**Software Requirement Specifications(SRS) Document**

**Technical Story Card**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version No.** | **Date** | **Prepared by / Modified by** | **Significant Changes** |
| **1** | **8 Sept 2022** | **Ashini**  **Anuradha**  **Lithila**  **Sakith**  **Mohammed** | **Report completion** |

**Table of Contents**

[**1**](#_gjdgxs) **Introduction 3**

[**2**](#_44sinio) **Scope of Change 3**

[**3**](#_2jxsxqh) **List of impacted modules 3**

[**4**](#_z337ya) **Design and Detailed technical updates 4**

[4.1](#_3j2qqm3) Process model 4

[4.1.1](#_tyjcwt) Use case Model 4

[4.1.2](#_1t3h5sf) Sequence diagram 4

[4.2](#_1y810tw) Proposed user Interface design 4

[4.3](#_4i7ojhp) Database design changes 4

[4.4](#_4d34og8) Refactoring related changes 4

[4.5](#_2xcytpi) Construction strategy and re-use 4

[**5**](#_1ci93xb) **Details of Alternative Design Approach 4**

[**6**](#_3whwml4) **Other Technical changes 5**

[6.1](#_2bn6wsx) Automation tasks/changes 5

[6.2](#_17dp8vu) CI / Build relates tasks/changes 5

[6.3](#_3rdcrjn) Non-functional related changes 5

[**7**](#_26in1rg) **Additional details 5**

[7.1](#_lnxbz9) Open Questions/clarifications / Assumptions 5

[7.2](#_35nkun2) Additional notes to technical team 5

[**8**](#_1ksv4uv) **References 5**

1. **Introduction**

Everyone has a few books, and some people may have many. It would be helpful to have a software program to manage them as one's collection grows. It could create reports on the books already in the collection, help the user locate specific volumes from the shelf fast, and so on. People also want an effective method of managing these electronic papers since it may be quite challenging to locate the particular document needed in a collection of hundreds or thousands of electronic documents because the file names for these documents sometimes have no significance. Building a feature-rich, high-quality software program to assist people in managing their books is the objective of this project.

1. **Scope of Change**

The suggested book management system is a database tool that aids in book cataloging. Based on the ISBN or other information supplied, it should be able to get book information from the Internet, such as the book title, author, and cover images, reducing the amount of input required from the users. A typical user should have little trouble utilizing the product without extra training since it should have a nice user interface.

1. **List of impacted modules**

**System Interface**

The system will utilize HTTP to get information about books from the Internet; the user will then review the information that has been retrieved and provide new information before the system records everything in the database.

**User Interface**

The system should have an intuitive user interface with commonly used GUI components including menus, toolbars, dialog boxes, shortcut keys, etc. A multi-language user interface architecture should also be enabled so that users from all over the world can select the languages they prefer.

**Software Interface**

The interface between the program and the backend database is the main software interface in this system. For all database activities, they will communicate via Standard Query Language.

**Communication Interface**

Hypertext Transfer Protocol (HTTP) communication interfaces between this system and distant Internet servers will be built.

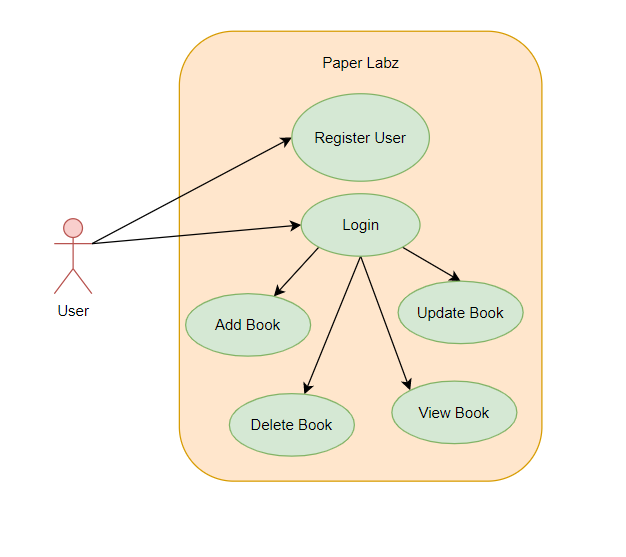
**Memory Constraints**

There are no explicit memory restrictions on the system. Any personal computer that is capable of running the operating system itself should be able to run it quickly enough.

1. **Design and Detailed technical updates**

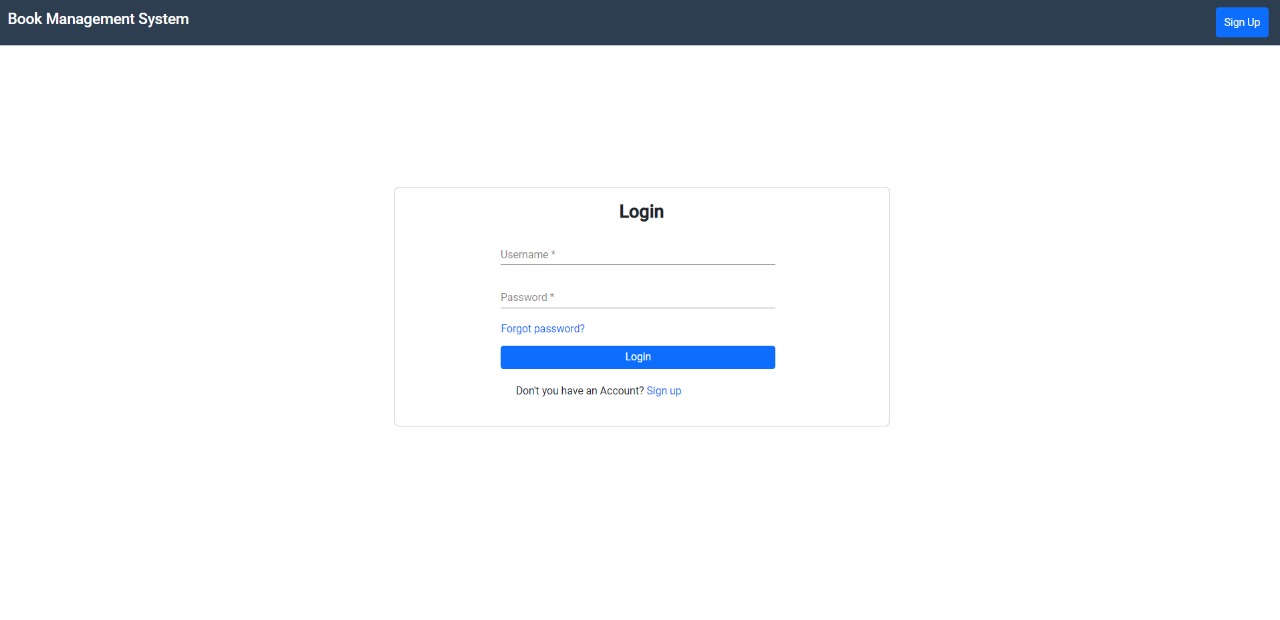
**Process Model**

**User Case Model**



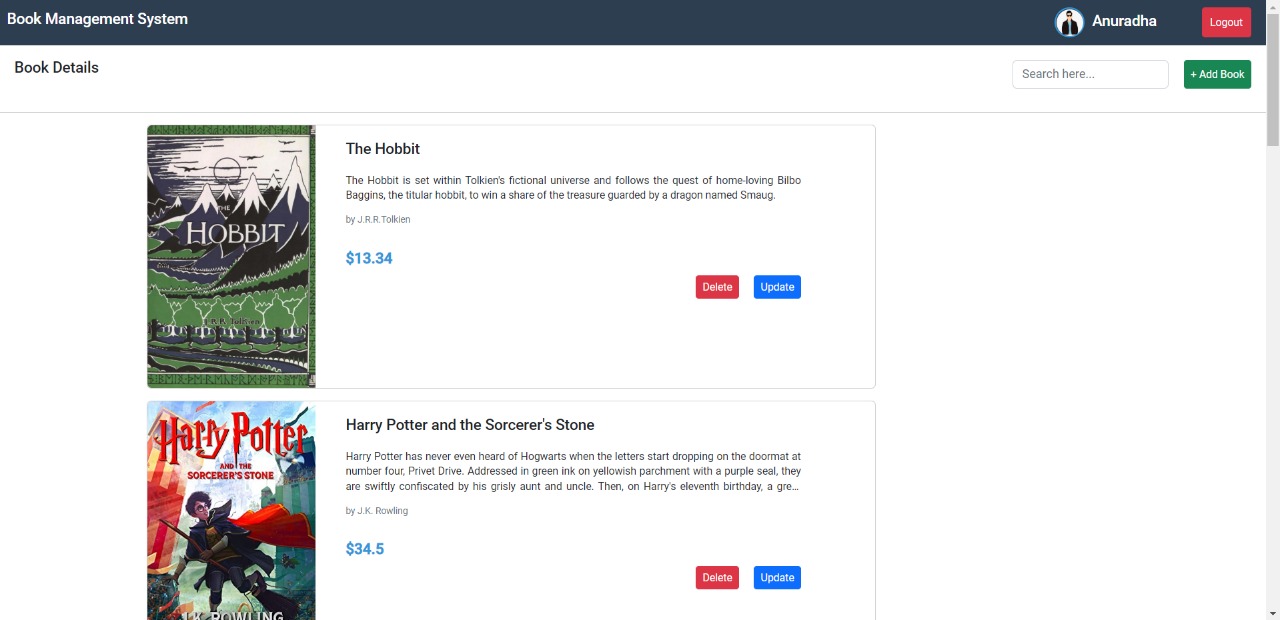
**Login Page**

|  |  |
| --- | --- |
| Brief Description | **Login** |
| Basic Flow | This use case describes how a user login into the system   1. The user has to register himself into the system. 2. After the successful login, users will be taken to the appropriate landing page. 3. The following information is required to login.  * Username * Password |
| Alternate Flow | 1. The system will validate the credentials provided. If credentials are invalid, the login form will be redirected again with an error message. |
| Validation | 1. Valid Username 2. Valid Password |
| Pre-Conditions | Users should have network access and a Browser with the latest updates. |
| Post-Conditions | Landing page has to be displayed. |



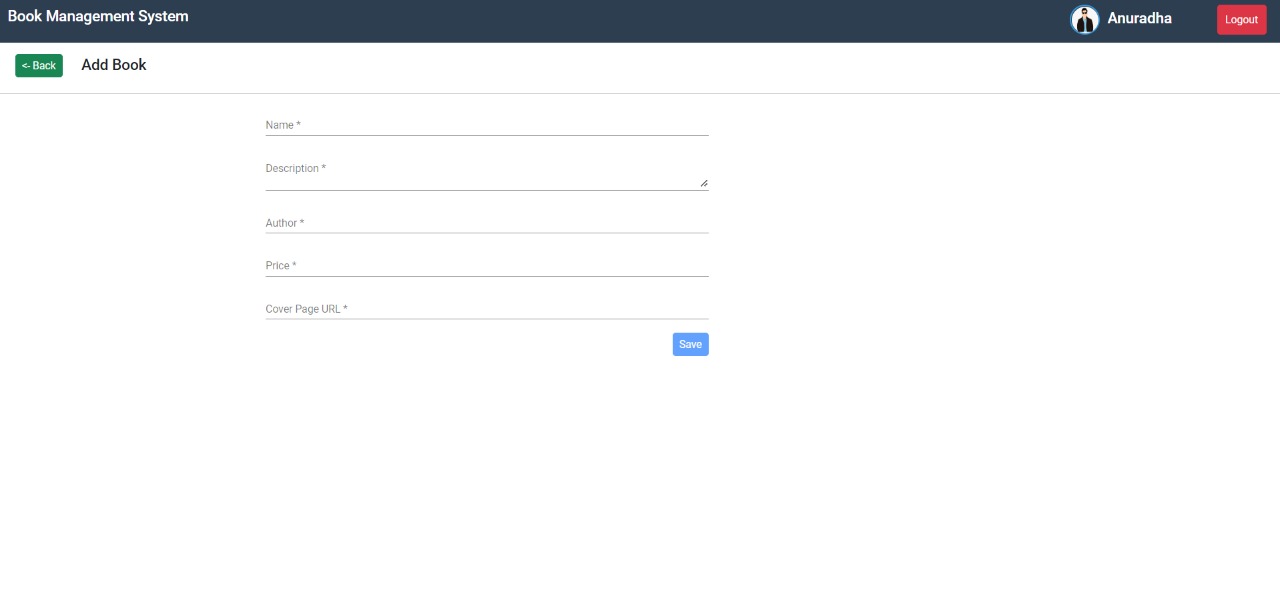
**Book Details Page**

|  |  |
| --- | --- |
| Brief Description | **Book Details** |
| Basic Flow | This use case describes how book details show in the system. |
| Alternate Flow | 1. The system will validate the information provided. If any invalid data is found, the input form will be redirected with an error message. |
| Validation |  |
| Pre-Conditions | Users should have already logged into the system. |
| Post-Conditions | Success message should be shown. |



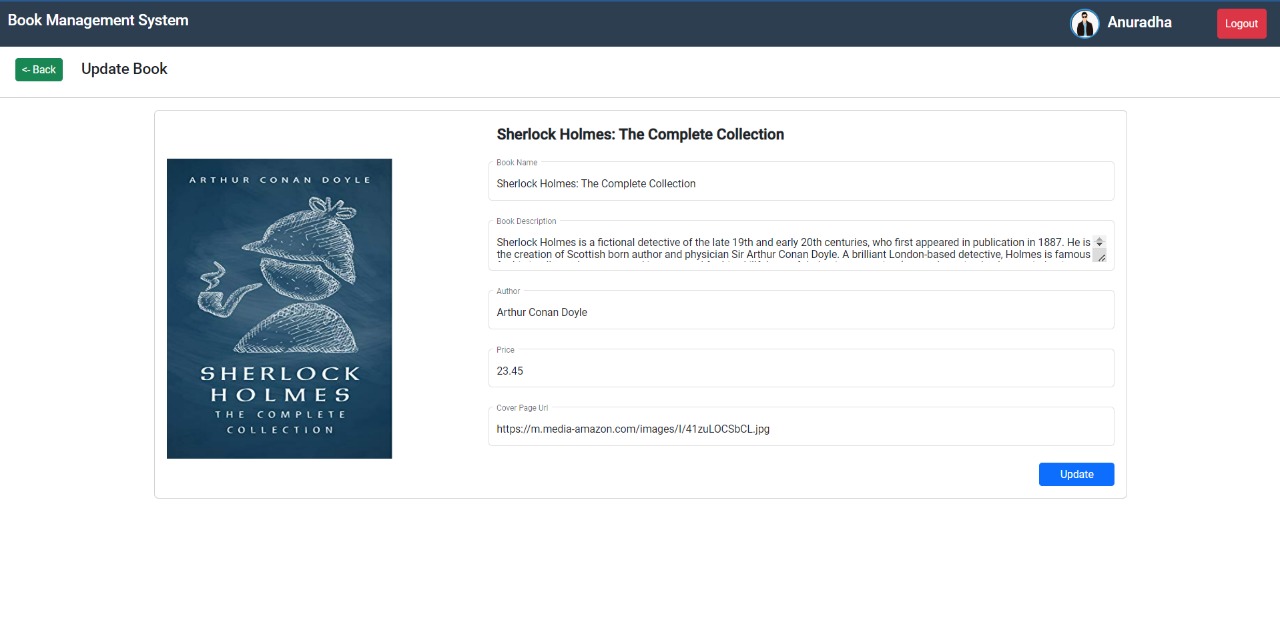
**Add Book Page**

|  |  |
| --- | --- |
| Brief Description | **Add Book** |
| Basic Flow | This use case describes how a user can add a book to the system.   1. The user has to login into the system. 2. The following information has to be displayed.  * Name * Description * Author * Price * Cover Page URL |
| Alternate Flow | Not Applicable |
| Validation | Not Applicable |
| Pre-Conditions | Users should have already logged into the system. |
| Post-Conditions | Success message should be shown. |



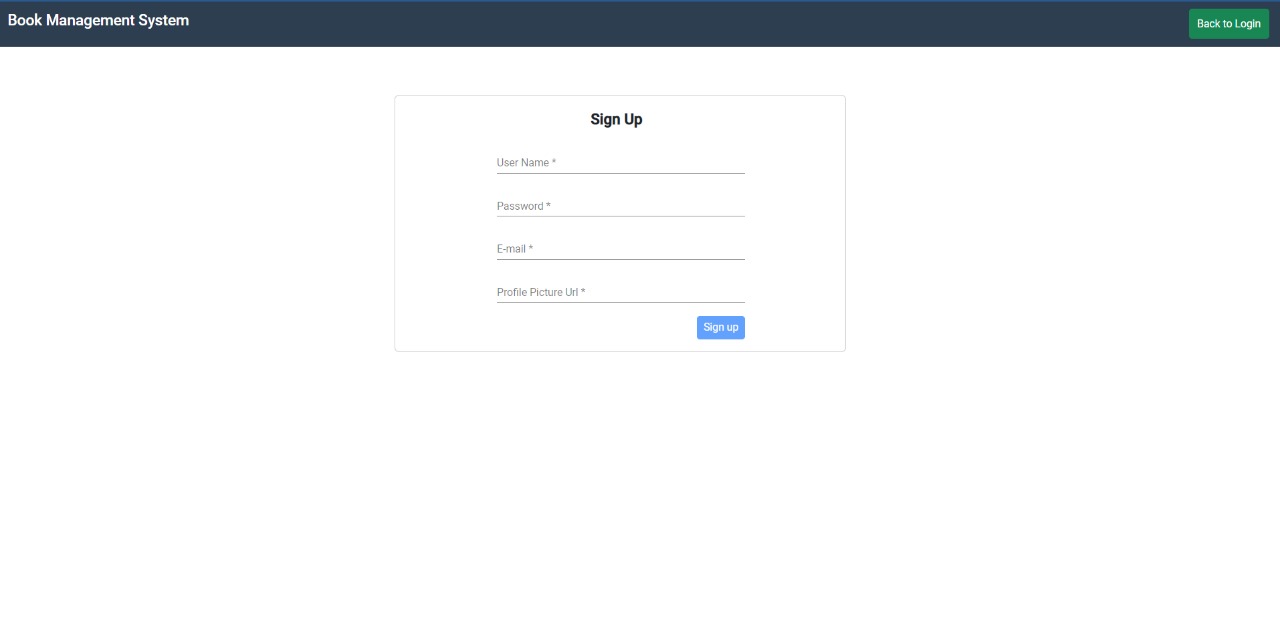
**Update Book Page**

|  |  |
| --- | --- |
| Brief Description | **Update Book** |
| Basic Flow | This use case describes how a user can update details of books.   1. The supplier has to login into the system. 2. The following information has to be displayed.  * Book Name * Book Description * Author * Price * Cover Page URL.  1. Details have to be updated. |
| Alternate Flow | Not Applicable |
| Validation | Not Applicable |
| Pre-Conditions | Users should have already logged into the system. |
| Post-Conditions | Success message should be shown. |



**Signup Page**

|  |  |
| --- | --- |
| Brief Description | **Sign Up** |
| Basic Flow | 1. This use case describes how a sign-up into the system 2. The user has to register himself into the system. 3. The following information is required to sign up.  * Username * Password * Email * Profile Picture URL |
| Alternate Flow | 1. The system will validate the credentials provided. If credentials are invalid, the login form will be redirected again with an error message. |
| Validation | 1. Valid Username 2. Valid Password 3. Required email must be a valid email type |
| Pre-Conditions | Not Applicable |
| Post-Conditions | Success message should be shown. |

**Architecture Diagram**



**Logical View**

**Technology/ Framework**

**Layer**



UI Components

Presentation Layer

Validations

HTML/ CSS/ JavaScript/ AJAX/ Angular

Resources

Report



Web-Server Classes

Java 1.8/ Spring MVC

Application/ Business Layer

Controllers

Configuration

REST Controllers

Service

Entity/ Model Classes



Data Access Layer

Hibernate ORM

Repository classes

H2

Database

**Presentation Layer:**

This layer consists of all UI components.

* Validation component: All basic level data validations should be done at UI level.
* Resources: All HTML/CSS/ images, which are required for the page design.

**Application Layer:**

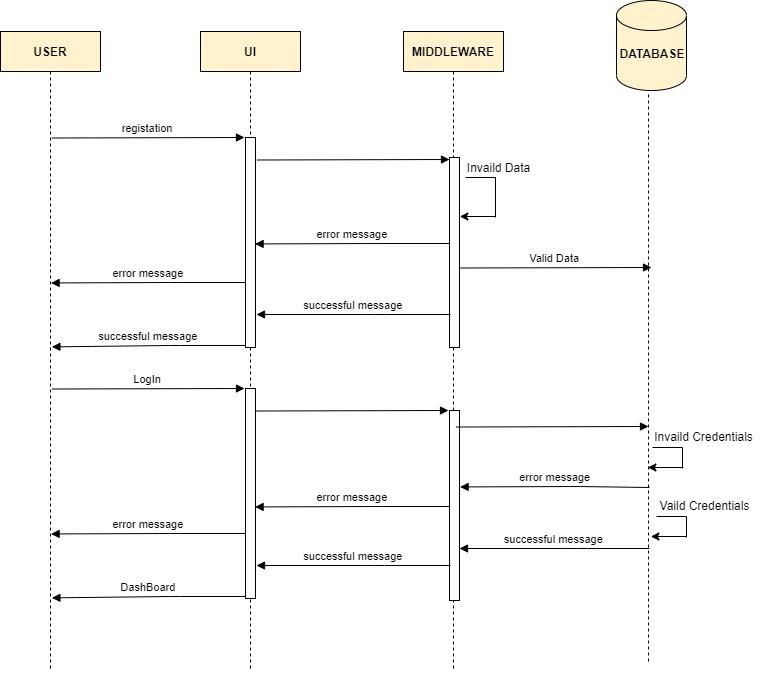
This layer comprises all server and business classes.

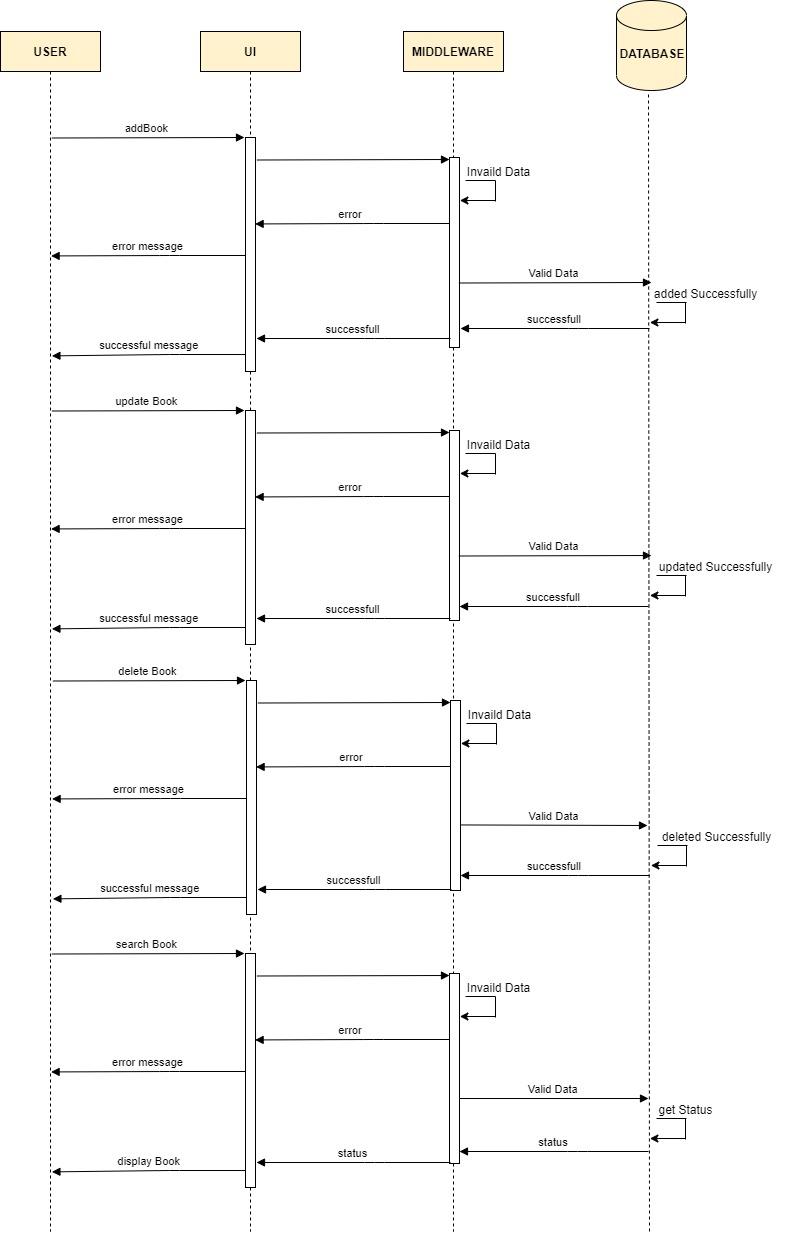
* Configuration settings (XML or Class based) will define the application and server configuration.
* Spring Controllers define the server classes, which are required to process the incoming http requests.
* Service classes are required to perform the required business services.
* REST Controllers to process the HTTP AJAX requests.
* Model classes are used to define the functions of the entities present in the system.

**Data Layer:**

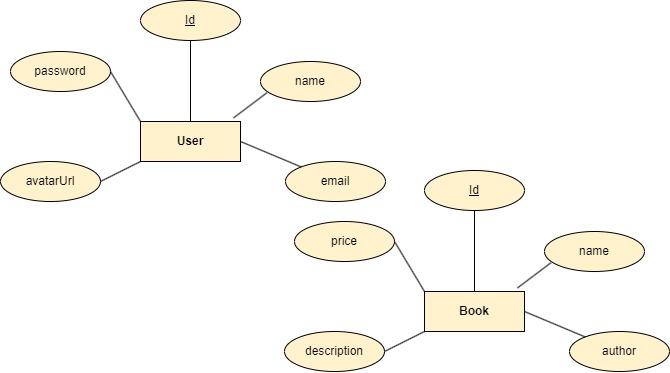
Data layer is implemented through Hibernate ORM. It will contain the repository classes, which provide an interface to the table.

**Sequence Diagram**

****

****

**ER-Diagram**



1. **Other Technical changes**

## CI / Build relates tasks

N/A

## Non-functional related changes

N/A

1. **Additional details**

## Open Questions/clarifications / Assumptions

N/A

## Additional notes to technical team

N/A

1. **References**

* Codevolution.(2020,June 08).Angular Authentication Tutorial [Playlist]. YouTube.https://www.youtube.com/playlist?list=PLC3y8-rFHvwg2RBz6UplKTGIXREj9dV0G.