks, Bookish offers a diverse selection to cater to every reading preference.

**1. Requirement Analysis:**

**a. User Requirements:**

**User Roles:** Define two user roles: customers and sellers where sellers can sell their books.

**User Authentication:** Users should be able to register, log in, and manage their accounts securely. We will handle two types of authentications user and seller.

**Profile Management:** Users should be able to update their profile data, view order history, and manage preferences. We will keep track of books viewed by users to make further predictions.

**Product Browsing:** Users should be able to search, filter, and browse books based on genre, author, and other attributes.

**Shopping Cart:** Users should be able to add, remove, and manage items in the shopping cart. When the user makes a count of book zero then it should not be removed from the cart.

**Checkout Process:** Users should be guided through a seamless checkout process, with multiple payment options.

**b. System Requirements:**

**Performance:** The system should handle a large number of concurrent users and transactions efficiently. For performance, there are different factors such as Cloud infrastructure, Distributed architecture, Caching mechanisms to store frequently accessed data in memory, or fast storage systems. In Bookish we will use Cloudinary to store images and video data.

**Scalability:** The system should be scalable to accommodate growth in user base and inventory. **Security:** The system should ensure data encryption like, secure authentication, and protection against common security threats. The system will use tools like Bcrypt, and JWT. Bookish will use robust authentication mechanisms to verify the identity of users. This includes strong password policies, multi-factor authentication (MFA), and CAPTCHA challenges to prevent unauthorized access to user accounts.

**Availability:** The system should have high availability, with minimal downtime for maintenance and updates.

**Compatibility:** The system should be compatible with popular web browsers and devices. The Bookish web application will be available on any popular web browsers such as Chrome, Brave, etc.

**2. Functional Specifications:**

The landing page of our website would have different books, ongoing offers, etc. Users can browse books without registering. Users can only access some functionalities after logging in such as add to cart, Buy, etc.

**a. User Management:**

Users have to register on the Bookish web app, after successful registration users can perform any activities such as browsing books, adding them to a cart, etc.

**Registration:** Users can register by providing the necessary details.

**Authentication:** Users can log in using email/password or social media accounts.

Profile Management: Users can update personal information, manage addresses, and view order history.

**b. Product Management:**

**Book Listings:** Sellers can add, edit, and delete book listings with details like title, author, description, and images. For the customer book listing page will contain only books and it can not update it.

**Search and Filter:** Users can search for books and filter results by genre, author, price range, etc.

**Product Details:** Users can view detailed information about each book, including ratings, reviews, and descriptions.

**c. Shopping Cart and Checkout:**

**Add to Cart:** Users can add books to the shopping cart and adjust quantities. The shopping cart would contain the total no of books added by the user and the user can adjust the quantities.

**Checkout Process:** Users can review their cart, select shipping options, and complete the purchase securely.

**d. Order Management:**

**Order History:** Users can view their order history, track shipments, and manage returns.

**Order Fulfillment:** Sellers can process orders, update order status, and manage inventory.

**3. External Interface Specifications:**

**a. User Interface:**

Web Interface: The application will have a responsive web interface accessible from desktop and mobile browsers.

**b. Communication Protocol:**

HTTP/HTTPS Protocol: The application shall use HTTP/HTTPS protocol for communication between clients (web browsers) and the server to ensure secure data transmission.

RESTful API: The server shall expose RESTful APIs to facilitate communication with client-side components for retrieving and updating data.

WebSockets: Optionally, WebSockets may be implemented for real-time communication features such as chat or notifications.

**C.Hardware Interface:**

Device Compatibility: The application shall be compatible with various hardware devices, including desktop computers, laptops, tablets, and smartphones, ensuring a seamless user experience across different platforms.

Responsive Design: The user interface shall adapt to different screen sizes and resolutions, optimizing layout and functionality for each device.

**d. Database Backend:**

The database backend stores and manages data related to books, users, orders, and other application entities.

Relational Database: The backend shall use a relational database management system (RDBMS) such as MySQL or PostgreSQL to store structured data with ACID properties.

**User Data Model:** we will take email, full name, password, mobile number, and profile picture as avatar.

**Book Data Model:** In the book data model we will have a title, author, publish date, description price, and stock. We store all images of books on the Cloudinary platform.

**Cart Data Model:** The cart will have a reference to a book data model.

**4. Technical Specifications:**

1. **Performance Constraints:**

Performance constraints define the limitations and requirements related to the speed, responsiveness, and scalability of the Bookish web application.

**Response Time:** The application shall respond to user interactions within a maximum of **2** seconds under normal load conditions.

**Page Load Time:** Web pages shall load within **3** seconds for optimal user experience.

**Scalability**: The application shall be designed to handle a minimum of **10,000** concurrent users without significant degradation in performance.

**Throughput:** The system shall support a minimum of **1000** transactions per minute during peak usage periods.

**Resource Utilization:** CPU and memory utilization shall not exceed 70% under maximum load conditions to ensure system stability and responsiveness.

1. **Memory and Operating System (OS):**

Memory and OS requirements specify the hardware and software environment necessary to run the Bookish web application efficiently.

**Memory:** The server hosting the application shall have a minimum of 4 GB RAM to ensure smooth operation and handle concurrent user requests effectively.

**Operating System:** The application shall be compatible with major operating systems such as Linux, Windows, and macOS for both development and production environments.

**Web Server:** The application shall be deployed on a web server compatible with the chosen operating system, such as Apache or Nginx, configured to handle HTTP/HTTPS traffic efficiently.

**Middleware:** Optionally, middleware components such as Node.js or Django may be used to enhance application performance and functionality.

**Browser Compatibility:** The client-side components of the application shall be compatible with modern web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari, ensuring consistent performance and user experience across different platforms.