**Library Management System**

**TEAM**

Tauseef Indikar

Nicolas Neitzel

Gaurav Shukla

Sanketh Shetty

**Library Management System**

Table of Contents

[1 Project Summary 3](#_Toc433065592)

[2 Project Requirements 3](#_Toc433065593)

[2.1 Business Requirements 3](#_Toc433065594)

[2.2 User Requirements 3](#_Toc433065595)

[2.3 Functional Requirements 3](#_Toc433065596)

[2.4 Non-Functional Requirements 4](#_Toc433065597)

[3 Users and Tasks (Use Cases) 4](#_Toc433065598)

[4 Activity Diagram 7](#_Toc433065599)

[5 Data Storage 8](#_Toc433065600)

[5.1 User Credentials Table 8](#_Toc433065601)

[5.2 Inventory Table 8](#_Toc433065602)

[5.3 Request Table 8](#_Toc433065603)

[5.4 Checkout Table 8](#_Toc433065604)

[6 UI Mockups 9](#_Toc433065605)

[6.1 Login UI 9](#_Toc433065606)

[6.2 Student UI 9](#_Toc433065607)

[6.3 Librarian UI 11](#_Toc433065608)

[6.4 Admin UI 12](#_Toc433065609)

[7 User Interactions 13](#_Toc433065610)

[7.1 Sequence Diagram: Admin adding/updating a Librarian 13](#_Toc433065611)

[7.2 Sequence Diagram: Admin adding/updating a book 14](#_Toc433065612)

[7.3 Sequence Diagram: Librarian Checking out a book 14](#_Toc433065613)

[8 Class Diagrams 15](#_Toc433065614)

# Project Summary

We will be building a website which serves as a Library Management System. The UI will be designed using HTML, JavaScript and CSS. The Server-side-scripting will be done using an MVC framework based on C#. MS SQL Server will be used to store the database.

There will be three types of users i.e. Administrator, Librarian and Student. An Administrator will have access to add/modify the books in the database as well as add/modify Librarian accounts. Students will be able to search the database for books based on three criteria ISBN, Title or Author and request a book. Librarians will be able to search the database for books and checkout requested books for the Student after the system verifies availability of the book.

We aim to showcase the MVC design model as well object oriented concepts which are useful for reuse and extension of the Library Management System.

# Project Requirements

## Business Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***ID*** | ***Requirement*** | ***Topic Area*** | ***User*** | ***Priority*** |
| *BR-01* | All signups should be through students ‘@colorado.edu’ email-id | \*Signup  \*Authentication | Student | Critical |
| *BR-02* | Only Administrator can add/modify a Librarians account | \*Librarian Signup | Administrator | Medium |
| *BR-03* | Only Administrator can add/modify books in the database | \*Books Database | Administrator | Medium |
| *BR-04* | Students can at most checkout 3 books at any given time | \*Request Book | Student | High |
| *BR-05* | Due date to be set for 2 months from Checkout Date | \*Checkout | System | Critical |

## User Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ***ID*** | ***User*** | ***Description*** | ***Priority*** |
| *US-01* | Administrator | I want to be able to add, delete and update books in the database. | Medium |
| *US-02* | Administrator | I need to be able to add, delete and update the Librarian profiles. | Medium |
| *US-03* | Librarian | I need to see a dashboard with all requests that are current. | High |
| *US-04* | Librarian | I want to be able to easily search and locate a book. | Low |
| *US-05* | Student | I need to be able to search for a book based on multiple criteria. | Medium |
| *US-06* | Student | I need to be able to request a book for today or a day in the future. | High |

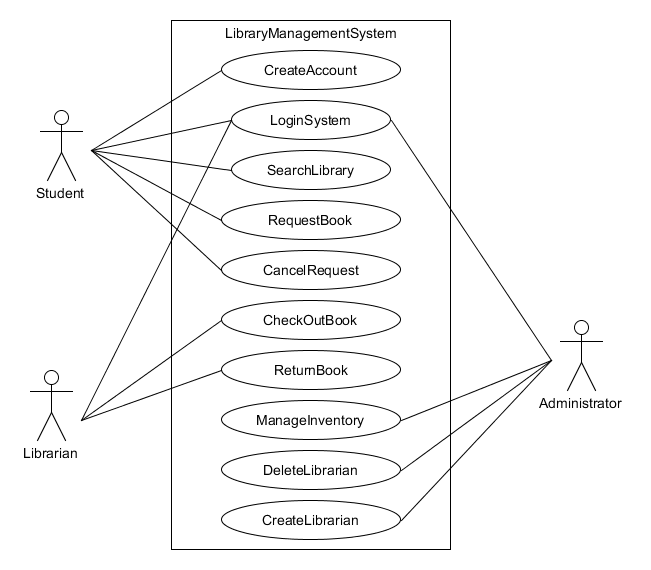
## Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ***ID*** | ***User*** | ***Description*** | ***Priority*** |
| *FR-01* | Librarian | Landing page should show all current requests. | High |
| *FR-02* | Librarian | After checkout take back to landing page with success message. | Medium |
| *FR-03* | Administrator | While creating librarian account mail librarian with a random temp password. | Medium |
| *FR-02* | Student | Landing page should show selection of books w.r.t topic. | High |
| *FR-03* | Student | Requests for a book should populate request date with today’s date. | High |

## Non-Functional Requirements

|  |  |  |
| --- | --- | --- |
| ***ID*** | ***Description*** | ***Priority*** |
| *NFR-01* | *Security*: All passwords must be hashed before saving in database. | Critical |
| *NFR-02* | *Platform Constraints*: All basic functionality for librarian and user must be supported across browsers (Firefox, Chrome, Opera, IE and Safari). | High |
| *NFR-03* | *Usability*: The search feature should work even without specifying criteria by searching across all parameters. | Nice-to-have |

# Users and Tasks (Use Cases)



|  |  |
| --- | --- |
| ***Use case name*** | **CreateAccount** |
| *Participating Actors* | Initiated by **Student** |
| Flow of Events | 1. Student selects sign up tab |
|  | 2. **Student** inputs information to create account |
| *Entry Condition* | N/A |
| *Quality Requirements* | System verifies **Student** has @colorado.edu email |

|  |  |
| --- | --- |
| ***Use case name*** | **CreateBook** |
| *Participating Actors* | Initiated by **Administrator** |
| *Flow of Events* | 1. **Administrator** selects option to add book to library |
|  | 2. **Administrator** enters book information onto screen |
|  | 3. **Administrator** submits information |
| *Entry Condition* | **Administrator** is logged intoLibrary Management System |
| *Quality Requirements* | **Administrator** is informed of successful addition |

|  |  |
| --- | --- |
| ***Use case Name*** | **CreateLibrarian** |
| *Participating actors* | Initiated by **Administrator** |
| *Flow of Events* | 1. **Administrator** pulls up Create Librarian screen |
|  | 2. **Administrator** enters email id and password for new user |
| *Entry Conditions* | **Administrator** is logged into Library Management System |
| *Exit Condition* | **Librarian** account created succesfully |
| *Quality Requirements* | **Administrator** receives confirmation of created account |

|  |  |
| --- | --- |
| ***Use Case name*** | **RequestBook** |
| Paticipating actors | Initiated by **Student** |
| *Flow of Events* | 1. **Student** logs into the System |
|  | 2. **Student** searches for book by category |
|  | 2a. Search by Title |
|  | 2b. Search by Author |
|  | 2c. Search by ISBN |
|  | 3.**Student** request a book from the search results |
| *Entry Conditions* | **Student** is logged into **LibraryManagementSystem** |
| *Quality Conditions* | **Student** is informed of Request status |

|  |  |
| --- | --- |
| ***Use case name*** | **ReturnBook** |
| *Participating Actors* | Initiated by **Librarian** |
| *Flow of Events* | 1. **Librarian** receives returned book |
|  | 2. **Librarian** selects return book option |
|  | 3. **Librarian** enters ISBN and student ID |
|  | 4. **Librarian** receives confirmation book was returned succesfully |
| *Entry Condition* | **Librarian** is logged into **Library Management System** |
| *Exit Condition* | **Librarian** receives confirmation of return |
| *Quality Conditions* | Success message is displayed for proper return |

|  |  |
| --- | --- |
| ***Use Case Name*** | **DeleteLibrarian** |
| *Participating Actors* | Initiated by **Administrator** |
| *Flow of Events* | 1. **Administrator** selects delete librarian tab |
|  | 2. **Administrator** selects librarian to delete |
|  | 3. **Administrator** submits request |
| *Entry Condition* | **Administrator** is logged into **LibraryManagementSystem** |
| *Quality Requirements* | **Administrator** receives confirmation of operation |

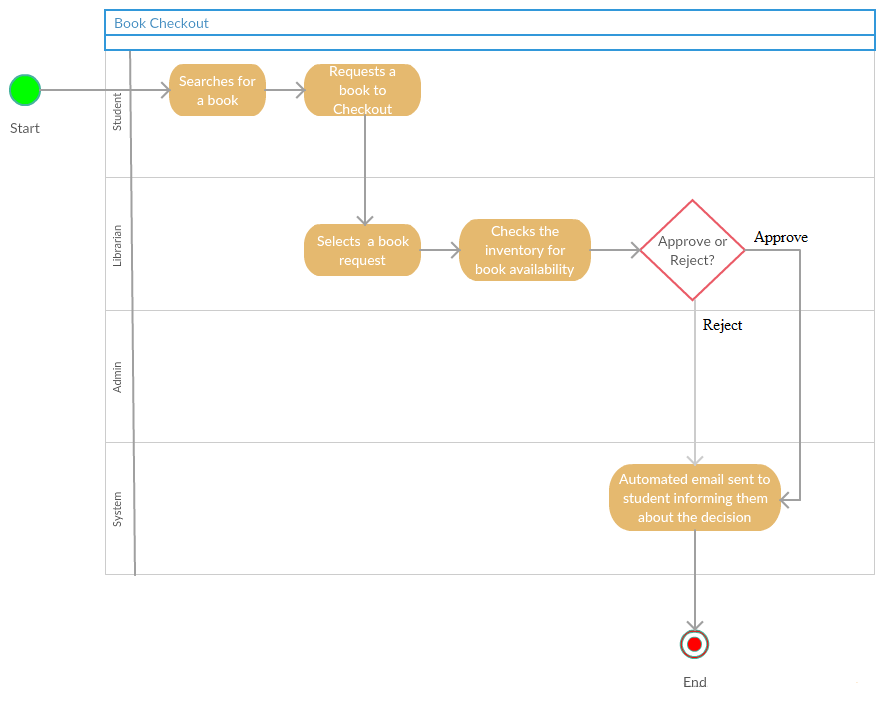
|  |  |
| --- | --- |
| ***Use case name*** | **CancelRequest** |
| *Participating Actors* | Initiated by **Student** |
| *Flow of Events* | 1. **Student** selects cancel request tab |
|  | 2. **Student** selects book request to be cancelled |
|  | 3. **Student** submits request |
| *Entry Conditions* | **Student** is logged into **LibraryManagement System** |

|  |  |
| --- | --- |
| ***Use case name*** | **SearchLibrary** |
| *Participating actors* | Initiated by **Student** |
| *Flow of Events* | 1. **Student** selects search tab |
|  | 2. **Student** selects category to perform search |
|  | 3. **Student** enters text to search |
|  | 4. **Student** hits search button |
| *Entry Conditions* | **Student** is logged into **LibraryManagementSystem** |
| *Quality Requirements* | Query can be performed on database |

|  |  |
| --- | --- |
| ***Use case name*** | **LoginSystem** |
| *Participating actors* | Initiated by **Librarian/Administrator/Student** |
| *Flow of events* | 1. **User** enters login information |
|  | 2. **User** hits submit button |
| *Entry Conditions* | N/A |

|  |  |
| --- | --- |
| ***Use case name*** | **Add/RemoveCopiesofBooks** |
| Participating Actors | **Administrator** |
| *Flow of Events* | 1. **Administrator** selects remove/add tab |
|  | 2. **Administrator** increments/decrements numbers of book in library |
| *Entry Conditions* | **Administrator** is logged into **LibraryManagement System** |

# Activity Diagram



# Data Storage

We are using MS SQL Server database to store tables. Data storage transactions will be performed through ***Library*** Class in Model.

The following will be the primary tables:

## User Credentials Table

This table contains login credentials and details of the user which include Name, Email, Password Hash, Login Type etc.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr.No. | First Name | Last Name | Email ID | Password Hash | Login Type | Books Allowed | Books Borrowed |
| 1 | Tauseef | Indikar | [tain5575@colorado.edu](mailto:tain5575@colorado.edu) | XXXXXXX | Admin |  |  |
| 2 | Gaurav | Shukla | [gash7618@colorado.edu](mailto:gash7618@colorado.edu) | XXXXXXX | Student | 3 | 2 |

**Example Credentials Table**

## Inventory Table

This table contains book details. Additional columns might be added later.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sr.No. | ISBN | Title | Author | Category | Quantity Available | Quantity Checked Out | Location |
| 1 | XXX-XX-XX | Hacking | Daniel Regalado | Computer Science | 3 | 1 | Aisle 1  Shelf A1 |
| 2 | XXX-XX-XX | Harry Potter | J.K. Rowling | Novel | 1 | 0 | Aisle 10  Shelf A1 |

**Example Inventory Table**

## Request Table

This table contains requests for a book made by a student

|  |  |  |  |
| --- | --- | --- | --- |
| Sr.No. | Email ID | ISBN | Request Date |
| 1 | [gash7618@colorado.edu](mailto:gash7618@colorado.edu) | XXXXXXX | 2015-10-18 |

**Example Request Table**

## Checkout Table

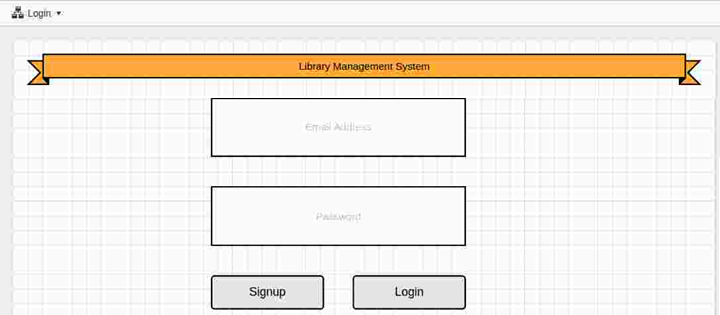
This table contains books currently checked out.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No. | Email ID | ISBN | Checkout Date | Due Date |
| 1 | [gash7618@colorado.edu](mailto:gash7618@colorado.edu) | XXXXXXX | 2015-10-18 | 2015-12-18 |

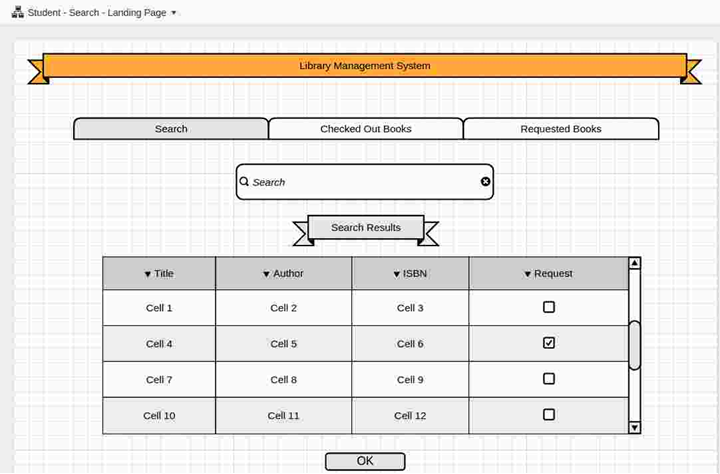
**Example Checkout Table**

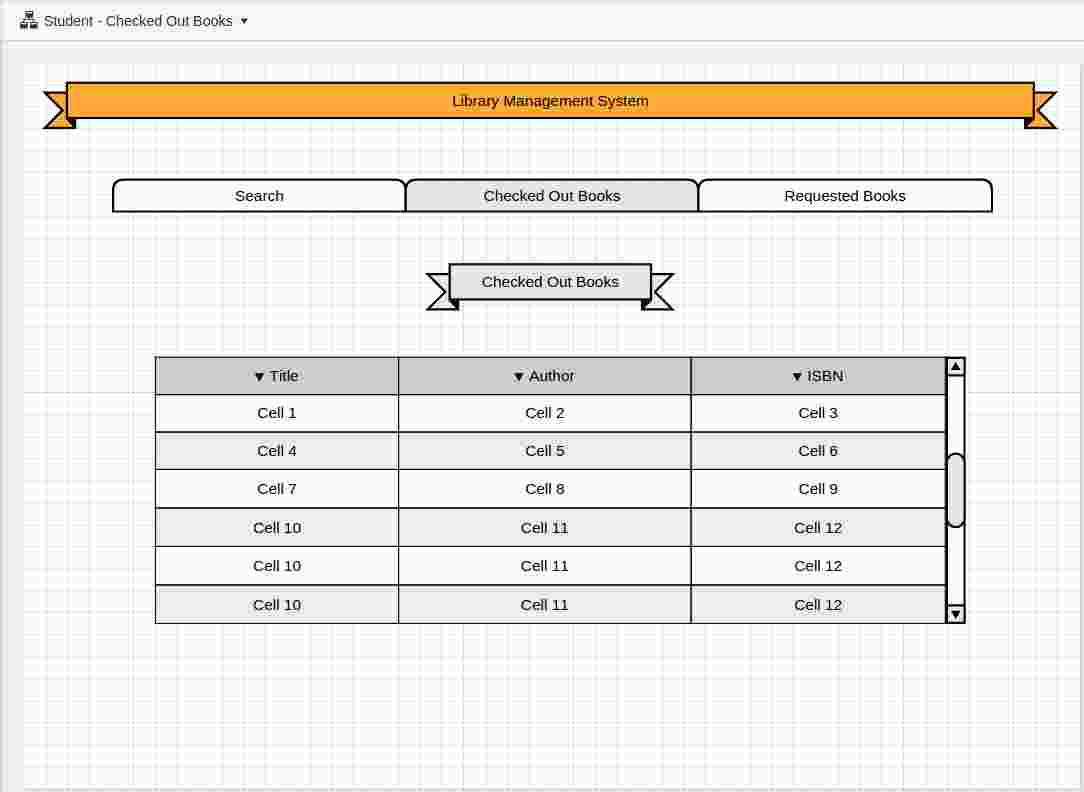
# UI Mockups

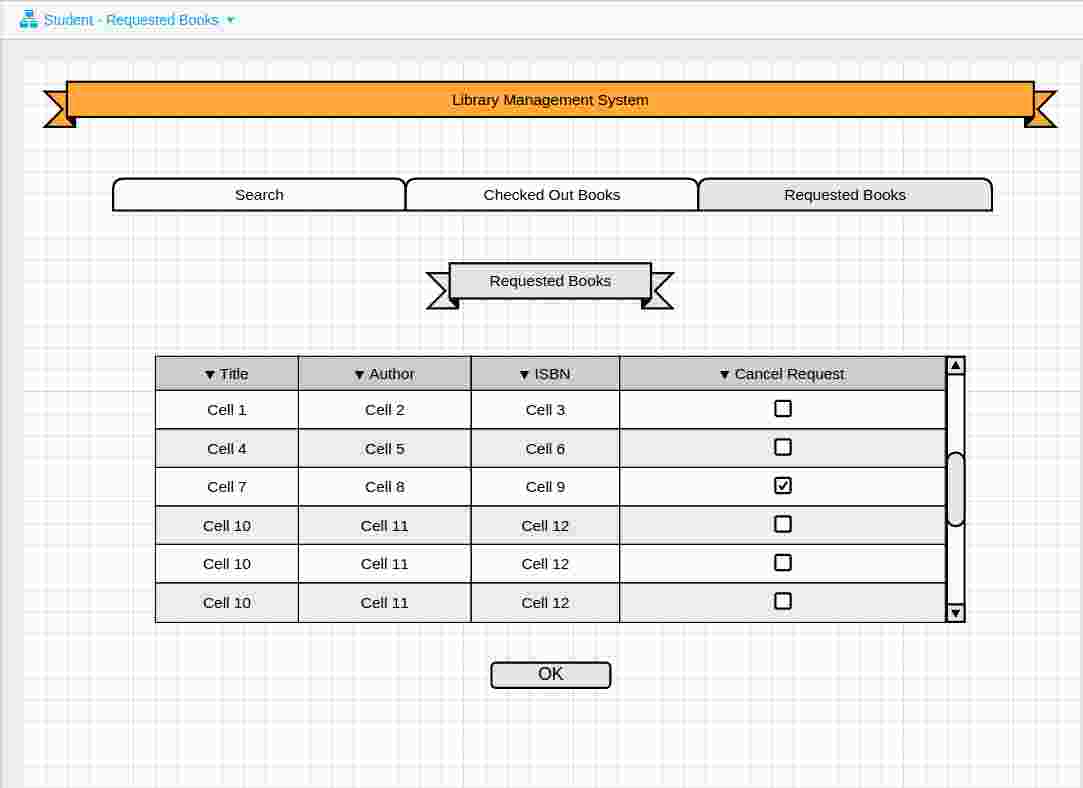
## Login UI



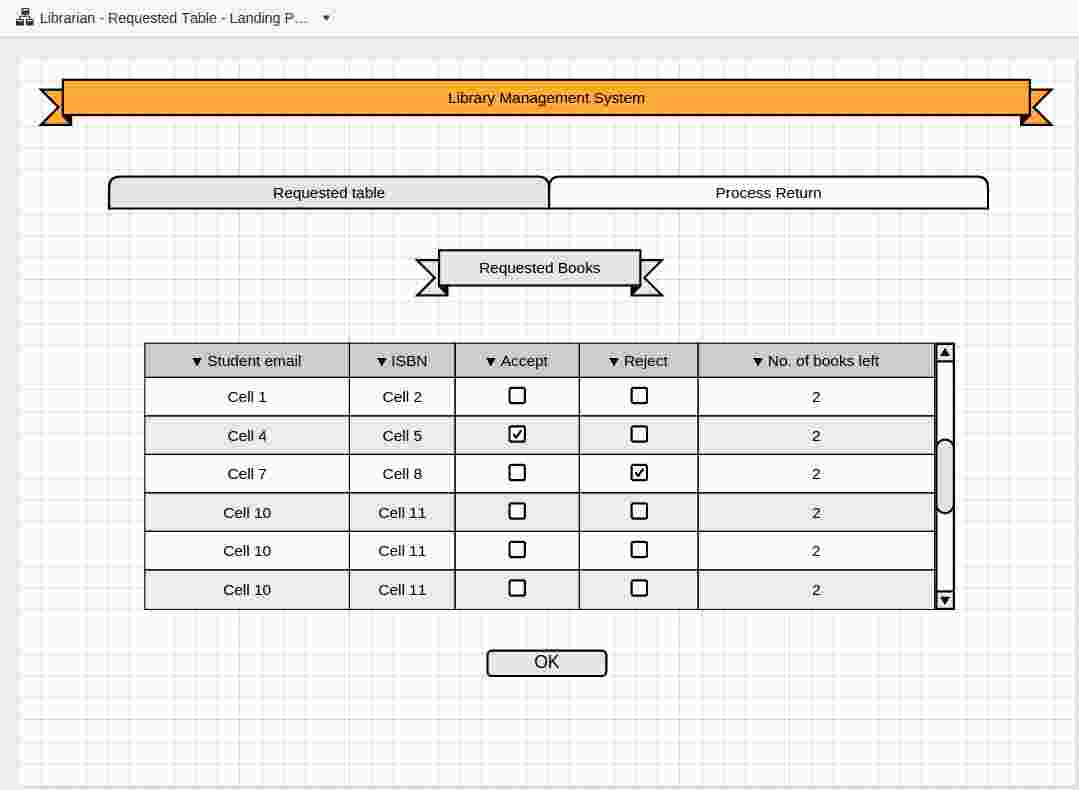
## Student UI

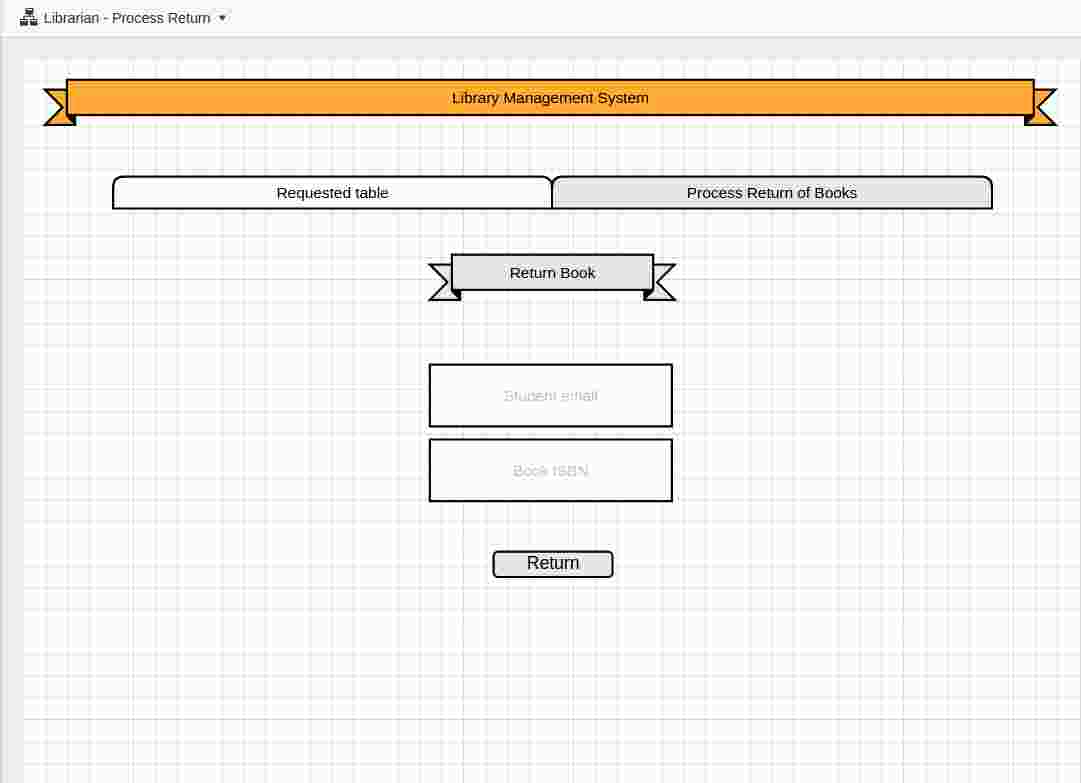




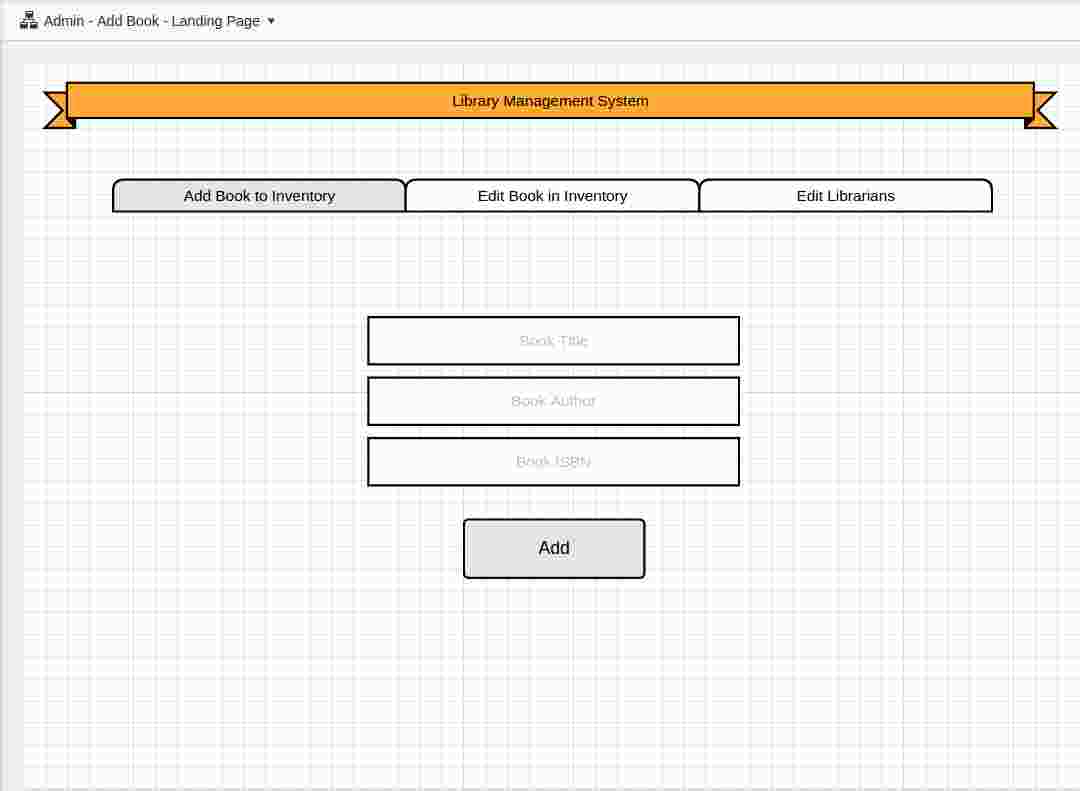


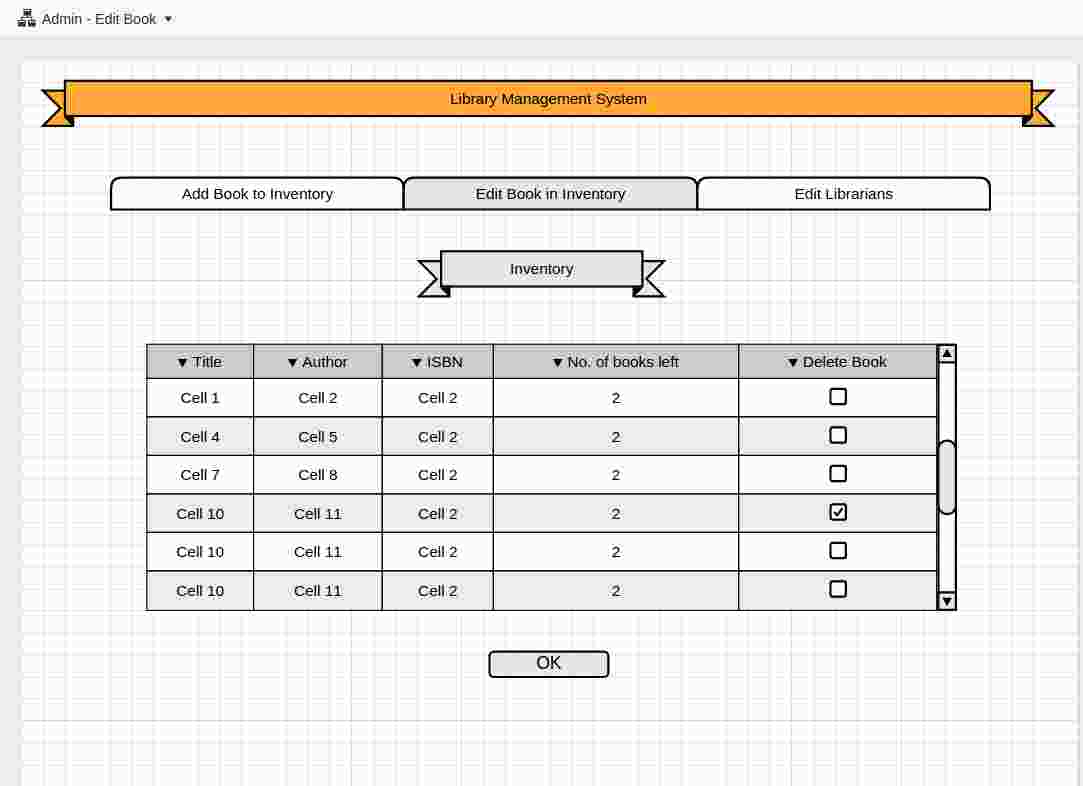
## Librarian UI

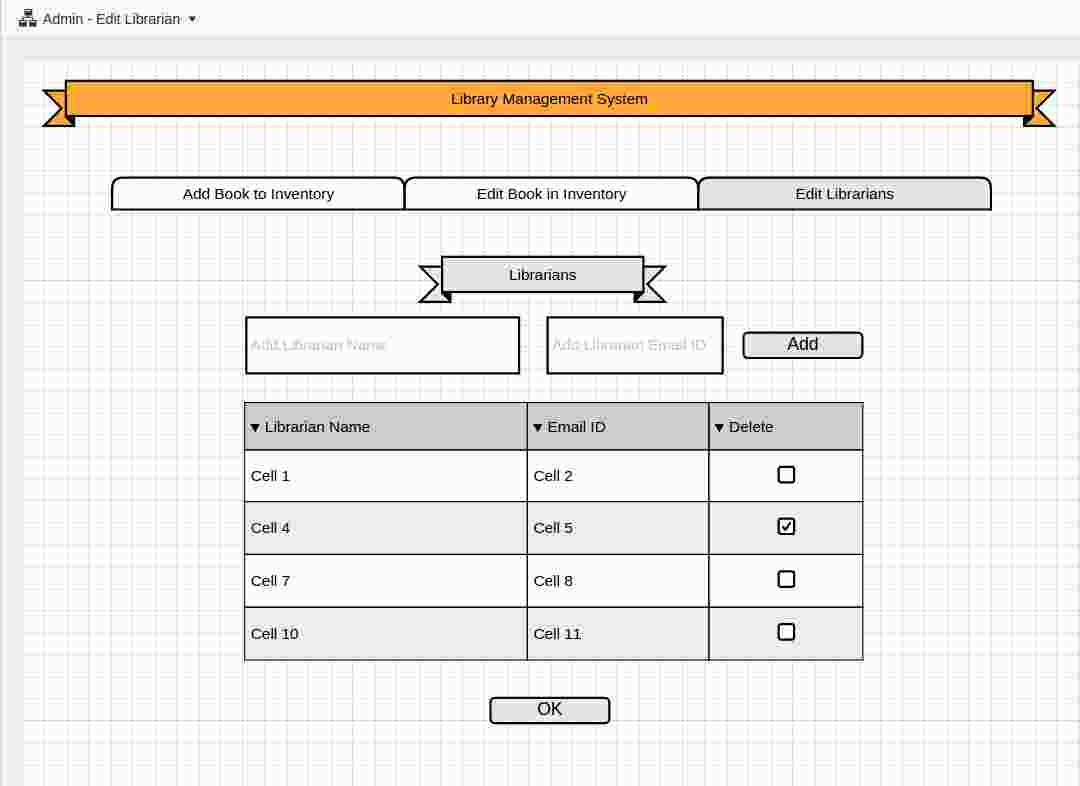




## Admin UI

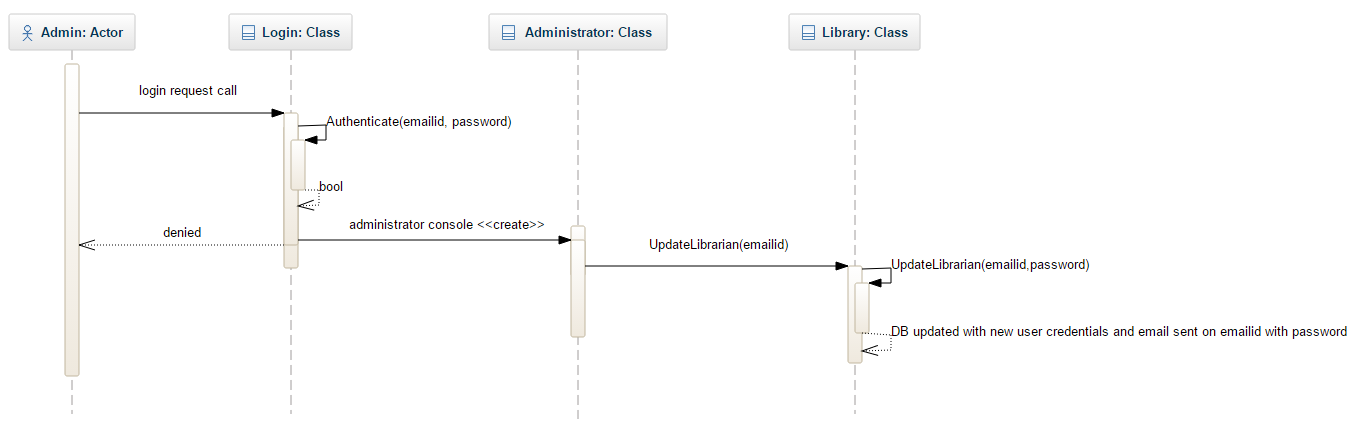






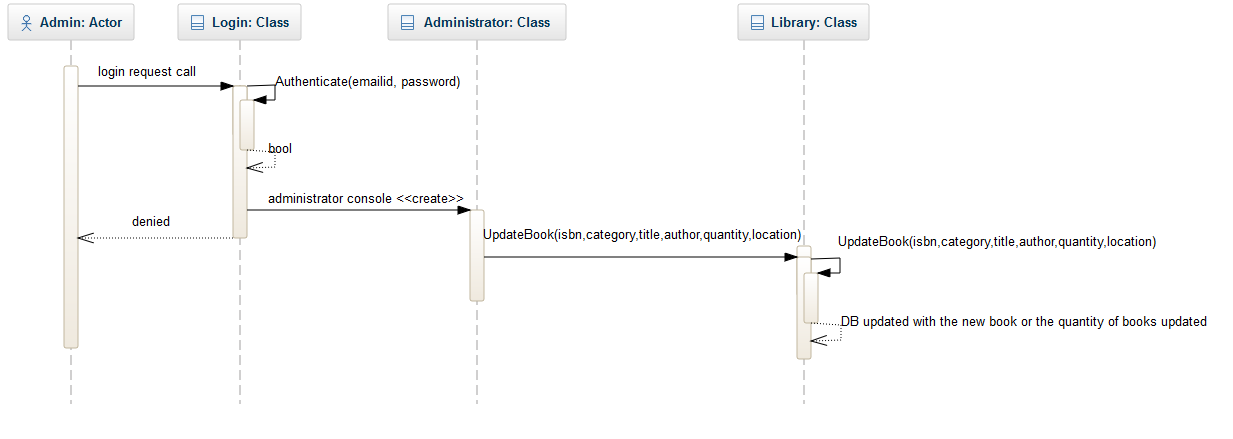
# User Interactions

## Sequence Diagram: Admin adding/updating a Librarian



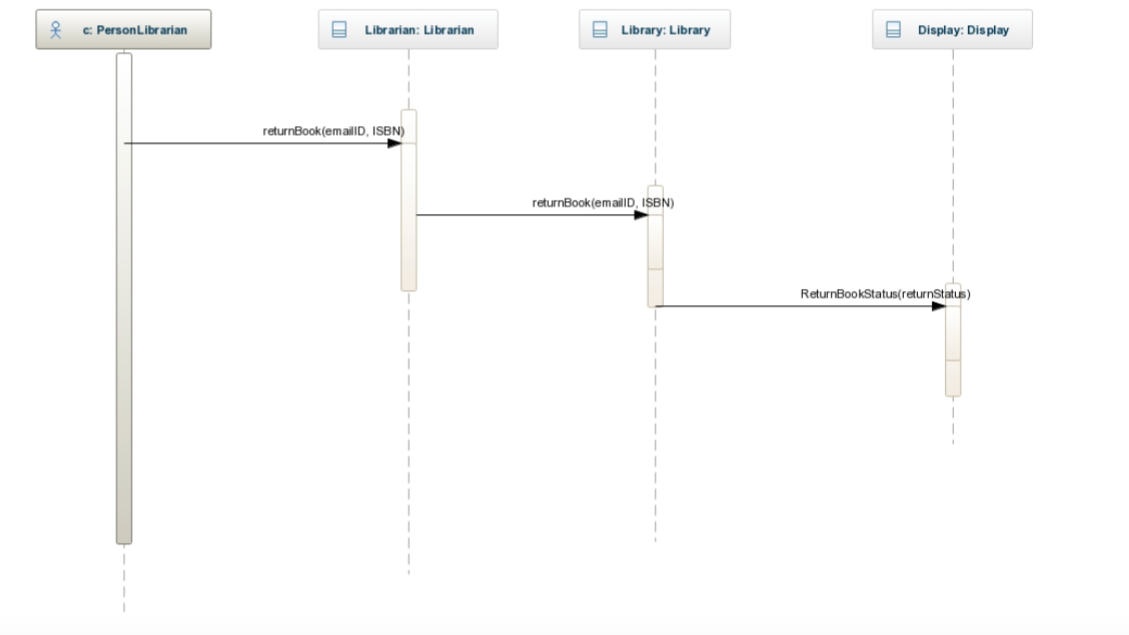
Admin adding a librarian

## Sequence Diagram: Admin adding/updating a book



Admin adding a book

## Sequence Diagram: Librarian Checking out a book



Librarian checking out a book

# Class Diagrams

