**Book Management System - Output Report**

Name: Yash Sudhakar Jadhav

Superset ID: 5298089

College: Imperial College of Engineering and Research, Pune

Email: yashj98819@example.com

Date & Time: July 11, 2025 - 05:00 PM

This document contains visual outputs of the Book Management System project, including:

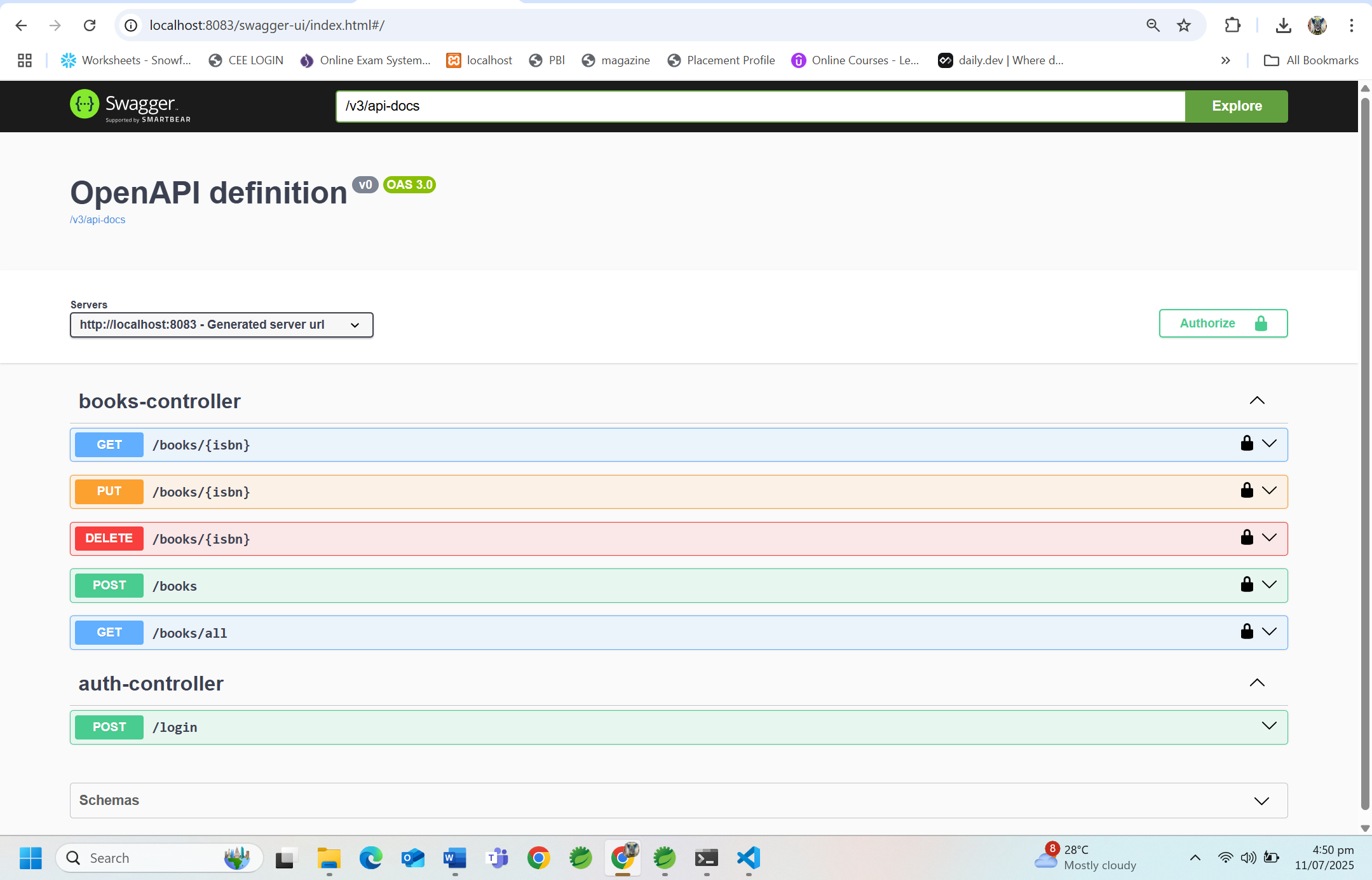
- Swagger/API Screens

- Database Output

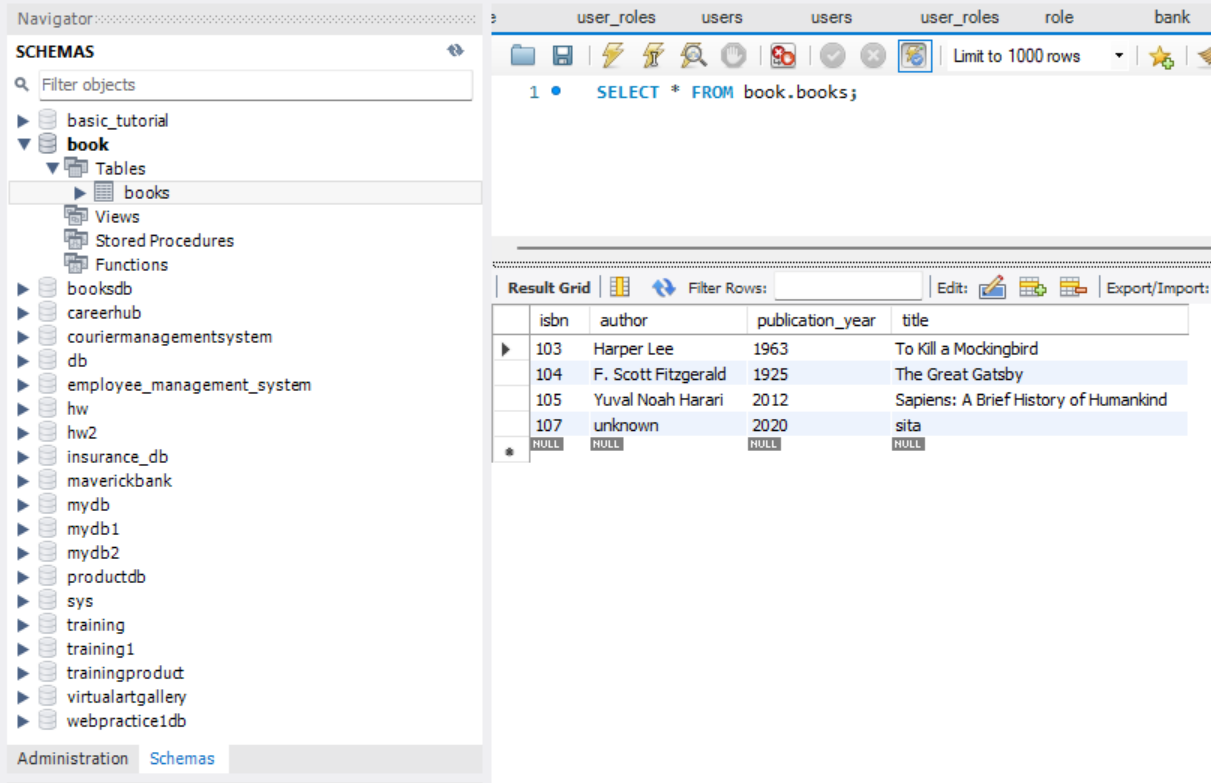
- User Interface Screenshots

- **Book Management System - Complete Project Documentation**

# - Swagger/API Screens

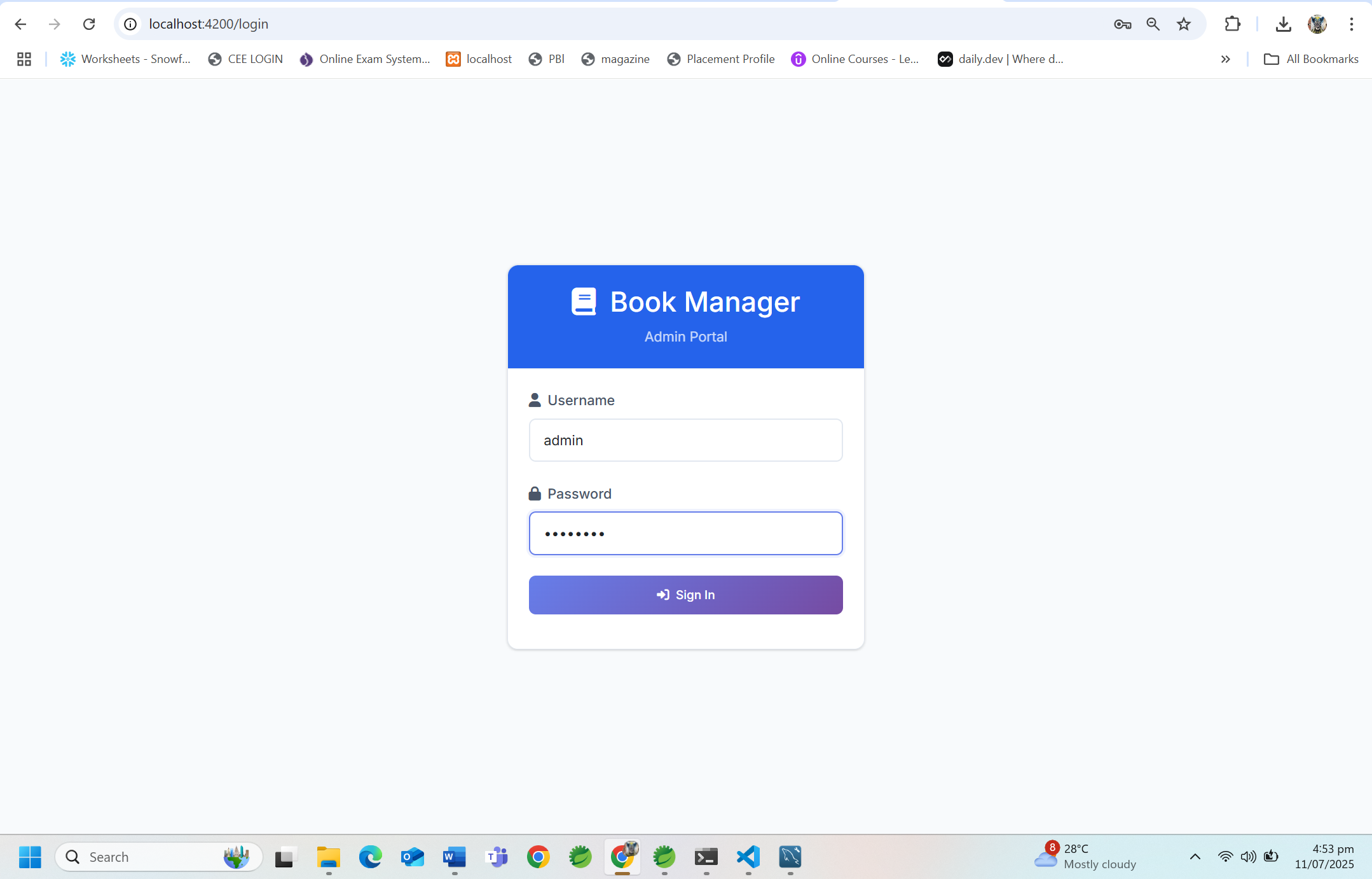


# - Database Output

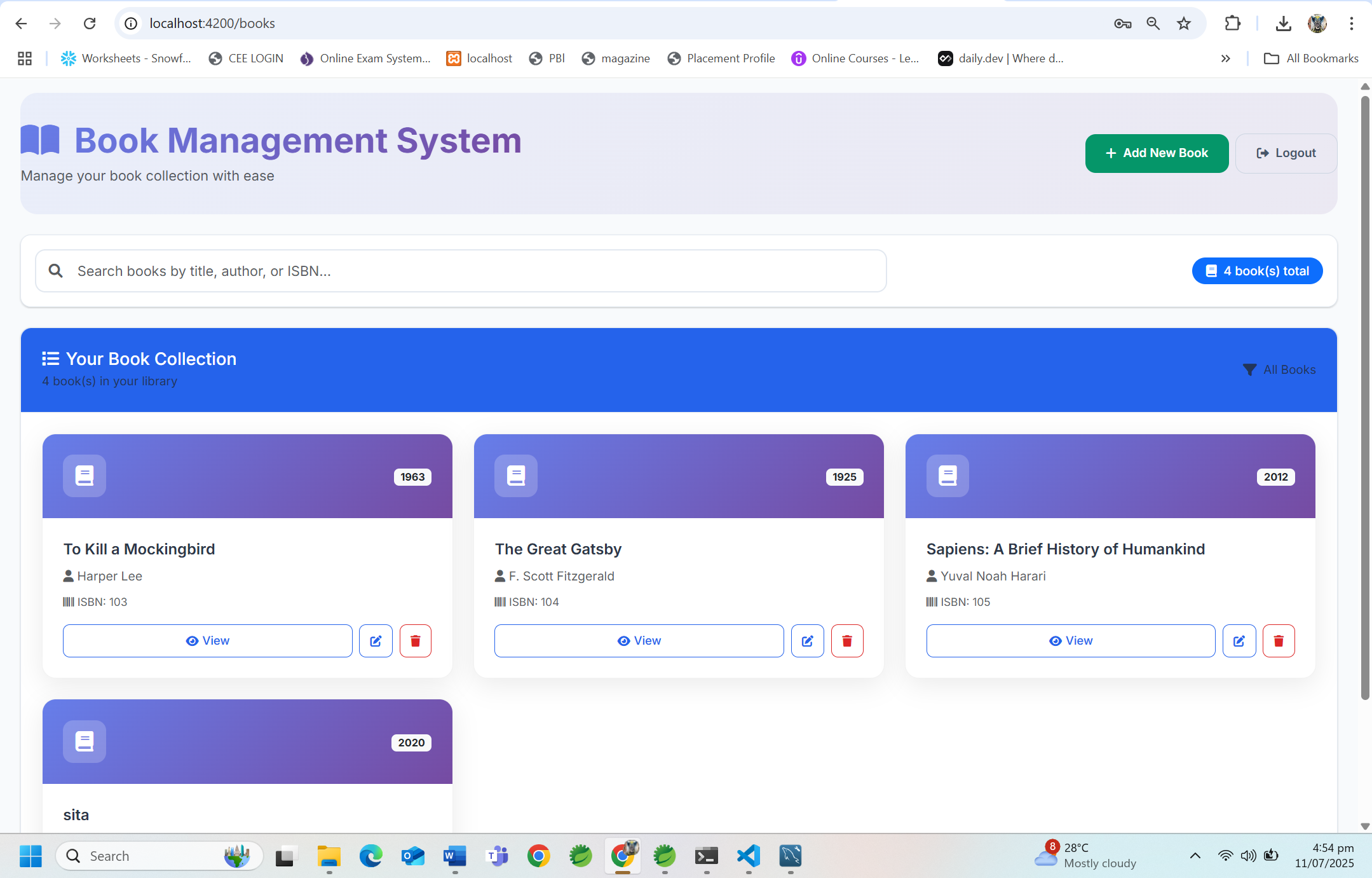


# - User Interface Screenshots

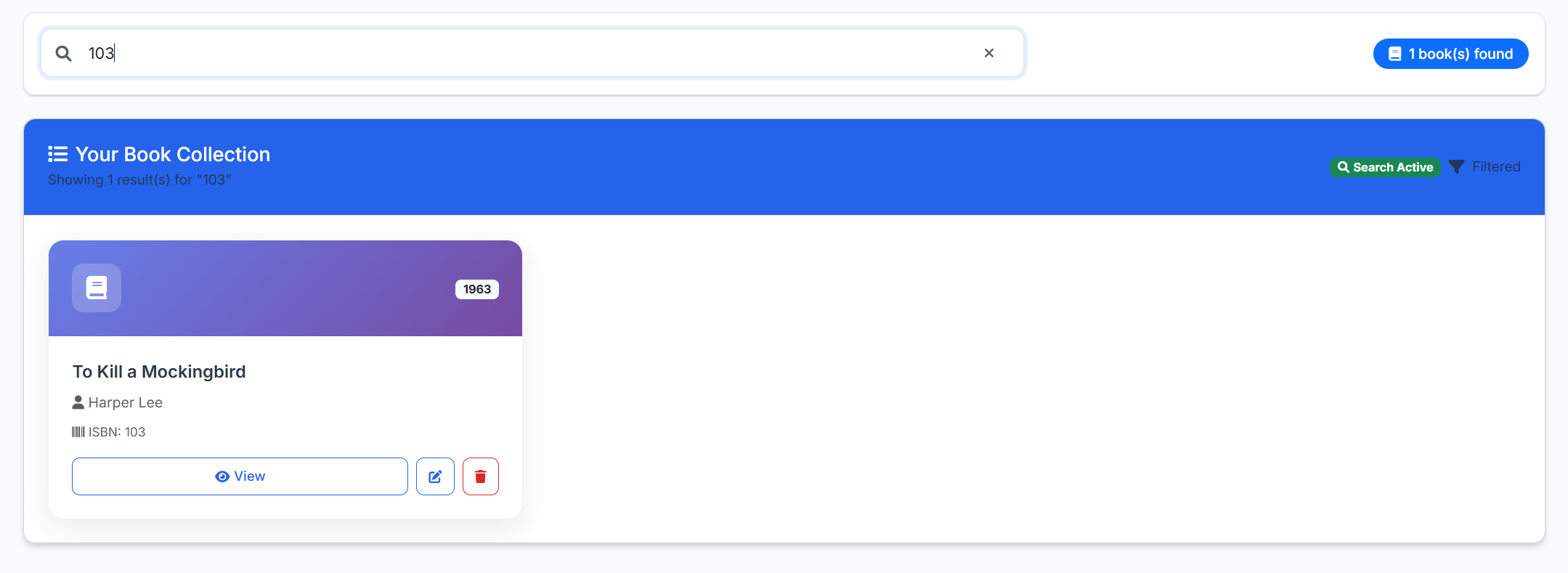
## Login page



## Dashboard



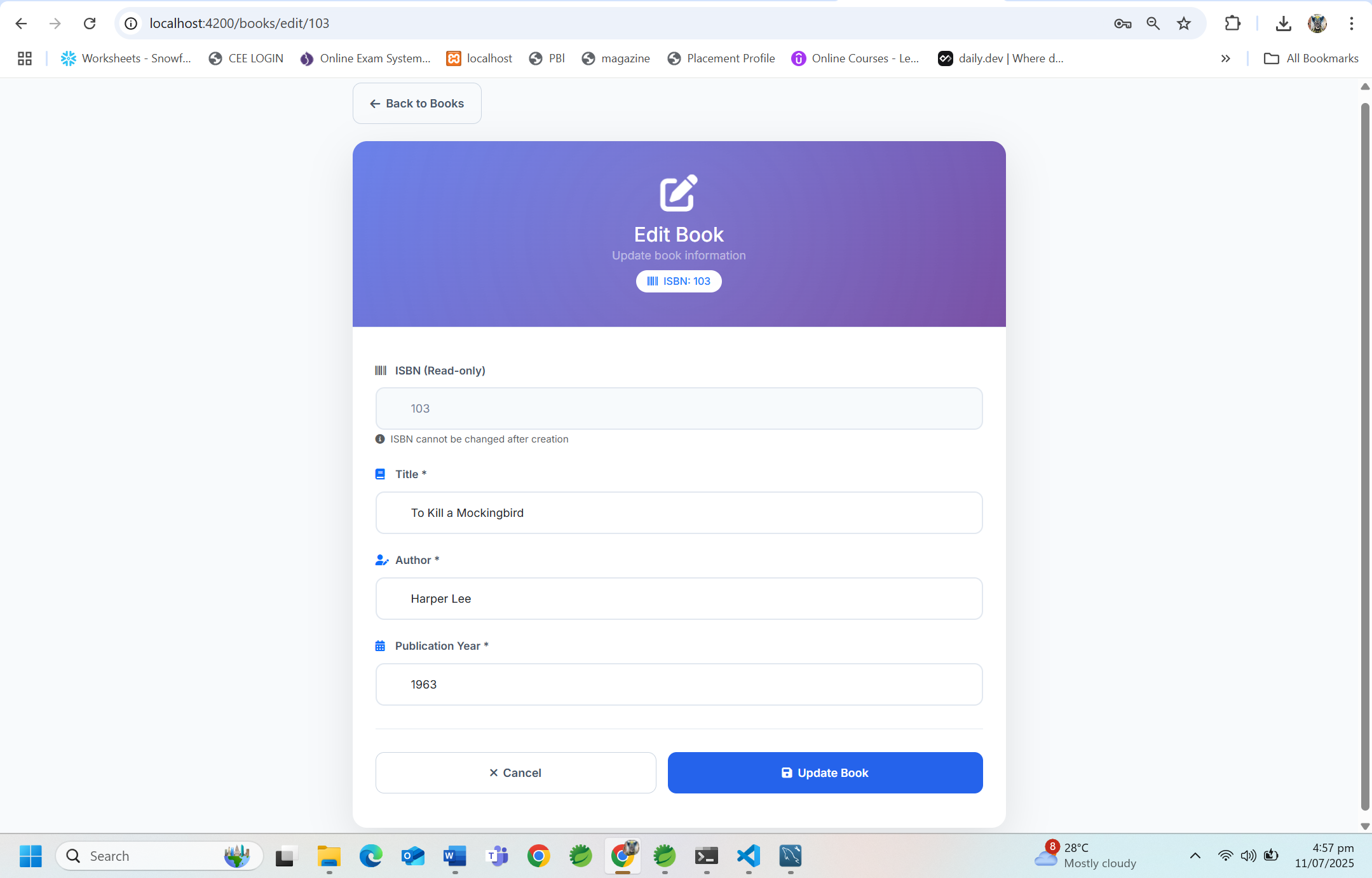
## Search by ISBN Number



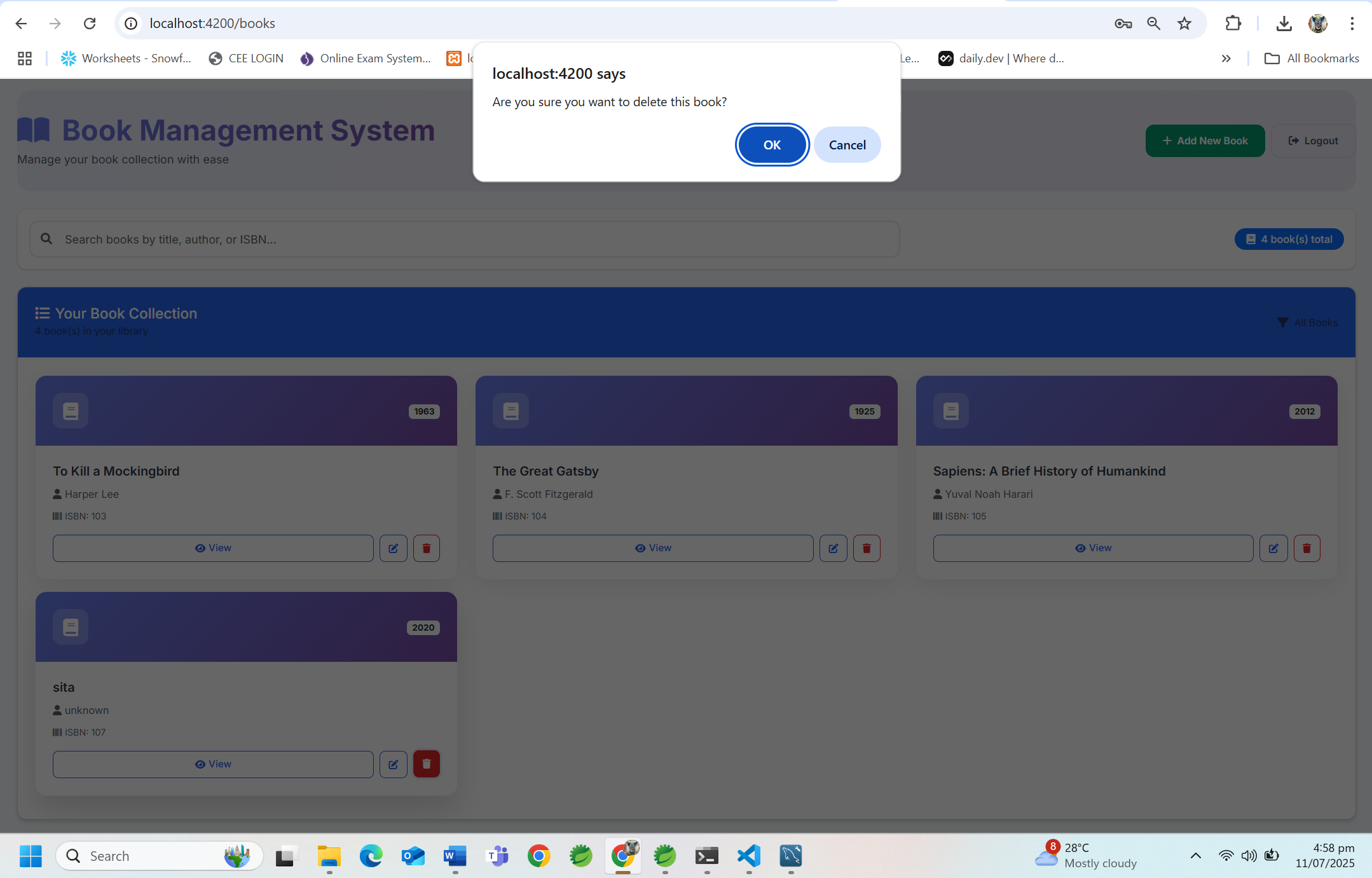
## Book Details



## Book Update



## Book Deletion



# Book Management System - Complete Project Documentation

## Table of Contents

1. Project Overview
2. System Architecture
3. Technology Stack
4. Backend Implementation
5. Database Design
6. Frontend Implementation
7. API Documentation
8. Security Implementation
9. UI/UX Design
10. Installation and Setup
11. Usage Guide with Screenshots
12. Project Screenshots
13. Testing and Troubleshooting

## Project Overview

### Project Name

Book Management System - Full Stack Application

### Description

A comprehensive full-stack web application for managing book collections with secure authentication, CRUD operations, and modern responsive design. The system includes both frontend (Angular) and backend (Spring Boot) components with MySQL database integration.

### Project Goals

* Create a complete full-stack book management solution
* Implement secure JWT-based authentication
* Provide efficient book search and filtering capabilities
* Ensure responsive design for all devices
* Maintain professional, modern UI/UX standards
* Demonstrate full-stack development best practices

### Key Benefits

* Full-Stack Solution: Complete frontend and backend integration
* Security: JWT-based authentication with secure API endpoints
* Performance: Optimized database queries and efficient frontend
* Usability: Modern, intuitive interface design
* Scalability: Modular architecture for future enhancements

### Technology Highlights

* Frontend: Angular 18+ with TypeScript
* Backend: Spring Boot with Java
* Database: MySQL with JPA/Hibernate
* Authentication: JWT tokens
* API: RESTful web services

## System Architecture

### Full-Stack Architecture Overview

┌─────────────────────────────────────────┐  
│ Client Layer (Browser) │  
├─────────────────────────────────────────┤  
│ • Angular Application │  
│ • TypeScript Components │  
│ • Responsive UI/UX │  
│ • JWT Token Management │  
└─────────────────────────────────────────┘  
 │  
 │ HTTP/HTTPS Requests  
 │ REST API Calls  
 │  
┌─────────────────────────────────────────┐  
│ Application Layer (Server) │  
├─────────────────────────────────────────┤  
│ • Spring Boot Application │  
│ • REST Controllers │  
│ • JWT Authentication │  
│ • Service Layer │  
│ • Data Validation │  
└─────────────────────────────────────────┘  
 │  
 │ JPA/Hibernate  
 │ Database Queries  
 │  
┌─────────────────────────────────────────┐  
│ Data Layer (Database) │  
├─────────────────────────────────────────┤  
│ • MySQL Database │  
│ • Book Entity Table │  
│ • User Authentication Table │  
│ • Indexes and Relationships │  
└─────────────────────────────────────────┘

### Data Flow Architecture

1. User Request → Angular Frontend
2. HTTP Request → Spring Boot Backend
3. JWT Validation → Security Layer
4. Business Logic → Service Layer
5. Database Query → MySQL Database
6. Response Processing → JSON Response
7. UI Update → Angular Components

## Technology Stack

### Frontend Technologies

|  |  |  |
| --- | --- | --- |
| Technology | Version | Purpose |
| Angular | 18+ | Frontend Framework |
| TypeScript | Latest | Programming Language |
| HTML5/CSS3 | Latest | Markup and Styling |
| Bootstrap | 5.x | UI Framework |
| Font Awesome | 6.0 | Icon Library |
| RxJS | Latest | Reactive Programming |

### Backend Technologies

|  |  |  |
| --- | --- | --- |
| Technology | Version | Purpose |
| Java | 17+ | Programming Language |
| Spring Boot | 3.x | Backend Framework |
| Spring Security | 6.x | Security Framework |
| Spring Data JPA | 3.x | Data Access Layer |
| Hibernate | 6.x | ORM Framework |
| Maven | 3.x | Build Tool |

### Database & Infrastructure

|  |  |  |
| --- | --- | --- |
| Technology | Version | Purpose |
| MySQL | 8.x | Primary Database |
| JWT | Latest | Authentication Tokens |
| Tomcat | Embedded | Application Server |
| Jackson | Latest | JSON Processing |

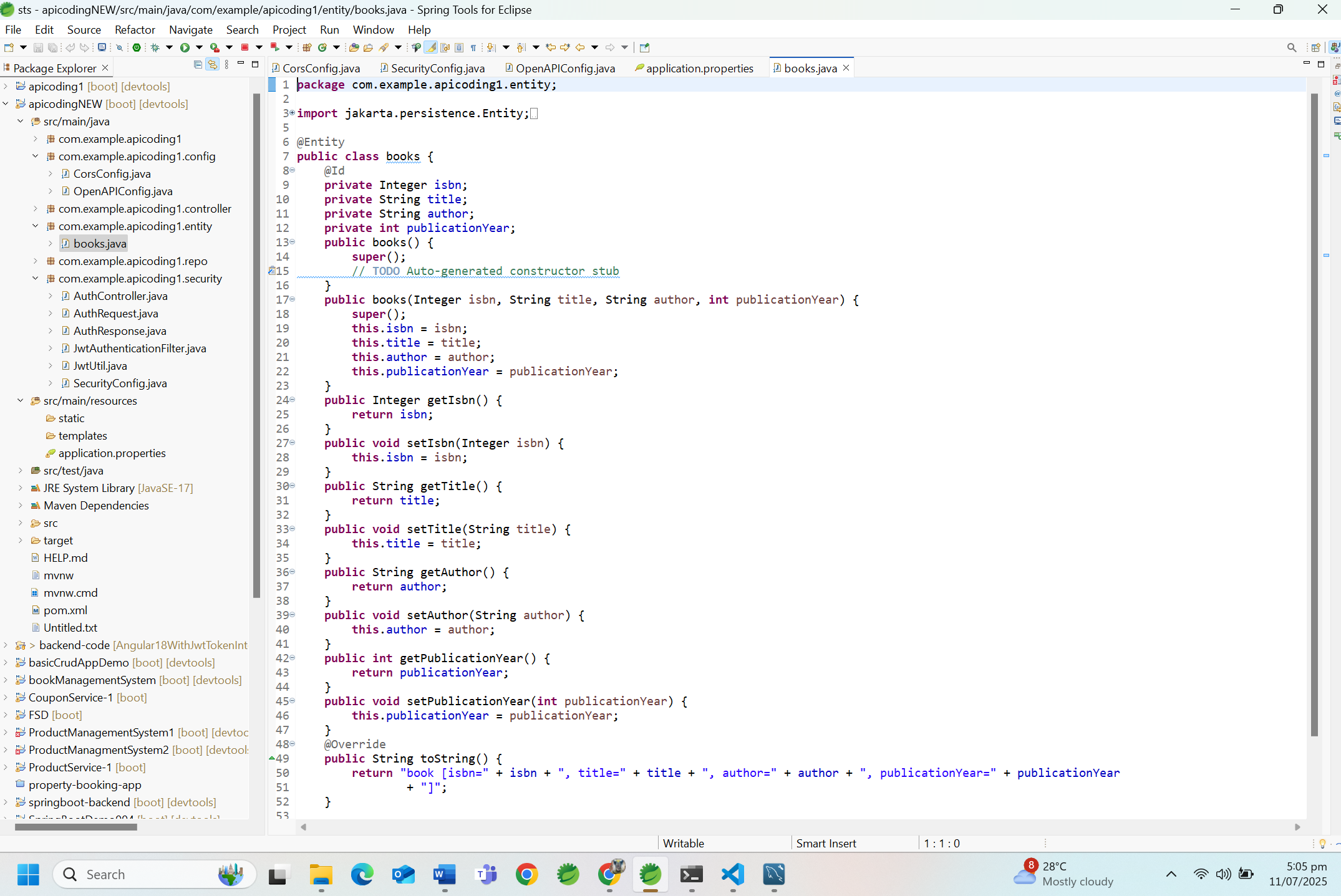
## Backend Implementation

### Project Structure

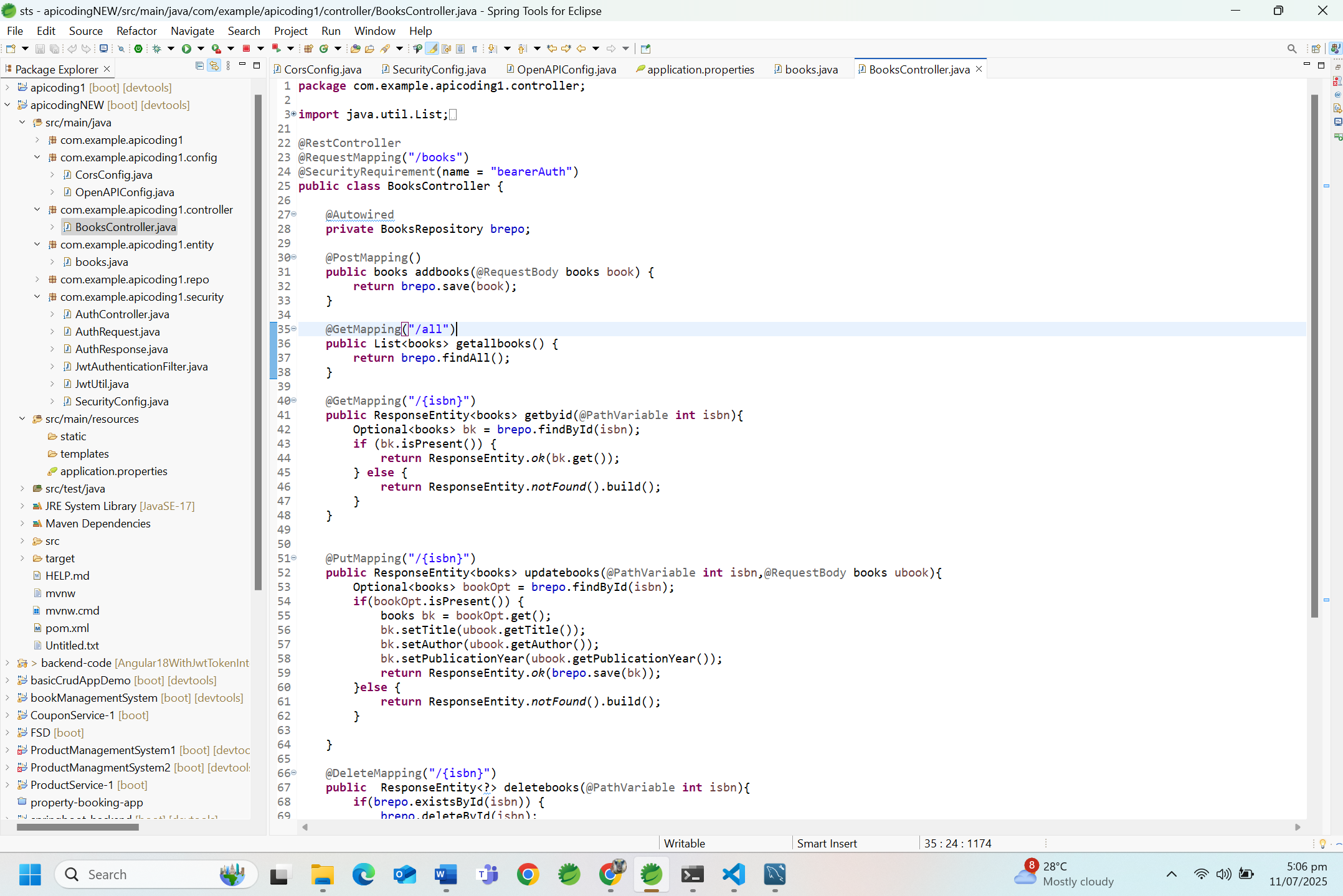
book-management-backend/  
├── src/main/java/com/bookmanagement/  
│ ├── controller/  
│ │ ├── AuthController.java  
│ │ ├── BookController.java  
│ │ └── GlobalExceptionHandler.java  
│ ├── service/  
│ │ ├── AuthService.java  
│ │ ├── BookService.java  
│ │ └── JwtService.java  
│ ├── repository/  
│ │ ├── BookRepository.java  
│ │ └── UserRepository.java  
│ ├── entity/  
│ │ ├── Book.java  
│ │ ├── User.java  
│ │ └── BaseEntity.java  
│ ├── dto/  
│ │ ├── BookDTO.java  
│ │ ├── LoginRequest.java  
│ │ └── AuthResponse.java  
│ ├── config/  
│ │ ├── SecurityConfig.java  
│ │ ├── CorsConfig.java  
│ │ └── JwtAuthenticationFilter.java  
│ └── BookManagementApplication.java  
├── src/main/resources/  
│ ├── application.properties  
│ └── data.sql  
└── pom.xml

### Key Backend Components

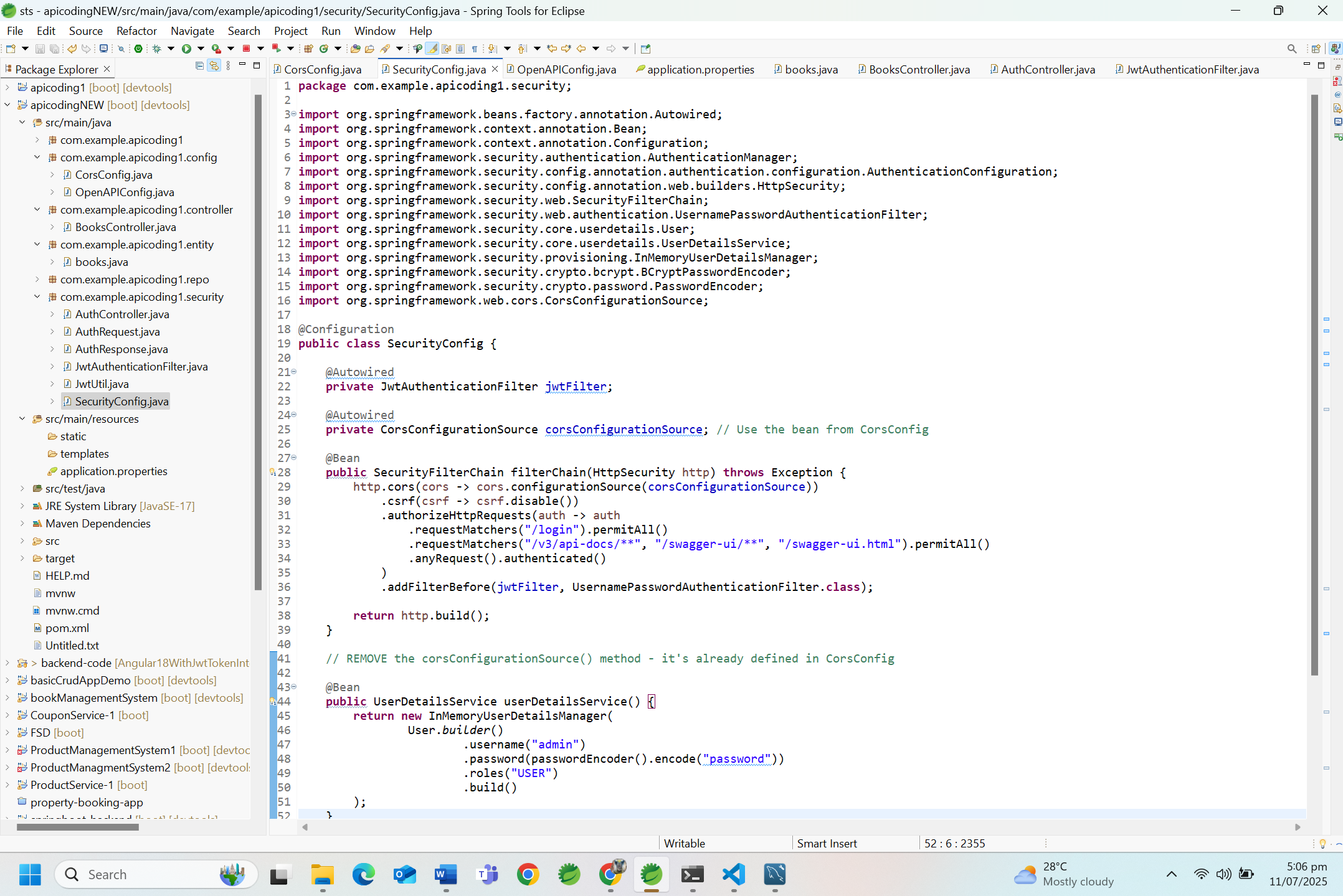
#### 1. Book Entity (Book.java)



#### 2. Book Controller (BookController.java)



#### 3. Security Configuration

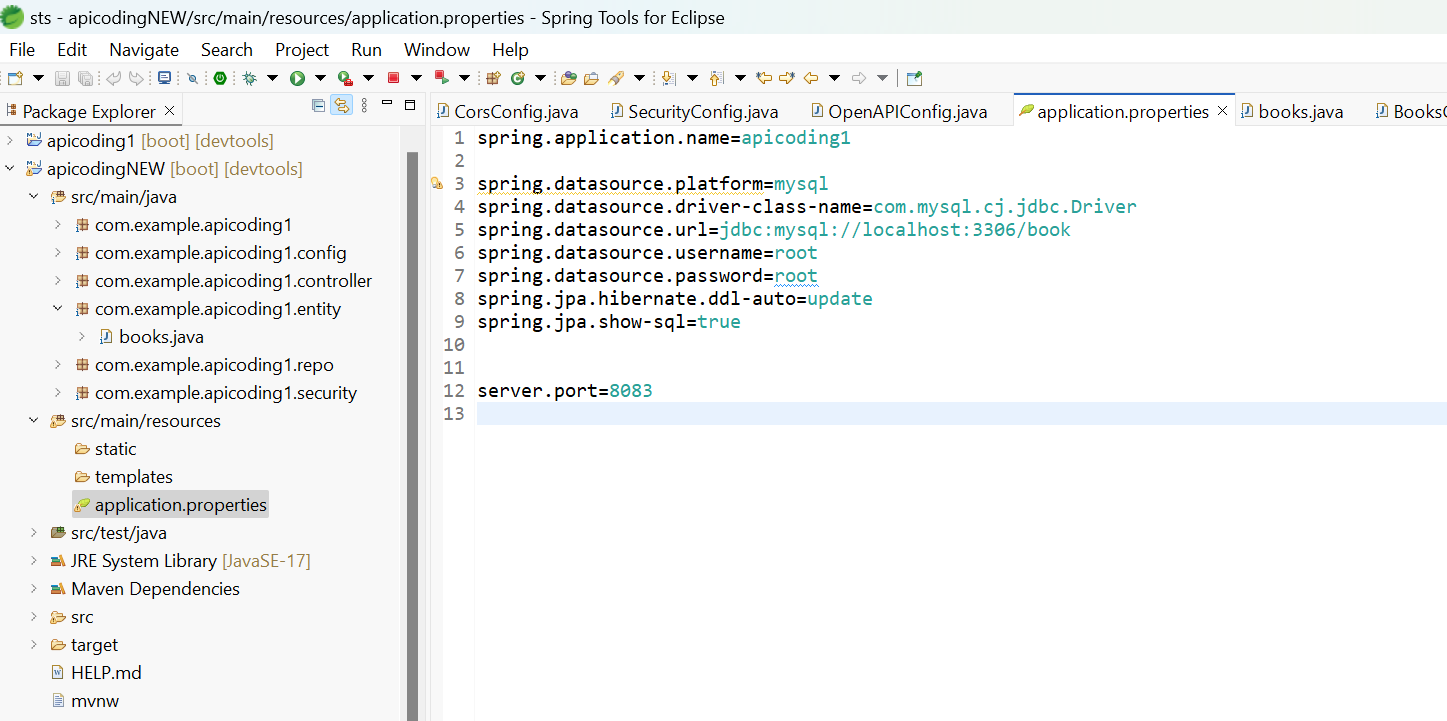


### Backend Features

* RESTful API Design: Standard HTTP methods and status codes
* JWT Authentication: Secure token-based authentication
* CORS Configuration: Cross-origin request handling
* Exception Handling: Global error handling with proper HTTP responses
* Data Validation: Input validation and sanitization
* Logging: Comprehensive logging for debugging and monitoring

### Database Configuration

#### Application Properties



### Database Features

* Primary Keys: Unique identifiers for all entities
* Foreign Keys: Proper relationships between tables
* Indexes: Optimized queries for better performance
* Constraints: Data integrity and validation
* Timestamps: Audit trails for data changes

## Frontend Implementation

### Component Architecture

#### 1. App Component (app.ts)

Purpose: Root application component  
- Application initialization and routing setup

#### 2. Authentication Components

* Login Component: User authentication interface
* Auth Guard: Route protection
* JWT Interceptor: Automatic token management

#### 3. Book Management Components

* Book List: Display all books with search functionality
* Book Add: Create new book entries
* Book Update: Edit existing book information
* Book Detail: View detailed book information

#### 4. Services

* Auth Service: Authentication and token management
* Book Service: All book-related API operations

### Key Frontend Features

* Responsive Design: Works on all devices
* Real-time Search: Instant book filtering
* Form Validation: Client-side input validation
* Error Handling: User-friendly error messages
* Loading States: Visual feedback during operations

## Security Implementation

### JWT Authentication Flow

1. User submits credentials to /auth/login
2. Backend validates credentials
3. Backend generates JWT token
4. Frontend stores token in localStorage
5. All API requests include token in Authorization header
6. Backend validates token for each protected endpoint

### Security Features

* Password Encryption: BCrypt hashing
* Token Expiration: 24-hour token validity
* CORS Configuration: Controlled cross-origin access
* Input Validation: Server-side data validation
* SQL Injection Prevention: JPA/Hibernate protection

## UI/UX Design

### Design Philosophy

* Minimalist: Clean, professional interface
* Responsive: Mobile-first design approach
* Accessible: High contrast and readable fonts
* Intuitive: Easy navigation and clear actions

## Installation and Setup

### Prerequisites

* Java: JDK 17 or higher
* Node.js: Version 18 or higher
* MySQL: Version 8.x
* Maven: Version 3.x
* Angular CLI: Latest version

### Access the Application

* Frontend: http://localhost:4200
* Backend API: http://localhost:8083
* Database: localhost:3306/book

## Testing and Troubleshooting

### Quick Testing Guide

### Common Issues

1. CORS Errors: Check backend CORS configuration
2. Authentication Issues: Verify JWT token and credentials
3. Database Connection: Check MySQL service and credentials
4. Port Conflicts: Ensure ports 4200 and 8083 are available

## Future Enhancements

### Planned Features

* Advanced Search: Filter by publication year range
* Book Categories: Genre-based organization
* User Management: Multiple user support
* Export/Import: CSV functionality

## Conclusion

This Book Management System demonstrates a complete full-stack application with:  
- Modern Frontend: Angular with TypeScript  
- Robust Backend: Spring Boot with MySQL  
- Security: JWT authentication  
- Professional UI: Responsive design  
- Best Practices: Clean code and architecture

The system provides a solid foundation for book collection management with room for future enhancements and scalability.

Document Version: 2.0  
Last Updated: July 11, 2025  
Project Status: Production Ready  
Maintained By: Yash Jadhav