System Design Specification

Library Management System

15th November 2016

Team Members
140265D-P.Y.M. Jayawardana
140268N- H.M.G.N.Jayawardena
140161F-P.R.L.Fernando
140660J-D.M.Weerasooriya
140066T-K.P.T.K. Bandara

1. Introduction

1.1 Purpose and Scope

The purpose of this document is to provide a detailed and technical description of how LMS will be constructed and implemented. It contains all required information for a software developer to start implementing the Library Management System (LMS).

The document was prepared based on the Software Requirements Specification (SRS) document. All entities and requirements identified in it were taken into consideration, ensuring that the current design meets all requirements of the system as well as any implicit requirements identified later.

1.2 References

- SRS - Software Requirements Specification Document

1.3 Overview

This document outlines the System Overview, System Architecture, Data design, component design, Human Interface Design, and traces the requirements.

- System architecture is described using Class Diagrams, Activity Diagrams and Sequence Diagrams.
- Database Design is provided using ER Model diagrams and Relational Table design.
- Component Design section provides detailed information on the components defined in the system and how they interact and perform functions.
- Essential user interfaces and their design prototypes are included in the Human Interface Design Section
- The Requirements Traceability Matrix is provided in the latter part of the document.

1.4 Constraints

- The system should operate 24/7, and at least 90% uptime is required.
- Necessary student information should be retrieved from Student Information System upon registration and stored in a MySQL database.
- The book information from the Book Information System should be maintained in the MySQL database, and any updates should be synchronized among them.
- The system should be accessible via any device and browser supporting HTML5 and JavaScript, hence should have a responsive design.
- The requesting of unavailable items should be done via an interface that connects to the Expographic Books Service.

2. System Overview

This product is an extension for the currently existing Book Inventory System, Student Information System and Expographic Book Service. It uses the features of all three systems to create a library management platform available for the students to be able to request books from Expographic, reserve books from the library and check availability of books using a single system. The library administration can use this product to purchase books from Expographic company based on the requests of students.

There are two types of user classes for this system. Namely, Students and Library Administration.

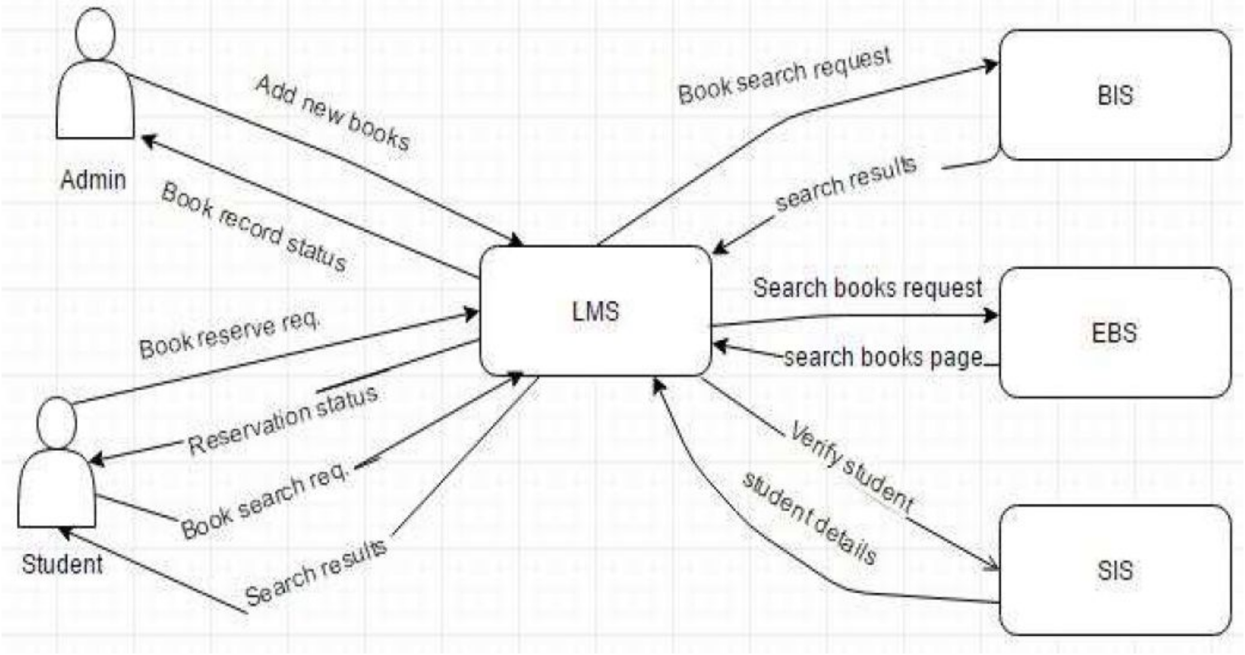
Students:

- The students use LMS to see available books in the library, make book reservations, suggest new books to the library and to view borrowed books, fines, etc. They are restricted from any administrative functions provided for library administration.

Library administration:

- The library administration use LMS to lend and retrieve books from the library, monitor student borrowed books and due dates, enforce fines on students, order books from EBS and to update BIS when new books are added to the library. Furthermore, handling password reset requests, library membership renewal and temporary student profile creation are also conducted by the library administration.

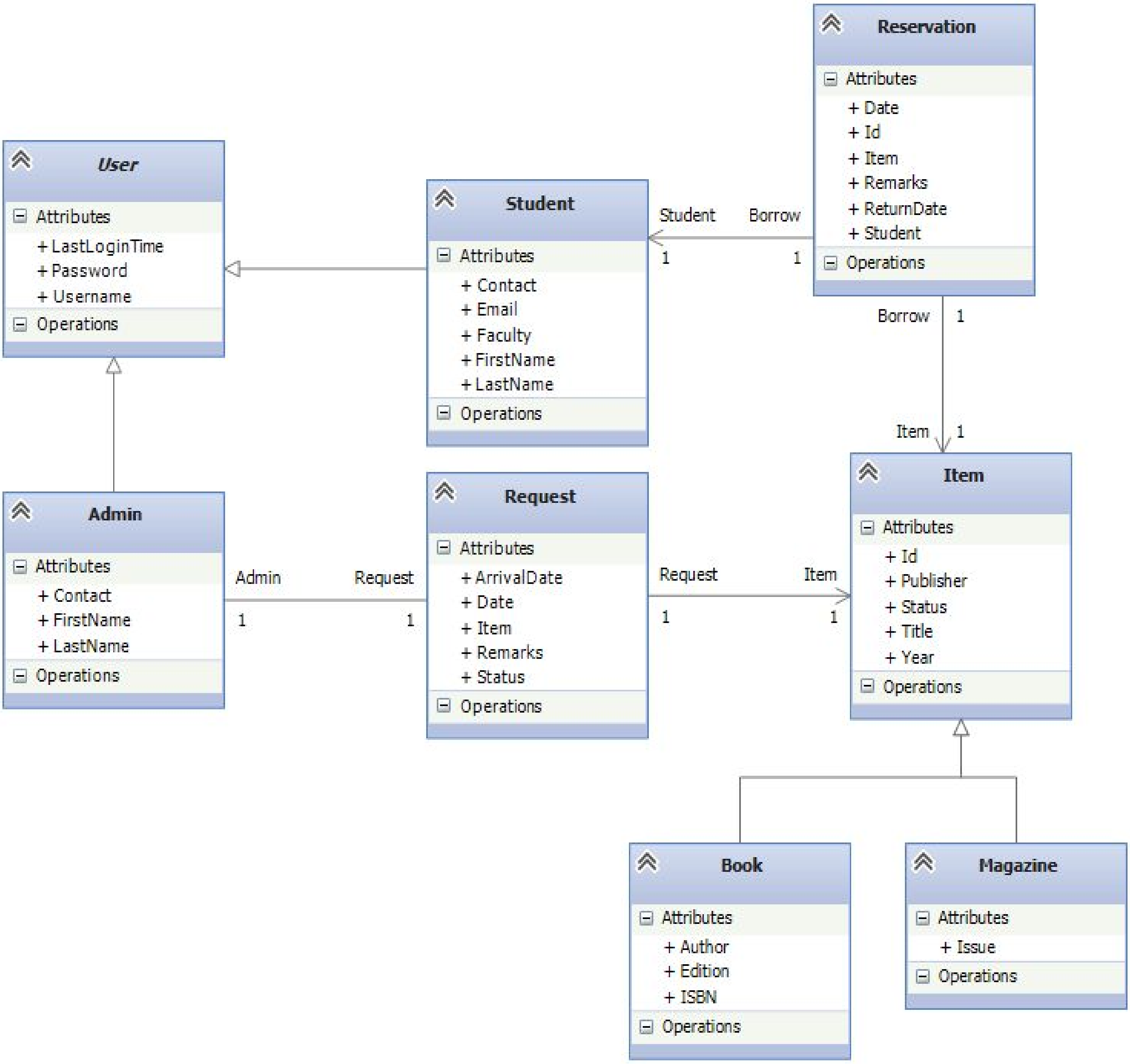
Any student currently registered in the university and has an index number can register and have access to the system from any location or device. The Library Administration can access LMS only from within the university network to ensure optimal security.



3. System Architecture

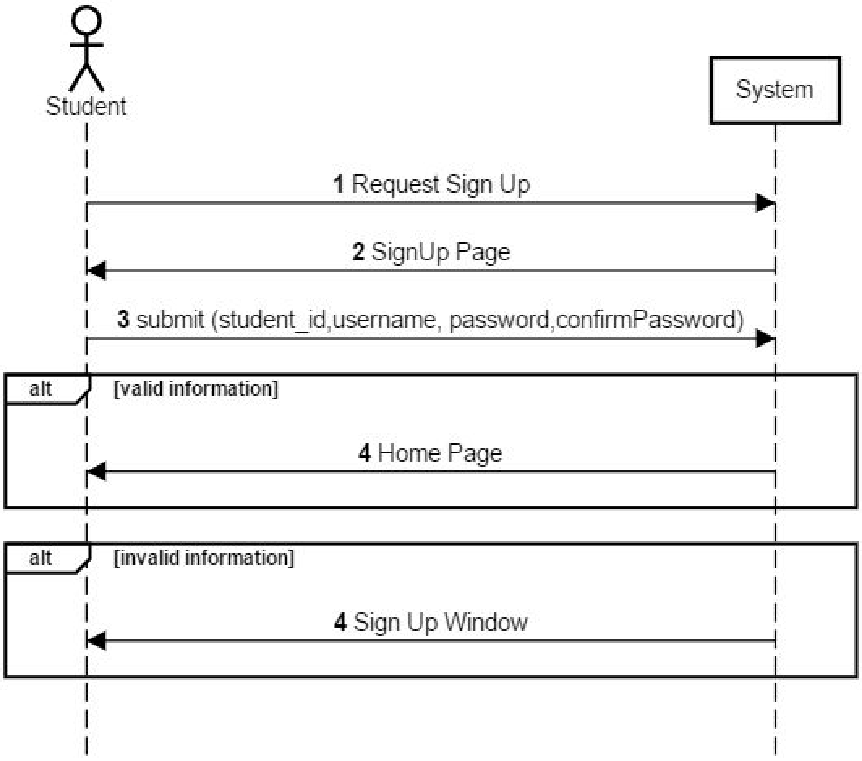
3.1 Architectural Design

3.1.1 Domain Class Diagram

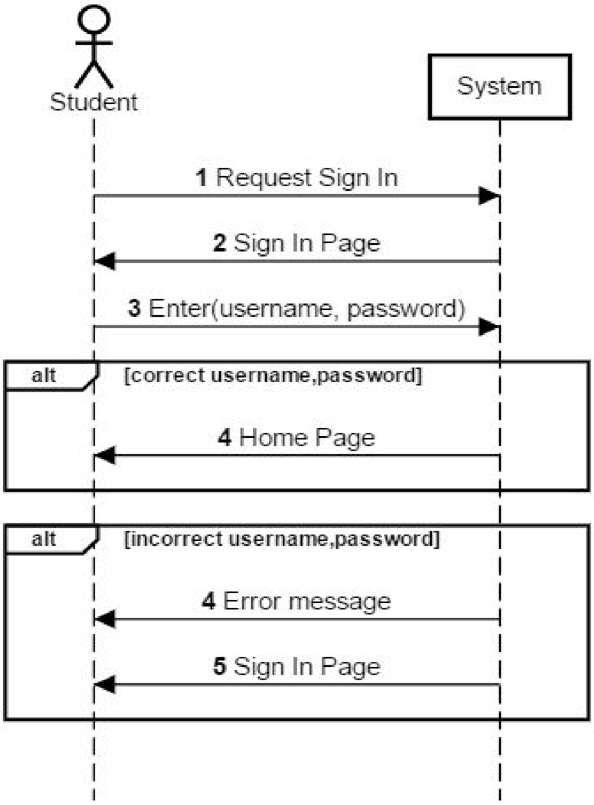


3.1.2 Sequence Diagrams

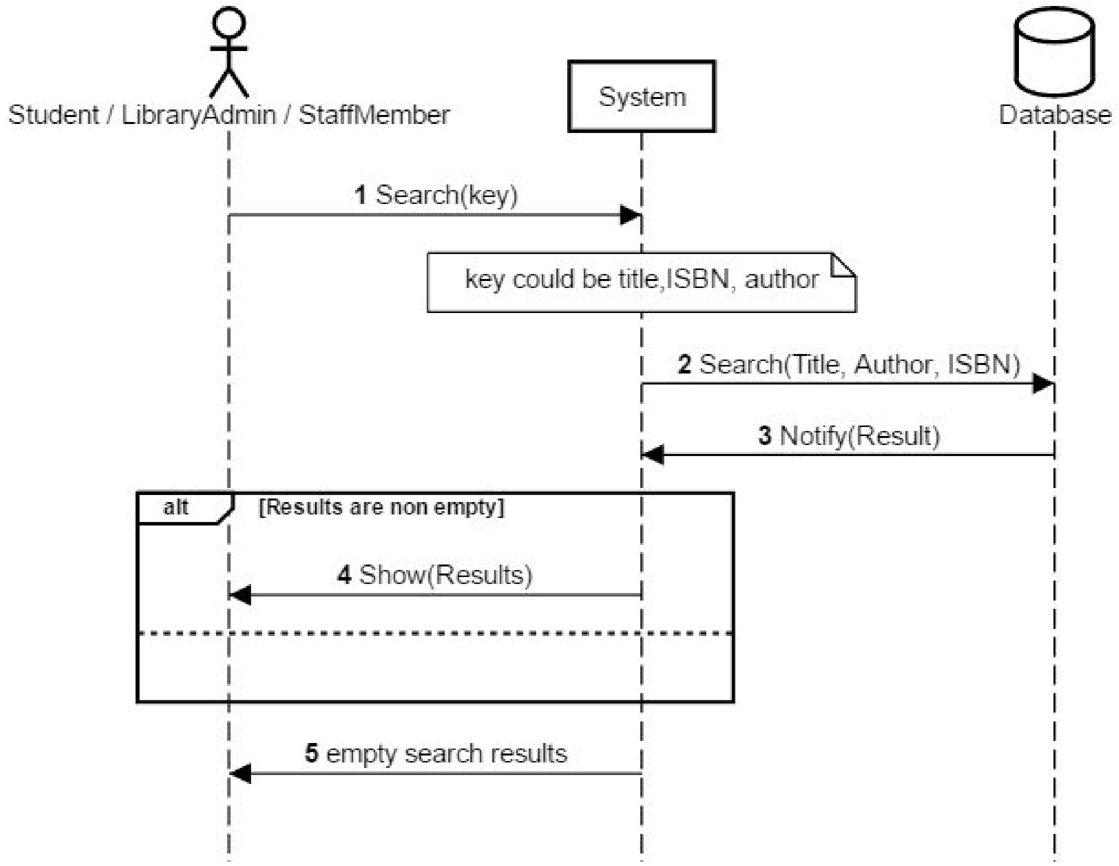
3.1.2.1 Student Sign Up



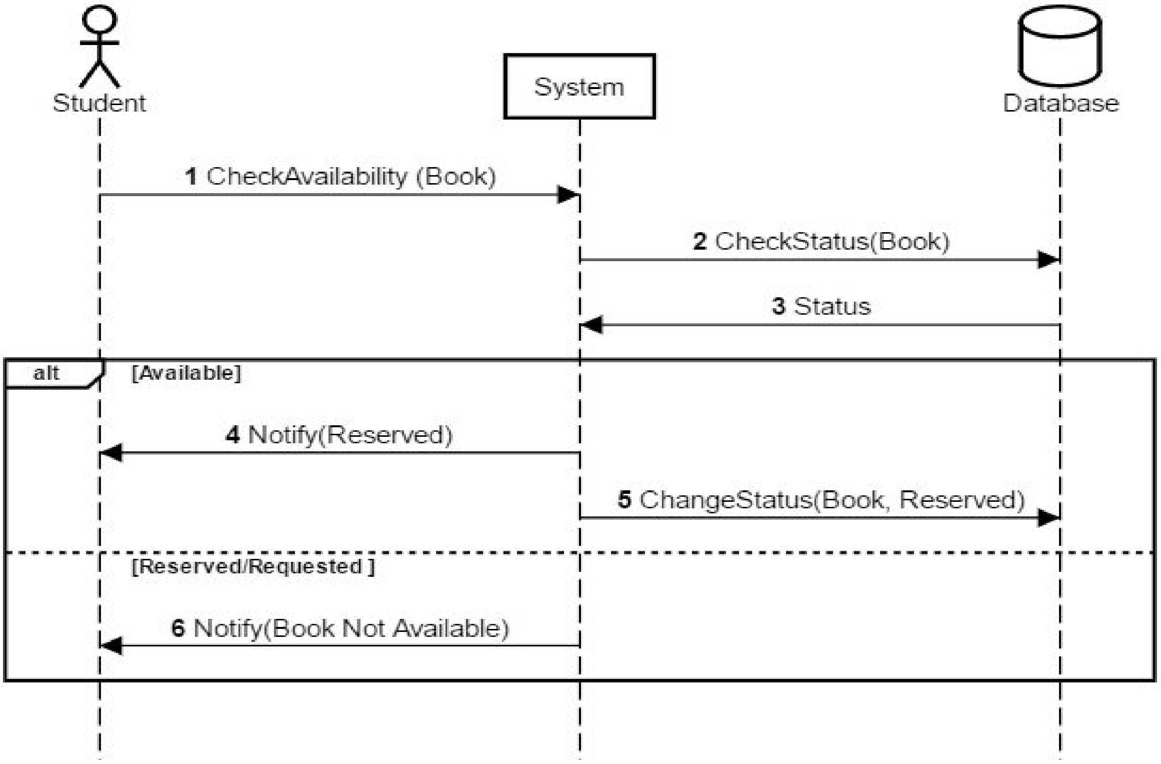
3.1.2.2 Student sign in



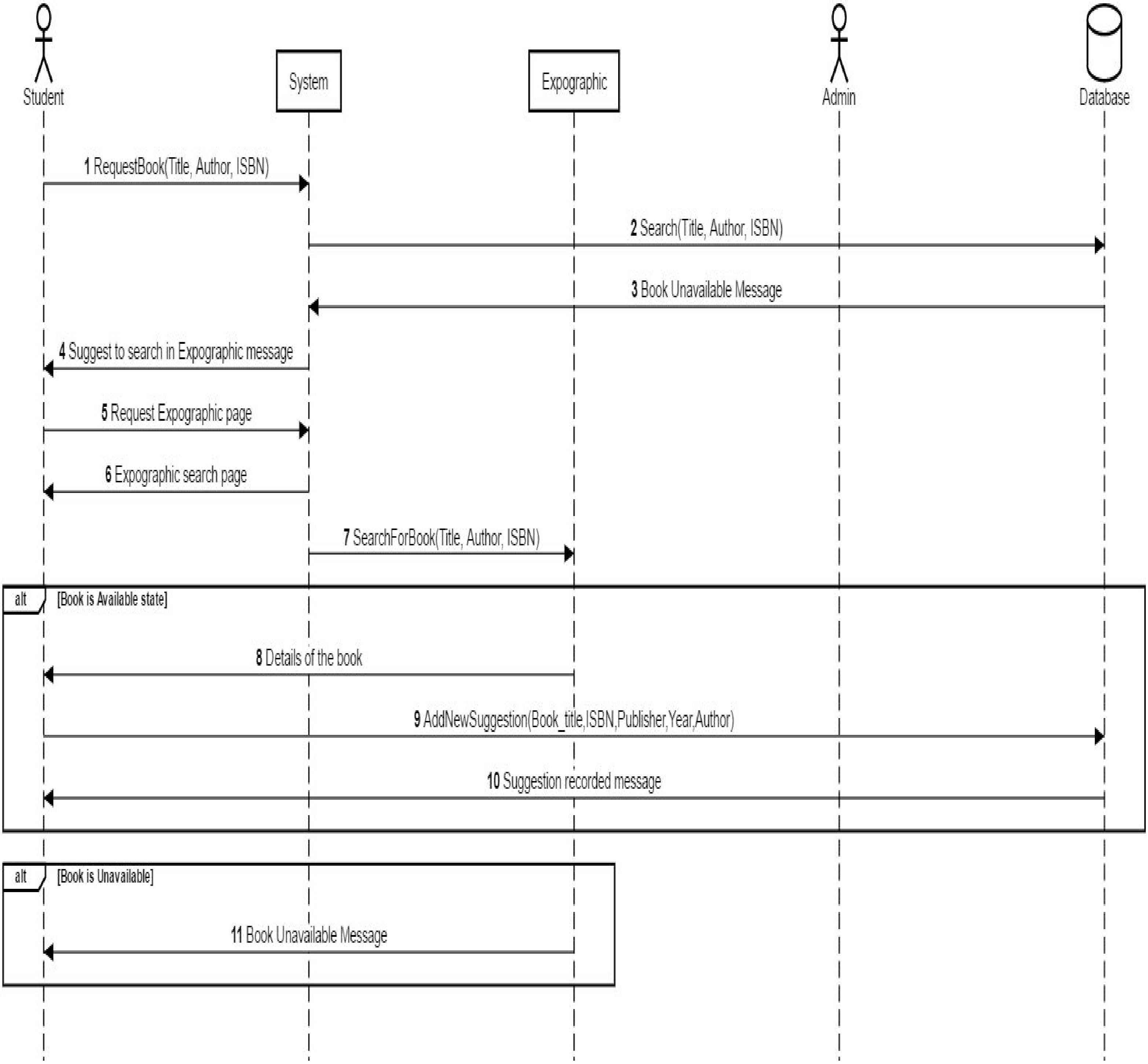
3.1.2.3 Search a book



3.1.2.4 Reserve Book

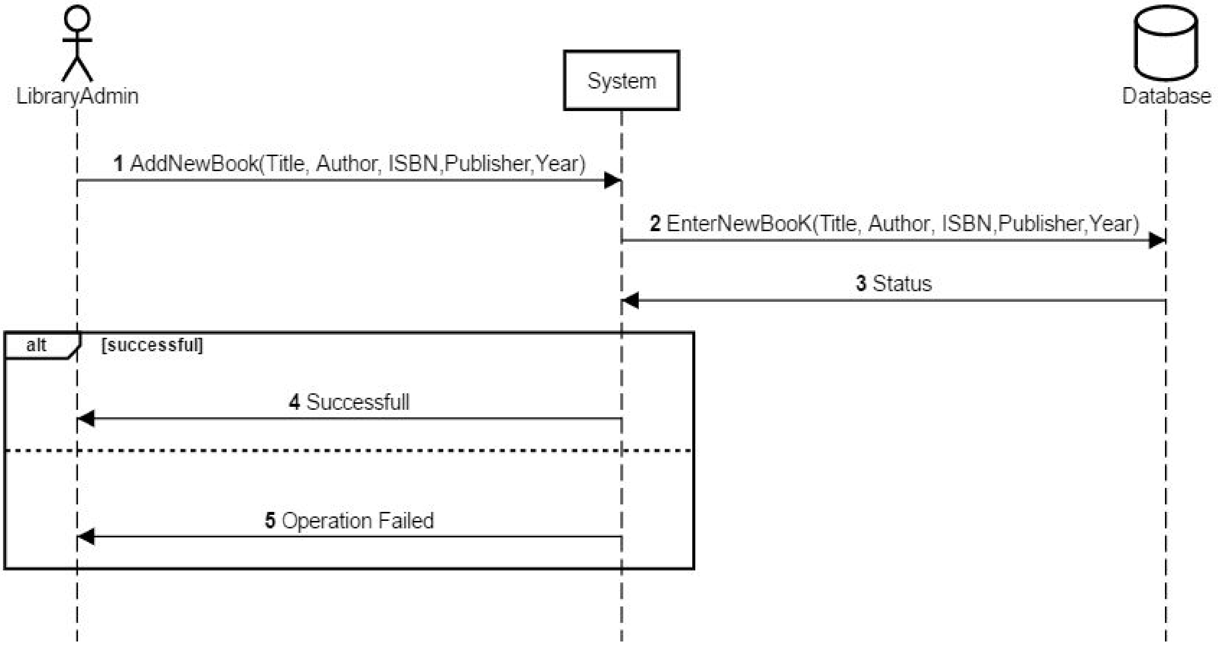


3.1.2.5 Add a book suggestion

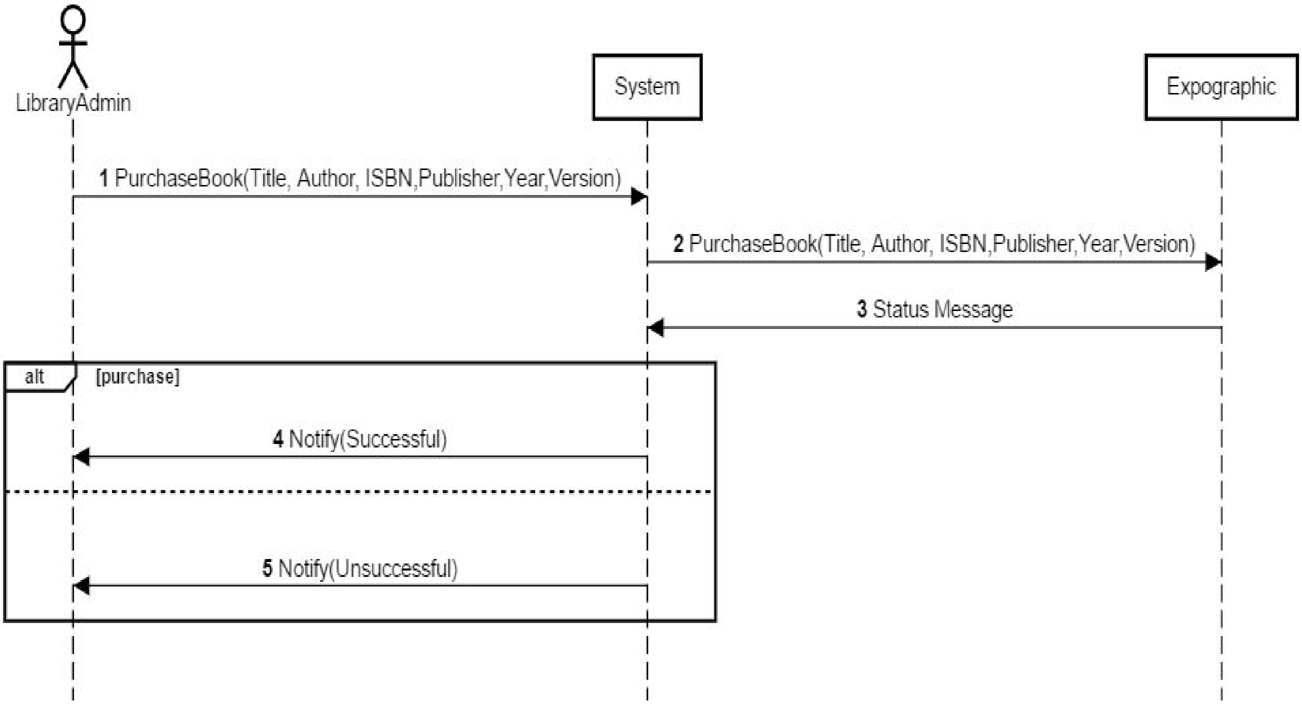


Student
System
Expographic
Admin
Database
1 RequestBook(Title, Author, ISBN)
2 Search(Title, Author, ISBN)
3 Book Unavailable Message
4 Suggest to search in Expographic message
5 Request Expographic page
6 Expographic search page
7 SearchForBook(Title, Author, ISBN)
alt
[Book is Available state]
8 Details of the book
9 AddNewSuggestion(Book_title,ISBN,Publisher,Year,Author)
10 Suggestion recorded message
alt
[Book is Unavailable]
11 Book Unavailable Message

3.1.2.6 Add New Book


LibraryAdmin
System
Database
1 AddNewBook(Title, Author, ISBN,Publisher,Year)
2 EnterNewBooK(Title, Author, ISBN,Publisher,Year)
3 Status
alt
[successful]
4 Successfull
5 Operation Failed

3.1.2.7 **Purchase a book from Expographic**


LibraryAdmin
System
Expographic
1 PurchaseBook(Title, Author, ISBN,Publisher,Year,Version)
2 PurchaseBook(Title, Author, ISBN,Publisher,Year,Version)
3 Status Message
alt
[purchase]
4 Notify(Successful)
5 Notify(Unsuccessful)

3.1.3 Design Patterns

Enterprise Service Bus Architecture
ESB allows communication between different platforms. ESB is used in interchanging data with Expographic Book Service, Book Inventory System, Student Information System and IMS Data Service.

Model-View-Controller Architecture
MVC is used to connect the business logic of the LMS with the user interfaces.
1) Domain Model
2) User Interface
3) Controller
MVC separates the business logic and user interactions.

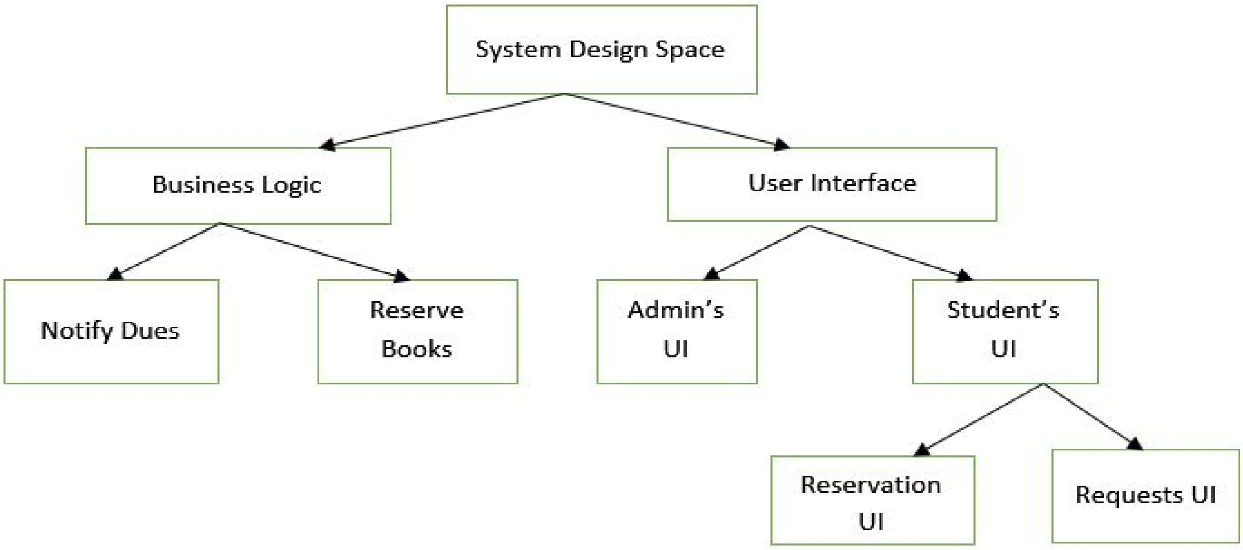
3.2 Architectural Alternatives

Broker Design Pattern (alternative for MVC architecture)
In MVC, controller class show similarities with the Broker class in Broker Design Pattern. They both communicate with different architectural elements.

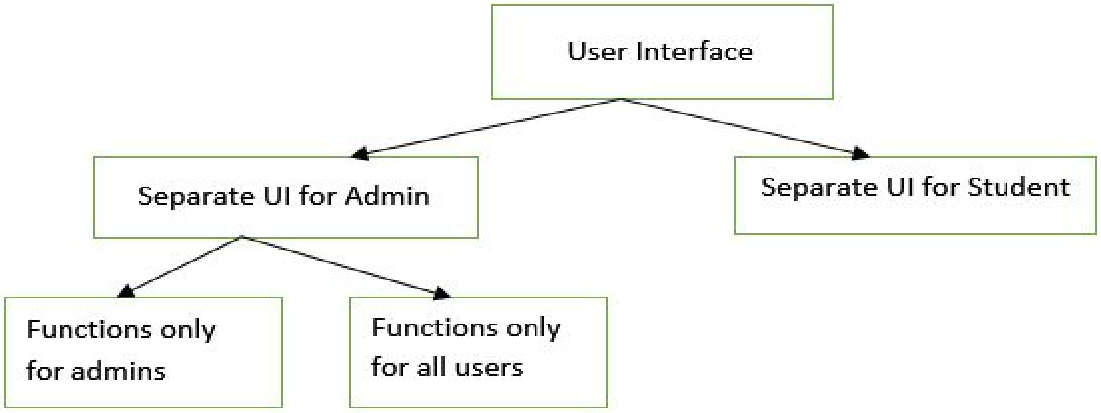
MVC was used because it is scalable in long run compared to Broker.
In MVC, components can be mapped to LMS's components directly. But Broker does not allow direct mapping of architectural elements.

3.3 Design Rationale

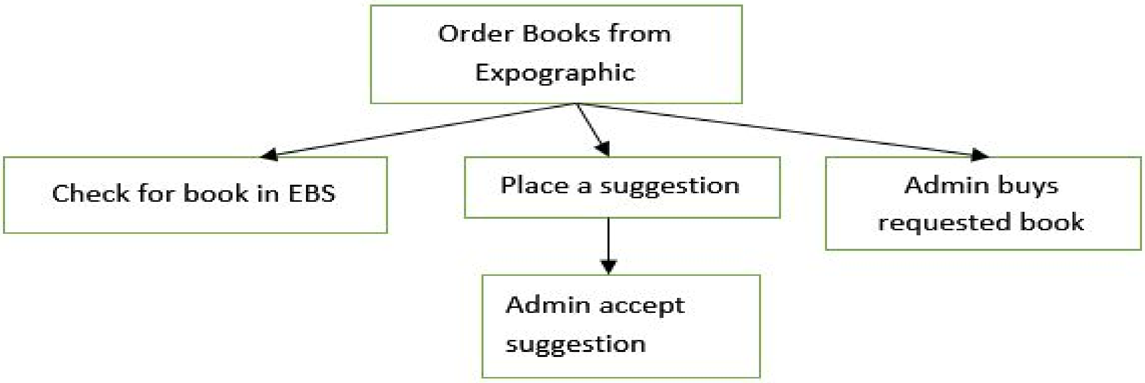
Discuss the rationale for selecting the architecture described in 3.1, including the critical issues and trade/offs that were considered. You can use design space analysis techniques such as design tree and questions options criteria.



3.3.1 User Interface Decision



3.3.2 Business Logic Decision



4. Data Design

4.1 Database Description

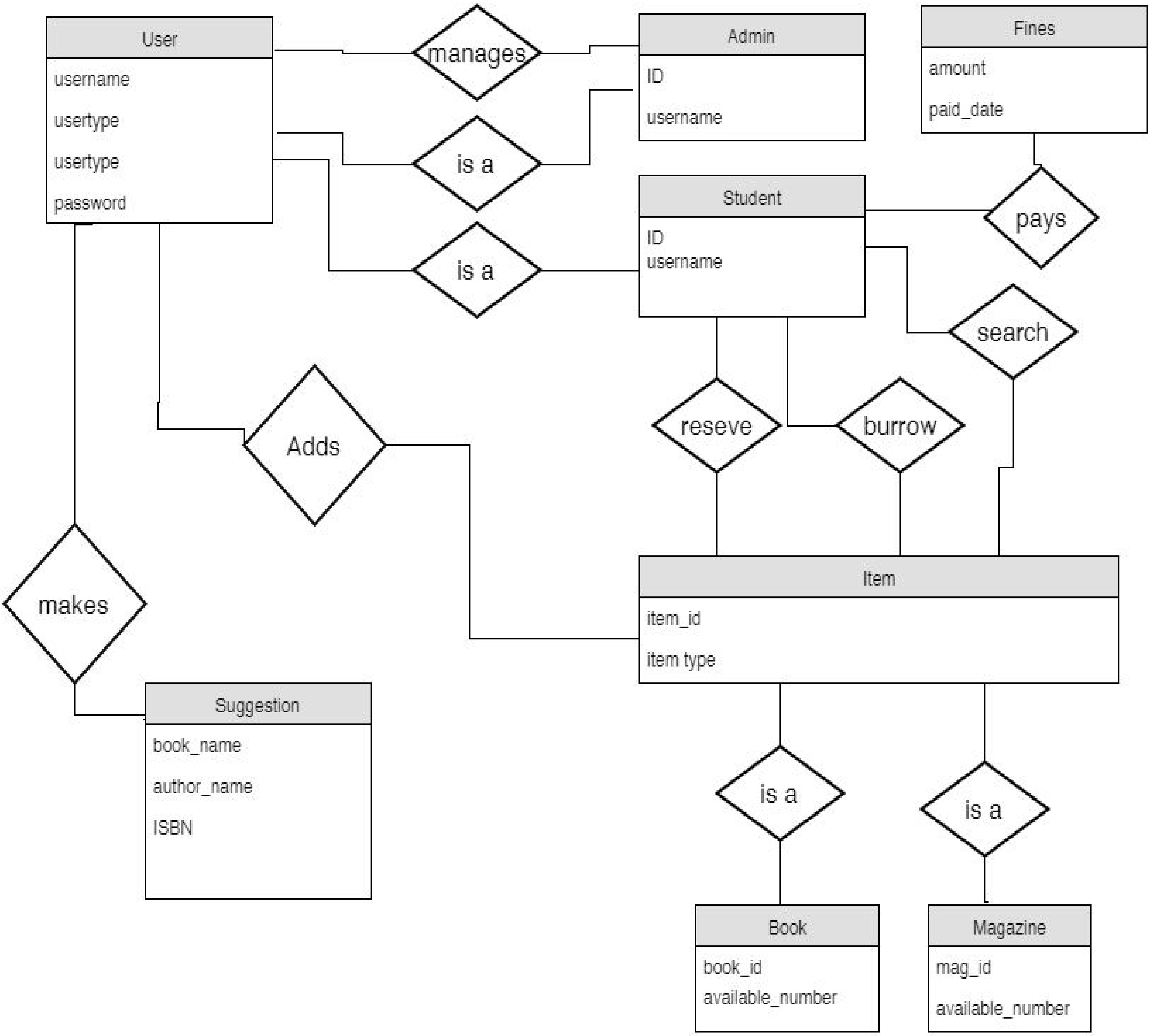


Figure: Entity Relationship Diagram – Complete System

4.2 Global Data Structures

User		
Field name	**Data type**	**Description**
username	Varchar(45)	Unique username
password	varchar(45)	Not null (encrypted)
user_type	varchar(45)	Not null- an admin or a student

Admin		
Field name	**Data type**	**Description**
username	Varchar(45)	Unique- foreign key References-user table
status	boolean	Currently an admin or not

Student		
Field name	**Data type**	**Description**
username	Varchar(45)	Unique- foreign key References-user table
s_id	varchar(10)	Unique

Item		
Field name	**Data type**	**Description**
item_id	Varchar(45)	Unique item_id
type	varchar(45)	Not null-book or a magazine

Book		
Field name	**Data type**	**Description**
item_id	Varchar(45)	Unique- foreign key References-item table
availability	boolean	Not null-availability in the library

Magazine		
Field name	**Data type**	**Description**
item_id	Varchar(45)	Unique- foreign key References-item table
availability	boolean	Not null-availability in the library

Reservations		
Field name	**Data type**	**Description**
student_id	Varchar(45)	Not null- foreign key
item_id	boolean	Not null - foreign key
date	date	Not null
status	varchar(45)	Not null

Borrow		
Field name	**Data type**	**Description**
student_id	Varchar(45)	Not null- foreign key
item_id	boolean	Not null- foreign key
date_borrowed	date	Not null
due_date	date	Not null

Suggestions		
Field name	**Data type**	**Description**
book_name	Varchar(45)	Not null
ISBN	varchar(60)	Not null
author	varchar(50)	Not null
publisher	varchar(50)	Not null

New_Book		
Field name	**Data type**	**Description**
book_id	varchar(10)	unique
book_name	Varchar(45)	Not null
isbn	varchar(60)	Not null
author	varchar(50)	Not null
publisher	varchar(50)	Not null

Fines		
Field name	**Data type**	**Description**
amount	varchar(10)	Not null
due_date	Varchar(45)	Not null
item_id	varchar(60)	Not null
student_id	varchar(50)	Not null- foreign key

5. Component Design

5.1 Database

Identity	LMS.DB
Purpose	storage of data for the application.
Function	receives SQL commands such as INSERT, UPDATE, SELECT from the application and fetch data as necessary
Subordinate	1) Student ● User_name ● student_id 2) User ● Username ● Password ● User_type 3)Admin ● Username ● Status 4) Item ● Item_id ● Item_type 5) Book ● Book_id ● availability 6) Magazine ● Item_id ● availability 7)Reservations ● student_id ● item_id ● Date ● Status 8) Borrow ● student_id ● item_id ● date_borrowed ● due_date 9) Suggestions ● book_name ● isbn ● author

	● Publisher 10)New_Book ● book_id ● book_name ● ISBN ● author ● publisher 11)Fine ● amount ● due_date ● item_id ● student_id
Dependencies	An administrative user should set up the database, modifying the database tables so that the software can store data properly.
Interface	All database transactions will be done through the provided application interfaces through the MySQL connectors.
Data	Data will be added through the interfaces as descripted above. For each query there will either be a result set, an update / insert function or an exception if any errors are present in the query.

5.2 Data Access Objects

Identity	LMS.DAO
Purpose	Contains a package of classes to connect the interface with the database.
Function	Provide an Object oriented approach for the database tables through relevant classes.
Subordinate	All classes inside will inherit the properties of the Data Access Object defined under the interfaces section. The class breakdown is as follows. 1) User class 2) Student class 3) Item class 4) Book class extends Item class 5) Magazine class extends Item class 6) Reservation class

	7)Borrow class 8) Suggestion class 9)New Book class 10)Fine class
Dependencies	The software uses the class library to handle data. The Server software will use Data Access Objects in order to load and store data from the database
Interface	All DAO subclasses should follow the above defined class that defines the basic CRUD operations.
Data	All subclasses will be described in following sections.

5.2.1 User DAO

Identity	LMS.DAO.User
Purpose	DAO class to provide transactions with User table
Function	DAO class to provide transactions with User table
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interface	Not available
Data	Not available

5.2.2 Student DAO

Identity	LMS.DAO.Student
Purpose	DAO class to access and modify Student table
Function	Database operations to manipulate data from Student table. All the fields of the Student table are attached to the class as attributes.
Subordinate	Not available

Dependencies	Interface and controller classes will interact with this object to access the database.
Interface	Not available
Data	Not available

5.2.3 Item DAO

Identity	LMS.DAO.Item
Purpose	DAO class to access and modify Item table
Function	Act as the superclass for subclasses 1) Book 2)Magazine
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interface	Not available
Data	Not available

5.3.1 Book DAO

Identity	TerraLMS.DAO.Item.Book
Purpose	DAO class to access and modify Book table
Function	Extends the Item class and creates a Book that is specific inside the application.
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interfaces	Extends the Item class
Data	Not available

5.3.2 Magazine DAO

Identity	LMS.DAO.Item.Magazine
Purpose	DAO class to access and modify Magazine table
Function	Extends the Item class and represent a Journal is specific inside the application.
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interfaces	Extends the Item class
Data	Not available

5.2.4 Reservation DAO

Identity	LMS.DAO.Reservation
Purpose	DAO class to access and modify Reservation table
Function	Basic DB operations related to Reserve table. Has all the fields of the Reserve table attached to the class as attributes.
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interfaces	Not available
Data	Not available

5.2.5 Borrow DAO

Identity	LMS.DAO.Borrow
Purpose	DAO class to access and modify Borrow table

Function	Basic DB operations related to Borrow table. Has all the fields of the Borrow table attached to the class as attributes.
Subordinate	N/A
Dependencies	Interface and controller classes will interact with this object to access the database.
Interfaces	N/A
Data	N/A

5.2.6 Suggestion DAO

Identity	LMS.DAO.Suggestion
Purpose	DAO class to access and modify Suggestion table
Function	Basic DB operations related to Suggestion table. Has all the fields of the Suggestion table attached to the class as attributes.
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interfaces	Not available
Data	Not available

5.2.5 NEW BOOK DMO

Identity	LMS.DAO.Reserve
Purpose	DAO class to access and modify New Book table
Function	Basic DB operations related to New Book table. Has all the fields of the New Book table attached to the class as attributes.
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interfaces	Not available

Data	Not available

FINE DMO

Identity	LMS.DAO.Fine
Purpose	DAO class to access and modify Fine table
Function	Basic DB operations related to Fine table. Has all the fields of the Fine table attached to the class as attributes.
Subordinate	Not available
Dependencies	Interface and controller classes will interact with this object to access the database.
Interfaces	Not available
Data	Not available

6. Human Interface Design (Screens)

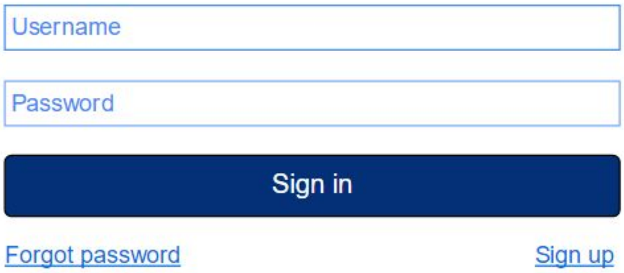
6.1 Overview of the User Interface

Two main User Interfaces.
i. Admin Interface
ii.Student Interface

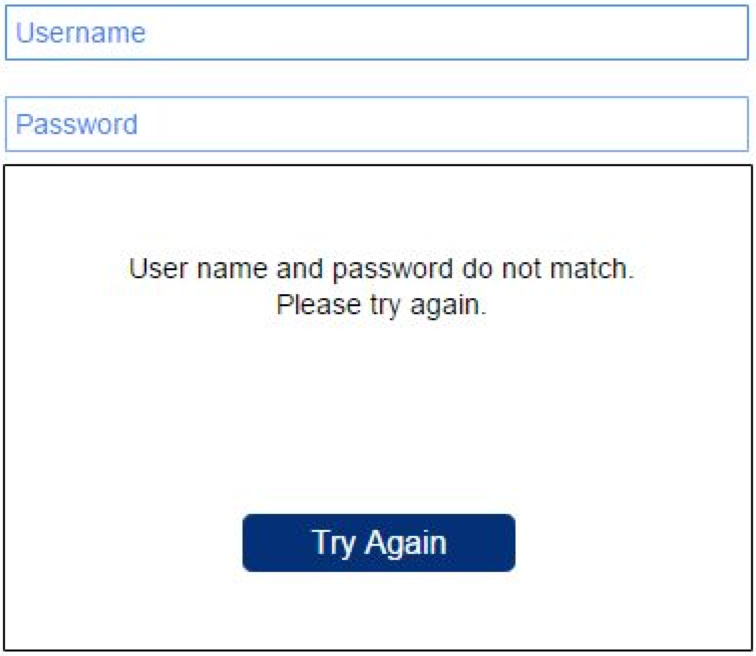
6.2 Screen Images

Screenshots showing the interface from the user's perspective. (This section can be unified with section 6.3)

These can be hand-drawn or you can use an automated drawing tool. Just make them as accurate as possible.

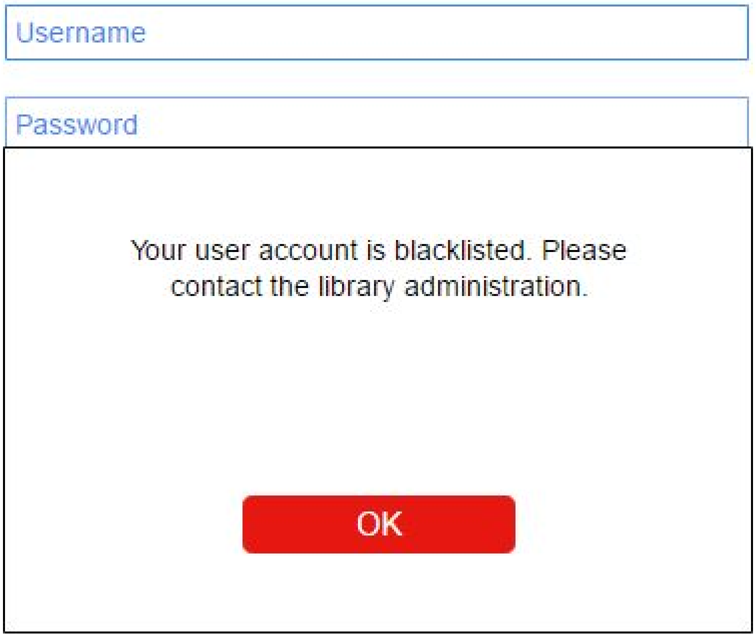


User login will be same for the all type of users. The access control function will determine the level of access based on the user type. The user type will be triggered by the user ID, and the appropriate menu will be displayed. The above screen will be displayed for the initial login to the system. The data submitted will be verified via the access control function.



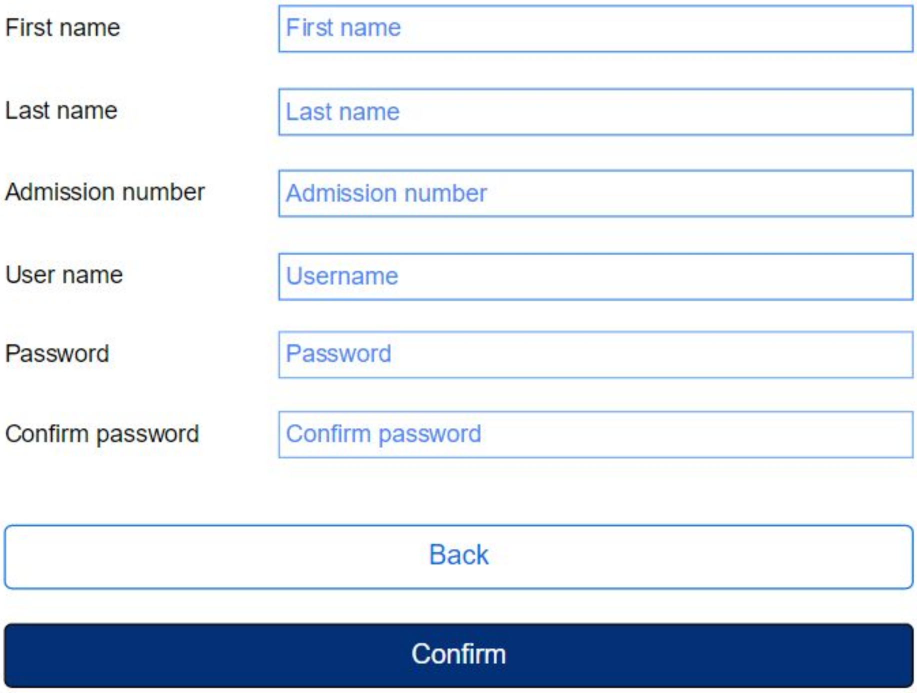
User name and password do not match.
Please try again.

If the Username and Password does not match above error message will be displayed.

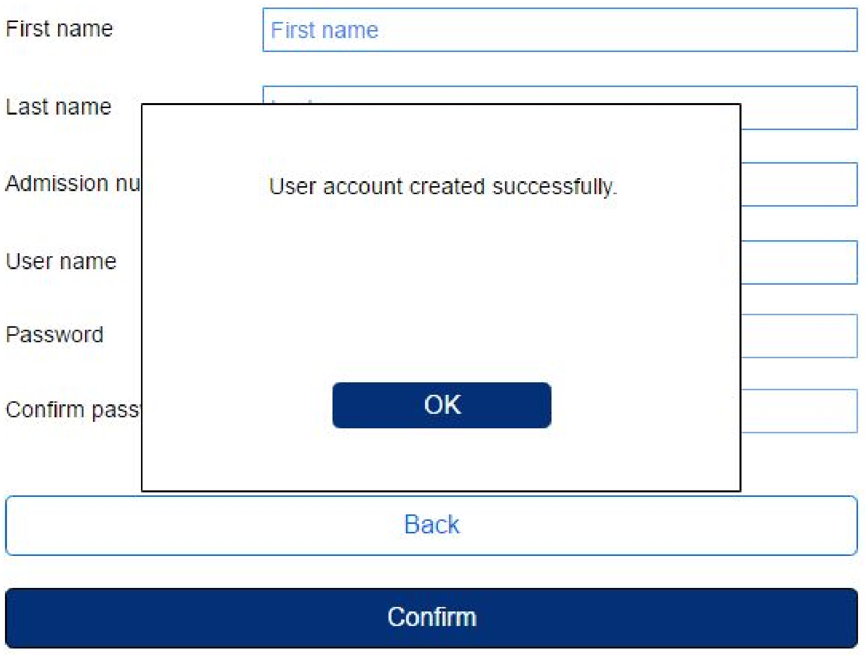


Your user account is blacklisted. Please contact the library administration.

If the User account was blocked by the admin then the above error message will be displayed.

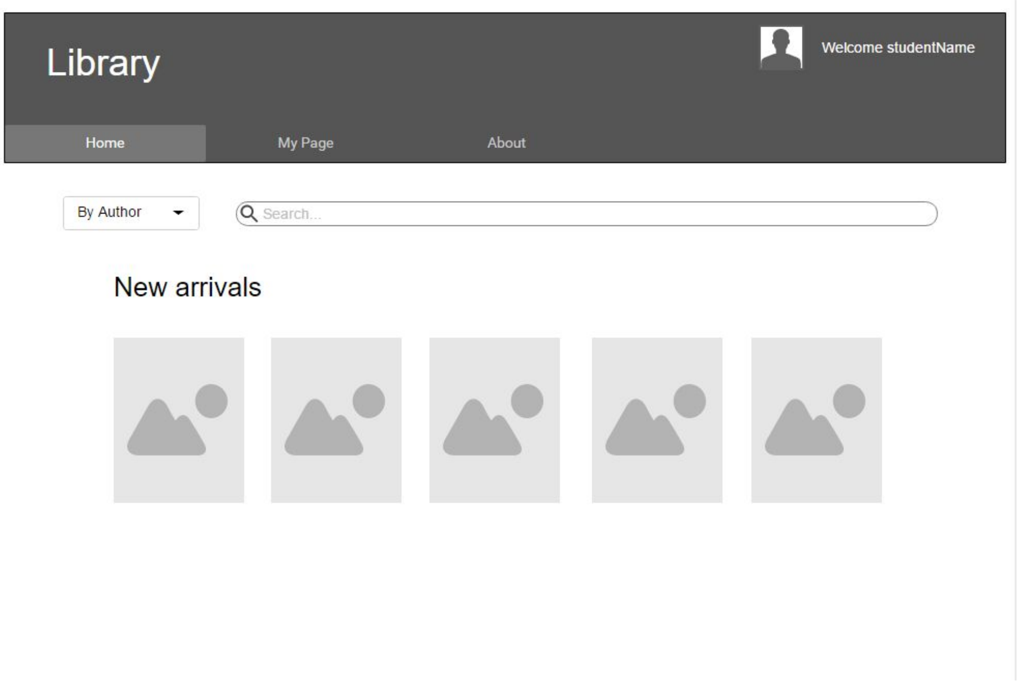


If user not existing then he/she can sign up with clicking sign up button. Then below interface will displayed.

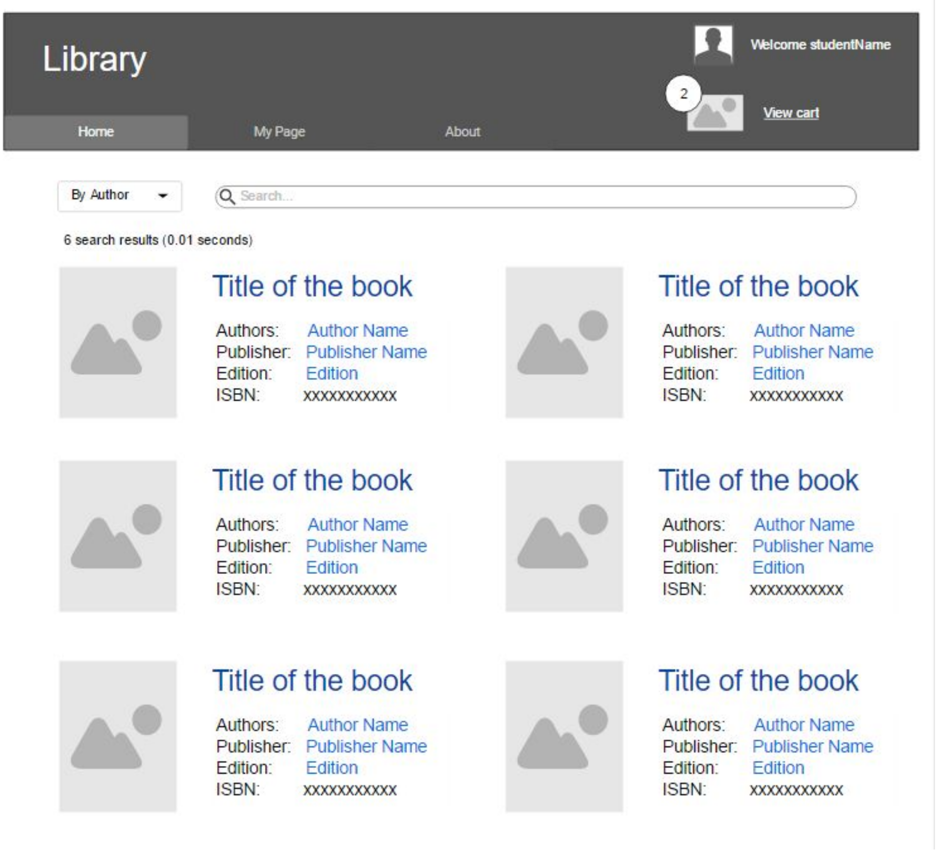


User account created successfully.

If user account created successfully then above message is displayed. And user will redirect to the main windows after clicking OK button.



After user sign up or sign in above interface will be displayed to Student. He/she can search books and he/she can change the searching parameter. Search parameters include Author, Title, and Any.



6 search results (0.01 seconds)

Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

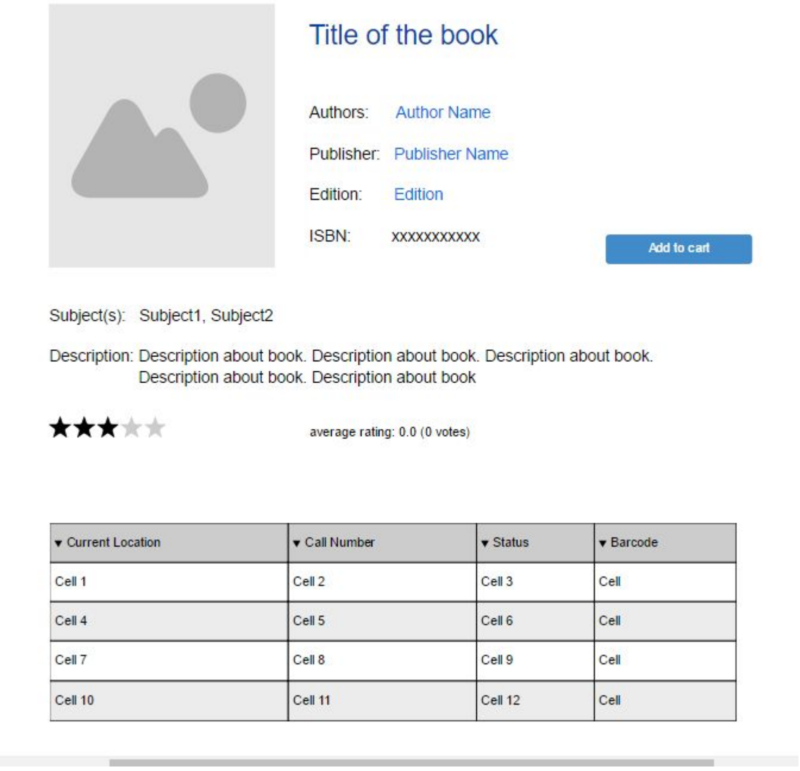
Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

Above window will display with the search results. User can view detailed description of a book by clicking on the book title.



Title of the book

Authors: Author Name

Publisher: Publisher Name

Edition: Edition

ISBN: xxxxxxxxxxx

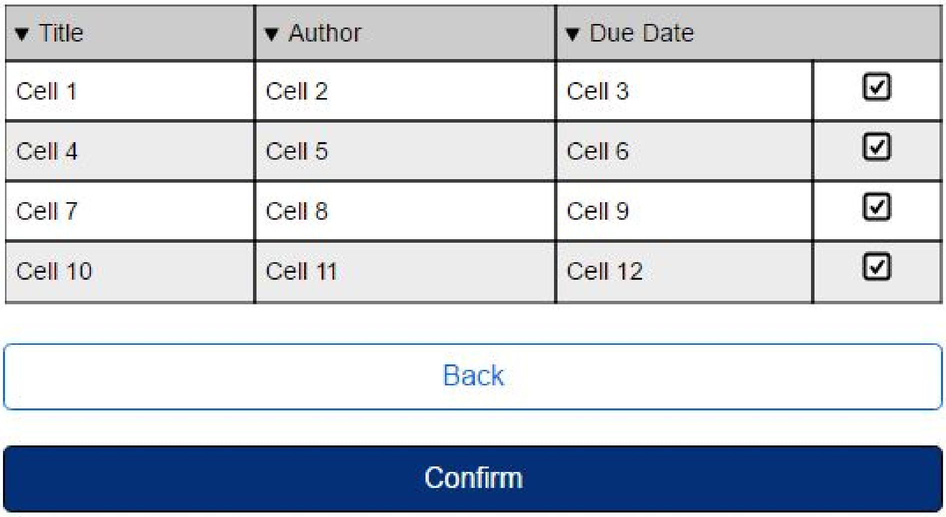
Subject(s): Subject1, Subject2

Description: Description about book. Description about book. Description about book. Description about book. Description about book

★★★☆☆ average rating: 0.0 (0 votes)

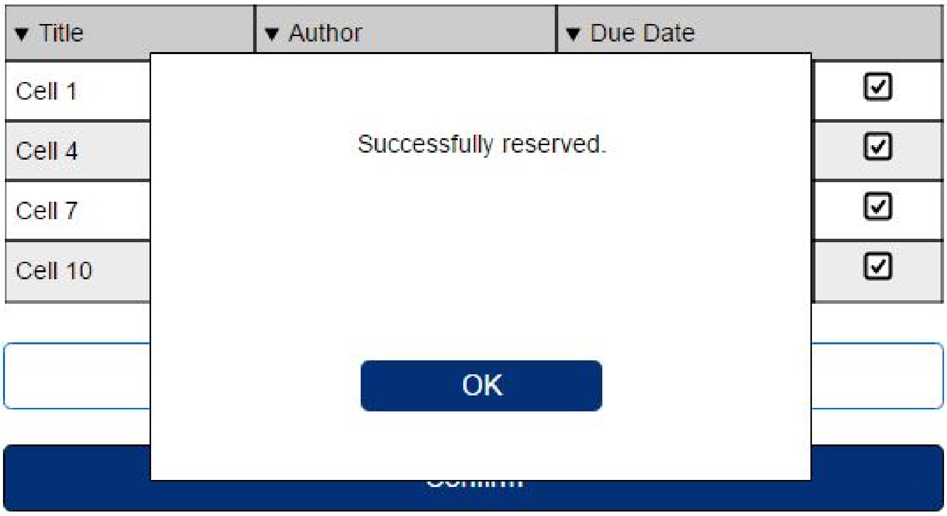
▾ Current Location	▾ Call Number	▾ Status	▾ Barcode
Cell 1	Cell 2	Cell 3	Cell
Cell 4	Cell 5	Cell 6	Cell
Cell 7	Cell 8	Cell 9	Cell
Cell 10	Cell 11	Cell 12	Cell

Above window display the detailed description of the selected book. There he/she can reserve a book by clicking the Add to cart button.



▼ Title	▼ Author	▼ Due Date	
Cell 1	Cell 2	Cell 3	☑
Cell 4	Cell 5	Cell 6	☑
Cell 7	Cell 8	Cell 9	☑
Cell 10	Cell 11	Cell 12	☑

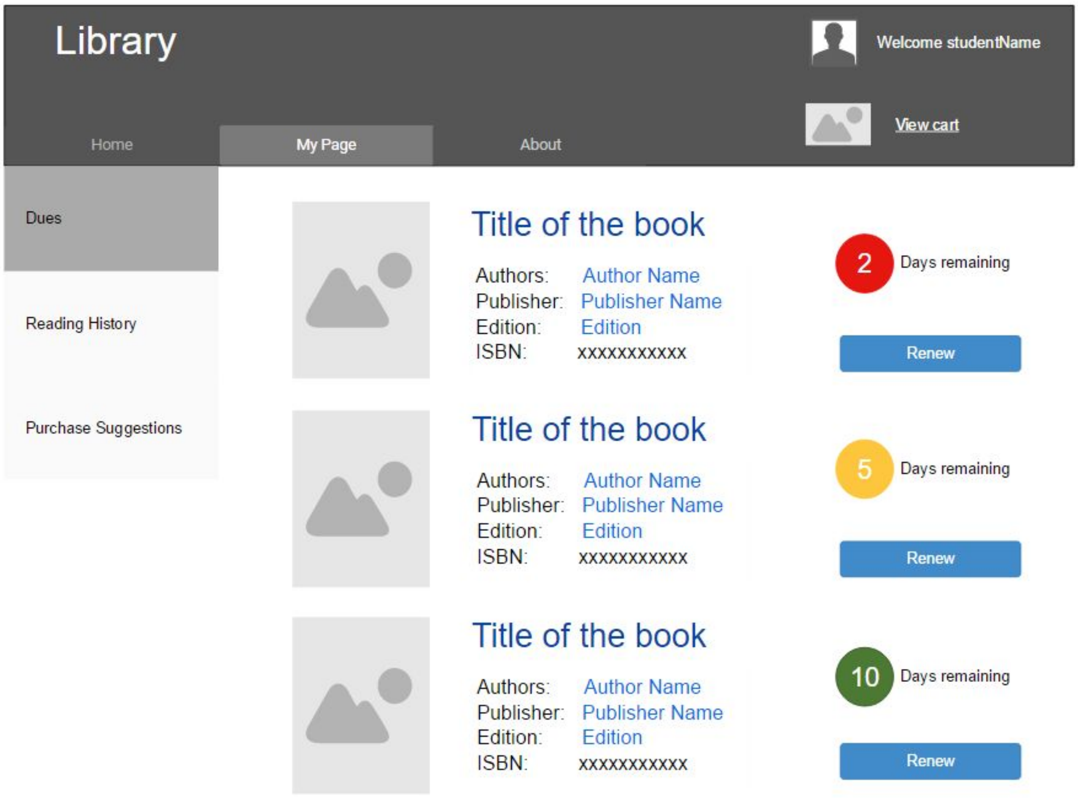
User can view the cart by clicking the view cart link which is displayed below the user avatar.



▼ Title	▼ Author	▼ Due Date	
Cell 1			☑
Cell 4			☑
Cell 7			☑
Cell 10			☑

Successfully reserved.

After clicking on the confirm button above message will display with confirmation details.



Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

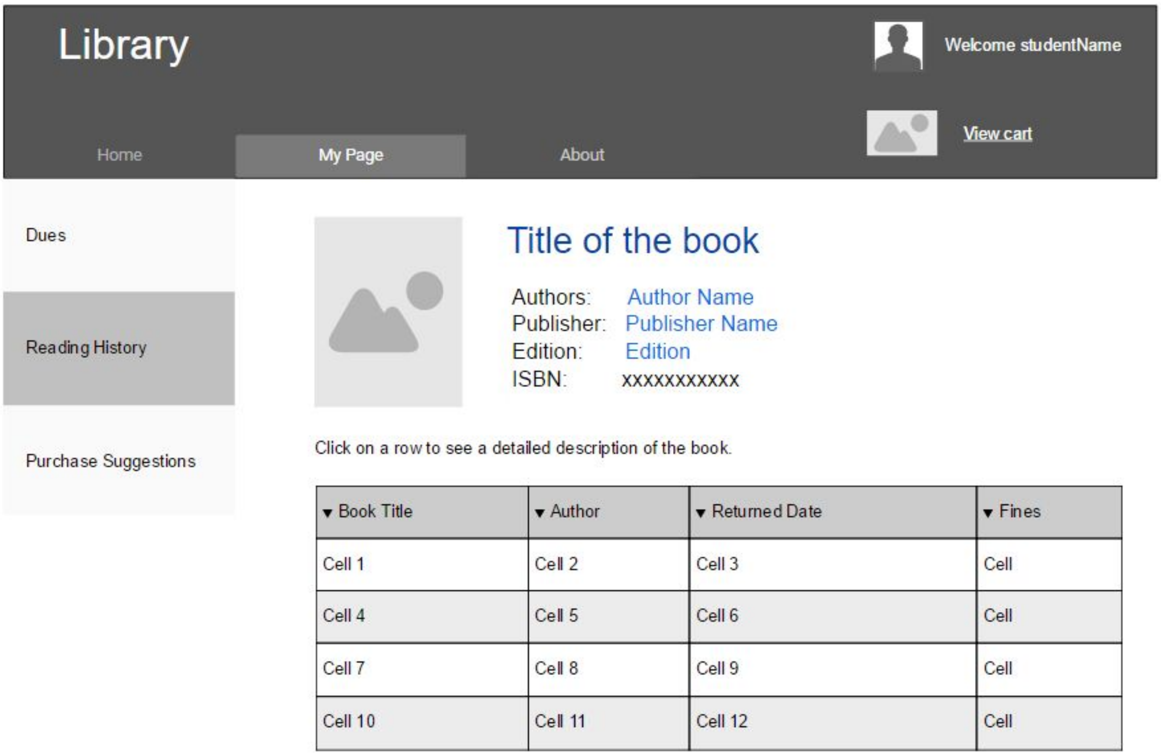
Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: xxxxxxxxxxx

This window display the dues.

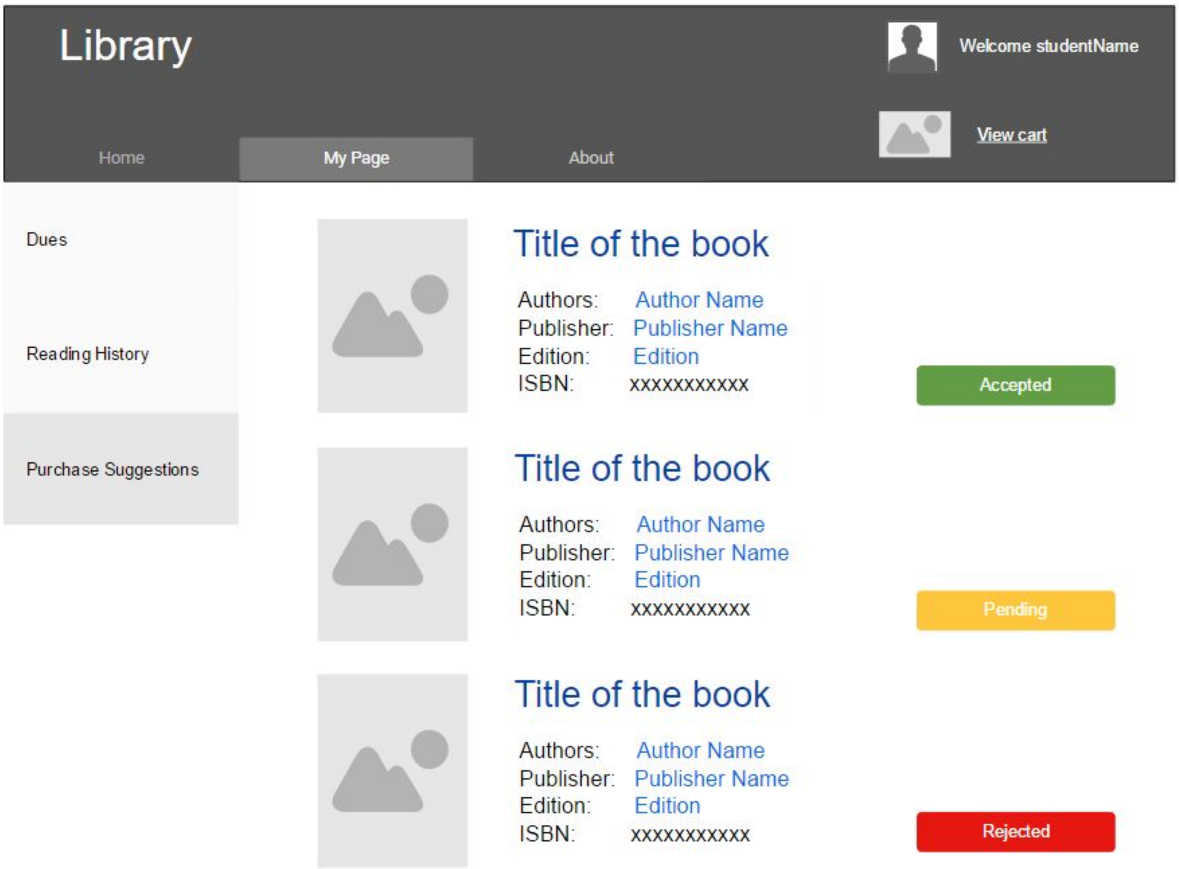


Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: XXXXXXXXXXX

Click on a row to see a detailed description of the book.

▾ Book Title	▾ Author	▾ Returned Date	▾ Fines
Cell 1	Cell 2	Cell 3	Cell
Cell 4	Cell 5	Cell 6	Cell
Cell 7	Cell 8	Cell 9	Cell
Cell 10	Cell 11	Cell 12	Cell



Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: XXXXXXXXXXX

Accepted

Title of the book

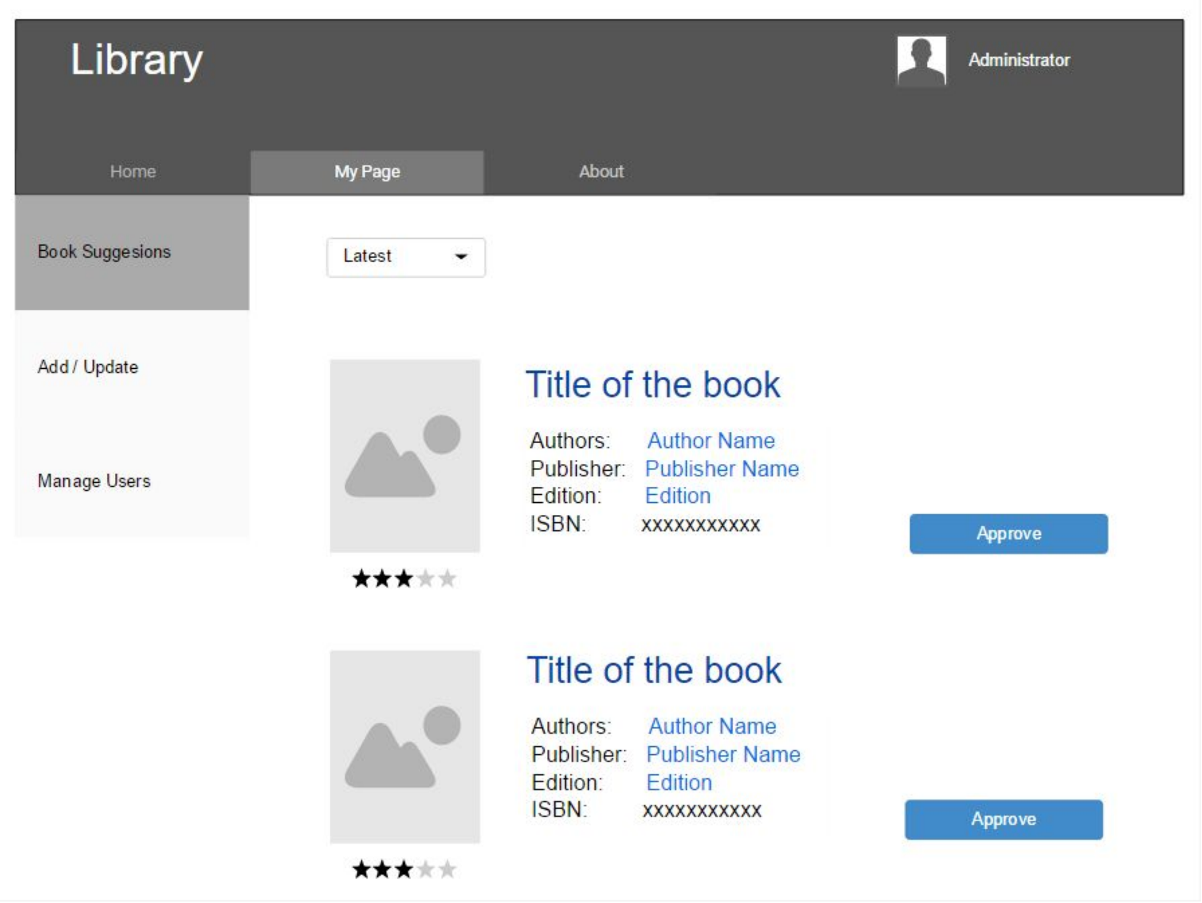
Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: XXXXXXXXXXX

Pending

Title of the book

Authors: Author Name
Publisher: Publisher Name
Edition: Edition
ISBN: XXXXXXXXXXX

Rejected



7. Requirements Traceability Matrix

The following table shows which system components satisfy each of the functional requirements mentioned in the section 3.4 of System Requirement Specification

	Use case Number	**Use case Name**	**Requirement**
Class 1: Student			
signUp()	4.1.1	Sign Up	Must be a student of UOM
signIn()	4.1.2	Sign In	Student must have registered in the system
reserveBook()	4.1.5	Add to cart	Book must available and exist in the LMS
searchBook(title, author, ISBN)	4.1.3	Search books,magazines	At least one of title,author,ISBN need to be entered

			and wildcard can be used.
bookSuggestion()	4.1.7	Request a book from Expographic	View Expographic inventory and Book must not exist in the LMS
Class 2: Admin			
searchBook()	4.1.3	Search books,magazines	At least one of title,author,ISBN need to be entered and wildcard can be used.
issueBook()	4.3.4	Issue a book	Book must available and exist in the LMS
reviewFines()	4.3.2	View Over Dues and blacklist	System must notify overdues
Class 3: Book			
displayBookDetail(book)	4.1.4	Check availability of a book	Book must exist in BIS
updateAvailability(book)	4.2.2	Update book availability, Issue a book, Return a book	Book must exist in the LMS
addNewBook(title, author, ISBN,publisher ,year)	4.3.3	Add new book details	

8. Definitions, Acronyms, and Abbreviations

1) DAO - Data Access Object
2) API - Application Programming Interface
3) ORM - Object Relational Mapping
4) DB - Database
5) LMS - Library Management System
6) BIS- Book Inventory System
7) EBS- Expographic Book Service
8) ESB- Enterprise Service Bus

Notes and Explanations on the SDS

A couple of checklists courtesy of Construx Software Builders, Inc. (modified slightly)
These questions are NOT for answering, they are only to assist you to think on your design…

Architecture

- Is the overall program organization clear, including a good architectural overview and justification?
- Are modules well defined including their functionality and interfaces to other modules?
- Are all the functions that are listed in the requirements covered sensibly, neither by too many nor too few modules?
- Are all major data structures described and justified?
- Are major data structures hidden with access functions?
- Is the database organization and content specified?
- Are all key algorithms described and justified?
- Are all major objects described and justified?
- Is the user interface modularized so that changes in it won't affect the rest of the program?
- Is a strategy for handling user input described?
- Are key aspects of the user interface defined?
- Are memory use estimates and a strategy for memory management described and justified?
- Is a strategy for handling I/O described and justified?
- Is a coherent error-handling strategy included?
- Are error messages managed as a set to present a clean user interface?
- Is a level of robustness specified?
- Are necessary buy vs. build decisions included?
- Is the architecture designed to accommodate likely changes?
- Is any part over- or under-architected?
- Are the major system goals clearly stated?
- Does the complete architecture hang together conceptually?

- Is the top-level design independent of the machine and language that will be used to implement it?
- Are motivations given for all major decisions?
- Are you, as a programmer who will implement the system, comfortable with the architecture?

High-Level Design

- Have you used round-trip design, selecting the best of several attempts rather than the first attempt?
- Is the design of the current subprogram consistent with the design of related subprograms?
- Does the design adequately address issues that were identified and deferred at the architectural level?
- Are you satisfied with the way the program has been decomposed into modules or objects?
- Are you satisfied with the way that modules have been decomposed into routines?
- Are subprogram boundaries well defined?
- Are subprograms designed for minimal interaction with each other?
- Does the design make sense both from the top down and the bottom up?
- Does the design differentiate between the problem-domain component, the user-interface component, the task-management component and the data-management component?
- Is the design intellectually manageable?
- Does the design have low complexity?
- Does the design keep the degree of component coupling as low as possible?
- Does the design keep the degree of component cohesion as high as possible?
- Will the program be easy to maintain?
- Does the design hold connections among subprograms to a minimum?
- Does the design account for future extensions to the program?
- Are subprograms designed so that you can use them in other systems?
- Will the design be easy to port to another environment?
- Is the design lean? Are all of its parts strictly necessary?
- Is the design stratified into layers?
- Does the design use standard techniques and avoid exotic, hard-to-understand elements?