Bộ Giáo Dục Và Đào Tạo

**Trường Đại Học Ngoại Ngữ - Tin Học TP-Hồ Chí Minh**

**---o0o---**



**BÁO CÁO KẾT THÚC MÔN HỌC**

***ĐỀ TÀI:* WEBSITE BÁN ĐIỆN THOẠI**

Giảng viên hướng dẫn:  Ths. Phan Thị Ngọc Hân

Sinh viên thực hiện:

Nguyễn Trần An Khang MSSV: 20DH112028

Lê Thanh Tùng MSSV: 20DH111303

Nguyễn Tiến Phát MSSV:20DH111676

Tp. Hồ Chí Minh - Tháng ...

**LỜI NGÕ**

Trong thời gian học tập và làm báo cáo với đề tài “Xây dựng website điện thoại online” này, em muốn gửi những lời cám ơn và biết ơn chân thành nhất của mình tới tất cả những người đã hỗ trợ, giúp đỡ em về kiến thức và tinh thần trong quá trình thực hiện đồ án.

Trước hết, em xin chân thành cám ơn côThs. Phan Thị Ngọc Hân. Giảng viên Khoa Công Nghệ Thông Tin, Trường Đại học Ngoại Ngữ - Tin Học Thành phố Hồ Chí Minh, người đã trực tiếp hướng dẫn, nhận xét, giúp đỡ em trong suốt quá trình thực hiện đồ án.

Xin chân thành cảm ơn các thầy cô trong Khoa Công Nghệ Thông Tin và các phòng ban nhà trường đã tạo điều kiện tốt nhất cho em cũng như các bạn khác trong suốt thời gian học tập và làm đồ án.

Cuối cùng em xin gửi lời cảm ơn đến gia đình, bạn bè, người thân đã giúp đỡ động viên em rất nhiều trong quá trình học tập và làm đồ án công nghệ phần mềm nâng cao đúng thời gian quy định.

Do thời gian thực hiện có hạn, kiến thức còn nhiều hạn chế nên đồ án thực hiện chắc chắn không tránh khỏi những thiếu sót nhất định.

Em rất mong nhận được ý kiến đóng góp của thầy cô giáo và các bạn để em có thêm kinh nghiệm và tiếp tục hoàn thiện đồ án của mình. Em xin chân thành cảm ơn tất cả mọi người.!

TP. Hồ Chí Minh, ngày… tháng… năm 2023

PHIẾU CHẤM ĐIỂM MÔN THI VẤN ĐÁP

Điểm phần trình bày – Điểm hệ 10



|  |  |  |
| --- | --- | --- |
|  | **CBCT1** | **CBCT2** |
| **Họ tên CBCT** | …………………………………………..  **Chữ ký**: ……………………………....... | …………………………………………..  **Chữ ký**: ……………………………....... |
| **Điểm** | …………………………………………..  **Bằng chữ**: ……………………………... | …………………………………………..  **Bằng chữ:** ……………………………... |
| **Nhận xét** |  |  |

 Điểm quá trình – Điểm hệ 10

**Họ tên CBCT:** ……………………………………Ký tên )

**Điểm tổng kết:**……………………………….(**Bằng chữ** )



**MỤC LỤC**

[DANH MỤC HÌNH 6](#_Toc135734934)

[DANH MỤC BẢNG 7](#_Toc135734935)

[1. XÁC ĐỊNH YÊU CẦU 8](#_Toc135734936)

[1.1. Mô tả hệ thống và yêu cầu chức năng 8](#_Toc135734937)

[1.2. Phân tích Use Case 9](#_Toc135734938)

[1.2.1. Use case tổng quát 9](#_Toc135734939)

[1.2.2. Sơ đồ chi tiết quản lý sản phẩm 10](#_Toc135734940)

[1.2.3. Sơ đồ chi tiết đăng nhập, đăng kí 10](#_Toc135734941)

[1.2.4. Sơ đồ chi tiết quản lý tài khoản 11](#_Toc135734942)

[1.3. Product Backlog 12](#_Toc135734943)

[1.3.1. Thành viên và vai trò 12](#_Toc135734944)

[1.3.2. Product Backlog 12](#_Toc135734945)

[1.3.3. Sprint Backlog 13](#_Toc135734946)

[2. THIẾT KẾ 17](#_Toc135734947)

[2.1. Màn hình 17](#_Toc135734948)

[2.2. Module 23](#_Toc135734949)

[2.2.1. Code BundleConfig 23](#_Toc135734950)

[2.2.2. Filter Config 24](#_Toc135734951)

[2.2.3. IdentityConfig 25](#_Toc135734952)

[2.2.4. RoundConfig 28](#_Toc135734953)

[2.2.5. Startup.Auth 30](#_Toc135734954)

[2.2.6. AccountController 31](#_Toc135734955)

[2.2.7. AdvController 35](#_Toc135734956)

[2.2.8. CategoryController 37](#_Toc135734957)

[2.2.9. HomeController 39](#_Toc135734958)

[2.2.10. NewsController 40](#_Toc135734959)

[2.2.11. OrderController 43](#_Toc135734960)

[2.2.12. PostController 45](#_Toc135734961)

[2.2.13. ProductcategoriController 48](#_Toc135734962)

[2.2.14. ProductImageController 50](#_Toc135734963)

[2.2.15. ProductController 51](#_Toc135734964)

[2.2.16. RoleController 55](#_Toc135734965)

[2.2.17. SettingSystemController 56](#_Toc135734966)

[2.2.18. StatisticalController 60](#_Toc135734967)

[2.2.19. AccountController 62](#_Toc135734968)

[2.2.20. ArticleController 79](#_Toc135734969)

[2.2.21. ContactController 80](#_Toc135734970)

[2.2.22. HomeController 81](#_Toc135734971)

[2.2.23. ManageController 83](#_Toc135734972)

[2.2.24. MenuController 97](#_Toc135734973)

[2.2.25. NewsController 98](#_Toc135734974)

[2.2.26. ProductController 99](#_Toc135734975)

[2.2.27. ShoppingcartController 101](#_Toc135734976)

[3. THỰC HIỆN PHẦN MỀM 107](#_Toc135734977)

[3.1. Kiến trúc phần mềm 107](#_Toc135734978)

[3.2. Phương pháp cài đặt 107](#_Toc135734979)

[3.3. Môi trường cài đặt 107](#_Toc135734980)

[3.4. Một số lưu ý khi cài đặt 107](#_Toc135734981)

[TÀI LIỆU THAM KHẢO 108](#_Toc135734982)

# DANH MỤC HÌNH

[Hình 1. Use case tổng quát 10](#_Toc135735808)

[Hình 2. Usecase quản lý sản phẩm 11](#_Toc135735809)

[Hình 3. Usecase đăng nhập, đăng kí 11](#_Toc135735810)

[Hình 4. Usecase quản lý tài khoản 12](#_Toc135735811)

[Hình 5. Màn hình chính 18](#_Toc135735812)

[Hình 6. Màn hình tìm kiếm sản phẩm 19](#_Toc135735813)

[Hình 7. Màn hình chi tiết sản phẩm 20](#_Toc135735814)

[Hình 8. Màn hình thông tin cửa hàng 21](#_Toc135735815)

[Hình 9. Màn hình quản lý sản phẩm 22](#_Toc135735816)

[Hình 10. Màn hình quản lý danh mục 23](#_Toc135735817)

# DANH MỤC BẢNG

[Bảng 1. Thành viên và vai trò 13](#_Toc135735776)

[Bảng 2. Product backlock 14](#_Toc135735777)

[Bảng 3. Sprint backlock 1 15](#_Toc135735778)

[Bảng 4. Sprint backlock 2 16](#_Toc135735779)

[Bảng 5. Sprint backlock 3 17](#_Toc135735780)

[Bảng 6. Code BundleConfig 25](#_Toc135735781)

[Bảng 7. Filter Config 25](#_Toc135735782)

[Bảng 8. IdentityConfig 28](#_Toc135735783)

[Bảng 9. RoundConfig 31](#_Toc135735784)

[Bảng 10. StartUpAuth 32](#_Toc135735785)

[Bảng 11. AccountController 35](#_Toc135735786)

[Bảng 12. AdvController 38](#_Toc135735787)

[Bảng 13. CategoryController 40](#_Toc135735788)

[Bảng 14. HomeController 41](#_Toc135735789)

[Bảng 15. NewsController 44](#_Toc135735790)

[Bảng 16. OrderController 47](#_Toc135735791)

[Bảng 17. PostController 49](#_Toc135735792)

[Bảng 18. ProductCategoriController 51](#_Toc135735793)

[Bảng 19. ProductImageController 53](#_Toc135735794)

[Bảng 20. ProductController 56](#_Toc135735795)

[Bảng 21. RoleController 58](#_Toc135735796)

[Bảng 22. SettingSystemController 61](#_Toc135735797)

[Bảng 23. StatisticalController 63](#_Toc135735798)

[Bảng 24. AccountController 81](#_Toc135735799)

[Bảng 25. ArticleController 81](#_Toc135735800)

[Bảng 26. ContactController 82](#_Toc135735801)

[Bảng 27. HomeController 84](#_Toc135735802)

[Bảng 28. ManageController 99](#_Toc135735803)

[Bảng 29. MenuController 100](#_Toc135735804)

[Bảng 30. NewsController 101](#_Toc135735805)

[Bảng 31. ProductController 103](#_Toc135735806)

[Bảng 32. ShoppingCartController 109](#_Toc135735807)

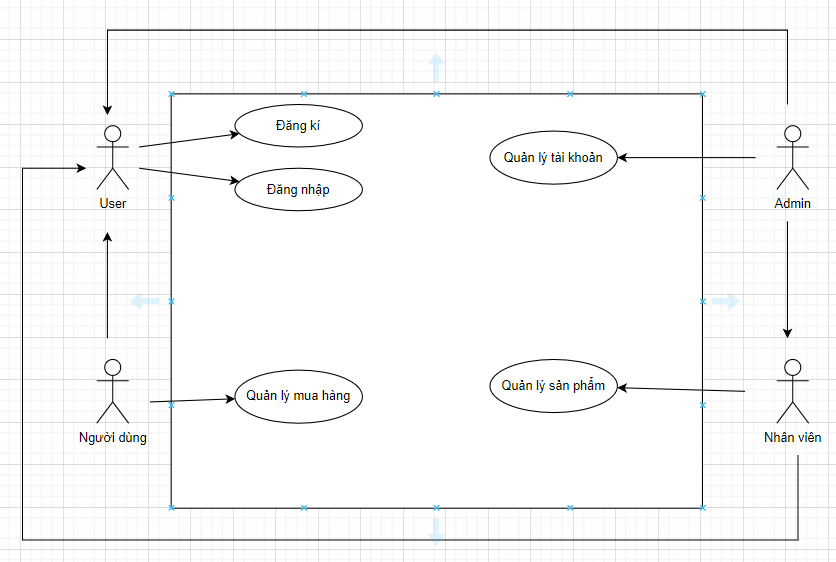
1. **XÁC ĐỊNH YÊU CẦU**

## Mô tả hệ thống và yêu cầu chức năng

* Hệ thống bán điện thoại online bao gồm một trang web
* Hệ thống cho phép người dùng có thể vào trang web để xem thông tin sản phẩm
* Người dùng có thể đăng ký tài khoản để mua sản phẩm hoặc nếu không đăng ký tài khoản thì vẫn cho phép người dùng mua và hỏi người dùng đồng ý sử dụng thông tin giao hàng để tạo tài khoản mới?
* Hệ thống quản lý từng user dựa vào địa chỉ email
* Hệ thống cho phép người dùng xem lại thông tin cá nhân, thay đổi một số thông tin cá nhân cần thiết như: địa chỉ, điện thoại,… nhưng không cho phép thay đổi email.
* Hệ thống có tính năng quản lý đơn hàng phía User:
  + Hệ thống cho phép người dùng chọn một trong các phương thức thanh toán cho đơn hàng như: giao hàng & thanh toán, thanh toán bằng tài khoản ngân hàng,…
  + Hệ thống cho phép người dùng xem lại các đơn hàng đã mua, chi tiết đơn hàng đã mua.
  + Đơn hàng bao gồm các trạng thái sau: đặt hàng thàng công, đang duyệt, đã duyệt, đang giao, đã giao, hoàn tất và đã hủy.
  + Người mua có thể hủy đơn hàng khi đơn hàng vẫn chưa được giao
  + Đối với các đơn hàng đã giao thì trạng thái đơn hàng sẽ là: đã giao
  + Đối với các đơn hàng đã hủy thì trạng thái đơn hàng sẽ là: đã hủy
* Hệ thống có tính năng quản lý đơn hàng phía Admin:
  + Admin được phép xem tất cả các đơn hàng với chọn lựa xem theo khách hàng, hoặc theo ngày, hoặc theo trạng thái của đơn hàng
  + Đơn hàng khi được đặt thì hệ thống sẽ thông báo cho admin biết để admin xử lý các đơn hàng
* Hệ thống có tính năng cho phép người dùng đánh giá (feedback) về các sản phẩm họ đã mua
* Hệ thống cho phép tạo các khảo sát về ý kiến người dùng
* Hệ thống cho phép tạo các đợt giảm giá sản phẩm theo chương trình khuyến mãi hoặc theo sản phẩm, hoặc theo loại khách hàng
* Hệ thống cho phép phân loại khách hàng theo mức chi tiêu của họ làm các loại: khách hàng kim cương, khách hàng vàng, khách hàng bạc và khách hàng đồng

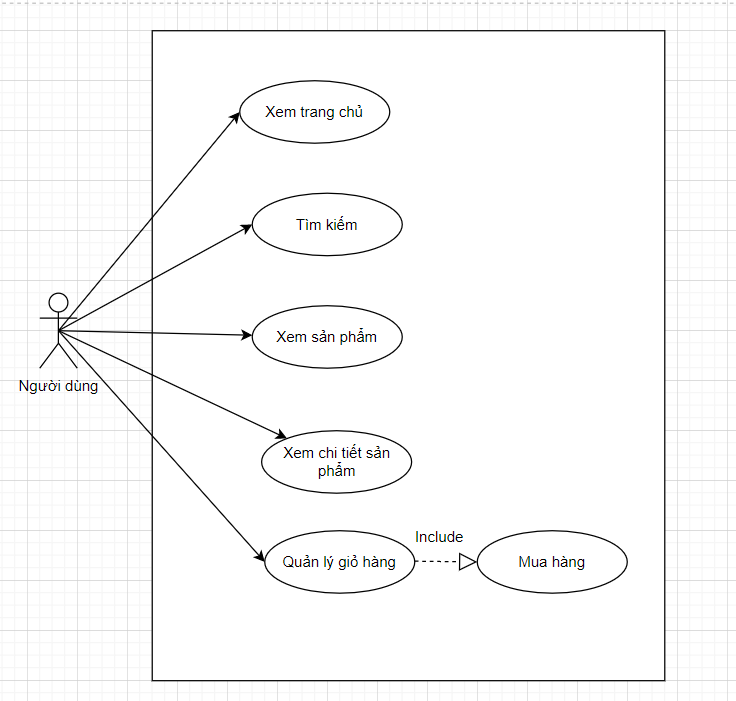
## Phân tích Use Case

### Use case tổng quát



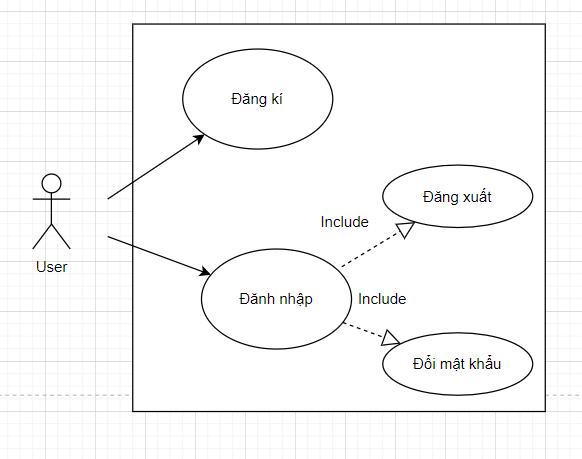
Hình . Use case tổng quát

### Sơ đồ chi tiết quản lý sản phẩm



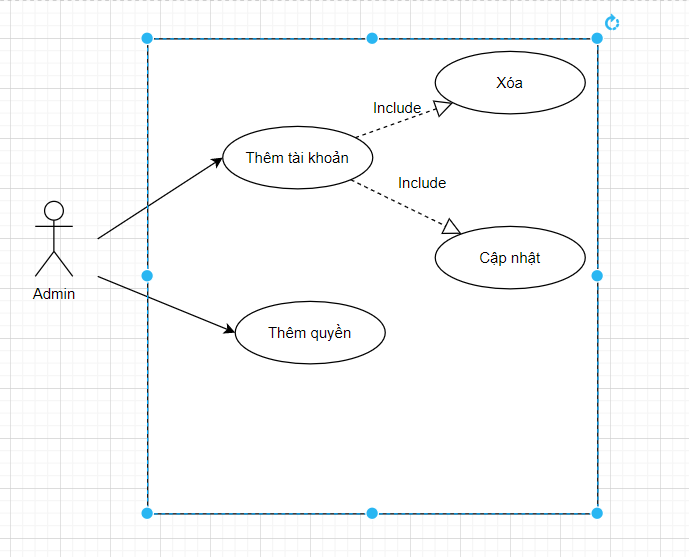
Hình . Usecase quản lý sản phẩm

### Sơ đồ chi tiết đăng nhập, đăng kí



Hình . Usecase đăng nhập, đăng kí

### Sơ đồ chi tiết quản lý tài khoản



Hình . Usecase quản lý tài khoản

## Product Backlog

### Thành viên và vai trò

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TEAM DEVELOPMENT STRUCTURE** | | | | | |
|  |  |  |  |  |  |
| **No.** | **Name** | **Account** | **Roles** | **DateStart** | **Date End** |
| 1 | Nguyễn Trần An Khang | khangnguyen | Dev/ Leader | 09/2/2023 | 18/5/2023 |
| 2 | Lê Thanh Tùng | tungle | Scrum master | 09/2/2023 | 18/5/2023 |
| 3 | Nguyễn Tiến Phát | phatnguyen | Product Owner | 09/2/2023 | 18/5/2023 |

Bảng . Thành viên và vai trò

### Product Backlog

|  |  |  |
| --- | --- | --- |
| Story ID | Story Name | Points |
| **Actor: là khách hàng vãng lai họ muốn** | | |
| 1 | Xem trang chủ | 2 |
| 2 | Xem danh sách sản phẩm | 2 |
| 3 | Xem chi tiết sản phẩm | 2 |
| 4 | Quản lý giỏ hàng | 5 |
| 5 | Đặt hàng | 2 |
| 6 | Đăng kí tài khoản | 2 |
| 7 | Xem thông tin mua hàng | 2 |
| 8 | Thanh toán đơn hàng | 2 |
| 16 | Thanh toán trực tuyến | 5 |
| **Actor: là thành viên họ muốn** | | |
| 12 | Đăng nhập | 2 |
| 13 | Đặt hàng | 2 |
| 14 | Quản lý đơn hàng | 2 |
| 15 | Xem trạng thái đơn hàng | 2 |
| 16 | Thanh toán trực tuyến | 5 |
| 17 | Đăng xuất | 1 |
| 18 | Đánh giá sản phẩm đã mua | 3 |
| 20 | Thay đổi thông tin cá nhân | 2 |
| 21 | Thanh toán đơn hàng | 2 |
| 22 | Xem đơn hàng | 2 |
| 23 | Xem trạng thái đơn hàng | 2 |
| **Actor: là nhân viên xử lý đơn hàng họ muốn** | | |
| 24 | Xem thông tin đơn đặt hàng | 2 |
| 25 | Quản lý đơn hàng | 2 |
| 26 | Quản lý vận chuyển | 2 |
| **Actor: là quản lý họ muốn** | | |
| 27 | Quản lý sản phẩm | 2 |
| 19 | Phân loại khách hàng vàng, bạc, đồng,... | 2 |
| 29 | Quản lý vận chuyễn | 2 |
| 30 | Quản lý giá | 2 |
|  | Quản quản lý đánh giá | 2 |
| **Actor: là quản trị trang web họ muốn** | | |
| 31 | Cấp quyền tài khoản | 3 |
| 32 | Quản lý tài khoản | 3 |

Bảng . Product backlock

### Sprint Backlog

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **E- sCOMMERCE PROJECT - SPRINT BACKLOG** | | | | | | | | | |
| **Story ID** | **Story Name** | **Points** | **Name Of Dev** | **Start Date** | **Endate** | **Act Start Date** | **ActEndate** | **Act Points** | **Notes** |
| **Sprint 1** | | | | | | | | | |
| 1 | Xem trang chủ | 2 | Phat | 09/02/2023 | 12/02/2023 | 25/2/2023 | 1/3/2023 | 2 |  |
| 2 | Đăng ký | 2 | Khang | 09/02/2023 | 12/02/2023 | 2/3/2023 | 3/3/2020 | 2 |  |
| 3 | Đăng nhập | 2 | Khang | 09/02/2023 | 12/02/2023 | 2/3/2023 | 3/3/2020 | 2 |  |
| 4 | Quản lý sản phẩm | 2 | Tung | 13/02/2023 | 16/02/2023 | 4/3/2023 | 7/3/2023 | 2 |  |
| 5 | Quản lý tài khoản | 3 | Tung | 13/02/2023 | 16/02/2023 | 4/3/2023 | 7/3/2023 | 3 |  |
| 6 | Xem danh sách sản phẩm | 2 | Phat | 17/02/2023 | 18/02/223 | 9/3/2023 | 12/2023 | 2 |  |
| 7 | Xem chi tiết sản phẩm | 2 | Phat | 17/02/2023 | 18/02/2023 | 9/3/2023 | 12/3/2023 | 2 |  |
| 8 | Thay đổi thông tin cá nhân | 2 | Tung | 19/02/2023 | 20/02/2023 | 13/3/2023 | 15/3/2023 | 2 |  |
| 9 | Đăng xuất | 1 | Khang | 21/02/2023 | 21/02/2023 | 2/3/2023 | 3/3/2023 | 1 |  |
| 10 | Đặt hàng | 2 | Tung | 22/02/2023 | 24/02/2023 | 15/3/2023 | 20/3/2023 | 2 |  |
|  | | | | | | | | | |

Bảng . Sprint backlock 1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Story ID** | **Story Name** | **Points** | **Name Of Dev** | **Start Date** | **Endate** | **Act Start Date** | | **ActEndate** | **Act Points** | **Notes** |
| **Sprint 2** | | | | | | | | | | |
| 1 | Quản lý đơn hàng | 2 | Tung | 25/02/2023 | 28/02/2023 | | 20/3/2023 | 25/3/2023 | 2 |  |
| 2 | Xem tình trạng vận chuyển | 2 | Phat | 1/03/2023 | 4/03/2023 | | 25/3/2023 | 28/3/2023 | 2 |  |
| 3 | Thanh toán trực tuyến | 5 | Khang | 5/03/2023 | 11/03/2023 | | 29/3/2023 | 5/4/2023 | 5 |  |
| 4 | Đánh giá sản phẩm đã mua | 3 | Khang | 12/03/2023 | 16/3/2023 | | 5/4/2023 | 10/4/2023 | 3 |  |
| 5 | Xem đơn hàng | 2 | Phat | 17/03/2023 | 20/03/2023 | | 11/4/2023 | 13/4/2023 | 2 |  |
| 6 | Quản lý đơn hàng | 2 | Tung | 21/03/2023 | 24/03/2023 | | 14/4/2023 | 17/4/2023 | 2 |  |
| 7 | Quản lý vận chuyển | 2 | Tung | 25/03/2023 | 28/03/2023 | | 18/4/2023 | 19/4/2023 | 2 |  |

Bảng . Sprint backlock 2

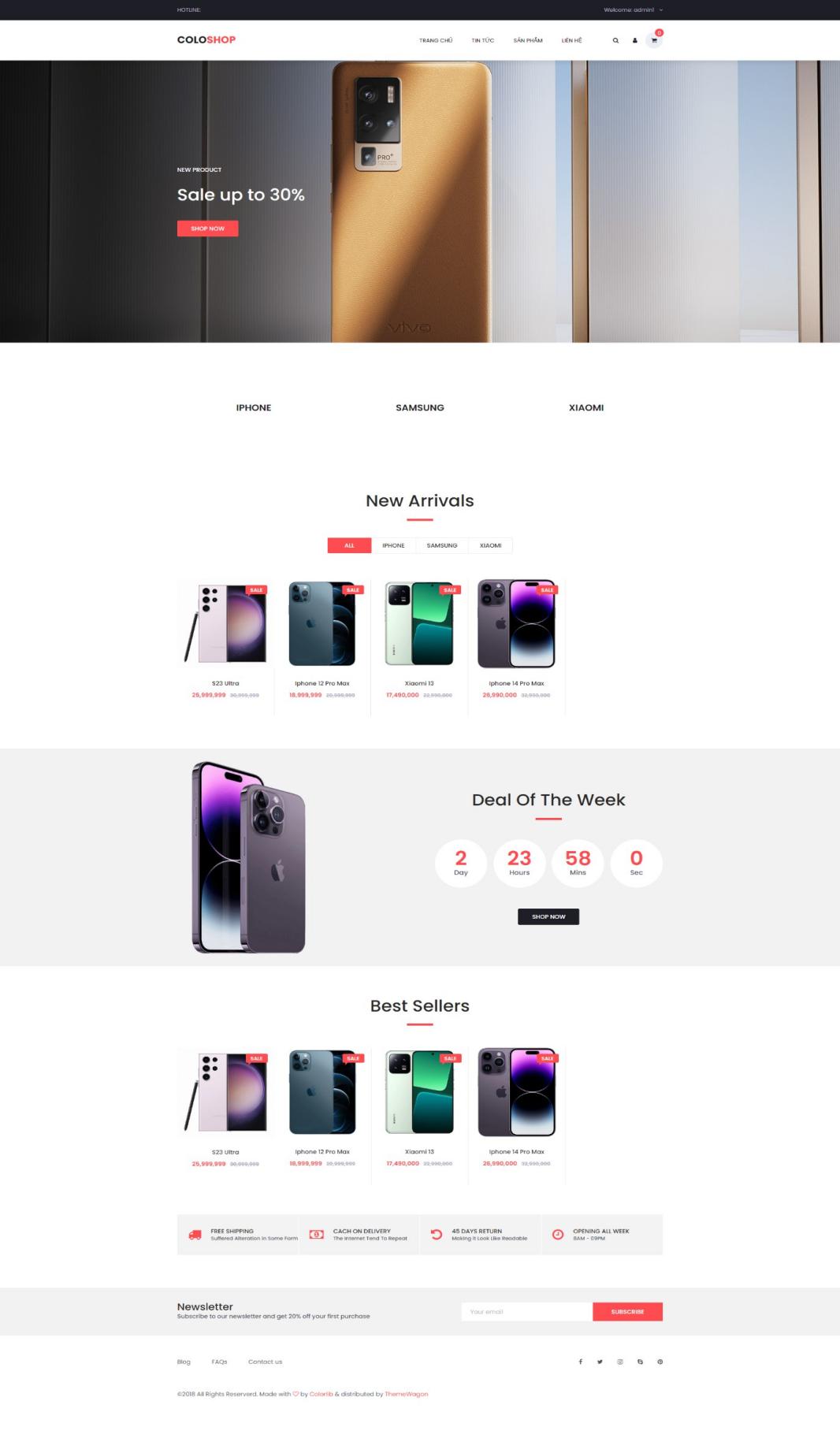
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Story ID** | **Story Name** | **Points** | **Name Of Dev** | **Start Date** | | **Endate** | **Act Start Date** | **ActEndate** | **Act Points** | **Notes** |
| **Sprint 3** | | | | | | | | | | |
| 1 | Phân hạng khách hàng | 2 | Khang | 30/03/2023 | 21/04/2023 | | 22/4/2023 | 25/4/2023 | 2 |  |
| 2 | Cấp quyền tài khoản | 3 | Khang | 3/04/2023 | 22/4/2023 | | 20/04/2023 | 21/04/2024 | 3 |  |

Bảng . Sprint backlock 3

# THIẾT KẾ

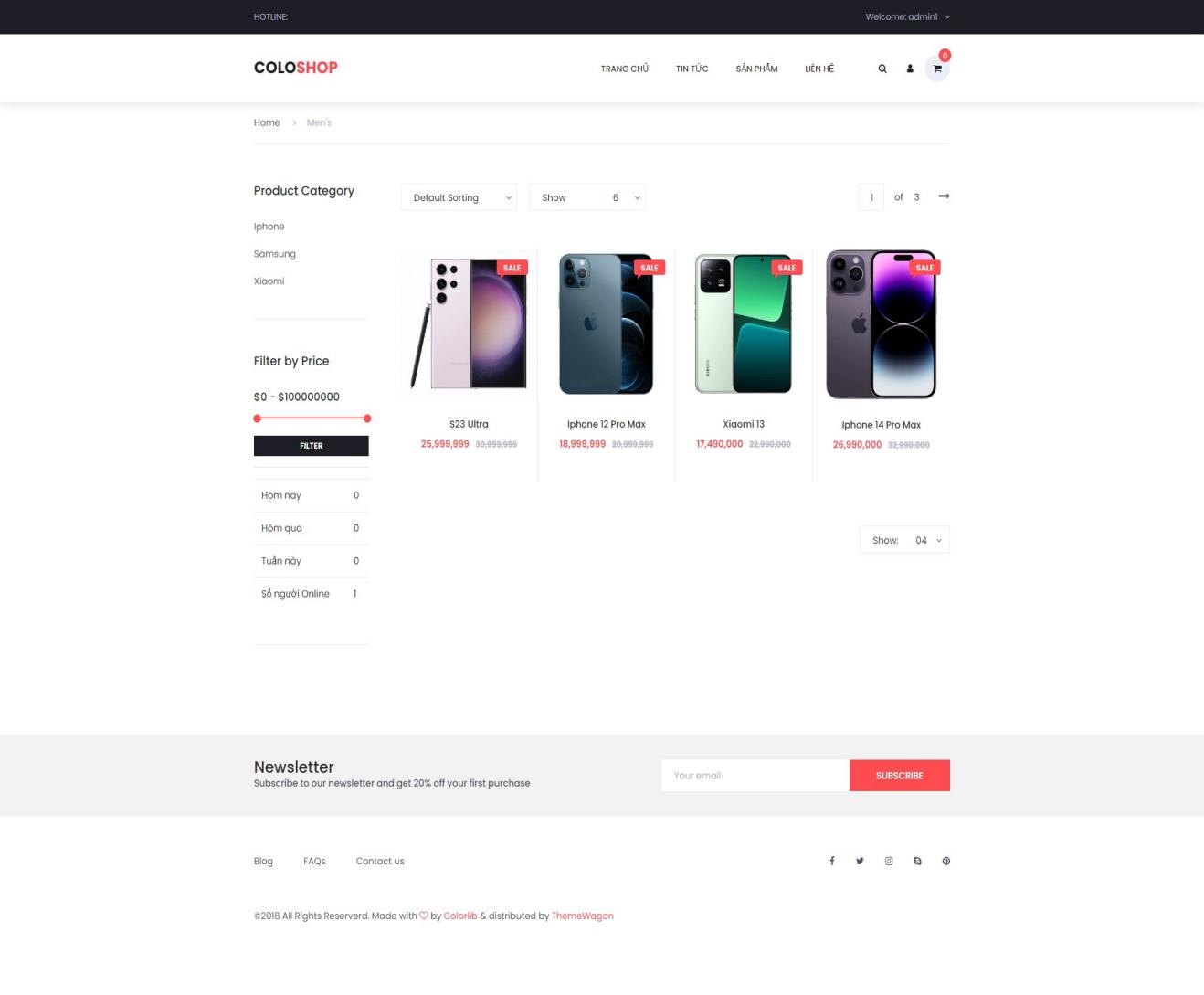
## Màn hình

* **Màn hình chính**



Hình . Màn hình chính

* **Màn hình tìm kiếm sản phẩm**



Hình . Màn hình tìm kiếm sản phẩm

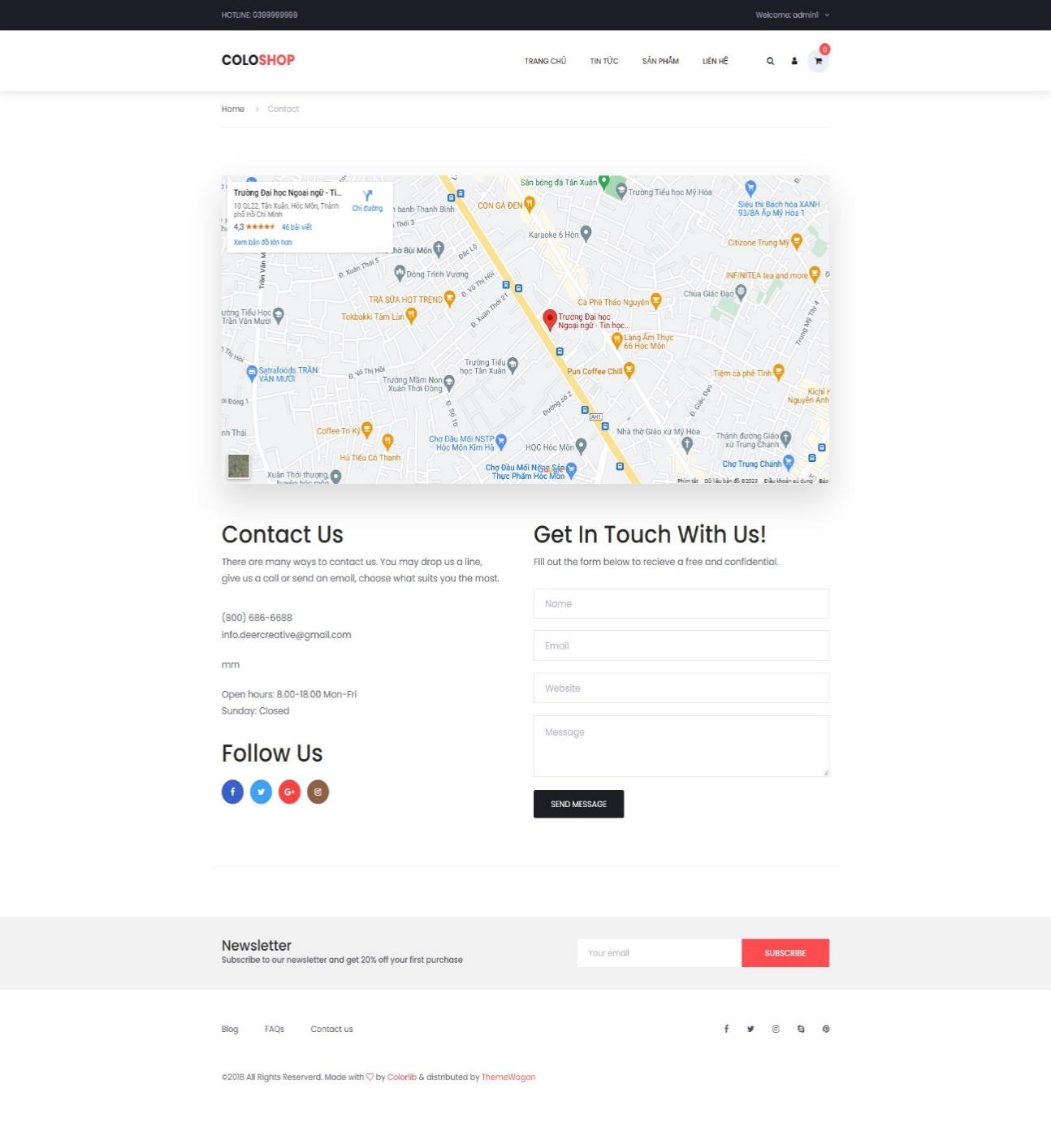
* **Màn hình chi tiết sản phẩm**

Graphical user interface

Description automatically generated

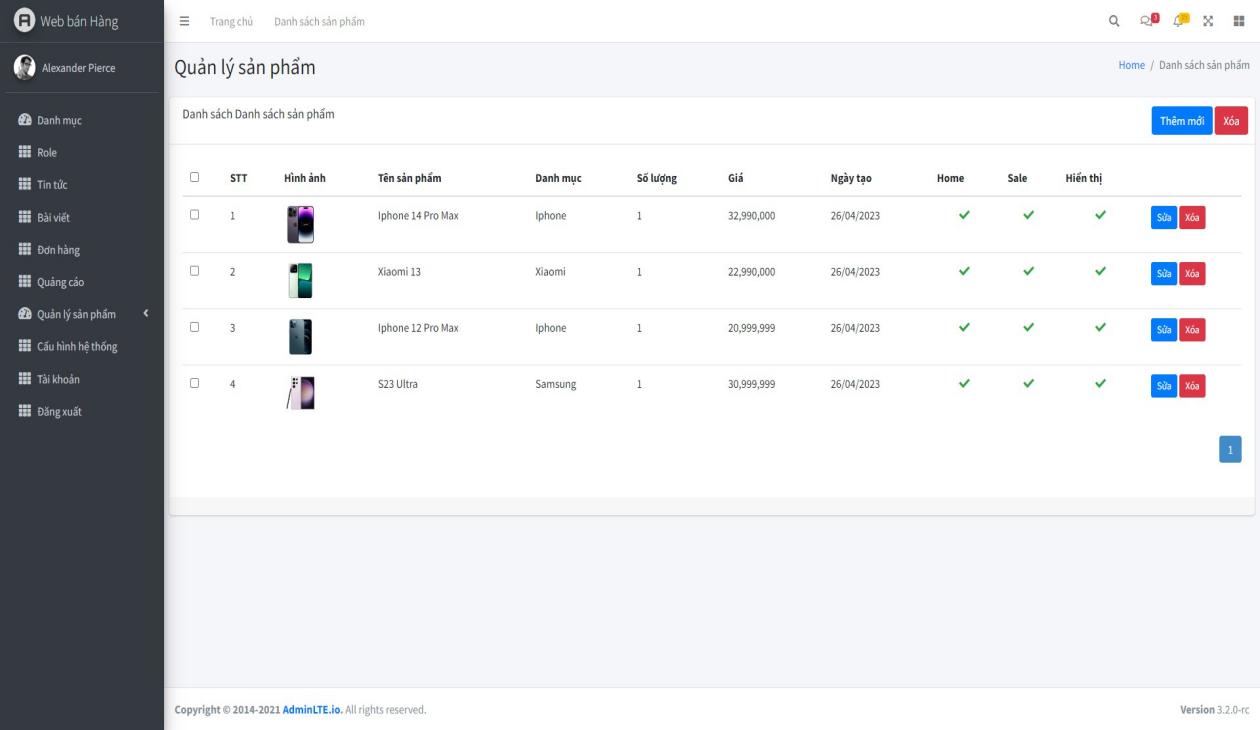
Hình . Màn hình chi tiết sản phẩm

* **­Màn hình thông tin cửa hàng**



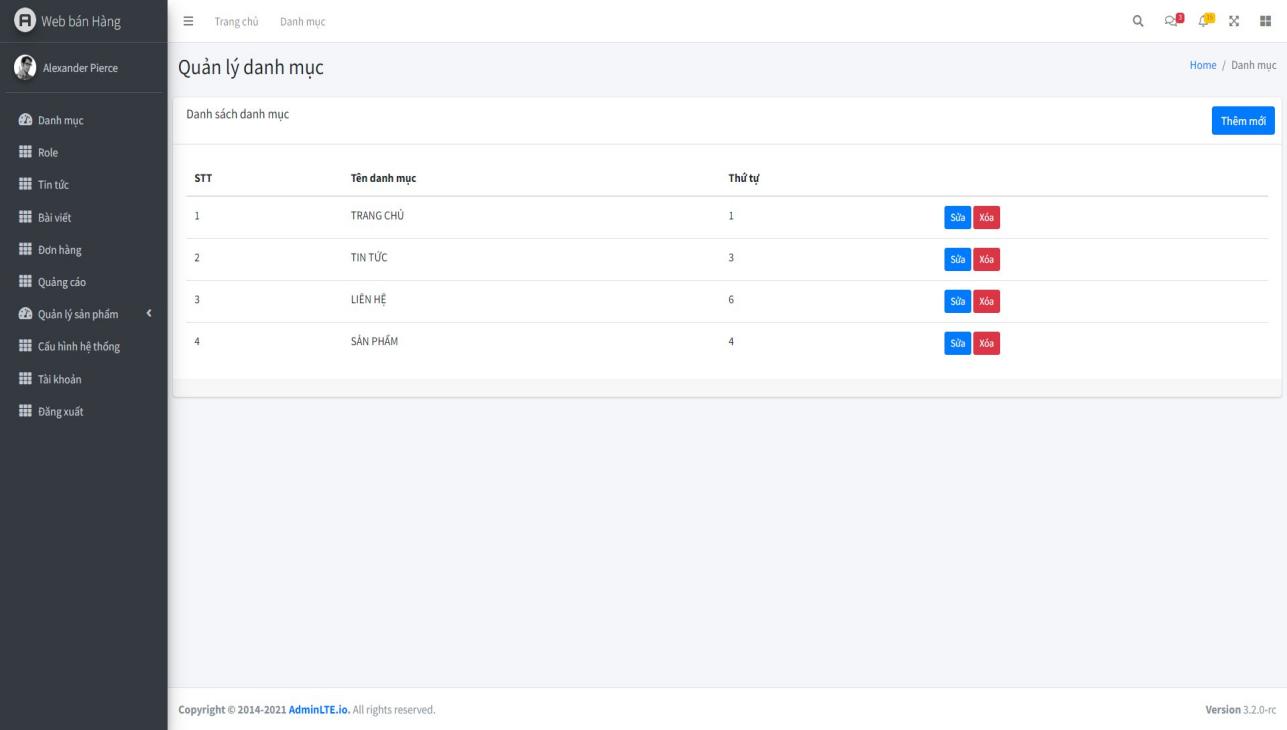
Hình . Màn hình thông tin cửa hàng

* **Màn hình quản lý sản phẩm**



Hình . Màn hình quản lý sản phẩm

* **Màn hình quản lý danh mục**



Hình . Màn hình quản lý danh mục

## Module

### Code BundleConfig

|  |  |
| --- | --- |
|  | using System.Web;  using System.Web.Optimization;  namespace WebBanHangOnline  {  public class BundleConfig  {  public static void RegisterBundles(BundleCollection bundles)  {  bundles.Add(new ScriptBundle("~/bundles/jquery").Include(  "~/Scripts/jquery-{version}.js"));  bundles.Add(new ScriptBundle("~/bundles/jqueryval").Include(  "~/Scripts/jquery.validate\*",  "~/Scripts/jquery.validate.unobtrusive\*"));    bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(  "~/Scripts/modernizr-\*"));  bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include(  "~/Scripts/bootstrap.js"));  bundles.Add(new StyleBundle("~/Content/css").Include(  "~/Content/bootstrap.css",  "~/Content/site.css"));  }  }  } |
| * **bundles.Add(new ScriptBundle("~/bundles/jquery").Include("~/Scripts/jquery-{version}.js"));** tạo ra một gói (bundle) tên là "jquery". Gói này chứa tệp tin JavaScript có tên là "jquery-{version}.js" từ thư mục Scripts. * **bundles.Add(new ScriptBundle("~/bundles/jqueryval").Include("~/Scripts/jquery.validate\*", "~/Scripts/jquery.validate.unobtrusive\*"));** tạo ra một gói tên là "jqueryval". Gói này chứa các tệp tin JavaScript liên quan đến việc xác thực dữ liệu trên trang web. * **Bundles.Add(new ScriptBundle("~/bundles/modernizr").Include("~/Scripts/modernizr-\*"));** tạo ra một gói tên là "modernizr". Gói này chứa tệp tin JavaScript của thư viện Modernizr, được sử dụng để kiểm tra tính năng hỗ trợ của trình duyệt. * **bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include("~/Scripts/bootstrap.js"));** tạo ra một gói tên là "bootstrap". Gói này chứa tệp tin JavaScript của framework Bootstrap, được sử dụng để tạo giao diện đẹp và linh hoạt trên trang web. * **bundles.Add(new StyleBundle("~/Content/css").Include("~/Content/bootstrap.css", "~/Content/site.css"));** tạo ra một gói tên là "css". Gói này chứa các tệp tin CSS, bao gồm tệp tin "bootstrap.css" và "site.css" từ thư mục Content. | |

Bảng . Code BundleConfig

### Filter Config

|  |  |
| --- | --- |
|  | using System.Web;  using System.Web.Mvc;  namespace WebBanHangOnline  {  public class FilterConfig  {  public static void RegisterGlobalFilters(GlobalFilterCollection filters)  {  filters.Add(new HandleErrorAttribute());  }  }  } |
| * **filters.Add(new HandleErrorAttribute());**lệnh này được sử dụng để xử lý các lỗi trong ứng dụng và hiển thị trang lỗi tương ứng khi có lỗi xảy ra. | |

Bảng . Filter Config

### IdentityConfig

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Data.Entity;  using System.Linq;  using System.Security.Claims;  using System.Threading.Tasks;  using System.Web;  using Microsoft.AspNet.Identity;  using Microsoft.AspNet.Identity.EntityFramework;  using Microsoft.AspNet.Identity.Owin;  using Microsoft.Owin;  using Microsoft.Owin.Security;  using WebBanHangOnline.Models;  namespace WebBanHangOnline  {  public class EmailService : IIdentityMessageService  {  public Task SendAsync(IdentityMessage message)  {    return Task.FromResult(0);  }  }  public class SmsService : IIdentityMessageService  {  public Task SendAsync(IdentityMessage message)  {  return Task.FromResult(0);  }  }  public class ApplicationUserManager : UserManager<ApplicationUser>  {  public ApplicationUserManager(IUserStore<ApplicationUser> store)  : base(store)  {  }  public static ApplicationUserManager Create(IdentityFactoryOptions<ApplicationUserManager> options, IOwinContext context)  {  var manager = new ApplicationUserManager(new UserStore<ApplicationUser>(context.Get<ApplicationDbContext>()));  manager.UserValidator = new UserValidator<ApplicationUser>(manager)  {  AllowOnlyAlphanumericUserNames = false,  RequireUniqueEmail = true  };  manager.PasswordValidator = new PasswordValidator  {  RequiredLength = 6,  RequireNonLetterOrDigit = false,  RequireDigit = false,  RequireLowercase = false,  RequireUppercase = false,  };  manager.UserLockoutEnabledByDefault = true;  manager.DefaultAccountLockoutTimeSpan = TimeSpan.FromMinutes(5);  manager.MaxFailedAccessAttemptsBeforeLockout = 5;  manager.RegisterTwoFactorProvider("Phone Code", new PhoneNumberTokenProvider<ApplicationUser>  {  MessageFormat = "Your security code is {0}"  });  manager.RegisterTwoFactorProvider("Email Code", new EmailTokenProvider<ApplicationUser>  {  Subject = "Security Code",  BodyFormat = "Your security code is {0}"  });  manager.EmailService = new EmailService();  manager.SmsService = new SmsService();  var dataProtectionProvider = options.DataProtectionProvider;  if (dataProtectionProvider != null)  {  manager.UserTokenProvider =  new DataProtectorTokenProvider<ApplicationUser>(dataProtectionProvider.Create("ASP.NET Identity"));  }  return manager;  }  }  public class ApplicationSignInManager : SignInManager<ApplicationUser, string>  {  public ApplicationSignInManager(ApplicationUserManager userManager, IAuthenticationManager authenticationManager)  : base(userManager, authenticationManager)  {  }  public override Task<ClaimsIdentity> CreateUserIdentityAsync(ApplicationUser user)  {  return user.GenerateUserIdentityAsync((ApplicationUserManager)UserManager);  }  public static ApplicationSignInManager Create(IdentityFactoryOptions<ApplicationSignInManager> options, IOwinContext context)  {  return new ApplicationSignInManager(context.GetUserManager<ApplicationUserManager>(), context.Authentication);  }  }  } |
| * Các phương thức trong code này giúp xây dựng hệ thống quản lý người dùng, xác thực và bảo mật đáng tin cậy cho ứng dụng web ASP.NET. Chúng cung cấp các chức năng như quản lý người dùng, quản lý đăng nhập của người dùng. | |

Bảng . IdentityConfig

### RoundConfig

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using System.Web.Routing;  namespace WebBanHangOnline  {  public class RouteConfig  {  public static void RegisterRoutes(RouteCollection routes)  {  routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");  routes.MapRoute(  name: "Contact",  url: "lien-he",  defaults: new { controller = "Contact", action = "Index", alias = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "CheckOut",  url: "thanh-toan",  defaults: new { controller = "ShoppingCart", action = "CheckOut", alias = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "ShoppingCart",  url: "gio-hang",  defaults: new { controller = "ShoppingCart", action = "Index", alias = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "CategoryProduct",  url: "danh-muc-san-pham/{alias}-{id}",  defaults: new { controller = "Products", action = "ProductCategory", id = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "BaiViet",  url: "post/{alias}",  defaults: new { controller = "Article", action = "Index", alias = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "detailProduct",  url: "chi-tiet/{alias}-p{id}",  defaults: new { controller = "Products", action = "Detail", alias = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "Products",  url: "san-pham",  defaults: new { controller = "Products", action = "Index", alias = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "DetailNew",  url: "{alias}-n{id}",  defaults: new { controller = "News", action = "Detail", id = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "NewsList",  url: "tin-tuc",  defaults: new { controller = "News", action = "Index", alias = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  routes.MapRoute(  name: "Default",  url: "{controller}/{action}/{id}",  defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional },  namespaces: new[] { "WebBanHangOnline.Controllers" }  );  }  }  } |
| * Code này thể hiện tạo ra các đường liên kết * **Ví dụ**   + Trang có tên là contac và đường dẫn là liên hệ | |

Bảng . RoundConfig

* + 1. **Startup.Auth**

|  |  |
| --- | --- |
|  | using System;  using Microsoft.AspNet.Identity;  using Microsoft.AspNet.Identity.Owin;  using Microsoft.Owin;  using Microsoft.Owin.Security.Cookies;  using Microsoft.Owin.Security.Google;  using Owin;  using WebBanHangOnline.Models;  namespace WebBanHangOnline  {  public partial class Startup  {  public void ConfigureAuth(IAppBuilder app)  {  app.CreatePerOwinContext(ApplicationDbContext.Create);  app.CreatePerOwinContext<ApplicationUserManager>(ApplicationUserManager.Create);  app.CreatePerOwinContext<ApplicationSignInManager>(ApplicationSignInManager.Create);  app.UseCookieAuthentication(new CookieAuthenticationOptions  {  AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,  LoginPath = new PathString("/Account/Login"),  Provider = new CookieAuthenticationProvider  {  OnValidateIdentity = SecurityStampValidator.OnValidateIdentity<ApplicationUserManager, ApplicationUser>(  validateInterval: TimeSpan.FromMinutes(30),  regenerateIdentity: (manager, user) => user.GenerateUserIdentityAsync(manager))  }  });  app.UseExternalSignInCookie(DefaultAuthenticationTypes.ExternalCookie);  app.UseTwoFactorSignInCookie(DefaultAuthenticationTypes.TwoFactorCookie, TimeSpan.FromMinutes(5));    app.UseTwoFactorRememberBrowserCookie(DefaultAuthenticationTypes.TwoFactorRememberBrowserCookie);    }  }  } |
| Code này giúp xác thực và quản lý đăng nhập trong ứng dụng, bao gồm xác thực bằng cookie, đăng nhập bên ngoài và xác thực hai yếu tố. | |

Bảng . StartUpAuth

* + 1. **AccountController**

|  |  |
| --- | --- |
|  | using Microsoft.AspNet.Identity;  using Microsoft.AspNet.Identity.Owin;  using Microsoft.Owin.Security;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Threading.Tasks;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin")]  public class AccountController : Controller  {  private ApplicationSignInManager \_signInManager;  private ApplicationUserManager \_userManager;  private ApplicationDbContext db = new ApplicationDbContext();  public AccountController()  {  }  public AccountController(ApplicationUserManager userManager, ApplicationSignInManager signInManager)  {  UserManager = userManager;  SignInManager = signInManager;  }  public ApplicationSignInManager SignInManager  {  get  {  return \_signInManager ?? HttpContext.GetOwinContext().Get<ApplicationSignInManager>();  }  private set  {  \_signInManager = value;  }  }  public ApplicationUserManager UserManager  {  get  {  return \_userManager ?? HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>();  }  private set  {  \_userManager = value;  }  }  public ActionResult Index()  {  var ítems = db.Users.ToList();  return View(ítems);  }  [AllowAnonymous]  public ActionResult Login(string returnUrl)  {  ViewBag.ReturnUrl = returnUrl;  return View();  }  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)  {  if (!ModelState.IsValid)  {  return View(model);  }    var result = await SignInManager.PasswordSignInAsync(model.UserName, model.Password, model.RememberMe, shouldLockout: false);  switch (result)  {  case SignInStatus.Success:  return RedirectToLocal(returnUrl);  case SignInStatus.LockedOut:  return View("Lockout");  case SignInStatus.RequiresVerification:  return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe = model.RememberMe });  case SignInStatus.Failure:  default:  ModelState.AddModelError("", "Invalid login attempt.");  return View(model);  }  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult LogOff()  {  AuthenticationManager.SignOut(DefaultAuthenticationTypes.ApplicationCookie);  return RedirectToAction("Index", "Home");  }  [AllowAnonymous]  public ActionResult Create()  {  ViewBag.Role = new SelectList(db.Roles.ToList(), "Name", "Name");  return View();  }  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Create(CreateAccountViewModel model)  {  if (ModelState.IsValid)  {  var user = new ApplicationUser  {  UserName = model.UserName,  Email = model.Email,  FullName = model.FullName,  Phone = model.Phone  };  var result = await UserManager.CreateAsync(user, model.Password);  if (result.Succeeded)  {  UserManager.AddToRole(user.Id, model.Role);    return RedirectToAction("Index", "Home");  }  AddErrors(result);  }  ViewBag.Role = new SelectList(db.Roles.ToList(), "Name", "Name");  return View(model);  }  private IAuthenticationManager AuthenticationManager  {  get  {  return HttpContext.GetOwinContext().Authentication;  }  }  private ActionResult RedirectToLocal(string returnUrl)  {  if (Url.IsLocalUrl(returnUrl))  {  return Redirect(returnUrl);  }  return RedirectToAction("Index", "Home");  }  private void AddErrors(IdentityResult result)  {  foreach (var error in result.Errors)  {  ModelState.AddModelError("", error);  }  }  }  } |
| * Code này cho phép người dùng quản lý tài khoản đăng nhập, đăng xuất. * Cho phép admin tạo tài khoản người dùng mới. Admin nhập thông tin cần thiết như tên người dùng, mật khẩu, email, vai trò. Sau khi tạo thành công người dùng có thể thêm vào hệ thống. * Hiển thị danh sách tất cả người dùng trong hệ thốn. Chức năng này được giới bởi admin mới có quyền truy cập. | |

Bảng . AccountController

* + 1. **AdvController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin,Employee")]  public class AdvController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/Posts  public ActionResult Index()  {  var items = db.Posts.ToList();  return View(items);  }  public ActionResult Add()  {  return View();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Add(Adv model)  {  if (ModelState.IsValid)  {  model.CreatedDate = DateTime.Now;  model.ModifiedDate = DateTime.Now;  db.Advs.Add(model);  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  public ActionResult Edit(int id)  {  var item = db.Advs.Find(id);  return View(item);  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Edit(Adv model)  {  if (ModelState.IsValid)  {  model.ModifiedDate = DateTime.Now;  db.Advs.Attach(model);  db.Entry(model).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  [HttpPost]  public ActionResult Delete(int id)  {  var item = db.Advs.Find(id);  if (item != null)  {  db.Advs.Remove(item);  db.SaveChanges();  return Json(new { success = true });  }  return Json(new { success = false });  }    [HttpPost]  public ActionResult DeleteAll(string ids)  {  if (!string.IsNullOrEmpty(ids))  {  var items = ids.Split(',');  if (items != null && items.Any())  {  foreach (var item in items)  {  var obj = db.Advs.Find(Convert.ToInt32(item));  db.Advs.Remove(obj);  db.SaveChanges();  }  }  return Json(new { success = true });  }  return Json(new { success = false });  }  }  } |
| Code này cung cấp các phương thức và chức năng để thao tác với quảng cáo với quyền admin của ứng dụng web | |

Bảng . AdvController

* + 1. **CategoryController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin")]  public class CategoryController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/Category  public ActionResult Index()  {  var items = db.Categories;  return View(items);  }  public ActionResult Add()  {  return View();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Add(Category model)  {  if (ModelState.IsValid)  {  model.CreatedDate = DateTime.Now;  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.Categories.Add(model);  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  public ActionResult Edit(int id)  {  var item = db.Categories.Find(id);  return View(item);  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Edit(Category model)  {  if (ModelState.IsValid)  {  db.Categories.Attach(model);  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.Entry(model).Property(x => x.Title).IsModified = true;  db.Entry(model).Property(x => x.Description).IsModified = true;  db.Entry(model).Property(x => x.Link).IsModified = true;  db.Entry(model).Property(x => x.Alias).IsModified = true;  db.Entry(model).Property(x => x.SeoDescription).IsModified = true;  db.Entry(model).Property(x => x.SeoKeywords).IsModified = true;  db.Entry(model).Property(x => x.SeoTitle).IsModified = true;  db.Entry(model).Property(x => x.Position).IsModified = true;  db.Entry(model).Property(x => x.ModifiedDate).IsModified = true;  db.Entry(model).Property(x => x.Modifiedby).IsModified = true;  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  [HttpPost]  public ActionResult Delete(int id)  {  var item = db.Categories.Find(id);  if (item != null)  {    db.Categories.Remove(item);  db.SaveChanges();  return Json(new { success = true });  }  return Json(new { success = false });  }  }  } |
| * Code này cung cấp các phương thức và chức năng để thao tác với danh mục trong khu vực "Admin" của ứng dụng web. | |

Bảng . CategoryController

* + 1. **HomeController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin,Employee")]  public class HomeController : Controller  {  // GET: Admin/Home  public ActionResult Index()  {  return View();  }  }  } |
| * Code này là một phần của ứng dụng web trong việc quản lý trang chủ (home page) trong khu vực "Admin". Cụ thể, nó chỉ định hành động Index để hiển thị trang chủ cho khu vực "Admin". | |

Bảng . HomeController

* + 1. **NewsController**

|  |  |
| --- | --- |
|  | using PagedList;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin,Employee")]  public class NewsController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/News  public ActionResult Index(string Searchtext, int? page)  {  var pageSize = 10;  if (page == null)  {  page = 1;  }  IEnumerable<News> items = db.News.OrderByDescending(x => x.Id);  if (!string.IsNullOrEmpty(Searchtext))  {  items= items.Where(x=>x.Alias.Contains(Searchtext) || x.Title.Contains(Searchtext));  }  var pageIndex = page.HasValue ? Convert.ToInt32(page) : 1;  items = items.ToPagedList(pageIndex, pageSize);  ViewBag.PageSize = pageSize;  ViewBag.Page = page;  return View(items);  }  public ActionResult Add()  {  return View();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Add(News model)  {  if (ModelState.IsValid)  {  model.CreatedDate = DateTime.Now;  model.CategoryId = 3;  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.News.Attach(model);  db.Entry(model).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  public ActionResult Edit(int id)  {  var item = db.News.Find(id);  return View(item);  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Edit(News model)  {  if (ModelState.IsValid)  {  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.News.Attach(model);  db.Entry(model).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  [HttpPost]  public ActionResult Delete(int id)  {  var item = db.News.Find(id);  if (item != null)  {  db.News.Remove(item);  db.SaveChanges();  return Json(new { success = true });  }  return Json(new { success = false });  }  [HttpPost]  public ActionResult IsActive(int id)  {  var item = db.News.Find(id);  if (item != null)  {  item.IsActive = !item.IsActive;  db.Entry(item).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return Json(new { success = true, isAcive = item.IsActive });  }  return Json(new { success = false });  }  [HttpPost]  public ActionResult DeleteAll(string ids)  {  if (!string.IsNullOrEmpty(ids))  {  var items = ids.Split(',');  if (items != null && items.Any())  {  foreach (var item in items)  {  var obj = db.News.Find(Convert.ToInt32(item));  db.News.Remove(obj);  db.SaveChanges();  }  }  return Json(new { success = true });  }  return Json(new { success = false });  }  }  } |
| * Code này cung cấp các phương thức và chức năng để thao tác với tin tức trong khu vực "Admin" của ứng dụng web. | |

Bảng . NewsController

* + 1. **OrderController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using PagedList;  using System.Globalization;  using System.Data.Entity;  using WebBanHangOnline.Models.ViewModels;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin")]  public class OrderController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/Order  public ActionResult Index(int? page)  {  var items = db.Orders.OrderByDescending(x => x.CreatedDate).ToList();  if (page == null)  {  page = 1;  }  var pageNumber = page ?? 1;  var pageSize = 10;  ViewBag.PageSize = pageSize;  ViewBag.Page = pageNumber;  return View(items.ToPagedList(pageNumber, pageSize));  }    public ActionResult View(int id)  {  var item = db.Orders.Find(id);  return View(item);  }  public ActionResult Partial\_SanPham(int id)  {  var items = db.OrderDetails.Where(x => x.OrderId == id).ToList();  return PartialView(items);  }  [HttpPost]  public ActionResult UpdateTT(int id, int trangthai)  {  var item = db.Orders.Find(id);  if (item != null)  {  db.Orders.Attach(item);  item.TypePayment = trangthai;  db.Entry(item).Property(x => x.TypePayment).IsModified = true;  db.SaveChanges();  return Json(new { message = "Success", Success = true });  }  return Json(new { message = "Unsuccess", Success = false });  }  public void ThongKe(string fromDate, string toDate)  {  var query = from o in db.Orders  join od in db.OrderDetails on o.Id equals od.OrderId  join p in db.Products  on od.ProductId equals p.Id  select new  {  CreatedDate = o.CreatedDate,  Quantity = od.Quantity,  Price = od.Price,  OriginalPrice = p.Price  };  if (!string.IsNullOrEmpty(fromDate))  {  DateTime start = DateTime.ParseExact(fromDate, "dd/MM/yyyy", CultureInfo.GetCultureInfo("vi-VN"));  query = query.Where(x => x.CreatedDate >= start);  }  if (!string.IsNullOrEmpty(toDate))  {  DateTime endDate = DateTime.ParseExact(toDate, "dd/MM/yyyy", CultureInfo.GetCultureInfo("vi-VN"));  query = query.Where(x => x.CreatedDate < endDate);  }  var result = query.GroupBy(x => DbFunctions.TruncateTime(x.CreatedDate)).Select(r => new  {  Date = r.Key.Value,  TotalBuy = r.Sum(x => x.OriginalPrice \* x.Quantity), // tổng giá bán  TotalSell = r.Sum(x => x.Price \* x.Quantity) // tổng giá mua  }).Select(x => new RevenueStatisticViewModel  {  Date = x.Date,  Benefit = x.TotalSell - x.TotalBuy,  Revenues = x.TotalSell  });  }  }  } |
| * Code này cung cấp các phương thức và chức năng để thao tác với đơn hàng trong khu vực "Admin" của ứng dụng web. | |

Bảng . OrderController

* + 1. **PostController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin,Employee")]  public class PostsController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/Posts  public ActionResult Index()  {  var items = db.Posts.ToList();  return View(items);  }  public ActionResult Add()  {  return View();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Add(Posts model)  {  if (ModelState.IsValid)  {  model.CreatedDate = DateTime.Now;  model.CategoryId = 3;  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.Posts.Add(model);  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  public ActionResult Edit(int id)  {  var item = db.Posts.Find(id);  return View(item);  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Edit(Posts model)  {  if (ModelState.IsValid)  {  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.Posts.Attach(model);  db.Entry(model).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  [HttpPost]  public ActionResult Delete(int id)  {  var item = db.Posts.Find(id);  if (item != null)  {  db.Posts.Remove(item);  db.SaveChanges();  return Json(new { success = true });  }  return Json(new { success = false });  }  [HttpPost]  public ActionResult IsActive(int id)  {  var item = db.Posts.Find(id);  if (item != null)  {  item.IsActive = !item.IsActive;  db.Entry(item).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return Json(new { success = true, isAcive = item.IsActive });  }  return Json(new { success = false });  }  [HttpPost]  public ActionResult DeleteAll(string ids)  {  if (!string.IsNullOrEmpty(ids))  {  var items = ids.Split(',');  if (items != null && items.Any())  {  foreach (var item in items)  {  var obj = db.Posts.Find(Convert.ToInt32(item));  db.Posts.Remove(obj);  db.SaveChanges();  }  }  return Json(new { success = true });  }  return Json(new { success = false });  }  }  } |
| Code này cung cấp các phương thức và chức năng để thao tác quản lý bài viếttrong khu vực "Admin" của ứng dụng web. | |

Bảng . PostController

* + 1. **ProductcategoriController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  public class ProductCategoryController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/ProductCategory  public ActionResult Index()  {  var items = db.ProductCategories;  return View(items);  }  public ActionResult Add()  {  return View();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Add(ProductCategory model)  {  if (ModelState.IsValid)  {  model.CreatedDate = DateTime.Now;  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.ProductCategories.Add(model);  db.SaveChanges();  return RedirectToAction("Index");  }  return View();  }  public ActionResult Edit(int id)  {  var item = db.ProductCategories.Find(id);  return View(item);  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Edit(ProductCategory model)  {  if (ModelState.IsValid)  {  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.ProductCategories.Attach(model);  db.Entry(model).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return RedirectToAction("Index");  }  return View();  }  }  } |
| Code này cung cấp các phương thức và chức năng để quản lý danh mục sản phẩm trong khu vực "Admin" của ứng dụng web. | |

Bảng . ProductCategoriController

* + 1. **ProductImageController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  public class ProductImageController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/ProductImage  public ActionResult Index(int id)  {  ViewBag.ProductId = id;  var items = db.ProductImages.Where(x => x.ProductId == id).ToList();  return View(items);  }  [HttpPost]  public ActionResult AddImage(int productId,string url)  {  db.ProductImages.Add(new ProductImage {  ProductId=productId,  Image=url,  IsDefault=false  });  db.SaveChanges();  return Json(new { Success=true});  }  [HttpPost]  public ActionResult Delete(int id)  {  var item = db.ProductImages.Find(id);  db.ProductImages.Remove(item);  db.SaveChanges();  return Json(new { success = true });  }  }  } |
| * Code này cung cấp các phương thức và chức năng để quản lý danh mục hình ảnh trong khu vực "Admin" của ứng dụng web. | |

Bảng . ProductImageController

* + 1. **ProductController**

|  |  |
| --- | --- |
|  | using PagedList;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin,Employee")]  public class ProductsController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/Products  public ActionResult Index(int? page)  {  IEnumerable<Product> items = db.Products.OrderByDescending(x => x.Id);  var pageSize = 10;  if (page == null)  {  page = 1;  }  var pageIndex = page.HasValue ? Convert.ToInt32(page) : 1;  items = items.ToPagedList(pageIndex, pageSize);  ViewBag.PageSize = pageSize;  ViewBag.Page = page;  return View(items);  }  public ActionResult Add()  {  ViewBag.ProductCategory = new SelectList(db.ProductCategories.ToList(), "Id", "Title");  return View();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Add(Product model, List<string> Images, List<int> rDefault)  {  if (ModelState.IsValid)  {  if (Images != null && Images.Count > 0)  {  for (int i = 0; i < Images.Count; i++)  {  if (i + 1 == rDefault[0])  {  model.Image = Images[i];  model.ProductImage.Add(new ProductImage  {  ProductId = model.Id,  Image = Images[i],  IsDefault = true  });  }  else  {  model.ProductImage.Add(new ProductImage  {  ProductId = model.Id,  Image = Images[i],  IsDefault = false  });  }  }  }  model.CreatedDate = DateTime.Now;  model.ModifiedDate = DateTime.Now;  if (string.IsNullOrEmpty(model.SeoTitle))  {  model.SeoTitle = model.Title;  }  if (string.IsNullOrEmpty(model.Alias))  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.Products.Add(model);  db.SaveChanges();  return RedirectToAction("Index");  }  ViewBag.ProductCategory = new SelectList(db.ProductCategories.ToList(), "Id", "Title");  return View(model);  }  public ActionResult Edit(int id)  {  ViewBag.ProductCategory = new SelectList(db.ProductCategories.ToList(), "Id", "Title");  var item = db.Products.Find(id);  return View(item);  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Edit(Product model)  {  if (ModelState.IsValid)  {  model.ModifiedDate = DateTime.Now;  model.Alias = WebBanHangOnline.Models.Common.Filter.FilterChar(model.Title);  db.Products.Attach(model);  db.Entry(model).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return RedirectToAction("Index");  }  return View(model);  }  [HttpPost]  public ActionResult Delete(int id)  {  var item = db.Products.Find(id);  if (item != null)  {  var checkImg = item.ProductImage.Where(x => x.ProductId == item.Id);  if (checkImg != null)  {  foreach(var img in checkImg)  {  db.ProductImages.Remove(img);  db.SaveChanges();  }  }  db.Products.Remove(item);  db.SaveChanges();  return Json(new { success = true });  }  return Json(new { success = false });  }  [HttpPost]  public ActionResult IsActive(int id)  {  var item = db.Products.Find(id);  if (item != null)  {  item.IsActive = !item.IsActive;  db.Entry(item).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return Json(new { success = true, isAcive = item.IsActive });  }  return Json(new { success = false });  }  [HttpPost]  public ActionResult IsHome(int id)  {  var item = db.Products.Find(id);  if (item != null)  {  item.IsHome = !item.IsHome;  db.Entry(item).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return Json(new { success = true, IsHome = item.IsHome });  }  return Json(new { success = false });  }  [HttpPost]  public ActionResult IsSale(int id)  {  var item = db.Products.Find(id);  if (item != null)  {  item.IsSale = !item.IsSale;  db.Entry(item).State = System.Data.Entity.EntityState.Modified;  db.SaveChanges();  return Json(new { success = true, IsSale = item.IsSale });  }  return Json(new { success = false });  } |
| * Code này cung cấp các phương thức và chức năng để quản lý sản phẩm trong khu vực "Admin" của ứng dụng web. | |

Bảng . ProductController

* + 1. **RoleController**

|  |  |
| --- | --- |
|  | using Microsoft.AspNet.Identity;  using Microsoft.AspNet.Identity.EntityFramework;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  [Authorize(Roles = "Admin")]  public class RoleController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/Role  public ActionResult Index()  {  var items = db.Roles.ToList();  return View(items);  }  public ActionResult Create()  {  return View();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Create(IdentityRole model)  {  if (ModelState.IsValid)  {  var roleManager = new RoleManager<IdentityRole>(new RoleStore<IdentityRole>(db));  roleManager.Create(model);  return RedirectToAction("Index");  }  return View(model);  }  public ActionResult Edit(int id)  {  var item = db.Roles.Find(id);  return View(item);  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult Edit(IdentityRole model)  {  if (ModelState.IsValid)  {  var roleManager = new RoleManager<IdentityRole>(new RoleStore<IdentityRole>(db));  roleManager.Update(model);  return RedirectToAction("Index");  }  return View(model);  }  }  } |
| * Code này cung cấp các phương thức và chức năng để quản lý vai trò trong khu vực "Admin" của ứng dụng web. | |

Bảng . RoleController

* + 1. **SettingSystemController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  public class SettingSystemController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/SettingSystem  public ActionResult Index()  {  return View();  }  public ActionResult Partial\_Setting()  {  return PartialView();  }  [HttpPost]  public ActionResult AddSetting(SettingSystemViewModel req)  {  SystemSetting set = null;  var checkTitle = db.SystemSettings.FirstOrDefault(x => x.SettingKey.Contains("SettingTitle"));  if (checkTitle == null)  {  set = new SystemSetting();  set.SettingKey = "SettingTitle";  set.SettingValue = req.SettingTitle;  db.SystemSettings.Add(set);  }  else  {  checkTitle.SettingValue = req.SettingTitle;  db.Entry(checkTitle).State = System.Data.Entity.EntityState.Modified;  }  //logo  var checkLogo = db.SystemSettings.FirstOrDefault(x => x.SettingKey.Contains("SettingLogo"));  if (checkLogo == null)  {  set = new SystemSetting();  set.SettingKey = "SettingLogo";  set.SettingValue = req.SettingLogo;  db.SystemSettings.Add(set);  }  else  {  checkLogo.SettingValue = req.SettingLogo;  db.Entry(checkLogo).State = System.Data.Entity.EntityState.Modified;  }  //Email  var email = db.SystemSettings.FirstOrDefault(x => x.SettingKey.Contains("SettingEmail"));  if (email == null)  {  set = new SystemSetting();  set.SettingKey = "SettingEmail";  set.SettingValue = req.SettingEmail;  db.SystemSettings.Add(set);  }  else  {  email.SettingValue = req.SettingEmail;  db.Entry(email).State = System.Data.Entity.EntityState.Modified;  }  //Hotline  var Hotline = db.SystemSettings.FirstOrDefault(x => x.SettingKey.Contains("SettingHotline"));  if (Hotline == null)  {  set = new SystemSetting();  set.SettingKey = "SettingHotline";  set.SettingValue = req.SettingHotline;  db.SystemSettings.Add(set);  }  else  {  Hotline.SettingValue = req.SettingHotline;  db.Entry(Hotline).State = System.Data.Entity.EntityState.Modified;  }  //TitleSeo  var TitleSeo = db.SystemSettings.FirstOrDefault(x => x.SettingKey.Contains("SettingTitleSeo"));  if (TitleSeo == null)  {  set = new SystemSetting();  set.SettingKey = "SettingTitleSeo";  set.SettingValue = req.SettingTitleSeo;  db.SystemSettings.Add(set);  }  else  {  TitleSeo.SettingValue = req.SettingTitleSeo;  db.Entry(TitleSeo).State = System.Data.Entity.EntityState.Modified;  }  //DessSeo  var DessSeo = db.SystemSettings.FirstOrDefault(x => x.SettingKey.Contains("SettingDesSeo"));  if (DessSeo == null)  {  set = new SystemSetting();  set.SettingKey = "SettingDesSeo";  set.SettingValue = req.SettingDesSeo;  db.SystemSettings.Add(set);  }  else  {  DessSeo.SettingValue = req.SettingDesSeo;  db.Entry(DessSeo).State = System.Data.Entity.EntityState.Modified;  }  //KeySeo  var KeySeo = db.SystemSettings.FirstOrDefault(x => x.SettingKey.Contains("SettingKeySeo"));  if (KeySeo == null)  {  set = new SystemSetting();  set.SettingKey = "SettingKeySeo";  set.SettingValue = req.SettingKeySeo;  db.SystemSettings.Add(set);  }  else  {  KeySeo.SettingValue = req.SettingKeySeo;  db.Entry(KeySeo).State = System.Data.Entity.EntityState.Modified;  }  db.SaveChanges();  return View("Partial\_Setting");  }  }  } |
| * Code này cung cấp các phương thức và chức năng để quản lý cài đặt hệ thống trong khu vực "Admin" của ứng dụng web. | |

Bảng . SettingSystemController

* + 1. **StatisticalController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Data.Entity;  using System.Globalization;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Areas.Admin.Controllers  {  public class StatisticalController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Admin/Statistical  public ActionResult Index()  {  return View();  }  [HttpGet]  public ActionResult GetStatistical(string fromDate, string toDate)  {  var query = from o in db.Orders  join od in db.OrderDetails  on o.Id equals od.OrderId  join p in db.Products  on od.ProductId equals p.Id  select new  {  CreatedDate = o.CreatedDate,  Quantity = od.Quantity,  Price = od.Price,  OriginalPrice = p.OriginalPrice  };  if (!string.IsNullOrEmpty(fromDate))  {  DateTime startDate = DateTime.ParseExact(fromDate, "dd/MM/yyyy", null);  query = query.Where(x => x.CreatedDate >= startDate);  }  if (!string.IsNullOrEmpty(toDate))  {  DateTime endDate = DateTime.ParseExact(toDate, "dd/MM/yyyy", null);  query = query.Where(x => x.CreatedDate < endDate);  }  var result = query.GroupBy(x => DbFunctions.TruncateTime(x.CreatedDate)).Select(x => new  {  Date = x.Key.Value,  TotalBuy = x.Sum(y => y.Quantity \* y.OriginalPrice),  TotalSell = x.Sum(y => y.Quantity \* y.Price),  }).Select(x => new  {  Date = x.Date,  DoanhThu = x.TotalSell,  LoiNhuan = x.TotalSell - x.TotalBuy  });  return Json(new { Data = result }, JsonRequestBehavior.AllowGet);  }  }  } |
| * Code này cung cấp các phương thức và chức năng thống kê dữ liệu trong khu vực "Admin" của ứng dụng web. | |

Bảng . StatisticalController

* + 1. **AccountController**

|  |  |
| --- | --- |
|  | using System;  using System.Globalization;  using System.Linq;  using System.Security.Claims;  using System.Threading.Tasks;  using System.Web;  using System.Web.Mvc;  using Microsoft.AspNet.Identity;  using Microsoft.AspNet.Identity.Owin;  using Microsoft.Owin.Security;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Controllers  {  [Authorize]  public class AccountController : Controller  {  private ApplicationSignInManager \_signInManager;  private ApplicationUserManager \_userManager;  public AccountController()  {  }  public AccountController(ApplicationUserManager userManager, ApplicationSignInManager signInManager )  {  UserManager = userManager;  SignInManager = signInManager;  }  public ApplicationSignInManager SignInManager  {  get  {  return \_signInManager ?? HttpContext.GetOwinContext().Get<ApplicationSignInManager>();  }  private set  {  \_signInManager = value;  }  }  public ApplicationUserManager UserManager  {  get  {  return \_userManager ?? HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>();  }  private set  {  \_userManager = value;  }  }  //  // GET: /Account/Login  [AllowAnonymous]  public ActionResult Login(string returnUrl)  {  ViewBag.ReturnUrl = returnUrl;  return View();  }  //  // POST: /Account/Login  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)  {  if (!ModelState.IsValid)  {  return View(model);  }  // This doesn't count login failures towards account lockout  // To enable password failures to trigger account lockout, change to shouldLockout: true  var result = await SignInManager.PasswordSignInAsync(model.UserName, model.Password, model.RememberMe, shouldLockout: false);  switch (result)  {  case SignInStatus.Success:  return RedirectToLocal(returnUrl);  case SignInStatus.LockedOut:  return View("Lockout");  case SignInStatus.RequiresVerification:  return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe = model.RememberMe });  case SignInStatus.Failure:  default:  ModelState.AddModelError("", "Invalid login attempt.");  return View(model);  }  }  //  // GET: /Account/VerifyCode  [AllowAnonymous]  public async Task<ActionResult> VerifyCode(string provider, string returnUrl, bool rememberMe)  {  // Require that the user has already logged in via username/password or external login  if (!await SignInManager.HasBeenVerifiedAsync())  {  return View("Error");  }  return View(new VerifyCodeViewModel { Provider = provider, ReturnUrl = returnUrl, RememberMe = rememberMe });  }  //  // POST: /Account/VerifyCode  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> VerifyCode(VerifyCodeViewModel model)  {  if (!ModelState.IsValid)  {  return View(model);  }  // The following code protects for brute force attacks against the two factor codes.  // If a user enters incorrect codes for a specified amount of time then the user account  // will be locked out for a specified amount of time.  // You can configure the account lockout settings in IdentityConfig  var result = await SignInManager.TwoFactorSignInAsync(model.Provider, model.Code, isPersistent: model.RememberMe, rememberBrowser: model.RememberBrowser);  switch (result)  {  case SignInStatus.Success:  return RedirectToLocal(model.ReturnUrl);  case SignInStatus.LockedOut:  return View("Lockout");  case SignInStatus.Failure:  default:  ModelState.AddModelError("", "Invalid code.");  return View(model);  }  }  //  // GET: /Account/Register  [AllowAnonymous]  public ActionResult Register()  {  return View();  }  //  // POST: /Account/Register  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> Register(RegisterViewModel model)  {  if (ModelState.IsValid)  {  var user = new ApplicationUser { UserName = model.Email, Email = model.Email };  var result = await UserManager.CreateAsync(user, model.Password);  if (result.Succeeded)  {  await SignInManager.SignInAsync(user, isPersistent:false, rememberBrowser:false);    // For more information on how to enable account confirmation and password reset please visit https://go.microsoft.com/fwlink/?LinkID=320771  // Send an email with this link  // string code = await UserManager.GenerateEmailConfirmationTokenAsync(user.Id);  // var callbackUrl = Url.Action("ConfirmEmail", "Account", new { userId = user.Id, code = code }, protocol: Request.Url.Scheme);  // await UserManager.SendEmailAsync(user.Id, "Confirm your account", "Please confirm your account by clicking <a href=\"" + callbackUrl + "\">here</a>");  return RedirectToAction("Index", "Home");  }  AddErrors(result);  }  // If we got this far, something failed, redisplay form  return View(model);  }  //  // GET: /Account/ConfirmEmail  [AllowAnonymous]  public async Task<ActionResult> ConfirmEmail(string userId, string code)  {  if (userId == null || code == null)  {  return View("Error");  }  var result = await UserManager.ConfirmEmailAsync(userId, code);  return View(result.Succeeded ? "ConfirmEmail" : "Error");  }  //  // GET: /Account/ForgotPassword  [AllowAnonymous]  public ActionResult ForgotPassword()  {  return View();  }  //  // POST: /Account/ForgotPassword  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> ForgotPassword(ForgotPasswordViewModel model)  {  if (ModelState.IsValid)  {  var user = await UserManager.FindByNameAsync(model.Email);  if (user == null || !(await UserManager.IsEmailConfirmedAsync(user.Id)))  {  // Don't reveal that the user does not exist or is not confirmed  return View("ForgotPasswordConfirmation");  }  // For more information on how to enable account confirmation and password reset please visit https://go.microsoft.com/fwlink/?LinkID=320771  // Send an email with this link  // string code = await UserManager.GeneratePasswordResetTokenAsync(user.Id);  // var callbackUrl = Url.Action("ResetPassword", "Account", new { userId = user.Id, code = code }, protocol: Request.Url.Scheme);  // await UserManager.SendEmailAsync(user.Id, "Reset Password", "Please reset your password by clicking <a href=\"" + callbackUrl + "\">here</a>");  // return RedirectToAction("ForgotPasswordConfirmation", "Account");  }  // If we got this far, something failed, redisplay form  return View(model);  }  //  // GET: /Account/ForgotPasswordConfirmation  [AllowAnonymous]  public ActionResult ForgotPasswordConfirmation()  {  return View();  }  //  // GET: /Account/ResetPassword  [AllowAnonymous]  public ActionResult ResetPassword(string code)  {  return code == null ? View("Error") : View();  }  //  // POST: /Account/ResetPassword  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> ResetPassword(ResetPasswordViewModel model)  {  if (!ModelState.IsValid)  {  return View(model);  }  var user = await UserManager.FindByNameAsync(model.Email);  if (user == null)  {  // Don't reveal that the user does not exist  return RedirectToAction("ResetPasswordConfirmation", "Account");  }  var result = await UserManager.ResetPasswordAsync(user.Id, model.Code, model.Password);  if (result.Succeeded)  {  return RedirectToAction("ResetPasswordConfirmation", "Account");  }  AddErrors(result);  return View();  }  //  // GET: /Account/ResetPasswordConfirmation  [AllowAnonymous]  public ActionResult ResetPasswordConfirmation()  {  return View();  }  //  // POST: /Account/ExternalLogin  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public ActionResult ExternalLogin(string provider, string returnUrl)  {  // Request a redirect to the external login provider  return new ChallengeResult(provider, Url.Action("ExternalLoginCallback", "Account", new { ReturnUrl = returnUrl }));  }  //  // GET: /Account/SendCode  [AllowAnonymous]  public async Task<ActionResult> SendCode(string returnUrl, bool rememberMe)  {  var userId = await SignInManager.GetVerifiedUserIdAsync();  if (userId == null)  {  return View("Error");  }  var userFactors = await UserManager.GetValidTwoFactorProvidersAsync(userId);  var factorOptions = userFactors.Select(purpose => new SelectListItem { Text = purpose, Value = purpose }).ToList();  return View(new SendCodeViewModel { Providers = factorOptions, ReturnUrl = returnUrl, RememberMe = rememberMe });  }  //  // POST: /Account/SendCode  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> SendCode(SendCodeViewModel model)  {  if (!ModelState.IsValid)  {  return View();  }  // Generate the token and send it  if (!await SignInManager.SendTwoFactorCodeAsync(model.SelectedProvider))  {  return View("Error");  }  return RedirectToAction("VerifyCode", new { Provider = model.SelectedProvider, ReturnUrl = model.ReturnUrl, RememberMe = model.RememberMe });  }  //  // GET: /Account/ExternalLoginCallback  [AllowAnonymous]  public async Task<ActionResult> ExternalLoginCallback(string returnUrl)  {  var loginInfo = await AuthenticationManager.GetExternalLoginInfoAsync();  if (loginInfo == null)  {  return RedirectToAction("Login");  }  // Sign in the user with this external login provider if the user already has a login  var result = await SignInManager.ExternalSignInAsync(loginInfo, isPersistent: false);  switch (result)  {  case SignInStatus.Success:  return RedirectToLocal(returnUrl);  case SignInStatus.LockedOut:  return View("Lockout");  case SignInStatus.RequiresVerification:  return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe = false });  case SignInStatus.Failure:  default:  // If the user does not have an account, then prompt the user to create an account  ViewBag.ReturnUrl = returnUrl;  ViewBag.LoginProvider = loginInfo.Login.LoginProvider;  return View("ExternalLoginConfirmation", new ExternalLoginConfirmationViewModel { Email = loginInfo.Email });  }  }  //  // POST: /Account/ExternalLoginConfirmation  [HttpPost]  [AllowAnonymous]  [ValidateAntiForgeryToken]  public async Task<ActionResult> ExternalLoginConfirmation(ExternalLoginConfirmationViewModel model, string returnUrl)  {  if (User.Identity.IsAuthenticated)  {  return RedirectToAction("Index", "Manage");  }  if (ModelState.IsValid)  {  // Get the information about the user from the external login provider  var info = await AuthenticationManager.GetExternalLoginInfoAsync();  if (info == null)  {  return View("ExternalLoginFailure");  }  var user = new ApplicationUser { UserName = model.Email, Email = model.Email };  var result = await UserManager.CreateAsync(user);  if (result.Succeeded)  {  result = await UserManager.AddLoginAsync(user.Id, info.Login);  if (result.Succeeded)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  return RedirectToLocal(returnUrl);  }  }  AddErrors(result);  }  ViewBag.ReturnUrl = returnUrl;  return View(model);  }  //  // POST: /Account/LogOff  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult LogOff()  {  AuthenticationManager.SignOut(DefaultAuthenticationTypes.ApplicationCookie);  return RedirectToAction("Index", "Home");  }  //  // GET: /Account/ExternalLoginFailure  [AllowAnonymous]  public ActionResult ExternalLoginFailure()  {  return View();  }  protected override void Dispose(bool disposing)  {  if (disposing)  {  if (\_userManager != null)  {  \_userManager.Dispose();  \_userManager = null;  }  if (\_signInManager != null)  {  \_signInManager.Dispose();  \_signInManager = null;  }  }  base.Dispose(disposing);  }  #region Helpers  // Used for XSRF protection when adding external logins  private const string XsrfKey = "XsrfId";  private IAuthenticationManager AuthenticationManager  {  get  {  return HttpContext.GetOwinContext().Authentication;  }  }  private void AddErrors(IdentityResult result)  {  foreach (var error in result.Errors)  {  ModelState.AddModelError("", error);  }  }  private ActionResult RedirectToLocal(string returnUrl)  {  if (Url.IsLocalUrl(returnUrl))  {  return Redirect(returnUrl);  }  return RedirectToAction("Index", "Home");  }  internal class ChallengeResult : HttpUnauthorizedResult  {  public ChallengeResult(string provider, string redirectUri)  : this(provider, redirectUri, null)  {  }  public ChallengeResult(string provider, string redirectUri, string userId)  {  LoginProvider = provider;  RedirectUri = redirectUri;  UserId = userId;  }  public string LoginProvider { get; set; }  public string RedirectUri { get; set; }  public string UserId { get; set; }  public override void ExecuteResult(ControllerContext context)  {  var properties = new AuthenticationProperties { RedirectUri = RedirectUri };  if (UserId != null)  {  properties.Dictionary[XsrfKey] = UserId;  }  context.HttpContext.GetOwinContext().Authentication.Challenge(properties, LoginProvider);  }  }  #endregion  }  } |
| * Code này cho phép quyền đăng nhập đăng kí tài khoản, đăng xuất của khách hàng. | |

Bảng . AccountController

* + 1. **ArticleController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Controllers  {  public class ArticleController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Article  public ActionResult Index(string alias)  {  var item = db.Posts.FirstOrDefault(x => x.Alias == alias);  return View(item);  }  }  } |
| * Code này xử lý có nhiệm vụ xử lý yêu cầu liên quan đến bài viết trên trang web. | |

Bảng . ArticleController

* + 1. **ContactController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Controllers  {  public class ContactController : Controller  {    // GET: Contact  public ActionResult Index(string id)  {  return View();  }  }  } |
| Code này xử lý có nhiệm vụ xử lý yêu cầu liên quan đến liên hệ trên trang web. | |

Bảng . ContactController

### HomeController

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Controllers  {  public class HomeController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  public ActionResult Index()  {    //WebBanHangOnline.Common.Common.SendMail("ABC", "AAAA", "AAAA", "ngohoang29@gmail.com");  return View();  }  public ActionResult Partial\_Subcrice()  {  return PartialView();  }  [HttpPost]  public ActionResult Subscribe(Subscribe req)  {  if (ModelState.IsValid)  {  db.Subscribes.Add(new Subscribe { Email = req.Email, CreatedDate = DateTime.Now });  db.SaveChanges();  return Json(new {Success=true });  }  return View("Partial\_Subcrice", req);  }  public ActionResult About()  {  ViewBag.Message = "Your application description page.";  return View();  }  public ActionResult Refresh()  {  var item = new ThongKeModel();  ViewBag.Visitors\_online = HttpContext.Application["visitors\_online"];  var hn = HttpContext.Application["HomNay"];  item.HomNay = HttpContext.Application["HomNay"].ToString();  item.HomQua = HttpContext.Application["HomQua"].ToString();  item.TuanNay = HttpContext.Application["TuanNay"].ToString();  item.TuanTruoc = HttpContext.Application["TuanTruoc"].ToString();  item.ThangNay = HttpContext.Application["ThangNay"].ToString();  item.ThangTruoc = HttpContext.Application["ThangTruoc"].ToString();  item.TatCa = HttpContext.Application["TatCa"].ToString();  return PartialView(item);  }  public ActionResult Contact()  {  ViewBag.Message = "Your contact page.";  return View();  }  }  } |
| * Code này xử lý có nhiệm vụ xử lý yêu cầu liên quan đến trang chủ trên trang web. | |

Bảng . HomeController

* + 1. **ManageController**

|  |  |
| --- | --- |
|  | using System;  using System.Linq;  using System.Threading.Tasks;  using System.Web;  using System.Web.Mvc;  using Microsoft.AspNet.Identity;  using Microsoft.AspNet.Identity.Owin;  using Microsoft.Owin.Security;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Controllers  {  [Authorize]  public class ManageController : Controller  {  private ApplicationSignInManager \_signInManager;  private ApplicationUserManager \_userManager;  public ManageController()  {  }  public ManageController(ApplicationUserManager userManager, ApplicationSignInManager signInManager)  {  UserManager = userManager;  SignInManager = signInManager;  }  public ApplicationSignInManager SignInManager  {  get  {  return \_signInManager ?? HttpContext.GetOwinContext().Get<ApplicationSignInManager>();  }  private set  {  \_signInManager = value;  }  }  public ApplicationUserManager UserManager  {  get  {  return \_userManager ?? HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>();  }  private set  {  \_userManager = value;  }  }  //  // GET: /Manage/Index  public async Task<ActionResult> Index(ManageMessageId? message)  {  ViewBag.StatusMessage =  message == ManageMessageId.ChangePasswordSuccess ? "Your password has been changed."  : message == ManageMessageId.SetPasswordSuccess ? "Your password has been set."  : message == ManageMessageId.SetTwoFactorSuccess ? "Your two-factor authentication provider has been set."  : message == ManageMessageId.Error ? "An error has occurred."  : message == ManageMessageId.AddPhoneSuccess ? "Your phone number was added."  : message == ManageMessageId.RemovePhoneSuccess ? "Your phone number was removed."  : "";  var userId = User.Identity.GetUserId();  var model = new IndexViewModel  {  HasPassword = HasPassword(),  PhoneNumber = await UserManager.GetPhoneNumberAsync(userId),  TwoFactor = await UserManager.GetTwoFactorEnabledAsync(userId),  Logins = await UserManager.GetLoginsAsync(userId),  BrowserRemembered = await AuthenticationManager.TwoFactorBrowserRememberedAsync(userId)  };  return View(model);  }  //  // POST: /Manage/RemoveLogin  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> RemoveLogin(string loginProvider, string providerKey)  {  ManageMessageId? message;  var result = await UserManager.RemoveLoginAsync(User.Identity.GetUserId(), new UserLoginInfo(loginProvider, providerKey));  if (result.Succeeded)  {  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user != null)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  }  message = ManageMessageId.RemoveLoginSuccess;  }  else  {  message = ManageMessageId.Error;  }  return RedirectToAction("ManageLogins", new { Message = message });  }  //  // GET: /Manage/AddPhoneNumber  public ActionResult AddPhoneNumber()  {  return View();  }  //  // POST: /Manage/AddPhoneNumber  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> AddPhoneNumber(AddPhoneNumberViewModel model)  {  if (!ModelState.IsValid)  {  return View(model);  }  // Generate the token and send it  var code = await UserManager.GenerateChangePhoneNumberTokenAsync(User.Identity.GetUserId(), model.Number);  if (UserManager.SmsService != null)  {  var message = new IdentityMessage  {  Destination = model.Number,  Body = "Your security code is: " + code  };  await UserManager.SmsService.SendAsync(message);  }  return RedirectToAction("VerifyPhoneNumber", new { PhoneNumber = model.Number });  }  //  // POST: /Manage/EnableTwoFactorAuthentication  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> EnableTwoFactorAuthentication()  {  await UserManager.SetTwoFactorEnabledAsync(User.Identity.GetUserId(), true);  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user != null)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  }  return RedirectToAction("Index", "Manage");  }  //  // POST: /Manage/DisableTwoFactorAuthentication  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> DisableTwoFactorAuthentication()  {  await UserManager.SetTwoFactorEnabledAsync(User.Identity.GetUserId(), false);  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user != null)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  }  return RedirectToAction("Index", "Manage");  }  //  // GET: /Manage/VerifyPhoneNumber  public async Task<ActionResult> VerifyPhoneNumber(string phoneNumber)  {  var code = await UserManager.GenerateChangePhoneNumberTokenAsync(User.Identity.GetUserId(), phoneNumber);  // Send an SMS through the SMS provider to verify the phone number  return phoneNumber == null ? View("Error") : View(new VerifyPhoneNumberViewModel { PhoneNumber = phoneNumber });  }  //  // POST: /Manage/VerifyPhoneNumber  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> VerifyPhoneNumber(VerifyPhoneNumberViewModel model)  {  if (!ModelState.IsValid)  {  return View(model);  }  var result = await UserManager.ChangePhoneNumberAsync(User.Identity.GetUserId(), model.PhoneNumber, model.Code);  if (result.Succeeded)  {  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user != null)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  }  return RedirectToAction("Index", new { Message = ManageMessageId.AddPhoneSuccess });  }  // If we got this far, something failed, redisplay form  ModelState.AddModelError("", "Failed to verify phone");  return View(model);  }  //  // POST: /Manage/RemovePhoneNumber  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> RemovePhoneNumber()  {  var result = await UserManager.SetPhoneNumberAsync(User.Identity.GetUserId(), null);  if (!result.Succeeded)  {  return RedirectToAction("Index", new { Message = ManageMessageId.Error });  }  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user != null)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  }  return RedirectToAction("Index", new { Message = ManageMessageId.RemovePhoneSuccess });  }  //  // GET: /Manage/ChangePassword  public ActionResult ChangePassword()  {  return View();  }  //  // POST: /Manage/ChangePassword  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> ChangePassword(ChangePasswordViewModel model)  {  if (!ModelState.IsValid)  {  return View(model);  }  var result = await UserManager.ChangePasswordAsync(User.Identity.GetUserId(), model.OldPassword, model.NewPassword);  if (result.Succeeded)  {  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user != null)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  }  return RedirectToAction("Index", new { Message = ManageMessageId.ChangePasswordSuccess });  }  AddErrors(result);  return View(model);  }  //  // GET: /Manage/SetPassword  public ActionResult SetPassword()  {  return View();  }  //  // POST: /Manage/SetPassword  [HttpPost]  [ValidateAntiForgeryToken]  public async Task<ActionResult> SetPassword(SetPasswordViewModel model)  {  if (ModelState.IsValid)  {  var result = await UserManager.AddPasswordAsync(User.Identity.GetUserId(), model.NewPassword);  if (result.Succeeded)  {  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user != null)  {  await SignInManager.SignInAsync(user, isPersistent: false, rememberBrowser: false);  }  return RedirectToAction("Index", new { Message = ManageMessageId.SetPasswordSuccess });  }  AddErrors(result);  }  // If we got this far, something failed, redisplay form  return View(model);  }  //  // GET: /Manage/ManageLogins  public async Task<ActionResult> ManageLogins(ManageMessageId? message)  {  ViewBag.StatusMessage =  message == ManageMessageId.RemoveLoginSuccess ? "The external login was removed."  : message == ManageMessageId.Error ? "An error has occurred."  : "";  var user = await UserManager.FindByIdAsync(User.Identity.GetUserId());  if (user == null)  {  return View("Error");  }  var userLogins = await UserManager.GetLoginsAsync(User.Identity.GetUserId());  var otherLogins = AuthenticationManager.GetExternalAuthenticationTypes().Where(auth => userLogins.All(ul => auth.AuthenticationType != ul.LoginProvider)).ToList();  ViewBag.ShowRemoveButton = user.PasswordHash != null || userLogins.Count > 1;  return View(new ManageLoginsViewModel  {  CurrentLogins = userLogins,  OtherLogins = otherLogins  });  }  //  // POST: /Manage/LinkLogin  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult LinkLogin(string provider)  {  // Request a redirect to the external login provider to link a login for the current user  return new AccountController.ChallengeResult(provider, Url.Action("LinkLoginCallback", "Manage"), User.Identity.GetUserId());  }  //  // GET: /Manage/LinkLoginCallback  public async Task<ActionResult> LinkLoginCallback()  {  var loginInfo = await AuthenticationManager.GetExternalLoginInfoAsync(XsrfKey, User.Identity.GetUserId());  if (loginInfo == null)  {  return RedirectToAction("ManageLogins", new { Message = ManageMessageId.Error });  }  var result = await UserManager.AddLoginAsync(User.Identity.GetUserId(), loginInfo.Login);  return result.Succeeded ? RedirectToAction("ManageLogins") : RedirectToAction("ManageLogins", new { Message = ManageMessageId.Error });  }  protected override void Dispose(bool disposing)  {  if (disposing && \_userManager != null)  {  \_userManager.Dispose();  \_userManager = null;  }  base.Dispose(disposing);  }  #region Helpers  // Used for XSRF protection when adding external logins  private const string XsrfKey = "XsrfId";  private IAuthenticationManager AuthenticationManager  {  get  {  return HttpContext.GetOwinContext().Authentication;  }  }  private void AddErrors(IdentityResult result)  {  foreach (var error in result.Errors)  {  ModelState.AddModelError("", error);  }  }  private bool HasPassword()  {  var user = UserManager.FindById(User.Identity.GetUserId());  if (user != null)  {  return user.PasswordHash != null;  }  return false;  }  private bool HasPhoneNumber()  {  var user = UserManager.FindById(User.Identity.GetUserId());  if (user != null)  {  return user.PhoneNumber != null;  }  return false;  }  public enum ManageMessageId  {  AddPhoneSuccess,  ChangePasswordSuccess,  SetTwoFactorSuccess,  SetPasswordSuccess,  RemoveLoginSuccess,  RemovePhoneSuccess,  Error  }  #endregion  }  } |
| * Code này xử lý có nhiệm vụ xử lý yêu cầu liên quan đến quản lý thông tin người dùng như thay đổi mật khẩu, thêm/ xóa số điện thoại,…. | |

Bảng . ManageController

* + 1. **MenuController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Controllers  {  public class MenuController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Menu  public ActionResult Index()  {  return View();  }  public ActionResult MenuTop()  {  var items = db.Categories.OrderBy(x=>x.Position).ToList();  return PartialView("\_MenuTop", items);  }  public ActionResult MenuProductCategory()  {  var items = db.ProductCategories.ToList();  return PartialView("\_MenuProductCategory", items);  }  public ActionResult MenuLeft(int? id)  {  if (id != null)  {  ViewBag.CateId = id;  }  var items = db.ProductCategories.ToList();  return PartialView("\_MenuLeft", items);  }  public ActionResult MenuArrivals()  {  var items = db.ProductCategories.ToList();  return PartialView("\_MenuArrivals", items);  }  }  } |
| * Code này xử lý có nhiệm vụ xử lý yêu cầu liên quan đến quản lý menu và hiển thị các menu tương ứng trên giao diện. | |

Bảng . MenuController

* + 1. **NewsController**

|  |  |
| --- | --- |
|  | using PagedList;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Controllers  {  public class NewsController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: News  public ActionResult Index(int? page)  {  var pageSize = 1;  if (page == null)  {  page = 1;  }  IEnumerable<News> items = db.News.OrderByDescending(x=>x.CreatedDate);  var pageIndex = page.HasValue ? Convert.ToInt32(page) : 1;  items = items.ToPagedList(pageIndex, pageSize);  ViewBag.PageSize = pageSize;  ViewBag.Page = page;  return View(items);  }  public ActionResult Detail(int id)  {  var item = db.News.Find(id);  return View(item);  }  public ActionResult Partial\_News\_Home()  {  var items = db.News.Take(3).ToList();  return PartialView(items);  }  }  } |
| * Code này xử lý có nhiệm vụ xử lý yêu cầu liên đến hiển thị danh sách tin tức trên giao diện người dùng. | |

Bảng . NewsController

* + 1. **ProductController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  namespace WebBanHangOnline.Controllers  {  public class ProductsController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: Products  public ActionResult Index()  {  var items = db.Products.ToList();    return View(items);  }  public ActionResult Detail(string alias,int id)  {  var item = db.Products.Find(id);  if (item != null)  {  db.Products.Attach(item);  item.ViewCount = item.ViewCount + 1;  db.Entry(item).Property(x => x.ViewCount).IsModified = true;  db.SaveChanges();  }    return View(item);  }  public ActionResult ProductCategory(string alias,int id)  {  var items = db.Products.ToList();  if (id > 0)  {  items = items.Where(x => x.ProductCategoryId == id).ToList();  }  var cate = db.ProductCategories.Find(id);  if (cate != null)  {  ViewBag.CateName = cate.Title;  }  ViewBag.CateId = id;  return View(items);  }  public ActionResult Partial\_ItemsByCateId()  {  var items = db.Products.Where(x => x.IsHome && x.IsActive).Take(12).ToList();  return PartialView(items);  }  public ActionResult Partial\_ProductSales()  {  var items = db.Products.Where(x => x.IsSale && x.IsActive).Take(12).ToList();  return PartialView(items);  }  }  } |
| * Code này xử lý có nhiệm vụ quản lý sản phẩm và hiển thị danh sách sản phẩm trên giao diện web. | |

Bảng . ProductController

* + 1. **ShoppingcartController**

|  |  |
| --- | --- |
|  | using System;  using System.Collections.Generic;  using System.Configuration;  using System.Linq;  using System.Web;  using System.Web.Mvc;  using WebBanHangOnline.Models;  using WebBanHangOnline.Models.EF;  namespace WebBanHangOnline.Controllers  {  public class ShoppingCartController : Controller  {  private ApplicationDbContext db = new ApplicationDbContext();  // GET: ShoppingCart  public ActionResult Index()  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null && cart.Items.Any())  {  ViewBag.CheckCart = cart;  }  return View();  }  public ActionResult CheckOut()  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null && cart.Items.Any())  {  ViewBag.CheckCart = cart;  }  return View();  }  public ActionResult CheckOutSuccess()  {  return View();  }  public ActionResult Partial\_Item\_ThanhToan()  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null && cart.Items.Any())  {  return PartialView(cart.Items);  }  return PartialView();  }  public ActionResult Partial\_Item\_Cart()  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null && cart.Items.Any())  {  return PartialView(cart.Items);  }  return PartialView();  }  public ActionResult ShowCount()  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null)  {  return Json(new { Count = cart.Items.Count }, JsonRequestBehavior.AllowGet);  }  return Json(new { Count = 0 }, JsonRequestBehavior.AllowGet);  }  public ActionResult Partial\_CheckOut()  {  return PartialView();  }  [HttpPost]  [ValidateAntiForgeryToken]  public ActionResult CheckOut(OrderViewModel req)  {  var code = new { Success = false, Code = -1 };  if (ModelState.IsValid)  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null)  {  Order order = new Order();  order.CustomerName = req.CustomerName;  order.Phone = req.Phone;  order.Address = req.Address;  order.Email = req.Email;  cart.Items.ForEach(x => order.OrderDetails.Add(new OrderDetail  {  ProductId = x.ProductId,  Quantity = x.Quantity,  Price = x.Price  }));  order.TotalAmount = cart.Items.Sum(x => (x.Price \* x.Quantity));  order.TypePayment = req.TypePayment;  order.CreatedDate = DateTime.Now;  order.ModifiedDate = DateTime.Now;  order.CreatedBy = req.Phone;  Random rd = new Random();  order.Code = "DH" + rd.Next(0, 9) + rd.Next(0, 9) + rd.Next(0, 9) + rd.Next(0, 9);  //order.E = req.CustomerName;  db.Orders.Add(order);  db.SaveChanges();  //send mail cho khachs hang  var strSanPham = "";  var thanhtien = decimal.Zero;  var TongTien = decimal.Zero;  foreach (var sp in cart.Items)  {  strSanPham += "<tr>";  strSanPham += "<td>" + sp.ProductName + "</td>";  strSanPham += "<td>" + sp.Quantity + "</td>";  strSanPham += "<td>" + WebBanHangOnline.Common.Common.FormatNumber(sp.TotalPrice, 0) + "</td>";  strSanPham += "</tr>";  thanhtien += sp.Price \* sp.Quantity;  }  TongTien = thanhtien;  string contentCustomer = System.IO.File.ReadAllText(Server.MapPath("~/Content/templates/send2.html"));  contentCustomer = contentCustomer.Replace("{{MaDon}}", order.Code);  contentCustomer = contentCustomer.Replace("{{SanPham}}", strSanPham);  contentCustomer = contentCustomer.Replace("{{NgayDat}}", DateTime.Now.ToString("dd/MM/yyyy"));  contentCustomer = contentCustomer.Replace("{{TenKhachHang}}", order.CustomerName);  contentCustomer = contentCustomer.Replace("{{Phone}}", order.Phone);  contentCustomer = contentCustomer.Replace("{{Email}}", req.Email);  contentCustomer = contentCustomer.Replace("{{DiaChiNhanHang}}", order.Address);  contentCustomer = contentCustomer.Replace("{{ThanhTien}}", WebBanHangOnline.Common.Common.FormatNumber(thanhtien, 0));  contentCustomer = contentCustomer.Replace("{{TongTien}}", WebBanHangOnline.Common.Common.FormatNumber(TongTien, 0));  WebBanHangOnline.Common.Common.SendMail("ShopOnline", "Đơn hàng #" + order.Code, contentCustomer.ToString(), req.Email);  string contentAdmin = System.IO.File.ReadAllText(Server.MapPath("~/Content/templates/send1.html"));  contentAdmin = contentAdmin.Replace("{{MaDon}}", order.Code);  contentAdmin = contentAdmin.Replace("{{SanPham}}", strSanPham);  contentAdmin = contentAdmin.Replace("{{NgayDat}}", DateTime.Now.ToString("dd/MM/yyyy"));  contentAdmin = contentAdmin.Replace("{{TenKhachHang}}", order.CustomerName);  contentAdmin = contentAdmin.Replace("{{Phone}}", order.Phone);  contentAdmin = contentAdmin.Replace("{{Email}}", req.Email);  contentAdmin = contentAdmin.Replace("{{DiaChiNhanHang}}", order.Address);  contentAdmin = contentAdmin.Replace("{{ThanhTien}}", WebBanHangOnline.Common.Common.FormatNumber(thanhtien, 0));  contentAdmin = contentAdmin.Replace("{{TongTien}}", WebBanHangOnline.Common.Common.FormatNumber(TongTien, 0));  WebBanHangOnline.Common.Common.SendMail("ShopOnline", "Đơn hàng mới #" + order.Code, contentAdmin.ToString(), ConfigurationManager.AppSettings["EmailAdmin"]);  cart.ClearCart();  return RedirectToAction("CheckOutSuccess");  }  }  return Json(code);  }  [HttpPost]  public ActionResult AddToCart(int id, int quantity)  {  var code = new { Success = false, msg = "", code = -1, Count = 0 };  var db = new ApplicationDbContext();  var checkProduct = db.Products.FirstOrDefault(x => x.Id == id);  if (checkProduct != null)  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart == null)  {  cart = new ShoppingCart();  }  ShoppingCartItem item = new ShoppingCartItem  {  ProductId = checkProduct.Id,  ProductName = checkProduct.Title,  CategoryName = checkProduct.ProductCategory.Title,  Alias = checkProduct.Alias,  Quantity = quantity  };  if (checkProduct.ProductImage.FirstOrDefault(x => x.IsDefault) != null)  {  item.ProductImg = checkProduct.ProductImage.FirstOrDefault(x => x.IsDefault).Image;  }  item.Price = checkProduct.Price;  if (checkProduct.PriceSale > 0)  {  item.Price = (decimal)checkProduct.PriceSale;  }  item.TotalPrice = item.Quantity \* item.Price;  cart.AddToCart(item, quantity);  Session["Cart"] = cart;  code = new { Success = true, msg = "Thêm sản phẩm vào giở hàng thành công!", code = 1, Count = cart.Items.Count };  }  return Json(code);  }  [HttpPost]  public ActionResult Update(int id, int quantity)  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null)  {  cart.UpdateQuantity(id, quantity);  return Json(new { Success = true });  }  return Json(new { Success = false });  }  [HttpPost]  public ActionResult Delete(int id)  {  var code = new { Success = false, msg = "", code = -1, Count = 0 };  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null)  {  var checkProduct = cart.Items.FirstOrDefault(x => x.ProductId == id);  if (checkProduct != null)  {  cart.Remove(id);  code = new { Success = true, msg = "", code = 1, Count = cart.Items.Count };  }  }  return Json(code);  }  [HttpPost]  public ActionResult DeleteAll()  {  ShoppingCart cart = (ShoppingCart)Session["Cart"];  if (cart != null)  {  cart.ClearCart();  return Json(new { Success = true });  }  return Json(new { Success = false });  }  }  } |
| * Code này xử lý có nhiệm vụ quản lý giỏ hàng người dùng và xử lý chức năng liên quan đến giỏ hàng. | |

Bảng . ShoppingCartController

# 3. THỰC HIỆN PHẦN MỀM

Trong bản demo này chỉ cài đặt cái chức năng chủ yếu như lấy thông tin từ database, thay đổi, thêm, xóa các thông tin đó, đăng nhập vào hệ thống, thông báo lỗi.

Các chức năng khác sẽ được phát triển trong các phiên bản sau.

## 3.1. Kiến trúc phần mềm

* 1 – Tier, MVC

## Phương pháp cài đặt

* Database First

## Môi trường cài đặt

* Hệ điều hành Windows
* Công cụ phát triển: Visual Studio, Microsoft SQL Server
* Công cụ quản lý: Github

## Một số lưu ý khi cài đặt

Ngôn ngữ lập trình: C#, HTML, JAVASCRIPT, SQL

# TÀI LIỆU THAM KHẢO

1. Microsoft, ASP.NET documentation   
   https://docs.microsoft.com/vi-vn/aspnet/core/?view=aspnetcore-5.0
2. A. Freeman, "Professional ASP.NET MVC 5," 2013.
3. L. Taylor, "ASP.NET MVC with Entity Framework," 2016.
4. S. Feuerstein, "Oracle PL/SQL Best Practices: Optimizing Oracle Code," 2001.
5. S. Feuerstein, "Advanced Oracle PL/SQL," 1996.