Library Management System

Backend (RESTful API for a Book Store):

1. Authentication and Authorization:

- Implement JWT authentication for user login and registration.
- Use bcrypt for securely storing user passwords.
- Extend user details to include additional information like name, address, etc., and store them securely in the 'users' collection.

2. Book Store Functionality:

- **Sell Book:**
  - Display title, description, price, and availability from the 'books' collection.
  - Ensure that only books with units greater than 0 are displayed for sale.
  - Implement email notifications to inform users about successful purchases.
  - Generate receipts for each purchase transaction and send them to the user's email.

- **Purchase Book:**
  - Store purchase details including user email and book ID in the 'purchases' collection.
  - Deduct the purchased book's units from the stock after a successful purchase.

- **Add More Books to Stock:**
  - Allow authorized users to add new books to the stock.
  - If the book is not present in the booklist, create a new document for that book in the 'books' collection.
  - Send email notifications to the admin or relevant personnel about new book additions.

3. Unit Testing and Documentation:

- Write comprehensive unit tests for all API endpoints using Mocha and Chai.
- Cover edge cases and error scenarios to ensure robustness.
- Include detailed documentation for the API, describing each endpoint, its purpose, expected inputs, and outputs.
- Provide examples of API usage and response formats for better understanding.

4. Security and Usability Enhancements:

- Implement rate limiting and throttling to prevent abuse and ensure fair usage of the API.
- Apply input validation and sanitise user inputs to prevent injection attacks.
- Utilize HTTPS for secure communication between the front end and back end.

- Implement logging and monitoring mechanisms to track API usage and identify potential issues.
- Set up CORS (Cross-Origin Resource Sharing) to restrict access to the API from unauthorized domains.
- Include error handling middleware to provide meaningful error messages to the users.

Frontend (ReactJS Interface):

1. User Authentication:

- Create login and registration forms for users to authenticate.
- Integrate JWT authentication with the backend API for secure user authentication.

2. Book Store Interface:

- Design an intuitive and user-friendly interface to display available books for sale.
- Implement features like search, sorting, and filtering to enhance user experience.
- Provide options for users to view book details, add books to a cart, and proceed with the purchase.
- Display notifications for successful purchases and errors.

3. Admin Dashboard:

- Develop an admin dashboard to manage book inventory and user accounts.
- Allow admins to add new books, update existing ones, and view purchase history.
- Implement role-based access control to restrict admin functionalities to authorized users.

4. Responsive Design:

- Ensure that the frontend interface is responsive and accessible across different devices and screen sizes.
- Utilize CSS frameworks like Bootstrap or Material UI for building responsive layouts and components.

5. Integration with Backend:

- Integrate frontend components with backend API endpoints for fetching book data, user authentication, and purchase transactions.
- Handle asynchronous operations and manage state effectively using React hooks or Redux.

6. Error Handling and Feedback:

- Provide users with meaningful error messages and feedback in case of invalid inputs or failed operations.
- Implement loading indicators and progress bars to indicate ongoing operations and improve user experience.