Book Allocation System – Full StackProject

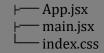
Tech Stack:

- Frontend: React.js, Tailwind CSS, React Router, Axios, React-Toastify
- Backend: Node.js, Express.js, MongoDB, Mongoose, JWT, bcrypt
- UX Enhancements: React-Toastify (for notifications), React-Spinners (for loaders)

Folder Structure

Backend (book-backend/)

Frontend (book-frontend/)



□ Backend Setup

* Environment Variables (.env)

ini CopyEdit PORT=5000 MONGO_URI=your_mongodb_connection JWT_SECRET=your_jwt_secret_key

* Install & Run

bash CopyEdit cd book-backend npm install npm run dev

Auth Routes (/api/auth)

- POST /register Register new user
- POST /login Login and receive token in HTTP-only cookie

Book Routes (/api/books)

- · All routes are protected by JWT middleware
- GET / Get all books
- POST / Add a book
- PUT /:id Update a book (only by uploader)
- DELETE /:id Delete a book (only by uploader)

Frontend Setup

★ Install & Run

bash CopyEdit cd book-frontend npm install npm run dev

Axios Setup (src/services/api.js)

```
js
CopyEdit
import axios from "axios";

const API = axios.create({
  baseURL: "http://localhost:5000/api",
  withCredentials: true,
});

export default API;
```

Application Workflow

Auth Flow

- 1. **Register** \rightarrow Validates input \rightarrow Sends data to /api/auth/register
- 2. **Login** → Validates input → Receives token cookie and username → Saves username to localStorage
- 3. **Auth Middleware** checks JWT from cookie in every protected route

M Book Management Flow

1. Home Page Loads

- o Auth-protected route (/home)
- o Fetches books via GET /books
- o Shows loader using react-spinners

2. Uploading a Book

- Form with title & author
- o Calls POST /books
- Success → toast + refresh list

3. **Editing a Book**

- 。 Opens modal
- Edits fields
- o Calls PUT /books/:id
- Success → toast + refresh list

4. Deleting a Book

- Calls DELETE /books/:id
- Success → toast + refresh list

☑ ☐ UI Features

- A Gradient background and modern card design
- Toasts for every action using react-toastify
- 2 Loading spinner for data fetching
- Reusable components (BookCard, EditModal, Header)
- Protected route using token presence (/home)
- Present Logout clears cookie and local storage

Deployment Tips

Backend

- Host on Render, Railway, or VPS
- Set jwt_secret, Mongo_uri in deployed env

Frontend

- Host on Vercel or Netlify
- Set API_BASE_URL in .env and update api.js:

```
js
CopyEdit
const API = axios.create({
  baseURL: import.meta.env.VITE_API_BASE_URL,
  withCredentials: true,
});
```

Future Enhancements

- Add file/image upload with Multer (e.g. book cover)
- Pagination or search in book list
- Theme toggle (light/dark)
- Admin panel for managing users/books
- Profile page for each user