**Software Requirement Engineering Project**

****

**Iraj Fatima BSEF23M502**

**Sajjal Fatima BSEF23M527**

**M.Talha BSEF23M523**

**Software Requirement Specification (SRS) for Library Management System**

**Table of Contents:**

1. **Introduction**
   * 1.1 Purpose
   * 1.2 Scope
   * 1.3 Definitions, Acronyms, and Abbreviations
   * 1.4 References
   * 1.5 Overview
2. **Overall Description**
   * 2.1 Product Perspective
   * 2.2 Product Functions
   * 2.3 User Characteristics
   * 2.4 Constraints
   * 2.5 Assumptions and Dependencies
3. **Specific Requirements**
   * 3.1 External Interfaces
   * 3.2 Functional Requirements
   * 3.3 Performance Requirements
   * 3.4 Logical Database Requirements
   * 3.5 Design Constraints
   * 3.6 Software System Quality Attributes
   * 3.7 Object-Oriented Models
4. **Appendices**
5. **Index**

**1. Introduction**

**1.1 Purpose**

The Library Management System (LMS) aims to streamline the operations of managing a library by providing an integrated platform to handle user registration, book inventory, borrowing and returning processes, overdue tracking, fine calculations, and reporting. The LMS is designed to reduce manual effort, ensure efficiency, and enhance the user experience for both library administrators and members.

**1.2 Scope**

This system will:

* Automate library processes such as registering users, managing book inventory, lending, and returning books.
* Provide an intuitive user interface for administrators and members.
* Track overdue books and calculate fines accurately.
* Generate reports on inventory, borrowing history, overdue books, and fines collected.
* Notify members about overdue books or fines via email or SMS.
* Support scalability to accommodate growing library needs.

**1.3 Definitions, Acronyms, and Abbreviations**

* **Admin**: A user with administrative privileges to manage library operations.
* **Member**: A registered user who can borrow, return, and manage books.
* **Book Inventory**: A database of books, including details like title, author, genre, and availability.
* **Borrowing History**: A log of books borrowed and returned by members.
* **Overdue Books**: Books not returned by the due date.
* **Fine Management**: System for calculating and tracking fines for overdue books.
* **Notification System**: Sends alerts about overdue books or fines.
* **LMS**: Library Management System.

**1.4 References**

* IEEE 830-1998 Standard, *Software Requirements Specification*.
* Example LMS systems for inspiration.

**1.5 Overview**

The LMS provides a comprehensive solution to handle library operations effectively. Admins will manage users and books, track borrowing and returning activities, and generate reports. Members will have access to borrow and return books, view borrowing history, and receive notifications. The system integrates with external notification services and ensures scalability and performance to meet library demands.

**2. Overall Description**

**2.1 Product Perspective**

The LMS is a standalone web-based application designed for modern libraries. It integrates with external systems for email and SMS notifications and operates on a secure, scalable architecture. The system supports multiple user roles (Admin and Member) and provides seamless functionality for each role.

**2.2 Product Functions**

**2.2.1 User Management**

* Admins can register new members, update profiles, and delete inactive members.
* Members can update their personal details and reset passwords.

**2.2.2 Book Inventory Management**

* Admins can add new books with details like title, author, genre, ISBN, and availability status.
* Update book details or mark books as removed when unavailable.
* View book inventory with search and filter options.

**2.2.3 Borrowing and Returning**

* Members can borrow books (subject to availability and borrowing limits).
* Admins can manage the borrowing process, including due date assignments.
* Books are marked as returned upon member submission.

**2.2.4 Overdue and Fine Management**

* Track overdue books and calculate fines based on library policies.
* Notify members about overdue books and unpaid fines via email/SMS.

**2.2.5 Reporting**

* Generate reports on book inventory, borrowing history, overdue books, and fines collected.
* Reports can be exported in formats like PDF or CSV for further analysis.

**2.2.6 Search and Filter**

* Search books by title, author, or genre.
* Filter borrowing history by member name or date range.

**2.3 User Characteristics**

* **Admins**: Expected to have basic computer literacy and understanding of library operations.
* **Members**: General library users with minimal technical knowledge. The interface will be simple and intuitive for easy navigation.

**2.4 Constraints**

* Must support up to 500 concurrent users.
* Ensure compatibility with modern web browsers (e.g., Chrome, Firefox, Safari).
* Operate on standard desktop and mobile devices.

**2.5 Assumptions and Dependencies**

* Users have access to the internet.
* Integration with notification services (e.g., email/SMS) is reliable.
* The database server is maintained regularly for performance.

**3. Specific Requirements**

**3.1 External Interfaces**

* **Email/SMS Notification Service**: Integrates with services like SendGrid or Twilio for member notifications.
* **Database Management System (DBMS)**: A relational database will store user and book data.
* **Web Browser**: The system is accessible through modern browsers.

**3.2 Functional Requirements**

1. **Admin Functions**
   * Add, update, or remove members.
   * Add, update, or remove books.
   * Track overdue books and notify members.
   * Generate and export reports.
2. **Member Functions**
   * Borrow and return books.
   * View borrowing history.
   * Receive notifications about overdue books or fines.
     1. **Detailed Use Case Descriptions**

* **UC1: Login and Registration(by M.Talha)**

**Scope**: Library Management System  
**Level**: User Goal  
**Stakeholders and Interests**:

* **User**: Wants seamless access to the system.
* **Admin**: Needs to verify user access rights.
* **System**: Ensures secure login and user identity management.

**Primary Actor**: User  
**Secondary Actor**: Admin  
**Offstage Actor**: Authentication Service

**Preconditions**:

* The system must be operational.
* User/Admin credentials must exist for login.

**Postconditions**:

* User/Admin is logged into the system with appropriate privileges.

**Main Success Scenario**:

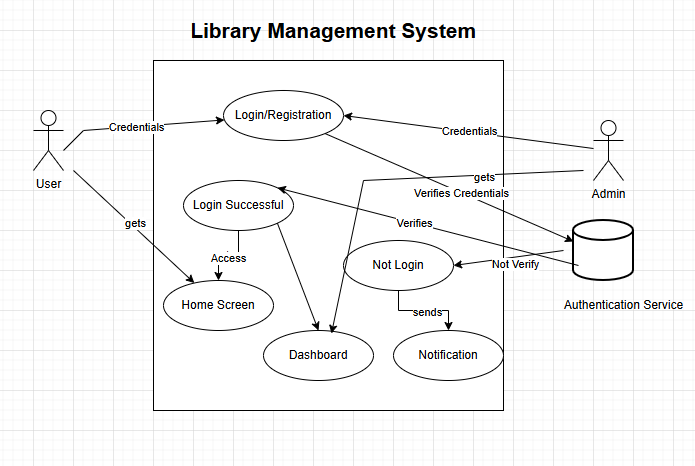
1. User accesses the system and selects **Login/Register**.
2. System prompts for credentials or registration details.
3. User provides valid credentials or completes registration.
4. System verifies and logs the user in, displaying the dashboard.

**Alternative Scenario**:

* **Invalid Credentials**: System prompts for re-entry or password recovery.
* **Registration Fails**: User fails validation checks (e.g., incomplete data).

**System Functions**

* + - Authenticate users based on roles (Admin, Member).
    - Calculate fines for overdue books.



* **UC2: Add Book: (by Sajjal Fatima)**

**Scope:** Library Management System

**Primary Actor:** Librarian

**Secondary Actor:** Library Management System (LMS)

**Goal:** To successfully add a new book entry into the library system.

**Preconditions:**

1. The librarian is logged into the library management system.

2. The librarian is on the "Add A Book" dashboard interface.

**Trigger:** The librarian chooses to add a new book to the library collection by navigating to the "Add A Book" form.

**Main Success Scenario (Basic Flow):**

1. The system displays the "Add A Book" form on the dashboard.

2. The librarian inputs the Book Title into Text Box 1.

3. The librarian inputs the Author Name into Text Box 2.

4. The librarian selects the Genre from the Dropdown list.

5. The librarian inputs the Available Number of Copies into Text Box 3.

6. The librarian enters the Rack Number (location) into Text Box 4.

7. The librarian enters the Shelf Number (location) into Text Box 5.

8. The librarian uploads the Book Image (front cover) using the File Input field.

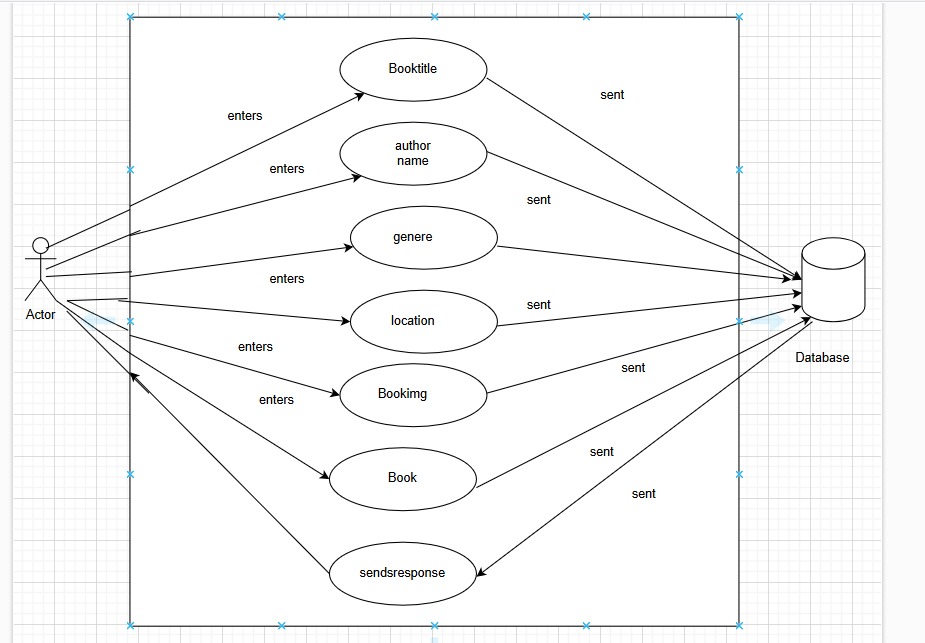
9. The librarian clicks the "Add Book" Button to submit the form.

10. The system validates all the input fields to ensure they are complete and correctly formatted.

11. If validation passes:

* + The system saves the new book record in the database.
  + The system updates the availability status based on the number of copies.
  + The system confirms successful addition of the book and provides a success message.

12. The librarian sees the confirmation message: "Book successfully added to the library system."



* **UC3: Update Book (by Iraj Fatima)**

**Scope**: Library Management System  
**Level**: User Goal  
**Stakeholders and Interests**:

* **Admin**: Needs to update Book details.
* **System**: Ensures the successful update.

**Primary Actor**: Librarian  
**Secondary Actor**: Library management system  
**Offstage Actor**: Authentication Service

**Preconditions**:

* The librarian must be logged into the Library Management System.
* The book to be updated must exist in the library database.

**Postconditions**:

* User/Admin is logged into the system with appropriate privileges.

**Main Success Scenario**:

1. User accesses the system and selects **Login/Register**.
2. System prompts for credentials or registration details.
3. User provides valid credentials or completes registration.
4. System verifies and logs the user in, displaying the dashboard.

**Alternative Scenario**:

* **Invalid Credentials**: System prompts for re-entry or password recovery.
* **Registration Fails**: User fails validation checks (e.g., incomplete data).

**System Functions**

* + Authenticate users based on roles (Admin, Member).
  + Calculate fines for overdue books.

**Extensions (Alternate Flows):**

1. Field Validation Errors:

* + - If any mandatory fields are left blank, the system highlights the incomplete fields and displays an error message: "Please complete all required fields."
    - The librarian corrects the input and resubmits the form.

2. Invalid Number of Copies (Text Box 3):

* + - If the number of copies entered is not a valid positive integer, the system displays an error: "Number of copies must be a positive whole number."
    - The librarian corrects the input and resubmits.

3. Invalid File Upload:

* + - If the uploaded file is not an image or exceeds the file size limit, the system displays an error: "Please upload a valid image file (JPG, PNG)."
    - The librarian uploads a valid image file.

4. System Error (Database Issue):

* + - If the system fails to save the book due to a database error, it displays: "An error occurred while adding the book. Please try again later."
    - The librarian retries after some time.

**Postconditions:**

1. The new book is successfully added to the library system.

2. The library system reflects the book's details, availability status, and location.

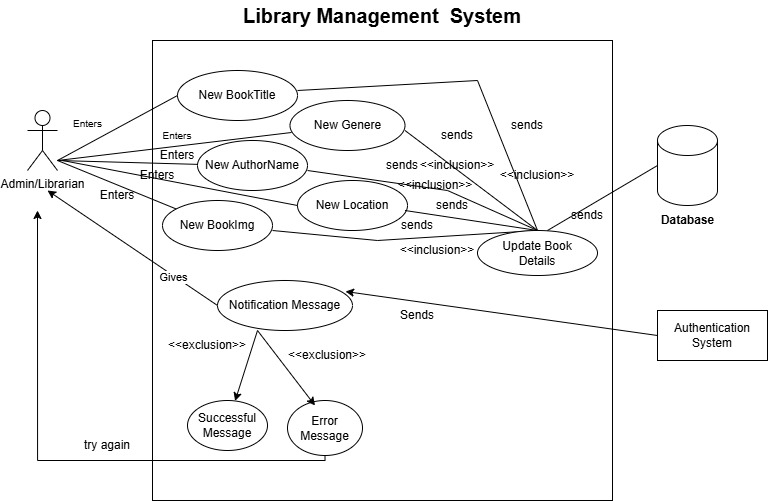
**Business Rules:**

1. All input fields must be validated before the book can be added.

2. The number of copies must be a positive integer.

3. Only image files (e.g., JPG, PNG) can be uploaded for the book cover.

4. Book title and author name must be non-empty.

****

**3.3 Performance Requirements**

* Response time should not exceed 2 seconds under peak load.
* The system should handle up to 500 concurrent users seamlessly.

**3.4 Logical Database Requirements**

* **Tables**:
  + Members: Stores member details.
  + Books: Stores book details.
  + Borrowing History: Logs borrowing and returning activities.
  + Fines: Tracks overdue fines.

**3.5 Design Constraints**

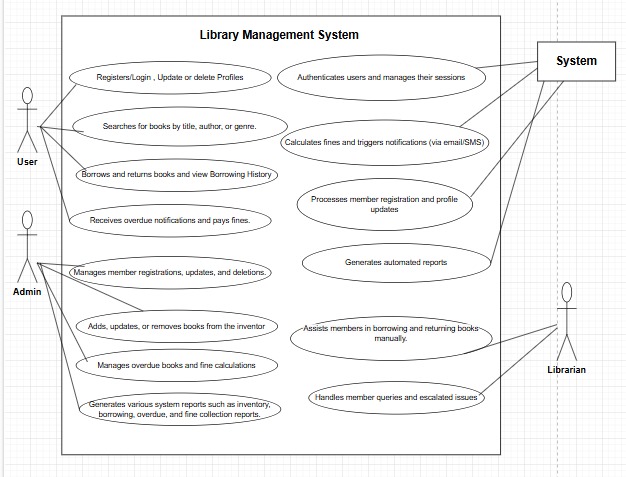
* Must be web-based and developed using secure frameworks.
* Ensure scalability to support future library growth.

**3.6 Software System Quality Attributes**

* **Reliability**: High uptime and minimal downtime.
* **Scalability**: Supports growing numbers of users and books.
* **Security**: Implements encryption for sensitive data and role-based access control.
* **Usability**: Intuitive interface for non-technical users.

**3.7 Object-Oriented Models**

1. **Use Case Diagram**: Represents actions like "Borrow Books," "Return Books," and "Generate Reports."



1. **Domain Model**:

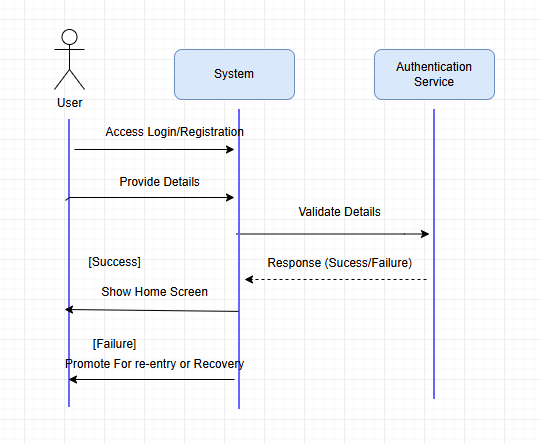


1. **Class Diagram**: Includes classes like Member, Admin, Book, and BorrowingHistory with attributes and methods.

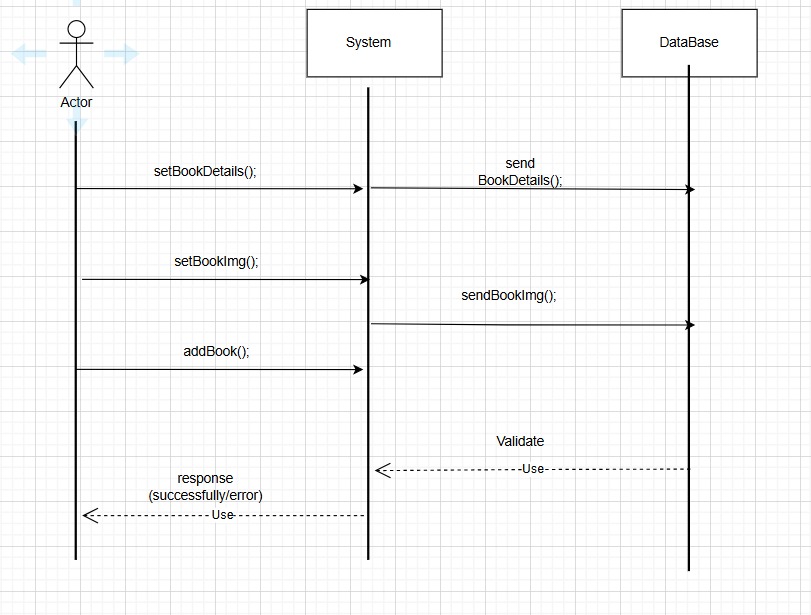


1. **System Diagram**: Demonstrates interactions during borrowing and returning processes.

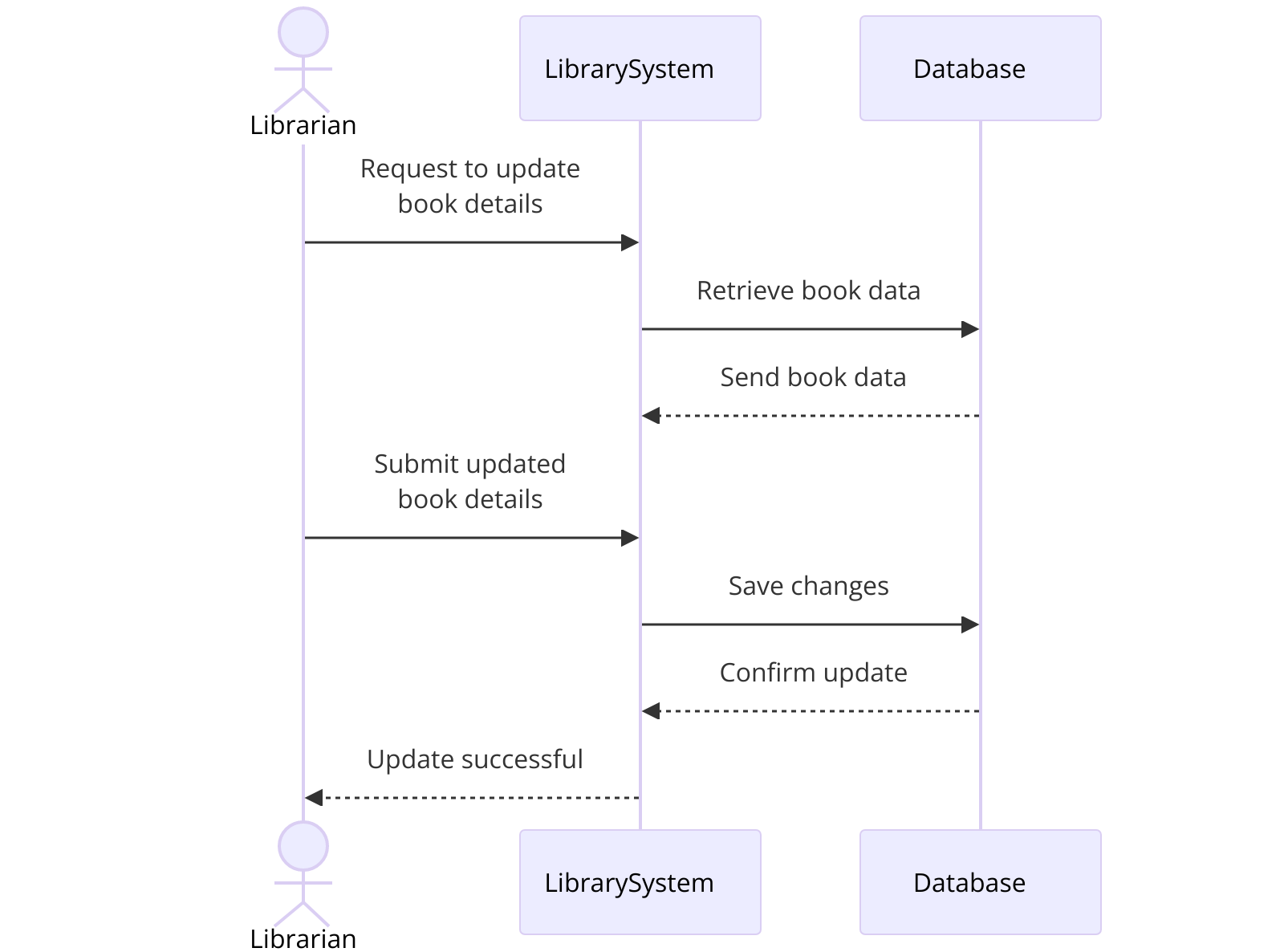
**UC1: System Diagram (SD)** (By M. Talha)

****

**UC2: System Diagram (SD)** (By Sajjal Fatima)

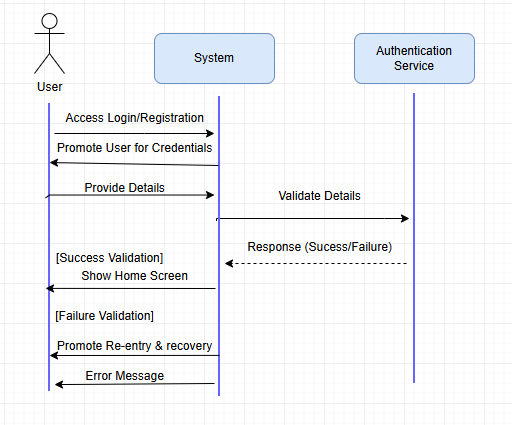


**UC3: System Diagram (SD)** (By Iraj Fatima)

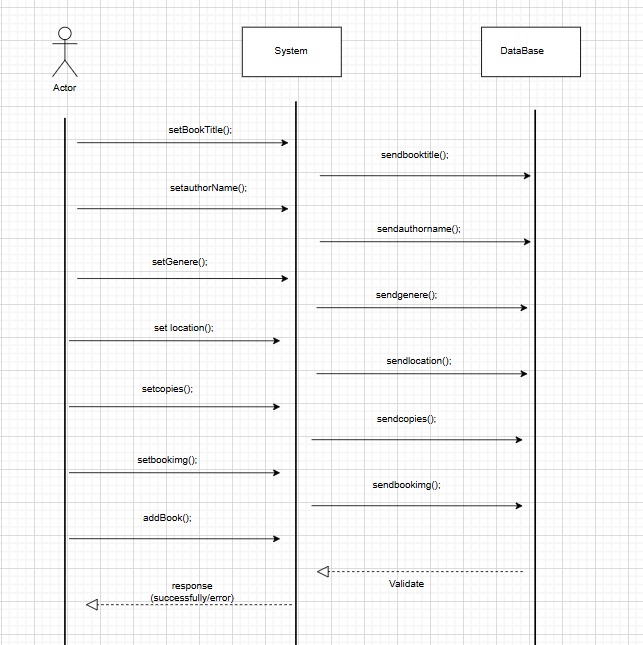


1. **System Sequence Diagrams:**

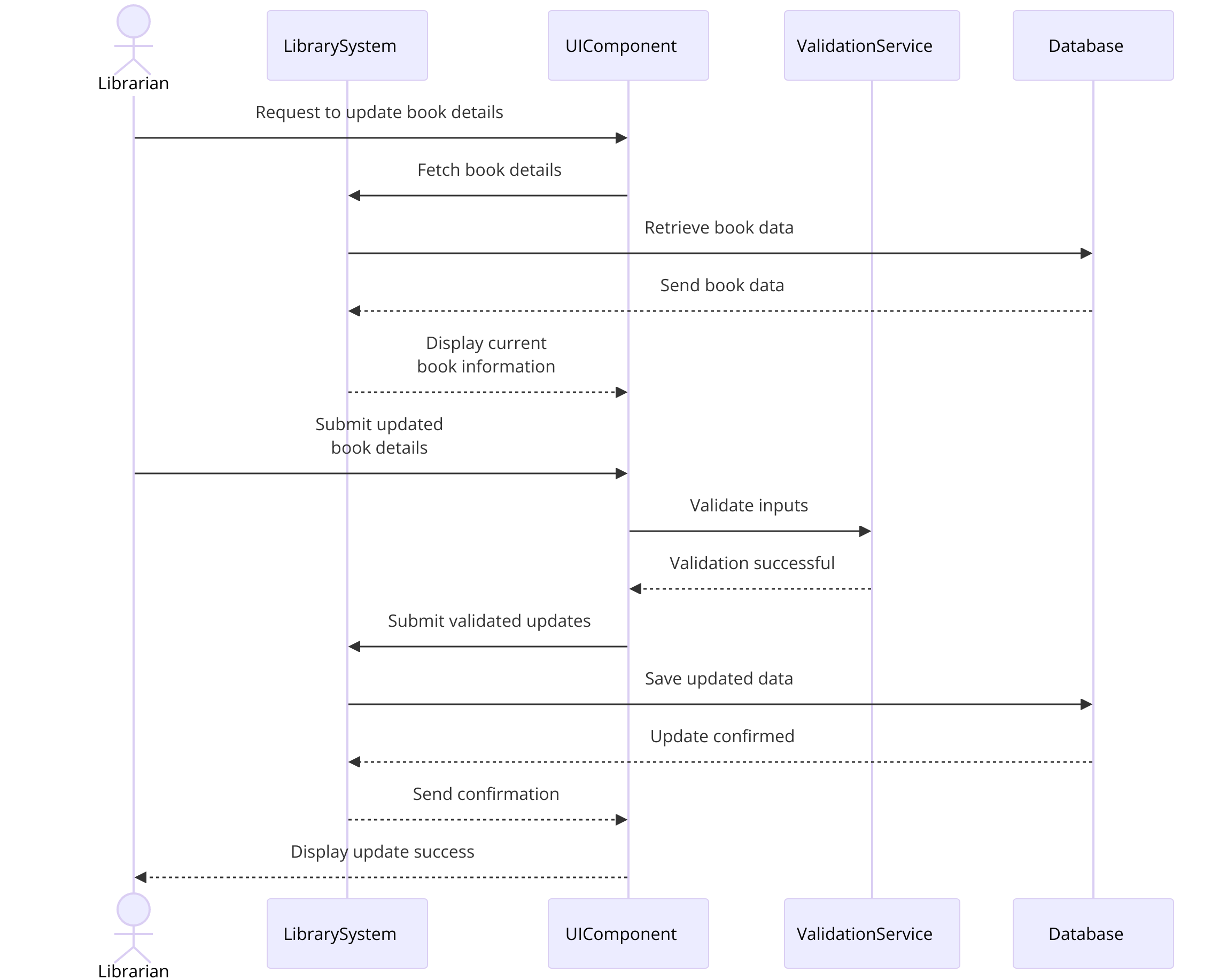
UC1: System Sequence Diagram (SD)(By M.Talha)



UC2: System Sequence Diagram (SD)(By Sajjal Fatima)



UC3: System Sequence Diagram (SD)(By Iraj Fatima)



**4. Appendices**

**4.1 Database Schema**

* **Members Table**: MemberID, Name, Email, Phone, Address.
* **Books Table**: BookID, Title, Author, Genre, Availability.
* **BorrowingHistory Table**: BorrowID, MemberID, BookID, BorrowDate, ReturnDate.
* **Fines Table**: FineID, MemberID, Amount, Status.

**4.2 Glossary of Terms**

* **Admin**: A user responsible for managing library operations.
* **Member**: A registered library user.
* **Borrowing Limit**: The maximum number of books a member can borrow.
* **Fine**: A penalty for late book returns.

**4.3 External System Integration**

* **Email Notifications**: For overdue reminders.
* **SMS Notifications**: For urgent alerts.

**5. Index**

* **Introduction**………………………………………………………… Page 2
* **Overall Description**……………………………………………. Page 4
* **Specific Requirements……………………………..** Page 6
* **Appendices**……………………………………………………………..Page 10