BOOKNEST

Educational Library

Project Name**: Booknest Library Management System**

By**: Mudusaritha Gangamina Kumaranyake**

Date**:** 2025.02.25

Introduction

The Booknest Library Management System is a web-based application designed to help users manage book records efficiently. The system enables authentication, book listing, addition, updating, and deletion functionalities.

Project Objectives

- Implement a full-stack application using React, TypeScript, .NET, and SQLite.
- Provide authentication and authorization via JWT.
- Develop a responsive UI for a smooth user experience.
- Ensure data persistence with SQLite and Entity Framework.

Technologies Used

- Frontend: React, TypeScript, Tailwind CSS, DaisyUI
- Backend: C# .NET Web API, Entity Framework Core
- Database: SQLite
- Authentication: ASP.NET Identity, JWT

Development Process

Backend Implementation

Tech Stack:

- C# .NET Web API
- Entity Framework Core
- SQLite Database

Database Setup:

The project uses two DbContexts:

1. AuthDbContext – Manages user authentication.
2. LMSDbContext – Manages book data.

Authentication & Authorization:

- Users register and log in using **JWT-based authentication**.
- Role-based access control is implemented with **Reader** and **Writer** roles.
- API endpoints require authentication tokens for secured access.

API Endpoints:

Method	Endpoint	Description
POST	/api/auth/register	User Registration
POST	/api/auth/login	User Login
GET	/api/books	Fetch book list
POST	/api/books	Add a new book
PUT	/api/books/{id}	Update book details
DELETE	/api/books/{id}	Delete a book

Frontend Implementation

Tech Stack:

- React, TypeScript
- Tailwind CSS, DaisyUI
- React Router for page navigation
- Axios for API integration

Key UI Components:

- Login/Register Forms – Allow users to authenticate.
- Manage Books Page – Displays books retrieved from the backend.and provide View, Update, Delete functionalities.
- Add Book Page – Provides functionality of creating a new book into the list..
- Modal Forms – Used for providing updating form and delete confirmation.
- Book Cards - Used for viewing the book image and full description of the book along with the author.

Challenges & Solutions

Challenge	Solution
Learning curve of C# .NET	Used 3 days to study and understand the basic concepts of the framework.
Tailwind CSS setup problems due to Version 4	Had to use version 3.4.17 due to difficulties with the newer version.
Authentication issues	Fixed JWT setup & token handling
Managing API requests	Used Axios with authentication headers
Seeded data was not editable	Switched from HasData() to dynamic insertion

Key Insights Learned

- **Entity Framework Core & Multiple DbContexts:** Managing authentication and book data separately improved project structure.
- **React & TypeScript:** Strengthened understanding of frontend component-based development.
- **JWT Authentication:** Learned how to secure APIs using tokens.
- **State Management with React Hooks:** Improved handling of UI state changes.
- **Axios for API Calls:** Used authentication headers for secure communication.
- **.NET learning curve:** Learned a lot about a backend technology which was very new to me and improved my full stack scope.

Conclusion

Project Summary:

The Library Management System successfully provides a simple and effective way to manage book records. The project combines secure authentication, a structured database, and a responsive frontend for a smooth user experience.

Future Improvements:

- Implement **pagination and search** for book listings.
- Add **profile management** for users.
- Enhance **UI styling** with animations and transitions.
- **Deploy** the application to a cloud platform.