Title:
Library Management System — Full Stack Project

Submitted by:
Akhila Induwara
Computing and Information Systems
Sabaragamuwa University of Sri Lanka

Course: Software Development Assignment
Technologies: C# .NET 10, React + TypeScript, SQLite, Entity Framework
Submitted to: Expernetic

Date:
08/12/2025

Introduction

This project is a full-stack **Library Management System** developed using:

- **C# .NET 10** for backend API
- **SQLite** with **Entity Framework Core**
- **React + TypeScript** for frontend
- **Axios** for API communication

The goal was to build a complete system demonstrating CRUD functionality, API integration, UI development, and clean architecture.

Project Overview

The system allows users to:

- Add new books
- Edit book details
- Delete books
- View all books
- Communicate with a backend API
- Handle data using SQLite

This project demonstrates independent development, integration skills, and the ability to deliver a complete working solution.

System Architecture

- **Frontend:** sends requests using Axios
- **Backend:** exposes REST endpoints
- **Database:** stores book records
- **EF Core:** handles ORM and migrations

Backend Implementation

a. Technologies Used

- .NET 10
- C#
- Entity Framework Core
- SQLite

b. Project Structure

- Controllers/
- Models/
- Data/
- Program.cs (Dependency injection)

c. Core Features

- REST API
- CRUD endpoints
- Model validation
- Database migrations

Frontend Implementation

a. Technologies Used

- React 19
- TypeScript
- React Router
- Axios

b. Features

- Add Book form
- Edit Book form

- Delete confirmation
- List view table
- API calls to backend

Database Design

Database: SQLite

Include a table design diagram:

Column	Type	Description
Id	int	Primary Key
Title	text	Book title
Author	text	Author name
Year	int	Published year
Genre	text	Category

CRUD Operations Explanation

Create (POST)

User adds a new book → sent to backend → stored in SQLite.

Read (GET)

Frontend fetches book list.

Update (PUT)

User edits book details → API updates row.

Delete (DELETE)

Book removed via ID.

Error Handling & Validation

Backend:

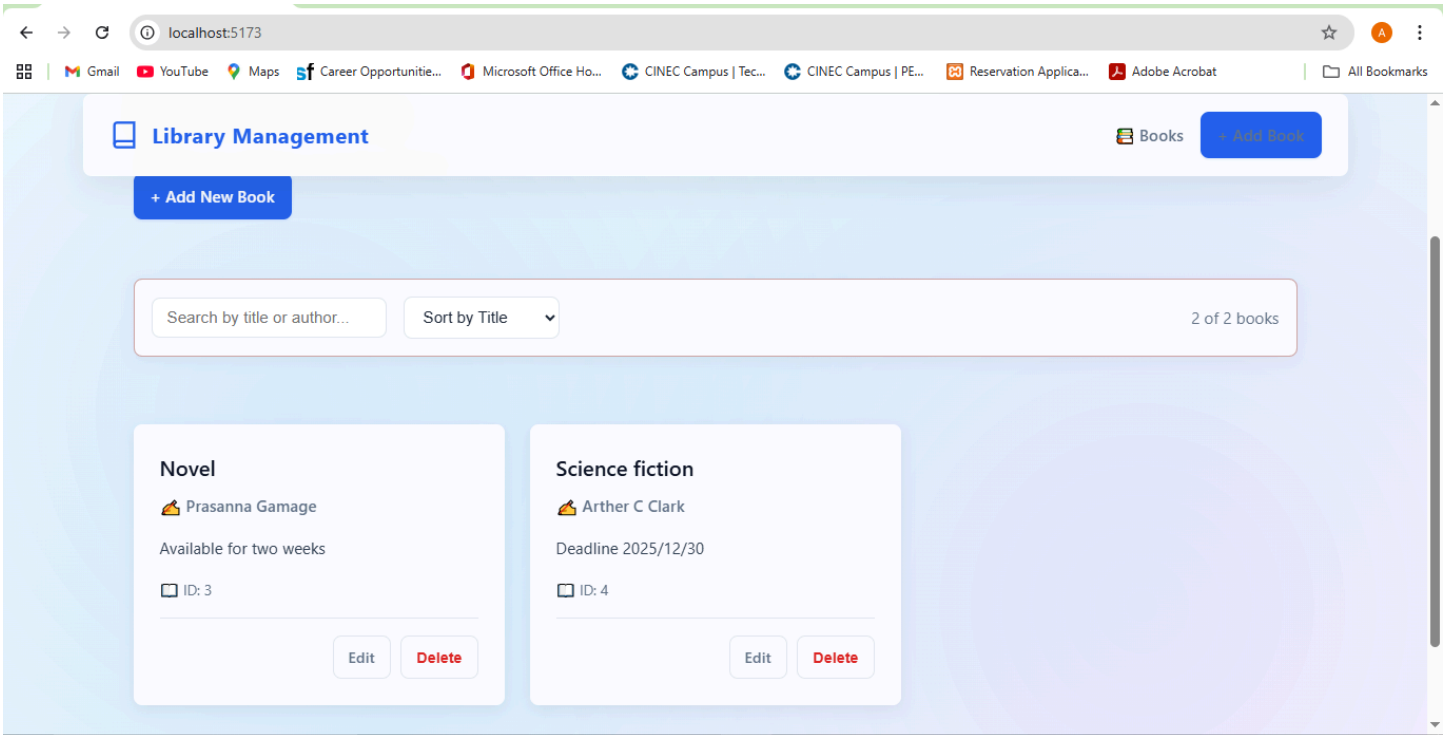
- ModelState validation
- Try–catch
- Return meaningful error messages

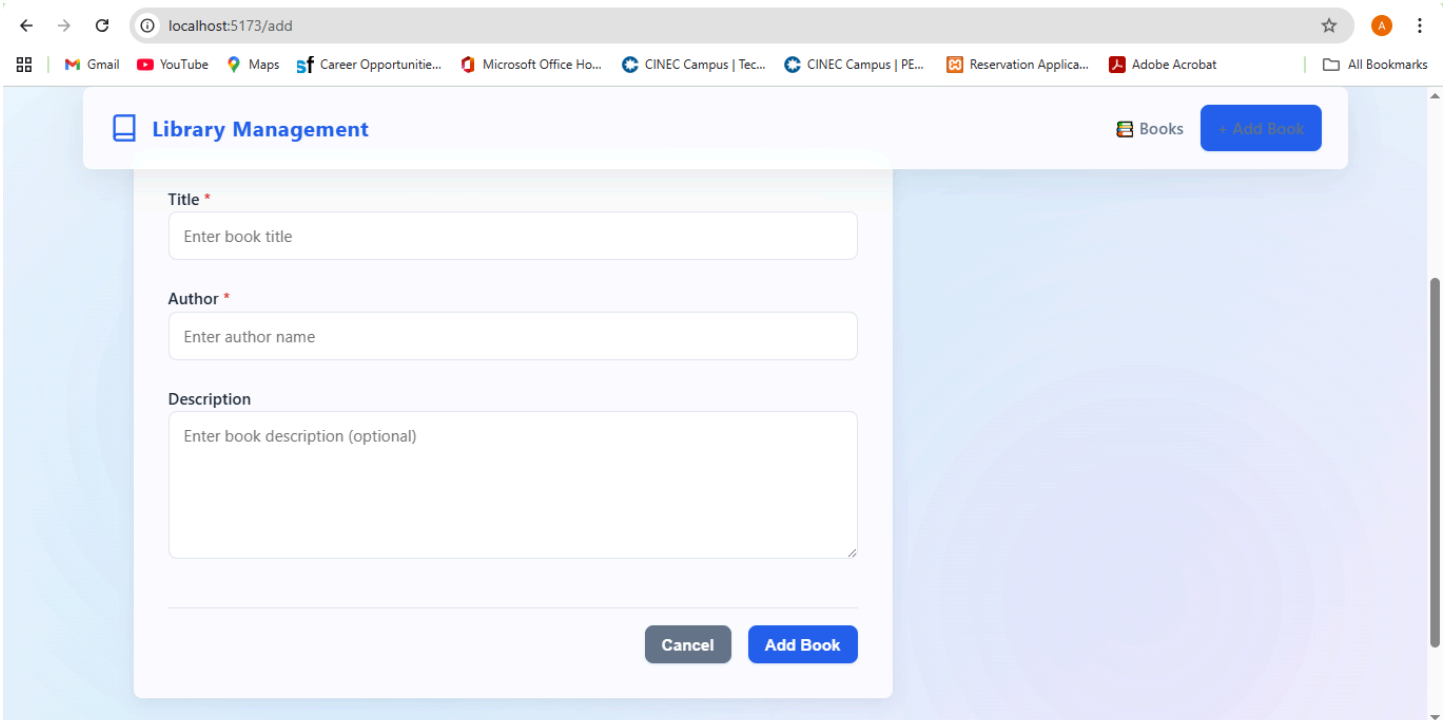
Frontend:

- Required fields
- Alerts on failure
- Disabled submit button

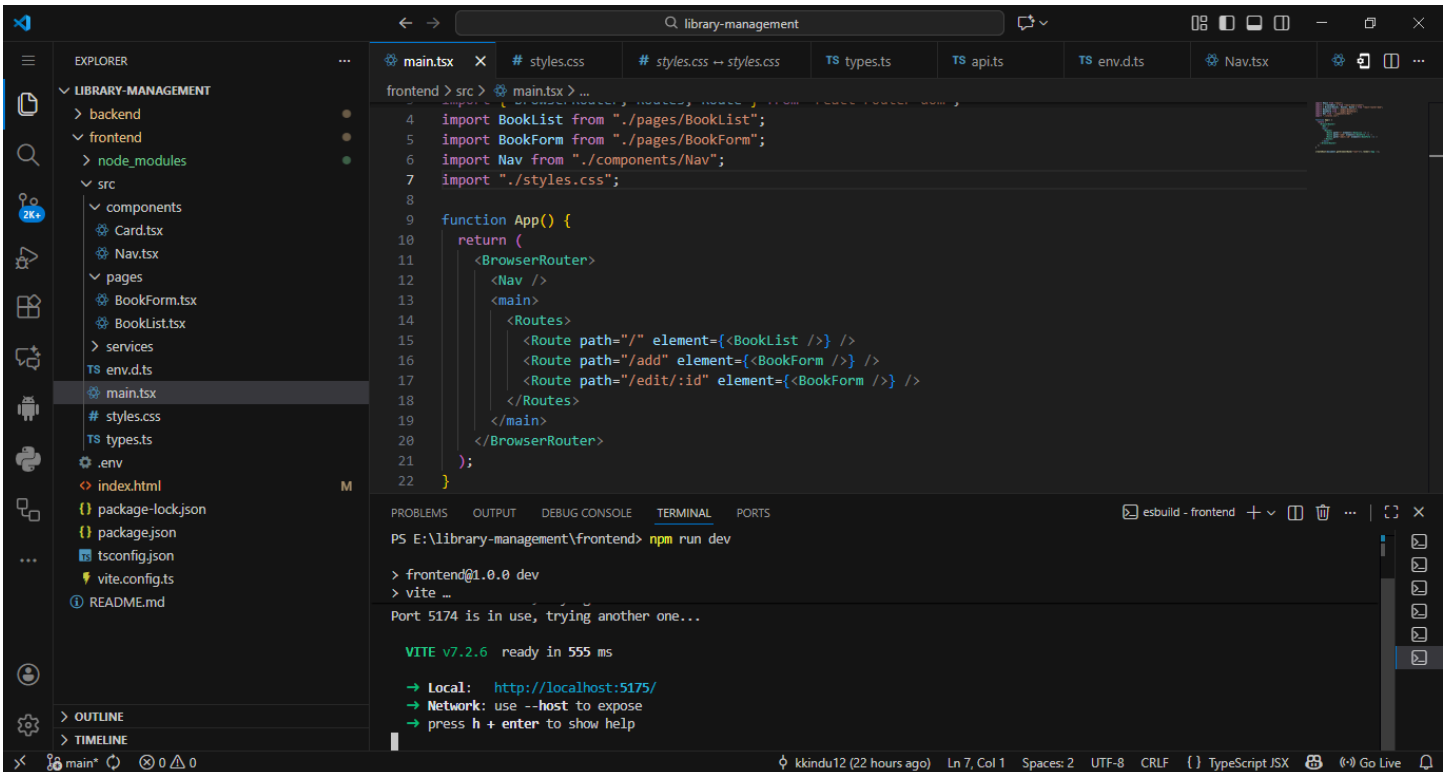
UI/UX and Responsiveness

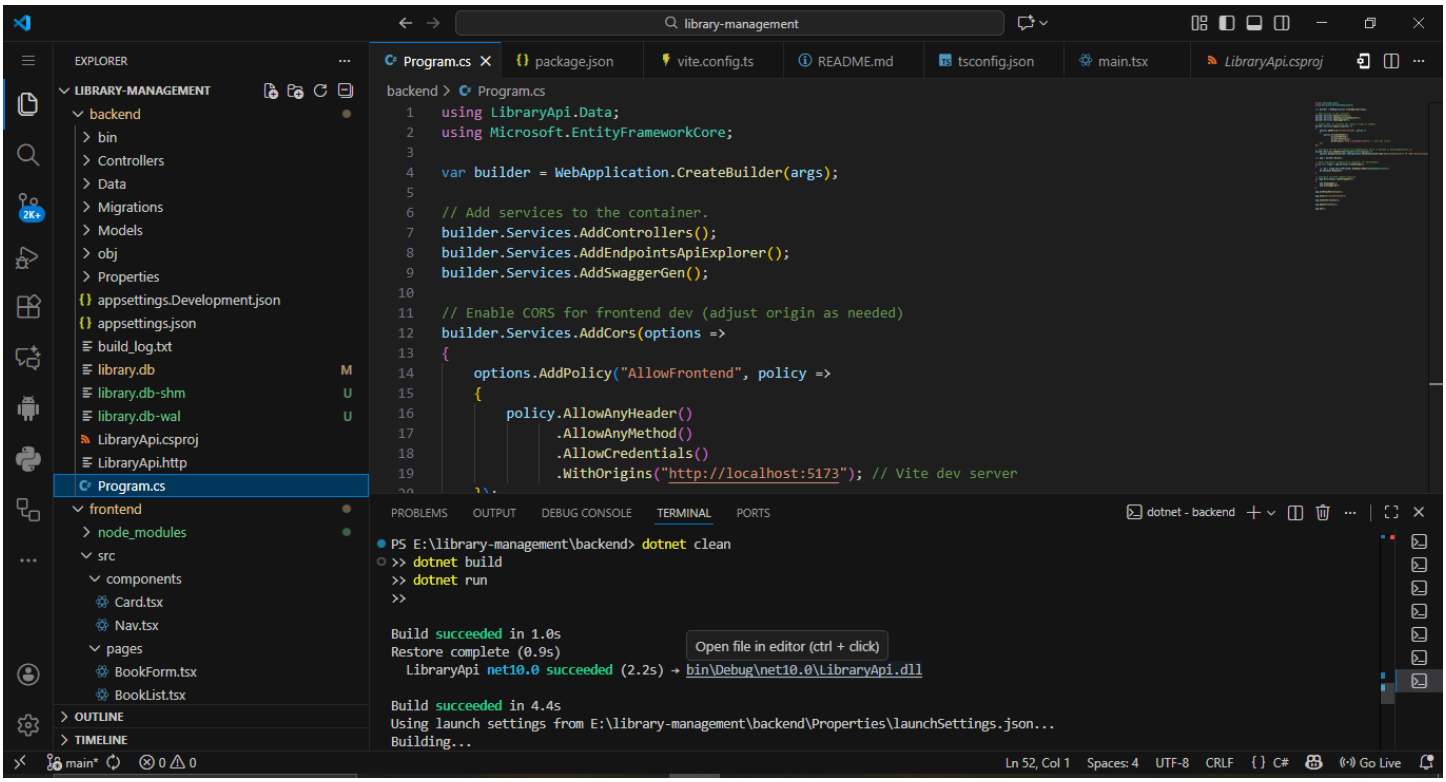
- Clean design
- Responsive layout
- Form validation
- Table for listing
- Simple and Professionally




localhost:5173/add
Gmail
YouTube
Maps
Career Opportunitie...
Microsoft Office Ho...
CINEC Campus | Tec...
CINEC Campus | PE...
Reservation Applica...
Adobe Acrobat
All Bookmarks
Library Management
Books
+ Add Book
Title *
Enter book title
Author *
Enter author name
Description
Enter book description (optional)
Cancel
Add Book

Running


library-management
EXPLORER
LIBRARY-MANAGEMENT
backend
frontend
node_modules
src
components
Card.tsx
Nav.tsx
pages
BookForm.tsx
BookList.tsx
services
env.d.ts
main.tsx
styles.css
types.ts
.env
index.html
package-lock.json
package.json
tsconfig.json
vite.config.ts
README.md
OUTLINE
TIMELINE
main.tsx
styles.css
styles.css ↔ styles.css
types.ts
api.ts
env.d.ts
Nav.tsx
frontend > src > main.tsx > ...
import BookList from "./pages/BookList";
import BookForm from "./pages/BookForm";
import Nav from "./components/Nav";
import "./styles.css";
function App() {
return (
<BrowserRouter>
<Nav />
<main>
<Routes>
<Route path="/" element={<BookList />} />
<Route path="/add" element={<BookForm />} />
<Route path="/edit/:id" element={<BookForm />} />
</Routes>
</main>
</BrowserRouter>
);
}
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
esbuild - frontend
PS E:\library-management\frontend> npm run dev
> frontend@1.0.0 dev
> vite …
Port 5174 is in use, trying another one...
VITE v7.2.6 ready in 555 ms
→ Local: http://localhost:5175/
→ Network: use --host to expose
→ press h + enter to show help
main*
kkindu12 (22 hours ago)
Ln 7, Col 1
Spaces: 2
UTF-8
CRLF
TypeScript JSX
Go Live


library-management
EXPLORER
LIBRARY-MANAGEMENT
backend
bin
Controllers
Data
Migrations
Models
obj
Properties
appsettings.Development.json
appsettings.json
build_log.txt
library.db
library.db-shm
library.db-wal
LibraryApi.csproj
LibraryApi.http
Program.cs
frontend
node_modules
src
components
Card.tsx
Nav.tsx
pages
BookForm.tsx
BookList.tsx
OUTLINE
TIMELINE
Program.cs
package.json
vite.config.ts
README.md
tsconfig.json
main.tsx
LibraryApi.csproj
backend > Program.cs
using LibraryApi.Data;
using Microsoft.EntityFrameworkCore;
var builder = WebApplication.CreateBuilder(args);
// Add services to the container.
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();
// Enable CORS for frontend dev (adjust origin as needed)
builder.Services.AddCors(options =>
{
options.AddPolicy("AllowFrontend", policy =>
{
policy.AllowAnyHeader()
.AllowAnyMethod()
.AllowCredentials()
.WithOrigins("http://localhost:5173"); // Vite dev server
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
dotnet - backend
PS E:\library-management\backend> dotnet clean
>> dotnet build
>> dotnet run
>>
Build succeeded in 1.0s
Restore complete (0.9s)
LibraryApi net10.0 succeeded (2.2s) → bin\Debug\net10.0\LibraryApi.dll
Open file in editor (ctrl + click)
Build succeeded in 4.4s
Using launch settings from E:\library-management\backend\Properties\launchSettings.json...
Building...
main*
0 0
Ln 52, Col 1
Spaces: 4
UTF-8
CRLF
C#
Go Live