**User-Side Responsibilities**

### **Functional Requirements:**

**Account and Profile Management**

* Users can register with email, password, and phone, with optional social media login.
* Users can log in securely with email and password, with optional multi-factor authentication.
* Users can update personal details, manage addresses, and view order history.
* Implementation Possibility: The development team can do that.

**Product Interaction**

* Products display titles, descriptions, images, prices, stock quantities, and categories.
  + Implementation Possibility: (Simple)
* Users can search for products with autocomplete and filter by price, brand, rating, etc.
  + Implementation Possibility: (Basic checks, feasible)
* Users can submit and view reviews and ratings on product pages.
  + Implementation Possibility: Easy to implement.

**Shopping Cart and Checkout**

* Users can add, remove, and adjust items in the shopping cart.
  + Implementation Possibility: Easy with backend team.
* The checkout process allows address selection, payment method choice, and order review.
* Secure integration with Stripe payment gateway to handle payments, supporting multiple payment methods.
  + Implementation Possibility: Possible with form validation and basic payment API.

**Order Management**

* Users can track their orders through stages like Processing, Shipped, and Delivered.
  + Implementation Possibility: Easy, feasible with backend team.
* Users can initiate and track return or refund requests.
  + Implementation Possibility: Possible with Customer Support team.
* Past orders are displayed with product details, order date, and payment information.
  + Implementation Possibility: Feasible with correct data storage.

**Customer Support**

* FAQ section addresses common issues and provides assistance.
* Users can access live chat or submit tickets for help.
  + Implementation Possibility: Possible by providing an interactive chat or social media.

**Engagement and Promotions**

* Users can apply discounts and coupons at checkout.
* Automated emails are sent for promotions, abandoned carts, and product recommendations.
* Recommendations display personalized suggestions based on browsing and purchase history.
  + Implementation Possibility: Achievable with basic backend rules and logic.

### **Non-Functional Requirements:**

**Usability**

* The interface is user-friendly, responsive, and consistent across mobile and desktop.
  + Implementation Possibility: Possible.
* Clear error messages guide users during login, checkout, or other issues.
  + Implementation Possibility: Possible.

**Performance**

* Pages and actions, like search or add-to-cart, load in under 2 seconds.
  + Implementation Possibility: Possible.

**Accessibility**

* The platform is compatible with screen readers, high contrast modes, and keyboard navigation.
  + Implementation Possibility: Possible.

**System-Side Responsibilities**

### **Functional Requirements:**

**Scalability and Performance**

* The system handles high concurrent users and traffic spikes without performance degradation.
  + Implementation Possibility: Difficult; requires advanced infrastructure, possibly in the future.
* Real-time updates maintain accurate product stock and inventory levels.
  + Implementation Possibility: Possible with a simple database setup.

**Security and Data Protection**

* Sensitive data like passwords and payment details is encrypted.
  + Implementation Possibility: Achievable with basic encryption methods.
* Role-based permissions control access to data and features by user roles (admin, seller, customer).
  + Implementation Possibility: Possible with simple backend configurations.

**Reliability and Uptime**

* 99.9% uptime ensures uninterrupted shopping and order processing.
  + Implementation Possibility: Requires strong hosting; possible in the future.

**Compliance and Policies**

* The system complies with GDPR, CCPA, and data protection standards.
  + Implementation Possibility: Possible by adding a consent option during registration.
* Product listing policies ensure compliance with legal and regulatory requirements.
  + Implementation Possibility: Achievable with basic rules in product management.

### **Non-Functional Requirements:**

**Scalability**

* Elastic infrastructure adapts to handle increasing traffic and user growth.
  + Implementation Possibility: Possible with cloud tools like AWS; can be implemented in the future.

**Security**

* Personal and payment data is encrypted at rest and in transit following industry standards.
  + Implementation Possibility: Feasible using built-in cloud encryption options.
* Fraud detection mechanisms monitor and prevent unauthorized activities.
  + Implementation Possibility: Possible in the future.

**Maintainability**

* The system’s architecture is modular, enabling easier updates and feature additions.
  + Implementation Possibility: Achievable with well-organized code structure.

**Compliance and Governance**

* Designed to meet data privacy standards, adapting to future regulatory changes.
  + Implementation Possibility: Basic compliance is achievable by handling data securely and updating policies.

**Inverse Requirements**

**No Guest Checkout**

* **User-Side**: Users must register and log in before proceeding to purchase, ensuring a secure and trackable shopping experience.
* **System-Side**: The system enforces login before checkout, disabling guest checkout options and maintaining session consistency for order tracking and support.
  + Implementation Possibility: Implementable with basic PHP or JavaScript session management.

**No Unsecured Payment Processing**

* **User-Side**: Users complete payments only through secure, integrated gateways to protect financial data.
* **System-Side**: All payment processing is routed through PCI DSS-compliant, integrated payment gateways, blocking external or insecure payment methods.
  + Implementation Possibility: Achievable using APIs.

**No Unauthorized Access to Admin Features**

* **User-Side**: Only authenticated administrators have access to features like order management, inventory, and reports.
* **System-Side**: Role-based permissions restrict access to admin functions, preventing unauthorized users from accessing management features.
  + Implementation Possibility: Start by defining roles (e.g., "user" and "admin").

**No User-to-User Communication**

* **User-Side**: Users cannot interact or communicate directly with one another, ensuring privacy.
* **System-Side**: No direct messaging or communication channels exist between users, enforcing privacy and security.
  + Implementation Possibility: Avoid designing chat features between users.

**No Modification of Critical Data by Users**

* **User-Side**: Users cannot alter historical order details or transaction records post-completion, maintaining data accuracy.
  + Implementation Possibility: Add a field in the database to indicate if data can be modified post-order completion.

**Design Constraints**

**Security Compliance**

* **User-Side**: Users’ personal and financial data is securely stored and handled, complying with GDPR and PCI DSS regulations.
  + Implementation Possibility: Feasible by using secure payment gateways and updating privacy policies.
* **System-Side**: All personal data is stored in encrypted formats, using secure hashing for passwords, ensuring compliance.
  + Implementation Possibility: Achievable with libraries for encryption and hashing.

**Mobile Compatibility**

* **User-Side**: The platform provides a responsive, optimized experience on mobile devices, ensuring functionality across devices.
  + Implementation Possibility: Achievable using responsive design frameworks like Bootstrap.
* **System-Side**: The user interface adapts to various screen sizes, enhancing accessibility.
  + Implementation Possibility: Feasible with responsive CSS and frameworks; accessible for beginners with basic web development skills.