BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC GIA ĐỊNH**

**KHOA: CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**TIỂU LUẬN**

**ĐỀ TÀI:**

**MẪU THIẾT KẾ QUẢN LÝ DANH SÁCH CÁC LOẠI SÁCH**

Ngành: **CÔNG NGHỆ THÔNG TIN**

Chuyên ngành: **KỸ THUẬT PHẦN MỀM**

Giảng viên hướng dẫn: **ThS. LÊ HUỲNH PHƯỚC**

Sinh viên thực hiện: **HỒ MINH TRÍ**

MSSV: **22140066**

Lớp: **221401**

TP. Hồ Chí Minh, tháng 07 năm 2024

**Khoa/Viện: Công Nghệ Thông Tin**

**NHẬN XÉT VÀ CHẤM ĐIỂM CỦA GIẢNG VIÊN**

**TIỂU LUẬN MÔN: Mẫu thiết kế cho phần mềm**

1. **Họ và tên sinh viên: Hồ Minh Trí**
2. **Tên đề tài**: Mẫu thiết kế quản lý danh sách các loại sách

1. **Nhận xét**:

***Những kết quả đạt được:***

***Những hạn chế:***

1. **Điểm đánh giá** *(theo thang điểm 10, làm tròn đến 0.5):*

Sinh viên: 22140066 – Hồ Minh Trí

Điểm số: ……….……

Điểm chữ: ……………………………………………………..

|  |  |
| --- | --- |
|  | *TP. HCM, ngày … tháng 7 năm 2024*  **Giảng viên chấm thi**  *(Ký và ghi rõ họ tên)*  Lê Huỳnh Phước |

**LỜI NÓI ĐẦU**

Ngày nay, công nghệ thông tin đóng vai trò quan trọng trong việc quản lý và vận hành các hệ thống thông tin. Một trong những lĩnh vực được hưởng lợi từ sự phát triển của công nghệ thông tin là quản lý thư viện. Việc áp dụng các giải pháp phần mềm vào quản lý sách thư viện không chỉ giúp nâng cao hiệu quả công việc mà còn mang lại trải nghiệm tốt hơn cho người dùng.

Ứng dụng quản lý sách thư viện là một hệ thống phần mềm được thiết kế nhằm tự động hóa các quy trình quản lý sách, từ việc nhập liệu, tra cứu, mượn trả sách cho đến việc báo cáo, thống kê. Ứng dụng này giúp giảm thiểu sai sót trong quá trình quản lý, tiết kiệm thời gian và nguồn lực, đồng thời nâng cao tính chính xác và minh bạch.

Một số lợi ích cụ thể của ứng dụng quản lý sách thư viện bao gồm:

* Tự động hóa quy trình: Giảm thiểu công việc thủ công, giúp nhân viên thư viện tập trung vào các nhiệm vụ quan trọng hơn.
* Quản lý thông tin hiệu quả: Dễ dàng quản lý thông tin sách, người mượn, và các giao dịch mượn trả sách.
* Tra cứu nhanh chóng: Người dùng có thể dễ dàng tra cứu thông tin sách, kiểm tra tình trạng sẵn có của sách một cách nhanh chóng và chính xác.
* Thống kê và báo cáo: Cung cấp các báo cáo chi tiết về tình hình mượn trả sách, giúp quản lý dễ dàng đánh giá và đưa ra các quyết định phù hợp.

Ứng dụng quản lý sách thư viện được phát triển dựa trên các nguyên tắc và mẫu thiết kế phần mềm hiện đại, bao gồm mô hình 3-layer, mẫu thiết kế Observer, Command, và MVC. Việc áp dụng các nguyên tắc này giúp ứng dụng có cấu trúc rõ ràng, dễ bảo trì, và mở rộng trong tương lai.

**MỤC LỤC**

[**CHƯƠNG 1. LÝ THUYẾT VỀ MẪU THIẾT KẾ** 1](#_Toc173143987)

[**1.1.** **Hiện trạng** 1](#_Toc173143988)

[**1.2.** **Yêu cầu** 1](#_Toc173143989)

[**1.3.** **Mô hình hoá yêu cầu** 1](#_Toc173143990)

[**1.4.** **Định nghĩa lý thyết về các mẫu thiết kế được áp dụng.** 2](#_Toc173143991)

[**1.4.1** **Command**. 2](#_Toc173143992)

[**1.4.2** **Observer** (**Publisher – Subcriber).** 3](#_Toc173143993)

[**1.4.4**  **Three – Layer Architecture (Cấu trúc 3 tầng).** 3](#_Toc173143994)

[**CHƯƠNG 2. PHÂN TÍCH, THIẾT KẾ CHƯƠNG TRÌNH** 5](#_Toc173143995)

[**2.1. Sơ đồ Use Case** 5](#_Toc173143996)

[**2.1.1.**  **Sơ đồ Use Case tổng quát** 5](#_Toc173143997)

[**2.1.2. Thêm sách** 6](#_Toc173143998)

[**2.1.3**  **Xoá sách** 6](#_Toc173143999)

[**2.1.4. Sửa thông tin sách** 7](#_Toc173144000)

[**2.1.5. Tìm kiếm sách** 7](#_Toc173144001)

[**2.1.6. Tính tổng thành tiền.** 7](#_Toc173144002)

[**2.1.7. Tính trung bình cộng sách giáo khoa.** 8](#_Toc173144003)

[**2.1.8. Tính trung bình cộng sách tham khảo.** 8](#_Toc173144004)

[**2.2. Sơ đồ lớp** 8](#_Toc173144005)

[**2.2.1. Tầng Presentation** 8](#_Toc173144006)

[**2.2.2. Tầng Domain** 9](#_Toc173144007)

[**2.2.3 Tầng Persistence** 10](#_Toc173144008)

[**2.3. Sơ đồ tuần tự (Sequency Diagram)** 11](#_Toc173144009)

[**2.3.1. Thêm sách** 11](#_Toc173144010)

[**2.3.2** **Xoá sách** 12](#_Toc173144011)

[**2.3.3. Sửa thông tin sách** 13](#_Toc173144012)

[**2.3.4** **Tìm kiếm sách** 14](#_Toc173144013)

[**2.3.5** **Tính tổng số lượng thành tiền .** 15](#_Toc173144014)

[**2.3.6.**  **Tính trung bình cộng sách giáo khoa.** 16](#_Toc173144015)

[**2.3.7. Tính trung bình cộng sách tham khảo.** 17](#_Toc173144016)

[**CHƯƠNG 3: CÀI ĐẶT THỬ NGHIỆM** 18](#_Toc173144017)

[**3.1. Giao diện chính** 18](#_Toc173144018)

[**3.2. Tìm kiếm sách theo tên Nhà Xuất Bản** 18](#_Toc173144019)

[**3.4. Xoá Thông tin sách** 20](#_Toc173144020)

[**3.5. Tổng Thành Tiền Sách Giáo Khoa và Sách Tham Khảo.** 20](#_Toc173144021)

[**3.6. Trung bình cộng đơn giá của sách tham khảo** 22](#_Toc173144022)

[**3.7. Tìm kiếm theo mã sách** 22](#_Toc173144023)

[**CHƯƠNG 4. KẾT LUẬN VÀ ĐÁNH GIÁ** 25](#_Toc173144024)

[**4.1. Kết quả đạt được** 25](#_Toc173144025)

[**4.2. Đánh giá ưu điểm và nhược điểm** 25](#_Toc173144026)

[**4.3. Tổng kết** 26](#_Toc173144027)

[**TÀI LIỆU THAM KHẢO** 27](#_Toc173144028)

# **CHƯƠNG 1. LÝ THUYẾT VỀ MẪU THIẾT KẾ**

* 1. **Hiện trạng**

Thư viên X quản lý danh sách các loại sách . Thông tin về các loại sách:

-Sách giáo khoa: ID sách, ngày nhập (ngày, tháng, năm), đơn giá, số lượng, nhà xuất bản, tình trạng (mới, cũ).

* Nếu tình trạng sách là mới thì: Tổng tiền = số lượng \* đơn giá.
* Nếu tình trạng sách là cũ thì: Tổng tiền = số lượng \* đơn giá \* 50%

-Sách tham khảo: ID sách, ngày nhập (ngày, tháng, năm), đơn giá, số lượng, nhà xuất bản, thuế. Tổng tiền = số lượng \* giá + thuế

* 1. **Yêu cầu**
* Thêm, sửa, xoá, in , tìm kiếm.
* Tính tổng tổng tiền từng loại
* Tính trung bình giá của các sách tham khảo
* Xuất ra các sách giáo khoa của nhà xuất bản được nhập vào ô tìm kiếm
  1. **Mô hình hoá yêu cầu**



Hình 1. 1: Mô hình hoá yêu cầu

* 1. **Định nghĩa lý thyết về các mẫu thiết kế được áp dụng.**
     1. **Command**.

Giới thiệu:

-Mẫu thiết kế Command là một trong những mẫu thiết kế hành vi (behavioral pattern) thuộc nhóm mẫu thiết kế hướng đối tượng. Mẫu này giúp bạn biến một yêu cầu thành một đối tượng độc lập chứa tất cả các thông tin về yêu cầu đó. Điều này cho phép bạn tham số hóa các phương thức với các yêu cầu khác nhau, trì hoãn hoặc sắp xếp lại việc thực hiện yêu cầu.

Vấn đề:

Trong một hệ thống phần mềm, các yêu cầu cần được thực thi có thể phức tạp và đa dạng. Việc quản lý, truyền tải, trì hoãn hoặc hoàn tác các yêu cầu này có thể gặp khó khăn nếu không có một cơ chế rõ ràng và linh hoạt. Các yêu cầu có thể bao gồm các thao tác khác nhau như tạo mới, cập nhật, xóa dữ liệu hoặc thực hiện các hành động cụ thể. Việc trực tiếp xử lý các yêu cầu này có thể dẫn đến mã nguồn phức tạp, khó bảo trì và mở rộng.

Giải pháp:

-Sử dụng mẫu thiết kế Command để biến mỗi yêu cầu thành một đối tượng độc lập chứa tất cả thông tin về yêu cầu đó. Bằng cách này, bạn có thể dễ dàng:

**Truyền tải các yêu cầu dưới dạng đối số của phương thức**: Việc chuyển đổi các yêu cầu thành các đối tượng độc lập giúp bạn dễ dàng truyền tải chúng dưới dạng đối số của các phương thức, làm cho mã nguồn trở nên linh hoạt và dễ quản lý hơn.

**Trì hoãn hoặc xếp hàng chờ để thực thi yêu cầu**: Bạn có thể lưu trữ các đối tượng yêu cầu trong một hàng chờ và thực hiện chúng theo thứ tự ưu tiên hoặc vào thời điểm thích hợp. Điều này giúp bạn kiểm soát tốt hơn việc thực thi các yêu cầu trong hệ thống.

**Hỗ trợ các thao tác có thể hoàn tác**: Với mẫu thiết kế Command, bạn có thể dễ dàng lưu trữ trạng thái của hệ thống trước khi thực thi một yêu cầu và khôi phục lại trạng thái này nếu cần thiết. Điều này giúp hỗ trợ các thao tác có thể hoàn tác, cải thiện trải nghiệm người dùng và đảm bảo tính toàn vẹn của hệ thống.

* + 1. **Observer** (**Publisher – Subcriber).**

**Observer** là một mối quan hệ một-nhiều giữa các đối tượng để khi một đối tượng thay đổi trạng thái, tất cả các đối tượng phụ thuộc của nó được thông báo và cập nhật tự động. Mẫu thiết kế này thường được sử dụng trong các hệ thống mà một thay đổi trong một đối tượng cần được phản ánh trong các đối tượng khác mà không cần chúng có mối liên kết chặt chẽ với nhau.

- **Vấn đề:** Nhiều thiết bị giám sát cần được thông báo khi một thiết bị thay đổi trạng thái. Số lượng và loại thiết bị giám sát này có thể thay đổi liên tục. Chúng ta muốn tránh việc kiểm tra trạng thái liên tục (polling) và không muốn mã của thiết bị phải biết chi tiết về số lượng hoặc loại thiết bị giám sát.

**- Giải pháp:** Thiết bị sẽ giữ một danh sách các thiết bị giám sát đã đăng ký. Các thiết bị giám sát có thể khác nhau, nhưng tất cả đều phải có một hàm update() để thiết bị có thể gọi khi thay đổi trạng thái. Danh sách này sẽ được quản lý bởi một lớp cơ bản Publisher có thể tái sử dụng. Một lớp cơ bản trừu tượng Subscriber sẽ định nghĩa giao diện mà các thiết bị giám sát cần triển khai.

### **1.4.4 Three – Layer Architecture (Cấu trúc 3 tầng).**

Three-Layer Architecture là một mô hình phân chia ứng dụng thành ba tầng chính:

**- Tầng Giao diện**: Tầng này chịu trách nhiệm tương tác với người dùng, bao gồm giao diện đồ họa (GUI), Nó nhận đầu vào từ người dùng và gửi yêu cầu tới Domain Layer để xử lý.

**- Tầng Xử lý logic nghiệp vụ:** Tầng này chứa các quy tắc, logic và tính toán cốt lõi của ứng dụng. Nó xử lý các yêu cầu từ Presentation Layer và tương tác với Persistence Layer để lấy hoặc lưu dữ liệu.

**- Tầng Dữ liệu**: Tầng này lưu trữ dữ liệu chịu trách nhiệm về truy xuất và thao tác dữ liệu từ cơ sở dữ liệu hoặc các nguồn dữ liệu khác. Nó cung cấp giao diện để Domain Layer tương tác với cơ sở dữ liệu.

Vấn đề:

* Phức tạp trong quản lý mã nguồn: Khi ứng dụng trở nên lớn hơn và phức tạp hơn, việc quản lý mã nguồn và sự phụ thuộc giữa các thành phần trở nên khó khăn.
* Khó bảo trì và mở rộng: Một hệ thống không được tổ chức tốt sẽ gây khó khăn trong việc bảo trì và mở rộng. Thay đổi một phần của hệ thống có thể ảnh hưởng đến nhiều phần khác.
* Tái sử dụng mã nguồn hạn chế: Mã nguồn không được tách biệt rõ ràng có thể làm giảm khả năng tái sử dụng và kiểm thử.

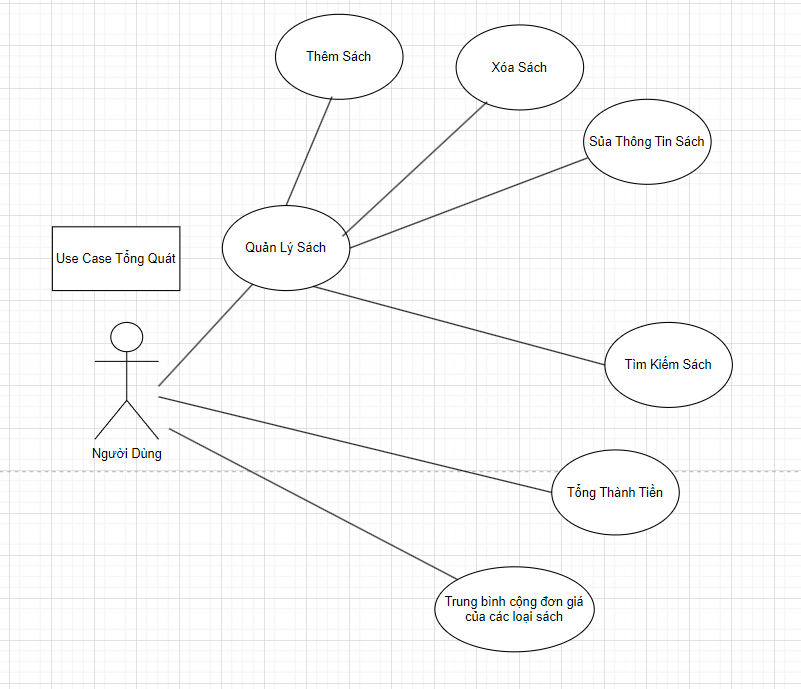
Giải pháp:

* Tách biệt các tầng: Phân chia ứng dụng thành ba tầng rõ ràng giúp mã nguồn trở nên dễ hiểu và dễ quản lý hơn. Mỗi tầng chịu trách nhiệm riêng của mình, giúp giảm sự phụ thuộc và tương tác giữa các tầng.
* Tăng tính bảo trì và mở rộng: Khi các thành phần được tách biệt rõ ràng, việc thay đổi hoặc mở rộng một phần của hệ thống trở nên dễ dàng hơn mà không ảnh hưởng đến các phần khác.
* Tăng khả năng tái sử dụng và kiểm thử: Với mô hình ba tầng, mã nguồn ở mỗi tầng có thể được tái sử dụng trong các ứng dụng khác. Ngoài ra, việc kiểm thử từng tầng riêng biệt cũng trở nên dễ dàng và hiệu quả hơn.

# **CHƯƠNG 2. PHÂN TÍCH, THIẾT KẾ CHƯƠNG TRÌNH**

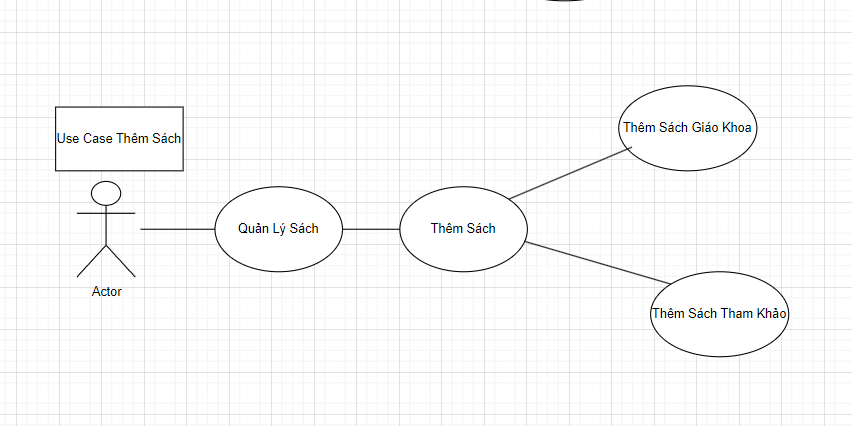
## **2.1. Sơ đồ Use Case**

### **2.1.1. Sơ đồ Use Case tổng quát**



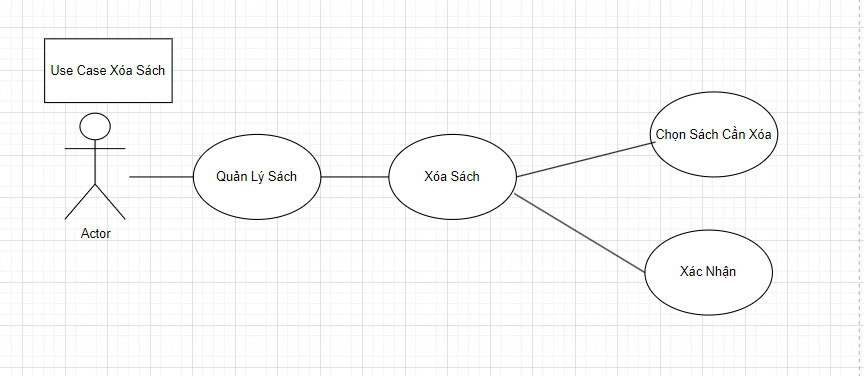
Hình 2. 1: Sơ đồ Use Case tổng quát

### **2.1.2. Thêm sách**



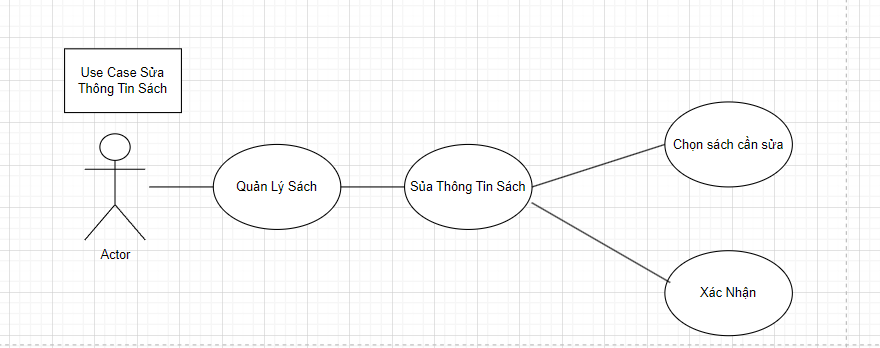
Hình 2. 2: Sơ đồ Use Case thêm sách

### **2.1.3 Xoá sách**



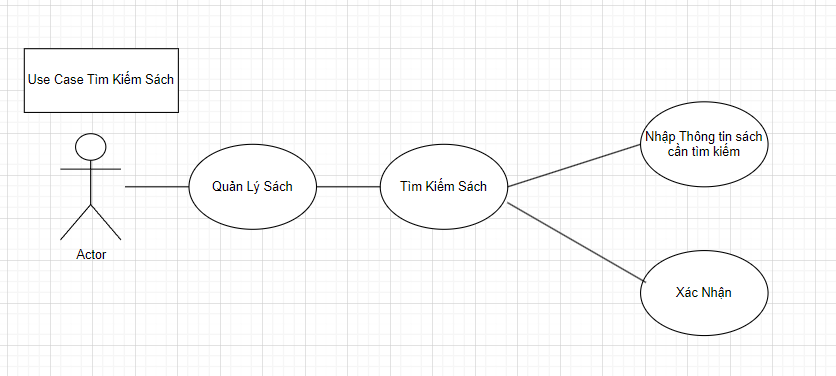
Hình 2. 3: Sơ đồ Use Case xoá sách

### **2.1.4. Sửa thông tin sách**



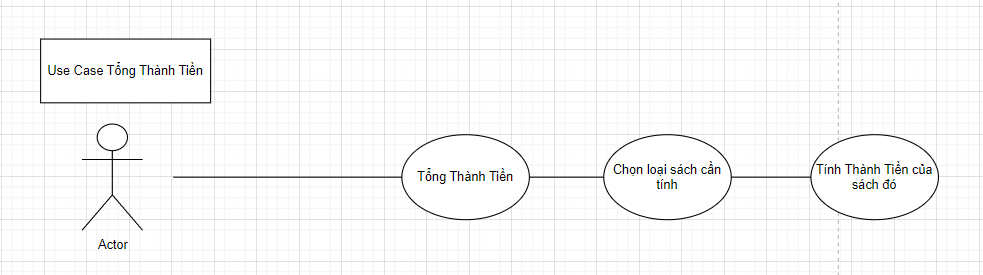
Hình 2. 4: Sơ đồ Use Case sửa thông tin sách

### **2.1.5. Tìm kiếm sách**



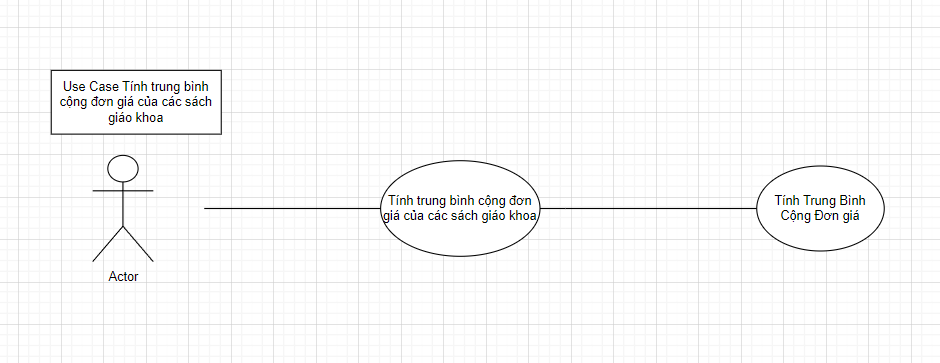
Hình 2. 5: Sơ đồ Use Case tìm kiếm sách

### **2.1.6. Tính tổng thành tiền.**



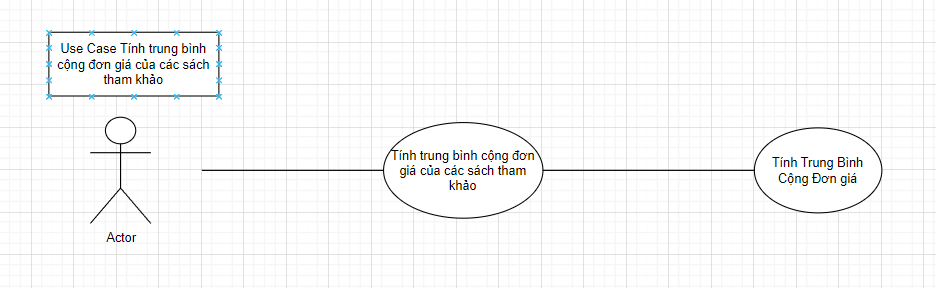
Hình 2. 6: Sơ đồ Use Case tính tổng thành tiền

### **2.1.7. Tính trung bình cộng sách giáo khoa.**



Hình 2. 7: Sơ đồ Use Case tính trung bình cộng sách giáo khoa

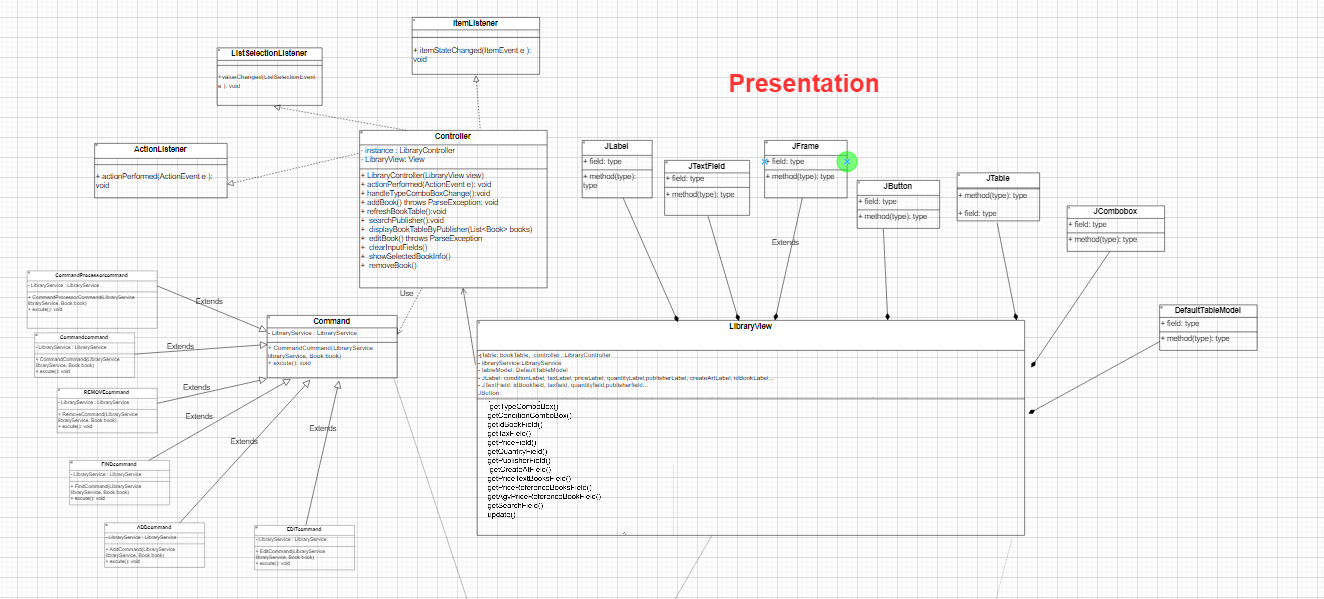
### **2.1.8. Tính trung bình cộng sách tham khảo.**



*Hình 2. 8: Sơ đồ Use Case tính trung bình cộng sách tham khảo*

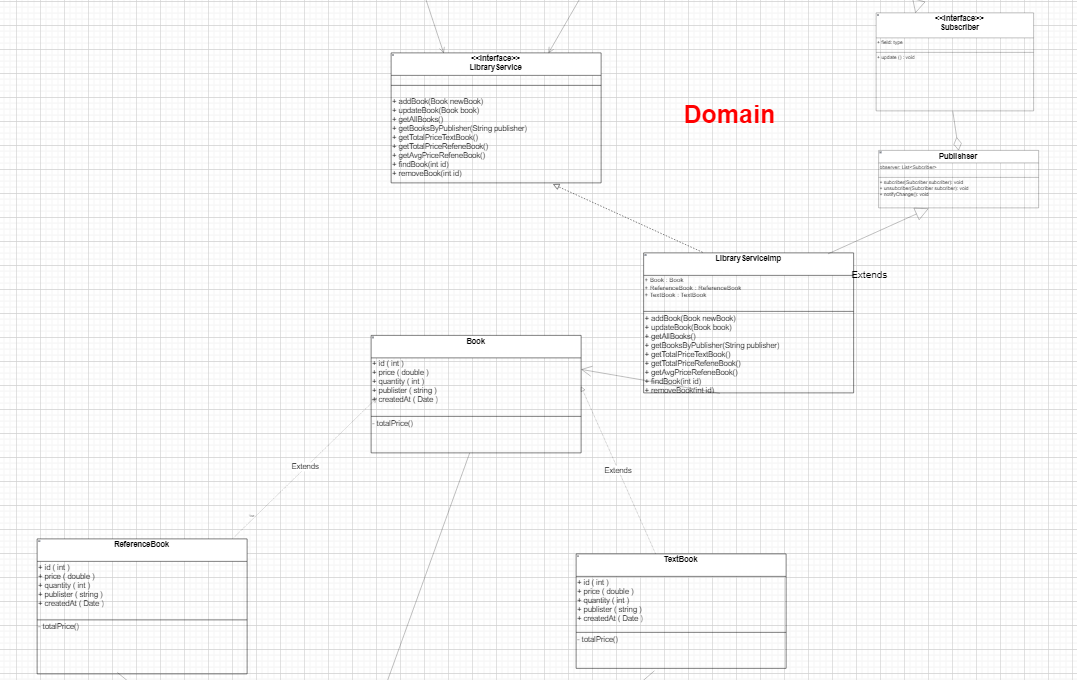
## **2.2. Sơ đồ lớp**

### **2.2.1. Tầng Presentation**



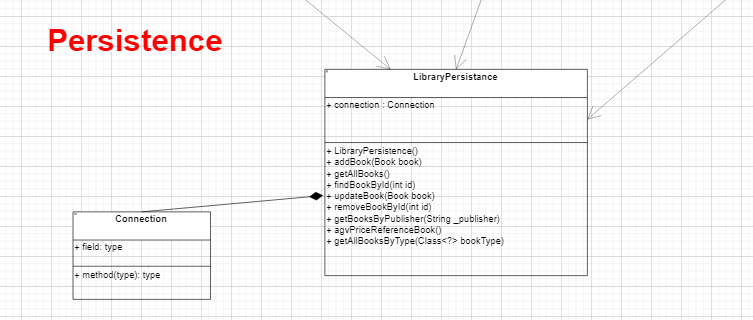
Hình 2. 8: Tầng Presentation

### **2.2.2. Tầng Domain**



Hình 2. 9: Tầng Domain

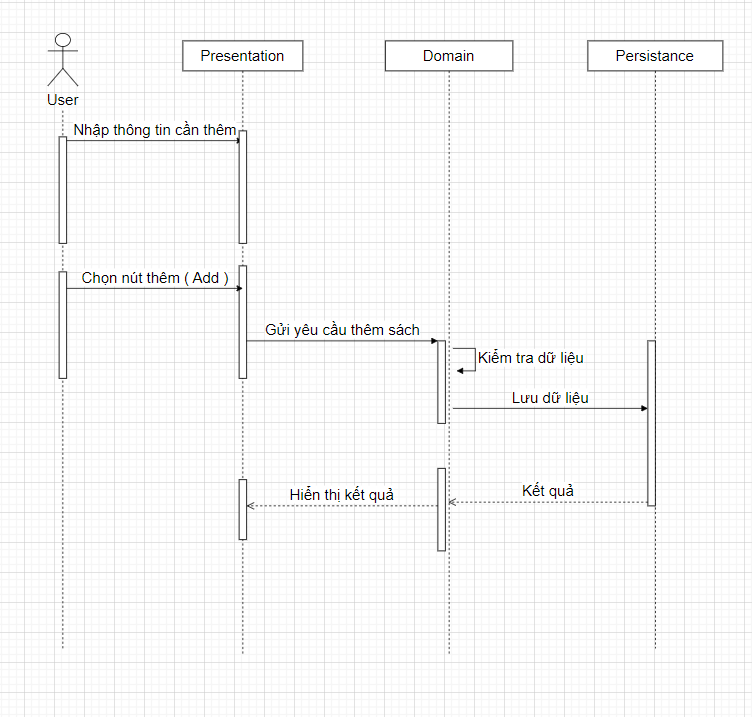
### **2.2.3 Tầng Persistence**



Hình 2. 10: Tầng Persistence

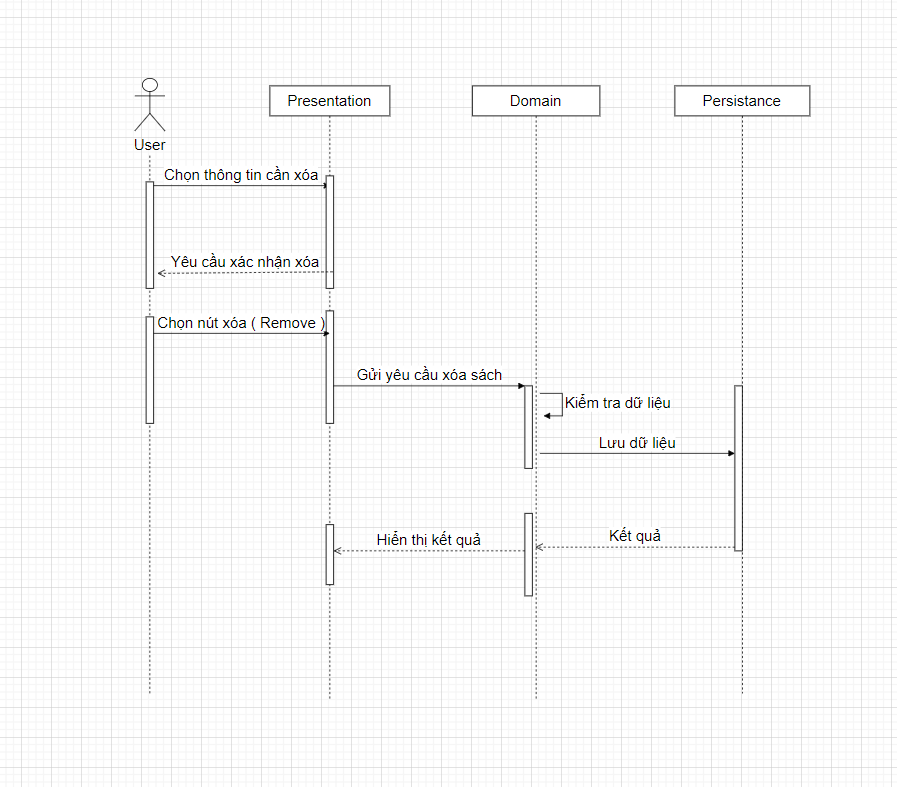
## **2.3. Sơ đồ tuần tự (Sequency Diagram)**

### **2.3.1. Thêm sách**



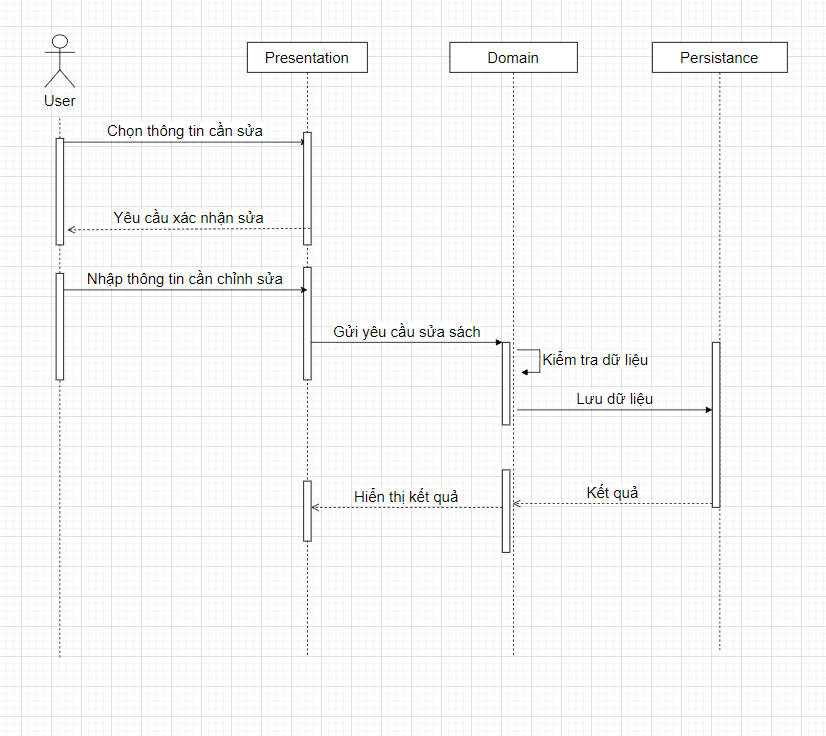
Hình 2. 11: Sơ tuần thêm sách

### **2.3.2 Xoá sách**



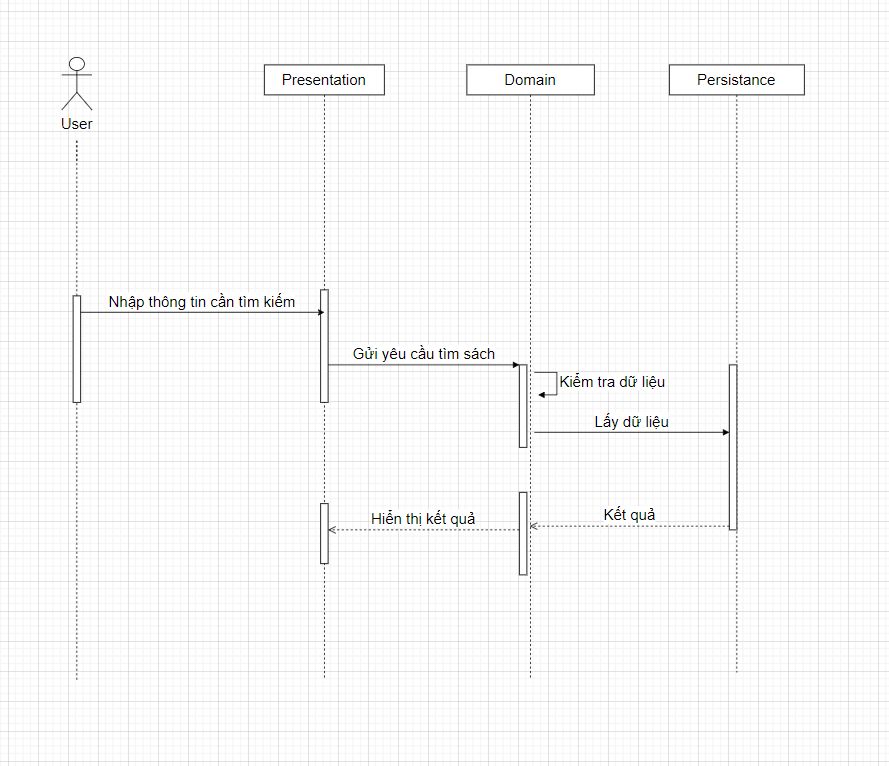
Hình 2. 12: Sơ tuần tự xoá hoá đơn

### **2.3.3. Sửa thông tin sách**



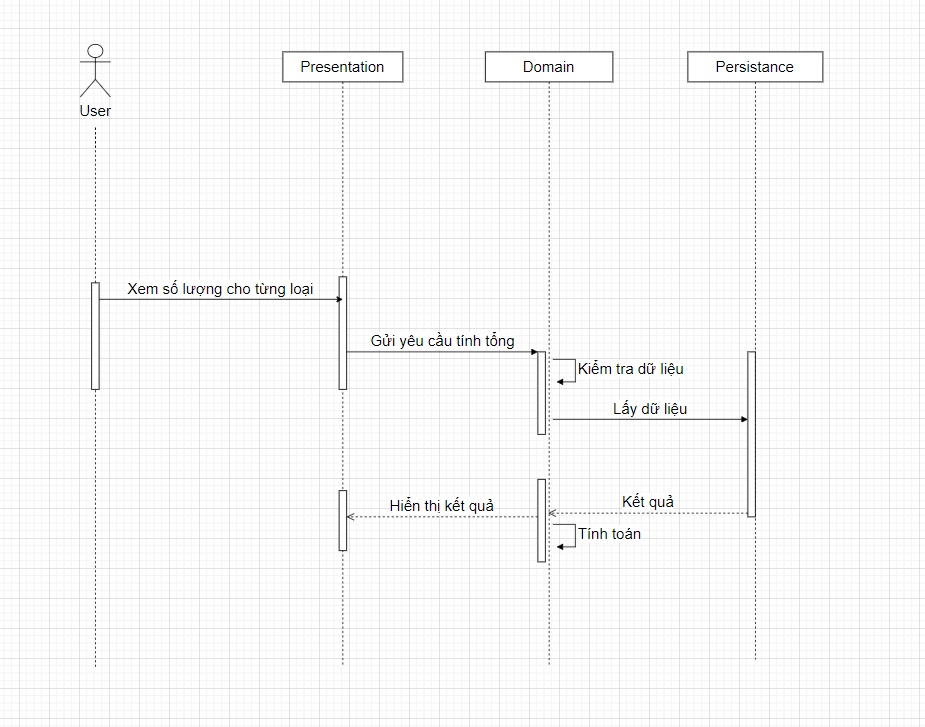
Hình 2. 13: Sơ tuần tự sửa thông tin sách

### **2.3.4 Tìm kiếm sách**



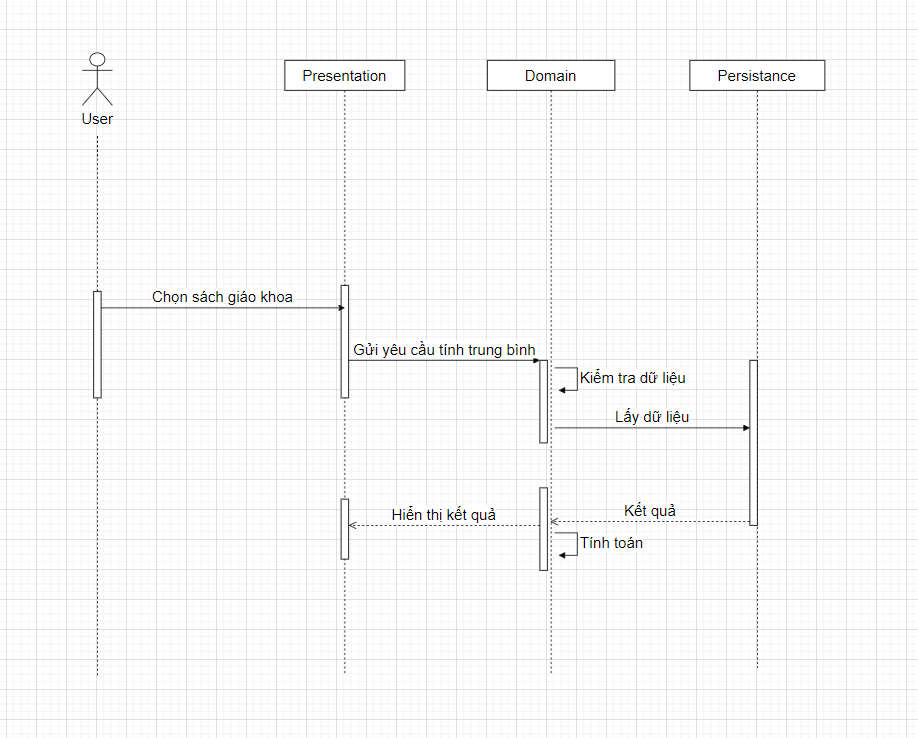
Hình 2. 14: Sơ tuần tự tìm kiếm sách

### **2.3.5 Tính tổng số lượng thành tiền .**



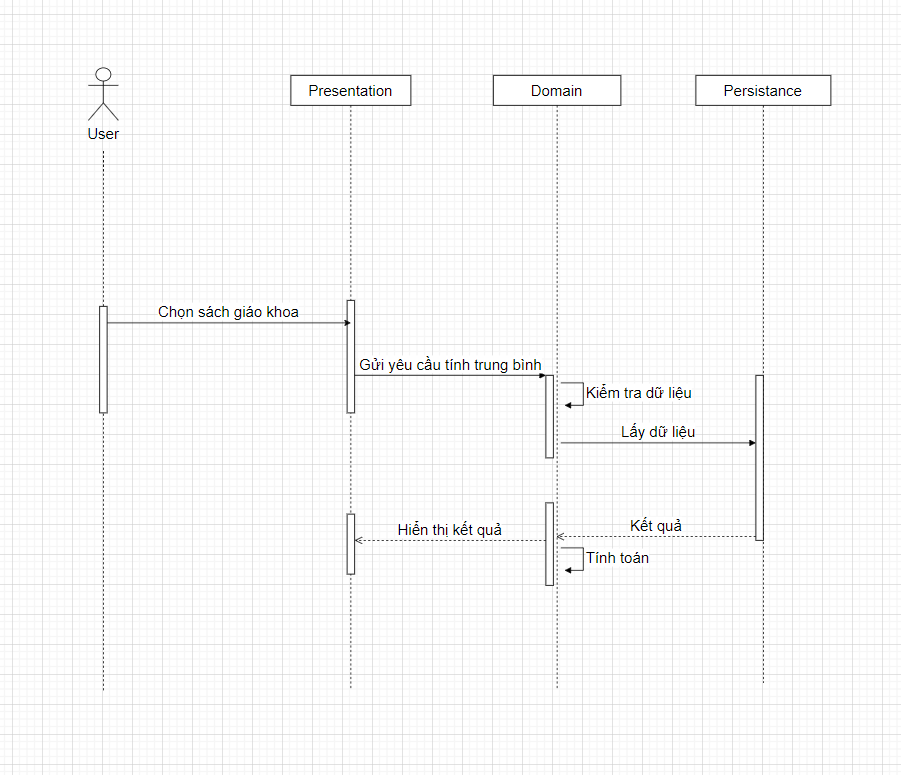
Hình 2. 15: Sơ tuần tự tính tổng thành tiền từng loại

### **2.3.6. Tính trung bình cộng sách giáo khoa.**



Hình 2. 16: Sơ tuần tự tính tổng trung bình sách giáo khoa.

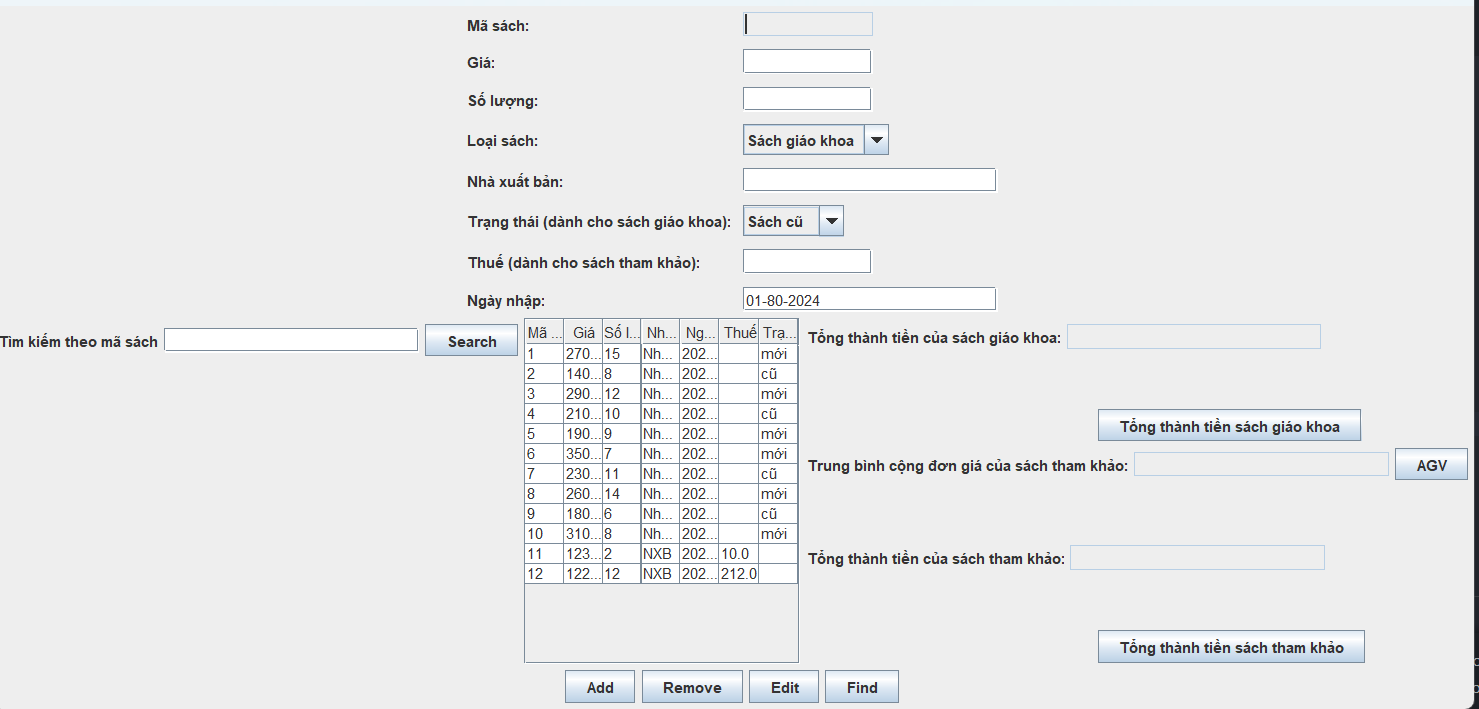
### **2.3.7. Tính trung bình cộng sách tham khảo.**



Hình 2. 17: Sơ tuần tự tính tổng trung bình sách tham khảo.

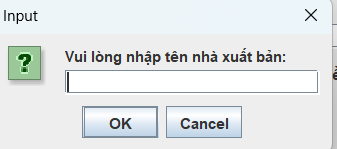
# **CHƯƠNG 3: CÀI ĐẶT THỬ NGHIỆM**

## **3.1. Giao diện chính**



Hình 3. 1: Giao diện chính

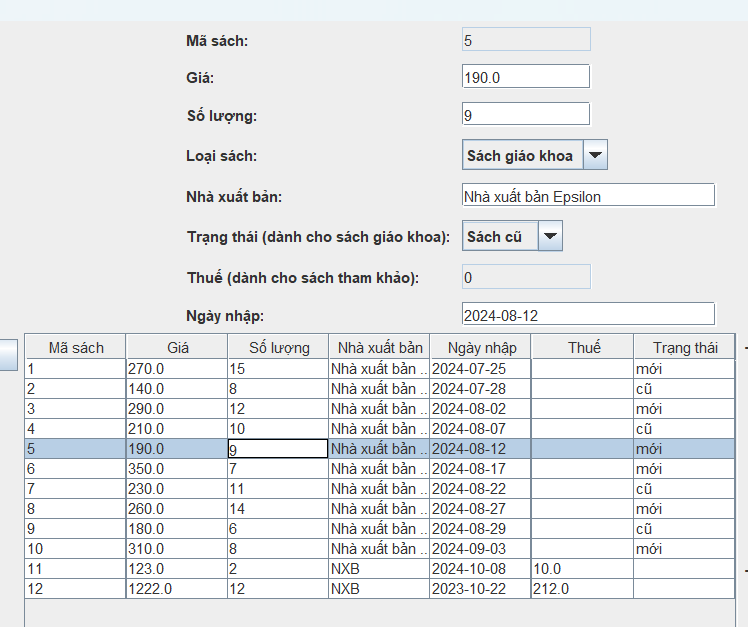
## **3.2. Tìm kiếm sách theo tên Nhà Xuất Bản**



Hình 3.2: Tìm kiếm sách theo tên Nhà Xuất Bản

Mô tả:

* Nhập tên Nhà Xuất Bản
* Chọn ok để tìm kiếm
* Nhập Cancel để thoát.

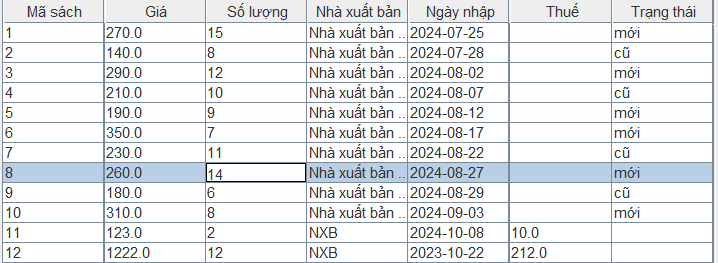
**3.3. Sửa thông tin sách :** 

Hình 3.3: cập nhập lại thông tin sách

Mô tả

* Nhấn Chọn vào thông tin sách cần cập nhập
* Nhập sửa đổi vào ô textbox thông tin tương ứng
* Nhấn chọn edit để thực hiện sửa đổi
* Cập nhập thành công.

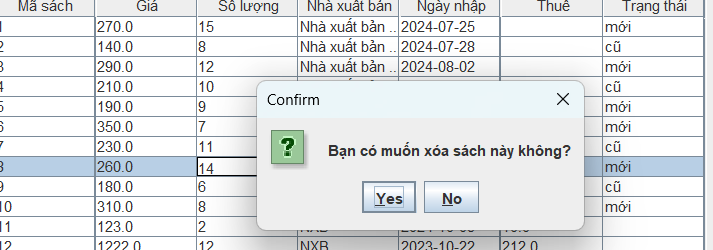
**3.4. Xoá Thông tin sách**

****

Hình 3.4: Xoá thông tin sách

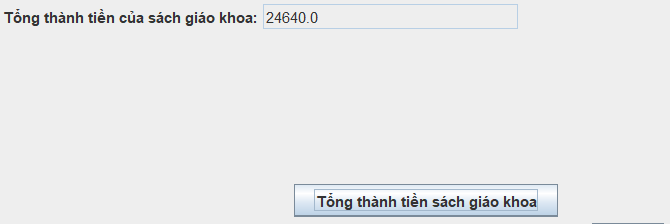
Mô tả:

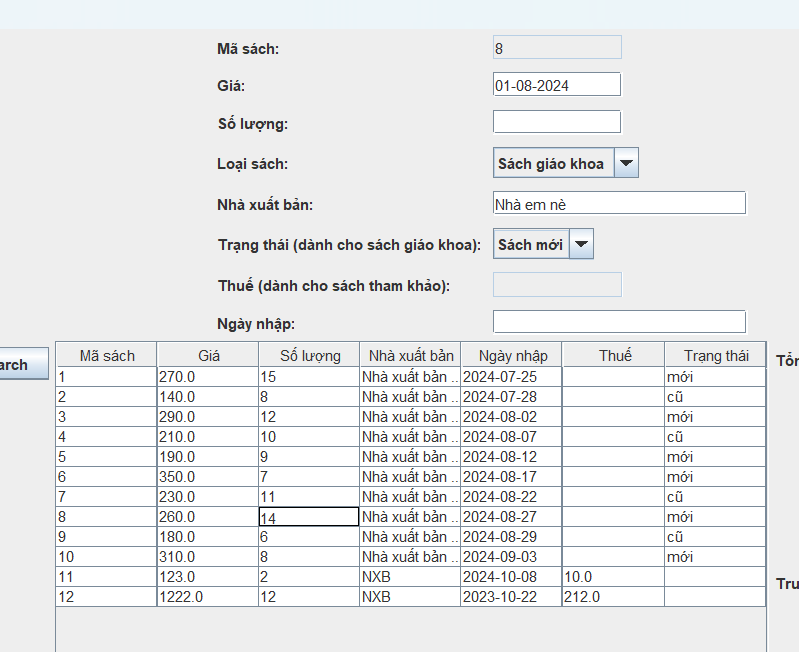
* Nhấn chọn vào thông tin sách cần xóa.
* Chọn button Remove để khởi động chương trình xác nhận
* Nhấn Yes để xác nhận để xóa thông tin sách hoặc No để hủy bỏ.



Hình 3.5: Xác nhận xóa thông tin sách

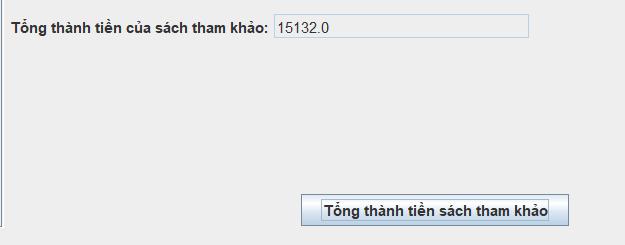
**3.5. Tổng Thành Tiền Sách Giáo Khoa và Sách Tham Khảo.**

****

Hình 3.: Tổng thành tiền của sách giáo khoa

**Mô tả**

* Nhấn chọn tổng thành tiền sách giáo khoa để tính tổng thành tiền của sách giáo khoa



Hình 3.7: Tổng thành tiền của sách tham khảo

**Mô tả:**

* Nhấn button để khỏi động chức năng tổng thành tiền sách giáo tham khảo

**3.6. Trung bình cộng đơn giá của sách tham khảo**



Hình 3.8: Trung bình cộng đơn giá của sách tham khảo

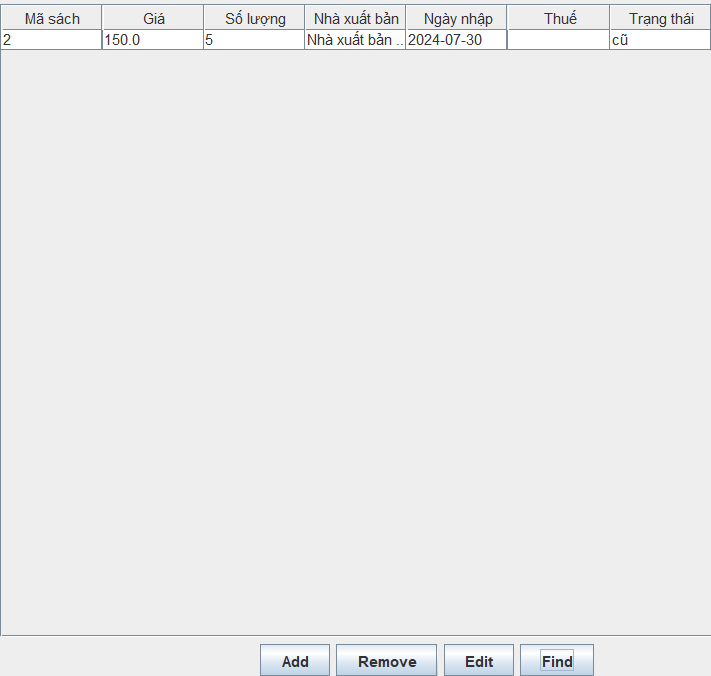
**Mô tả:**

* Nhấn chọn AGV để tính trung bình cộng đơn giá của sách tham khảo

**3.7. Tìm kiếm theo mã sách**



Hình 3.9: Tìm kiếm theo mã sách

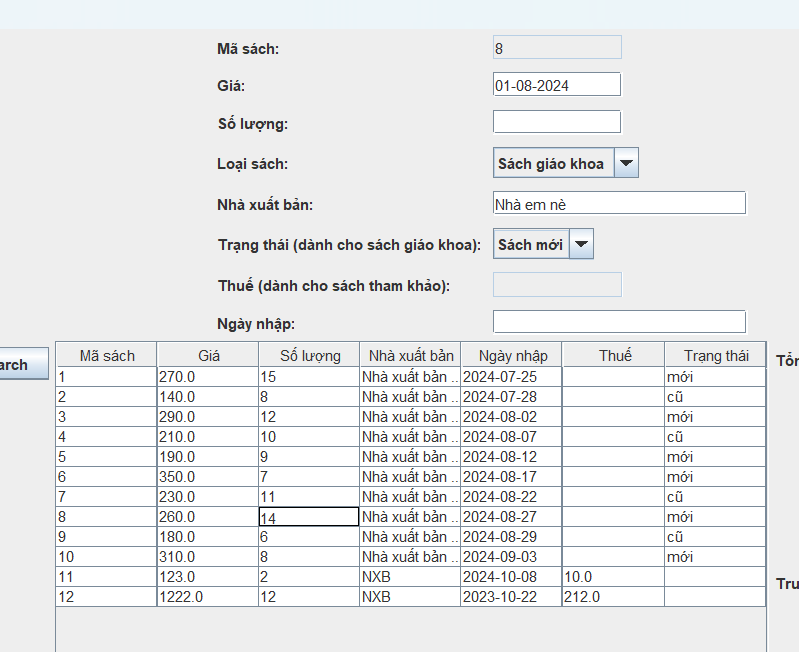


Hình 3.10: Kết quả sau khi tìm kiếm

**Mô tả**

* Nhập vào ô tìm kiếm mã sách
* Nhấn chọn button Find để bắt đầu tìm kiếm
* Kết quả sẽ hiện lên ô danh sách

**3.8. Thêm Thông tin sách**



Hình 3.11: Giao diện nhập thông tin sách cần thêm

**Mô tả**

* Nhập thông tin sách( Giá , Số lượng, Loại sách, Nhà xuất bản, Trạng thái,...) cần thêm
* Nhấn button Add để xác nhận thêm sách vào dữ liệu.
* Thông báo đã thêm dữ liệu thành công

# **CHƯƠNG 4. KẾT LUẬN VÀ ĐÁNH GIÁ**

## **4.1. Kết quả đạt được**

Trong quá trình phát triển ứng dụng quản lý sách, tôi đã áp dụng nhiều mẫu thiết kế phần mềm và đạt được những kết quả đáng kể. Việc sử dụng các mẫu thiết kế này giúp tôi tổ chức mã nguồn một cách rõ ràng hơn, tách biệt các thành phần và giảm thiểu sự phụ thuộc giữa chúng. Cụ thể:

**-Mô hình 3-layer**: Đã giúp tôi phân chia mã nguồn thành các phần rõ ràng, bao gồm giao diện người dùng, xử lý nghiệp vụ và truy cập dữ liệu. Điều này cải thiện tính tổ chức và dễ bảo trì của mã nguồn, đồng thời giảm thiểu sự phụ thuộc giữa các thành phần của ứng dụng.

**-Mẫu thiết kế Observer**: Cho phép ứng dụng phản hồi nhanh chóng với các thay đổi dữ liệu, đảm bảo giao diện luôn hiển thị thông tin mới nhất mà không cần phải làm mới toàn bộ ứng dụng. Điều này nâng cao trải nghiệm người dùng và đảm bảo tính nhất quán của thông tin.

Các mẫu thiết kế này đã mang lại tính linh hoạt, dễ bảo trì và cải thiện hiệu suất cho ứng dụng. Đồng thời, việc áp dụng chúng cũng đã giúp tôi hiểu sâu hơn về cách tổ chức và phát triển phần mềm một cách chuyên nghiệp, tối ưu hóa việc quản lý tài nguyên và nâng cao trải nghiệm người dùng. Qua quá trình này, tôi đã củng cố kiến thức về các mẫu thiết kế và nâng cao kỹ năng lập trình của mình, góp phần tạo ra một ứng dụng chất lượng cao và hiệu quả.

## **4.2. Đánh giá ưu điểm và nhược điểm**

**4.2.1. Ưu điểm**

Thực hiện hóa đa số giao diện của đề bài yêu cầu.

Giao diện tối giản, dễ sử dụng.

**4.2.2. Nhược điểm**

Giao diện chưa được tối ưu.

Chưa đáp ứng được việc sử dụng trên nhiều thiết bị.

Hệ thống thô sơ, thiếu tính bảo mật an toàn.

Vẫn còn tính năng không hoạt động đúng theo yêu cầu.

## **4.3. Tổng kết**

Sau khi hoàn thành bài thiết kế ứng dụng quản lý sách, tôi đã đạt được những kết quả đáng kể nhờ vào việc áp dụng các mẫu thiết kế phần mềm. Cụ thể:

**Mô hình 3-layer** đã giúp tôi tổ chức mã nguồn một cách rõ ràng, phân chia thành ba lớp riêng biệt: giao diện người dùng, xử lý nghiệp vụ và truy cập dữ liệu. Điều này không chỉ cải thiện tính tổ chức và dễ bảo trì của mã nguồn mà còn giúp giảm thiểu sự phụ thuộc giữa các thành phần của ứng dụng.

**Mẫu Observer** cho phép ứng dụng tự động cập nhật giao diện người dùng khi có sự thay đổi dữ liệu, mà không cần phải làm mới toàn bộ ứng dụng. Điều này nâng cao trải nghiệm người dùng và đảm bảo tính nhất quán của thông tin.

Ngoài việc tối ưu hóa hiệu suất và quản lý tài nguyên, các mẫu thiết kế đã mang lại sự linh hoạt cho ứng dụng, giúp ứng dụng phản hồi nhanh chóng và hiệu quả hơn với các thay đổi dữ liệu. Quá trình áp dụng các mẫu thiết kế này không chỉ giúp tôi củng cố kiến thức về cách tổ chức và phát triển phần mềm một cách chuyên nghiệp mà còn nâng cao kỹ năng lập trình của mình. Nhờ vậy, tôi đã tạo ra một ứng dụng chất lượng cao và hiệu quả, đồng thời nâng cao trải nghiệm người dùng bằng giao diện thân thiện và khả năng phản hồi kịp thời với các thay đổi.

# **TÀI LIỆU THAM KHẢO**

1. Đỗ Xuân Lâm, Nguyễn Thanh Bình. *Mẫu Thiết Kế Hướng Đối Tượng*. Nhà xuất bản Đại học Quốc gia TP.HCM, 2019.
2. Trần Đình Quế. *Kỹ Thuật Lập Trình Hướng Đối Tượng*. Nhà xuất bản Bách Khoa, 2015.
3. Học viện Công nghệ Bưu chính Viễn thông. "Báo cáo về kiến trúc 3 tầng". Truy cập ngày 29 tháng 7 năm 2024. <https://www.ptit.edu.vn/>
4. Trường Đại học Bách Khoa Hà Nội. "Các mẫu thiết kế phần mềm". Truy cập ngày 29 tháng 7 năm 2024. <https://www.hust.edu.vn/>
5. Nguyễn Văn Ngọc. *Kiến Trúc Phần Mềm*. Nhà xuất bản Khoa học và Kỹ thuật, 2015.
6. Trần Đình Quế. *Thiết Kế Hệ Thống Thông Tin*. Nhà xuất bản Bách Khoa Hà Nội, 2017.