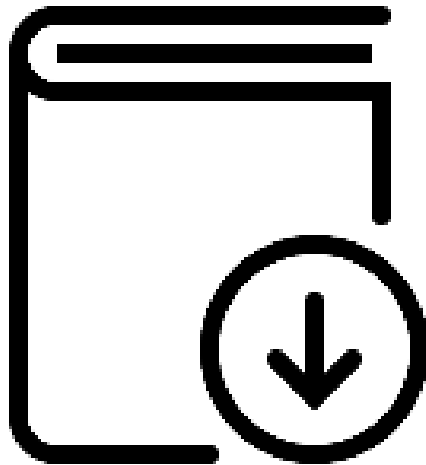


# BooknBorrow

Book-borrowing Management



---

## TABLE OF CONTENTS

---

Project presentation .....	3
BooknBorrow book-borrowing Management .....	3
Members .....	3
Database diagrams .....	4
Entity-relationship diagram .....	4
Table structure diagram.....	4
Planned gantt diagram.....	5
Use case diagrams .....	7
Borrowing Process Use Case Diagram .....	7
Library and Book Management Use Case Diagram.....	8
User Management Use Case Diagram:.....	9
Activity diagram.....	10
Borrowing Process .....	10
User Management .....	11
Sequence diagrams.....	12
Book Filing.....	12
Login .....	13
Wireframe .....	14
Front page .....	14
Book list page .....	14
Component diagrams.....	15
Layers Component Diagram .....	15
User Actions .....	16
Class diagram .....	17

---

## Project presentation

---

### BooknBorrow book-borrowing Management



Website for book-borrowing. BooknBorrow is a platform for users to borrow books from multiple libraries and keep records of their readings and eventual fines if they bring back books too late.

### Members

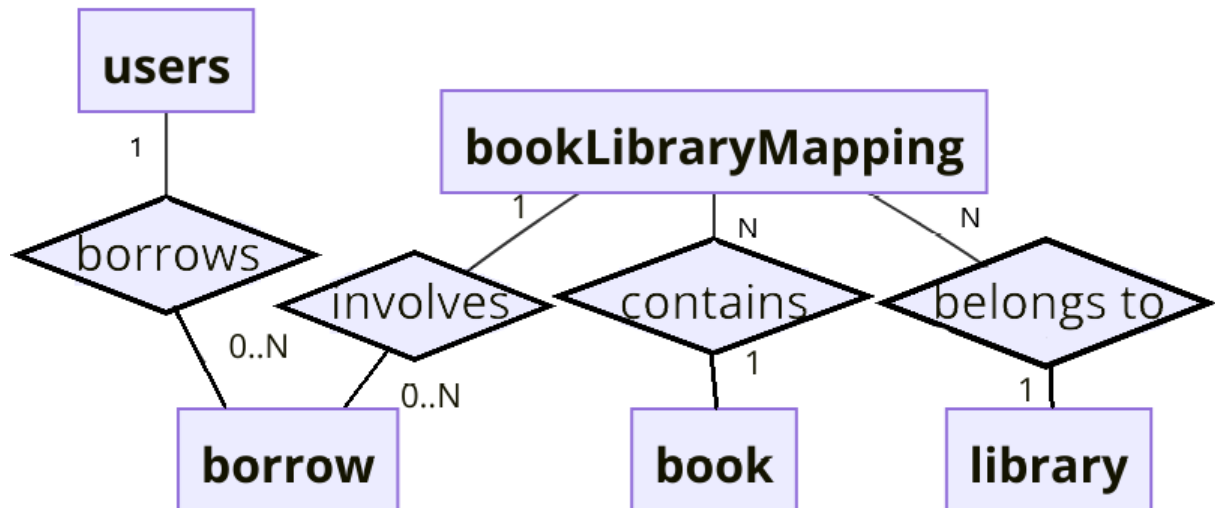
- LAURENT Sacha
- SIMON Eliot

---

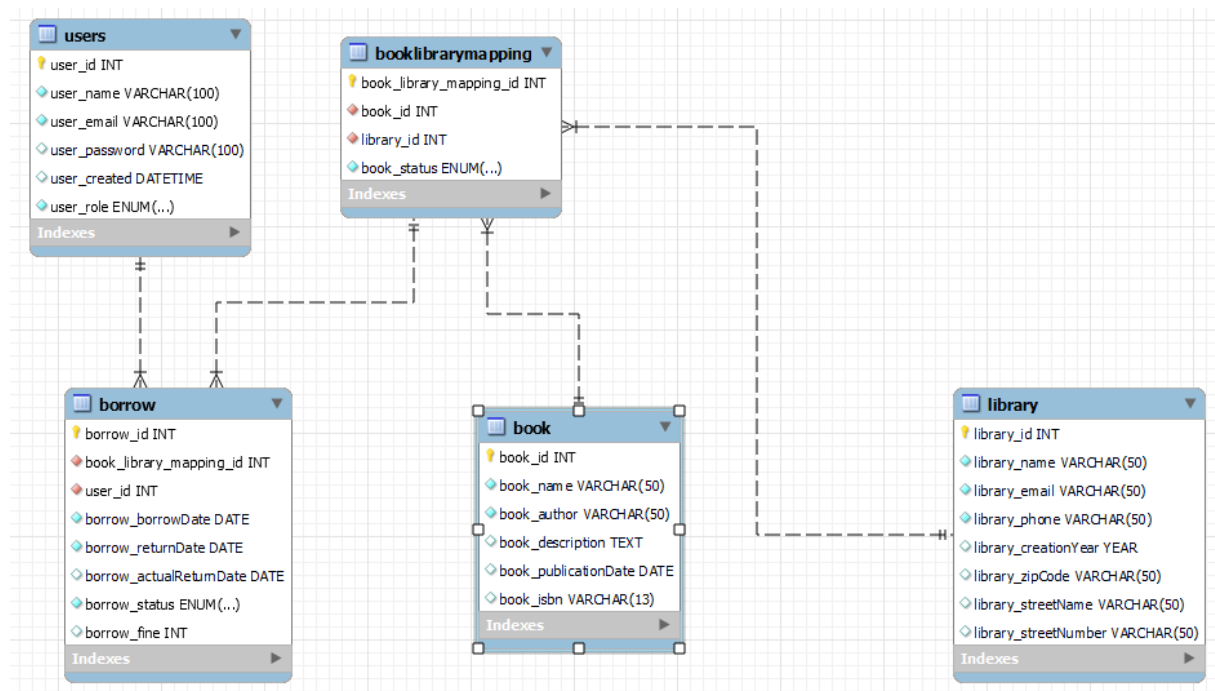
Database diagrams

---

## Entity-relationship diagram

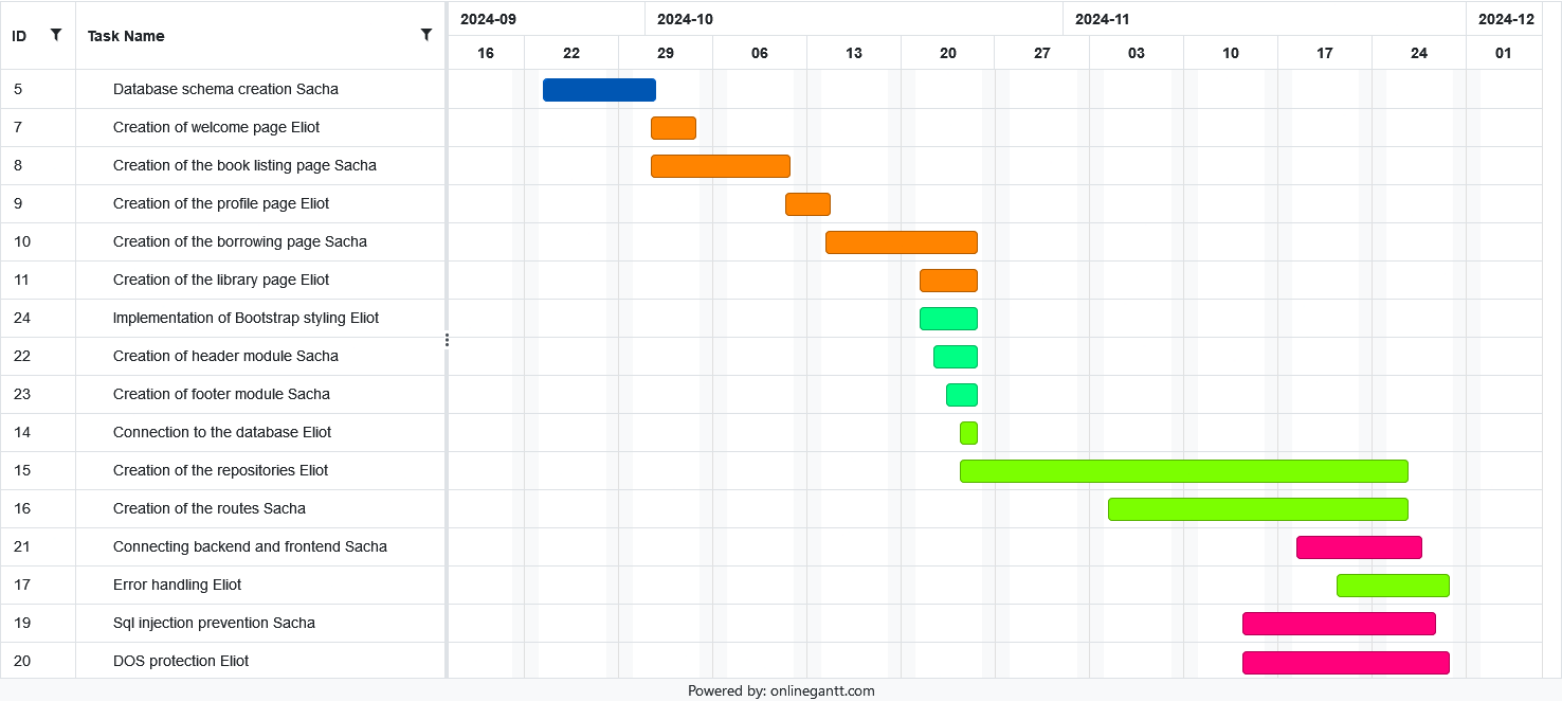


## Table structure diagram



Gantt diagrams

Planned gantt diagram



Legend:

	Clientside
	Styling
	Serverside
	Safety

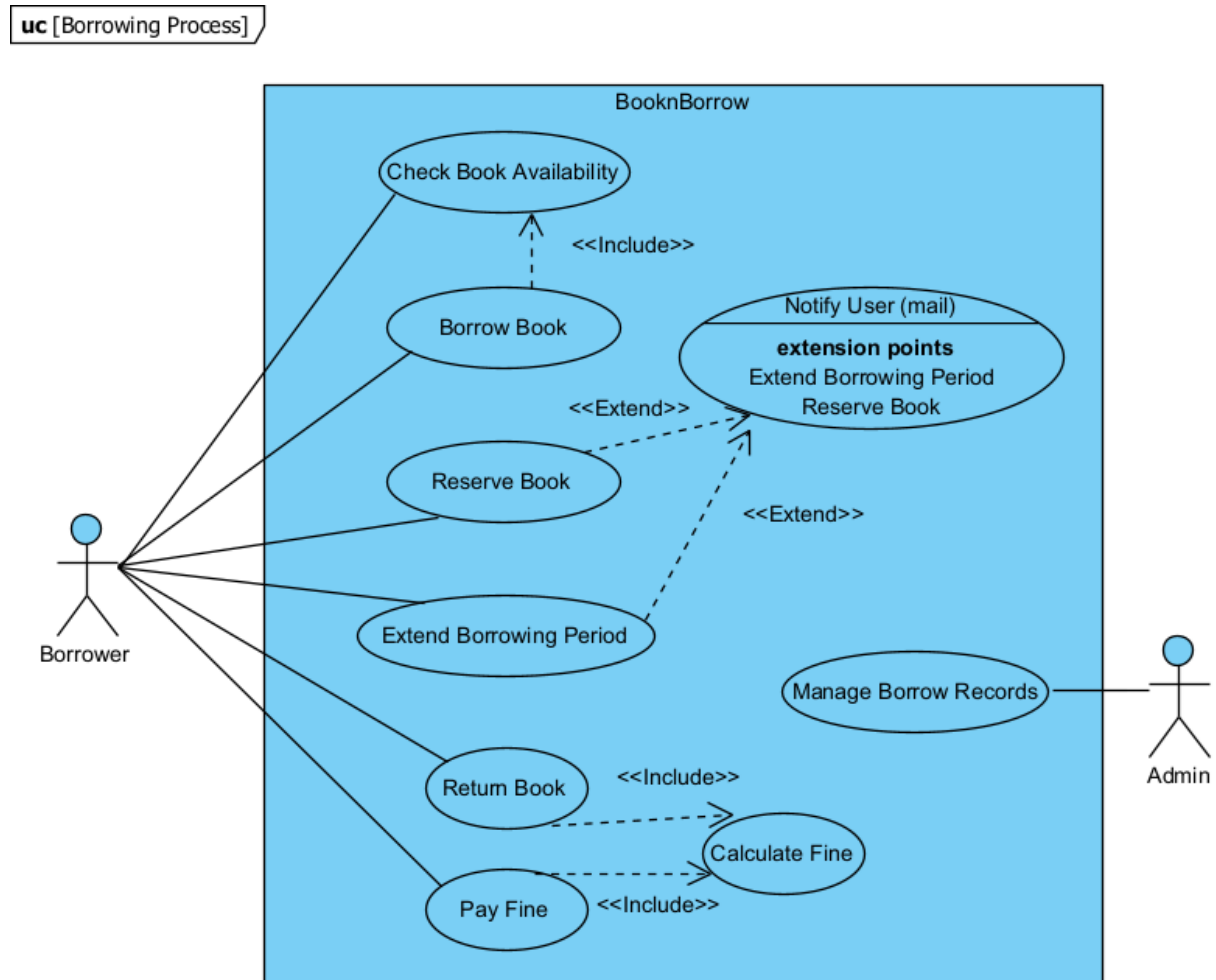
Actual gant diagram



	Frontend
	APIs
	Database

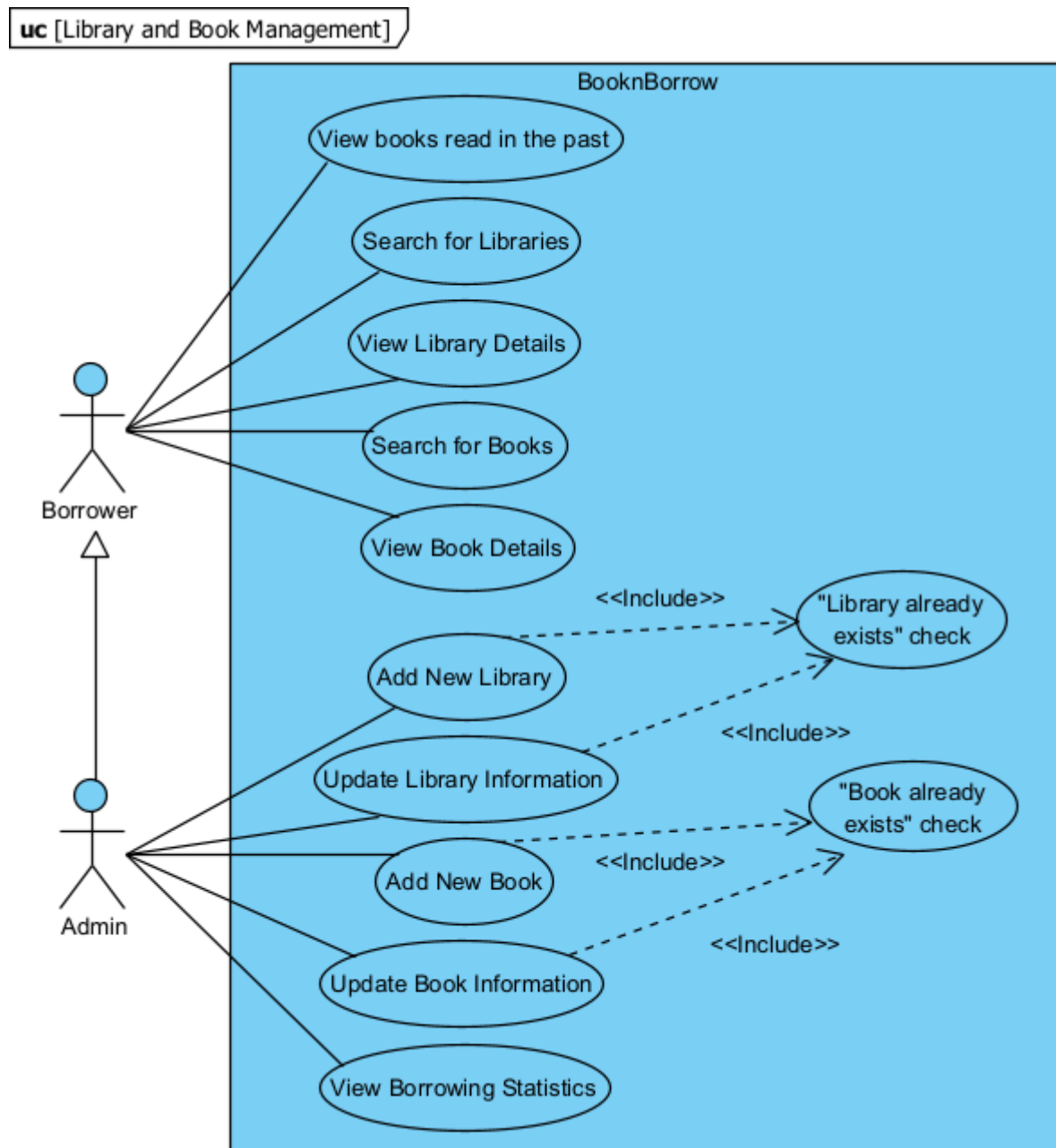
## Use case diagrams

### Borrowing Process Use Case Diagram



- **Actors:** User, Admin
- **Use Cases:** Borrow Book, Reserve Book, Return Book, Check Book Availability, Extend Borrowing Period, Pay Fine (User), Manage Borrow Records (Admin).

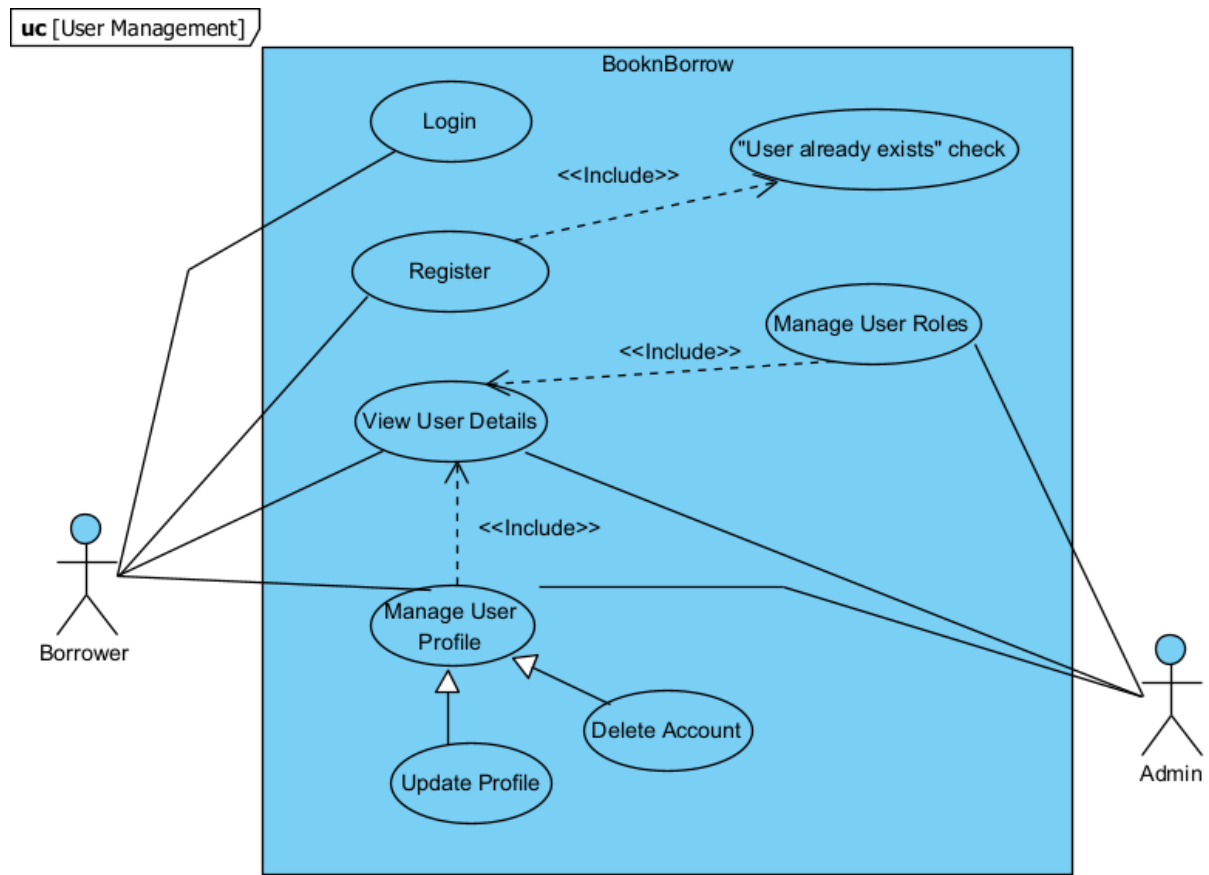
## Library and Book Management Use Case Diagram



- **Actors:** Admin, User
- **Use Cases:** Add New Library (Admin), View Library Details, Add New Book (Admin), View Book Details, View books read in the past, Update Book Information (Admin), Update Library Information (Admin), Search for Books, Search for Libraries, View Borrowing Statistics (Admin).



## User Management Use Case Diagram:

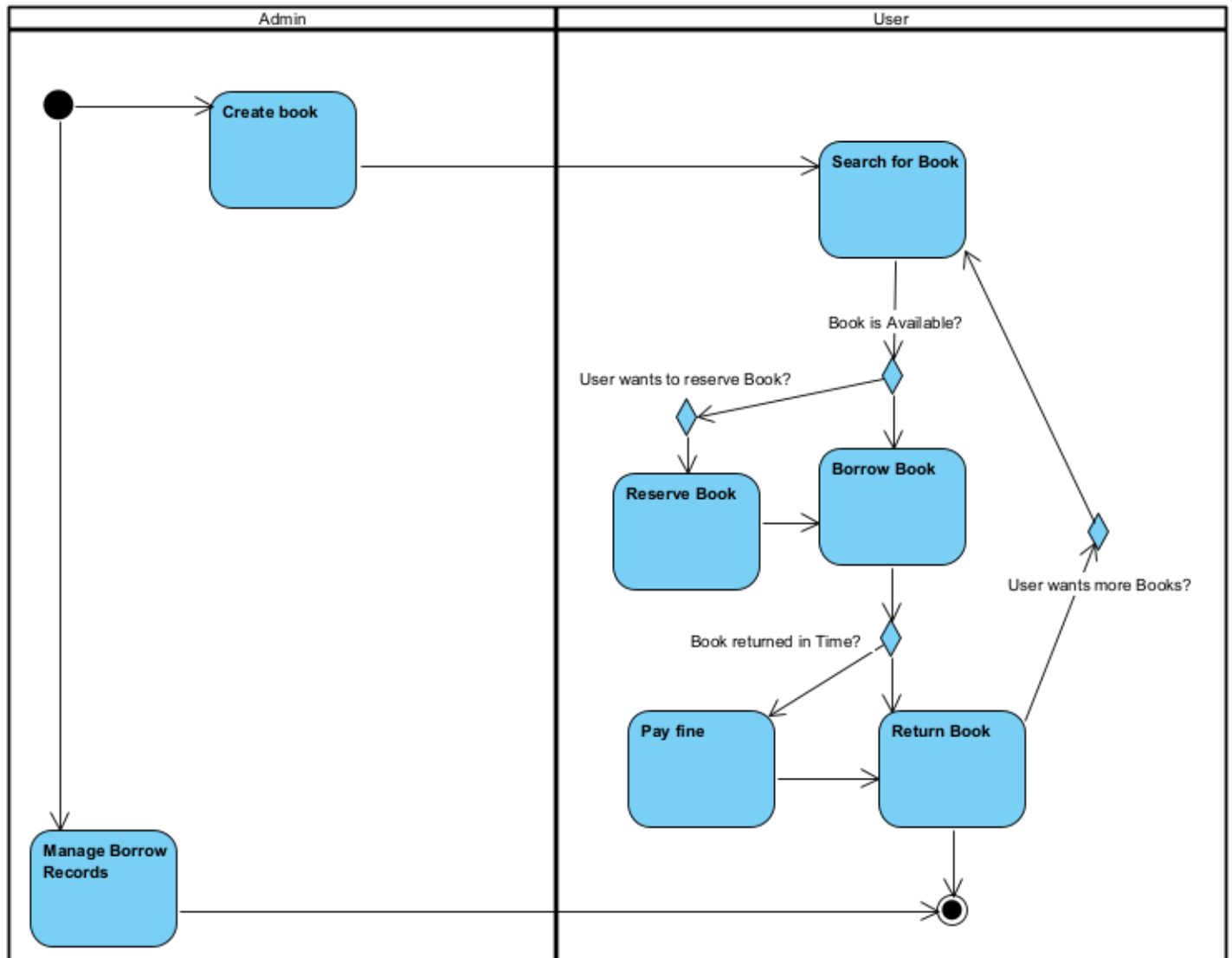


- **Actors:** Admin, User
- **Use Cases:** Register, Login, Update Profile, View User Details, Manage User Roles (Admin only), Delete Account, Manage Users (Admin only).

## Activity diagram

### Borrowing Process

act [Borrowing Process]

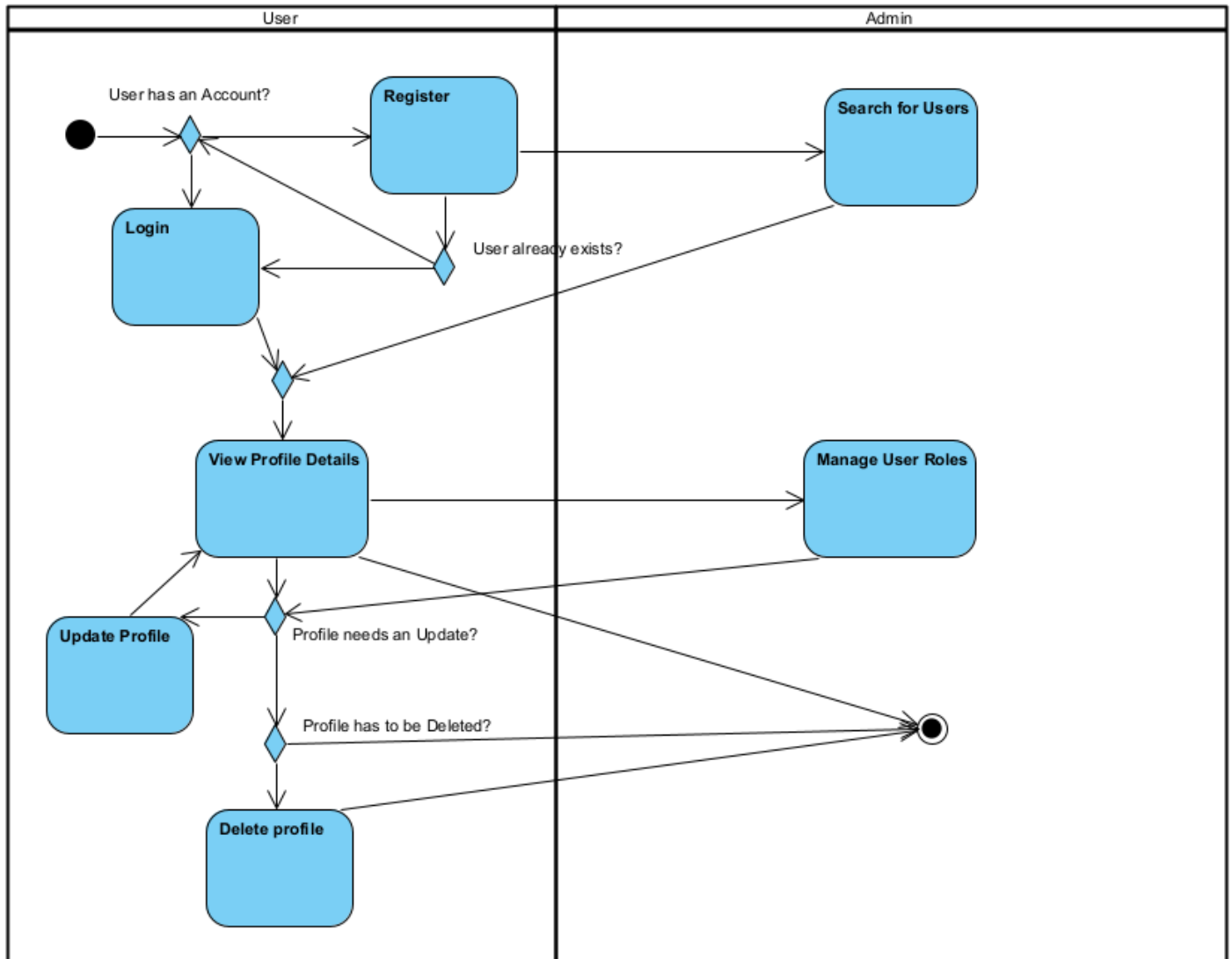


**Actors:** Admin, User

The Admin can initiate the process by creating a new book entry, which becomes accessible to the User for searching and borrowing. The User starts by searching for a book. If the book is available, the User can choose to borrow it or reserve it if needed. Once borrowed, the User must return the book on time. If returned late, a fine is imposed. The process also allows the User to borrow additional books after completing a transaction. Finally, the Admin manages the borrowing records to complete the process.

## User Management

act [User Management]



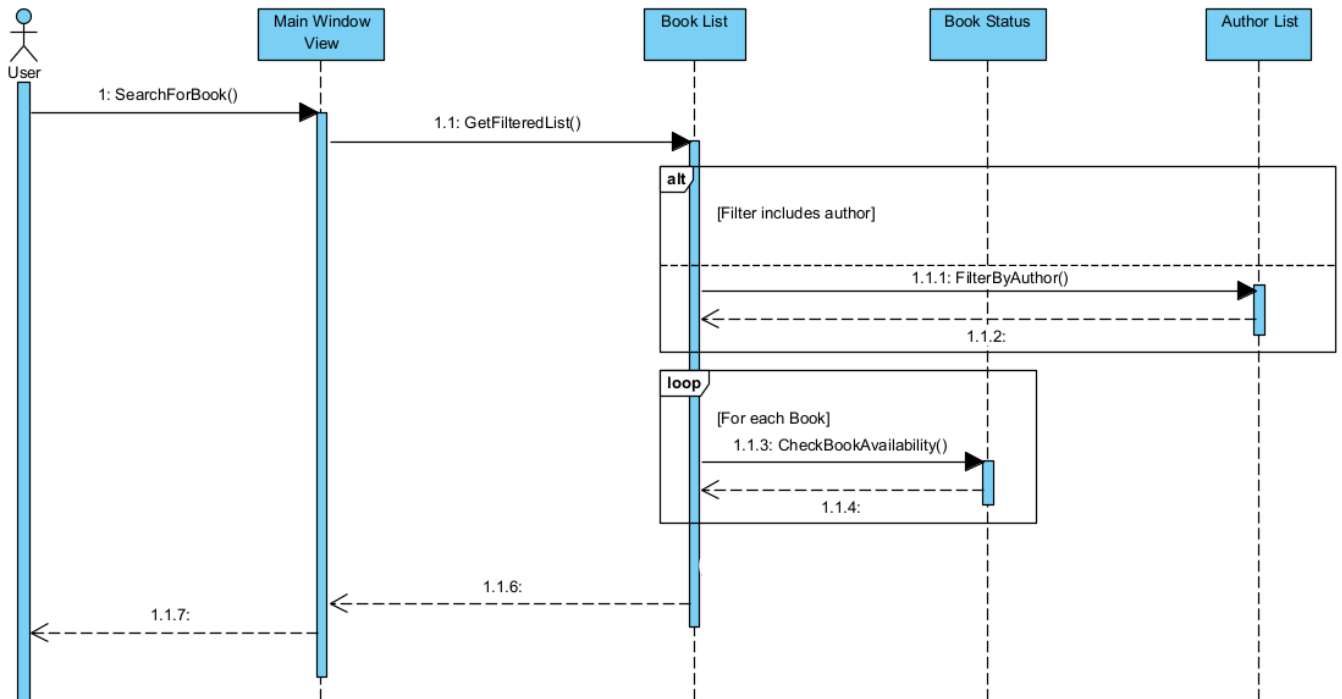
**Actors:** Admin, User

For Users, the process begins by checking if they already have an account. If they don't, they are prompted to register. If an account exists, they proceed to log in. Once logged in, the User can view their profile details. If necessary, the profile can be updated or deleted, depending on the User's needs.

On the Admin side, they can search for users and manage user roles, ensuring proper access control within the system. The diagram shows the various pathways that can be followed depending on the User's actions, such as updating or deleting their profile, while also illustrating the admin's ability to oversee user roles and account management.

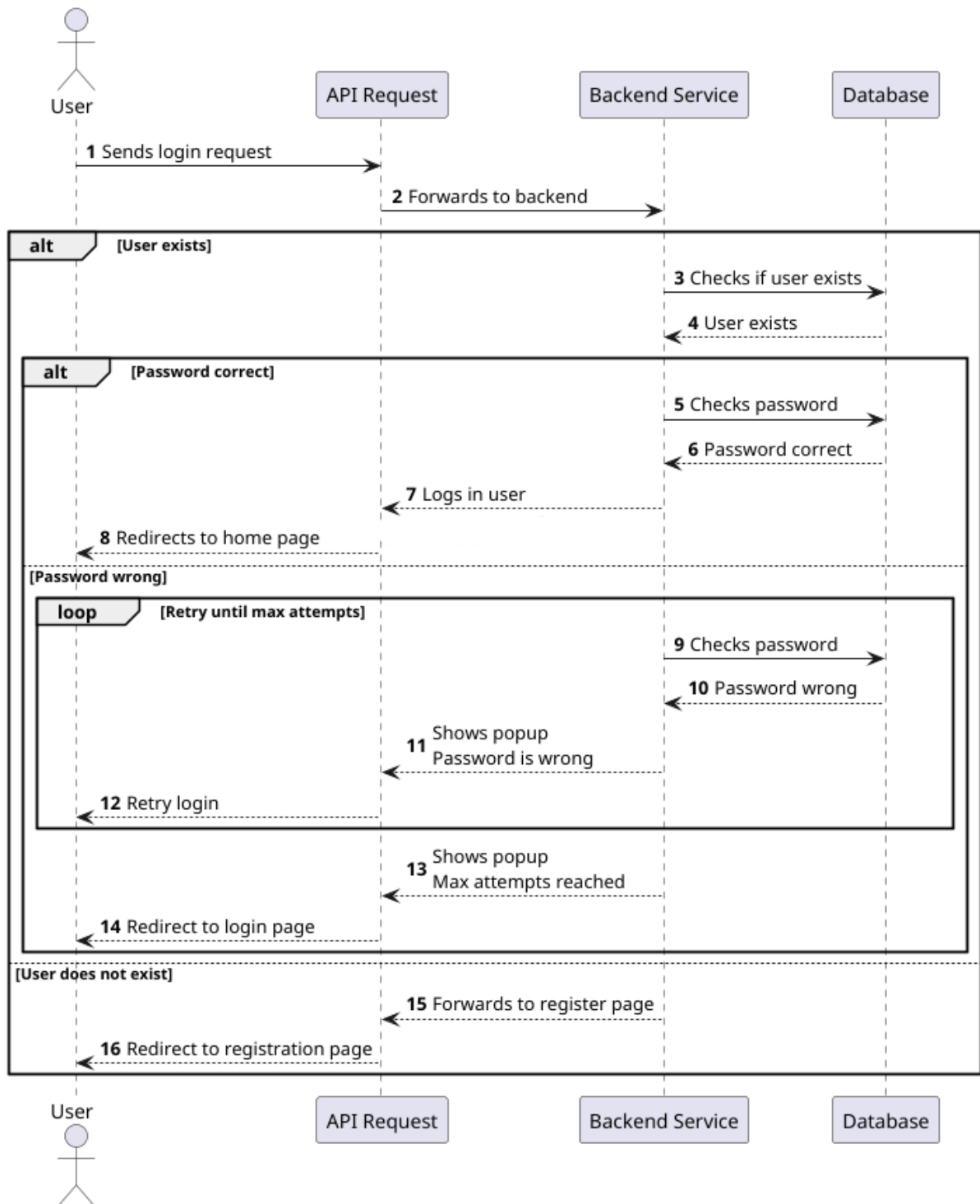
## Sequence diagrams

### Book Filing

**sd** [Book Filtering]

The Sequence begins with the User initiating a `SearchForBook()` request in the Main Window View. The Main Window View then sends a request to retrieve a filtered list of books (`GetFilteredList()`) to the Book List component. If the filter criteria include an author, the Book List interacts with the Author List to apply `FilterByAuthor()`, updating the list accordingly. Once filtered, the Book List updates, and the filtered list is returned back through the Main Window View to the User.

## Login

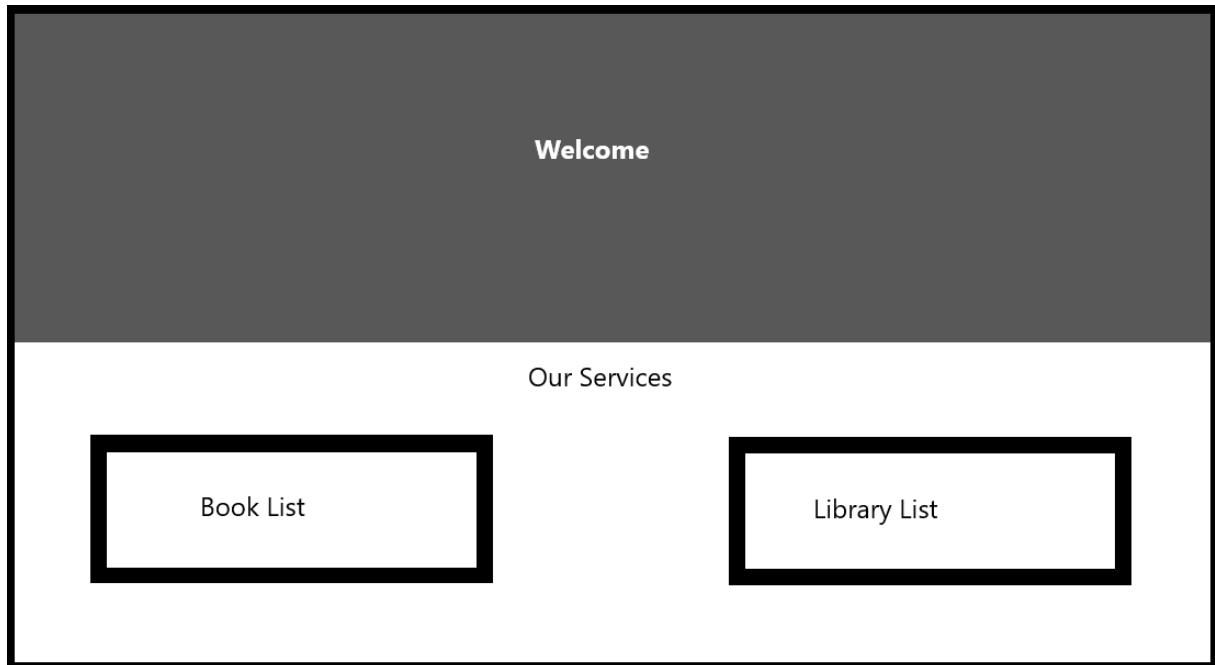


---

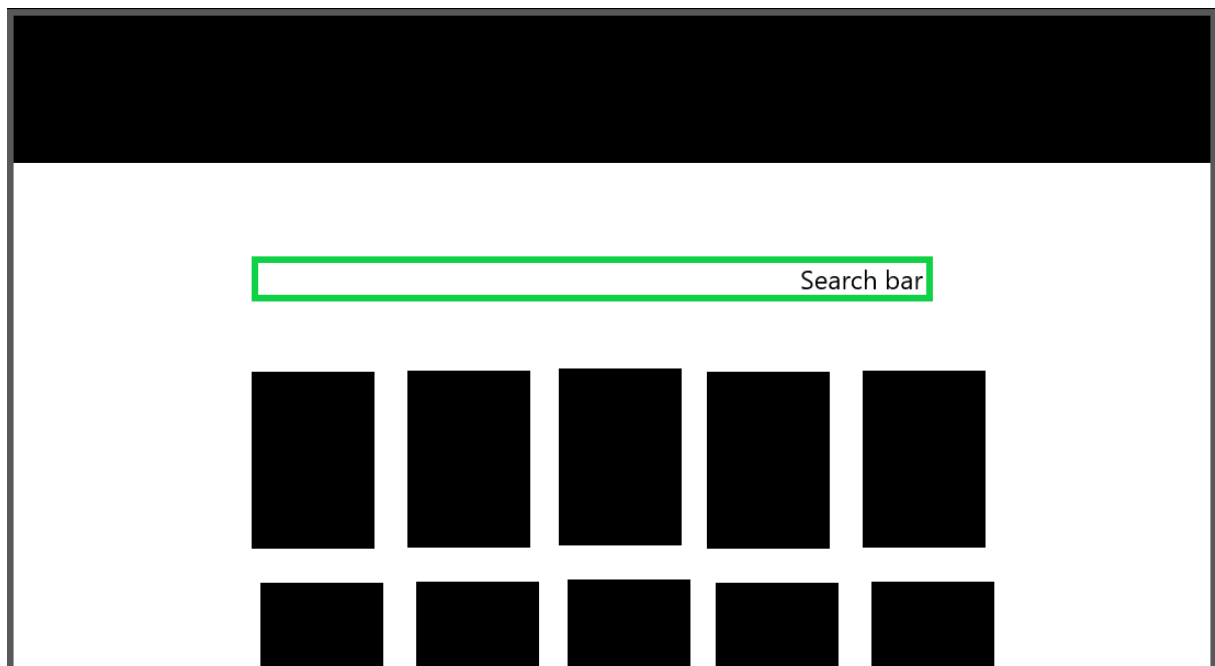
## Wireframe

---

### Front page

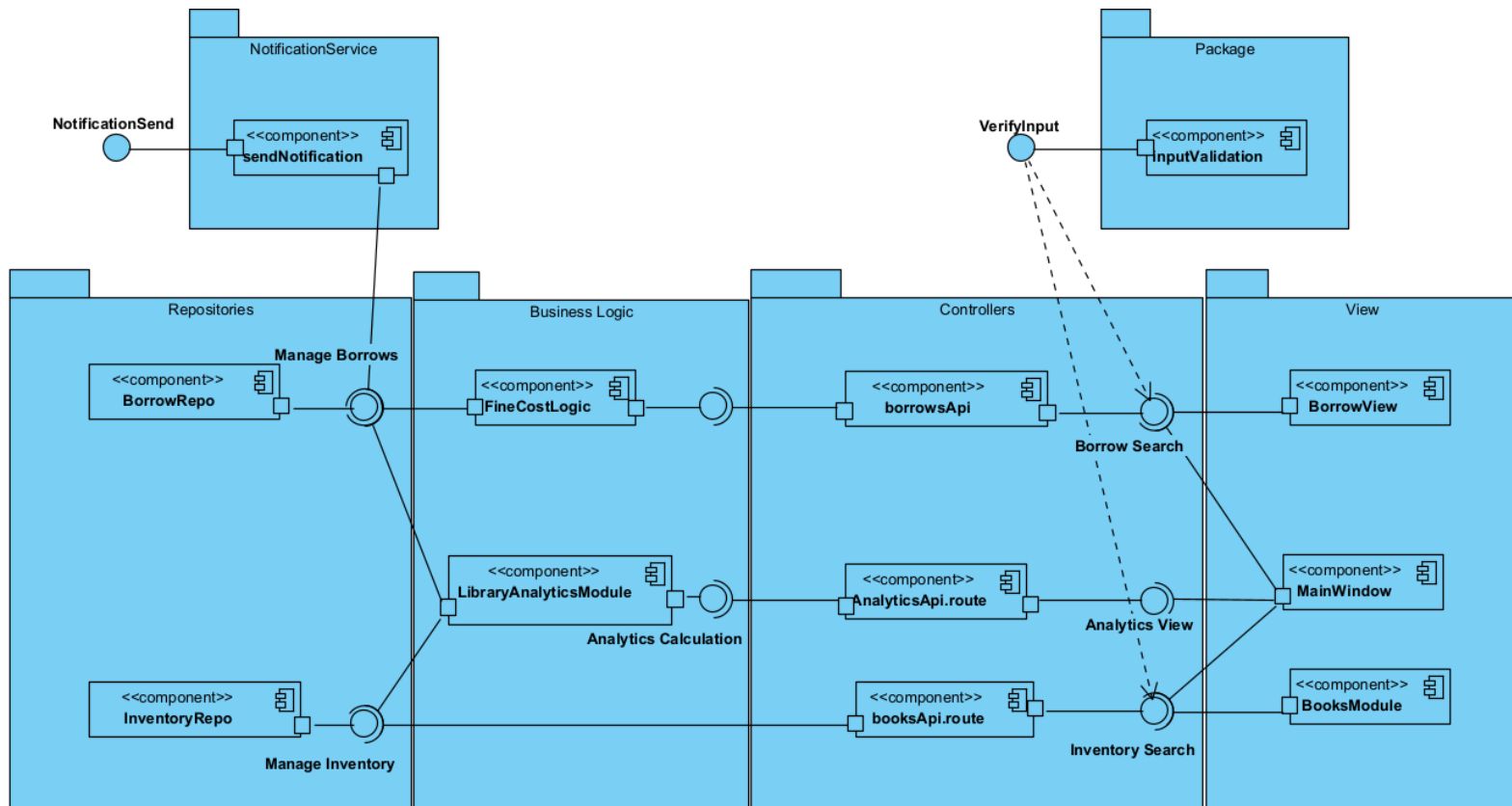


### Book list page



## Component diagrams

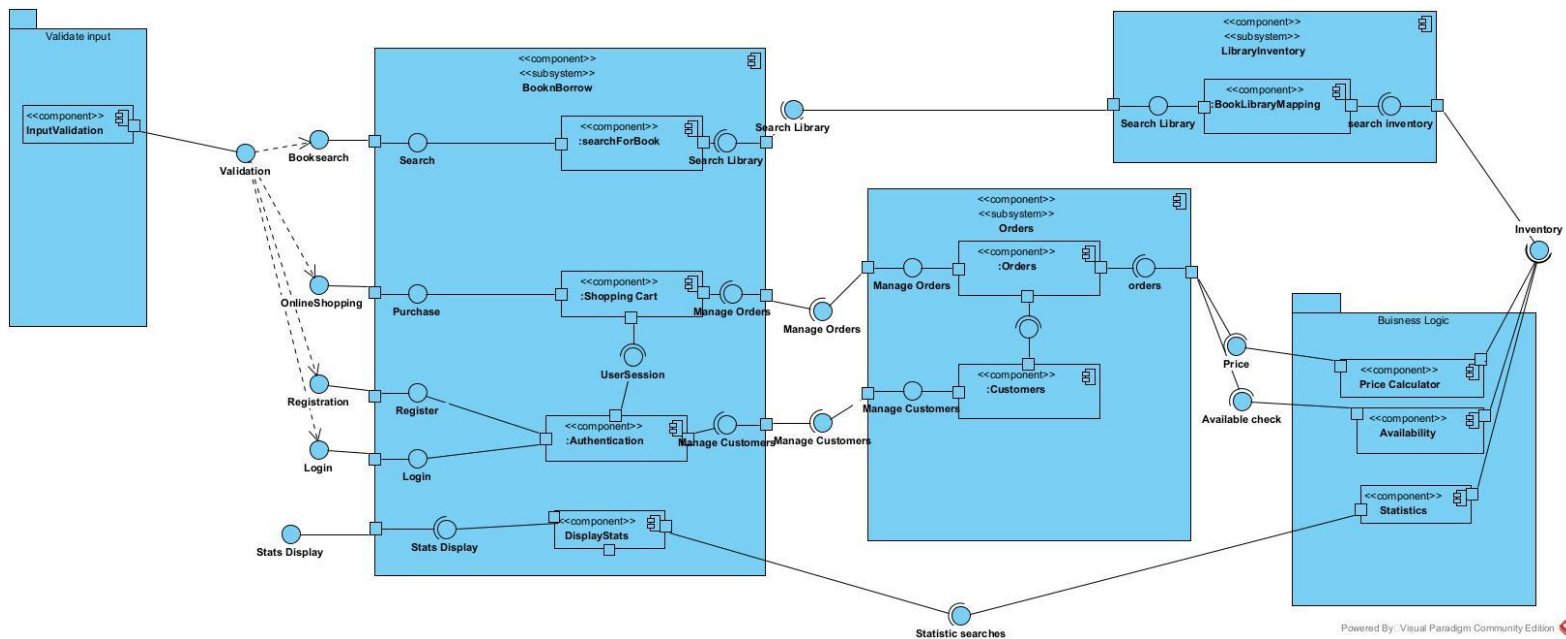
### Layers Component Diagram



#### Subsystems:

- **Repository Layer** consists of components like `books.repository`, `libraries.repository`, and `booklibrarymapping.repository`, which manage database interactions for specific entities.
- **Business Logic Layer** includes a `FineCostLogic` component to handle fine calculations.
- **Controllers Layer** contains API route components such as `booksApi.route` and `librariesApi.route`, which coordinate between the business logic and the user-facing modules.
- **View Layer** comprises modules like `BooksModule` and `LibraryModule`, responsible for presenting data and interacting with the controllers.
- **InputValidation package** makes the system immune to SQL injection

## User Actions



- **Input Validation**
  - Validates all user inputs to ensure data integrity and proper functionality across the system.
- **BooknBorrow Subsystem**
  - `SearchForBook`: Enables users to search for books available in the library.
  - `Shopping Cart`: Facilitates book borrowing or purchasing activities.
  - `Authentication`: Manages user login functionality.
  - `Register`: Handles new user registrations.
  - `DisplayStats`: Provides statistical data for users and system administrators.
- **Orders Subsystem**
  - `Orders`: Manages customer orders, including order placement and processing.
  - `Customers`: Stores and manages customer-related data for personalization and order tracking.
- **LibraryInventory Subsystem**
  - `BookLibraryMapping`: Maps and retrieves book data to check library availability.
- **Business Logic Subsystem**
  - `Price Calculator`: Calculates costs for book purchases or borrowing.
  - `Availability`: Checks inventory to confirm book availability.
  - `Statistics`: Gathers and presents analytical data for the system.



---

## Class diagram

---

Link to the diagram:

<https://github.com/Hormone4/BooknBorrow-Library-Management/blob/main/diagrams/class/class-diagram.png>

### Core Classes:

- Book, Library, Users, Admin, and Payment are the primary classes, with BookLibraryMapping, Borrow, and Reservation linking their relationships and functionality.

### Relationships:

- A Library holds many Books via BookLibraryMapping. The BookLibraryMapping tracks the status (available, borrowed, reserved) of each book in specific libraries.
- Users interact with books in two ways:
  - They can borrow a book through the Borrow class, which records dates (ongoing, finished, overdue) and calculates fines.
  - They can reserve a book using the Reservation class, which manages expiration dates and cancellations.
- Admin is a specialized role of Users, with additional responsibilities to manage users and generate reports.

### Interactions and Processes:

- Borrowing or reserving a book may lead to a Payment, which handles amounts, payment methods, and processing.
- Notifications are managed by the Notifiable interface, enabling the system to send alerts to users (e.g., overdue books or reservation updates).

### Inheritance and Methods:

- Admin extends Users with administrative functionalities.
- Shared methods such as getBookInfo() (in Book), processPayment() (in Payment), and updateStatus() (in BookLibraryMapping) ensure modular and reusable functionality.