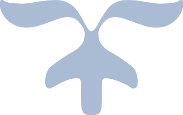


Library Management

System

SRS Document



Abdelrahman Mahmoud 230105776 Gamal Ramadan 230105450

Fady Mahrous 230103236

Kirellos Samir 230102253

Mohamed Ahmed 230101143

November 5,2024 TO

Dr. Ahmed Mostafa

**Table of Contents**

1. Introduction ....................................................................................................................................................... 1

1.1 Purpose ......................................................................................................................................................... 1

1.2 Document Conventions .......................................................................................................................... 1

1.3 Intended Audience and Reading Suggestions ................................................................................ 1

1.4 Project Scope .............................................................................................................................................. 1

1.5 References ................................................................................................................................................... 1

2. Overall Description .......................................................................................................................................... 2

2.1 Product Perspective ................................................................................................................................. 2

2.2 Product Features ....................................................................................................................................... 2

2.3 User Classes and Characteristics......................................................................................................... 2

2.4 Operating Environment .......................................................................................................................... 3

2.5 Design and Implementation Constraints ......................................................................................... 3

2.6 User Documentation ................................................................................................................................ 3

2.7 Assumptions and Dependencies ......................................................................................................... 4

3. System Features ................................................................................................................................................ 4

3.1 User Registration & check........................................................................................................................5

Description & Priority........................................................................................................................ 5

3.2 User Login & Validate account............................................................................................................. 7

Description & Priority.....................................................................................................................7

3.3 Search ............................................................................................................................................................ 9

Description & Priority.................................................................................................................... 9

3.4 Pay for book ............................................................................................................................................... 10

Description & Priority.................................................................................................................... 10

3.5 Edit User ……………………………………................................................................................................... 11

Description & Priority................................................................................................................... 11

3.6 Add Penalty................................................................................................................................................. 12

Description & Priority................................................................................................................... 12

3.7 borrow……………………............................................................................................................................. 14

Description & Priority.................................................................................................................... 14

3.8 Add User, check users............................................................................................................................ 16

Description & Priority.................................................................................................................... 16

3.9 Delete User….............................................................................................................................................. 17

Description & Priority.................................................................................................................... 17

3.10 Reservation…........................................................................................................................................... 18

Description & Priority.................................................................................................................... 18

3.11 Create book............................................................................................................................................... 20

Description & Priority.................................................................................................................... 20

3.12 Edit book.................................................................................................................................................... 21

Description & Priority.................................................................................................................... 21

3.13 Delete book......................................................................................................................................... 22

Description & Priority.................................................................................................................... 22

4. External Interface Requirements .............................................................................................................. 23

4.1 User Interfaces ........................................................................................................................................ 23

4.2 Hardware Interfaces ............................................................................................................................. 24

4.3 Software Interfaces ............................................................................................................................... 24

4.4 Communication Interfaces ................................................................................................................. 25

5. Other Nonfunctional Requirements ....................................................................................................... 25

5.1 Safety Requirements ............................................................................................................................. 25

5.2 Security Requirements ........................................................................................................................ 25

5.3 Software Quality Attributes ............................................................................................................... 25

**1. Introduction**

**1.1 Purpose**

The purpose of this website is to serve as a digital platform for managing a university's library system, catering to two main user roles: Admin and Student. Administrators oversee library operations, including managing books, student accounts, and borrowing records, ensuring data integrity and system performance. Students can search for books, check availability, borrow items, and track their borrowing history through a user-friendly interface. By streamlining library management and improving access to resources, the platform enhances efficiency and the overall user experience for both administrators and students.

**1.2 Document Conventions**

This Software Requirements Specification (SRS) follows the following standards and typographical conventions:

* **Font**: The default font used throughout the document is Cambria.
* **Highlighting**: Important terms or concepts are highlighted using bold or italic

formatting.

* **Priority Inheritance**: It is assumed that the priorities assigned to higher-level

requirements are inherited by detailed requirements unless explicitly stated otherwise.

**1.3 Intended Audience and Reading Suggestions**

This document is intended for developers, testers, and project managers involved in the development of the University Library Management System. It is also aimed at stakeholders, such as university administrators and librarians, who wish to understand the system's requirements and features.

**1.4 Project Scope**

The University Library Management System is a web-based application designed to streamline library operations and enhance user interaction with library resources. The system provides role-based access, allowing administrators to manage books and user accounts through CRUD operations, impose penalties, and oversee borrowing and purchasing activities. Students can search for books, check availability, borrow or purchase books, and reserve unavailable ones, ensuring comprehensive access to library services.

**1.5 References**

* [User Stories](file:///C:/Users/original/Downloads/user_stories.md)

**2. Overall Description**

**2.1 Product Perspective**

The University Library Management System is a standalone web application that provides users with a responsive web interface for seamless interaction. It features a backend server that processes user requests, manages user roles and permissions, and maintains system data. The system connects to a library database to fetch, update, and manage book and user information. Administrators can perform tasks such as managing books and user accounts, imposing penalties, and monitoring system activities, while students can borrow, purchase, and reserve unavailable books, ensuring efficient library operations and enhanced user experience.

**2.2 Product Features**

The University Library Management System will include the following features:

* **User Registration:** Allows students and administrators to create and manage accounts.
* **Book Borrowing, Purchasing, and Reservation:** Students can borrow books, purchase them, or reserve unavailable books for future use.
* **User Authentication:** Ensures secure access to the system through login credentials.
* **Role-Based Access:** Grants different levels of functionality for administrators and students based on their roles.
* **CRUD Methods:** Enables administrators to create, read, update, and delete books and user accounts.
* **Penalty Management:** Allows administrators to impose penalties on students for overdue books or other violations.
* **Search Functionality:** Lets students search for books by title, author, or category.

**2.3 User Classes and Characteristics**

The system will have the following user classes:

* **User:** A person with an account on the platform. Users have a role (e.g., Student or Admin) and can log in and log out. Users have personal information such as name, email, phone, password, and age.
* **Admin:** A user responsible for managing the library system. Admins can add, remove, and update books. They can also add, remove, and update student accounts. Admins have the ability to impose penalties on students and manage payments.
* **Student:** A user who can search, view, borrow, purchase, and reserve books from the library. Students can also make payments for fines or purchases. They have a list of books and can view detailed book information.
* **Book:** Represents an item available in the library. Each book has a unique ID, title, description, time duration for borrowing, and is associated with a category.
* **Category:** A classification used to organize books. Each category has a name, description, and a list of books associated with it.
* **Borrow:** Represents the borrowing activity of a student. It includes details like the borrowed book, student information, borrow date, return date, and status.
* **Reservation:** Represents the reservation activity of a student. It includes details like the reserved book, student information, reservation date, and status.
* **Purchase:** Represents the purchase activity of a student. It includes details like the purchased book, student information, and purchase date.
* **Payment:** Tracks payments made by students for fines or purchases. It includes student details, payment amount, and payment date.
* **Penalty:** Represents fines imposed on students by the admin. It includes details such as penalty amount, reason, date, and status.

**2.4 Operating Environment**

The University Library Management System will be a web-based application that can be accessed through modern web browsers such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge. The system will be hosted on a web server that supports the required technologies.

**2.5 Design and Implementation Constraints**

The Movie Recommendation System will be implemented using the following technologies:   
- **Frontend**: HTML, CSS, JavaScript - **Backend**: Php Laravel - **Database**: MySQL

**2.6 User Documentation**

**For Students:**

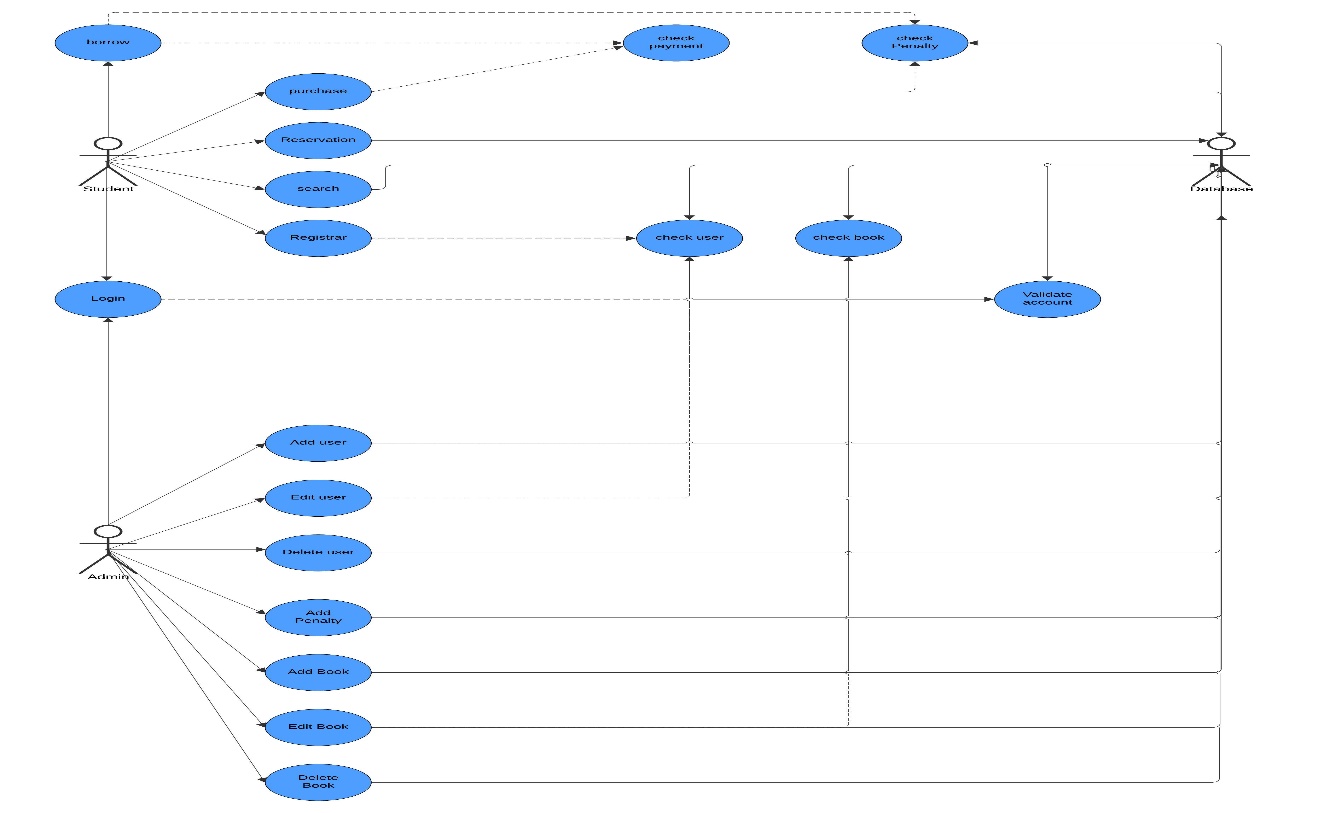
* **Creating an Account:** Instructions on how to register for a library account.
* **Searching for Books:** Steps to find books using the search functionality.
* **Borrowing Books:** How to borrow books from the library.
* **Purchasing Books:** Guidelines for purchasing books through the system.
* **Reserving Books:** Instructions for reserving unavailable books for future use.

**For Administrators:**

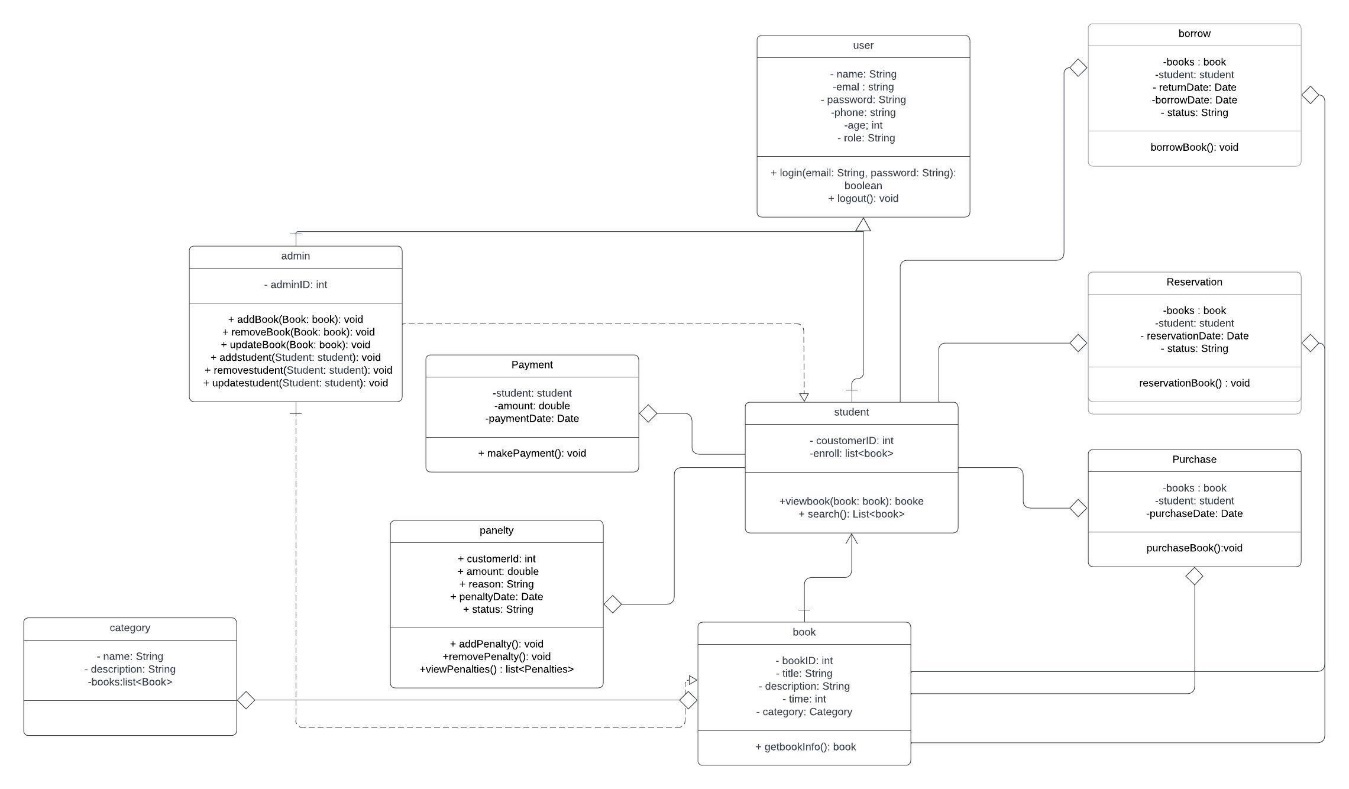
* **Managing Users**: Instructions on creating, updating, and deleting student accounts.
* **Managing Books:** Steps to add, update, or remove books from the library catalog.
* **Imposing Penalties:** Guidelines for applying penalties for overdue or lost books.
* **System Maintenance:** Ensuring the library system operates smoothly.

**2.7 Assumptions and Dependencies**

The University Library Management System assumes that users have a basic understanding of how to use a web application and are familiar with concepts such as creating an account, purchase in books, and managing Books

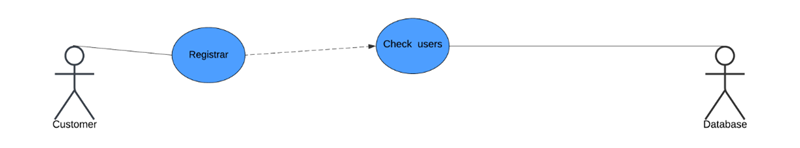
**3. System Features**

*Figure 2: Overall System Use Case Diagram.*

****

*Figure 3: Overall System UML Class Diagram*

**3.1 User Registration & Check**



**Description & Priority**

Users can create an account on the platform by providing their username and password. This feature is of **High priority** as it is required for users to access the system.

Use Case #: CMS-Register

Use Case name: User Registration

Purpose: To allow new users to register for an account in the system.

Primary actor: Customer

Secondary actor: None

Input parameters: username , email , password , more information.

Output parameters: None

Precondition:

• The username and email must be unique and not already in use.

• The password must meet security requirements (e.g., minimum length, complexity).

• Implement all parameters

Post-condition:

• A new user account is created, and the user can log in to the system.

Successful scenario:

1. The user submits registration details (username, email, password , more info).

2. The parameters (username , email ) are passed to the check user functionality to find if there is someone use the email and username before.

3. The result of check user, a failure, is passed back to the user

4. The system validates the input for uniqueness and security requirements.

5. The user account is created, and a confirmation message is returned.

Exceptions:

• Username or email is already in use.

• Password does not meet security requirements.

• Invalid email format.

Additional remarks:

• Passwords should be securely hashed before storage.

Use Case #: CMS-Check

Use Case name: Check User

Purpose: To get user from the database

Primary actor: None

Secondary actor: Database

Input parameters: Email , password

Output parameters: User

Precondition:

• Both parameters are provided.

• Both parameters are in proper format.

Post-condition:

• If successful, the user account is retrieved from the database

Successful scenario:

1. An external process invokes this use case by passing the Email and Password.

2. The system ensures that both parameters are in proper format.

3. The user account is retrieved from the database and returned to the calling process.

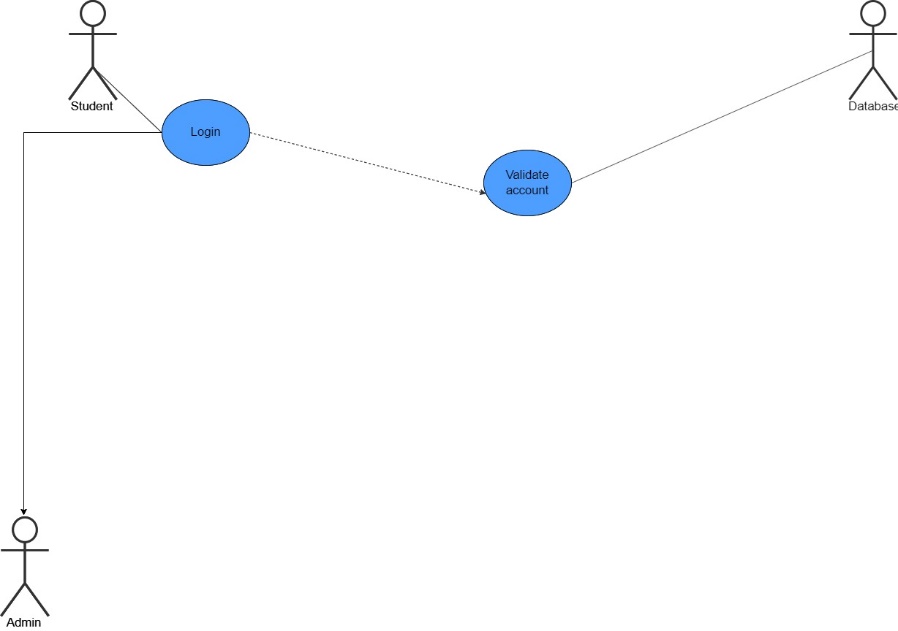
Exceptions:

• One or both parameters are missing.

• One or both parameters are in incorrect format

Additional remarks: None

**3.2 User Login &** **Validate account**



**Description & Priority**

Users can log in to the platform using their username and password. This feature is of **High priority** as it is required for users to access their account.

**Use Case #:** CMS-L1   
**Use Case name:** Login   
**Purpose:** To login into an account.   
**Primary actor:** Admin, Student   
**Secondary actor:** None   
**Input parameters:** Email, Password   
**Output parameters:** None   
**Precondition:**

* Input parameters must be in the appropriate format.
* Both input parameters must be provided.

**Post-condition:**

* If successful, a confirmation is sent to the user for successful login.

**Successful scenario:**

1. User submits the two parameters: email and password.
2. The two parameters are validated for format and completeness.
3. The two parameters are passed to the validate\_account functionality to validate user access.
4. The result of validate\_account, a confirmation, is passed back to the user.
5. Upon success, the user is navigated to the appropriate dashboard.

**Exceptions:**

* One or both parameters are missing.
* One or both parameters are in the incorrect format.

**Additional remarks:**

* A Dashboard page may be displayed after a successful login.
* The welcome page may vary based on the type of user.

**Use Case #:** CMS-V   
**Use Case name:** Validate Account   
**Purpose:** To validate an account against the stored accounts in the database during login.   
**Primary actor:** None   
**Secondary actor:** Database   
**Input parameters:** email , password

**Output parameters:** account

**Precondition:**

* Both parameters are provided.
* Both parameters are in proper format.

**Post-condition:**

* If successful, the account is retrieved from the database

**Successful scenario:**

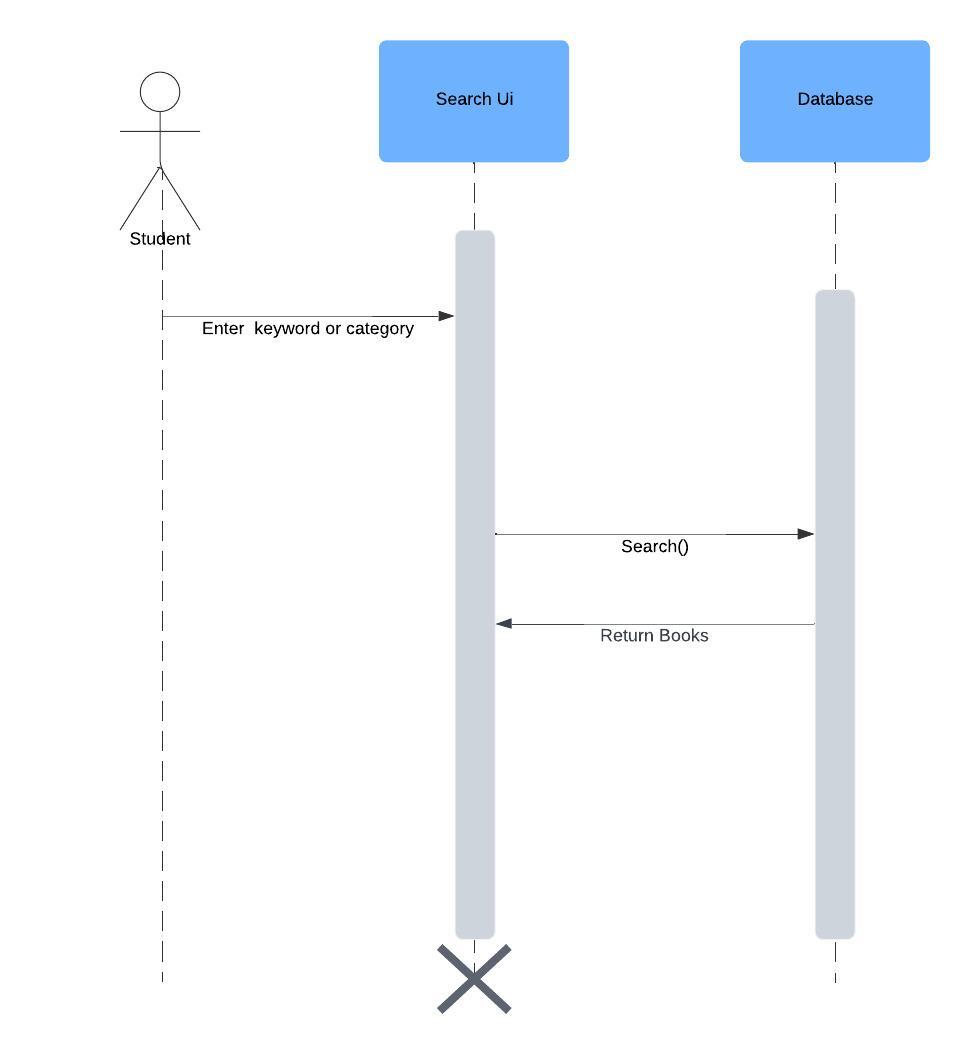
1. One of the other use cases invokes this use case by passing the email and Password.
2. The system ensures that both parameters are in proper format.
3. The account is retrieved from the database and is returned to the calling use case.

**Exceptions:**

* One or both parameters are missing.
* One or both parameters are in incorrect format.
* Account does not exist for the given email and password.

**Additional remarks:** None

**3.3 Search**



**Description & Priority**

Users will be able to search for Books using various parameters such as catalog, author, or keywords, **High Priority**: The search feature is essential for enhancing user experience and accessibility, allowing users to quickly find Books that meet their needs.

**Use Case #:** CMS-Search   
**Use Case name:** Search Book   
**Purpose:** To allow users to search for available Books.   
**Primary actor:** Customer  
**Secondary actor:** Database   
**Input parameters:** Keyword , Category , author,title

**Output parameters:** List of Books

**Precondition:**

* At least one search parameter (keyword, category, author) should be provided.

**Post-condition:**

* If successful, at least one search parameter (keyword, category, or author) should be provided.

**Successful scenario:**

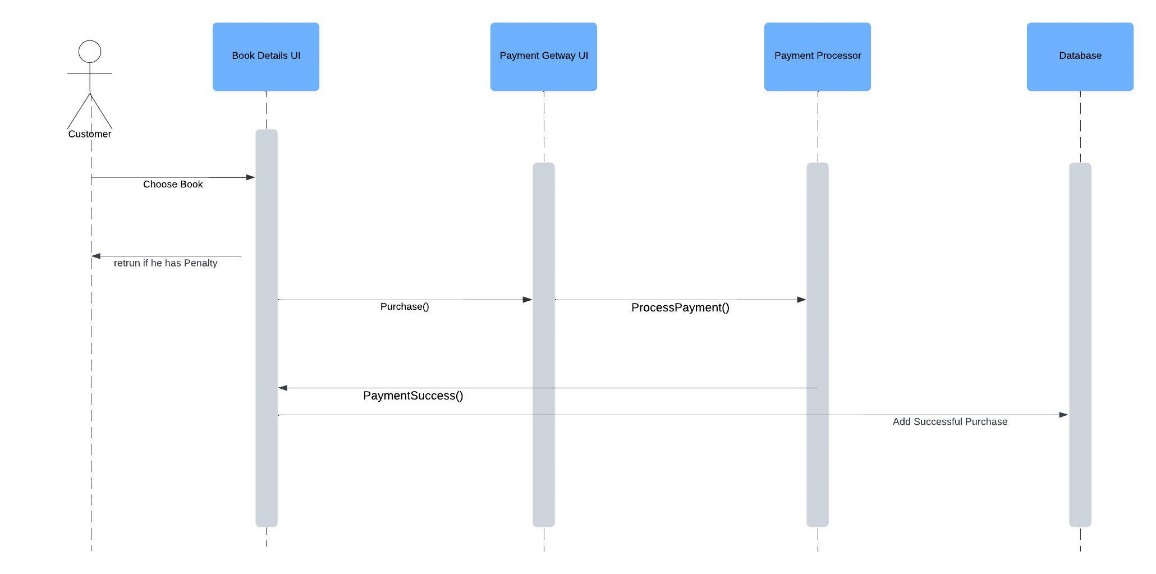
1. User submits the search parameters (keyword, category, author).
2. System validates that at least one search parameter is provided.
3. System searches for books based on the given parameters.
4. System returns a list of matching books to the user.

**Exceptions:**

* No parameters provided for search.
* No books found matching the search criteria.

**Additional remarks:** None

**3.4 Pay for book**



**Description & Priority**

Users will be able to pay for the book and proceed to a secure payment gateway to complete their purchase, provided the book is available, **High Priority**: The payment feature is critical for the functionality of the Library Management System

**Use Case #:** CMS-pay   
**Use Case name:** Purchase   
**Purpose:** To allow users to make a payment for a book.   
**Primary actor:** Customer   
**Secondary actor:** Database   
**Input parameters:** user\_id , book\_id

**Output parameters:** None

**Precondition:**

* User must be logged in.
* book ID and payment details must be valid.

**Post-condition:**

* If payment is successful, a is generated, and access to the book is granted to the user.

**Successful scenario:**

1. The user selects a book to pay for and submits payment method.
2. The system verifies the book ID and payment  method.
3. The system processes the payment through the payment gateway.
4. Upon successful payment, the system generates a receipt and grants the user access to the book.

**Exceptions:**

* Invalid book ID or user ID.

**Additional remarks:**

* The payment gateway can be integrated with external APIs for real transactions. The function could be extended with notifications or access control for different book types.

**3.5 Edit User**

**A close-up of a computer screen

Description automatically generated**

**Description & Priority**

The Library Management System will enable admins to manually edit user details (such as username, email, password, and role).  
**High Priority:** This functionality is essential for maintaining up-to-date user information and supporting administrative control over the platform.

**Use Case #: CMS-EditU**  
**Use Case Name:** Edit User  
**Purpose:** To allow an admin to update the details of an existing user in the system.  
**Primary Actor:** Admin  
**Secondary Actor:** Database

**Input Parameters: user\_id**

**Output Parameters:**

* Confirmation message indicating successful user update or an error message.

**Precondition:**

* The existing username must be provided.
* At least one parameter to be updated must be provided.
* If the email is being updated, it must be in a valid format.
* If the username is being updated, it must be unique and not already in use.

**Post-condition:**

* If successful, the user's details are updated and stored in the database.

**Successful Scenario:**

1. The admin submits the existing username and the new details to be updated (e.g., email, password, role).
2. The system verifies the existing username and checks for potential conflicts (e.g., duplicate username or email).
3. If verification passes, the system updates the user details in the database.
4. A confirmation message is returned to the admin.

**Exceptions:**

* The existing username is not found.
* The email is not in a valid format.
* The new username is already taken.
* No parameters are provided for updating.

**Additional Remarks:** None

**3.6 Add Penalty**

**A close-up of a computer screen

Description automatically generated**

**Description & Priority**

The Library Management System will allow admins to add penalties to users for overdue books, lost items, or rule violations.  
**High Priority:** This functionality is essential for enforcing library policies and ensuring compliance with borrowing rules.

**Use Case #: CMS-AddPenalty**  
**Use Case Name:** Add Penalty  
**Purpose:** To enable an admin to impose penalties on users for rule violations or overdue books.  
**Primary Actor:** Admin  
**Secondary Actor:** Database

**Input Parameters:user\_id, Penalty**

**Output Parameters:**

* Confirmation message indicating successful penalty addition or an error message.

**Precondition:**

* The user\_id must exist in the system.
* The penalty must be provided.

**Post-condition:**

* If successful, the penalty is recorded in the database and associated with the specified user account.

**Successful Scenario:**

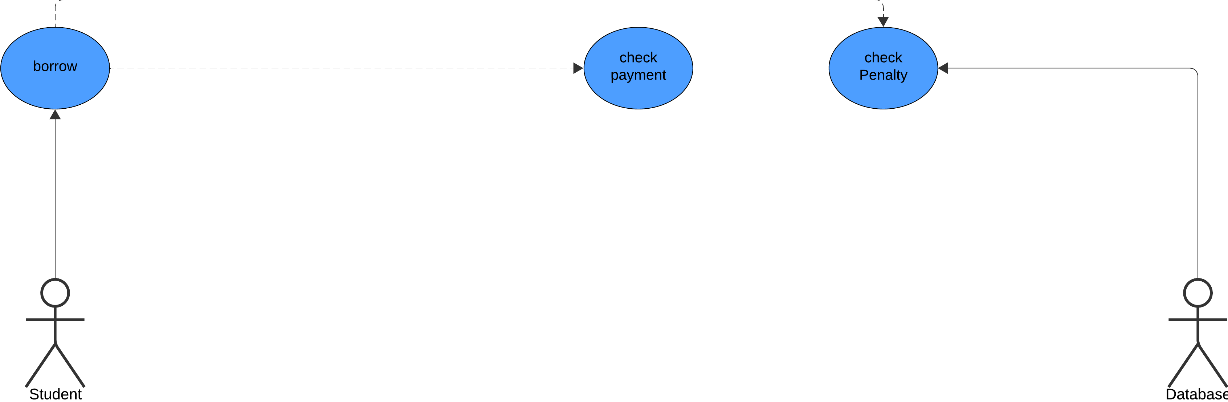
1. The admin submits the user\_id , Penalty.
2. The system verifies the user exists in the database.
3. The penalty details are recorded and associated with the user account.
4. A confirmation message is returned to the admin.

**Exceptions:**

* The user\_id does not exist in the system.
* One or more required parameters are missing.

**Additional Remarks:** None

**3.7 borrow**

****

**Description & Priority**

The Library Management System will allow students to borrow books, provided they have no pending penalties.  
**High Priority:** This functionality is critical for ensuring efficient book lending while maintaining compliance with library policies.

**Use Case #: CMS-BorrowBook**  
**Use Case Name:** Borrow Book  
**Purpose:** To allow a student to borrow a book after verifying eligibility and availability.  
**Primary Actor:** Student  
**Secondary Actor:** Database

**Input Parameters:**user\_id , book\_id , borrow

**Output Parameters:**

* Confirmation message indicating successful borrowing or an error message.

**Precondition:**

* The user must exist in the system.
* The book must be available in the library database.
* The user must have no pending penalties.

**Post-condition:**

* If successful, the book is marked as borrowed, the borrowing record is added to the database, and the due date is calculated.

**Successful Scenario:**

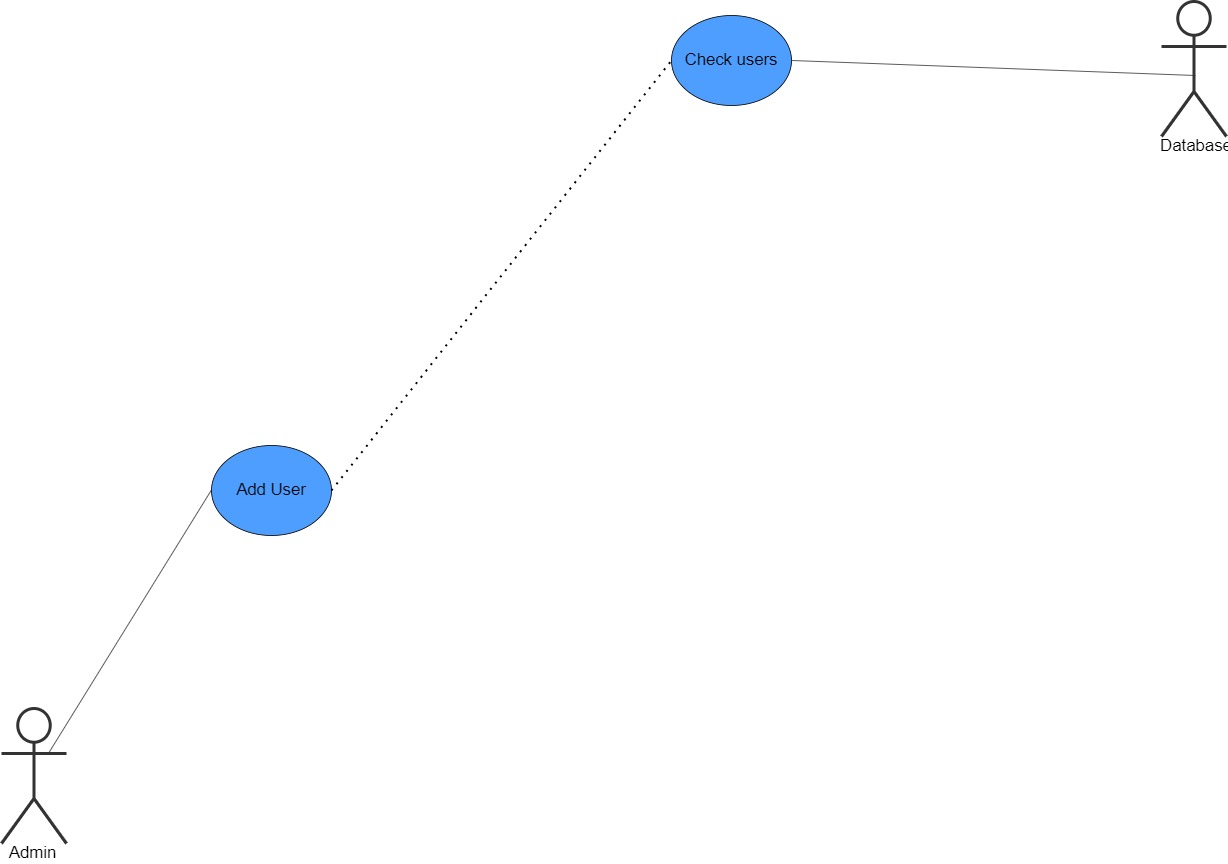
1. The student submits their user ID and the Book ID.
2. The system calls the **Check Penalty** function to verify if the student has any pending penalties.
   * If a penalty exists, an error message is returned, and the process ends.
3. The system calls the **Check Book** function to verify the availability of the book.
   * If the book is unavailable, an error message is returned, and the process ends.
4. If the penalty check and book availability check pass, the system processes the borrowing request.
5. The system integrates with the **Payment** function (if applicable) to finalize the borrowing transaction.
6. The borrowing record, including the username, Book ID, borrow date, and due date, is added to the database.
7. A confirmation message is returned to the student.

**Exceptions:**

* The user does not exist in the system.
* The student has pending penalties.
* The book is unavailable.
* Payment (if required) fails or is incomplete.

**Additional Remarks:**  
This use case ensures proper management of book borrowing while maintaining compliance with library rules and payment processes.

**3.8** **Add User, check users**



**Description & Priority**

The Library Management System will enable admins to manually add new users (such as Admin, customers) and check existing user profiles, **High Priority**: This functionality is essential for managing users and supporting administrative control over the platform.

**Use Case #:** CMS-AddU

**Use Case name:** Add User

**Purpose:** To allow an admin to add a new user to the system.

**Primary actor:** Admin

**Secondary actor:** Database

**Input parameters:** username , email ,password , role

**Output parameters:**

* Confirmation message indicating successful user addition or an error message.

**Precondition:**

* All input parameters must be provided.
* The email must be in the proper format.
* The username must be unique and not already in use.

**Post-condition:**

* If successful, a new user account is created and stored in the database.

**Successful scenario:**

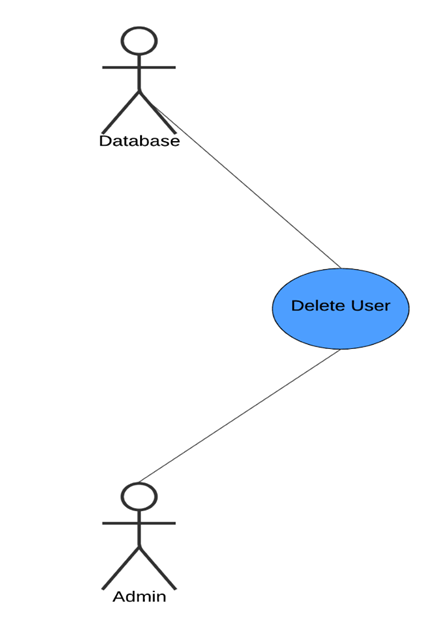
1. The admin submits user details (username, email, password, role).
2. The parameters (username , email ) are passed to the check user functionality to find if there is someone use the email and username before.
3. The result of check user, a failure, is passed back to the user
4. A new user is created and added to the database.
5. A confirmation message is returned to the admin.

**Exceptions:**

* One or more parameters are missing.
* The email is not in a valid format.
* The username is already taken.

**Additional remarks: None**

**3.9 Delete User**



**Description & Priority**

The Library Management System will provide admins with the ability to delete user accounts, **Medium Priority**: This feature is important for maintaining user management.

**Use Case #:** CMS-D1

**Use Case name:** Delete User

**Purpose:** To allow an admin to remove a user from the system.

**Primary actor:** Admin

**Secondary actor:** Database

**Input parameters:** username

**Output parameters:**

* Confirmation message indicating successful user deletion or an error message.

**Precondition:**

* The input parameter must be provided.
* The user specified by the username must exist in the database.

**Post-condition:**

* If successful, the user account is removed from the database.

**Successful scenario:**

1. The admin submits the username of the user to be deleted.
2. The system checks if the user exists in the database.
3. The user is removed from the database.
4. A confirmation message is returned to the admin.

**Exceptions:**

* The username parameter is missing.
* The user does not exist in the database.

**Additional remarks:** None

**A long thin line on a white background

Description automatically generated3.10 reservation**

**Description & Priority**

The Library Management System will allow students to reserve unavailable books.   
**High Priority:** This functionality ensures fair access to books and maintains library rule compliance through penalties and payments.

**Use Case #: CMS-ReserveBook**  
**Use Case Name:** Reserve Book  
**Purpose:** To allow a student to reserve an unavailable book after verifying eligibility and completing the reservation fee payment.  
**Primary Actor:** Student  
**Secondary Actor:** Database

**Input Parameters:user\_id , Book\_id, reservation**

**Output Parameters:**

* Confirmation message indicating successful reservation or an error message.

**Precondition:**

* The user must exist in the system.
* The book must be unavailable for borrowing but eligible for reservation.
* The user must have no pending penalties.

**Post-condition:**

* If successful, the book is marked as reserved, the reservation record is added to the database, and a confirmation is sent to the student.

**Successful Scenario:**

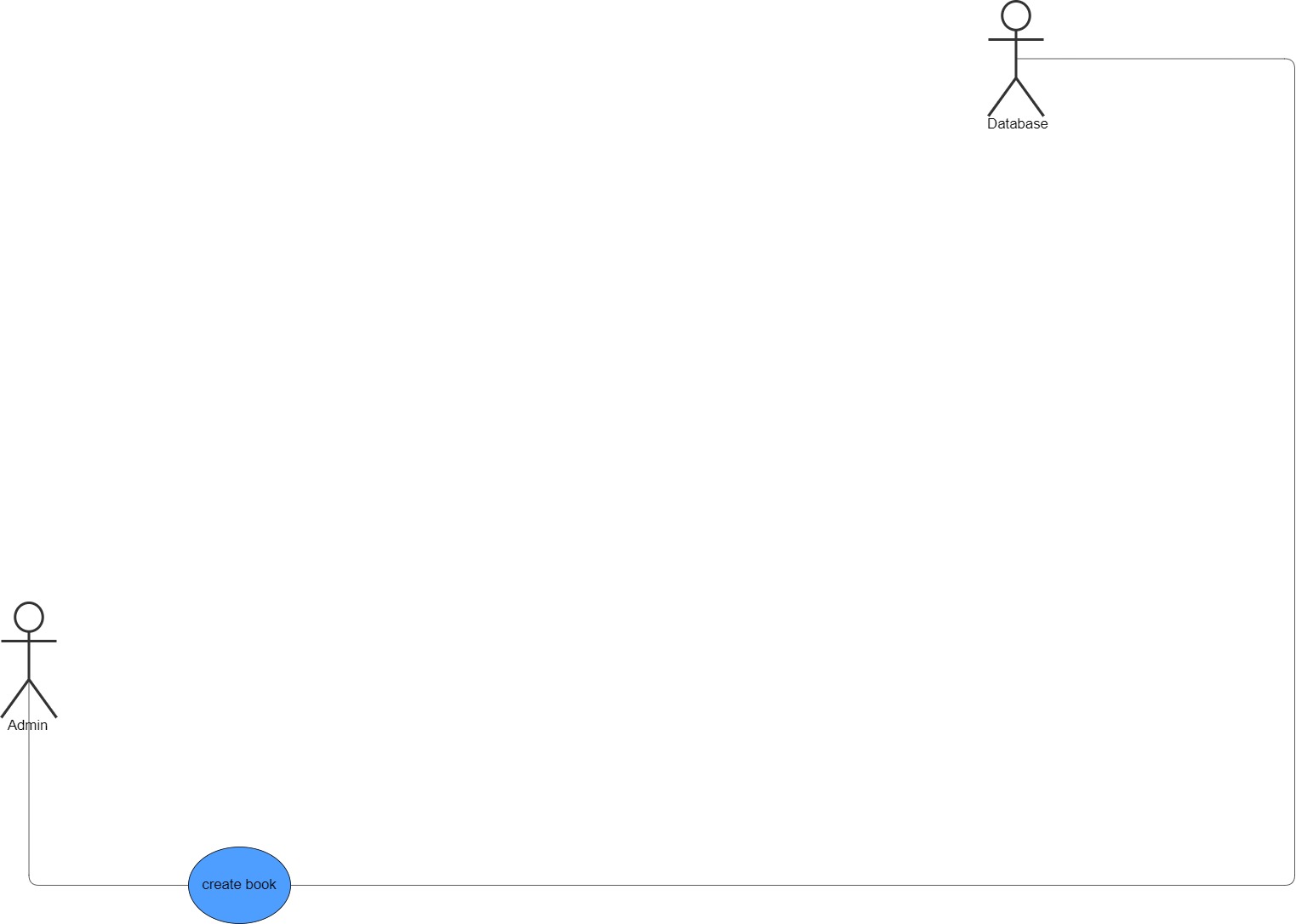
1. The student submits their username and the Book ID for reservation.
2. The system calls the **Check Penalty** function to verify if the student has any pending penalties.
   * If a penalty exists, an error message is returned, and the process ends.
3. The reservation record, including the username, Book ID , reservation , is added to the database.
4. A confirmation message is returned to the student.

**Exceptions:**

* The user does not exist in the system.
* The student has pending penalties.
* The book is not eligible for reservation.

**Additional Remarks:**  
This use case ensures a fair reservation system while incorporating penalty checks and payment processing to maintain order and accountability.

**3.11 Create book**



**Description & Priority**

The Library Management System will provide authorized users, such as admins, the ability to add new books to the platform, **High Priority**: The add book feature is a core function.

**Use Case #:** CMS-AddB

**Use Case name:** Add Book

**Purpose:** To allow an admin to add a new book to the system.

**Primary actor:** Admin

**Secondary actor:** Admin

**Input parameters:** book name , title of book , prices, content

**Output parameters:**

* Confirmation message successful send to admin or an error message.

**Precondition:**

* All input parameters must be provided.
* The prices Currency type .

**Post-condition:**

* If successful, a new Book is stored in the admin.

**Successful Scenario:**

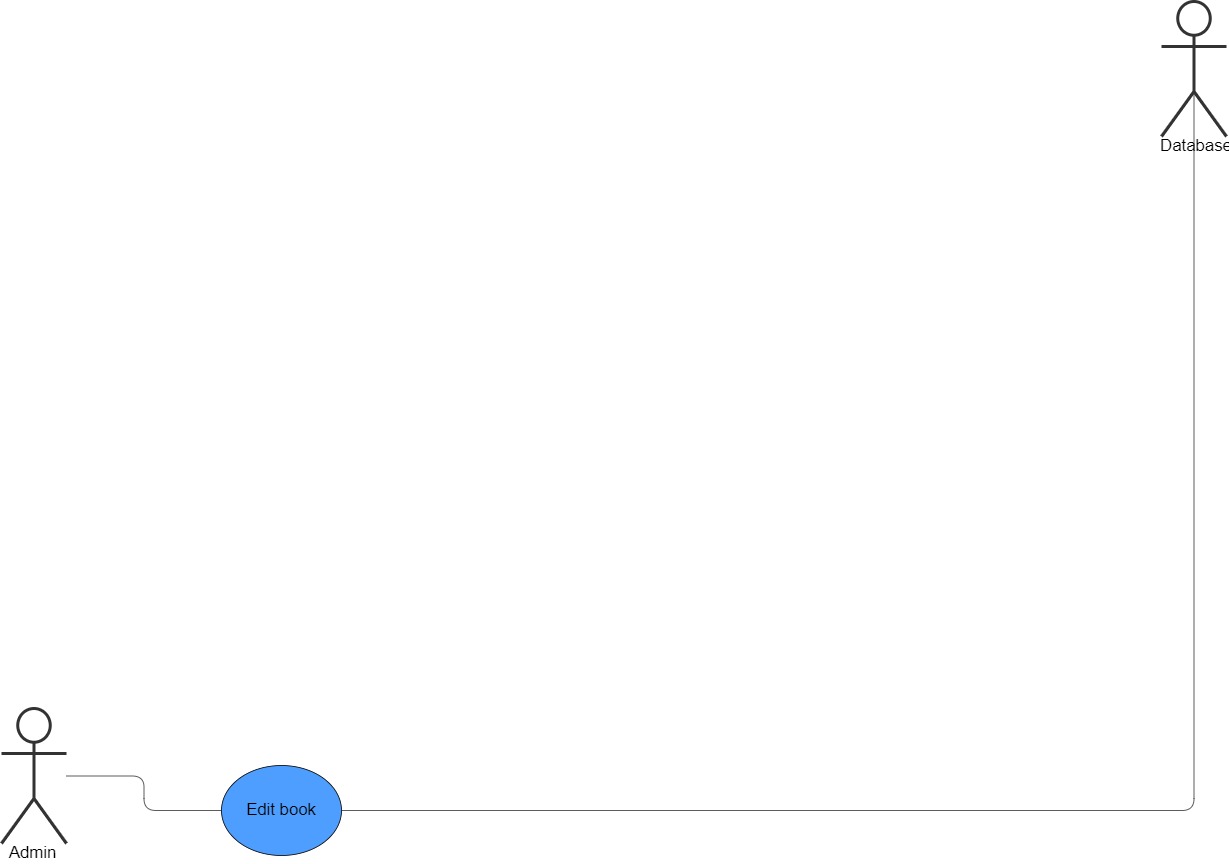
1. The Admin submits Book details.
2. The system verifies that all input parameters are provided and valid.
3. The system verifies the prices' currency type.
4. If all validations pass, the new book is added to the database.

**Exceptions:**

* One or more parameters are missing.
* The prices are not in the correct currency type.

**Additional Remarks:** None

**3.12 Edit book**



**Description & Priority**

The Library Management System will allow authorized users, such as admins to edit Book details, **High Priority**: As book may need updates over time for accuracy, relevance, and quality.

### Use Case #: CMS-EditB

**Use Case Name:** Edit Book

**Purpose:**  
To allow a Admin to edit an existing book in the system.

**Primary Actor:** Admin

**Secondary Actor:** Database

**Input Parameters: book\_id**

**Output Parameters:**

* Confirmation message indicating successful book edit or an error message.

**Preconditions:**

* The book ID must be provided.
* The book ID must exist in the system.
* If provided, the updated prices must be in the correct currency type.

**Post-condition:**

* If successful, the book information is updated in the system.

**Successful Scenario:**

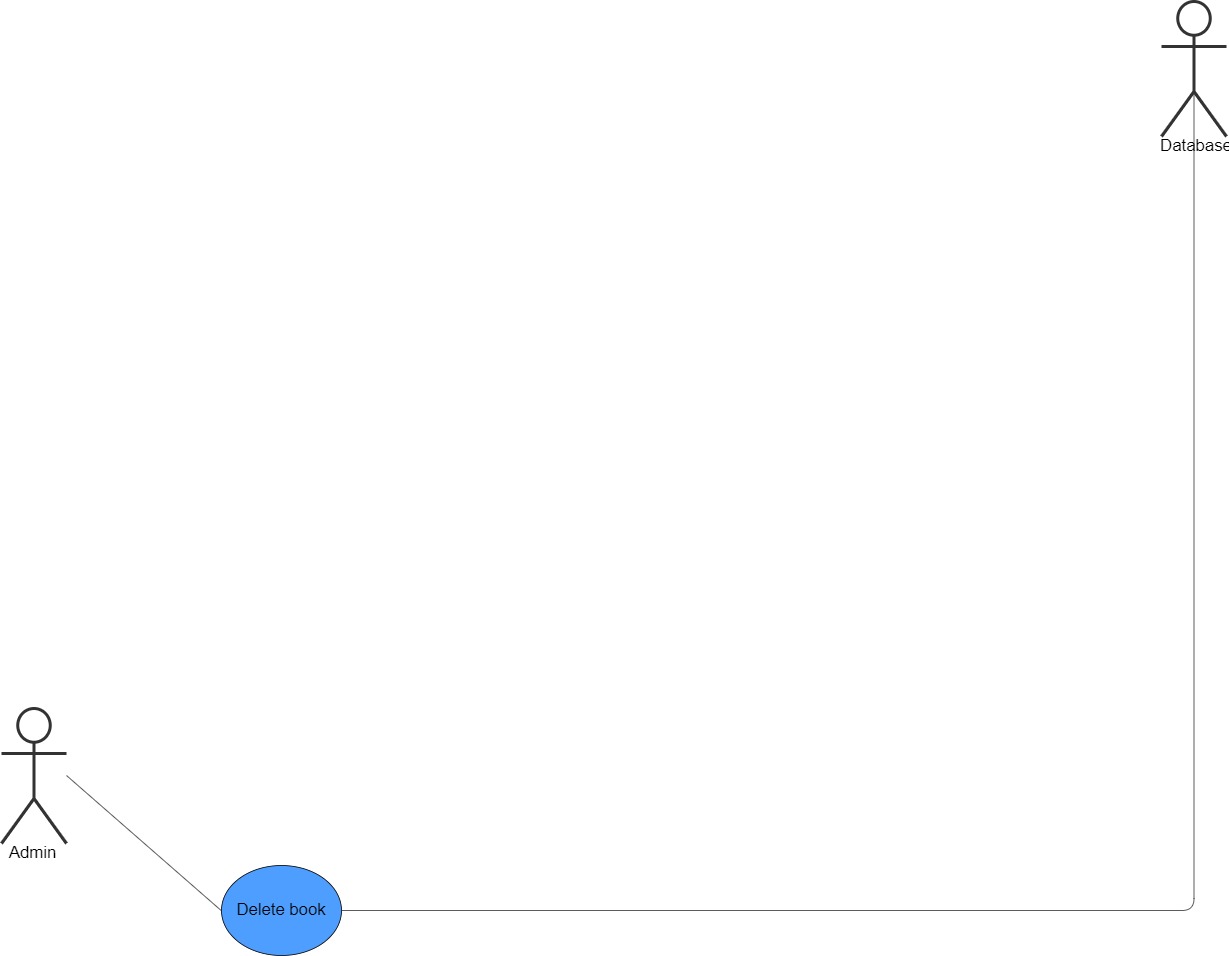
1. The system verifies that the book ID exists.
2. The system verifies the updated prices' currency type (if provided).
3. If all validations pass, the book information is updated in the database.
4. A confirmation message is sent to the admin.

**Exceptions:**

* The book ID does not exist.
* The updated prices are not in the correct currency type.

**Additional Remarks:**  
None

**3.13 Delete book**



**Description & Priority**

The Library Management System will include a feature that allows admins to delete a book from the platform**, Medium Priority**: While not critical for the initial system operation

### Use Case #: CMS-DeleteB

**Use Case Name:** Delete book

**Purpose:**  
To allow a Admin to delete an existing book from the system.

**Primary Actor:**  
Admin

**Secondary Actor:**  
Database

**Input Parameters:**

* Book ID

**Output Parameters:**

* Confirmation message indicating successful book deletion or an error message.

**Preconditions:**

* The book ID must be provided.
* The book ID must exist in the system.

**Post-condition:**

* If successful, the book is removed from the system.

**Successful Scenario:**

1. The Admin submits the book ID for the book they wish to delete.
2. The system verifies that the book ID exists.
3. If the book ID is valid, the book is deleted from the database.

**Exceptions:**

* The book ID does not exist.

**Additional Remarks:**  
None

**4. External Interface Requirements**

**4.1 User Interfaces**

The University Library Management System will include the following user interfaces:

* **Registration Page:** Allows students and administrators to create accounts by providing their personal details.
* **Login Page:** Enables users to log in using their username and password.
* **Home Page:** See Most borrowed books, Most reserved books, and Most purchase books.
* **Browse Page:** Allows students to search and browse books by title, author, or category, with results pulled from the database.
* **Book Details Page:** Displays detailed information about a selected book, including its availability status. Provides options to borrow, purchase, or reserve the book.
* **User Management Page (Admin Only):** Allows administrators to view, add, delete, and update user accounts, as well as validate student accounts.
* **Book Management Page (Admin Only):** Enables administrators to add, edit, or delete books in the library catalog.
* **Penalty Management Section:** Provides administrators with tools to manage and apply penalties for overdue or lost books.

**4.2 Hardware Interfaces**

The University Library Management System will be a web-based application that can be accessed through a modern web browser. It does not have any specific hardware requirements.

**4.3 Software Interfaces**

The University Library Management System will interact with the following software components:

* **Database**:
  + Stores user information, book data, borrow records, and purchase records.
  + Supports user authentication, book search and reservation records.
* **Authentication Service**:
  + Manages user login, registration, and session management.
  + Ensures secure access to role-based functionalities (e.g., Student, Admin).
* **Payment Gateway API**:
  + Manages payment processing for books.
  + Receives ProcessPayment() requests and returns PaymentSuccess() or PaymentFailure() responses to confirm or deny purchase.
* **Search Service**:
  + Supports book search functionality by handling Search() requests and returning relevant books based on keyword or category.
  + Pulls book data from the database and filters based on user-provided search criteria.

**4.4 Communication Interfaces**

The books Management System will communicate with users through a web interface. The system will use HTTP requests and responses to interact with the frontend and backend components.

**5. Other Nonfunctional Requirements**

**5.1 Safety Requirements**

The University Library Management System shall not store sensitive user information in plain text. The system shall use secure hashing algorithms to store user passwords.

**5.2 Security Requirements**

The University Library Management System shall encrypt user passwords using a secure hashing algorithm. The system shall use HTTPS to secure communication between the frontend and backend components.

**5.3 Software Quality Attributes**

The University Library Management System shall be user-friendly and intuitive to use. The system shall be responsive and provide real-time updates to users.