|  |
| --- |
| **A**  **PROJECT REPORT ON** |
|  |
|  |
| Library Management System |
|  |
|  |
| SUBMITTED IN FULFILLMENT OF  **DIPLOMA IN ADVANCED COMPUTING (PG-DAC)** |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| **BY**  **Shreyas Khodre**  **Prajwal Nirmal**  **Nimish Jais**  **Janhavi Mulane**  **Prajakta Padaval** |
|  |
|  |
| **UNDER THE GUIDENCE OF**  **Mrs. Lalita Shinde** |
|  |
|  |
|  |
| **AT**  **SUNBEAM INSTITUTE OF INFORMATION TECHNOLOGY, PUNE** |

|  |
| --- |
|  |
| **SUNBEAM INSTITUTE OF INFORMATION TECHNOLOGY,**  **PUNE.** |
|  |
|  |
|  |
|  |
| **CERTIFICATE** |
| This is to certify that the project |
|  |
| **Library Management System**  Has been submitted by  **Shreyas Khodre**  **Prajwal Nirmal**  **Nimish Jais**  **Janhavi Mulane**  **Prajakta Padaval**   |  |  | | --- | --- | | In fulfillment of the requirement for the Course of **PG Diploma in Advanced Computing (PG-DAC AUG2024)** as prescribed by The **CDAC** ACTS, PUNE. | | |  | | |  | | | Place: Pune | Date: 11-FEB-2025 | |  | | |
| **UNDER THE GUIDENCE OF**  **Mrs. Lalita Shinde** |
|  |

**ACKNOWLEDGEMENT**

A project usually falls short of its expectation unless aided and guided by the right persons at the right time. We avail this opportunity to express our deep sense of gratitude towards Mr. Nitin Kudale (Center Coordinator, SIIT, Pune), Mr. Yogesh Kolhe (Course Coordinator, SIIT ,Pune) and our project guide Mrs. Lalita Shinde ma’am(Mentor, SIIT , Pune) .

We are deeply indebted and grateful to them for their guidance, encouragement and deep concern for our project. Without their critical evaluation and suggestions at every stage of the project, this project could never have reached its present form .

Last but not the least we thank the entire faculty and the staff members of Sunbeam Institute of Information Technology, Pune for their support.

**INDEX**

|  |  |  |
| --- | --- | --- |
|  | **INTRODUCTION** |  |
|  | 1.1 Introduction | 6 |
|  | **REQUIREMENTS** |  |
|  | 2. Use case diagram | 7 |
|  | **2.1 Customer flow for library management System** | 8 |
|  | 2.1.1 Home Page |  |
|  | 2.1.2 Book Selection |  |
|  | 2.1.3 Book Borrowing & Reservation |  |
|  | 2.1.4 Cart Interaction |  |
|  | 2.1.5 Borrowing Confirmation Page |  |
|  | 2.1.6 Profile Page |  |
|  | **2.2 Admin flow for library management system** | 10 |
|  | 2.2.1 Admin Dashboard |  |
|  | 2.2.2 Book Management |  |
|  | 2.2.3 User Management |  |
|  | 2.2.4 Borrowing & Reservation Management |  |
|  | 2.2.5 Reports & Analysis |  |
|  | 2.2.6 Admin Profile & Settings |  |
|  | **NON-FUNCTIONAL REQUIREMENTS** | 12 |
|  | 3.1 Interface , Performance , Constraints |  |
|  | 3.2 Software Requirements |  |
|  | 3.3 Hardware Requirements |  |
|  | **DATABASE DESIGN** | 14 |
|  | **CODING STANDARDS** | 17 |
|  | **E-R DIAGRAM** | 19 |
|  | **USER INTERFACE** | 20 |
|  | **REFERENCES** | 29 |

**LIST OF TABLES**

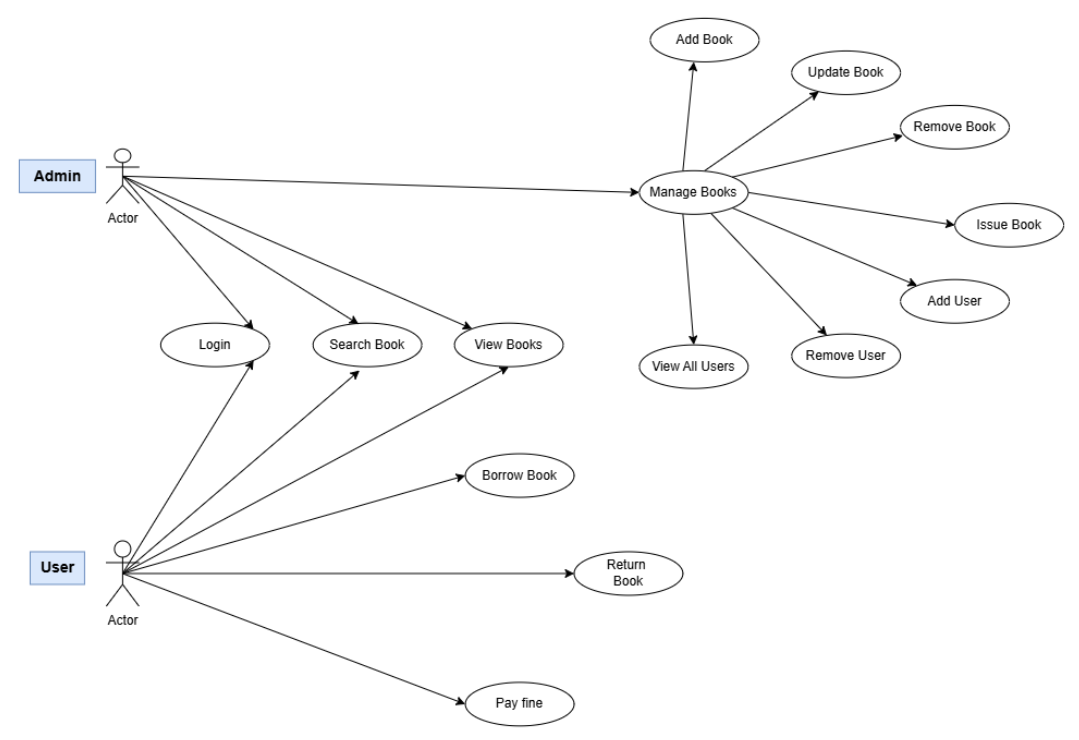
|  |  |  |
| --- | --- | --- |
| **Section** | **Table Title** | **Page** |
| 1 | Users Table | 14 |
| 2 | Books table | 14 |
| 3 | Reservations Table | 14 |
| 4 | Fines Table | 15 |
| 5 | Borrowing\_records Table | 15 |
| 6 | Transactions Table | 15 |
| 7 | Logs table | 16 |
| 8 | Books\_suggestions | 16 |
| 9 | Coding Standards | 16 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Section** | **Figure Title** | **Page** |
| 1 | Use case Diagram | 7 |
| 2 | E-R Diagram | 19 |

**1.INTRODUCTION TO PROJECT-**

The Library Management System is a comprehensive web application designed to efficiently manage and organize library resources. This system provides a user-friendly interface for librarians, students, and administrators to facilitate book cataloging, borrowing, returning, and tracking library assets. It streamlines operations by automating key tasks such as inventory management, member registration, and overdue book tracking. The system ensures seamless communication between users and administrators while enhancing the overall efficiency of library management.

**2. REQUIREMENTS :**

1. Use Case Diagram

**2.1 Customer Flow for Library Management System**

**2.1.1 Home Page**

**Objective:** Display a list of available books and resources.  
**Features:**

* View a list of books, e-books, and other resources.
* Search and filter books based on title, author, genre, or availability.

**2.1.2 Book Selection**

**Objective:** Select a book to view its details.  
**Features:**

* Click on a book to view details such as author, summary, availability, and reviews.
* Check if the book is available for borrowing or reservation.

**2.1.3 Book Borrowing & Reservation**

**Objective:** Allow customers to borrow or reserve books.  
**Features:**

* If the book is available:
  + Borrow the book (if eligible and logged in).
  + Choose a borrowing duration.
* If the book is not available:
  + Reserve the book for future borrowing.

**2.1.4 Cart Interaction (For Multiple Borrowing Requests)**

**Objective:** Manage borrowing requests before finalizing.  
**Features:**

* If Logged In:
  + Navigate to the borrowing cart to review selected books.
* If Not Logged In:
  + Display a login prompt.
  + If not registered, prompt registration before borrowing.

**2.1.5 Borrowing Confirmation Page**

**Objective:** Finalize the borrowing process.  
**Features:**

* View selected books for borrowing.
* Choose a pickup location (if applicable).
* Confirm borrowing or reservation.
* Redirect to the home page after confirming the borrowing request.

**2.1.6 Profile Page**

**Objective:** Manage user profile, borrowed books, and history.  
**Features:**

* If Not Logged In:
  + Display a sign-in prompt.
* If Logged In:
  + View and update profile details.
  + View borrowed books, due dates, and past borrowing history.
  + Renew books (if applicable).
  + Submit book reviews.
  + Logout.

**2.2 Admin Flow for Library Management System**

**2.2.1 Admin Dashboard**

**Objective:** Provide an overview of library operations.  
**Features:**

* View total books, borrowed books, reservations, and registered users.
* Search and filter books, users, and borrowing records.
* View notifications for overdue books and pending reservations.

**2.2.2 Book Management**

**Objective:** Manage the library's book collection.  
**Features:**

* Add new books with details like title, author, genre, and availability.
* Edit or update book information.
* Remove books from the system.
* Manage book categories and genres.

**2.2.3 User Management**

**Objective:** Manage library members.  
**Features:**

* View a list of registered users.
* Search and filter users by name, email, or membership status.
* Edit user details (e.g., contact info, membership status).
* Activate or deactivate user accounts.

**2.2.4 Borrowing & Reservation Management**

**Objective:** Oversee book borrowing and reservations.  
**Features:**

* View and approve/reject book reservations.
* Track currently borrowed books and their due dates.
* Mark books as returned and update availability status.
* Send reminders for overdue books.

**2.2.5 Reports & Analytics**

**Objective:** Monitor library activity and generate reports.  
**Features:**

* Generate reports on borrowed books, overdue books, and user activity.
* Track the most borrowed books and popular categories.
* Export reports in different formats (PDF, Excel).

**2.2.6 Admin Profile & Settings**

**Objective:** Manage admin account and system settings.  
**Features:**

* Update admin profile details.
* Change password and security settings.
* Configure library settings (e.g., borrowing limits, late fees).
* Logout.

**3. Non-Functional Requirements**

**3.1 Interface**

* User interfaces must be intuitive and user-friendly for both library users and administrators.
* Detailed UI designs are provided in Appendix B.

**3.2 Performance**

* **Number of Concurrent Users:** The system should handle at least 1000 transactions/inquiries per second.
* **System Resilience:** The application should be resilient to temporary server failures and ensure data consistency.

**3.3 Constraints**

* The system should maintain performance standards of handling 1000 transactions/inquiries per second.
* The database should be optimized for fast retrieval of books, users, and borrowing records.

**3.4 Other Requirements**

**3.4.1 Hardware Interfaces**

**Minimum Requirements:**

* Processor: Intel Core i5 or higher (or AMD equivalent)
* RAM: 8 GB or more
* Storage: 512 GB SSD or larger

**3.4.2 Software Interfaces**

* **Operating Systems:** Windows 11, Ubuntu 22.04
* **Database:** MySQL
* **Server:** Embedded Tomcat
* **Browsers:** Compatible with modern web browsers (Chrome, Firefox, Edge

**3.4.3 System Design**

**Architecture**

* **Front-End:** Developed using React.js and Redux for state management.
* **Back-End:** Built with Spring Boot for server-side logic.
* **Database:** MySQL for storing user details, book inventory, and borrowing records.
* **Server:** Embedded Tomcat for hosting the application.

**4. DESIGN**

4.1 Database Design The following table structures depict the database design.

**1] Users Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | name | varchar | 50 | 0 |
| 0 | email | varchar | 25 | 0 |
| 0 | password | varchar | 10 | 0 |
| 0 | role | varchar | 10 | 0 |

**2] Books Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | title | varchar | 30 | 0 |
| 0 | author | varchar | 30 | 1 |
| 0 | publisher | varchar | 30 | 1 |
| 0 | isbn | varchar | 10 | 0 |
| 0 | genre | varchar | 20 | 0 |
| 0 | total\_copies | int | - | 0 |
| 0 | available\_copies | int | - | 0 |

**3] Reservations Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | user\_id | bigint | - | 0 |
| 0 | book\_id | bigint | - | 0 |
| 0 | reservation\_date | date | - | 0 |
| 0 | expiration\_date | date | - | 0 |
| 0 | status | varchar | 10 | 0 |

**4] Fines Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | user\_id | bigint | - | 0 |
| 0 | record\_id | bigint | - | 0 |
| 0 | amount | int | - | 0 |
| 0 | payment\_status | varchar | 10 | 1 |

**5] Borrowing\_records Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | user\_id | bigint | - | 0 |
| 0 | book\_id | bigint | - | 0 |
| 0 | borrow\_date | date | - | 0 |
| 0 | due\_date | date | - | 0 |
| 0 | return\_date | date | - | 0 |
| 0 | status | varchar | 10 | 0 |

**6] Transactions Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | user\_id | bigint | - | 0 |
| 0 | book\_id | bigint | - | 0 |
| 0 | transaction\_amount | int | - | 0 |
| 0 | payment\_method | varchar | 10 | 0 |
| 0 | transaction\_type | varchar | 10 | 0 |
| 0 | status | varchar | 10 | 0 |

**7] Logs Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | user\_id | bigint | - | 0 |
| 0 | action | varchar | 20 | 0 |
| 0 | timestamp | date | - | 0 |
| 0 | details | varchar | 50 | 0 |

**8] Book\_Suggestions Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key Type/Constraint** | **Column Name** | **Data Type** | **Length** | **Allow Null (1=Yes; 0=No)** |
| 3 | id | bigint | - | 0 |
| 0 | user\_id | bigint | - | 0 |
| 0 | title | varchar | 50 | 0 |
| 0 | author | varchar | 20 | 0 |
| 0 | timestamp | varchar | 10 | 0 |

**5. CODING STANDARDS IMPLEMENTED :**

Naming and Capitalization Below summarizes the naming recommendations for identifiers in Pascal casing is used mainly (i.e. capitalize first letter of each word) with camel casing (capitalize each word except for the first one) being used in certain circumstances.

|  |  |  |  |
| --- | --- | --- | --- |
| Identifier | Case | Examples | Additional Notes |
| Class | Pascal | User, Order,  UserController | Class names should be based on "objects" or "real things" and should generally be nouns. No‘\_’signs allowed. Do not use type prefixes like ‘C’ for class. |
| Method | Camel | signUp , signIn,  addReview | Methods should use verbs or verb phrases. |
| Parameter | Camel | firstName, lastName,  email, password | Use descriptive parameter names. Parameter names should be descriptive enough that the name of the parameter and its type can be used to determine its meaning in most scenarios. |
| Interfaces | Pascal with I prefix | UserRepository,  BookRepository | Do not use the “\_” sign |
| Annotations | Pascal | @SpringBootApllication | Use @ at start of annotation |
| DTOs | Camel | ApiResponseDTO | Used to transfer data between the processes |
| Exception Class | Pascal with “Exception” suffix | ResourceNotFoundException |  |

**Comments :**

• Comment each type, each non-public type member, and each region declaration.

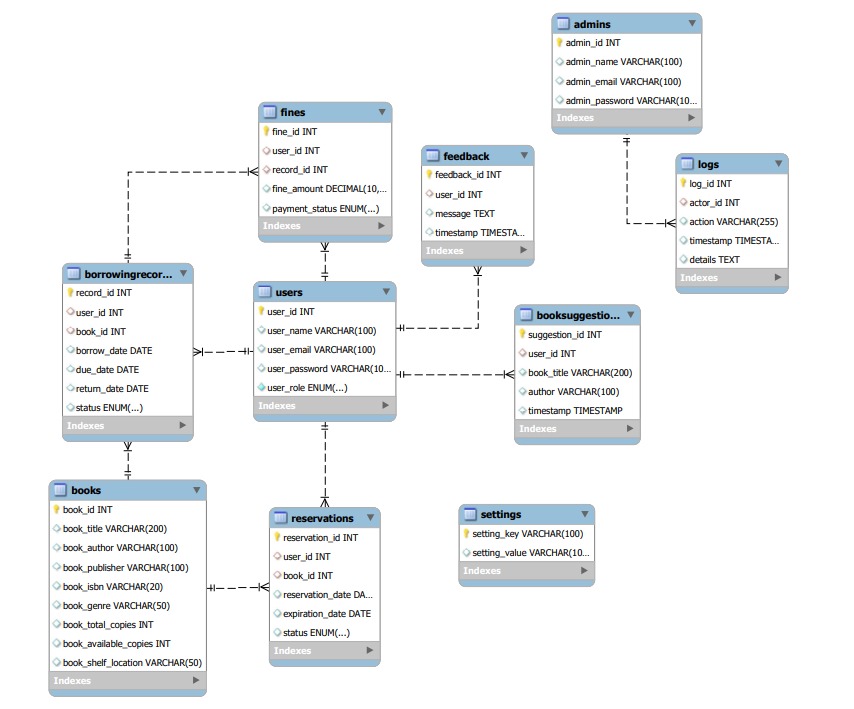
• Use end-line comments only on variable declaration lines. End-line comments are comments that follow code on a single line.

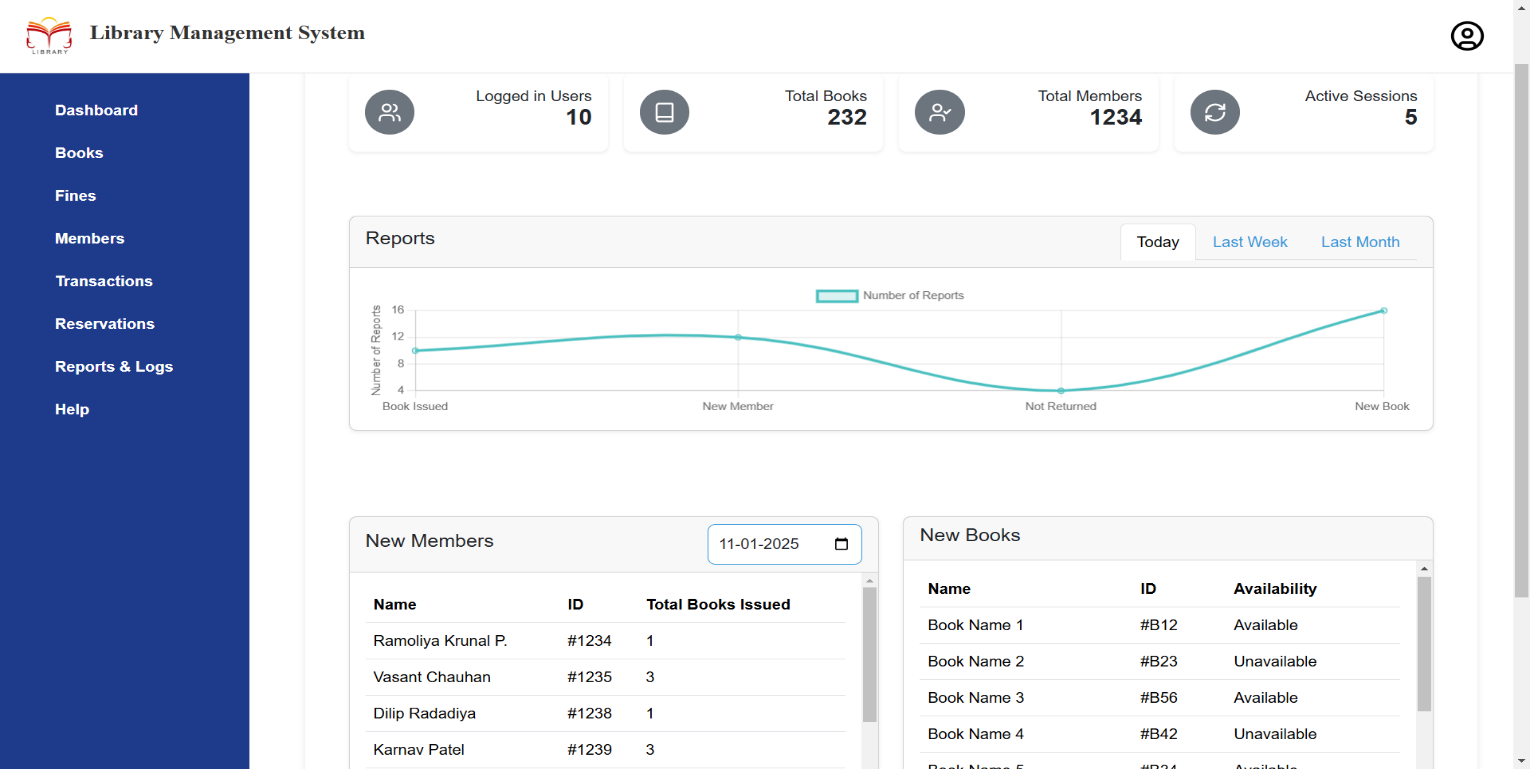
• Separate comments from comment delimiters (apostrophe) or//with one space.

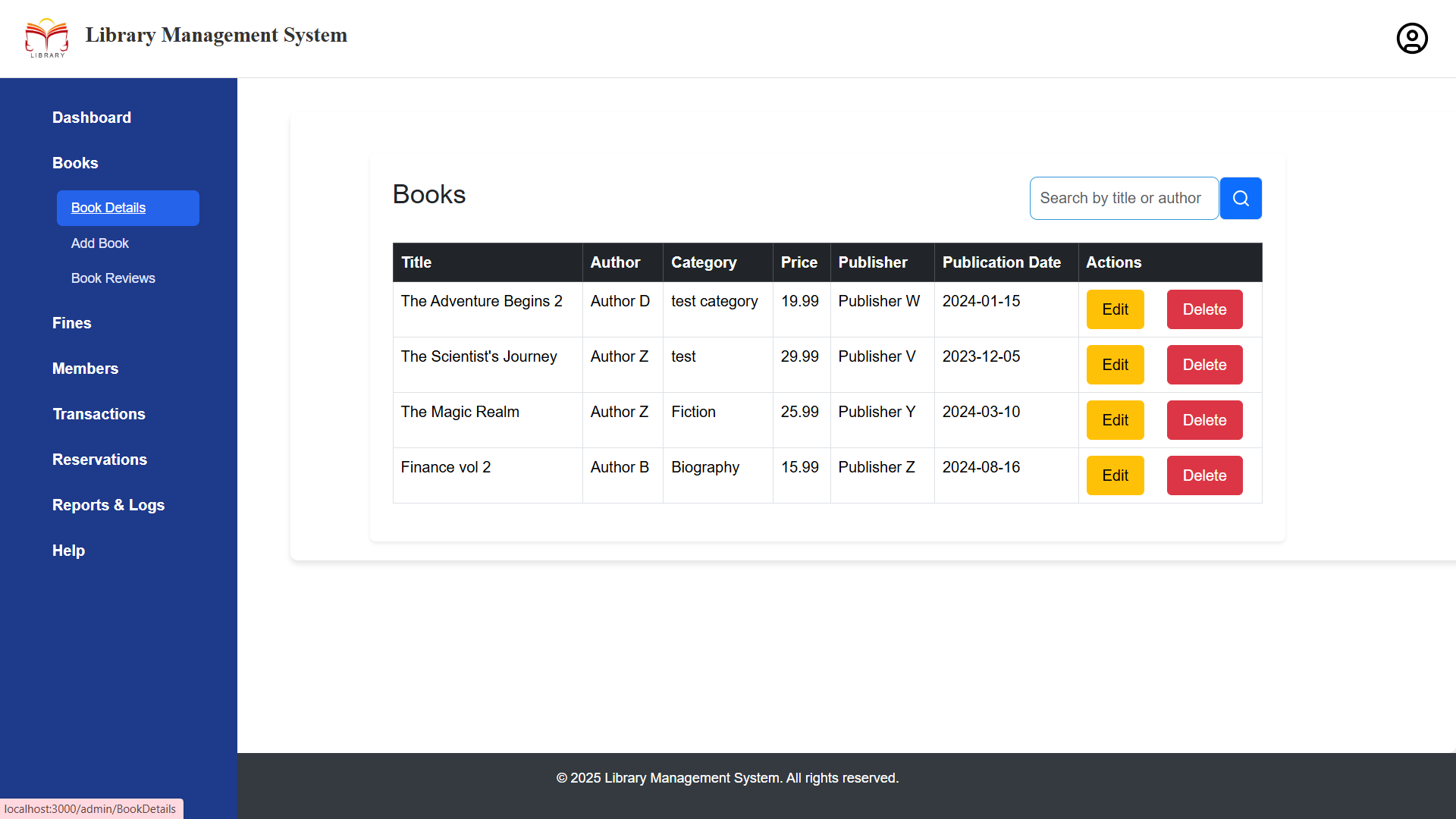
• Begin the comment text with an uppercase letter.

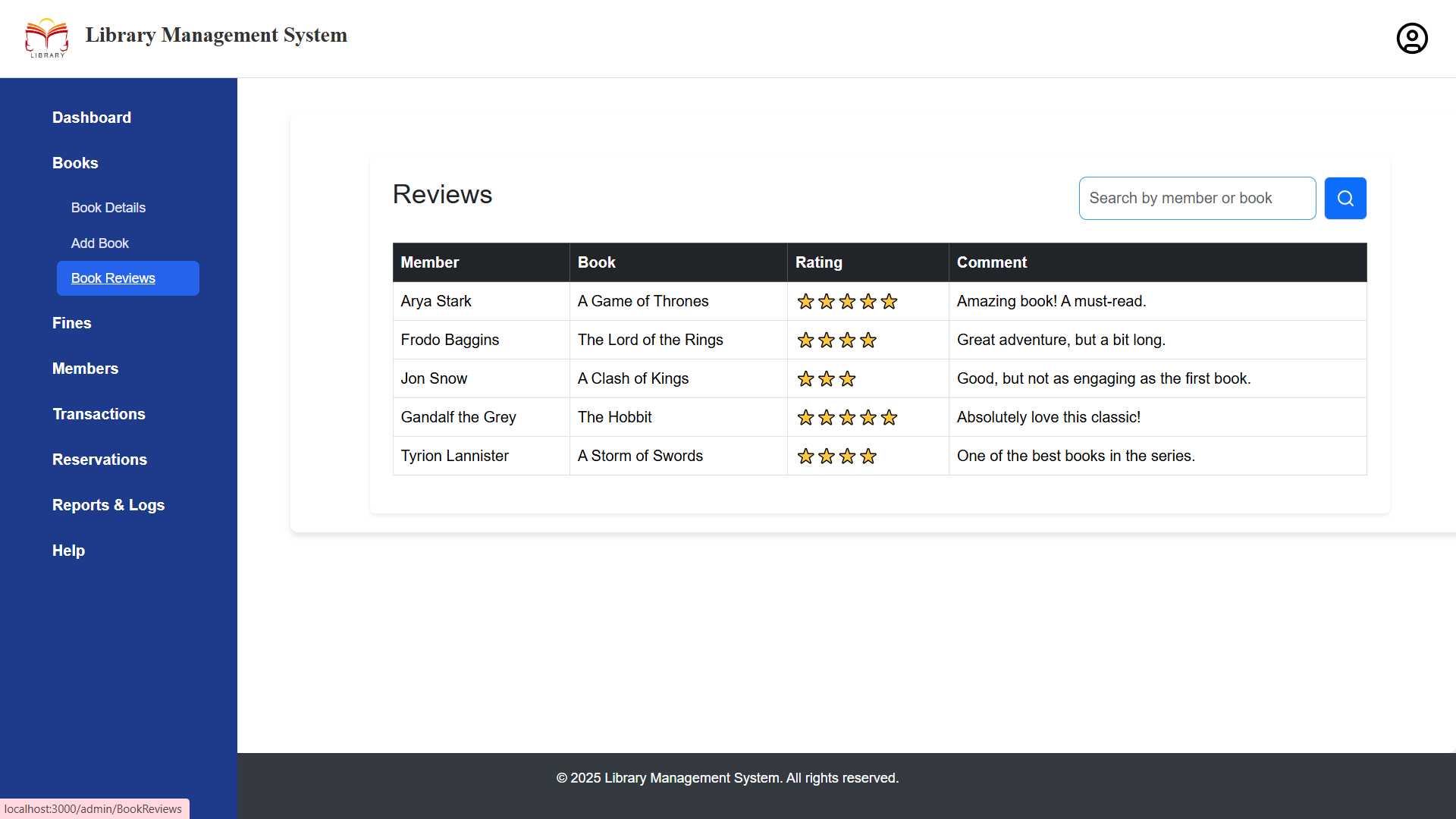
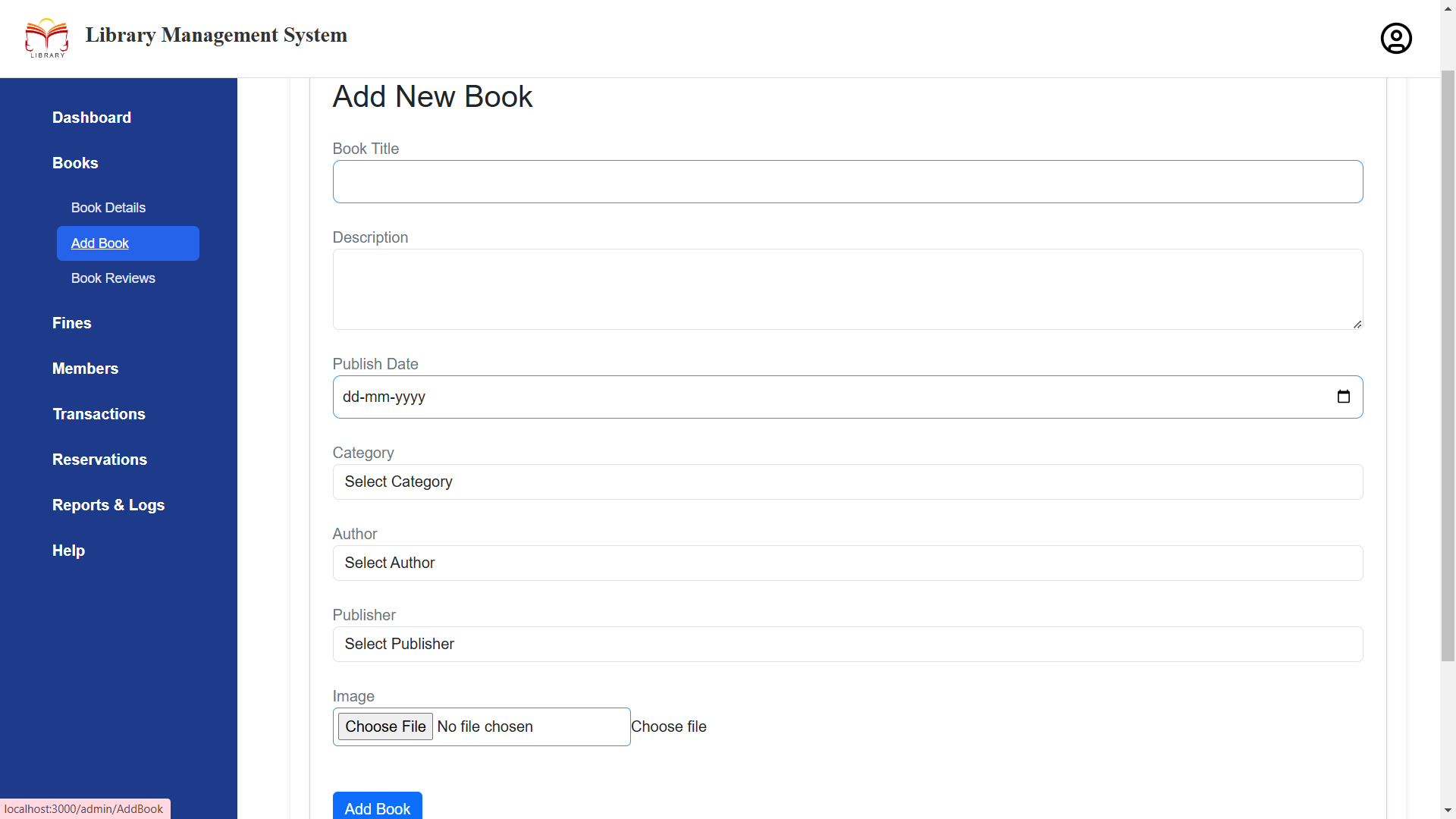
• End the comment with a period.

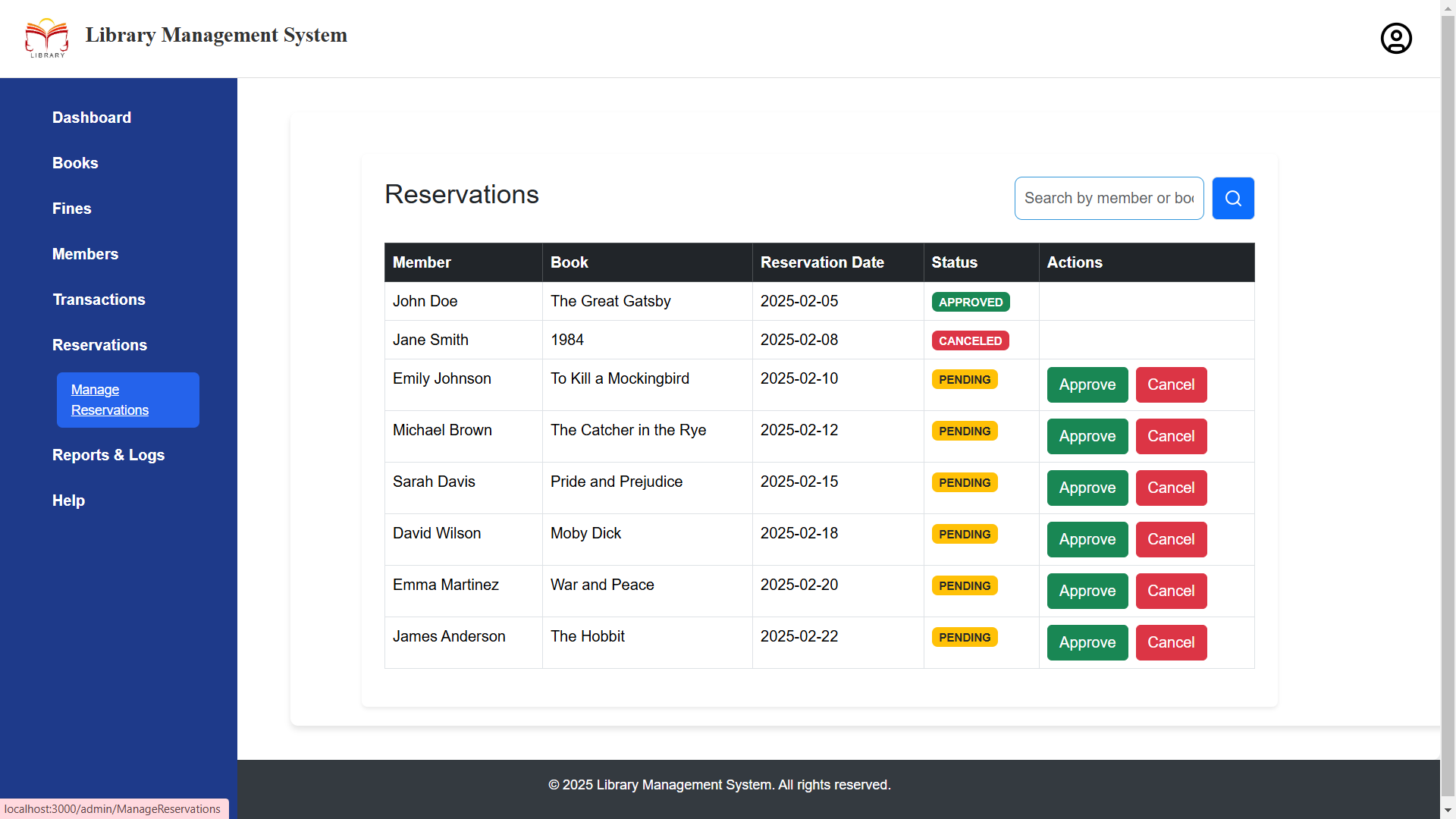
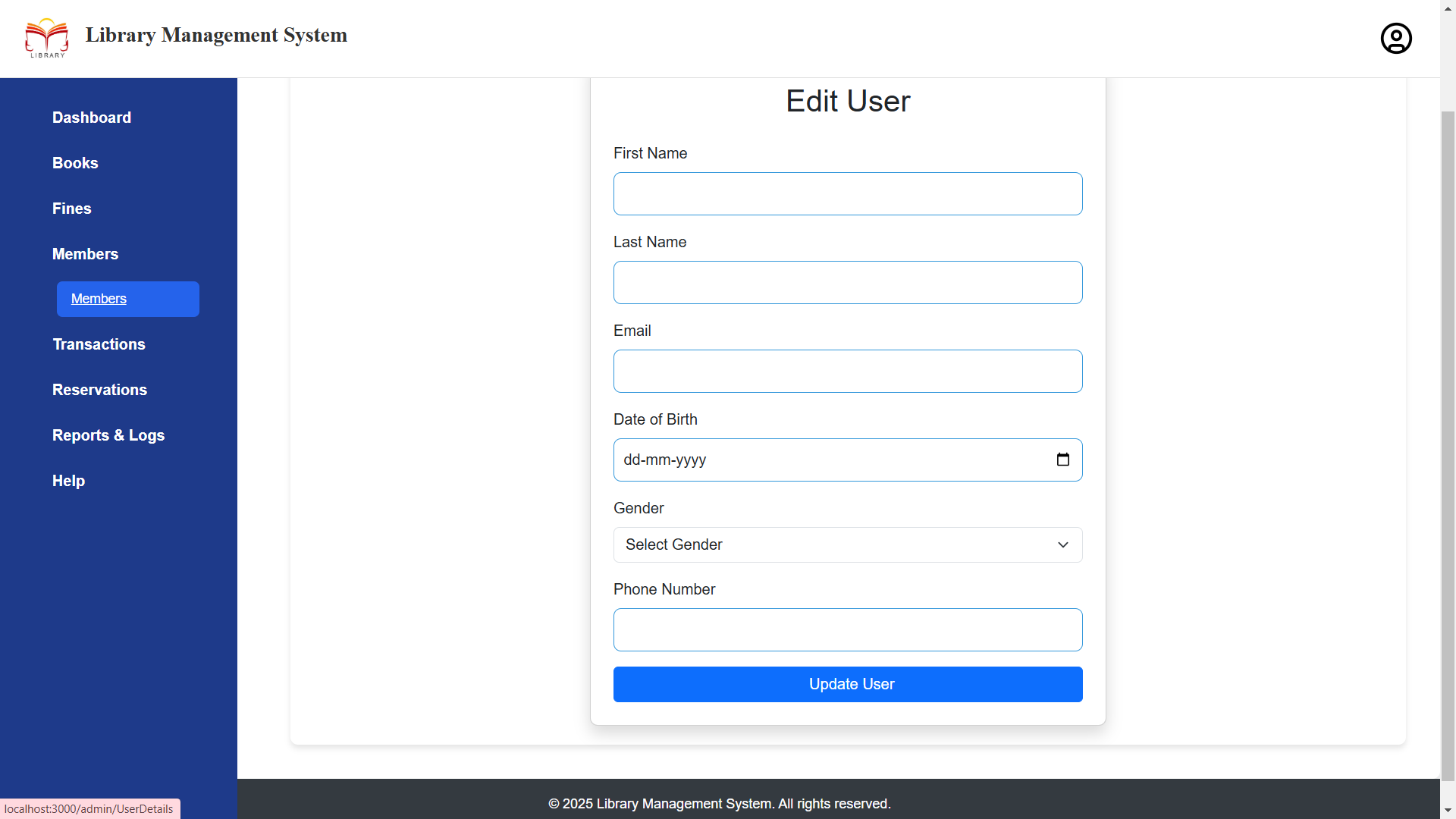
• Explain the code; do not repeat it.

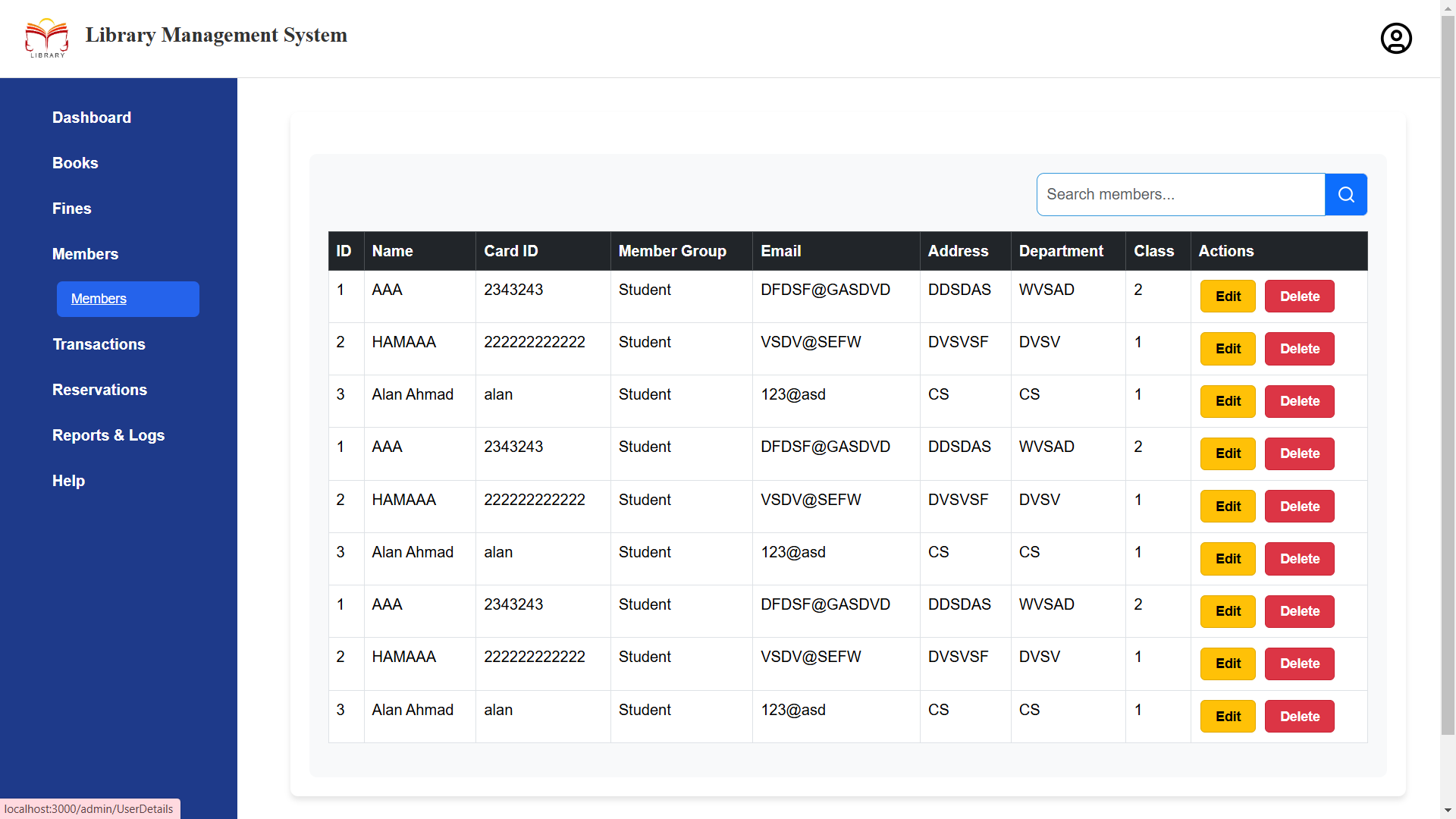
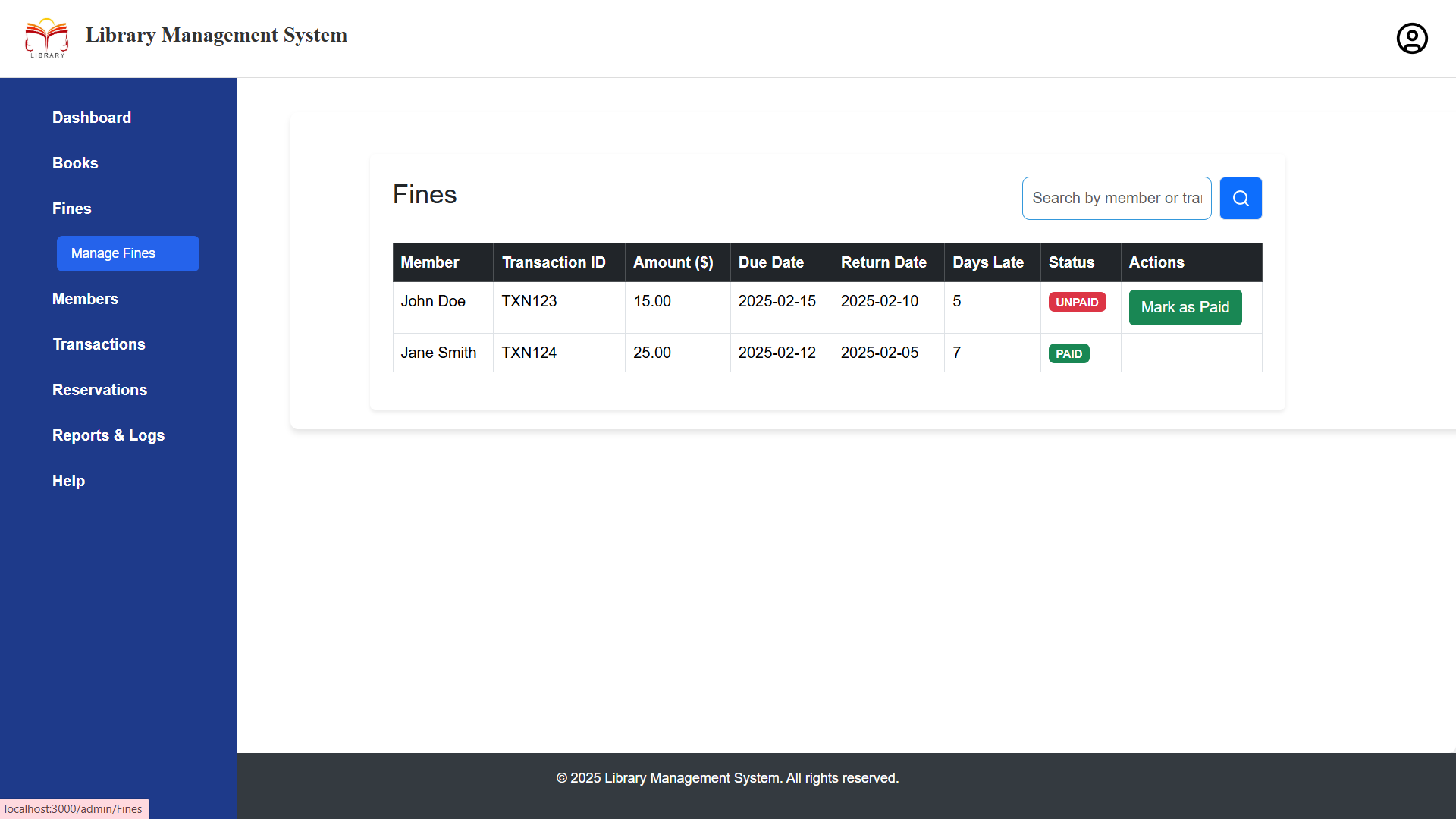
**6 .E-R Diagram :**

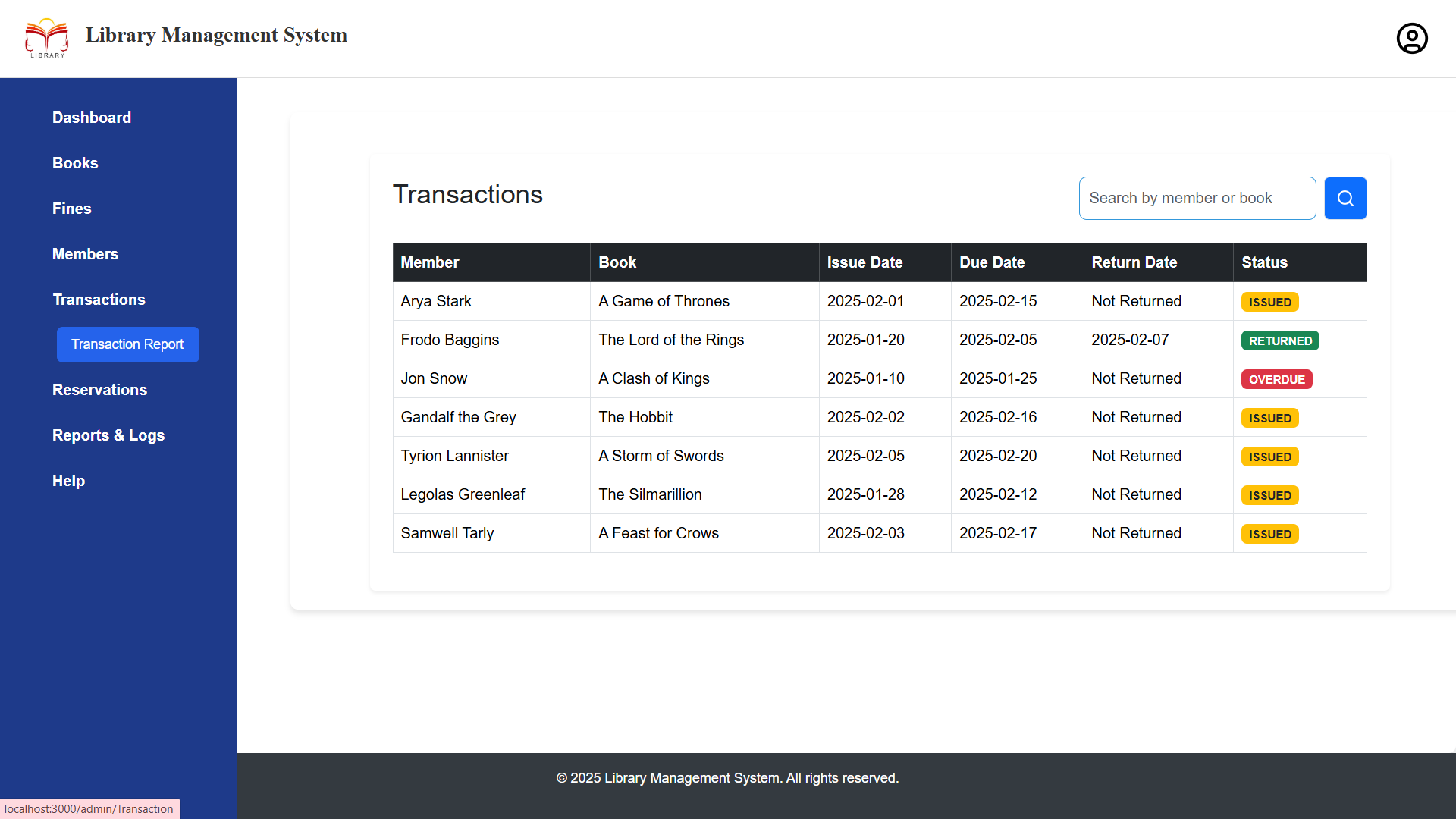
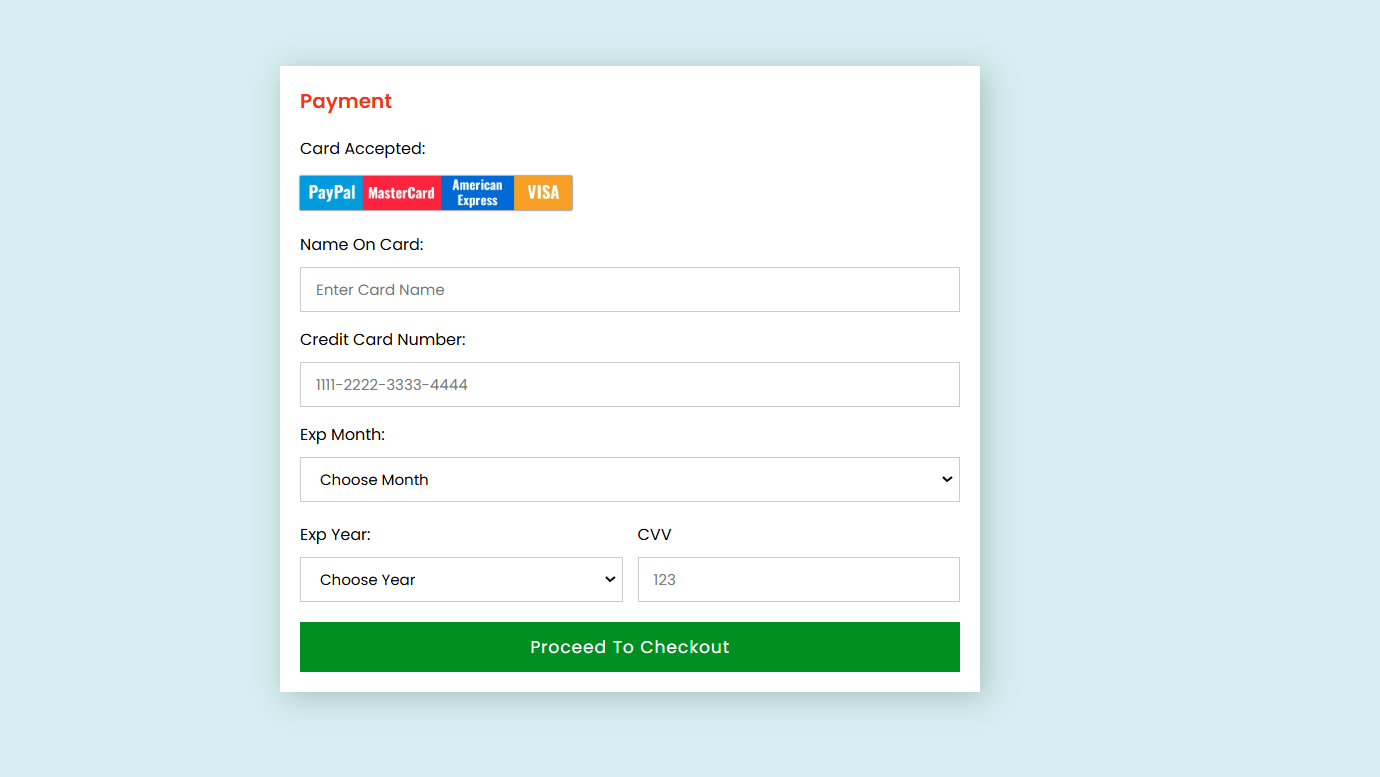
**7. USER INTERFACE :**

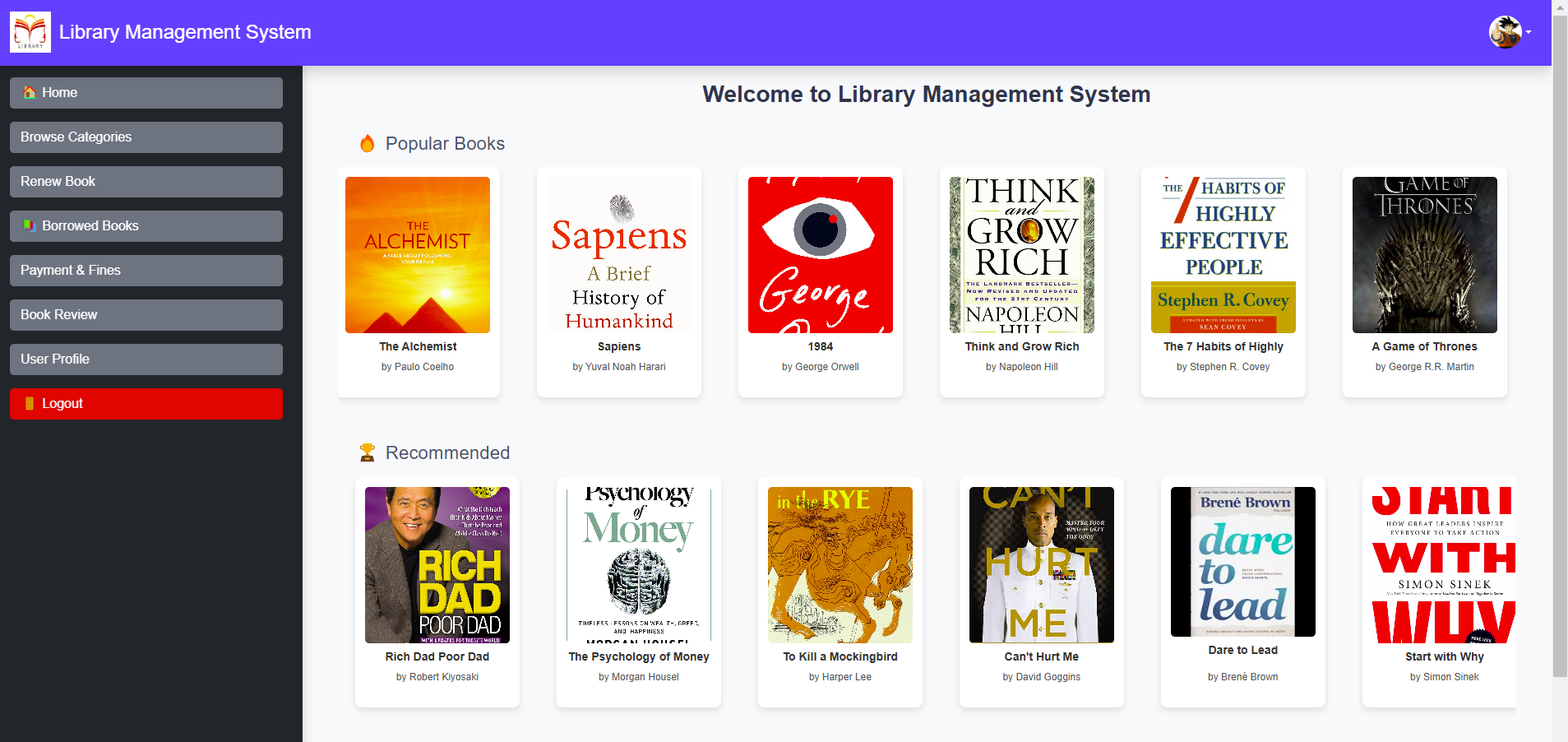
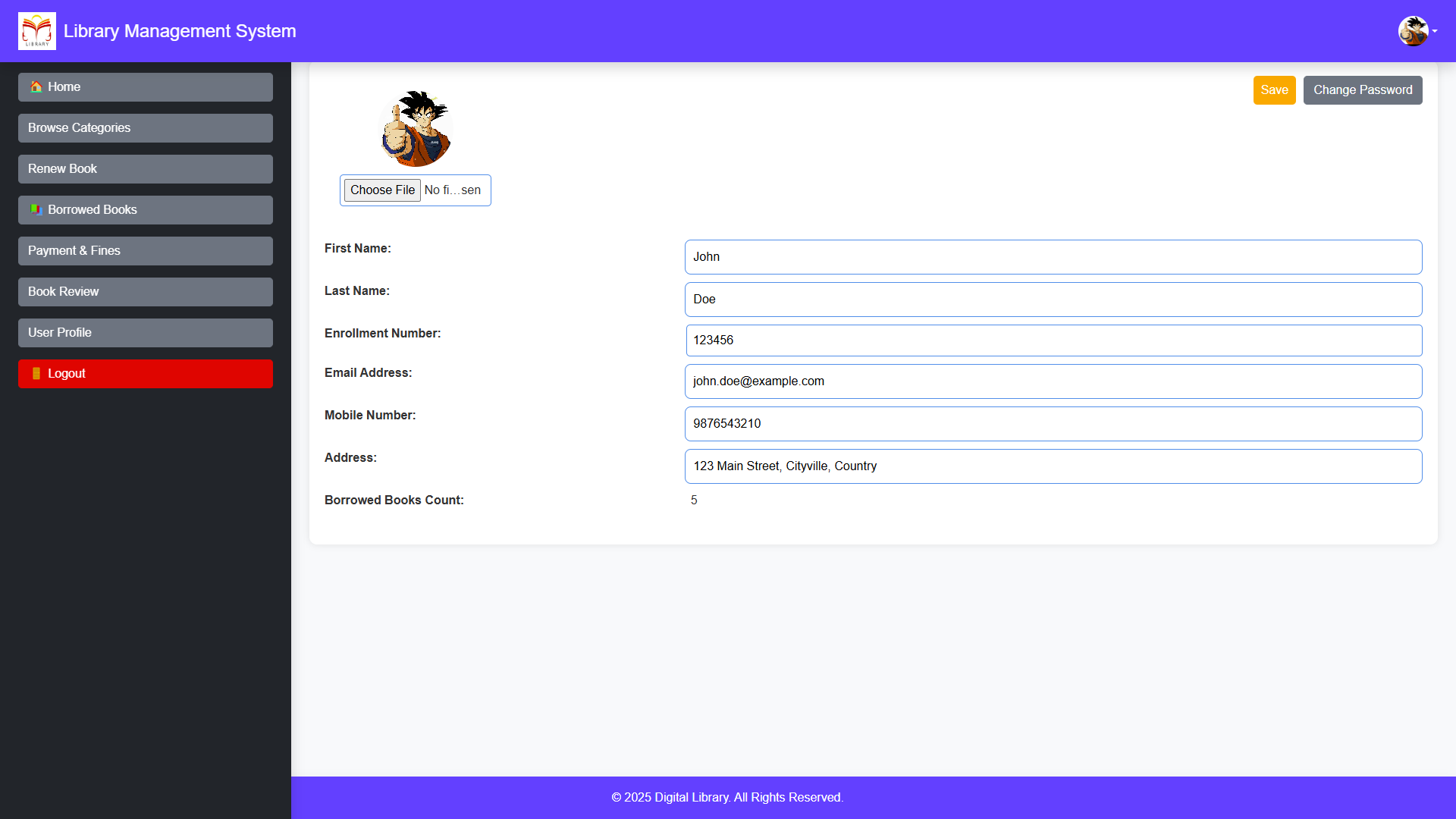


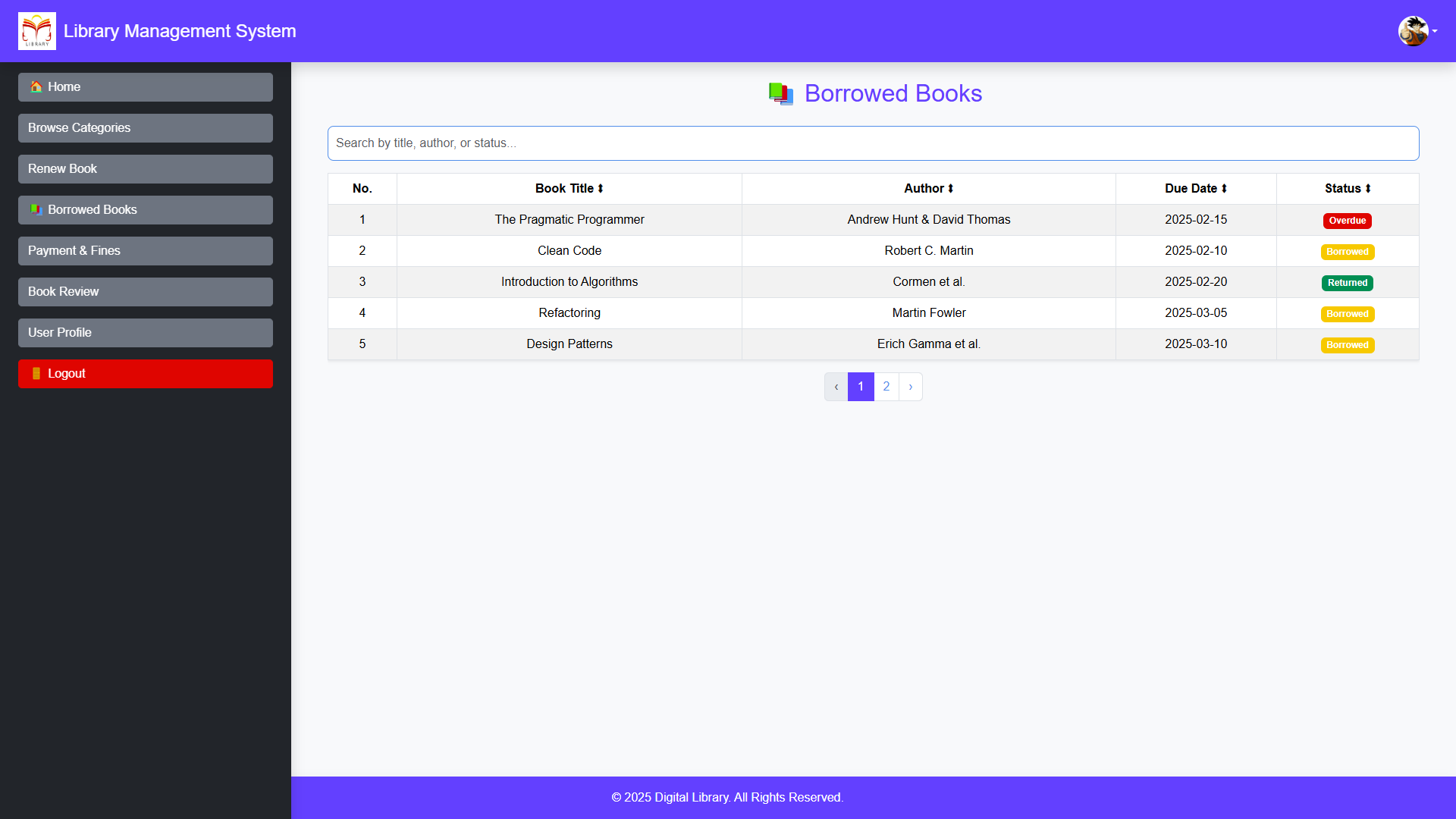
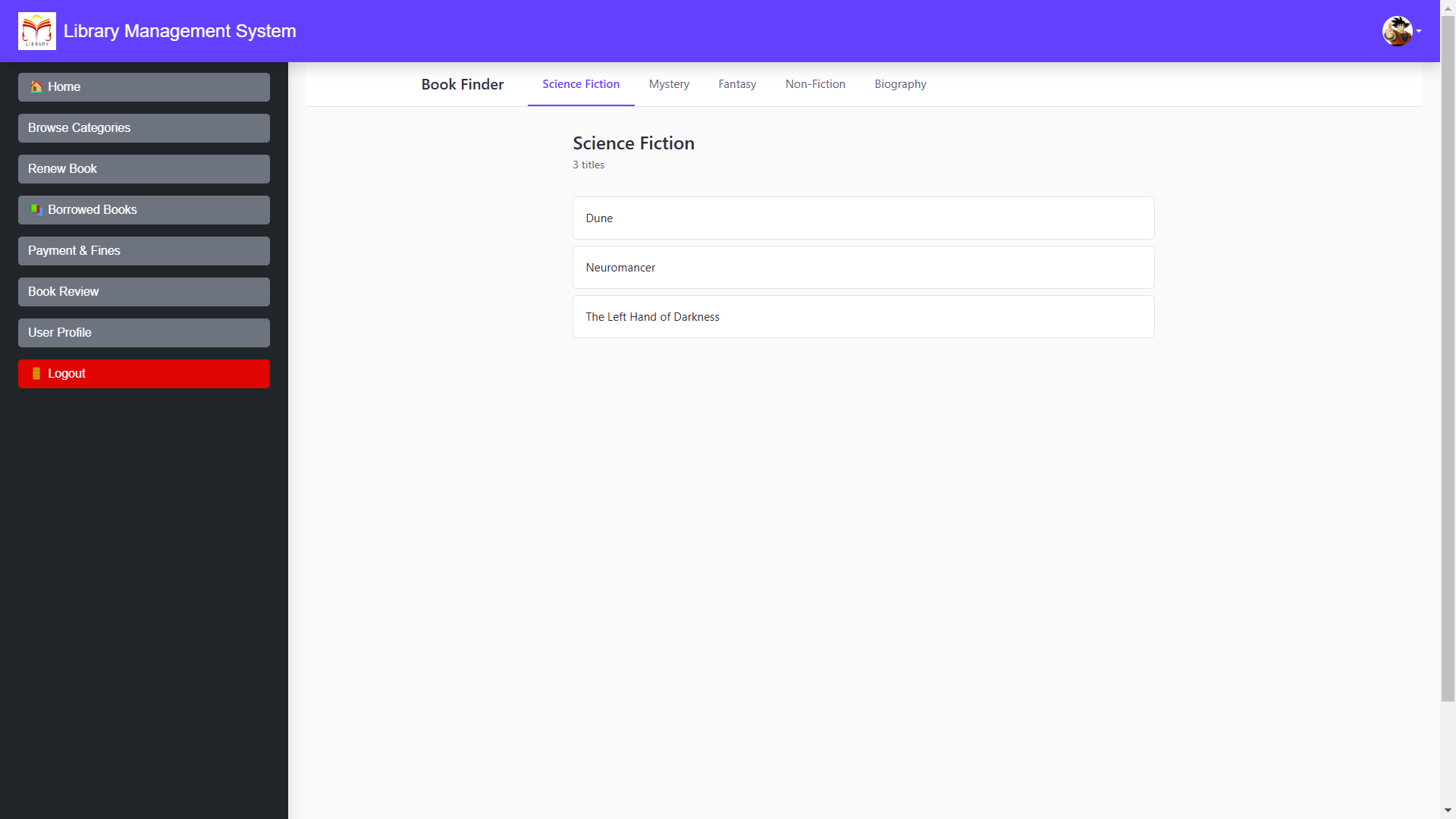


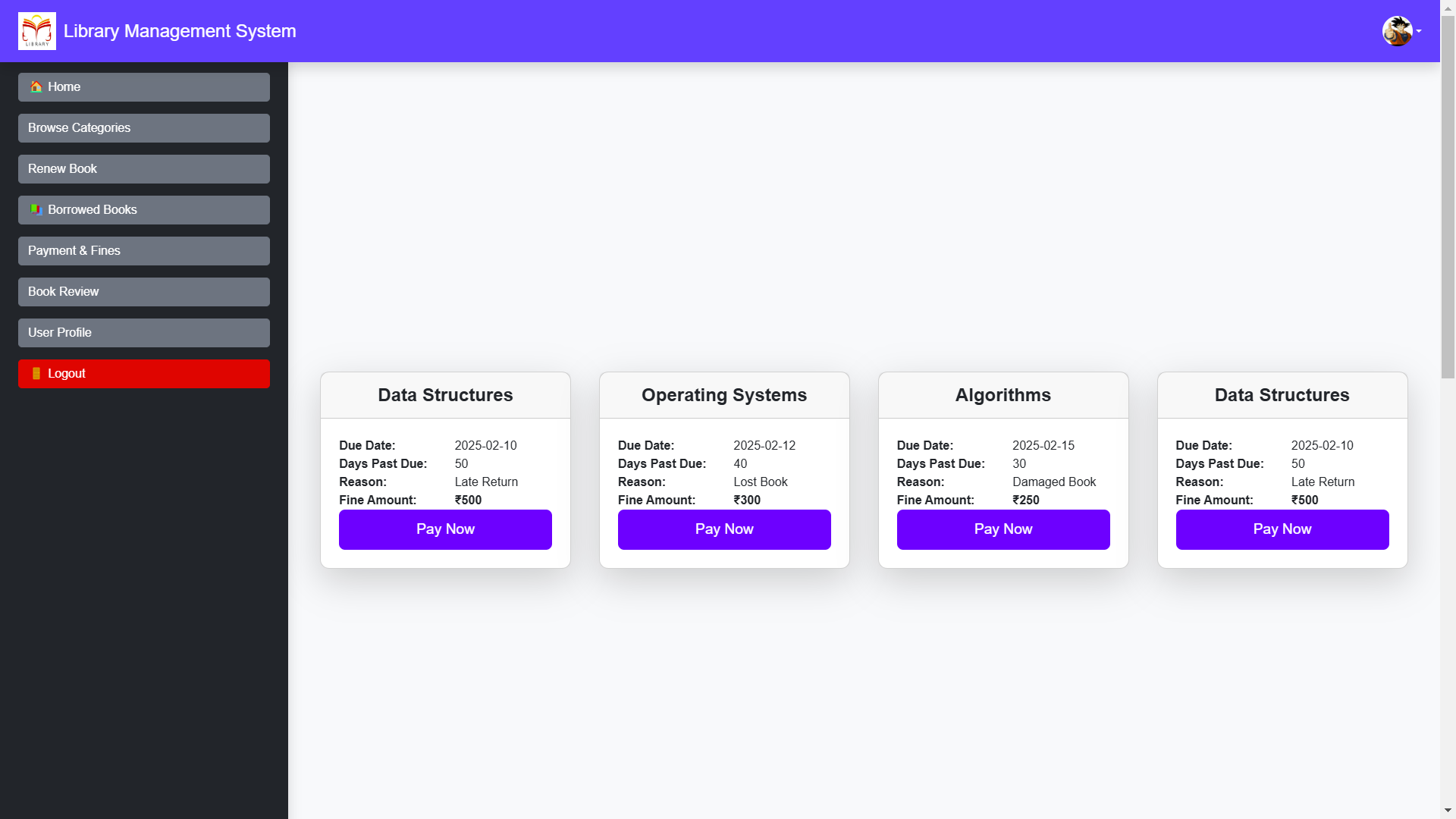
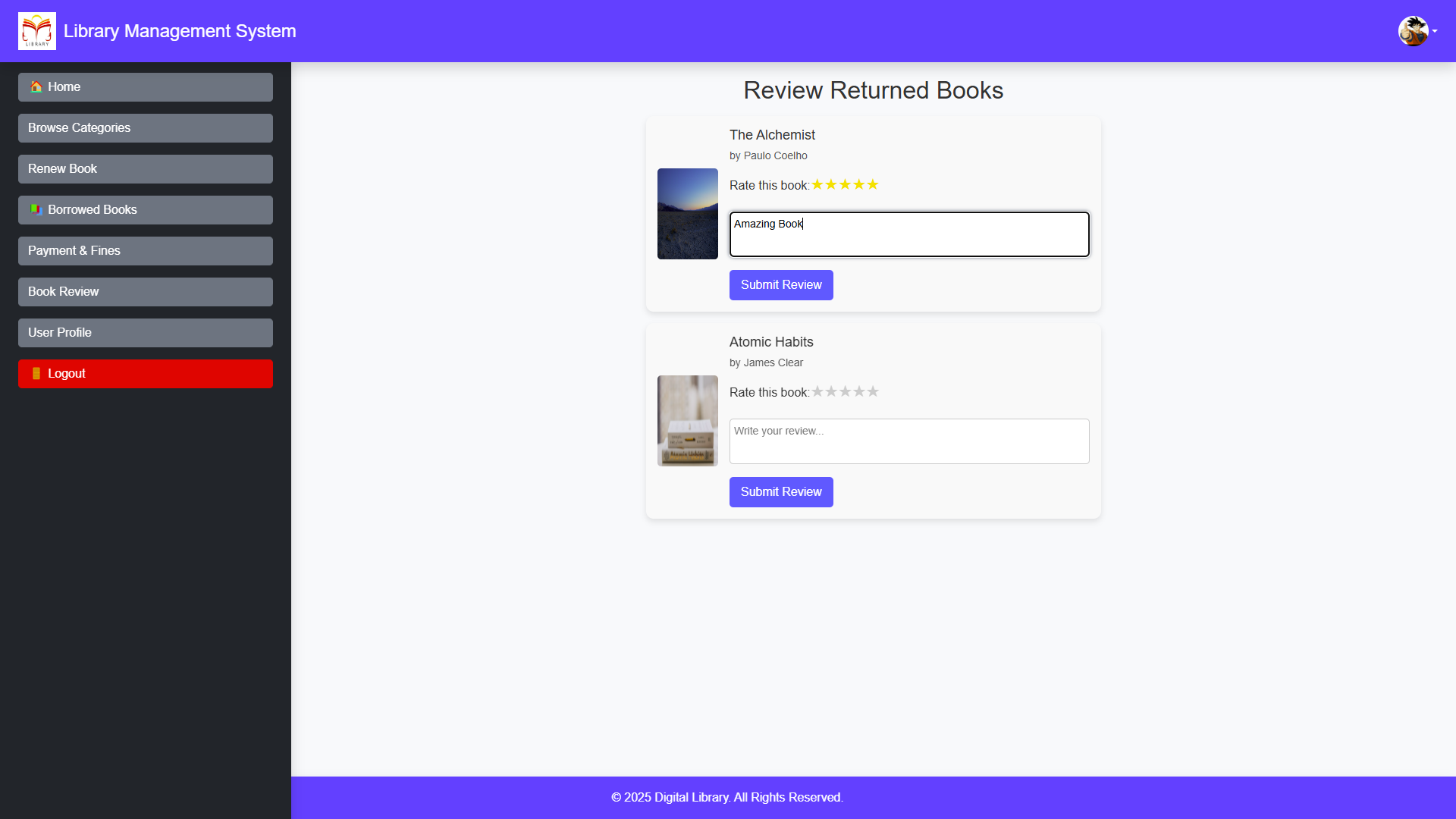


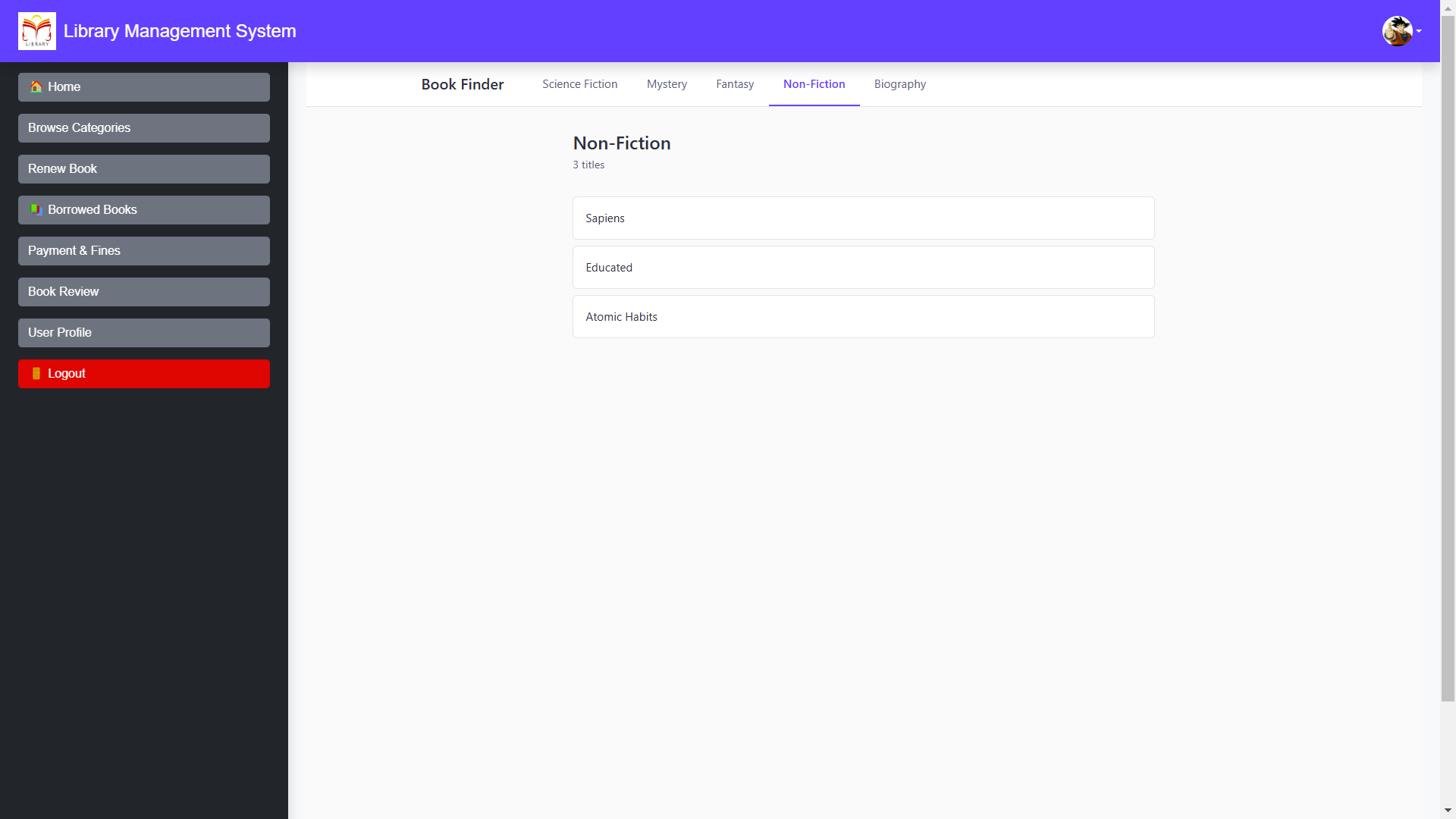
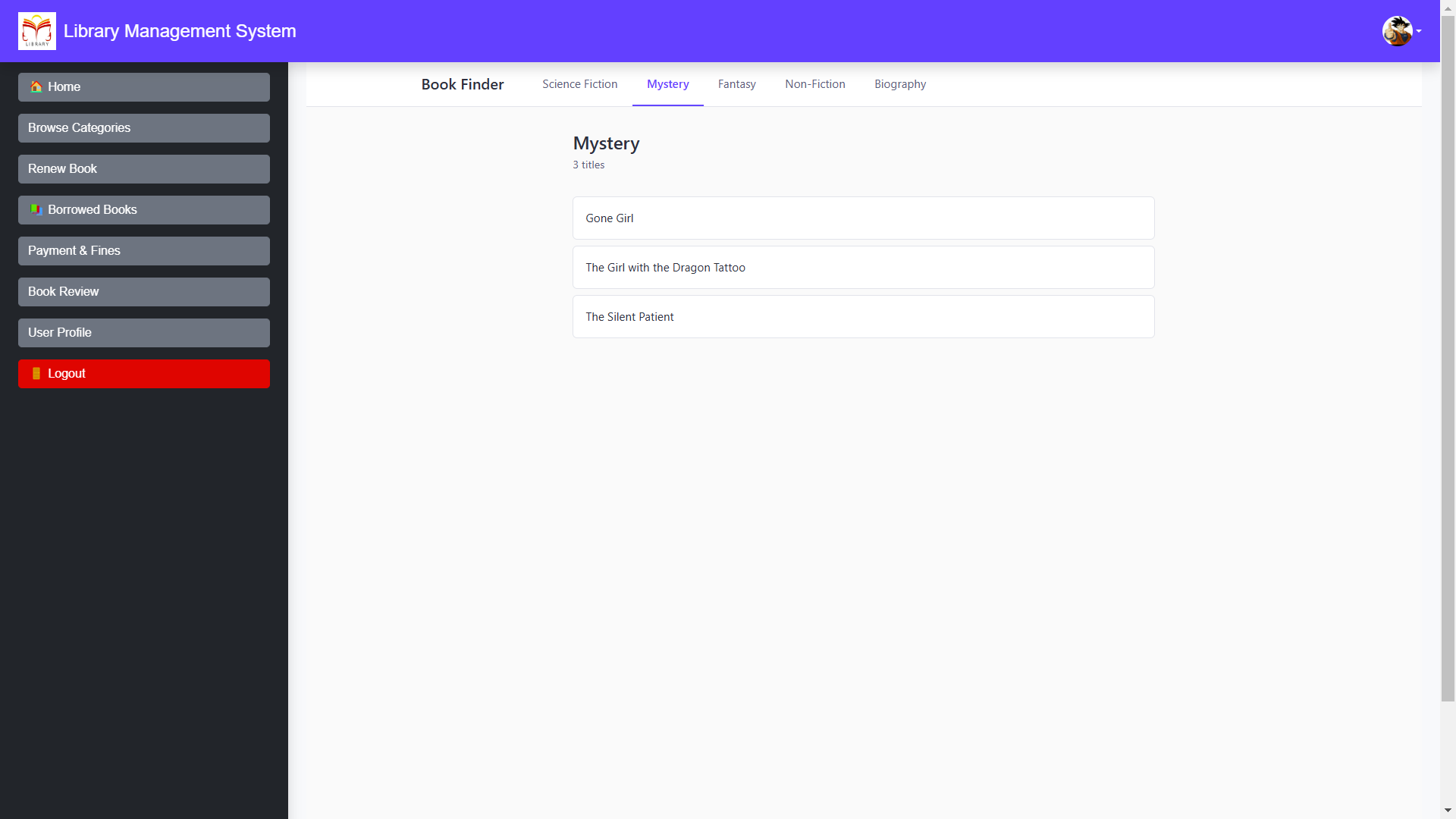












**8. REFERENCES :**

1. Spring Boot Documentation URL: <https://spring.io/projects/spring-boot>

2. React.js Documentation URL: <https://reactjs.org/docs/getting-started.html>

3. Redux Documentation URL: <https://redux.js.org>

4. Java Programming Language URL: <https://www.oracle.com/java/>

5. MySQL Workbench Documentation URL: https://dev.mysql.com/doc/workbench/en/ 6. Swagger Documentation for Spring Boot URL: https://springdoc.org/

7. MDN Web Docs URL: <https://developer.mozilla.org/>

8. React Redux Integration Guide URL: https://react-redux.js.org/