Internship Assignment Report

Candidate: Heshan Jeewantha

Internship Program: Software Engineering Internship

Assignment: Full Stack Development of a Library Management System

Github repository: Library Management System

🧠 Overview

This report documents the development of a full-stack **Library Management System** designed as part of my software engineering internship assignment. The project includes a **C# .NET 8 backend** using **SQLite** and a **React + TypeScript frontend**, implementing essential CRUD functionality along with optional JWT-based user authentication. The system was built with scalability, maintainability, and user experience in mind.

A summary of this project is also available in the README.md file, which contains detailed backend and frontend setup steps, API documentation, and usage instructions.

🎯 Objectives

- Design and develop a functional and user-friendly system to manage library books.
- Implement clean, RESTful backend APIs with proper error handling.
- Build a responsive frontend interface using modern web technologies.
- Secure user access with JWT tokens (optional, included).
- Demonstrate independent full-stack development skills.

⚙️ **Backend Implementation**

- **Technology Stack**: C# .NET 8, Entity Framework Core, SQLite
- **Project Structure**:
    - Controllers: AuthController, BookController
    - Models: Book, User
    - Data: AppDbContext
    - Services: AuthService (login, register, token generation)
- **Key Features**:
    - RESTful API endpoints for books and user auth
    - JWT token-based authentication
    - Swagger integration for API testing
    - Validation and error handling (e.g., missing fields, duplicate users)
- **Database**:
    - Used SQLite for simplicity and embedded file-based storage
    - Applied migrations via dotnet ef

  Example Endpoint: POST /api/book creates a new book record

🎨 Frontend Implementation

- **Technology Stack**: React, TypeScript, Vite, Bootstrap
- **Pages Implemented**:
  - Login / Register
  - Home (list books)
  - AddBook, UpdateBook
- **Component Structure**:
  - Navbar.tsx: Responsive navigation bar with logo and auth links
  - BookForm.tsx: Shared form component for adding/updating books
  - BookList.tsx: Displays list of books with edit/delete actions
- **State Management**:
  - Used React's built-in state hooks (useState, useEffect)
  - Token stored in localStorage to persist user sessions
- **Styling**:
  - Bootstrap classes for form controls, spacing, and buttons
  - Responsive layout for all screen sizes

🔐 Authentication Implementation

- JWT token generated on login
- Stored in browser's localStorage
- Passed in Authorization header for protected book API requests
- Logout clears the token and redirects user to login screen

🧩 Challenges Faced

1. **CORS Configuration**:
    - Needed to enable CORS in Program.cs for frontend API calls to succeed.
2. **Token Handling**:
    - Initial bug: token was being set statically in Axios config → fixed with dynamic authHeader() method.
3. **SQLite Migrations**:
    - Required understanding of EF Core CLI commands and migration flow.
4. **Protected Routes in React**:
    - Implemented ProtectedRoute.tsx to guard routes based on login state.
5. **Version Control**:
    - Initially faced merge conflicts and untracked file issues during Git operations. Learned to use .gitignore, resolve conflicts manually, and commit logically grouped changes.
6. **Error Handling**:
    - Encountered runtime errors such as 401 (unauthorized) due to missing or outdated JWT tokens. Resolved by improving token validation and adding user feedback in the UI.

✨ Additional Features

- ✅ Responsive layout with Bootstrap
- ✅ Token persistence across sessions
- ✅ Reusable book form component
- ✅ Graceful error messages

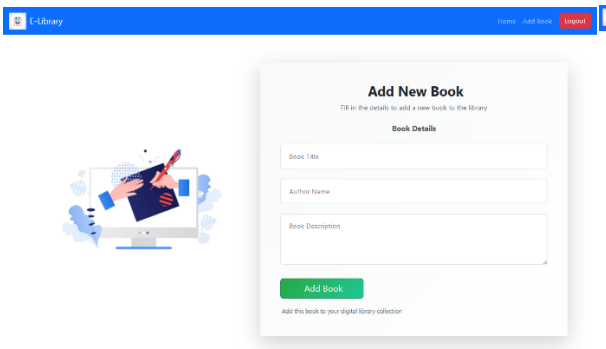
💡 Key Learnings

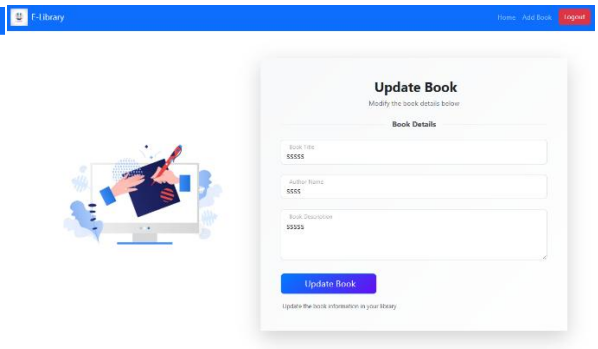
- Gained hands-on experience in designing clean, decoupled REST APIs.
- Improved frontend architectural decisions using React and component reuse.
- Understood the practical aspects of JWT authentication.
- Enhanced debugging and problem-solving when handling API integration issues.
- Learned to manage source code effectively with Git and GitHub.

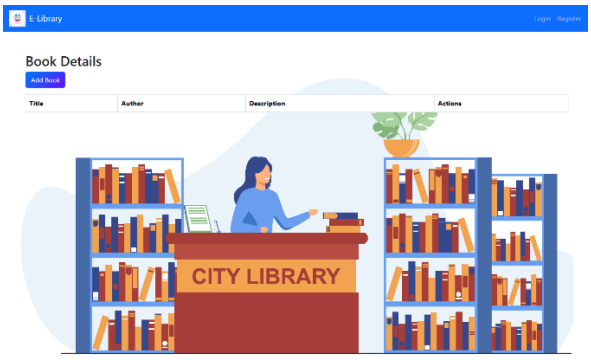
✅ Conclusion

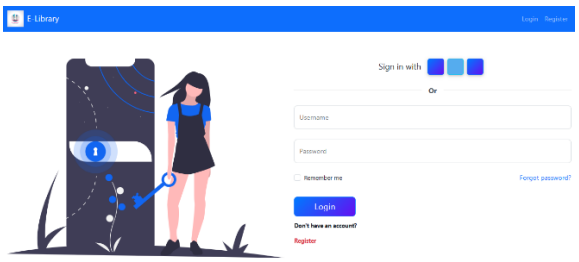
This assignment allowed me to independently develop a full-stack application, apply modern web technologies, and solve real-world integration problems. It also prepared me to take ownership of tasks in a real development environment and reinforced my understanding of full-stack application workflows.

Ui









Prepared by:

Heshan jeewantha
heshanJeewantha@gmail.com