* Lưu Product trong List Product và List Product được lưu trong Session Storage

Tạo CartItemViewModel trong Models

public class CartItemViewModel

{

public List<CartItemModel> CartItems { get; set;} // Là List Products để chứa Product

public decimal GrandTotal {get; set;} // Tổng tiền của List Products

}

* Tạo Class Session để lưu List Products

using Microsoft.AspNetCore.Http;

using Newtonsoft.Json;

namespace TechLaptop.Repository

{

public static class SessionExtensions

{

public static void SetJson(this ISession session, string key, object value) // chuyển Object thành Json rồi lưu lại trong Session

{

session.SetString(key, JsonConvert.SerializeObject(value));

}

public static T GetJson<T>(this ISession session, string key) // lấy json trong Session rồi chuyển lại thành Object

{

var sessionData = session.GetString(key); // nếu Key không tồn tại sessionData trả về NULL

return sessionData == null ? default(T) : JsonConvert.DeserializeObject<T>(sessionData);

}

}

}

gọi nó ở Controller

List <CartItemModel> cartItems = HttpContext.Session.GetJson<List<CartItemModel>>("Cart") ?? new List<CartItemModel>();

ở [grogram.cs](http://grogram.cs) phải cấu hình

builder .Services.AddDistributedMemoryCache(); //

builder.Services.AddSession(options =>

{

options.IdleTimeout = TimeSpan.FromMinutes(30);

options.Cookie.HttpOnly = true;

options.Cookie.IsEssential = true;

});

var app = builder.Build();

app.UseSession(); đặt cái này sau app

**AddDistributedMemoryCache()** → tạo **nơi lưu trữ session** (RAM tạm thời trên server).

**AddSession(...)** → **cấu hình session**, như:

* Thời gian sống (IdleTimeout)
* Bảo mật cookie (HttpOnly)
* Cookie bắt buộc (IsEssential)
* Controller Shopping Cart

namespace TechLaptop.Controllers{

public class CartController : Controller

{

private readonly DataContext \_dataContext;

public CartController(DataContext \_context)

{

\_dataContext = \_context;

}

// action này dùng :lấy List Product trong session lưu vào CartItemViewModel (Models )

public IActionResult Index() {

List<CartItemModel> cartItems = HttpContext.Session.GetJson<List<CartItemModel>>("Cart") ?? new List<CartItemModel>();

CartItemViewModel cartVM = new CartItemViewModel

{

CartItems = cartItems,

GrandTotal = cartItems.Sum(x => x.Quantity \* x.Price)

};

return View(cartVM);

}

// action này dùng để thêm Product vào List Products và sau đó lưu List Products vàp trong session

public async Task<IActionResult> Add(int Id)

{ // tìm Product theo Id

ProductModel product = await \_dataContext.Products.FindAsync(Id);

List<CartItemModel> cart = HttpContext.Session.GetJson<List<CartItemModel>>("Cart") ?? new List<CartItemModel>();

CartItemModel cartItems = cart.Where(c => c.ProductId == Id).FirstOrDefault();

// nếu chưa có Product trong List Products thì thêm Product đó vào List Products và nếu có rồi chỉ cần tăng số lượng của Product đó

if (cartItems == null)

{

cart.Add(new CartItemModel(product));

}

else

{

cartItems.Quantity++;

}

HttpContext.Session.SetJson("Cart", cart);

TempData ["Success"] = "Product added to cart successfully!";

return Redirect(Request.Headers["Referer"].ToString());

}

// Giảm số lượng Product đó

public async Task<IActionResult> Decrease(int Id)

{

List<CartItemModel> cart = HttpContext.Session.GetJson<List<CartItemModel>>("Cart") ?? new List<CartItemModel>();

CartItemModel cartItems = cart.Where(c => c.ProductId == Id).FirstOrDefault();

// nếu số lượng của Product đó >1 thì giảm còn nếu <= 1 thì remove Product đó trong List Products

if (cartItems.Quantity > 1)

{

cartItems.Quantity--;

}

else

{

cart.RemoveAll(c => c.ProductId == Id);

}

if (cart.Count == 0)

{

HttpContext.Session.Remove("Cart"); //

}

else

{

HttpContext.Session.SetJson("Cart", cart);

}

TempData ["Success"] = "Product quantity updated successfully!";

return RedirectToAction("Index");

}

public async Task<IActionResult> Increase(int Id)

{

List<CartItemModel> cart = HttpContext.Session.GetJson<List<CartItemModel>>("Cart") ?? new List<CartItemModel>();

CartItemModel cartItems = cart.Where(c => c.ProductId == Id).FirstOrDefault();

if (cartItems.Quantity >= 1)

{

cartItems.Quantity++;

}

else

{

cart.RemoveAll(c => c.ProductId == Id); //

}

if (cart.Count == 0)

{

HttpContext.Session.Remove("Cart");

}

else

{

HttpContext.Session.SetJson("Cart", cart);

}

TempData ["Success"] = "Product quantity updated successfully!";

return RedirectToAction("Index");

}

public async Task<IActionResult> Remove(int Id)

{

List<CartItemModel> cart = HttpContext.Session.GetJson<List<CartItemModel>>("Cart") ?? new List<CartItemModel>();

cart.RemoveAll(c => c.ProductId == Id);

if (cart.Count == 0)

{

HttpContext.Session.Remove("Cart");

}

else

{

HttpContext.Session.SetJson("Cart", cart);

}

TempData["Success"] = "Product removed from cart successfully!";

return RedirectToAction("Index");

}

public async Task<IActionResult> Clear(int Id)

{

HttpContext.Session.Remove("Cart");

TempData["Success"] = "Cart cleared successfully!";

return RedirectToAction("Index");

}

}

}