Q. Write a SRS document for an ecommerce application.

Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose

The purpose of this document is to outline the functional, non-functional, and technical requirements for the development of an eCommerce application. This application will enable users to browse and purchase products online while providing an administrative interface for managing products, orders, and users.

1.2 Scope

The eCommerce application will consist of two interfaces:

1. **User Interface (UI):** For customers to browse products, manage their cart, and place orders.
2. **Admin Panel:** For administrators to manage inventory, orders, and users.

The system will include core functionalities such as user registration, product browsing, secure checkout, order tracking, and payment integration.

1.3 Audience

This document is intended for the following stakeholders:

- Project Managers
- Development Team
- Quality Assurance Team
- Business Analysts

---

2. Overall Description

2.1 Product Perspective

This application will be developed as a standalone system with RESTful APIs to interact with frontend applications. It will be scalable to support web and mobile platforms.

2.2 Product Features

- User registration and login (including social logins).
- Product browsing with advanced search and filters.
- Shopping cart functionality.
- Secure payment gateway integration.
- Order management and tracking.
- Admin dashboard for product and order management.

2.3 User Classes and Characteristics

1. **End Users:**
    - Casual or frequent buyers with minimal technical expertise.
2. **Admin Users:**
    - Moderators and business owners responsible for managing the platform.

2.4 Operating Environment

- **Frontend:** Compatible with modern browsers (Chrome, Firefox, Safari).
- **Backend:** Hosted on a cloud platform (e.g., AWS or GCP) with APIs built in Django.
- **Database:** PostgreSQL for relational data and MongoDB for catalog flexibility.
- **Payment Gateway:** Stripe, PayPal, or Razorpay.

---

3. Functional Requirements

3.1 User Management

- Users can register using email or social logins.
- Users can log in, log out, and manage their profiles.
- Forgot password and account recovery options.

3.2 Product Management

- Products will be displayed with images, descriptions, and prices.
- Users can search and filter products by categories, price range, and ratings.
- Real-time stock updates for each product.

3.3 Shopping Cart

- Users can add, update, or remove items from their cart.
- The cart will display item quantity, price, and total cost.
- Option to save cart items for later.

3.4 Order Processing

- Users can proceed to checkout, enter shipping details, and place orders.
- Orders will have statuses: Pending, Shipped, Delivered, Cancelled.
- Notifications for order updates via email/SMS.

3.5 Payment Integration

- Secure payment gateway for processing credit/debit cards, UPI, and wallets.
- Generate payment confirmation receipts.

- Support for multiple currencies.

3.6 Admin Dashboard

- Manage products (add, update, delete).
- View and process orders.
- Generate sales reports and analytics.
- User management (view, block, or delete users).

---

4. Non-Functional Requirements

4.1 Performance

- The system should handle 1000 concurrent users.
- API responses must be under 2 seconds.

4.2 Security

- Implement JWT-based authentication for APIs.
- Use HTTPS for all communications.
- Follow OWASP guidelines to prevent vulnerabilities.

4.3 Usability

- The application will have an intuitive user interface.
- Provide multi-language support for international users.

4.4 Scalability

- The system should scale horizontally to handle increased traffic.

4.5 Availability

- The system should maintain 99.9% uptime.

---

5. Assumptions and Constraints

5.1 Assumptions

- Users have access to stable internet.
- Payment gateways will handle fraud detection and compliance.
- Admins are responsible for maintaining accurate inventory records.

5.2 Constraints

- Development will follow a six-month timeline.
- APIs will adhere to RESTful standards.

---

6. Appendices

- **A. Glossary:**
    - JWT: JSON Web Token
    - API: Application Programming Interface
- **B. References:**
    - Google Maps API for location services.
    - Stripe API documentation for payment processing.