CS3332 - Library Management System

Presented by:

Group 5 - IHAB
Nguyen Viet Thien Quoc 1670020
Le Van Hai 1669834
Nguyen Tien Dat 1669692
Do Van Quy 1670022
Tran Ba Thu Hoang 1669875

Instructor:
Du Dinh Vien

2024

Degree of Bachelor of Computer Science of Troy University

Table of Contents

List of tables

List of figures

Foreword

In the dynamic landscape of information management and technology, libraries are stepping beyond traditional roles and embracing new digital paradigms and tools to enhance their services to their communities. This transformation is driven by the integration of advanced library management systems, which enable seamless access to extensive repositories of knowledge and resources.

This report delves into the software engineering process to create a software library management system. As libraries strive to adapt to the dynamic demands of the digital age, the significance of a well-implemented library management system cannot be overstated. It is not merely a tool for cataloging and tracking resources but a cornerstone for fostering an environment where information is readily accessible, organized, and secure.

Introduction

Reason for Selecting LMS:

The selection of the Library Management System as our research topic arose from the compelling convergence of contemporary challenges and opportunities in the realm of education and information management. Our interest was sparked by the increasing integration of technology in education and the imperative for streamlined access to knowledge resources. We aim to explore a comprehensive solution that bridges the gap between traditional libraries and the digital age. Libraries have transcended being mere repositories of physical books to become dynamic information hubs that facilitate access to a diverse array of digital resources, services, and learning opportunities. The challenges posed by copyright concerns, efficient resource management, and the user experience in the evolving landscape of academia served as driving forces in our choice of this pertinent topic.

Objective:

The purpose of this report is to conduct a comprehensive analysis of the software engineering process utilized in creating and integrating a Library Management System (LMS). The report is set to accomplish the following key objectives:

a) **Detailing the Software Engineering Life Cycle:**
   - To outline the complete software engineering life cycle (SELC) as applied to the LMS project, including requirements analysis, design, development, testing, deployment, and maintenance.

b) **Defining Requirements and Scope:**
   - To document and analyze the functional and non-functional requirements of the LMS, ensuring that the system addresses the specific needs of the library and its users.
   - To establish the scope of the project, including key features, system constraints, and performance criteria.

c) **Design and Implement:**

- To describe the design and architectural decisions made during the development of the LMS, including system architecture, database design, user interface design, and integration strategies.
- To illustrate how the design aligns with best practices and addresses the requirements identified.
- To detail the development process, including coding standards, technologies used, and development methodologies employed (in this project is Agile).
- To provide insights into the implementation phase, including challenges encountered and solutions applied.

d) **Testing and Quality Assurance:**

- To outline the testing strategies and methodologies employed to ensure the LMS meets quality standards and performs as expected.

e) **Deployment and Integration:**

- To describe the deployment process, including the steps taken to roll out the LMS to the production environment.
- To discuss integration with existing systems and data migration strategies, if applicable.

f) **Maintenance and Support:**

- To address the ongoing maintenance and support strategies for the LMS, including bug fixes, updates, and user support mechanisms.
- To outline the plan for handling future enhancements and system upgrades.

Members responsibilities:

Project Manager: Nguyen Viet Thien Quoc

Business Analyst: Le Van Hai, Do Van Quy

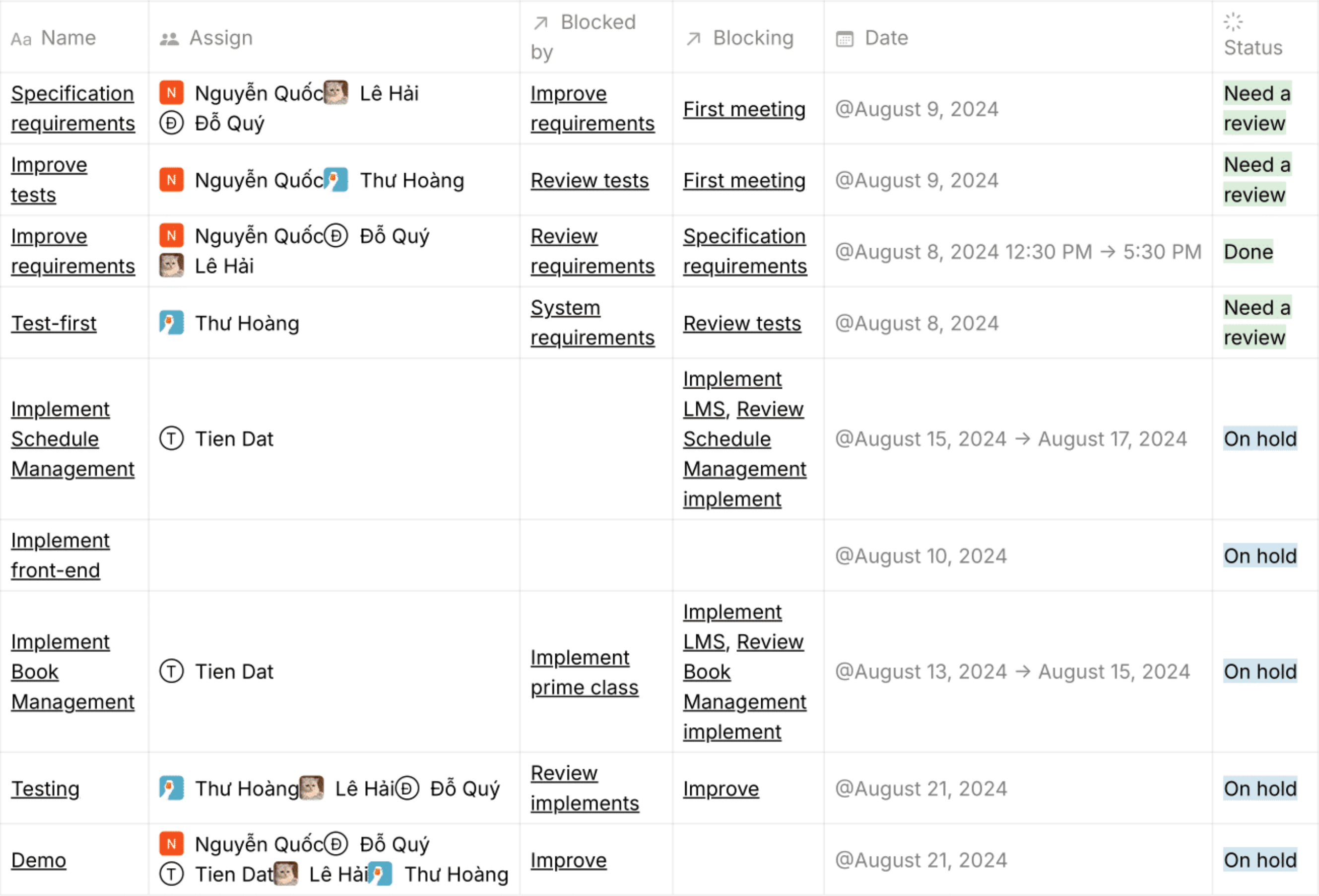
Developer: Nguyen Tien Dat

Tester: Tran Ba Thu Hoang

Task Distribution: Board of tasks

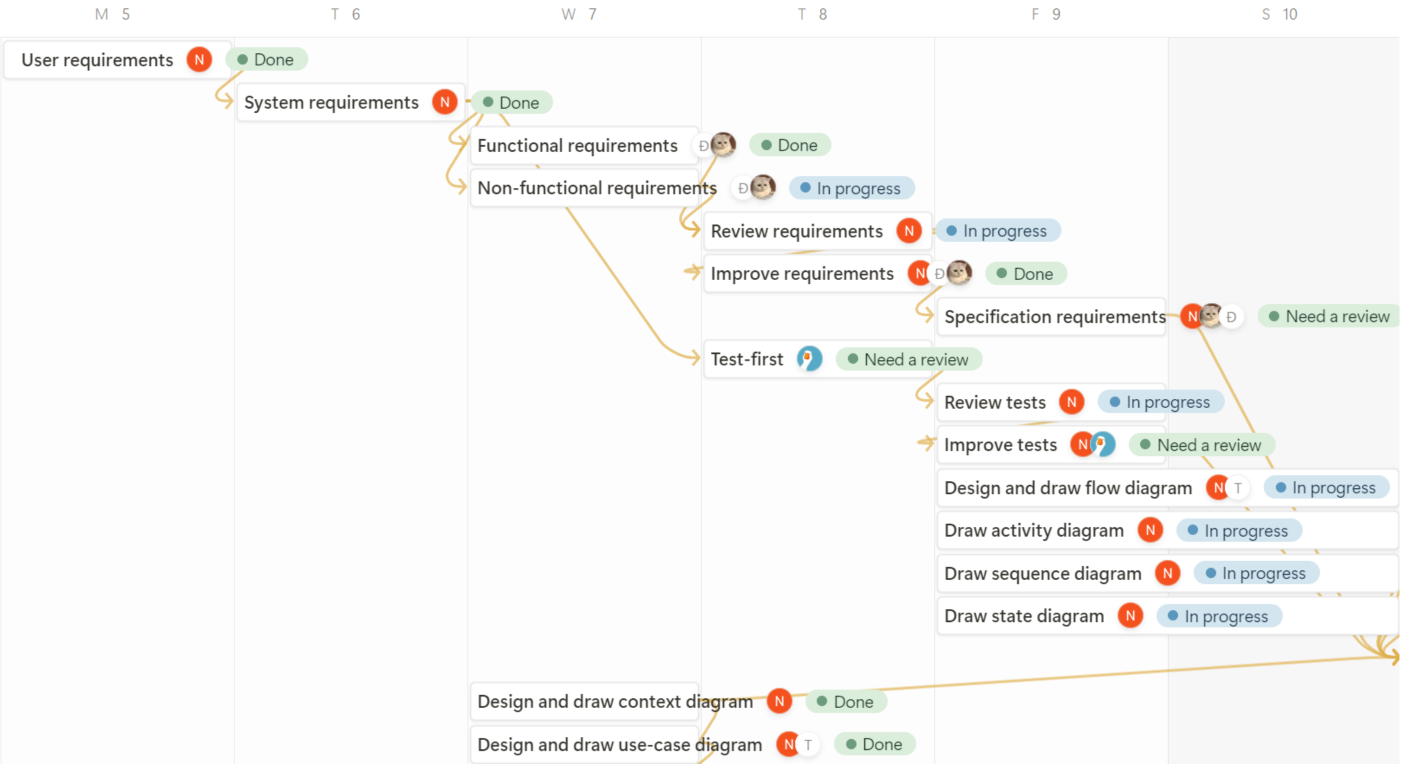
Aa Name	Assign	Blocked by	Blocking	Date	Status
User requirements	N Nguyễn Quốc		System requirements	@August 5, 2024	Done
System requirements	N Nguyễn Quốc	User requirements	Functional requirements, Non-functional requirements, Test-first	@August 6, 2024	Done
Functional requirements	Đ Đỗ Quý Lê Hải	System requirements	Review requirements	@August 7, 2024	Done
Non-functional requirements	Đ Đỗ Quý Lê Hải	System requirements	Review requirements	@August 7, 2024	In progress
Design class diagram	T Tien Dat	Design and draw use-case diagram, Design and draw context diagram	Review class design	@August 8, 2024	Done
Implement prime class	T Tien Dat	Review class design	Implement Book Management, Review units implement	@August 10, 2024 → August 11, 2024	Done
Review requirements	N Nguyễn Quốc	Functional requirements, Non-functional requirements	Improve requirements	@August 8, 2024 8:00 AM → 11:30 AM	In progress
Review class design	N Nguyễn Quốc T Tien Dat	Design class diagram	Implement prime class	@August 9, 2024	Done
Design and draw flow diagram	N Nguyễn Quốc T Tien Dat		First meeting	@August 9, 2024 → August 10, 2024	In progress
Design and draw use-case diagram	N Nguyễn Quốc T Tien Dat		Design class diagram	@August 7, 2024	Done
Review tests	N Nguyễn Quốc	Test-first	Improve tests	@August 9, 2024	In progress
Design and draw context diagram	N Nguyễn Quốc		Design class diagram, First meeting	@August 7, 2024	Done
Draw activity diagram	N Nguyễn Quốc		First meeting	@August 9, 2024 → August 10, 2024	In progress
Draw sequence diagram	N Nguyễn Quốc		First meeting	@August 9, 2024 → August 10, 2024	In progress
Draw state diagram	N Nguyễn Quốc		First meeting	@August 9, 2024 → August 10, 2024	In progress

Name	Assign	Blocked by	Blocking	Date	Status
Implement Administrator Management	Tien Dat		Implement LMS, Review User Management implement	@August 11, 2024 → August 12, 2024	Not started
Implement Student Mangement	Tien Dat		Implement LMS, Review User Management implement	@August 12, 2024 → August 13, 2024	Not started
Implement LMS	Tien Dat	Implement Student Mangement, Implement Administrator Management, Implement Book Management, Implement Schedule Management	Review implements	@August 19, 2024 → August 20, 2024	Not started
Presentation				@August 23, 2024	Not started
Review implements	Nguyễn Quốc Đỗ Quý Tien Dat Lê Hải Thư Hoàng	Implement LMS	Testing	@August 20, 2024	Not started
Improve	Đỗ Quý Lê Hải Thư Hoàng Tien Dat	Testing	Demo	@August 21, 2024	Not started
First meeting	Nguyễn Quốc Lê Hải Đỗ Quý Tien Dat Thư Hoàng	Specification requirements, Draw state diagram, Draw sequence diagram, Draw activity diagram, Design and draw flow diagram, Improve tests, Design and draw context diagram		@August 11, 2024	Done
Review Book Management implement		Implement Book Management		@August 15, 2024	Not started
Review Schedule Management implement		Implement Schedule Management		@August 17, 2024	Not started
Review units implement		Implement prime class		@August 11, 2024	In progress



Aa Name	Assign	Blocked by	Blocking	Date	Status
Specification requirements	N Nguyễn Quốc Lê Hải Đ Đỗ Quý	Improve requirements	First meeting	@August 9, 2024	Need a review
Improve tests	N Nguyễn Quốc Thư Hoàng	Review tests	First meeting	@August 9, 2024	Need a review
Improve requirements	N Nguyễn Quốc Đ Đỗ Quý Lê Hải	Review requirements	Specification requirements	@August 8, 2024 12:30 PM → 5:30 PM	Done
Test-first	Thư Hoàng	System requirements	Review tests	@August 8, 2024	Need a review
Implement Schedule Management	T Tien Dat		Implement LMS, Review Schedule Management implement	@August 15, 2024 → August 17, 2024	On hold
Implement front-end				@August 10, 2024	On hold
Implement Book Management	T Tien Dat	Implement prime class	Implement LMS, Review Book Management implement	@August 13, 2024 → August 15, 2024	On hold
Testing	Thư Hoàng Lê Hải Đ Đỗ Quý	Review implements	Improve	@August 21, 2024	On hold
Demo	N Nguyễn Quốc Đ Đỗ Quý T Tien Dat Lê Hải Thư Hoàng	Improve		@August 21, 2024	On hold

Timeline: Timeline


M 5 T 6 W 7 T 8 F 9 S 10

User requirements N Done

System requirements N Done

Functional requirements Đ Done

Non-functional requirements Đ In progress

Review requirements N In progress

Improve requirements N Đ Done

Specification requirements N Đ Need a review

Test-first Need a review

Review tests N In progress

Improve tests N Need a review

Design and draw flow diagram N T In progress

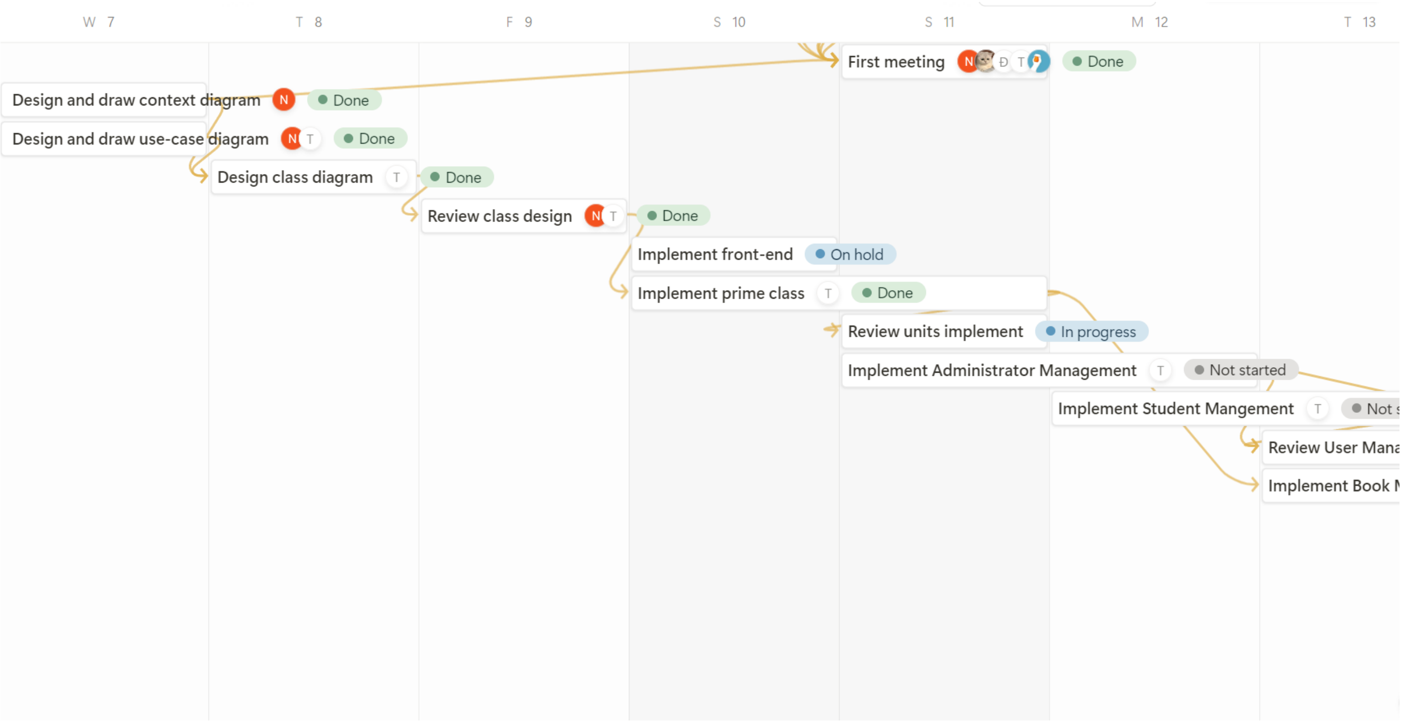
Draw activity diagram N In progress

Draw sequence diagram N In progress

Draw state diagram N In progress

Design and draw context diagram N Done

Design and draw use-case diagram N T Done


W 7 T 8 F 9 S 10 S 11 M 12 T 13

First meeting N Đ T Done

Design and draw context diagram N Done

Design and draw use-case diagram N T Done

Design class diagram T Done

Review class design N T Done

Implement front-end On hold

Implement prime class T Done

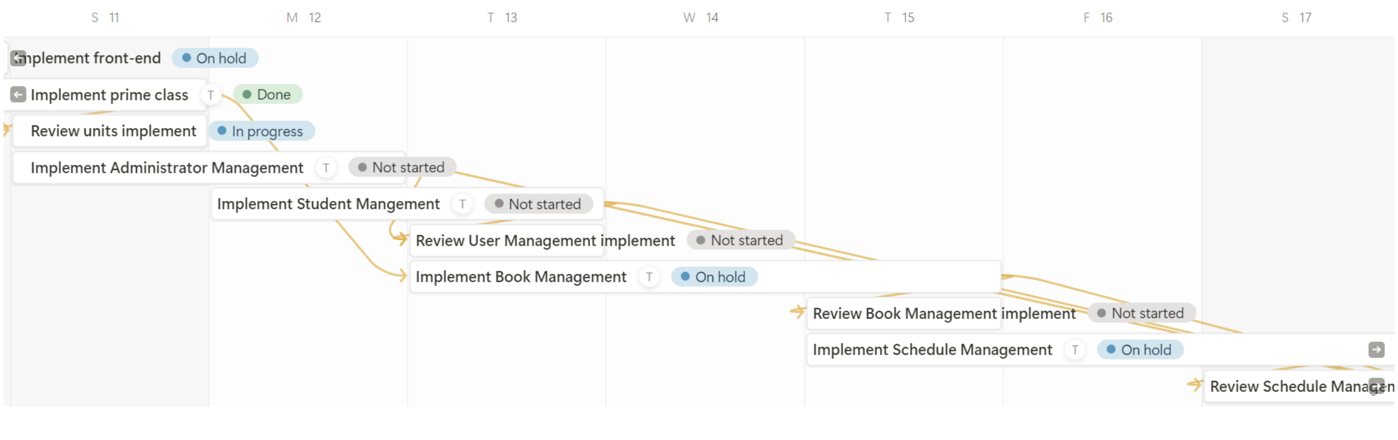
Review units implement In progress

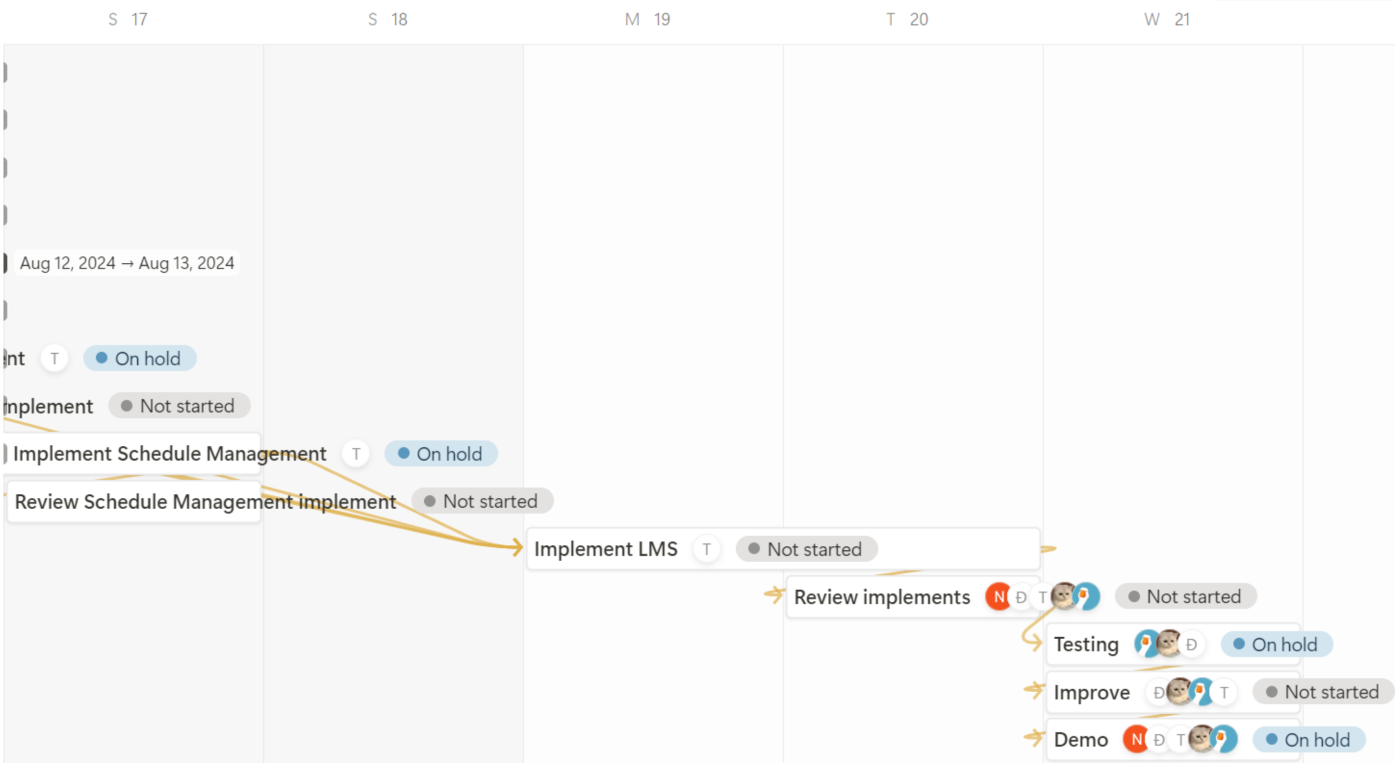
Implement Administrator Management T Not started

Implement Student Mangement T Not s

Review User Mana

Implement Book M





Workspace:

Notion: LMS Notion

Google Sheet: LMS Requirements Sheet

Draw.io: LMS Chart, Diagram

Figma: Ideas and brainstorms

GitHub: LMS GitHub

General Description

HaNoi University of Science and Technology (HUST):

is a prestigious institution located in Hanoi, Vietnam, established in 1956. As one of the leading universities in the country, HUST is renowned for its dedication to advancing knowledge and research in fields such as engineering, technology, and applied sciences.

Due to its expanding academic and research endeavors, HUST intends to improve its library operations by introducing a sophisticated Library Management System (LMS). The main objectives of this new system are to simplify library procedures, enhance resource management, and provide a more effective and user-friendly experience for both library staff and students.

The university needs a complete LMS solution that can manage cataloging, circulation, user management, and integration of digital resources. In addition, it should provide strong support and training. The goal is to support HUST's mission of academic excellence and research innovation by improving the management of its library resources.

User requirements:

- The LMS is designed for the use of librarians and only students in the university. It will check if the user is a school student through the login and student data provided.
- Using a library management system, librarians easily manage books in the library, borrowed books, and students who borrow books.
- This software should contain all the book information of the university's library in the system, which provides student book information to borrow and return borrowed books.
- The system also provides a search book function, book information, and books available to borrow.
- After viewing all book information, students will able to schedule a duration to borrow books on the system and come to the library and borrow books.
- At the end of the borrow duration, the system sends a reminder to the user email, alert about the returning book.

- The system must provide a Root Administrator account, which can create a normal administrator account.
- Only the root admin can create an administrator account and manage them, normal administrators can manage student accounts and managed books, but not manage other administrator accounts.

System requirements:

- For student users, the provided student database is restricted to creating user accounts only for university students.
- For administrator users, only the root admin can create an admin account, and the admin account is not restricted to any database.
- Students can search for books, view book information, and schedule book borrow duration.
- Normal administrator users can manage books, manage student accounts, and manage schedule time.
- The root administrator user is an administrator user but he can manage the administrator.
- All Unit Management has 3 main functions: Add, Edit, and Remove unity. Units like Book, User, and Borrow Time have smaller functions which would specify in functional requirement
- Unit Management needs to separate each other, prime unit like User only access to some system function

Specification:

Functional requirements:

- Login: The system must provide login functionality to authenticate users and grant access based on their role.
    - Login Process:
        - Users will enter their credentials (username and password) into designated fields.

          The system must validate the credentials against the stored user data.
    - Authentication:
        - Students:

        - The system must check if the provided credentials match a record in the university's student database.
          Access will be granted if the credentials are valid and the user is a current student.
      - For Administrators (Normal and Root):
        - The system must check if the provided credentials match a record in the system's administrator database.
        - Access will be granted if the credentials are valid and the user is an authorized administrator.
    - Access Control:
      - Students:
        - should have access to student-specific functionalities (e.g., searching for books, borrowing books).
      - Normal administrators:
        - should have access to administrator functionalities (e.g., managing books, managing student accounts).
      - The root administrator:
        - should have access to all functionalities, including the ability to create and manage other administrator accounts.
    - Error Handling:
      - If login credentials are invalid, the system must display an error message indicating that the credentials are incorrect.
      - The system should not disclose specific details about which part of the credentials is incorrect (e.g., username or password).
    - Security:
      - Passwords must be securely hashed and stored.
      - The login process should be protected against common security threats such as brute force attacks and SQL injection.

- Logout: The system must allow users to safely end their session and exit the system, ensuring that no further actions can be performed until they log in again.
    - Logout process:

        - User click on log out button.
        - The system must show logout verify.
        - If user verify log out then log out navigate to log in interface, else stay on the work page.
    - Session Termination:
        - All session data, including authentication tokens or session identifiers, must be invalidated to prevent unauthorized access.
    - Security:
        - Any attempt to access restricted areas after logout should redirect the user to the login page.

- Search Book: The system should include a search function for finding books in the library. It must support multiple criteria and provide relevant results based on user input.
    - Search Criteria:
        - Title: Users can enter a book title to find matching books.
        - Author: Users can enter an author's name to locate books written by that author.
        - ISBN: Users can input the International Standard Book Number (ISBN) to find a specific book.
        - Genre: Users can select a genre to find books categorized under that genre.
        - Publication Date: Users can filter search results based on the book's publication date.
        - Availability: Users can filter by availability to find books that are currently available for borrowing.
        - Language: Users can filter by the language of the book.
        - Edition: Users can search for specific editions of a book.
    - Search Results:
        - Relevance: The system should display search results ranked by relevance to the search criteria.
        - Book Details: Each search result should include the book's title, author, genre, ISBN, availability status, and location within the library.

        - Pagination: The system should support pagination for search results to handle large numbers of matching books.
        - Sorting: Users should be able to sort search results by title, author, publication date, and availability.
    - Advanced Search:
        - Multi-Criteria Search: Users can combine multiple search criteria (e.g., author and genre) to narrow down search results.
        - Boolean Search: The system should support Boolean operators (AND, OR, NOT) for more refined searches.
    - Search Optimization:
        - Auto-Suggestions: The system should provide auto-suggestions as users type in the search field.
        - Spell Check: The system should include a spell check feature that suggests corrections for misspelled search terms.
        - Synonym Support: The system should recognize synonyms and return relevant results (e.g., searching for "novel" should also return results for "book").

- View Book Information: The system allows users to search for books and view detailed information about them, including availability, author, genre, and publication details. It also enables users to see related books and, optionally, access book reviews if available.
    - Display Book Details:
        - When a student or administrator selects a book from the search results, the system should display detailed information about the book.
        - The detailed information should include the book title, author, publication year, ISBN, genre, summary, and the number of available copies.
    - Check Availability:
        - The system should clearly indicate whether the book is available for borrowing or if it is currently checked out.
        - If the book is checked out, the system should display the expected return date.

  - View Related Books:
    - The system should provide a list of related books based on the genre or author to help students find similar books of interest.
    - This feature should be accessible on the book details page.
  - Access Book Reviews (Optional):
    - If available, the system should allow students to view reviews and ratings left by other users for the book.
    - This feature should also be accessible on the book details page.

- Schedule Book Borrow Time: The system allows users to select and reserve a specific time slot for borrowing a book from the library, with features for confirming, modifying, canceling, and receiving reminders about the scheduled borrow time.
  - Allow users to schedule a book borrow duration
    - The system should allow students to view the available time slots for borrowing a book.
    - Students can select a specific time slot and schedule when they want to borrow a book.
    - The system should check if the selected time slot is available before confirming the schedule.
  - Display available borrow time slots
    - The system should display available time slots based on the book's current availability.
    - If a book is currently borrowed, the system should automatically exclude that time slot from the available options.
  - Confirmation of borrow time
    - Once a student schedules a borrow time, the system should send a confirmation message to the student’s email.
    - The confirmation should include the book's details, the scheduled borrow time, and instructions for picking up the book.
  - Modify or cancel scheduled borrow time
    - The system should allow students to modify or cancel their scheduled borrow time before the scheduled time begins.

        - Students should receive an updated confirmation if they modify the schedule, or a cancellation notice if they cancel the schedule.
    - Reminder notifications for scheduled borrow time
        - The system should send a reminder notification to the student's email before the scheduled borrow time.
        - The reminder should include the book's details, the scheduled borrow time, and a reminder to pick up the book at the library.
    - Conflict detection for scheduling
        - The system should detect conflicts in scheduling if multiple students attempt to borrow the same book at the same time.
        - In the case of a conflict, the system should notify the student and suggest alternative available times.
    - Error Handling
        - If an error occurs during the editing process (e.g., invalid dates, database errors), the system must provide a clear error message and guide the user on how to resolve the issue or try again.
        - The system must ensure that no duplicate or conflicting schedules are created during the editing process.
    - Schedule Confirmation:
        - After making changes to the schedule, the system must display a summary of the updated schedule for confirmation.
        - The user must confirm the changes before the system updates the schedule in the database.
    - Update System Database
        - Upon confirmation, the system must update the library's database to reflect the removal of the borrowing schedule.
        - The book's availability status must be updated to reflect that it is available for borrowing again.

- **Edit Schedule:** The system allows students and administrators to modify an existing book borrowing schedule, including changing the borrowing dates or details.
    - **Access Control:**

    - Only the student who created the schedule and administrators (both normal and root) have access to edit the borrowing schedule.
    - The system must authenticate the user’s credentials and verify their permissions before allowing access to edit the schedule.
- **Allow users to schedule a borrow period**
    - The system should allow users to view the available time slots for borrowing a book.
    - Users can select a specific time slot and schedule when they want to borrow a book.
    - The system should check if the selected time slot is available before confirming the schedule.
- **Schedule Selection:**
    - The user must be able to view their existing schedules or all schedules if they are an administrator.
    - The system must provide a search interface or list view for users to select the schedule they wish to remove.
- **Edit Details:**
    - Once a schedule is selected, the system must display the current borrowing details, including book title, start date, end date, and any other relevant information.
    - The user must be able to modify the borrowing period by selecting new start and end dates, as well as making any necessary adjustments to other details (e.g., book condition notes).
- **Reminder notifications for scheduled borrow time:**
    - The system should send a reminder notification to the student’s email before the scheduled borrowing time.
    - The reminder should include the book's details, the scheduled borrow time, and a reminder to pick up the book at the library.
- **Conflict detection for scheduling:**
    - The system should detect conflicts in scheduling if multiple students attempt to borrow the same book at the same time.

        - In the case of a conflict, the system should notify the student and suggest alternative available times.
    - **Error Handling:**
        - If an error occurs during the editing process (e.g., invalid dates, database errors), the system must provide a clear error message and guide the user on how to resolve the issue or try again.
        - The system must ensure that no duplicate or conflicting schedules are created during the editing process.
    - **Schedule Confirmation:**
        - After making changes to the schedule, the system must display a summary of the updated schedule for confirmation.
        - The user must confirm the changes before the system updates the schedule in the database.
    - **Update System Database:**
        - Upon confirmation, the system must update the library's database to reflect the removal of the borrowing schedule.
        - The book's availability status must be updated to reflect that it is available for borrowing again.

- **Remove Schedule:** The system allows students and administrators to remove an existing book borrowing schedule.
    - **Access Control:**
        - Only the student who created the schedule and administrators (both normal and root) have access to edit the borrowing schedule.
        - The system must authenticate the user's credentials and verify their permissions before allowing access to edit the schedule.
    - **Schedule Removal Process:**
        - Once a schedule is selected, the system must display a confirmation prompt, including details of the schedule (e.g., book title, borrowing period).
        - The user must confirm the removal action by clicking a confirmation button.
    - **Schedule Selection:**

        - The user must be able to view their existing schedules or all schedules if they are an administrator.
        - The system must provide a search interface or list view for users to select the schedule they wish to remove.
    - **Access Logs:**
        - The system must log all actions related to removing schedules, including details such as the book ID, schedule details, time of removal, and the user who performed the action.
        - These logs must be accessible to administrators for audit purposes and to track changes in the scheduling system.
    - **Reminder notifications for scheduled borrow time:**
        - The system should send a reminder notification to the student's email before the scheduled borrowing time.
        - The reminder should include the book's details, the scheduled borrow time, and a reminder to pick up the book at the library.
    - **Error Handling:**
        - If an error occurs during the editing process (e.g., invalid dates, database errors), the system must provide a clear error message and guide the user on how to resolve the issue or try again.
        - The system must ensure that no duplicate or conflicting schedules are created during the editing process.
    - **Update System Database:**
        - Upon confirmation, the system must update the library's database to reflect the removal of the borrowing schedule.
        - The book's availability status must be updated to reflect that it is available for borrowing again.
- **Load Schedule:** The system enables to load and display the book borrowing schedules from the database, providing information on current and upcoming book borrowings.
    - **Access Control:**
        - Only authorized users (administrators) can access and load schedule data. Students can view their own borrowing schedules but cannot access data for other students.

    - The system must enforce access controls to ensure that users only see schedule data they are permitted to view.

- **Schedule Display:**
    - The system must display the loaded schedule data in a user-friendly format, such as a table or calendar view, allowing administrators and students to view current and upcoming book borrowings.
    - For each schedule entry, the system must show relevant details, including book title, student name, borrowing dates, and status (e.g., pending, active, completed).
- **Data Retrieval:**
    - The system must retrieve borrowing schedule data from the database, including details such as book ID, student ID, start date, end date, and borrowing status.
    - The system must be able to load schedules for specific books, students, or a range of dates, based on the query parameters provided by the administrator or the system itself.
- **Filtering and Sorting:**
    - The system must provide options to filter and sort the schedule data based on criteria such as book title, student name, or borrowing date.
    - The filtering and sorting options must be intuitive and easy to use, allowing users to quickly find specific schedules or view schedules in a preferred order.
- **Data Accuracy:**
    - The system must ensure that the schedule data loaded from the database is accurate and up-to-date.
    - Any discrepancies or errors in the schedule data must be flagged and reported, allowing administrators to correct or investigate issues.

- **Save Schedule:** The system enables to save the borrowing schedule data into the database, ensuring that all borrowing arrangements are properly recorded and managed.
    - **Save Schedule Data:**
        - The system must save the details of each borrowing schedule, including the student's ID, the book's ID, start date, end date, and any special notes or requests.
        - The schedule data must be stored in a structured format within the database to allow for easy retrieval and management.
    - **Data Validation:**
        - Before saving the schedule, the system must validate the data to ensure that it complies with library policies (e.g., no overlapping borrow periods, valid dates).
        - The system must check for conflicts and ensure that the book is available for the requested period.
    - **Update Existing Schedules:**
        - If a student needs to modify an existing schedule (e.g., change the borrowing period), the system must allow updates and save the new schedule data.
        - The system must keep track of changes to the schedule, including the date and time of modification and the administrator who made the change.
    - **Error Handling:**
        - If an error occurs during the saving process (e.g., database connection issues, validation errors), the system must provide a clear error message to the user and guide them on how to address the issue.
        - The system must ensure that no partial or corrupted schedule data is saved.
    - **Data Security:**
        - The system must ensure that schedule data is securely stored and protected from unauthorized access.

        - Access to schedule data must be restricted based on user roles, with only authorized administrators able to view or modify the data.

- **Add Book:** The system allows librarians (administrators) to add new books to the library management system. This feature ensures that the library's book collection is updated with the latest additions and maintains accurate records for students to access.
    - **Admin must be able to input book details:**
        - The system must provide a form where the administrator can enter the book's title, author, ISBN, publication date, and genre.
        - The form should validate that all required fields are filled out correctly before submission.
    - **Admin should receive confirmation upon successful addition:**
        - After a book is successfully added, the system should display a confirmation message to the administrator.
        - The confirmation should include the book’s details for verification.
    - **The system must store and display book information:**
        - Once the book details are submitted, the system should save the information to the database.
        - The newly added book should be immediately visible in the system’s book catalog and searchable by students and librarians.
    - **Admin should be able to upload book cover images:**
        - The system should provide an option to upload a cover image along with the book details.
        - Uploaded images should be stored and displayed with the book information in the catalog.
- **Remove Book:** The system allows authorized users to remove a book from the library's inventory.
    - **Authentication Check:**
        - Only administrators (both normal and root) can access the Remove Book functionality.

    - The system must authenticate the administrator's credentials before granting access to this feature.
- **Book Selection:**
    - The system must provide a search interface for administrators to locate the book to be removed.
    - The search should support querying by book ID, title, author, and ISBN.
- **Confirmation Before Removal:**
    - Once a book is selected, the system must display detailed information about the book for confirmation before proceeding with the removal.
    - The administrator must confirm the removal action by clicking a confirmation button.
- **Removal Process:**
    - Upon confirmation, the system must update the library's database to reflect the removal of the book.
    - The book's record must be permanently deleted from the inventory.
- **Notification of Removal:**
    - After a book is removed, the system must generate a notification log entry indicating the book's removal, including details such as the book ID, title, author, and the administrator who performed the action.
- **Error Handling:**
    - If the book is currently checked out, the system must prevent its removal and display an appropriate error message indicating that the book cannot be removed while it is on loan.
    - The system must handle errors gracefully and provide clear instructions for resolving issues, such as contacting support.
- **Access Control:**
    - The root administrator has the additional capability to manage and monitor all book removals conducted by normal administrators.

        - The system must log all actions related to book removal, including timestamps and user information, for audit purposes.

- **Edit Book:** The system allows authorized users to edit or update the information of a book in the library's inventory.
    - **Authentication Check:**
        - Only administrators (both normal and root) can access the Remove Book functionality.
        - The system must authenticate the administrator's credentials before granting access to this feature.
    - **Book Selection:**
        - The system must provide a search interface for administrators to locate the book to be removed.
        - The search should support querying by book ID, title, author, and ISBN.
    - **Display Book Information:**
        - Once a book is selected, the system must display all current information about the book, including title, author, ISBN, publication date, genre, availability status, and any other relevant details.
        - The displayed information must be editable through a form interface.\
    - **Update Book Information:**
        - Administrators can update any of the book's details, including changing its availability status (e.g., available, checked out, reserved).
        - The system must validate the input to ensure all required fields are correctly filled and that no invalid data is entered (e.g., invalid ISBN format).

    - **Confirmation Before Saving Changes:**
        - After editing, the system must prompt the administrator to confirm the changes before they are saved.

        - The confirmation prompt should display a summary of the changes made.
    - **Save Changes and Update Database:**
        - Upon confirmation, the system must save the updated information and reflect the changes in the library’s database.
        - The system should ensure that all related records (e.g., loan history, availability status) are updated accordingly.
    - **Notification and Logging:**
        - The system must generate a notification log entry for each edit, detailing the changes made, including the book ID, title, author, the edited fields, and the administrator who performed the action.
        - The root administrator should have access to view and manage these logs.
    - **Error Handling:**
        - If the system encounters an error during the update process, it must provide a clear error message indicating the problem and suggest possible solutions.
        - The system should ensure that no partial updates are saved in the case of an error, maintaining data integrity.

- **Load Book Data:** The system allows the system to retrieve and display book data from the library's database for both students and administrators.
    - **Database Connection:**
        - The system must establish a secure connection to the library's database to retrieve book data.
        - The connection must be optimized to handle multiple simultaneous queries without significant performance degradation.
    - **Data Retrieval:**
        - The system should retrieve book data including book ID, title, author, publication date, genre, ISBN, availability status, book cover.

        - The data retrieval process must be efficient, ensuring minimal load times for users
    - **Display of Book Information:**
        - The retrieved book data must be displayed in a user-friendly format on the interface for both students and administrators.
        - The display should include sortable columns and filtering options (e.g., by title, author, genre) to help users quickly find specific books.
    - **Error Handling:**
        - If the system encounters an issue during data retrieval (e.g., database connection failure), it must display a clear error message to the user.
        - The system should provide options for retrying the operation or contacting support if the problem persists.
    - **Data Integrity:**
        - The system must ensure that the book data displayed is accurate and up to date, reflecting the current state of the library's inventory.
        - Any discrepancies found in the data should be logged for review by the root administrator.

- **Save Book Data:** The system allows authorized users to save or update book information in the system’s database.
    - **Authentication Check:**
        - Only administrators (both normal and root) can access the Save Book Data functionality.
        - The system must authenticate the administrator’s credentials before granting access to this feature.
    - **Input Validation:**
        - The system must validate all input fields (e.g., title, author, ISBN, publication year) for completeness and correctness before saving the book data.
        - If any required field is missing or contains invalid data, the system must prompt the user to correct the information.

- **Data Integrity:**
    - The system must ensure that each book has a unique identifier (e.g., ISBN) in the database to prevent duplication.
    - If a book with the same identifier already exists in the database, the system should provide an option to update the existing record or cancel the operation.
- **Save Operation:**
    - Once the input is validated, the system must save the book information to the database.
    - The system should confirm that the save operation was successful and display a confirmation message to the administrator.
- **Error Handling:**
    - If the save operation fails (e.g., due to a database connection issue), the system must display an error message and log the failure for troubleshooting.
    - The system should also allow the administrator to retry the save operation after resolving the issue.
- **Notification of Changes:**
    - After saving the book data, the system should generate a notification log entry indicating the changes made, including details such as the book ID, title, author, and the administrator who performed the save operation.
- **Access Control:**
    - The root administrator has the additional capability to review and audit all book data save operations conducted by normal administrators.
    - The system must log all actions related to saving book data, including timestamps and user information, for audit purposes.

- **Add Student:** The system allows authorized administrators to add a new student to the system, enabling them to access and use the library's resources.
    - **Authentication Check:**

    - Only administrators (both normal and root) can access the Save Book Data functionality.
    - One exception is Student create account, allow Student to access this function.
    - The system must authenticate the administrator’s credentials before granting access to this feature.
  - **Student Data Validation:**
    - The system must validate the student’s data against the university’s student database to ensure the individual is a registered student.
    - Required fields for adding a student include student ID, name, email, and course details. The system must check that all required fields are provided and correctly formatted.
  - **Duplicate Check:**
    - Before adding a new student, the system must check for existing records with the same student ID to prevent duplicate entries.
    - If a duplicate is found, the system must display an error message and prevent the student from being added.
  - **Account Creation:**
    - Upon successful validation, the system will create a student account in the - LMS, assigning a unique user ID.
    - The system should automatically generate and send login credentials (username and temporary password) to the student’s registered email.
  - **Confirmation and Notification:**
    - After the student account is created, the system must display a confirmation message to the administrator.
    - The system should log the action, recording details such as the student ID, name, and the administrator who performed the action for audit purposes.
  - **Error Handling:**
    - If there is an issue with the student data (e.g., invalid email format, student not found in the university’s database), the

system must provide a clear error message and guide the administrator on how to correct the issue.
        - The system must handle all errors gracefully and ensure that no incomplete or invalid student accounts are created.
    - **Access Control:**
        - The root administrator has the additional capability to view all student accounts and their creation history, including those added by normal administrators.
        - The system must log all actions related to adding students, including timestamps and user information, for future reference and audits.

- **Edit Student:** The system allows administrators to edit or update information related to a student's book borrowing history and scheduled borrowing sessions.
    - **Authentication Check:**
        - Only administrators (both normal and root) can access the Save Book Data functionality.
        - One exception is Student create account, allow Student to access this function.
        - The system must authenticate the administrator's credentials before granting access to this feature.
    - **Student Selection:**
        - The system must provide a search interface for administrators to locate the student whose borrowing information needs to be edited.
        - The search should support querying by student ID, name, and email address.
    - **View Borrowing History:**
        - Once a student is selected, the system must display a detailed view of the student's borrowing history, including currently borrowed books, past borrowed books, and upcoming scheduled borrowings.
    - **Edit Borrowing Information:**

        - Administrators can edit the return date of currently borrowed books if necessary.
        - The system must allow the administrator to update the status of a book return, including marking it as returned or extending the borrowing period.
    - **Edit Scheduled Borrowing:**
        - The system must allow administrators to edit the scheduled borrowing duration for future book borrowings.
        - Administrators can change the scheduled pickup date, the duration of the borrowing period, or cancel a scheduled borrowing session entirely.
    - **Notification of Changes:**
        - After any edits are made, the system must automatically send a notification email to the student, detailing the changes made to their borrowing or scheduling information.
        - The notification must include information such as the updated return date, changes to the scheduled borrowing, or any cancellations.
    - **Access Control and Logging:**
        - All actions related to editing a student's borrowing information must be logged, including the administrator's details, the changes made, and timestamps.
        - The root administrator has the ability to review logs of all edits made by normal administrators for auditing purposes.

- **Search Student:** The system allows administrators to search for and view detailed information about students registered in the system.
    - **Authentication Check:**
        - Only administrators (both normal and root) can access the Save Book Data functionality.
        - One exception is Student create account, allow Student to access this function.
        - The system must authenticate the administrator's credentials before granting access to this feature.

  - **Search Criteria:**
    - The system must provide a search interface that allows administrators to search for students by various criteria such as student ID, name, department, or email address.
    - The search should support partial matches and return a list of students that meet the search criteria.
  - **Display Results:**
    - The system must display a list of students matching the search criteria, including basic details such as student ID, name, department, and email.
    - Administrators can select a student from the list to view more detailed information.
  - **View Student Details:**
    - When a student is selected from the search results, the system must display detailed information about the student, including contact information, borrowing history, and any active borrowed books.
    - The system must allow the administrator to view all associated information in a clear and organized manner.
  - **Error Handling:**
    - If no students match the search criteria, the system must display a message indicating that no results were found.
    - The system must handle errors gracefully, providing clear instructions for resolving issues, such as refining search criteria.
  - **Access Control and Permissions:**
    - Normal administrators can view and manage student accounts but cannot delete or deactivate them.
    - Root administrators have additional permissions to modify, delete, or deactivate student accounts if necessary.

- **Load Student Data:** The system loads and verifies student data from the university's database for authentication and account creation purposes.
  - **Authentication Check:**

    - Only administrators (both normal and root) can access the Save Book Data functionality.
    - One exception is Student create account, allow Student to access this function.
    - The system must authenticate the administrator’s credentials before granting access to this feature.
- **Connection to University Database:**
    - The system must establish a secure connection to the university’s student database to retrieve the relevant student information.
    - The system must be able to handle potential connection issues and notify the administrator if there is a problem with the connection.
- **Data Retrieval:**
    - The system must load essential student information, including student ID, name, department, and current enrollment status.
    - The system must validate the data to ensure completeness and accuracy before it is used within the LMS.
- **Data Integration:**
    - Once the data is retrieved, the system must integrate it into the LMS to allow student account creation and verification.
    - The system must match the loaded student data with existing user accounts, flagging any discrepancies or potential issues.
- **Error Handling:**
    - If the system encounters any errors during the data retrieval process, such as missing or corrupted data, it must generate an error log and notify the administrator.
    - The system must provide clear instructions on how to resolve issues, including options to retry the data load or contact technical support.
- **Data Security:**
    - The system must ensure that all student data is securely stored and transmitted, using encryption and other security measures to protect sensitive information.

        - The system must comply with relevant data protection regulations, ensuring that student data is only accessible to authorized users.
    - **Access Control and Logging:**
        - Only the root administrator can view the logs and details of the data loading process.
        - The system must log all data loading activities, including timestamps, the administrator involved, and any errors encountered, for auditing purposes.

- **Save Student Data:** The system allows to save and manage student data within the database, ensuring that only authorized students can access the LMS.
    - **Authentication Check:**
        - Only administrators (both normal and root) can access the Save Book Data functionality.
        - One exception is Student create account, allow Student to access this function.
        - The system must authenticate the administrator's credentials before granting access to this feature.
    - **Data Input and Validation:**
        - Administrators must be able to input student data manually or import it from an existing database or file.
        - The system must validate the student data against the university's official records to ensure accuracy.
        - Fields required for each student entry include student ID, name, email, department, and enrollment status.
    - **Data Storage:**
        - The system must securely store the student data in the database, ensuring that it is accessible for student account creation, book borrowing, and other functionalities.
        - The student data must be indexed for quick retrieval and search operations.
    - **Data Update:**

    - Administrators must be able to update existing student data, including changes to personal information, department, or enrollment status.
    - The system must log all updates made to student data, including the time of the update and the administrator responsible for the change.
  - **Error Handling:**
    - If there is an error during data saving or validation (e.g., missing required fields, invalid student ID), the system must provide a clear error message to the administrator, highlighting the issues that need correction.
    - The system must prevent saving data if any validation errors exist.
  - **Access Control:**
    - Normal administrators can save, update, and manage student data, but they cannot delete student data.
    - Only the root administrator has the authority to delete student data if necessary. The system must log all actions related to saving and updating student data, including timestamps and user information, for audit purposes.

Non-functional requirements:

Overall:

- Usability:
  - The system is designed to be a user-friendly environment so that students and library staff can perform various tasks easily and efficiently without the need for special instructions or documents. any.
- Accuracy:

    - Stored data about books must be completely accurate and reliable, especially in terms of copyright. The search function must work perfectly and return the user exactly what they are looking for.
- Maintainability:
    - Software should be maintained, adding new features, and making changes to the software should be as simple as possible.
- Availability:
    - The system must be always available while the library is up and should be restored in less than an hour if the system fails. The system will respond to requests within three seconds or less.

Specification:

- Login: The system must provide a login functionality to authenticate users and grant access based on their role.
    - **Performance**:
        - Response Time: The login process should authenticate users and load the user's dashboard within 2 seconds after credentials are submitted
        - Concurrency: The system must support up to 10,000 concurrent login attempts without degradation in performance.
    - **Security**:
        - Encryption: All login credentials must be transmitted securely
        - Password storage: Passwords must be stored securely using a hashing algorithm
        - Account Lockout: The system must lock a user account after 5 consecutive unsuccessful login attempts
    - **Compliance:**
        - The login system must comply with GDPR (General Data Protection Regulation) on the handling of user data, ensuring that users' personal data is protected and not disclosed
    - **Reliability**:

        - Availability: The login service must have an uptime of 99.9% to ensure users can access the system at any time.
        - Failover: In case of primary system server failure, automatically switch to backup server.
    - **Usability**:
        - Error Messages: The system must provide clear and user-friendly error messages without revealing specific details that could aid unauthorized access attempts
        - Accessibility: The login interface must be accessible to a variety of interface types, including support for screen readers and keyboard navigation.

    - **Scalability**:
        - The system must be scalable to accommodate more than 70% of concurrent users without major architectural changes.

- **Logout:** The system must allow users to safely end their session and exit the system, ensuring that no further actions can be performed until they log in again.
    - **Performance**:
        - Response Time: The logout process must complete within 1 second, ensuring that the user is promptly redirected to the login or logout confirmation page
        - Scalability: The system must handle concurrent logout requests from up to 10,000 users without significant delays or impact on server performance.
    - **Security**:
        - Delete Data: Any temporary data stored on the client side must be deleted immediately upon logout
        - End of process: The system must completely terminate the user's session upon logout, ensuring that no further actions can be performed without re-authentication.

        - Automatic logout: After 10 minutes, the system will automatically log out and force the user to log back in to prevent unauthorized access from unattended sessions.
    - **Compliance:**
        - GDPR(General Data Protection Regulation) Compliance: The system must ensure that all personal data related to the session is processed in accordance with the requirements of the GDPR, ensuring that no data is accessible after logging out.
    - **Reliability**:
        - Error Handling: In the event of a server failure during the logout process, the system must ensure that all user data remains secure and notify the user whether the logout process was successful or needs to be retried.
        - Consistency: The logout process must be performed on all devices and browsers, ensuring that sessions are terminated completely across all platforms.
    - **Usability**:
        - User feedback: The system must provide clear feedback to users whether they have successfully or unsuccessfully logged out.
        - Accessibility: The logout functionality must be easily accessible from every page in the system, suitable for everyone.

- **Search book:** The system should include a search function for finding books in the library. It must support multiple criteria and provide relevant results based on user input.
    - **Performance**:
        - Response Time: The system must return search results within 2 seconds for typical queries under normal load conditions. If load conditions are high (during peak library/large-scale search hours), response time must not exceed 5 seconds.
        - Throughput: The search function must be able to handle at least 1000 concurrent search requests without performance degradation.

    - Efficiency: Use the most optimized search algorithm to minimize resource consumption
- **Security**:
    - Data Privacy: Search queries should not reveal any information history. Access to some restricted book information should be limited and marked as allowed to authorized users only (e.g., certain research articles or internal documents reserved for school staff).
    - Access Control: The system should grant explicit execution permissions to each different location of the system, ensuring that only authorized users can access or modify specific book details.
    - Audit Logging: All search queries and access to sensitive data should be recorded for auditing purposes, with logs kept secure and accessible only to authorized personnel.
- **Accuracy:**
    - Relevance: The search algorithm should prioritize and rank results based on relevance, with the most relevant books appearing first.
    - Fuzzy matching: The system should include fuzzy search capabilities to handle typos or minor errors in user input, providing relevant results even when search terms do not match exactly.
    - Sort and filter: Users should be able to sort results by criteria such as relevance, publication date, author name, or availability, and filter them by genre, language, or format (e.g., e-book, print book).
- **Availability**:
    - Uptime: The search function is always available during library business hours, ensuring users have access at almost all times. Planned maintenance should be scheduled outside of business hours and announced in advance to minimize disruption.

    - Failover and redundancy: The system should have a failover mechanism to ensure the search function remains available even in the event of a server failure.
- **Usability**:
    - Advanced Search Options: Users should be able to apply filters and combine multiple search criteria (e.g., title, author, genre, ISBN) to narrow results.
    - Search Suggestions: As users' type, the system should provide real-time search suggestions based on available book titles, authors, or genres.
    - User Interface (UI) Design: The search interface should be designed to be easy to use, with a clean, intuitive layout with clearly labeled fields for search criteria.
- **Scalability**:
    - Database Growth: The system must maintain performance as the number of books in the library database increases. Handle the database so that the time to search for books does not increase significantly as more records are added.
    - Elasticity: The system must be able to scale by adding more servers or processing power to accommodate increased user demand or database size.
- **Compatibility**:
    - Cross-platform functionality: The search functionality must be fully compatible with all major web browsers (Chrome, Firefox, Safari, Edge) and must work seamlessly on both desktop and mobile devices, including iOS and Android platforms.
- **Maintainability**:
    - Modular design: The search functionality should be implemented as a modular component of the system, allowing for easy updating, modification, or replacement without affecting other system functions.
    - Testability: The search functionality should be designed to allow for automated testing, including unit tests, integration

tests, and performance tests, to ensure continued reliability after updates or changes.

- **View Book Information:** The system allows users to search for books and view detailed information about them, including availability, author, genre, and publication details. It also enables users to see related books and, optionally, access book reviews if available.
    - **Performance**:
        - Response Time: The system must display detailed book information within 2 seconds of a user request, ensuring a smooth and efficient user experience.
        - Scalability: The system must handle up to 10,000 concurrent requests to view book information without sacrificing performance, meeting peak usage times.
    - **Security**:
        - Data protection: Any personal data related to book information, such as user-generated reviews, must be stored securely and in compliance with relevant data protection regulations, such as GDPR.
        - Access control: Only authenticated users have access to view detailed book information, especially for any restricted or sensitive material.
        - Audit logging: The system must maintain a log of who has accessed specific book information and when, to comply with organizational policies and for security auditing purposes.
    - **Compliance:**
        - Copyright Compliance: The system must ensure that book details, including excerpts or reviews, comply with copyright laws and only display publicly accessible content.
        - Regulatory Compliance: The display of book information must comply with any relevant regulations or standards that apply to the content, such as labeling requirements for certain genres or ratings.
    - **Reliability**:

        - Availability: The "View Book Info" function must be available round the clock, ensuring minimal downtime and uninterrupted access to book details.
        - Error Handling: If a book's information is unavailable, the system displays a user-friendly error message and provides other options to try again or report the problem.

    - **Usability**:
        - Easy to navigate: The interface for viewing book information should be intuitive, clearly labeled, and provide easy access to related books, reviews, and additional details.
        - UI consistency: The design and layout for viewing book information should be consistent with other parts of the system, ensuring a consistent experience across the platform.
    - **Localization**
        - Multiple Language Support: The system must support multiple languages to view book information, ensuring that users with different language backgrounds can access it.
    - **Maintainability**:
        - Code Modularity: The system's code base should be modular, allowing for easy updates or changes to how book information is retrieved or displayed without affecting other system functionality.
        - Documentation: Clear and comprehensive documentation should be provided, detailing how the "View Book Information" functionality is implemented, including any dependencies on external services or databases.

- **Schedule Borrow Periods:** The system allows users to select and reserve a specific time slot for borrowing a book from the library, with features for confirming, modifying, canceling, and receiving reminders about the scheduled borrow time.
    - **Performance**:

    - Response time: The system must ensure a smooth user experience without any delays when users schedule, modify or cancel a loan period within 3 seconds of their request.
    - Scalability: The system must support up to 10,000 concurrent users scheduling loan periods during peak times, without sacrificing performance.
- **Security**:
    - Data privacy: The system must protect users' personal data (e.g., student ID, contact information) during the scheduling process.
    - Access control: Only authenticated users are allowed to schedule loan periods, and only authorized users can view or manage these schedules.
    - Audit logging: All actions related to scheduling (e.g., creation, modification, cancellation) must be logged for audit purposes, ensuring accountability.
- **Compliance:**
    - Regulatory Compliance: The scheduling system must comply with relevant educational or institutional policies, such as restricting the borrowing of certain materials or enforcing maximum borrowing times
- **Reliability**:
    - Error handling: If a schedule conflict occurs, the system will immediately notify the user and suggest alternative time slots.
    - Consistency: Scheduled loan periods must be reliably reflected across all components of the system, including notifications and librarian interfaces.
- **Usability**:
    - Confirmation Feedback: The system must provide clear, immediate feedback (e.g., confirmation message) after scheduling, modifying, or canceling a loan period.
- **Interoperability**:
    - Calendar Integration: The scheduling system should support integration with popular calendar applications (e.g. Google

        Calendar, Outlook) to allow users to sync their borrowed calendars.

    - **Maintainability**:
        - Modularity: The scheduling component should be designed in a modular manner, allowing for easy updates or changes without affecting other parts of the system.
    - **Notification and Reminder System**:
        - Timeliness: Notifications and reminders related to scheduled borrow periods should be sent out promptly, ensuring that users receive them well before their scheduled time.
        - Customizability: Users should be able to customize their reminder preferences, such as choosing the time of reminder notifications or opting out of certain notifications.

- **Edit Schedule:** The system allows students and administrators to modify an existing book borrowing schedule, including changing the borrowing dates or details.
    - **Performance**:
        - Response time: The system must process and apply changes to the loan schedule within 3 seconds of the user request, ensuring a fast and responsive experience.
        - Scalability: The system must efficiently handle up to 5,000 users editing their schedules at the same time, especially during peak periods such as the beginning of the semester.
    - **Security**:
        - Data Privacy: The system must protect user data during schedule editing, ensuring that only authorized users can access changes.
        - Access Control: Only authenticated users are allowed to edit their own schedules.
        - Audit Logging: Any changes to the loan schedule must be logged for audit, including details of the changes made and the user who made them.
    - **Compliance:**

        - Regulatory Compliance: Editing functionality must comply with organizational policies.
        - Notification Requirements: Changes to borrowed calendars must trigger appropriate notifications.
    - **Reliability**:
        - Availability: The schedule editing functionality must be available 99.9% of the time to ensure users can edit their schedules without interruption.
        - Error Handling: If there is a conflict or issue with the new schedule (e.g., the new date is not available), the system must provide immediate feedback and suggest alternative options.
        - Consistency: Any changes to the loan schedule must be consistently reflected across all system components, including notifications and administrative views.
    - **Usability**:
        - User Interface: The interface should provide clear options for modifying dates. The interface should include visual cues such as calendars and drop-down lists to easily select dates.
        - Confirmation Response: After making changes to the loan schedule, the system should immediately display a confirmation message or summary of the updated details.
    - **Notifications and reminders**:
        - Timeliness: Notifications of changes to the borrowing schedule must be sent in a timely manner, ensuring that users and administrators are notified of timely updates.
        - Customizability: Users must be able to customize their notification preferences for schedule changes, including selecting different types of notifications or opting out of notifications.

- **Remove Schedule:** The system allows students and administrators to remove an existing book borrowing schedule.
    - **Performance**:

        - The calendar deletion action should be processed in less than 2 seconds, ensuring minimal latency and a smooth user experience.
    - **Security**:
        - Only authorized users (students with private schedules and administrators with schedules) can delete schedules. The system should implement appropriate authentication and authorization mechanisms.
    - **Logging and auditing:**
        - All actions related to calendar deletion should be logged with timestamp, user ID, and details of the deleted schedule for auditing.
    - **Reliability**:
        - The system should ensure that schedules are permanently deleted from the system without leaving any data behind. Additionally, the system should confirm the action with the user to avoid accidental deletion.
    - **Usability**:
        - The system should provide an intuitive and user-friendly interface for both students and administrators to easily find and delete loan schedules without much guidance or training.

- **Load Schedule:** The system enables to load and display the book borrowing schedules from the database, providing information on current and upcoming book borrowings.
    - **Performance**:
        - The system must load and display the loan calendar within 3 seconds, even during peak usage times. The loading process must be optimized to handle large data sets efficiently.
    - **Security**:
        - Access to calendar data must be restricted based on user role. Only authorized users can view certain details, with sensitive information protected under the system's security policy.
    - **Scalability:**

        - The system must be able to handle an increasing number of calendars as the library’s user base grows. The system must maintain performance and reliability as more calendars are added over time.
    - **Reliability**:
        - The system must consistently retrieve accurate and up-to-date calendar information from the database. The system must handle data retrieval errors gracefully, providing meaningful feedback to users if problems occur.
    - **Usability**:
        - Calendar information must be presented in a clear and organized manner, allowing users to quickly understand current and upcoming loans.

- **Save Schedule:** The system enables to save the borrowing schedule data into the database, ensuring that all borrowing arrangements are properly recorded and managed.
    - **Performance**:
        - The system must save the borrowed schedule data to the database within 2 seconds to ensure a smooth user experience, even under heavy load.
    - **Security**:
        - Saving the schedule data must comply with the system's security policies, ensuring that only authorized users can save or modify the schedule. Data encryption must be used during transmission and storage to protect sensitive information.
    - **Scalability:**
        - The system must be able to handle increasing numbers of schedule save operations as the user base and schedule volume increase without sacrificing performance.
    - **Reliability**:
        - The system must save the schedule data reliably even in the event of network interruptions or server failures. The system

must include mechanisms to retry the save operation or notify the user of any problems that prevent the data from being saved.

- **Data Integrity**:
  - The system must ensure that the borrowed schedule data is recorded accurately in the database without corruption or loss.
- **Auditability**:
  - The system must record all schedule save operations, recording who made the changes and when. This audit trail ensures accountability and helps diagnose any issues or discrepancies in schedule management.

- **Add Book:** The system allows librarians (administrators) to add new books to the library management system. This feature ensures that the library's book collection is updated with the latest additions and maintains accurate records for students to access.
  - **Performance**:
    - The system should allow librarians to add new books to the library database within 3 seconds. The performance must remain consistent regardless of the number of entries in the database.
  - **Security**:
    - Only authorized librarian users (administrators) should have access to the "Add Book" feature. Data entered into the system should be encrypted during transmission and storage to protect sensitive information.
  - **Scalability:**
    - The system must be able to handle a growing volume of book entries as the library expands its collection. The addition of new books should not impact the overall system performance or response time.
  - **Consistency**:
    - The system must ensure that once a new book is added, it is immediately available in all relevant views, such as search

results and browsing categories. Any discrepancies should be automatically flagged and resolved.

- **Data Integrity**:
  - The system must ensure that all book information is accurately recorded in the database. If a data entry error occurs, the system should provide clear error messages and prevent incomplete or incorrect data from being saved.
- **Auditability**:
  - The system should log all book additions, including details such as who added the book, when it was added, and any subsequent changes to the book's information. This audit trail helps in tracking changes and ensuring accountability.

- **Remove Book:** The system allows authorized users to remove a book from the library's inventory.
  - **Performance**:
    - The system must allow authorized users to delete a book from the library's inventory within 2 seconds. Response times must be consistent regardless of the size of the library database.
  - **Security**:
    - Only authorized users, such as librarians or administrators, can delete books.
  - **Scalability:**
    - After a book is deleted, the system must immediately update all related views and records, such as search results, inventory lists, and borrowing history, to reflect the deletion
  - **Reliability**:
    - The system must ensure that the deletion process is reliable and completes without errors. If an error occurs during the deletion process, the system must handle the error gracefully, ensuring that the book is completely deleted or the process is rolled back without leaving the system in an inconsistent state.
  - **Data Integrity**:

        - The system must ensure that after a book is deleted, all related records (e.g., borrowing history, reservations) are properly processed to avoid any data inconsistencies. The system must store the data or update related records accordingly.
    - **Auditability**:
        - The system must record every deletion of a book, including who deleted it, when it was deleted, and why. Administrators must be able to access this audit log for review and compliance.
    - **Maintainability:**
        - The system should be allowing for future modifications or enhancements without affecting the overall functionality of the system.

- **Edit Book:** The system allows authorized users to edit or update the information of a book in the library's inventory.
    - **Performance**:
        - The system must allow authorized users to update book information within 2 seconds, even when the system is under heavy load. This process must be smooth and responsive, ensuring minimal latency.
    - **Security**:
        - Only authorized users, such as librarians or administrators, can edit books.
    - **Error Handling:**
        - The system must include robust error handling to manage any issues that arise during the editing process, provide clear notifications to users, and log errors for further investigation.
    - **Consistency**:
        - Once a book's information is edited, all system views and records must be updated immediately to reflect the changes. This includes search results, book detail pages, and any reports or analytics related to the edited book.
    - **Data Integrity**:

        - Any edits must maintain the integrity of associated data, such as borrowing and reservation records.
    - **Auditability**:
        - Any edits made to book information must be logged, recording details such as who made the change, what was changed, and when the change occurred.
    - **Maintainability:**
        - The system must be designed to allow for easy updating and maintenance of the "Edit Book" feature

- **Load Book Data:** The system allows the system to retrieve and display book data from the library's database for both students and administrators.
    - **Performance**:
        - The system must retrieve and display book data within 2 seconds under normal conditions and within 5 seconds during peak usage.
    - **Security**:
        - Access to book data must be controlled based on user roles. The system must prevent unauthorized users from accessing sensitive book data, ensuring that only students and administrators can retrieve and view the data.
    - **Compliance**:
        - The system must comply with all relevant data privacy and security regulations, ensuring that the process of loading and displaying book data complies with legal and organizational requirements.
    - **Usability**:
        - The interface for loading and displaying book data must be intuitive and easy to navigate, providing clear and organized information for both students and administrators. The system must provide sorting, filtering, and searching options to help users find specific data quickly.
    - **Data Integrity**:

        - Any edits must maintain the integrity of associated data, such as borrowing and reservation records.
    - **Error Handling:**
        - The system must include robust error handling to manage issues that may arise during the data loading process. Clear error messages must be provided to the user and errors must be logged for further investigation.
    - **Maintainability:**
        - The “Load Book Data” feature must be easy to maintain and update, allowing for future enhancements or bug fixes without disrupting the functionality of the system.

- **Save Book Data:** The system allows authorized users to save or update book information in the system’s database.
    - **Performance**:
        - The system must save or update book information within 2 seconds under normal conditions. During peak usage, this operation must be completed within 5 seconds, ensuring that the system remains responsive.
    - **Security**:
        - Access to the "Save Book Data" function must be restricted to authorized users, such as librarians or administrators. The system must ensure that unauthorized users cannot modify or save book information, protecting the integrity of library data.
    - **Maintainability:**
        - The “Save Book Data” feature should be designed to be easily maintained, allowing for future updates or enhancements without significant disruption to the system.
    - **Scalability:**
        - The system must be able to handle an increasing number of save or update operations as the library grows without sacrificing performance. The database design must support efficient updates even as the volume of data increases.
    - **Error Handling:**

        - If an error occurs, the system should provide clear notification to the user, log the error, and ensure that incomplete or incorrect data is not saved.
    - **Availability:**
        - The system must ensure that the “Save Book Data” feature is available 99.9% of the time, minimizing downtime so that authorized users can save or update book information whenever necessary.
    - **Data Integrity**:
        - The system must ensure that all book data saved or updated in the database is accurate, consistent, and complete. Data validation must be performed before saving to avoid errors or inconsistencies.

- **Add Student:** The system allows authorized administrators to add a new student to the system, enabling them to access and use the library's resources.
    - **Performance**:
        - The system must process and save new student records within 3 seconds of submission to ensure a responsive user experience.
    - **Security**:
        - The system must restrict access to the "Add Student" functionality to authorized administrators only.
    - **Availability:**
        - The system should be available and functional 24/7, with minimal downtime, to allow administrators to add students at any time.
    - **Data Integrity**:
        - The system should prevent duplicate student records by verifying against existing entries.
    - **Maintainability:**
        - The system should be designed to be easily maintainable, allowing administrators to update or modify student records efficiently without requiring significant changes to the system.

  - **Compliance**:
    - The system must comply with relevant data protection laws, such as GDPR or CCPA, ensuring that student data is handled and stored securely, with appropriate consent and privacy measures in place.
  - **Auditability**:
    - The system must log all actions related to adding new students, including the username of the administrator, timestamp, and details of the student record added.

- **Edit Student:** The system allows administrators to edit or update information related to a student's book borrowing history and scheduled borrowing sessions.
  - **Performance**:
    - The system must update student information and related book borrowing history or scheduled sessions within 3 seconds of submission to ensure a responsive user experience.
  - **Security**:
    - The system must restrict access to the "Edit Student" functionality to authorized administrators only.
  - **Concurrency**:
    - The system must support concurrent editing of student information by multiple administrators without causing data conflicts or inconsistencies.
  - **Error Handling:**
    - The system must provide clear error messages and instructions if the editing process fails. It should handle input errors gracefully and guide users on how to correct them.
  - **Auditability**:
    - The system must log all changes made to student information, including the username of the administrator who made the edits, the timestamp, and the details of the changes. This helps in tracking modifications and ensuring accountability.
  - **Scalability**:

        - The system should be capable of handling a growing number of student records and updates without performance degradation.

- **Search Student:** The system allows administrators to search for and view detailed information about students registered in the system.
    - **Performance**:
        - The search functionality must return results within 2 seconds for common queries to ensure that administrators can quickly access student information.
    - **Maintainability**:
        - The search feature should be easy to maintain and update, allowing for modifications and improvements to search functionality without extensive rework.
    - **Integration**:
        - The search functionality should integrate seamlessly with other system components, such as the student database and user authentication system, to provide a cohesive user experience.
    - **Auditability**:
        - The system should log search activities, including the search criteria used and the user who performed the search. This audit trail helps in monitoring and reviewing access to student information.
    - **Usability**:
        - The search interface should be intuitive and user-friendly, allowing administrators to easily enter search criteria and view results. It should support various search filters (e.g., name, ID, enrollment status) and provide clear navigation for viewing detailed student information.
    - **Maintainability:**
        - The search feature should be easy to maintain and update, allowing for modifications and improvements to search functionality without extensive rework.

- **Load Student Data:** The system loads and verifies student data from the university's database for authentication and account creation purposes.
    - **Performance**:
        - Data Load Time: The system should load student data within 3 seconds to ensure quick processing for authentication and account creation.
        - Efficiency: The data retrieval process should be optimized to handle large datasets without significant delays.
    - **Data Integrity**:
        - The system must ensure that the student data loaded from the database is accurate and up-to-date. It should include validation checks to confirm the integrity of the data.
    - **Security**:
        - Data Protection: The data loading process must adhere to strict security measures to protect sensitive student information from unauthorized access. This includes encrypting data in transit and ensuring secure access controls.
        - Access Control: Only authorized systems or personnel should have the ability to initiate data loading processes.
    - **Error Handling:**
        - The system should handle errors gracefully during data loading, such as data format issues or connection problems, providing meaningful error messages and recovery options.
    - **Scalability**:
        - Handling Growth: The system should be able to scale efficiently to handle increasing volumes of student data as the number of students grows. It should maintain performance and reliability with larger datasets.
    - **Integration**:
        - Ensure seamless integration with the university's database for accurate data retrieval and loading.

- **Save Student Data:** The system allows to save and manage student data within the database, ensuring that only authorized students can access the LMS.
    - **Performance**:
        - Save Time: The system should save student data to the database within 2 seconds to ensure quick updates and minimal delays.
        - Efficiency: Data saving processes should be optimized to handle concurrent updates from multiple sources without impacting performance.
    - **Scalability**:
        - The system should be capable of scaling to accommodate increasing amounts of student data as the number of users grows, maintaining performance and responsiveness.
    - **Security**:
        - Data Protection: The system must ensure that student data is securely saved, using encryption for sensitive information both in transit and at rest. - Access Control: Only authorized users (such as administrators) should be allowed to save or update student data. Implement role-based access controls to enforce this.
    - **Compliance**:
        - Data Protection Regulations: The system must comply with relevant data protection laws, ensuring that student data is handled in accordance with legal and regulatory requirements.
        - Data Retention Policies: Adhere to organizational policies and regulations regarding data retention and deletion.
    - **Integration**:
        - Ensure seamless integration with the database for storing and retrieving student data. The system should handle various data formats and structures if needed.
    - **Auditability**: Maintain logs of data saving operations, including timestamps, user actions, and any errors encountered.

Design and implement

1. Context model:

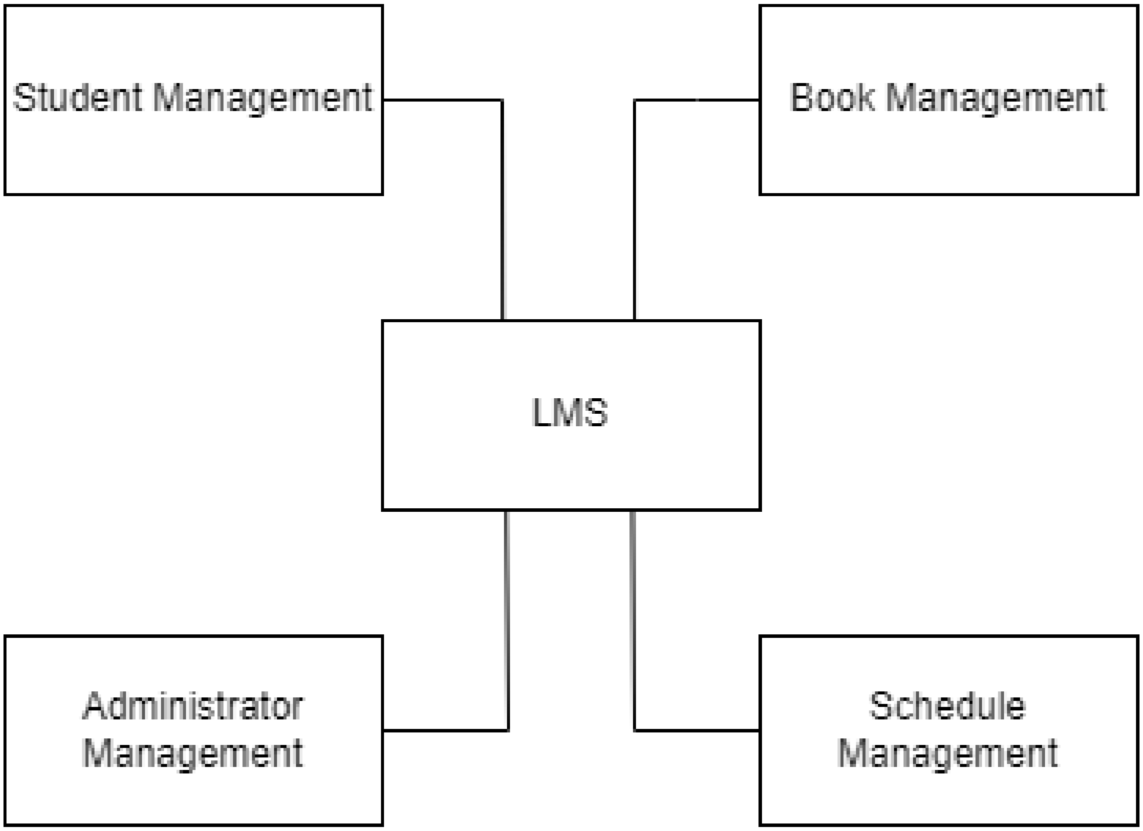


Figure 1: LMS (Context model)

2. Flowchart:

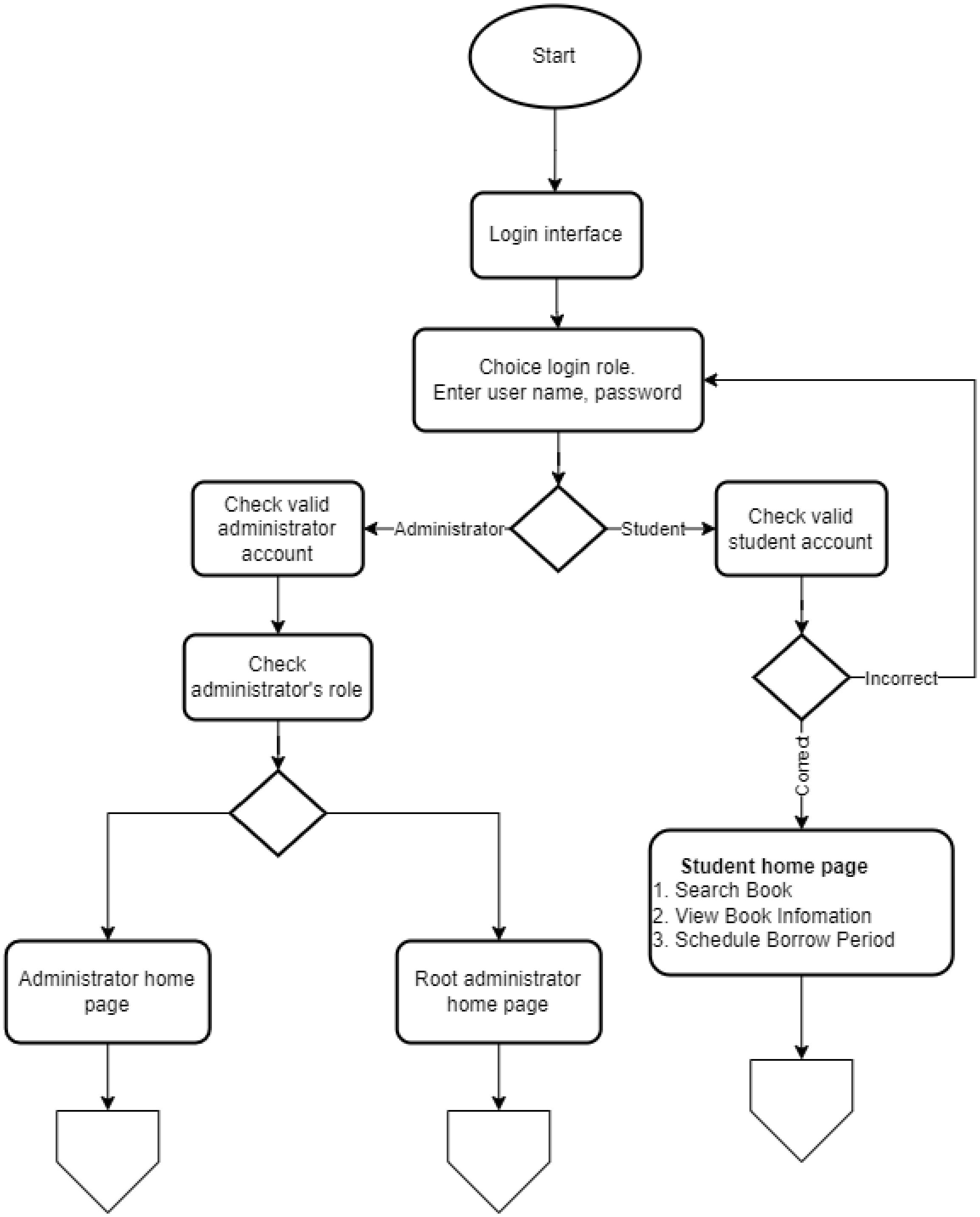

Start
Login interface
Choice login role.
Enter user name, password
Check valid administrator account
Administrator
Student
Check valid student account
Check administrator's role
Incorrect
Correct
Student home page
1. Search Book
2. View Book Infomation
3. Schedule Borrow Period
Administrator home page
Root administrator home page

Figure 2: Login (Flowchart)

Description (Figure 2)**:** This flowchart shows how to login to a system for two types of users: administrators and students:

1. **Start**: The process begins.
2. **Login Interface**: The user accesses the login interface.
3. **Choice Login Role**: The user is prompted to select their role (either Administrator or Student) and enter their username and password.
4. **Decision Point - Role Selection**:
    - If the user selects the **Administrator** role, the process moves to the left side of the flowchart.
    - If the user selects the **Student** role, the process moves to the right side of the flowchart.
5. **Administrator Path**:
    - **Check Valid Administrator Account**: The system checks if the administrator's account credentials are valid.
        - If the credentials are invalid, the process would loop back to the login interface (not explicitly shown in the chart).
    - **Check Administrator's Role**:
        - If the administrator has a standard role, they are directed to the **Administrator Home Page**.
        - If the administrator has root privileges, they are directed to the **Root Administrator Home Page**.
6. **Student Path**:
    - **Check Valid Student Account**: The system checks if the student's account credentials are valid.
        - If the credentials are incorrect, the process loops back, indicating an incorrect login attempt.

- If the credentials are correct, the student is directed to the **Student Home Page**.

7. **Student Home Page**:
   - The student can perform the following actions:
     - Search for books.
     - View book information.
     - Schedule a borrow period.

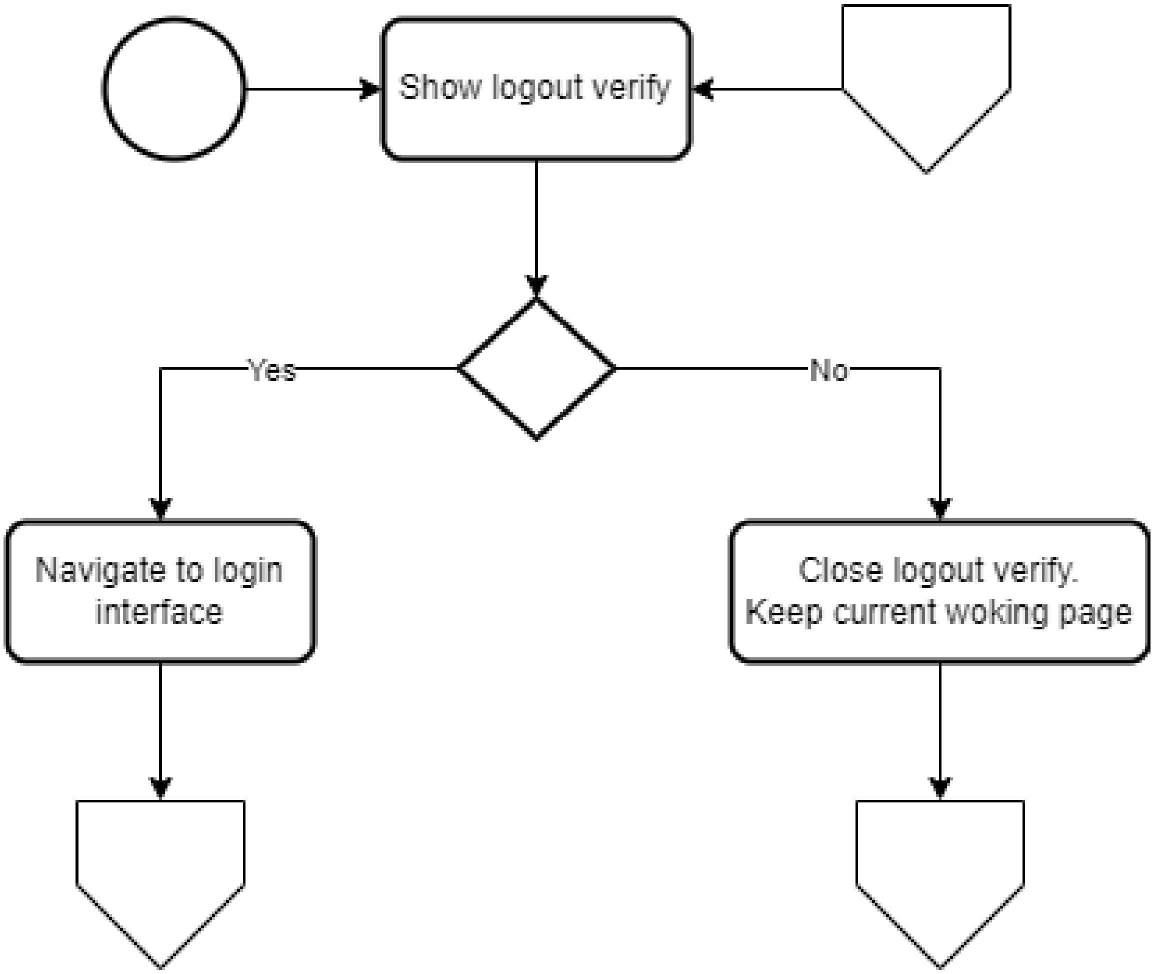

Show logout verify
Yes
No
Navigate to login interface
Close logout verify. Keep current woking page

Figure 3 Logout (Flowchart)

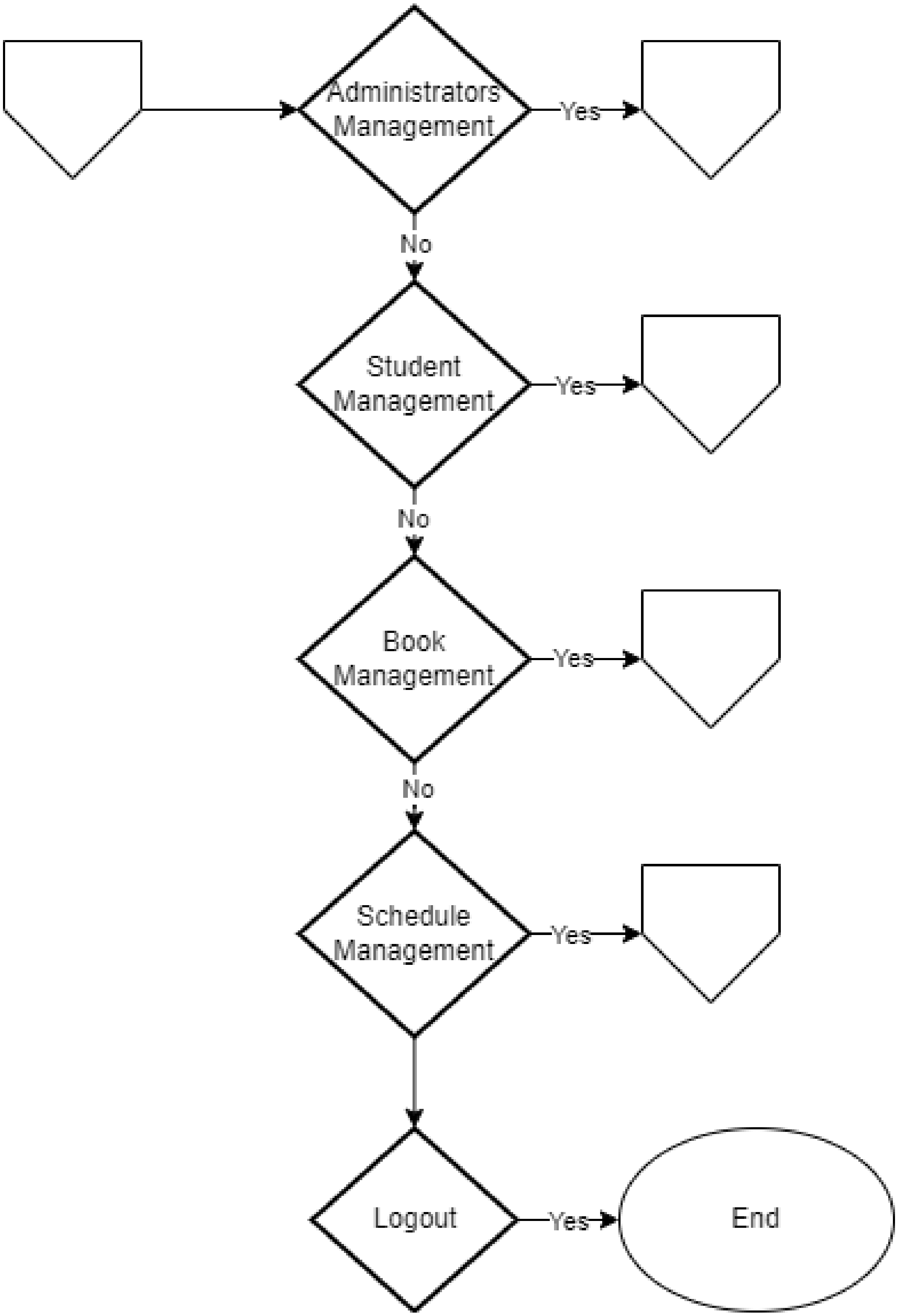


Figure 4 Root Administrator Page (Flowchart)

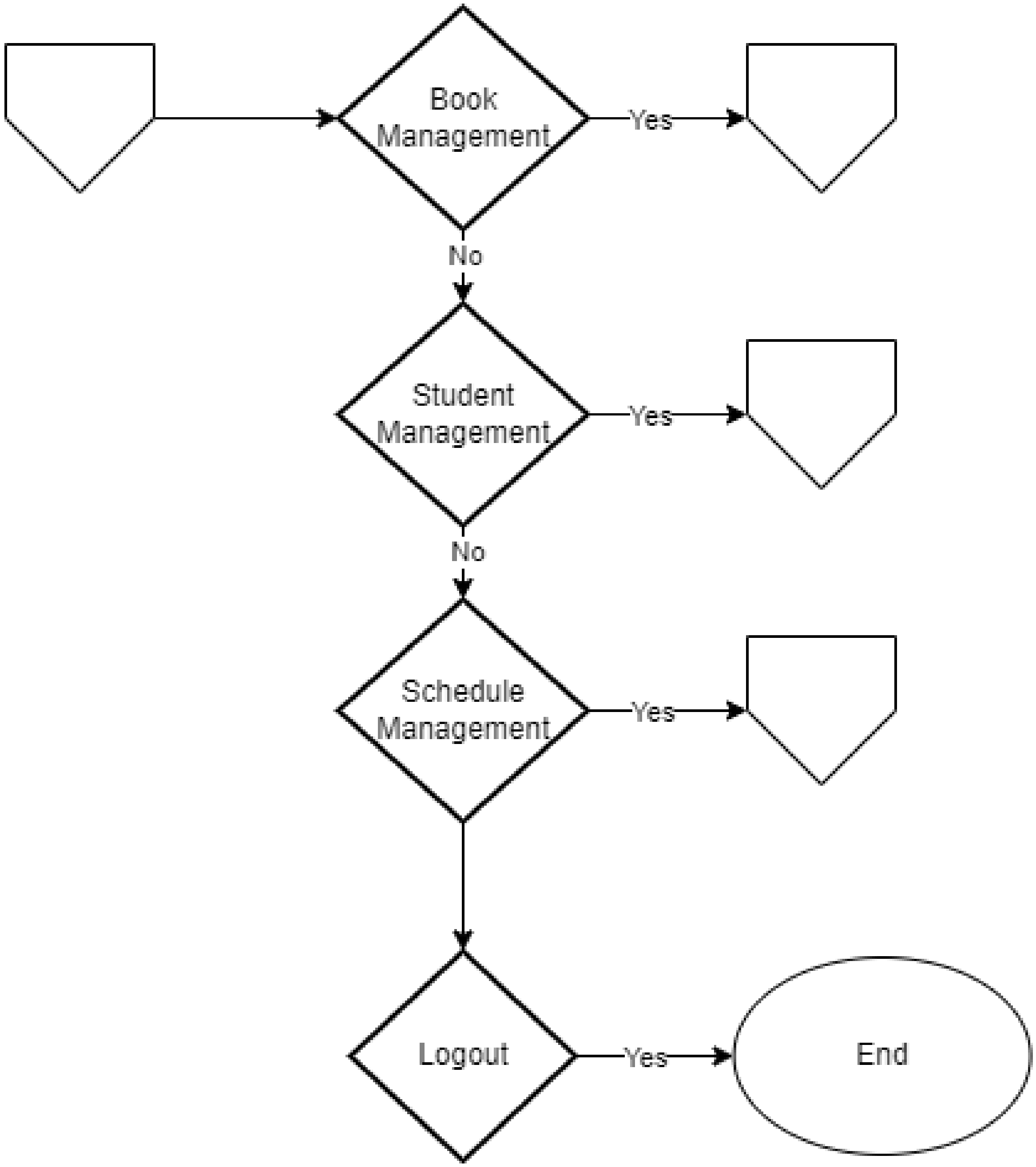


Figure 5 Administrator Page (Flowchart)

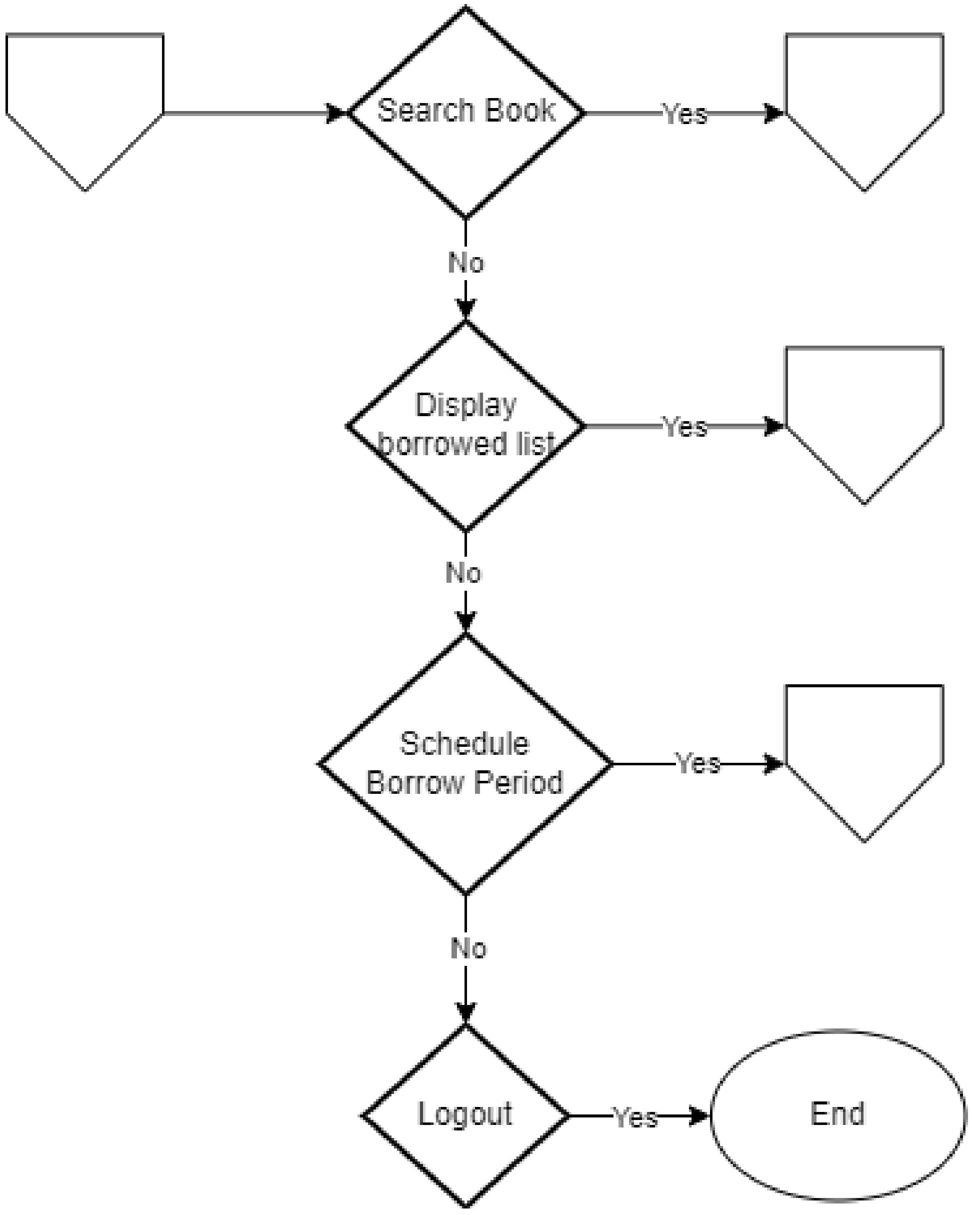


Figure 6 Student Page (Flowchart)

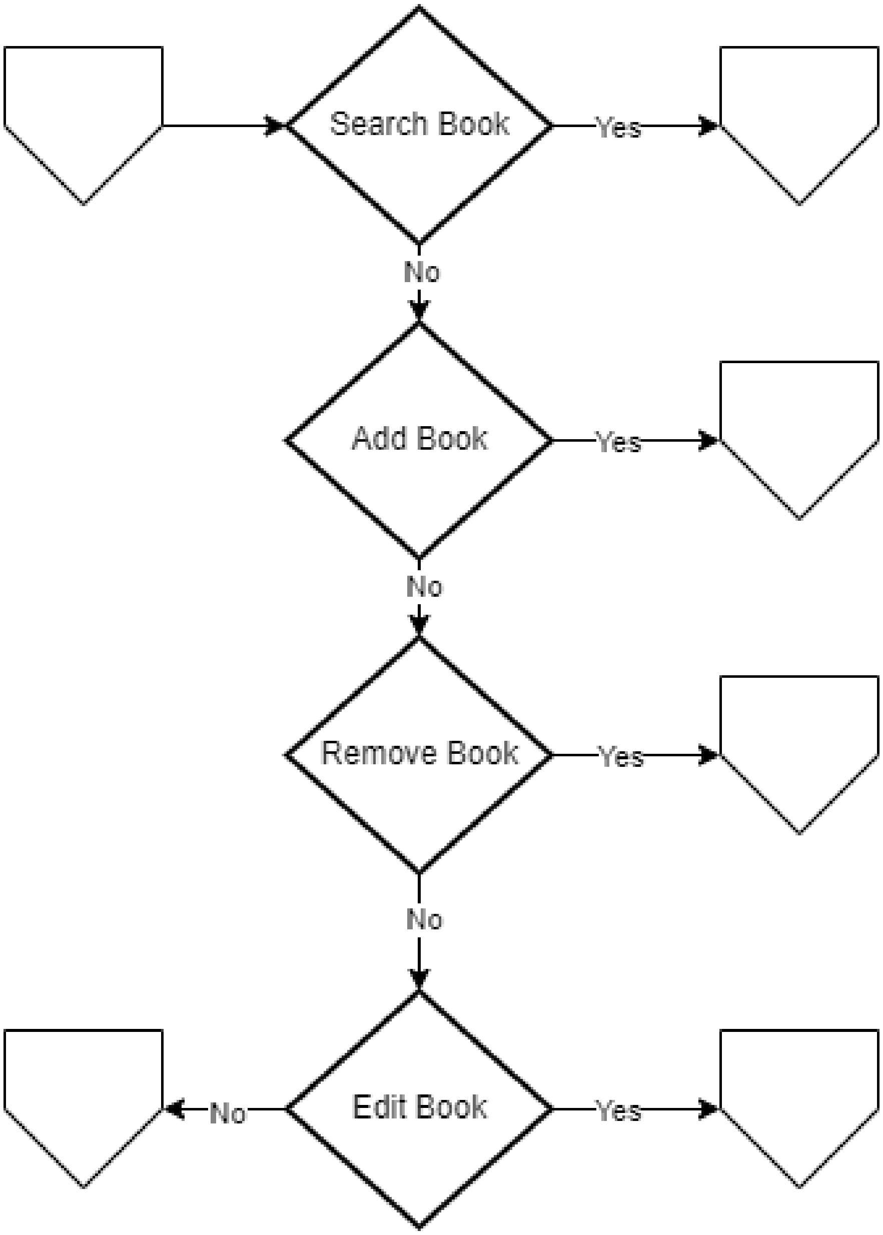


Figure 7 Book Management Page (Flowchart)

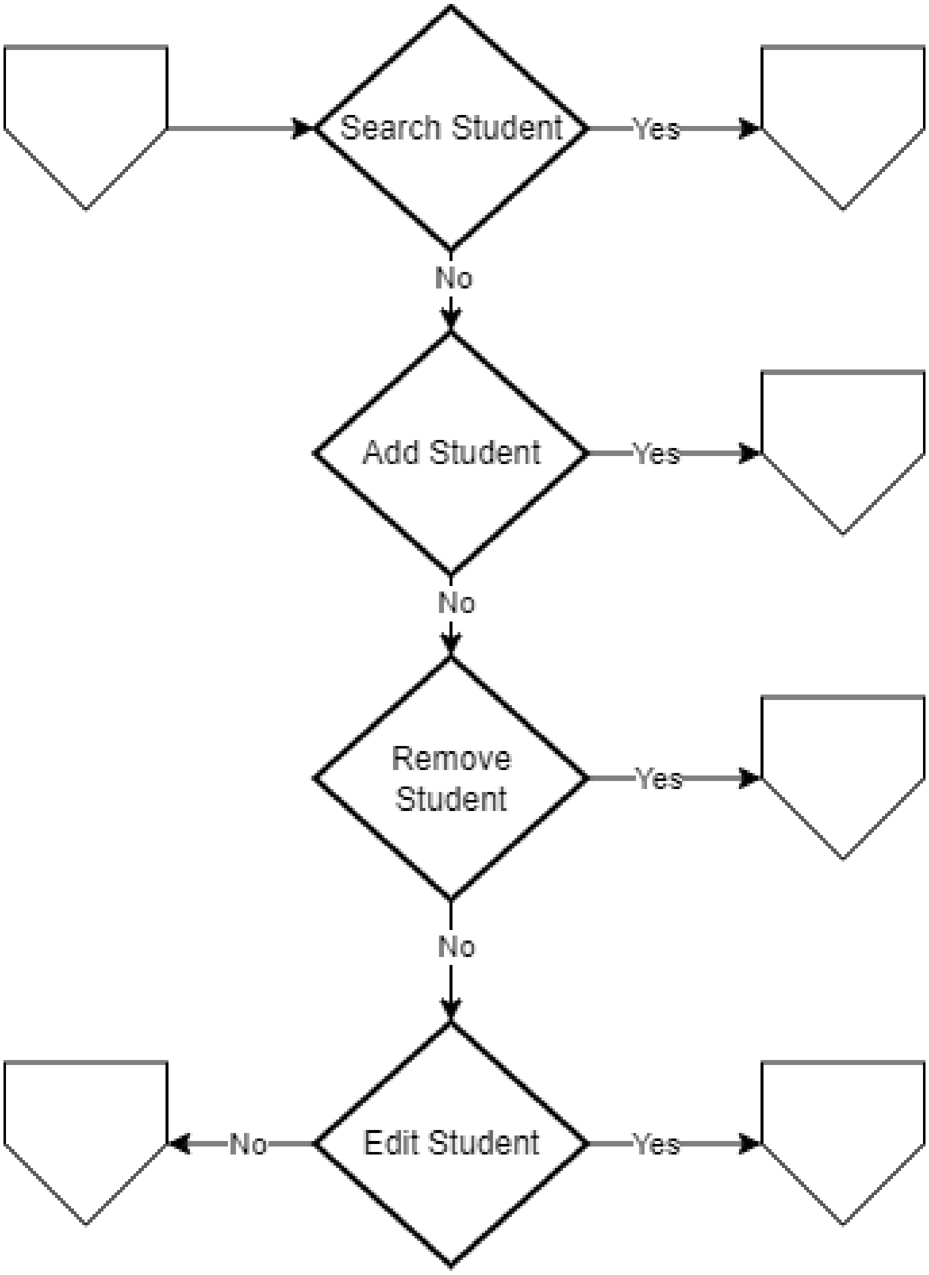


Figure 8 Student Management Page (Flowchart)

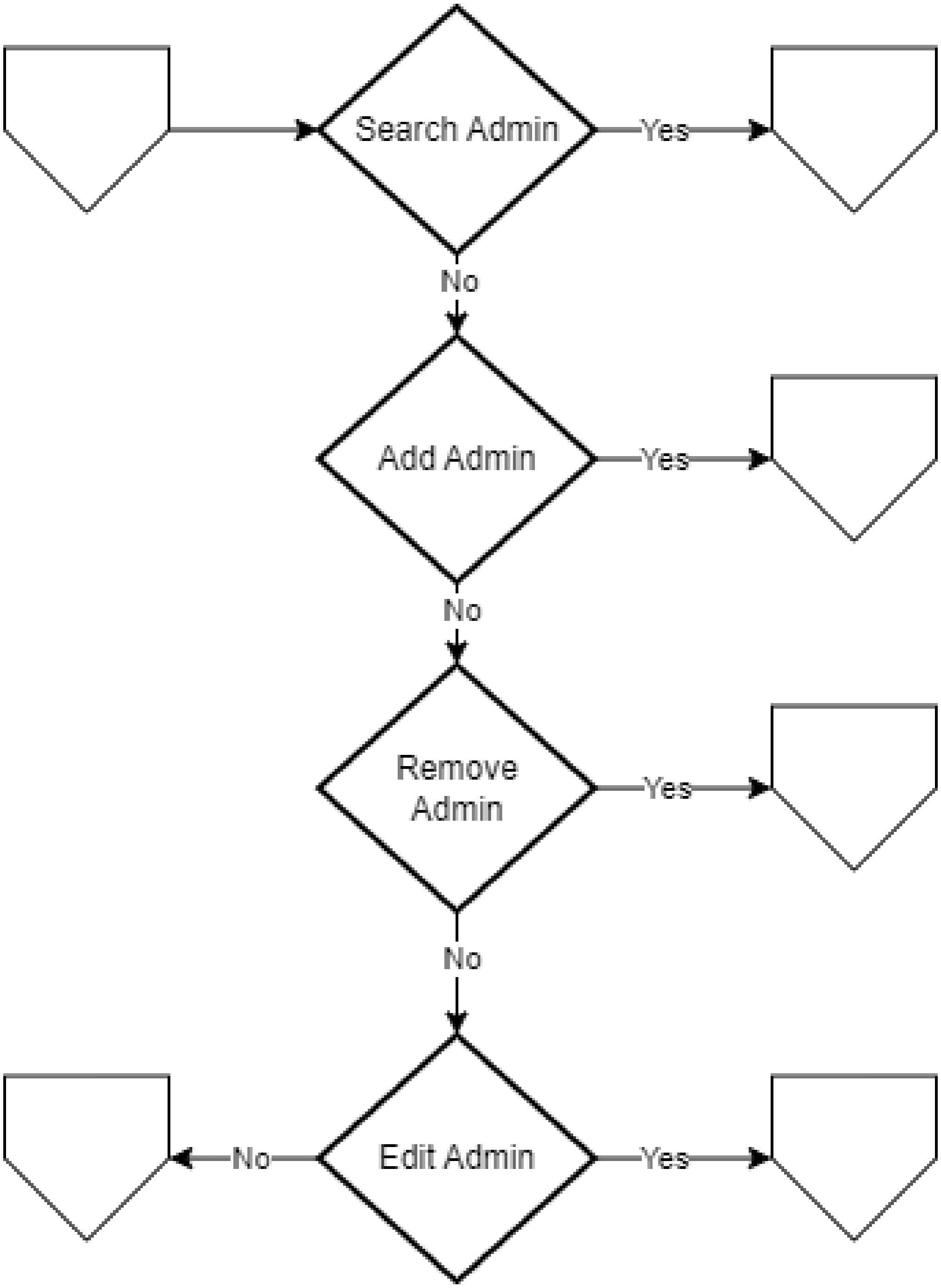


Figure 9: Administrator Management Page (Flowchart)

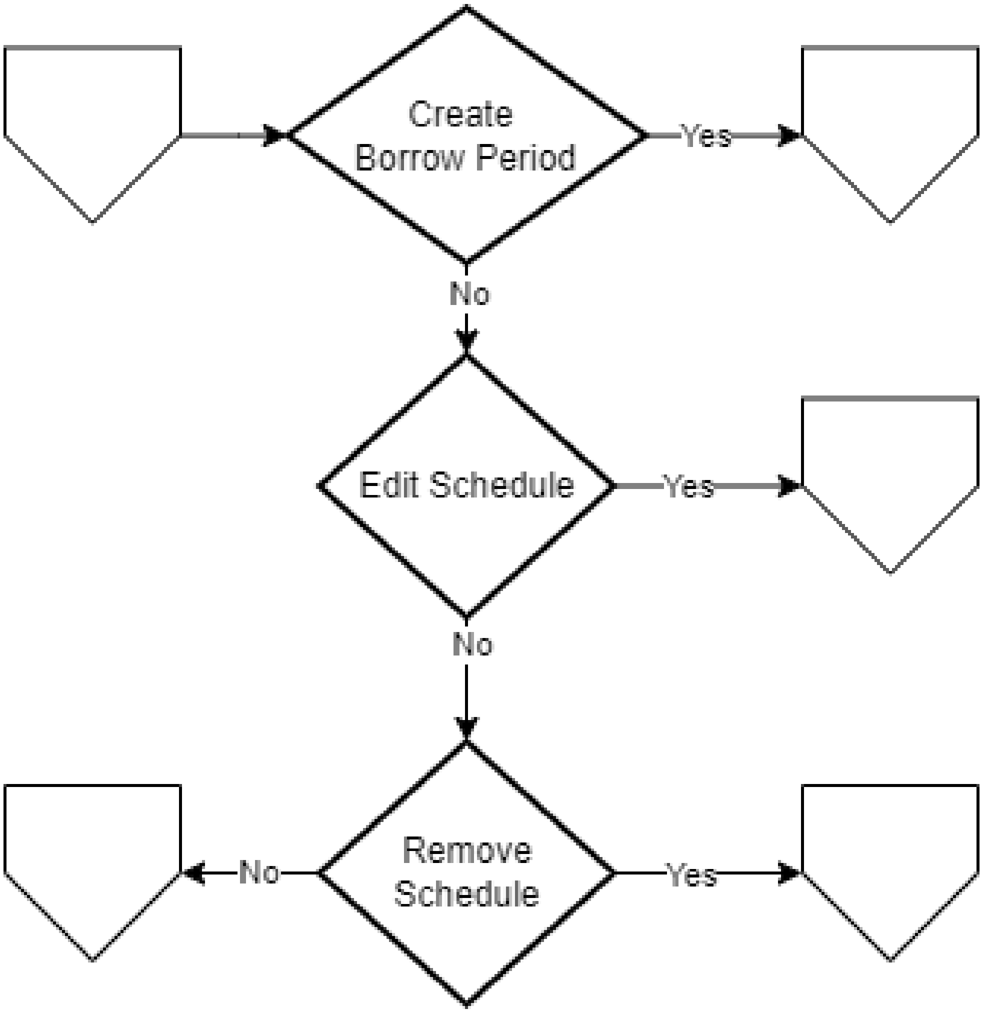


Figure 10 Schedule Management (Flowchart)

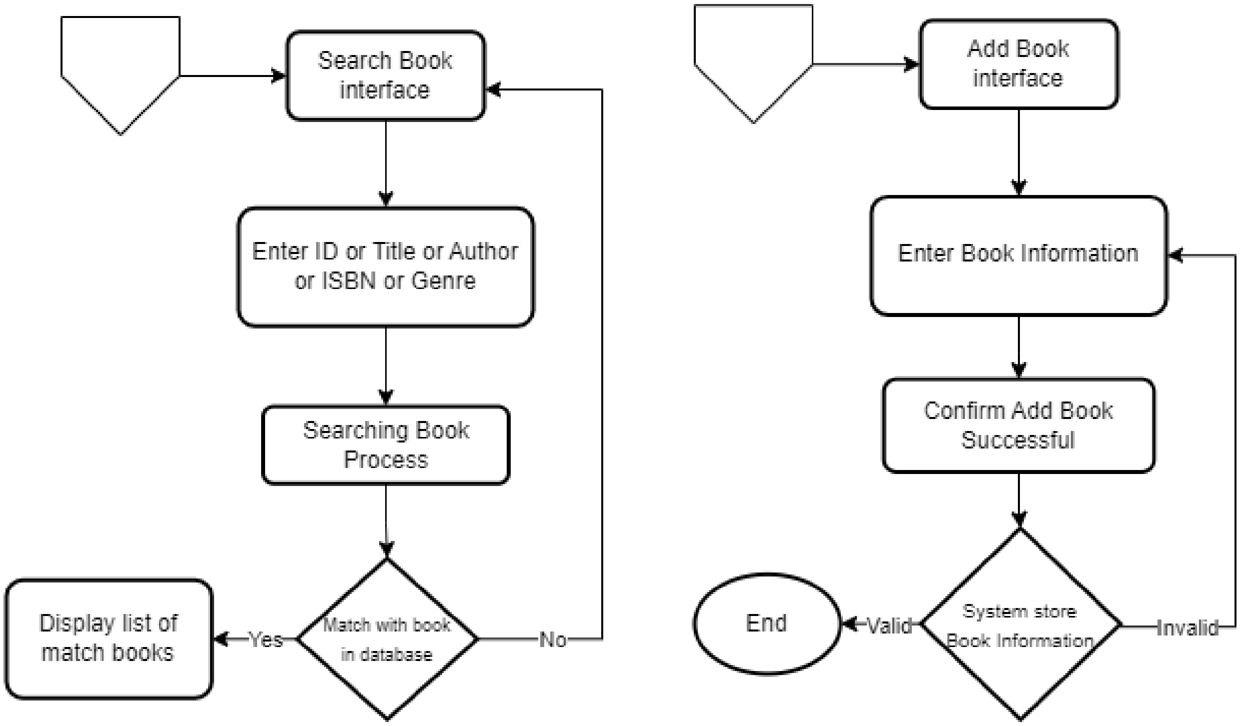

Search Book interface
Enter ID or Title or Author or ISBN or Genre
Searching Book Process
Match with book in database
Yes
No
Display list of match books
Add Book interface
Enter Book Information
Confirm Add Book Successful
System store Book Information
Valid
Invalid
End

Figure 11 Search, Add Book (Flowchart)

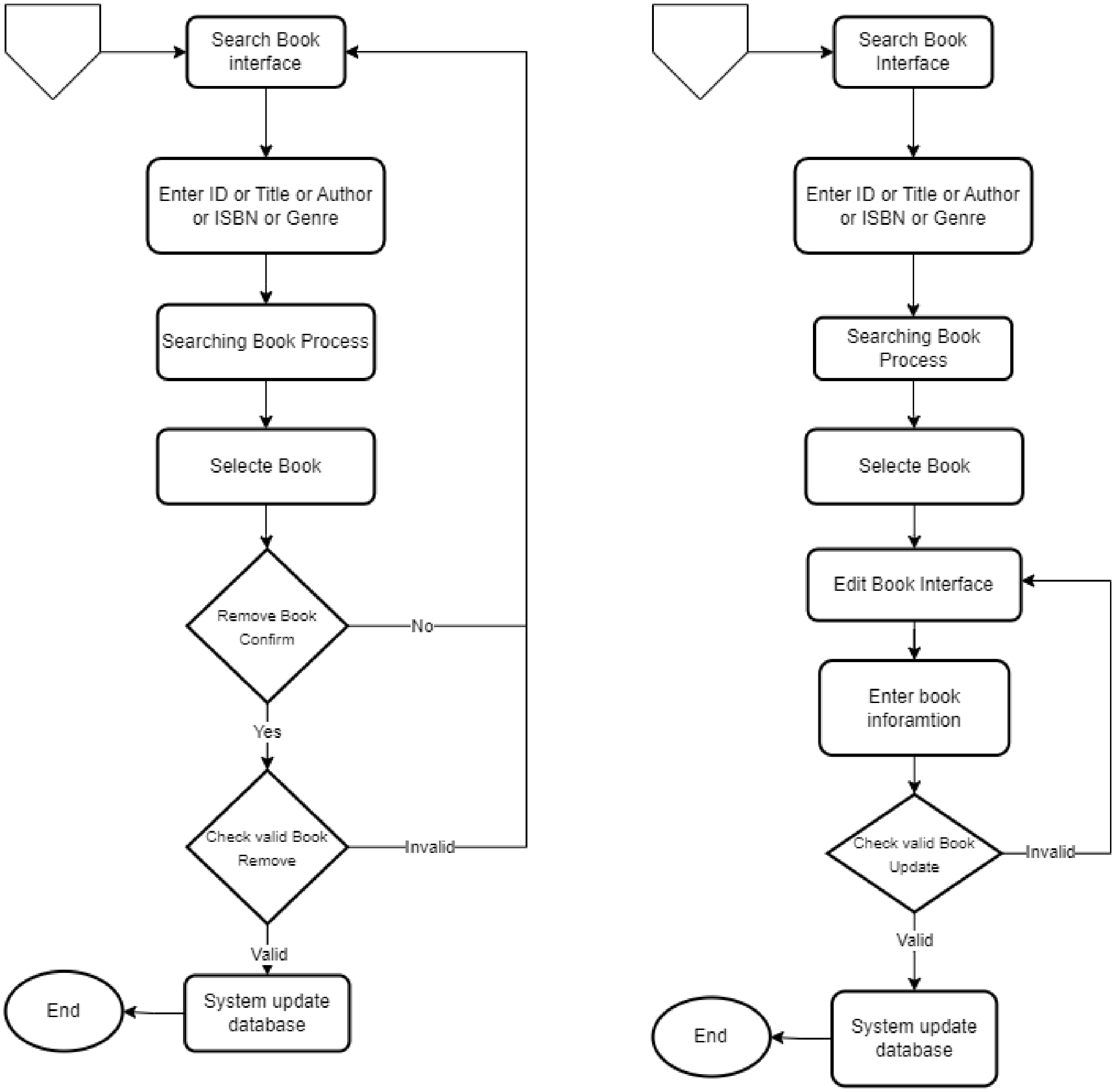


Figure 12 Remove, Edit Book (Flowchart)

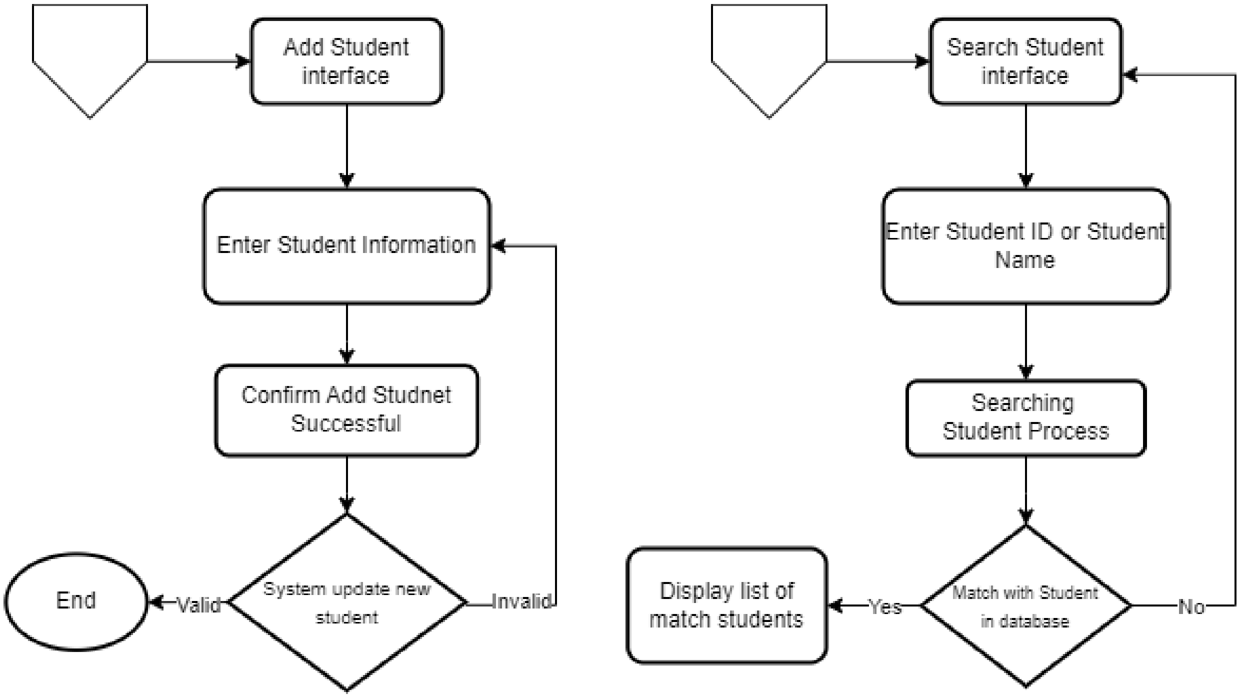

Add Student interface
Enter Student Information
Confirm Add Studnet Successful
System update new student
Valid
Invalid
End
Search Student interface
Enter Student ID or Student Name
Searching Student Process
Match with Student in database
Yes
No
Display list of match students

Figure 13 Search, Add Student (Flowchart)

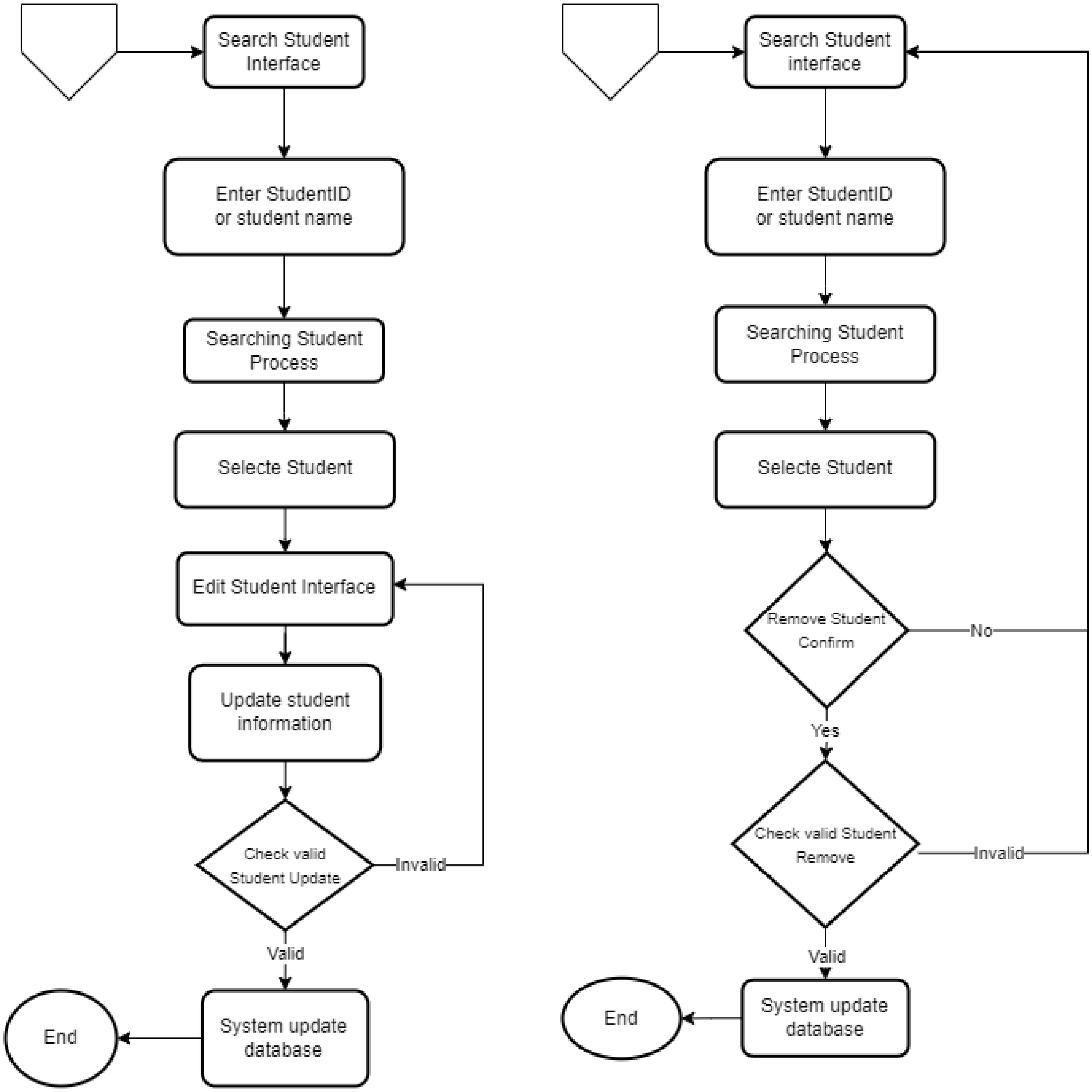


Figure 14 Remove, Edit Student flowchart

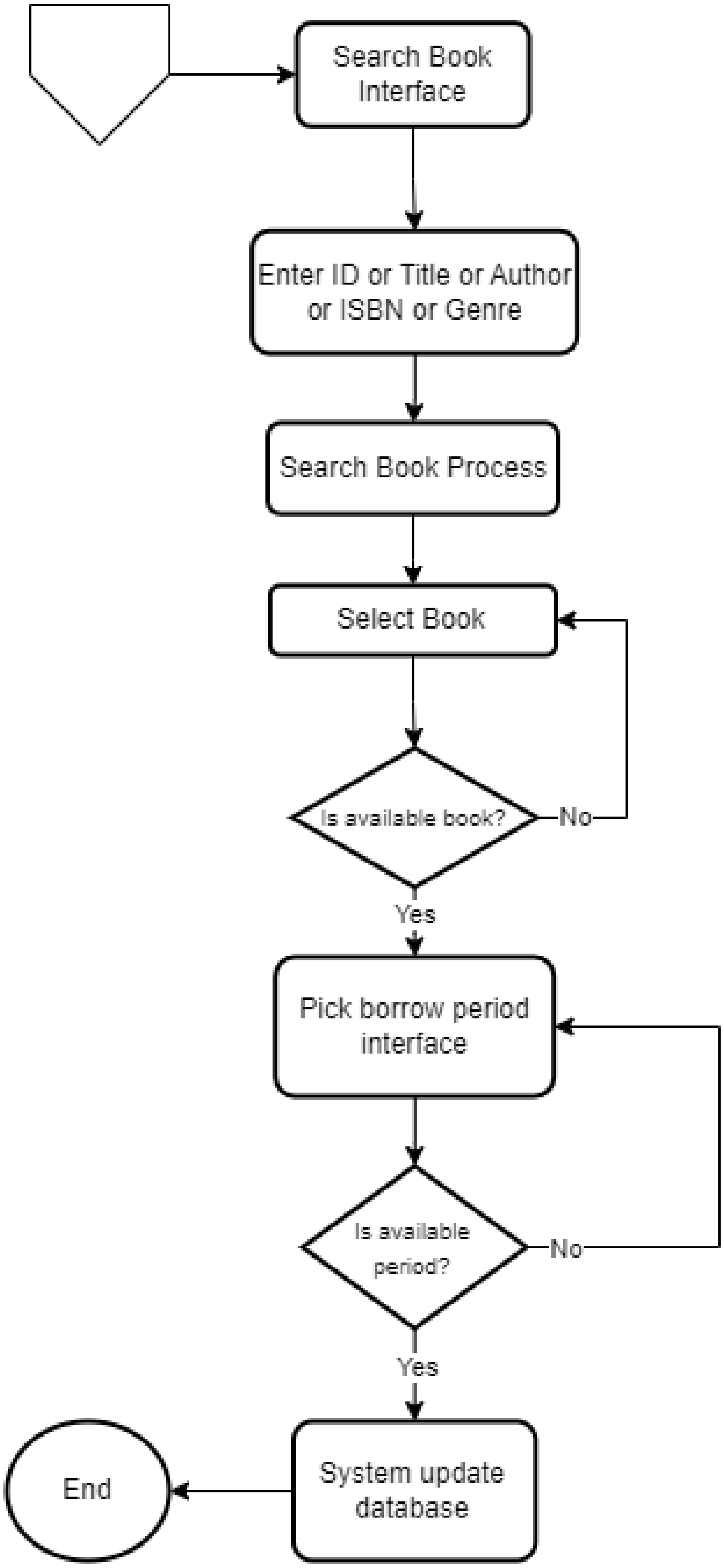


Figure 15 Create Borrow Period flowchart

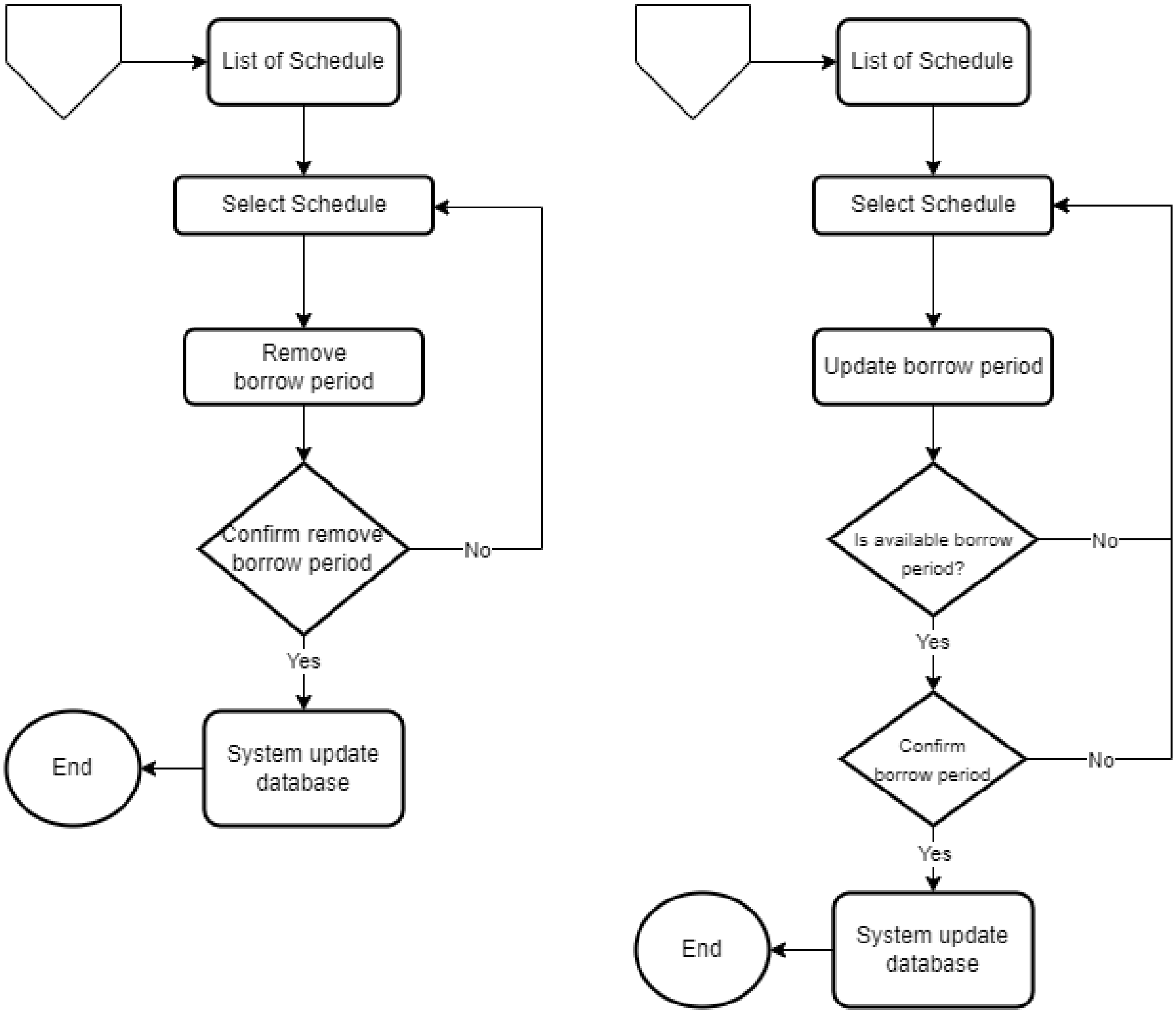


Figure 16 Remove, Edit Schedule flowchart

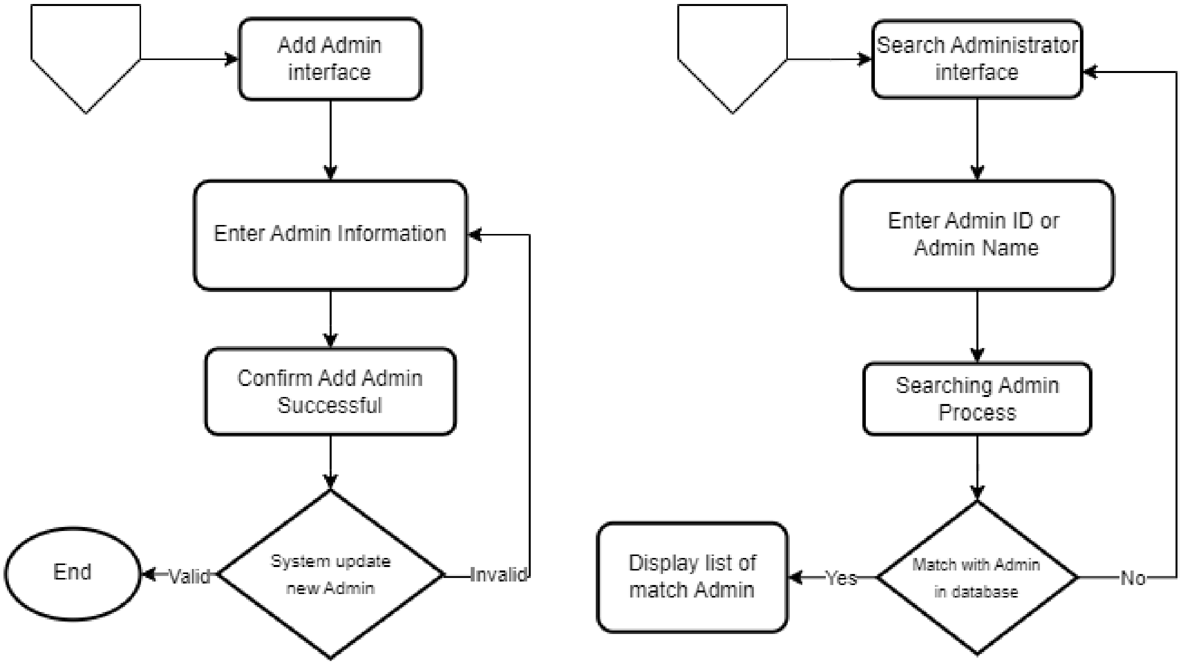

Add Admin interface
Enter Admin Information
Confirm Add Admin Successful
System update new Admin
Valid
Invalid
End
Search Administrator interface
Enter Admin ID or Admin Name
Searching Admin Process
Match with Admin in database
Yes
No
Display list of match Admin

Figure 17 Seach, Add Administrator flowchart

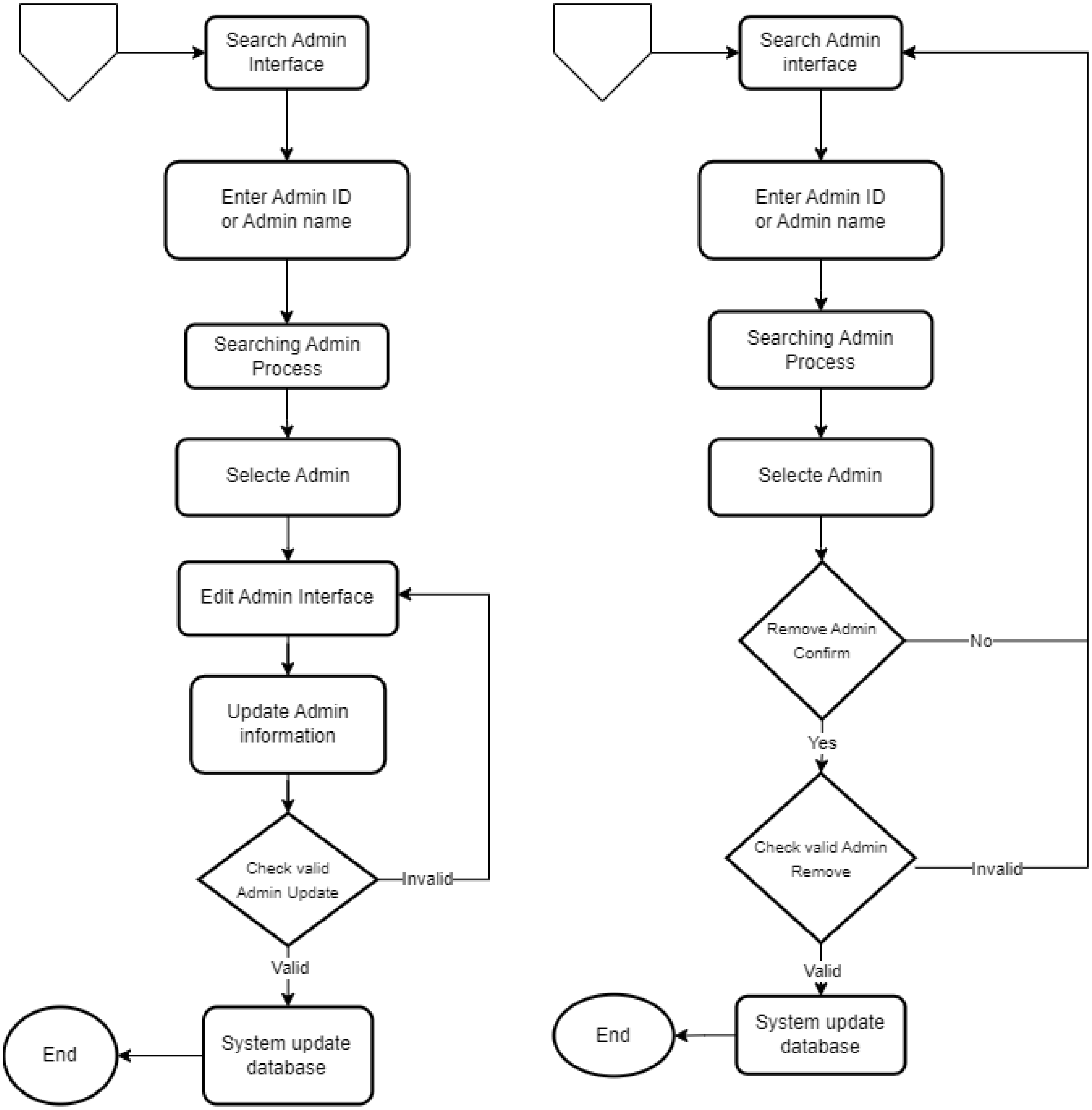


Figure 18 Edit, Remove Administrator flowchart

3. Use-case diagram:

LMS Use-case:

No	Actor	Description
1	Student	Once login, student can search for book, view book details, schedule borrow period, manage account, borrowed books list and logout
2	Normal Administrator	Once login, Administrator can access book management system, student management system, also schedule management.
3	Root Administrator	After login, Root Administrator can access all system of LMS, especially administrator management, also logout.

Table 1 LMS Use-case (Use-case diagram)

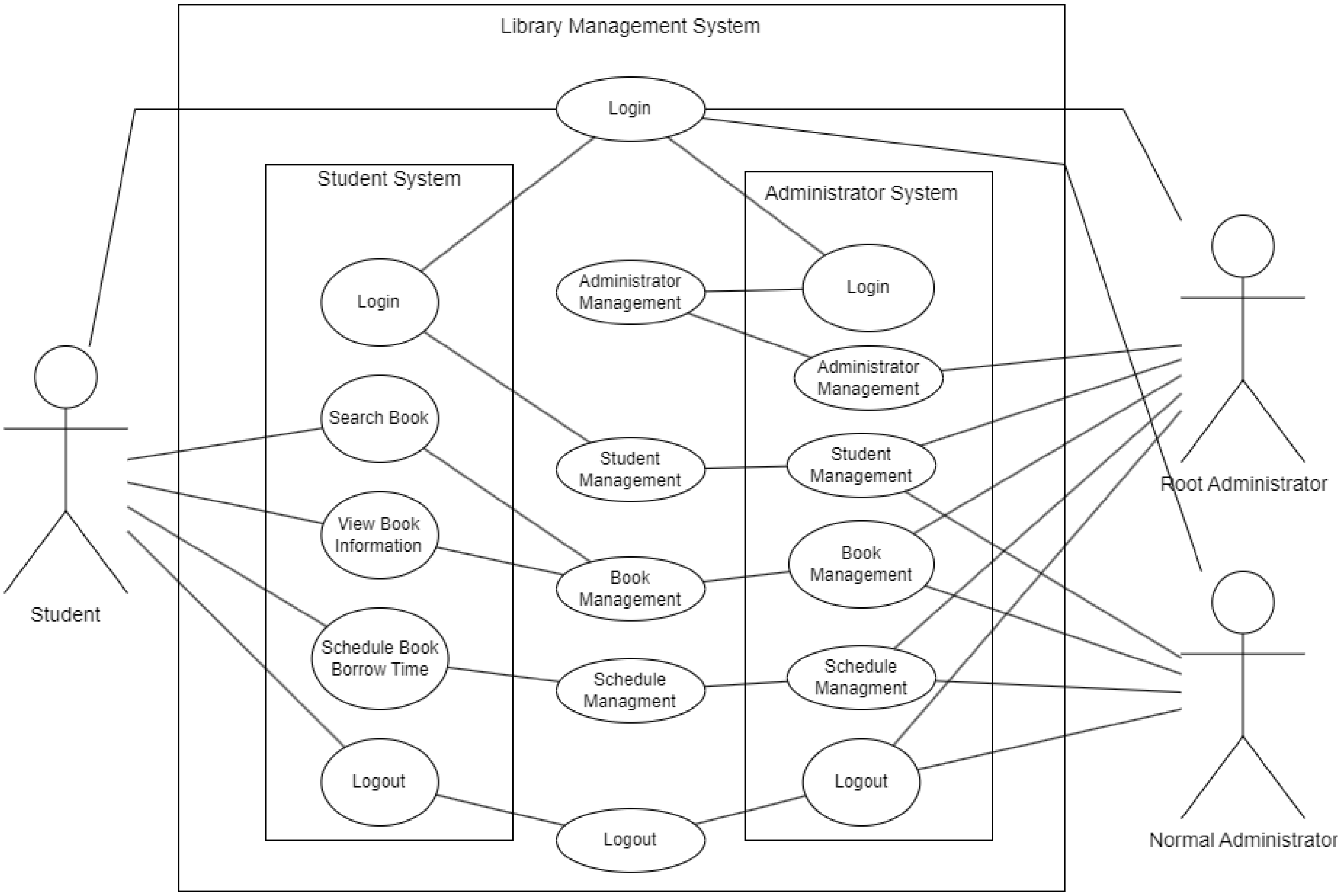


Figure 19 LMS Use-case (Use-case diagram)

Root Administrator use-case:

No	Actor	Description
1	Root Administrator	After login, Root Administrator can access all system of LMS, student management, book management, schedule management, administrator management and logout.

Table 2 Root Administrator Use-case (Use-case diagram)

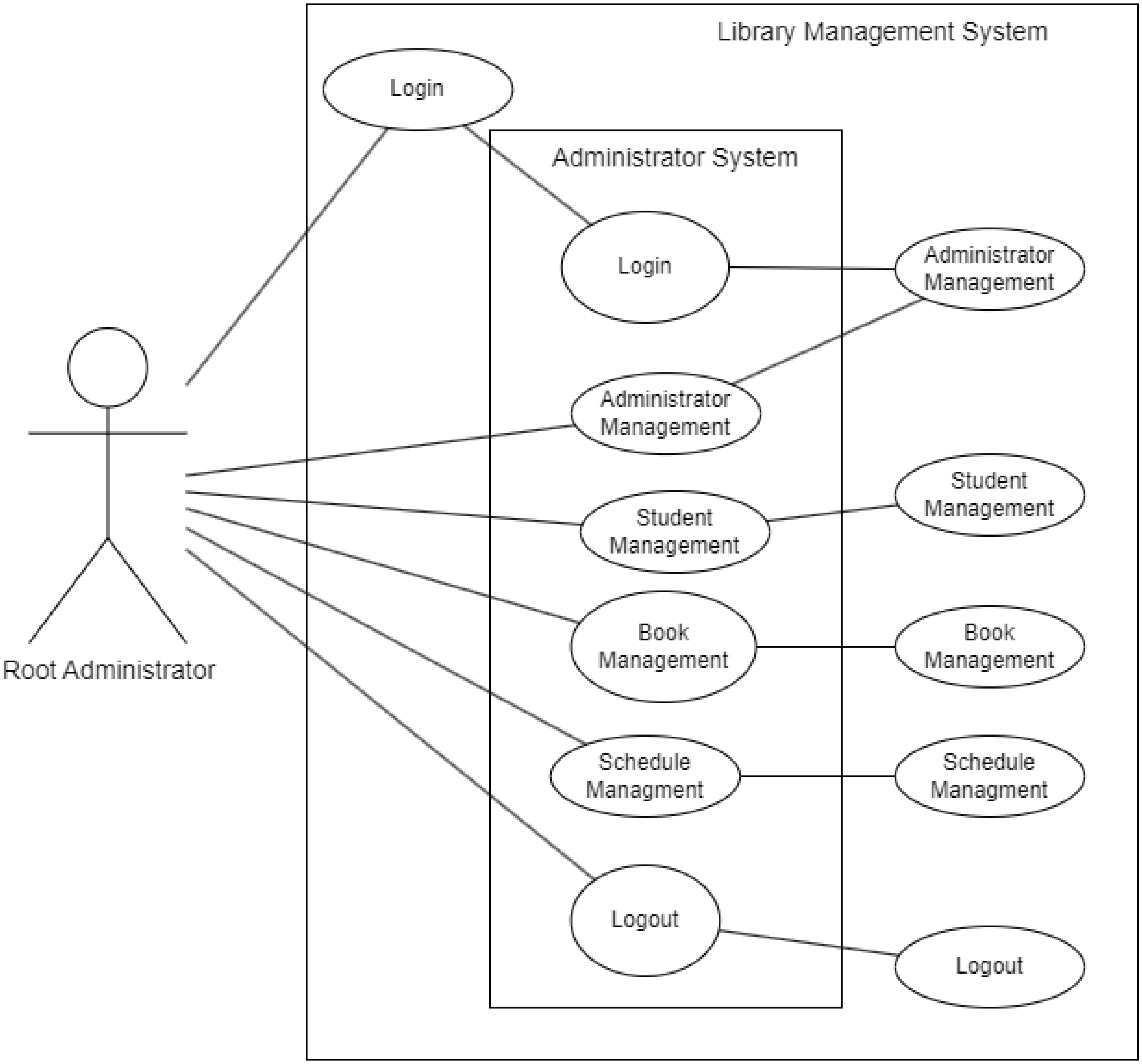


Figure 20 Root Administrator use-case (Use-case diagram)

Normal Administrator use-case:

No	Actor	Description
1	Normal Administrator	Once login, Administrator can access all system of LMS, except administrator management system.

Table 3 Normal Administrator Use-case (Use-case diagram)

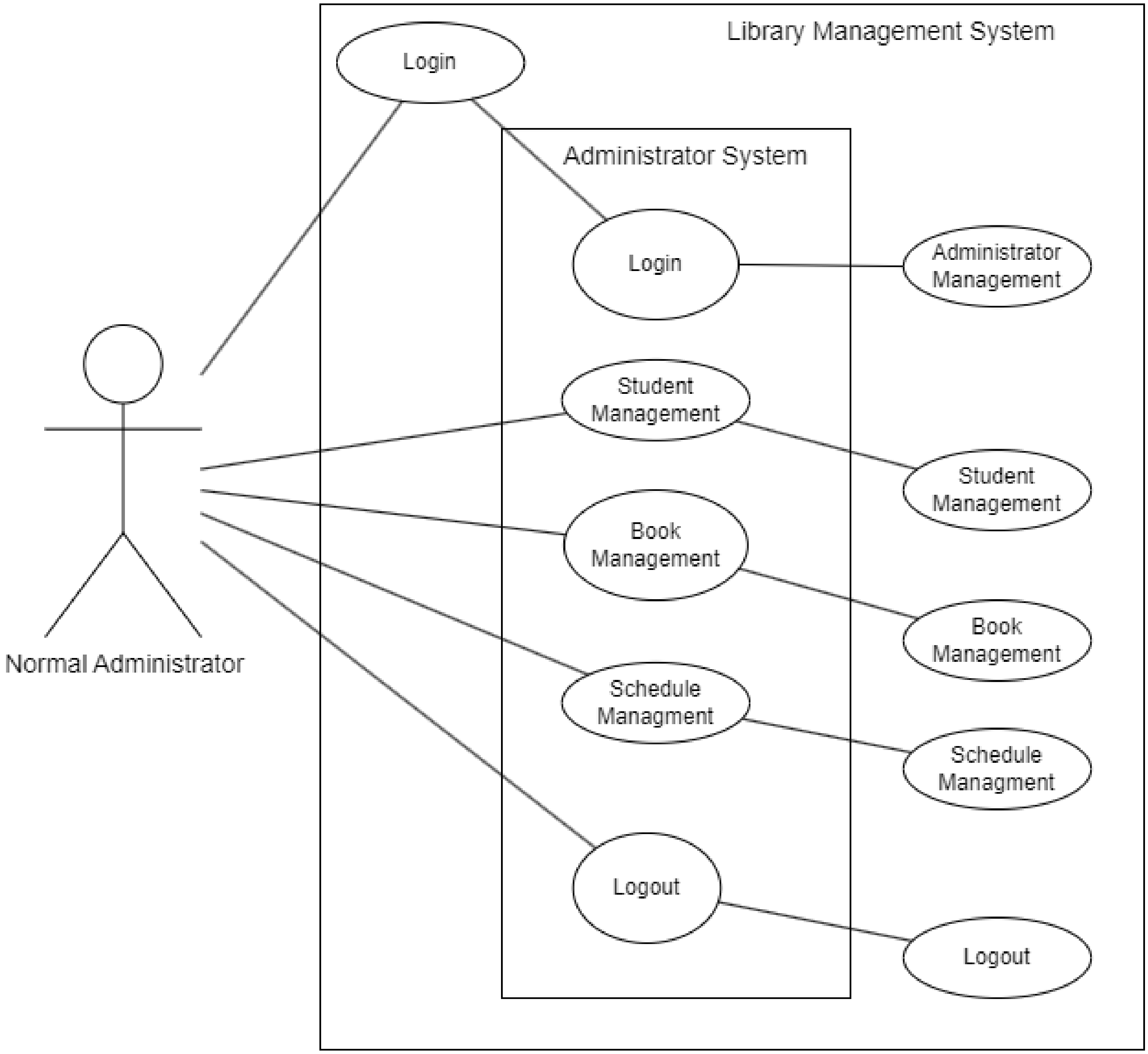


Figure 21 Normal Administrator use-case (Use-case diagram)

Student use-case:

No	Actor	Description
1	Student	Once login, Student can access some service: Search Book, View Book Information, Create, Remove, Eidt Schedule, and Logout.

Table 4 Student use-case (Use-case diagram)

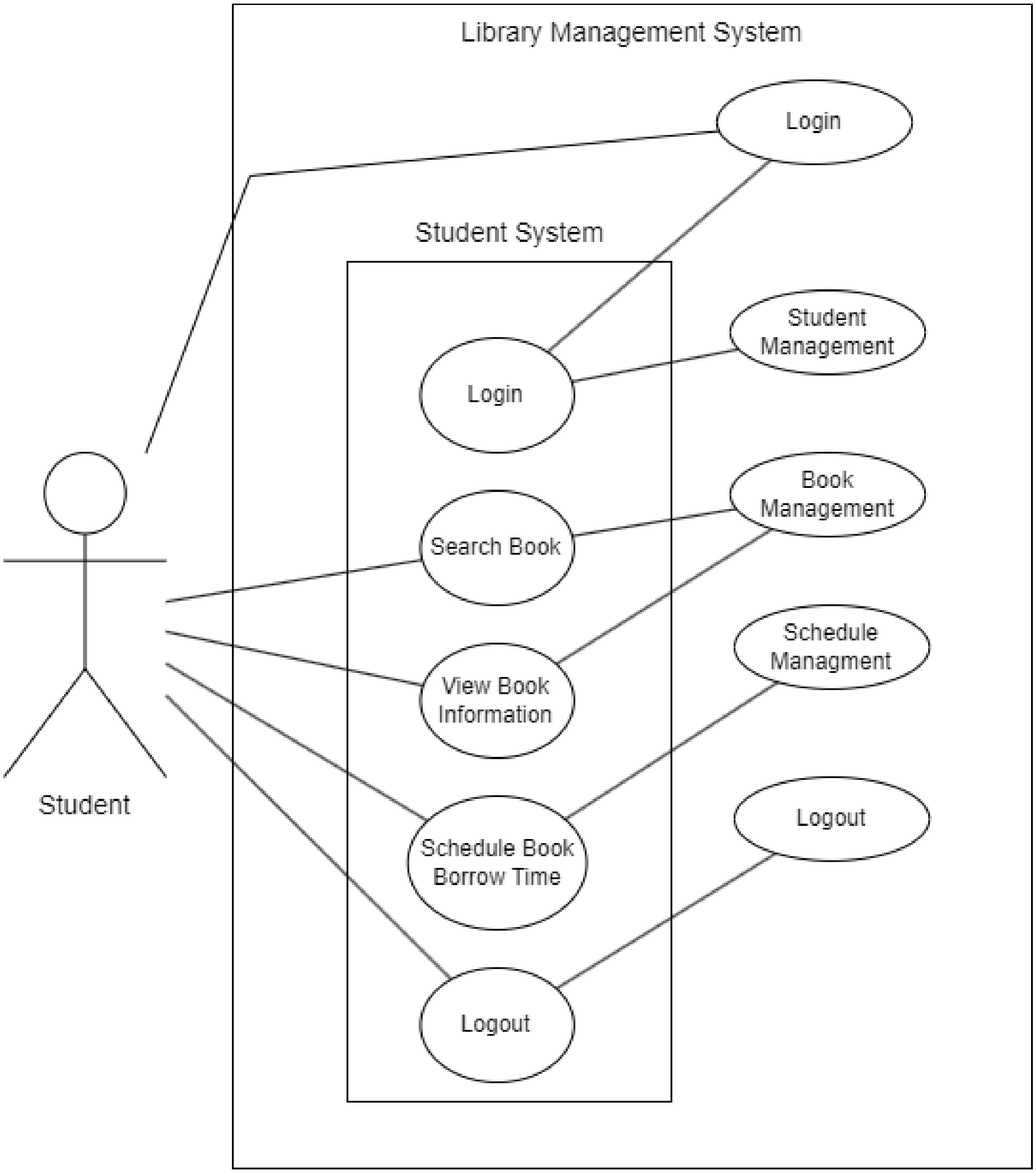


Figure 22 Student use-case (Use-case diagram)

4. Class diagram

Overall:

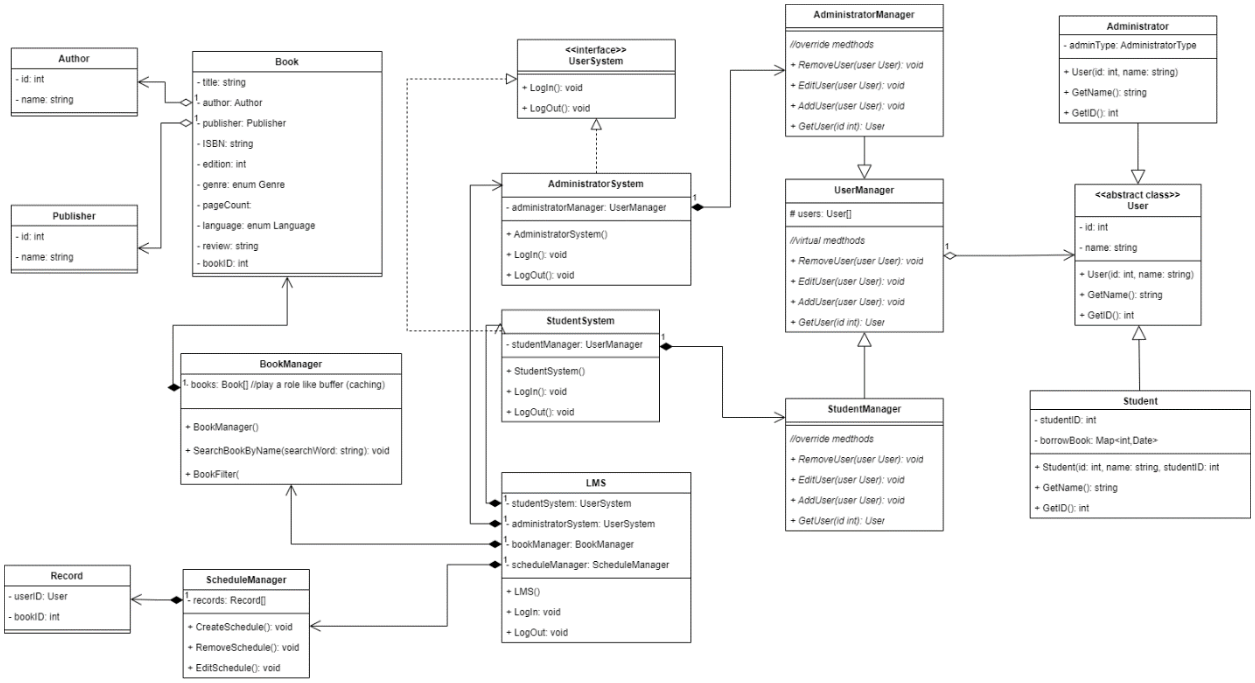


Figure 23 LMS Class diagram (Class diagram)

Specification:

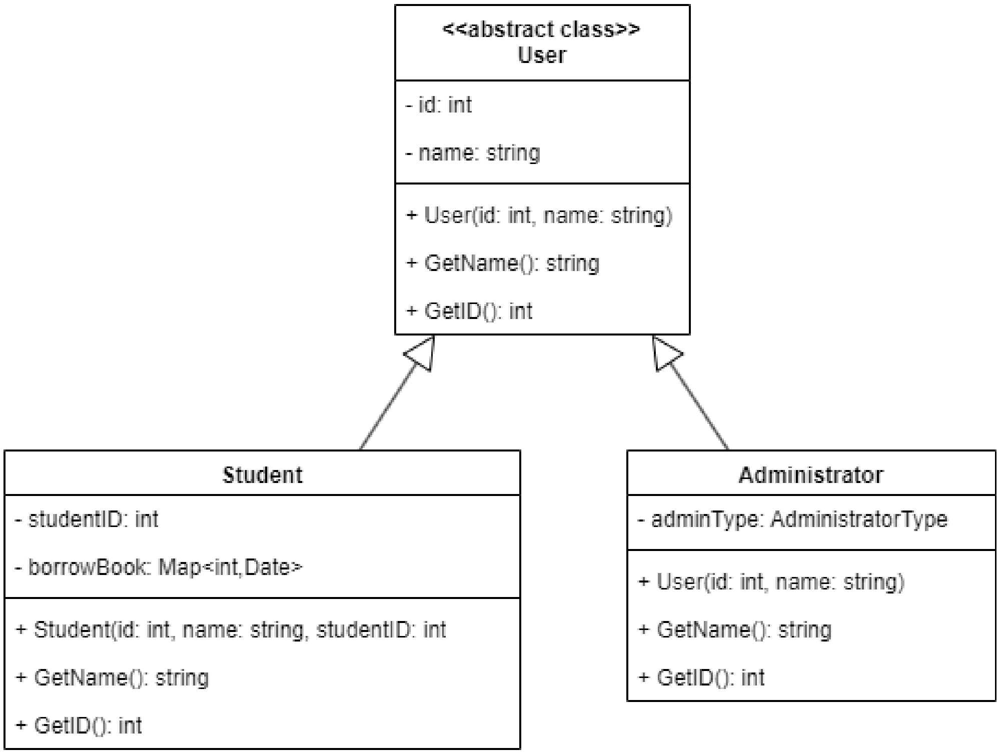


Figure 24 User class diagram (Class diagram)

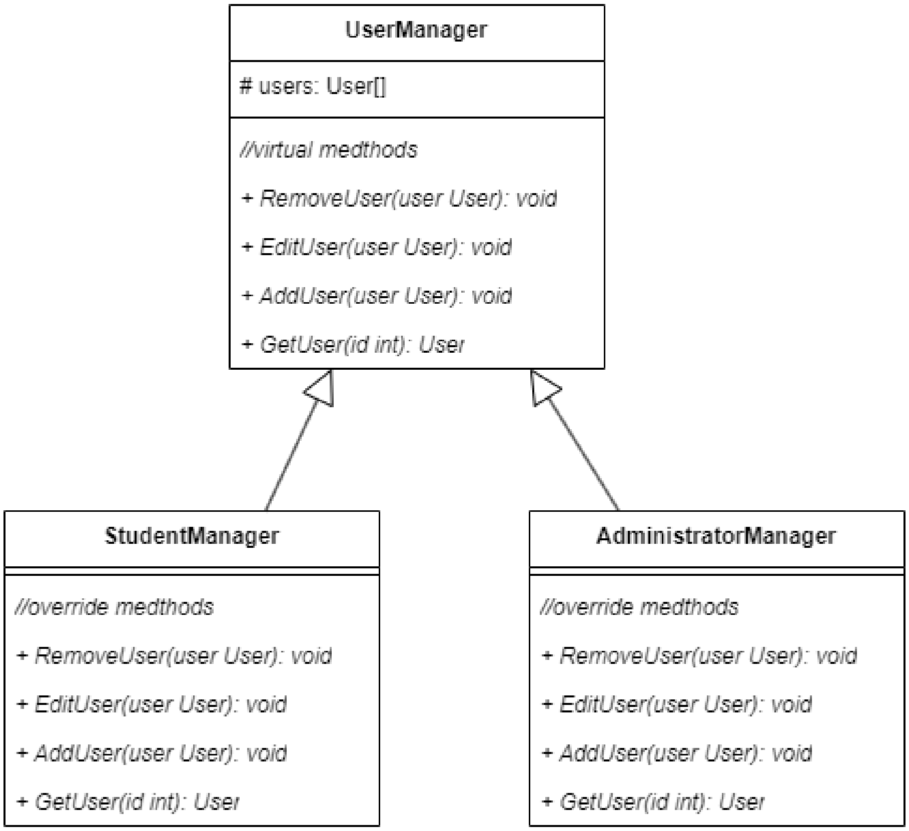

UserManager
# users: User[]
//virtual medthods
+ RemoveUser(user User): void
+ EditUser(user User): void
+ AddUser(user User): void
+ GetUser(id int): User
StudentManager
//override medthods
+ RemoveUser(user User): void
+ EditUser(user User): void
+ AddUser(user User): void
+ GetUser(id int): User
AdministratorManager
//override medthods
+ RemoveUser(user User): void
+ EditUser(user User): void
+ AddUser(user User): void
+ GetUser(id int): User

Figure 25 User Management class diagram (Class diagram)

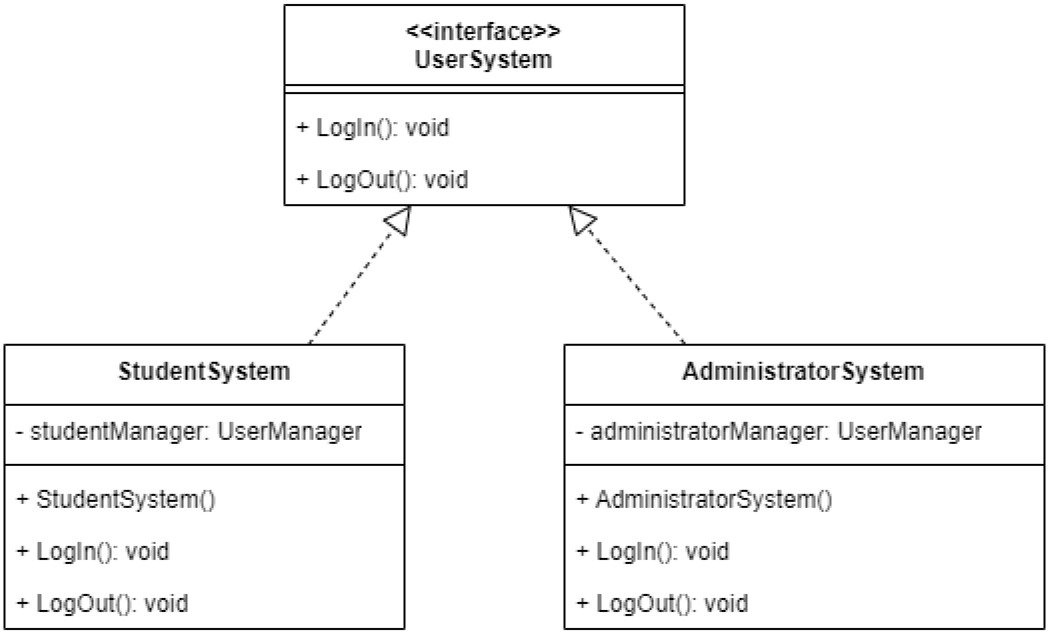


Figure 26 User System class diagram (Class diagram)

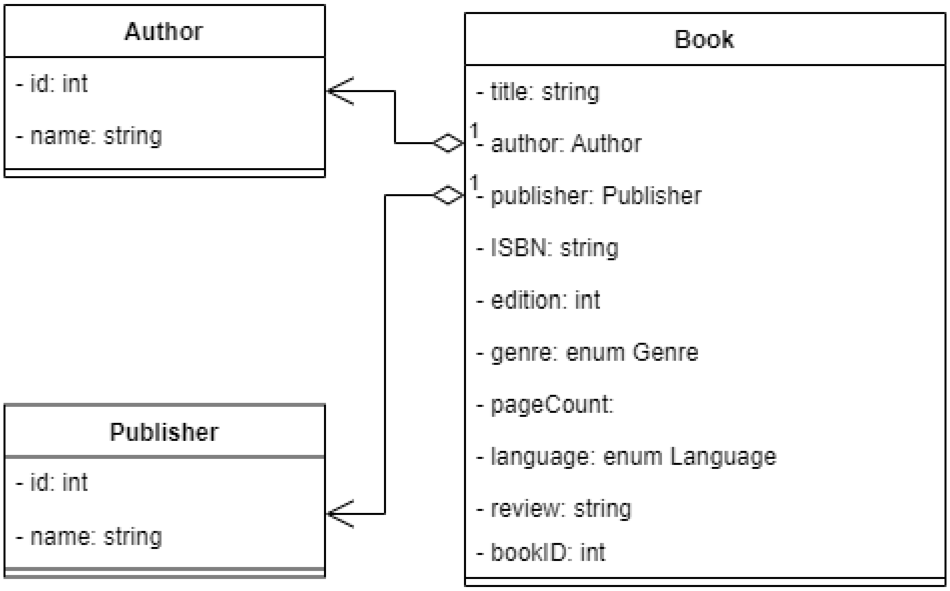


Figure 27 Book class diagram (Class diagram)

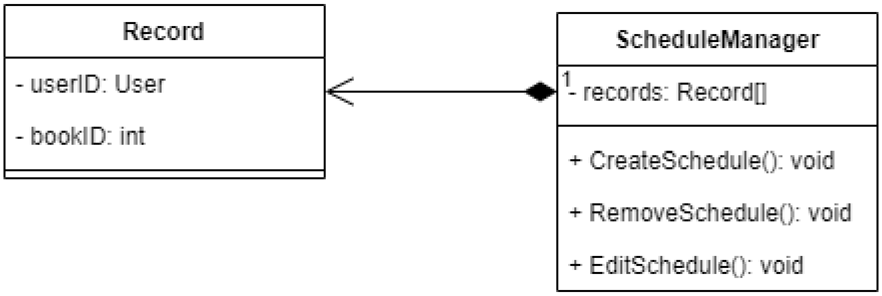


Figure 28 Schedule Management class diagram (Class diagram)

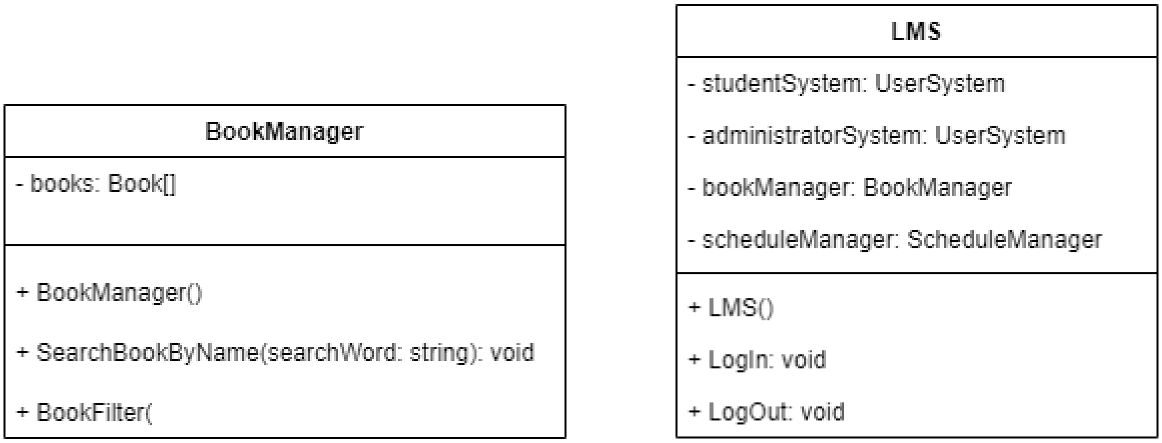

BookManager
- books: Book[]
+ BookManager()
+ SearchBookByName(searchWord: string): void
+ BookFilter(
LMS
- studentSystem: UserSystem
- administratorSystem: UserSystem
- bookManager: BookManager
- scheduleManager: ScheduleManager
+ LMS()
+ LogIn: void
+ LogOut: void

Figure 29 LMS and Book Management class diagram (Class diagram)

5. Activity diagram

Login/Logout

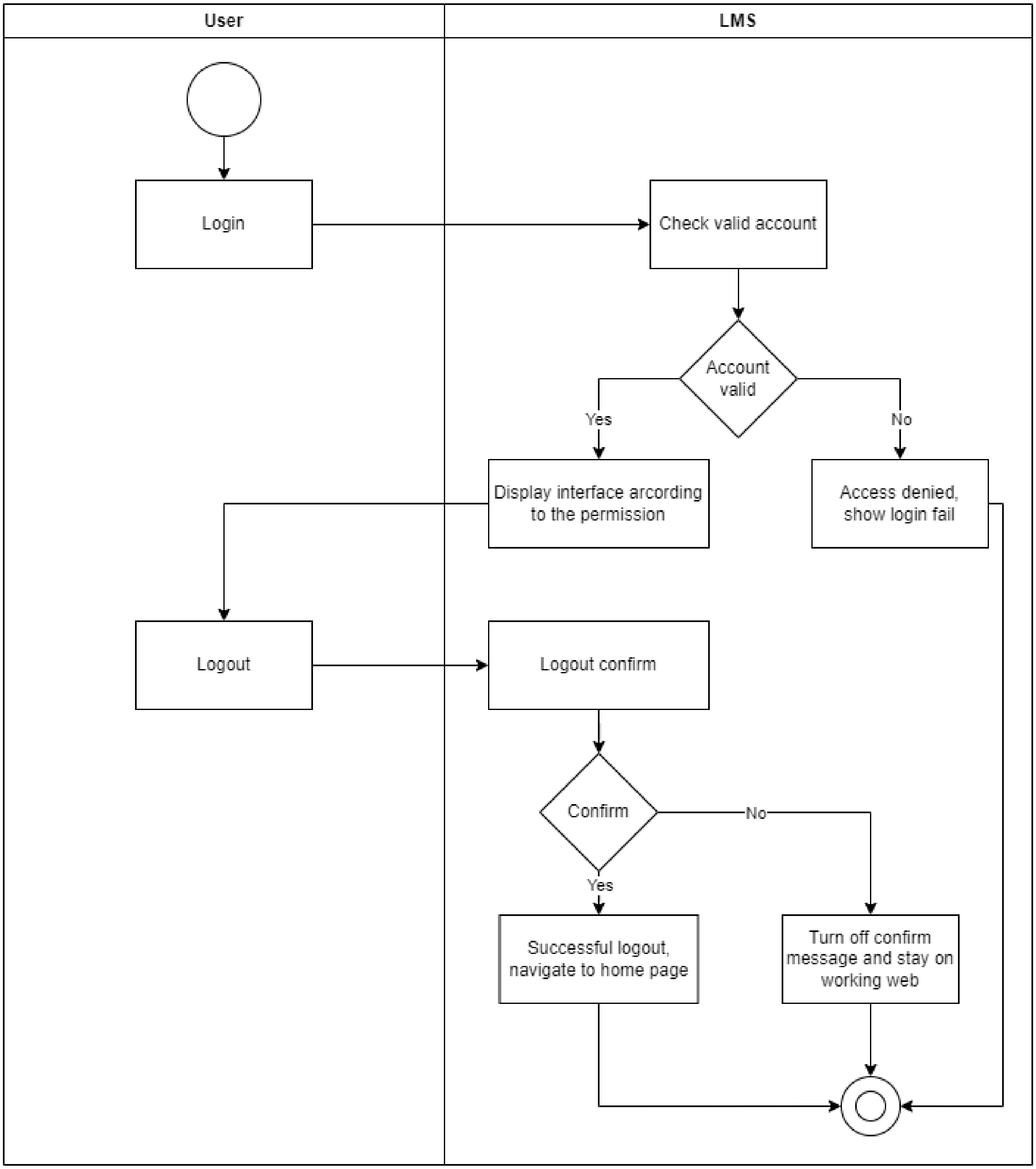

User
LMS
Login
Check valid account
Account valid
Yes
No
Display interface arcording to the permission
Access denied, show login fail
Logout
Logout confirm
Confirm
No
Yes
Successful logout, navigate to home page
Turn off confirm message and stay on working web

Figure 30 Login/Logout (Activity diagram)

Search Book

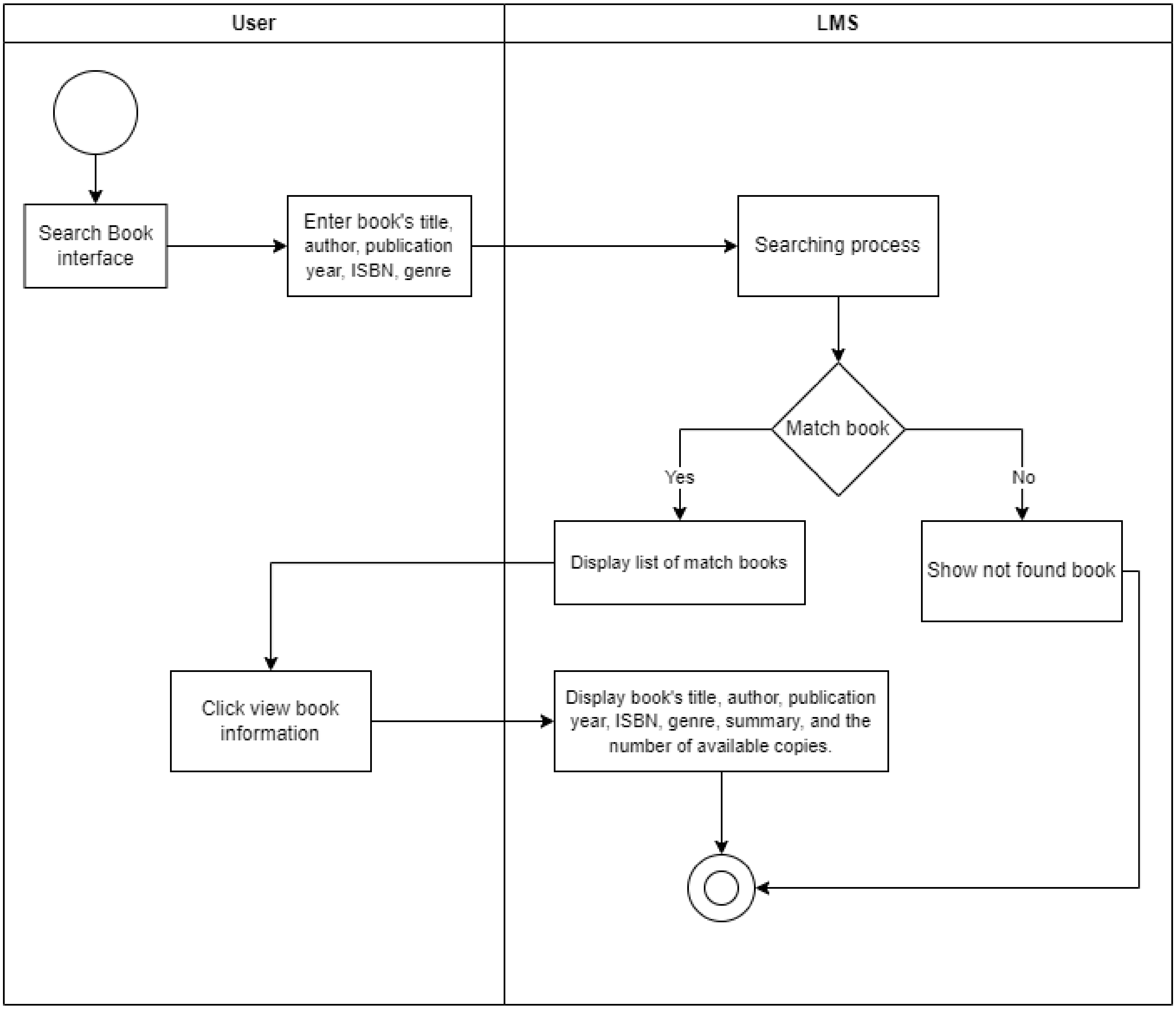

User
LMS
Search Book interface
Enter book's title, author, publication year, ISBN, genre
Searching process
Match book
Yes
No
Display list of match books
Show not found book
Click view book information
Display book's title, author, publication year, ISBN, genre, summary, and the number of available copies.

Figure 31 Search Book (Activity diagram)

Schedule Borrow Period

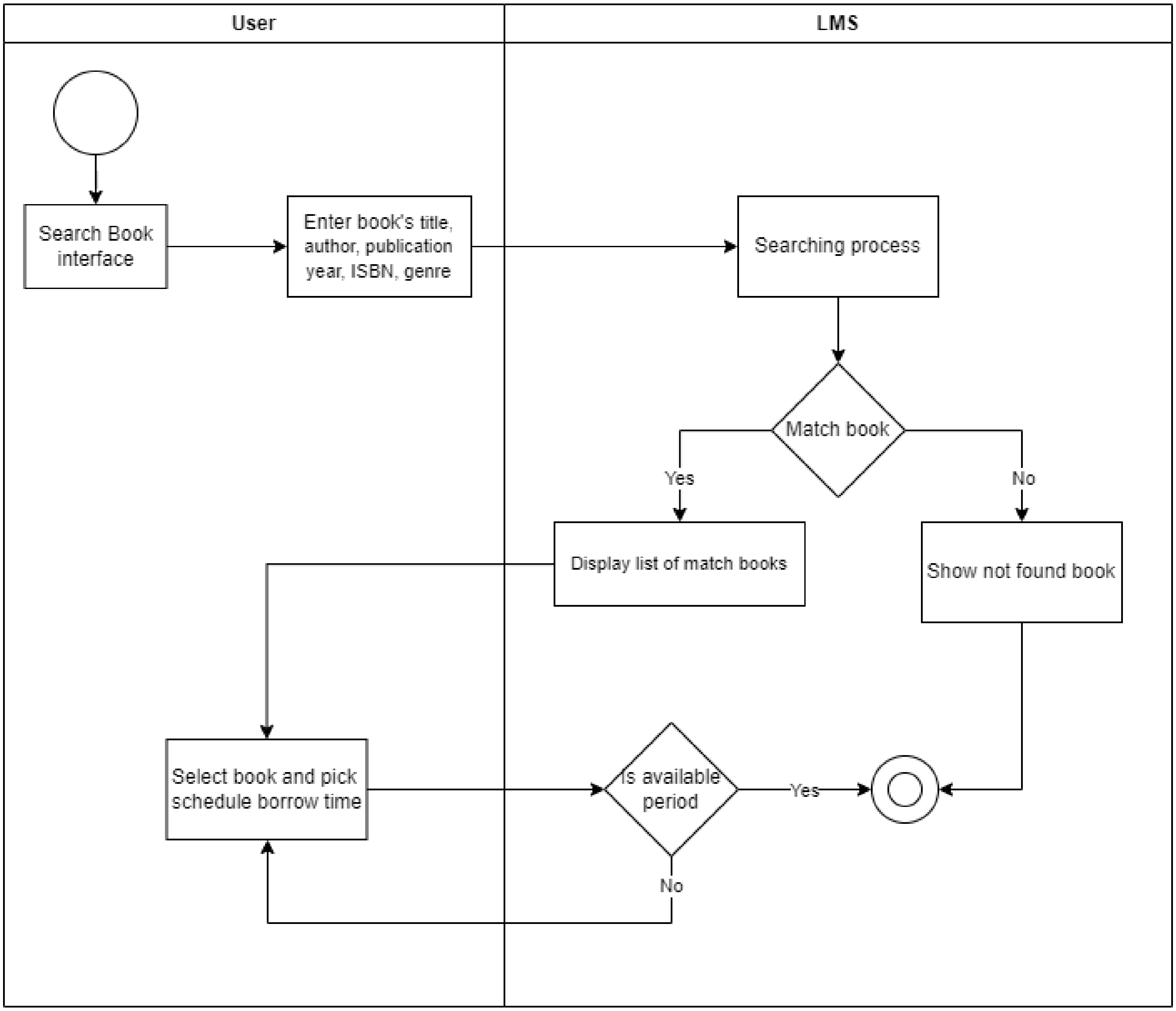

User
LMS
Search Book interface
Enter book's title, author, publication year, ISBN, genre
Searching process
Match book
Yes
No
Display list of match books
Show not found book
Select book and pick schedule borrow time
Is available period
Yes
No

Figure 32 Schedule Borrow Period (Activity diagram)

6. Sequence diagram

Login

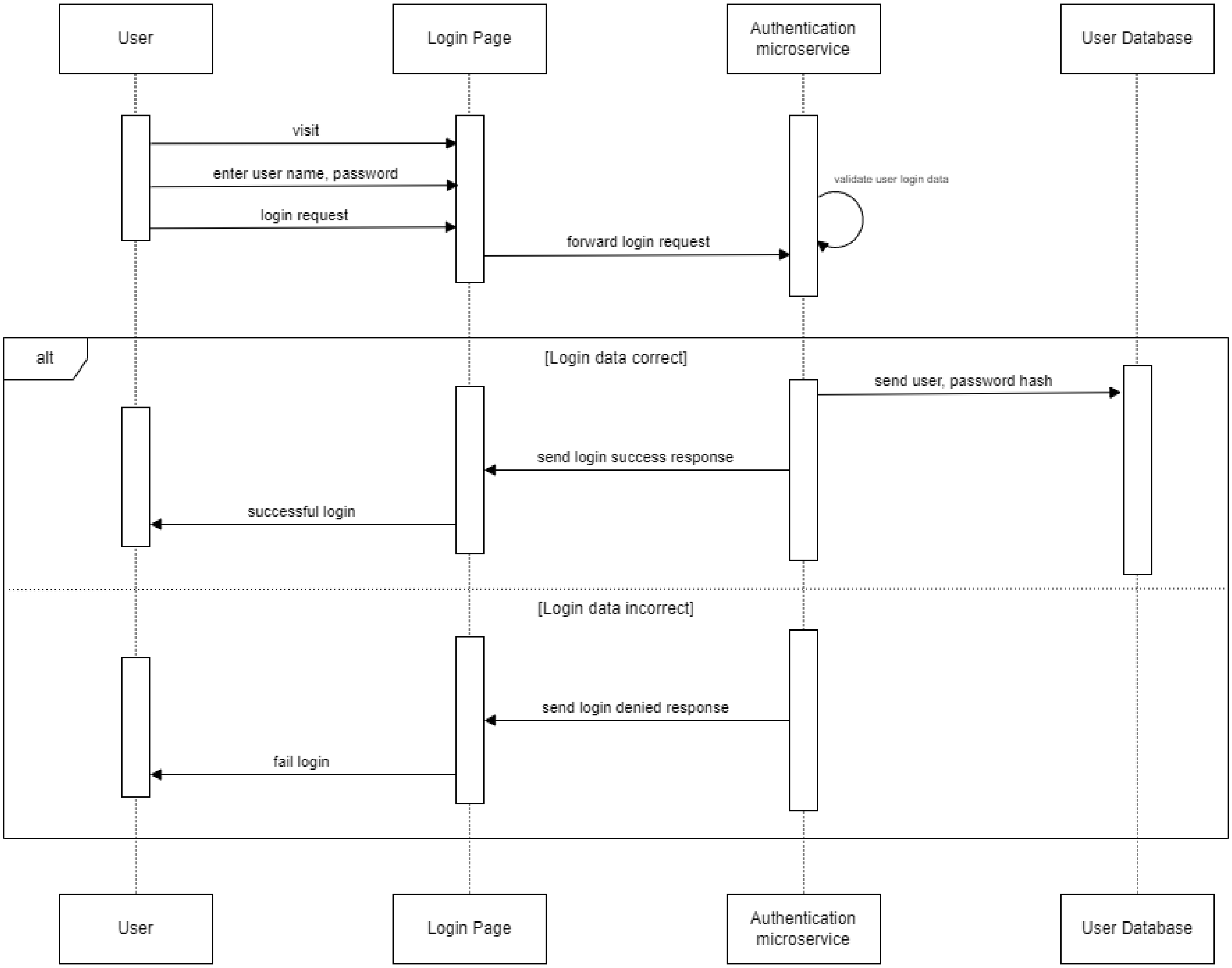


Figure 33 Login (Sequence diagram)

Search Book

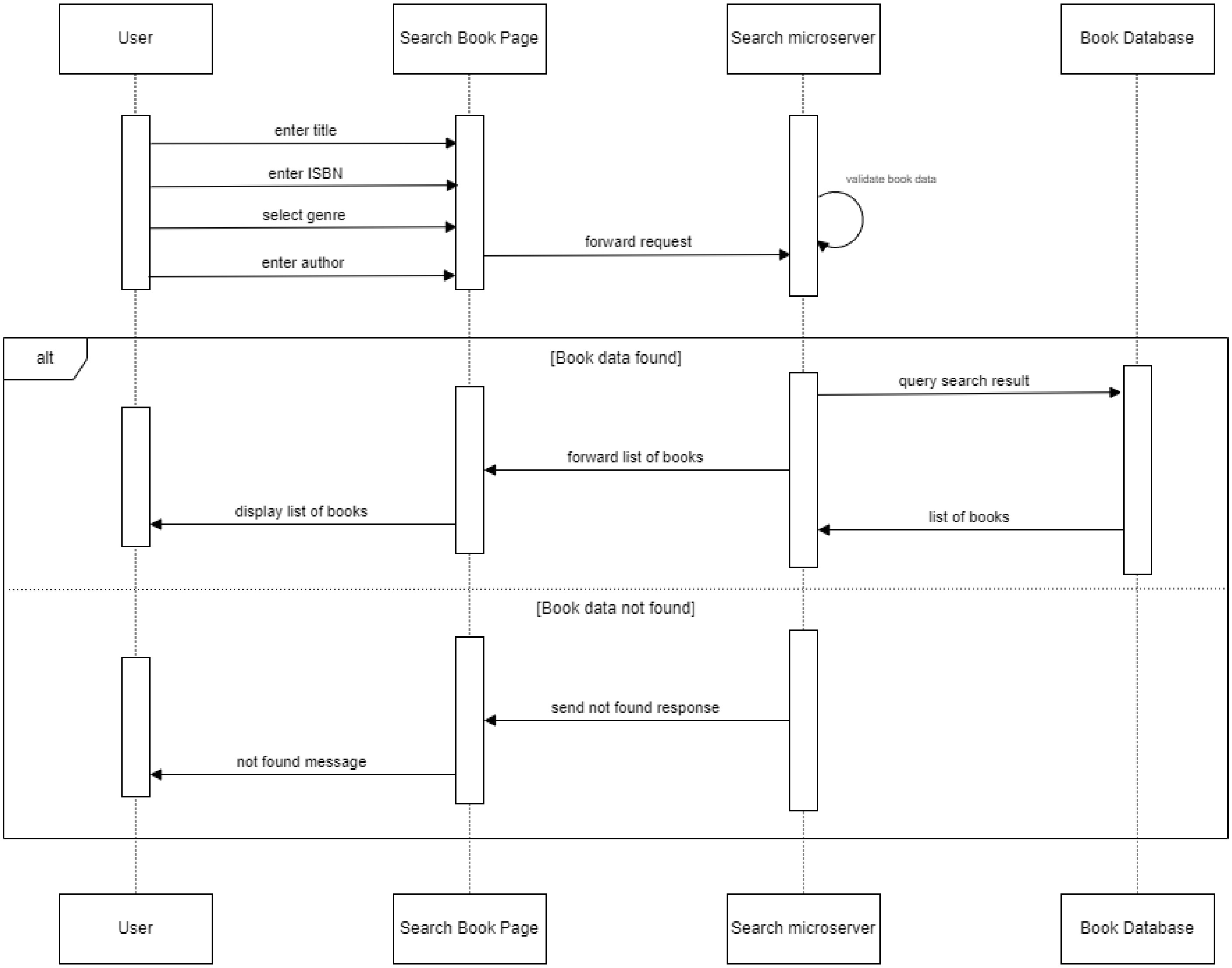


Figure 34 Search Book (Sequence diagram)

7.Sate diagram

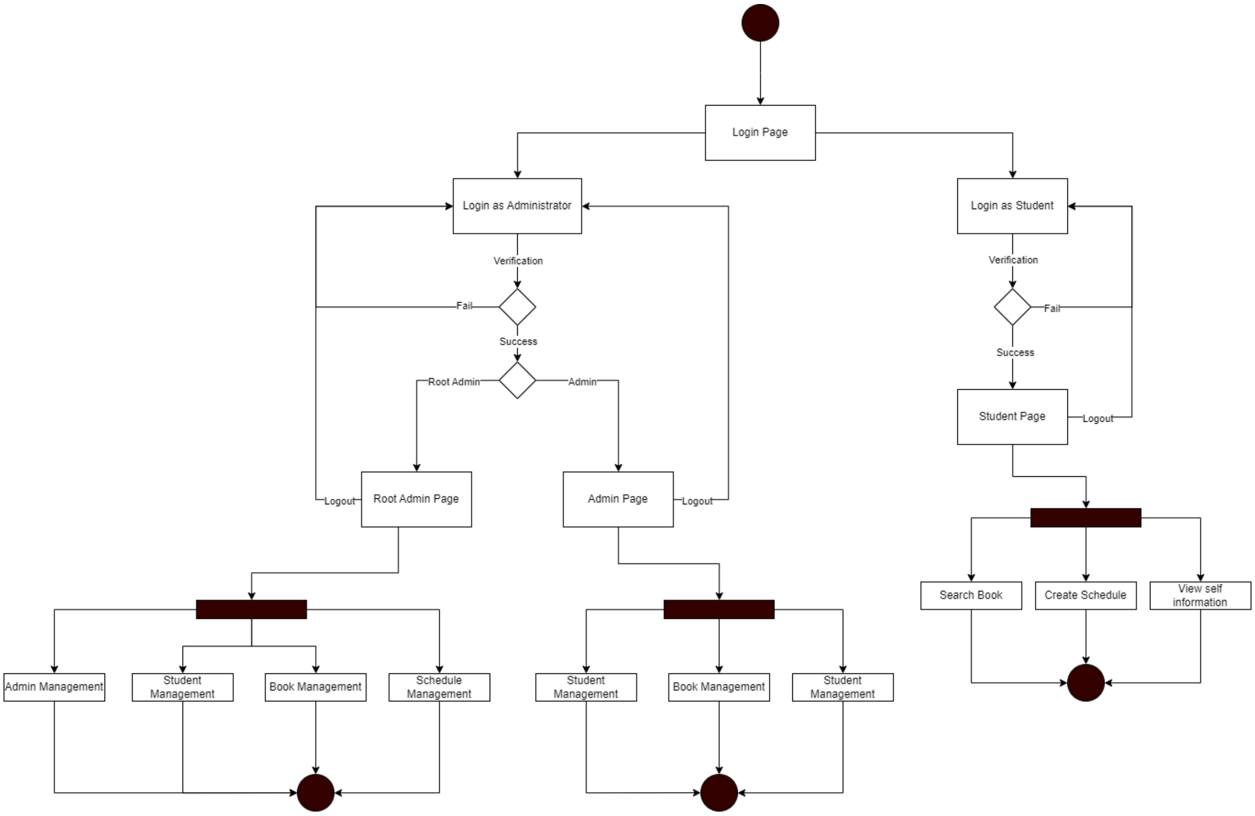


Figure 35 LMS overall state (State diagram)

Testing

Test-first:

Table 5 Microservice test-first

Function name	Test case 1	Test case 2	Test case 3
Login: The system must provide a login functionality to authenticate users and grant access based on their role.	Successful Login for a Valid Student User	Failed Login Due to Invalid Password	Failed Login for a Non-Student User
	Pre-conditions: - The user is a registered student in the university's database. - The user has valid credentials (username and password).	Pre-conditions: - The user is a registered student in the university's database. - The username is correct, but the password is incorrect.	Pre-conditions: - The username does not exist in the university's student database.
	Test data: - Username: student123 - Password: ValidPassword123	Test data: - Username: student123 - Password: InvalidPassword	Test data: - Username: nonstudent - Password: SomePassword
	Expected Result: - The system authenticates the user. - The user is granted access to student-specific functionalities. - The user is redirected to the student dashboard.	Expected Result: - The system does not authenticate the user. - An error message is displayed: "Invalid username or password." - The user remains on the login page.	Expected Result: - The system does not authenticate the user. - An error message is displayed: "Invalid username or password." - The user remains on the login page.

Logout: The system must allow users to safely end their session and exit the system, ensuring that no further actions can be performed until they log in again.	Successful Logout for a Student User	Logout Cancellation	Accessing Restricted Areas After Logout
	Pre-conditions: - The user is logged in as a student. - The user is currently on a student-specific page (e.g., student dashboard).	Pre-conditions: - The user is logged in as a student. - The user is currently on a student-specific page.	Pre-conditions: - The user was previously logged in but has successfully logged out.
	Test data: - Username: student123	Test data: - Username: student123	Test data: - Username: student123
	Expected Result: - The session is terminated, and all session data (authentication tokens or session identifiers) are invalidated. - The user is redirected to the login page. - The user is unable to access any restricted areas without logging in again.	Expected Result: - The session remains active. - The user stays on the current page. - The user can continue performing actions within the system.	Expected Result: - The system detects that there is no active session. - The user is redirected to the login page. - An error message may be displayed, indicating that login is required to access the page.
Search book: The system should include a search function for finding books	Search by Multiple Criteria (Title, Author, and Genre)	Search Results Functionality	Advanced Search with Multi-Criteria and Boolean Search
	Pre-conditions: - The library’s database includes books with	Pre-conditions: - The library’s database contains a large number	Pre-conditions: - The library’s database includes books with

in the library. It must support multiple criteria and provide relevant results based on user input.	various titles, authors, and genres. - The user is logged in as a student.	of books with varying details. - The user is logged in as a student.	various attributes. - The user is logged in as a student.
	Test data: - Search Criteria: - Title: Introduction to Algorithms - Author: Thomas H. Cormen - Genre: Computer Science	Test data: - Search Criteria: - Title: Programming - Author: John Doe - Genre: Programming - Publication Date: 2022 - Availability: Available	Test data: - Search Criteria: - Title: Data Science - Author: Alice - Genre: Technology - Boolean Search Query: ("Data Science" AND "Alice") OR ("Machine Learning" NOT "Bob")
	Expected Result: - The system displays a list of books that match all three criteria: title "Introduction to Algorithms," author "Thomas H. Cormen," and genre "Computer Science." - Each result includes the book's title, author, genre, ISBN, availability status, and location within the library. - The results are relevant and correctly filtered based on the	Expected Result: - The system displays search results ranked by relevance to the search criteria, with the most relevant results appearing at the top of the list. - If the number of search results exceeds the page limit, the system should support pagination. - The user should be able to navigate through pages of search results (e.g., Next, Previous, Page numbers).	Expected Result: - The system should display a list of books that match the combined criteria: +Books with the title "Data Science" and authored by "Alice." +Books with the title "Machine Learning" but not authored by "Bob." - Each result includes the book's title, author, genre, ISBN, availability status, and location within the library.

	provided search criteria.		
View Book Information: The system allows users to search for books and view detailed information about them, including availability, author, genre, and publication details. It also enables users to see related books and, optionally, access book reviews if available.	Display Book Details	Check Availability	View Related Books
	Pre-conditions: - The library's database contains detailed book information. - The user is logged in as a student.	Pre-conditions: - The library's database has up-to-date information on book availability. - The user is logged in as a student.	Pre-conditions: - The library's database includes books with genre and author information. - The user is logged in as a student.
	Test data: - Book Title: The Great Gatsby - Author: F. Scott Fitzgerald - Publication Year: 1925 - ISBN: 9780743273565 - Genre: Classic - Summary: A novel about the American dream and the disillusionment that comes with it. - Available Copies: 3	Test data: - Book Title: To Kill a Mockingbird - Availability Status: Checked out - Expected Return Date: 2024-08-25	Test data: - Book Title: 1984 - Genre: Dystopian - Author: George Orwell
	Expected Result: - The system displays the book's title, author, publication year, ISBN, genre, summary, and	Expected Result: - The system should display that "To Kill a Mockingbird" is checked out and show	Expected Result: - The system displays a list of related books based on the "Dystopian" genre or

	the number of available copies accurately.	the expected return date "2024-08-25".	other books by George Orwell.
Schedule Borrow Periods: The system allows users to select and reserve a specific time slot for borrowing a book from the library, with features for confirming, modifying, canceling, and receiving reminders about the scheduled borrow time.	Schedule a Borrow Period	Modify a Scheduled Borrow Time	Cancel a Scheduled Borrow Time
	Pre-conditions: - The library's database has up-to-date information on book availability and time slots. - The user is logged in as a student.	Pre-conditions: - The student has an existing scheduled borrow time. - The library's database has updated information on available time slots.	Pre-conditions: - The student has an existing scheduled borrow time.
	Test data: - Book Title: The Catcher in the Rye - Selected Time Slot: 2024-08-20 14:00 - 2024-08-20 15:00 - Availability Status: Available	Test data: - Book Title: To Kill a Mockingbird - Original Scheduled Time Slot: 2024-08-15 10:00 - 2024-08-15 11:00 - New Time Slot: 2024-08-16 11:00 - 2024-08-16 12:00 - Availability Status: Available for the new slot	Test data: - Book Title: Pride and Prejudice - Scheduled Time Slot: 2024-08-25 09:00 - 2024-08-25 10:00
	Expected Result: - The system should confirm that the selected time slot is available and reserve it for the user.	Expected Result: - The system should update the schedule with the new time slot and send an updated confirmation email to	Expected Result: - The system should cancel the scheduled borrow time and send a cancellation notice email to the student

	- The system should send a confirmation email to the student with the book's details, the scheduled borrow time, and instructions for picking up the book.	the student with the revised details.	with details of the cancellation.
Edit Schedule: The system allows students and administrators to modify an existing book borrowing schedule, including changing the borrowing dates or details.	Verify Access Control for Editing Schedule	Validate Conflict Detection During Schedule Editing	Verify Schedule Update and Confirmation Process
	Pre-conditions: - The user is logged in as a student who has previously created a borrowing schedule.	Pre-conditions: - The book "1984" is already scheduled to be borrowed by another student during the overlapping period.	Pre-conditions: - The user is logged in as an administrator.
	Test data: - User: Student (Jane Doe) - Schedule ID: 1234 - Book Title: "1984" - Borrowing Period: 2024-09-01 to 2024-09-10	Test data: - User: Student (Jane Doe) - Schedule ID: 1234 - Original Borrowing Period: 2024-09-01 to 2024-09-10 - New Borrowing Period: 2024-09-05 to 2024-09-15 (conflicts with another student's schedule)	Test data: - User: Administrator (Admin1) - Schedule ID: 5678 - Book Title: "Brave New World" - Original Borrowing Period: 2024-10-01 to 2024-10-10 - New Borrowing Period: 2024-10-05 to 2024-10-15
	Expected Result: - The system should authenticate Jane Doe's	Expected Result: - The system should detect the conflict with	Expected Result: - The system should display the updated

	credentials. - The system should allow Jane Doe to access and edit the selected schedule. - The system should display the current schedule details for editing.	another student's borrowing schedule. - The system should display a conflict notification and suggest alternative available time slots. - Jane Doe should not be able to confirm the conflicting schedule until the conflict is resolved.	schedule details for confirmation. - After confirmation, the system should update the schedule in the database. - The system should send a confirmation message to the student associated with the schedule. - The book's availability status should be updated accordingly.
Remove Schedule: The system allows students and administrators to remove an existing book borrowing schedule.	Verify Access Control for Schedule Removal	Validate Schedule Removal Process and Confirmation	Verify Access Logs and Error Handling During Schedule Removal
	Pre-conditions: - The user is logged in as the student who created the borrowing schedule.	Pre-conditions: - The user is logged in as an administrator.	Pre-conditions: - The user is logged in as the student who created the borrowing schedule, and the system is experiencing a temporary database issue.
	Test data: - User: Student (John Doe) - Schedule ID: 7890 - Book Title: "To Kill a Mockingbird" - Borrowing Period:	Test data: - User: Administrator (Admin1) - Schedule ID: 12345 - Book Title: "Pride and Prejudice" - Borrowing Period:	Test data: - User: Student (Jane Doe) - Schedule ID: 6789 - Book Title: "Moby Dick" - Borrowing Period:

	2024-08-25 to 2024-09-05	2024-09-10 to 2024-09-20	2024-10-01 to 2024-10-15
	Expected Result: - The system should authenticate John Doe's credentials. - The system should verify that John Doe is authorized to remove this schedule. - The system should display a confirmation prompt with the schedule details for John Doe to confirm the removal.	Expected Result: - The system should display the details of the schedule in the confirmation prompt. - Admin1 should be able to confirm the removal action by clicking a confirmation button. - The system should remove the schedule and update the database accordingly. - A confirmation message should be displayed, and the book’s availability status should be updated.	Expected Result: - The system should log the attempt to remove the schedule, including details such as the book ID, schedule details, time of removal attempt, and Jane Doe's user ID. - Upon encountering the error, the system should display a clear error message to Jane Doe, explaining the issue and offering options to retry or contact support. - The system should ensure that no partial or duplicate removals occur. - If the issue persists, the system should log the error for further investigation by administrators.
Load Schedule: The system enables to load and display the book	Successful Loading and Display of Book Borrowing Schedules	Access Control Verification	Data Retrieval and Accuracy
	Pre-conditions: - The database contains	Pre-conditions: - The user is logged in	Pre-conditions: - The database contains

borrowing schedules from the database, providing information on current and upcoming book borrowings.	up-to-date and accurate borrowing schedule data. - The user is logged in as an administrator.	as a student or administrator. - The student should only be able to view their own borrowing schedules.	correct borrowing schedule data. - The user is logged in as an administrator.
	Test data: - Book ID: 56789 - Student ID: STU12345 - Start Date: 2024-08-15 - End Date: 2024-08-30 - Status: Active	Test data: - Student ID (for access check): STU54321 (not authorized for current user) - Current User Role: Student	Test data: - Query Parameters: Book ID 56789, Date Range 2024-08-01 to 2024-08-31
	Expected Result: - The system should display the borrowing schedule in a table or calendar view. - For each schedule entry, the display should include book title, student name, borrowing dates, and status. - The format should be clear and user-friendly, making it easy to view current and upcoming borrowings.	Expected Result: - The student should only see their own borrowing schedule and be restricted from viewing others' schedules. - The administrator should be able to view and manage all schedules. - Access controls should enforce permissions accurately and restrict unauthorized access.	Expected Result: - The system should correctly retrieve and display the borrowing schedules for the specified book and date range. - The displayed data should include accurate details such as book ID, student ID, start date, end date, and borrowing status. - Any discrepancies or errors in the data should be flagged, with an option to correct or report them.

Save Schedule: The system enables to save the borrowing schedule data into the database, ensuring that all borrowing arrangements are properly recorded and managed.	Successful Saving of a New Borrowing Schedule	Data Validation and Conflict Checking	Updating an Existing Schedule
	Pre-conditions: - The student is logged in and authenticated. - The book is available for borrowing during the requested period.	Pre-conditions: - The student is logged in and authenticated. - The book is already borrowed for the entire period requested.	Pre-conditions: - The student is logged in and authenticated. - There is an existing borrowing schedule that can be modified.
	Test data: - Student ID: STU12345 - Book ID: 56789 - Start Date: 2024-08-15 - End Date: 2024-08-22 - Special Notes: Requires a large print edition	Test data: - Student ID: STU54321 - Book ID: 56789 - Start Date: 2024-08-15 - End Date: 2024-08-20	Test data: - Student ID: STU67890 - Book ID: 12345 - Original Start Date: 2024-08-10 - Original End Date: 2024-08-15 - New Start Date: 2024-08-12 - New End Date: 2024-08-17
	Expected Result: - The system should successfully save the schedule with the details entered. - The saved schedule should be retrievable from the database with the correct details (student ID, book ID, start and end dates, and	Expected Result: - The system should validate the requested schedule and identify the conflict with the existing booking. - The system should display a clear error message indicating that the book is not available for the requested period.	Expected Result: - The system should successfully update the existing schedule with the new dates. - The updated schedule should be saved in the database, reflecting the new start and end dates. - The system should keep a log of the

	special notes). - The system should confirm that the schedule has been saved successfully.	- The schedule should not be saved, and the student should be guided to select an alternative time slot.	changes, including the date and time of modification and the user who made the changes. - The student should receive a confirmation message with the updated schedule details.
Add Book: The system allows librarians (administrators) to add new books to the library management system. This feature ensures that the library's book collection is updated with the latest additions and maintains accurate records for	Successfully Adding a New Book	Form Validation for Required Fields	Uploading and Displaying a Book Cover Image
	Pre-conditions: - The administrator is logged in and authenticated. - The system is operational, and the book catalog is accessible.	Pre-conditions: - The administrator is logged in and authenticated. - The system is operational.	Pre-conditions: - The administrator is logged in and authenticated. - The system is operational and capable of handling file uploads.
	Test data: - Title: "The Great Adventure" - Author: "John Doe" - ISBN: "978-3-16-148410-0" - Publication Date: "2024-09-01" - Genre: "Adventure"	Test data: - Title: " " - Author: "Jane Smith" - ISBN: "978-1-23-456789-0" - Publication Date: "2023-05-15" - Genre: "Science Fiction"	Test data: - Title: "The Modern Classics" - Author: "Emily Clark" - ISBN: "978-0-14-312854-0" - Publication Date: "2024-01-20" - Genre: "Classics" - Cover Image: (A valid

students to access.	- Cover Image: (A valid image file)	- Cover Image: (No image)	image file named "modern_classics.jpg")
	Expected Result: - The system should successfully add the book to the database. - A confirmation message should be displayed, including the book's details. - The new book should be immediately visible in the system's book catalog. - The book cover image should be displayed with the book information.	Expected Result: - The system should display validation error messages indicating which required fields are missing. - The book should not be added to the database. - The form should not submit until all required fields are filled out correctly.	Expected Result: - The book should be successfully added to the database with the provided details. - The cover image should be uploaded, stored, and displayed alongside the book information in the catalog. - The system should ensure that the cover image is correctly linked to the book and visible to users.
Remove Book: The system allows authorized users to remove a book from the library's inventory.	Successfully Removing a Book	Error Handling for Checked-Out Books	Confirmation and Logging of Book Removal
	Pre-conditions: - The administrator is logged in and authenticated. - The book to be removed exists in the library's inventory. - The book is not currently checked out.	Pre-conditions: - The administrator is logged in and authenticated. - The book to be removed exists in the library's inventory and is currently checked out.	Pre-conditions: - The administrator is logged in and authenticated. - The book to be removed exists in the library's inventory and is not checked out.

	Test data: - Book ID: B987654 - Title: The Lost Chronicles - Author: Emily Stone - ISBN: 978-1-23-456789-0	Test data: - Book ID: B123456 - Title: History of Art - Author: Michael Clark - ISBN: 978-0-14-312854-0	Test data: - Book ID: B654321 - Title: Advanced Programming - Author: Sarah Lee - ISBN: 978-3-16-148410-0
	Expected Result: - The system provides a search interface where the administrator can locate the book using the provided data. - Upon selecting the book, detailed information is displayed for confirmation. - After confirming the removal, the system updates the database and permanently deletes the book's record. - A notification log entry is generated, including details such as the book ID, title, author, and the administrator who performed the action.	Expected Result: - The system identifies that the book is checked out and prevents its removal. - An error message is displayed, indicating that the book cannot be removed while it is on loan. The system provides clear instructions for resolving the issue, such as contacting support or returning the book first.	Expected Result: - The system provides a search interface where the administrator can locate the book. - After selecting the book, detailed information is displayed for confirmation. - Upon confirmation, the system updates the database and permanently deletes the book's record. - A notification log entry is created with details about the removal, including the book ID, title, author, and the administrator who performed the action. - The root administrator is able to access and review the logs of all book removals.

Edit Book: The system allows authorized users to edit or update the information of a book in the library's inventory.	Authentication Check	Update Book Information	Error Handling - Invalid Data
	Pre-conditions: - User must be logged into the system. - The system should have valid credentials for different user roles (administrator, normal user, and root administrator).	Pre-conditions: - User is logged in as an administrator. - The book to be edited must exist in the system.	Pre-conditions: - User is logged in as an administrator. - The book to be edited must exist in the system.
	Test data: - Non-Admin User: student_user - Admin User: admin_user - Root Admin User: root_admin	Test data: - Book to Edit: ISBN: 978-1-23-456789-0 Title: "Old Title" Author: "Old Author" Publication Date: "2023-05-15" Genre: "Science Fiction" - Updated Book Details: Title: "New Title" Author: "New Author" Publication Date: "2024-01-01"	Test data: - Book to Edit: ISBN: 978-1-23-456789-0 - Invalid Data: ISBN: 123456789 (Invalid format)
	Expected Result: - Access is denied. The student user receives an error message stating that only administrators can access this feature.	Expected Result: - The system displays a summary of the changes made. - The book's information is updated	Expected Result: - The system should validate the input and display an error message indicating that the ISBN format is

	- Access is granted. The admin user can view and interact with the book edit interface and manage additional permissions if applicable.	in the database. The new details are reflected in the book's record.	invalid. - The system prevents the changes from being saved due to invalid data. The user is notified to correct the input before saving.
Load Book Data: The system allows the system to retrieve and display book data from the library's database for both students and administrators.	Successful Data Retrieval and Display	Database Connection Failure	Data Integrity and Accuracy
	Pre-conditions: - The database connection is established and secure. - The database contains up-to-date book information.	Pre-conditions: - Simulate a database connection failure.	Pre-conditions: - The database contains accurate and up-to-date book information. - There are known discrepancies in the database that need to be checked.
	Test data: - Book ID: 12345 - Book Title: The Great Gatsby - Author: F. Scott Fitzgerald - Publication Date: 1925-04-10 - Genre: Fiction - ISBN: 9780743273565 - Availability Status: Available	Test data: - N/A (Simulation of a connection failure)	Test data: - Book Title: Moby-Dick - Expected Author: Herman Melville - Expected Publication Date: 1851-10-18 - Expected Genre: Adventure

	- Book Cover: URL to cover image		
	Expected Result: - The system should display the book information, including title, author, publication date, genre, ISBN, availability status, and cover image. - The display should be user-friendly with sortable columns and filtering options.	Expected Result: - The system should display a clear error message indicating a connection failure. - The error message should include options to retry the operation or contact support if the problem persists. - No book data should be displayed, and the interface should handle the error gracefully without crashing.	Expected Result: - The system should display accurate and up-to-date information for Moby-Dick, including title, author, publication date, and genre. - Any discrepancies between displayed data and the actual data should be logged for review by the root administrator. - Data integrity checks should ensure that the information is consistent with the library's inventory records.
Save Book Data: The system allows authorized users to save or update book information in	Successfully Saving a New Book	Handling Duplicate ISBN	Error Handling During Save Operation
	Pre-conditions: - The administrator is logged in and authenticated. - The book being added does not already exist in the database.	Pre-conditions: - The administrator is logged in and authenticated. - A book with the same ISBN already exists in the database.	Pre-conditions: - The administrator is logged in and authenticated. - A temporary issue (e.g., a database connection error) occurs

the system’s database.			during the save operation.
	Test data: - Title: Understanding AI - Author: Jane Doe - ISBN: 978-1-23-456789-1 - Publication Year: 2024 - Genre: Technology	Test data: - Title: Advanced Data Science - Author: John Smith - ISBN: 978-1-23-456789-1 - Publication Year: 2023 - Genre: Science	Test data: - Title: Modern Algorithms - Author: Alice Johnson - ISBN: 978-0-12-345678-9 - Publication Year: 2022 - Genre: Computing
	Expected Result: - The system validates the completeness and correctness of the provided data. - The system ensures that the ISBN is unique in the database. - The system successfully saves the book data, updates the database, and displays a confirmation message. - A notification log entry is generated, capturing the book details and the administrator's information.	Expected Result: - The system identifies the duplicate ISBN and prompts the administrator with options to update the existing record or cancel the operation. - The system does not save any new data if the operation is canceled. - If the administrator chooses to update, the system validates the new data and updates the existing record.	Expected Result: - The system detects the save operation failure and displays an appropriate error message to the administrator. - The system logs the failure for troubleshooting and provides options to retry the save operation. - No partial or invalid data is saved in the database during the failure.

Add Student: The system allows authorized administrators to add a new student to the system, enabling them to access and use the library's resources.	Successfully Adding a New Student	Data Validation for Required Fields	Duplicate Student ID Check
	Pre-conditions: - The administrator is logged in and authenticated. - The system is operational, and the student database is accessible.	Pre-conditions: - The administrator is logged in and authenticated. - The system is operational.	Pre-conditions: - The administrator is logged in and authenticated. - The student database is accessible and contains existing records.
	Test data: - Student ID: S789012 - Name: Alice Smith - Email: alice.smith@example.com - Course Details: Mathematics	Test data: - Student ID: S123456 - Name: `` - Email: invalid-email - Course Details: Computer Science	Test data: - Existing Student ID: S123456 - New Student Data: - Student ID: S123456 - Name: John Doe - Email: john.doe@example.com - Course Details: Engineering
	Expected Result: - The system should successfully add the student to the database. - A confirmation message should be displayed, including the student's details. - The student account should be created, and login credentials should be sent to the student's	Expected Result: - The system should display validation error messages indicating which required fields are missing or contain invalid data. - The student should not be added to the database. - The form should not submit until all required	Expected Result: - The system should check for duplicate student IDs and find that the ID already exists. - An error message should be displayed, indicating that the student ID is already in use. - The new student

	email. - The system should log the addition of the new student for auditing purposes.	fields are correctly filled out.	should not be added to the database, and the form should not submit with a duplicate ID.
Edit Student: The system allows administrators to edit or update information related to a student's book borrowing history and scheduled borrowing sessions.	Successfully Editing a Student's Borrowing History	Editing Scheduled Borrowing Information	Handling Cancellations of Scheduled Borrowings
	Pre-conditions: - The administrator is logged in and authenticated. - The student and their borrowing history are present in the system.	Pre-conditions: - The administrator is logged in and authenticated. - The student has future scheduled borrowings in the system.	Pre-conditions: - The administrator is logged in and authenticated. - The student has a scheduled borrowing session that can be canceled.
	Test data: - Student ID: S123456 - Book ID: B7890 - New Return Date: 2024-09-15 - New Status: Returned	Test data: - Student ID: S789012 - Book ID: B4567 - New Pickup Date: 2024-10-01 - New Duration: 14 days	Test data: - Student ID: S345678 - Book ID: B2345
	Expected Result: - The system should update the borrowing record with the new return date and status. - A notification email should be sent to the student, detailing the	Expected Result: - The system should update the scheduled borrowing with the new pickup date and duration. - A notification email should be sent to the	Expected Result: - The system should cancel the scheduled borrowing session and update the records accordingly. - A notification email should be sent to the

	updated return date and status. - The system should log the changes, including the administrator’s details and the timestamps of the edits.	student, including details about the updated pickup date and borrowing period. - The system should log the changes, including the administrator’s details and timestamps of the edits.	student, informing them of the cancellation and any relevant details. - The system should log the cancellation action, including the administrator’s details and timestamps.
Search Student: The system allows administrators to search for and view detailed information about students registered in the system.	Successfully Searching for a Student by ID	Searching with Partial Matches	Handling No Results Found
	Pre-conditions: - The administrator is logged in and authenticated. - The student with the specified ID is registered in the system.	Pre-conditions: - The administrator is logged in and authenticated. - Students with names partially matching the search criteria are registered in the system.	Pre-conditions: - The administrator is logged in and authenticated.
	Test data: - Search Criteria: Student ID S123456	Test data: - Search Criteria: Name "John"	Test data: - Search Criteria: Email "nonexistent@example.com"
	Expected Result: - The system should return a list with the student matching the ID. - The displayed results should include the student's basic details	Expected Result: - The system should return a list of students whose names contain "John". - The results should include basic details such as ID, name,	Expected Result: - The system should display a message indicating that no students match the search criteria. - The message should suggest refining the

	(ID, name, department, email). - Clicking on the student's name should display detailed information, including contact details, borrowing history, and any active borrowed books.	department, and email. - Clicking on a student’s name should display detailed information about the selected student.	search criteria. - The system should handle the scenario gracefully, without errors or crashes.
Load Student Data: The system loads and verifies student data from the university's database for authentication and account creation purposes.	Successful Data Retrieval and Integration	Handling Connection Issues	Data Security and Compliance
	Pre-conditions: - The administrator is logged in and authenticated. - The university’s student database is accessible and operational.	Pre-conditions: - The administrator is logged in and authenticated. - Simulate a connection issue with the university’s database.	Pre-conditions: - The administrator is logged in and authenticated. - Student data is being retrieved from the university’s database.
	Test data: - Student ID: S123456 - Name "Alice Johnson" - Department "Computer Science" - Enrollment Status "Active".	Test data: - Student ID: S123456	Test data: - Student ID: S123456 - Data Encryption: Verify encryption methods used.
	Expected Result: - The system	Expected Result: - The system should	Expected Result: - The system should use

	establishes a secure connection to the university database. - The system retrieves the student data (ID, name, department, enrollment status) accurately. - The retrieved data is integrated into the LMS, allowing account creation or verification for the student. - The system should confirm successful data retrieval and integration, and any discrepancies should be flagged.	detect the connection issue and generate an error message indicating the problem. - An error log should be created detailing the connection failure. - The system should provide options to retry the connection or contact technical support. - The administrator should be informed of the issue with clear instructions for resolution.	encryption and other security measures to protect student data during transmission and storage. - The data should be securely integrated into the LMS with proper access controls. - The system should log all data loading activities, including timestamps and administrator details. - Only authorized users (e.g., root administrator) should be able to view data loading logs and details.
Save Student Data: The system allows to save and manage student data within the database, ensuring that only authorized students can access the LMS.	Successfully Saving New Student Data	Data Validation and Error Handling	Updating Existing Student Data
	Pre-conditions: - The administrator is logged in and authenticated. - The system is operational and connected to the university's database.	Pre-conditions: - The administrator is logged in and authenticated. - The system is operational.	Pre-conditions: - The administrator is logged in and authenticated. - The student data exists in the system.
	Test data: - Student ID: S654321 - Name: John Doe	Test data: - Student ID: S999999 - Name: Alice	Test data: - Student ID: S123456 - New Name: Alice

	- Email: john.doe@example.com - Department: Mathematics - Enrollment Status: Active	Wonderland - Email: alice.wonderland - Department: Literature - Enrollment Status: (Missing)	Johnson - New Department: Computer Science - New Enrollment Status: Graduated
	Expected Result: - The system successfully saves the student data into the database. - A confirmation message is displayed, verifying the data has been saved. - The student data is indexed and available for future retrieval and search operations.	Expected Result: - The system should detect that the enrollment status is missing. - An error message should be displayed, indicating the missing field and requesting correction. - The system should not save the data until all required fields are completed correctly.	Expected Result: - The system successfully updates the existing student data with the new information. - A confirmation message is displayed, indicating that the update was successful. - The update is logged with details such as the time of the update and the administrator responsible for the change. - The system ensures that all updated data is stored securely and remains accessible.

Conclusion:

The development of the Library Management System (LMS) is a significant advancement in meeting the evolving needs of education and information management. This sophisticated system bridges the gap between traditional library methods and the digital era, integrating modern technological capabilities with the fundamental functions of traditional libraries.

For students, the LMS offers an easy-to-use interface for browsing the book collection, searching book online, managing borrow time, and interacting with borrowed books. For administrators, it provides efficient tools for managing books, schedule, student, and other administrative tasks, ensuring smooth library operations.

Our extensive validation process, which included various scenarios and test cases, has confirmed the system's functionality, accuracy, and user-friendliness. Each component, from user login to book management, has been thoughtfully designed and thoroughly tested to ensure a seamless experience and precise information handling.

In a rapidly advancing technological landscape characterized by information abundance, the LMS stands as a testament to the power of innovation in the educational sector. By overcoming geographical limitations and addressing issues related to copyright, the LMS transforms library management. We are confident that the LMS will continue to redefine student interactions with literature and streamline library administration, making a significant contribution to the field of digital libraries and education.