* Model: dữ liệu ứng dụng, lưu giữ csdl
* View: html, hiển thị ra màn hình người dùng
* Controller: xử lý yêu cầu trên thanh địa chỉ
* Router: chọn controller đúng để xử lý yêu cầu.

Yêu cầu ở thanh địa chỉ sẽ được gửi về webserver, ở đó router sẽ xử lý để đưa đến controller tương ứng, controller sẽ trỏ đến action/method (phương thức) đúng với yêu cầu, xử lý và đưa kết quả vào view tương ứng, và phần view sẽ hiển thị ra màn hình người dùng.

Khi sử dụng 1 view trong 1 controller, asp.net sẽ tìm đến view đó trong folder có tên trùng với controller.

*http://localhost:xxx/controller/action (method)/Id (tham số)*

app.UseMvc(routes =>

{

routes.MapRoute(

name: "default",

template: "{controller=Home}/{action=Index}/{id?}");

});

Khi return view trong controller bị lỗi thì có nghĩa là chưa có view đó trong application. Cần phải tạo trong folder view

Mặc định nếu không có controller nào thì sẽ đưa lên HomeController. Và nếu không có action nào thì sẽ được đưa đến action index (nằm trong HomeController). Vì thế kết quả của các URL sau đều đến 1 trang như nhau.

<http://localhost:xxx>

<http://localhost:xxx>/home

<http://localhost:xxx>/home/index

truyền dữ liệu từ browse đến server bằng query string (khung url), form data (dữ liệu từ 1 form trong trang web) bằng phương thức get hoặc post. Dùng get khi muốn lấy dữ liệu của 1 tài nguyên xuống (tải hình ảnh, video trên mạng xuống), dùng post khi muốn tạo tài nguyên (vd tải hình ảnh, video lên mạng).

phuowng thuwsc http get post

controller truyền dữ liệu qua view bằng cách nàoTrong 1 solution thì sẽ có các project và DOL, project là các app hoặc web có thể chạy được còn DOL thì không.

Mọi thay đổi trong solution đều không ảnh hưởng đến file và folder thật, VD tạo 1 folder trong solution thì không có folder đó tồn tại trong file explorer mà folder đó chỉ tồn tại trong visual st.

Để tải DOL thì vào tools -> NuGet Package Manager

Mỗi controller sẽ cần 1 view tương ứng cùng tên thì mới chạy được.

Truyền dữ liệu từ controller đến view thông qua model

--------------/

Tạo model, controller, view

Xuất ra ten, age, gender cứng

Xuất ra bằng cách nhập dữ liệu từ browser

Xuất ra giới tính nam hoặc nữ bằng if else trong view

..... bằng ? trong view

Xuất ra male hoặc female bằng ? trong controller với viewData và viewBag

--------------------------/

//Trước mỗi controller đều phải có phần:

namespace WebGoldenSea.Controllers

{

    public class (tên controller)Controller : GsControllerBase<HomeController>

    {

        public (tên controller)Controller(GOLDEN\_SEAContext context) : base(context)

        {

        }

public IActionResult Index()

        {

//Cần thiết lập trước ở file Starup.cs (cuối solution);

public void ConfigureServices(IServiceCollection services)

{

............

............

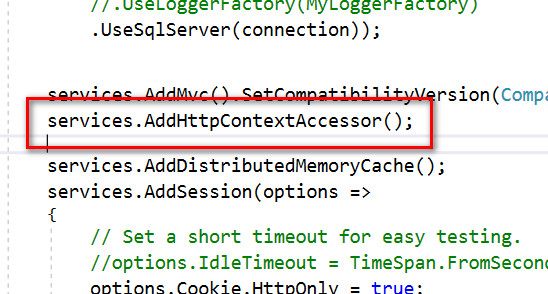
services.AddHttpContextAccessor();

............

............

}

//Hình thực tế:



*Tuần 2*

*Trang product: truyền vào ID của 1 sản phẩm để lấy những thông tin của sản phẩm đó trong CSDL. Viết 1 review và lưu thông tin (tên người dùng và nội dung review) vào csdl, đồng thời xuất thông tin đó ra màn hình sử dụng ajax.*

HomeController: Có thể tự tạo Controller trong thư mục Controllers để viết.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.EntityFrameworkCore;

using WebGoldenSea.EfModels;

using WebGoldenSea.Models;

namespace WebGoldenSea.Controllers

{

    public class ProductDetailController : GsControllerBase<HomeController>

    {

        public ProductDetailController(GOLDEN\_SEAContext context) : base(context)

        {

        }

public async Task<IActionResult> Product(string id)

        {

            ProductDetailViewModel model = new ProductDetailViewModel();

            model.title = "Leo Leo Quân";

            model.product = await \_context.WProduct.SingleOrDefaultAsync(o => o.Id == id);

// lấy một item

            model.listProductImage = await \_context.WProductImage.Where(u => u.FkWproduct == id).OrderBy(o => o.ShowIndex).ToListAsync();

//lấy list item, listProductImage là tên param tự đặt vì muốn lấy nhiều hình ảnh.

            model.listProductReview = await \_context.WReviewOfProduct.Where(u => u.FkWproduct == id && u.CheckReview == true).OrderBy(o => o.ShowIndex).ToListAsync();

//lấy list item và kiểm tra điều kiện

            return View(model);

--------------/

            // Get one

            var singleItem = await \_context.WProduct.SingleOrDefaultAsync(o => o.Id == id);

            //// Get List

            var listItem = await \_context.WProduct.Where(u => u.CreatedDate < DateTime.Now).OrderBy(o => o.Id).ToListAsync();

//lấy list item có điều kiện

            // Update

            if (singleItem != null)

            {

                singleItem.ProductCode = "NEW CODE";

                \_context.SaveChanges();

            }

------------------/

            // Insert

            WProduct newProduct = new WProduct();

            newProduct.Id = Guid.NewGuid().ToString();

            newProduct.FkWcategory = "FK\_WCategory";

            newProduct.ProductCode = "CODE001";

            newProduct.ProductName = "Máy tính xách tay";

            newProduct.ProductPrice = 0;

            newProduct.ProductPriceSale = 0;

            newProduct.ProductDesShor = "Mô tả ngắn sản phẩm";

            newProduct.ProductDes = "Mô tả đầy đủ về sản phẩm";

            newProduct.Productindex = 0;

            newProduct.ProductView = 0;

            newProduct.CreatedDate = DateTime.Now;

            newProduct.ModifiedDate = DateTime.Now;

            newProduct.CreatedBy = "Người tạo";

            newProduct.ModifiedBy = "Người sửa";

            \_context.WProduct.Add(newProduct);

            \_context.SaveChanges();

            // Delete

            \_context.WProduct.Remove(singleItem);

            \_context.SaveChanges();

            //var listItem = await \_context.RequestFindProduct.ToListAsync();

        }

//Lưu review vào csdl

        [HttpPost]

        public async Task<IActionResult> NewReview(string fk\_productid, string yourname, string yourcontent, int rating)

//tên các param cần truyền vào khi nhập review (họ tên, nội dung của review, đánh giá-rating).

        {

            try

            {

                WReviewOfProduct review = new WReviewOfProduct();

                review.Id = Guid.NewGuid().ToString();

//Guid.NewGuid() hàm tạo số ngẫu nhiên không trùng.

                review.FkWproduct = fk\_productid;

// FkWproduct là tên thuộc tính trong bảng WreviewOfProduct, fk\_productid là tên tự đặt dựa theo id của các input và textarea (ô nhập chữ, review) bên trang View.

                review.ReviewName = yourname; //tương tự trên

                review.ReviewContent = yourcontent; //tương tự

                review.Rating = rating;//

//từ rating trở xuống thì ko phải truyền param nữa nên gán giá trị luôn.

                review.ShowIndex = 0;

                review.CheckReview = false;

                review.DateReview = DateTime.Now;

                \_context.WReviewOfProduct.Add(review);

                await \_context.SaveChangesAsync();

                return Json(true);

            }

            catch (Exception ex)//dùng để bắt lỗi, kết hợp với ajax bên View

            {

                return Json(new { errorMessage = ex.ToString() });

            }

        }

------------------/

Model: gom các thành phần ở Controller để truyền sang View

using System;

using System.Collections.Generic;

using WebGoldenSea.EfModels;

namespace WebGoldenSea.Models

{

    public class ProductDetailViewModel

    {

        public WProduct product { get; set; }

        public List<WProductImage> listProductImage { get; set; }

        public List<WReviewOfProduct> listProductReview { get; set; }

        public string title { get; set; }

    }

}

//Wproduct, WProductImage, WReviewOfProduct là table trong csdl, dùng list<> khi lấy nhiều thuộc tính của bảng đó trong Controller.

--------------/

View:

@model WebGoldenSea.Models.ProductDetailViewModel

// Địa chỉ model sẽ sử dụng

@{

    Layout = "~/Views/Shared/\_Layout.cshtml";

    ViewBag.Title = Model.title;

}

: @model WebGoldenSea.Models.ProductDetailViewModel

Đếm số review về sản phẩm: @Model.listProductReview.Count.ToString()

Gọi param trong view: @Model.tênparamtạobênController.tênthuộctínhcầngọitrongtable

Foreach sản phẩm:

@foreach(var abc in Model.listProduct)

{

//code về sản phẩm

Gọi param trong foreach thì đều phải bắt đầu bằng @abc.tênparammuốngọi , trong đó abc là param tạo trong foreach.

}

View (ajax):

@section scripts {

    <script>

        $(document).ready(function () {

            var \_urlPostComment = '@Url.Action("NewReview", "Home")';

//Home là tên Controller, NewReview là action trong Home.

            var $fkProductID = $('#fk\_productId');

//$fkProductID là tên param tạo dựa trên id của các input và textarea (ô nhập chữ, review), #fk\_productId là id của ô input cần nhập.

            var $yourName = $('#reviewYourName');//tương tự

            var $yourContent = $('#reviewYourContent');//tương tự

            var $rating = 1;

            // Log

            console.log($yourName.val());

            console.log($yourContent.val());

            console.log($fkProductID.val());

            $("#button-review").click(function () {

                // Post Comment

                $.ajax({

                    url: \_urlPostComment,

                    type: 'POST',

                    data: {

                        fk\_productid: $fkProductID.val(),

                        yourname: $yourName.val(),

                        yourcontent: $yourContent.val(),

                        rating: $rating

                    },

                    dataType: 'json',

                    error: function (arg1, arg2, arg3) {

                        alert('Process error')

                    },

                    success: function (data) {

                        console.log(data);

                        if (data.errorMessage) {

                            alert('Lỗi khi lưu dữ liệu');

                        }

                        else {

                            alert('Đăng bình luận thành công')

                        }

                    },

                });

            });

        });

    </script>

--------------------------------------------/

*TRANG CATEGORY: lấy ra danh sách những sản phẩm có cùng mã loại, liên kết với trang Product để xem chi tiết sản phẩm.*

Model:

using System;

using System.Collections.Generic;

using WebGoldenSea.EfModels;

namespace WebGoldenSea.Models

{

    public class CategoryViewModel

    {

        public WCategory category { get; set; }

        public List<WProduct> listProduct { get; set; }

    }

}

CategoryController:

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using WebGoldenSea.EfModels;

using WebGoldenSea.Models;

namespace WebGoldenSea.Controllers

{

    public class CategoryController : GsControllerBase<CategoryController>

    {

        public CategoryController(GOLDEN\_SEAContext context) : base(context)

        {

        }

        public async Task<IActionResult> Index(string id)

        {

            CategoryViewModel model = new CategoryViewModel();

            model.category = await \_context.WCategory.SingleOrDefaultAsync(o => o.Id == id);

            model.listProduct = await \_context.WProduct.Where(u => u.FkWcategory == id).OrderBy(o => o.ProductName).ToListAsync();

            return View(model);

        }

        public async Task<IActionResult> GetCategory(string id, int recordLimit)

        {

            CategoryViewModel model = new CategoryViewModel();

            model.category = await \_context.WCategory.SingleOrDefaultAsync(o => o.Id == id);

            model.listProduct = await \_context.WProduct.Where(u => u.FkWcategory == id).OrderBy(o => o.ProductName).ToListAsync();

            return View(model);

        }

    }

}

View:

@model WebGoldenSea.Models.CategoryViewModel

@{

    Layout = "~/Views/Shared/\_Layout.cshtml";

    ViewBag.Title = Model.category.CategoryName;

}

Liên kết trang: href="~/Home/Product/@product.Id"

------------------------------------------------------------------/

*Trang Contact: nhập thông tin liên hệ vào ô, sau đó bấm nút để lưu những thông tin đó vào csdl, đồng thời gửi 1 mail đến địa chỉ email đã ghi vào ô.*

ContactController:

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Net;

using System.Net.Mail;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Options;

using WebGoldenSea.EfModels;

using WebGoldenSea.Models;

namespace WebGoldenSea.Controllers

{

    public class ContactController : GsControllerBase<ContactController>

    {

        private readonly EmailSettings \_emailSettings;

        public ContactController(GOLDEN\_SEAContext context, IOptions<EmailSettings> emailSettings) : base(context)

        {

            \_emailSettings = emailSettings.Value;

        }

        public IActionResult Index()

        {

//giúp lấy dữ liệu từ table Wsetting thông qua SettingHelper đổ vào param model, từ đó return model qua view (Contact) để hiển thị data.

            ContactViewModel model = new ContactViewModel();

            model.setting = await SettingHelper.ReadServerOptionAsync(\_context);

            return View(model);

        }

        [HttpPost]

        public async Task<IActionResult> NewContact(string yourname, string email, string enquirycontent)

//có 3 thông tin cần nhập vào trong 3 ô nên cần truyền 3 param

        {

            WContact contact = new WContact();

            contact.Id = Guid.NewGuid().ToString();//hàm tạo số ngẫu nhiên không trùng

            contact.YourName = yourname;

            contact.Email = email;

            contact.Enquiry = enquirycontent;

// YourName, Email, Enquiry là tên đặt, dựa theo yourname, email, enquirycontent là name của các ô input và area cần nhập dữ liệu ở trang View.

            contact.CreatedDate = DateTime.Now;

            contact.CreatedBy = "người tạo";

            contact.Tag = "lienhe";

            \_context.WContact.Add(contact);

            await \_context.SaveChangesAsync();

       ////// Phần Gửi email (1)

            string toEmail = contact.Email; //địa chỉ nhận mail

            string toSubject = "Test gửi mail"; //tiêu đề mail

            string toMessage = "<h1>Nội dung</h1>"; //nội dung mail

            try

            {

                // Credentials

                var credentials = new NetworkCredential(\_emailSettings.Sender, \_emailSettings.Password); //tự động lấy địa chỉ mail và password ở file appsettings.json (để đăng nhập).

                // Mail message

                var mail = new MailMessage()

                {

                    From = new MailAddress(\_emailSettings.Sender, \_emailSettings.SenderName), //địa chỉ gửi, lấy email và username của email.

                    Subject = toSubject,

                    Body = toMessage,

                    IsBodyHtml = true

                };

                mail.To.Add(new MailAddress(toEmail)); //gửi mail mới

                //mail.CC.Add(new MailAddress(toEmail));

                //mail.Attachments.Add() image

                // Smtp client

                var client = new SmtpClient()

                {

                    Port = \_emailSettings.MailPort,

                    DeliveryMethod = SmtpDeliveryMethod.Network,

                    UseDefaultCredentials = false,

                    Host = \_emailSettings.MailServer,

                    EnableSsl = \_emailSettings.EnableSSL,

                    Credentials = credentials

                };

                // Send it...

                await client.SendMailAsync(mail);

            }

            catch (Exception ex)

            {

                // co loi khi gui

                throw new InvalidOperationException(ex.Message);

            }

            return Redirect("Index");

        }

        public async Task<IActionResult> TestSendMail(string email = "truongpq197@gmail.com", string subject = "Test gửi mail", string message = "<h1>Nội dung</h1>")

        {

            try

            {

                // Credentials

                var credentials = new NetworkCredential(\_emailSettings.Sender, \_emailSettings.Password);

                // Mail message

                var mail = new MailMessage()

                {

                    From = new MailAddress(\_emailSettings.Sender, \_emailSettings.SenderName),

                    Subject = subject,

                    Body = message,

                    IsBodyHtml = true

                };

                mail.To.Add(new MailAddress(email));

                // Smtp client

                var client = new SmtpClient()

                {

                    Port = \_emailSettings.MailPort,

                    DeliveryMethod = SmtpDeliveryMethod.Network,

                    UseDefaultCredentials = false,

                    Host = \_emailSettings.MailServer,

                    EnableSsl = \_emailSettings.EnableSSL,

                    Credentials = credentials

                };

                // Send it...

                await client.SendMailAsync(mail);

            }

            catch (Exception ex)

            {

                // TODO: handle exception

                throw new InvalidOperationException(ex.Message);

            }

            return View();

        }

    }

}

View:

@model WebGoldenSea.Models.ContactViewModel

@{

    ViewBag.Title = "Contact";

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

....

@Model.setting.CONTACT\_TT

@Model.setting.CONTACT\_DC

@Model.setting.CONTACT\_SDT

@Model.setting.CONTACT\_DES

//lấy thông tin ở trong csdl, setting thuộc SettingHelper.cs (thư mục Helpers), setting được khởi tạo trong ContactViewModel:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebGoldenSea.Models

{

    public class ContactViewModel

    {

        public SettingHelper setting { get; set; }

    }

}

//setting làm nhiệm vụ trung gian giúp lấy data ở csdl cho View (Contact), vì data nằm ở table WSetting trong csdl mà SettingHelper lại kết nối với table Wsetting (code trong SettingHelper.cs):

public static async Task<SettingHelper> ReadServerOptionAsync(GOLDEN\_SEAContext context)

//hàm để lấy data từ Wsetting

{

    List<WSetting> lsSetting = await context.WSetting.ToListAsync();

    SettingHelper serverSetting = new SettingHelper();

.....

//Trong SettingHelper.cs đã khai báo thêm những data mới:

public string CONTACT\_TT { get; set; }

public string CONTACT\_DC { get; set; }

public string CONTACT\_SDT { get; set; }

public string CONTACT\_DES { get; set; }

//Cách lấy data từ table WSetting để vào view (Contact), thêm nội dung vào Index() của ContactController :

public IActionResult Index()

        {

//giúp lấy dữ liệu từ table Wsetting thông qua SettingHelper đổ vào param model (của ContactViewModel), từ đó return model qua view (Contact) để hiển thị data.

            ContactViewModel model = new ContactViewModel();

            model.setting = await SettingHelper.ReadServerOptionAsync(\_context);

            return View(model);

        }

Ở phần button bổ sung thêm:

<**button** class="btn btn-default buttonGray" type="submit" **asp-controller**="Contact" **asp-action**="NewContact">

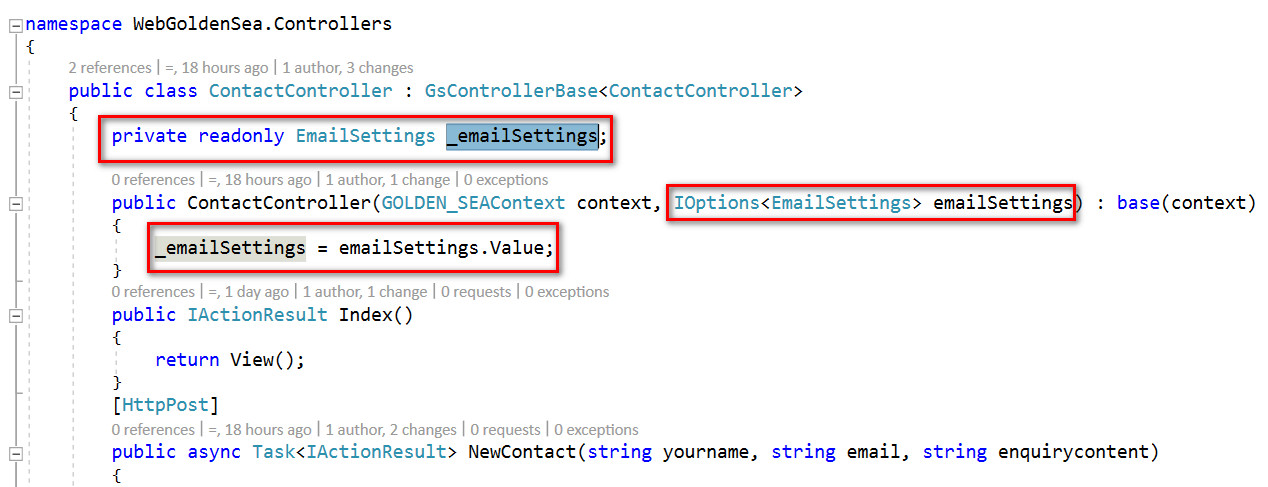
      <span>Submit</span>

</**button**>

// **asp-controller**="Contact" **asp-action**="NewContact" để nhận biết rằng nút submit sẽ cần chạy đến ContactController, sau đó chạy vào action NewContact để thực thi lệnh.

*Phần gửi mail (2)*

Phần đầu của ContactController, thêm:



private readonly EmailSettings \_emailSettings;

IOptions<EmailSettings> emailSettings

\_emailSettings = emailSettings.Value;

File Startup.cs (nằm gần cuối solution), thêm vào cuối:

public void ConfigureServices(IServiceCollection services)

{

.....

.....

services.AddOptions();

services.Configure<Models.EmailSettings>(Configuration.GetSection("EmailSettings"));

}

//gọi hồn file EmailSetting.cs

File EmailSetting.cs (nằm trong Models), thêm:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebGoldenSea.Models //tênsolution.Models

{

    public class EmailSettings

    {

        public string MailServer { get; set; }

        public int MailPort { get; set; }

        public string SenderName { get; set; }

        public string Sender { get; set; }

        public string Password { get; set; }

        public bool EnableSSL { get; set; }

    }

}

File appsettings.json (nằm gần cuối solution), thêm vào:

{

  "ConnectionStrings": {

    "DefaultConnection": "Server=101.99.32.48,8899;Database=GOLDEN\_SEA\_Identity;User Id=sa;Password=1@qweQAZ",

    "GOLDEN\_SEAContext": "Server=101.99.32.48,8899;Database=GOLDEN\_SEA;User Id=sa;Password=1@qweQAZ"

  },

  "Logging": {

    "LogLevel": {

      "Default": "Warning"

    }

  },

  "AllowedHosts": "\*",

  "EmailSettings": {

    "MailServer": "mail.goldensealogistics.com",

    "MailPort": 25,

    "SenderName": "Golden Sea",

    "Sender": "info@goldensealogistics.com",

    "Password": "Tbv!@#$1234",

    "EnableSSL": false

  }

}

*Tuần 3:*

*Trang Blog: Giống trang Category nhưng có thêm 1 menu category.*

Model:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using WebGoldenSea.Models;

using WebGoldenSea.EfModels;

namespace WebGoldenSea.Models

{

    public class BlogViewModel

    {

        public WBlogCaterogy categoryBlog { get; set; }//lấy blog theo FK\_BlogCategory (mã loại blog).

        public List<WBlogCaterogy> categories { get; set; }//lấy danh sách tên các loại blog để làm menu category.

        public List<WBlog> blog { get; set; }//lấy danh sách các blog theo category.

    }

}

Controller

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using WebGoldenSea.EfModels;

using WebGoldenSea.Models;

namespace WebGoldenSea.Controllers

{

    public class BlogController : GsControllerBase<HomeController>

    {

        public BlogController(GOLDEN\_SEAContext context) : base(context)

        {

        }

        public async Task<IActionResult> Index(string id)//tham số truyền vào

        {

            BlogViewModel model = new BlogViewModel();

            model.categoryBlog = await \_context.WBlogCaterogy.SingleOrDefaultAsync(o => o.Id == id); }//lấy blog theo FK\_BlogCategory (mã loại blog).

            model.categories = await \_context.WBlogCaterogy.ToListAsync();//lấy danh sách tên các loại blog để làm menu category.

            model.blog = await \_context.WBlog.Where(u => u.FkBlogCaterogy == id).OrderBy(o => o.BlogName).ToListAsync();//lấy danh sách các blog theo category.

            return View(model);

        }

    }

}

View

//xuất ra màn hình tên category trên menu category, khi bấm category nào thì sẽ chuyển sang trang của category đó. (xuất tên và gắn link cho mỗi tên category).

@foreach(var category in Model.categories)

{

 <li class="list-group-item"> <a href="~/blog?id=@category.Id" class="group-item active">@category.BlogCategoryName</a></li>

}

*Trang BlogDetail: Giống trang ProductDetail.*

Model:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using WebGoldenSea.EfModels;

namespace WebGoldenSea.Models

{

    public class BlogDetailViewModel

    {

        public WBlog detail { get; set; }//lấy chi tiết của 1 blog trong table WBlog

        public List<WBlogCaterogy> category { get; set; }//lấy danh sách tên các loại blog để làm menu category.

    }

}

Controller

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Newtonsoft.Json;

using WebGoldenSea.EfModels;

using WebGoldenSea.Models;

namespace WebGoldenSea.Controllers

{

    public class BlogDetailController : GsControllerBase<HomeController>

    {

        public BlogDetailController(GOLDEN\_SEAContext context) : base(context)

        {

        }

        public async Task<IActionResult> Index(string id)

        {

            BlogDetailViewModel model = new BlogDetailViewModel();

//lấy chi tiết của 1 blog trong table WBlog

           model.detail = await \_context.WBlog.SingleOrDefaultAsync(o => o.Id == id);

//lấy danh sách tên các loại blog để làm menu category.

           model.category = await \_context.WBlogCaterogy.ToListAsync();

           return View(model);

        }}}

View: Tương tự trang Blog

*Trang Cart: Bấm “Add to Cart” thì lưu thông tin cần thiết vào Session, sau đó lấy thông tin vừa được lưu trong Session để xuất ra bên View() – trang Cart.*

*Ví dụ sử dụng Session:*

Chuẩn bị:

Ở file Startup.cs (gần cuối solution) thêm vào:

public void ConfigureServices(IServiceCollection services)

{

...............

...............

services.AddDistributedMemoryCache();

        services.AddSession(options =>

            {

                // Set a short timeout for easy testing.

                //options.IdleTimeout = TimeSpan.FromSeconds(10);

                options.Cookie.HttpOnly = true;

                // Make the session cookie essential

                options.Cookie.IsEssential = true;

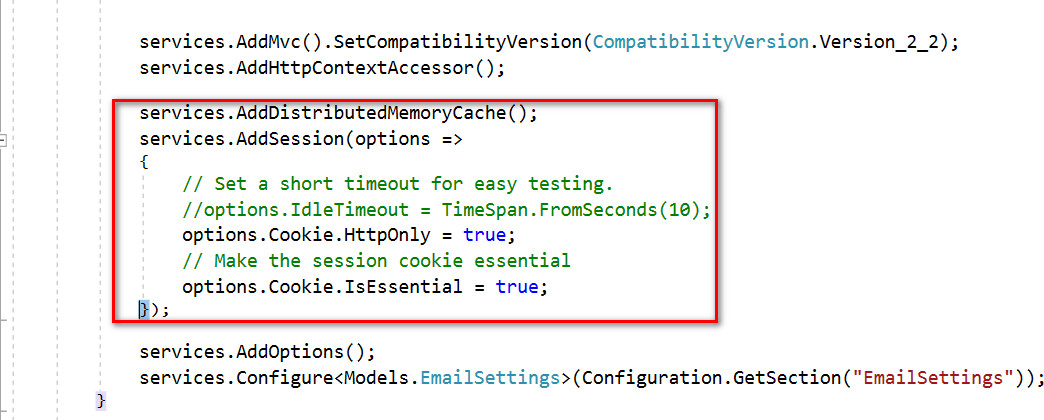
            });

.............

.............

}

Hình thực tế:



Và thêm vào:

public void Configure(IApplicationBuilder app, IHostingEnvironment env)

{

............

............

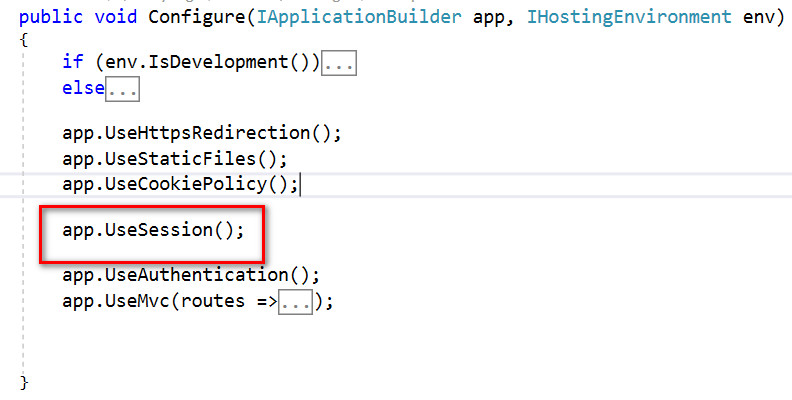
app.UseSession();

............

............

}

Hình thực tế:



Vào làm:

Trong phần action của 1 controller:

public IActionResult Index()

{

    //Tes gán gia tri

    var tempObj = new EmailSettings

    {

        SenderName = "sdfsdf",

        EnableSSL = true,

        Password = DateTime.Now.ToString()

    };

    //Searialize object to string-ép tất cả sang dạng string-put to Session

    var jsonTempObj = JsonConvert.SerializeObject(tempObj);

    HttpContext.Session.SetString("keyABC", jsonTempObj);

    return View();

}

Ở phần action của 1 controller khác:

public async Task<IActionResult> Index(string id)

{

//tiếp nhận chuỗi vừa được ép kiểu, nhờ vào khóa (keyABC) đã đánh dấu.

    var json = HttpContext.Session.GetString("keyABC");

//giải trừ ép kiểu, trả chuỗi về dạng ban đầu

    var obj = JsonConvert.DeserializeObject<EmailSettings>(json);

}

----------------------------------------------------------------/

Model: Shoping Cart.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebGoldenSea.Models

{

    public class ShopingCart

    {

        public string ProductId { get; set; }

        public string ImagePath { get; set; }

        public string ProductName { get; set; }

        public string ProductCode { get; set; }

        public int Quantity { get; set; }

        public decimal Price { get; set; }

        public decimal TotalPrice { get; set; }

    }

}

Model: CartViewModel.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebGoldenSea.Models

{

    public class CartViewModel

    {

//gom lại vào listShopingCart

        public List<ShopingCart> listShopingCart { get; set; }

    }

}

Ví dụ 2:

//Ở 1 homecontroller:

public IActionResult Index()

        {

            List<ShopingCart> listCart = new List<ShopingCart>();

            for (int i = 0; i < 10; i++)//xuất ra 10 sản phẩm

            {

                ShopingCart cart = new ShopingCart();

                cart.ProductId = Guid.NewGuid().ToString();

                cart.ProductCode = "SP" + i.ToString();

                cart.ProductName = "Ten san pham " + i.ToString();

                cart.Quantity = i + 1;

                cart.Price = (i + 1) \* 1000;

                cart.TotalPrice = cart.Price \* cart.Quantity;

                cart.ImagePath = "image" + i.ToString();

                listCart.Add(cart);

            }

         // Tao ra bien tam, ép kiểu

            var jsonListCart = JsonConvert.SerializeObject(listCart);

            HttpContext.Session.SetString("Cart", jsonListCart);

//remove from cart

            listCart.RemoveAt(0);

            jsonListCart = JsonConvert.SerializeObject(listCart);

            HttpContext.Session.SetString("Cart", jsonListCart);

            return View();

        }

//Ở Cartcontroller:

public IActionResult Index()

{

    CartViewModel model = new CartViewModel();

    var json = HttpContext.Session.GetString("Cart");

    model.listShopingCart = JsonConvert.DeserializeObject<List<ShopingCart>>(json);

    //model.Address = "Truong";

    return View(model);

}

//chạy “~/cart” thì sẽ thấy 10 sản phẩm trong giỏ hàng

*Làm trang Cart*

Model: ShopingCart.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebGoldenSea.Models

{

    public class ShopingCart

    {

        public string ProductId { get; set; }

        public string ImagePath { get; set; }

        public string ProductName { get; set; }

        public string ProductCode { get; set; }

        public int Quantity { get; set; }

        public decimal Price { get; set; }

        public decimal TotalPrice { get; set; }

    }

}

Model: CartViewModel.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebGoldenSea.Models

{

    public class CartViewModel

    {

//gom lại vào listShopingCart

        public List<ShopingCart> listShopingCart { get; set; }

    }

}

Trong CartController

public class CartController : GsControllerBase<HomeController>

    {

        public CartController(GOLDEN\_SEAContext context) : base(context)

        {

        }

        public IActionResult Index()

        {

            CartViewModel model = new CartViewModel();

            var json = HttpContext.Session.GetString("Cart");

            model.listShopingCart = JsonConvert.DeserializeObject<List<ShopingCart>>(json);

            //model.Address = "Truong";

            return View(model);

        }

//thêm sản phẩm vào trang Cart

[HttpPost]

public async Task<IActionResult> AddProductToShopingCart(string idProduct)

{

    try

    {

//khai báo product là param thuộc Wproduct dựa vào id

        WProduct product = \_context.WProduct.FirstOrDefault(u => u.Id == idProduct);

        ShopingCart cartItem = new ShopingCart();

        cartItem.ProductId = product.Id;

        cartItem.ImagePath = product.ProductImage;

        cartItem.ProductName = product.ProductName;

        cartItem.ProductCode = product.ProductCode;

        cartItem.Quantity = 1;

        cartItem.Price = product.ProductPrice;

        cartItem.TotalPrice = cartItem.Quantity \* cartItem.Price;

//đầu tiên Get Session (lấy Session), nếu đại diện cho Session(“Cart”) là json mà null thì sẽ tạo biến currentListShopingCart thuộc List<ShopingCart> sau đó add các thông tin lấy được ở trên vào currentListShopingCart. Còn nếu json đã có dữ liệu rồi thì sẽ add thẳng vào currentListShopingCart. Cuối cùng đẩy cả currentListShopingCart vào Session (đã ép kiểu).

        // Get Session

        var json = HttpContext.Session.GetString("Cart");

        List<ShopingCart> currentListShopingCart;

        if (string.IsNullOrEmpty(json))

        {

            // Ko co shoping cart

            currentListShopingCart = new List<ShopingCart>();

        }

        else

        {

            // Co shoping Cart

            currentListShopingCart = JsonConvert.DeserializeObject<List<ShopingCart>>(json);

        }

        // Add New Item

        currentListShopingCart.Add(cartItem);

        // Put to Session

        var jsonListCart = JsonConvert.SerializeObject(currentListShopingCart);

        HttpContext.Session.SetString("Cart", jsonListCart);

        return Json(true);

    }

    catch (Exception ex)

    {

        return Json(new { errorMessage = ex.ToString() });

    }

}

[HttpPost]

public async Task<IActionResult> RemoveProductToShopingCart(string idProduct)

{

    try

    {

        // Get Session

        var json = HttpContext.Session.GetString("Cart");

        List<ShopingCart> currentListShopingCart = JsonConvert.DeserializeObject<List<ShopingCart>>(json);

        // Get Item in Session with ID

        ShopingCart removeItem = currentListShopingCart.FirstOrDefault(u => u.ProductId == idProduct);

        currentListShopingCart.Remove(removeItem);

        // Put to Session

        var jsonListCart = JsonConvert.SerializeObject(currentListShopingCart);

        HttpContext.Session.SetString("Cart", jsonListCart);

        return Json(true);

    }

    catch (Exception ex)

    {

        return Json(new { errorMessage = ex.ToString() });

    }

}

//do sản phẩm nằm ở trang Category nên ajax thêm sản phẩm vào cart sẽ nằm ở trang view của Category

@section scripts {

    <script>

        $(document).ready(function () {

//Dẫn đến tên action và controller

            var \_urlCart = '@Url.Action("AddProductToShopingCart", "Cart")';

//Dùng để debug, khi bấm vào nút có class addToCart thì sẽ hiển thị ra số lượng tất cả nút có cùng class ở phần console (trong F12). Class trong bài: class="addToCart btn-button"

            console.log($(".addToCart").length);

//khi vùng có class tên addToCart được click thì sẽ chạy

            $(".addToCart").click(function () {

//Tạo biến lưu giữ thông tin của sản phẩm được bấm. Thông tin được lưu giữ là id của sản phẩm, cùng vị trí với class, tạo thêm data-product-id="@product.Id", trong đó data- định nghĩa 1 loại dữ liệu có tên product-id, dữ liệu này sẽ lưu giữ thông tin là id của sản phẩm được bấm vào.

                var $clickedButton = $(this);

                var productId = $clickedButton.data('product-id');

                $.ajax({

                    url: \_urlCart,

                    type: 'POST',

                    data: {

//đẩy id lấy được ở trên truyền qua cho idProduct – đây cũng là param của hàm AddProductToShopingCart bên CartController.

                        idProduct: productId,

                    },

                    dataType: 'json',

                    error: function (arg1, arg2, arg3) {

                        alert('Process error')

                    },

                    success: function (data) {

                        console.log(data);

                        if (data.errorMessage) {

                            alert('Lỗi khi lưu dữ liệu');

                        }

                        else {

                            alert('Lưu thành công')

                        }

                    },

                });

            });

        });

    </script>

}

--------------------------------/

Trong trang Cart: Xóa 1 sản phẩm khỏi danh sách sẩn phẩm trong giỏ hàng. Thêm vào cuối view của Cart ajax:

@section scripts {

    <script>

        $(document).ready(function () {

//gọi đến controller Cart và action RemoveProductToShopingCart trong Cart.

            var \_urlCart = '@Url.Action("RemoveProductToShopingCart", "Cart")';

//ajax sẽ hoạt động khi bấm vào nơi có class btn-danger

            $(".btn-danger").click(function () {

                var $clickedButton = $(this);

//tìm đến data trên View có tên product-id (VD: data-product-id=”@acx. productId”) và lấy giá trị của nó, ở trường hợp này thì lấy Id của sản phẩm đó.

                var productId = $clickedButton.data('product-id');

                $.ajax({

                    url: \_urlCart,

                    type: 'POST',

                    data: {

                        idProduct: productId,

                    },

                    dataType: 'json',

                    error: function (arg1, arg2, arg3) {

                        alert('Process error')

                    },

                    success: function (data) {

                        console.log(data);

                        if (data.errorMessage) {

                            alert('Lỗi khi lưu dữ liệu');

                        }

                        else {

                            // Remove item ra khoi cart

//số lượng parent() phụ thuộc vào số lượng thẻ (tag) mà chỗ nút bấm có class btn-danger nằm ở trong. VD: @foreach( var ... in ...){

.....

.....

<button type="button" data-product-id="@cartItem.ProductId" class="btn btn-danger"><i class="fa fa-times-circle"></i></button>

                                        </span>1

                                    </div>2

                                </td>3

                                <td class="text-right">$@cartItem.Price</td>

                                <td class="text-right">$@cartItem.TotalPrice</td>

                            </tr>4

=>4 thẻ =>cần 4 parent()

}

                            var $parrent = $clickedButton.parent().parent().parent().parent();

                            $parrent.remove();

                        }

                    },

                });

            });

        });

    </script>

}

*Trang wishlist: Giống trang Cart, tạo Session+Action Add-Remove+Ajax cho Add-Remove.*

WishlistViewModel:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebGoldenSea.Models

{

    public class WishlistViewModel

    {

//cũng giống CartViewModel, tạo param mới thuộc ShopingCart để dùng lại các thuộc tính của ShopingCart, vì Wishlist sẽ chứa nhiều sản phẩm nên sẽ là List<ShopingCart>

        public List<ShopingCart> listWishlist { get; set; }

    }

}

WishlistController:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Newtonsoft.Json;

using WebGoldenSea.EfModels;

using WebGoldenSea.Models;

namespace WebGoldenSea.Controllers

{

    public class WishlistController : GsControllerBase<HomeController>

    {

        public WishlistController(GOLDEN\_SEAContext context) : base(context)

        {

        }

        public IActionResult Index()

        {

//tên Model, tên Session phải đúng

            WishlistViewModel model = new WishlistViewModel();

            var json = HttpContext.Session.GetString("Wishlist");

            model.listWishlist = JsonConvert.DeserializeObject<List<ShopingCart>>(json);

            return View(model);

        }

        [HttpPost]

        public async Task<IActionResult> AddProductToWishlist(string idProduct)

        {

            try

            {

                WProduct product = \_context.WProduct.FirstOrDefault(u => u.Id == idProduct);

                ShopingCart wishlistItem = new ShopingCart();

                wishlistItem.ProductId = product.Id;

                wishlistItem.ImagePath = product.ProductImage;

                wishlistItem.ProductName = product.ProductName;

                wishlistItem.ProductCode = product.ProductCode;

                wishlistItem.Price = product.ProductPrice;

                // Get Session, tên Session phải đúng

                var json = HttpContext.Session.GetString("Wishlist");

                List<ShopingCart> currentListWishlist;

                if (string.IsNullOrEmpty(json))

                {

                    // Ko co shoping cart

                    currentListWishlist = new List<ShopingCart>();

                }

                else

                {

                    // Co shoping Cart

                    currentListWishlist = JsonConvert.DeserializeObject<List<ShopingCart>>(json);

                }

                // Add New Item

                currentListWishlist.Add(wishlistItem);

                // Put to Session

                var jsonListCart = JsonConvert.SerializeObject(currentListWishlist);

                HttpContext.Session.SetString("Wishlist", jsonListCart);

                return Json(true);

            }

            catch (Exception ex)

            {

                return Json(new { errorMessage = ex.ToString() });

            }

        }

        [HttpPost]

        public async Task<IActionResult> RemoveProductFromWishlist(string idProduct)

        {

            try

            {

                // Get Session

                var json = HttpContext.Session.GetString("Wishlist");

                List<ShopingCart> currentListShopingCart = JsonConvert.DeserializeObject<List<ShopingCart>>(json);

                // Get Item in Session with ID

                ShopingCart removeItem = currentListShopingCart.FirstOrDefault(u => u.ProductId == idProduct);

                currentListShopingCart.Remove(removeItem);

                // Put to Session

                var jsonListCart = JsonConvert.SerializeObject(currentListShopingCart);

                HttpContext.Session.SetString("Wishlist", jsonListCart);

                return Json(true);

            }

            catch (Exception ex)

            {

                return Json(new { errorMessage = ex.ToString() });

            }

        }

    }

}

Wishlist-View-Ajax: Chức năng Add thẳng sản phẩm vào Session Cart để hiển thị trên trang Cart và chức năng xóa sản phẩm ra khỏi Wishlist. Cách chèn 2 hoặc nhiều ajax vào 1 script.

@section scripts {

    <script>

        $(document).ready(function () {

            var \_urlCart = '@Url.Action("AddProductToShopingCart", "Cart")';

            var \_urlRemoveWishlist = '@Url.Action("RemoveProductFromWishlist", "Wishlist")';

            $(".btn-primary").click(function () {

                var $clickedButton = $(this);

                var productId = $clickedButton.data('product-id');//xem ở trên để biết ý nghĩa.

                $.ajax({

                    url: \_urlCart,

                    type: 'POST',

                    data: {

                        idProduct: productId,

                    },

                    dataType: 'json',

                    error: function (arg1, arg2, arg3) {

                        alert('Process error')

                    },

                    success: function (data) {

                        console.log(data);

                        if (data.errorMessage) {

                            alert('Lỗi khi lưu dữ liệu');

                        }

                        else {

                            alert('Lưu thành công')

                        }

                    },

                });

            });

            $(".btn-danger").click(function () {

                var $clickedButton = $(this);

                var productId = $clickedButton.data('product-id');

                $.ajax({

                    url: \_urlRemoveWishlist,

                    type: 'POST',

                    data: {

                        idProduct: productId,

                    },

                    dataType: 'json',

                    error: function (arg1, arg2, arg3) {

                        alert('Process error')

                    },

                    success: function (data) {

                        console.log(data);

                        if (data.errorMessage) {

                            alert('Lỗi khi lưu dữ liệu');

                        }

                        else {

                            // Remove item ra khoi cart, xem ở trên để biết ý nghĩa.

                   var $parrent = $clickedButton.parent().parent().parent().parent();

                            $parrent.remove();

                        }

                    },

                });

            });

        });

    </script>

}

------------------------------------------------/

*Trang Compare tương tự Wishlist: Tạo Model. Tạo Session, hàm Add+Remove trong Controller. Viết Ajax trong phần view với chức năng xóa sản phẩm khỏi danh sách Compare và thêm thẳng sản phẩm vào Cart.*