**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NGOẠI NGỮ-TIN HỌC THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**Logo

Description automatically generated**

**BÁO CÁO MÔN HỌC: MẪU THIẾT KẾ**

**ĐỀ TÀI: WEB BÁN SÁCH**

**GIẢNG VIÊN HƯỚNG DẪN: Lương Văn Minh**

**SINH VIÊN THỰC HIỆN:**

**Đỗ Trần Quốc Duy – 18DH110880**

**TP.HỒ CHÍ MINH - 2021**

LỜI CẢM ƠN

🙞🕮🙜

Để hoàn thành đồ án này trước hết em xin gửi đến quý thầy, cô giáo trong khoa công nghệ phần mềm. trường Đại học ngoại ngữ tin học lời cảm ơn chân thành, em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô, gia đình và bạn bè.

Em xin gửi lời cảm ơn chân thành đến thầy Lương Văn Minh, giảng viên Bộ môn Công nghệ thông tin- người đã tận tình hướng dẫn, chỉ bảo em trong suốt quá trình làm khoá luận.

Em cũng xin chân thành cảm ơn các thầy cô giáo trong khoa công nghệ phần mềm. trường Đại học ngoại ngữ tin học nói chung, các thầy cô trong Bộ môn Mẫu Thiết Kế nói riêng đã dạy dỗ cho em kiến thức từ căn bản đến chuyên môn giúp em có được cơ sở lý thuyết vững vàng và tạo điều kiện giúp đỡ em trong suốt quá trình học tập.

Cuối cùng, em xin chân thành cảm ơn gia đình và bạn bè, đã luôn tạo điều kiện, quan tâm, giúp đỡ, động viên em trong suốt quá trình học tập và hoàn thành báo cáo đồ án.

***TP. Hồ Chí Minh, tháng 12 năm 2021***

MỤC LỤC

[Chương 1. Tổng Quan Đề Tài 3](#_Toc7724)

[1. Bối cảnh 3](#_Toc11999)

[2. Công nghệ sử dụng 3](#_Toc6820)

[3. Chức năng phần mềm 4](#_Toc22122)

[Chương 2. Mẫu Thiết kế 5](#_Toc23498)

[1. Giới thiệu design pattern 5](#_Toc16325)

[2. Creational patterns (nhóm khởi tạo) 6](#_Toc7411)

[3. Structural patterns (nhóm cấu trúc) 7](#_Toc31162)

[4. Behavioral patterns (nhóm hành vi/ tương tác) 8](#_Toc5990)

[5. Ưu điểm và nhược điểm khi sử dụng design pattern 9](#_Toc12173)

[Chương 3. Áp dụng mẫu thiết kế 10](#_Toc11271)

[1. Singleton Pattern Design-(Creational Pattern) 10](#_Toc24191)

[2. Template Pattern Design -(Behavior Pattern) 15](#_Toc11091)

[3. Prototype Pattern Design 16](#_Toc17487)

[4. Iterator Pattern Design 18](#_Toc13922)

[5. Facade Pattern Design 20](#_Toc14019)

[6. Decorator Pattern Design 24](#_Toc18628)

[7. Proxy Pattern Design 27](#_Toc21635)

[Chương 4. Giao diện website bán sách 31](#_Toc14962)

[1. Trang web User 31](#_Toc16847)

[2. Trang web Admin 35](#_Toc16945)

[KẾT LUẬN 39](#_Toc30671)

[1. Những vấn đề đã làm được 39](#_Toc1043)

[TÀI LIỆU THAM KHẢO 40](#_Toc9115)

# Tổng Quan Đề Tài

## Bối cảnh

. Tiếp nhận kiến thức là một việc rất quan trọng trong đó việc đọc sách là không thể thiếu.Việc đọc sách giúp bản thân sẽ nhìn nhận những nội dung trong sách theo các khía cạnh khác nhau. Tầm quan trọng của sách là việc có thể giúp cho người đọc phát huy sự sáng tạo, áp dụng những kiến thức trong sách vào ứng dụng thực tế của bản thân để xử lý trong các tình huống khác nhau.Dù xã hội có phát triển đến đâu, các phương thức lưu giữ thông tin khác có thể phát triển nhưng việc lưu giữ sách và hiểu được tầm quan trọng của sách sẽ giúp cho xã hội ngày càng văn minh, tiến bộ hơn. Đọc sách có chọn lọc và hiểu hết các giá trị trong cuốn sách đó sẽ giúp người đọc ngày càng phát triển được bản thân.

Vì vậy, em đã quyết định thực hiện đề tài “Xây dựng Website bán sách online”. -Vinabook. Việc xây dựng website sẽ giúp cho công việc kinh doanh trở nên thuận lợi và dễ dàng hơn, đáp ứng nhu cầu làm việc mọi lúc, mọi nơi, đồng thời thỏa mãn nhu cầu xem thông tin, mua bán sách online… của mọi đối tượng khách hàng.

## Công nghệ sử dụng

Xây dựng và thiết kế website bán sách theo ngôn ngữ lập trình c#, cơ sở dữ liệu sql server, framwork asp.Net trên nền tảng. Xây dựng những công nghệ đưa vào nhằm mục đích giúp cho lập trình viên dễ dàng tiếp cận nhanh chóng với công nghệ thực tiễn không phải nhập nhằn trong việc lựa chọn. ASP.NET là back-end framework lớn thứ hai trên internet, chỉ sau PHP. Đây là ứng dụng dịch vụ của Microsoft dành cho các trang web động và C# là ngôn ngữ chính được sử dụng để lập trình với ASP.NET framework. Website được tạo ra ngày càng phổ biến nó sẽ tiết kiệm nhân lực, chi phí khách hàng lẫn cả lập trình viên hiệu quả, nhanh chóng, mạnh mẽ

Website bán sách VinaBook được sử dụng theo mô hình mvc gồm có model, controller, View. Trong đó Model là bộ phận có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng. View là phần giao diện (theme) dành cho người sử dụng. Controller là Bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua view, đưa ra dữ liệu phù hợp với người dùng.

## Chức năng phần mềm

* Đăng nhập, đăng kí, giỏ hàng
* Dịch vụ chăm sóc và hỏi đáp
* Thanh toán
* Tìm kiếm tra cứu
* Giao diện màn hình trang chủ bán sách
* Quản lí sách
* Quản lí khách hàng
* Quản lí nhà xuất bản
* Quản lí khuyến mãi
* Quản lí đơn hàng
* Quản lí chủ đề

# Mẫu Thiết kế

1. Giới thiệu design pattern

Design Patterns (mẫu thiết kế) là một kỹ thuật trong lập trình hướng đối tượng, nó khá quan trọng là một giải pháp tổng thể cho các vấn đề chung trong thiết kế phần mềm. Design Pattern được sử dụng thường xuyên trong các ngôn ngữ OOP.

Các mẫu thiết kế hướng đối tượng thường cho thấy mối quan hệ và sự tương tác giữa các lớp hay các đối tượng, mà không cần chỉ rõ các lớp hay đối tượng của từng ứng dụng cụ thể.

**Tại sao cần học design pattern ?**

* Thay đổi code cũ mà không phát sinh các lỗi mới hoặc các bugs ko mong muốn.
* Cung cấp các mẫu thiết kế có thể áp dụng vào thiết kế của bạn và giải quyết các vấn đề chung.
* Phát triển phần mềm trở nên dễ dàng hơn
* Tránh lặp code, thiết kế không chặt chẽ và nhiều nguy cơ tiềm ẩn trong một số class mà bạn sử dụng.

-Design pattern có tổng cộng 32 mẫu thiết kế và được chia làm 3 nhóm chính:

+Creational Pattern (nhóm khởi tạo – 5 mẫu)

+Structural Pattern (nhóm cấu trúc – 7 mẫu)

+Behavioral Pattern (nhóm tương tác/ hành vi – 11 mẫu)

1. Creational patterns (nhóm khởi tạo)

CP Là các mẫu thiết kế đối ứng với các cơ chế tạo đối tượng, cố gắng tạo các đối tượng theo cách phù hợp với từng tình huống nhất định

Creational Patterns có liên quan đến việc cách khởi tạo các object. Chúng giảm độ phức tạp và bất ổn khi tạo các objects trong 1 cách thức kiểm soát.

* Singleton - Đảm bảo rằng chỉ có duy nhất 1 instance của object tồn tại suốt ứng dụng.
* Factory Method - Tạo các đối tượng của một số classes liên quan mà không chỉ định chính xác đối tượng được tạo.
* Abstract Factory - Tạo ra các đối tượng phụ thuộc liên quan giống nhau.
* Builder - Xây dựng các đối tượng phức tạp bằng cách sử dụng phương pháp từng bước.
* Prototype - khởi tạo một đối tượng bằng cách clone một đối tượng đã tồn tại thay vì khởi tạo với từ khoá new

1. Structural patterns (nhóm cấu trúc)

SP là các pattern giải quyết các vấn đề liên quan tới cách tổ chức các lớp, đối tượng sao cho linh hoạt, ngăn nắp để dễ dàng thay đổi, hay mở rộng code sau này.

Structural Patterns giúp đơn giản việc thiết kế bằng cách xác định cách đơn giản để thể hiện được các mối quan hệ giữa các thực thể

* Adapter: là kết nối giao diện của các lớp khác nhau.
* Bridge: phân cách giao diện của các đối tượng từ việc triển khai.
* Composite: cấu trúc cây sự đơn giản và phức hợp của đối tượng.
* Decorator: bổ sung trách nhiệm cho các đối tượng động.
* Facade: một lớp đơn đại diện cho toàn bộ một hệ thống con.
* Private Class Data: hạn chế truy cập accessor/mutator.
* Proxy: một đối tượng đại diện cho đối tượng khác.

1. Behavioral patterns (nhóm hành vi/ tương tác)

* BP Là các mẫu thiết kế xác định và định nghĩa các mẫu giao tiếp chung giữa các đối tượng
* Behavioral Patterns có liên quan sẽ giúp bạn quản lý hành vi, trách nhiệm của các lớp dễ dàng hơn..
* Chain of responsibility - là phương án để chuyển yêu cầu giữa một chuỗi các đối tượng.
* Command - cho phép chuyển yêu cầu thành đối tượng độc lập, có thể được sử dụng để tham số hóa các đối tượng với các yêu cầu khác nhau như log, queue (undo/redo), transtraction
* Interpreter - giúp người lập trình có thể “xây dựng” những đối tượng “động” bằng cách đọc mô tả về đối tượng rồi sau đó “xây dựng” đối tượng đúng theo mô tả đó.
* Iterator - truy cập liên tục các thành phần của bộ lựa chọn.
* Mediator - sử dụng để giảm sự phức tạp trong "giao tiếp" giữa các lớp và các đối tượng.
* Memento - cho phép chúng ta lưu trữ và khôi phục trạng thái của một đối tượng mà không tiết lộ chi tiết bên trong
* Observer - duy trì một danh sách các thành phần phụ thuộc nó và thông báo tới chúng một cách tự động về bất cứ thay đổi nào.
* State - thay đổi hành vi của đối tượng khi trạng thái thay đổi
* Strategy - Đóng gói thuật toán bên trong một lớp.
* Template method - Định nghĩa bộ khung gồm các bước chính xác của một thuật toán cho từng phân lớp
* Visitor - Cho phép thay đổi, mở rộng các thao tác cho đối tượng mà không thay đổi cấu trúc, nội dung bên trong đối tượng

1. Ưu điểm và nhược điểm khi sử dụng design pattern

**Ưu điểm**

* giảm được thời gian và công sức suy nghĩ ra các cách giải quyết cho những vấn đề đã có lời giải.
* giúp chương trình chạy uyển chuyển hơn, dễ dàng quản lý tiến trình hoạt động, dễ nâng cấp bảo trì, …

**Nhược điểm**

* design pattern là nó luôn là một lĩnh vực khá khó nhằn và hơi trừu tượng
* Khi sử dụng những mẫu design pattern có sẵn thì code sẽ chạy chậm

# Áp dụng mẫu thiết kế

1. Singleton Pattern Design-(Creational Pattern)

Singleton đảm bảo chỉ duy nhất một thể hiện (instance) được tạo ra và nó sẽ cung cấp cho bạn một method để có thể truy xuất được thể hiện duy nhất đó mọi lúc mọi nơi trong chương trình

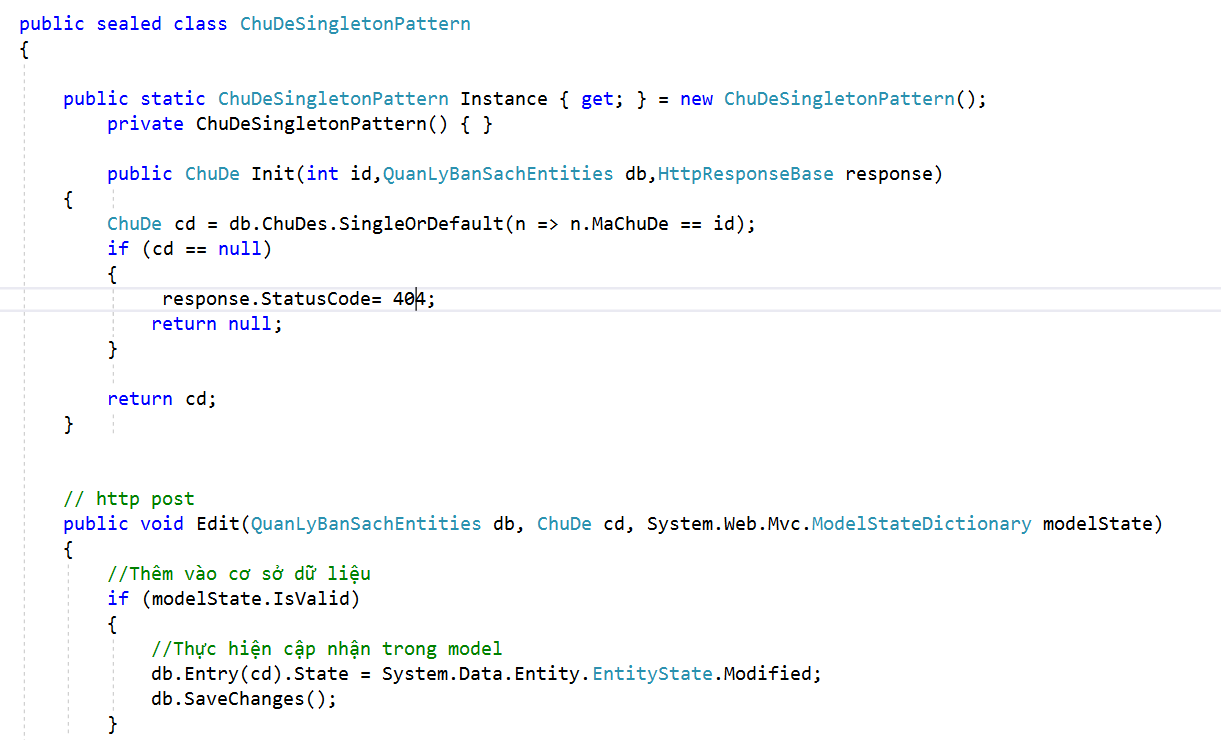
Sử dụng singleton mục đích là:

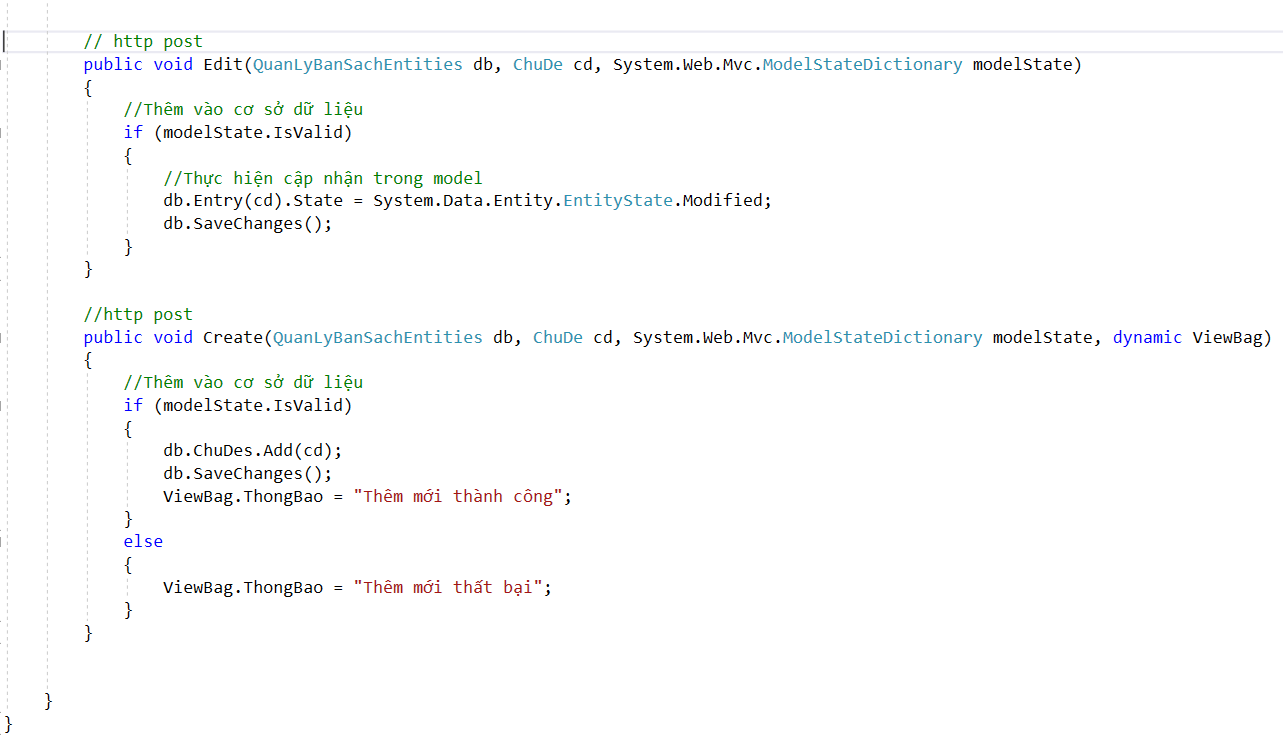
* Đảm bảo rằng chỉ có một instance của lớp.
* Việc quản lý việc truy cập tốt hơn vì chỉ có một thể hiện duy nhất.
* Có thể quản lý số lượng thể hiện của một lớp trong giới hạn chỉ định.

Ví dụ : Quản lí chủ đề, khuyến mãi, nhà sản xuất, sách,…..

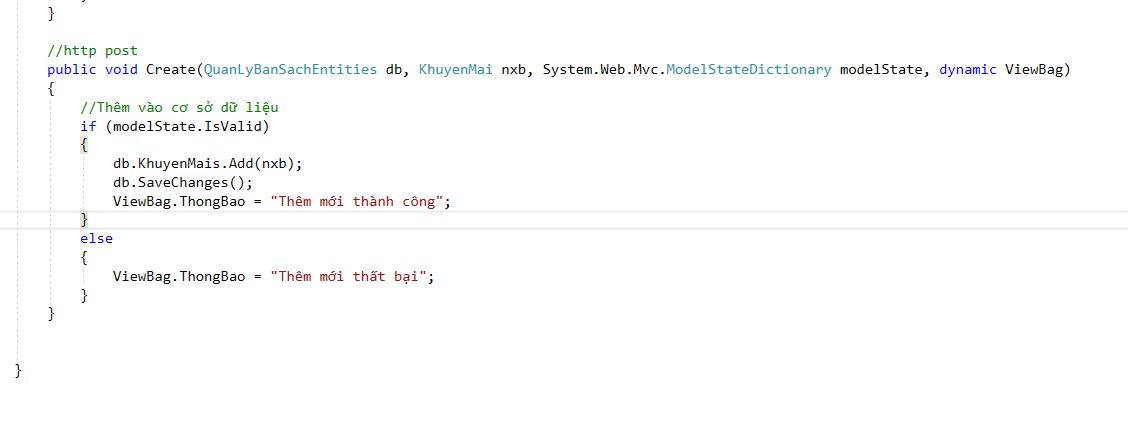
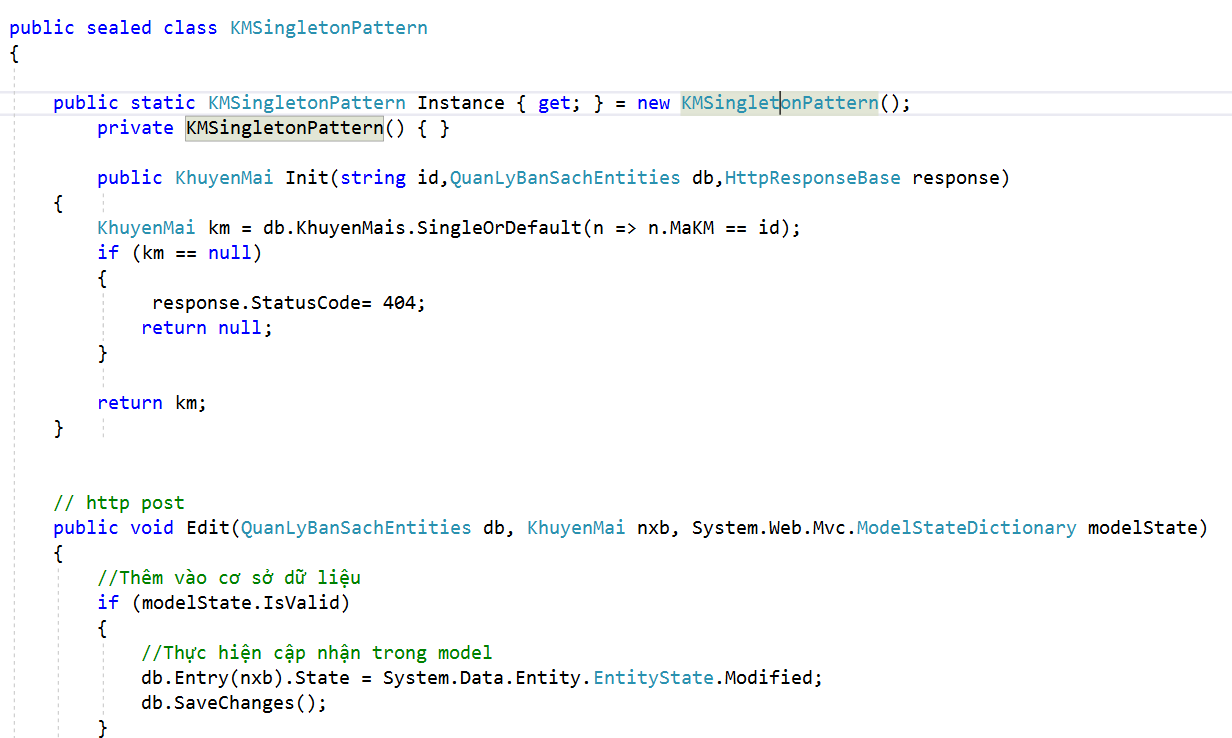
**Một số đoạn code:**

##### Quản lí chủ đề





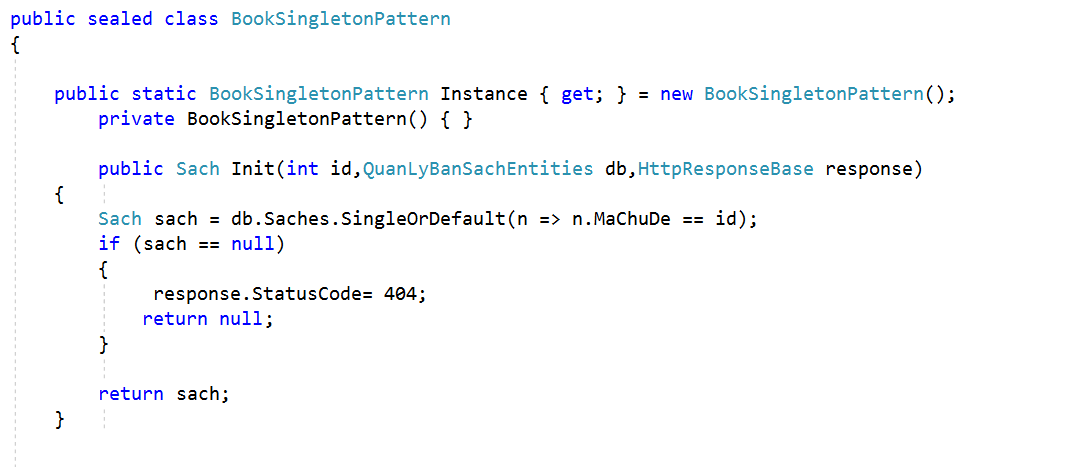
##### Quản lí khuyến mãi

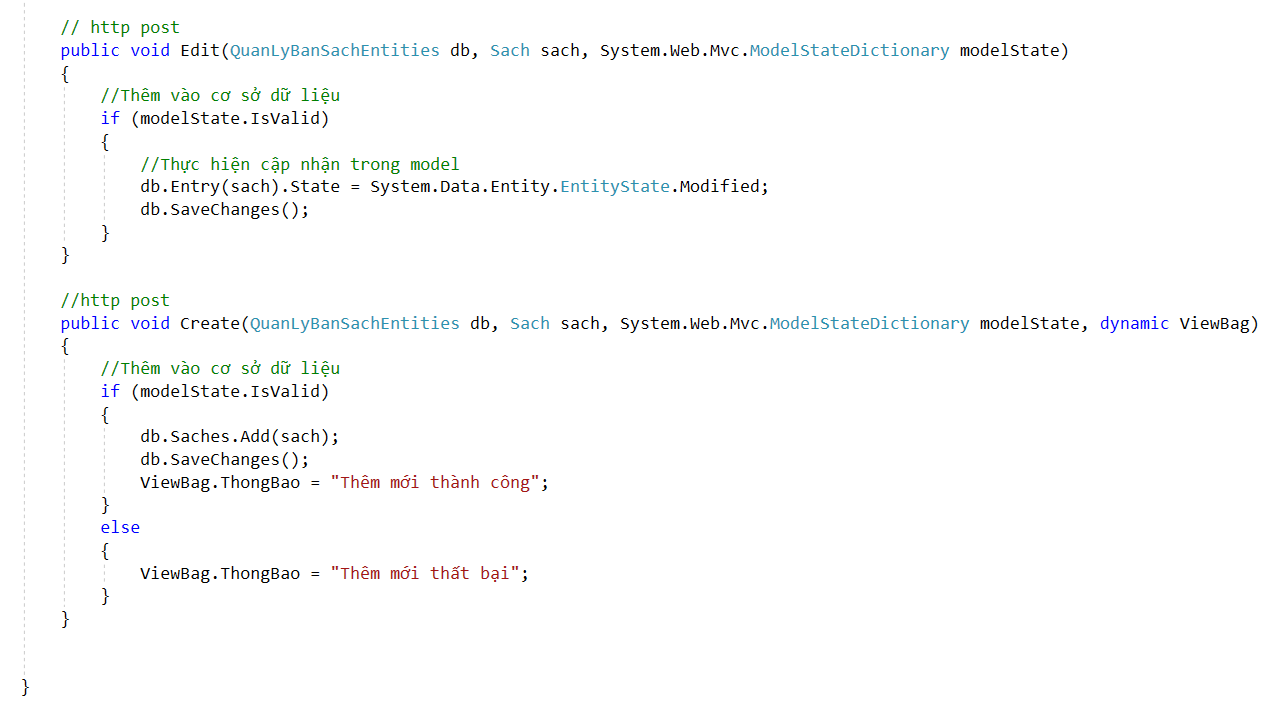


##### Quản lí NSX



##### Quản lí sách

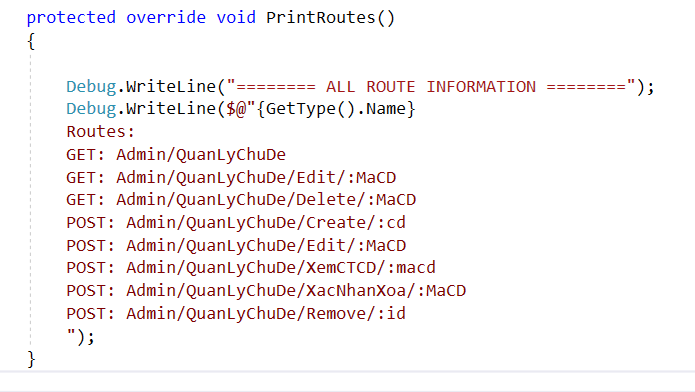
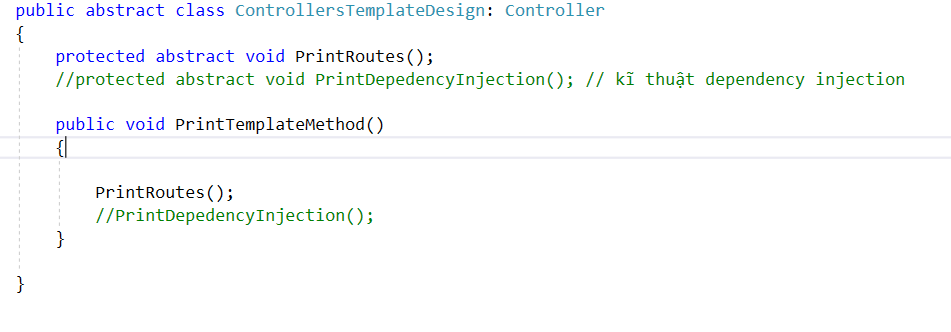




1. Template Pattern Design -(Behavior Pattern)

Template method giúp cho chúng ta tạo nên một bộ khung (template) cho một vấn đề đang cần giải quyết. Trong đó các đối tượng cụ thể sẽ có cùng các bước thực hiện, nhưng trong mỗi bước thực hiện đó có thể khác nhau.

Sử dụng template mục đích là:

* Khi có một thuật toán với nhiều bước và mong muốn cho phép tùy chỉnh trong lớp con.
* Mong muốn chỉ có một triển khai phương thức trừu tượng duy nhất của một thuật toán.
* Mong muốn hành vi chung giữa các lớp con nên được đặt ở một lớp chung.
* Các lớp cha có thể gọi các hành vi trong các lớp con của chúng một cách thống nhất (step by step).  
  ví dụ :tạo ra controllers Design Pattern
* 

1. Prototype Pattern Design

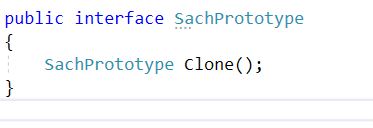
Khởi tạo một đối tượng bằng cách clone một đối tượng đã tồn tại thay vì khởi tạo với từ khoá new. Đối tượng mới là một bản sao có thể giống 100% với đối tượng gốc, chúng ta có thể thay đổi dữ liệu của nó mà không ảnh hưởng đến đối tượng gốc.

Prototype Pattern được dùng khi việc tạo một object tốn nhiều chi phí và thời gian trong khi bạn đã có một object tương tự tồn tại.

Sử dụng prototype mục đích là:

* Giảm độ phức tạp cho việc khởi tạo đối tượng: do mỗi lớp chỉ implement cách clone của chính nó.
* Giảm việc phân lớp, tránh việc tạo nhiều lớp con cho việc khởi tạo đối tượng
* Khởi tạo object mới bằng cách thay đổi một vài thuộc tính của object

Ví dụ tạo chức năng sao chép sách tương tư



1. Iterator Pattern Design

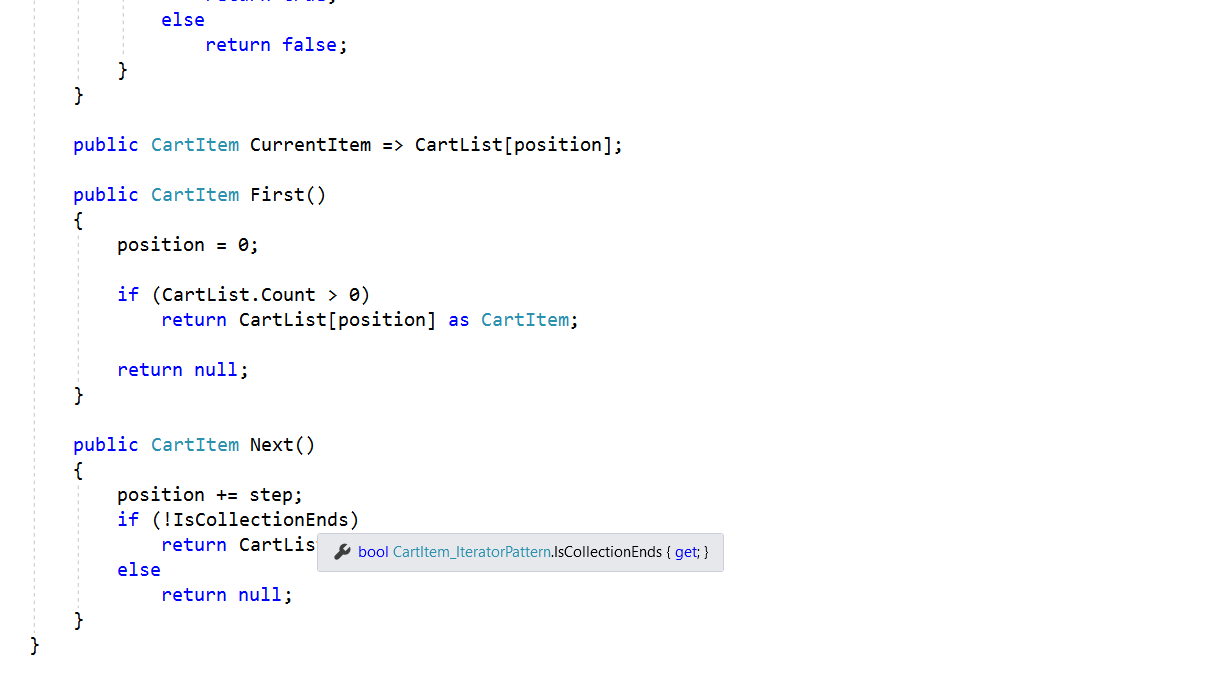
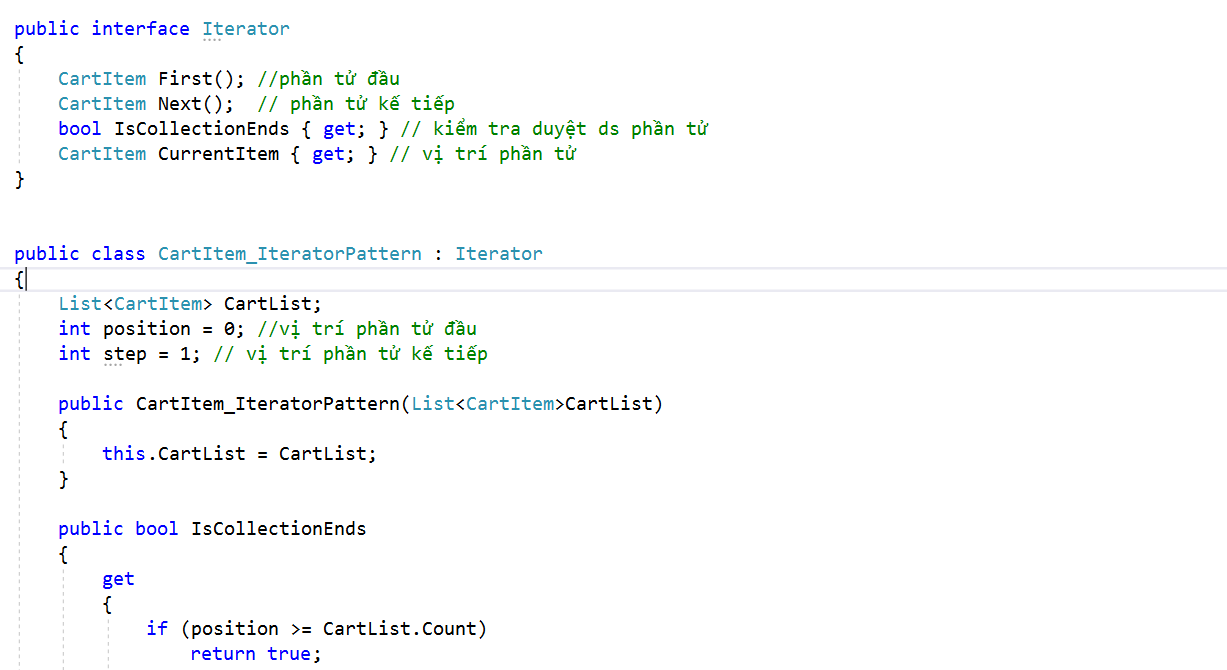
Cung cấp một cách thức truy cập tuần tự tới các phần tử của một đối tượng tổng hợp, mà không cần phải tạo dựng riêng các phương pháp truy cập cho đối tượng tổng hợp này

Prototype Pattern được dùng với ý tưởng thiết kế này là một trong những kỹ thuật được gọi là “đơn trách nhiệm – Single responsibility principle (SRP)” – một lớp chỉ có duy nhất một công việc để làm.

Sử dụng Iterator mục đích là:

* Đảm bảo nguyên tắc Single responsibility principle (SRP) .
* Đảm bảo nguyên tắc Open/Closed Principle (OCP)
* Chúng ta có thể truy cập song song trên cùng một tập hợp vì mỗi đối tượng iterator có chứa trạng thái riêng của nó.

Ví dụ tạo chức năng thêm giỏ hàng



1. Facade Pattern Design

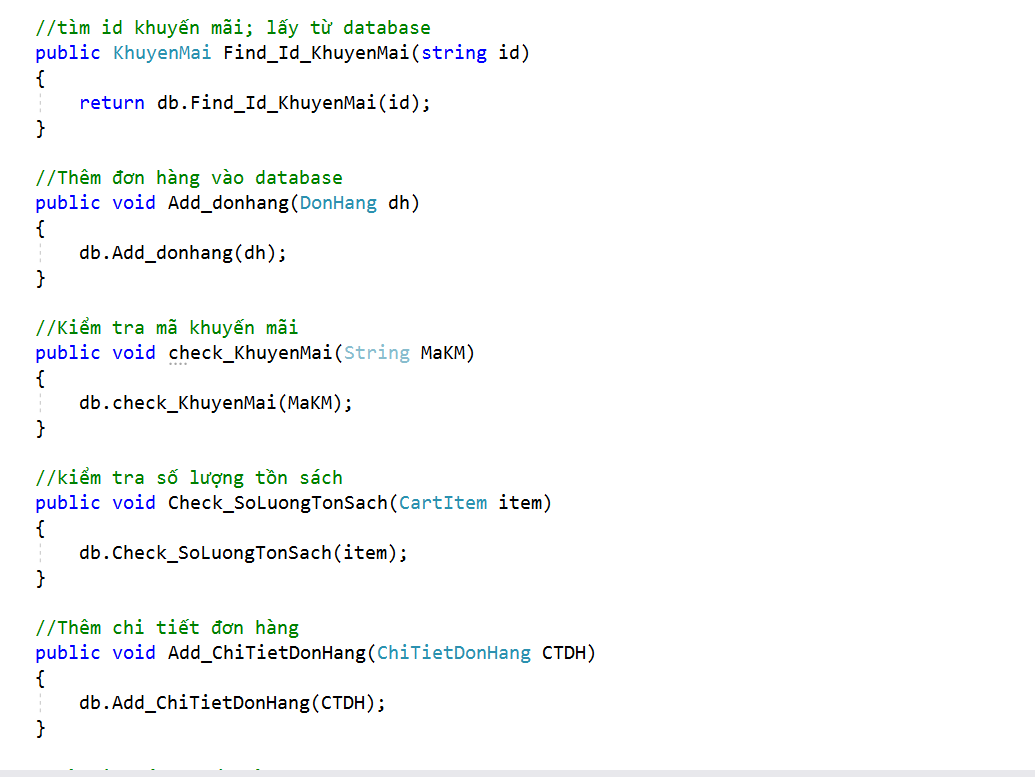
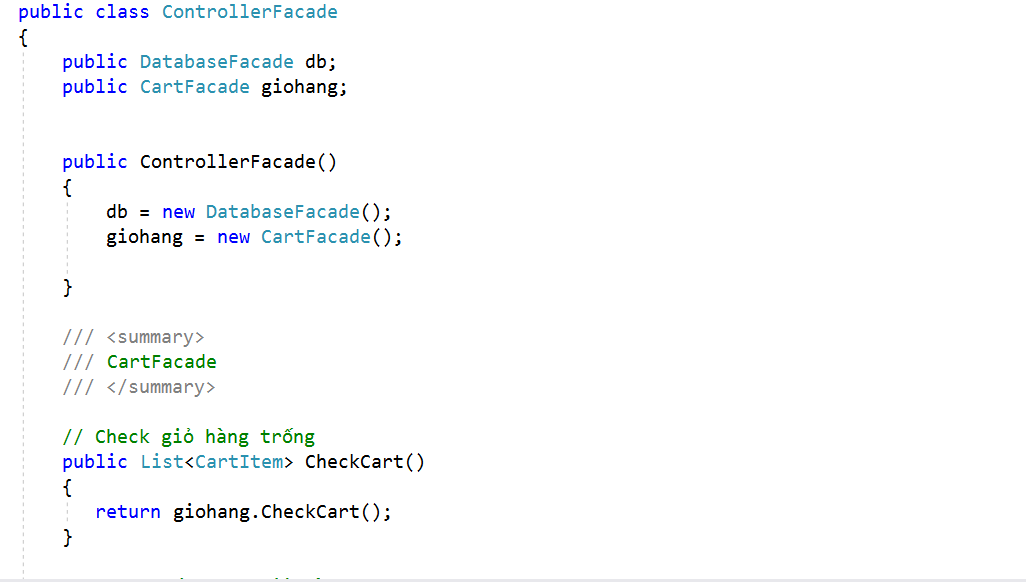
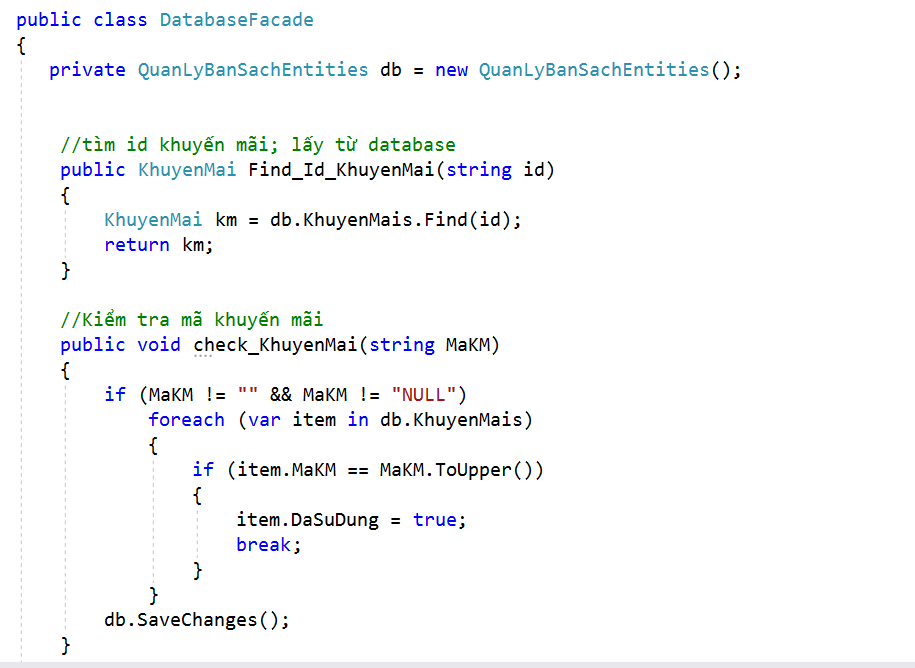
Che giấu các hoạt động phức tạp bên trong hệ thống con, làm cho hệ thống con dễ sử dụng hơn.

Facade Pattern được dùng với ý tưởng thiết kế này là một trong những kỹ thuật được gọi là “đơn trách nhiệm – Single responsibility principle (SRP)” – một lớp chỉ có duy nhất một công việc để làm.

Sử dụng Facade mục đích là:

* Giúp cho hệ thống của bạn trở nên đơn giản hơn trong việc sử dụng và trong việc hiểu nó
* Giảm sự phụ thuộc của các mã code bên ngoài với hiện thực bên trong của thư viện
* Đóng gói tập nhiều hàm API được thiết kế không tốt bằng một hàm API đơn có thiết kế tốt hơn.

Ví dụ tạo chức năng hệ thống giỏ hàng và dữ liệu database



1. Decorator Pattern Design

Cho phép người dùng thêm chức năng mới vào đối tượng hiện tại mà không muốn ảnh hưởng đến các đối tượng khác. Kiểu thiết kế này có cấu trúc hoạt động như một lớp bao bọc (wrap) cho lớp hiện có.

Decorator pattern sử dụng composition thay vì inheritance (thừa kế) để mở rộng đối tượng. Decorator pattern còn được gọi là Wrapper hay Smart Proxy.

Sử dụng Decorator mục đích là:

* Một đối tượng có thể được bao bọc bởi nhiều wrapper cùng một lúc. .
* Cho phép thêm hoặc xóa tính năng của một đối tượng lúc thực thi
* khả năng mở rộng của đối tượng .bằng cách implement trên các lớp mới.

Không thấy sự đặc biệcđưa cho nó một wrapper thay vì đối tượng gốc.

Ví dụ : chức năng chọn loại sách theo loại khuyến mãi, giảm giá, mới về,…



1. Proxy Pattern Design

Mục đích xây dựng Proxy pattern cũng chính vì muốn tạo ra một đối tượng sẽ ủy quyền, thay thế cho một đối tượng khác.

Proxy Pattern mẫu thiết kế mà ở đó tất cả các truy cập trực tiếp đến một đối tượng nào đó sẽ được chuyển hướng vào một đối tượng trung gian (Proxy Class). Mẫu Proxy (người đại diện) đại diện cho một đối tượng khác thực thi các phương thức, phương thức đó có thể được định nghĩa lại cho phù hợp với múc đích sử dụng.

Phân loại proxy gồm có:

+Virtual Proxy

+Protection Proxy

+Remote Proxy

+Monitor Proxy

+Firewall Proxy

+Cache Proxy

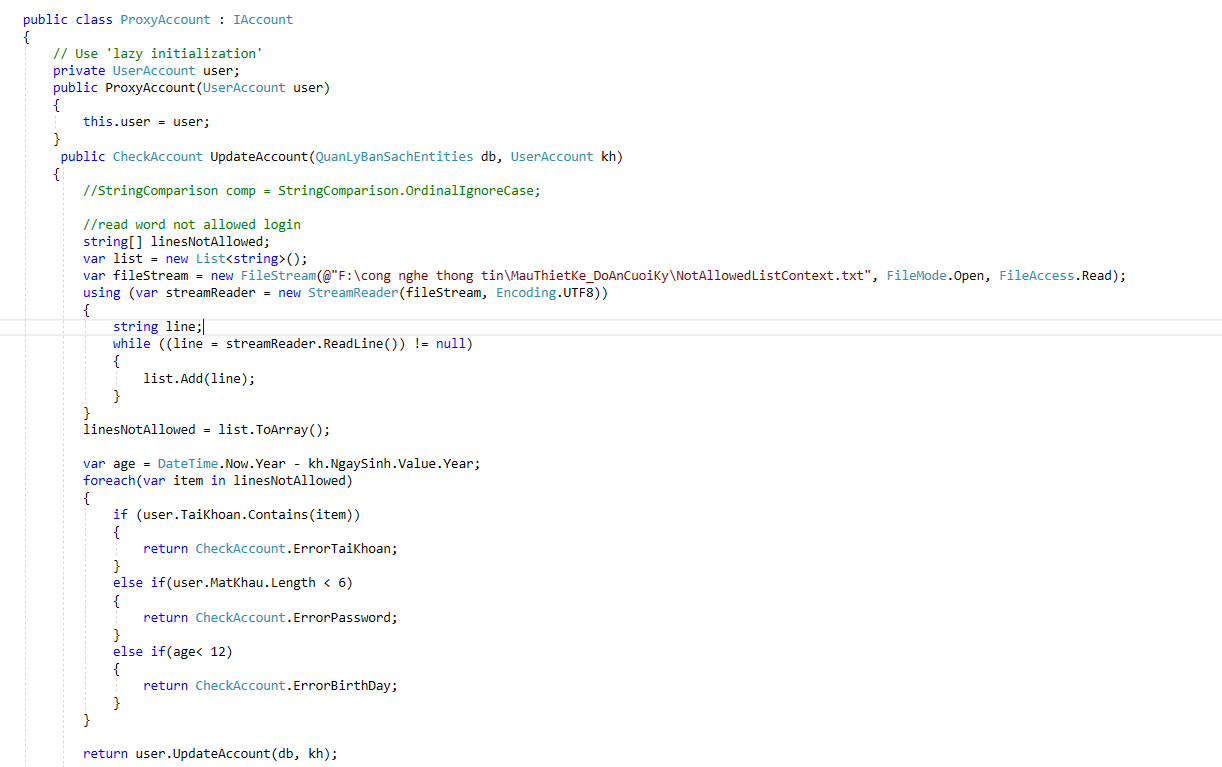
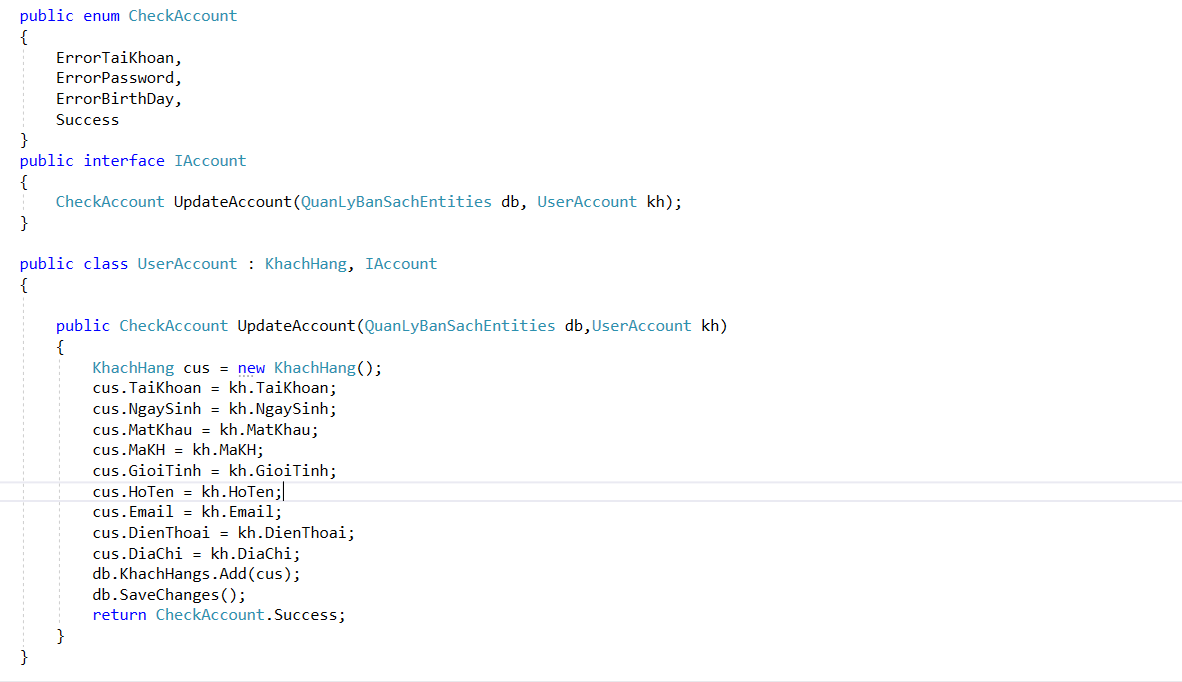
+Smart Reference Proxy

+Synchronization Proxy

+Copy-On-Write Proxy

Sử dụng Iterator mục đích là:

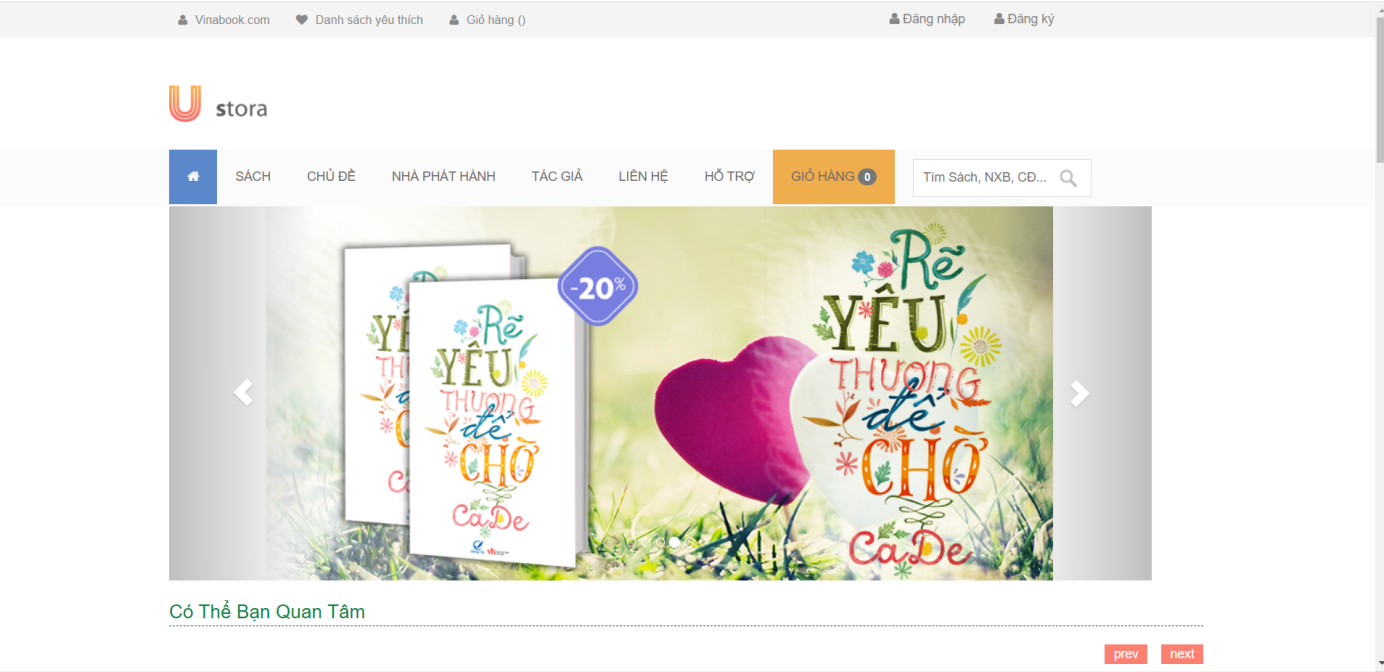
* Theo dõi trạng thái và vòng đời đối tượng .
* đối tượng gốc tồn tại trong môi trường từ xa
* yêu cầu muốn bảo vệ quyền truy xuất vào các phương thức của object thực sự chậm trễ khi tải một số tài nguyên nhất định (lazy loading)
* có nhiều truy cập vào đối tượng có chi phí khởi tạo ban đầu lớn

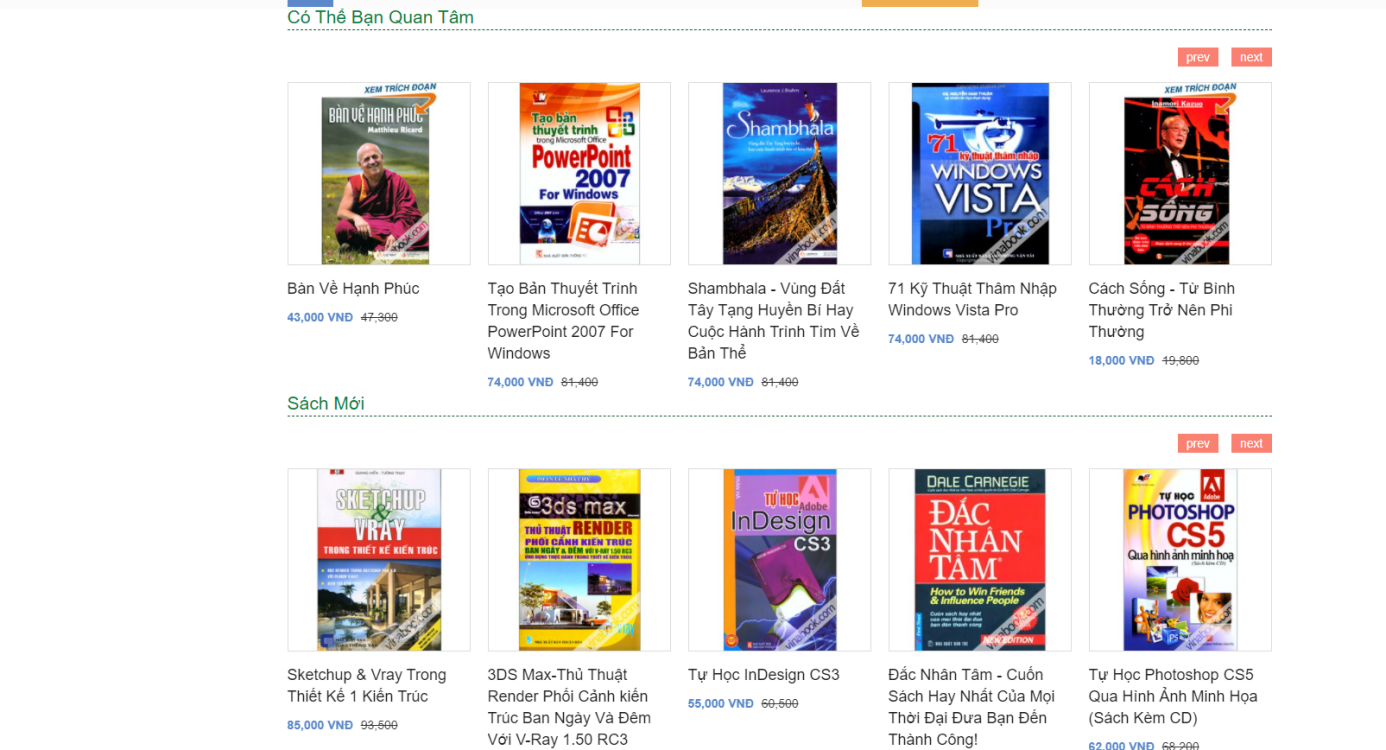
Ví dụ tạo chức năng cập nhật user account với từ khóa, mật khẩu, năm sinh, v.v

# Giao diện website bán sách

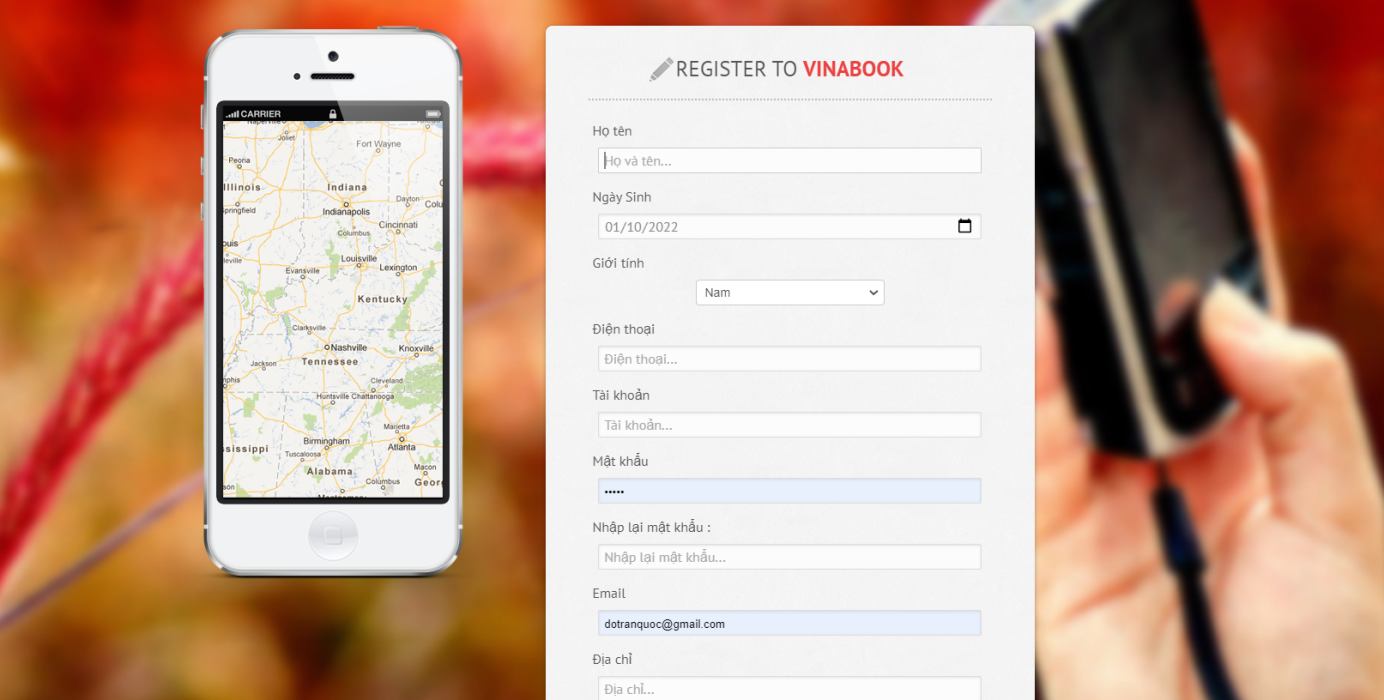
1. Trang web User

##### Trang chủ

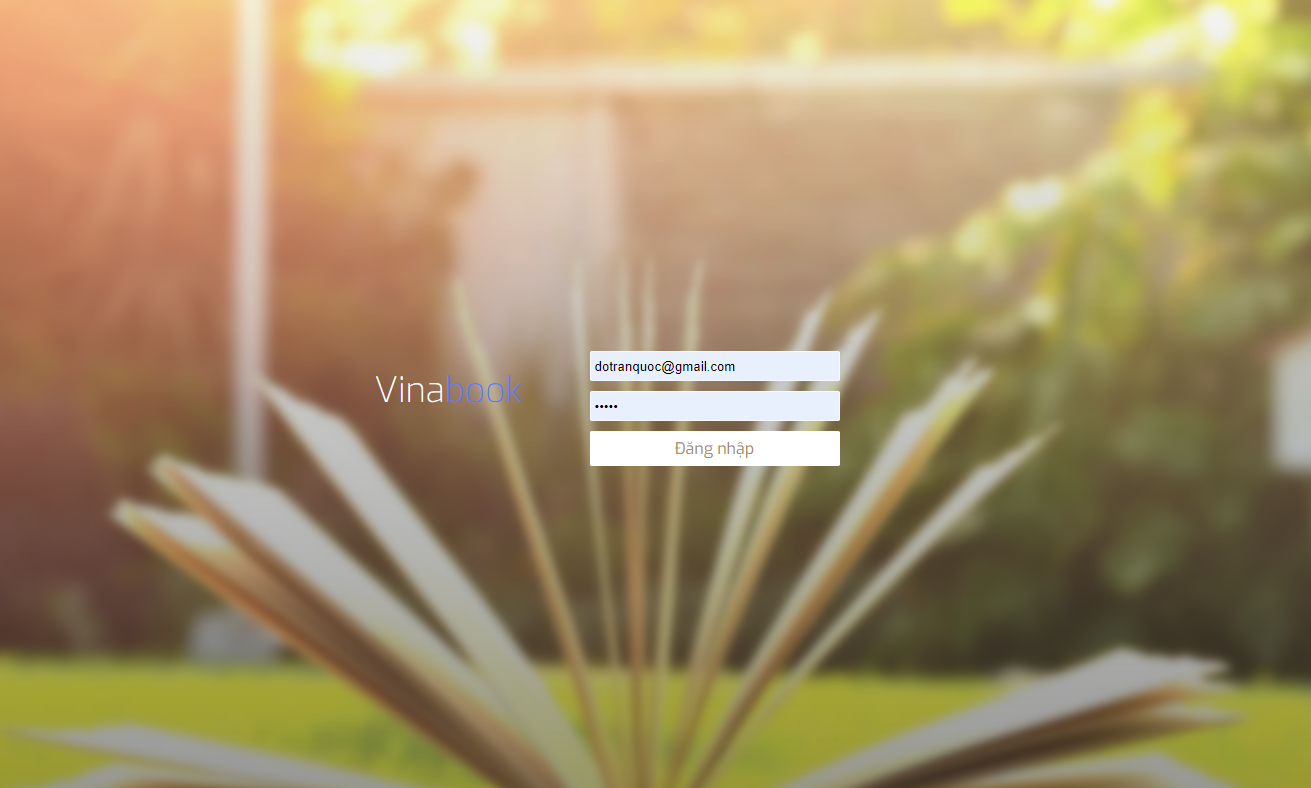




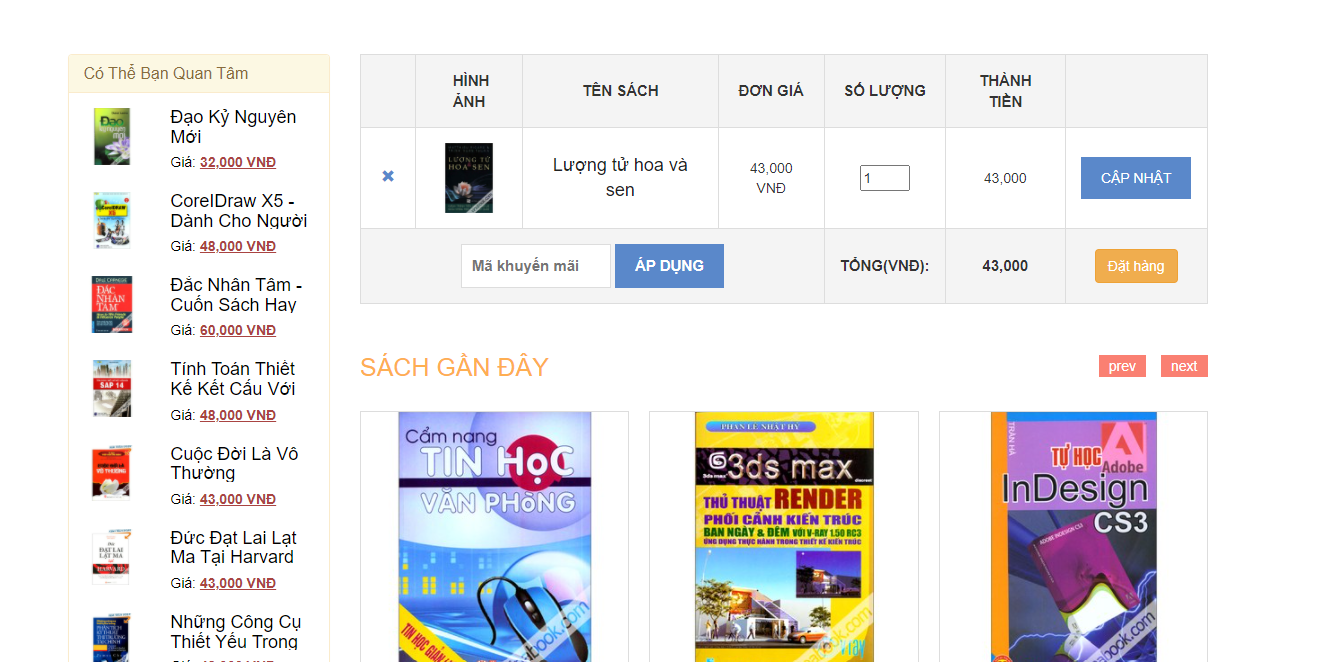
##### Đăng kí



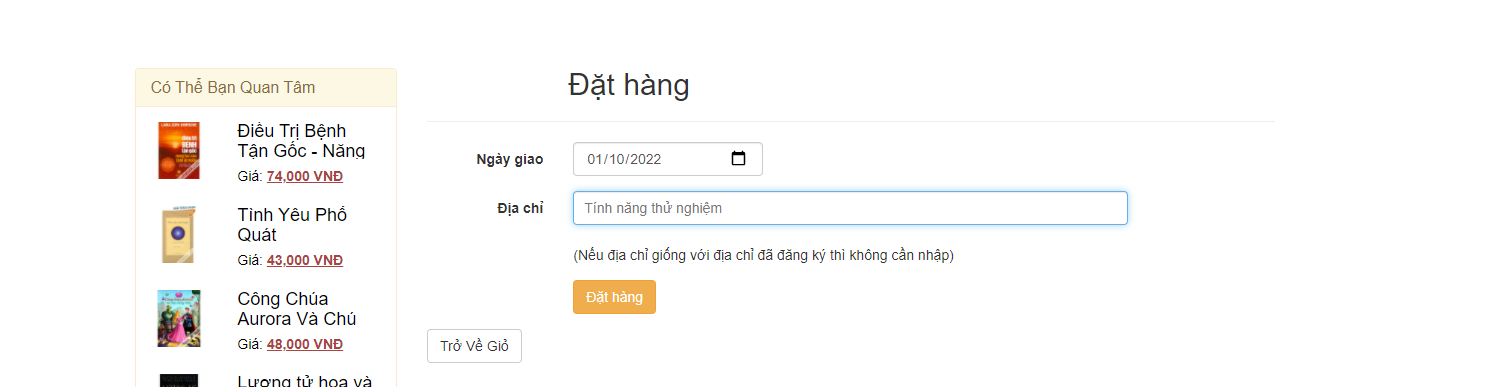
##### Đăng nhập



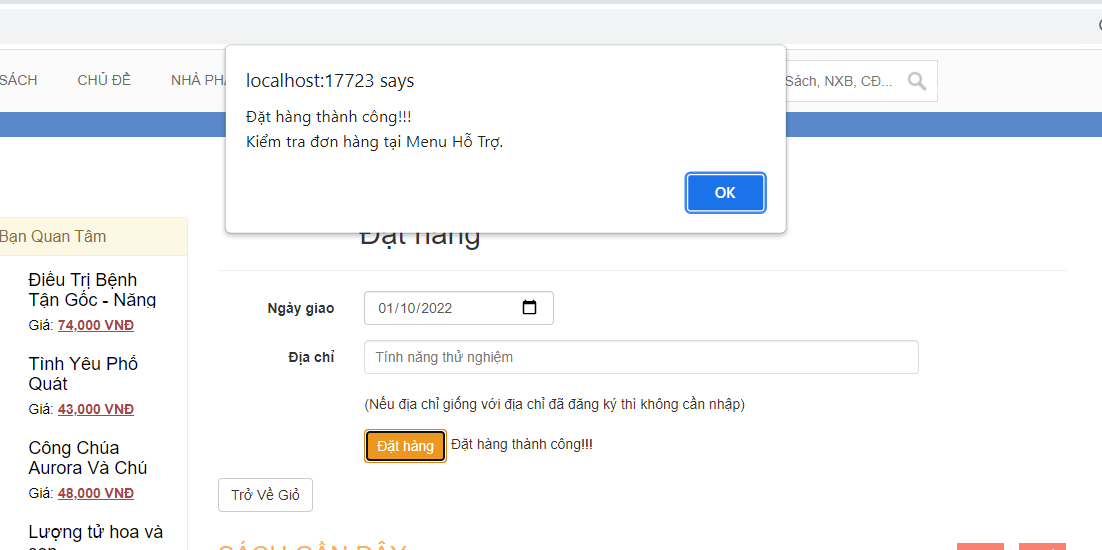
##### Giỏ hàng



##### Đặt hàng

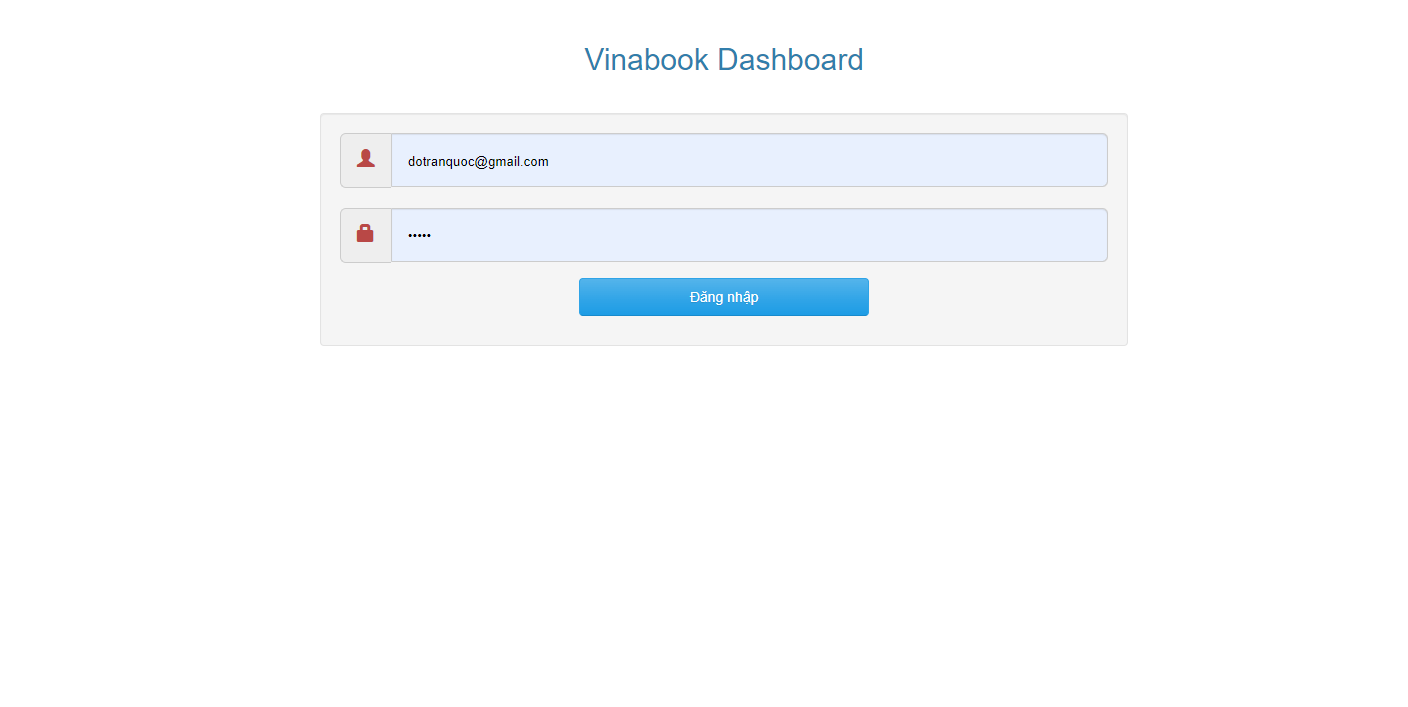


##### Đặt hàng thành công

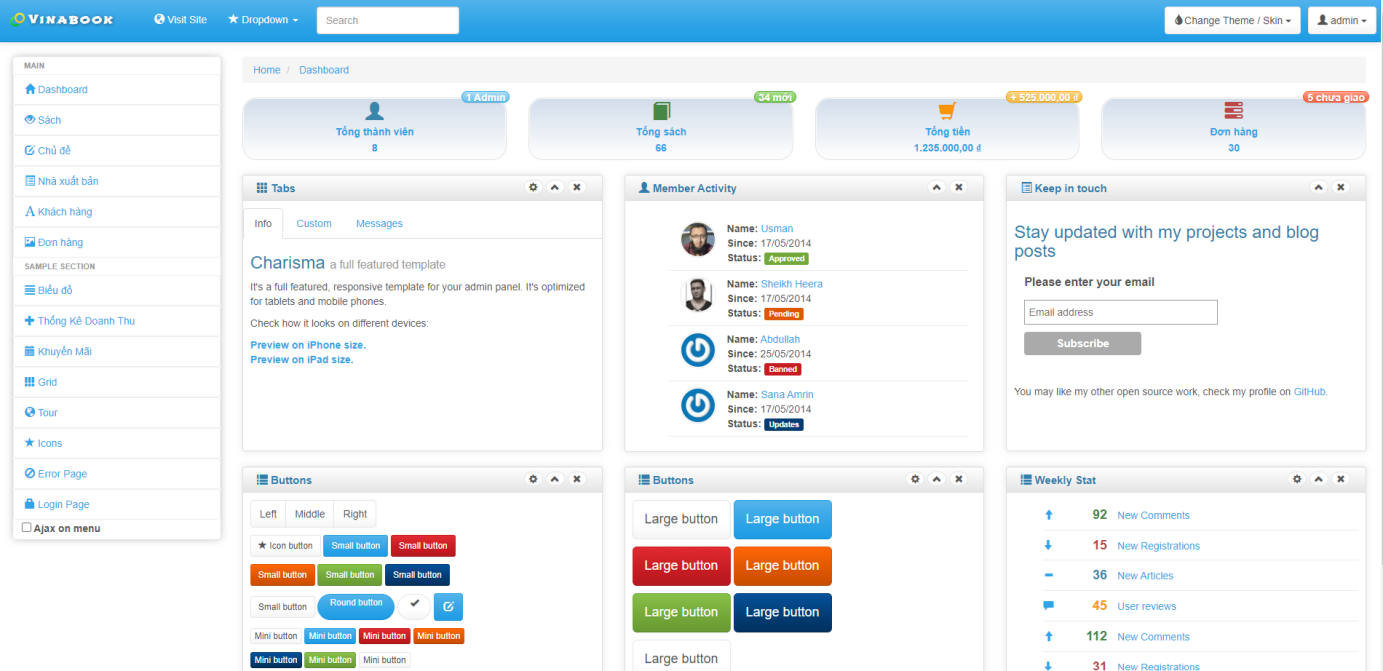


1. Trang web Admin

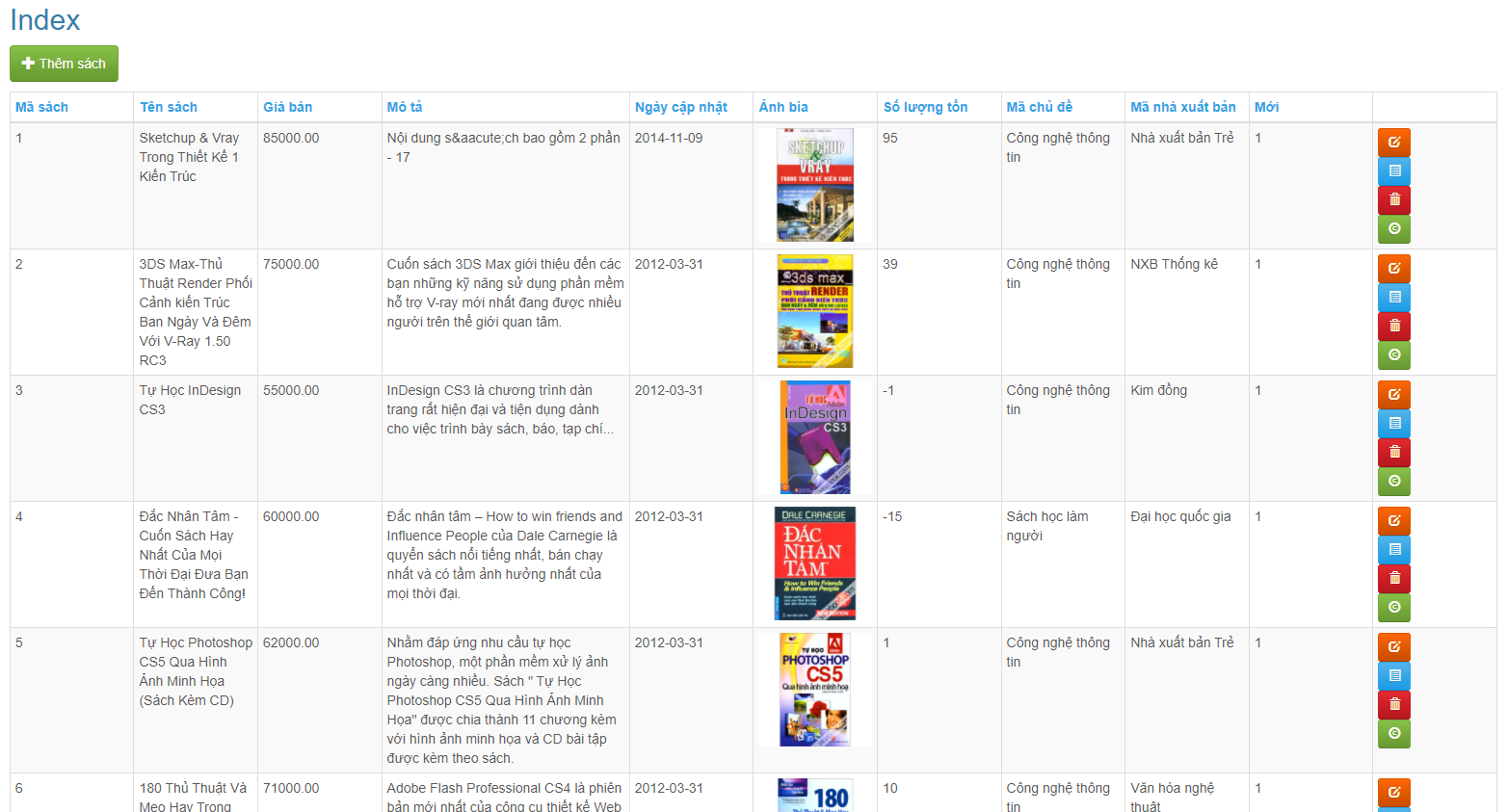
##### Đăng nhập admin



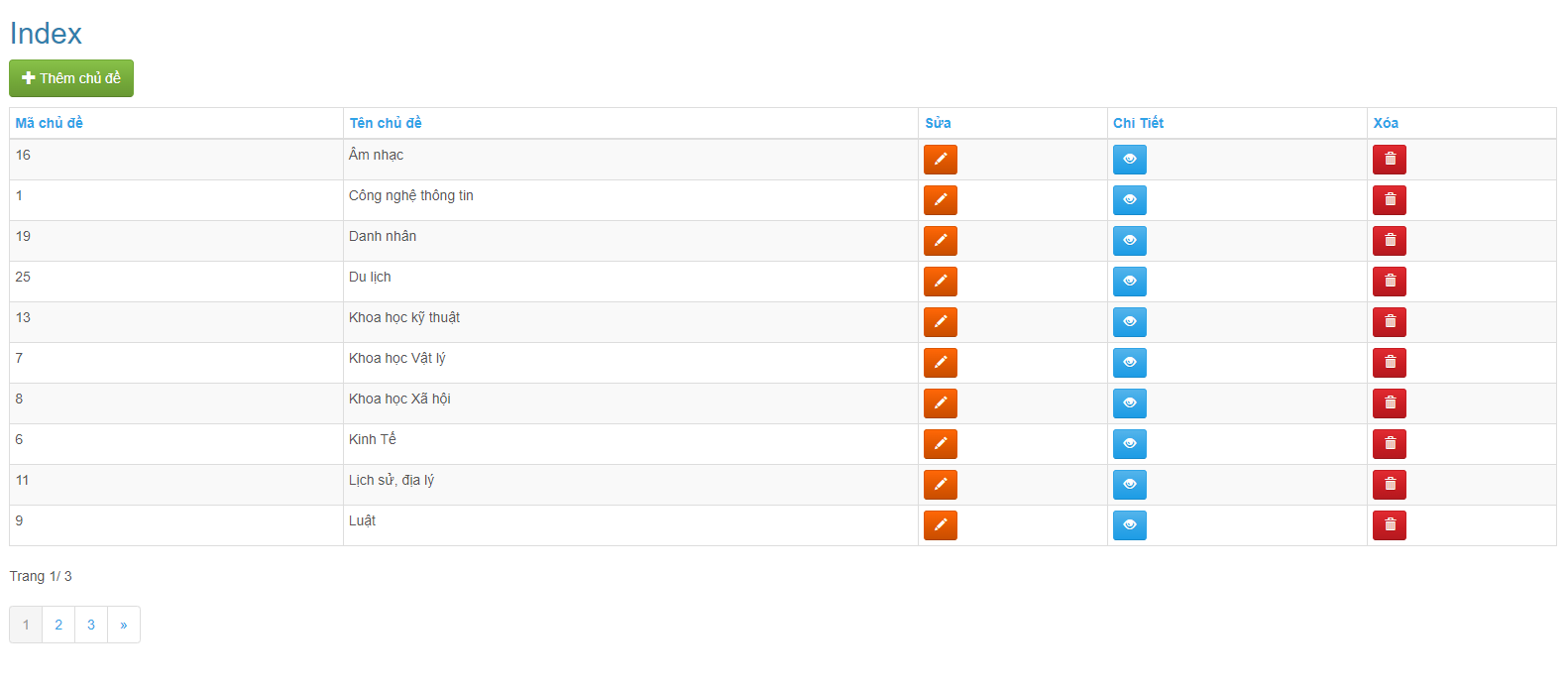
##### Giao diện admin



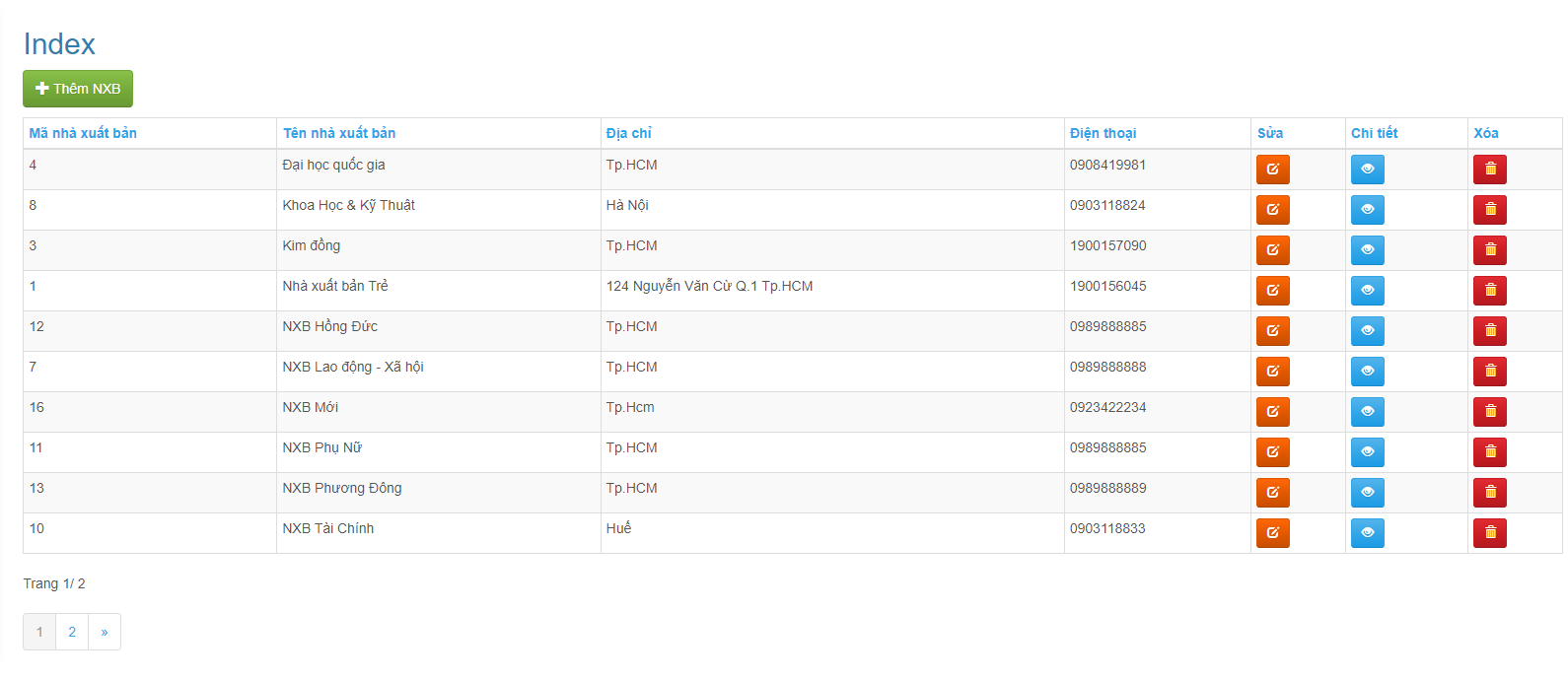
##### Quản lí sách



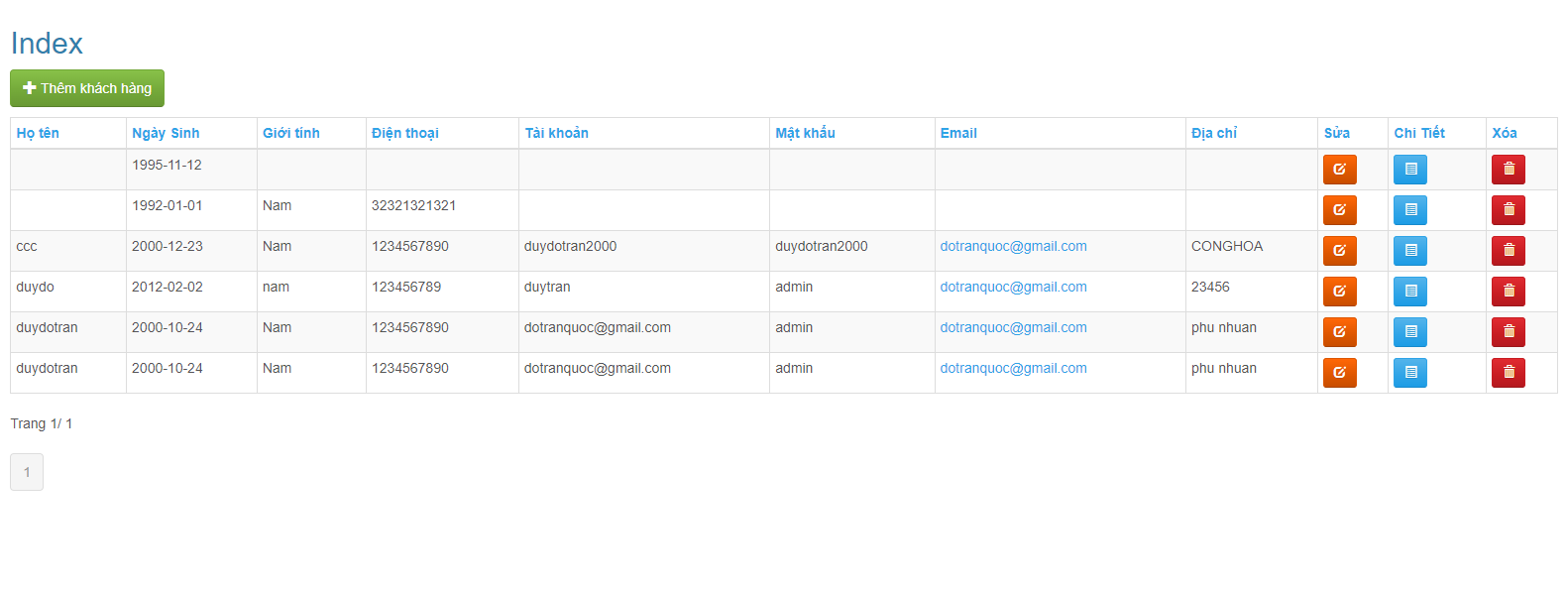
Quản lí chủ đề



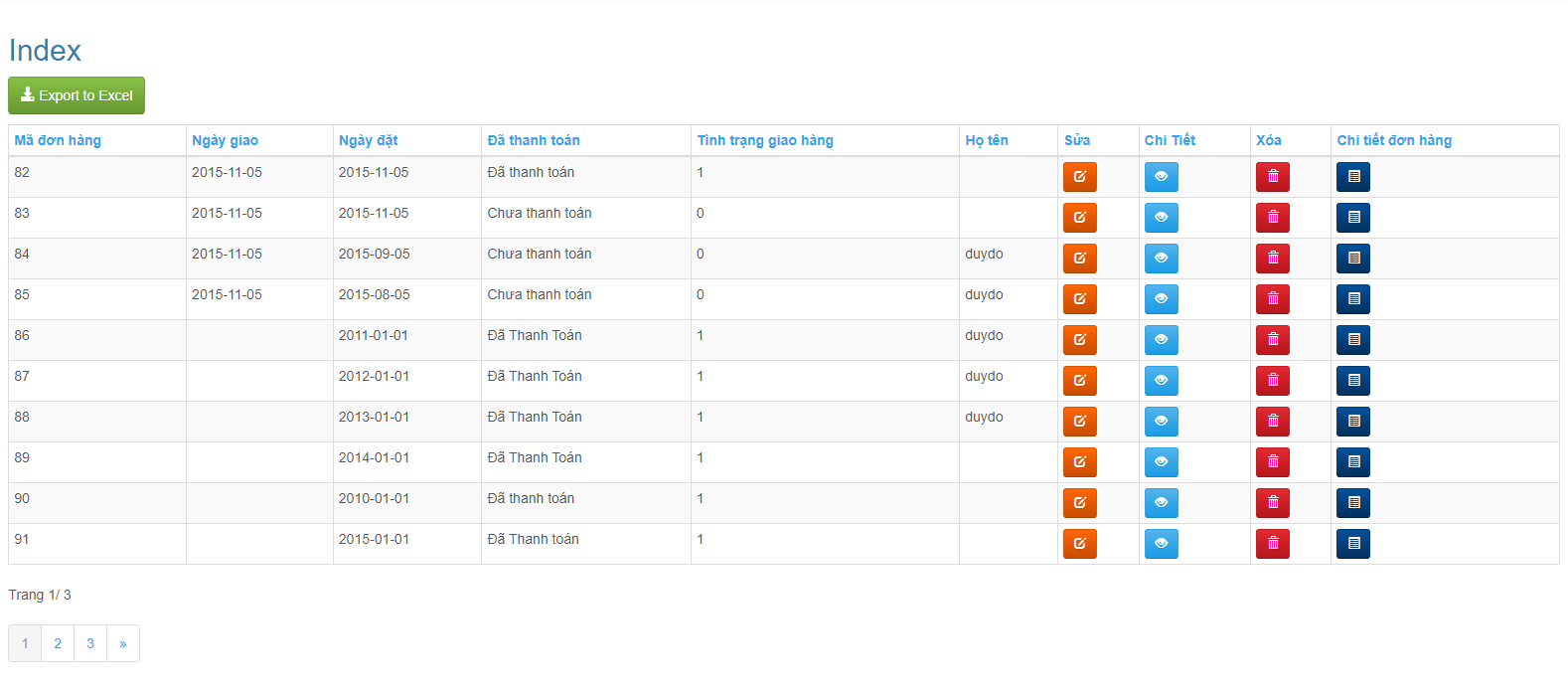
##### Quản lí nhà xuất bản



##### Quản lí khách hàng



##### Quản lí đơn hàng



# KẾT LUẬN

1. **Những vấn đề đã làm được**

Qua môn học này, Em đã hoàn thành các chức năng đồ án web bán sách như sau :

* + Trang chủ/ danh sách sản ohaarm
  + giỏ hàng (trống, có hàng)
  + chi tiết sản phẩm
  + Trang thanh toán
  + Trang Đăng kí, đăng nhập
  + Quản lí user account
  + Quản lí sách
  + Quản lí nhà xuất bản
  + Quản lí chủ đề

1. **Những vấn đề chưa làm được**

* Vẫn còn mắc lỗi trong việc triển khai các mẫu thiết kế trong quá trình chạy
* Chưa vận dụng tốt về chức năng khi áp dụng mẫu thiết kế.

1. **Hướng phát triển**

* Luyện tập thói quen và vận dụng nhiều mẫu thiết kế, phát huy sáng tạo
* Hiểu rõ vận dụng mẫu thiết kế tối đa, tránh bị sử dụng lặp lại kế thừa đối tượng nhiều lần
* Một số mẫu thiết kế dùng trong đồ án có thể tái sử dụng nhiều lần

# TÀI LIỆU THAM KHẢO

<https://phambinh.net/bai-viet/design-pattern-la-gi/>

<https://topdev.vn/blog/design-pattern-la-gi/>

<https://refactoring.guru/design-patterns/csharp>

<https://www.c-sharpcorner.com/UploadFile/bd5be5/design-patterns-in-net/>

<https://www.dofactory.com/net/design-patterns>

<https://viblo.asia/p/tong-hop-cac-bai-huong-dan-ve-design-pattern-23-mau-co-ban-cua-gof-3P0lPQPG5ox>