INTERNATIONAL IT TRAINING SYSTEM

APTECH VIET NAM

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

GRADUATION REPORT

TOPIC

SHOES SALE WEBSITE DESIGN

LANGUAGE OF ASP.NET MVC PROGRAMMING LANGUAGE

Class: Batch 153

Member: Đặng Văn Thịnh (C)

Nguyễn Đức Vĩnh Thắng

Huỳnh Minh Hiếu

Instructors: Nguyễn Khánh

Da Nang, 12 - 04 - 2021

**PREFACE**

This project is done by team members in the international .NET programmer course of the Softech Aptech training program.

The reason for choosing the topic is that when creating a sales website, students can design many features and research on the most realistic website that can interact with users .

The goal of choosing the topic is that students can create a shoe selling website to help shoe shops have a website to sell online.

All of this project are self-conceptualized and implemented by members, the languages ​​used are HTML, CSS, BOOTSTRAP, JAVASCRIPT, JQUERY, AJAX, LINQ, C #, ASP.NET, ENTITY FRAMEWORK , SQL SERVER .

TABLE OF CONTENTS

1. DIVISION INTERPRETATION 2

1.1. Current state 2

1.2. Request generation system 2

1.3. Use case 3

1.3.1. Functional specification 3

1.3.2. Specification of entities 3

2. DESIGN 7

2.1. Database design 7

2.1.1. Entity relationship diagram 7

2.1.2. Detailed design of the entities 9

2.2. Design the theme 14

2.2.1. Design the admin section 14

2.2.2. Design the customer part 25

3. Execute coding 33

3.1. Write code for the administration 33

3.1.1. Products Management 33

3.1.2. Categories management 43

3.1.3. Orders management 44

3.1.4. Users management 48

3.1.5. Authorization management 51

3.2. Write the code for the user section 52

3.2.1. Home 52

3.2.2. Account management 54

3.2.3. Products management 63

3.2.4. Cart management 67

3.2.5. Checkout management 68

# DIVISION INTERPRETATION

## Current state

Currently, online shoe stores are present in all provinces of Vietnam, and sell a variety of shoes. These online stores currently do not have small-scale websites to introduce their products to users. They also do not have an automated management environment to make their business easier.

## Request generation system

Aptech-Shose-Shop wants to have a website to introduce products and collect opinions of consumers about products in business. Website also has functions for administrators so that they can manage products, classify products, manage customers, manage customer comments about products as well as inventory of goods by type ...

Specific requirements are set out as follows

* Website sales for consumers
* Structure them all of the web pages in the right website
* Contains a list of goods for customers to easily search for goods by type
* Search for goods by name
* Show list of items to buy most
* Login / consumer information after logging in
* The main page of the website

Display special items designated by admin staff. Special items can be displayed in a way that is easy to see (slideshow can be used)

* Display page of goods

When the user selects the product category or searches by name, a list of goods that satisfy the requirements is displayed. Each item must show picture, name, unit price and discount ne u there. When users click on the photo of the goods will go to a page showing detailed tiế t of goods and increase the number of views.

* Detail of goods
* Sales website for administrators
* Management pages

These pages should be equipped with proper view, add, delete, edit, search and pagination functions.

* Security requirements
* All entry forms must be controlled data logically

Customer not logged in not view account information, do not pour i password, update account information

* Only the administrators are allowed to perform the administrative functions
* Requirements for technology environment

- Website must be built with ASP.NET and SQL Server

## Use case

### Functional specification

**1.3.1.1. Functions for customers**

- The general structure of the web pages

* Search by category, by name
* Filter products
* View detailed information
* Add products to cart
* Account management

**1.3.1.2. Functions for administration**

- Manage products

* Manage categories
* Manage orders
* Manage users and authorization

### Specification of entities

The following are the entities and their properties required to build the Shose-shop website

* 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

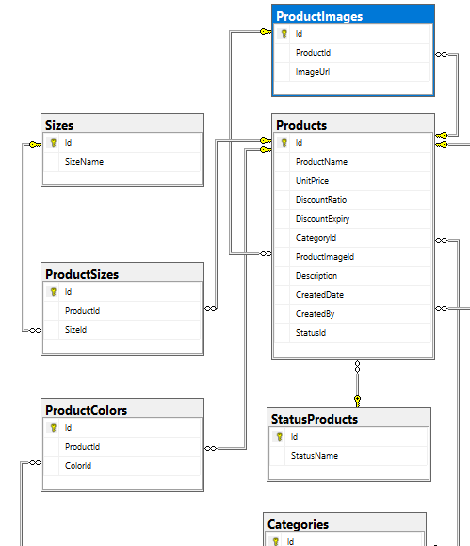
# DESIGN

## Database design

Based on the entity specification and specifically working with Shose-shop, you can design a detailed and relational diagram of the entities so that you have enough information to install the database onto SQL Server.

### Entity relationship diagram

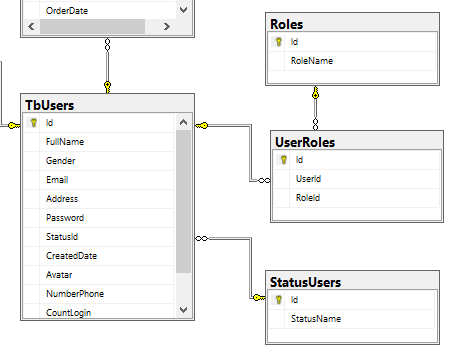
* Model entities in a relationship with the Products table:

****

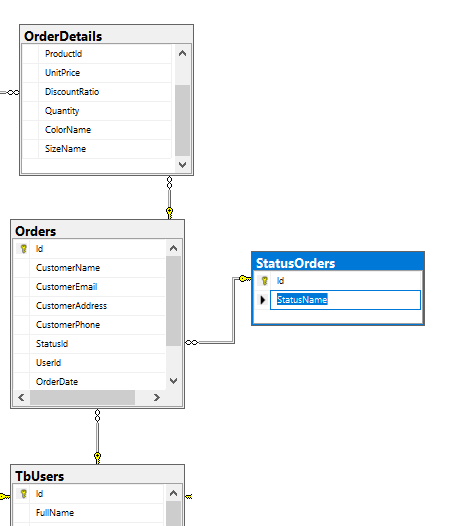
* Table Products – ProductImages: Id(Products ) – ProductId(ProductImages)

ProductImageId(Products) – Id(ProductImages)

* Table Products – StatusProducts: StatusId(Products) – Id(StatusProducts)
* Table Products – Categories: CategoryId(Products ) – Id(Categories)
* Table Products – ProductSizes: Id(Products ) – ProductId(ProductSizes)
* Table Products – ProductColors: Id(Products ) – ProductId(ProductColors)
* A model of entities related to the Users table



* Table TbUsers – StatusUsers: StatusId(TbUsers) –Id(StatusUsers)
* Table TbUsers – UserRoles: Id(TbUsers) – UserId(UserRoles)
* A model of the entities related to the Orders table



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

### Detailed design of the entities

**2.1.2.1. Products**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| 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**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| 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**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| 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**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk,not null, tự tang | Mã Size |
| SizeName | string | Not null | Tên Size |

**2.1.2.5. ProductSizes**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk, not null, tự tăng | Mã ProductSize |
| ProductId | Int | Fk, not null | Khóa ngoại đến Table Products |
| SizeId | int | Fk,not null | Khóa ngoại đến Table Size |

**2.1.2.6. Colors**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk, not null, tự tăng | Mã màu |
| ColorName | string | Not null | Tên màu |

**2.1.2.7. ProductColors**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk, not null, tự tăng | Id |
| ColorId | Int | FK,not null | Khóa ngoại đến Table Colors |
| ProductId | int | FK,not null | Khóa ngoại đến Table Products |

**2.1.2.8. StatusProducts**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk, not null, tự tăng | Mã tình trạng |
| StatusName | string | Not null | Tên tình trạng |

**2.1.2.9. Orders**

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

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| 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 | Note mua hàng |

**2.1.2.10. OrderDetails**

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

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| 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**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| 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**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| 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**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk, not null, tự tăng | Id |
| RoleName | string | not null | Tên role |

**2.1.2.14. UserRoles**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk, not null, tự tăng | Id |
| UserId | Int | FK,not null | Khóa ngoại đến Table TblUser |
| RoleId | int | FK,not null | Khóa ngoại đến Table Roles |

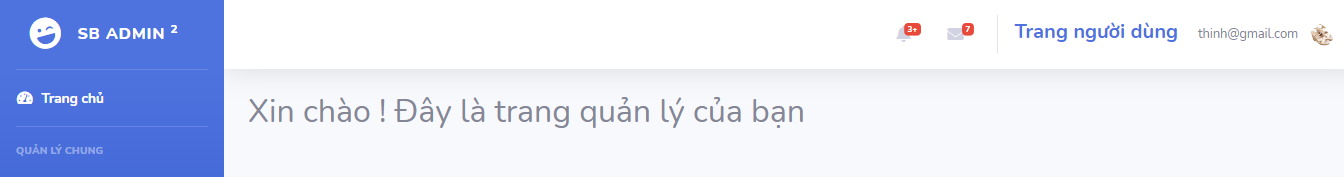
**2.1.2.15. StatusUsers**

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Type | Constraint | Note |
| Id | Int | Pk, not null, tự tăng | Id |
| StatusName | string | not null | Tình trạng User |

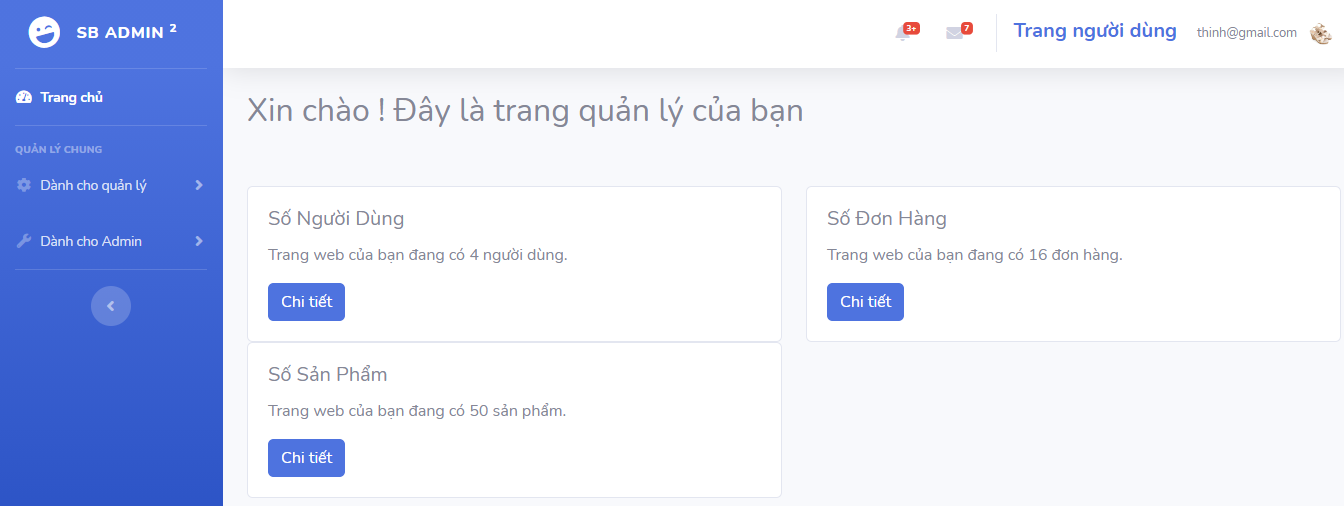
## Design the theme

### Design the admin section

**2.2.1.1. General structure (layout)**



**2.2.1.2. Home**

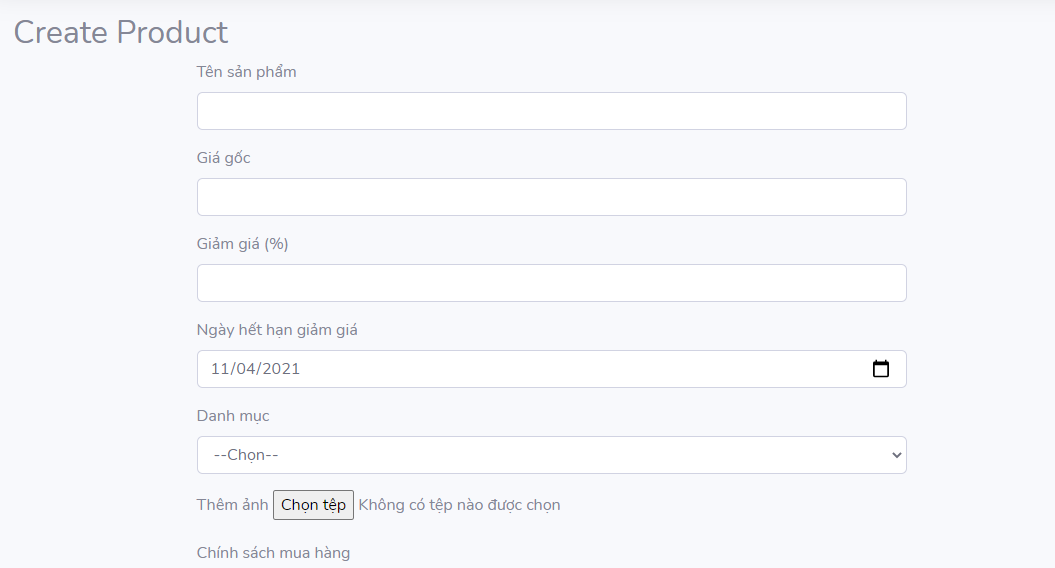


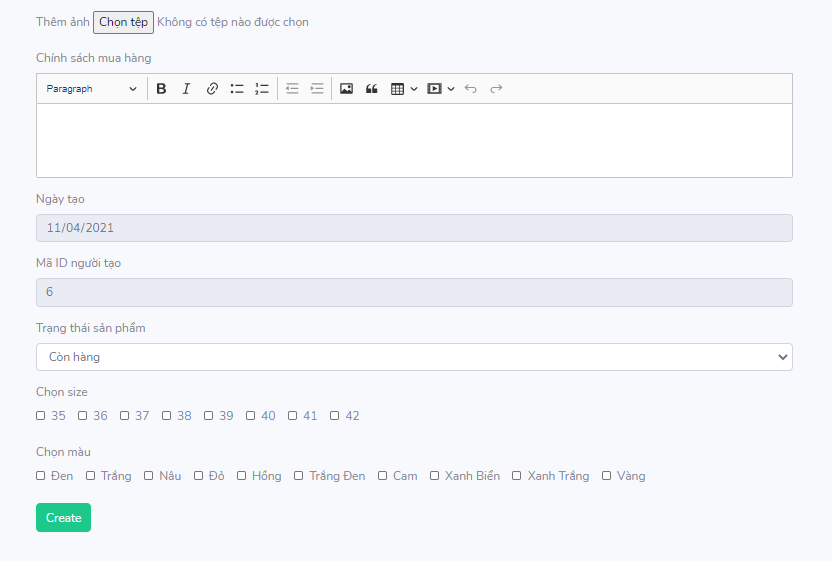
**2.2.1.3. Products Management**

Show page:

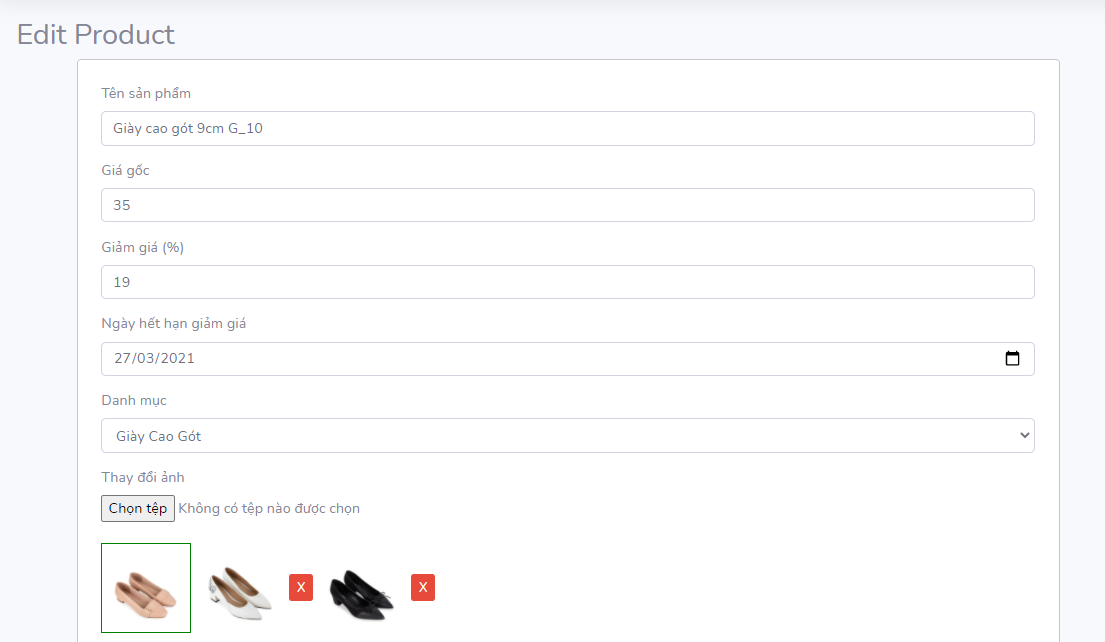


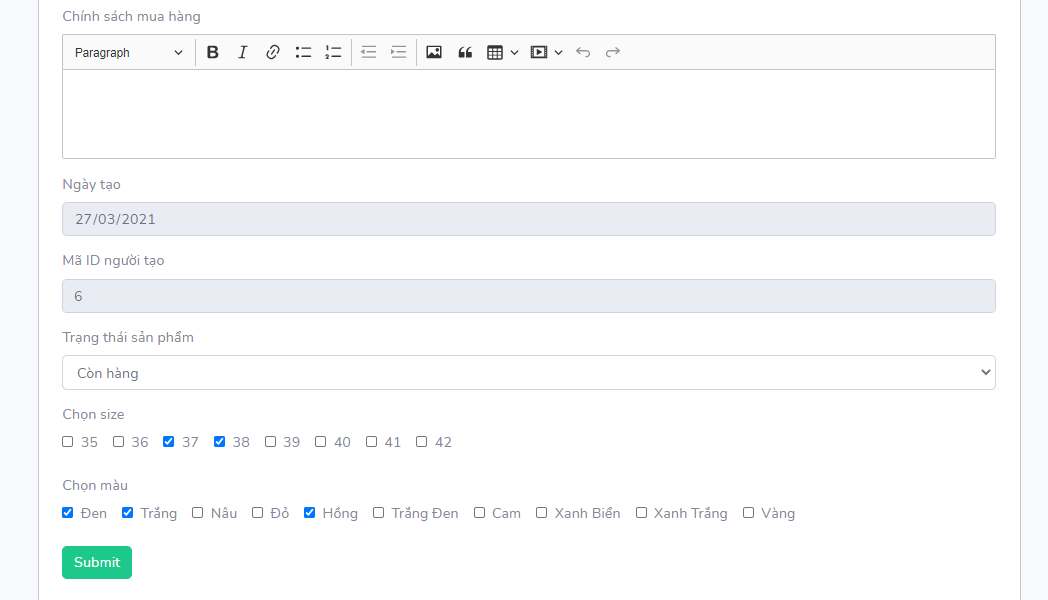
Create new





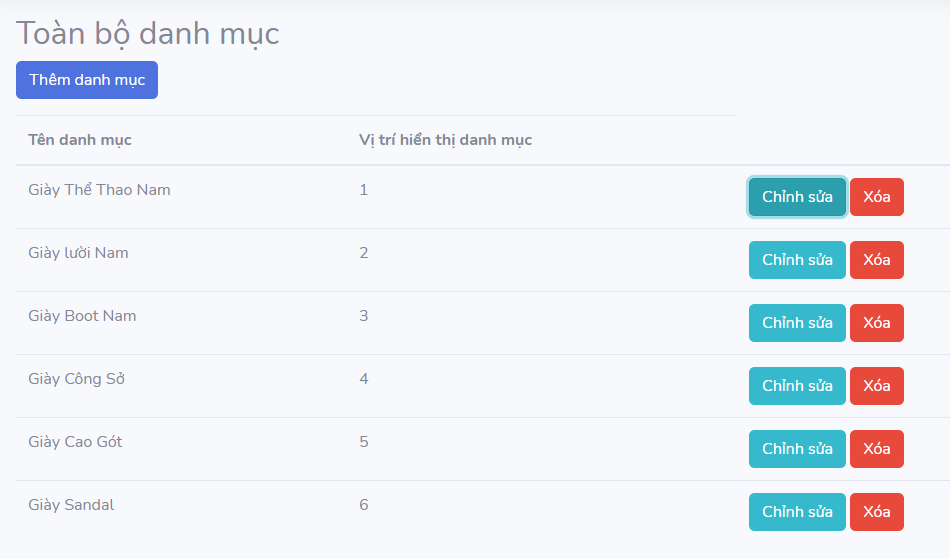
Edit



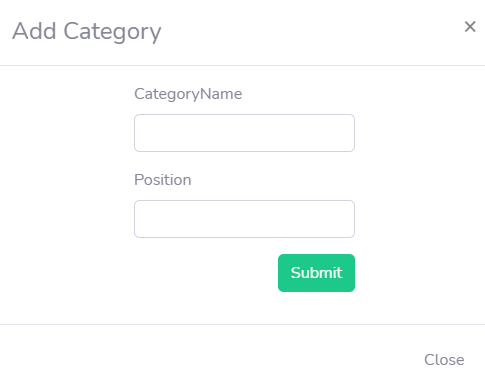


**2.2.1.4. Categories Management**

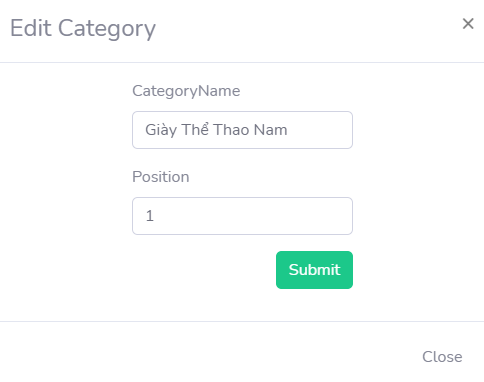
Show page:



Add:

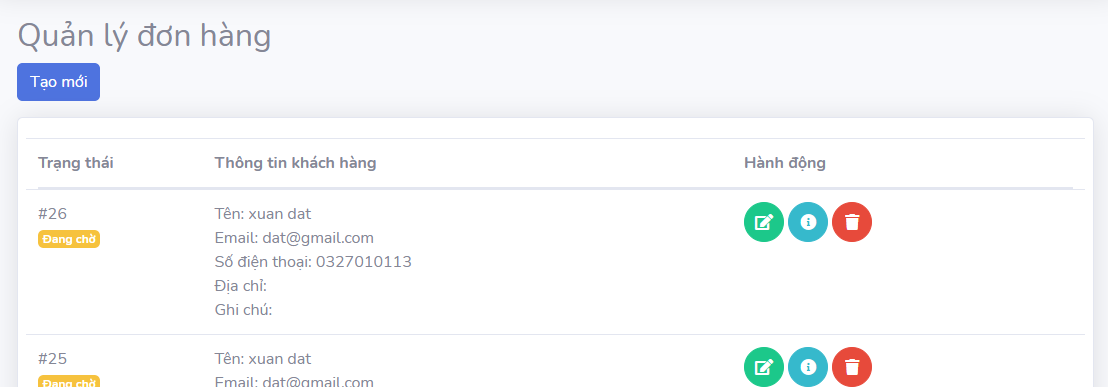


Edit

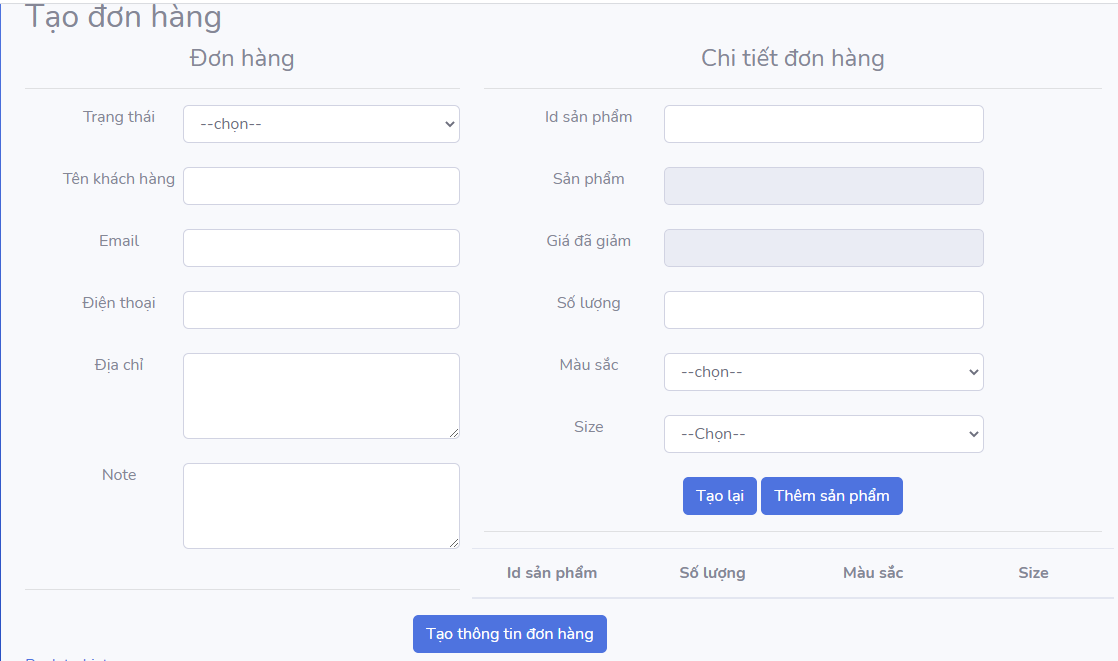


**2.2.1.5. Orders Management**

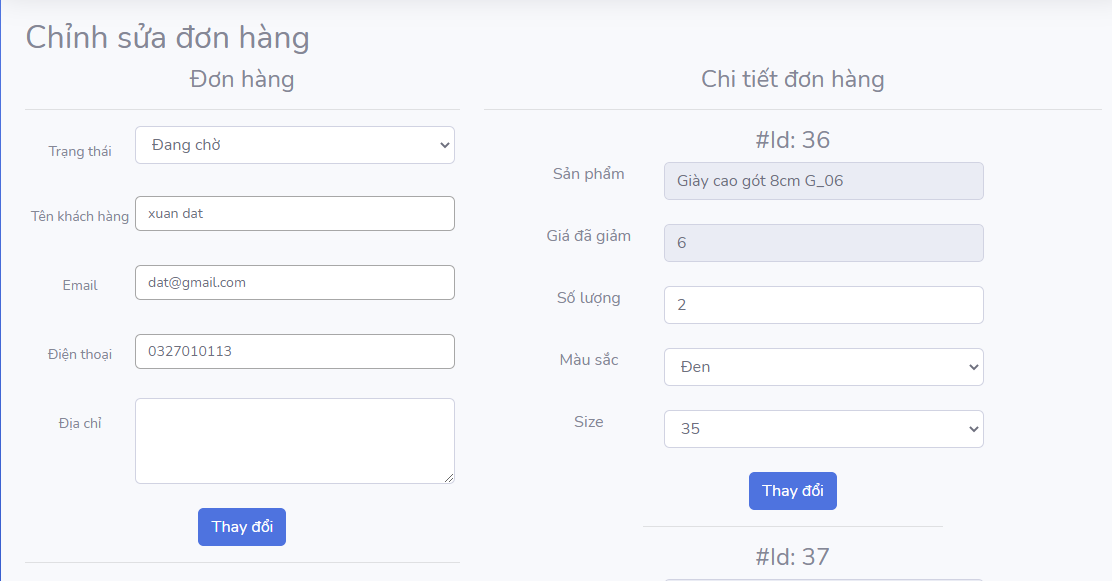
Show page:



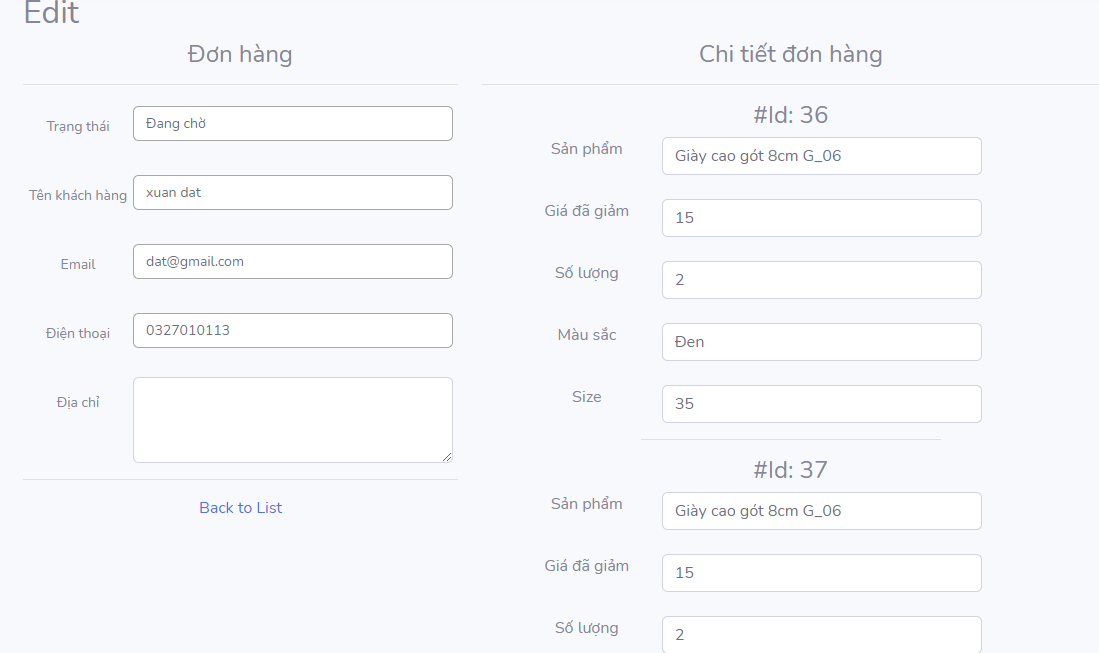
Create new:



Edit

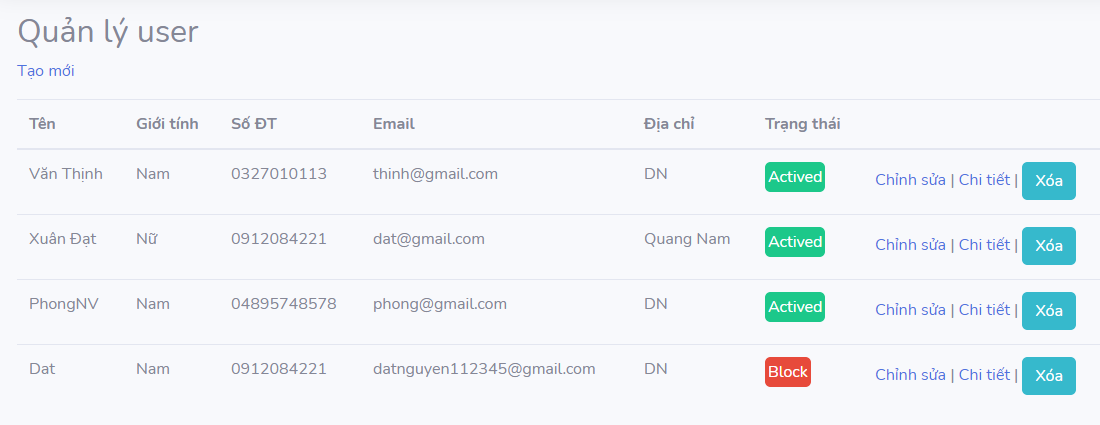


Detail:

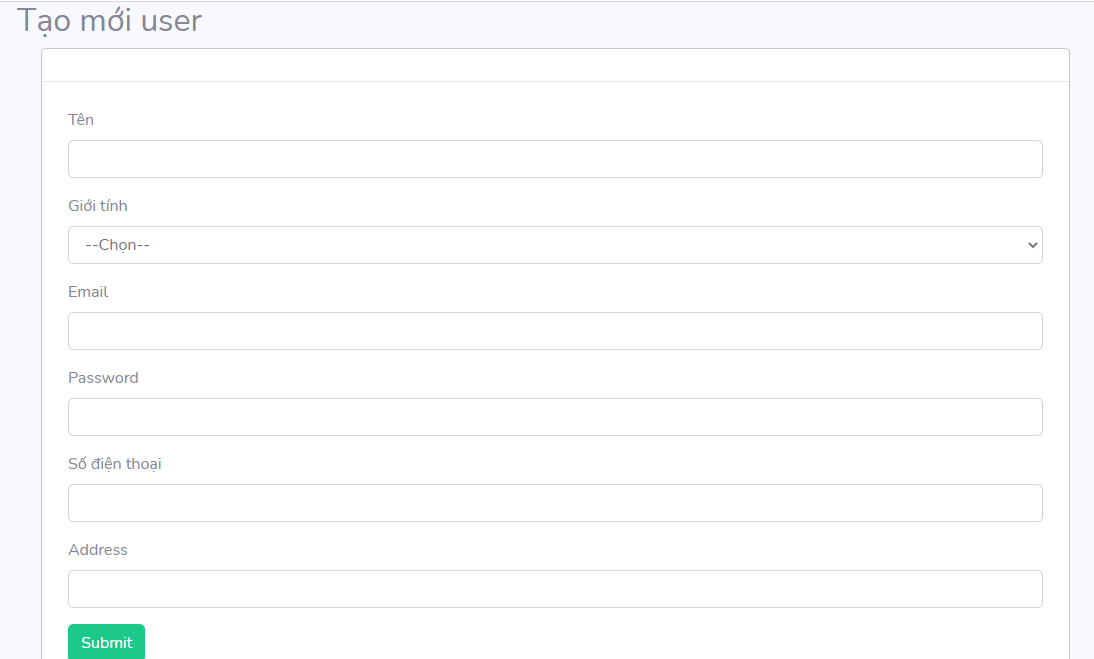


**2.2.1.6. Users management**

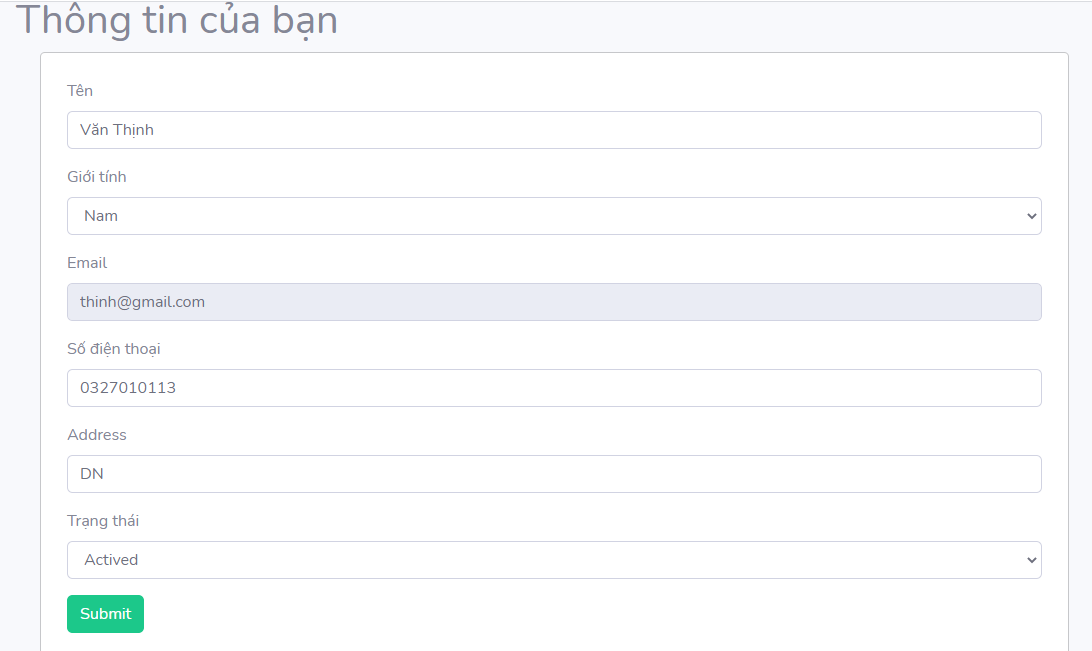
Show page:



Add user

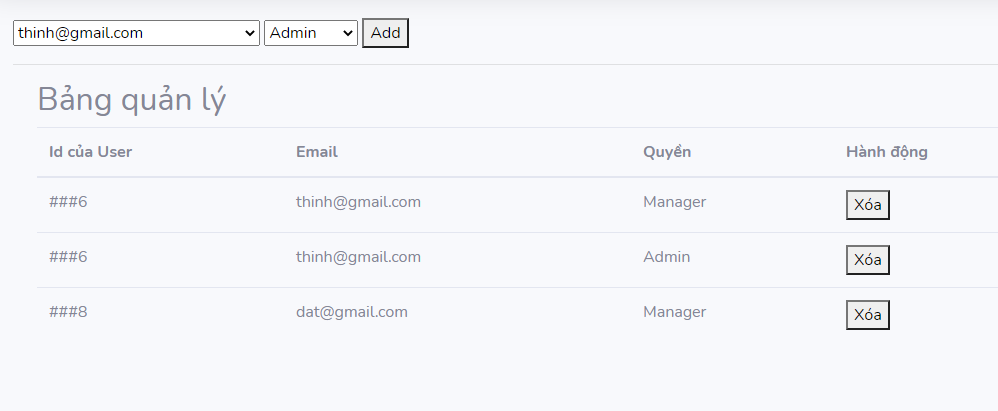


Edit user

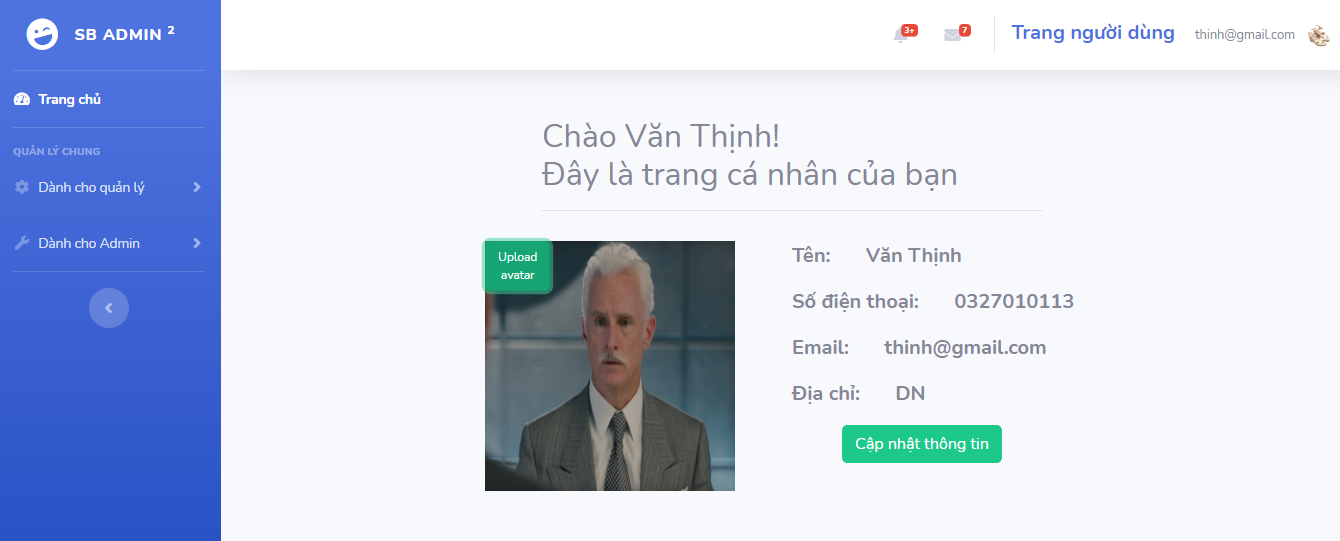


**2.2.1.7. Autorization management**

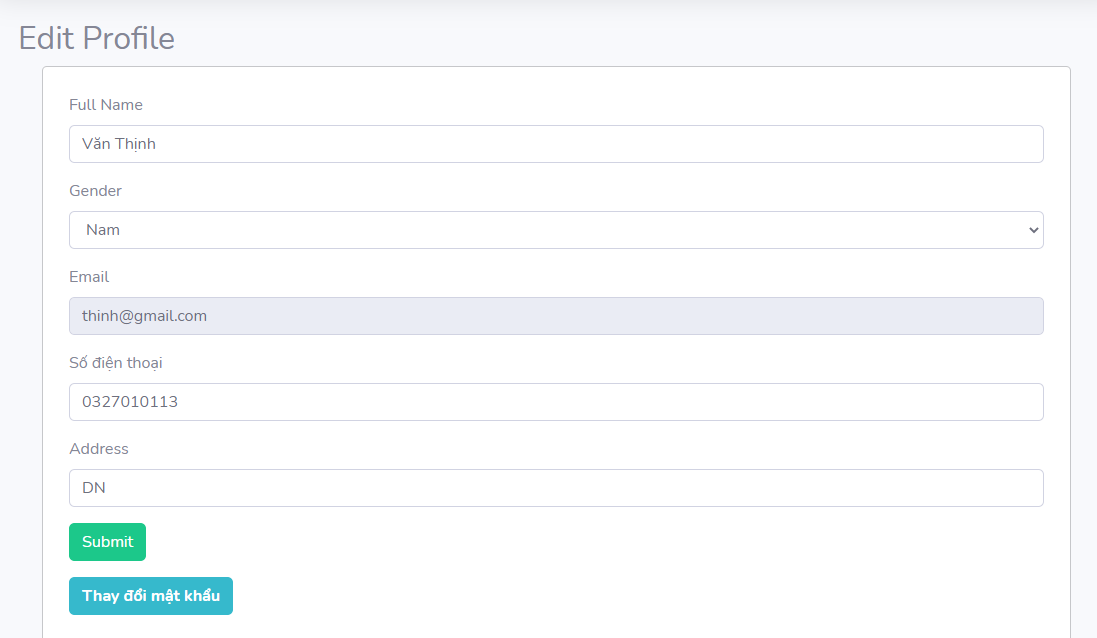
Show page:



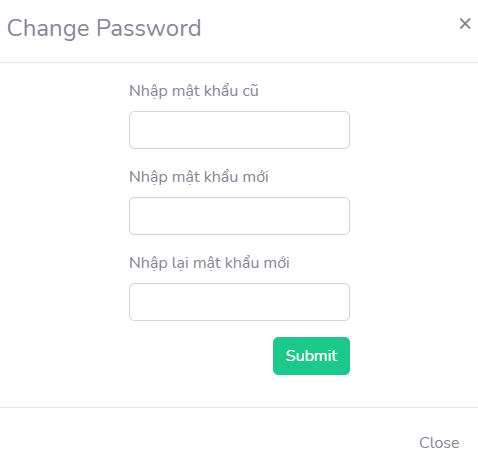
**2.2.1.8. Account management**



**2.2.1.8.1. Update account**

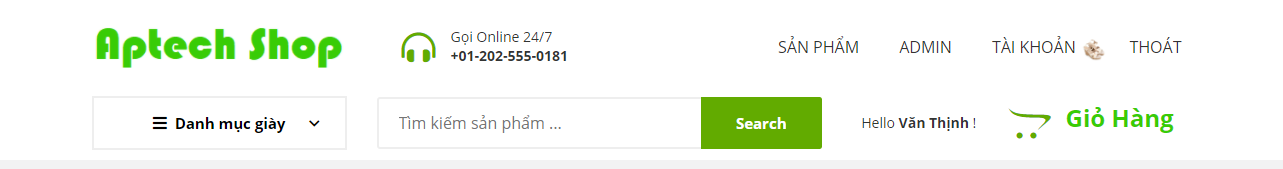


**2.2.1.8.2. Change password**

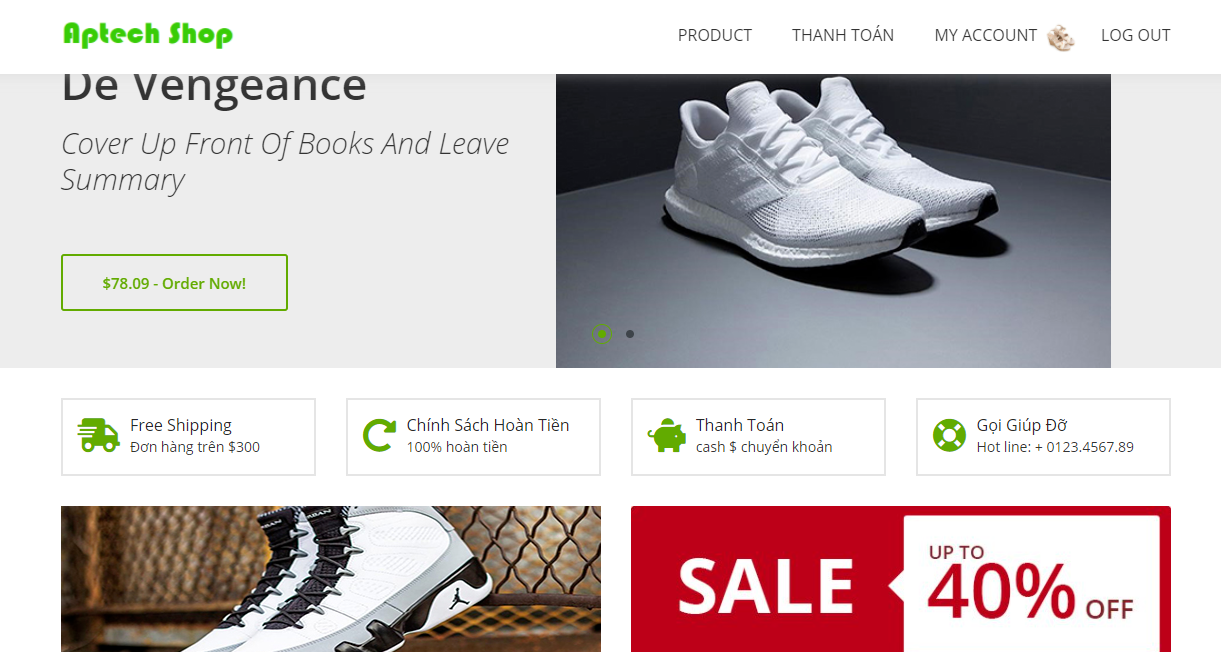


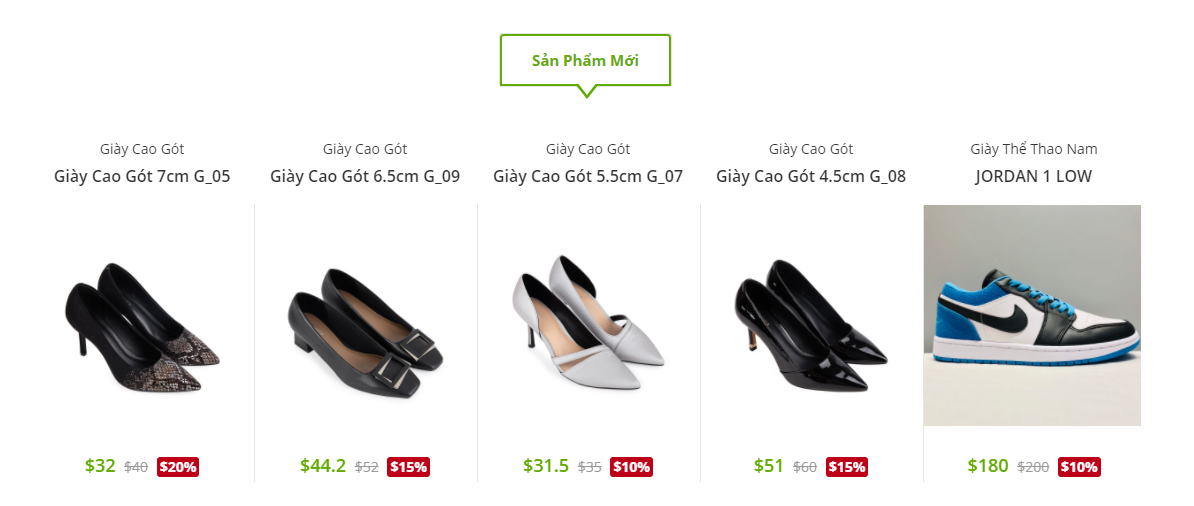
### Design the customer part

**2.2.2.1. General structure (layout)**

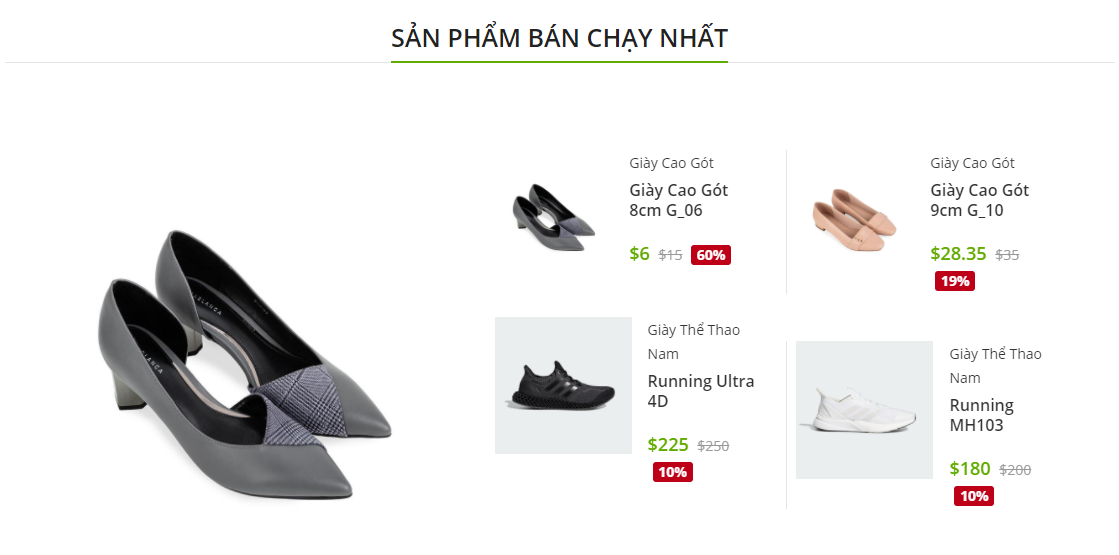


**2.2.2.2. Home**

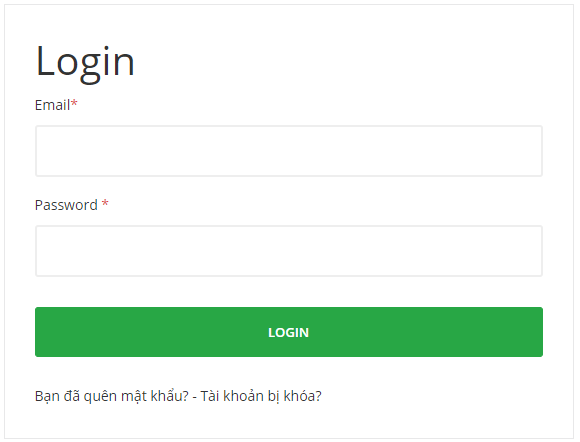




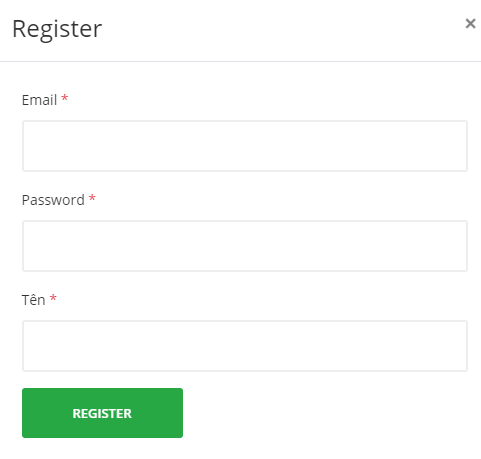




**2.2.2.3. Login**



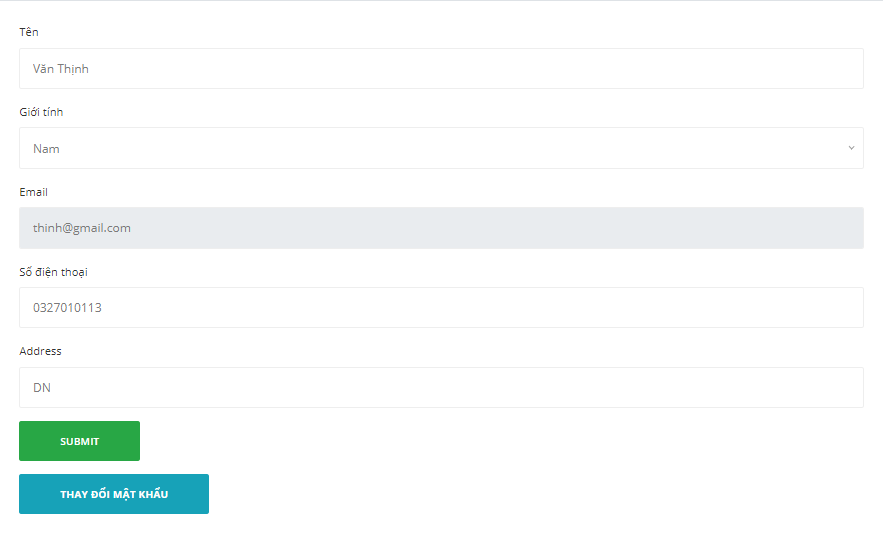
**2.2.2.4. Register**



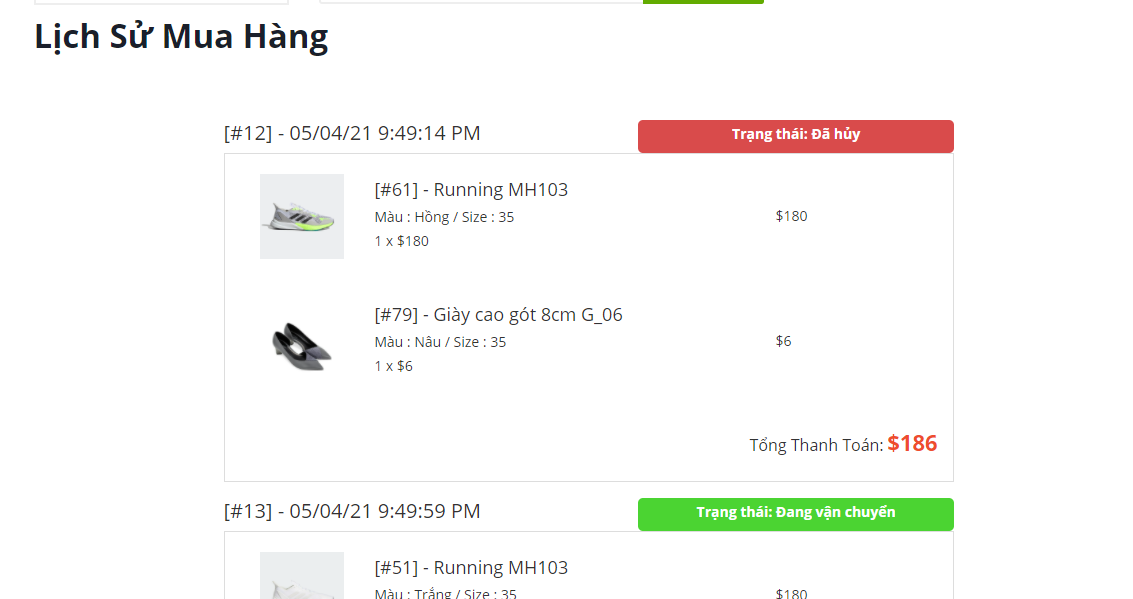
**2.2.2.5. Account management**



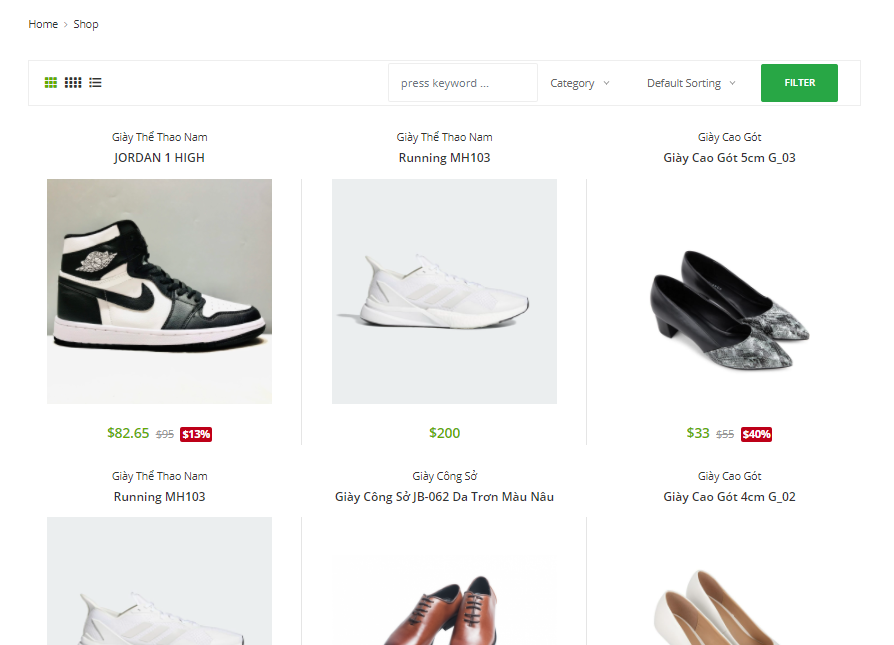
**2.2.2.5.1. Update account**



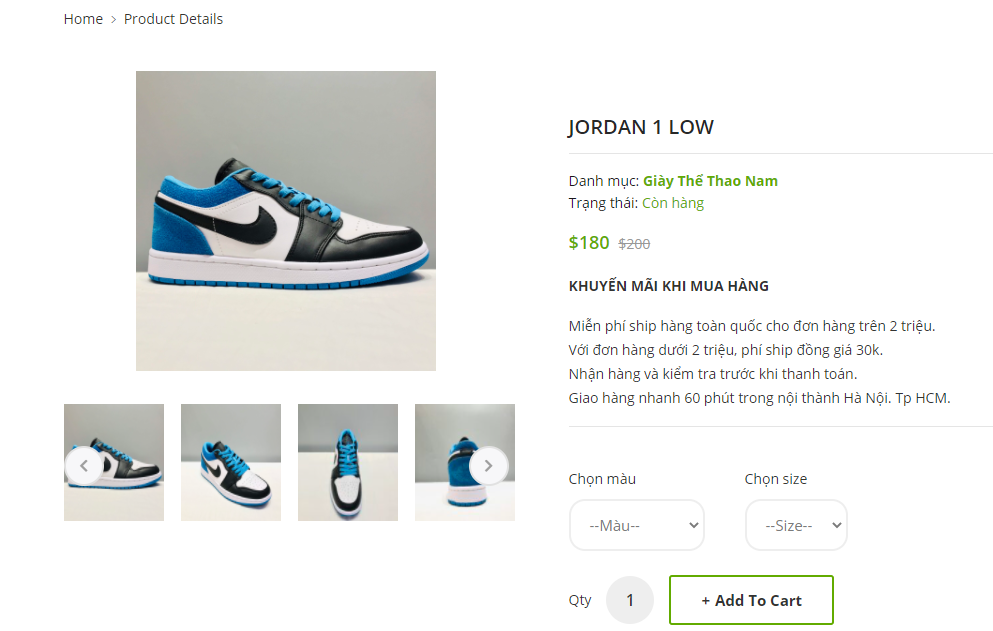
**2.2.2.5.2. Purchase history**



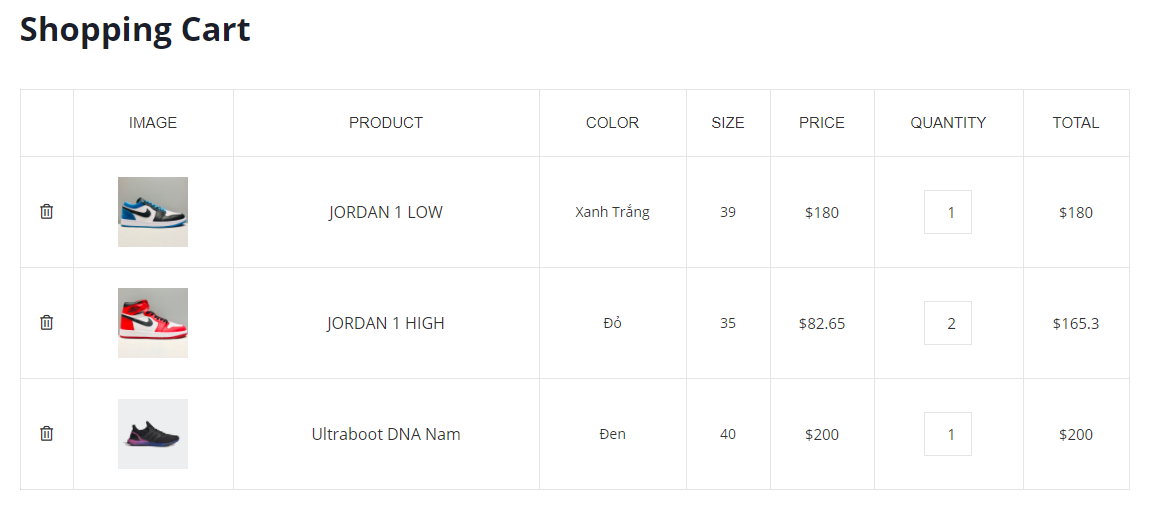
**2.2.2.6. Product**

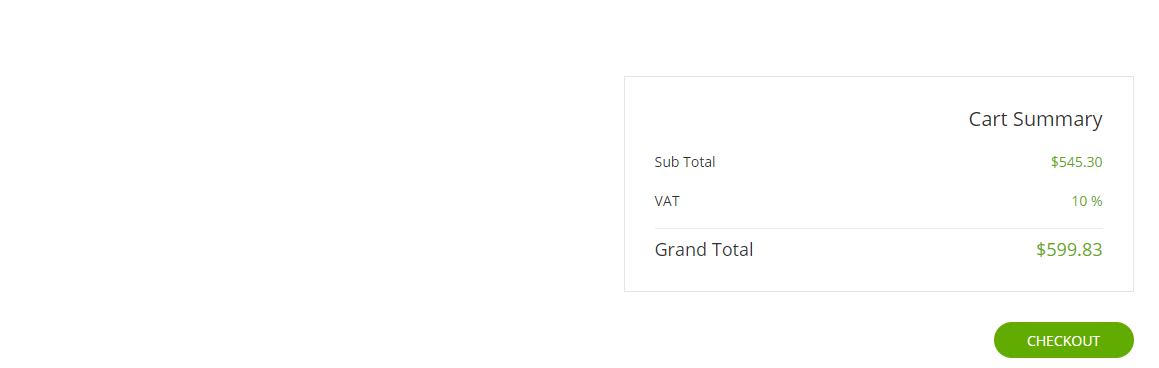


**2.2.2.7. Product detail**

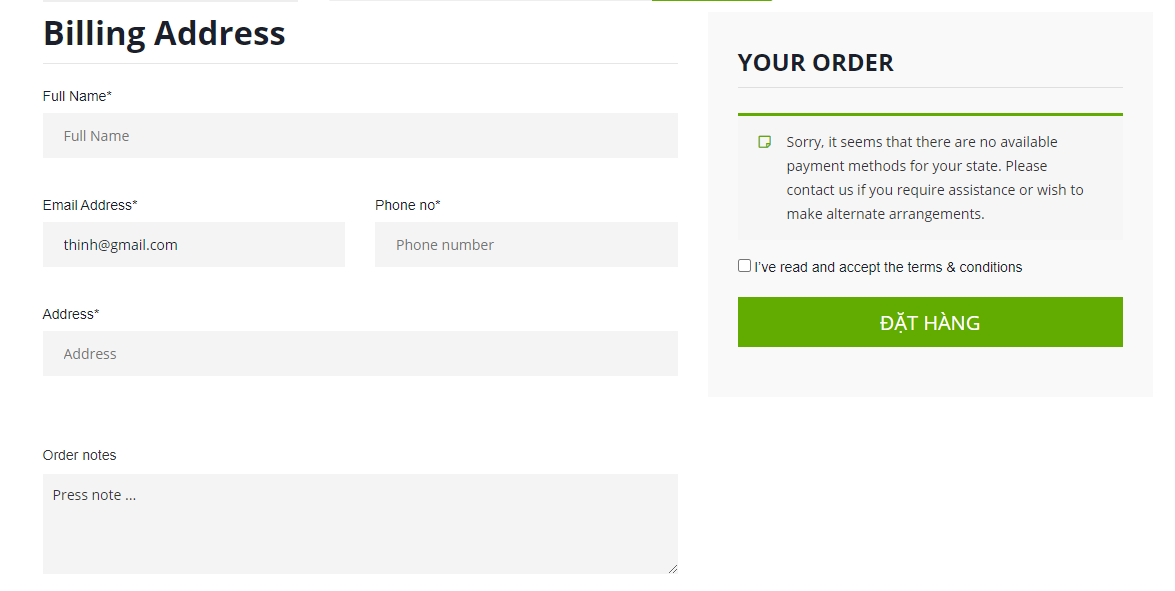


**2.2.2.8. Cart**





**2.2.2.9. Check out**



# Execute coding

## Write code for the administration

### Products Management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | [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));  } |
| Add new | [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");  } |
| Update by Id | [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 });  } |
| Detail by Id | 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);  } |
| Deletel by Id | 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 Table 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 Table UserRole  var account = db.UserRoles.Where(x => x.UserId == userId).FirstOrDefault();  ///Check null  if (account == null)  {  return Json(error);  }    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();  }  } |

### Categories management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | [HttpGet]  public ActionResult Index()  {  return View(db.Categories.OrderBy(x => x.Position).ToList());  } |
| Add new | [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");  } |
| Update by Id | [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");  } |
| Deletel by Id | public ActionResult DeleteCate(int id)  {  Category t = db.Categories.Find(id);  db.Categories.Remove(t);  db.SaveChanges();  return Content("OK");  } |

### Orders management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | 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));  } |
| Add new | [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);  } |
| Update by Id | [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!");  } |
| Detail by Id | [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);  } |
| Deletel by Id | 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");  } |

### Users management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | [HttpGet]  public ActionResult Index()  {  var tbUsers = db.TbUsers.Include(t => t.Status);  return View(tbUsers.ToList());  } |
| Add new | [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");  } |
| Update by Id | [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");  } |
| Detail by Id | 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);  } |
| Deletel by Id | public ActionResult Delete(int id)  {  var user = db.TbUsers.Find(id);  db.TbUsers.Remove(user);  db.SaveChanges();  return Content("OK");  } |

### Authorization management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | [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());  } |
| Add new | [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");  } |
| Deletel by Id | [HttpPost]  public ActionResult Delete(int id)  {  var rs = db.UserRoles.Find(id);  if (rs != null)  {  db.UserRoles.Remove(rs);  db.SaveChanges();  }  return RedirectToAction("Index");  } |

## Write the code for the user section

### Home

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | [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();  } |

### Account management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| 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");  } |
| Profile | [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. Update account**

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| 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. View and update order history**

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | [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");  } |

### Products management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Show all in index | [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);  } |

### Cart management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Binding products to cart | [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);  } |

### Checkout management

|  |  |
| --- | --- |
| **Action** | **Code statement** |
| Check out | [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");  } |