**Front Page**

Index

[Chapte-1: Introduction 3](#_Toc13015033)

[1.1 Problem Statement 3](#_Toc13015034)

[1.2 Proposed Solution 3](#_Toc13015035)

[1.3 Project Structure 3](#_Toc13015036)

[Chapter-2: Related Projects 5](#_Toc13015037)

[2.1 Project-1: 5](#_Toc13015038)

[2.2 Project-2: 5](#_Toc13015039)

[Chapter-3: Analysis 5](#_Toc13015040)

[3.1 Requirement Gather 5](#_Toc13015041)

[3.1.1 Document Reading 5](#_Toc13015042)

[3.1.2 Project Requirement Observations 5](#_Toc13015043)

[3.1.3 Questionaries’ 5](#_Toc13015044)

[3.1.4 Interview 5](#_Toc13015045)

[3.2 Feasibility Study 5](#_Toc13015046)

[3.3 Functional and Non-Function Requirements 5](#_Toc13015047)

[Chapter-4: Diagrams 6](#_Toc13015048)

[4.1 Use Case Diagram 6](#_Toc13015049)

[4.2 Activity Diagram 10](#_Toc13015050)

[4.3 Sequence Diagram 10](#_Toc13015051)

[4.4 Class Diagram 11](#_Toc13015052)

[Chapter-5: Evaluation 11](#_Toc13015053)

[Chapter-6: Conclusion 11](#_Toc13015054)

# Chapte-1: Introduction

## 1.1 Problem Statement

A Library Management System is built to reduce the cost and human resource to maintain a Library and to increase user experience through fast and efficient process. A Library Management System helps to keep the records of whole transactions of the books available in the library. Library Management System makes everyday library tasks more efficient. This means more work can be done in less time. Consequently, this decreases operational costs. This also minimizes paperwork and manual tasks, thus allowing library personnel to concentrate on other things such as interaction with users. The major benefit of Library Management System is, tracking facilities. A library management system is able to track the amount of books in the library, amount of books borrowed by a user, availability of books so easily.

## 1.2 Proposed Solution

Software systems are built to make human tasks easier, efficient securely. So our project is mainly focused on the User Friendly Interaction and Security of the system. As this project is mainly focused on User Friendly Flavor so we tried to keep the User Interface as simple as possible. So any user can access and interact with the system more efficiently than any other system. Our application designed as module based for future development. Which makes our application more loosely coupled than any other systems. This project is created in JAVA, which makes the application platform independent. In 1.3 we have described about the project structure in more detailed.

## 1.3 Project Structure

Our Project is based on JAVA Backend. We have Used Spring Framework for this project. If some small libraries use traditional library management system, they may waste resources into the system. According to the present situation of multi-hierarchical architecture development of information system, we had analyzed thoroughly the spring framework. It integrated the frameworks to design a set of sufficient flexible, loose coupling, expandable Library Management System.

Through the Spring Framework We have separated the application into several layer. Below a diagram for the Project Structure is given:



**Fig 1.3.a: Project Structure Overview.**

# Chapter-2: Related Projects

## 2.1 Project-1:

## 2.2 Project-2:

# Chapter-3: Analysis

## 3.1 Requirement Gather

### 3.1.1 Document Reading

### 3.1.2 Project Requirement Observations

Requirement Observations are given below:

1. Multi User Application

2. Role Base User Access

3. Export Book Information in CSV

4. Import Book Information from CSV

5. SSL Based Communication

6. Store secure information as encrypted

7. Student/Teacher Account Activation/De-Activation

8. CRUD Operations over books information’s

9. CRUD Operations over Users information’s

10. Bulk Delete Operation for Books.

11. Bulk Delete Operation for Users.

### 3.1.3 Questionaries’

### 3.1.4 Interview

## 3.2 Feasibility Study

## 3.3 Functional and Non-Function Requirements

# Chapter-4: Diagrams

## 4.1 Use Case Diagram

**Fig 4.1.a: Library Management System Use Case Diagram.**

Use Case Steps Description:

|  |  |
| --- | --- |
| Use Case: | Login |
| Introduction : | To interact with the system, The application shall validate provided credentials with this system. It also defines the actions a user can perform based on User Role. |
| Actors: | Librarian/Admin  Student/Teacher/User |
| Pre-Conditions: | User must have a compatible browser to access the system through web. |
| Post-Conditions: | User should be redirected to the user dashboard page |
| Basic Flow: | 1. Show Login Page 2. Enter User Id and Password 3. User Credentials Validation Successful 4. Redirect to User Dashboard |
| Alternative Flow: | 1. Show Login Page 2. Enter User Id and Password 3. User Credentials Validation Un-successful 4. Redirect to Login Page |
| Special Requirement: | User Must be a Registered User |

|  |  |
| --- | --- |
| Use Case: | View/Search Users |
| Introduction : | This use case enable a user to view users list or search particular user matching the pattern of the search text. |
| Actors: | Admin/Librarian |
| Pre-Conditions: | 1. User Must be Logged In 2. User must have Admin Role |
| Post-Conditions: | 1. Show All Users list 2. Show users that have been matched with the provided search texts. |
| Basic Flow: | 1. Show All Users in a list with details of these users. |
| Alternative Flow: | 1. Show specific users whose username is matched with the search pattern. |
| Special Requirement: | User Must Have Admin Role |

|  |  |
| --- | --- |
| Use Case: | Add, Delete & Modify Users |
| Introduction : | This use case permits a user to add, delete or modify any Users. |
| Actors: | Admin/Librarian |
| Pre-Conditions: | 1. User Must be Logged In 2. User must have Admin Role |
| Post-Conditions: | 1. Shows a form to add new user if user clicks into the add button. 2. Removes a user if any user is selected and clicked on Delete Button. 3. Shows a form with existing data of a user and let’s to modify the contents. |
| Basic Flow: | Basic CRUD operations. |
| Alternative Flow: | If any invalid data is submitted into the add/modify form then the submission will be rejected with an error message. |
| Special Requirement | User Must Have Admin Role |

|  |  |
| --- | --- |
| Use Case: | Add, Delete & Modify Books |
| Introduction : | This use case permits a user to add, delete or modify any Books. |
| Actors: | Admin/Librarian |
| Pre-Conditions: | 1. User Must be Logged In 2. User must have Admin Role |
| Post-Conditions: |  |
| Basic Flow: |  |
| Alternative Flow: | If any invalid data is submitted into the add/modify form then the submission will be rejected with an error message. |
| Special Requirement | User Must Have Admin Role |

|  |  |
| --- | --- |
| Use Case: | Issue Book |
| Introduction : | By this use case a user is able to issue a book to someone. |
| Actors: | Admin/Librarian |
| Pre-Conditions: | 1. User Must be Logged In 2. User must have Admin Role |
| Post-Conditions: |  |
| Basic Flow: |  |
| Alternative Flow: |  |
| Special Requirement |  |

|  |  |
| --- | --- |
| Use Case: | Return Book |
| Introduction : | This use case stores the data |
| Actors: | Admin/Librarian |
| Pre-Conditions: | 1. User Must be Logged In 2. User must have Admin Role |
| Post-Conditions: |  |
| Basic Flow: |  |
| Alternative Flow: |  |
| Special Requirement |  |

|  |  |
| --- | --- |
| Use Case: | View Book |
| Introduction : |  |
| Actors: | Librarian/Admin  Student/Teacher/User |
| Pre-Conditions: |  |
| Post-Conditions: |  |
| Basic Flow: |  |
| Alternative Flow: |  |
| Special Requirement |  |

|  |  |
| --- | --- |
| Use Case: | Request Book |
| Introduction : |  |
| Actors: | Student/Teacher/User |
| Pre-Conditions: |  |
| Post-Conditions: |  |
| Basic Flow: |  |
| Alternative Flow: |  |
| Special Requirement |  |

## 4.2 Activity Diagram

## 4.3 Sequence Diagram

## 4.4 Class Diagram

**Fig 4.4.a: Library Management Class Diagram**

# Chapter-5: Evaluation

# Chapter-6: Conclusion