HỆ THỐNG ĐÀO TẠO CNTT QUỐC TẾ

APTECH VIỆT NAM

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

BÁO CÁO TỐT NGHIỆP

ĐỀ TÀI

THIẾT KẾ WEBSITE BÁN GIÀY

BẰNG NGÔN NGỮ LẬP TRÌNH ASP.NET MVC

Lớp: Batch 153

Thành viên: Đặng Văn Thịnh (Nhóm trưởng)

Nguyễn Đức Vĩnh Thắng

Huỳnh Minh Hiếu

Giáo viên hướng dẫn: Nguyễn Khánh

Đà Nẵng, ngày 12 tháng 4 năm 2021

**LỜI NÓI ĐẦU**

Đồ án này được các thành viên trong nhóm thực hiện trong khóa học lập trình viên .NET quốc tế thuộc chương trình đào tạo của trung tâm Softech Aptech.

Lý do chọn đề tài là khi tạo 1 trang web bán hàng, các học viên có thể thiết kế được nhiều tính năng và được nghiên cứu về 1 trang web thực tế nhất có thể tương tác với người dùng.

Mục tiêu chọn đề tài là các học viên có thể tạo ra 1 trang web bán giày để có thể giúp cho các shop bán giày có 1 trang web để bán hàng online.

Toàn bộ đồ án này đều được các thành viên tự lên ý tưởng và thực hiện, các ngôn ngữ được sử dụng là HTML, CSS, BOOTSTRAP, JAVASCRIPT, JQUERY, AJAX, LINQ,   
C#, ASP.NET, ENTITYFRAMEWORK, SQLSERVER.

**MỤC LỤC**

1. PHÂN TÍCH 2

1.1. Hiện trạng 2

1.2. Yêu cầu hệ thống 2

1.3. Use case 4

1.3.1. Đặc tả chức năng 4

1.3.2. Đặc tả các thực thể 6

2. THIẾT KẾ 9

2.1. Thiết kế cơ sở dữ liệu 9

2.1.1. Sơ đồ quan hệ thực thể 9

2.1.2. Thiết kế chi tiết các thực thể 11

2.2. Thiết kế giao diện 16

2.2.1. Thiết kế phần quản trị 16

2.2.2. Thiết kế phần khách hàng 26

3. Thực hiện viết mã 34

3.1. Viết mã lập trình cho phần quản trị 34

3.1.1. Quản lý sản phẩm 34

3.1.2. Quản lý danh mục 44

3.1.3. Quản lý đơn hàng 45

3.1.4. Quản lý người dùng 49

3.1.5. Quản lý phân quyền 52

3.2. Viết mã lập trình cho phần người dùng 53

3.2.1. Quản lý trang chủ 53

3.2.2. Quản lý tài khoản 55

3.2.3. Quản lý sản phẩm 64

3.2.4. Quản lý giỏ hàng 68

3.2.5. Quản lý thanh toán 69

# PHÂN TÍCH

## Hiện trạng

Hiện tại, các cửa hàng giày online hiện đang có mặt trên khắp các tỉnh thành cả nước Việt Nam, kinh doanh đa dạng nhiều loại giày khác nhau. Hiện các cửa hàng online đó chưa có những trang web với quy mô nhỏ để có thể giới thiệu sản phẩm của họ đến người dùng. Họ cũng không có một môi trường quản lý tự động để giúp cho việc kinh doanh trở nên thuận lợi hơn.

## Yêu cầu hệ thống

Aptech-Shose-Shop mong muốn có một website nhằm giới thiệu sản phẩm và thu nhận ý kiến của người tiêu dùng về các sản phẩm đang kinh doanh. Website cũng có các chức năng cho người quản trị để họ có thể quản lý sản phẩm, phân loại sản phẩm, quản lý khách hàng, quản lý các bình luận từ khác hàng về các sản phẩm

cũng như thống kê hàng hóa theo loại…

Yêu cầu cụ thể được đặt ra như sau

* Website bán hàng dành cho người tiêu dùng
* Cấu trúc chúng của tất cả các trang web trong website phải
* Chứa danh mục loại hàng hóa để khách hàng dễ dàng tìm kiếm hàng hóa theo từng loại
* Tìm kiếm hàng hóa theo tên
* Hiển thị danh sách các mặt hàng được mua nhiều nhất
* Đăng nhập/thông tin người tiêu dùng sau khi đã đăng nhập
* Trang chính của website

Trưng bày các mặt hàng đặc biệt được nhân viên quản trị chỉ định. Các mặt hàng đặc biệt có thể được trình bày theo cách nào đó sao cho dễ nhìn (có thể sử dụng slideshow)

* Trang trưng bày hàng hóa

Khi người dùng chọn loại hàng hoặc tìm kiếm theo tên thì danh sách hàng hóa thỏa mãn yêu cầu được hiện ra. Mỗi mặt hàng cần hiển thị hình ảnh, tên, đơn giá và giảm giá nếu có. Khi người dùng nhấp vào hình của hàng hóa thì sẽ chuyển đến trang hiển thị chi tiết hàng hóa đồng thời tăng số lần xem.

* Chi tiết hàng hóa

Yêu cầu của trang web này như sau

* Thông tin chi tiết của mặt hàng được chọn
* Danh sách các mặt hàng cùng loại với mặt hàng được chọn
* Khách hàng đặt mua sản phẩm trong trang này
* Các trang quản lý tài khoản sử dụng
* Đăng nhập
* Đăng ký
* Quên mật khẩu – Mở khóa tài khoản
* Đổi mật khẩu
* Cập nhật thông tin tài khoản
* Website bán hàng dành cho người quản trị
* Các trang quản lý

Các trang này cần được trang bị các chức năng xem, thêm, xóa, sửa, tìm kiếm và phân trang một cách hợp lý

* Quản lý hàng hóa
* Quản lý danh mục
* Quản lý khách hàng
* Quản lý đơn hàng
* Quản lý phân quyền
* Tổng hợp thống kê
* Thống kê số lượng hàng hóa
* Thống kê số lượng người đăng ký
* Thống kê số lượng đơn hàng
* Yêu cầu về bảo mật
* Tất cả các form nhập phải được kiểm soát dữ liệu một cách hợp lý

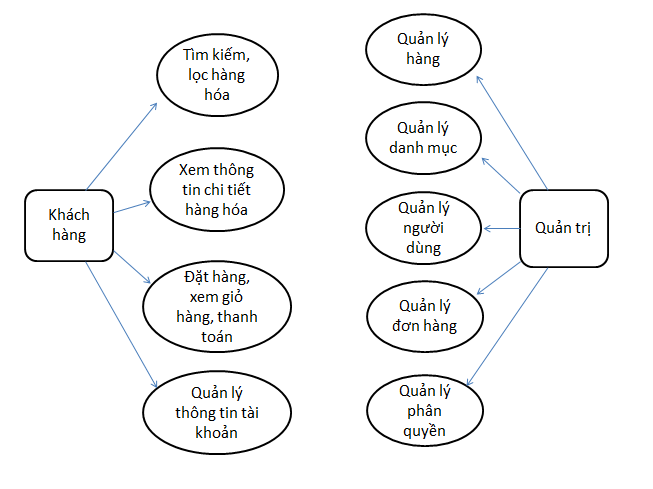
Khách hàng chưa đăng nhập không được xem thông tin tài khoản, không được đổi mật khẩu, cập nhật thông tin tài khoản

* Chỉ có các nhân viên quản trị mới được phép thực hiện các chức năng quản trị
* Yêu cầu về môi trường công nghệ

- Website phải được xây dựng với ASP.NET và SQL Server

## Use case

Use case là sơ đồ mô tả các tình huống sử dụng của người dùng trong hệ thống. Theo yêu cầu của website chúng ta có thể phác thảo sơ đồ use case như sau:



### Đặc tả chức năng

**1.3.1.1. Chức năng dành cho khách hàng**

- Cấu trúc chung của các trang web

Hiển thị thông tin liên lạc, form tìm kiếm theo tên, tìm theo danh mục, mục tài khoản của người dùng, logout, giỏ hàng, thanh toán. Nếu người đăng nhập thuộc quản trị thì có hiển thị link đến trang Admin

* Tìm theo danh mục, theo tên

Liệt kê ra các mặt hàng theo đúng với nhu cầu tìm kiếm. Bao gồm ảnh, đơn giá, giá giảm, liên kết đến trang chi tiết sản phẩm

* Lọc sản phẩm

Lọc sản phẩm theo tên, danh mục, đơn giá sản phẩm từ a – z, từ z – a, …. Thông tin sản phẩm như phần tìm sản phẩm

* Xem thông tin chi tiết

Hiển thị thông tin chi tiết của mặt hàng được chọn bởi khách hàng và các mặt hàng cùng loại có gắn liên kết đến trang chi tiết hàng hóa để khách hàng có thể xem luôn mà không cần phải trở lại trang liệt kê hàng hóa

* Thêm sản phẩm vào giỏ hàng
* Trong trang chi tiết sản phẩm khách hàng có thể chọn thông tin sản phẩm theo nhu cầu rồi thêm trực tiếp vào trong giỏ hàng
* Trong trang giỏ hàng, khách hàng có thể sửa lại số lượng và có thể xóa sản phẩm trong giỏ
* Ngoài ra, khách hàng còn có thể quay lại trang sản phẩm để đặt mua thêm một số hàng hóa rồi sau đó vào kiểm trang giỏ hàng 1 lần nữa, xem những thông tin sản phẩm và số tiền phải trả rồi tiếp tục đi đến thanh toán
* Quản lý tài khoản
* Đăng ký: Đăng ký thành viên mới với vai trò là khách hàng, tức không được phép sử dụng các chức năng trong phần quản trị.
* Đăng nhập: để thực hiện các chức năng như quản lý thông tin tài khoản, đặt hàng, xem lịch sử đặt hàng.
* Đổi mật khẩu, quên mật khẩu: Để đảm bảo tính an toàn cho tài khoản. Ngoài ra, tài khoản đăng nhập sai nhiều lần sẽ bị khóa, nên có thêm 1 chức năng mở khóa tài khoản
* Cập nhật thông tin tài khoản: đổi ảnh đại diện, thông tin cá nhân
* Đăng xuất: để thoát tài khoản cho người khác không vào sử dụng tài khoản của mình được

**1.3.1.2. Chức năng dành cho quản trị**

- Quản lý hàng hóa:

Thực hiện các chức năng thêm, xóa, sửa hàng hóa

* Quản lý danh mục

Thực hiện các chức năng thêm, xóa, sửa danh mục

* Quản lý đơn hàng

Thực hiện các chức năng thêm, xóa, sửa đơn hàng

* Quản lý người dùng và phân quyền

Thực hiện các chức năng thêm, xóa, sửa và nhân viên không thể xem phần này

### Đặc tả các thực thể

Sau đây là các thực thể và thuộc tính của chúng cần thiết cho việc xây dựng website Shose-shop

* Products:
* Id
* ProductName
* UnitPrice
* DiscountRatio
* DiscountExpiry
* CategoryId
* ProductImageId
* Description
* CreatedDate
* CreatedBy
* StatusId
* ProductImages:
* Id
* ProductId
* ImageUrl
* StatusProducts:
* Id
* StatusName
* Categories
* Id
* CategoryName
* Position
* Colors:
* Id
* ColorName
* ProductColors:
* Id
* ProductId
* ColorId
* Sizes:
* Id
* SizeName
* ProductSizes:
* Id
* ProductId
* SizeId
* TbUsers:
* Id
* FullName
* Gender
* Email
* Address
* Password
* StatusId
* CreatedDate
* Avatar
* NumberPhone
* CountLogin
* TimeLock
* StatusUsers:
* Id
* StatusName
* Roles:
* Id
* RoleName
* UserRoles:
* Id
* UserId
* RoleId
* Orders:
* Id
* CustomerName
* CustomerEmail
* CustomerAddress
* CustomerPhone
* StatusId
* UserId
* OrderDate
* OrderNote
* OrderDetails:
* Id
* OrderId
* ProductId
* UnitPrice
* DiscountRatio
* Quantity
* ColorName
* SizeName
* StatusOrders
* Id
* StatusName

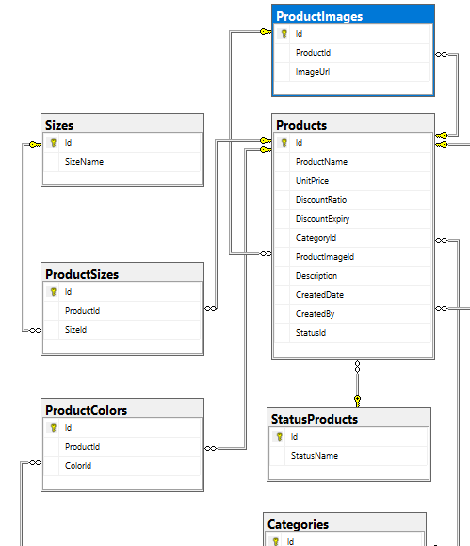
# THIẾT KẾ

## Thiết kế cơ sở dữ liệu

Dựa vào đặc tả thực thể và cụ thể làm việc với Shose-shop bạn có thể thiết kế sơ đồ quan hệ và chi tiết các thực thể để từ có có đủ thông tin cài đặt cơ sở dữ liệu lên SQL Server.

### Sơ đồ quan hệ thực thể

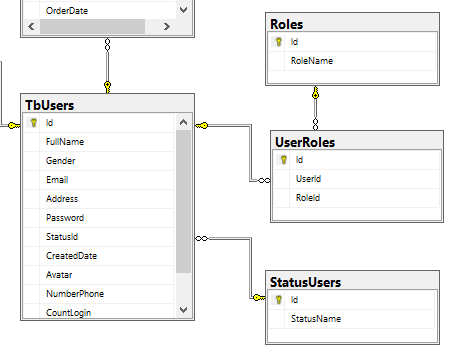
* Mô hình các thực thể có quan hệ với bảng Products:

****

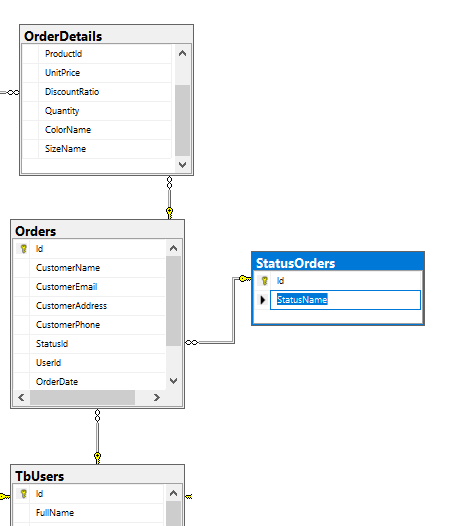
* Bảng Products – ProductImages: Id(Products ) – ProductId(ProductImages)

ProductImageId(Products) – Id(ProductImages)

* Bảng Products – StatusProducts: StatusId(Products) – Id(StatusProducts)
* Bảng Products – Categories: CategoryId(Products ) – Id(Categories)
* Bảng Products – ProductSizes: Id(Products ) – ProductId(ProductSizes)
* Bảng Products – ProductColors: Id(Products ) – ProductId(ProductColors)
* Mô hình các thực thể có quan hệ với bảng Users



* Bảng TbUsers – StatusUsers: StatusId(TbUsers) –Id(StatusUsers)
* Bảng TbUsers – UserRoles: Id(TbUsers) – UserId(UserRoles)
* Mô hình các thực thể có quan hệ với bảng Orders



* Bảng Orders – OrderDetails: Id(Orders) – OrderId(OrderDetails)
* Bảng Orders – StatusOrders: StatusId(Orders) – Id(StatusOrders)
* Bảng Orders – TbUsers: UserId(Orders) – Id(TbUsers)

### Thiết kế chi tiết các thực thể

**2.1.2.1. Products**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Mã sản phẩm |
| ProductName | string | Not null | Tên sản phẩm |
| UnitPrice | double | Not null | Đơn giá |
| DiscountRatio | double | allow null | Giảm giá |
| DiscountExpiry | DateTime | Not null | Ngày kết thúc giảm giá |
| CategoryId | int | Not null | Mã thể loại |
| ProductImageId | int | Not null | Id hình ảnh |
| Description | string | Not null | Mô tả sản phẩm |
| CreatedDate | DateTime | Not null | Ngày tạo |
| CreatedBy | int | Not null | Id người tạo |
| StatusId | int | Not null | Id tình trạng sản phẩm |

**2.1.2.2. ProductImages**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Id |
| ProductId | int | FK, not null | Khóa khoại xác định sản phẩm |
| ImageUrl | string | Not null | Đường dẫn ảnh |

**2.1.2.3. Categories**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk,not null, tự tăng | Mã thể lọa |
| CategoryName | String | Not null | Tên thể loại |
| Position | int | Allow null | Vị trí sắp xếp |

**2.1.2.4. Sizes**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk,not null, tự tang | Mã Size |
| SizeName | string | Not null | Tên Size |

**2.1.2.5. ProductSizes**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Mã ProductSize |
| ProductId | Int | Fk, not null | Khóa ngoại đến bảng Products |
| SizeId | int | Fk,not null | Khóa ngoại đến bảng Size |

**2.1.2.6. Colors**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Mã màu |
| ColorName | string | Not null | Tên màu |

**2.1.2.7. ProductColors**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Id |
| ColorId | Int | FK,not null | Khóa ngoại đến bảng Colors |
| ProductId | int | FK,not null | Khóa ngoại đến bảng Products |

**2.1.2.8. StatusProducts**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Mã tình trạng |
| StatusName | string | Not null | Tên tình trạng |

**2.1.2.9. Orders**

Bảng Orders lưu thông tin đơn hàng có cấu trúc như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | INT | PK, not null, tự tăng | Mã đơn hàng |
| CustomerName | STRING | Not null | Tên khách hàng |
| CustomerEmail | STRING | Not null | Email khách hàng |
| CustomerAddress | STRING | Not null | Địa chỉ khách hàng |
| CustomerPhone | STRING | Not null | Số điện thoại khách hàng |
| StatusId | INT | FK, not null | Mã trạng thái đơn hàng |
| UserId | INT | FK, not null | Mã người dùng |
| OrderDate | DATETIME | Not null | Ngày đặt hàng |
| OrderNote | STRING | Null | Ghi chú mua hàng |

**2.1.2.10. OrderDetails**

Bảng OrderDetails lưu chi tiết đơn hàng có cấu trúc như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | INT | PK, not null, tự tăng | Mã chi tiết đơn hàng |
| OrderId | INT | FK, not null | Mã đơn hàng |
| ProductId | INT | FK, not null | Mã sản phẩm |
| UnitPrice | FLOAT | Not null | Giá bán sản phẩm hiện tại |
| DiscountRatio | FLOAT | Not null | Phần trăm giảm giá sản phẩm |
| Quantity | INT | Not null | Số lượng sản phẩm đặt |
| ColorName | STRING | Not null | Màu sản phẩm |
| SizeName | STRING | Not null | Size sản phẩm |

**2.1.2.11. StatusOrders**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | INT | PK, not null, tự tăng | Mã trạng thái đơn hàng |
| StatusName | STRING | Not null | Tên trạng thái đơn hàng |

**2.1.2.12. TbUsers**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Id,Khóa chính |
| FullName | string | Not null | Tên đầy đủ |
| Gender | Bool | Null | Giới tính |
| NumberPhone | string | Null | Số điện thoại |
| Email | String | Null | Email |
| Address | String | Null | Địa chỉ |
| Password | String | Not null | Mật khẩu |
| StatusId | int | Fk, not null | Tình trạng |
| CreateDate | DateTime | Not null | Ngày tạo |
| Avatar | string | Null | Ảnh đại diện |
| CountLogin | int | Allow null | Số lần đăng nhập |
| TimeLock | DateTime | Allow null | Thời gian bị khóa |

**2.1.2.13. Roles**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Id |
| RoleName | string | not null | Tên role |

**2.1.2.14. UserRoles**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Id |
| UserId | Int | FK,not null | Khóa ngoại đến bảng TblUser |
| RoleId | int | FK,not null | Khóa ngoại đến bảng Roles |

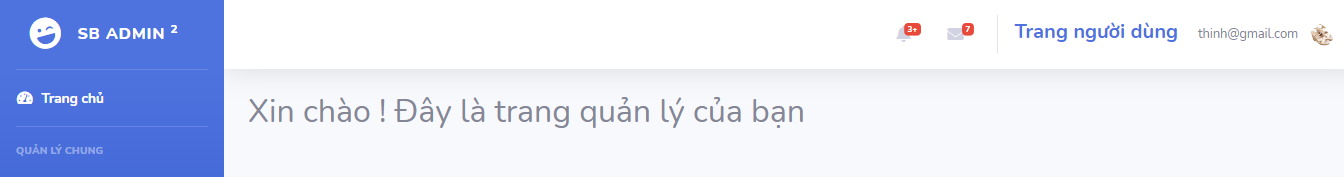
**2.1.2.15. StatusUsers**

|  |  |  |  |
| --- | --- | --- | --- |
| Tên cột | Kiểu | Ràng buộc | Ghi chú |
| Id | Int | Pk, not null, tự tăng | Id |
| StatusName | string | not null | Tình trạng User |

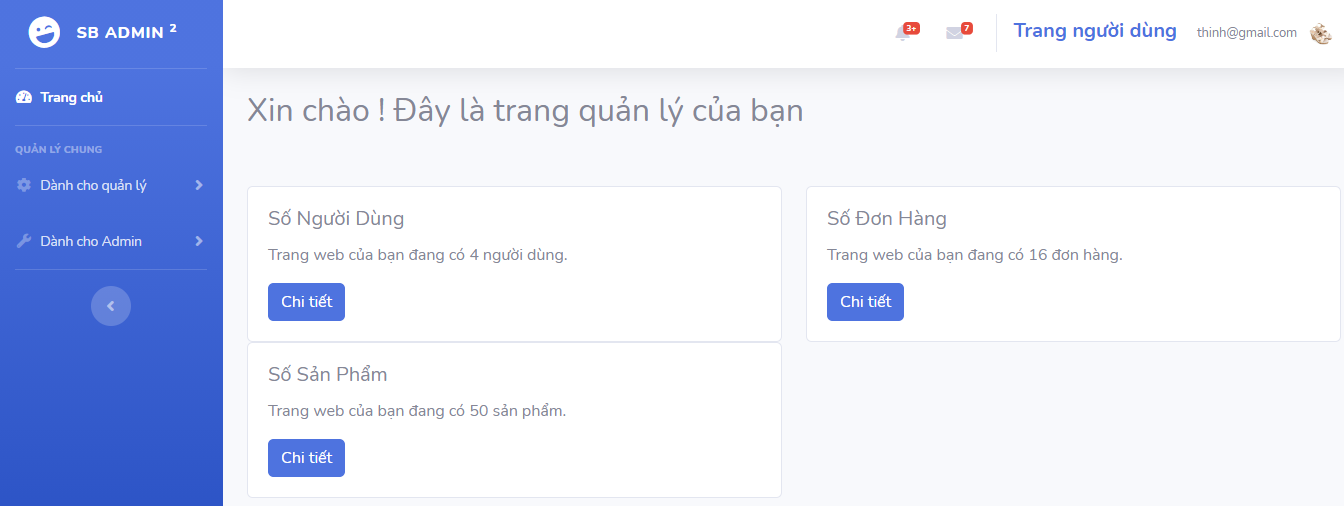
## Thiết kế giao diện

### Thiết kế phần quản trị

**2.2.1.1. Cấu trúc chung (layout)**



**2.2.1.2. Trang chủ**

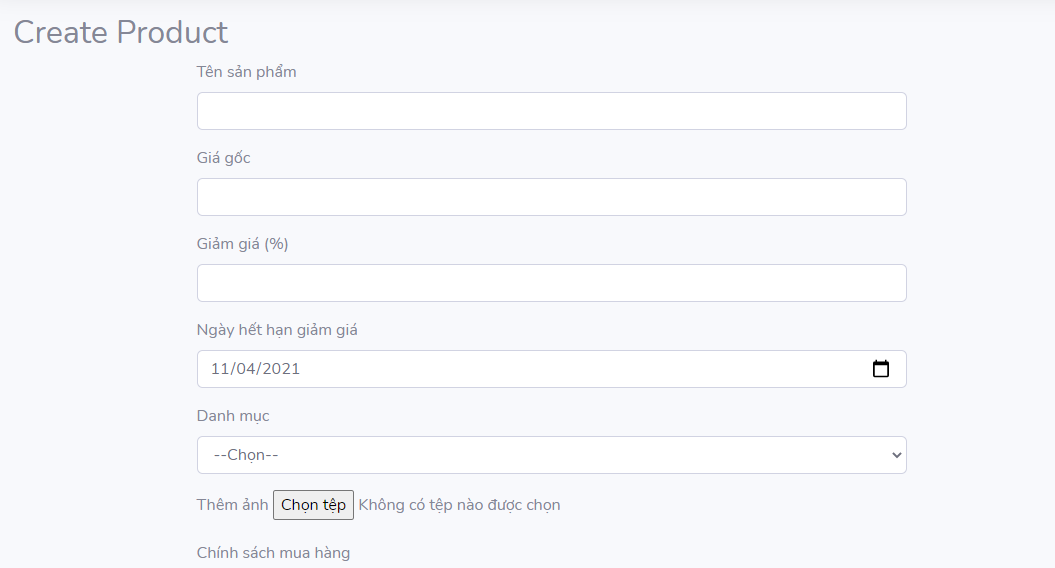


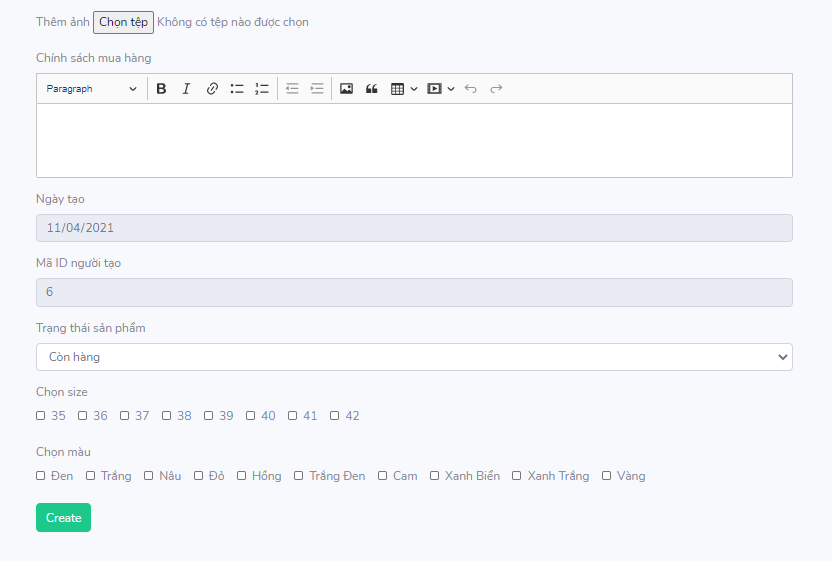
**2.2.1.3. Quản lý sản phẩm**

Trang hiển thị:

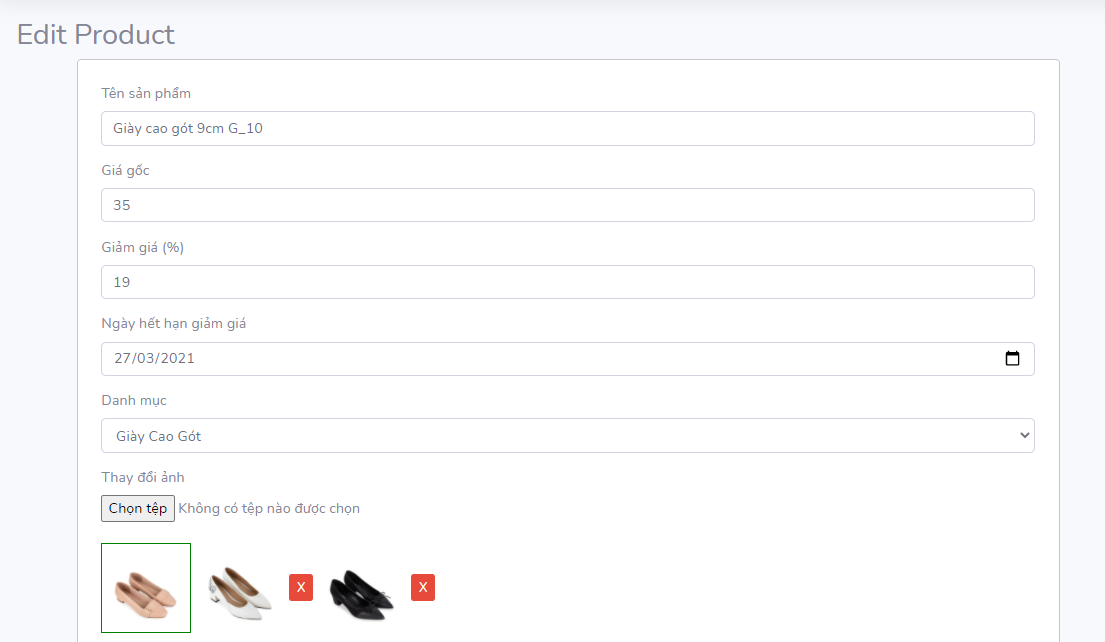


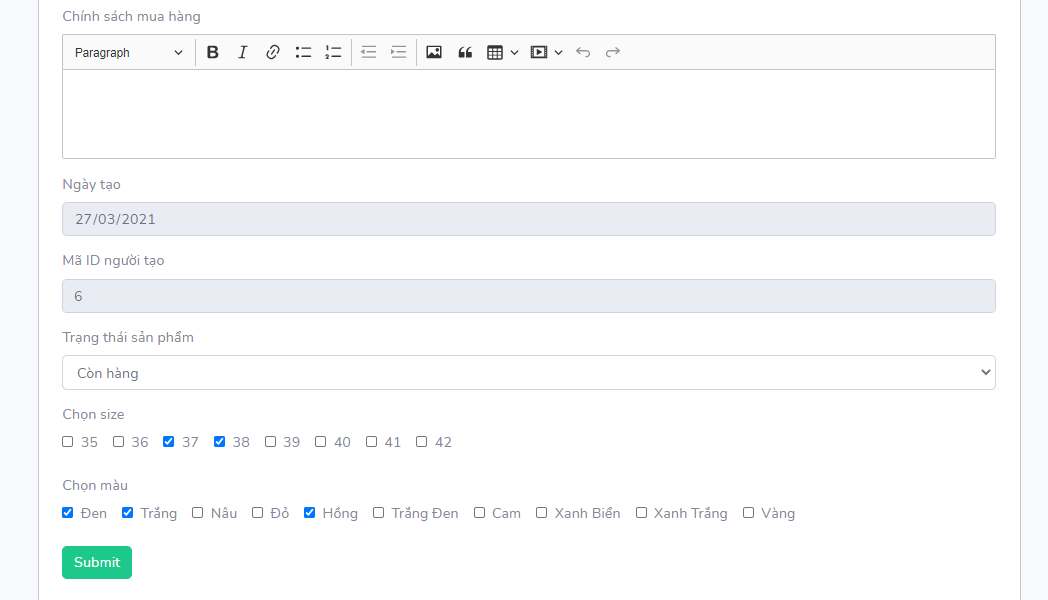
Tạo mới sản phẩm





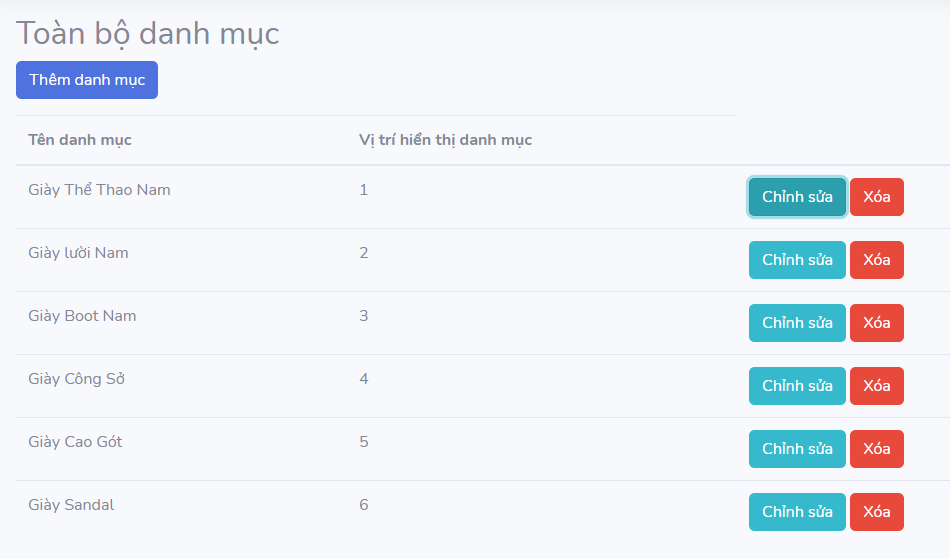
Chỉnh sửa sản phẩm



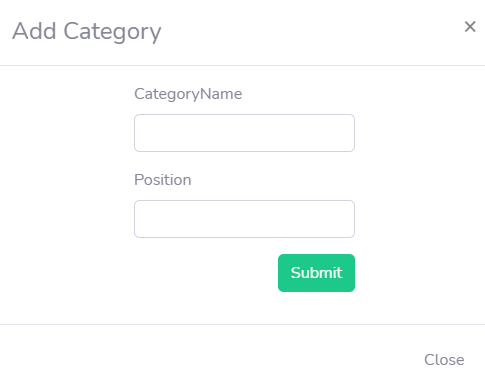


**2.2.1.4. Quản lý danh mục**

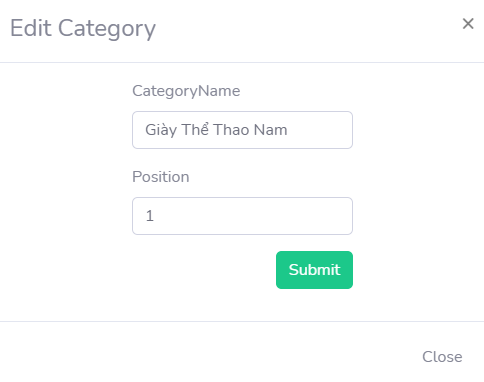
Trang hiển thị:



Thêm danh mục:

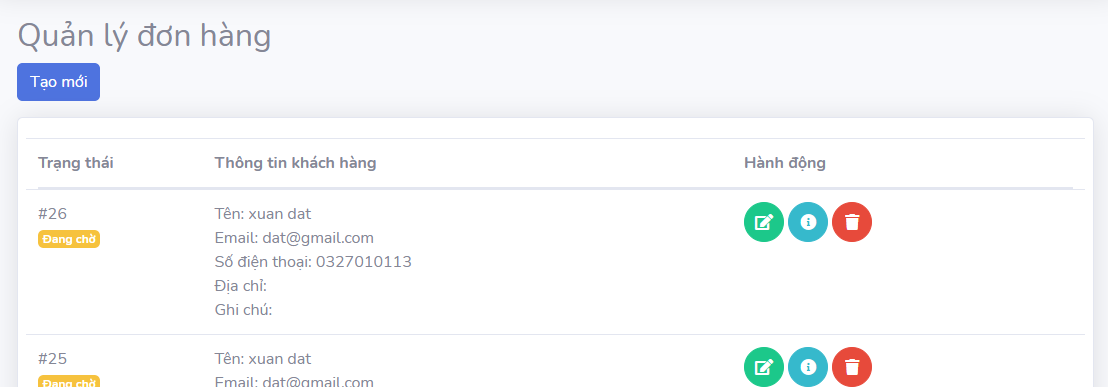


Sửa danh mục:

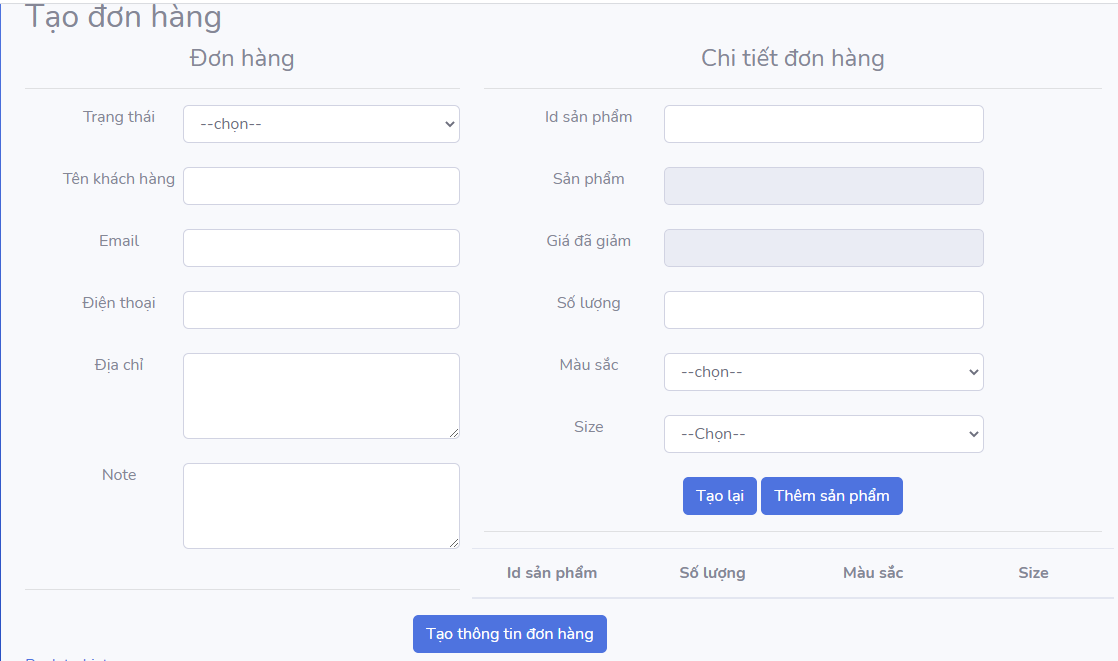


**2.2.1.5. Quản lý đơn hàng**

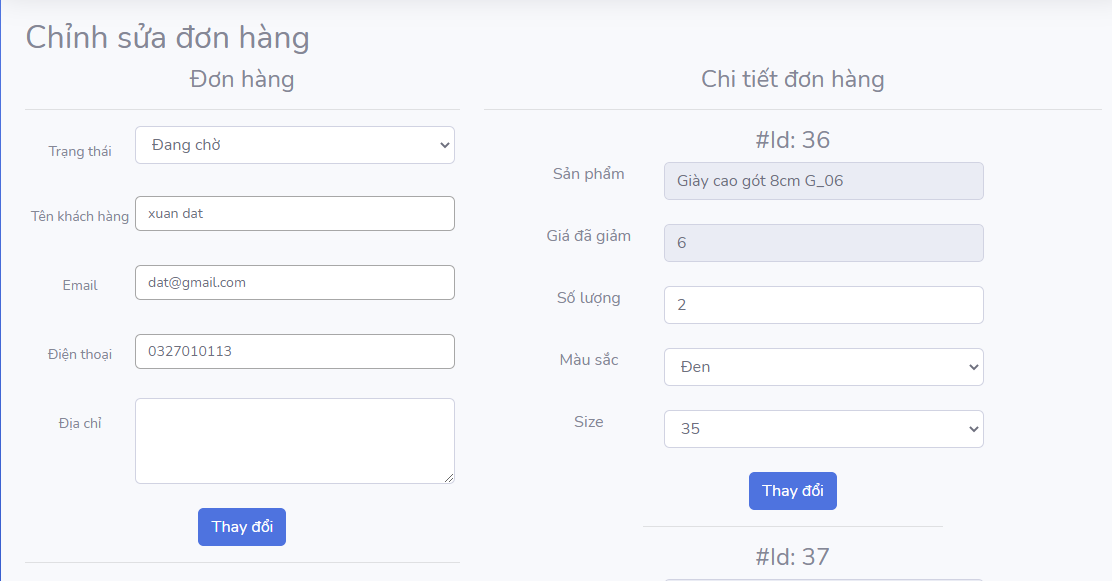
Trang hiển thị:



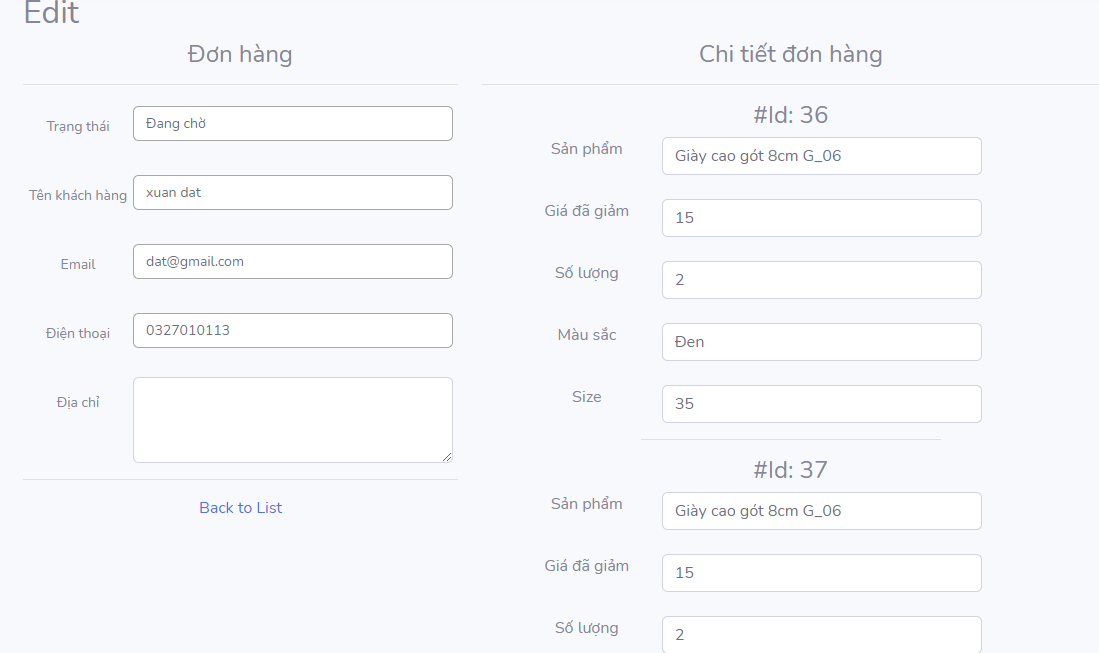
Tạo mới đơn hàng:



Chỉnh sửa đơn hàng

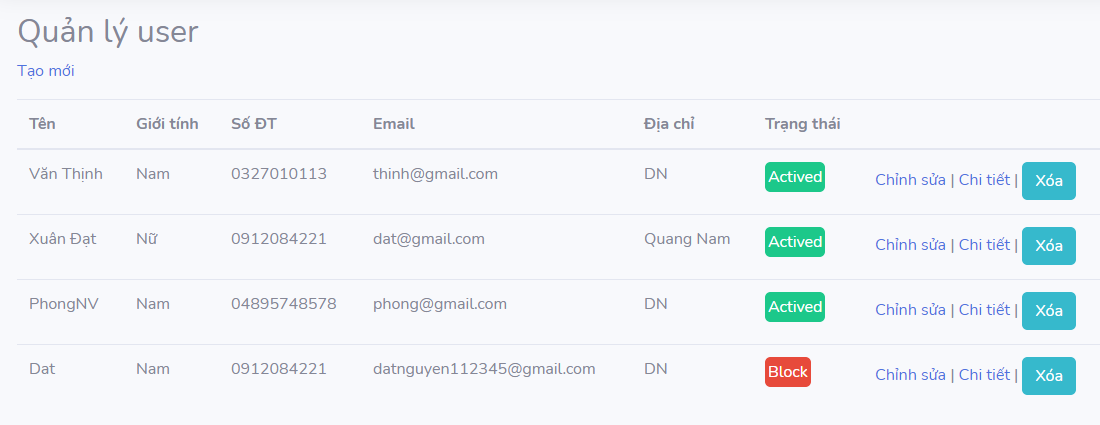


Chi tiết đơn hàng:

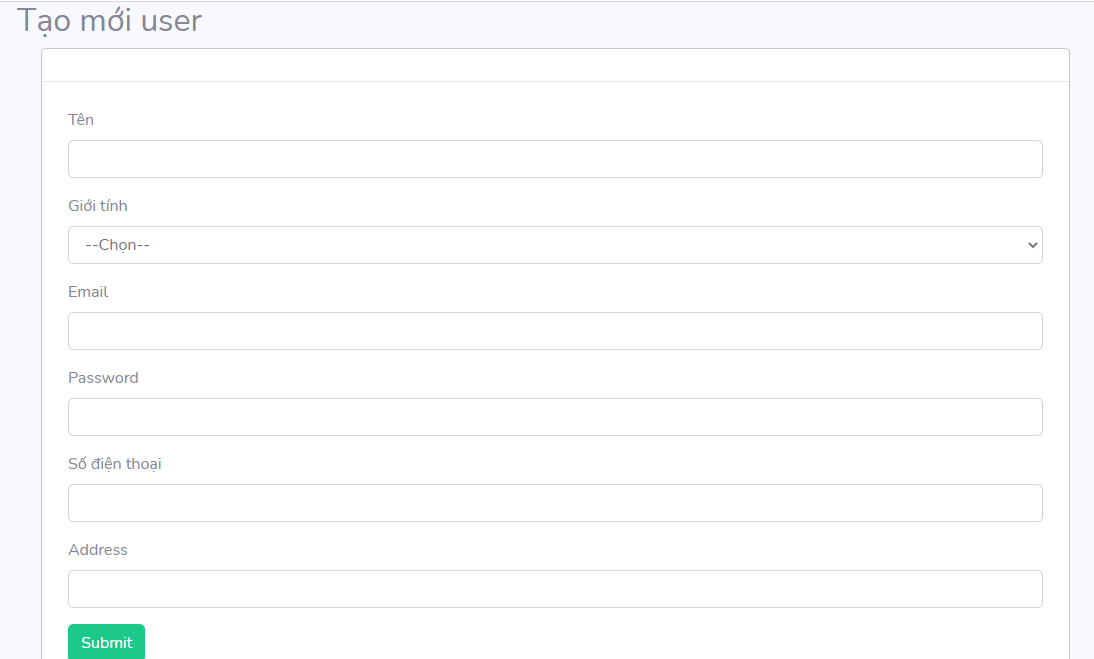


**2.2.1.6. Quản lý người dùng**

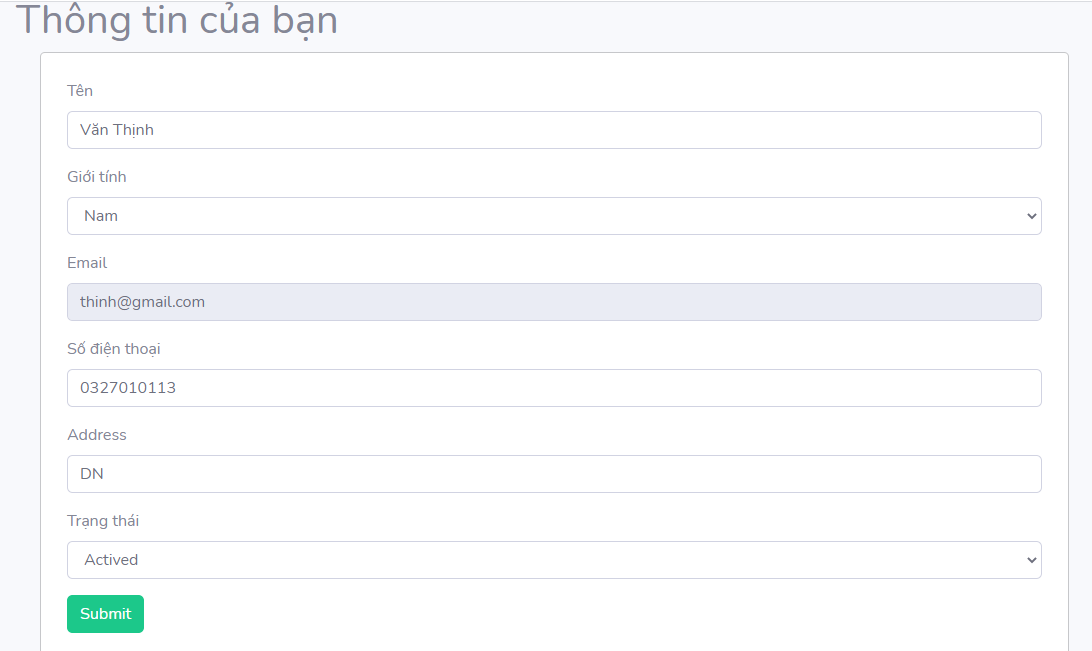
Trang hiển thị:



Thêm mới user

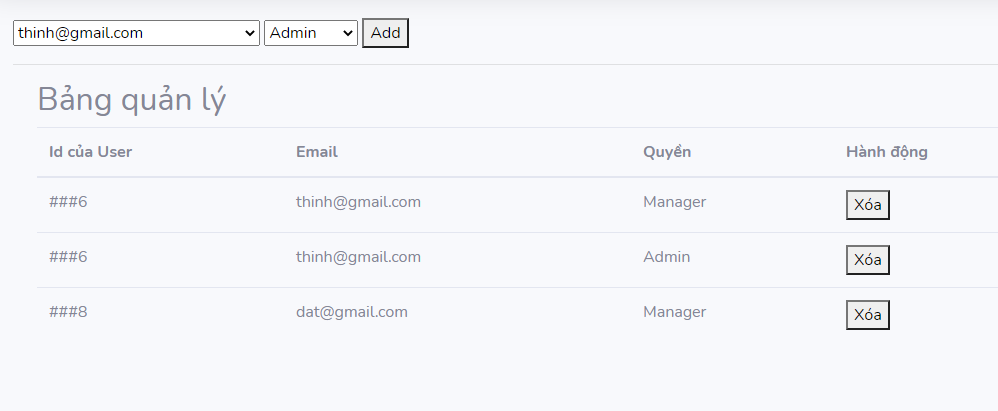


Chỉnh sửa user

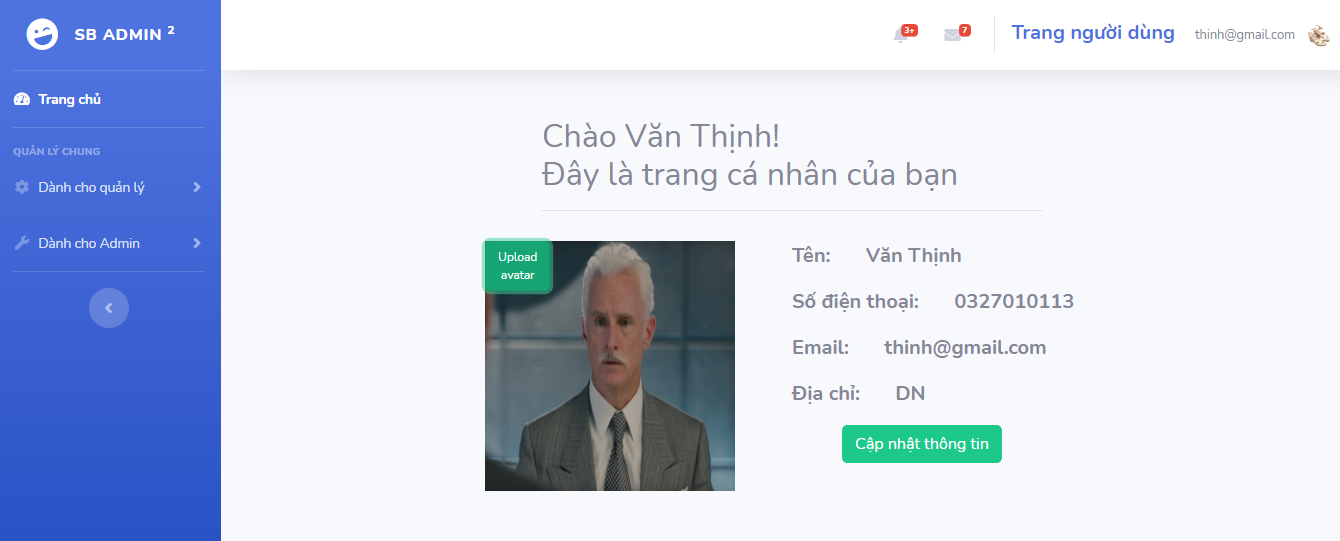


**2.2.1.7. Quản lý phân quyền**

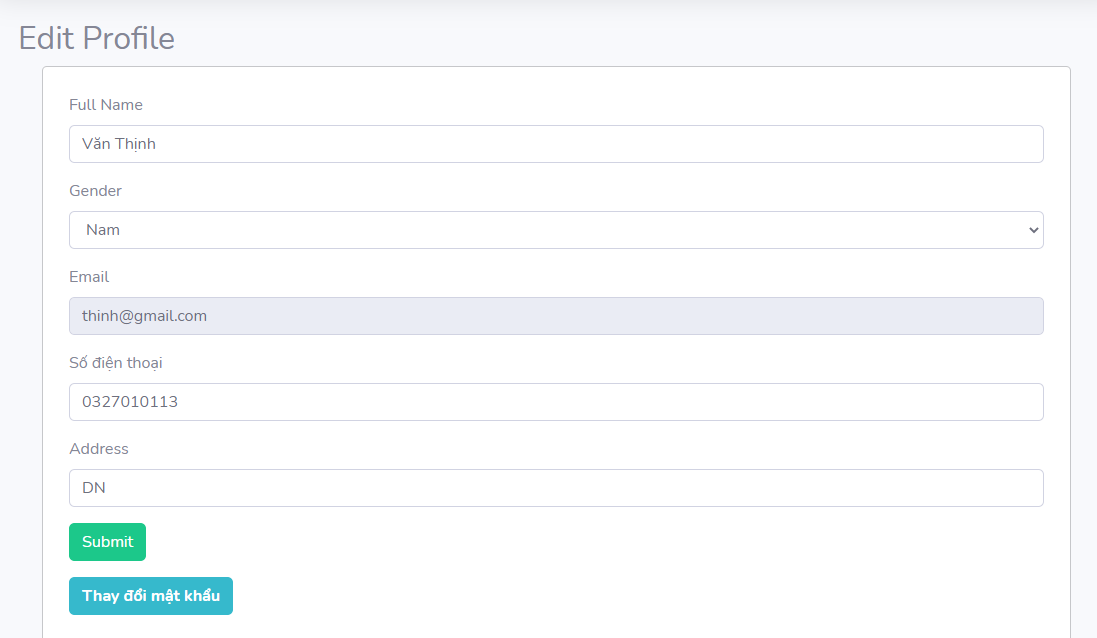
Trang hiển thị:



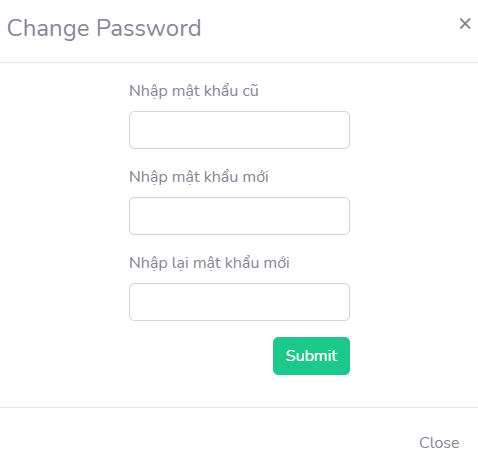
**2.2.1.8. Quản lý tài khoản cá nhân**



**2.2.1.8.1. Cập nhật tài khoản**

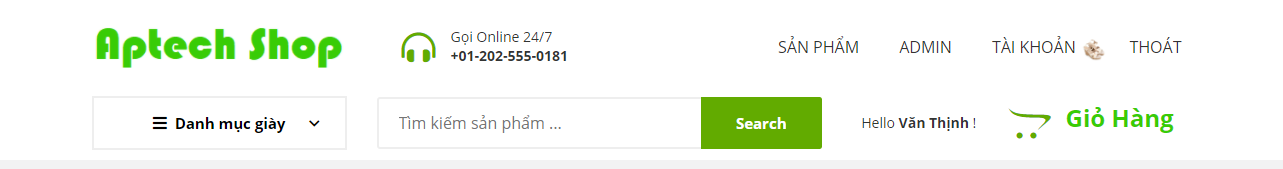


**2.2.1.8.2. Đổi mật khẩu**

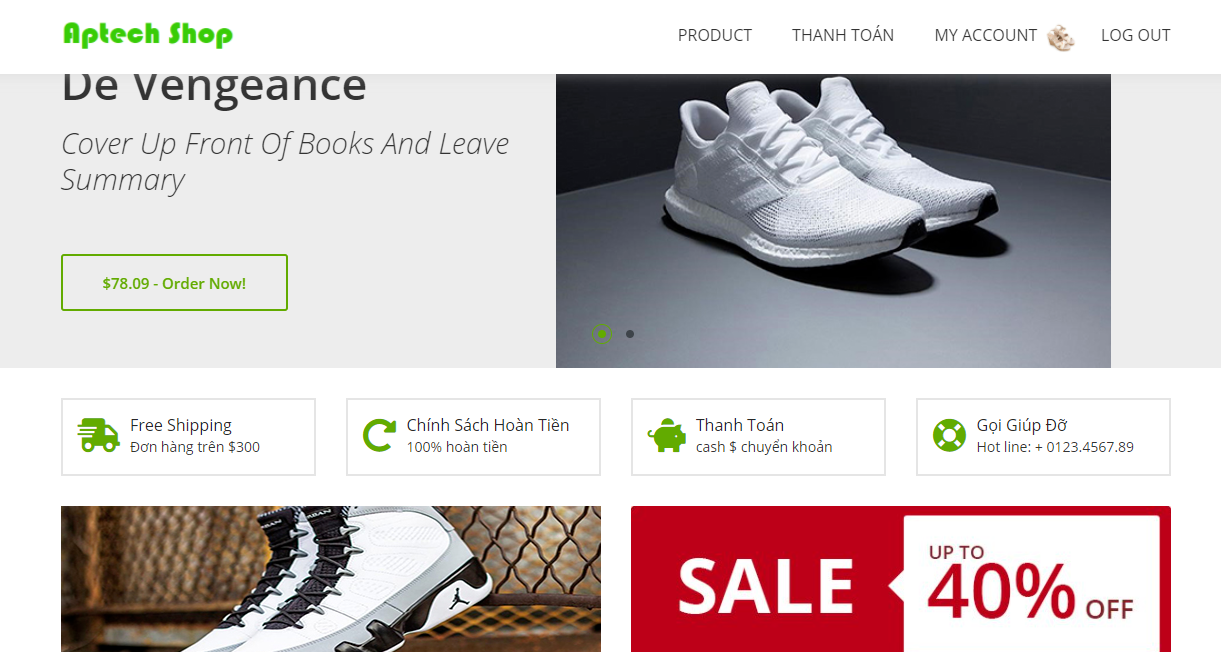


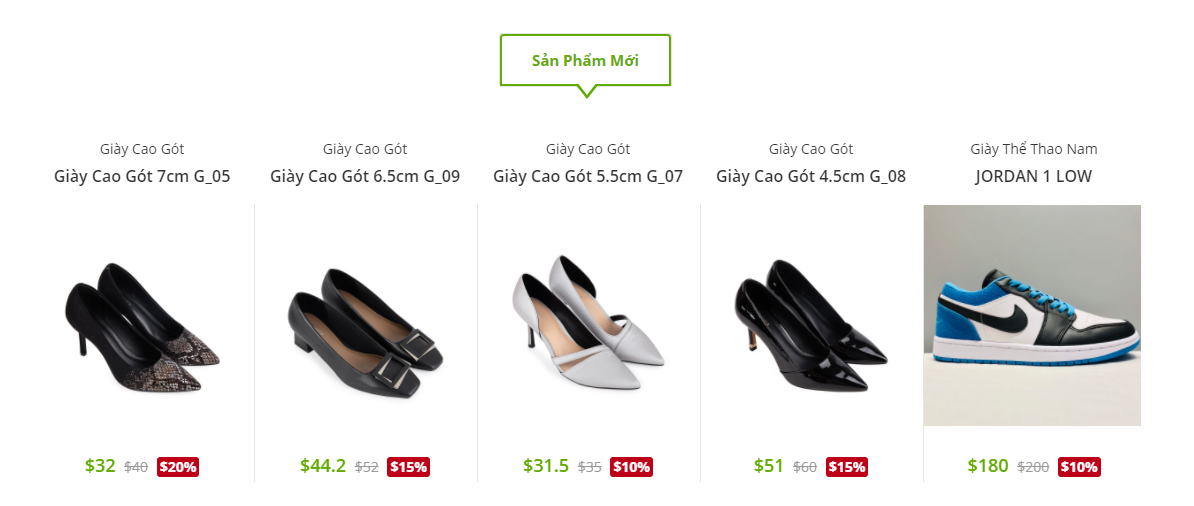
### Thiết kế phần khách hàng

**2.2.2.1. Cấu trúc chung (layout)**

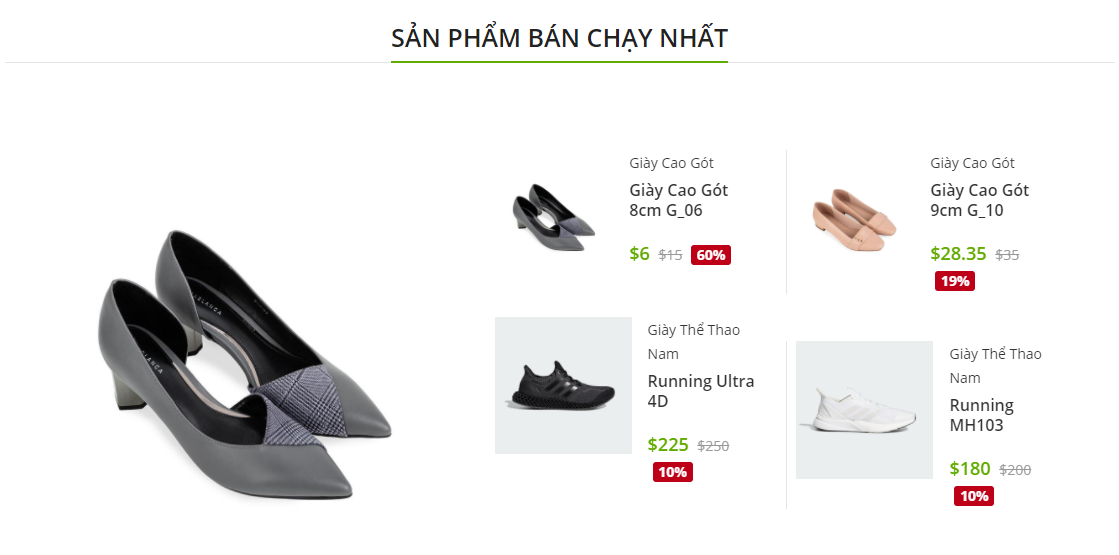


**2.2.2.2. Trang chủ**

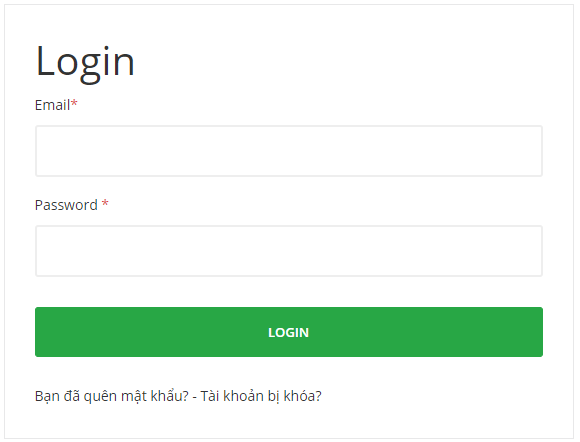




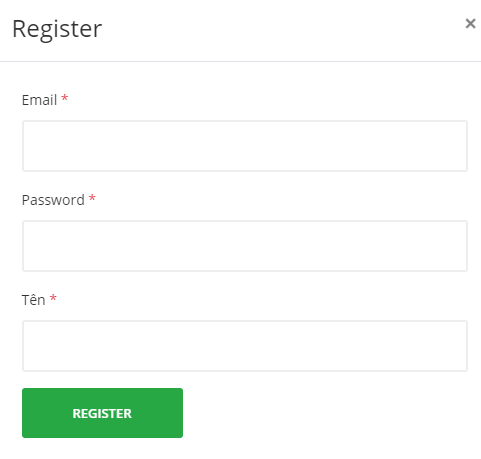




**2.2.2.3. Đăng nhập**



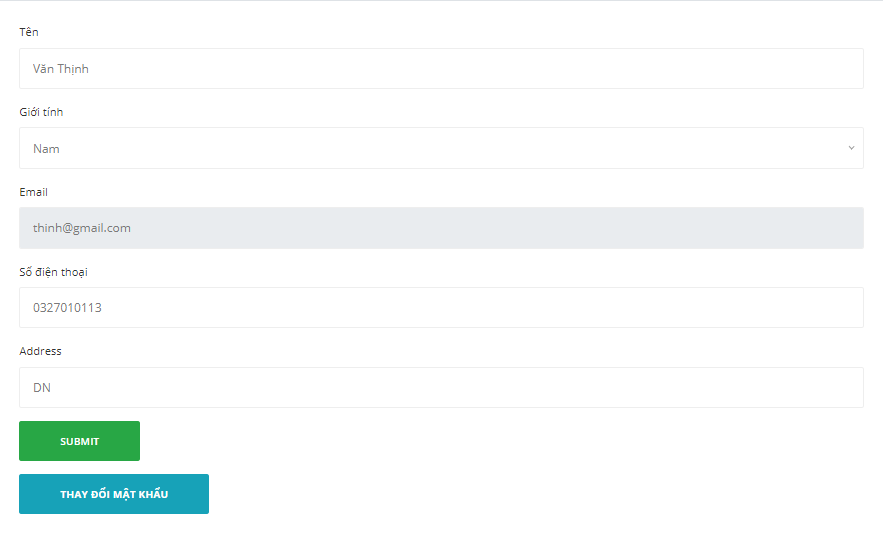
**2.2.2.4. Đăng ký**



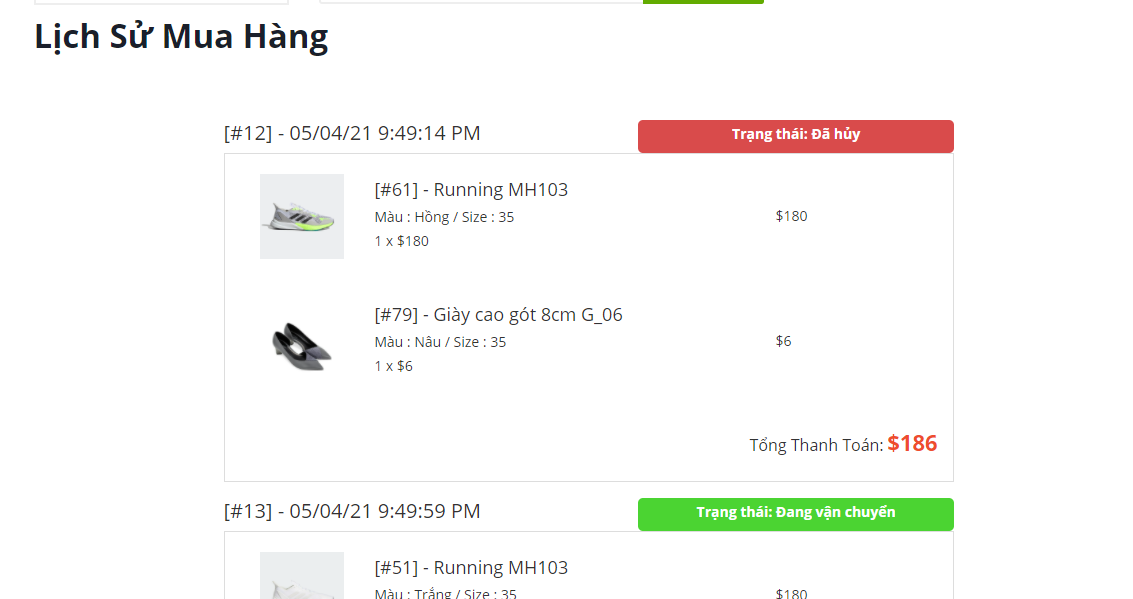
**2.2.2.5. Quản lý thông tin tài khoản**



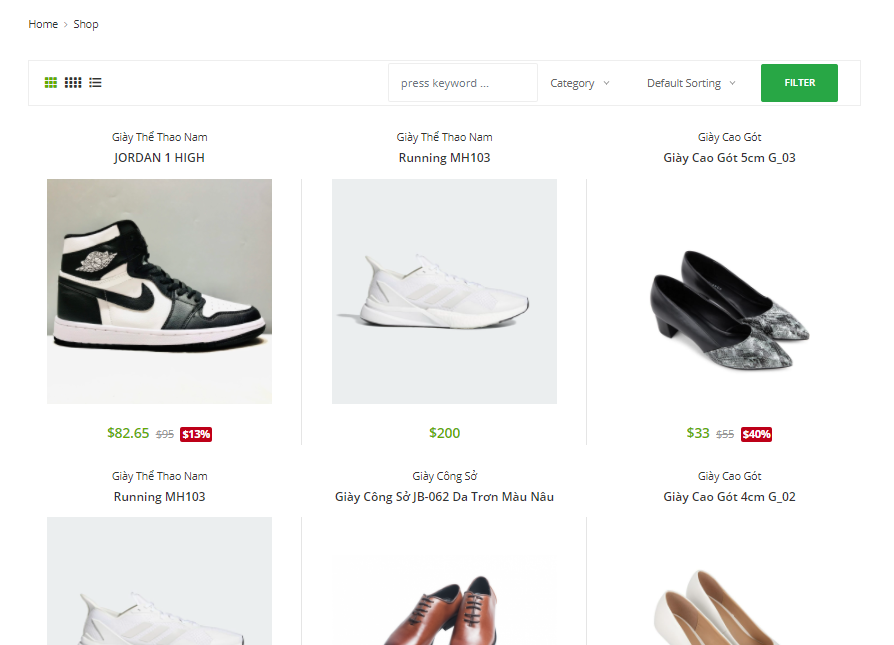
**2.2.2.5.1. Cập nhật thông tin tài khoản**



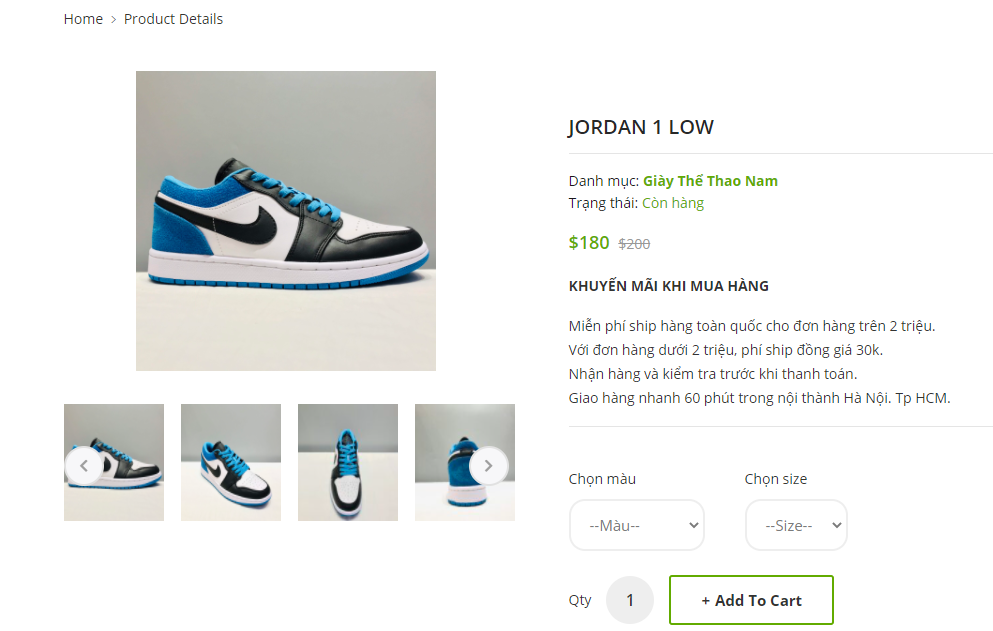
**2.2.2.5.2. Lịch sử mua hàng**



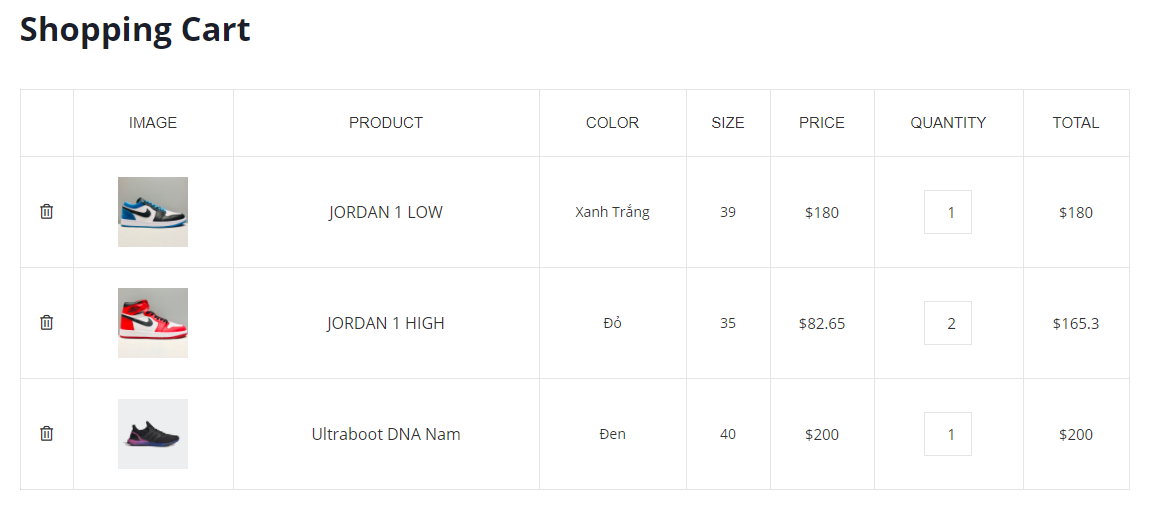
**2.2.2.6. Sản phẩm**

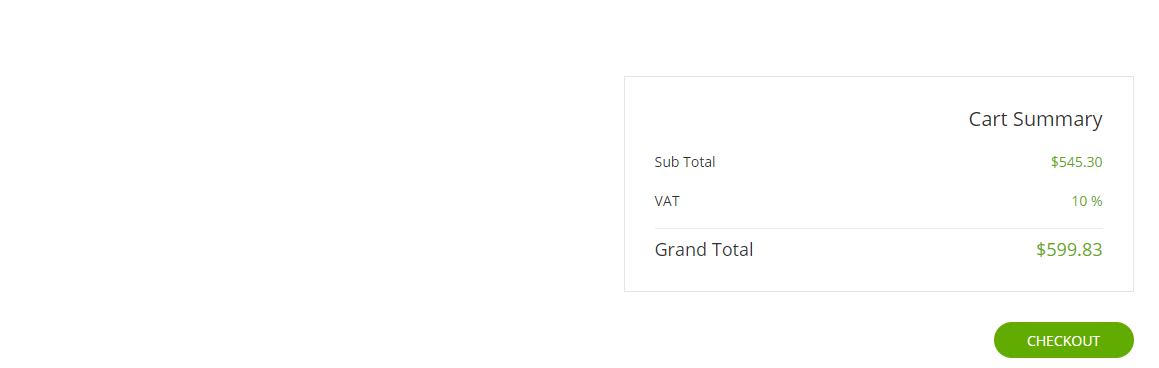


**2.2.2.7. Chi tiết sản phẩm**

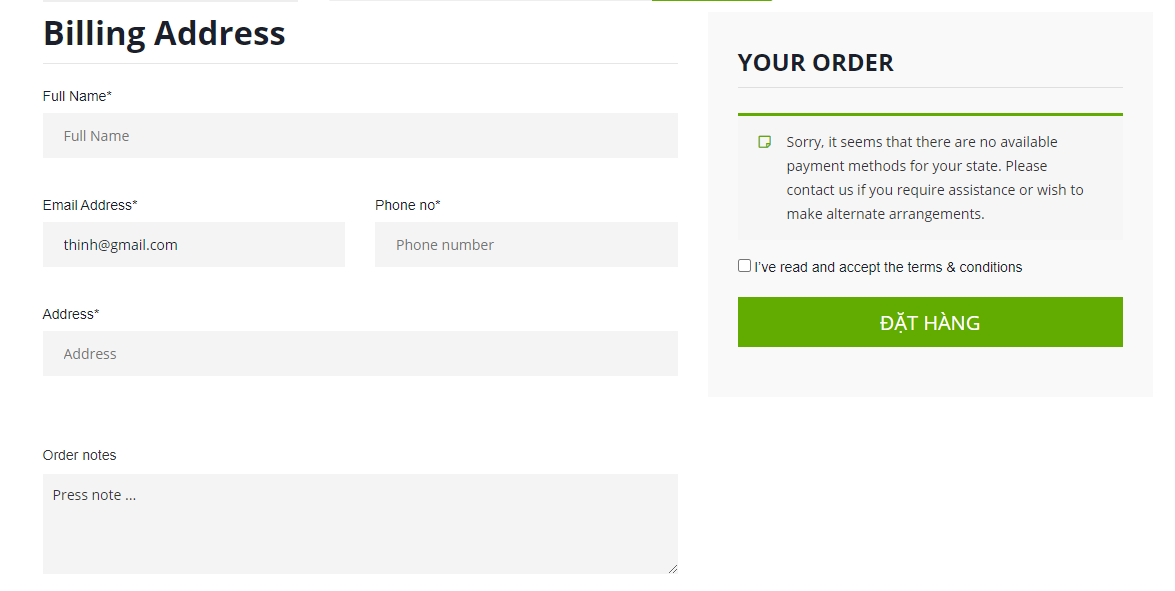


**2.2.2.8. Giỏ hàng**





**2.2.2.9. Thanh toán**



# Thực hiện viết mã

## Viết mã lập trình cho phần quản trị

### Quản lý sản phẩm

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | [HttpGet]  public ActionResult Index(int? page)  {  int pageSize = 10;  int pageNumber = page ?? 1;  var product = db.Products.OrderByDescending(x => x.Id);  return View(product.ToPagedList(pageNumber, pageSize));  } |
| Thêm mới | [HttpGet]  public ActionResult Create()  {  int id = int.Parse(User.Identity.Name);  ViewBag.UserID = id;  var showCate = db.Categories.OrderBy(x => x.Position);  List<StatusProduct> statusPro = db.StatusProducts.ToList();  SelectList statusProList = new SelectList(statusPro, "Id", "StatusName", "1");  ViewBag.StatusPro = statusProList;  List<Size> size = db.Sizes.ToList();  ViewBag.Size = size;  List<Color> color = db.Colors.ToList();  ViewBag.Color = color;  string date = DateTime.Today.ToString("yyyy-MM-dd");  ViewBag.date = date;  return View(showCate);  }  [HttpPost]  [ValidateInput(false)]  public ActionResult Create(ProductVM product, HttpPostedFileBase[] ProductImageId, List<int> SizeId, List<int> ColorId)  {  Product newPro = new Product()  {  ProductName = product.ProductName,  UnitPrice = product.UnitPrice,  DiscountRatio = product.DiscountRatio != null ? product.DiscountRatio : 0,  DiscountExpiry = product.DiscountExpiry,  CategoryId = product.CategoryId,  Description = product.Description,  CreatedDate = product.CreatedDate,  CreatedBy = product.CreatedBy,  StatusId = product.StatusId  };  db.Products.Add(newPro);  db.SaveChanges();  if (SizeId != null && SizeId.Count > 0)  {  ProductSize proSize;  foreach (var item in SizeId)  {  proSize = new ProductSize();  proSize.ProductId = newPro.Id;  proSize.SizeId = item;  db.ProductSizes.Add(proSize);  }  }  if (ColorId != null && ColorId.Count > 0)  {  ProductColor proColor;  foreach (var item in ColorId)  {  proColor = new ProductColor();  proColor.ProductId = newPro.Id;  proColor.ColorId = item;  db.ProductColors.Add(proColor);  }  }  if (ProductImageId[0] == null)  {  ///ViewBag.AddImage = "Bạn chưa thêm ảnh";  return RedirectToAction("Index");  }  ProductImage proImage;  foreach (var item in ProductImageId)  {  proImage = new ProductImage();  proImage.ImageUrl = item.FileName;  proImage.ProductId = newPro.Id;  db.ProductImages.Add(proImage);  }  db.SaveChanges();  string findUrl = ProductImageId[0].FileName;  var defaultImage = db.ProductImages.Where(x => x.ProductId == newPro.Id).Where(x => x.ImageUrl == findUrl).FirstOrDefault();  newPro.ProductImageId = defaultImage.Id;  db.SaveChanges();  string strFolder = Server.MapPath("~/Data/Products/" + newPro.Id);  if (!Directory.Exists(strFolder))  {  Directory.CreateDirectory(strFolder);  }  foreach (var item in ProductImageId)  {  item.SaveAs(strFolder + @"\" + item.FileName);  }  return RedirectToAction("Index");  } |
| Cập nhật theo mã | [HttpGet]  public ActionResult Edit(int id, string ErrorCreatBy)  {  if (ErrorCreatBy != null)  {  ViewBag.ErrorCreatBy = "Bạn không được edit nếu không phải là người tạo";  }  var product = db.Products.Find(id);  if (product == null)  {  return RedirectToAction("Index");  }  var listImage = db.ProductImages.Where(x => x.ProductId == id).ToList();  ViewBag.ListImage = listImage;  ViewBag.Url = "/Data/products/" + product.Id + "/" + product.ProductImage.ImageUrl;  var cates = db.Categories.ToList();  ViewBag.Cates = cates;  var listProStatus = db.StatusProducts.ToList();  ViewBag.ListProStatus = listProStatus;  var listSize = db.Sizes.ToList();  ViewBag.ListSize = listSize;  var listProSize = db.ProductSizes.Where(x => x.ProductId == id).ToList();  List<int> listSizeChecked = new List<int>();  foreach (var item in listProSize)  {  listSizeChecked.Add(item.SizeId);  }  ViewBag.ListSizeChecked = listSizeChecked;  var listColor = db.Colors.ToList();  ViewBag.ListColor = listColor;  var listProColor = db.ProductColors.Where(x => x.ProductId == id).ToList();  List<int> listColorChecked = new List<int>();  foreach (var item in listProColor)  {  listColorChecked.Add(item.ColorId);  }  ViewBag.ListColorChecked = listColorChecked;  return View(product);  }  [HttpPost]  [ValidateInput(false)]  public ActionResult Edit(ProductVM pro, HttpPostedFileBase[] ProductImageId, int[] ImgDelete, List<int> SizeId, List<int> ColorId)  {  int userId = int.Parse(User.Identity.Name);  ///Lấy user Admin  var userRole = db.UserRoles.Where(x => x.UserId == userId).Where(x => x.RoleId == 1).SingleOrDefault();  var product = db.Products.Find(pro.Id);  if (product == null)  {  return RedirectToAction("Index");  }  if (userId != product.CreatedBy && userRole == null)  {  return RedirectToAction("Edit", new { id = pro.Id, ErrorCreatBy = "Error" });  }  product.ProductName = pro.ProductName;  product.UnitPrice = pro.UnitPrice;  product.DiscountRatio = pro.DiscountRatio != null ? pro.DiscountRatio : 0;  product.DiscountExpiry = pro.DiscountExpiry;  product.CategoryId = pro.CategoryId;  product.Description = pro.Description;  product.CreatedDate = pro.CreatedDate;  product.CreatedBy = pro.CreatedBy;  product.StatusId = pro.StatusId;  string strFolder = "";  if (ProductImageId[0] != null)  {  ProductImage proImage;  foreach (var item in ProductImageId)  {  proImage = new ProductImage();  proImage.ImageUrl = item.FileName;  proImage.ProductId = pro.Id;  db.ProductImages.Add(proImage);  strFolder = Server.MapPath("~/Data/Products/" + pro.Id);  item.SaveAs(strFolder + @"\" + item.FileName);  }  }  foreach (var item in ImgDelete)  {  if (item > 0)  {  var proDelete = db.ProductImages.Find(item);  strFolder = Server.MapPath("~/data/products/" + proDelete.ProductId);  if (System.IO.File.Exists(strFolder + @"\" + proDelete.ImageUrl))  {  System.IO.File.Delete(strFolder + @"\" + proDelete.ImageUrl);  }  db.ProductImages.Remove(proDelete);  }  }  if (SizeId != null && SizeId.Count > 0)  {  List<int> listOldSize = new List<int>();  var proOldSize = db.ProductSizes.Where(x => x.ProductId == pro.Id).ToList();  foreach (var item in proOldSize)  {  listOldSize.Add(item.SizeId);  }  ProductSize proSize;  foreach (var item in SizeId)  {  if (!listOldSize.Contains(item))  {  proSize = new ProductSize();  proSize.ProductId = pro.Id;  proSize.SizeId = item;  db.ProductSizes.Add(proSize);  }  }  foreach (var item in proOldSize)  {  if (!SizeId.Contains(item.SizeId))  {  db.ProductSizes.Remove(item);  }  }  }  if (ColorId != null && ColorId.Count > 0)  {  List<int> listOldColor = new List<int>();  var proOldColor = db.ProductColors.Where(x => x.ProductId == pro.Id).ToList();  foreach (var item in proOldColor)  {  listOldColor.Add(item.ColorId);  }  ProductColor proColor;  foreach (var item in ColorId)  {  if (!listOldColor.Contains(item))  {  proColor = new ProductColor();  proColor.ProductId = pro.Id;  proColor.ColorId = item;  db.ProductColors.Add(proColor);  }  }  foreach (var item in proOldColor)  {  if (!ColorId.Contains(item.ColorId))  {  db.ProductColors.Remove(item);  }  }  }  db.SaveChanges();  return RedirectToAction("Edit", new { id = pro.Id });  } |
| Xem chi tiết theo mã | public ActionResult Detail(int id)  {  var listImage = db.ProductImages.Where(x => x.ProductId == id).ToList();  ViewBag.ListImage = listImage;  var product = db.Products.Find(id);  ViewBag.Url = "/Data/products/" + product.Id + "/" + product.ProductImage.ImageUrl;  var cates = db.Categories.ToList();  ViewBag.Cates = cates;  var listProStatus = db.StatusProducts.ToList();  ViewBag.ListProStatus = listProStatus;  var listSize = db.Sizes.ToList();  ViewBag.ListSize = listSize;  var listProSize = db.ProductSizes.Where(x => x.ProductId == id).ToList();  List<int> listSizeChecked = new List<int>();  foreach (var item in listProSize)  {  listSizeChecked.Add(item.SizeId);  }  ViewBag.ListSizeChecked = listSizeChecked;  var listColor = db.Colors.ToList();  ViewBag.ListColor = listColor;  var listProColor = db.ProductColors.Where(x => x.ProductId == id).ToList();  List<int> listColorChecked = new List<int>();  foreach (var item in listProColor)  {  listColorChecked.Add(item.ColorId);  }  ViewBag.ListColorChecked = listColorChecked;  return View(product);  } |
| Xóa theo mã | public ActionResult DeletePro(int id)  {  ///Tìm sản phẩm  var product = db.Products.Find(id);  string error = "Bạn không được quyền xóa sản phẩm này";  ///Tìm id đang đăng nhập  if (User.Identity.IsAuthenticated)  {  int userId = int.Parse(User.Identity.Name);  ///So sánh id đăng nhập có trùng với id tạo sp k  ///Nếu trùng thì mới cho xóa  if (product.CreatedBy == userId)  {  ///Set NULL cho trường này thì mới xóa được trong bảng ProImage  product.ProductImageId = null;  ///Hàm xóa ảnh  DeleteProId(product.Id);  db.Products.Remove(product);  db.SaveChanges();  return Content("OK");  }  ///Lấy ra object của id đăng nhập trong bảng UserRole  var account = db.UserRoles.Where(x => x.UserId == userId).FirstOrDefault();  ///Check null  if (account == null)  {  return Json(error);  }  ///Nếu = 1, tức là = Admin được được phép xóa tất cả ảnh  if (account.RoleId == 1)  {  product.ProductImageId = null;  DeleteProId(product.Id);  db.Products.Remove(product);  db.SaveChanges();  return Content("OK");  }  else  {  return Json(error);  }  }  return Json(error);  }  public void DeleteProId(int productId)  {  var RmProductImage = db.ProductImages.Where(x => x.ProductId == productId).ToList();  foreach (var item in RmProductImage)  {  db.ProductImages.Remove(item);  db.SaveChanges();  }  } |

### Quản lý danh mục

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | [HttpGet]  public ActionResult Index()  {  return View(db.Categories.OrderBy(x => x.Position).ToList());  } |
| Thêm mới | [HttpGet]  public ActionResult CreateCate()  {  return View();  }  [HttpPost]  public ActionResult CreateCate(Category cate)  {  Category newCate = new Category()  {  CategoryName = cate.CategoryName,  Position = cate.Position  };  db.Categories.Add(newCate);  db.SaveChanges();  return RedirectToAction("Index");  } |
| Cập nhật theo mã | [HttpGet]  public ActionResult \_EditCate(int id)  {  var cate = db.Categories.Find(id);  return PartialView(cate);  }  [HttpPost]  public ActionResult EditCate(int id, string categoryName, int position)  {  var cate = db.Categories.Find(id);  cate.CategoryName = categoryName;  cate.Position = position;  db.SaveChanges();  return RedirectToAction("Index");  } |
| Xóa theo mã | public ActionResult DeleteCate(int id)  {  Category t = db.Categories.Find(id);  db.Categories.Remove(t);  db.SaveChanges();  return Content("OK");  } |

### Quản lý đơn hàng

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | public ActionResult Index(int? page)  {  int pageSize = 10;  int pageNumber = page ?? 1;  var orders = db.Orders.OrderByDescending(x => x.Id);  return View(orders.ToPagedList(pageNumber, pageSize));  } |
| Thêm mới | [HttpGet]  public ActionResult Create()  {  var status = db.StatusOrders.ToList();  ViewBag.Status = status;  var colors = db.Colors.ToList();  ViewBag.Colors = colors;  var sizes = db.Sizes.ToList();  ViewBag.Sizes = sizes;  return View();  }  [HttpPost]  public ActionResult Create(OrderAdminVM postOrder)  {  Order newOrder = new Order()  {  StatusId = postOrder.StatusId,  CustomerName = postOrder.CustomerName,  CustomerEmail = postOrder.CustomerEmail,  CustomerPhone = postOrder.CustomerPhone,  CustomerAddress = postOrder.CustomerAddress,  OrderNote = postOrder.OrderNote,  OrderDate = DateTime.Now  };  db.Orders.Add(newOrder);  foreach (var item in postOrder.orderDetailAdminVMs)  {  var product = db.Products.Find(item.ProductId);  OrderDetail newProDetail = new OrderDetail()  {  OrderId = newOrder.Id,  ProductId = product.Id,  UnitPrice = product.UnitPrice,  DiscountRatio = product.DiscountRatio,  Quantity = item.Quantity,  ColorName = item.ColorName,  SizeName = item.SizeName  };  db.OrderDetails.Add(newProDetail);  }  db.SaveChanges();  return Json(data: "Bạn đã thêm 1 đơn hàng mới", JsonRequestBehavior.AllowGet);  } |
| Cập nhật theo mã | [HttpGet]  public ActionResult Edit(int id)  {  var order = db.Orders.Find(id);  var status = db.StatusOrders.ToList();  ViewBag.Status = status;  var oldOrderDetails = db.OrderDetails.Where(x => x.OrderId == id).ToList();  OrderDetailVM OrderDetail;  List<OrderDetailVM> listOrderDetail = new List<OrderDetailVM>();  double subTotal = 0;  foreach (var item in oldOrderDetails)  {  OrderDetail = new OrderDetailVM();  OrderDetail.Id = item.Id;  OrderDetail.ProductName = item.Product.ProductName;  OrderDetail.UnitPrice = item.UnitPrice - ((item.UnitPrice \* item.DiscountRatio.Value) / 100);  OrderDetail.Quantity = item.Quantity;  OrderDetail.Color = item.ColorName;  OrderDetail.Size = item.SizeName;  listOrderDetail.Add(OrderDetail);  double Tien = OrderDetail.UnitPrice \* OrderDetail.Quantity;  subTotal += Tien;  }  ViewBag.listOrderDetail = listOrderDetail;  ViewBag.SubTotal = (subTotal \* 1.1).ToString();  int FindProId = oldOrderDetails[0].ProductId;  var colors = db.ProductColors.Where(x => x.ProductId == FindProId).ToList();  ViewBag.Colors = colors;  var sizes = db.ProductSizes.Where(x => x.ProductId == FindProId).ToList();  ViewBag.Sizes = sizes;  return View(order);  }  [HttpPost]  public ActionResult Edit(Order order)  {  var editOrder = db.Orders.Find(order.Id);  if (editOrder == null)  {  return View("Index");  }  editOrder.CustomerName = order.CustomerName;  editOrder.CustomerAddress = order.CustomerAddress;  editOrder.CustomerEmail = order.CustomerEmail;  editOrder.CustomerPhone = order.CustomerPhone;  editOrder.StatusId = order.StatusId;  db.SaveChanges();  return RedirectToAction("Edit");  }  [HttpPost]  public ActionResult EditDetail(ChangeOrderDetail NewOrder)  {  var changeOrder = db.OrderDetails.Find(NewOrder.Id);  changeOrder.Quantity = NewOrder.Quantity;  changeOrder.ColorName = NewOrder.Color;  changeOrder.SizeName = NewOrder.Size;  db.SaveChanges();  return Json("Bạn đã thay đổi đơn hàng!");  } |
| Xem chi tiết theo mã | [HttpGet]  public ActionResult Detail(int id)  {  var order = db.Orders.Find(id);  var orderDetail = order.OrderDetails.ToList();  double subTotal = 0;  foreach (var item in orderDetail)  {  double unitPrice = item.UnitPrice - ((item.UnitPrice \* item.DiscountRatio.Value) / 100);  int quantity = item.Quantity;  double Tien = unitPrice \* quantity;  subTotal += Tien;  }  ViewBag.SubTotal = (subTotal \* 1.1).ToString();  return View(order);  } |
| Xóa theo mã | public ActionResult Delete(int id)  {  var order = db.Orders.Find(id);  var orderDetail = db.OrderDetails.Where(x => x.OrderId == id).ToList();  foreach (var item in orderDetail)  {  db.OrderDetails.Remove(item);  };  db.Orders.Remove(order);  db.SaveChanges();  return Content("OK");  } |

### Quản lý người dùng

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | [HttpGet]  public ActionResult Index()  {  var tbUsers = db.TbUsers.Include(t => t.Status);  return View(tbUsers.ToList());  } |
| Thêm mới | [HttpGet]  public ActionResult Create()  {  return View();  }  [HttpPost]  public ActionResult Create(string FullName, int Gender, string Email, string Password, string NumberPhone, string Address)  {  bool? GenderV2 = null;  if (Gender == 1)  {  GenderV2 = true;  }  else if (Gender == 0)  {  GenderV2 = false;  }  var emailUser = db.TbUsers.Where(x => x.Email == Email).SingleOrDefault();  if (emailUser != null)  {  ModelState.AddModelError("", "Tài khoản đã tồn tại");  return View();  }  TbUser newUser = new TbUser()  {  FullName = FullName,  Gender = GenderV2,  Email = Email,  Password = MySecurity.EncryptPassword(Password),  NumberPhone = NumberPhone,  CreatedDate = DateTime.Now,  Address = Address,  StatusId = 1,  CountLogin = 0  };  db.TbUsers.Add(newUser);  db.SaveChanges();  return RedirectToAction("Index");  } |
| Cập nhật theo mã | [HttpGet]  public ActionResult Edit(int id)  {  var user = db.TbUsers.Find(id);  var status = db.StatusUsers.ToList();  ViewBag.Status = status;  return View(user);  }  [HttpPost]  public ActionResult Edit(int Id, string FullName, int Gender, string NumberPhone, string Address, int StatusId)  {  bool? GenderV2 = null;  if (Gender == 1)  {  GenderV2 = true;  }  else if (Gender == 0)  {  GenderV2 = false;  }  var newUser = db.TbUsers.Find(Id);  newUser.FullName = FullName;  newUser.Gender = GenderV2;  newUser.NumberPhone = NumberPhone;  newUser.Address = Address;  newUser.StatusId = StatusId;  db.SaveChanges();  return RedirectToAction("Index");  } |
| Xem chi tiết theo mã | public ActionResult Details(int? id)  {  if (id == null)  {  return new HttpStatusCodeResult(HttpStatusCode.BadRequest);  }  TbUser tbUser = db.TbUsers.Find(id);  if (tbUser == null)  {  return HttpNotFound();  }  return View(tbUser);  } |
| Xóa theo mã | public ActionResult Delete(int id)  {  var user = db.TbUsers.Find(id);  db.TbUsers.Remove(user);  db.SaveChanges();  return Content("OK");  } |

### Quản lý phân quyền

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | [HttpGet]  public ActionResult Index(string checkRole, string checkEmail)  {  if (checkEmail != null)  {  ViewBag.HasAuthor = "Quyền " + checkRole + " đã được phân cho " + checkEmail;  }  var listAuthor = db.UserRoles;  return View(listAuthor.ToList());  } |
| Thêm mới | [HttpPost]  public ActionResult AddRole(int RoleId, int UserId)  {  var check = db.UserRoles.Where(x => x.RoleId == RoleId && x.UserId == UserId).FirstOrDefault();  if (check == null)  {  UserRole u = new UserRole()  {  RoleId = RoleId,  UserId = UserId  };  db.UserRoles.Add(u);  db.SaveChanges();  }  else  {  return RedirectToAction("Index", new { checkRole = check.Role.RoleName, checkEmail = check.User.Email });  }  return RedirectToAction("Index");  } |
| Xóa theo mã | [HttpPost]  public ActionResult Delete(int id)  {  var rs = db.UserRoles.Find(id);  if (rs != null)  {  db.UserRoles.Remove(rs);  db.SaveChanges();  }  return RedirectToAction("Index");  } |

## Viết mã lập trình cho phần người dùng

### Quản lý trang chủ

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | [HttpGet]  public ActionResult Index()  {  if (User.Identity.IsAuthenticated)  {  int userId = int.Parse(User.Identity.Name);  TbUser u = db.TbUsers.Find(userId);  };  var policies = db.Policies.ToList();  ViewBag.Policies = policies;  var products = db.Products.Where(x => x.StatusId == 1).OrderByDescending(x => x.Id).Take(12);  var lastedProducts = products.Select(s => new ProductClientVM  {  Id = s.Id,  ProductName = s.ProductName,  ImgUrl = s.ProductImageId != null ? "/Data/Products/" + s.Id + "/" + s.ProductImage.ImageUrl : "https://martialartsplusinc.com/wp-content/uploads/2017/04/default-image-620x600.jpg",  UnitPrice = s.UnitPrice.ToString(),  DiscountRatio = s.DiscountRatio.ToString(),  salePrice = (s.UnitPrice - ((s.UnitPrice \* s.DiscountRatio) / 100)).ToString(),  CategoryName = s.Category.CategoryName  });  ViewBag.LastedProducts = lastedProducts.ToList();  var disProducts = db.Products.Where(x => x.DiscountRatio > 20).OrderByDescending(x => x.DiscountRatio).Take(10);  var listDisProducts = disProducts.Select(s => new ProductClientVM  {  Id = s.Id,  ProductName = s.ProductName,  ImgUrl = s.ProductImageId != null ? "/Data/Products/" + s.Id + "/" + s.ProductImage.ImageUrl : "https://martialartsplusinc.com/wp-content/uploads/2017/04/default-image-620x600.jpg",  UnitPrice = s.UnitPrice.ToString(),  DiscountRatio = s.DiscountRatio.ToString(),  salePrice = (s.UnitPrice - ((s.UnitPrice \* s.DiscountRatio) / 100)).ToString(),  CategoryName = s.Category.CategoryName  });  ViewBag.ListDisProducts = listDisProducts.ToList();  var bestProducts = db.Products.Where(x => x.StatusId == 1);  var bestseller = db.OrderDetails.AsEnumerable()  .GroupBy(x => x.ProductId)  .Select(t => new { ID = t.Key, Value = t.Sum(s => s.Quantity) })  .OrderByDescending(x => x.Value).Select(s => s.ID);  var BestProducts = (from a in bestseller  join b in bestProducts  on a equals b.Id  select b).Take(4);  var listBestProducts = BestProducts.Select(s => new ProductClientVM  {  Id = s.Id,  ProductName = s.ProductName,  ImgUrl = s.ProductImageId != null ? "/Data/Products/" + s.Id + "/" + s.ProductImage.ImageUrl : "https://martialartsplusinc.com/wp-content/uploads/2017/04/default-image-620x600.jpg",  UnitPrice = s.UnitPrice.ToString(),  DiscountRatio = s.DiscountRatio.ToString(),  salePrice = (s.UnitPrice - ((s.UnitPrice \* s.DiscountRatio) / 100)).ToString(),  CategoryName = s.Category.CategoryName  });  ViewBag.ListBestProducts = listBestProducts.ToList();  return View();  } |

### Quản lý tài khoản

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Login | [HttpGet]  public ActionResult Login()  {  if (User.Identity.IsAuthenticated)  {  var id = int.Parse(User.Identity.Name);  ///Check role  var userRole = db.UserRoles.Where(x => x.UserId == id);  if (userRole != null)  {  foreach (var item in userRole)  {  if (item.RoleId == 1 || item.RoleId == 2)  {  return RedirectToAction("Index", "DashBoard", new { area = "Admin" });  }  }  }  return RedirectToAction("Index", "Home", new { area = string.Empty });  }  return View();  }  [HttpPost]  public ActionResult Login(string email, string password)  {  var user = db.TbUsers.Where(x => x.Email.Equals(email)).FirstOrDefault();  if (string.IsNullOrEmpty(email) && string.IsNullOrEmpty(password))  {  ModelState.AddModelError("", "Không được để trống ! ");  return View();  }  if (user == null)  {  ModelState.AddModelError("", "Tài khoản không tồn tại");  return View();  }  if (user.TimeLock != null)  {  var timeCurrent = DateTime.Now;  if (timeCurrent < user.TimeLock)  {  user.CountLogin += 1;  if (user.CountLogin >= 6)  {  user.StatusId = 2;  user.TimeLock = null;  db.SaveChanges();  ModelState.AddModelError("", "Tài khoản đã bị khóa, để lấy lại mật khẩu hãy click vào \"tài khoản đang bị khóa\"");  return View();  }  db.SaveChanges();  ModelState.AddModelError("", $"Tài khoản bị khóa cho đến {user.TimeLock}");  return View();  }  else  {  user.TimeLock = null;  db.SaveChanges();  }  }  if (user.StatusId == 2)  {  ModelState.AddModelError("", "Tài khoản đang bị khóa");  return View();  }  if (user.Password.Equals(MySecurity.EncryptPassword(password)))  {  Authen(user.Id);  user.CountLogin = 0;  db.SaveChanges();  return RedirectToAction("Index", "Home");  }  else  {  user.CountLogin += 1;  if (user.CountLogin == 3)  {  DateTime currentTime = DateTime.Now;  user.TimeLock = currentTime.AddMinutes(15);  db.SaveChanges();  ModelState.AddModelError("", "Tài khoản đã bị khóa sau 15 phút");  return View();  }  if (user.CountLogin >= 6)  {  user.StatusId = 2;  user.TimeLock = null;  db.SaveChanges();  ModelState.AddModelError("", "Tài khoản đã bị khóa, để lấy lại mật khẩu hãy click vào \"tài khoản đang bị khóa\"");  return View();  }  db.SaveChanges();  ModelState.AddModelError("", "Mật khẩu không đúng");  }  return View();  } |
| Register | [HttpPost]  public ActionResult Register(TbUser user)  {  var emailUser = db.TbUsers.Where(x => x.Email == user.Email).SingleOrDefault();  if (emailUser != null)  {  return Json("Email này đã tồn tại");  }  if (user.Password == null)  {  return Json("Email của bạn hợp lệ!");  }  TbUser newUser = new TbUser()  {  FullName = user.FullName,  Email = user.Email,  Password = MySecurity.EncryptPassword(user.Password),  StatusId = 1,  CreatedDate = DateTime.Now,  CountLogin = 0  };  db.TbUsers.Add(newUser);  db.SaveChanges();  Authen(newUser.Id);  //sendmail  EmailManagement.SendMail(user.Email, "Aptech Shose Shop",  "<h1>Hello [Name]! bạn đã đăng ký tài khoản thành công tại Aptech Shose Shop</h1>".Replace("[Name]", newUser.FullName));  return RedirectToAction("Index", "Home");  } |
| Hiển thị trang cá nhân | [HttpGet]  public ActionResult Index()  {  int userId = int.Parse(User.Identity.Name);  var user = db.TbUsers.Find(userId);  ViewBag.Url = "/Data/users/" + user.Id + "/" + user.Avatar;  return View(user);  } |
| Logout | [HttpGet]  public ActionResult Logout()  {  Session.Abandon();  Response.Cookies["UserId"].Expires = DateTime.Now.AddDays(-10);  FormsAuthentication.SignOut();  return RedirectToAction("Login");  } |
| Login to Checkout | [HttpPost]  public ActionResult LoginCkout(string Email, string Password)  {  var user = db.TbUsers.Where(x => x.Email.Equals(Email)).SingleOrDefault();  if (user == null)  {  ModelState.AddModelError("", "Tài khoản không tồn tại");  }  if (!string.IsNullOrEmpty(Email) && !string.IsNullOrEmpty(Password))  {  if (user.Password.Equals(MySecurity.EncryptPassword(Password)))  {  Response.Cookies["UserId"].Value = user.Id.ToString();  FormsAuthentication.SetAuthCookie(user.Id.ToString(), true);  var c = db.UserRoles.Where(x => x.UserId == user.Id).FirstOrDefault();  if (c != null && (c.RoleId == 1 || c.RoleId == 2)) //1 laf admin  {  return RedirectToAction("Index", "DashBoard", new { area = "Admin" });  }  else  {  return RedirectToAction("Index", "Checkout");  }  }  else  {  ModelState.AddModelError("", "Mật khẩu không đúng");  }  }  return Redirect(Request.UrlReferrer.ToString());  } |

**3.2.2.1. Cập nhật tài khoản**

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Upoad Avatar: | [HttpPost]  public ActionResult UploadAvatar(HttpPostedFileBase file)  {  string fileName = file.FileName;  int userId = int.Parse(User.Identity.Name);  TbUser user = db.TbUsers.Find(userId);  string strFolder = Server.MapPath("~/data/users/" + user.Id);  if (System.IO.File.Exists(strFolder + @"\" + user.Avatar))  {  System.IO.File.Delete(strFolder + @"\" + user.Avatar);  }  if (!Directory.Exists(strFolder))  {  Directory.CreateDirectory(strFolder);  }  file.SaveAs(strFolder + @"\" + fileName);  user.Avatar = fileName;  db.SaveChanges();  return Content("/data/users/" + user.Id + "/" + fileName);  } |
| Change Password: | [HttpGet]  public ActionResult ChangePW(int id)  {  TbUser user = db.TbUsers.Find(id);  return View(user);  }  [HttpPost]  public ActionResult ChangePW(ChangePW changePW)  {  int userId = int.Parse(User.Identity.Name);  TbUser user = db.TbUsers.Find(userId);  if (!user.Password.Equals(MySecurity  .EncryptPassword(changePW.oldPassword)))  {  return Json("Mật khẩu cũ không đúng, hãy nhập lại!");  }  user.Password = MySecurity.EncryptPassword(changePW.newPassword);  db.SaveChanges();  return Json("Mật khẩu đã thay đổi!");  } |
| Forgot Password: | [HttpPost]  public ActionResult ForgotPW(string email)  {  var user = db.TbUsers.Where(x => x.Email == email).SingleOrDefault();  if (user == null)  {  return Json("Email bạn nhập vào không đúng");  }  Random rdom = new Random();  var xPass = rdom.Next(100000, 1000000).ToString();  user.Password = MySecurity.EncryptPassword(xPass);  user.StatusId = 1;  user.CountLogin = 0;  db.SaveChanges();  EmailManagement.SendMail(user.Email, "Aptech Shose Shop",  "<h1>Hello [Name]! Mật khẩu mới của bạn là [newPass]</h1>"  .Replace("[Name]", user.FullName)  .Replace("[newPass]", xPass));  return Json("Bạn hãy kiểm tra email để lấy mật khẩu");  } |
| Edit Profile | [HttpGet]  public ActionResult EditProfile(int id)  {  var account = db.TbUsers.Find(id);  return View(account);  }  [HttpPost]  public ActionResult EditProfile(int Id, string FullName, int Gender, string NumberPhone, string Address)  {  bool? GenderV2 = null;  if (Gender == 1)  {  GenderV2 = true;  }  else if (Gender == 0)  {  GenderV2 = false;  }  var newUser = db.TbUsers.Find(Id);  newUser.FullName = FullName;  newUser.Gender = GenderV2;  newUser.NumberPhone = NumberPhone;  newUser.Address = Address;  db.SaveChanges();  return RedirectToAction("EditProfile");  } |

**3.2.2.2. Xem và cập nhật lịch sử đặt hàng**

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | [HttpGet]  public ActionResult Index()  {  if (User.Identity.IsAuthenticated)  {  List<OrderVM> listResult = new List<OrderVM>();  int userId = int.Parse(User.Identity.Name);  OrderVM o;  foreach (var item in db.Orders.Where(x => x.UserId == userId))  {  o = new OrderVM(item.Id, item.OrderDate.ToString(), item.Status.StatusName, listProduct(item.Id));  listResult.Add(o);  }  return View(listResult);  }  return RedirectToAction("Index", "Home", new { area = string.Empty });  }  [HttpGet]  public List<ProductOrderVM> listProduct(int orderId)  {  List<ProductOrderVM> list = new List<ProductOrderVM>();  double SubTotal = 0;  foreach (var item in db.OrderDetails.Where(x => x.OrderId == orderId))  {  var unitPrice = item.UnitPrice - ((item.UnitPrice \* item.DiscountRatio.Value) / 100);  var Url = item.Product.ProductImageId != null ? "/Data/products/" + item.Product.Id + "/" + item.Product.ProductImage.ImageUrl  : "linkImage";  double Tien = unitPrice \* item.Quantity;  var Total = Tien.ToString();  SubTotal += Tien;  list.Add(new ProductOrderVM(item.ProductId, Url, item.Product.ProductName, unitPrice.ToString(),  item.Quantity, item.SizeName, item.ColorName, Total.ToString(), SubTotal.ToString()));  }  return list;  } |
| Cancel | [HttpPost]  public ActionResult Cancel(int id)  {  var order = db.Orders.Find(id);  var orderDetail = db.OrderDetails.Where(x => x.OrderId == id).ToList();  order.StatusId = 4;  db.SaveChanges();  return Content("OK");  } |

### Quản lý sản phẩm

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Hiển thị toàn bộ | [HttpGet]  public ActionResult Index(string kw, int? cateid, string sortby, int? page)  {  int pageSize = 12;  int pageNumber = page ?? 1;  var showCate = db.Categories.ToList();  ViewBag.ShowCate = showCate;  ViewBag.Kw = kw;  ViewBag.Cateid = cateid;  ViewBag.SortBy = sortby;  IEnumerable<Product> products = db.Products.Where(x => x.StatusId == 1);  products = !string.IsNullOrEmpty(kw) ? getKw(kw, products) : products;  products = cateid != null ? getCate(cateid, products) : products;  products = !string.IsNullOrEmpty(sortby) ? getSortby(sortby, products) : products;  ///var listImage = db.ProductImages.Where(x => x.ProductId == s.Id)  var endProducts = products.Select(s => new ProductClientVM  {  Id = s.Id,  ProductName = s.ProductName,  ImgUrl = s.ProductImageId != null ? "/Data/Products/" + s.Id + "/" + s.ProductImage.ImageUrl : "https://martialartsplusinc.com/wp-content/uploads/2017/04/default-image-620x600.jpg",  UnitPrice = s.UnitPrice.ToString(),  DiscountRatio = s.DiscountRatio.ToString(),  salePrice = (s.UnitPrice - ((s.UnitPrice \* s.DiscountRatio) / 100)).ToString(),  CategoryName = s.Category.CategoryName  });  if (sortby == "name\_defau" || sortby == null)  {  Random rnd = new Random();  List<ProductClientVM> RdProduct = endProducts.ToList();  RdProduct = RdProduct.OrderBy(x => rnd.Next()).ToList();  return View(RdProduct.ToPagedList(pageNumber, pageSize));  }  return View(endProducts.ToPagedList(pageNumber, pageSize));  }  public IEnumerable<Product> getCate(int? cateid, IEnumerable<Product> p)  {  p = p.Where(x => x.CategoryId == cateid);  return p;  }  public IEnumerable<Product> getKw(string kw, IEnumerable<Product> p)  {  kw = kw.ToLower();  ViewBag.Kw = kw;  p = p.Where(x => x.ProductName.ToLower().Contains(kw));  return p;  }  public IEnumerable<Product> getSortby(string sortby, IEnumerable<Product> listProduct)  {  switch (sortby)  {  case "name":  listProduct = listProduct.OrderBy(s => s.ProductName);  break;  case "name\_desc":  listProduct = listProduct.OrderByDescending(x => x.ProductName);  break;  case "price":  listProduct = listProduct.OrderBy(x => x.UnitPrice);  break;  case "price\_desc":  listProduct = listProduct.OrderByDescending(x => x.UnitPrice);  break;  case "best\_sell":  var bestseller = db.OrderDetails.AsEnumerable()  .GroupBy(x => x.ProductId)  .Select(t => new { ID = t.Key, Value = t.Sum(s => s.Quantity) })  .OrderByDescending(x => x.Value).Select(s => s.ID);  var listProductMap = from a in bestseller  join b in listProduct  on a equals b.Id  select b;  listProduct = listProductMap.Concat(listProduct).Distinct().ToList();  break;  default:  listProduct = listProduct.OrderByDescending(x => x.Id);  break;  }  return listProduct;  } |
| Product Detail | [HttpGet]  public ActionResult Details(int id)  {  var product = db.Products.Find(id);  var ListUrl = db.ProductImages.Where(x => x.ProductId == product.Id).ToList();  List<string> MainListUrl = new List<string>();  for (int i = 0; i < ListUrl.Count; i++)  {  var eachUrl = "/Data/Products/" + product.Id + "/" + ListUrl[i].ImageUrl;  MainListUrl.Add(eachUrl);  }  ViewBag.MainListUrl = MainListUrl;  ViewBag.NewPrice = product.UnitPrice - (product.UnitPrice \* product.DiscountRatio) / 100;  var ListProCate = db.Products.Where(y => y.CategoryId == product.CategoryId)  .OrderByDescending(z => z.Id).Take(5).ToList();  var MainListProCate = ListProCate.Select(s => new ProductClientVM  {  Id = s.Id,  ProductName = s.ProductName,  ImgUrl = s.ProductImageId != null ? "/Data/Products/" + s.Id + "/" + s.ProductImage.ImageUrl : "https://martialartsplusinc.com/wp-content/uploads/2017/04/default-image-620x600.jpg",  UnitPrice = s.UnitPrice.ToString(),  DiscountRatio = s.DiscountRatio.ToString(),  salePrice = (s.UnitPrice - ((s.UnitPrice \* s.DiscountRatio) / 100)).ToString(),  CategoryName = s.Category.CategoryName  });  ViewBag.MainListProCate = MainListProCate.ToList();  var listColor = db.ProductColors.Where(x => x.ProductId == id).ToList();  ViewBag.MainListColor = listColor;  var listSize = db.ProductSizes.Where(x => x.ProductId == id).ToList();  ViewBag.MainListSize = listSize;  return View(product);  } |

### Quản lý giỏ hàng

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Chuyển sản phẩm vào giỏ hàng | [HttpGet]  public ActionResult Binding(string data)//string type  {  var cart\_items = JsonConvert.DeserializeObject<List<CartItem>>(data);  if (cart\_items == null || cart\_items.Count == 0)  {  return PartialView("\_Empty");  }  List<CartVM> lstCart = new List<CartVM>();  CartVM ca;  double subTotal = 0;  foreach (var item in cart\_items)  {  Product p = db.Products.Find(item.productid);  if (p != null)  {  double tien = (p.UnitPrice - ((p.UnitPrice \* p.DiscountRatio.Value) / 100)) \* item.quantity;  ca = new CartVM();  ca.ProductId = p.Id;  ca.ProductName = p.ProductName;  ca.ColorName = item.ColorName;  ca.SizeName = item.SizeName;  ca.UnitPrice = (p.UnitPrice - ((p.UnitPrice \* p.DiscountRatio) / 100)).ToString();  ca.Quantity = item.quantity;  ca.ImgUrl = p.ProductImageId != null ? "/Data/Products/" + p.Id + "/" + p.ProductImage.ImageUrl  : "https://martialartsplusinc.com/wp-content/uploads/2017/04/default-image-620x600.jpg";  ca.Total = tien.ToString();  lstCart.Add(ca);  subTotal = subTotal + tien;  }  }  double vat = 10;  ViewBag.VAT = vat;  ViewBag.SubTotal = subTotal.ToString("N2");  double tongToanBo = subTotal + (subTotal \* (vat / 100));  ViewBag.GrandTotal = tongToanBo.ToString("N2");  return PartialView("\_Cart", lstCart);  } |

### Quản lý thanh toán

|  |  |
| --- | --- |
| **Hành động** | **Câu mã lệnh code** |
| Thanh toán | [HttpPost]  public ActionResult Checkout(string data,  string CustomerName, string CustomerEmail,  string CustomerAddress,  string CustomerPhone, string OrderNote  )  {  if (CustomerName == "" || CustomerEmail == "" || CustomerPhone == "" || CustomerAddress == "")  {  return Content("Kiểm tra lại thông tin");  }  var cart\_items = JsonConvert.DeserializeObject<List<CartItem>>(data);  if (cart\_items == null || cart\_items.Count == 0)  {  return Content("Ko có sp trong giỏ hàng");  }  int? userName = null;  if (User.Identity.IsAuthenticated)  {  userName = int.Parse(User.Identity.Name);  }  Order orders = new Order()  {  CustomerName = CustomerName,  CustomerEmail = CustomerEmail,  CustomerAddress = CustomerAddress,  CustomerPhone = CustomerPhone,  OrderDate = DateTime.Now,  OrderNote = OrderNote,  UserId = userName != null ? userName : userName,  StatusId = 1  };  db.Orders.Add(orders);  foreach (var item in cart\_items)  {  Product p = db.Products.Find(item.productid);  OrderDetail od = new OrderDetail()  {  OrderId = orders.Id,  ProductId = item.productid,  UnitPrice = p.UnitPrice,  DiscountRatio = p.DiscountRatio,  Quantity = item.quantity,  ColorName = item.ColorName,  SizeName = item.SizeName  };  db.OrderDetails.Add(od);  }  db.SaveChanges();  return Content("OK");  } |