🏛 BookNest: Your Online Bookstore - Discover Amazing Books

`

Introduction:

Welcome to BookNest, a revolutionary online bookstore application meticulously crafted with the powerful MERN (MongoDB, Express.js, React, Node.js) Stack. BookNest is designed to seamlessly connect bibliophiles with an extensive collection of literary works, redefining how readers explore, discover, and indulge in their literary pursuits. Our platform offers a visually enchanting and interactive interface, blending robust functionality with an intuitive user experience for discovering new releases and revisiting timeless classics.

Built with a scalable and efficient database infrastructure powered by MongoDB, a responsive and efficient server established by Express.js, and high-performance, non-blocking I/O operations ensured by Node.js, BookNest promises a seamless and enjoyable user journey. At its heart, React provides a dynamic and feature-rich JavaScript library for the visually appealing and interactive user interface, which is responsive across desktops, tablets, and smartphones. BookNest transforms how you connect with literature, making the discovery of your next favorite read an effortless and enriching experience, where every book is just a click away.

Key Features:

BookNest offers a comprehensive set of features for various user roles, ensuring a streamlined and secure online bookstore experience.

- **Authentication:** Secure user registration and login are implemented using JWT (JSON Web Tokens) for stateless authentication. Protected routes require a valid JWT, and logging out clears the JWT from the client. The system supports roles, including 'user' (buyer) and 'admin'.
- **Book Management:**
    - **Browse & Search:** All users can browse and search for books based on title, author, description, and price. Users can also select specific categories or genres to filter their Browse experience.
    - **Admin-only Management:** Administrators have dedicated pages to create, edit, or delete book entries, each with a title, author, description, price, stock, and cover image.
- **Wishlist:** Authenticated users can add and remove books from their personal wishlist. The wishlist items are persisted across sessions by storing them in the browser's
- localStorage.
- **Cart & Orders:**
    - **Shopping Cart:** Users can add books to a shopping cart, which is also persisted in localStorage for continuity across sessions, even on reload.

  - **Order Placement:** On checkout, a simulated order is placed (without real payment). Order details, including items, quantities, total price, and date, are stored in the database.

  - **Order History:** Users can view their past and current orders, track shipments, and review purchased books.

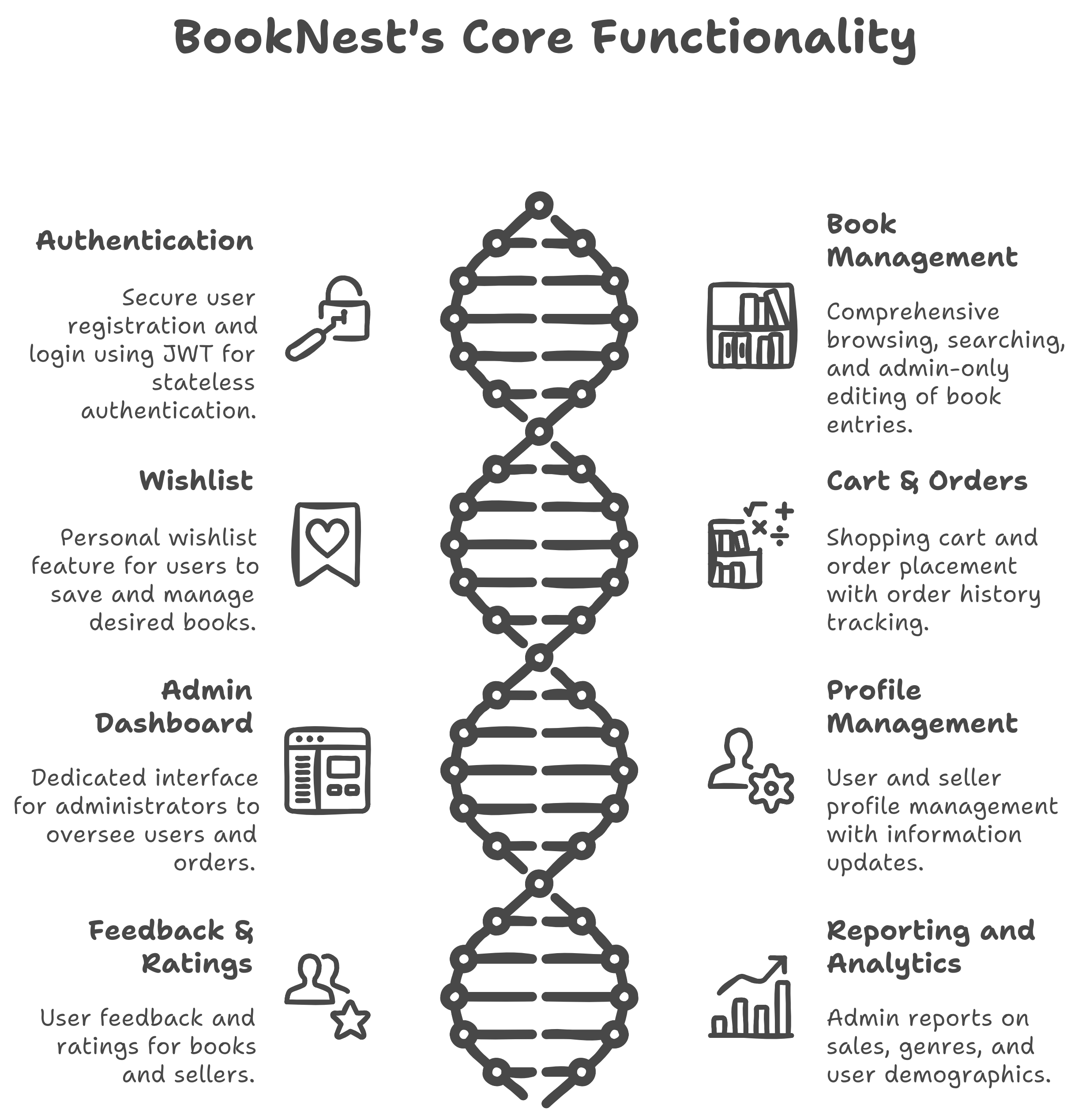
- **Admin Dashboard:** Administrators have a dedicated interface to oversee all users and all orders, and manage any book entry. Admin-specific routes are protected by middleware that verifies the

- req.user.role as 'admin'.

- **Profile Management:** Users can manage their profiles, updating information like email, name, and password. Sellers (if applicable) can manage their business name, email, and password.

- **Feedback & Ratings (https://www.google.com/search?q=Optional/Mentioned):** Users can provide feedback and ratings for purchased books and sellers.

- **Reporting and Analytics (Admin):** Admins can generate reports and analytics on book sales, popular genres, and user demographics to gain insights into platform usage and performance.


BookNest's Core Functionality
Authentication
Secure user registration and login using JWT for stateless authentication.
Book Management
Comprehensive browsing, searching, and admin-only editing of book entries.
Wishlist
Personal wishlist feature for users to save and manage desired books.
Cart & Orders
Shopping cart and order placement with order history tracking.
Admin Dashboard
Dedicated interface for administrators to oversee users and orders.
Profile Management
User and seller profile management with information updates.
Feedback & Ratings
User feedback and ratings for books and sellers.
Reporting and Analytics
Admin reports on sales, genres, and user demographics.

Description:

BookNest is a user-centric platform designed to make the process of discovering and purchasing books easy and efficient. The application connects readers with an extensive library of literary works through a streamlined digital interface, allowing users to search, filter by category/genre, and securely purchase books.

For everyday users (buyers), BookNest offers secure registration and login, a persistent wishlist, and a shopping cart that retains items across sessions. Automated order confirmation and the ability to view past orders enhance the purchasing experience. Sellers, upon admin approval, can manage their book listings, including adding new titles, updating inventory levels, and fulfilling orders. An integrated admin dashboard allows for comprehensive oversight, including user and order management, book listing approvals, and platform policy enforcement.

The application is built using the MERN stack: React for a responsive and intuitive frontend , coupled with Axios/Fetch for seamless backend communication. The backend, powered by Express.js, handles server-side logic, while MongoDB provides scalable data storage for user profiles, book details, and order information. Authentication is secured using JWT for session management and bcrypt for password hashing. The frontend is mobile-responsive, utilizing CSS media queries to ensure a consistent and delightful experience across various devices.

Scenario-Based Case Study:

1. User (Buyer) Registration and Book Purchase:
John, an avid reader, wants to find new fantasy novels. He visits the BookNest website. He clicks "Sign Up" and securely registers an account by providing his name, email, and a password. After successful registration, he logs in.

Once logged in, John navigates to the "Books" section. He uses the search bar to look for "fantasy novels" and filters by the "Fantasy" genre. He finds a book titled "The Dragon's Legacy" and clicks on it to view details. He decides to buy it and clicks "Add to Cart". The book is added to his shopping cart, and the cart contents persist even if he closes and reopens his browser.

John then proceeds to checkout. He reviews the items in his cart and confirms the purchase (simulated payment). An order is successfully placed, and he receives an immediate confirmation with his order ID and details. John can later go to his "Orders" page to view his purchase history.

2. Seller Listing a Book:
Sarah, who owns a small independent bookstore, decides to sell some of her inventory on BookNest. She registers as a seller on the platform, providing her business name, email, and password. Upon successful registration, her account awaits admin approval.

Once her seller account is approved by an administrator, Sarah logs in. She navigates to her seller dashboard (assuming a dedicated interface or access through the admin panel) and selects the option to "Add New Book." She fills in the details for a new book: title ("Ancient History Explained"), author ("Dr. Eleanor Vance"), description, price, stock quantity, and uploads a cover image. After submitting, the book is now listed for sale on BookNest and becomes visible to buyers. She can later update the stock or edit book details from her dashboard.

3. Admin Managing the Platform:
The BookNest administrator logs into the admin dashboard. The dashboard provides an overview of "Total Books," "Total Orders," "Total Users," and "Total Revenue".

The admin reviews new seller registrations and sees Sarah's pending application. After verifying her details, the admin approves her seller account, making her books available to users. The admin also monitors overall platform operations, addressing any issues and ensuring compliance with platform policies. They can also view a comprehensive list of all users and all orders placed on the platform. If a user reports an issue with a book listing, the admin can easily locate and edit or delete that book from the system using the book management features.

Technical Architecture:

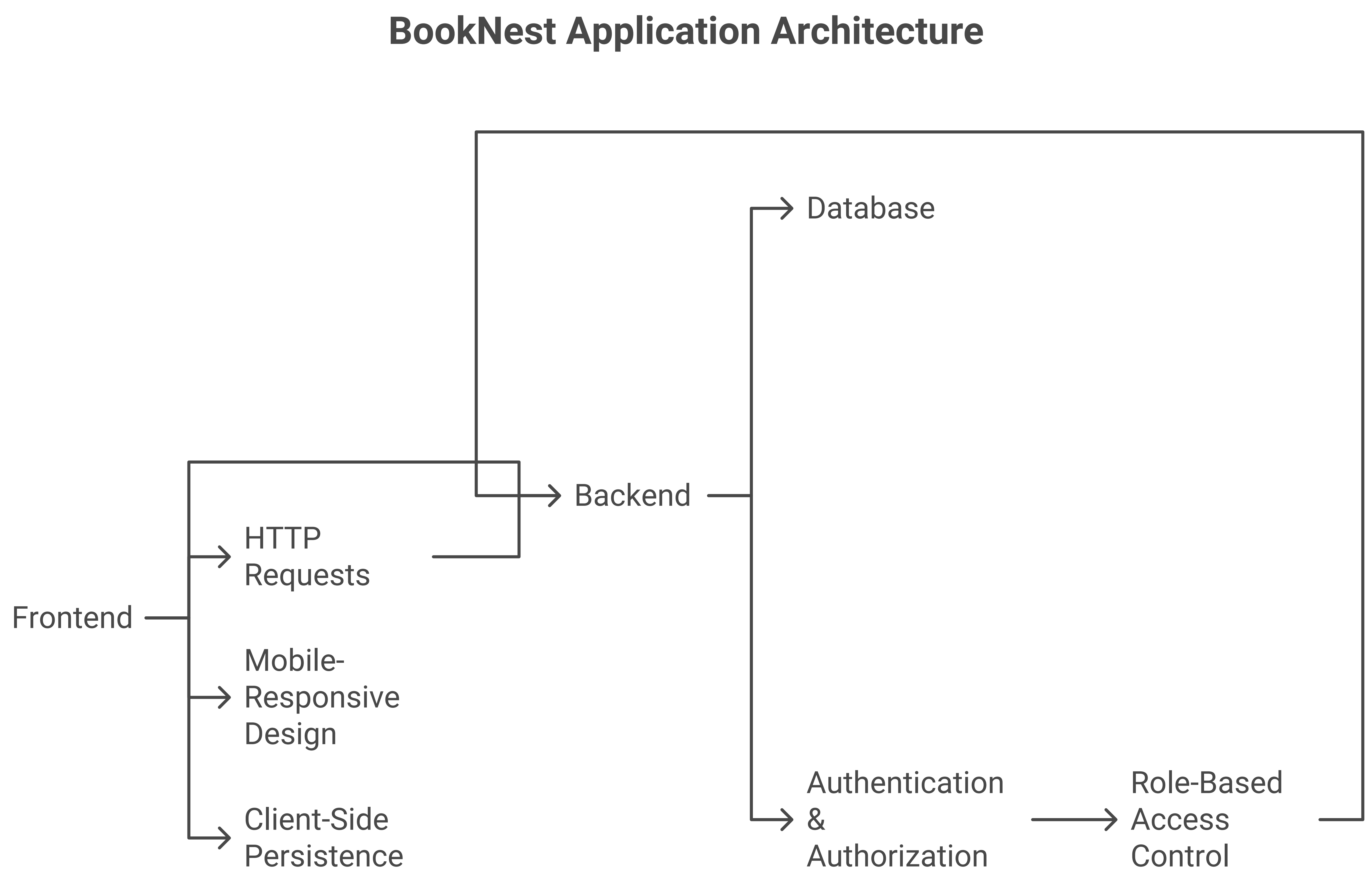
The BookNest application features a modern technical architecture based on a client-server model.

- **Frontend:** The frontend leverages **React.js** for building interactive and reusable user interfaces. It uses functional components with Hooks (e.g.,
- useState, useEffect) and **React Router** for navigation between pages (Home, Books, Wishlist, Cart, Orders, Login, Register, Admin Panel). For HTTP requests,
- **Axios** or the native Fetch API are used for seamless API communication. The user interface is mobile-responsive, achieved through plain
- **CSS** and **media queries**.

- **Backend:** The backend is powered by **Node.js** and **Express.js**, offering robust server-side logic and handling RESTful API endpoints. It exposes JSON REST endpoints and uses Mongoose schemas to model data. Key components include controllers for handling logic (e.g.,
- authController.js, bookController.js, orderController.js, wishlistController.js, userController.js), middleware for authentication and authorization, and routes for defining API endpoints.

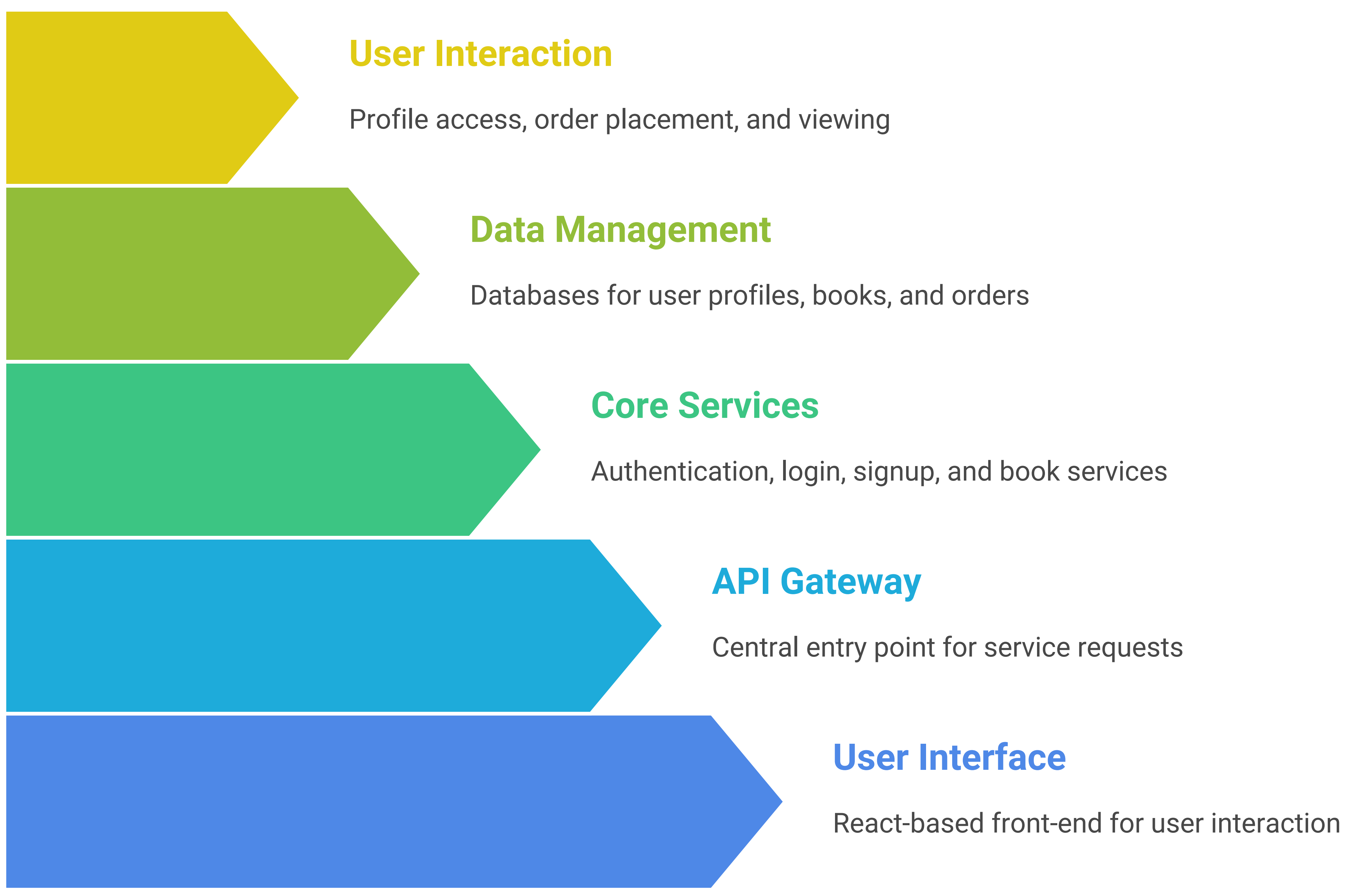
- **Database:** **MongoDB** is used as the NoSQL database for flexible and scalable storage of user data, book profiles, and order records.
- **Mongoose**, an Object-Document Mapping (ODM) library, is used to define schemas and interact with MongoDB from the Node.js backend.

- **Authentication & Authorization:** Secure authentication is handled using **JWT (JSON Web Tokens)** for stateless session management.
- **Bcrypt** is used for hashing passwords before storing them in the database, ensuring sensitive data security. Role-based access control (RBAC) is implemented via middleware to protect admin-specific routes.
- **Other Notable Technologies:**
  - localStorage is used for client-side persistence of the shopping cart and wishlist items across sessions.
  - cite_start A seeding script (seed/seed.js) can be used to populate initial books and an admin user.


BookNest Application Architecture
Database
Backend
HTTP Requests
Frontend
Mobile-Responsive Design
Client-Side Persistence
Authentication & Authorization
Role-Based Access Control

Technical Architecture Flow Diagram:

BookNest System Architecture


User Interaction
Profile access, order placement, and viewing
Data Management
Databases for user profiles, books, and orders
Core Services
Authentication, login, signup, and book services
API Gateway
Central entry point for service requests
User Interface
React-based front-end for user interaction

ER Diagram:

The Entity-Relationship (ER) diagram for the BookNest application represents key entities such as Users, Books, Orders, and their respective attributes and relationships.

- **User Entity:** Includes name, email (unique), password (hashed), role (user, admin), and wishlist (array of Book IDs). This represents both buyers and administrators.

- **Book Entity:** Includes title, author, description, price, stock, and coverImage (URL/path).

- **Order Entity:** Includes user (reference to User), items (array of objects, each with book reference, quantity, price), total, and createdAt.

Relationships:

- **User-Book Relationship (Wishlist):** A User can have multiple Books in their Wishlist (One-to-Many relationship implicitly, as wishlist is an array of Book IDs in the User schema).

- **User-Order Relationship:** A single User can place multiple Orders, but each Order belongs to one User (One-to-Many).

- **Book-Order Relationship (Items):** An Order can contain multiple Books (via its items array), and a Book can be part of many Orders (Many-to-Many, implicitly handled by the items array in the Order schema).
- **Book-Inventory Relationship:** Each book can have multiple copies in inventory, but each copy belongs to one book (One-to-Many). This is reflected by the
- stock attribute in the Book schema.
- **Book-Author Relationship:** A book can have multiple authors, and an author can write multiple books (Many-to-Many). (This relationship would typically require an intermediate table/model, as suggested in the provided text for "WrittenBy", though not explicitly defined in the provided schema snippets).
- **Book-Genre Relationship:** A book can belong to multiple genres, and a genre can have many books (Many-to-Many). (Similar to Book-Author, an intermediate "CategorizedAs" table would be ideal).
- **Review-User Relationship:** A review is written by one user, but a user can write many reviews (Many-to-One, with UserID as a foreign key in a Review table, as suggested).

(The provided ER-Diagram image is a Use Case Diagram, showing actors and their interactions, rather than a traditional Entity-Relationship Diagram with entities and attributes. The textual description of ER relationships provides the basis for the above ER description.)

Pre-Requisites:

To develop and run the BookNest application, the following prerequisites are essential:

- **Node.js and npm:** Required to run JavaScript on the server-side and manage project dependencies.
  - Download:
  - https://nodejs.org/en/download/

  - Installation instructions:
  - https://nodejs.org/en/download/package-manager/
- **MongoDB:** A NoSQL database to store application data like user profiles, book details, and order information.
  - Download:
  - https://www.mongodb.com/try/download/community

  - Installation instructions:
  - https://docs.mongodb.com/manual/installation/
- **Express.js:** A web application framework for Node.js, used for server-side routing and API development.
  - Installation:
  - npm install express
- **React.js:** A JavaScript library for building interactive and reusable user interfaces.

- Quick Start with Vite (recommended):
- npm create vite@latest, cd my-app, npm install, npm run dev

- Alternatively, Create React App:
- npx create-react-app your-app-name

- **HTML, CSS, and JavaScript:** Basic knowledge of these web technologies is crucial for structuring, styling, and adding interactivity to the user interface.

- **Database Connectivity (Mongoose):** An Object-Document Mapping (ODM) library to connect Node.js with MongoDB and perform CRUD operations.

- **Version Control (Git):** Essential for collaboration and tracking changes throughout development.

    - Download:
    - https://git-scm.com/downloads

- **Development Environment:** A code editor like Visual Studio Code, Sublime Text, or WebStorm.

Setup and Installation Instructions:

The BookNest application consists of separate frontend and backend components. Both need to be set up and run for the application to function correctly.
1. Clone the Project Repository: Download the project files from GitHub or clone the repository using Git. git clone <your-repository-url> (Replace <your-repository-url> with the actual URL of your project repository.)
2. Backend Configuration:

- **Navigate to Backend Directory:**

- cd Book-Store/backend (or bookstore-backend/ if following the structure from)

- **Install Dependencies:** npm install This installs necessary packages like
- express, mongoose, bcryptjs, jsonwebtoken, cors, dotenv, and multer.

- **Configure MongoDB:** Create a .env file in the backend directory. touch .env Add your MongoDB Atlas connection URI and a JWT secret to this file:

```
MONGODB_URI=mongodb+srv://<user>:<pass>@cluster0.mongodb.net/bookstore
JWT_SECRET=Your_Strong_JWT_Secret_Key
PORT=5000
```

- (Replace
- <user>, <pass>, cluster0.mongodb.net/bookstore with your actual MongoDB Atlas credentials and cluster details. Replace

- Your_Strong_JWT_Secret_Key with a strong, random string.)

- **(https://www.google.com/search?q=Optional) Seed Initial Data:** If you have a
- seed directory with seed.js (as described in the document), you can run it once to populate initial books and an admin user:

- node seed/seed.js

- **Start the Backend Server:**

- npm run dev (if using nodemon for development) or npm start.

- The backend API will listen on the specified PORT (e.g.,
- http://localhost:5000).

3. Frontend Configuration:

- **Navigate to Frontend Directory:** cd ../frontend (from the backend directory) or cd Book-Store/frontend
- **Install Dependencies:** npm install This installs required libraries like
- react, react-router-dom, axios, etc..

- **Set Up Environment Variables:** Create a .env.local file (or .env) in the frontend directory. touch .env.local Specify the backend API URL:

```
REACT_APP_API_URL=http://localhost:5000/api
```

- (Adjust the URL if your backend is running on a different port or deployed elsewhere.)

- **Start the Frontend Development Server:**

- npm run dev (if using Vite) or npm start (if using Create React App).

- The frontend application will typically be available at
- http://localhost:3000 or http://localhost:5173.

4. Verify the Application:

- **Check Frontend:** Open your web browser and go to http://localhost:3000 (or http://localhost:5173). The BookNest application should load with the landing page, allowing you to browse books, login, or register.

- **Check Backend:** Use a tool like Postman to test backend endpoints, such as POST http://localhost:5000/api/auth/register or GET http://localhost:5000/api/books to ensure the API services are running correctly.

Project Structure:

The project is structured into distinct Frontend and Backend components, ensuring a modular and organized codebase.

```
project-root/
├── backend/               # Node.js Express server
│   ├── config/            # Database connection setup (db.js)
│   │   └── db.js
│   ├── controllers/        # Business logic for routes (authController, bookController,
etc.)
│   │   ├── adminController.js
│   │   ├── authController.js
│   │   ├── bookController.js
│   │   ├── orderController.js
│   │   ├── userController.js
│   │   └── wishlistController.js
│   ├── middleware/         # JWT authentication and role-based access control
│   │   ├── authMiddleware.js
│   │   └── roleMiddleware.js
│   ├── models/            # Mongoose schemas for MongoDB collections (User, Book,
Order) [cite: 246, 255]
│   │   ├── Book.js
│   │   ├── Order.js
│   │   └── User.js
│   ├── routes/            # API endpoint definitions (auth, books, wishlist, orders,
admin) [cite: 246, 281]
│   │   ├── admin.js
│   │   ├── auth.js
│   │   ├── books.js
│   │   ├── orders.js
│   │   └── wishlist.js
│   ├── seed/              # Optional: Script for seeding initial data [cite: 246, 327]
│   │   └── seed.js
│   ├── uploads/           # Directory for uploaded files (e.g., book cover images)
│   ├── .env.example        # Example environment variables for backend
│   ├── app.js             # Express app setup, middleware, and route imports [cite:
246, 252]
│   ├── server.js          # Node.js server entry point [cite: 246, 250]
│   ├── package.json        # Backend dependencies and scripts
│   └── README.md          # Backend setup instructions
└── frontend/              # React.js client application
    ├── public/            # Static assets (index.html, images, etc.)
    │   └── index.html
    ├── src/               # React source code
    │   ├── assets/         # Images, icons
    │   ├── components/      # Reusable React components (Navbar, BookCard, etc.)
[cite: 245, 299]
    │   ├── pages/          # Route-specific components (Home, Books, Wishlist, Cart,
Orders, Admin, Login, Register) [cite: 245, 286, 287]
    │   │   ├── Admin/
    │   │   ├── Books/
    │   │   ├── Cart/
    │   │   ├── Home/
    │   │   ├── Login/
    │   │   ├── Orders/
    │   │   ├── Register/
    │   │   └── Wishlist/
    │   ├── services/       # API service modules (authService.js, bookService.js) [cite:
```

Application Flow:

The BookNest application supports three main types of users: Customer (Ordinary User), Seller (if implemented as a distinct role), and Admin, each with specific roles and responsibilities defined by API endpoints.

1. Customer/Ordinary User Flow:

- **Create an Account & Login:** Customers register using email and password. After registration, they log in to access the app.
  - API:
  - POST /api/auth/register, POST /api/auth/login

- **View & Search Books:** After logging in, customers see a list of available books in their dashboard/home page. They can search and filter books by genre or other criteria.
  - API:
  - GET /api/books (with optional ?search= query)

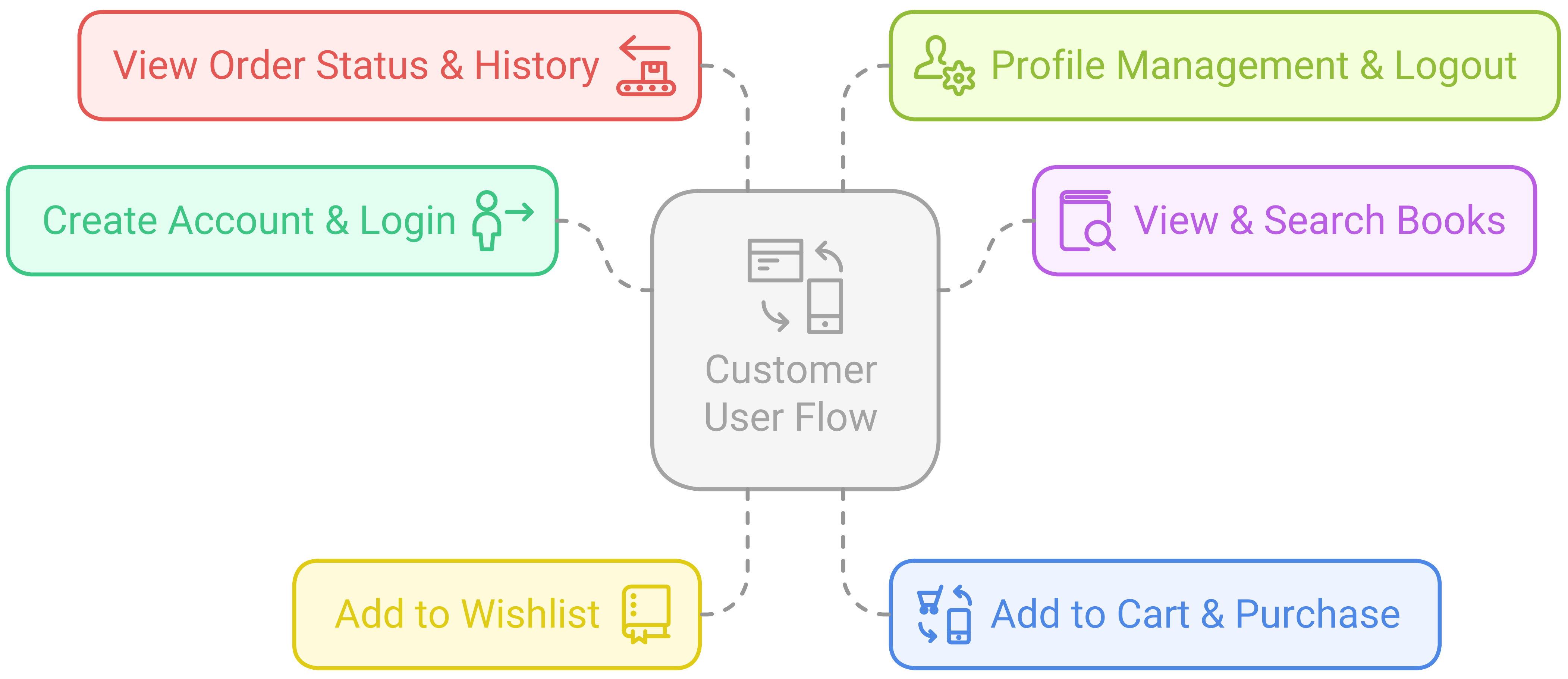
- **Add to Wishlist:** Authenticated users can add books to their wishlist for later.
  - API:
  - POST /api/wishlist/:bookId

- **Add to Cart & Purchase:** Customers select a book and add it to their shopping cart. When ready, they proceed to checkout to place an order.
  - API:
  - POST /api/orders (body includes cart items)

- **View Order Status & History:** Customers can track the status of their orders and view their past purchases.
  - API:
  - GET /api/orders

- **Profile Management & Logout:** Users can update their personal information and securely log out.

BookNest Customer User Flow



2. Seller Flow (if applicable, or via Admin-approved User):

- **Account Approval:** Sellers must receive approval from an admin before they can use the platform to list books.

- **Profile Management:** Sellers can manage their business details, email, and password.

- **Book Listing:** Once registered and approved, sellers can add new books, providing details like title, author, genre, description, price, and quantity.
  - API:
  - POST /api/books (Admin only, implies seller functionality is likely managed via admin or a dedicated seller API not fully detailed, but book CRUD is admin only)

- **Inventory Management:** Sellers can update stock levels and manage their active listings.
  - API:
  - PUT /api/books/:id

- **Order Fulfillment:** Sellers are responsible for processing and shipping orders placed by customers.
  - (Details on API for seller-specific order management are not explicitly provided in the source but would be necessary for this role).
- **Logout:** Sellers can log out from the application.

3. Admin Flow:

- **Monitor All Operations:** Admins oversee the entire platform, including user, seller, and order management.

- **Approve Seller Applications:** Admins review and approve new seller registrations, making them active on the platform.
    - (API for approving sellers is not explicitly detailed but falls under GET /api/admin/users or similar admin user management).

- **User Management:** Admins can manage user accounts, including creating, updating, or deleting them, and have authority over user ratings.
    - API:
    - GET /api/admin/users

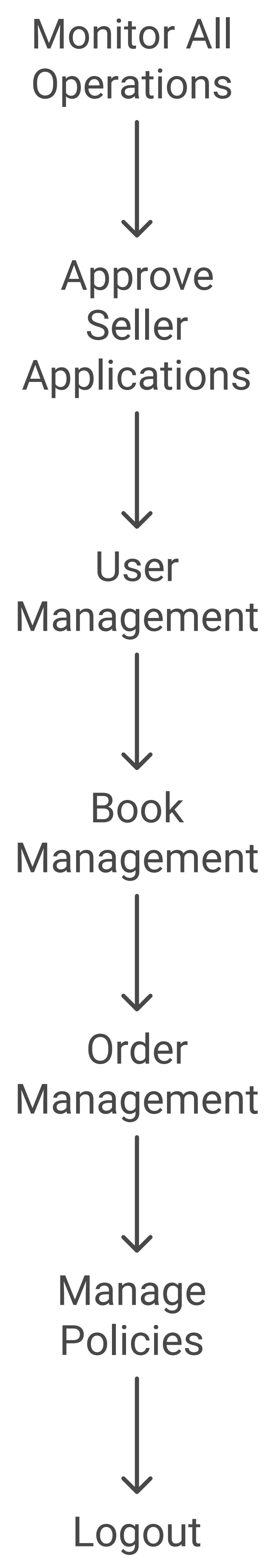
- **Book Management:** Admins can add new books, update existing book details, and remove inactive listings.
    - API:
    - POST /api/books, PUT /api/books/:id, DELETE /api/books/:id

- **Order Management:** Admins can view all orders placed by all users.
    - API:
    - GET /api/admin/orders

- **Manage Policies:** Admins enforce platform policies, terms of service, and privacy regulations.

- **Logout:** Admins can log out from the application.

Admin Flow in BookNest



Project Flow (Milestones):

The BookNest project development is structured into distinct milestones:
Milestone 1: Project Setup and Configuration:

- Install required tools: Node.js, MongoDB, and
- npm create vite@latest (or npx create-react-app).

- Create structured project folders for the client (frontend) and server (backend).

Milestone 2: Backend Development:

- Set up the Express.js server: Install
- express and create app.js (or server.js).

- Configure MongoDB: Install
- mongoose and establish the database connection.

- Create Mongoose Models: Define schemas for User, Book, and Order.

- Implement API Endpoints: Define RESTful routes under

- /api and implement CRUD (Create, Read, Update, Delete) operations for books, user authentication, wishlists, and orders.

- Test API Endpoints: Utilize tools like Postman to ensure all routes function as intended.

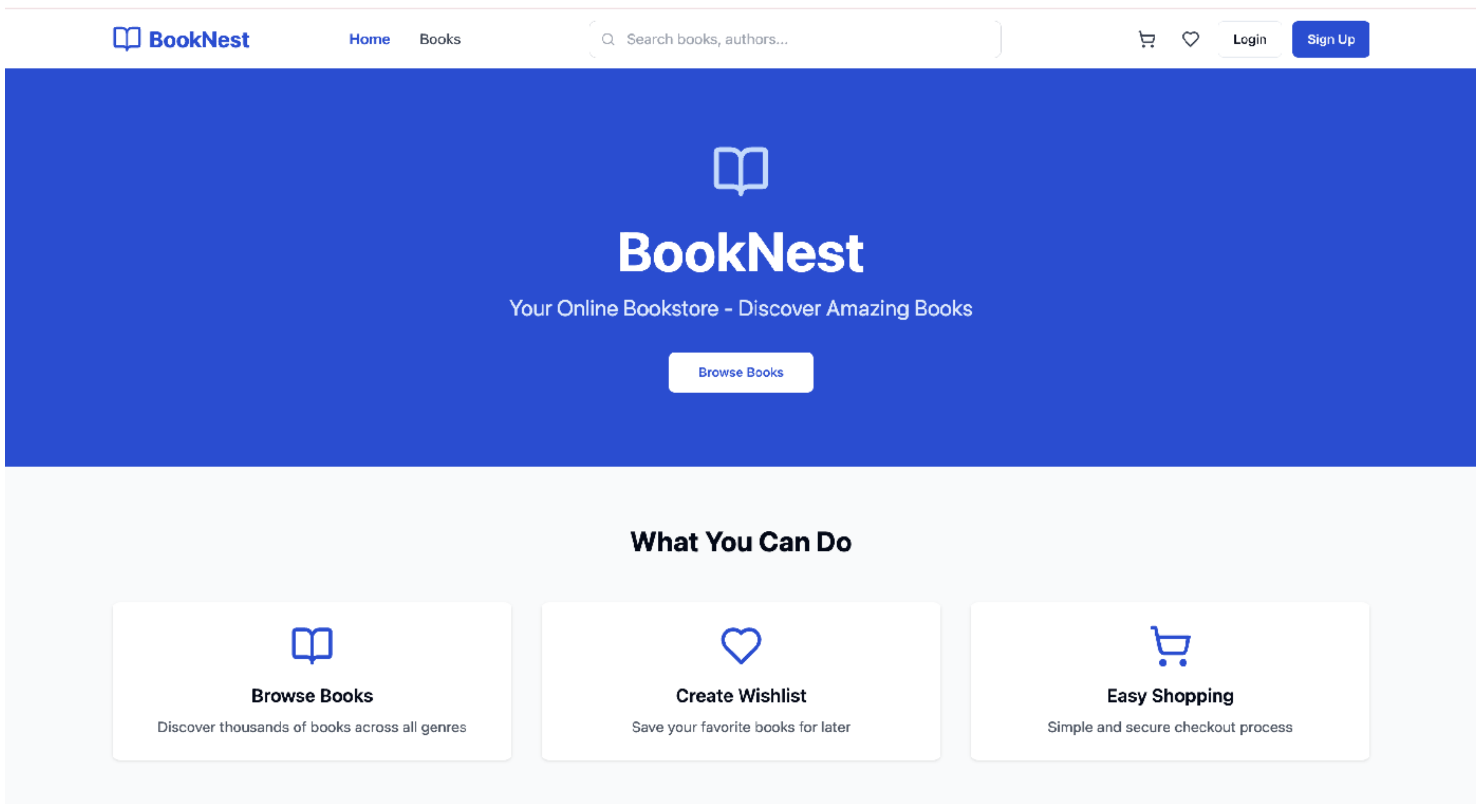
Milestone 3: Frontend Development:

- Set up React Application: Create the React app, configure React Router for navigation, and install required libraries (e.g.,
- react-router-dom, axios).

- Design UI Components: Create reusable React components (e.g., Navbar, BookCard, forms, dashboards) and implement styling using CSS.

- Implement Frontend Logic: Integrate with API endpoints using Axios/Fetch, implement data binding, and manage client-side state (including JWT in
- localStorage for authentication and cart/wishlist persistence).

Screenshots:



BookNest

Your Online Bookstore - Discover Amazing Books

What You Can Do

Browse Books

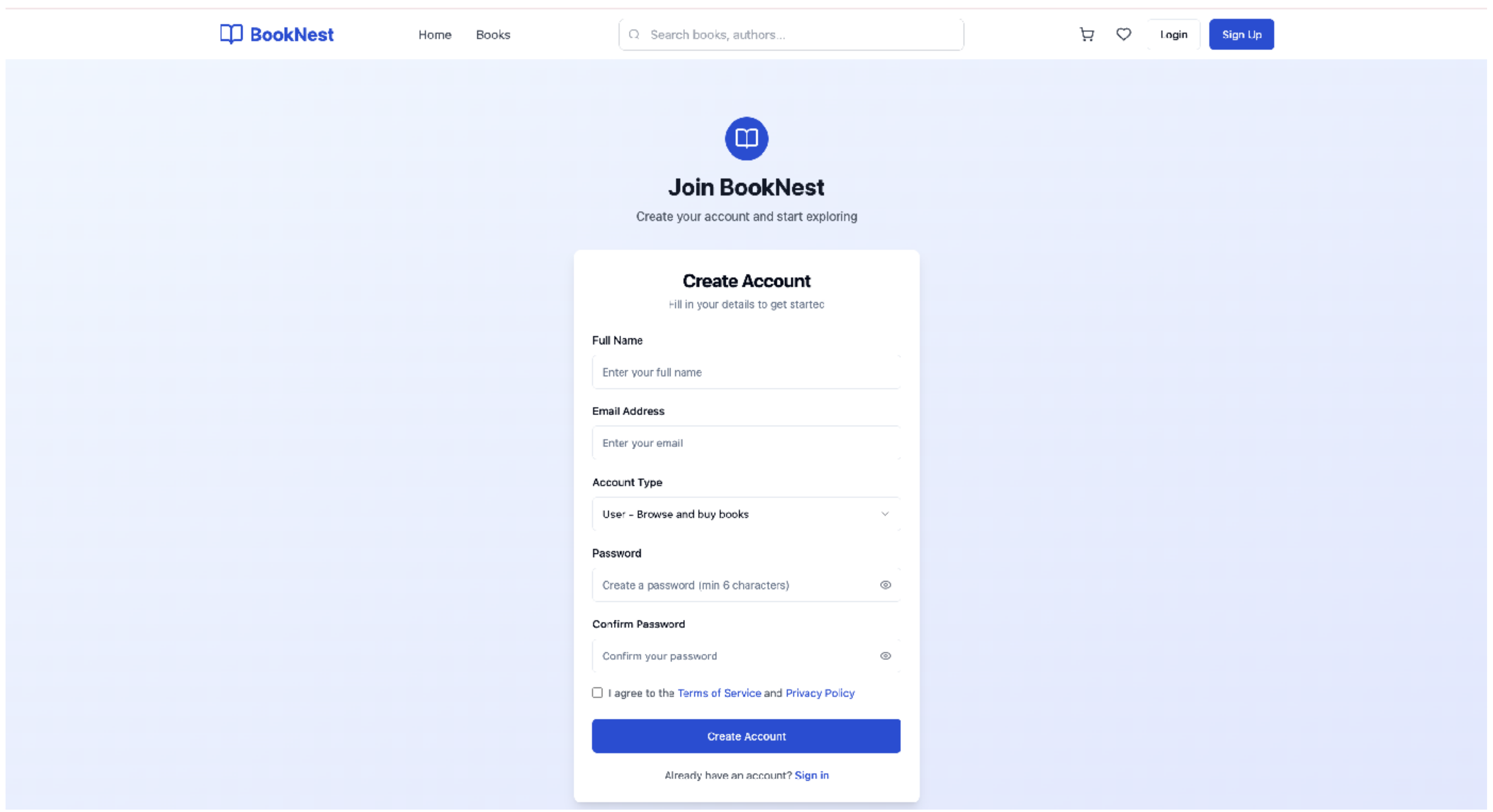
Discover thousands of books across all genres

Create Wishlist

Save your favorite books for later

Easy Shopping

Simple and secure checkout process



Join BookNest

Create your account and start exploring

Create Account

Fill in your details to get started

Full Name

Enter your full name

Email Address

Enter your email

Account Type

User - Browse and buy books

Password

Create a password (min 6 characters)

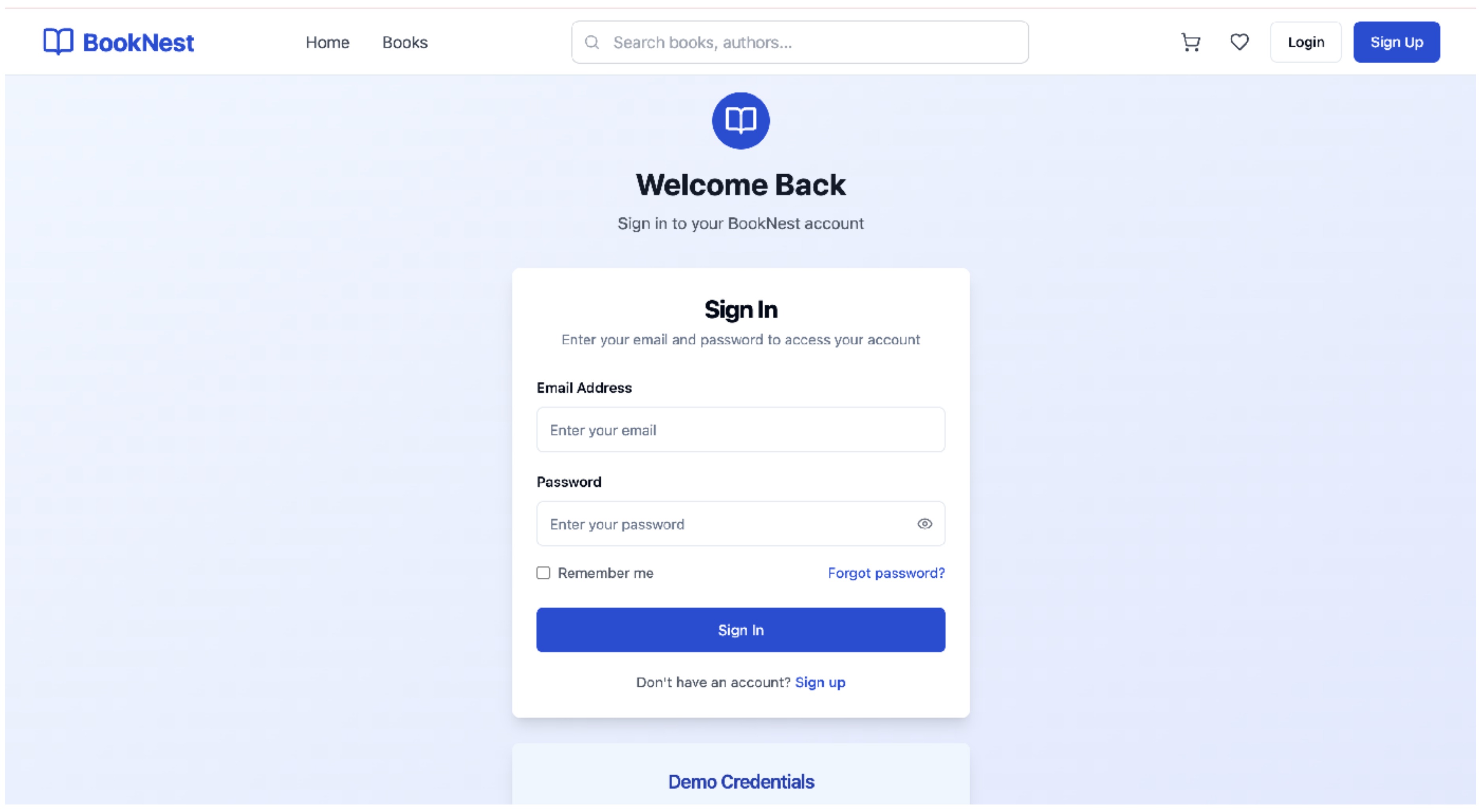
Confirm Password

Confirm your password

☐ I agree to the Terms of Service and Privacy Policy

Create Account

Already have an account? Sign in



BookNest Home Books Search books, authors... Login Sign Up

Welcome Back

Sign in to your BookNest account

Sign In

Enter your email and password to access your account

Email Address

Enter your email

Password

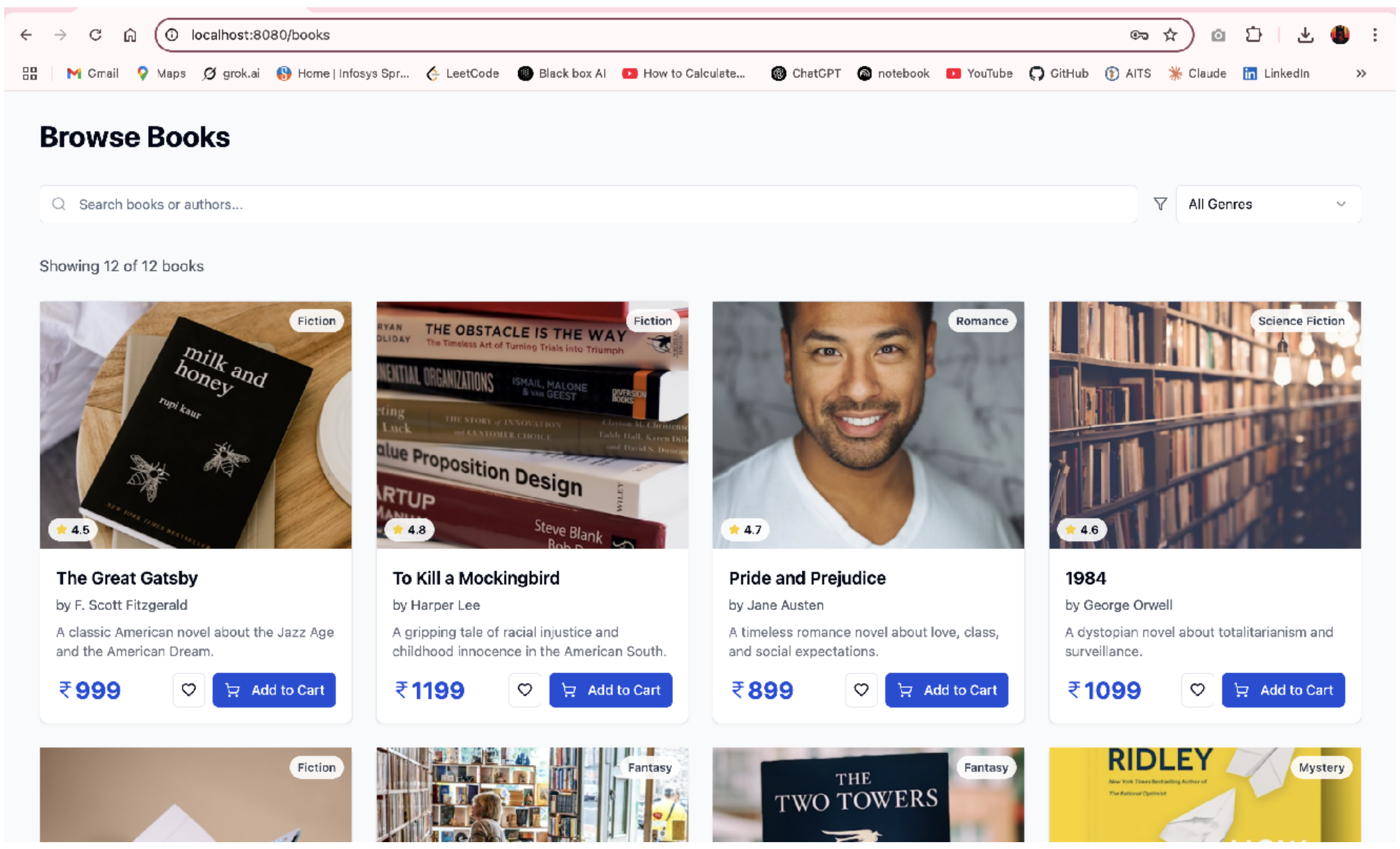
Enter your password

Remember me Forgot password?

Sign In

Don't have an account? Sign up

Demo Credentials



localhost:8080/books

Gmail Maps grok.ai Home | Infosys Spr... LeetCode Black box AI How to Calculate... ChatGPT notebook YouTube GitHub AITS Claude LinkedIn

Browse Books

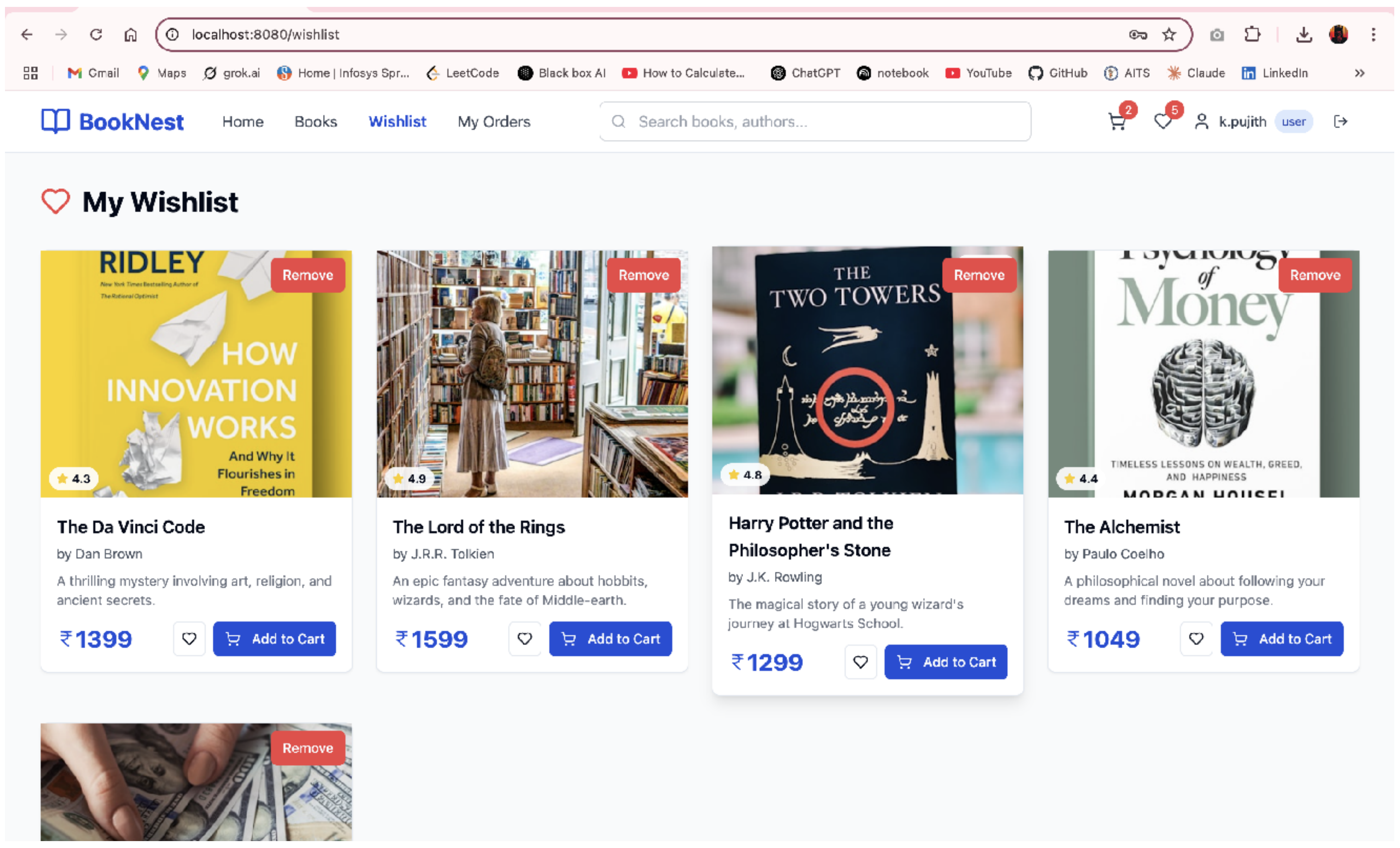
Search books or authors... All Genres

Showing 12 of 12 books

Genre	Rating	Title	Author	Description	Price
Fiction	4.5	The Great Gatsby	by F. Scott Fitzgerald	A classic American novel about the Jazz Age and the American Dream.	₹999
Fiction	4.8	To Kill a Mockingbird	by Harper Lee	A gripping tale of racial injustice and childhood innocence in the American South.	₹1199
Romance	4.7	Pride and Prejudice	by Jane Austen	A timeless romance novel about love, class, and social expectations.	₹899
Science Fiction	4.6	1984	by George Orwell	A dystopian novel about totalitarianism and surveillance.	₹1099

Add to Cart

Fiction Fantasy Fantasy Mystery



localhost:8080/wishlist

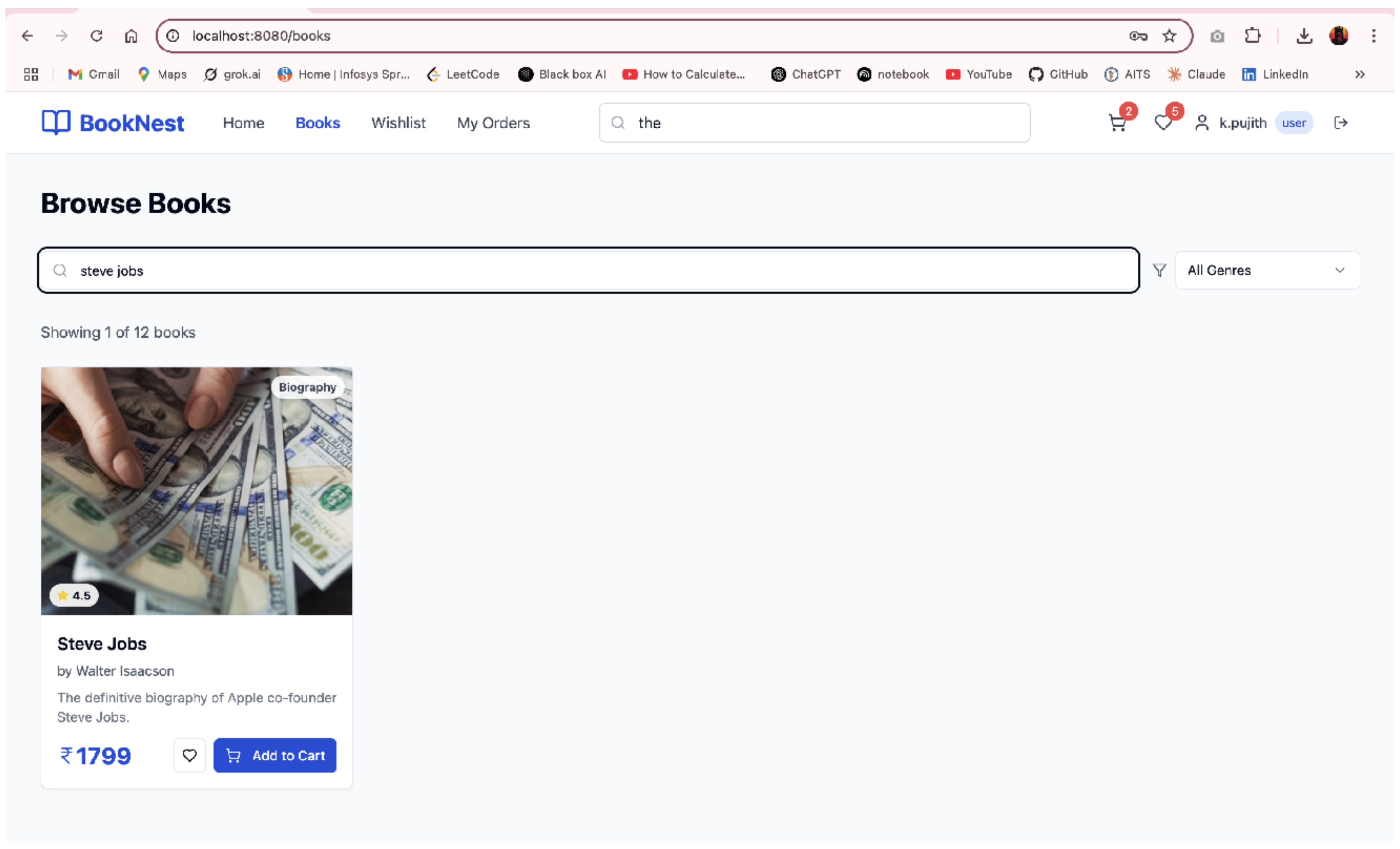
Gmail Maps grok.ai Home | Infosys Spr... LeetCode Black box AI How to Calculate... ChatGPT notebook YouTube GitHub AITS Claude LinkedIn

BookNest Home Books Wishlist My Orders Search books, authors... 2 5 k.pujith user

My Wishlist

Rating	Title	Author	Description	Price
4.3	The Da Vinci Code	by Dan Brown	A thrilling mystery involving art, religion, and ancient secrets.	₹1399
4.9	The Lord of the Rings	by J.R.R. Tolkien	An epic fantasy adventure about hobbits, wizards, and the fate of Middle-earth.	₹1599
4.8	Harry Potter and the Philosopher's Stone	by J.K. Rowling	The magical story of a young wizard's journey at Hogwarts School.	₹1299
4.4	The Alchemist	by Paulo Coelho	A philosophical novel about following your dreams and finding your purpose.	₹1049

Remove Add to Cart


localhost:8080/books

Gmail · Maps · grok.ai · Home | Infosys Spr... · LeetCode · Black box AI · How to Calculate... · ChatGPT · notebook · YouTube · GitHub · AITS · Claude · LinkedIn

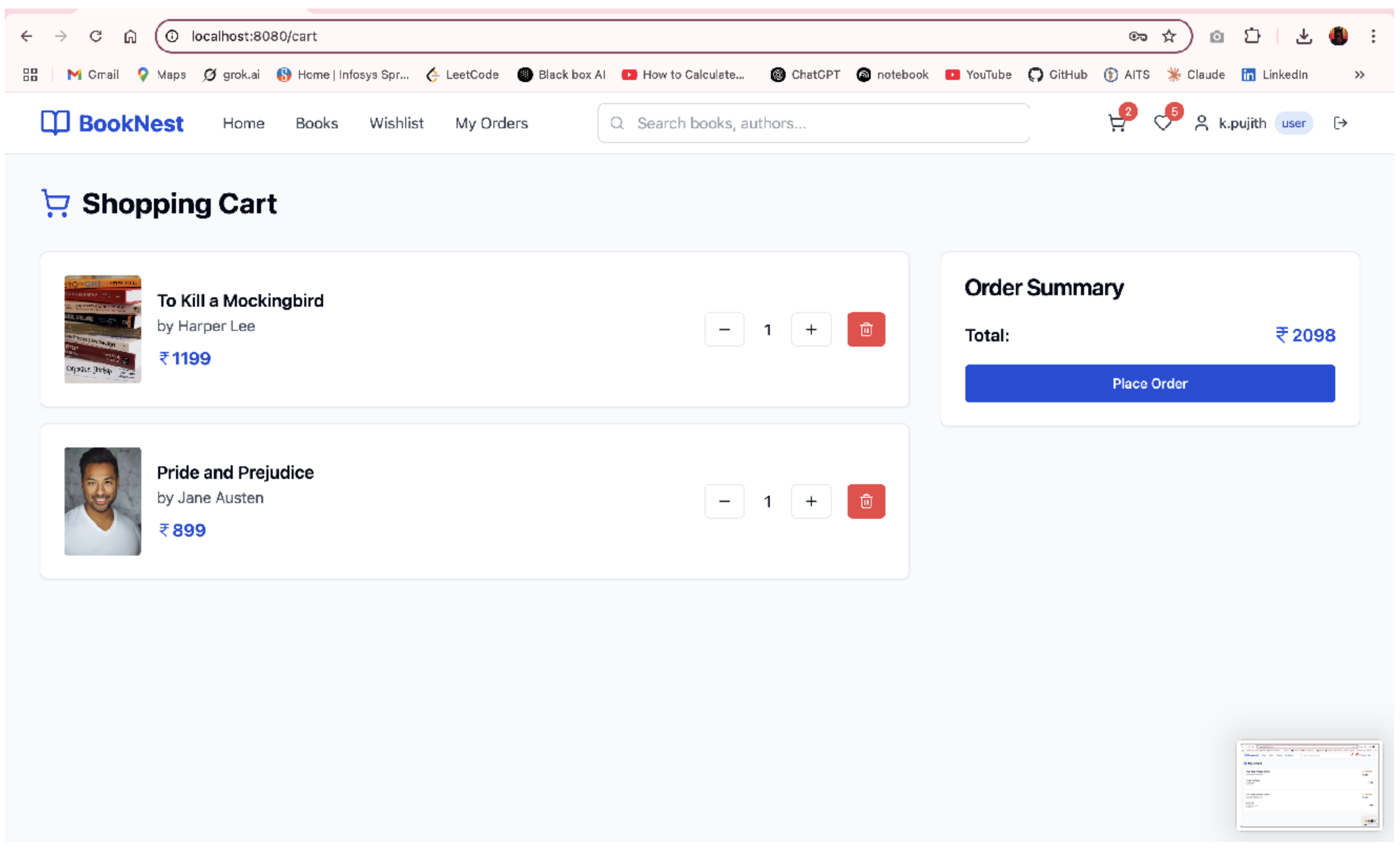
BookNest Home Books Wishlist My Orders the 2 5 k.pujith user

Browse Books

steve jobs · All Genres

Showing 1 of 12 books

Biography
4.5
Steve Jobs
by Walter Isaacson
The definitive biography of Apple co-founder Steve Jobs.
₹1799 Add to Cart


localhost:8080/cart

Gmail · Maps · grok.ai · Home | Infosys Spr... · LeetCode · Black box AI · How to Calculate... · ChatGPT · notebook · YouTube · GitHub · AITS · Claude · LinkedIn

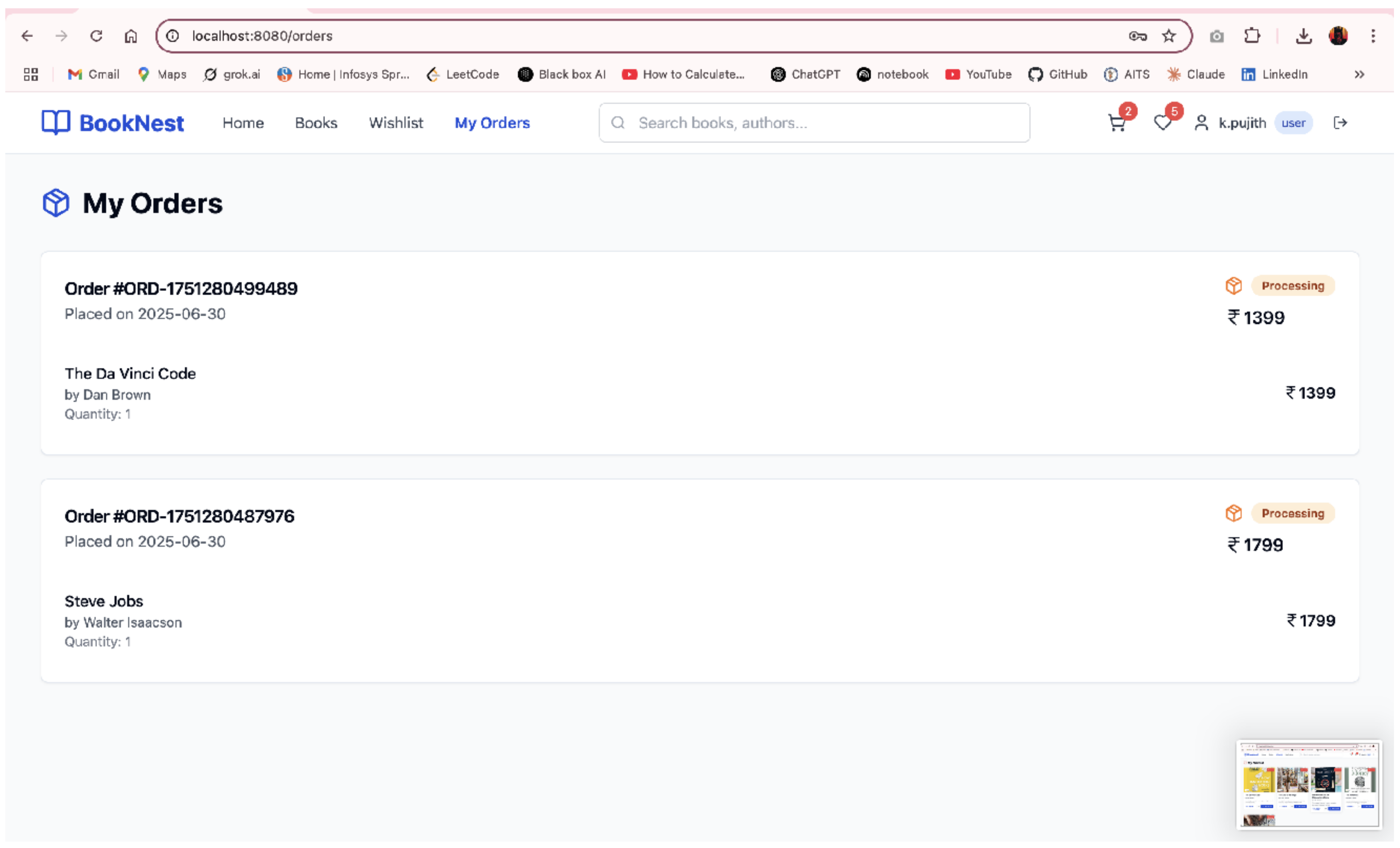
BookNest Home Books Wishlist My Orders Search books, authors... 2 5 k.pujith user

Shopping Cart

To Kill a Mockingbird
by Harper Lee
₹1199 − 1 +

Pride and Prejudice
by Jane Austen
₹899 − 1 +

Order Summary
Total: ₹2098
Place Order


localhost:8080/orders

Gmail · Maps · grok.ai · Home | Infosys Spr... · LeetCode · Black box AI · How to Calculate... · ChatGPT · notebook · YouTube · GitHub · AITS · Claude · LinkedIn

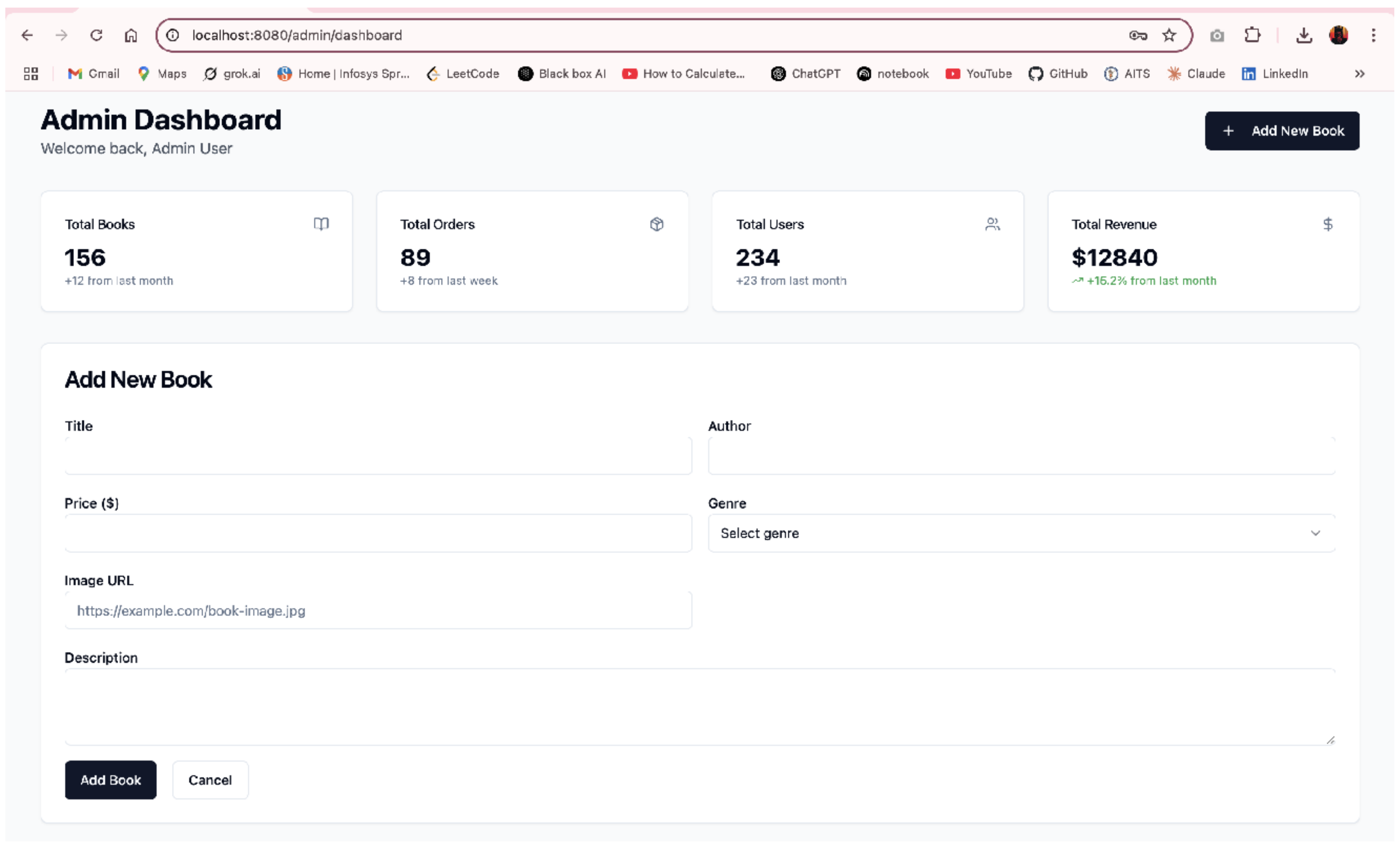
BookNest Home Books Wishlist My Orders Search books, authors... 2 5 k.pujith user

My Orders

Order #ORD-1751280499489
Placed on 2025-06-30
Processing
₹1399
The Da Vinci Code
by Dan Brown
Quantity: 1
₹1399

Order #ORD-1751280487976
Placed on 2025-06-30
Processing
₹1799
Steve Jobs
by Walter Isaacson
Quantity: 1
₹1799

Made with Napkin



Admin Dashboard

Welcome back, Admin User

+ Add New Book

Total Books	Total Orders	Total Users	Total Revenue
156	89	234	$12840
+12 from last month	+8 from last week	+23 from last month	+15.2% from last month

Add New Book

Title:

Author:

Price ($):

Genre: Select genre

Image URL: https://example.com/book-image.jpg

Description:

Add Book | Cancel

Made with Napkin