**Project Name:**

**LIBRARY MANAGEMENT SYSTEM PROJECT**

**Date:30-12-2022**

Copyright Notice

This document contains proprietary information of HCL Technologies Ltd. No part of this document may be reproduced, stored, copied, or transmitted in any form or by means of electronic, mechanical, photocopying or otherwise, without the express consent of HCL Technologies. This document is intended for internal circulation only and not meant for external circulation.

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version No | Date | Prepared By | Significant changes |
|  | 30-12-2022 | Avusodapu Ashritha | - |
|  |  | Bhateri | - |
|  |  | Ch.Charan | - |
|  |  | S.Jaya | - |
|  |  | Muskan Hayath | - |

**Table of Contents**

[1.1 Project Overview 3](#_Toc76987238)

[1.2 Scope 3](#_Toc76987239)

[1.3 Out of Scope 3](#_Toc76987240)

[1.4 Intended Audience 3](#_Toc76987241)

[1.5 High Level Use Cases 3](#_Toc76987242)

[1.6 Use Cases detailed 3](#_Toc76987243)

[1.7 User Interface Modules 3](#_Toc76987244)

[1.8 Technical Architecture 3](#_Toc76987245)

[1.9 Technical Detailed Description 3](#_Toc76987246)

[1.10 Sequence Diagram 3](#_Toc76987247)

[1.11 Database Entity Relationship Diagram 3](#_Toc76987248)

[1.12 Standards 3](#_Toc76987249)

[1.13 Non Functional Requirements 3](#_Toc76987250)

**Project Overview**:

The title of the project is “Library Management System”. The activities and functioning of library will be managed by this project. Library management encompasses normal management tasks as well as returning books, feedback, and issuing books. It is used by the librarian/Admin/Student to categorically manage the library by the virtue of using a computerized system where he/she can record various transactions like issue of books, return of books, addition of new books, addition of new students etc. It facilitates of recording information regarding issuing book status, returning, all the input information given by the end user.

The processes and approaches that will be employed throughout the lifecycle of coding are described in this project.

**Scope** :-

|  |  |  |
| --- | --- | --- |
| **Modules** | **Roles and application** | **Description** |
| Registration  Login | User/Admin | If there is a new user who wants to visit the library application can register by providing username, password, Mobile Number, Address. If the admin wants to visit the application then he/she should enter username and password. |
| Admin | Admin | The admin who has already registered in the application can perform various activities like they can add another user, add book, can log out from admin page and can search for books, proceed with the payment and provide feedback on books |
| Student | Student | The student can enroll their details, can enroll book details, they can search for books, authors, books can be issued to them, they can return the book, can also view the history of status of books |
| Search | Admin | The admin who is registered can search for books, can view the search history also can clear the viewed history |
| Wish List | Admin | The admin can add their favorite books to wish list, can remove from it and also can provide feedback. |
| Payment | Payment | The admin can buy a book by choosing available payment options. |

**Out of Scope:** -

These features are not to be tested because they are not included in the Software Requirement Specification.

* User interface
* Hardware Interface
* Software Interface

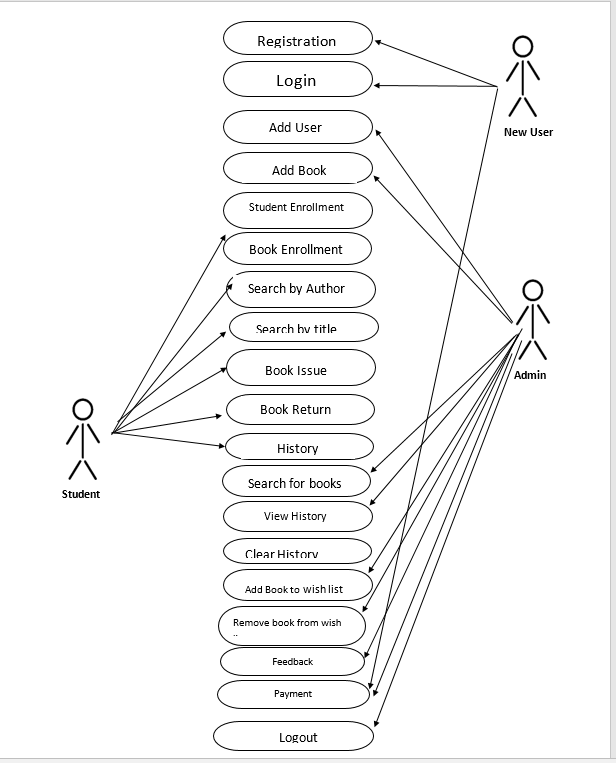
**Intended Audience**: -

* Admin
* Student

In this Library management system, it is accessible to all users irrespective of their age but they need to have and should give valid credentials. The main objective of the project is store all the data, credentials and display the history of the books to the user with the help of full-fledged computer software, maintaining up with the requirements so that the data can be stored for ages and can be accessible. On the contrary, the project essentially outlines how to manage for improved services and performance.

High Level Use Case:

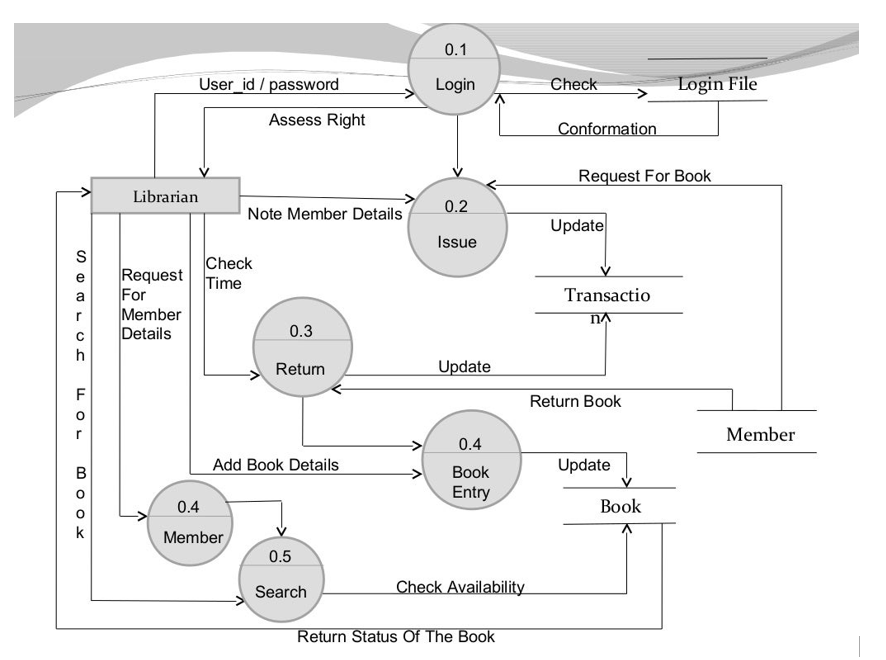
UML use case diagram of Library Management System



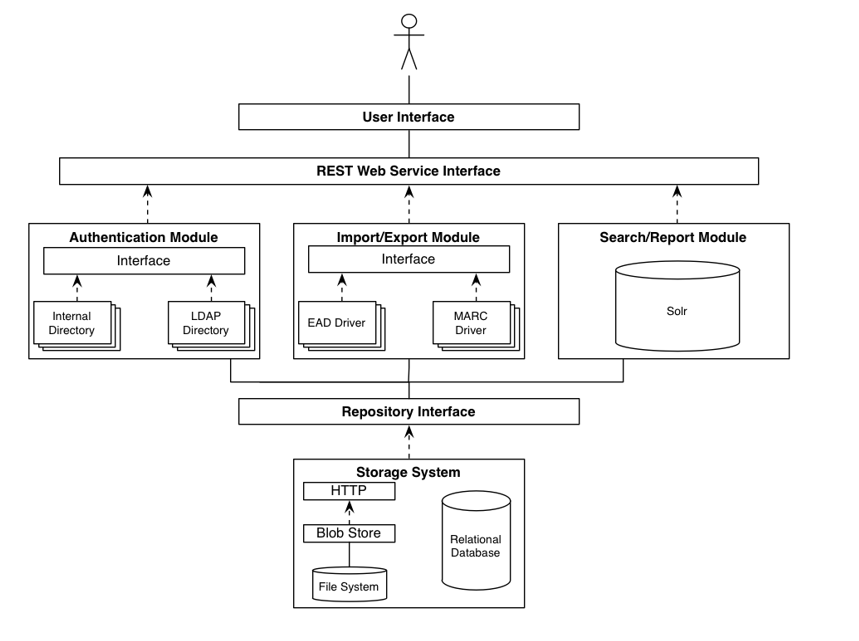
Use Cases detailed: -

* **Registration**: Registration is a functionality where a new user can enroll his/her credentials like Username, Password, Mobile Number, Address and can become as admin.
* **Login**: According to this use case, the system's login feature serves to verify user identity. So, the admin should provide valid credentials.
* **Add User**: This use case defines that admin can add another user by providing valid credentials like user id, name, gender, mobile number, book category.
* **Add Book**: This use case defines that admin can book by providing valid credentials like book name, category, author name, language, publisher’s name.
* **Student Enrollment**: This use case defines that student will be able to add his/her details by providing valid data like roll number, name, department.
* **Book Enrollment**: This use case defines that student will be able to add the book credentials by providing valid data like book id, book title, author of book, availability.
* **Search by author**: This use case defines that student will be able to search by author’s name of first 3 characters.
* **Search by title**: This use case defines that student will be able to search by book title’s name of first 3 characters.
* **Book Issue**: This use case defines that the book can be issued to the student by entering data like roll number, book id.
* **Book Return**: This use case defines that the book can be returned to the student by entering data like roll number, book id.
* **History**: This use case defines that the student can view the history of the book status by entering roll number.
* **Search for books**: This use case defines that the admin can search for books.
* **View History**: This use case defines that the admin can view the history of books.
* **Clear History**: This use case defines that the admin can clear the history by entering choices.
* **Add to Wish list**: This use case defines that the admin will be able to add book details by providing credentials like book name, author’s name.
* **Remove to wish list**: This use case defines that the admin will be able to remove book details by providing credentials like book name, author’s name.
* **Feedback on book**: This use case defines that the admin can give feedback on the respective book by entering book name.
* **Payment**: This use case defines that the admin can pay for the respective books by choosing the payment method.
* **Logout**: This use case defines that the admin can logout by choosing respective option.

**User Interface Modules:**



**Technical Architecture: -**



Registration:

If the new user wants to become admin, they can register themselves by providing username, password, Mobile Number, Address.

Login:

If the admin wants to login to the application, then he/she can enter valid credentials and login to the page which navigates to admin page.

Admin: If the admin wants to add new user, add new book then he/she can provide valid credentials of the choice selected functionality also as admin has already registered they can search for books, clear the history, view the history of books, can also add favorite books to wish list, can also remove books from wish list, can give feedback on respective book and admin can also proceed with the payment for books and can logout if they want to.

Student:

The student sub field which has various functionalities like student can enroll their details, book details, can search for book, search for author, book issuance can be done, book can be returned, can also view the history of status of books.

Search:

The admin can search for books, can view the history also can clear the history

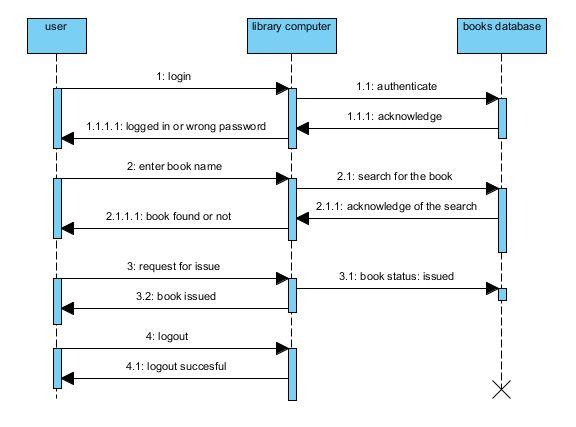
Wish List:

The admin can add his/her favorite books to wish list, can remove them from wish list and can also give feedback on books.

Payment:

The admin can proceed with the payment with various online payment methods by providing valid credentials.

Sequence Diagram:



**Database Entity Relationship Diagram:** -This School Management System ER Diagram example depicts the relationship between elements such as admin, teacher, parent and student information, as well as how they are related.



**Standards**

* Class names start with an upper case letter.
* The comment should include a description of the method, the name and description of each parameter, a description of the return value, and the name and description of any exceptions thrown within the method using Javadoc keywords and formatting.
* Detailed description line by line:
* Begins with a slash, followed by two asterisks (/\*\*)
* One asterisk (\*), followed by @parameter, followed by the name of the parameter, followed by a description of the parameter (Omit if there are no parameters. Use one line for each parameter, so two parameters will have two lines. 3 parameters will have 3 lines. Etc.)
* One asterisk (\*), followed by return, followed by a description of the return variable (Omit if the return value is void.)
* One asterisk (\*), followed by @exception, followed by the class of the exception, followed by a description of when the exception is thrown. Only checked exceptions are required to be listed. Unchecked exceptions are not required to be listed. Omit exceptions if there are no exceptions that are thrown. (In other words, if you catch the exception within the method, you do not need to list the exception.) Use one line for each exception, so two exceptions will have two lines. 3 exceptions will have 3 lines. Etc.
* Ends with one asterisk, followed by one slash (\*/)

**NON-FUNCTIONAL REQUIRMENTS**

1.Usability

**Usability** is the main non-functional requirement for a library management system. The UI should be simple enough for everyone to understand and get the relevant information without any special training. Different languages can be provided based on the requirements.

2.Accuracy

**Accuracy** is another important non-functional requirement for the library management system. The data stored about the books and the fines calculated should be correct, consistent, and reliable.

3.Maintainability

**Maintainability**  
The software should be easily maintainable and adding new features and making changes to the software must be as simple as possible. In addition to this, the software must also be portable.

•Reliability

**Reliability Requirement**

The system has to be 100% reliable due to the importance of data and the damages that can because by incorrect or incomplete data. The system will run 7 days a week, 24 hours a day.