**Full Stack Development with MERN**

**Book Store**

**1. Introduction**

* **Project Title**: BookStore
* **Team Members**:

1. Akshat Raj
2. Falguni Jain
3. No 3
4. No 4

**2. Project Overview**

* **Purpose**:  
  To simplify book exchange by providing a platform that promotes sustainability, reduces book waste, and fosters a community of readers through easy sharing and borrowing.
* **Features**:
  + User Authentication (Sign Up/Login)
  + Add & List Books
  + Browse/Search Books
  + Request and Accept Book Exchanges
  + Responsive Design
  + Book Management Dashboard

**3. Architecture**

* **Frontend**:  
  Built using **React.js** and styled with **Tailwind CSS** and **Radix UI** for reusable components. Designed as a responsive SPA.
* **Backend**:  
  Developed using **Node.js** and **Express.js**, structured as RESTful APIs, with secure routing and data validation using **Zod**.
* **Database**:  
  **MongoDB** is used to store user data, book details, and transactions. Interactions are managed via **Mongoose** or **Drizzle ORM** (if ported from SQL).

**4. Setup Instructions**

* **Prerequisites**:
  + Node.js (v18+)
  + MongoDB (Atlas or local instance)
  + npm or yarn
  + Git
* **Installation**:

cd BookStore

cd client && npm install

cd ../server && npm install

* **Environment Setup**:  
  Create a .env file in the server folder with:

MONGO\_URI=your\_mongodb\_connection\_string

JWT\_SECRET=your\_jwt\_secret

**5. Folder Structure**

* **Client/**
  + src/components/ – Reusable UI components
  + src/pages/ – Page components like Home, BookList, Profile
  + src/utils/ – Helpers and API functions
* **Server/**
  + controllers/ – Request handlers
  + routes/ – API route definitions
  + models/ – Mongoose schemas
  + middleware/ – Authentication, error handling

**6. Running the Application**

* **Frontend**:

bash

CopyEdit

cd client

npm start

* **Backend**:

bash

CopyEdit

cd server

npm start

**7. API Documentation**

|  |  |  |
| --- | --- | --- |
| **Method** | **Endpoint** | **Description** |
| POST | /api/auth/register | Register new user |
| POST | /api/auth/login | Authenticate user |
| GET | /api/books | Get all listed books |
| POST | /api/books | Add a new book |
| PUT | /api/books/:id | Update book info |
| DELETE | /api/books/:id | Delete book |
| POST | /api/exchange/request | Send book exchange request |

Example Response:

{

"success": true,

"data": { "title": "Atomic Habits", "owner": "JohnDoe" }

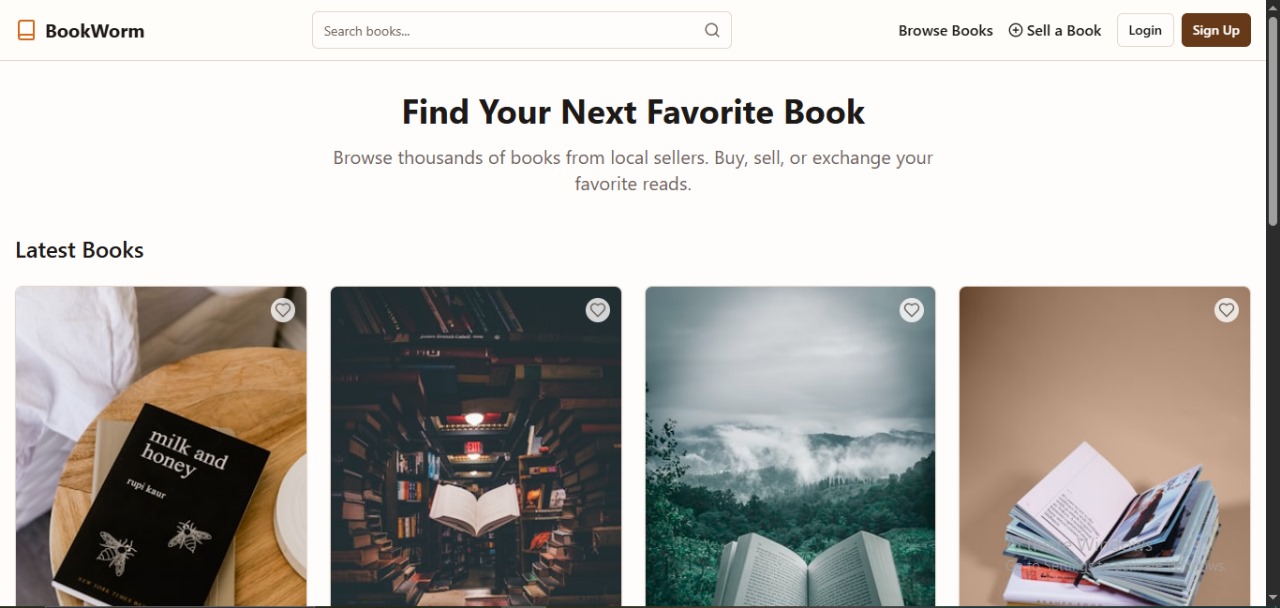
}

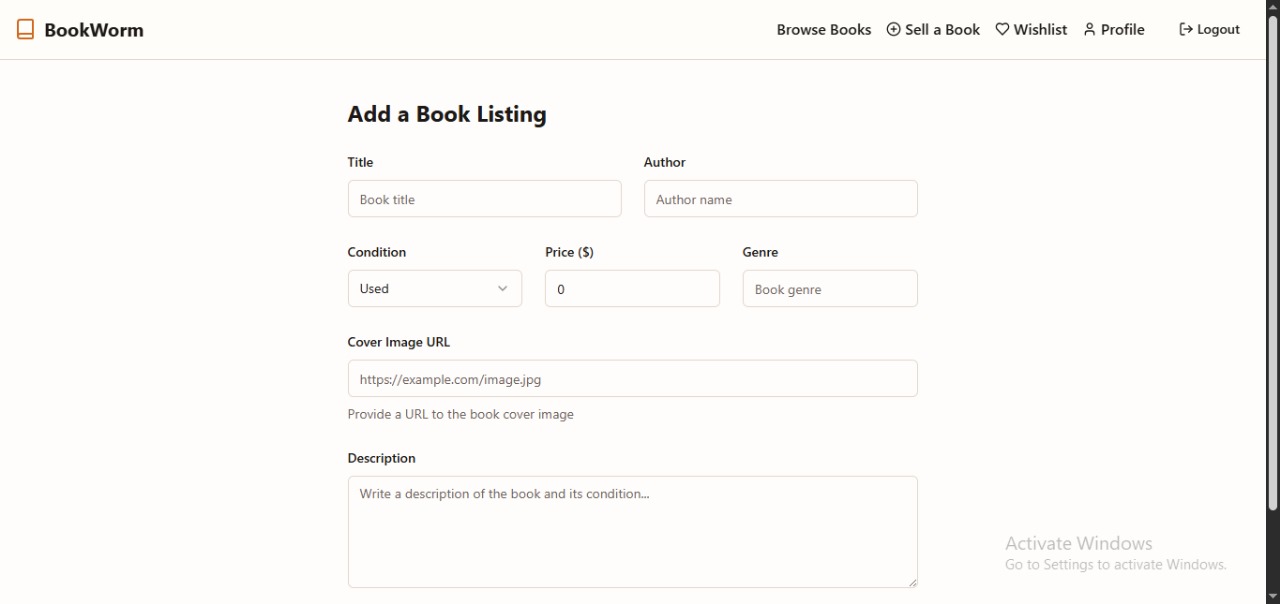
**8. Authentication**

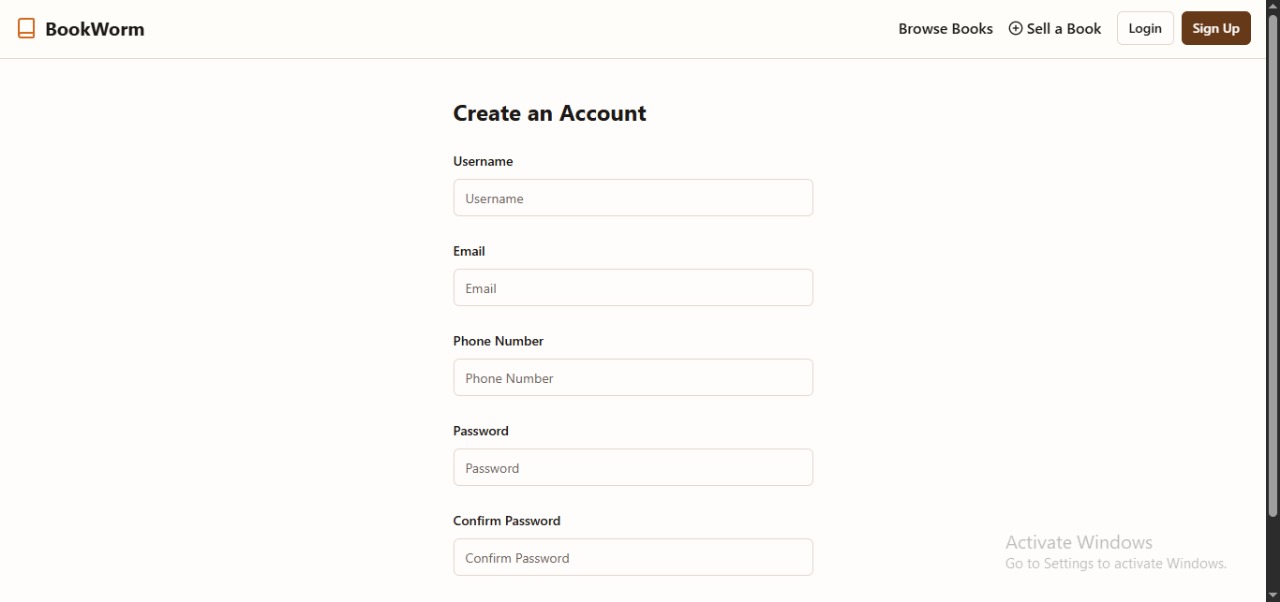
* Handled using **JWT tokens** for secure session management.
* Auth middleware validates token presence and integrity on protected routes.
* Tokens are stored in localStorage on the client side.

**9. User Interface**

* Clean and modern UI using Tailwind CSS & Radix UI
* Components: Navbar, Book Cards, Profile Pages, Exchange Modal
* Mobile Responsive Design



****

****

**10. Testing**

* Manual testing of UI and flow
* Backend API tested via Postman
* Unit tests with **Jest** or **Mocha** (Optional setup)

**11. Screenshots or Demo**

* GitHub Repo: <https://github.com/FalguniJain/BookStore>
* Live Demo (Replit or Vercel): <https://drive.google.com/file/d/1Aa71aYigDDnYBcx3ZWH2Z4cqukD6ZWur/view?usp=sharing>

**12. Known Issues**

* No built-in trust or rating system between users
* Cannot upload book cover images yet
* Mobile app version not available

**13. Future Enhancements**

* Add messaging system between users
* Implement user reviews & trust score
* Mobile app version (React Native)
* Gamification: badges for book exchanges
* Add filters by genre, location, author