Project Name: Library Management

Date: 17/01/2023

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 distribution.

Revision History

| Version No | Date | Prepared by / Modified by | Significant Changes |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Table Of Contents**

[1.1 Project Overview 3](#_Toc76987238)

[1.2 Scope 3](#_Toc76987239)

[1.3 Out of Scope 3](#_Toc76987240)

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

[1.5 Use Cases detailed 3](#_Toc76987243)

[1.6 Technical Detailed Description 3](#_Toc76987246)

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

[1.8 Standards 3](#_Toc76987249)

## Project Overview

This project aims to create Library Management system by using Java code. It will involve Librarian to Log in and Log out and also allow Librarian to add books and it can maintain the records of issue and return of books. The project will have the following objectives: To Reducing costs of maintaining books, to consume less time to keep records of customers, to increases the revenue and to deliver the quality product and services. The project scope includes Log in, Log out add books, issue of books and return of books. The project will deliver the following: A new project for library management.

## Scope

The scope of this project includes the following:

* A project to develop a new software application might have a scope that includes designing and building the application, testing it, and delivering it to the customer.

## Out of Scope

The following elements are outside of the scope of this project:

Non-functional testing like stress, performance is beyond scope of this project.

Automation testing is beyond scope.

External interfaces are in out of the scope and no need to be tested

The system can run on all browsers.

## High Level Use Cases

A user named venkat wants to log in to the library management system website. He enters into system and enter the username and passwords and successfully log in into system. The he tries to add the book by entering details of the book. The system accepts the data and create and store the data in database, it successfully adds the book details in database. The system collects the data and allow venkat perform to add book or he can exit the system.

## Use Cases detailed

1. A librarian venkat wants to log in to his library management system website. He enters his username and password and the system verifies the combination. The system grants access to venkat to perform functionalities in system.
2. After that librarian wants to perform Add books and system ask him to give Book-ID and Book Name he enters the details and system add book successfully.
3. After that librarian wants to see the books. So he wants to choose View books and system ask him to give Book-ID and Book Name he enters the details and system display the books available in database successfully
4. After that librarian wants to issue the books and system ask him to give Book-ID and Book Name and also name of the person who is going to take he enters the details and system Notes the given date and book details successfully
5. After that librarian wants to view the issue books and system and the system display the issued books with ID and name including with the given date.
6. After that librarian wants to perform recollect the books and system ask him to give Book-ID and Book Name he enters the details and system displays the return date.

## Technical Detailed Description

## Overall Architecture

* The overall architecture of the Library Management system is based on a new user fills the book details and add the book to the store and view list of issued books and maintain the records of issue and return date of the books.

It consists of the following components:

* Log in User - This component in system can allow user to login into the page
* Add Books - This component in system can allow librarian to add books into the system
* Issue books - This Component marks the issued books to customer
* View Books - This component shows the list of books present in database
* View issued Book - This component maintains the records of issued books to customers
* Return Books - This component marks the return books from customer
* Log Out - This component allows the user to log out from system

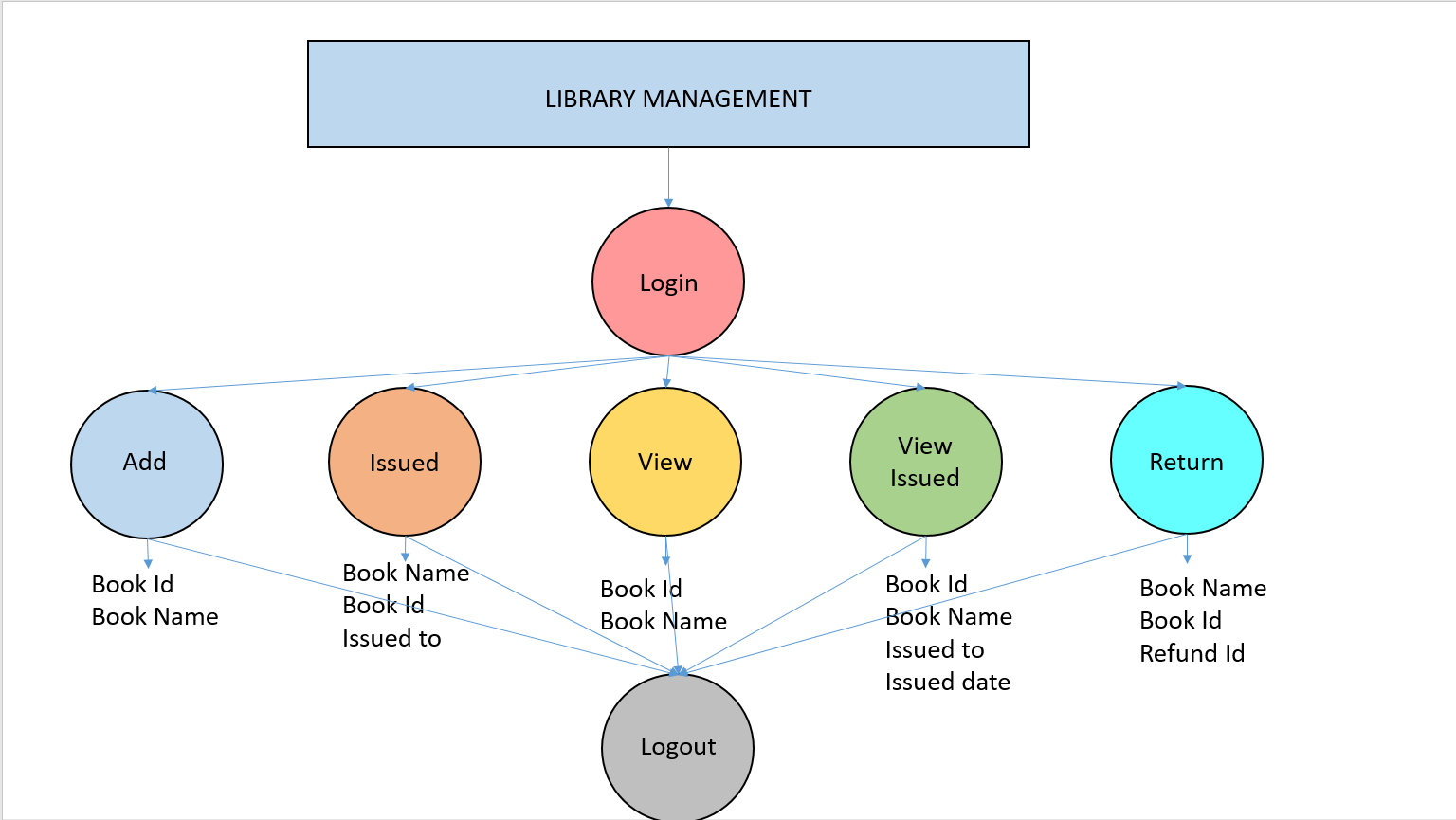
## Data Architecture

The Library Management system uses My SQL data storage mechanism to persist data. The data model consists of the following entities:

* [Eclipse IDE] - The Role of this entity is develop code of the system
* [MY SQL connector jars] - This entity is used to create the connection between the eclipse IDE and MY SQL
* [My SQL] -This entity database creates the table and store the data in database

The data access layer is implemented using [technology or framework used], and provides [description of the data access mechanism, such as an API or set of stored procedures].

## Database Entity Relationship Diagram



## Standards

The Library Management system adheres to the following standards:

1. Class names start with an upper case letter. Variable name start with a lower case letter.
2. Use descriptive and appropriate name for all identifiers (variables, method, names, class names, constants, etc.).
3. Open braces (i.e. “{”) do not start a new line. Close braces (i.e. “}”) do start a new line, and are intended with the code they close
4. Put single space before every “{”.
5. Avoid the use of the default package
6. Comment every 3-7 lines of code.
7. Multi-word identifiers are internally capitalized.
8. Do not use hyphens or underscores to separate multi-word identifiers (except for constants, which have all upper case letters).
9. In-line comments should be used to explain complicated sections of code, such as loops. Use the // comment delimiter for in-line comments. Do not comment generally known features of the java language.
10. Use two blank lines to separate each method within a class definition. Use one blank line to separate logical sections of code within a method.