



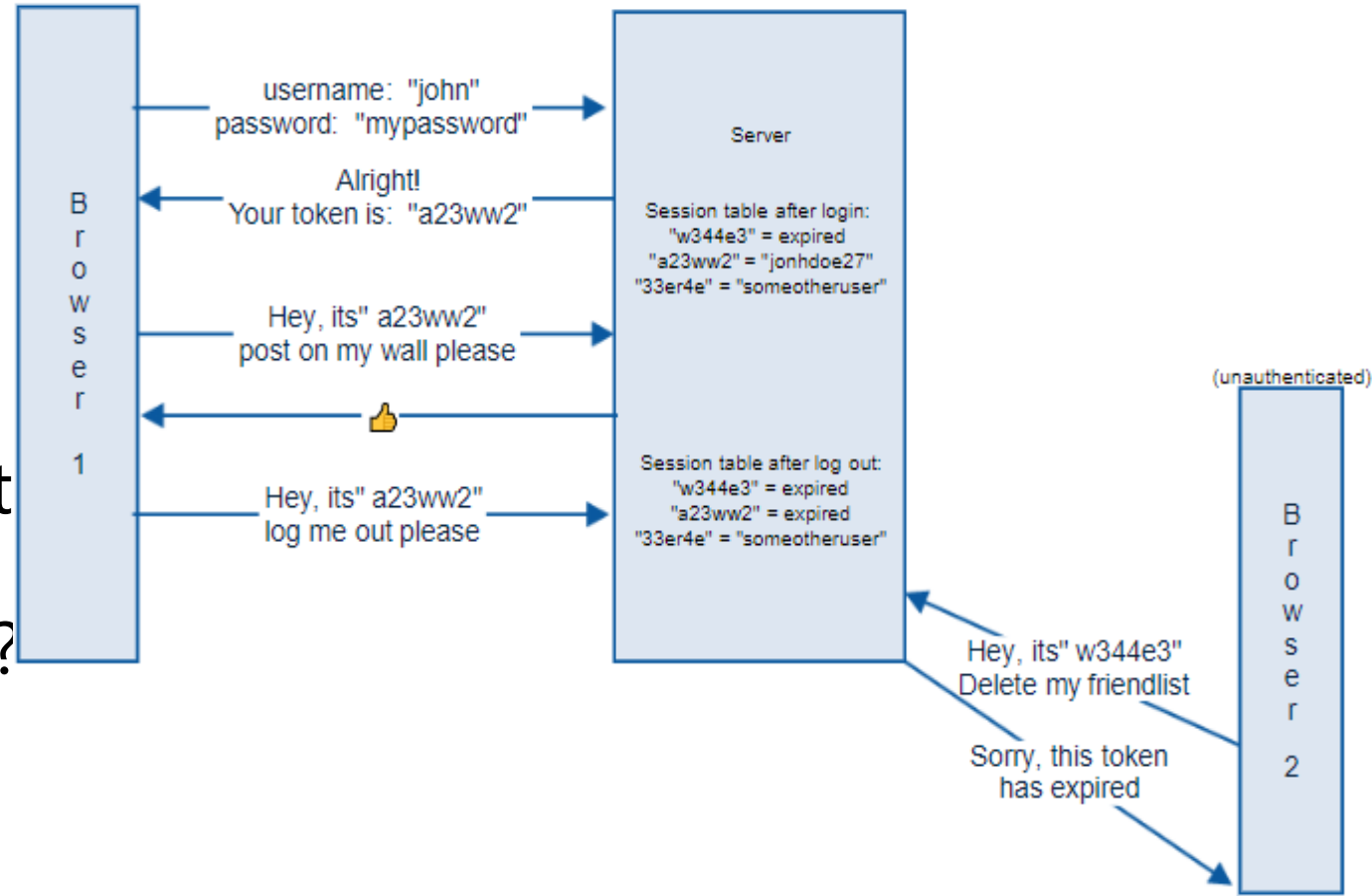
# **LẬP TRÌNH C# 4**

## **BÀI 8: SESSIONS – SHOPPING CART (P1)**

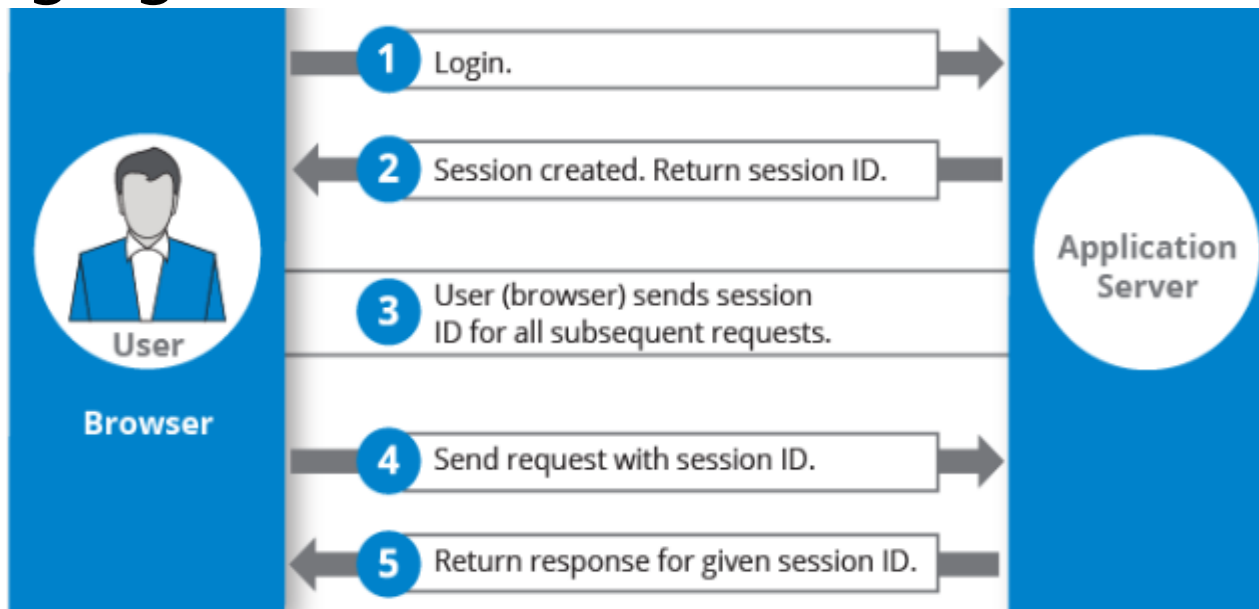
- ⊙ Khái niệm Session
- ⊙ Phiên người dùng đăng nhập
- ⊙ Giỏ hàng



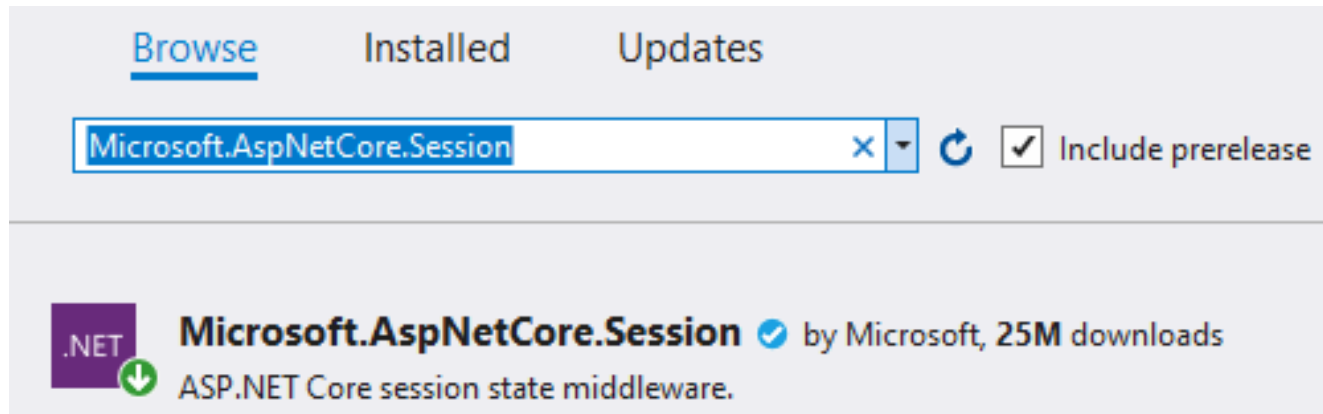
- ❑ Việc giao tiếp giữa trình duyệt với máy được thực hiện thông qua hàng loạt các router trên mạng internet. Vấn đề đặt ra trong quá trình giao tiếp này là làm sao để phân biệt được giữa các trình duyệt (máy tính) khác nhau?



- ❑ Session là cơ chế để lưu lại dữ liệu của phiên làm việc của ứng dụng - ứng với từng khách truy cập: thông tin đăng nhập, trao đổi dữ liệu từ trang này qua trang khác, thông tin giỏ hàng...
- ❑ Dữ liệu Session lưu trữ trên Server có thể là ở bộ nhớ Cache, có thể là ở CSDL SQL Server hoặc những nguồn lưu cache khác nhau



- ❑ Sử dụng Nuget thêm Session package “Microsoft.AspNetCore.Session” vào dự án



- ❑ Thêm dịch vụ session service vào phương thức ConfigureServices

```
services.AddSession(options => {  
    options.IdleTimeout = TimeSpan.FromMinutes(1); //You can set Time  
});
```

## ❑ Thêm Request Pipeline cho session

```
app.UseStaticFiles();  
  
app.UseSession();  
  
app.UseMvc(routes =>  
{  
    routes.MapRoute(  
        name: "default",  
        template: "{controller=Home}/{action=Index}/{id?}");  
});
```

- ❑ Lưu và đọc dữ liệu session được thực hiện thông qua đối tượng có kiểu **Isession**. Đối tượng này lấy được từ thuộc tính của HttpContext.
- ❑ Một số phương thức, thuộc tính của Isession

Member	Ý nghĩa
ID	Thuộc tính lấy ID của Session, ID này có gửi về lưu ở Cookie
Clear()	Xóa bỏ các giá trị lưu trong Session
Set(String key, Byte[])	Lưu mảng byte vào Session với key chỉ ra
TryGetValue(String key, Byte[])	Lấy dữ liệu lưu trong key (trả về false là có thành công)
SetString(String key, String s)	Lưu chuỗi vào key
GetString(String key)	Lấy chuỗi lưu trong key
SetInt32(String key, Int32 val)	Lưu số kiểu int32 vào key
GetInt32(String key)	Lấy số int32 trong key

- ❑ Ví dụ khởi tạo lưu session tại action Index và đọc session tại action About

```
public IActionResult Index()
{
    HttpContext.Session.SetString("name", "Phan Viet The");
    HttpContext.Session.SetString("email", "thepv@uit.edu.vn.com");
    return View();
}
```

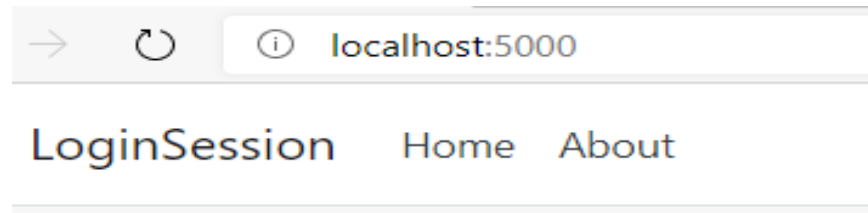
0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions

```
public IActionResult About()
{
    ViewBag.Name = HttpContext.Session.GetString("name");
    ViewBag.Email = HttpContext.Session.GetString("email");
    ViewData["Message"] = "Your about page, please refresh page after one minute";
    ViewData["Title"] = "Demo session login";

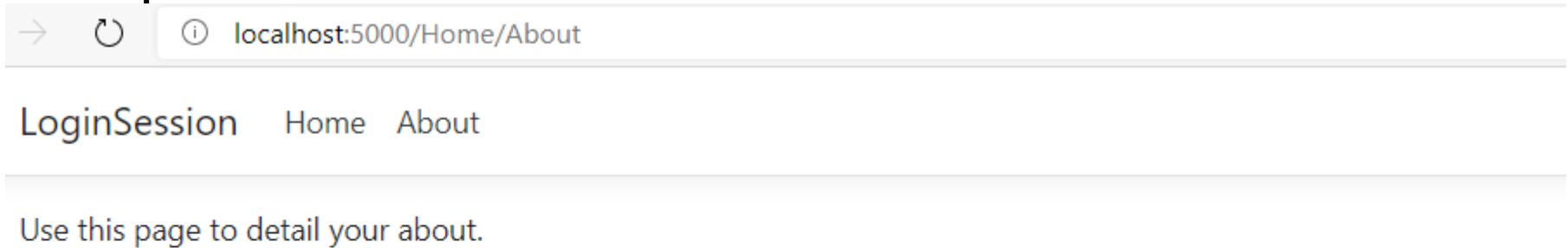
    return View();
}
```



- ❑ Chạy /home/index thì session được khởi tạo



- ❑ Click chọn about, dữ liệu session được truyền qua about



**Title: Demo session login**

**Message: Your about page, please refresh page after one minute**

**name: Phan Viet The**

**email: thepv@uit.edu.vn.com**



**DEMO**

- Hiện thực ví dụ Login với session  
Qua page Index và About



- ❑ Session hỗ trợ nhiều kiểu dữ liệu khác nhau trong đó có kiểu đối tượng
- ❑ Có thể chuyển đổi đối tượng phức tạp thành JSON và lưu trữ nó dưới dạng chuỗi. Sau đó, truy xuất nó dưới dạng một chuỗi và chuyển hóa thành đối tượng ban đầu
- ❑ Sử dụng `SerializeObject` và `DeserializeObject` của lớp `JsonConvert` để xử lý dữ liệu đối tượng dưới định dạng Json, sau đó lưu và đọc cho session

## ❑ Tạo lớp User

```
namespace ComplexDataSession.Models
{
    3 references | 0 changes | 0 authors, 0 changes
    public class User
    {
        1 reference | 0 changes | 0 authors, 0 changes | 0 exceptions
        public string Name { get; set; }
        1 reference | 0 changes | 0 authors, 0 changes | 0 exceptions
        public double Percentage { get; set; }
    }
}
```

# VÍ DỤ LƯU VÀ ĐỌC THÔNG TIN ĐỐI TƯỢNG USER VỚI SESSION

- ❑ Tạo lớp có 2 phương thức hỗ trợ chuyển đổi đối tượng sang json và ngược lại, sau đó tương tác với session

```
public static class SessionExtensions
{
    1 reference | 0 changes | 0 authors, 0 changes | 0 exceptions
    public static T GetComplexData<T>(this ISession session, string key)
    {
        var data = session.GetString(key);
        if (data == null)
        {
            return default(T);
        }
        return JsonConvert.DeserializeObject<T>(data);
    }

    1 reference | 0 changes | 0 authors, 0 changes | 0 exceptions
    public static void SetComplexData(this ISession session, string key, object value)
    {
        session.SetString(key, JsonConvert.SerializeObject(value));
    }
}
```

- ❑ Khởi tạo session cho đối tượng User tại action Index và hiển thị thông tin tại action GetComplexData

```
public class HomeController : Controller
{
    [Route("home/GetComplexData")]
    0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions
    public IActionResult GetComplexData()
    {
        ViewBag.data = HttpContext.Session.GetComplexData<User>("UserData");
        return View();
    }
    0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions
    public IActionResult Index()
    {
        User user = new User
        {
            Name = "Phan Viết Thế",
            Percentage = 90.45
        };

        HttpContext.Session.SetComplexData("UserData", user);
        return View();
    }
}
```

## ❑ Kiểm tra tại home/GetComplexData

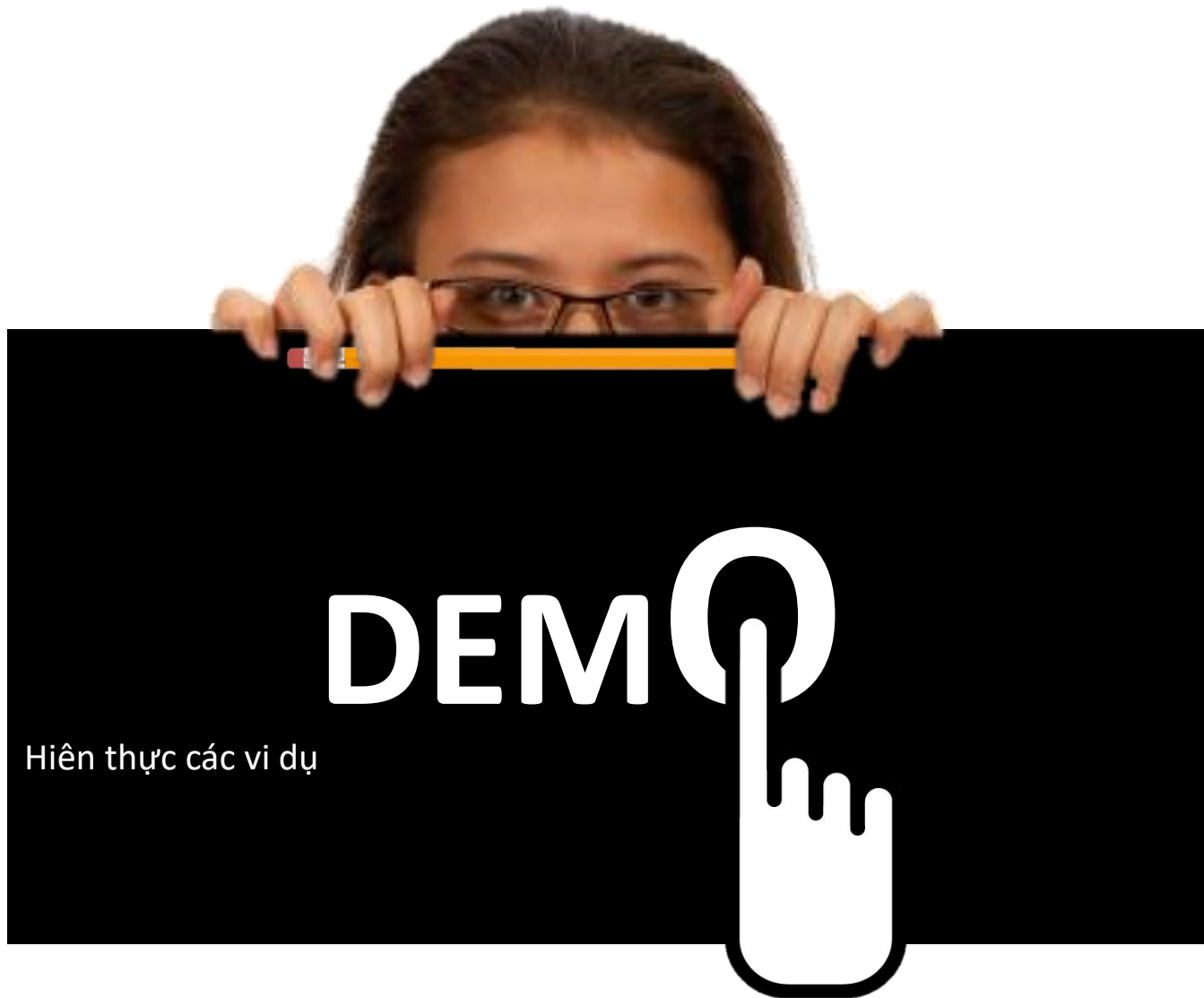
```
GetComplexData.cshtml  User.cs  SessionExtensions.cs
1
2  @{
3      ViewData["Title"] = "GetComplexData";
4  }
5
6  <h1>GetComplexData</h1>
7
8  <h3>name: @ViewBag.data.Name</h3>
9
```

→ ↻ ⓘ localhost:5000/home/GetComplexData

ComplexDataSession Home Privacy

# GetComplexData

name: Phan Viết Thế








# **LẬP TRÌNH C# 4**

## **BÀI 8: SESSIONS – SHOPPING CART (P2)**

## □ Các bước thực hiện:

1. Tạo csdl Product
2. Dùng EF core hiển thị sản phẩm với các model
3. Tạo controller Product, tạo view index cho Product
4. Tạo lớp SessionHelper hỗ trợ chuyển đổi đối tượng sang json và ngược lại, sau đó tương tác với session
5. Tạo controller Cart với các action: buy, remove, isExist
6. Tạo View index cho cart

## ❑ Structure of Product Table

	Column Name	Data Type	Allow Nulls
	Id	int	<input type="checkbox"/>
	Name	varchar(250)	<input checked="" type="checkbox"/>
	Price	money	<input checked="" type="checkbox"/>
	Quantity	int	<input checked="" type="checkbox"/>
	Status	bit	<input checked="" type="checkbox"/>

## ❑ Data of Product Table

Id	Name	Price	Quantity	Status
1	Name 1	20.0000	4	True
2	Name 2	12.0000	8	False
3	Name 3	4.0000	3	True
4	Name 4	17.0000	8	True

Khai báo kết nối

```
"ConnectionStrings": {  
  "DefaultConnection": "Server=.;Initial Catalog=Xoa;Integrated Security=True;I  
}
```

```
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)  
{  
    var builder = new ConfigurationBuilder()  
        .SetBasePath(Directory.GetCurrentDirectory())  
        .AddJsonFile("appsettings.json");  
    var configuration = builder.Build();  
    optionsBuilder.UseSqlServer(configuration["ConnectionStrings:DefaultConnection"]);  
}  
  
2 references | 0 changes | 0 authors, 0 changes | 0 exceptions  
public DbSet<Product> Product { get; set; }
```

Tạo DataContext

# DÙNG EF CORE HIỂN THỊ SẢN PHẨM

```
public class Product
```

```
{
```

1 reference | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public int Id { get; set; }
```

0 references | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public string Name { get; set; }
```

1 reference | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public decimal Price { get; set; }
```

0 references | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public int Quantity { get; set; }
```

0 references | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public bool Status { get; set; }
```

```
}
```

```
public class Item
```

```
{
```

4 references | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public Product Product { get; set; }
```

4 references | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public int Quantity { get; set; }
```

```
}
```

Tạo class Product

Phương thức truy  
vấn tất cả sản phẩm

Tạo ProductModel

```
public class ProductModel
```

```
{
```

```
private DbContext db = new DbContext();
```

1 reference | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public List<Product> FindAll()
```

```
{
```

```
var a = db.Product.ToList();
```

```
return a;
```

```
}
```

2 references | 0 changes | 0 authors, 0 changes | 0 exceptions

```
public Product Find(int id)
```

```
{
```

```
var a = db.Product.Find(id);
```

```
return a;
```

```
}
```

```
L
```

```
}
```

Phương thức truy vấn  
sản phẩm theo ID

Lớp đại diện lưu thông  
tin giỏ hàng

- ❑ Tạo controller Product, truyền danh sách product cho view Index

```
[Route("product")]
```

0 references | 0 changes | 0 authors, 0 changes

```
public class ProductController : Controller
```

```
{
```

```
    private DataContext db = new DataContext();
```

```
    [Route("")]
```

```
    [Route("index")]
```

```
    [Route("~/")]
```

0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions

```
    public IActionResult Index()
```

```
    {
```

```
        ProductModel productModel = new ProductModel();
```

```
        ViewBag.products = productModel.FindAll();
```

```
        return View();
```

```
    }
```

```
}
```

- ❑ Sử dụng Link Tag Helper trong view Index kết hợp ghi nhận sản phẩm được chọn và truyền Id sản phẩm cho controller "cart" và action "buy"

## Product List

Id	Name	Price	Quantity	Status	Action
1	thepv	20.0000	4	True	<a href="#">Buy Now</a>
2	Name 2	12.0000	8	False	<a href="#">Buy Now</a>
3	Name 3	4.0000	3	True	<a href="#">Buy Now</a>
4	Name 4	17.0000	8	True	<a href="#">Buy Now</a>

```
@foreach (var product in ViewBag.products)
{
    <tr>
        <td>@product.Id</td>
        <td>@product.Name</td>
        <td>@product.Price</td>
        <td>@product.Quantity</td>
        <td>@product.Status</td>
        <td align="center">
            <a asp-controller="cart" asp-action="buy" asp-route-id="@product.Id">Buy Now</a>
        </td>
    </tr>
}
```

Link Tag Helper

- ❑ Tạo lớp SessionHelper hỗ trợ chuyển đổi đối tượng sang json và ngược lại, sau đó tương tác với session

```
public static void SetObjectAsJson(this ISession session, string key, object value)
{
    session.SetString(key, JsonConvert.SerializeObject(value));
}
5 references | 0 changes | 0 authors, 0 changes | 0 exceptions
public static T GetObjectFromJson<T>(this ISession session, string key)
{
    var value = session.GetString(key);
    return value == null ? default(T) : JsonConvert.DeserializeObject<T>(value);
}
```



- ❑ Action 'isExist' kiểm tra một sản phẩm có tồn tại trong giỏ hàng chưa
- ❑ Khi người dùng chọn sản phẩm từ Index:
  - ❖ Nếu sp tồn tại rồi thì tăng số lượng
  - ❖ Nếu chưa tồn tại thì thêm vào giỏ hàng
- ❑ Trả về vị trí sp trong giỏ hàng khi người dùng xóa sp

```
private int isExist(int id)
{
    List<Item> cart = SessionHelper.GetObjectFromJson<List<Item>>(HttpContext.Session, "cart");
    for (int i = 0; i < cart.Count; i++)
    {
        if (cart[i].Product.Id.Equals(id))
        {
            return i;
        }
    }
    return -1;
}
```

- ❑ Action 'Buy(int id)' thêm sp vào giỏ hàng, nhận tham số đầu vào là 'id' từ view Index của Product. Sau khi xử lý trả kết quả về view Index của Cart
  - ❖ Nếu giỏ hàng rỗng: thêm sp vào với số lượng 1
  - ❖ Nếu giỏ hàng đã có sp:
    - Nếu sản phẩm được chọn đã tồn tại trong giỏ hàng: tăng số lượng thêm 1
    - Nếu sản phẩm được chọn là sp chưa tồn tại trong giỏ hàng: thêm sp vào giỏ hàng với số lượng 1

```
[Route("buy/{id}")]
```

0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions

```
public IActionResult Buy(int id)
```

```
{
```

```
    ProductModel productModel = new ProductModel();
```

```
    if (SessionHelper.GetObjectFromJson<List<Item>>(HttpContext.Session, "cart") == null)
```

```
    {
```

```
        List<Item> cart = new List<Item>();
```

```
        cart.Add(new Item { Product = productModel.Find(id), Quantity = 1 });
```

```
        SessionHelper.SetObjectAsJson(HttpContext.Session, "cart", cart);
```

```
    }
```

```
    else
```

```
    {
```

```
        List<Item> cart = SessionHelper.GetObjectFromJson<List<Item>>(HttpContext.Session, "cart");
```

```
        int index = isExist(id);
```

```
        if (index != -1)
```

```
        {
```

```
            cart[index].Quantity++;
```

```
        }
```

```
        else
```

```
        {
```

```
            cart.Add(new Item { Product = productModel.Find(id), Quantity = 1 });
```

```
        }
```

```
        SessionHelper.SetObjectAsJson(HttpContext.Session, "cart", cart);
```

```
    }
```

```
    return RedirectToAction("Index");
```

```
}
```

Nếu giỏ hàng rỗng: thêm sp vào với số lượng 1

Nếu sản phẩm được chọn đã tồn tại trong giỏ hàng: tăng số lượng thêm 1

- ❑ Action 'Remove(int id)' nhận tham số đầu vào là 'id' sp được chọn xóa trong giỏ hàng, Sau khi xử lý trả kết quả về view Index của Cart

```
[Route("remove/{id}")]
0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions
public IActionResult Remove(int id)
{
    List<Item> cart = SessionHelper.GetObjectFromJson<List<Item>>(HttpContext.Session, "cart");
    int index = isExist(id);
    cart.RemoveAt(index);
    SessionHelper.SetObjectAsJson(HttpContext.Session, "cart", cart);
    return RedirectToAction("Index");
}
```

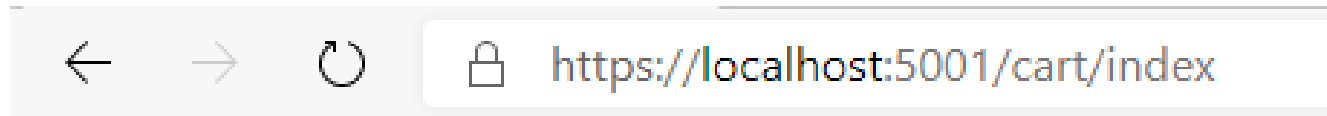
- ❑ Action 'Index' lấy danh sách sp và số lượng có trong giỏ hàng truyền cho view Index của cart

```
[Route("cart")]
0 references | 0 changes | 0 authors, 0 changes
public class CartController : Controller
{
    [Route("index")]
    0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions
    public IActionResult Index()
    {
        var cart = SessionHelper.GetObjectFromJson<List<Item>>(HttpContext.Session, "cart");
        ViewBag.cart = cart;
        ViewBag.total = cart.Sum(item => item.Product.Price * item.Quantity);
        return View();
    }
}
```

- ❑ Hiển thị danh sách sp giỏ hàng:
- ❑ Kết hợp link tag helper, khi người dùng chọn remove sẽ ghi nhận 'id' và truyền id này cho action 'remove' của controller 'cart'

```
@foreach (var item in ViewBag.cart)
{
    <tr>
        <td><a asp-controller="cart" asp-action="remove" asp-route-id="@item.Product.Id">Remove</a></td>
        <td>@item.Product.Id</td>
        <td>@item.Product.Name</td>
        <td>@item.Product.Price</td>
        <td>@item.Quantity</td>
        <td>@(item.Product.Price * item.Quantity)</td>
    </tr>
}
<tr>
    <td align="right" colspan="6">Sum</td>
    <td>
        @ViewBag.total
    </td>
</tr>
```

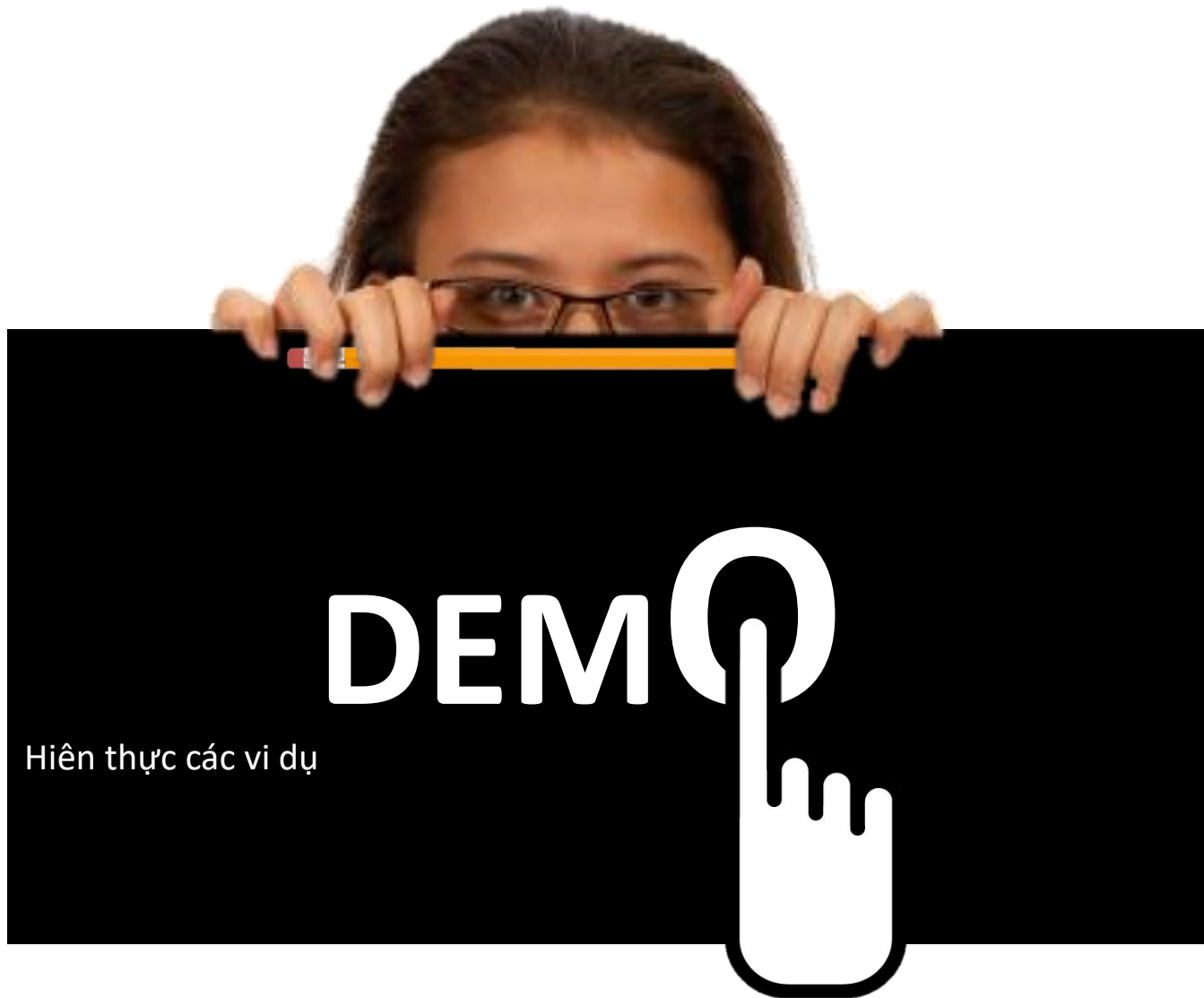
❑ Hiển thị danh sách sp giỏ hàng:



## Cart Page

Option	Id	Name	Price	Quantity	SubTotal	Total
<a href="#">Remove</a>	2	Name 2	12.0000	3	36.0000	
<a href="#">Remove</a>	4	Name 4	17.0000	1	17.0000	
<a href="#">Remove</a>	1	thepv	20.0000	1	20.0000	
Sum						73.0000

[Continue Shopping](#)





## Tổng kết bài học

- ◎ Khái niệm Session
- ◎ Phiên người dùng đăng nhập
- ◎ Giỏ hàng





**KẾT THÚC**