**Library Management System**

By: -

Shubham (46002223)

**Table of Contents**

**1. Abstract**

**2. Technology Used**

**3. Scope**

3.1. Abstract Level Use Case Diagram

**4. Project Design**

**5. Activity Diagram**

6.1. Student Activity Diagram

6.2. Librarian Activity Diagram

**6. Entity Relation Diagram**

**7. Complete Use Case Diagram**

**8. Class Description**

8.1 Attributes

8.2 Methods

**Abstract**

This project is aimed to develop an application for Library management. This is an integrated web-based application that contains a student and a librarian as users. This lets new students to register or existing students to sign in and proceed with taking books. The application ensures that student can take book as per their choice of genre, title or author. Further, he can return the taken book before 15 days from day of issue, for no overdue charges. Overdue charges are calculated as Rs 10/day for each day exceeding return date.

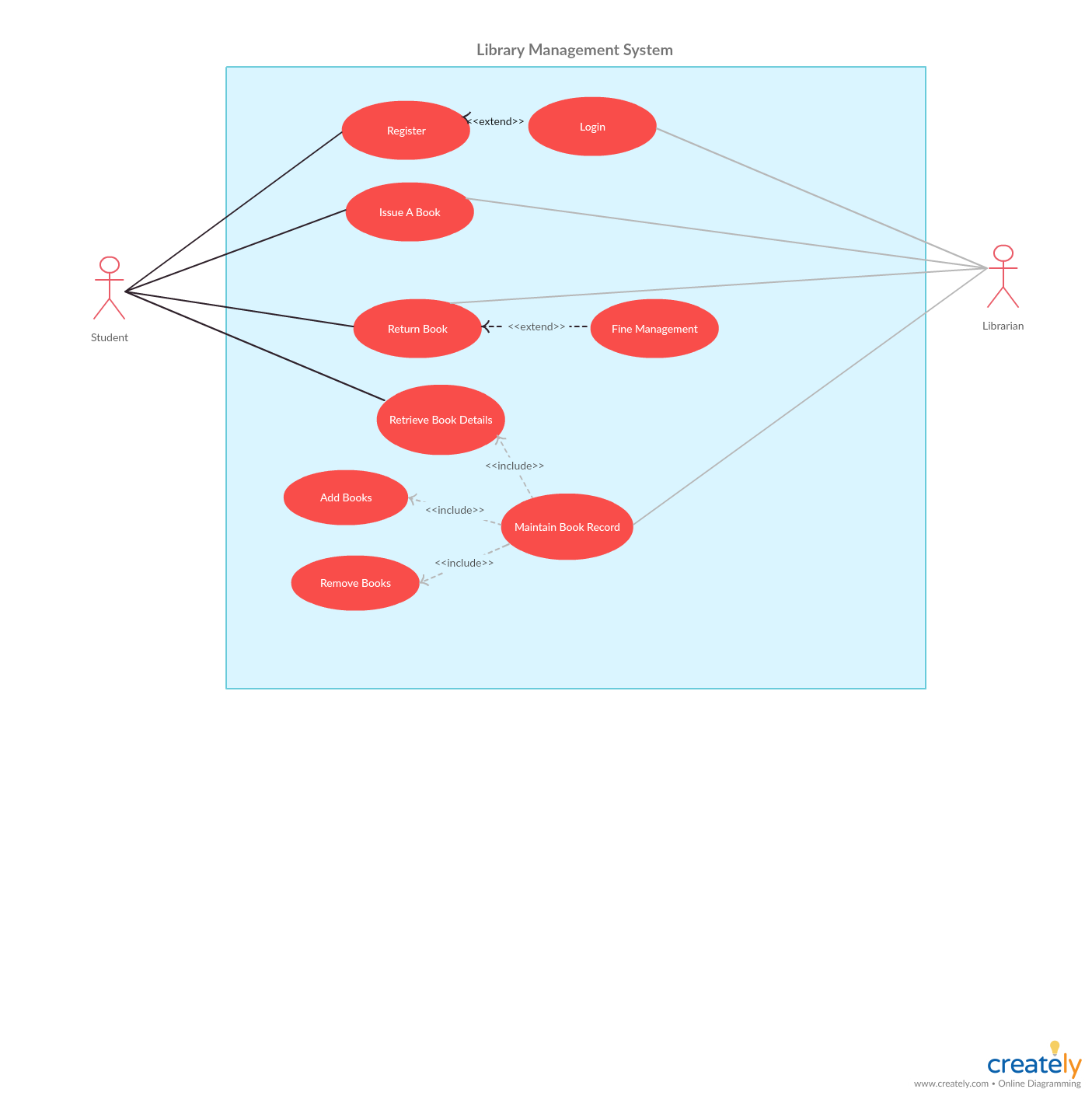
Librarian is able to add new books, retrieve details of the book and remove a book. Further, he can see how many persons have taken a particular book and also how many books are taken by an individual. He can collect overdue charges based on number of days exceeded than scheduled return date.

**Technology Used**

* Java 8(base programming language)
* Jdk1.8.0\_65
* Jre1.8.0\_65
* Log4j-api-2.11.2.jar
* Junit 4.12
* Spring Web 5.1.10.RELEASE
* Spring Boot 2.1.9
* Spring Data 2.1.9
* My SQL 5.5.43(database storage)
* GIT 2.23.0(GIT Bash version)
* Postman 7.11.0(for testing RESTful service)
* Node JS 10.11.0(Javascript Runtime Environment)
* Angular CLI 6.2.9(Framework to develop web apps)
* VsCode 1.39.1

**Scope**

There are two categories of people who would access the system viz. students and librarians. Each one of them would have some exclusive privileges.



Abstract Level Use Case Diagram

**Librarian:**

1. Login to the system using his/her credentials.
2. Sign up by using valid details.
3. Manage books which include adding, retrieving and removing a book.
4. View all the transaction history.
5. Collect overdue charges (if any) from student.

**Student:**

1. Login into the system using his/her credentials.
2. Sign up by using valid details.
3. Issue a book based on title, genre or author.
4. Can view his/her past book issues and returns.
5. Return the book along with overdue charges (if any).

**Project Design**

In this project, there are three microservices which specialize in handling three different activities/functions:

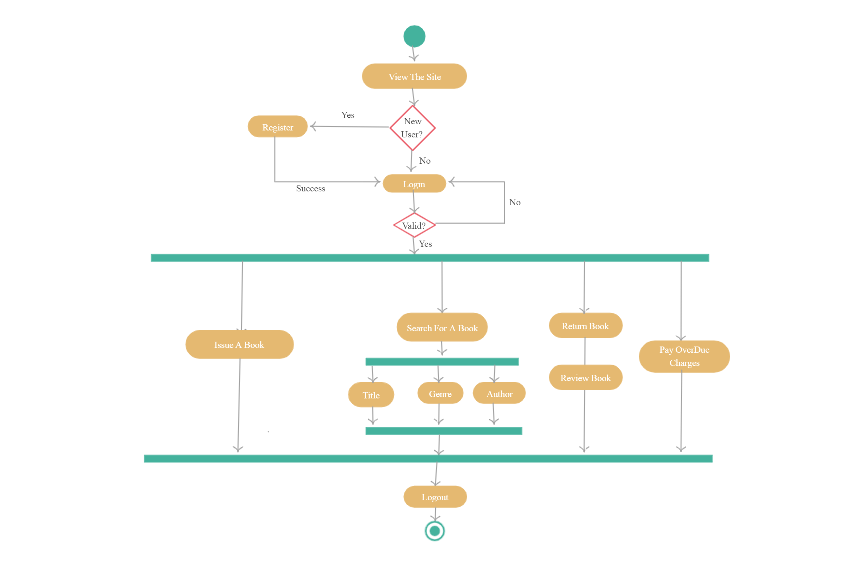
1. **Managing Users Module**: The operations defined in this module are login and signup for students.

**Sign Up:** User details are added on signup page and these details are injected into the corresponding tables into the database. Only valid data are added to the tables else appropriate error messages are displayed. For students, personal details like name, date-of-birth, gender, contact number, email-ID will be added to the database.

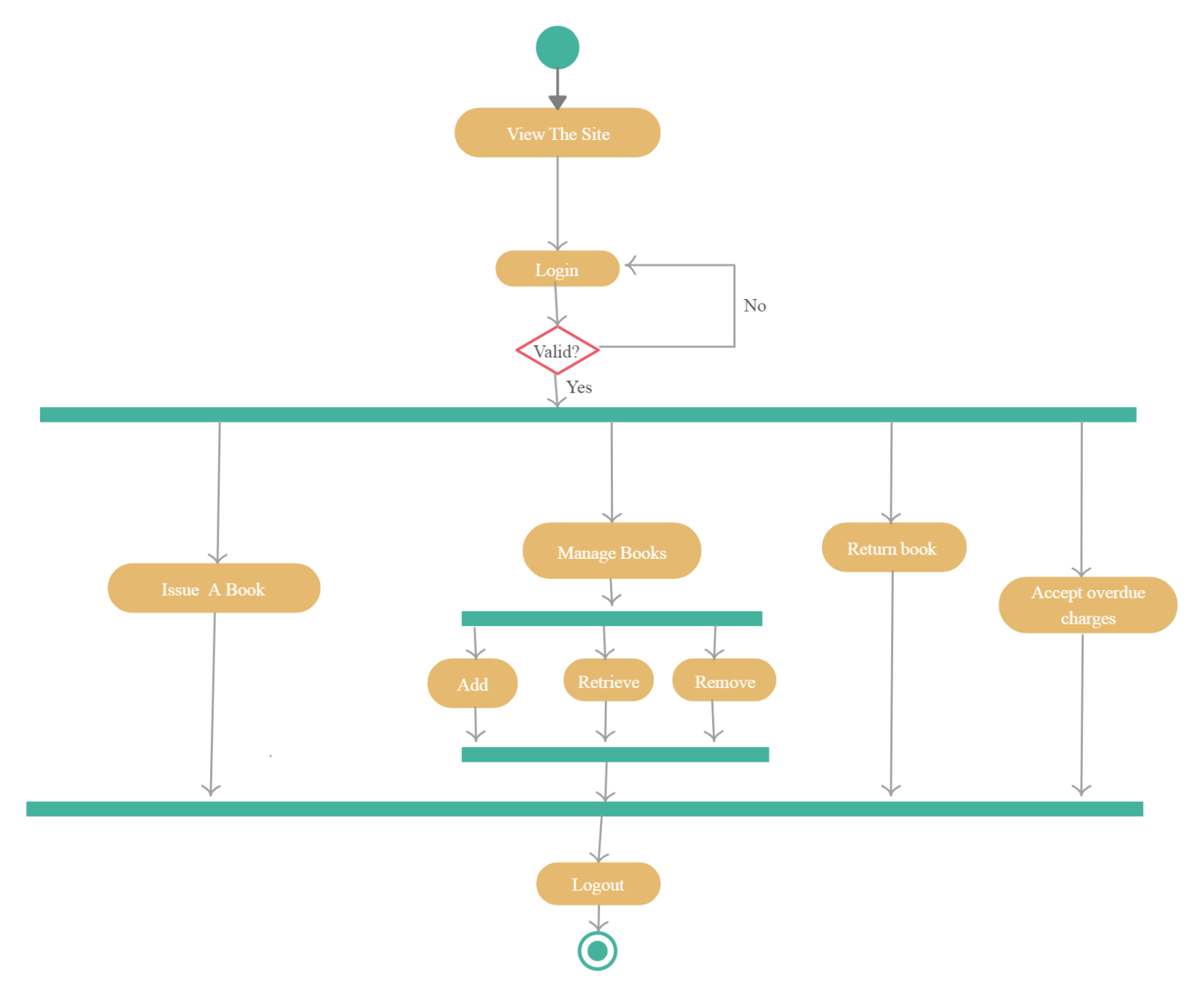
**Login:** Students can login into their account by giving correct username and password. Both students and librarian with valid credentials can login into their account.

1. **Managing Books Module (For Librarian):** In this module, librarian can add new books along with the details like title, author, genre, ISBN, number of pages, number of copies and price. He/She can retrieve details of a particular book or for all books. Further, he/she can remove a book, provided it has not been issued to anyone, on his wish.
2. **Managing Transactions Module:** The functionality of this module is to manage all the transactions. A transaction is defined as issuing a book and then returning it back along with overdue charges, if any. Librarian can see how many persons have taken a particular book. Further, he can keep track of all the transactions- both completed and going on.

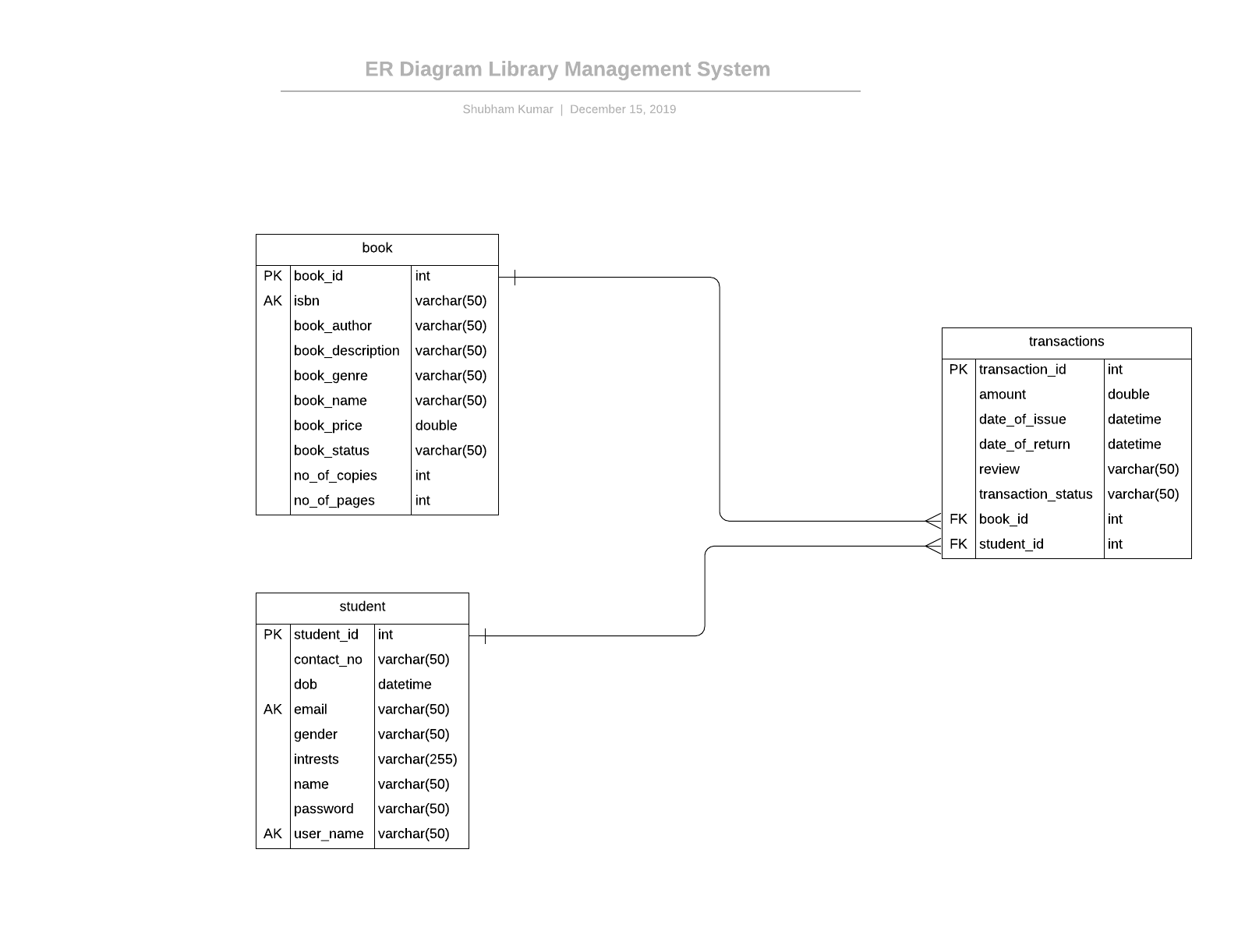
ACTIVITY DIAGRAM (For Student)



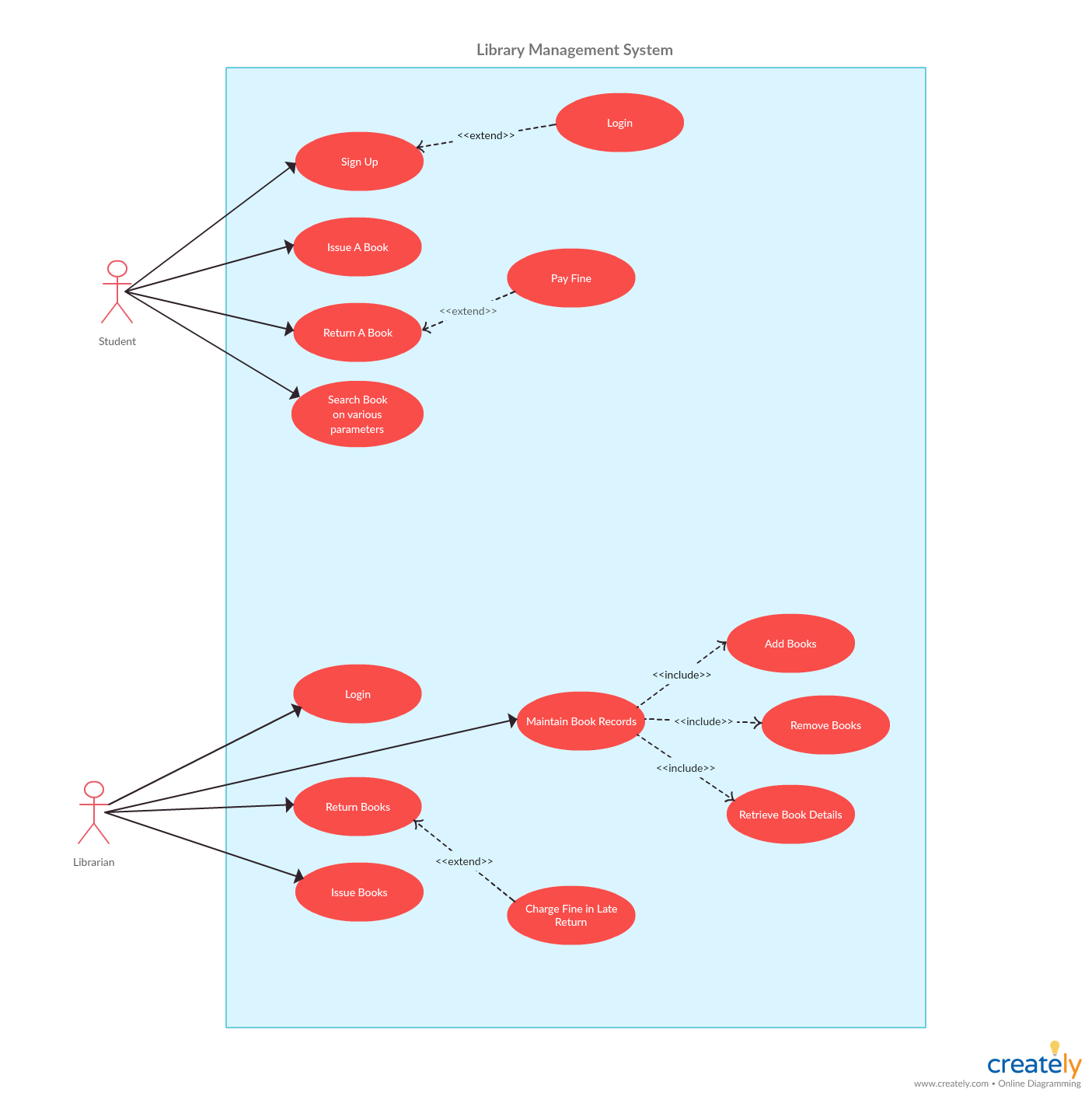
Activity Diagram (For Librarian)

****

Entity relationship diagram

****

Use Case Diagram

****

Snapshots of working App

