Book Store

Descreption : Introducing "BookstoreHub": Your go-to destination for all things literary, powered by a robust Spring Boot REST API. Browse an extensive collection of books, effortlessly place orders, and enjoy a seamless shopping experience from start to finish. With user-friendly navigation, secure payments, and real-time inventory updates, BookstoreHub simplifies book buying. Explore new releases, classics, and bestsellers, and discover your next favorite read with ease. Whether you're a bookworm or casual reader, BookstoreHub has something for everyone. Say goodbye to bookstore queues and hello to convenient online shopping. Dive into the world of literature today with BookstoreHub!

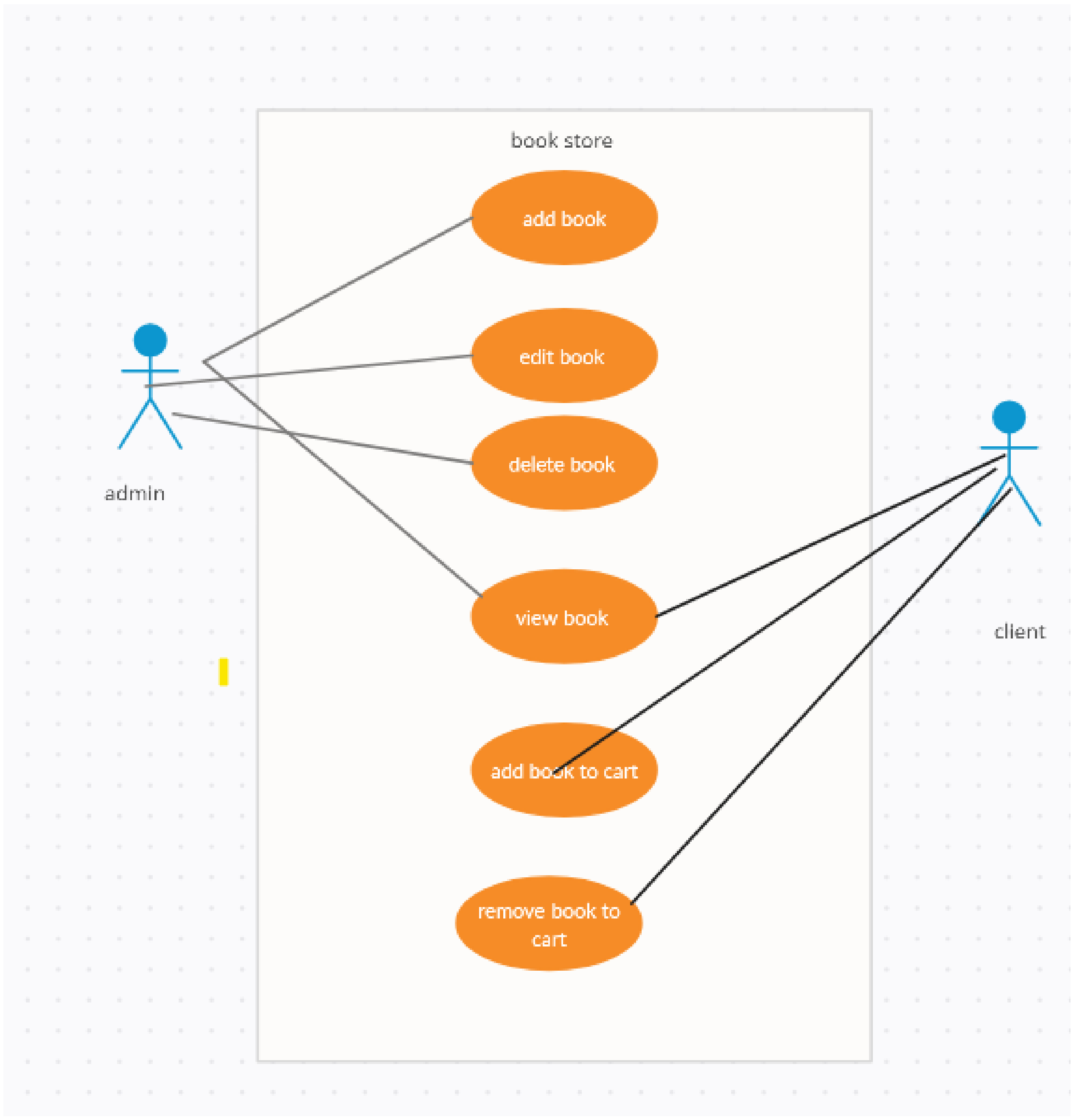
Functional Requirements:

1. **Book Browsing and Search:**
    - Users should be able to browse through a comprehensive collection of books.
    - The app should support efficient search functionality by title, author, genre, and other relevant criteria.
2. **Order Placement:**
    - Users should be able to add books to their cart and proceed to checkout.
    - The system should facilitate order placement, including specifying quantities and delivery details.
3. **Inventory Management:**
    - The app should maintain real-time inventory updates, reflecting accurate stock levels.
    - Users should be notified if a selected book is out of stock or unavailable.
4. **User Profile Management:**
    - Users should be able to view and edit their profiles, including personal information and order history.
    - The system should allow users to track the status of their orders.
5. **Admin Panel:**
    - Administrators should have access to an admin panel to manage books, users, and orders.
    - Admins should be able to add new books, update inventory, and view sales reports.

Non-Functional Requirements:

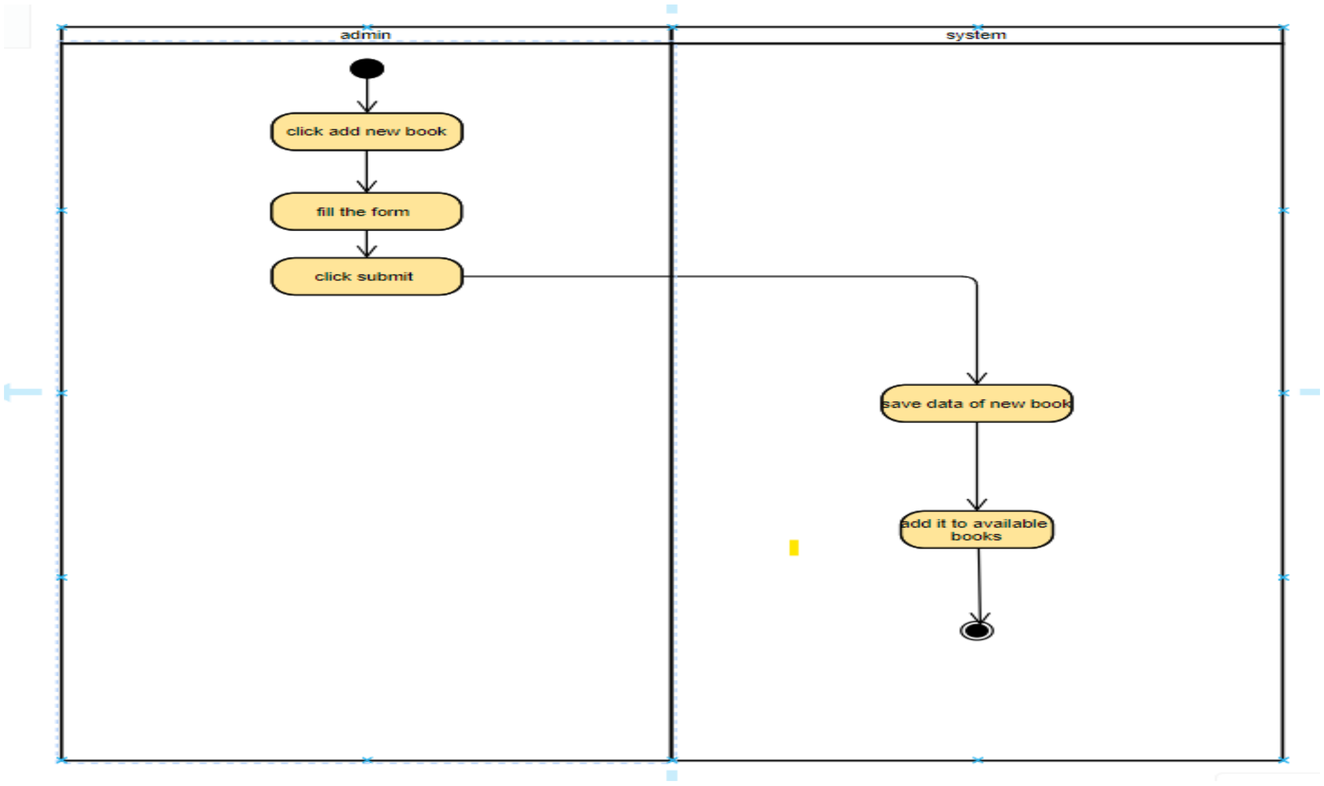
1. **Performance:**
    - The system should respond promptly to user interactions, with minimal loading times.
    - It should handle simultaneous user requests efficiently, ensuring smooth navigation and transaction processing.
2. **Security:**
    - User data, including personal information and payment details, must be encrypted and stored securely.
    - The application should implement measures to prevent unauthorized access and protect against common security threats like SQL injection and cross-site scripting (XSS).
3. **Scalability:**
    - The system should be scalable to accommodate a growing user base and increasing book inventory.
    - It should be designed with scalability in mind, allowing for easy deployment of additional servers or resources as needed.
4. **Reliability:**
    - The application should be highly reliable, minimizing downtime and ensuring continuous availability.
    - It should have mechanisms in place to handle errors gracefully, providing informative error messages to users when necessary.
5. **Usability:**
    - The user interface should be intuitive and easy to navigate, catering to users of all levels of technical proficiency.
    - The application should adhere to accessibility standards, ensuring inclusivity for users with disabilities.
6. **Compatibility:**
    - The app should be compatible with a wide range of devices and web browsers to reach a diverse user base.
    - It should support responsive design principles, optimizing the user experience across various screen sizes and resolutions.
7. **Compliance:**
    - The system should comply with relevant regulations and industry standards for e-commerce and data protection, such as GDPR and PCI DSS.
    - It should provide mechanisms for users to manage their consent preferences regarding data usage and marketing communications.

1.0 Use case

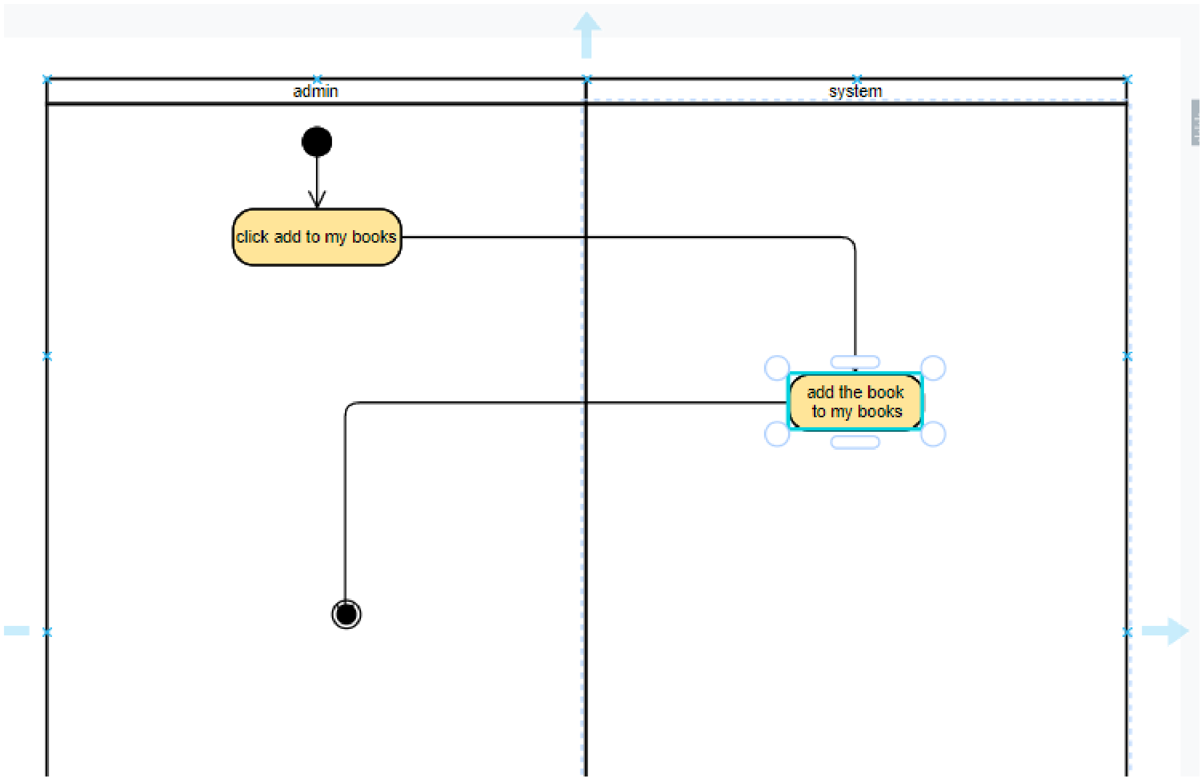

book store
add book
edit book
delete book
view book
add book to cart
remove book to cart
admin
client

2.0 activity diagram

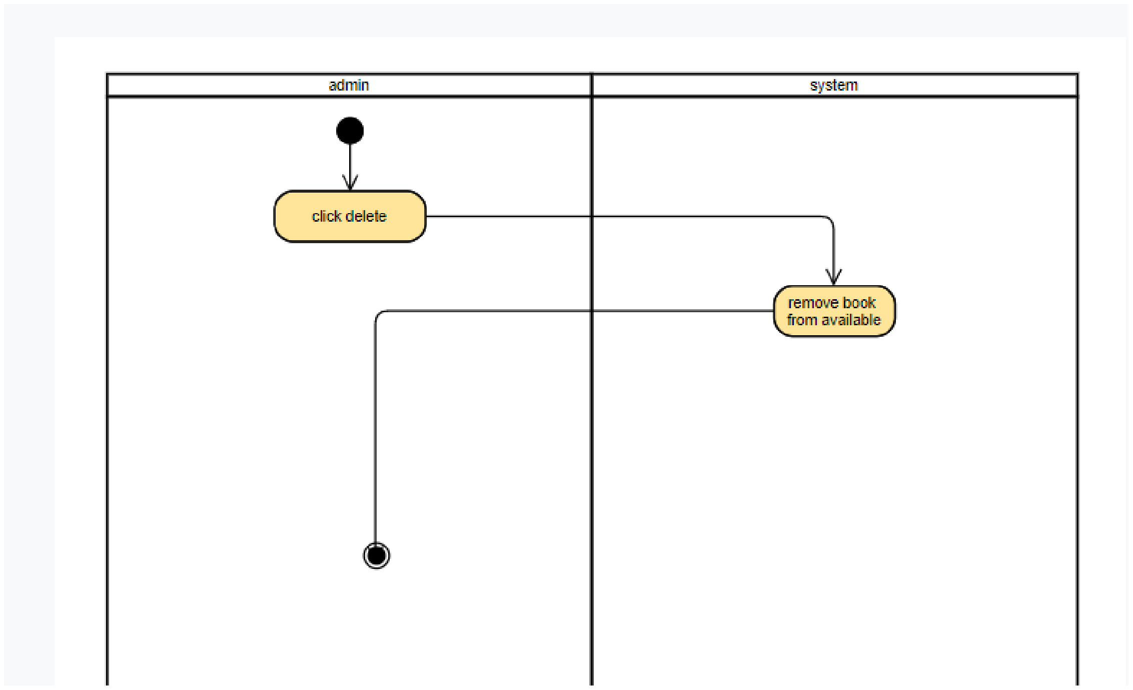
2.1 add new book



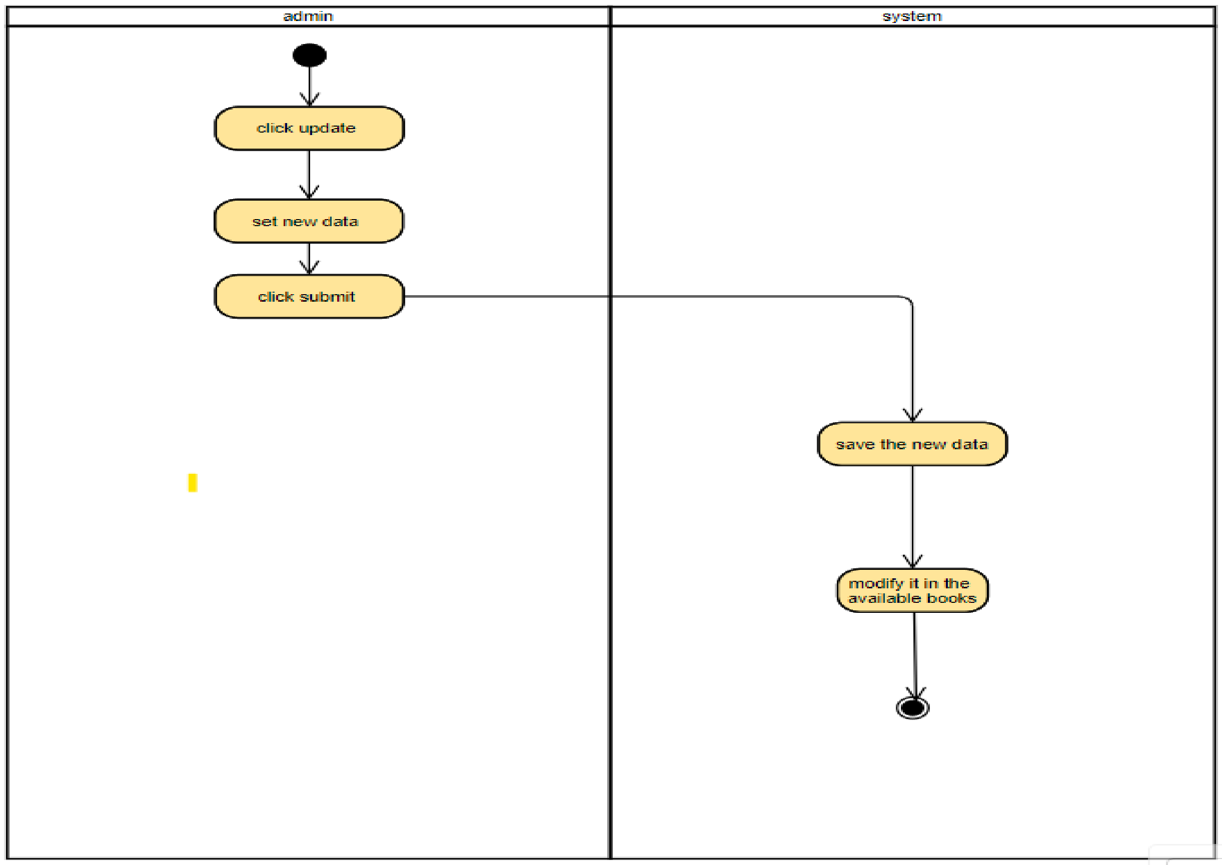
2.2 add to my books



2.3 activity diagram delete book

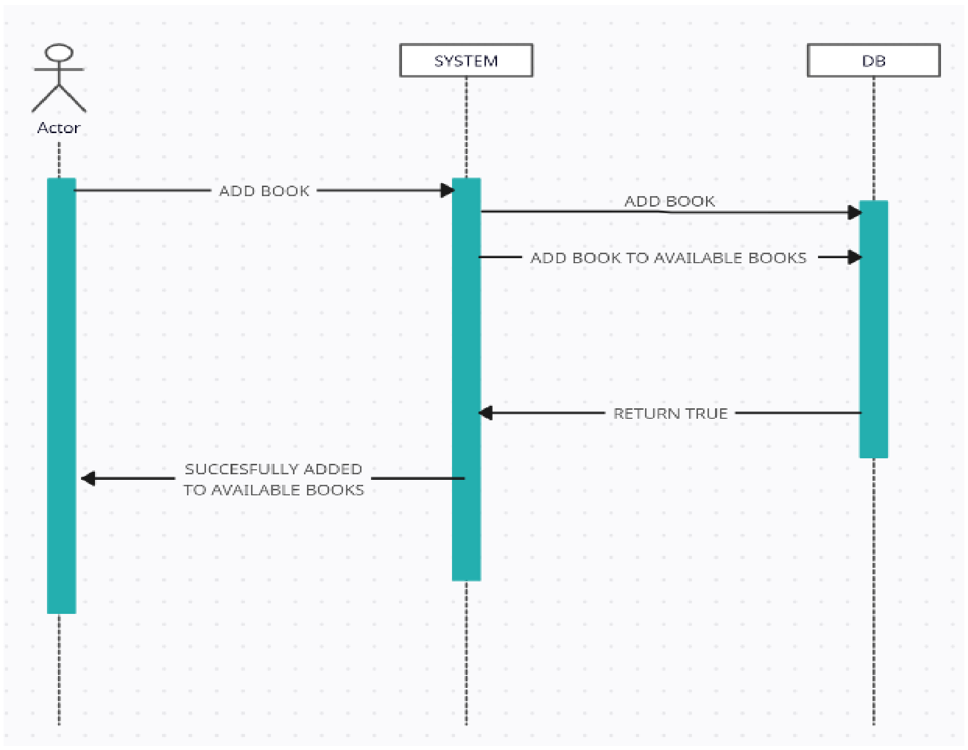


2.4 activity diagram update book

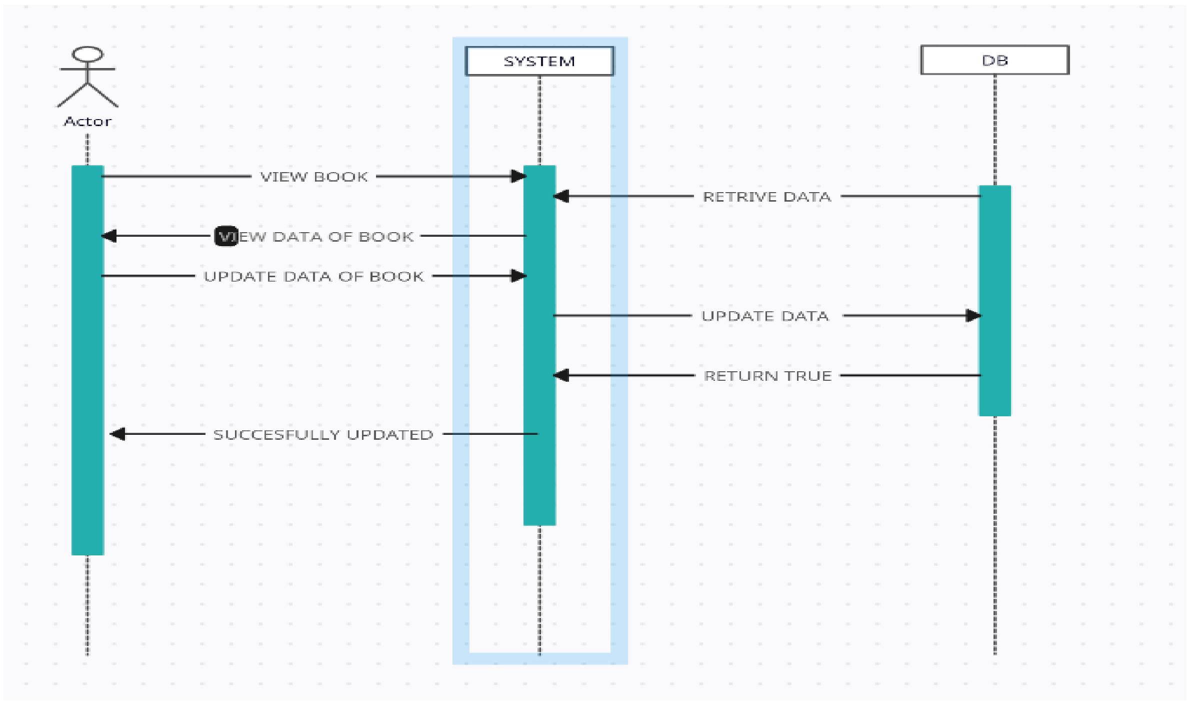


3.0 Sequence diagram

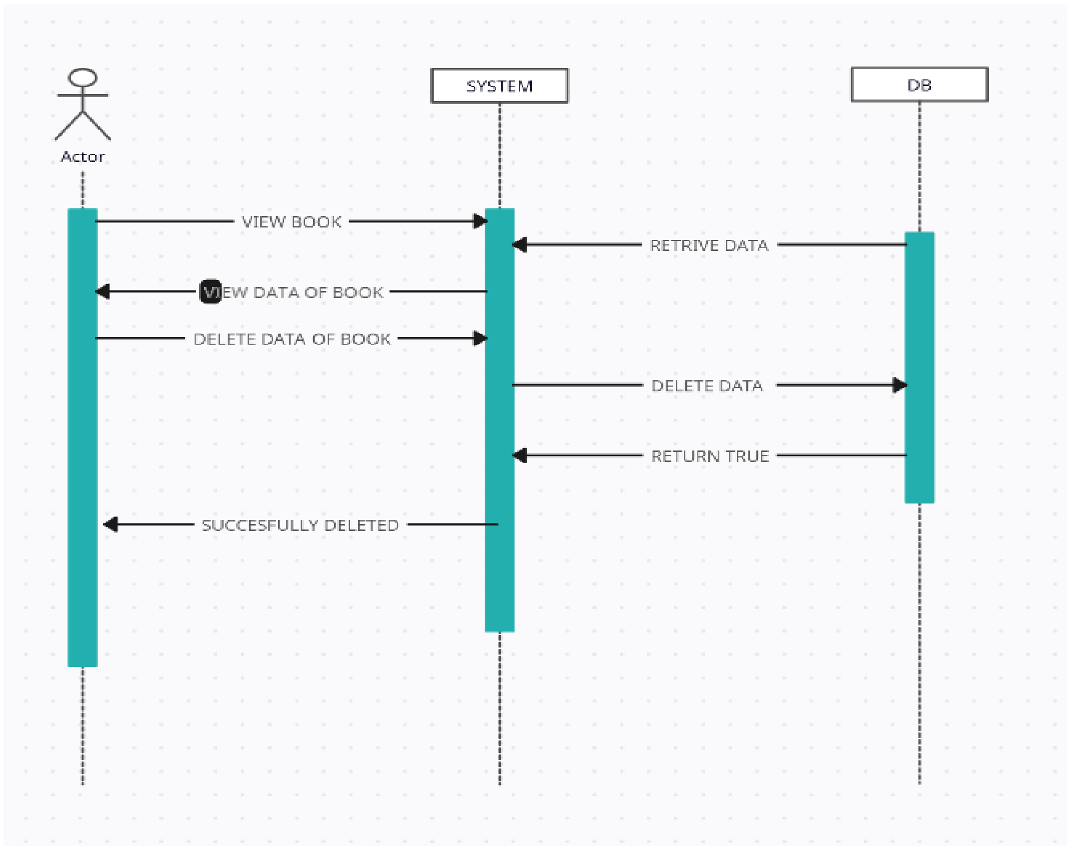
3.1 sequence diagram add book


Actor
SYSTEM
DB
ADD BOOK
ADD BOOK
ADD BOOK TO AVAILABLE BOOKS
RETURN TRUE
SUCCESFULLY ADDED TO AVAILABLE BOOKS

3.2 sequence diagram update book

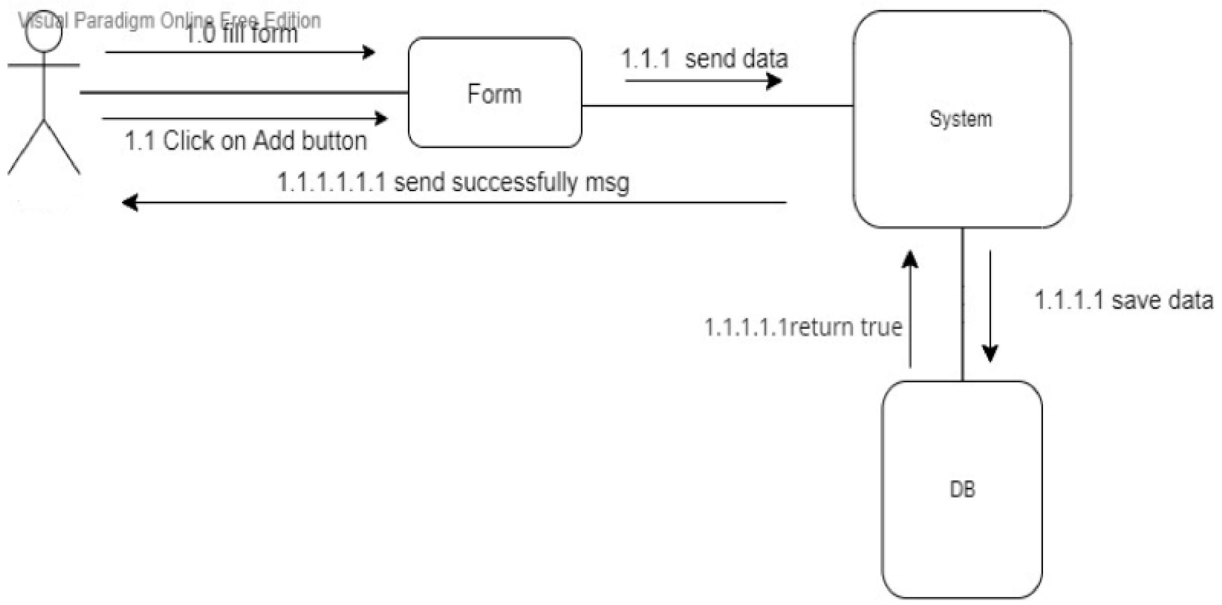


3.3 sequence diagram delete book

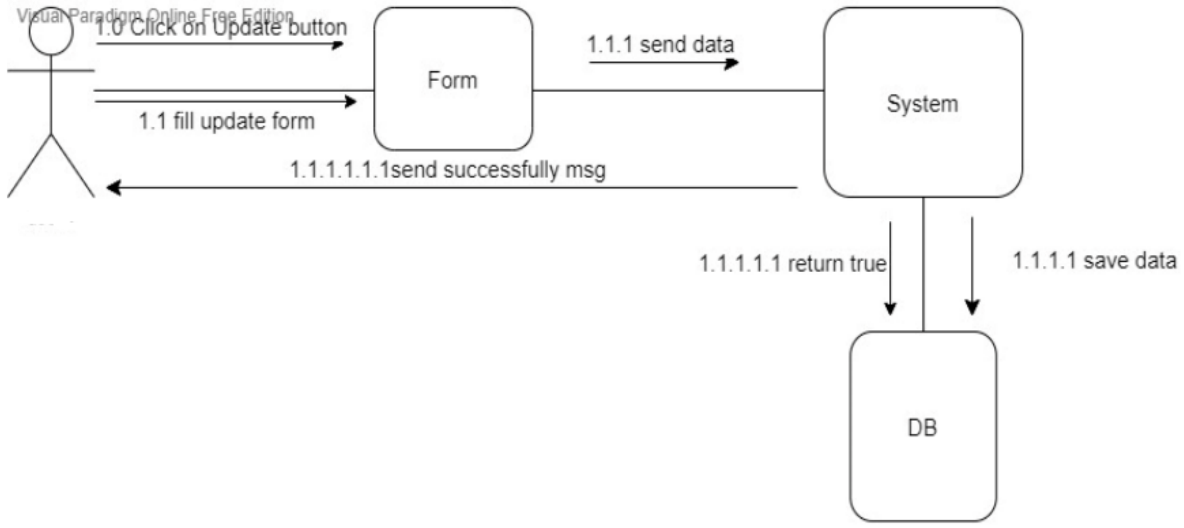


4.0 communication diagram

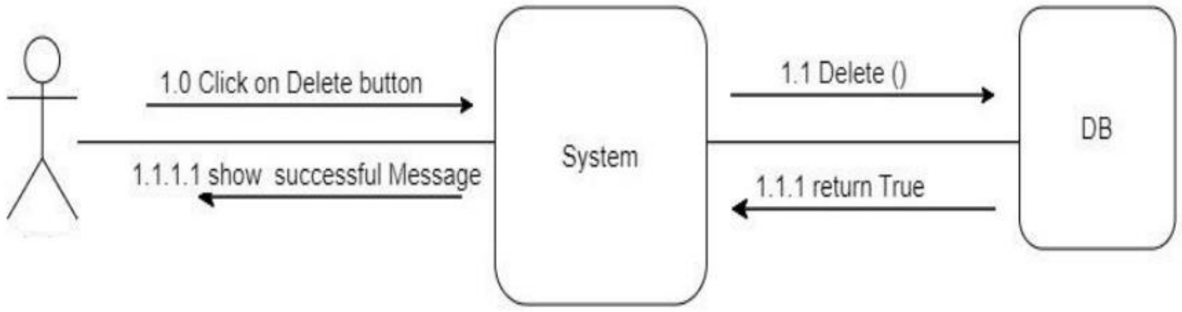
4.1 communication diagram add book



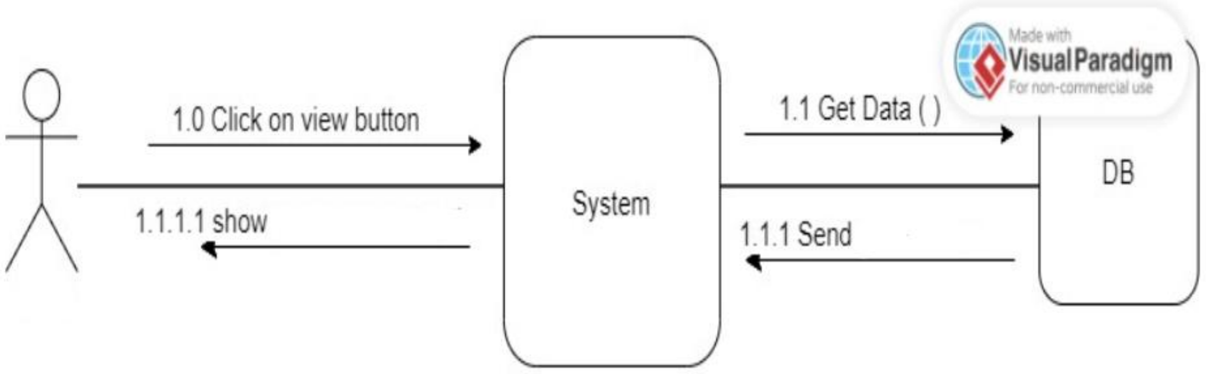
4.2 communication diagram update book


Visual Paradigm Online Free Edition
1.0 Click on Update button
Form
1.1.1 send data
System
1.1 fill update form
1.1.1.1.1.1send successfully msg
1.1.1.1.1 return true
1.1.1.1 save data
DB

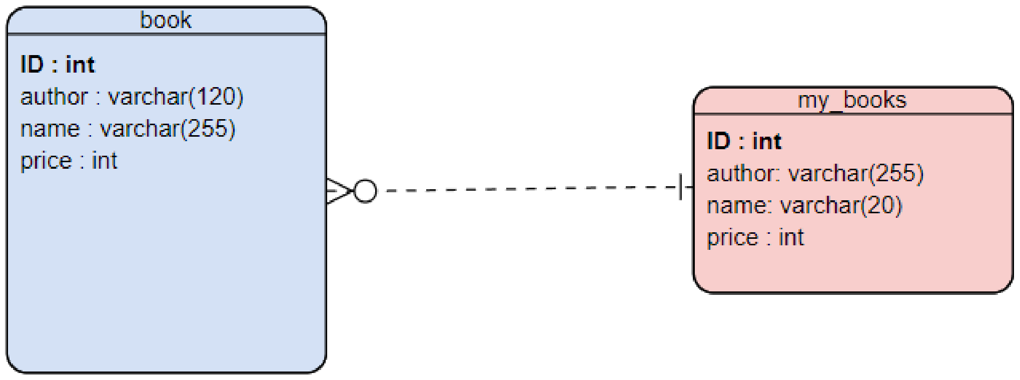
4.3 communication diagram delete book



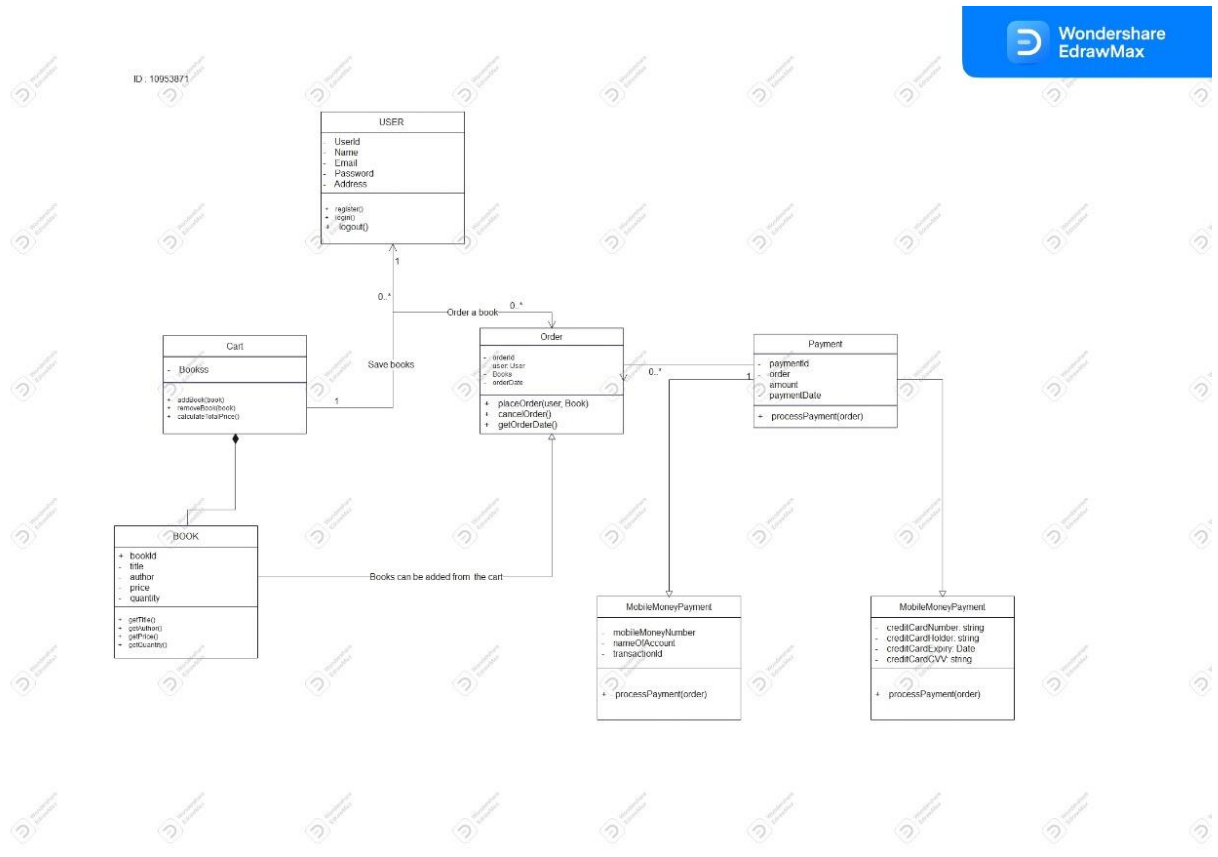
4.4 communication diagram view book



5.0 ERD diagram



6.0 class diagram



7.0 OCL Constrains:

7.1: context Cart

```
  def: operation getBooks(): Set(Book) =
    self.book->asSet();
```

7.2: context USER

```
  def: operation hasCart(): Boolean =
    self.cart->notEmpty();
```

7.3: context Cart

```
  def: operation getTotalPrice(): Real =
    self.book->collect(price)->sum();
```

7.4: context BOOK

```
  inv: self.quantity >= 0;
```