# Phân trang, tìm kiếm, Sắp xếp và Ajax

Mục Lục

[Phân trang, tìm kiếm, Sắp xếp và Ajax 1](#_Toc173833254)

[I. Phân trang, tìm kiếm, sắp xếp 2](#_Toc173833255)

[1. Phân trang Product Admin 2](#_Toc173833256)

[2. tìm kiếm trong trang Product Admin 12](#_Toc173833257)

[3. Sắp xếp Product Admin 19](#_Toc173833258)

[4. Phân trang Product ở trang cửa hàng 26](#_Toc173833259)

[5. Sắp xếp Product trang cửa hàng 32](#_Toc173833260)

[II. AJAX 40](#_Toc173833261)

[6. Ajax xóa sản phẩm trong giỏ hàng 41](#_Toc173833262)

[7. Ajax thêm, bớt số lượng sản phầm trong giỏ hàng 54](#_Toc173833263)

[8. Ajax thêm sản phầm vào giỏ hàng 60](#_Toc173833264)

## Phân trang, tìm kiếm, sắp xếp

### Phân trang Product Admin

**Admin/ProductAdmin/Index**

Tạo Model mới

**Areas/Admin/Models/ProductAdmin.cs**

using Microsoft.AspNetCore.Mvc.Rendering;

using System.ComponentModel.DataAnnotations;

namespace ShopManager.Areas.Admin.Models

{

public class ProductAdmin

{

[Key]

public int Id { get; set; }

[Display(Name = "Title")]

[Required(ErrorMessage = "\*")]

[MaxLength(100, ErrorMessage = "Tối đa 100 kí tự")]

public string Title { get; set; }

[Display(Name = "Content")]

[Required(ErrorMessage = "\*")]

[MaxLength(500, ErrorMessage = "Tối đa 500 kí tự")]

public string Content { get; set; }

[Display(Name = "Image")]

public string Img { get; set; }

[Display(Name = "Price")]

[Required(ErrorMessage = "\*")]

public int Price { get; set; }

[Display(Name = "Rating")]

[Required(ErrorMessage = "\*")]

public double Rate { get; set; }

[Display(Name = "Create At")]

public DateTime CreateAt { get; set; }

[Display(Name = "Update At")]

public DateTime UpdateAt { get; set; }

[Display(Name = "Category Id")]

[Required(ErrorMessage = "\*")]

public int CategoryId { get; set; }

[Display(Name = "Category Title")]

[Required(ErrorMessage = "\*")]

public string CategoryTitle { get; set; }

}

public class ProductFormAdmin : ProductAdmin

{

public string IdCategorySelected { get; set; }

public List<SelectListItem>? ListCategory { get; set; }

}

public class ProductAdminModel

{

///<summary>

/// Gets or sets ProductAdmin.

///</summary>

public List<ProductAdmin> ProductAdmins { get; set; }

///<summary>

/// Gets or sets CurrentPageIndex.

///</summary>

public int CurrentPageIndex { get; set; }

///<summary>

/// Gets or sets PageCount.

///</summary>

public int PageCount { get; set; }

}

}

**Areas/Admin/DAL/ProductAdminDAL.cs**

using Microsoft.CodeAnalysis.Elfie.Diagnostics;

using Microsoft.Data.SqlClient;

using ShopManager.Areas.Admin.Models;

using ShopManager.Database;

using System.Data;

namespace ShopManager.Areas.Admin.DAL

{

public class ProductAdminDAL

{

DBConnect connect = new DBConnect();

public List<ProductAdmin> getAll()

{

....

}

public bool AddNew(ProductFormAdmin productNew)

{

....

}

public ProductAdmin GetProductById(int Id)

{

....

}

public bool UpdateProduct(ProductFormAdmin productNew, int Id)

{

....

}

public bool DeleteProduct(int Id)

{

....

}

// Lấy danh sách sản phầm theo phân trang (số trang hiện tại, số hàng trong 1 trang)

public List<ProductAdmin> getProduct\_Pagination(int pageIndex, int pageSize)

{

connect.openConnection();

List<ProductAdmin> list = new List<ProductAdmin>();

using (SqlCommand command = new SqlCommand())

{

command.Connection = connect.getConnecttion();

command.CommandType = System.Data.CommandType.Text;

//Truy vấn lồng nhau

// ROW\_NUMBER() OVER(ORDER BY a.id asc) AS RowNumber tạo thêm 1 cột để lưu index (tương tự số thứ tự)

// sau đó sử dụng câu lệnh BETWEEN(start, end) để lấy dữ liệu có RowNumber (index ở trên):

// start <= RowNumber <= end

string query = @"

SELECT \* FROM (

SELECT ROW\_NUMBER() OVER(ORDER BY a.id asc) AS RowNumber,

a.id as ProductId, a.title as ProductTitle,

a.content as ProductContent, a.img as ProductImg, a.price as ProductPrice,

a.rate as ProductRate, a.createAt as ProductCreateAt, a.updateAt as ProductUpdateAt,

b.id as CategoryId, b.title as CategoryTitle

FROM product a join category b on a.categoryId = b.Id

) TableResult

WHERE TableResult.RowNumber BETWEEN( @PageIndex -1) \* @PageSize + 1

AND @PageIndex \* @PageSize ";

command.CommandText = query;

command.Parameters.AddWithValue("@PageIndex", pageIndex);

command.Parameters.AddWithValue("@PageSize", pageSize);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ProductAdmin product = new ProductAdmin()

{

Id = Convert.ToInt32(reader["ProductId"]),

Title = reader["ProductTitle"].ToString() ?? "",

Content = reader["ProductContent"].ToString() ?? "",

Img = reader["ProductImg"].ToString() ?? "",

Price = Convert.ToInt32(reader["ProductPrice"]),

Rate = Convert.ToDouble(reader["ProductRate"].ToString()),

CreateAt = DateTime.Parse(reader["ProductCreateAt"]?.ToString()),

UpdateAt = DateTime.Parse(reader["ProductUpdateAt"]?.ToString()),

CategoryId = Convert.ToInt32(reader["CategoryId"]),

CategoryTitle = reader["CategoryTitle"].ToString(),

};

list.Add(product);

}

}

connect.closeConnection();

return list;

}

}

}

**Giải thích câu truy vấn SQL:**

SELECT ROW\_NUMBER() OVER(ORDER BY a.id desc) AS RowNumber,

a.id as ProductId, a.title as ProductTitle,

a.content as ProductContent, a.img as ProductImg, a.price as ProductPrice,

a.rate as ProductRate, a.createAt as ProductCreateAt, a.updateAt as ProductUpdateAt,

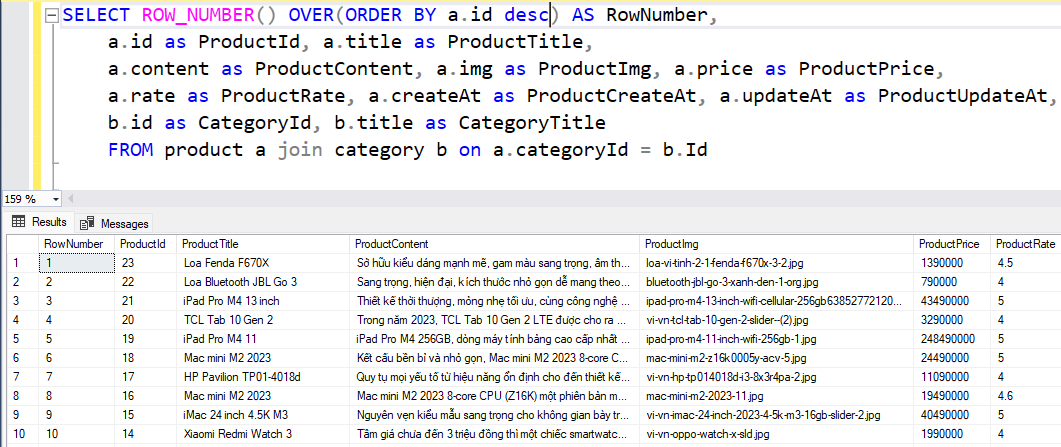
b.id as CategoryId, b.title as CategoryTitle

FROM product a join category b on a.categoryId = b.Id

Trong đó

ROW\_NUMBER() OVER(ORDER BY a.id desc) AS RowNumber

Tạo thêm 1 cột tên RowNumber hiển thị Số thứ tự (index)



Sử dụng thêm 1 câu truy vấn để giới hạn số lượng theo trang từ câu lệnh truy vấn trước đó

-- khai báo

Declare @PageIndex int, @PageSize int;

set @PageIndex = 1

set @PageSize = 5

-- truy vấn

SELECT \* FROM (

SELECT ROW\_NUMBER() OVER(ORDER BY a.id desc) AS RowNumber,

a.id as ProductId, a.title as ProductTitle,

a.content as ProductContent, a.img as ProductImg, a.price as ProductPrice,

a.rate as ProductRate, a.createAt as ProductCreateAt, a.updateAt as ProductUpdateAt,

b.id as CategoryId, b.title as CategoryTitle

FROM product a join category b on a.categoryId = b.Id

) TableResult

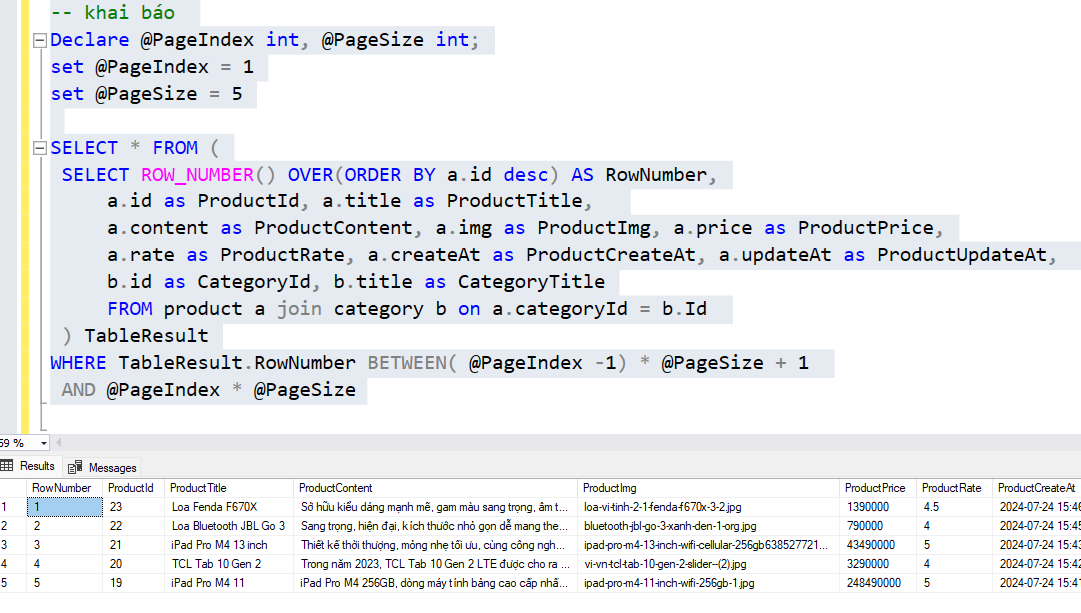
WHERE TableResult.RowNumber BETWEEN( @PageIndex -1) \* @PageSize + 1

AND @PageIndex \* @PageSize

Khai báo cố định **PageIndex = 1** và **PageSize = 5**

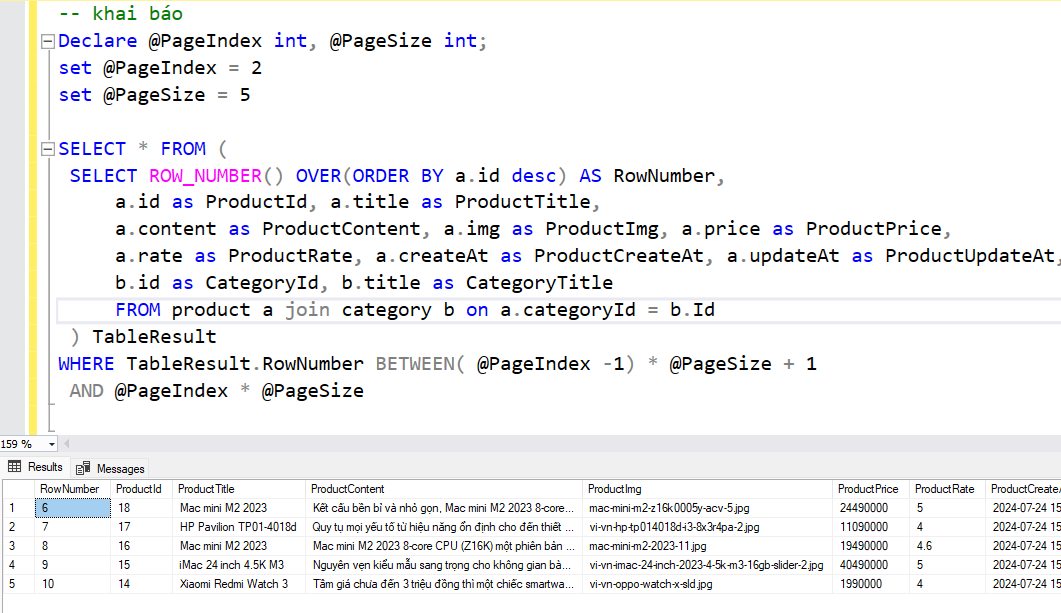
Hiển thị **trang số 1**, trong trang đó **hiển thị 5 hàng** dữ liệu **(Row)**

Sử dụng **BETWEEN(start, end)** để xác định dữ liệu sẽ lấy bắt đầu từ **start** và kết thúc tại **end**



Hiển thị trang 2, số hàng trong trang là 5

PageIndex = 2, PageSize = 5



**Areas/Admin/Controllers/ProductAdminController.cs** (thay đổi Action Index)

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.Data.SqlClient;

using ShopManager.Areas.Admin.DAL;

using ShopManager.Areas.Admin.Models;

using ShopManager.Helper;

namespace ShopManager.Areas.Admin.Controllers

{

[Area("Admin")]

[Authorize(Roles = "Administrator")]

public class ProductAdminController : Controller

{

ProductAdminDAL productDAL = new ProductAdminDAL();

CategoryAdminDAL categoryDAL = new CategoryAdminDAL();

// GET: ProductAdminController

public ActionResult Index(int page = 1)

{

//Khai báo số lượng Row trong 1 trang

int pageSize = 5;

//Text hiển thị trong input sau khi tìm kiếm

ViewData["CurrentFilter"] = searchString;

// Lấy danh sách Product sau khi phân trang

List<ProductAdmin> products = new List<ProductAdmin>();

products = productDAL.getProduct\_Pagination(page, pageSize, searchString.Trim());

// Lấy số lượng Row sau khi truy vấn có thêm điều kiện (tìm kiếm)

int numRows = productDAL.getCountRow\_Pagination(page, pageSize, searchString.Trim());

//Tính số lượng trang sẽ có (làm tròn lên, vd: 4.3 -> 5)

double pageCount = (double)numRows / pageSize;

int maxPage = (int)Math.Ceiling(pageCount);

//Tạo model để hiển thị

ProductAdminModel model = new ProductAdminModel();

model.ProductAdmins = products;

model.CurrentPageIndex = page;

model.PageCount = maxPage;

return View(model);

}

// GET: ProductAdminController/Details/5

public ActionResult Details(int id)

{

...

}

...

}

}

**Areas/Admin/Views/ProductAdmin/Index.cshtml**

~~@\* @model IEnumerable<ShopManager.Areas.Admin.Models.ProductAdmin> \*@~~

@model ShopManager.Areas.Admin.Models.ProductAdminModel

@{

ViewData["Title"] = "Index";

}

<h1>Index</h1>

<p>

<**a** **asp-action**="Create" class="btn btn-primary">Create New</**a**>

</p>

<table class="table">

<thead>

<tr>

<th>

Id

</th>

<th>

Title

</th>

<th>

Content

</th>

<th>

Image

</th>

<th>

Price

</th>

<th>

Rate

</th>

<th>

Create At

</th>

<th>

Update At

</th>

<th>

CategoryId

</th>

<th>

CategoryTitle

</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach (var item in Model.ProductAdmins)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.Id)

</td>

<td>

@Html.DisplayFor(modelItem => item.Title)

</td>

<td>

<span style="line-height:1.5rem; height:4.5rem; display: -webkit-box; -webkit-line-clamp: 3; -webkit-box-orient: vertical; overflow: hidden;">

@Html.DisplayFor(modelItem => item.Content)

</span>

</td>

<td>

<**img** src="~/images/sanpham/@item.Img" class="img-fluid " style="width:100px; height: 60px; object-fit:contain" />

</td>

<td>

@item.Price.ToString("#,##0 VND")

</td>

<td>

@Html.DisplayFor(modelItem => item.Rate)

</td>

<td>

@Html.DisplayFor(modelItem => item.CreateAt)

</td>

<td>

@Html.DisplayFor(modelItem => item.UpdateAt)

</td>

<td>

@Html.DisplayFor(modelItem => item.CategoryId)

</td>

<td>

@Html.DisplayFor(modelItem => item.CategoryTitle)

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id=item.Id }) |

@Html.ActionLink("Details", "Details", new { id=item.Id }) |

@Html.ActionLink("Delete", "Delete", new { id=item.Id })

</td>

</tr>

}

</tbody>

</table>

<br />

<!-- Pagination Start-->

<div class="col-12">

<div class="pagination d-flex justify-content-center mt-5">

<!--Button Previous-->

@if (Model.CurrentPageIndex > 1)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@(Model.CurrentPageIndex -1)"

class="btn btn-outline-dark mx-2">

&laquo;

</**a**>

}

@for (int i = 1; i <= Model.PageCount; i++)

{

@if (i != Model.CurrentPageIndex)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@i"

class="btn btn-outline-dark mx-2">

@i

</**a**>

}

else

{

//Current Page

<a class="btn btn btn-outline-dark active mx-2">@i</a>

}

}

<!-- Button Next -->

@if (Model.PageCount != Model.CurrentPageIndex)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@(Model.CurrentPageIndex + 1)"

class="btn btn-outline-dark mx-2">

&raquo;

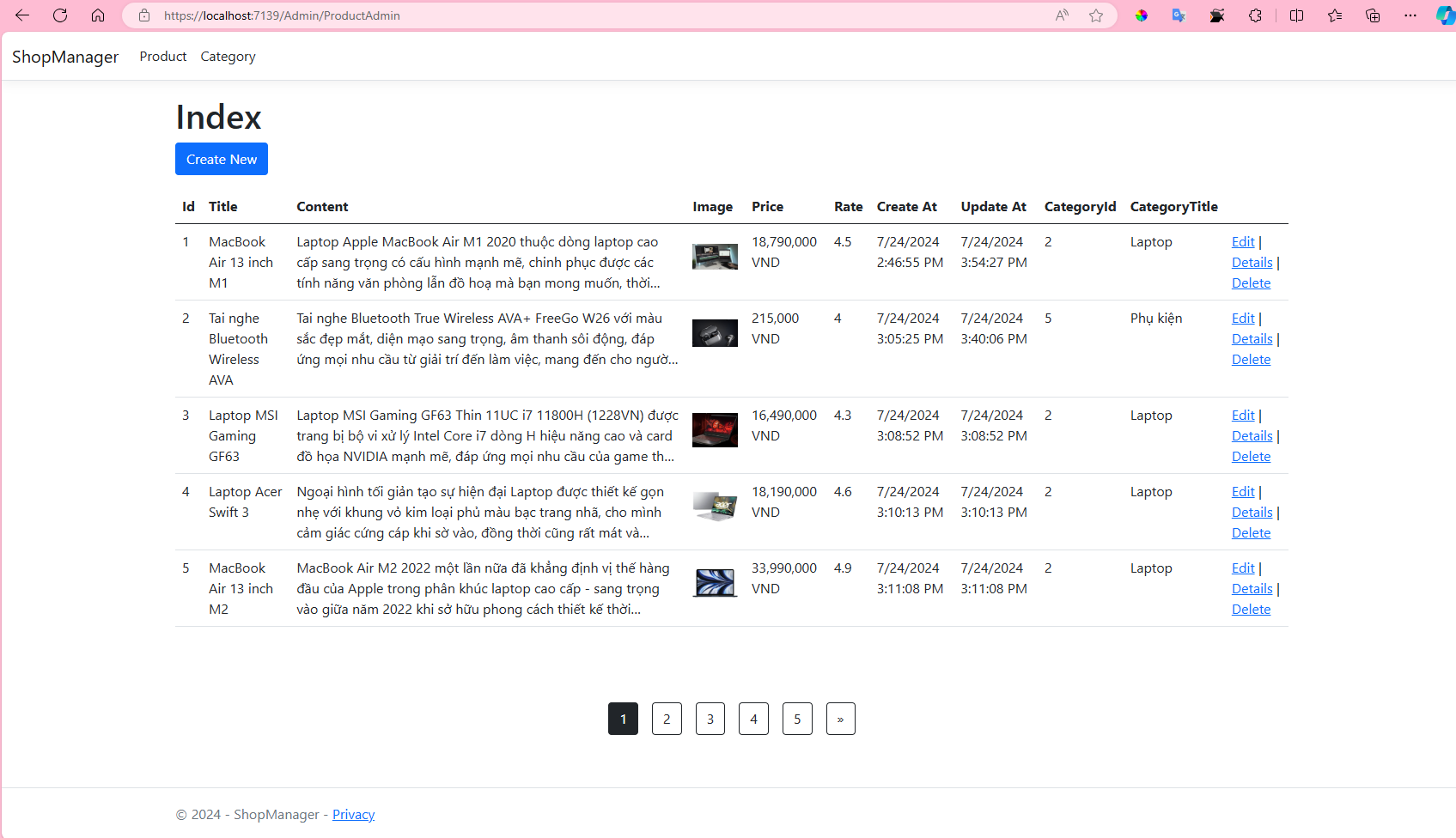
</**a**>

}

</div>

</div>

<!-- Pagination End-->



### tìm kiếm trong trang Product Admin

**Admin/ProductAdmin/Index**

Tạo truy vấn SQL

**Areas/Admin/DAL/ProductAdminDAL.cs**

using Microsoft.CodeAnalysis.Elfie.Diagnostics;

using Microsoft.Data.SqlClient;

using ShopManager.Areas.Admin.Models;

using ShopManager.Database;

using System.Data;

namespace ShopManager.Areas.Admin.DAL

{

public class ProductAdminDAL

{

DBConnect connect = new DBConnect();

...

public bool DeleteProduct(int Id)

{

....

}

// Lấy số lượng kết quả sau khi truy vấn có thêm điều kiện (Search)

public int getCountRow\_Pagination(int pageIndex, int pageSize, string? searchString)

{

connect.openConnection();

// khai báo biến lưu số lượng Row truy vấn được

int count = 0;

//Nếu chuỗi tìm kiếm có dữ liệu thì thêm câu lệnh SQL WHERE

string condition = "";

if (searchString != "" && searchString != null)

{

condition = @" Where a.title Like '%" + searchString + "%' or a.content like '%" + searchString + "%' ";

}

using (SqlCommand command = new SqlCommand())

{

command.Connection = connect.getConnecttion();

command.CommandType = System.Data.CommandType.Text;

//Câu truy vấn

string query = @"

SELECT Count(\*) as CountRow

FROM product a join category b on a.categoryId = b.Id ";

// nối thêm điều kiện, Nếu điều kiện không có thì condition=""

// query = query + ""; thì vẫn = query

query = query + condition;

command.CommandText = query;

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

count = Convert.ToInt32(reader["CountRow"]);

}

}

connect.closeConnection();

return count;

}

// Lấy danh sách sản phầm theo phân trang (số trang hiện tại, số hàng trong 1 trang, tìm kiếm)

public List<ProductAdmin> getProduct\_Pagination(int pageIndex,

int pageSize , string? searchString)

{

connect.openConnection();

List<ProductAdmin> list = new List<ProductAdmin>();

//Nếu chuỗi tìm kiếm có dữ liệu thì thêm câu lệnh SQL WHERE

string condition = "";

if (searchString != "" && searchString != null)

{

condition = @" Where a.title Like '%" + searchString + "%' or a.content like '%" + searchString + "%' ";

}

using (SqlCommand command = new SqlCommand())

{

command.Connection = connect.getConnecttion();

command.CommandType = System.Data.CommandType.Text;

//Truy vấn lồng nhau

// ROW\_NUMBER() OVER(ORDER BY a.id asc) AS RowNumber tạo thêm 1 cột để lưu index (tương tự số thứ tự)

// sau đó sử dụng câu lệnh BETWEEN(start, end) để lấy dữ liệu có RowNumber (index ở trên):

// start <= RowNumber <= end

string query = @"

SELECT \* FROM (

SELECT ROW\_NUMBER() OVER(ORDER BY a.id asc) AS RowNumber,

a.id as ProductId, a.title as ProductTitle,

a.content as ProductContent, a.img as ProductImg, a.price as ProductPrice,

a.rate as ProductRate, a.createAt as ProductCreateAt, a.updateAt as ProductUpdateAt,

b.id as CategoryId, b.title as CategoryTitle

FROM product a join category b on a.categoryId = b.Id

" + condition + @"

) TableResult

WHERE TableResult.RowNumber BETWEEN( @PageIndex -1) \* @PageSize + 1

AND @PageIndex \* @PageSize ";

command.CommandText = query;

command.Parameters.AddWithValue("@PageIndex", pageIndex);

command.Parameters.AddWithValue("@PageSize", pageSize);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ProductAdmin product = new ProductAdmin()

{

Id = Convert.ToInt32(reader["ProductId"]),

Title = reader["ProductTitle"].ToString() ?? "",

Content = reader["ProductContent"].ToString() ?? "",

Img = reader["ProductImg"].ToString() ?? "",

Price = Convert.ToInt32(reader["ProductPrice"]),

Rate = Convert.ToDouble(reader["ProductRate"].ToString()),

CreateAt = DateTime.Parse(reader["ProductCreateAt"]?.ToString()),

UpdateAt = DateTime.Parse(reader["ProductUpdateAt"]?.ToString()),

CategoryId = Convert.ToInt32(reader["CategoryId"]),

CategoryTitle = reader["CategoryTitle"].ToString(),

};

list.Add(product);

}

}

connect.closeConnection();

return list;

}

}

}

**Areas/Admin/Controllers/ProductAdminController.cs**

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.Data.SqlClient;

using ShopManager.Areas.Admin.DAL;

using ShopManager.Areas.Admin.Models;

using ShopManager.Helper;

namespace ShopManager.Areas.Admin.Controllers

{

[Area("Admin")]

[Authorize(Roles = "Administrator")]

public class ProductAdminController : Controller

{

ProductAdminDAL productDAL = new ProductAdminDAL();

CategoryAdminDAL categoryDAL = new CategoryAdminDAL();

// GET: ProductAdminController

public ActionResult Index(int page = 1, string searchString = "")

{

//Khai báo số lượng Row trong 1 trang

int pageSize = 5;

//Text hiển thị trong input sau khi tìm kiếm

ViewData["CurrentFilter"] = searchString;

// Lấy danh sách Product sau khi phân trang

List<ProductAdmin> products = new List<ProductAdmin>();

products = productDAL.getProduct\_Pagination(page

, pageSize , searchString.Trim());

// Lấy số lượng Row sau khi truy vấn có thêm điều kiện (tìm kiếm)

int numRows = productDAL.getCountRow\_Pagination(page, pageSize, searchString.Trim());

//Tính số lượng trang sẽ có (làm tròn lên, vd: 4.3 -> 5)

double pageCount = (double)numRows / pageSize;

int maxPage = (int)Math.Ceiling(pageCount);

//Tạo model để hiển thị

ProductAdminModel model = new ProductAdminModel();

model.ProductAdmins = products;

model.CurrentPageIndex = page;

model.PageCount = maxPage;

return View(model);

}

// GET: ProductAdminController/Details/5

public ActionResult Details(int id) {...}

...

}

}

Areas/Admin/Views/ProductAdmin/Index.cshtml

@model ShopManager.Areas.Admin.Models.ProductAdminModel

@{

ViewData["Title"] = "Index";

}

<h1>Index</h1>

<p>

<**a** **asp-action**="Create" class="btn btn-primary">Create New</**a**>

</p>

<**form** **asp-action**="index" method="get">

<div class="form-actions no-color">

<p>

Find by name: <input type="text" name="SearchString" value="@ViewData["CurrentFilter"]" />

<input type="submit" value="Search" class="btn btn-primary" /> |

<**a** **asp-action**="index" class="btn btn-outline-dark">Back to Full List</**a**>

</p>

</div>

</**form**>

<table class="table">

<thead>

<tr>

...

</tr>

</thead>

<tbody>

@foreach (var item in Model.ProductAdmins)

{

<tr>

...

</tr>

}

</tbody>

</table>

<br />

<!-- Pagination Start-->

<div class="col-12">

<div class="pagination d-flex justify-content-center mt-5">

<!--Button Previous-->

@if (Model.CurrentPageIndex > 1)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@(Model.CurrentPageIndex -1)"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

class="btn btn-outline-dark mx-2">

&laquo;

</**a**>

}

@for (int i = 1; i <= Model.PageCount; i++)

{

@if (i != Model.CurrentPageIndex)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@i"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

class="btn btn-outline-dark mx-2">

@i

</**a**>

}

else

{

//Current Page

<a class="btn btn btn-outline-dark active mx-2">@i</a>

}

}

<!-- Button Next -->

@if (Model.PageCount != Model.CurrentPageIndex)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@(Model.CurrentPageIndex + 1)"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

class="btn btn-outline-dark mx-2">

&raquo;

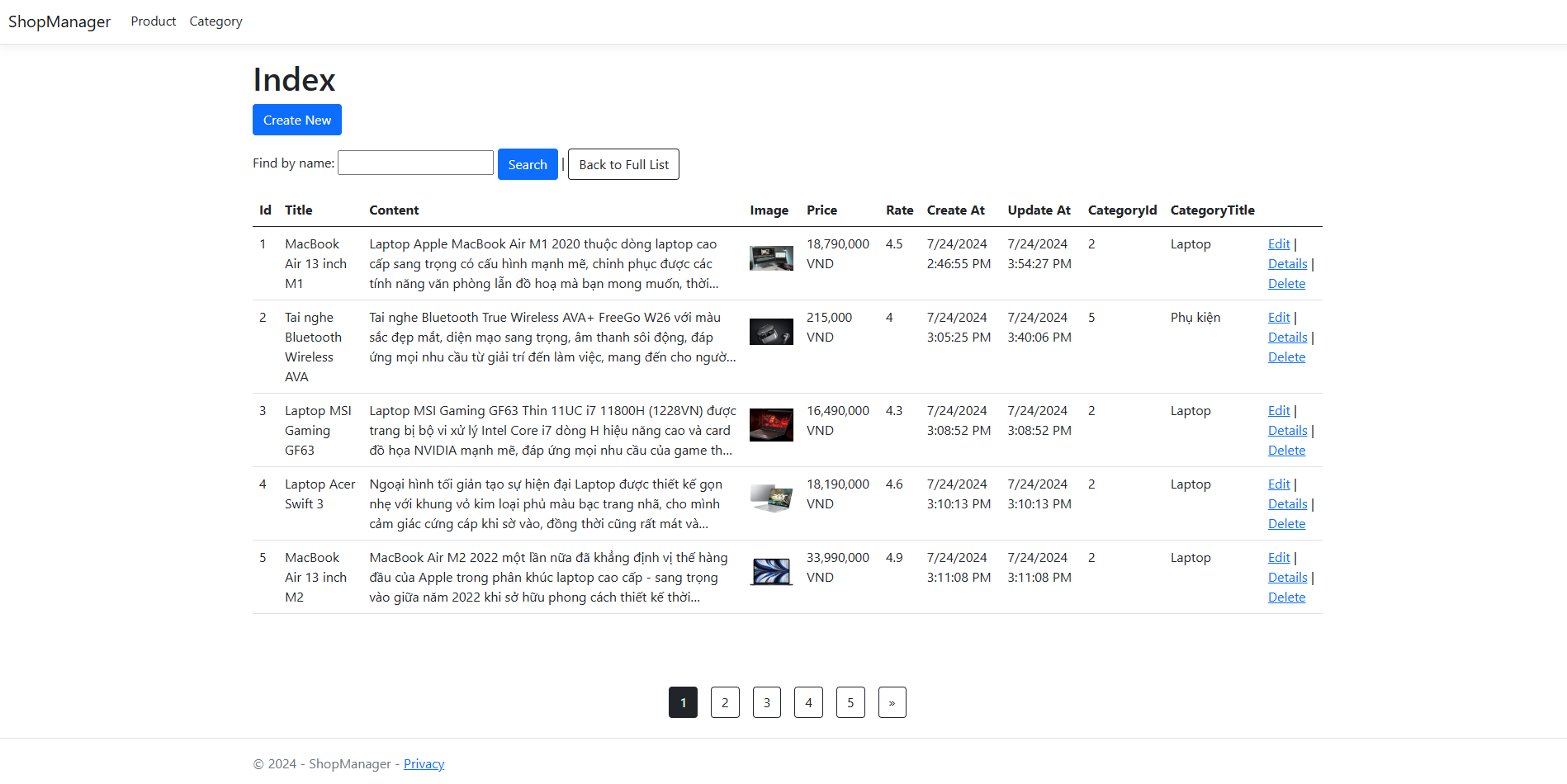
</**a**>

}

</div>

</div>

<!-- Pagination End-->



### Sắp xếp Product Admin

Sắp xếp danh sách Product theo Title, Price và Rate

Tạo truy vấn SQL

Areas/Admin/DAL/ProductAdminDAL.cs

using Microsoft.CodeAnalysis.Elfie.Diagnostics;

using Microsoft.Data.SqlClient;

using ShopManager.Areas.Admin.Models;

using ShopManager.Database;

using System.Data;

namespace ShopManager.Areas.Admin.DAL

{

public class ProductAdminDAL

{

DBConnect connect = new DBConnect();

...

public bool DeleteProduct(int Id)

{

....

}

// Lấy số lượng kết quả sau khi truy vấn có thêm điều kiện (Search)

public int getCountRow\_Pagination(int pageIndex, int pageSize, string? searchString)

{

....

}

// Lấy danh sách sản phầm theo phân trang (số trang hiện tại, số hàng trong 1 trang, tìm kiếm)

public List<ProductAdmin> getProduct\_Pagination(int pageIndex, int pageSize, string? searchString , string sortOrder)

{

connect.openConnection();

List<ProductAdmin> list = new List<ProductAdmin>();

//Nếu chuỗi tìm kiếm có dữ liệu thì thêm câu lệnh SQL WHERE

string condition = "";

if (searchString != "" && searchString != null)

{

condition = @" Where a.title Like '%" + searchString + "%' or a.content like '%" + searchString + "%' ";

}

//Truy vấn Sắp xếp

string sortQuery = " ORDER BY a.id ";

if (!string.IsNullOrEmpty(sortOrder))

{

switch (sortOrder)

{

case "id\_desc":

sortQuery = " ORDER BY a.id DESC ";

break;

case "title":

sortQuery = " ORDER BY a.title ";

break;

case "title\_desc":

sortQuery = " ORDER BY a.title DESC ";

break;

case "price":

sortQuery = " ORDER BY a.price ";

break;

case "price\_desc":

sortQuery = " ORDER BY a.price DESC ";

break;

case "rate":

sortQuery = " ORDER BY a.rate ";

break;

case "rate\_desc":

sortQuery = " ORDER BY a.rate DESC ";

break;

default:

sortQuery = " ORDER BY a.id ";

break;

}

}

using (SqlCommand command = new SqlCommand())

{

command.Connection = connect.getConnecttion();

command.CommandType = System.Data.CommandType.Text;

//Truy vấn lồng nhau

// ROW\_NUMBER() OVER(ORDER BY a.id asc) AS RowNumber tạo thêm 1 cột để lưu index (tương tự số thứ tự)

// sau đó sử dụng câu lệnh BETWEEN(start, end) để lấy dữ liệu có RowNumber (index ở trên):

// start <= RowNumber <= end

string query = @"

SELECT \* FROM (

SELECT ROW\_NUMBER() OVER ( "+ sortQuery + @") AS RowNumber,

a.id as ProductId, a.title as ProductTitle,

a.content as ProductContent, a.img as ProductImg, a.price as ProductPrice,

a.rate as ProductRate, a.createAt as ProductCreateAt, a.updateAt as ProductUpdateAt,

b.id as CategoryId, b.title as CategoryTitle

FROM product a join category b on a.categoryId = b.Id "

+ condition +

@") TableResult

WHERE TableResult.RowNumber BETWEEN( @PageIndex -1) \* @PageSize + 1

AND @PageIndex \* @PageSize ";

command.CommandText = query;

command.Parameters.AddWithValue("@PageIndex", pageIndex);

command.Parameters.AddWithValue("@PageSize", pageSize);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

ProductAdmin product = new ProductAdmin()

{

Id = Convert.ToInt32(reader["ProductId"]),

Title = reader["ProductTitle"].ToString() ?? "",

Content = reader["ProductContent"].ToString() ?? "",

Img = reader["ProductImg"].ToString() ?? "",

Price = Convert.ToInt32(reader["ProductPrice"]),

Rate = Convert.ToDouble(reader["ProductRate"].ToString()),

CreateAt = DateTime.Parse(reader["ProductCreateAt"]?.ToString()),

UpdateAt = DateTime.Parse(reader["ProductUpdateAt"]?.ToString()),

CategoryId = Convert.ToInt32(reader["CategoryId"]),

CategoryTitle = reader["CategoryTitle"].ToString(),

};

list.Add(product);

}

}

connect.closeConnection();

return list;

}

}

}

Areas/Admin/Controllers/ProductAdminController.cs

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.Data.SqlClient;

using ShopManager.Areas.Admin.DAL;

using ShopManager.Areas.Admin.Models;

using ShopManager.Helper;

namespace ShopManager.Areas.Admin.Controllers

{

[Area("Admin")]

[Authorize(Roles = "Administrator")]

public class ProductAdminController : Controller

{

ProductAdminDAL productDAL = new ProductAdminDAL();

CategoryAdminDAL categoryDAL = new CategoryAdminDAL();

// GET: ProductAdminController

public ActionResult Index(int page = 1, string searchString = "" , string sortOrder = "" )

{

//Khai báo số lượng Row trong 1 trang

int pageSize = 5;

//Text hiển thị trong input sau khi tìm kiếm

ViewData["CurrentFilter"] = searchString;

// Lưu Column được sắp xếp

ViewData["SortColumn"] = sortOrder;

//Value trả về trong button Sắp xếp

ViewData["IdSortParm"] = String.IsNullOrEmpty(sortOrder) ? "id\_desc" : "";

ViewData["TitleSortParm"] = sortOrder == "title" ? "title\_desc" : "title";

ViewData["PriceSortParm"] = sortOrder == "price" ? "price\_desc" : "price";

ViewData["RateSortParm"] = sortOrder == "rate" ? "rate\_desc" : "rate";

// Lấy danh sách Product sau khi phân trang

List<ProductAdmin> products = new List<ProductAdmin>();

products = productDAL.getProduct\_Pagination(page, pageSize, searchString.Trim(), sortOrder );

// Lấy số lượng Row sau khi truy vấn có thêm điều kiện (tìm kiếm)

int numRows = productDAL.getCountRow\_Pagination(page, pageSize, searchString.Trim());

//Tính số lượng trang sẽ có (làm tròn lên, vd: 4.3 -> 5)

double pageCount = (double)numRows / pageSize;

int maxPage = (int)Math.Ceiling(pageCount);

//Tạo model để hiển thị

ProductAdminModel model = new ProductAdminModel();

model.ProductAdmins = products;

model.CurrentPageIndex = page;

model.PageCount = maxPage;

return View(model);

}

// GET: ProductAdminController/Details/5

public ActionResult Details(int id)

{

...

}

...

}

}

Hiển thị button sắp xếp theo cột

**Areas/Admin/Views/ProductAdmin/Index.cshtml**

@model ShopManager.Areas.Admin.Models.ProductAdminModel

@{

ViewData["Title"] = "Index";

}

<h1>Index</h1>

<p>

<**a** **asp-action**="Create" class="btn btn-primary">Create New</**a**>

</p>

<**form** **asp-action**="index" method="get">

<div class="form-actions no-color">

<p>

Find by name: <input type="text" name="SearchString" value="@ViewData["CurrentFilter"]" />

<input type="submit" value="Search" class="btn btn-primary" /> |

<**a** **asp-action**="index" class="btn btn-outline-dark">Back to Full List</**a**>

</p>

</div>

</**form**>

<table class="table">

<thead>

<tr>

<th style="width:50px">

<**a** **asp-controller**="ProductAdmin" **asp-action**="Index"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

**asp-route-sortOrder**="@ViewData["IdSortParm"]">

Id

@if (ViewData["SortColumn"]!.ToString() == "")

{

<span>&darr;</span>

// mũi tên đi xuống

}

@if (ViewData["SortColumn"]!.ToString() == "id\_desc")

{

<span>&uarr;</span>

// mũi tên đi lên

}

</**a**>

</th>

<th>

<**a** **asp-controller**="ProductAdmin" **asp-action**="Index"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

**asp-route-sortOrder**="@ViewData["TitleSortParm"]">

Title

@if (ViewData["SortColumn"]!.ToString() == "title")

{

<span>&uarr;</span>

}

@if (ViewData["SortColumn"]!.ToString() == "title\_desc")

{

<span>&darr;</span>

}

</**a**>

</th>

<th>

Content

</th>

<th>

Image

</th>

<th>

<**a** **asp-controller**="ProductAdmin" **asp-action**="Index"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

**asp-route-sortOrder**="@ViewData["PriceSortParm"]">

Price

@if (ViewData["SortColumn"]!.ToString() == "price")

{

<span>&uarr;</span>

}

@if (ViewData["SortColumn"]!.ToString() == "price\_desc")

{

<span>&darr;</span>

}

</**a**>

</th>

<th style="width:70px">

<**a** **asp-controller**="ProductAdmin" **asp-action**="Index"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

**asp-route-sortOrder**="@ViewData["RateSortParm"]">

Rate

@if (ViewData["SortColumn"]!.ToString() == "rate")

{

<span>&uarr;</span>

}

@if (ViewData["SortColumn"]!.ToString() == "rate\_desc")

{

<span>&darr;</span>

}

</**a**>

</th>

<th>

Create At

</th>

<th>

Update At

</th>

<th>

CategoryId

</th>

<th>

CategoryTitle

</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach (var item in Model.ProductAdmins)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.Id)

</td>

<td>

@Html.DisplayFor(modelItem => item.Title)

</td>

<td>

<span style="line-height:1.5rem; height:4.5rem; display: -webkit-box; -webkit-line-clamp: 3; -webkit-box-orient: vertical; overflow: hidden;">

@Html.DisplayFor(modelItem => item.Content)

</span>

</td>

<td>

<**img** src="~/images/sanpham/@item.Img" class="img-fluid " style="width:100px; height: 60px; object-fit:contain" />

</td>

<td>

@item.Price.ToString("#,##0 VND")

</td>

<td>

@Html.DisplayFor(modelItem => item.Rate)

</td>

<td>

@Html.DisplayFor(modelItem => item.CreateAt)

</td>

<td>

@Html.DisplayFor(modelItem => item.UpdateAt)

</td>

<td>

@Html.DisplayFor(modelItem => item.CategoryId)

</td>

<td>

@Html.DisplayFor(modelItem => item.CategoryTitle)

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id=item.Id }) |

@Html.ActionLink("Details", "Details", new { id=item.Id }) |

@Html.ActionLink("Delete", "Delete", new { id=item.Id })

</td>

</tr>

}

</tbody>

</table>

<br />

<!-- Pagination Start-->

<div class="col-12">

<div class="pagination d-flex justify-content-center mt-5">

<!--Button Previous-->

@if (Model.CurrentPageIndex > 1)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@(Model.CurrentPageIndex -1)"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

**asp-route-sortOrder**="@ViewData["SortColumn"]"

class="btn btn-outline-dark mx-2">

&laquo;

</**a**>

}

@for (int i = 1; i <= Model.PageCount; i++)

{

@if (i != Model.CurrentPageIndex)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@i"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

**asp-route-sortOrder**="@ViewData["SortColumn"]"

class="btn btn-outline-dark mx-2">

@i

</**a**>

}

else

{

//Current Page

<a class="btn btn btn-outline-dark active mx-2">@i</a>

}

}

<!-- Button Next -->

@if (Model.PageCount != Model.CurrentPageIndex)

{

<**a** **asp-controller**="ProductAdmin" **asp-action**="index"

**asp-route-page**="@(Model.CurrentPageIndex + 1)"

**asp-route-searchString**="@ViewData["CurrentFilter"]"

**asp-route-sortOrder**="@ViewData["SortColumn"]"

class="btn btn-outline-dark mx-2">

&raquo;

</**a**>

}

</div>

</div>

<!-- Pagination End-->

### Phân trang Product trong trang cửa hàng

/Product/Index  
Tạo Model mới

**Models/Product.cs**

using System.ComponentModel.DataAnnotations;

namespace ShopManager.Models

{

public class Product

{

[Key]

public int Id { get; set; }

public string Title { get; set; }

public string Content { get; set; }

public string Img { get; set; }

public int Price { get; set; }

public double Rate { get; set; }

public DateTime CreateAt { get; set; }

public DateTime UpdateAt { get; set; }

public int CategoryId { get; set; }

public string CategoryTitle { get; set; }

}

public class ProductPagination

{

public List<Product> Products { get; set; }

public int CurrentPageIndex { get; set; }

public int PageCount { get; set; }

}

}

**DAL/ProductDAL.cs**

using Microsoft.Data.SqlClient;

using Microsoft.IdentityModel.Tokens;

using ShopManager.Areas.Admin.Models;

using ShopManager.Database;

using ShopManager.Models;

namespace ShopManager.DAL

{

public class ProductDAL

{

DBConnect connect = new DBConnect();

public List<Product> GetListProduct(int? idCategory)

{

...

}

public List<Product> GetProducts\_Pagination(int? idCategory, int pageIndex, int pageSize)

{

connect.openConnection();

List<Product> list = new List<Product>();

//Điều kiện Category

string categoryCondition = "";

if (idCategory.HasValue)

{

categoryCondition = @" WHERE a.categoryId = " + idCategory;

}

using (SqlCommand command = new SqlCommand())

{

command.Connection = connect.getConnecttion();

command.CommandType = System.Data.CommandType.Text;

string query = @"SELECT \* FROM (

SELECT ROW\_NUMBER() OVER( ORDER BY a.id ) AS RowNumber,

a.id as ProductId, a.title as ProductTitle,

a.content as ProductContent, a.img as ProductImg, a.price as ProductPrice,

a.rate as ProductRate, a.createAt as ProductCreateAt, a.updateAt as ProductUpdateAt,

b.id as CategoryId, b.title as CategoryTitle

FROM product a join category b on a.categoryId = b.Id "

+ categoryCondition + @"

) TableResult

WHERE TableResult.RowNumber BETWEEN( @PageIndex -1) \* @PageSize + 1

AND @PageIndex \* @PageSize

";

command.CommandText = query;

command.Parameters.AddWithValue("@PageIndex", pageIndex);

command.Parameters.AddWithValue("@PageSize", pageSize);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

Product product = new Product()

{

Id = Convert.ToInt32(reader["ProductId"]),

Title = reader["ProductTitle"].ToString() ?? "",

Content = reader["ProductContent"].ToString() ?? "",

Img = reader["ProductImg"].ToString() ?? "",

Price = Convert.ToInt32(reader["ProductPrice"]),

Rate = Convert.ToDouble(reader["ProductRate"].ToString()),

CreateAt = DateTime.Parse(reader["ProductCreateAt"]?.ToString()),

UpdateAt = DateTime.Parse(reader["ProductUpdateAt"]?.ToString()),

CategoryId = Convert.ToInt32(reader["CategoryId"]),

CategoryTitle = reader["CategoryTitle"].ToString(),

};

list.Add(product);

}

}

connect.closeConnection();

return list;

}

...

}

}

**Controllers/ProductController.cs**

using Microsoft.AspNetCore.Mvc;

using ShopManager.DAL;

using ShopManager.Models;

namespace ShopManager.Controllers

{

public class ProductController : Controller

{

ProductDAL productDAL = new ProductDAL();

//View List Product

public IActionResult **Index**(int? idCategory, int page = 1)

{

//Khai báo số lượng Row trong 1 trang

int pageSize = 5;

// Lưu Id Category

ViewData["IdCategory"] = idCategory;

// Lấy danh sách Product sau khi phân trang

List<Product> products = new List<Product>();

products = productDAL.GetProducts\_Pagination(idCategory, page, pageSize);

//Lấy tổng số lượng Product

int rowCount = productDAL.GetListProduct(idCategory).Count();

//Tính số lượng trang

double pageCount = (double)rowCount / pageSize;

int maxPage = (int)Math.Ceiling(pageCount);

//Tạo model để hiển thị

ProductPagination model = new ProductPagination();

model.Products= products;

model.CurrentPageIndex = page;

model.PageCount = maxPage;

return View(model);

}

//View Detail

public IActionResult Detail(int id) {...}

}

}

**Views/Product/Index.cshtml**

~~@\* @model IEnumerable<ShopManager.Models.Product> \*@~~

@model ShopManager.Models.ProductPagination

@{

ViewData["Title"] = "Index";

}

<!-- Single Page Header start -->

...

<!-- Single Page Header End -->

<!-- Fruits Shop Start-->

<div class="container-fluid fruite py-5">

<div class="container py-5">

<h1 class="mb-4">Fresh fruits shop</h1>

<div class="row g-4">

<div class="col-lg-12">

<div class="row g-4">

<div class="col-xl-3">

<div class="input-group w-100 mx-auto d-flex">

<input type="search"

class="form-control p-3"

placeholder="keywords"

aria-describedby="search-icon-1" />

<span id="search-icon-1" class="input-group-text p-3">

<i class="fa fa-search"></i>

</span>

</div>

</div>

<div class="col-6"></div>

<div class="col-xl-3">

<div class="bg-light ps-3 py-3 rounded d-flex justify-content-between mb-4">

<label for="fruits">Sắp xếp theo:</label>

<select id="fruit"

name="fruitlist"

class="border-0 form-select-sm bg-light me-3"

form="fruitform">

<**option**>

Tất cả

</**option**>

<**option**>

Giá cao đến thấp

</**option**>

<**option**>

Giá thấp đến cao

</**option**>

<**option**>

Đánh giá

</**option**>

</select>

</div>

</div>

</div>

<div class="row g-4">

<!-- Left Panel Start-->

<div class="col-lg-3">

<div class="row g-4">

<!-- Categories Start -->

<div class="col-lg-12">

@await Component.InvokeAsync("MenuCategory")

</div>

<!-- Categories End -->

<!-- Featured products Start -->

<div class="col-lg-12">

@await Component.InvokeAsync("FeaturedProducts",new { limit= 5})

</div>

<!-- Featured products End -->

</div>

</div>

<!-- Left Panel End -->

<!-- Right Panel Start-->

<div class="col-lg-9">

<div class="row g-4 justify-content-center">

@foreach (var item in Model.Products)

{

<!-- Item 1 Start -->

...

<!-- Item 1 End -->

}

<!-- Pagination Start-->

<div class="col-12">

<div class="pagination d-flex justify-content-center mt-5">

@if (Model.CurrentPageIndex > 1)

{

<**a** **asp-controller**="Product" **asp-action**="Index"

**asp-route-idCategory**="@ViewData["IdCategory"]"

**asp-route-page**="@(Model.CurrentPageIndex - 1)"

class="rounded">

&laquo;

</**a**>

}

@for (int i = 1; i <= Model.PageCount; i++)

{

if (i == Model.CurrentPageIndex)

{

<a href="#" class="active rounded">@i</a>

}

else

{

<**a** **asp-controller**="Product" **asp-action**="Index"

**asp-route-idCategory**="@ViewData["IdCategory"]"

**asp-route-page**="@i" class="rounded">

@i

</**a**>

}

}

@if (Model.CurrentPageIndex != Model.PageCount)

{

<**a** **asp-controller**="Product" **asp-action**="Index"

**asp-route-idCategory**="@ViewData["IdCategory"]"

**asp-route-page**="@(Model.CurrentPageIndex + 1)"

class="rounded">

&raquo;

</**a**>

}

</div>

</div>

<!-- Pagination End-->

</div>

</div>

<!-- Right Panel End -->

</div>

</div>

</div>

</div>

</div>

<!-- Fruits Shop End -->

### Sắp xếp Product trang cửa hàng

**DAL/ProductDAL.cs**

using Microsoft.Data.SqlClient;

using Microsoft.IdentityModel.Tokens;

using ShopManager.Areas.Admin.Models;

using ShopManager.Database;

using ShopManager.Models;

namespace ShopManager.DAL

{

public class ProductDAL

{

DBConnect connect = new DBConnect();

public List<Product> GetListProduct(int? idCategory)

{

...

}

public List<Product> GetProducts\_Pagination(int? idCategory, int pageIndex, int pageSize, string sortOrder)

{

connect.openConnection();

List<Product> list = new List<Product>();

//Điều kiện Category

string categoryCondition = "";

if (idCategory.HasValue)

{

categoryCondition = @" WHERE a.categoryId = " + idCategory;

}

//Truy vấn Sắp xếp

string sortQuery = " ORDER BY a.id ";

if (!string.IsNullOrEmpty(sortOrder))

{

switch (sortOrder)

{

case "price":

sortQuery = " ORDER BY a.price ";

break;

case "price\_desc":

sortQuery = " ORDER BY a.price DESC ";

break;

case "rate\_desc":

sortQuery = " ORDER BY a.rate DESC ";

break;

default:

sortQuery = " ORDER BY a.id ";

break;

}

}

using (SqlCommand command = new SqlCommand())

{

command.Connection = connect.getConnecttion();

command.CommandType = System.Data.CommandType.Text;

string query = @"SELECT \* FROM (

SELECT ROW\_NUMBER() OVER ( " + sortQuery + @" ) AS RowNumber,

a.id as ProductId, a.title as ProductTitle,

a.content as ProductContent, a.img as ProductImg, a.price as ProductPrice,

a.rate as ProductRate, a.createAt as ProductCreateAt, a.updateAt as ProductUpdateAt,

b.id as CategoryId, b.title as CategoryTitle

FROM product a join category b on a.categoryId = b.Id "

+ categoryCondition + @"

) TableResult

WHERE TableResult.RowNumber BETWEEN( @PageIndex -1) \* @PageSize + 1

AND @PageIndex \* @PageSize

";

command.CommandText = query;

command.Parameters.AddWithValue("@PageIndex", pageIndex);

command.Parameters.AddWithValue("@PageSize", pageSize);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

Product product = new Product()

{

Id = Convert.ToInt32(reader["ProductId"]),

Title = reader["ProductTitle"].ToString() ?? "",

Content = reader["ProductContent"].ToString() ?? "",

Img = reader["ProductImg"].ToString() ?? "",

Price = Convert.ToInt32(reader["ProductPrice"]),

Rate = Convert.ToDouble(reader["ProductRate"].ToString()),

CreateAt = DateTime.Parse(reader["ProductCreateAt"]?.ToString()),

UpdateAt = DateTime.Parse(reader["ProductUpdateAt"]?.ToString()),

CategoryId = Convert.ToInt32(reader["CategoryId"]),

CategoryTitle = reader["CategoryTitle"].ToString(),

};

list.Add(product);

}

}

connect.closeConnection();

return list;

}

....

}

}

**Controllers/ProductController.cs**

using Microsoft.AspNetCore.Mvc;

using ShopManager.DAL;

using ShopManager.Models;

namespace ShopManager.Controllers

{

public class ProductController : Controller

{

ProductDAL productDAL = new ProductDAL();

//View List Product

public IActionResult **Index**(int? idCategory, int page = 1, string sortOrder = "")

{

//Khai báo số lượng Row trong 1 trang

int pageSize = 5;

// Lưu Id Category

ViewData["IdCategory"] = idCategory;

// Lưu Cột được sắp xếp

ViewData["SortColumn"] = sortOrder;

// Lấy danh sách Product sau khi phân trang

List<Product> products = new List<Product>();

products = productDAL.GetProducts\_Pagination(idCategory, page, pageSize , sortOrder);

//Lấy tổng số lượng Product

int rowCount = productDAL.GetListProduct(idCategory).Count();

//Tính số lượng trang

double pageCount = (double)rowCount / pageSize;

int maxPage = (int)Math.Ceiling(pageCount);

//Tạo model để hiển thị

ProductPagination model = new ProductPagination();

model.Products= products;

model.CurrentPageIndex = page;

model.PageCount = maxPage;

return View(model);

}

//View Detail

public IActionResult Detail(int id){...}

}

}

**Views/Product/Index.cshtml**

@model ShopManager.Models.ProductPagination

@{

ViewData["Title"] = "Index";

}

<!-- Single Page Header start -->

...

<!-- Single Page Header End -->

<!-- Fruits Shop Start-->

<div class="container-fluid fruite py-5">

<div class="container py-5">

<h1 class="mb-4">Fresh fruits shop</h1>

<div class="row g-4">

<div class="col-lg-12">

<div class="row g-4">

...

<div class="col-6"></div>

<div class="col-xl-3">

<**form** **asp-controller**="Product" **asp-action**="Index" **asp-route-idCategory**="@ViewData["IdCategory"]">

<div class="bg-light ps-3 py-3 rounded d-flex justify-content-between mb-4">

<label for="fruits">Sắp xếp theo:</label>

<select id="dropdown\_shop\_id"

name="fruitlist"

class="border-0 form-select-sm bg-light me-3"

form="fruitform"

onchange="changeDropDown()" >

<**option** **value**="" >

Tất cả

</**option**>

<**option** **value**="price\_desc" >

Giá cao đến thấp

</**option**>

<**option** **value**="price">

Giá thấp đến cao

</**option**>

<**option** **value**="rate\_desc">

Đánh giá

</**option**>

</select>

<!-- Hidden Start-->

<input type="hidden" id="sort\_value\_id" name="sortOrder" />

<input type="hidden" id="sort\_column\_id" value="@ViewData["SortColumn"]" />

<input type="hidden" name="idCategory" value="@ViewData["IdCategory"]" />

<!-- Hidden End-->

</div>

</**form**>

</div>

</div>

<div class="row g-4">

<!-- Left Panel Start-->

<div class="col-lg-3">

<div class="row g-4">

<!-- Categories Start -->

<div class="col-lg-12">

@await Component.InvokeAsync("MenuCategory")

</div>

<!-- Categories End -->

<!-- Featured products Start -->

<div class="col-lg-12">

@await Component.InvokeAsync("FeaturedProducts",new { limit= 5})

</div>

<!-- Featured products End -->

</div>

</div>

<!-- Left Panel End -->

<!-- Right Panel Start-->

<div class="col-lg-9">

<div class="row g-4 justify-content-center">

@foreach (var item in Model.Products)

{

<!-- Item 1 Start -->

...

<!-- Item 1 End -->

}

<!-- Pagination Start-->

<div class="col-12">

<div class="pagination d-flex justify-content-center mt-5">

@if (Model.CurrentPageIndex > 1)

{

<**a** **asp-controller**="Product" **asp-action**="Index"

**asp-route-sortOrder**="@ViewData["SortColumn"]"

**asp-route-idCategory**="@ViewData["IdCategory"]"

**asp-route-page**="@(Model.CurrentPageIndex - 1)"

class="rounded">&laquo;</**a**>

}

@for (int i = 1; i <= Model.PageCount; i++)

{

if (i == Model.CurrentPageIndex)

{

<a href="#" class="active rounded">@i</a>

}

else

{

<**a** **asp-controller**="Product" **asp-action**="Index"

**asp-route-sortOrder**="@ViewData["SortColumn"]"

**asp-route-idCategory**="@ViewData["IdCategory"]"

**asp-route-page**="@i" class="rounded">@i</**a**>

}

}

@if (Model.CurrentPageIndex != Model.PageCount)

{

<**a** **asp-controller**="Product" **asp-action**="Index"

**asp-route-sortOrder**="@ViewData["SortColumn"]"

**asp-route-idCategory**="@ViewData["IdCategory"]"

**asp-route-page**="@(Model.CurrentPageIndex + 1)" class="rounded">&raquo;</**a**>

}

</div>

</div>

<!-- Pagination End-->

</div>

</div>

<!-- Right Panel End -->

</div>

</div>

</div>

</div>

<!-- SCRIPT Start -->

<script>

//----------------- Bắt sự kiện thay đổi giá trị ở ô dropDown ----------

function changeDropDown() {

valueDropdown = document.getElementById("dropdown\_shop\_id").value;

document.getElementById("sort\_value\_id").value = valueDropdown;

document.forms[0].submit();

}

// -------- set giá trị đã chọn cho Dropdown -------------------------

function selectElement(id, valueToSelect) {

let element = document.getElementById(id);

element.value = valueToSelect;

}

const sortColumnValue = document.getElementById("sort\_column\_id").value

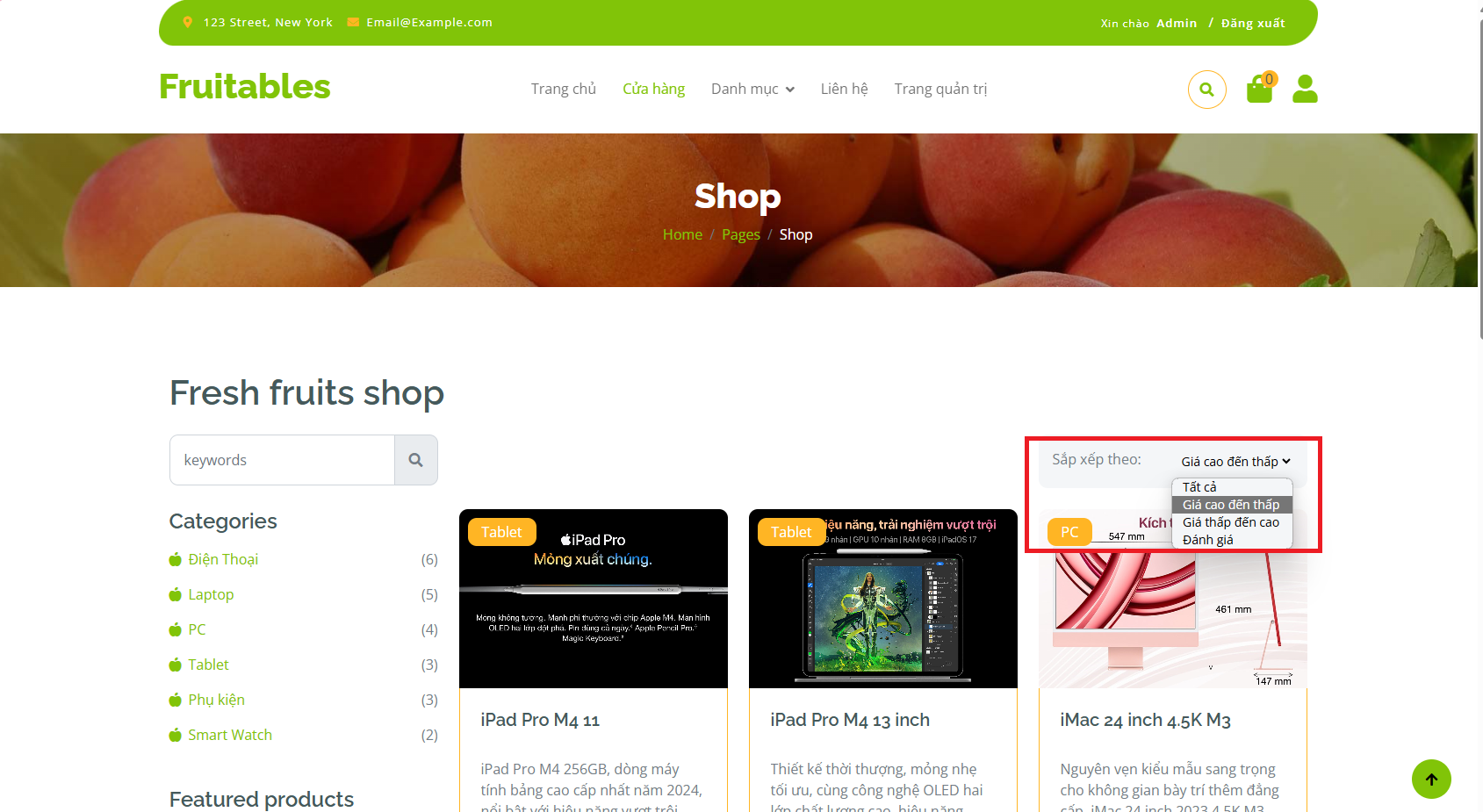
selectElement('dropdown\_shop\_id', sortColumnValue);

</script>

<!-- SCRIPT End -->

</div>

<!-- Fruits Shop End -->



## AJAX

**AJAX** là chữ viết tắt của cụm từ **Asynchronous Javascript and XML**. AJAX là phương thức trao đổi dữ liệu với máy chủ và cập nhật một hay nhiều phần của trang web, hoàn toàn không reload lại toàn bộ trang.

Ajax được viết bằng Javascript chạy trên client, tức là mỗi browser sẽ chạy độc lập hoàn toàn không ảnh hưởng lẫn nhau. Về mặt kỹ thuật, nó đề cập đến việc sử dụng các đối tượng XmlHttpRequest để tương tác với một máy chủ web thông qua Javascript.

**Cấu trúc một câu lệnh Ajax gọi đến Action trong Controller**

<script>

$.ajax({

url: '@Url.Action("Action","Controller")',

method: "GET",

data: { id: ProductId },

success: function (data) {

//call is successfully completed and we got result in data

},

error: function (xhr, ajaxOptions, thrownError) {

//some errror, some show err msg to user and log the error

alert(xhr.responseText);

}

});

</script>

**Giải thích:** Ajax được đặt trong thẻ <script> ... </script>

**Trong đó:**

url: '@Url.Action("Action","Controller")'

là đường dẫn trỏ tới Action muốn gọi

method: "GET",

là phương thức muốn gọi: GET, POST, PUT, DELETE, …

data: { id: ProductId },

là tham số muốn truyền vào Action đó

success: function (data) {

//call is successfully completed and we got result in data

},

Sự kiện trả về sau khi xử lý thành công

error: function (xhr, ajaxOptions, thrownError) {

//some errror, some show err msg to user and log the error

alert(xhr.responseText);

}

Sự kiện trả về khi xử lý Action thất bại

### Ajax xóa sản phẩm trong giỏ hàng

**Views/Cart/Index.cshtml**

@model IEnumerable<ShopManager.Models.CartItem>

<!-- Single Page Header start -->

<div class="container-fluid page-header py-5">

<h1 class="text-center text-white display-6">Cart</h1>

<ol class="breadcrumb justify-content-center mb-0">

<li class="breadcrumb-item"><a href="#">Home</a></li>

<li class="breadcrumb-item"><a href="#">Pages</a></li>

<li class="breadcrumb-item active text-white">Cart</li>

</ol>

</div>

<!-- Single Page Header End -->

<!-- Cart Page Start -->

<div class="container-fluid py-5">

<div class="container py-5">

<div class="table-responsive">

<table class="table">

<thead>

<tr>

<th scope="col">Products</th>

<th scope="col">Name</th>

<th scope="col">Price</th>

<th scope="col">Quantity</th>

<th scope="col">Total</th>

<th scope="col">Handle</th>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr id="idProduct-@item.IdProduct">

<th scope="row">

<div class="d-flex align-items-center">

<**img** src="~/images/SanPham/@item.Img" class="img-fluid me-5 rounded-circle" style="width: 80px; height: 80px;" alt="@item.Img">

</div>

</th>

<td>

<p class="mb-0 mt-4">@item.Name</p>

</td>

<td>

<p class="mb-0 mt-4">@item.Price.ToString("#,##0 VND")</p>

</td>

<td>

<div class="input-group quantity mt-4" style="width: 100px;">

<div class="input-group-btn">

<button class="btn btn-sm btn-minus rounded-circle bg-light border">

<i class="fa fa-minus"></i>

</button>

</div>

<input type="text"

id="quantityProduct"

class="form-control form-control-sm text-center border-0"

value="@item.Quantity">

<div class="input-group-btn">

<button class="btn btn-sm btn-plus rounded-circle bg-light border">

<i class="fa fa-plus"></i>

</button>

</div>

</div>

</td>

<td>

<p class="mb-0 mt-4">@item.Total.ToString("#,##0 VND")</p>

</td>

<td>

<button onclick="clickDeleteProduct(@item.IdProduct)"

class="btn btn-md rounded-circle bg-light border mt-4">

<i class="fa fa-times text-danger"></i>

</button>

</td>

</tr>

}

</tbody>

</table>

</div>

<div class="row g-4 justify-content-end">

<div class="col-8"></div>

<div class="col-sm-8 col-md-7 col-lg-6 col-xl-4">

<div class="bg-light rounded">

<div class="py-4 mb-4 border-top border-bottom d-flex justify-content-between">

<h5 class="mb-0 ps-4 me-4">Tổng tiền</h5>

<p class="mb-0 pe-4">@Model.Sum(p => p.Total).ToString("#,##0 VND")</p>

</div>

<button class="btn border-secondary rounded-pill px-4 py-3 text-primary text-uppercase mb-4 ms-4" type="button">Proceed Checkout</button>

</div>

</div>

</div>

</div>

</div>

<!-- Cart Page End -->

<script>

// ------------- Auto Close Alert Start----------------

const alert\_error = document.getElementById("alert\_message")

window.setTimeout(function () {

$(".alert").fadeTo(500, 0).slideUp(500, function () {

$(this).remove();

});

}, 2000);

// ------------- Auto Close Alert End----------------

//------------- Delete Product Start ---------------

function clickDeleteProduct(idProduct) {

$.ajax({

url: '@Url.Action("RemoveCart","Cart")',

data: { id: idProduct },

success: function (data) {

console.log("Delete Success ", idProduct);

//Không hiển thị hàng hóa đó nữa

const idName = "#idProduct-" + idProduct;

console.log(idName);

$(idName).remove();

//call is successfully completed and we got result in data

},

error: function (xhr, ajaxOptions, thrownError) {

//some errror, some show err msg to user and log the error

alert(xhr.responseText);

}

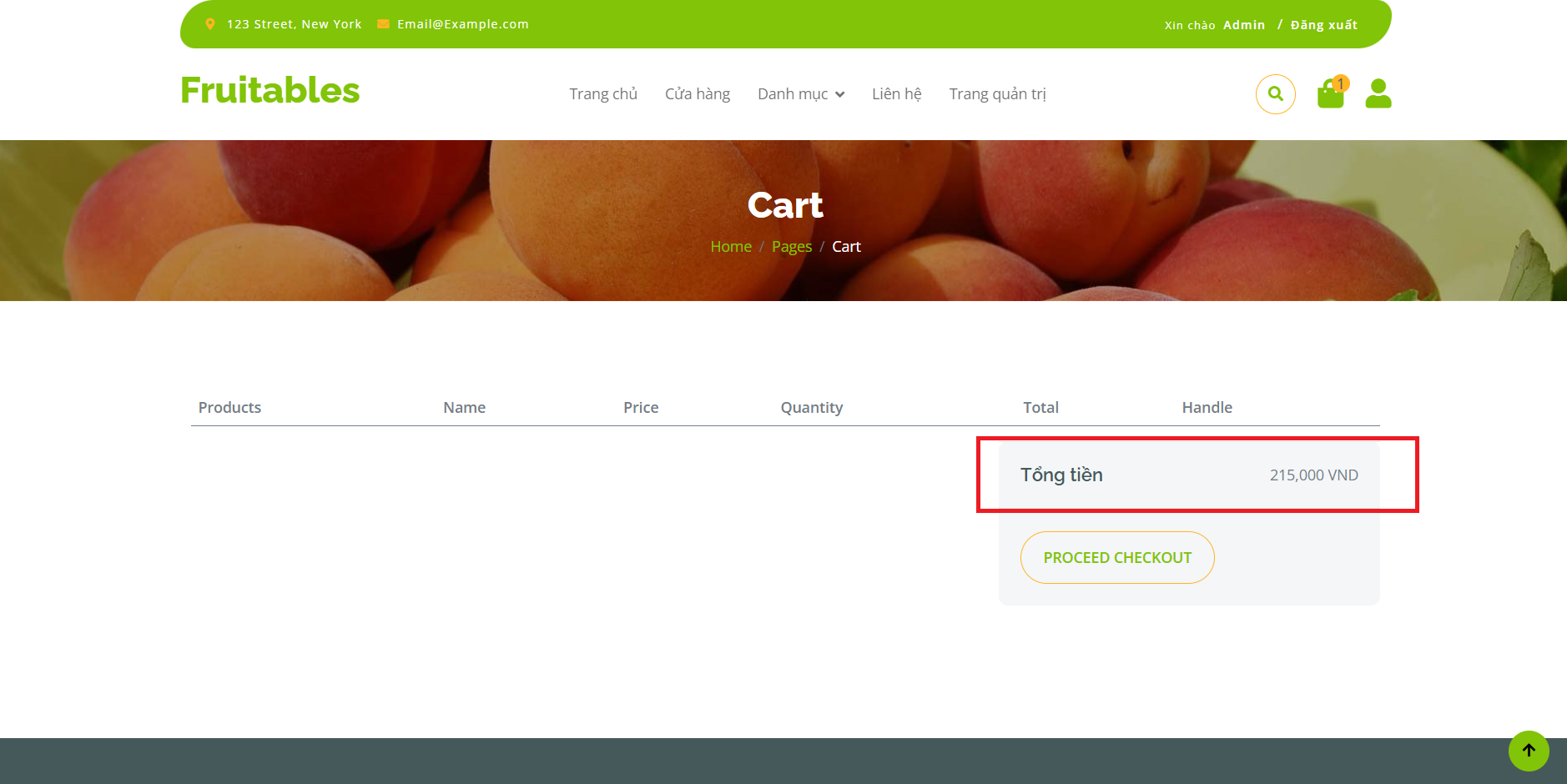
});

}

//------------- Delete Product End ---------------

</script>

Tuy nhiên sau khi xóa, Tổng tiền cần thanh toán vẫn không tự cập nhật



Tạo **Action** tính tổng tiền cần thanh toán trong **CartController**

**Controllers/CartController.cs**

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Cors.Infrastructure;

using Microsoft.AspNetCore.Mvc;

using ShopManager.DAL;

using ShopManager.Helper;

using ShopManager.Models;

using System.Runtime.CompilerServices;

using System.Security.Claims;

namespace ShopManager.Controllers

{

public class CartController : Controller

{

//Khai báo DAL

ProductDAL productDAL = new ProductDAL();

CustomerDAL customerDAL = new CustomerDAL();

CartDAL cartDAL = new CartDAL();

//Lấy thông tin giỏ hàng

public List<CartItem> Cart => HttpContext.Session.Get<List<CartItem>>(MyConst.CART\_KEY) ??

new List<CartItem>();

//Hiển thị Giỏ hàng

public IActionResult Index()

{

return View(Cart);

}

//Thêm vào giỏ hàng

public IActionResult AddToCart(int id, int quantity = 1)

{

...

}

public IActionResult RemoveCart(int id)

{

...

}

public IActionResult InputQuantityCart(int id, int quantity = 1)

{

... }

[Authorize] //Kiểm tra user đăng nhập ?

public IActionResult CheckOut()

{

... }

//Lấy tổng tiền thanh toán

public IActionResult GetTotalAmount()

{

// Giả sử bạn có phương thức GetTotal để tính tổng số tiền của giỏ hàng

int totalAmount = Cart.Sum(item => item.Total);

return Json(totalAmount);

}

}

}

Gọi Action **GetTotalAmount** với **Ajax**, sau đó **hiển thị giá trị nhận được**

**Views/Cart/Index.cshtml**

@model IEnumerable<ShopManager.Models.CartItem>

<!-- Single Page Header start -->

<div class="container-fluid page-header py-5">

<h1 class="text-center text-white display-6">Cart</h1>

<ol class="breadcrumb justify-content-center mb-0">

<li class="breadcrumb-item"><a href="#">Home</a></li>

<li class="breadcrumb-item"><a href="#">Pages</a></li>

<li class="breadcrumb-item active text-white">Cart</li>

</ol>

</div>

<!-- Single Page Header End -->

<!-- Cart Page Start -->

<div class="container-fluid py-5">

<div class="container py-5">

<div class="table-responsive">

<table class="table">

<thead>

<tr>

<th scope="col">Products</th>

<th scope="col">Name</th>

<th scope="col">Price</th>

<th scope="col">Quantity</th>

<th scope="col">Total</th>

<th scope="col">Handle</th>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr id="idProduct-@item.IdProduct">

<th scope="row">

<div class="d-flex align-items-center">

<**img** src="~/images/SanPham/@item.Img" class="img-fluid me-5 rounded-circle" style="width: 80px; height: 80px;" alt="@item.Img">

</div>

</th>

<td>

<p class="mb-0 mt-4">@item.Name</p>

</td>

<td>

<p class="mb-0 mt-4">@item.Price.ToString("#,##0 VND")</p>

</td>

<td>

<div class="input-group quantity mt-4" style="width: 100px;">

<div class="input-group-btn">

<button class="btn btn-sm btn-minus rounded-circle bg-light border">

<i class="fa fa-minus"></i>

</button>

</div>

<input type="text"

id="quantityProduct"

product-id-input="@item.IdProduct"

class="form-control form-control-sm text-center border-0"

value="@item.Quantity">

<div class="input-group-btn">

<button class="btn btn-sm btn-plus rounded-circle bg-light border">

<i class="fa fa-plus"></i>

</button>

</div>

</div>

</td>

<td>

<p class="mb-0 mt-4">@item.Total.ToString("#,##0 VND")</p>

</td>

<td>

<button onclick="clickDeleteProduct(@item.IdProduct)"

class="btn btn-md rounded-circle bg-light border mt-4">

<i class="fa fa-times text-danger"></i>

</button>

</td>

</tr>

}

</tbody>

</table>

</div>

<div class="row g-4 justify-content-end">

<div class="col-8"></div>

<div class="col-sm-8 col-md-7 col-lg-6 col-xl-4">

<div class="bg-light rounded">

<div id="totalCheckout" class="py-4 mb-4 border-top border-bottom d-flex justify-content-between">

<h5 class="mb-0 ps-4 me-4">Tổng tiền</h5>

<p class="mb-0 pe-4">@Model.Sum(p => p.Total).ToString("#,##0 VND")</p>

</div>

<button class="btn border-secondary rounded-pill px-4 py-3 text-primary text-uppercase mb-4 ms-4" type="button">Proceed Checkout</button>

</div>

</div>

</div>

</div>

</div>

<!-- Cart Page End -->

<script>

// ------------- Auto Close Alert Start----------------

const alert\_error = document.getElementById("alert\_message")

window.setTimeout(function () {

$(".alert").fadeTo(500, 0).slideUp(500, function () {

$(this).remove();

});

}, 2000);

// ------------- Auto Close Alert End----------------

//------------- Delete Product Start ---------------

function clickDeleteProduct(idProduct) {

$.ajax({

url: '@Url.Action("RemoveCart","Cart")',

data: { id: idProduct },

success: function (data) {

console.log("Delete Success ", idProduct);

//Không hiển thị hàng hóa đó nữa

const idName = "#idProduct-" + idProduct;

console.log(idName);

$(idName).remove();

//Load lại tổng tiền thanh toán

$.ajax({

url: '@Url.Action("GetTotalAmount","Cart")',

success: function (total) {

$("#totalCheckout p").text(total.toString().replace(/\B(?=(\d{3})+(?!\d))/g, ",") + " VND");

},

error: function (xhr, ajaxOptions, thrownError) {

alert("Error updating total amount.");

}

});

//call is successfully completed and we got result in data

},

error: function (xhr, ajaxOptions, thrownError) {

//some errror, some show err msg to user and log the error

alert(xhr.responseText);

}

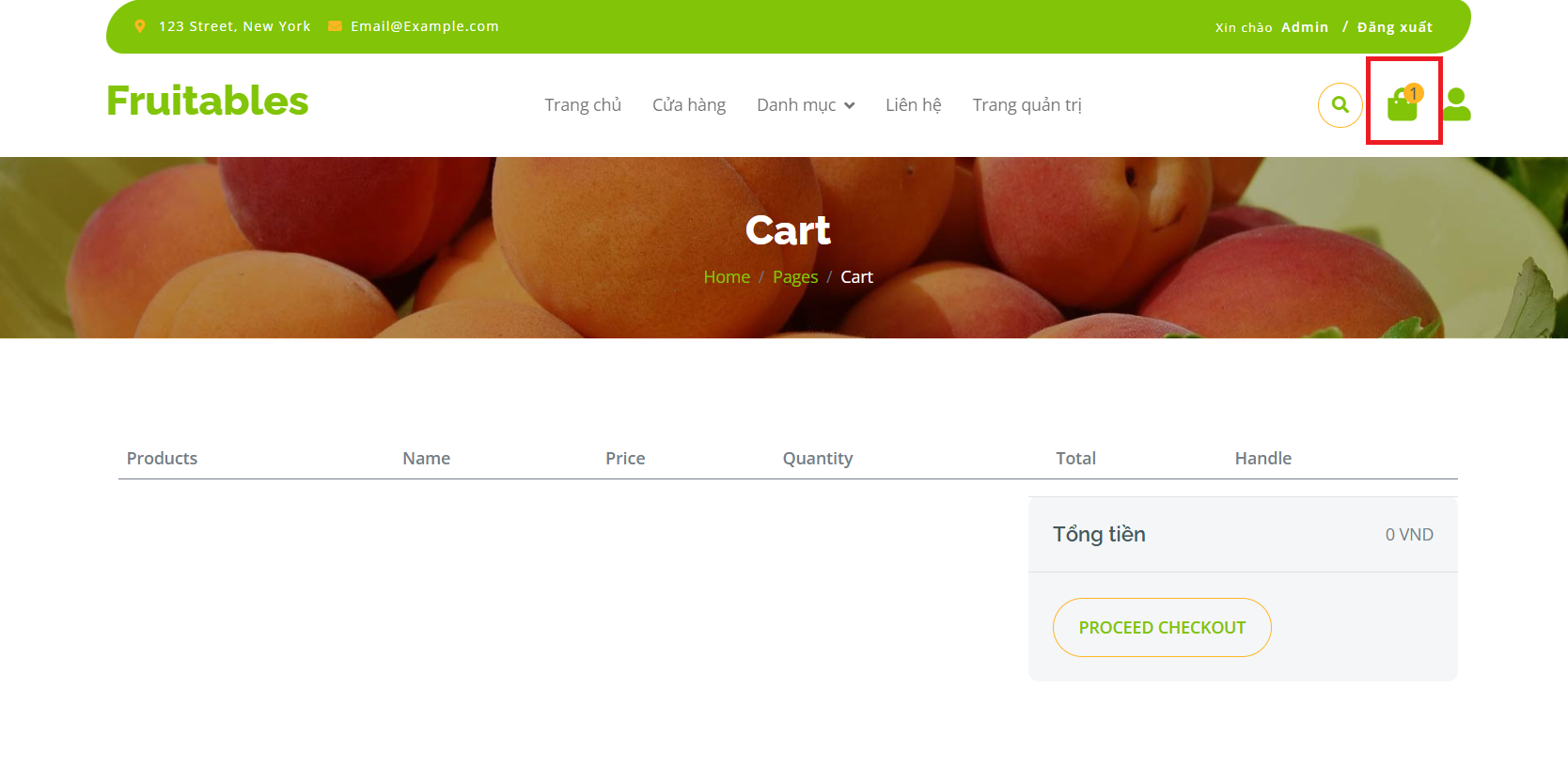
});

}

//------------- Delete Product End ---------------

</script>

View Component (CartViewComponent) hiển thị số lượng sản phầm vẫn chưa cập nhật lại



**Views/Shared/Components/MenuDynamic/MenuDynamic.cshtml**

Bọc **Cart ViewComponent** trong thẻ **<div>** với **id** là **cart\_view\_component**

@model IEnumerable<ShopManager.Models.NavbarItem>

<div class="container-fluid fixed-top">

<div class="container topbar bg-primary d-none d-lg-block">

...

</div>

<div class="container px-0">

<nav class="navbar navbar-light bg-white navbar-expand-xl">

<a href="index.html" class="navbar-brand"><h1 class="text-primary display-6">Fruitables</h1></a>

<button class="navbar-toggler py-2 px-3"

type="button"

data-bs-toggle="collapse"

data-bs-target="#navbarCollapse">

<span class="fa fa-bars text-primary"></span>

</button>

<div class="collapse navbar-collapse bg-white" id="navbarCollapse">

<div class="navbar-nav mx-auto">

@foreach (var item in Model)

{

...

}

</div>

<div class="d-flex m-3 me-0">

<button class="btn-search btn border border-secondary btn-md-square rounded-circle bg-white me-4"

data-bs-toggle="modal"

data-bs-target="#searchModal">

<i class="fas fa-search text-primary"></i>

</button>

<div id="cart\_view\_component">

@await Component.InvokeAsync("Cart")

</div>

<**a** **asp-area**="" **asp-controller**="Customer" **asp-action**="Profile" class="my-auto">

<i class="fas fa-user fa-2x"></i>

</**a**>

</div>

</div>

</nav>

</div>

</div>

Tạo **Action** trả về **CartViewComponent** (định dạng **html**)

**Controllers/CartController.cs**

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Cors.Infrastructure;

using Microsoft.AspNetCore.Mvc;

using ShopManager.DAL;

using ShopManager.Helper;

using ShopManager.Models;

using System.Runtime.CompilerServices;

using System.Security.Claims;

namespace ShopManager.Controllers

{

public class CartController : Controller

{

//Khai báo DAL

ProductDAL productDAL = new ProductDAL();

CustomerDAL customerDAL = new CustomerDAL();

CartDAL cartDAL = new CartDAL();

//Lấy thông tin giỏ hàng

public List<CartItem> Cart => HttpContext.Session.Get<List<CartItem>>(MyConst.CART\_KEY) ??

new List<CartItem>();

//Hiển thị Giỏ hàng

public IActionResult Index()

{

return View(Cart);

}

//Thêm vào giỏ hàng

public IActionResult AddToCart(int id, int quantity = 1) {...}

...

//Lấy tổng tiền thanh toán

public IActionResult GetTotalAmount()

{

// Giả sử bạn có phương thức GetTotal để tính tổng số tiền của giỏ hàng

int totalAmount = Cart.Sum(item => item.Total);

return Json(totalAmount);

}

//Refresh Cart View Component

public IActionResult RefreshCartViewComponent()

{

// Trả về ViewComponent("Tên ViewComponent muốn gọi", tham số truyền vào nếu có)

return ViewComponent("Cart");

}

}

}

Sử dụng **Ajax** để gọi Action vừa tạo, sau đó gán kết quả trả về cho **CartViewComponent**

**Cú pháp:**

$("#id thẻ muốn refresh html").html(dữ liệu mới dạng html);

**Views/Cart/Index.cshtml**

@model IEnumerable<ShopManager.Models.CartItem>

<!-- Single Page Header start -->

...

<!-- Single Page Header End -->

<!-- Cart Page Start -->

<div class="container-fluid py-5">

<div class="container py-5">

<div class="table-responsive">

<table class="table">

...

<tbody>

@foreach (var item in Model)

{

<tr id="idProduct-@item.IdProduct">

<th scope="row">

<div class="d-flex align-items-center">

<**img** src="~/images/SanPham/@item.Img" class="img-fluid me-5 rounded-circle" style="width: 80px; height: 80px;" alt="@item.Img">

</div>

</th>

<td>

<p class="mb-0 mt-4">@item.Name</p>

</td>

<td>

<p class="mb-0 mt-4">@item.Price.ToString("#,##0 VND")</p>

</td>

<td>

<div class="input-group quantity mt-4" style="width: 100px;">

<div class="input-group-btn">

<button

class="btn btn-sm btn-minus rounded-circle bg-light border">

<i class="fa fa-minus"></i>

</button>

</div>

<input type="text"

id="quantityProduct"

product-id-input="@item.IdProduct"

class="form-control form-control-sm text-center border-0"

value="@item.Quantity">

<div class="input-group-btn">

<button

class="btn btn-sm btn-plus rounded-circle bg-light border">

<i class="fa fa-plus"></i>

</button>

</div>

</div>

</td>

<td>

<p class="mb-0 mt-4">@item.Total.ToString("#,##0 VND")</p>

</td>

<td>

<button onclick="clickDeleteProduct(@item.IdProduct)"

class="btn btn-md rounded-circle bg-light border mt-4">

<i class="fa fa-times text-danger"></i>

</button>

</td>

</tr>

}

</tbody>

</table>

</div>

<div class="row g-4 justify-content-end">

<div class="col-8"></div>

<div class="col-sm-8 col-md-7 col-lg-6 col-xl-4">

<div class="bg-light rounded">

<div id="totalCheckout" class="py-4 mb-4 border-top border-bottom d-flex justify-content-between">

<h5 class="mb-0 ps-4 me-4">Tổng tiền</h5>

<p class="mb-0 pe-4">@Model.Sum(p => p.Total).ToString("#,##0 VND")</p>

</div>

<button class="btn border-secondary rounded-pill px-4 py-3 text-primary text-uppercase mb-4 ms-4" type="button">Proceed Checkout</button>

</div>

</div>

</div>

</div>

</div>

<!-- Cart Page End -->

<script>

// ------------- Auto Close Alert Start----------------

const alert\_error = document.getElementById("alert\_message")

window.setTimeout(function () {

$(".alert").fadeTo(500, 0).slideUp(500, function () {

$(this).remove();

});

}, 2000);

// ------------- Auto Close Alert End----------------

//------------- Delete Product Start ---------------

function clickDeleteProduct(idProduct) {

$.ajax({

url: '@Url.Action("RemoveCart","Cart")',

data: { id: idProduct },

success: function (data) {

console.log("Delete Success ", idProduct);

//Không hiển thị hàng hóa đó nữa

const idName = "#idProduct-" + idProduct;

console.log(idName);

$(idName).remove();

//Load lại tổng tiền thanh toán

$.ajax({

url: '@Url.Action("GetTotalAmount","Cart")',

success: function (total) {

$("#totalCheckout p").text(total.toString().replace(/\B(?=(\d{3})+(?!\d))/g, ",") + " VND");

},

error: function (xhr, ajaxOptions, thrownError) {

alert("Error updating total amount.");

}

});

//Load lại Cart ViewComponent

$.ajax({

url: '@Url.Action("RefreshCartViewComponent","Cart")',

success: function (data) {

console.log(data)

$("#cart\_view\_component").html(data);

},

error: function (xhr, ajaxOptions, thrownError) {

console.log(thrownError)

alert("Error updating Cart ViewComponent.");

}

});

//call is successfully completed and we got result in data

},

error: function (xhr, ajaxOptions, thrownError) {

//some errror, some show err msg to user and log the error

alert(xhr.responseText);

}

});

}

//------------- Delete Product End ---------------

<script>

### Ajax thêm, bớt số lượng sản phầm trong giỏ hàng

Tạo **Action** trả về tổng tiền theo **Product**

**Controllers/CartController.cs**

namespace ShopManager.Controllers

{

public class CartController : Controller

{

//Khai báo DAL

ProductDAL productDAL = new ProductDAL();

CustomerDAL customerDAL = new CustomerDAL();

CartDAL cartDAL = new CartDAL();

//Lấy thông tin giỏ hàng

public List<CartItem> Cart => HttpContext.Session.Get<List<CartItem>>(MyConst.CART\_KEY) ??

new List<CartItem>();

//Hiển thị Giỏ hàng

public IActionResult Index()

{

return View(Cart);

}

...

//Lấy tổng tiền thanh toán

public IActionResult GetTotalAmount()

{

// Giả sử bạn có phương thức GetTotal để tính tổng số tiền của giỏ hàng

int totalAmount = Cart.Sum(item => item.Total);

return Json(totalAmount);

}

//Refresh Cart View Component

public IActionResult RefreshCartViewComponent()

{

// Trả về ViewComponent("Tên ViewComponent muốn gọi", tham số truyền vào nếu có)

return ViewComponent("Cart");

}

//Lấy tổng tiền theo từng Product

public IActionResult GetTotalProduct(int idProduct)

{

// Giả sử bạn có phương thức GetTotal để tính tổng số tiền của giỏ hàng

var productFind = Cart.Find(item=>item.IdProduct == idProduct);

int totalAmount = 0;

if (productFind != null)

{

totalAmount = productFind.Total;

}

return Json(totalAmount);

}

}

}

**Views/Cart/Index.cshtml**

@model IEnumerable<ShopManager.Models.CartItem>

<!-- Single Page Header start -->

...

<!-- Single Page Header End -->

<!-- Cart Page Start -->

<div class="container-fluid py-5">

<div class="container py-5">

...

<div class="table-responsive">

<table class="table">

...

<tbody>

@foreach (var item in Model)

{

<tr id="idProduct-@item.IdProduct">

<th scope="row">

<div class="d-flex align-items-center">

<**img** src="~/images/SanPham/@item.Img" class="img-fluid me-5 rounded-circle" style="width: 80px; height: 80px;" alt="@item.Img">

</div>

</th>

<td>

<p class="mb-0 mt-4">@item.Name</p>

</td>

<td>

<p class="mb-0 mt-4">@item.Price.ToString("#,##0 VND")</p>

</td>

<td>

<div class="input-group quantity mt-4" style="width: 100px;">

<div class="input-group-btn">

<button product-id="@item.IdProduct"

onclick="clickBtnQuantity(false)"

class="btn btn-sm btn-minus rounded-circle bg-light border">

<i class="fa fa-minus"></i>

</button>

</div>

<input type="text"

id="quantityProduct"

product-id-input="@item.IdProduct"

class="form-control form-control-sm text-center border-0"

value="@item.Quantity">

<div class="input-group-btn">

<button product-id="@item.IdProduct"

onclick="clickBtnQuantity(true)"

class="btn btn-sm btn-plus rounded-circle bg-light border">

<i class="fa fa-plus"></i>

</button>

</div>

</div>

</td>

<td>

<p id="totalProduct" class="mb-0 mt-4">@item.Total.ToString("#,##0 VND")</p>

</td>

<td>

<button onclick="clickDeleteProduct(@item.IdProduct)"

class="btn btn-md rounded-circle bg-light border mt-4">

<i class="fa fa-times text-danger"></i>

</button>

</td>

</tr>

}

</tbody>

</table>

</div>

<div class="row g-4 justify-content-end">

<div class="col-8"></div>

<div class="col-sm-8 col-md-7 col-lg-6 col-xl-4">

<div class="bg-light rounded">

<div id="totalCheckout" class="py-4 mb-4 border-top border-bottom d-flex justify-content-between">

<h5 class="mb-0 ps-4 me-4">Tổng tiền</h5>

<p class="mb-0 pe-4">@Model.Sum(p => p.Total).ToString("#,##0 VND")</p>

</div>

<button class="btn border-secondary rounded-pill px-4 py-3 text-primary text-uppercase mb-4 ms-4" type="button">Proceed Checkout</button>

</div>

</div>

</div>

</div>

</div>

<!-- Cart Page End -->

<script>

// ------------- Auto Close Alert Start---------------- ...

// ------------- Auto Close Alert End----------------

//------------- Delete Product Start --------------- ...

//------------- Delete Product End ---------------

// ----------- Change Quantity With Two Buttons Start -----------------

function clickBtnQuantity(increment) {

// increment = true -> Nhấn button Tăng số lượng

// increment = false -> Nhấn button Giảm số lượng

// Lấy product-id từ nút được nhấn

var productId = event.target.getAttribute('product-id');

// Nếu nút không phải là thẻ con của nút (như <i> trong trường hợp này)

if (!productId) {

productId = event.target.parentElement.getAttribute('product-id');

}

// Ví dụ: Hiển thị productId và giá trị increment trong console

console.log('Product ID:', productId);

console.log('Increment:', increment);

// Ajax gọi action ChangeQuantityCart trong Cart Controller

$.ajax({

url: '@Url.Action("ChangeQuantityCart","Cart")',

data: { id: productId, isIncrement: increment, quantity: 1 },

success: function (data) {

console.log("ChangeQuantityCart Success ", productId, ' Increment: ', increment);

//Lấy giá trị trong ô Input theo ProductId

const queryString = '[product-id-input="' + productId + '"]';

const inputValue = document.querySelectorAll(queryString)[0].value;

// Nếu giá trị trong ô Input <= 0 thì remove hàng hóa đó

if (inputValue <= 0) {

const idName = "#idProduct-" + productId;

$(idName).remove();

} else {

//Load lại tổng tiền theo từng Product

$.ajax({

url: '@Url.Action("GetTotalProduct","Cart")',

data: { idProduct: productId },

success: function (total) {

$("#totalProduct").text(total.toString().replace(/\B(?=(\d{3})+(?!\d))/g, ",") + " VND");

},

error: function (xhr, ajaxOptions, thrownError) {

alert("Error updating total amount.");

}

});

}

//Load lại tổng tiền thanh toán

$.ajax({

url: '@Url.Action("GetTotalAmount","Cart")',

success: function (total) {

$("#totalCheckout p").text(total.toString().replace(/\B(?=(\d{3})+(?!\d))/g, ",") + " VND");

},

error: function (xhr, ajaxOptions, thrownError) {

alert("Error updating total amount.");

}

});

//Load lại Cart ViewComponent

$.ajax({

url: '@Url.Action("RefreshCartViewComponent","Cart")',

success: function (data) {

console.log(data)

$("#cart\_view\_component").html(data);

},

error: function (xhr, ajaxOptions, thrownError) {

console.log(thrownError)

alert("Error updating Cart ViewComponent.");

}

});

},

error: function (xhr, ajaxOptions, thrownError) {

//some errror, some show err msg to user and log the error

alert(xhr.responseText);

}

});

}// --------------- Change Quantity With Two Buttons End ---------------

</script>

### Ajax thêm sản phầm vào giỏ hàng

Sử dụng Action **RefreshCartViewComponent** trong **CartController.cs**

Thêm sự kiện gọi **Action** trên sau khi nhấn thêm vào giỏ hàng

**Views/Product/Index.cshtml**

@model ShopManager.Models.ProductPagination

@{

ViewData["Title"] = "Index";

}

<!-- Single Page Header start -->

<div class="container-fluid page-header py-5">

<h1 class="text-center text-white display-6">Shop</h1>

<ol class="breadcrumb justify-content-center mb-0">

<li class="breadcrumb-item"><a href="#">Home</a></li>

<li class="breadcrumb-item"><a href="#">Pages</a></li>

<li class="breadcrumb-item active text-white">Shop</li>

</ol>

</div>

<!-- Single Page Header End -->

<!-- Modal Start -->

<div class="modal fade" id="modalAddToCartSuccess" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="exampleModalLabel">Thông báo</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

Thêm vào giỏ hàng thành công

</div>

<div class="modal-footer">

</div>

</div>

</div>

</div>

<!-- Modal End -->

<!-- Fruits Shop Start-->

<div class="container-fluid fruite py-5">

<div class="container py-5">

<h1 class="mb-4">Fresh fruits shop</h1>

<div class="row g-4">

<div class="col-lg-12">

<div class="row g-4">

...

<div class="row g-4">

<!-- Left Panel Start-->

<div class="col-lg-3">

<div class="row g-4">

<!-- Categories Start -->

<div class="col-lg-12">

@await Component.InvokeAsync("MenuCategory")

</div>

<!-- Categories End -->

<!-- Featured products Start -->

<div class="col-lg-12">

@await Component.InvokeAsync("FeaturedProducts",new { limit= 5})

</div>

<!-- Featured products End -->

</div>

</div>

<!-- Left Panel End -->

<!-- Right Panel Start-->

<div class="col-lg-9">

<div class="row g-4 justify-content-center">

@foreach (var item in Model.Products)

{

<!-- Item 1 Start -->

<div class="col-md-6 col-lg-6 col-xl-4">

<div class="rounded position-relative fruite-item">

<div class="fruite-img">

<**a** **asp-area**="" **asp-controller**="Product" **asp-action**="Detail" **asp-route-id**="@item.Id">

<**img** src="~/Images/SanPham/@item.Img"

class="img-fluid w-100 rounded-top"

style="aspect-ratio:1.5; object-fit:cover"

alt="@item.Img" />

</**a**>

</div>

<div class="text-white bg-secondary px-3 py-1 rounded position-absolute"

style="top: 10px; left: 10px">

@item.CategoryTitle

</div>

<div class="p-4 border border-secondary border-top-0 rounded-bottom">

<h5 style="line-height:1.5rem; height:3rem; display: -webkit-box; -webkit-line-clamp: 2; -webkit-box-orient: vertical; overflow: hidden;">

@item.Title

</h5>

<p style="line-height:1.5rem; height:4.5rem; display: -webkit-box; -webkit-line-clamp: 3; -webkit-box-orient: vertical; overflow: hidden;">

@item.Content

</p>

<div class="d-flex justify-content-start flex-lg-wrap">

<p class="text-dark fs-6 fw-bold mb-0">@item.Price.ToString("#,##0 VND")</p>

</div>

<div class="d-flex justify-content-end flex-lg-wrap">

<button onclick="clickAddToCart(@item.Id)"

class="btn border border-secondary rounded-pill px-3 text-primary">

<i class="fa fa-shopping-bag me-2 text-primary"></i> Add to cart

</button>

</div>

</div>

</div>

</div>

<!-- Item 1 End -->

}

<!-- Pagination Start-->

...

<!-- Pagination End-->

</div>

</div>

<!-- Right Panel End -->

</div>

</div>

</div>

</div>

<!-- SCRIPT Start -->

<script>

// -------- set giá trị đã chọn cho Dropdown -------------------------

function selectElement(id, valueToSelect) {

let element = document.getElementById(id);

element.value = valueToSelect;

}

const sortColumnValue = document.getElementById("sort\_column\_id").value

selectElement('dropdown\_shop\_id', sortColumnValue);

//------------ Bắt sự kiện thay đổi giá trị ở ô dropDown ------------------

function changeDropDown() {

valueDropdown = document.getElementById("dropdown\_shop\_id").value;

document.getElementById("sort\_value\_id").value = valueDropdown;

document.forms[0].submit();

}

// --------------- AJAX thêm vào giỏ hàng -------------------

function clickAddToCart(idProduct) {

$.ajax({

url: '@Url.Action("AddToCart","Cart")',

data: { id: idProduct },

success: function (data) {

//Load lại Cart ViewComponent

$.ajax({

url: '@Url.Action("RefreshCartViewComponent","Cart")',

success: function (data) {

console.log(data)

$("#cart\_view\_component").html(data);

},

error: function (xhr, ajaxOptions, thrownError) {

console.log(thrownError)

alert("Error updating Cart ViewComponent.");

}

});

// // Hiển thị thông báo thêm thành công

$("#modalAddToCartSuccess").modal('show');

},

error: function (xhr, ajaxOptions, thrownError) {

//some errror, some show err msg to user and log the error

alert(xhr.responseText);

}

});

}

</script>

<!-- SCRIPT End -->

</div>

<!-- Fruits Shop End -->

**--- THE END ---**