**Full Stack Development with MERN**

**Project Documentation format**

**1. Introduction**

* **Project Title: BOOKNEST**
* **Team Members:**
  + Sanjay
  + Yavish
  + Preetham
  + Nageswara Rao

**2. Project Overview**

* **Purpose:** To create a comprehensive online bookstore allowing users to browse, purchase, and manage books, while providing dedicated portals for sellers and administrators.
* **Features:**
  + User registration and login
  + Book browsing
  + wishlist management
  + Order management
  + Seller and admin portals for inventory and order management
  + Admin functionalities for user and seller management

**3. Architecture**

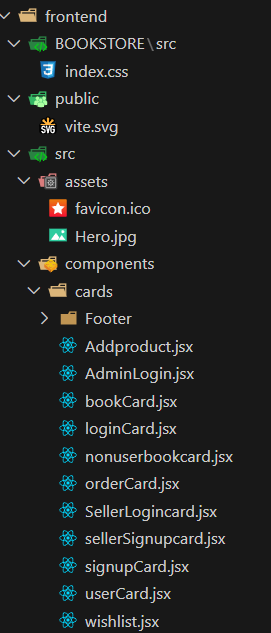
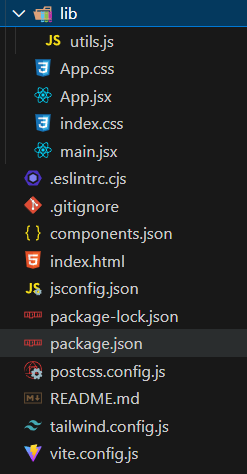
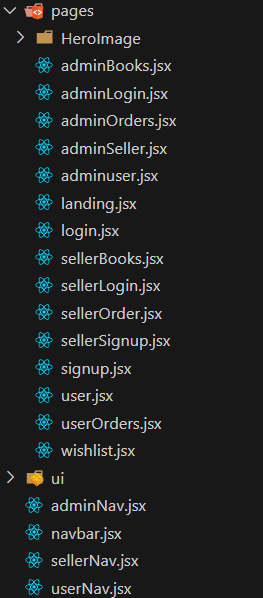
* Frontend: Built with React.js and styled using Tailwind CSS.
* Backend: Developed using Node.js and Express.js.
* Database: MongoDB for storing user data, book details, and order information.

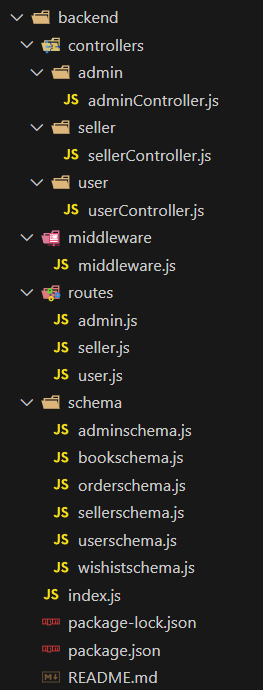
**4. Setup Instructions**

* **Prerequisites:**
  + Node.js
  + MongoDB
* **Installation:**
  + **Clone the repository.**
  + **Install dependencies for the backend:**
    - cd backend
    - npm i
    - node index.js
  + **Install dependencies for the frontend:**
    - cd frontend
    - npm i
    - npm run dev

**5. Folder Structure**

**Client (frontend):**

**** ****

* **Server (backend):**
* ****

**6. Running the Application**

* **Frontend:**

cd frontend

npm i

npm run dev

* **Backend:**

cd backend

npm i

node index.js

**7. API Documentation**

* **Endpoints:**

 **User Registration**

* **POST** /api/users/register
* Request: { "email": "example@example.com", "password": "password123" }
* Response: 200 OK

 **User Login**

* **POST** /api/users/login
* Request: { "email": "example@example.com", "password": "password123" }
* Response: 200 OK

 **Seller Registration**

* **POST** /api/sellers/register
* Request: { "shopName": "Example Shop", "email": "example@example.com", "password": "password123" }
* Response: 200 OK

 **Admin Login**

* **POST** /api/admin/login
* Request: { "email": "admin@example.com", "password": "admin123" }
* Response: 200 OK

 **Get Books**

* **GET** /api/books
* Response: 200 OK { "books": [...] }

 **Add Book (Seller/Admin)**

* **POST** /api/books
* Request: { "title": "Book Title", "author": "Author Name", "genre": "Fiction", "price": 19.99 }
* Response: 201 Created

 **Order Book**

* **POST** /api/orders
* Request: { "userId": "123", "bookId": "456", "address": "User Address" }
* Response: 200 OK { "message": "Order placed" }

 **Get Orders (User)**

* **GET** /api/orders/user/:userId
* Response: 200 OK { "orders": [...] }

 **Get Orders (Seller)**

* **GET** /api/orders/seller/:sellerId
* Response: 200 OK { "orders": [...] }

 **Get Orders (Admin)**

* **GET** /api/orders
* Response: 200 OK { "orders": [...] }

 **Delete Book (Seller/Admin)**

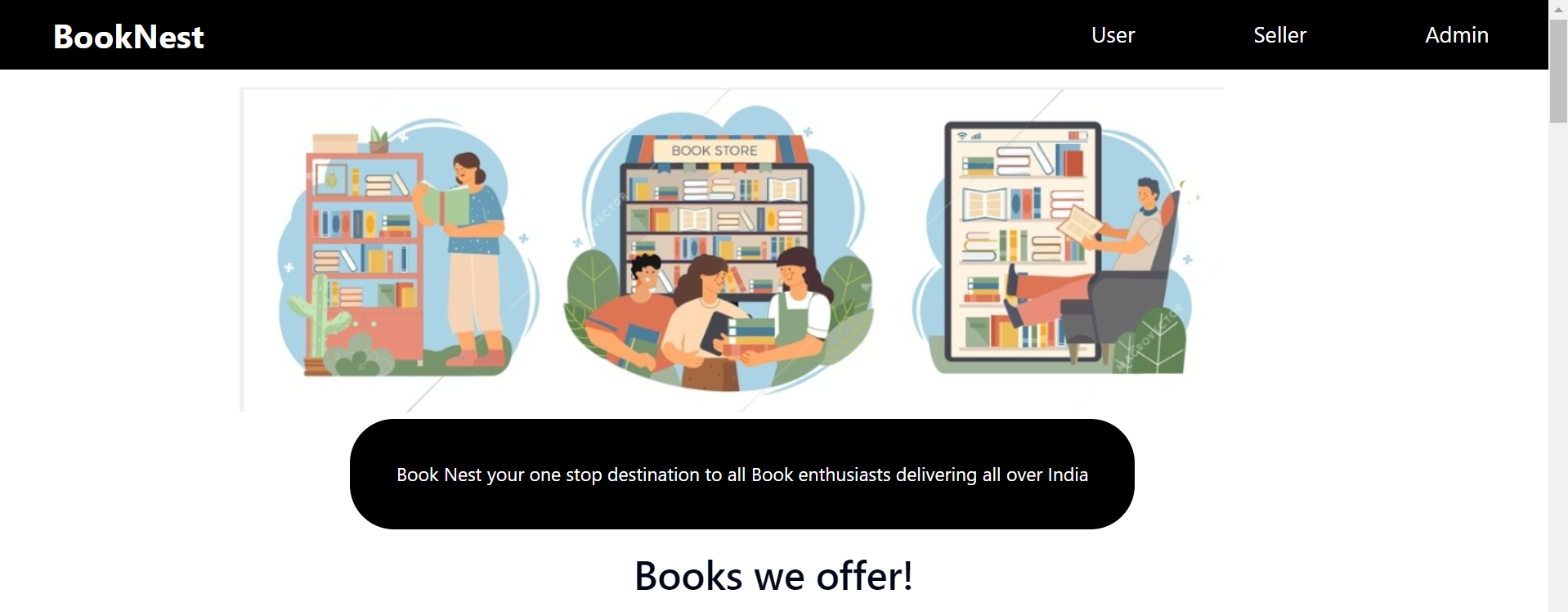
* **DELETE** /api/books/:bookId
* Response: 200 OK { "message": "Book deleted" }

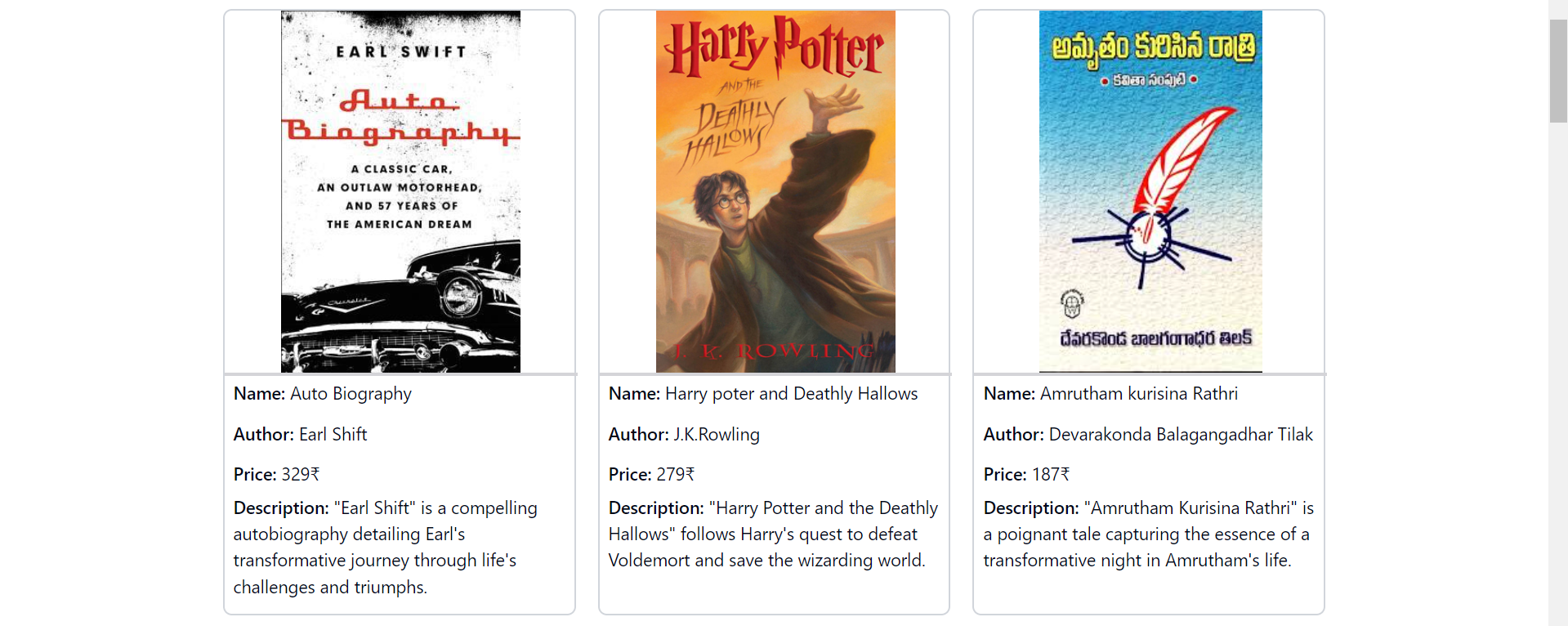
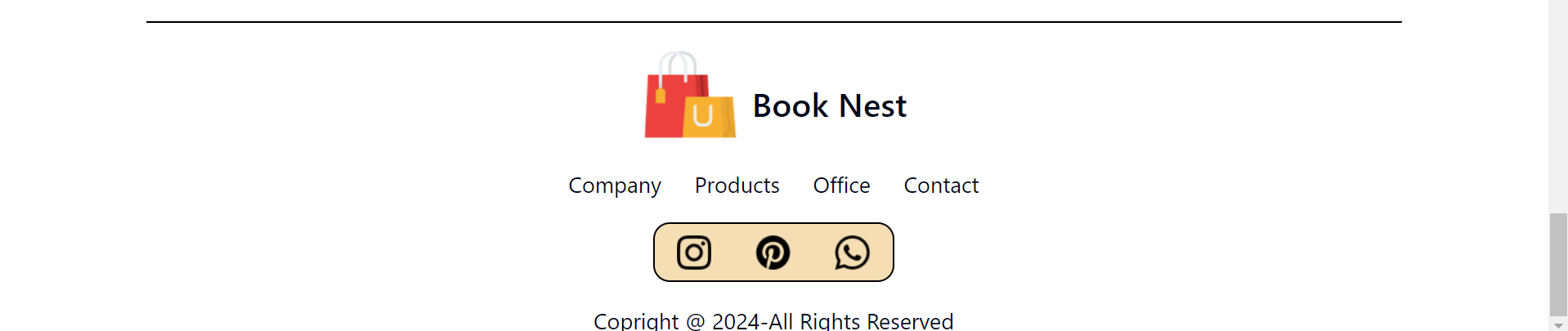
**8. Authentication**

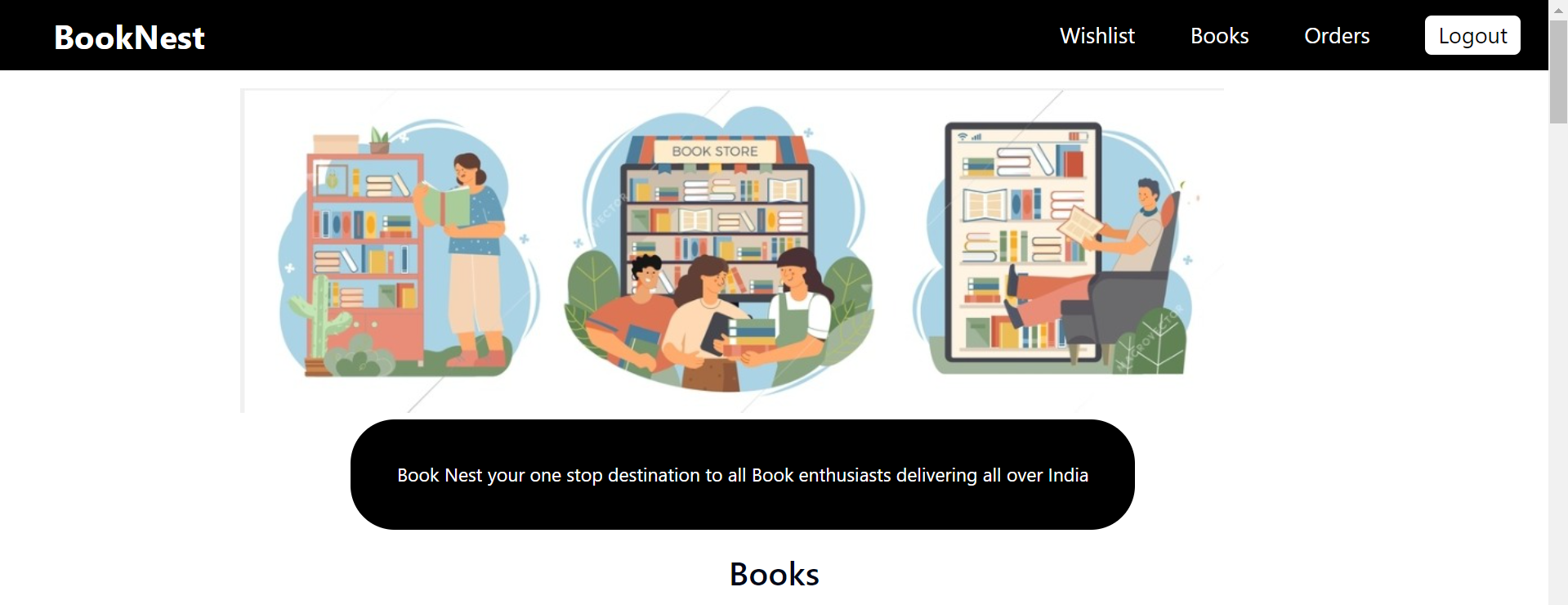
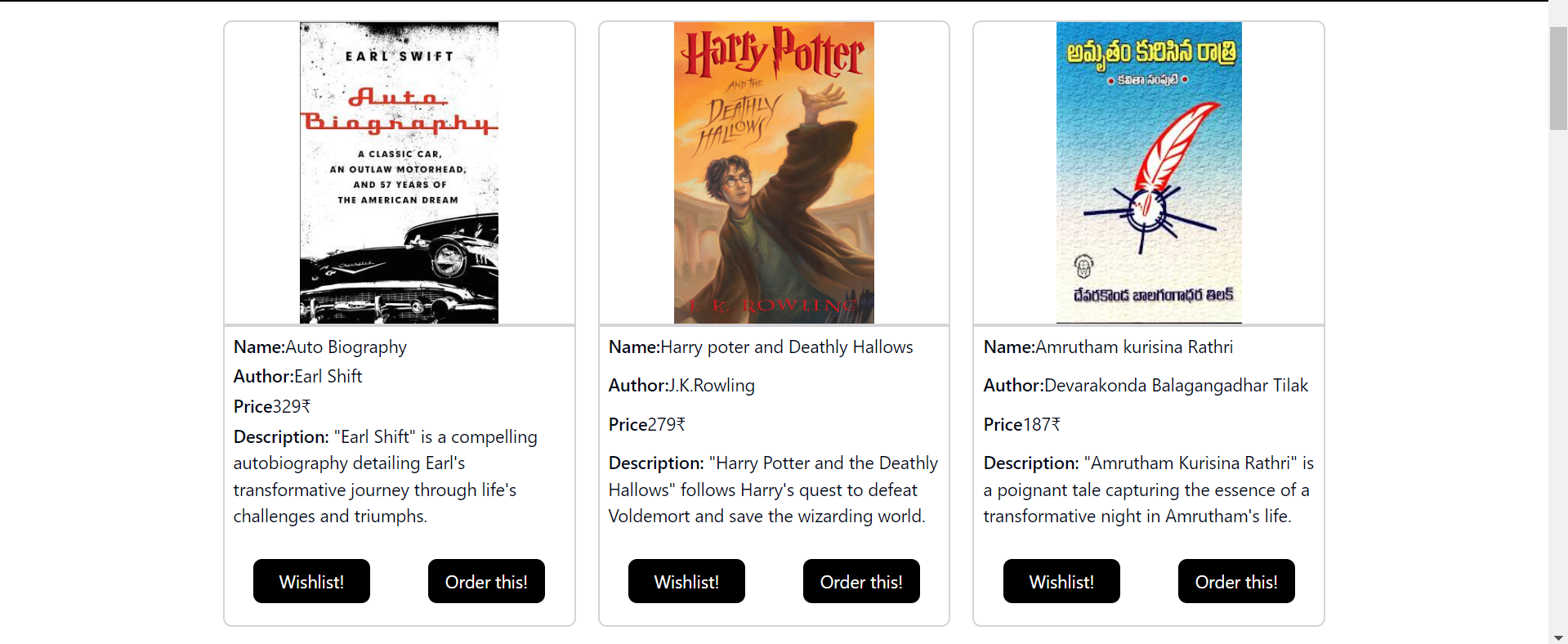
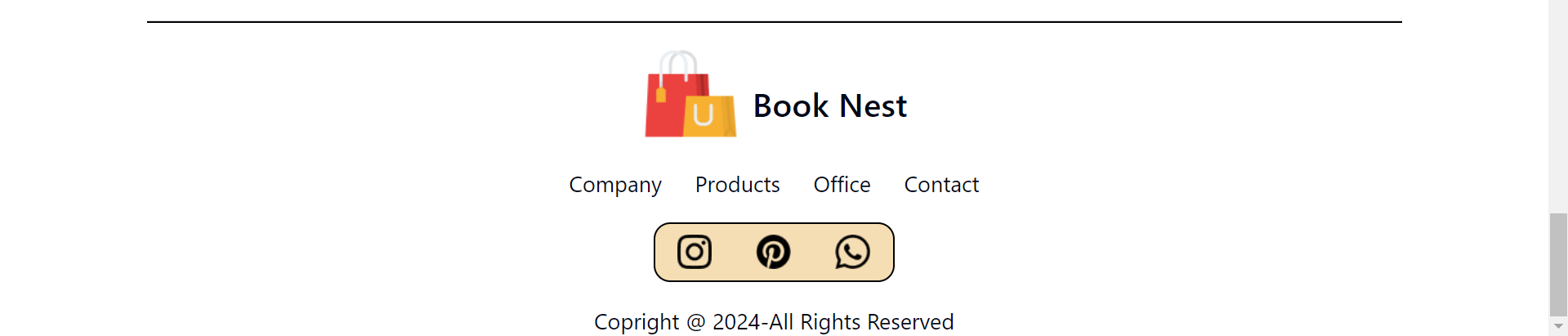
* Method: JSON Web Tokens (JWT) for securing user sessions.
* Process:
  1. User logs in or registers and receives a token.
  2. Token is included in the headers of subsequent requests for authentication.

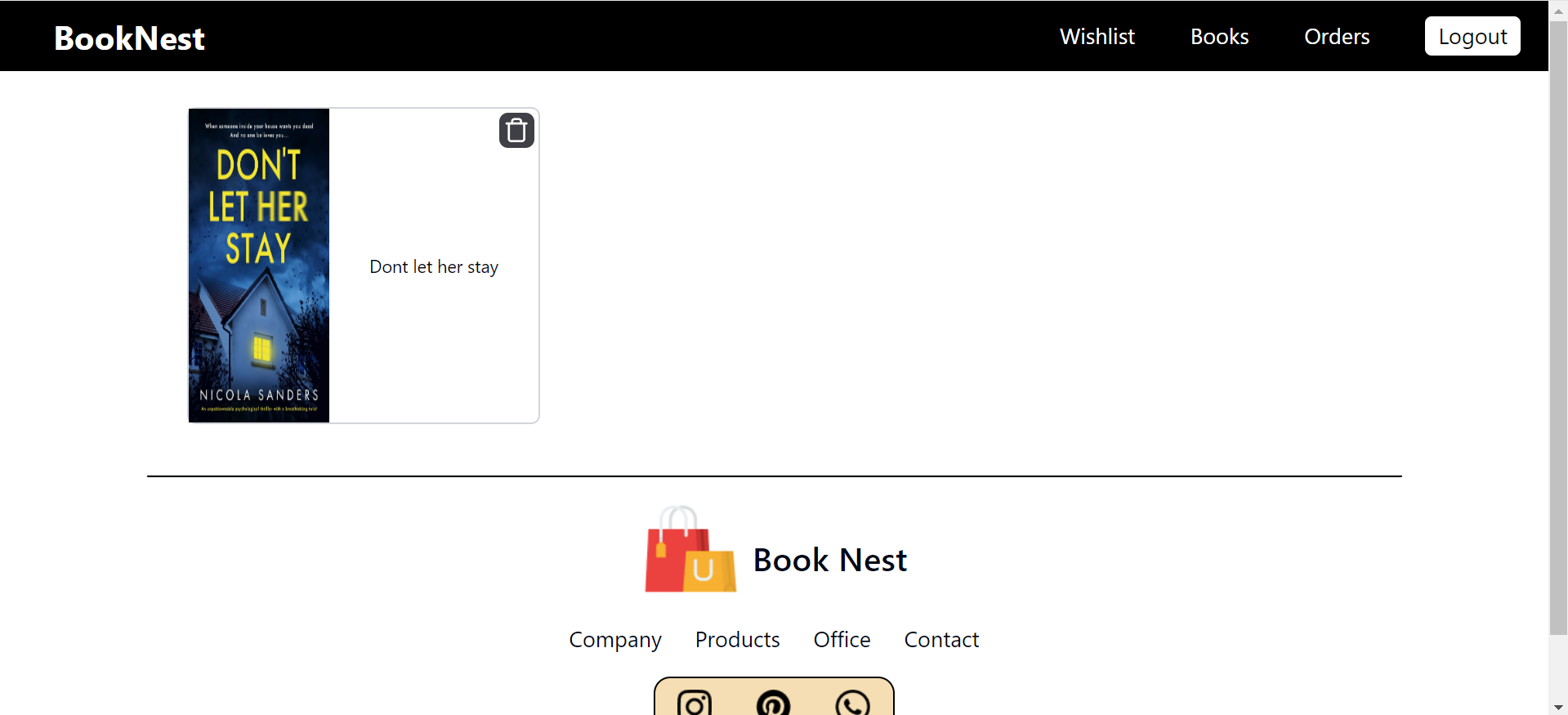
**9. User Interface**

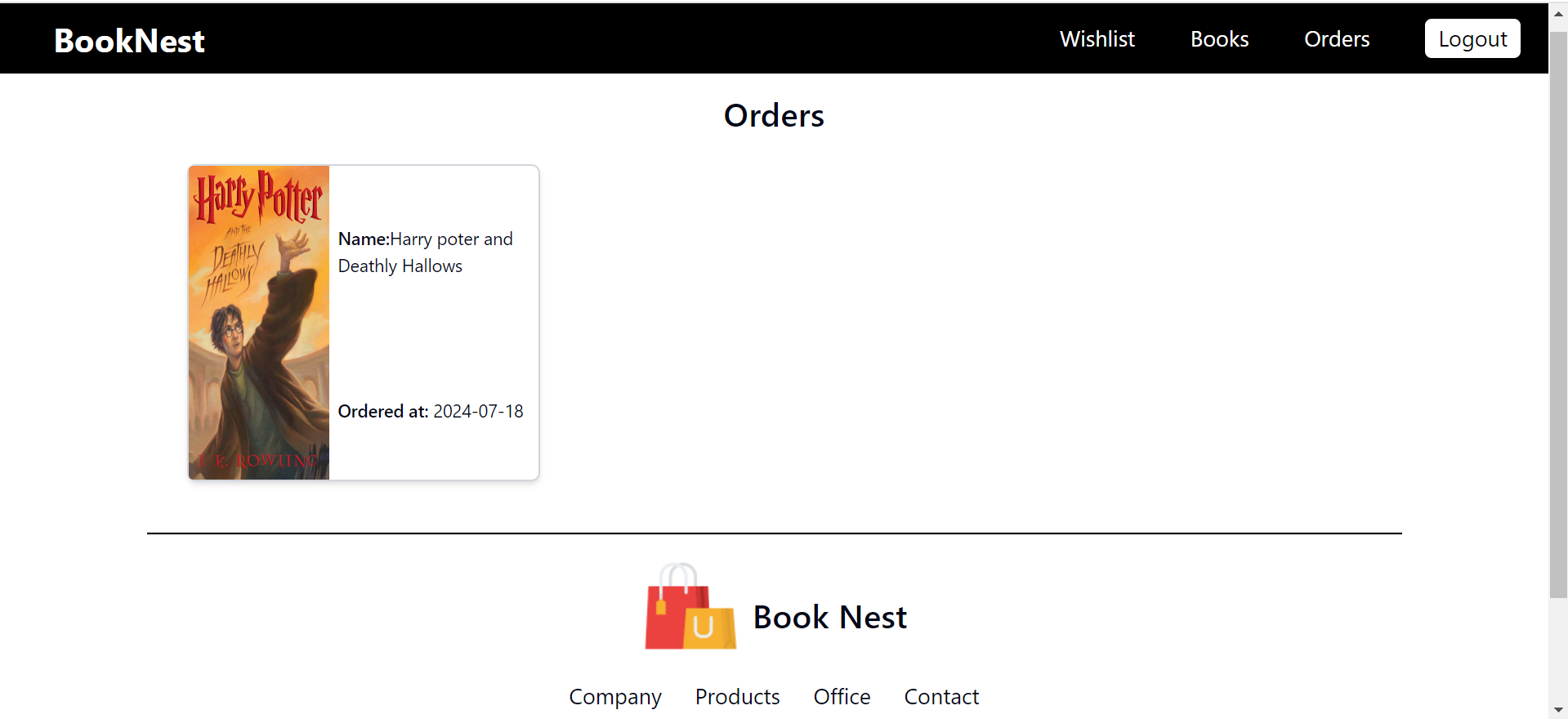
* **Landing Page:**

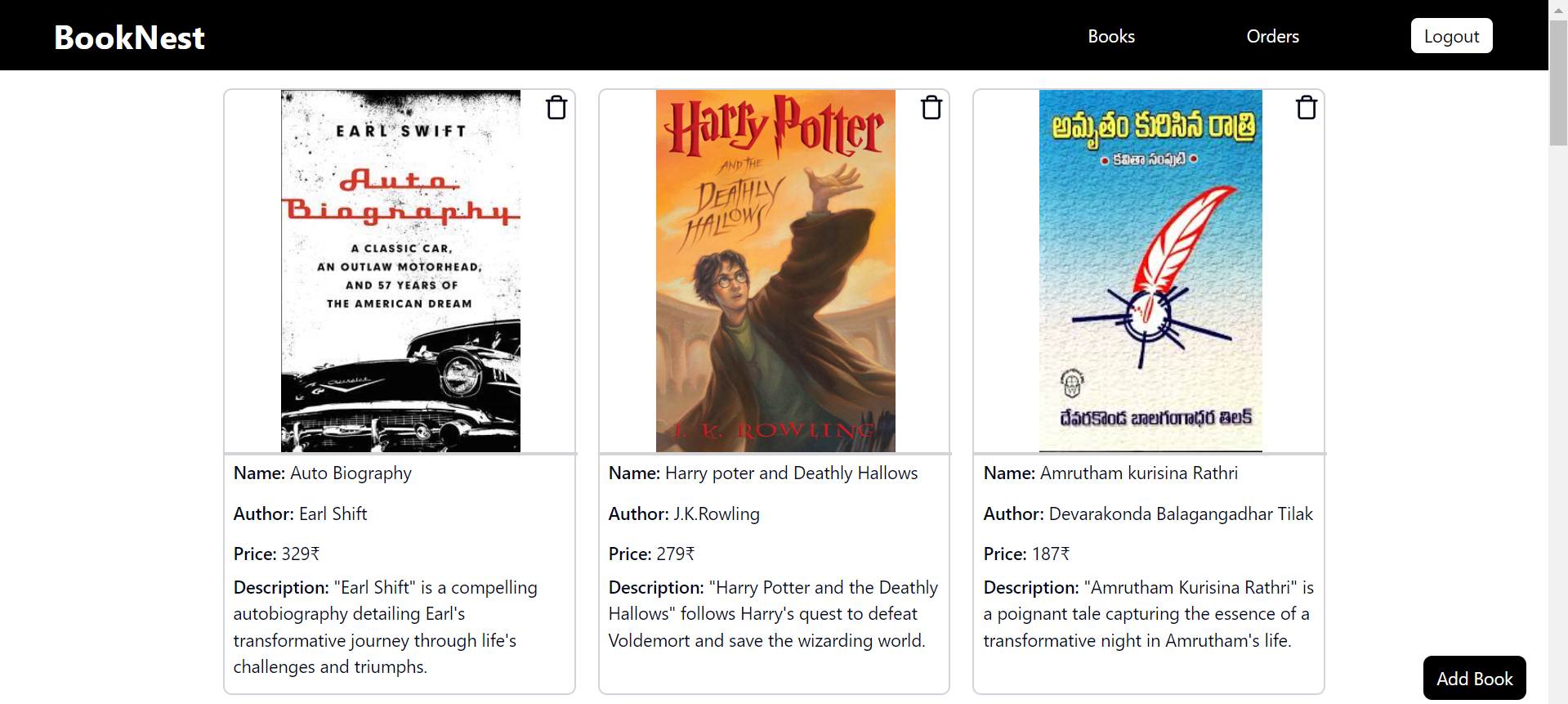
****

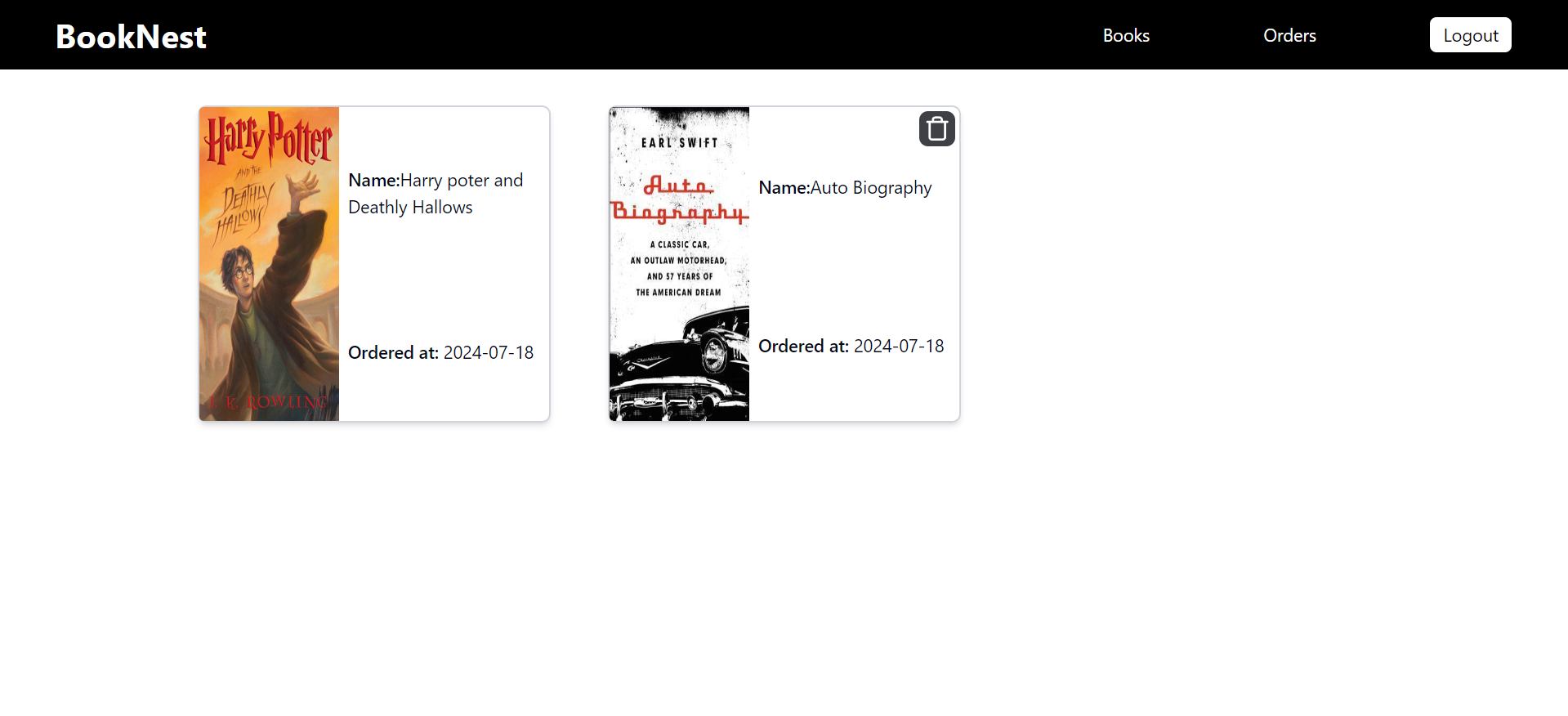
**** ****

* **User Page:** ****  

**Wishlist page:** ****

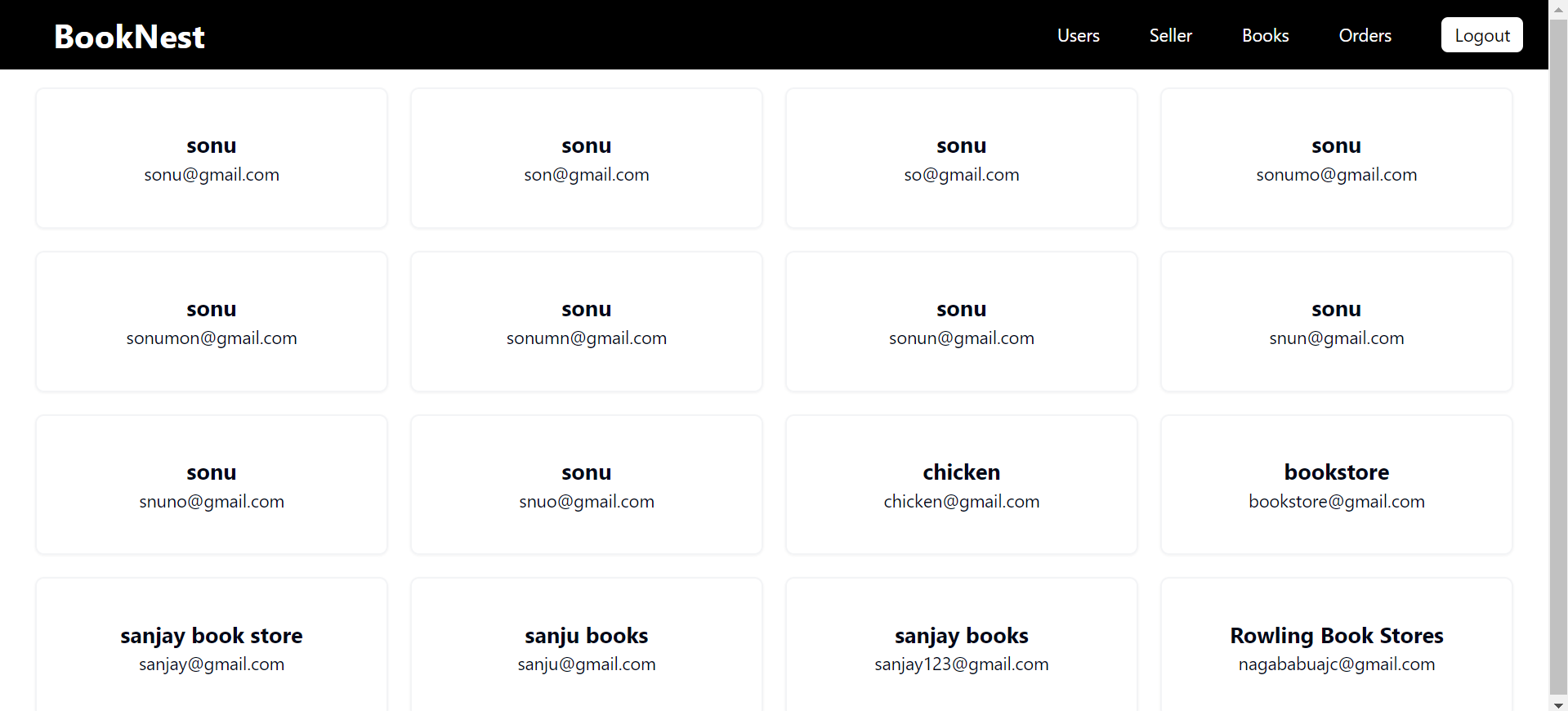
**User Order page:** 

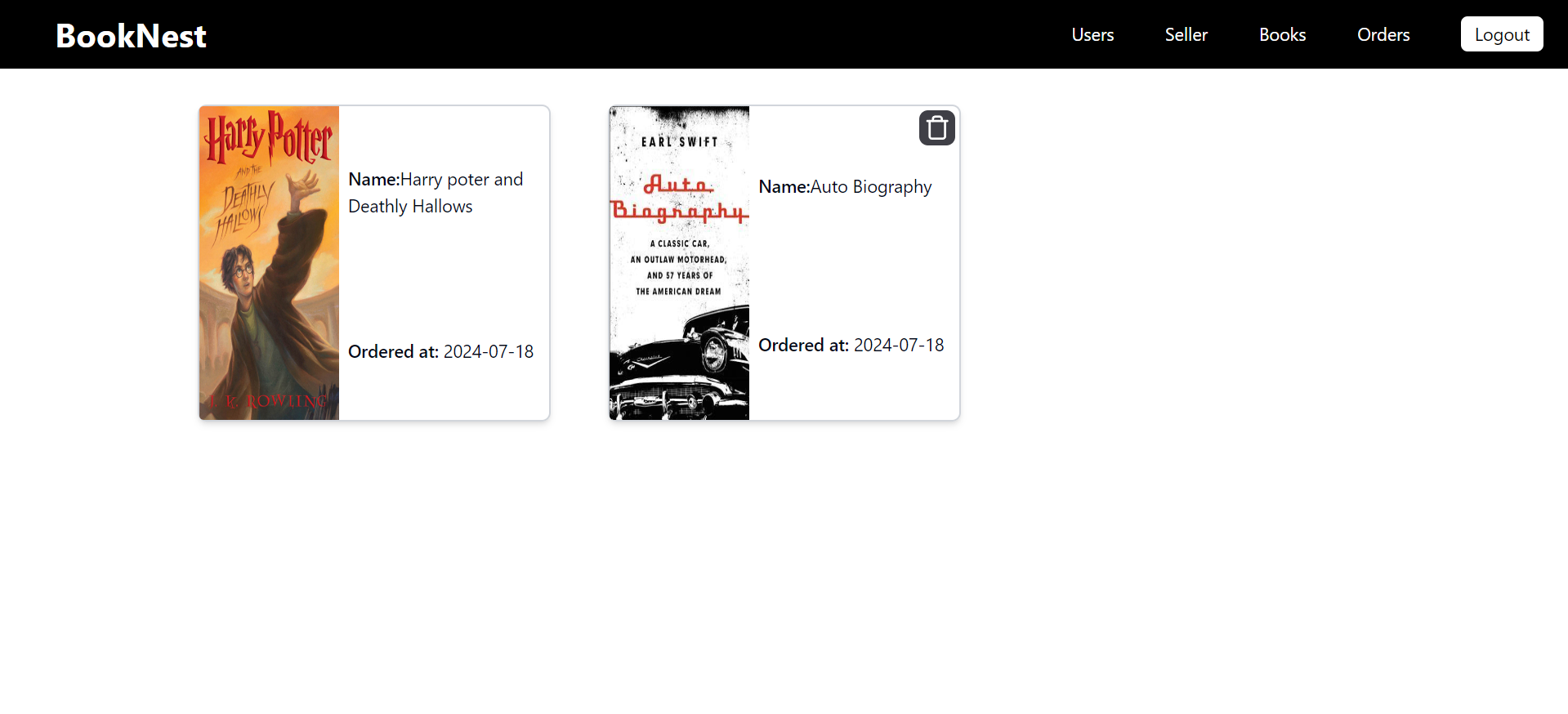
**Seller page:** 

**Seller order page:** ****

**Admin page:** ****

**Admin user page:** ****

**Admin Seller page:** ****

**Admin Orders page:** ****

**10. Testing**

* Strategy: Manual testing of user flows and functionalities.
* Tools: Postman for API testing.

**11. Screenshots or Demo**

* **Link to Demo: [**[click here to watch demonstration video](https://youtu.be/CUhEV_aI-hA)**]**

**12. Known Issues**

* Placing an order shows an alert "order placed" as a demo instead of performing a real transaction.

**13. Future Enhancements**

* Implement real payment gateway integration.
* Improve UI/UX with more responsive design elements.
* Add more detailed book descriptions and reviews.