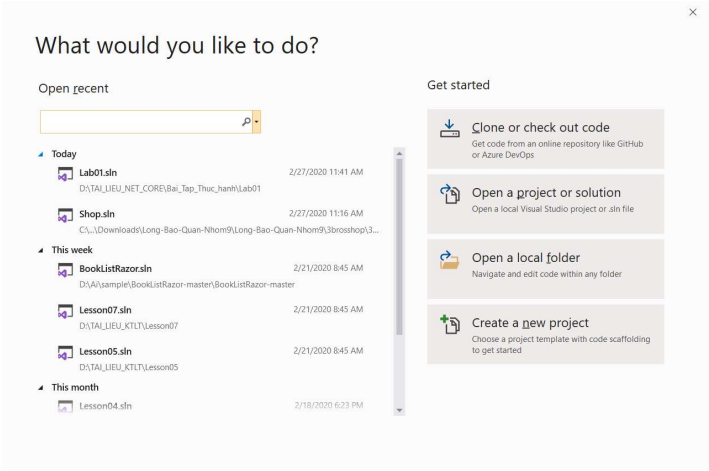
Bài Tập Thực Hành Đồ Án 1

Năm 2019-2020

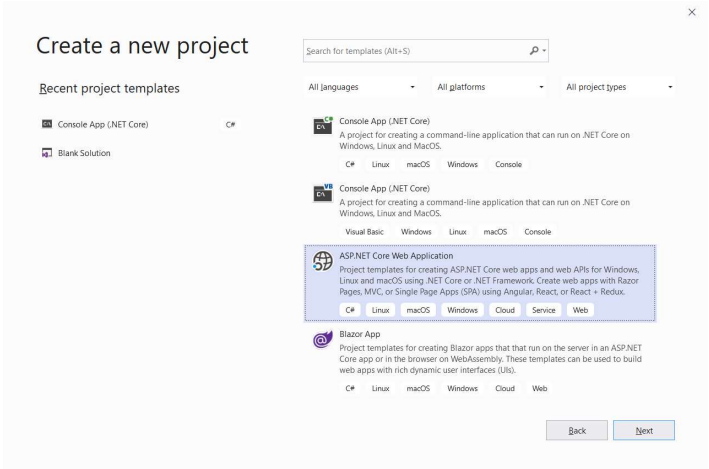
Lab01 – Làm quen với AspNet.Core, Xây dựng trang HomePage

Bước 1: Tạo project với VS2019 (.netcore 3.1)

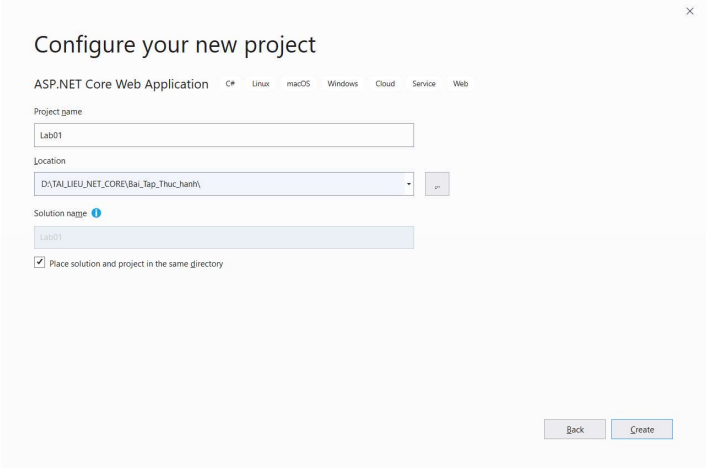
VS2019 -> Create new project



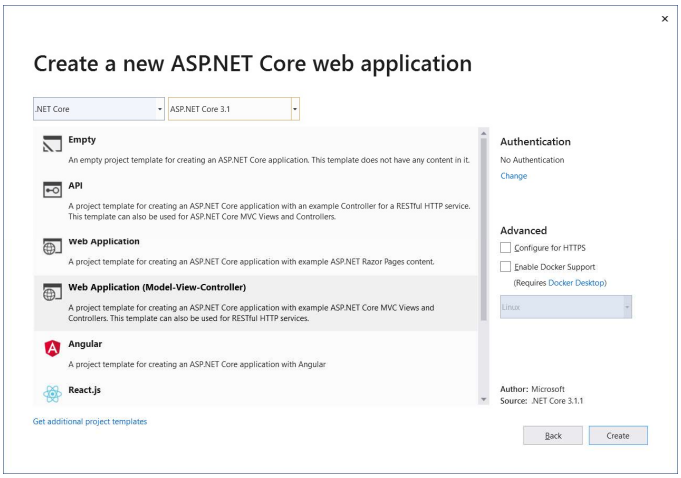
Bước 2: chọn Asp.Net Core Web Application



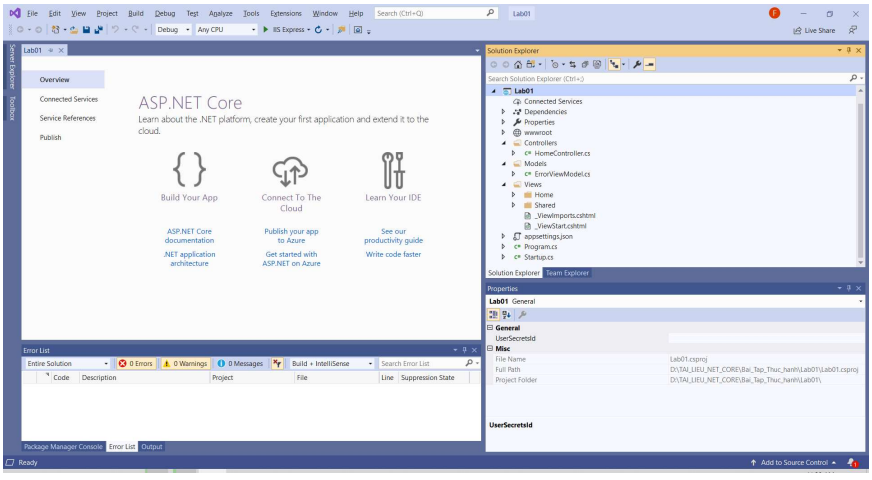
Bước 3: chọn đường dẫn lưu Project



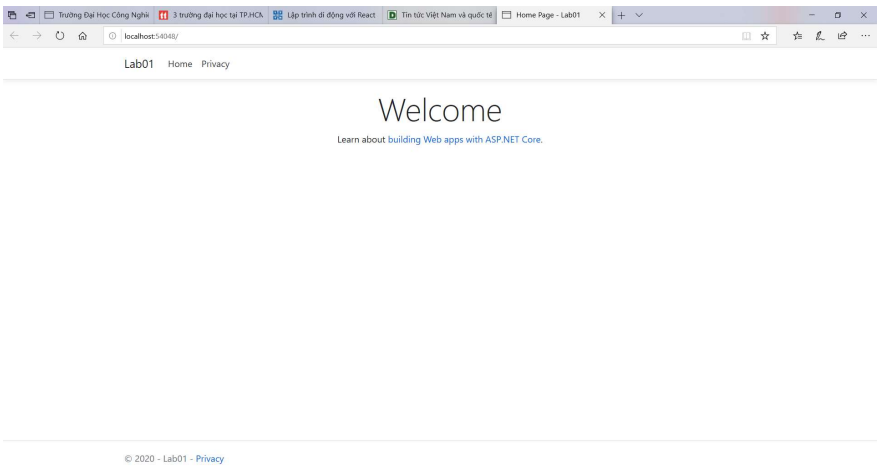
Bước 4: chọn template MVC cho Project



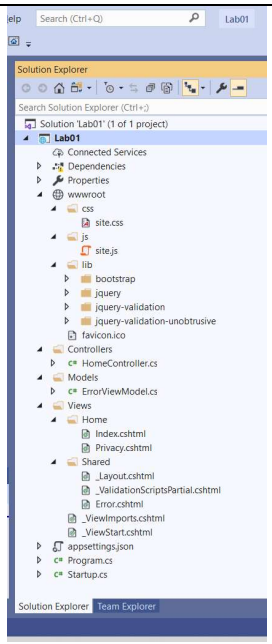
Bước 5: Cấu trúc Project



Bước 6: F5 để run chương trình và kết quả



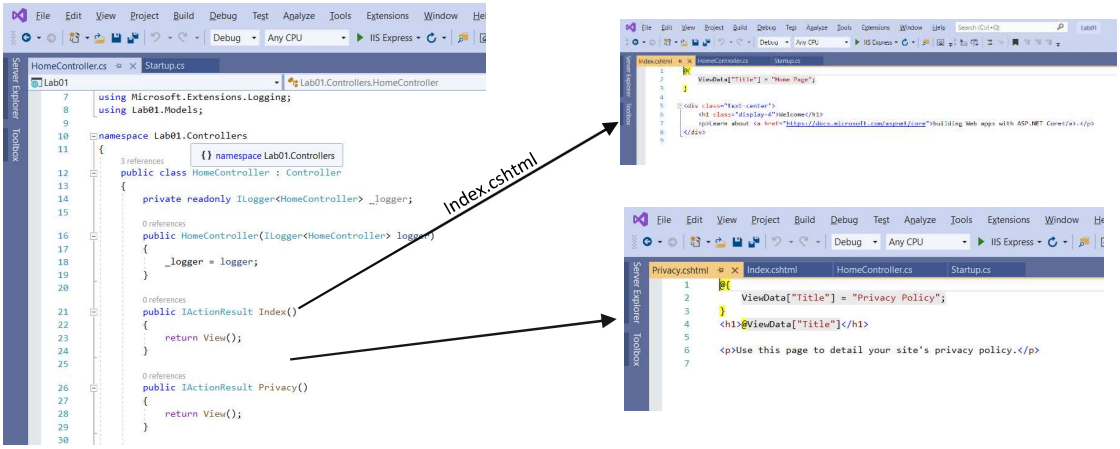
Bước 7: Quan sát cấu trúc Project



1- wwwroot: chứa css, bootstrap(4.0),...
2- Controllers: chứa các controller điều khiển các trang và chuyển hướng dữ liệu, lưu ý: định dạng file luôn có "Controller" ở cuối file.
3- Models: chứa phần dữ liệu liên kết database và controller, lưu ý: định dạng file luôn có "Model" ở cuối file.
4- Views: trang hiển thị
5- appsettings.json: chứa kết nối đến SQLserver
6- Program.cs : file chạy của Project
7- Startup.cs: file chứa các cấu hình của Project

Bước 8: Quan sát các bước liên kết giữa Controllers và Views

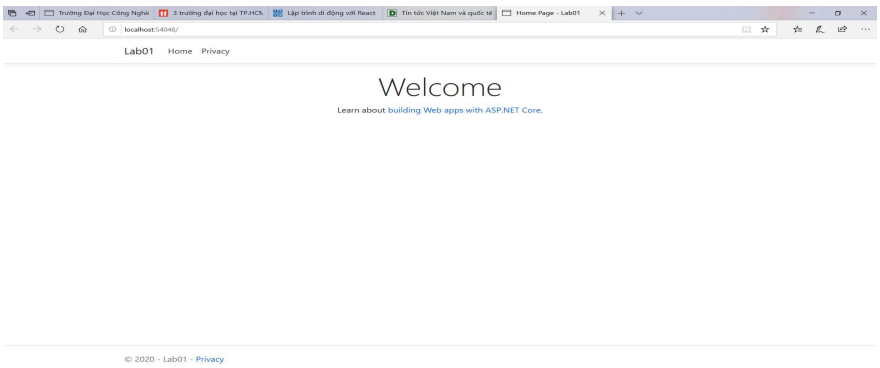
- Mỗi View sẽ tương ứng với 1 phương thức trong controllers.
- Một controller điều khiển View thông qua các phương thức và có thể chứa nhiều phương thức
- Mỗi phương thức trong controller kết thúc bằng return View() được dùng để hiển thị
- Tên của phương thức trong controller sẽ trùng tên với View



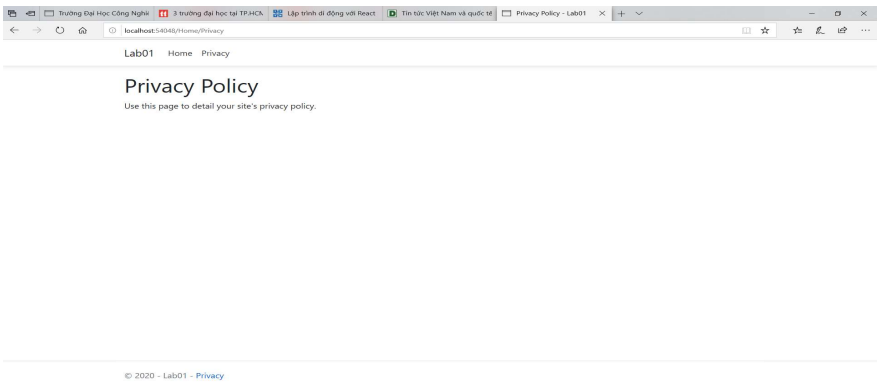
Privacy.cshtml

Kết quả:

http://localhost:54048/

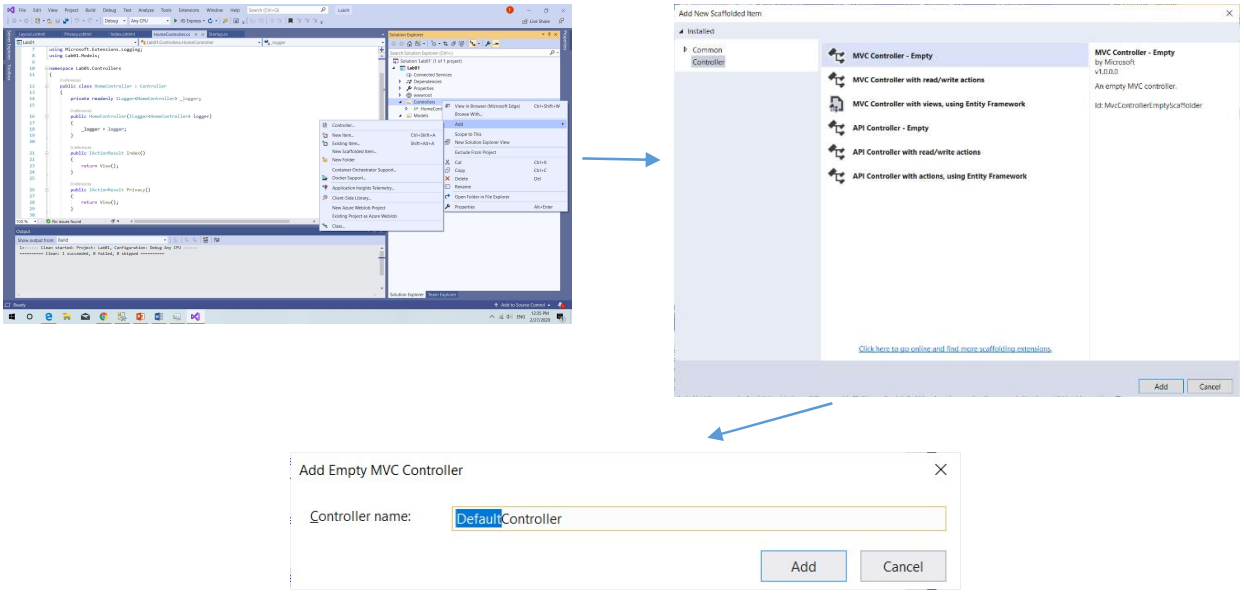


http://localhost:54048/Home/Privacy



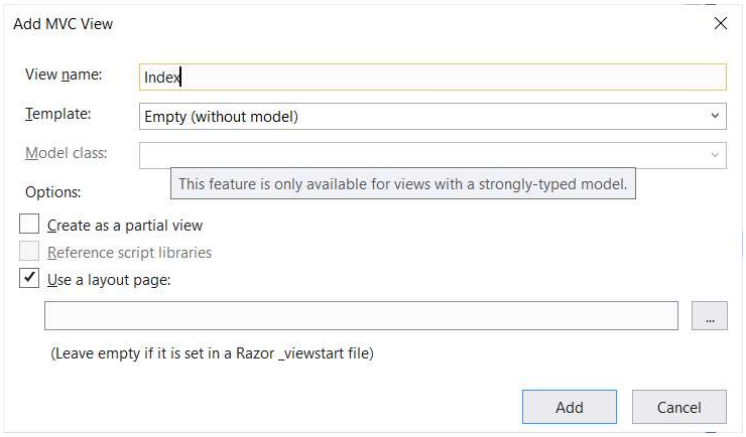
Bước 9 : Thêm một view mới vào Project

Bước 9.1: Right Click thư mục Controllers → Add → Controller→ MVC Contronller Empty → ProductController



Bước 9.2: Tạo Product View

Tạo thu mục có tên trùng với controller (Product) , RC vào View chọn Add → View→ Index



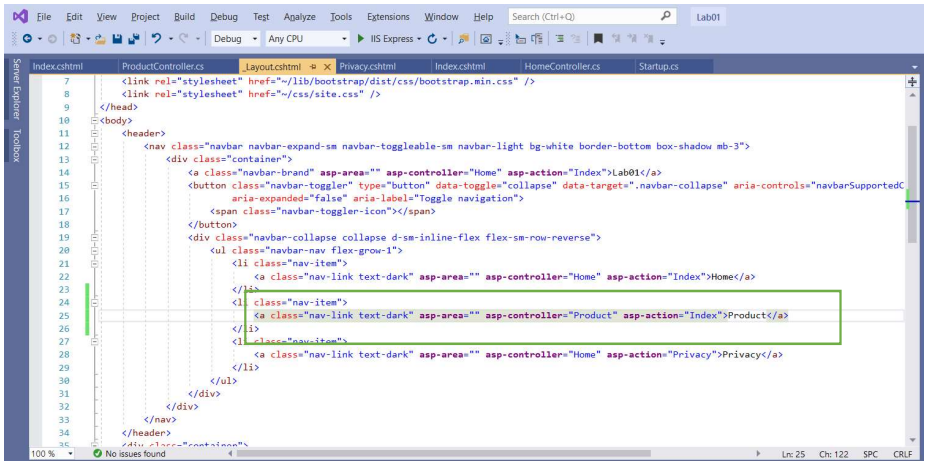
Bước 9.3: thêm Product vào menu Navigation link

vào Shared/_Layout.cshtml:

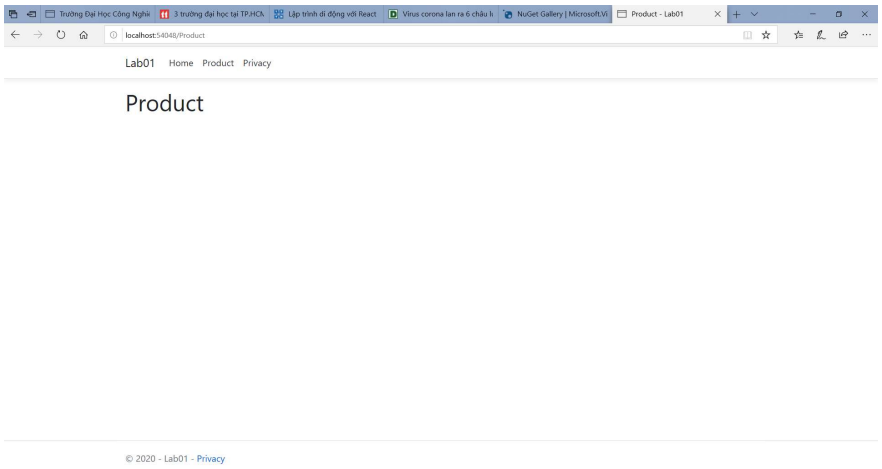
Thêm code như hình dưới

trong đó:

- `asp-controller="Product"` → Tên của controller
- `asp-action="Index"` → Tên phương thức trong controller



Bước 10: F5 kiểm tra kết quả:

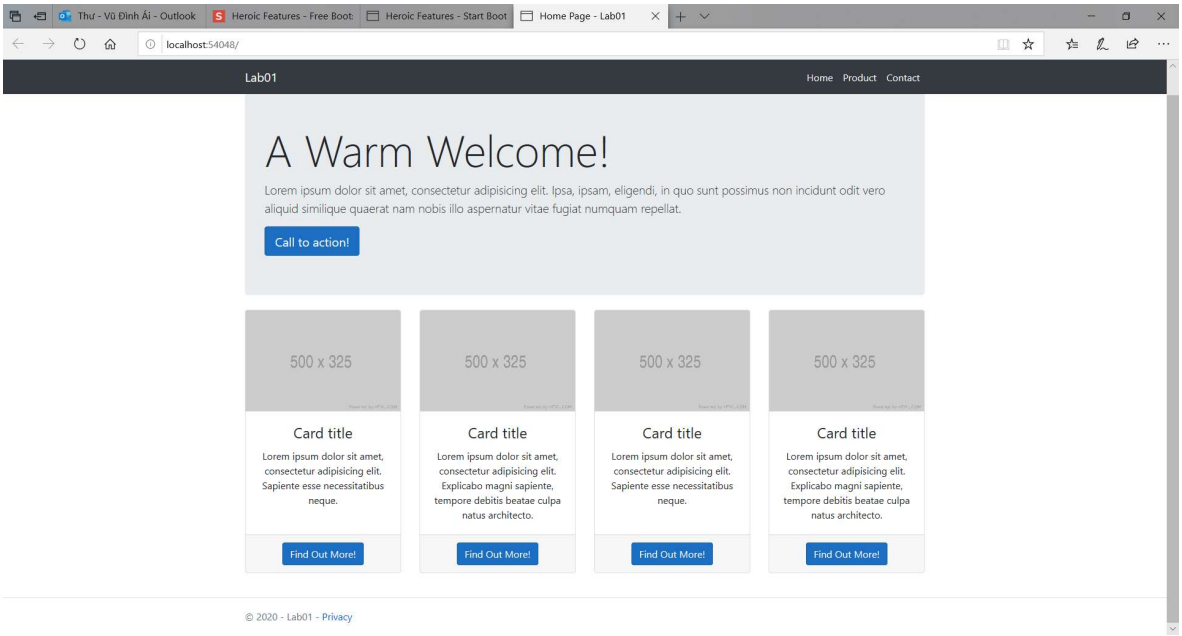


Yêu cầu:

Hãy thực hiện lại các bước trên

Hãy xây dựng một trang Homepage với layout tùy ý: (Vận dụng HTML, css, Bootstrap)

Ví dụ:



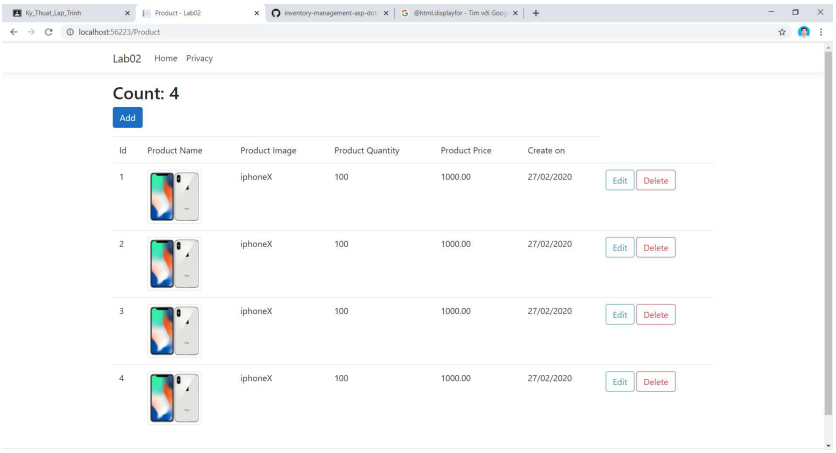
Tham khảo:

https://startbootstrap.com/templates/

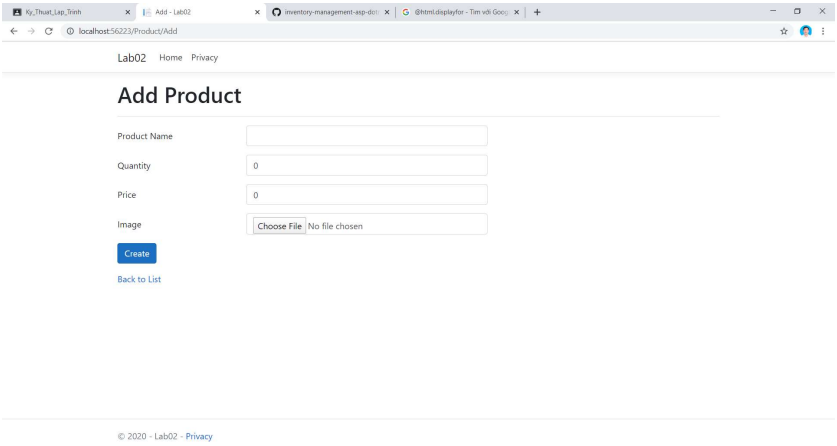
http://th-phone.azurewebsites.net

Lab 02: Xây dựng trang CRUD(Create, Update, Delete)

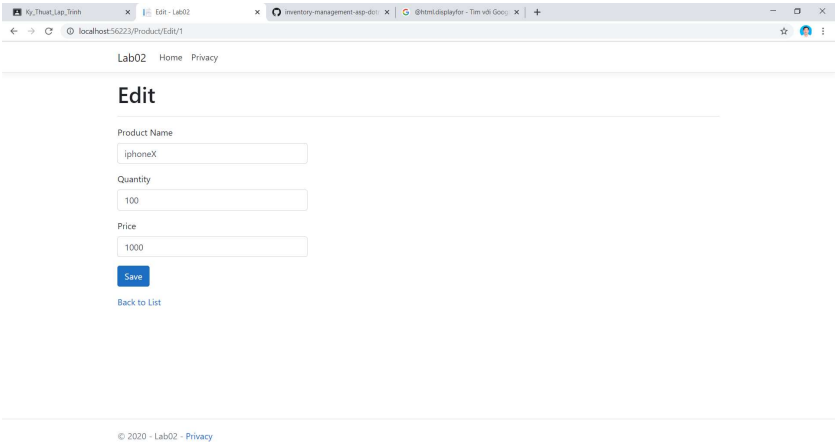
Hiên thị:



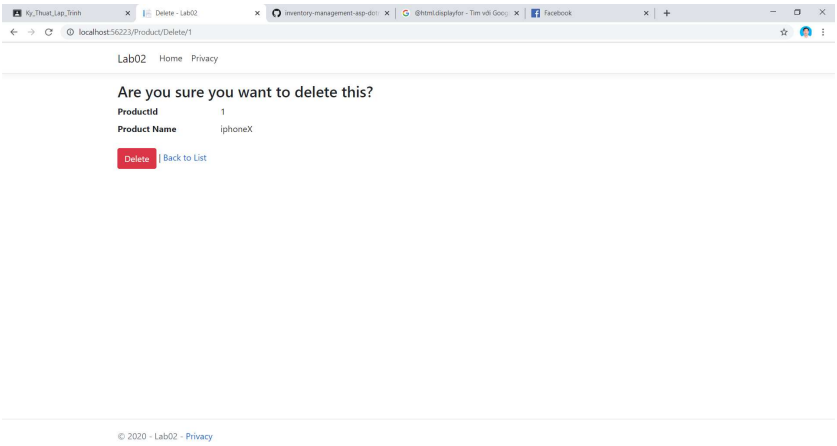
Thêm mới:



Cập nhật:



Xóa:



Hướng dẫn:

Bước 0: Thêm pattern Singleton vào Project

Startup.cs

```
public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();
            services.AddSingleton(provider =>
ProductData.initData());
        }
```

Bước 1: Tạo Model

ProductModel.cs

```
public class ProductModel{
        public int ProductId { get; set; }
        [DisplayName("Product Name")]
        [Required(ErrorMessage ="Not null")]
        [RegularExpression(@"^[A-Z]+[a-zA-Z""'\s-]*$")]
        public string ProductName { get; set; }
        [Required]
        [Range(0, 100)]
        [DisplayName("Quantity")]
        public int ProductQuantity { get; set; }
        [DisplayName("Image")]
        public string ProductImage { get; set; }
        [DisplayName("Category")]
        public int CategoryId { get; set; }
        [Required]
        [Range(0, 999.99)]
        [DisplayName("Price")]
        public double ProductPrice { get; set; }
```

```
        [DisplayName("Create on")]
        public DateTime CreateDate { get; set; }
        private static int nextId = 1;

        public ProductModel()
        {
            ProductId = nextId;
            nextId++;
        }

        public override int GetHashCode()
        {
            return ProductId;
        }
}
```

ProductData.cs (Dùng để lưu danh sách ProductModel.cs)

```
public class ProductData
    {
        public List<ProductModel> ProductList { get; set; }
        public static ProductData initData()
        {
            List<ProductModel> products = new
List<ProductModel>();
            products.AddRange(new List<ProductModel>
            {
                 new ProductModel()
                {
                    ProductName = "iphoneX",
                    ProductImage = "iphoneX.png",
                    ProductQuantity = 100,
                    ProductPrice = 1000.00,
                    CreateDate = DateTime.Now

                },
                 new ProductModel()
                {
                    ProductName = "iphoneX",
                    ProductImage = "iphoneX.png",
                    ProductQuantity = 100,
                    ProductPrice = 1000.00,
                   CreateDate = DateTime.Now

                },
                 new ProductModel()
                {
```

```
                    ProductName = "iphoneX",
                    ProductImage = "iphoneX.png",
                    ProductQuantity = 100,
                    ProductPrice = 1000.00,
                   CreateDate = DateTime.Now

                },
                 new ProductModel()
                {
                    ProductName = "iphoneX",
                    ProductImage = "iphoneX.png",
                    ProductQuantity = 100,
                    ProductPrice = 1000.00,
                    CreateDate = DateTime.Now

                }
            }); ;
            return new ProductData()
            {
                ProductList = products
            };
        }
    }
```

Bước 2: Tạo Controller

ProductController.cs

```
public class ProductController : Controller
    {
        ProductData productData;
        public ProductController(ProductData productData)
        {
            this.productData = productData;
        }
        [HttpGet]
        public IActionResult Index()
        {
            List<ProductModel> products =
productData.ProductList;
            return View(products);
        }
        [HttpGet]
        public IActionResult Add()
        {
            ProductModel productModel = new ProductModel();
```

```
            return View(productModel);
        }
        [HttpPost]
        public IActionResult Add(ProductModel
productModel,IFormFile photo)
        {
            if (ModelState.IsValid)
            {
                ProductModel newProduct = new ProductModel();
                if(photo == null || photo.Length == 0)
                {
                    newProduct.ProductImage = "abc.png";
                }
                else
                {
                    var path =
Path.Combine(Directory.GetCurrentDirectory(), "wwwroot/images",
photo.FileName);
                    var stream = new FileStream(path,
FileMode.Create);
                    photo.CopyToAsync(stream);
                    newProduct.ProductImage = photo.FileName;
                }
                newProduct.ProductName =
productModel.ProductName;
                newProduct.ProductQuantity =
productModel.ProductQuantity;
                newProduct.ProductPrice =
productModel.ProductPrice;
                newProduct.CreateDate = DateTime.Now;
                productData.ProductList.Add(newProduct);
                return RedirectToAction("Index","Product");
            }
            else
            {
                return View(productModel);
            }


        }
        [HttpGet]
        public IActionResult Edit(int id)
        {
            ProductModel oldProduct =
productData.ProductList.FirstOrDefault(p => p.ProductId == id);
```

```
            return View(oldProduct);
        }
        [HttpPost]
        public IActionResult Edit(int id,ProductModel
productModel)
        {
            if (ModelState.IsValid)
            {
                ProductModel oldProduct =
productData.ProductList.FirstOrDefault(p => p.ProductId == id);
                oldProduct.ProductName =
productModel.ProductName;
                oldProduct.ProductQuantity =
productModel.ProductQuantity;
                oldProduct.ProductPrice =
productModel.ProductPrice;
                oldProduct.CreateDate = DateTime.Now;
                ViewBag.Status = 1;
            }
          return View(productModel);
        }


        public IActionResult Delete(int id)
        {
            ProductModel oldProduct =
productData.ProductList.FirstOrDefault(p => p.ProductId == id);
            return View(oldProduct);
        }
        [HttpPost,ActionName("Delete")]
        public IActionResult DeleteConfirmed(int id)
        {
            productData.ProductList.RemoveAll(p => p.ProductId
== id);
            return RedirectToAction("Index", "Product");
        }
    }
```

Bước 3: Tao các View

Index.html

```
@{
    ViewData["Title"] = "Product";
}
@model List<ProductModel>
```

```
<h2>Count: @Model.Count</h2>
<p><a class="btn btn-primary" asp-controller="Product" asp-
action="Add">Add</a></p>
<table class="table">
    <thead>
        <tr>
            <td>Id</td>
            <td>Product Name</td>
            <td>Product Image</td>
            <td>Product Quantity</td>
            <td>Product Price</td>
            <td>Create on</td>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model)
        {
        <tr>
            <td>@item.ProductId</td>
            <td><img src="~/images/@item.ProductImage"
class="img-thumbnail" width="100px" height="100px"
alt="iphoneX" /></td>
            <td>@item.ProductName</td>
            <td>@item.ProductQuantity</td>
            <td>@item.ProductPrice.ToString("0.00")</td>
            <td>@item.CreateDate.ToString("dd/MM/yyyy")</td>
            <td><a class="btn btn-outline-info"asp-
controller="Product" asp-action="Edit" asp-route-
id="@item.ProductId">Edit</a>
                <a class="btn btn-outline-danger" asp-
controller="Product" asp-action="Delete" asp-route-
id="@item.ProductId">Delete</a></td>
        </tr>
        }
    </tbody>

</table>
```

Add.cshtml

```
@model Lab02.Models.ProductModel

@{
    ViewData["Title"] = "Add";
}
```

```
<h1>Add Product</h1>
<hr />
<div class="row">
    <div class="col-md-10">
        <form asp-action="Add" asp-controller="Product"
method="post" enctype="multipart/form-data">
            <div class="form-group row">
                <label asp-for="ProductName" class="col-sm-3
col-form-label"></label>
                <div class="col-sm-6">
                    <input asp-for="ProductName" class="form-
control">
                </div>
                <span asp-validation-for="ProductName"
class="text-danger"></span>
            </div>
            <div class="form-group row">
                <label asp-for="ProductQuantity" class="col-sm-
3 col-form-label"></label>
                <div class="col-sm-6">
                    <input asp-for="ProductQuantity"
class="form-control">
                </div>
                <span asp-validation-for="ProductQuantity"
class="text-danger"></span>
            </div>
            <div class="form-group row">
                <label asp-for="ProductPrice" class="col-sm-3
col-form-label"></label>
                <div class="col-sm-6">
                    <input asp-for="ProductPrice" class="form-
control">
                </div>
                <span asp-validation-for="ProductPrice"
class="text-danger"></span>
            </div>
            <div class="form-group row">
                <label class="col-sm-3 col-form-
label">Image</label>
                <div class="col-sm-6">
                    <input class="form-control" type="file"
name="photo">
                </div>
            </div>
```

```
            <div class="form-group">
                <input type="submit" value="Create" class="btn
btn-primary" />
            </div>
            <a asp-action="Index">Back to List</a>
        </form>
    </div>
</div>
```

Edit.cshtml

```
@model Lab02.Models.ProductModel

@{
    ViewData["Title"] = "Edit";
}

<h1>Edit</h1>
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Edit">
            <div asp-validation-summary="ModelOnly"
class="text-danger"></div>
            <div class="form-group" hidden>
                <label asp-for="ProductId" class="control-
label"></label>
                <input asp-for="ProductId" class="form-control"
/>
                <span asp-validation-for="ProductId"
class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="ProductName" class="control-
label"></label>
                <input asp-for="ProductName" class="form-
control" />
                <span asp-validation-for="ProductName"
class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="ProductQuantity"
class="control-label"></label>
                <input asp-for="ProductQuantity" class="form-
control" />
```

```
                <span asp-validation-for="ProductQuantity"
class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="ProductPrice" class="control-
label"></label>
                <input asp-for="ProductPrice" class="form-
control" />
                <span asp-validation-for="ProductPrice"
class="text-danger"></span>
            </div>
            <div class="form-group">
                <input type="submit" value="Save" class="btn
btn-primary" />
            </div>
            @if(ViewBag.Status == 1)
            {
                <p class="text-warning">Record saved </p>
            }
        </form>
    </div>
</div>

<div>
    <a asp-action="Index">Back to List</a>
</div>
```

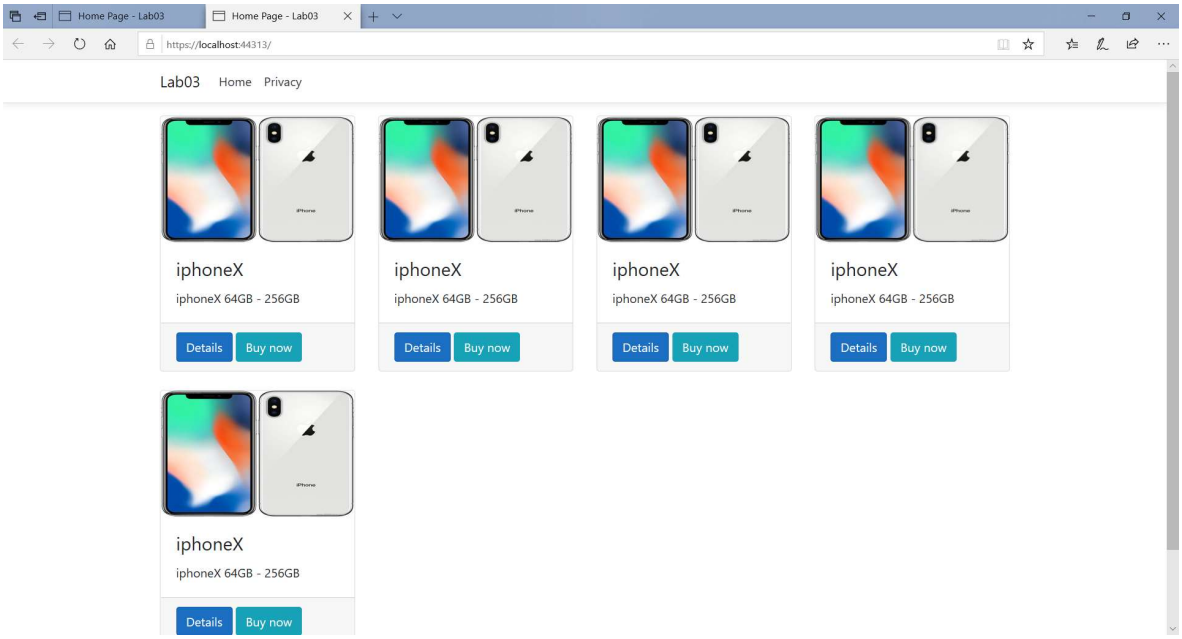
Delete.cshtml

```
@model Lab02.Models.ProductModel

@{
    ViewData["Title"] = "Delete";
}


<h3>Are you sure you want to delete this?</h3>
<div>
    <dl class="row">
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.ProductId)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.ProductId)
        </dd>
        <dt class="col-sm-2">
```

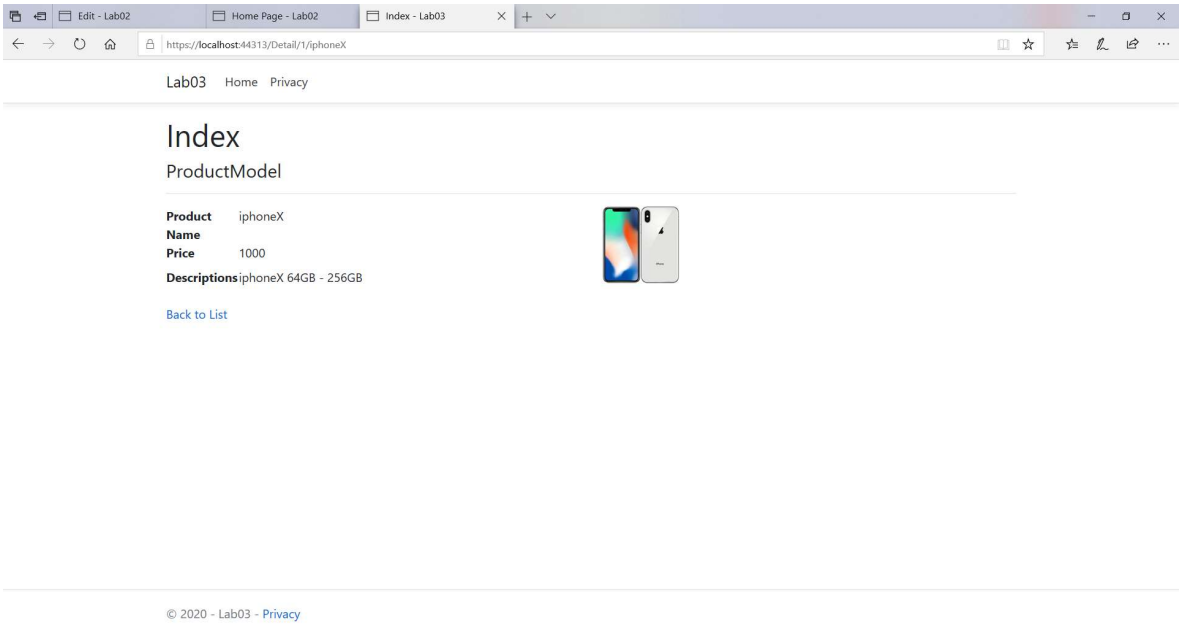
```
            @Html.DisplayNameFor(model => model.ProductName)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.ProductName)
        </dd>

    </dl>

    <form asp-action="Delete">
        <input type="hidden" asp-for="ProductId" />
        <input type="submit" value="Delete" class="btn btn-
danger" /> |
        <a asp-action="Index">Back to List</a>
    </form>
</div>
```

Lab 03: Xây dựng trang HomePage

Trang HomePage



Trang Detail sản phâm



Sử dụng lại Project Lab02 . Xây dựng trang HomePage như hình trên:

Bước 1: Xây dựng giao diện cho HomePage

Index.cshtml

```
@{
```

```
    ViewData["Title"] = "Home Page";
}
@model List<ProductModel>
                <div class="row">
                    @foreach (var item in Model)
                    {
                        <div class="col-lg-3 col-md-6 mb-4">
                            <div class="card h-100">
                                <img class="card-img-top"
src="~/images/@item.ProductImage" alt="">
                                <div class="card-body">
                                    <h4 class="card-
title">@item.ProductName</h4>
                                    <p class="card-
text">@item.Descriptions</p>
                                </div>
                                <div class="card-footer">
                                    <a asp-controller="Detail"
asp-action="Index" asp-route-id="@item.ProductId" asp-route-
name="@item.ProductName" class="btn btn-primary">Details</a>
                                    <a href="#" class="btn btn-
info">Buy now</a>
                                </div>
                            </div>
                        </div>
                    }
                </div>
```

HomeController.cs

```
public class HomeController : Controller
    {

        private ProductData data;

        public HomeController(ProductData productData)
        {
            data = productData;
        }

        public IActionResult Index()
        {
            return View(data.ProductList);
        }

        public IActionResult Privacy()
```

```
        {
            return View();
        }
    }
```

Bước 2: Xây dựng trang Detial

Detail.cshtml

```
@model Lab03.Models.ProductModel

@{
    ViewData["Title"] = "Index";
}

<h1>Index</h1>

<div>
    <h4>ProductModel</h4>
    <hr />
    <div class="row">
        <div class="col-sm-6">
            <dl class="row">
                <dt class="col-sm-2">
                    @Html.DisplayNameFor(model =>
model.ProductName)
                </dt>
                <dd class="col-sm-10">
                    @Html.DisplayFor(model =>
model.ProductName)
                </dd>
                <dt class="col-sm-2">
                    @Html.DisplayNameFor(model =>
model.ProductPrice)
                </dt>
                <dd class="col-sm-10">
                    @Html.DisplayFor(model =>
model.ProductPrice)
                </dd>
                <dt class="col-sm-2">
                    @Html.DisplayNameFor(model =>
model.Descriptions)
                </dt>
                <dd class="col-sm-10">
                    @Html.DisplayFor(model =>
model.Descriptions)
```

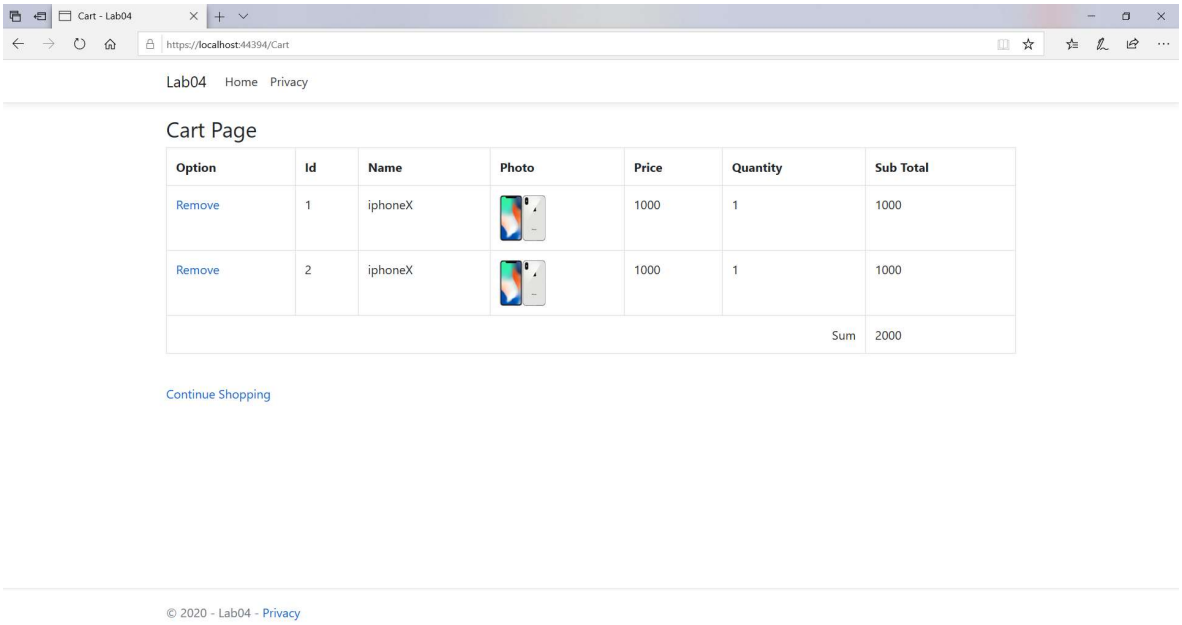
```
                </dd>
            </dl>
        </div>
        <div class="col-sm-4">

                <img src="~/images/@Model.ProductImage"
width="100" height="100" atl="image"/>


        </div>
    </div>

</div>
<div>

    <a asp-action="Index">Back to List</a>
</div>
```

DetailController.cs

```
public class DetailController : Controller
    {
        private readonly ProductData productData;
        public DetailController(ProductData productData)
        {
            this.productData = productData;
        }
        [HttpGet("Detail/{id}/{name}")]
        public IActionResult Index(int id)
        {
            ProductModel productModel =
productData.ProductList.FirstOrDefault(p => p.ProductId == id);
            return View(productModel);
        }
    }
```

Kết hợp Lab02 và Lab03 thành Project hoàn chỉnh

Lab04: Xây dựng trang Giỏ Hàng



Sử dụng Project sau khi hoàn chỉnh Lab02 và Lab03

Bước 1: Tạo thư mục Helper và session

SessionHelper.cs

```
public static class SessionHelper
    {
        public static void SetObjectAsJson(this ISession
session, string key, object value)
        {
            session.SetString(key,
JsonConvert.SerializeObject(value));
        }

        public static T GetObjectFromJson<T>(this ISession
session, string key)
        {
            var value = session.GetString(key);
            return value == null ? default(T) :
JsonConvert.DeserializeObject<T>(value);
        }
    }
```

Bước 2: Tạo CartController và giao diện Cart

CartController.cs

```
  public class CartController : Controller
    {
        private readonly ProductData productData;
        public CartController(ProductData productData)
        {
            this.productData = productData;
        }
        public IActionResult Index()
        {
            var cart =
SessionHelper.GetObjectFromJson<List<ProductToCart>>(HttpContex
t.Session, "cart");
            ViewBag.cart = cart;
            ViewBag.total = cart.Sum(item =>
item.ProductModel.ProductPrice * item.Quantity);
            return View();
        }

        [Route("buy/{id}")]
        public IActionResult Buy(int id)
        {

            if
(SessionHelper.GetObjectFromJson<List<ProductToCart>>(HttpConte
xt.Session, "cart") == null)
            {
                List<ProductToCart> cart = new
List<ProductToCart>();
                cart.Add(new ProductToCart { ProductModel =
productData.ProductList.FirstOrDefault(p => p.ProductId == id),
Quantity = 1 });

SessionHelper.SetObjectAsJson(HttpContext.Session, "cart",
cart);
            }
            else
            {
                List<ProductToCart> cart =
SessionHelper.GetObjectFromJson<List<ProductToCart>>(HttpContex
t.Session, "cart");
                int index = isExist(id);
                if (index != -1)
                {
                    cart[index].Quantity++;
                }
```

```
                else
                {
                    cart.Add(new ProductToCart { ProductModel =
productData.ProductList.FirstOrDefault(p => p.ProductId == id),
Quantity = 1 });
                }

SessionHelper.SetObjectAsJson(HttpContext.Session, "cart",
cart);
            }
            return RedirectToAction("Index");
        }

        [Route("remove/{id}")]
        public IActionResult Remove(int id)
        {
            List<ProductToCart> cart =
SessionHelper.GetObjectFromJson<List<ProductToCart>>(HttpContex
t.Session, "cart");
            int index = isExist(id);
            cart.RemoveAt(index);
            SessionHelper.SetObjectAsJson(HttpContext.Session,
"cart", cart);
            return RedirectToAction("Index");
        }

        private int isExist(int id)
        {
            List<ProductToCart> cart =
SessionHelper.GetObjectFromJson<List<ProductToCart>>(HttpContex
t.Session, "cart");
            for (int i = 0; i < cart.Count; i++)
            {
                if (cart[i].ProductModel.ProductId == id)
                {
                    return i;
                }
            }
            return -1;
        }

    }
```

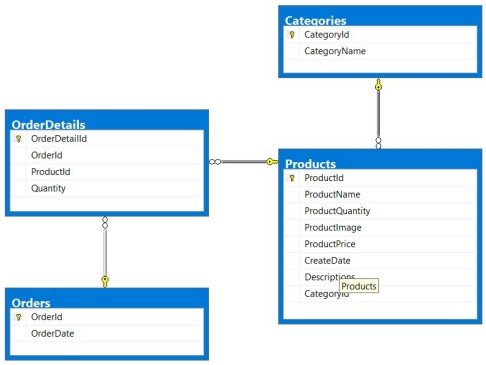
Index.cshtml

```
@{
    ViewData["Title"] = "Cart";
}

<h3>Cart Page</h3>
<table class="table-bordered table">
    <tr>
        <th>Option</th>
        <th>Id</th>
        <th>Name</th>
        <th>Photo</th>
        <th>Price</th>
        <th>Quantity</th>
        <th>Sub Total</th>
    </tr>
    @foreach (var item in ViewBag.cart)
    {
        <tr>
            <td><a asp-controller="cart" asp-action="remove"
asp-route-id="@item.ProductModel.ProductId">Remove</a></td>
            <td>@item.ProductModel.ProductId</td>
            <td>@item.ProductModel.ProductName</td>
            <td><img
src="~/images/@item.ProductModel.ProductImage" width="60"
height="60" /> </td>
            <td>@item.ProductModel.ProductPrice</td>
            <td>@item.Quantity</td>
            <td>@(item.ProductModel.ProductPrice *
item.Quantity)</td>
        </tr>
    }
    <tr>
        <td align="right" colspan="6">Sum</td>
        <td>
            @ViewBag.total
        </td>
    </tr>
</table>
<br>
<a asp-controller="Home" asp-action="index">Continue
Shopping</a>
```

Bước 3: Cập nhật Index của HomePage

```
@{
    ViewData["Title"] = "Home Page";
}
@model List<ProductModel>
<div class="row">
    @foreach (var item in Model)
    {
        <div class="col-lg-3 col-md-6 mb-4">
            <div class="card h-100">
                <img class="card-img-top"
src="~/images/@item.ProductImage" alt="">
                <div class="card-body">
                    <h4 class="card-
title">@item.ProductName</h4>
                    <p class="card-text">@item.Descriptions</p>
                </div>
                <div class="card-footer">
                     <a asp-controller="Detail" asp-
action="Index" asp-route-id="@item.ProductId" asp-route-
name="@item.ProductName" class="btn btn-primary">Details</a>

                    <a asp-controller="Cart" asp-action="Buy"
asp-route-id="@item.ProductId" class="btn btn-info">Buy now</a>
                </div>
            </div>
        </div>
    }
</div>
```

Lab05: Làm việc với EntityFramwork Core



Sử dụng lại Project ở Lab04

Bước 1: trong thư mục View/Domain, tạo các model sau:

Product.cs

```
  public class Product
    {
        public int ProductId { get; set; }
        public string ProductName { get; set; }
        public int ProductQuantity { get; set; }
        public string ProductImage { get; set; }
        public double ProductPrice { get; set; }
        public DateTime CreateDate { get; set; }
        public string Descriptions { get; set; }
        public Category Category { get; set; }
        public int CategoryId { get; set; }

    }
```

Category.cs

```
public class Category
    {
        public int CategoryId { get; set; }
        public string CategoryName { get; set; }
        public List<Product> Products { get; set; }
    }
```

Order.cs

```
public class Order
```

```
    {
        public int OrderId { get; set; }
        public DateTime OrderDate { get; set; }
        public List<OrderDetail> OrderDetails { get; set; }
    }
```

OrderDetail.cs

```
public class OrderDetail
    {
        public int OrderDetailId { get; set; }
        public int OrderId { get; set; }
        public int ProductId { get; set; }
        public int Quantity { get; set; }
        public Order Order { get; set; }
        public Product Product { get; set; }
    }
```

Bước 2: cài đặt các gói sau: (Sử dụng Package Manager Console)

Install-Package Microsoft.EntityFrameworkCore -Version 3.1.2

Install-Package Microsoft.EntityFrameworkCore.SqlServer

Install-Package Microsoft.EntityFrameworkCore.Tools

Bước 3:Trong thư mục Data, tạo DataContext sau:

DataContext.cs

```
public class DataContext:DbContext
    {
        public DataContext(DbContextOptions<DataContext>
options):base(options)
        {

        }

        public DbSet<Category> Categories { get; set; }
        public DbSet<Product> Products { get; set; }
        public DbSet<Order> Orders { get; set; }
        public DbSet<OrderDetail> OrderDetails { get; set; }

        protected override void OnModelCreating(ModelBuilder
builder)
        {
            base.OnModelCreating(builder);
            builder.Entity<Category>().HasData(
```

```
            new Category()
            {
                CategoryId = 1,
                CategoryName = "Iphone"
            },
            new Category()
            {
                CategoryId = 2,
                CategoryName = "SamSung"
            });
        builder.Entity<Product>().HasData(
            new Product()
            {
                ProductId = 1,
                ProductName = "IphoneX",
                ProductImage = "iphoneX.png",
                Descriptions = "iphoneX 64GB - 256GB",
                ProductQuantity = 100,
                ProductPrice = 1000.00,
                CreateDate = DateTime.Now,
                CategoryId = 1

            },
             new Product()
             {
                 ProductId = 2,
                 ProductName = "IphoneX",
                 ProductImage = "iphoneX.png",
                 Descriptions = "iphoneX 64GB - 256GB",
                 ProductQuantity = 100,
                 ProductPrice = 1000.00,
                 CreateDate = DateTime.Now,
                 CategoryId = 1

             },
              new Product()
              {
                  ProductId = 3,
                  ProductName = "IphoneX",
                  ProductImage = "iphoneX.png",
                  Descriptions = "iphoneX 64GB - 256GB",
                  ProductQuantity = 100,
                  ProductPrice = 1000.00,
                  CreateDate = DateTime.Now,
                  CategoryId = 1
```

```
                },
                 new Product()
                 {
                     ProductId = 4,
                     ProductName = "SamSung",
                     ProductImage = "samsung.png",
                     Descriptions = "samsung 64GB - 256GB",
                     ProductQuantity = 100,
                     ProductPrice = 1000.00,
                     CreateDate = DateTime.Now,
                     CategoryId = 2

                 });
        }
    }
```

Bước 4: Thêm ConnectString trong appsetting.json

appsetting.json

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=DESKTOP-MQB7M9K\\SQLEXPRESS;
Database=ShoppingDB; User=sa; Password=123456"
  },
    "Logging": {
      "LogLevel": {
        "Default": "Information",
        "Microsoft": "Warning",
        "Microsoft.Hosting.Lifetime": "Information"
      }
    },
    "AllowedHosts": "*"
  }
```

Bước 5: Cấu hình Startup.cs để sử dụng EF Core

```
public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();
            services.AddSingleton(p=>ProductData.initData());
            services.AddSession();
            services.AddDbContext<DataContext>(options =>
options.UseSqlServer(Configuration.GetConnectionString("Default
Connection")));
        }
```

Bước 6: Thực hiện migration (Lưu ý: phải build project trước khi thực hiện)

Trong Package Manager Console thực hiện :

```
Add-Migration InitData // tạo thư mục Migration
Update-Database // tạo database
Add-Migration SeedData // thêm dữ liệu mẫu
Update-Database // cập nhật thay đổi database
```

Bước 7: Kiểm tra trong SqlServer

Lab06: Thực hiện lại các thao tác CRUD với CSDL

Sử dụng lại Project Lab05

Bước 1: cài đặt AutoMapper vào Project

Install-Package AutoMapper.Extensions.Microsoft.DependencyInjection

Bước 2: Cập nhật lại Startup.cs

```
public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();
            services.AddSingleton(p=>ProductData.initData());
            services.AddSession();
            services.AddDbContext<DataContext>(options =>
options.UseSqlServer(Configuration.GetConnectionString("Default
Connection")));
            services.AddAutoMapper(typeof(Startup));
        }
```

Bước 3: Cập nhật lại ProductController

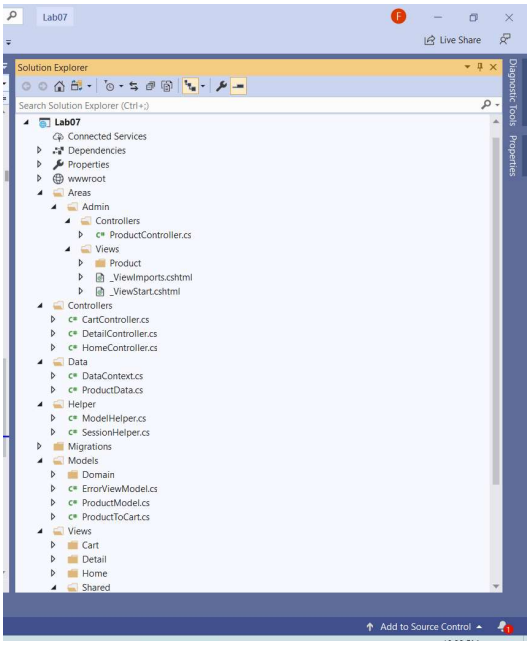
```
public class ProductController : Controller
    {
        ProductData productData;
        DataContext dataContext;
        IMapper mapper;
        public ProductController(ProductData productData,
DataContext dataContext, IMapper mapper)
        {
            this.productData = productData;
            this.dataContext = dataContext;
            this.mapper = mapper;
        }
        [HttpGet]
        public IActionResult Index()
        {
            List<ProductModel> products = new
List<ProductModel>();
            List<Product> productss =
dataContext.Products.ToList();
            foreach (var item in productss)
            {
                products.Add(mapper.Map<ProductModel>(item));
            }
            return View(products);
        }
```

```
//....
}
```

Thực hiện tiếp với các yêu cầu còn lại như Trang HomePage, Delete, Edit

Lab07: Xây dựng trang Admin với Areas

Sử dụng lại Project tại Lab06, sau đó thay đổi cấu trúc thư mục như hình sau



Bước 1: Cập nhật lai Startup.cs

```
public void Configure(IApplicationBuilder app,
IWebHostEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }
            else
            {
                app.UseExceptionHandler("/Home/Error");
            }
            app.UseStaticFiles();
            app.UseSession();
            app.UseRouting();

            app.UseAuthorization();

            app.UseEndpoints(endpoints =>
            {
                endpoints.MapAreaControllerRoute(
                    name: "MyArea",
                    areaName: "Admin",
```

```
                    pattern:
"{area:exists}/{controller=Product}/{action=Index}/{id?}");

                endpoints.MapControllerRoute(
                    name: "default",
                    pattern:
"{controller=Home}/{action=Index}/{id?}");


                //endpoints.MapControllers();

            });
        }
```

Bước 2: Cập nhật Areas/Admin/Controllers/Product

```
[Area("Admin")]
    public class ProductController : Controller
    {
        ProductData productData;
        DataContext dataContext;
        IMapper mapper;
        public ProductController(ProductData productData,
DataContext dataContext, IMapper mapper)
        {
            this.productData = productData;
            this.dataContext = dataContext;
            this.mapper = mapper;
        }

        public IActionResult Index()
        {
            List<ProductModel> products = new
List<ProductModel>();
            List<Product> productss =
dataContext.Products.ToList();
            foreach (var item in productss)
            {
                products.Add(mapper.Map<ProductModel>(item));
            }
            return View(products);
        }

//...
```

Bước 3: Cập nhật _Layout.cshtml

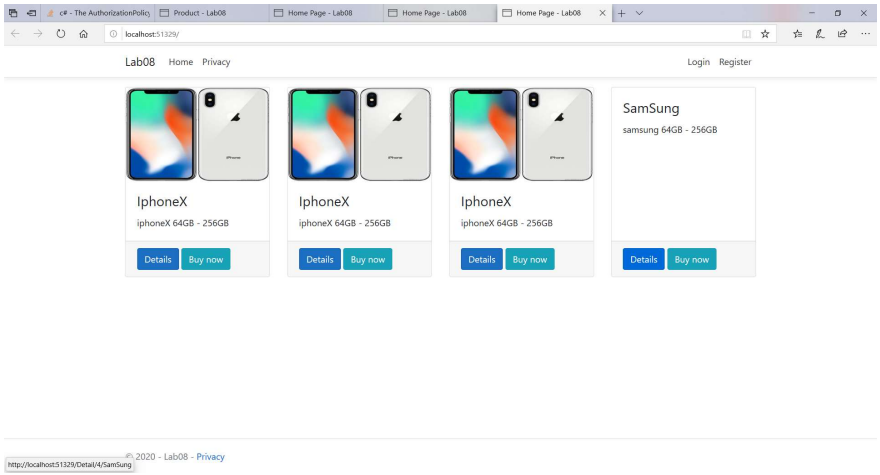
//...

```
   <li class="nav-item">
                              <a class="nav-link text-dark" asp-
area="Admin" asp-controller="Product" asp-
action="Index">Product</a>
                           </li>
```

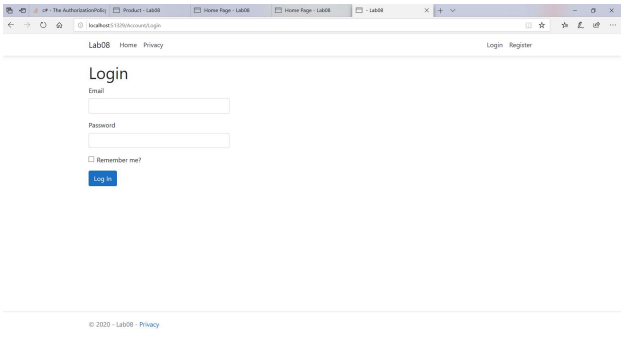
//..

Bước 4: Chạy kiểm tra kết quả

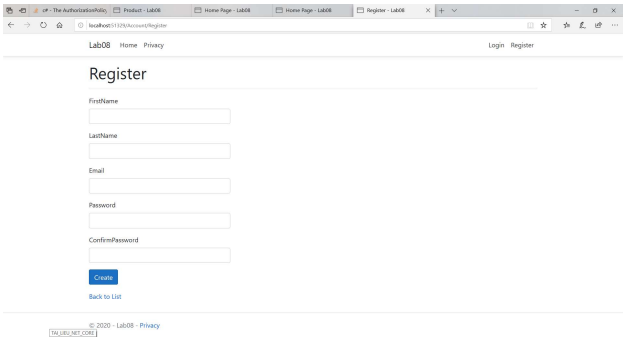
Lab 08: Làm việc với Identity



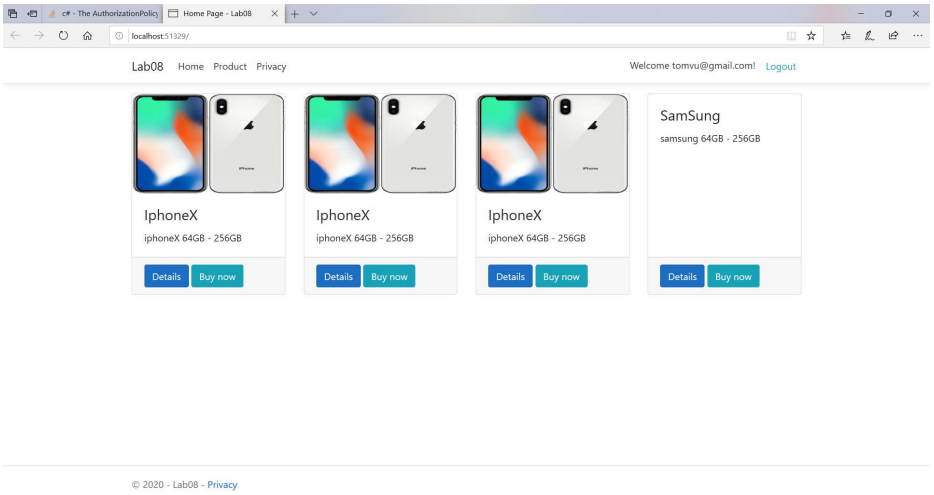
Giao diện Login:



Giao diện Register



Giao diện sau khi Login



Bước 1: Tạo User.cs trong Models/Domain

```
public class User:IdentityUser
    {
        public string FirstName { get; set; }
        public string LastName { get; set; }
    }
```

Bước 2: Cập nhật DataContext.cs

```
public class DataContext: IdentityDbContext<User>
    {
        public DataContext(DbContextOptions<DataContext>
options):base(options)
        {

        }
```

```
        public DbSet<Category> Categories { get; set; }
        public DbSet<Product> Products { get; set; }
        public DbSet<Order> Orders { get; set; }
        public DbSet<OrderDetail> OrderDetails { get; set; }

        protected override void OnModelCreating(ModelBuilder
builder)
        {
            base.OnModelCreating(builder);
            builder.Entity<Category>().HasData(
                new Category()
                {
                    CategoryId = 1,
                    CategoryName = "Iphone"
                },
                new Category()
                {
                    CategoryId = 2,
                    CategoryName = "SamSung"
                });
            builder.Entity<Product>().HasData(
                new Product()
                {
                    ProductId = 1,
                    ProductName = "IphoneX",
                    ProductImage = "iphoneX.png",
                    Descriptions = "iphoneX 64GB - 256GB",
                    ProductQuantity = 100,
                    ProductPrice = 1000.00,
                    CreateDate = DateTime.Now,
                    CategoryId = 1

                },
                new Product()
                 {
                     ProductId = 2,
                     ProductName = "IphoneX",
                     ProductImage = "iphoneX.png",
                     Descriptions = "iphoneX 64GB - 256GB",
                     ProductQuantity = 100,
                     ProductPrice = 1000.00,
                     CreateDate = DateTime.Now,
```

```
                    CategoryId = 1

                },
                new Product()
                {
                    ProductId = 3,
                    ProductName = "IphoneX",
                    ProductImage = "iphoneX.png",
                    Descriptions = "iphoneX 64GB - 256GB",
                    ProductQuantity = 100,
                    ProductPrice = 1000.00,
                    CreateDate = DateTime.Now,
                    CategoryId = 1

                },
                new Product()
                {
                    ProductId = 4,
                    ProductName = "SamSung",
                    ProductImage = "samsung.png",
                    Descriptions = "samsung 64GB - 256GB",
                    ProductQuantity = 100,
                    ProductPrice = 1000.00,
                    CreateDate = DateTime.Now,
                    CategoryId = 2

                });
            builder.Entity<IdentityRole>().HasData(
                new IdentityRole
            {
                Name = "Visitor",
                NormalizedName = "VISITOR"
            },
                new IdentityRole
            {
                Name = "Administrator",
                NormalizedName = "ADMINISTRATOR"
            });

        }
    }
```

Bước 3: Thực hiện thêm cấu hình Identity trong Startup.cs

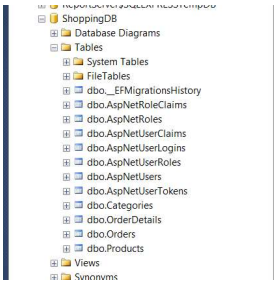
```
public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();
            services.AddSingleton(p=>ProductData.initData());
            services.AddSession();
            services.AddDbContext<DataContext>(options =>
options.UseSqlServer(Configuration.GetConnectionString("DefaultCo
nnection")));
            services.AddAutoMapper(typeof(Startup));

            services.AddIdentity<User, IdentityRole>( opt =>
            {
                opt.Password.RequiredLength = 8;
                opt.Password.RequireDigit = false;
                opt.Password.RequireUppercase = false;
                opt.Password.RequireNonAlphanumeric = false;
            }).AddEntityFrameworkStores<DataContext>();

           //
services.AddScoped<IUserClaimsPrincipalFactory<User>,
CustomClaimsFactory>();

        }
```

Bước 4: Thực hiện Migration tạo Identity cho Database

Add-Migration CreatingIdentity

Update-database

Kiểm tra trong SqlServer

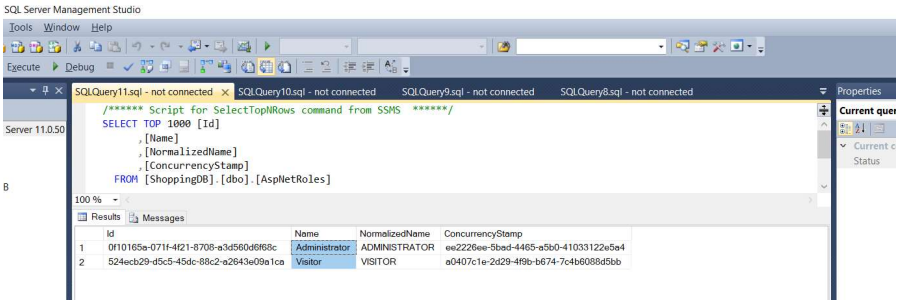


Bước 5: Thực hiện Migration để tạo Role cho User

Add-Migration InsertedRoles

Update-database

Kiểm tra trong SqlServer



Bước 6: Xây dựng AccountController.cs

```
public class AccountController : Controller
    {
        private readonly IMapper _mapper;
        private readonly UserManager<User> _userManager;
        private readonly SignInManager<User> _signInManager;

        public AccountController(IMapper mapper,
UserManager<User> userManager, SignInManager<User> signInManager)
        {
            _mapper = mapper;
            _userManager = userManager;
            _signInManager = signInManager;
        }

        [HttpGet]
        public IActionResult Register()
        {
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult>
Register(UserRegistrationModel userModel)
        {
            if (!ModelState.IsValid)
            {
                return View(userModel);
            }
```

```
            var user = _mapper.Map<User>(userModel);

            var result = await _userManager.CreateAsync(user,
userModel.Password);
            if (!result.Succeeded)
            {
                foreach (var error in result.Errors)
                {
                    ModelState.TryAddModelError(error.Code,
error.Description);
                }

                return View(userModel);
            }

            await _userManager.AddToRoleAsync(user, "Visitor");

            return RedirectToAction(nameof(HomeController.Index),
"Home");
        }

        [HttpGet]
        public IActionResult Login(string returnUrl = null)
        {
            ViewData["ReturnUrl"] = returnUrl;
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Login(UserLoginModel
userModel, string returnUrl = null)
        {
            if (!ModelState.IsValid)
            {
                return View(userModel);
            }

            var result = await
_signInManager.PasswordSignInAsync(userModel.Email,
userModel.Password, userModel.RememberMe, false);
```

```
            if (result.Succeeded)
            {
                return RedirectToLocal(returnUrl);
            }
            else
            {
                ModelState.AddModelError("", "Invalid UserName or
Password");
                return View();
            }
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Logout()
        {
            await _signInManager.SignOutAsync();

            return RedirectToAction(nameof(HomeController.Index),
"Home");
        }

        private IActionResult RedirectToLocal(string returnUrl)
        {
            if (Url.IsLocalUrl(returnUrl))
                return Redirect(returnUrl);
            else
                return
RedirectToAction(nameof(HomeController.Index), "Home");
        }
    }
```

Bước 7: Giao diện Register.cshtml

```
@model Lab08.Models.UserRegistrationModel

@{
    ViewData["Title"] = "Register";
}

<h1>Register</h1>

<hr />
```

```
<div class="row">
    <div class="col-md-4">
        <form asp-action="Register">
            <div asp-validation-summary="ModelOnly" class="text-
danger"></div>
            <div class="form-group">
                <label asp-for="FirstName" class="control-
label"></label>
                <input asp-for="FirstName" class="form-control"
/>
                <span asp-validation-for="FirstName" class="text-
danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="LastName" class="control-
label"></label>
                <input asp-for="LastName" class="form-control" />
                <span asp-validation-for="LastName" class="text-
danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Email" class="control-
label"></label>
                <input asp-for="Email" class="form-control" />
                <span asp-validation-for="Email" class="text-
danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Password" class="control-
label"></label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-
danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="ConfirmPassword" class="control-
label"></label>
                <input asp-for="ConfirmPassword" class="form-
control" />
                <span asp-validation-for="ConfirmPassword"
class="text-danger"></span>
            </div>
```

```
            <div class="form-group">
                <input type="submit" value="Create" class="btn
btn-primary" />
            </div>
        </form>
    </div>
</div>

<div>
    <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
    @{await
Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

Bước 8: Giao diện Login.cshtml

```
@model Lab08.Models.UserLoginModel

<h1>Login</h1>

<div class="row">
    <div class="col-md-4">
        <form asp-action="Login" asp-route-
returnUrl="@ViewData["ReturnUrl"]">
            <div asp-validation-summary="ModelOnly" class="text-
danger"></div>
            <div class="form-group">
                <label asp-for="Email" class="control-
label"></label>
                <input asp-for="Email" class="form-control" />
                <span asp-validation-for="Email" class="text-
danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Password" class="control-
label"></label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-
danger"></span>
```

```
            </div>
            <div class="form-group form-check">
                <label class="form-check-label">
                    <input class="form-check-input" asp-
for="RememberMe" /> @Html.DisplayNameFor(model =>
model.RememberMe)
                </label>
            </div>
            <div class="form-group">
                <input type="submit" value="Log In" class="btn
btn-primary" />
            </div>
        </form>
    </div>
</div>

@section Scripts {
    @{await
Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

Bước 9: Giao diện _LoginPartial.cshtml

```
@using Microsoft.AspNetCore.Identity
@inject SignInManager<User> SignInManager
@inject UserManager<User> UserManager

<ul class="navbar-nav">
    @if (SignInManager.IsSignedIn(User))
    {
        <li class="nav-item">
            <a class="nav-link text-dark" asp-controller="Home"
asp-action="Index"
               title="Welcome">Welcome @User.Identity.Name!</a>
        </li>
        <li class="nav-item">
            <form class="form-inline" asp-controller="Account"
asp-action="Logout" asp-area="">
                <button type="submit" class="nav-link btn btn-
link text-info">Logout</button>
            </form>
        </li>
    }
```

```
    else
    {
        <li class="nav-item">
            <a class="nav-link text-dark" asp-
controller="Account"
               asp-action="Login">Login</a>
        </li>
        <li class="nav-item">
            <a class="nav-link text-dark" asp-
controller="Account"
               asp-action="Register">Register</a>
        </li>
    }
</ul>
```

Bước 10: Cập nhật giao diện cho HomePage

//..

```
<header>
        <nav class="navbar navbar-expand-sm navbar-toggleable-sm
navbar-light bg-white border-bottom box-shadow mb-3">
            <div class="container">
                <a class="navbar-brand" asp-area="" asp-
controller="Home" asp-action="Index">Lab08</a>
                <button class="navbar-toggler" type="button"
data-toggle="collapse" data-target=".navbar-collapse" aria-
controls="navbarSupportedContent"
                        aria-expanded="false" aria-label="Toggle
navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="navbar-collapse collapse d-sm-inline-
flex flex-sm-row-reverse">
                    <partial name="_LoginPartial" />
                    <ul class="navbar-nav flex-grow-1">
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-
area="" asp-controller="Home" asp-action="Index">Home</a>
                        </li>

                        @if (User.Identity.IsAuthenticated)
                        {
```

```
                            <li class="nav-item">
                                <a class="nav-link text-dark"
asp-area="Admin" asp-controller="Product" asp-
action="Index">Product</a>
                            </li>
                        }

                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-
area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
                        </li>
                    </ul>
                </div>
```

//..

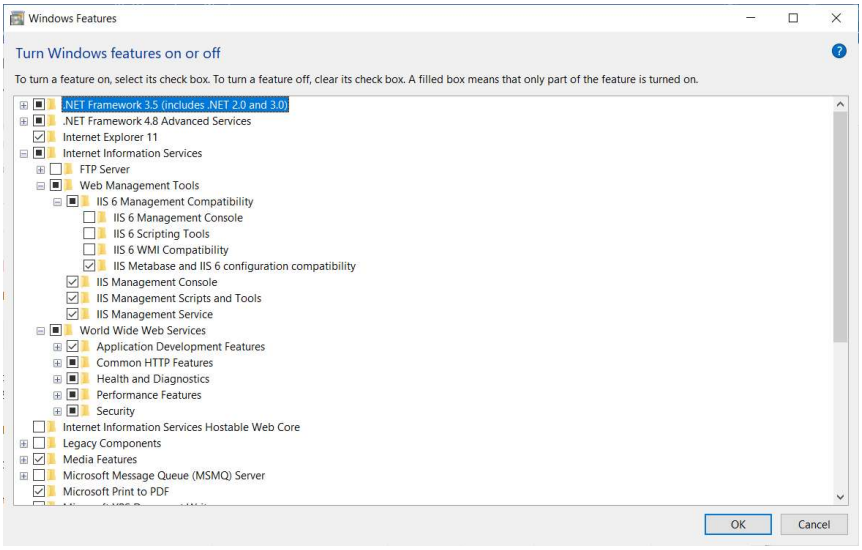
Lab09: Publish ứng dụng

Cách 1: IIS

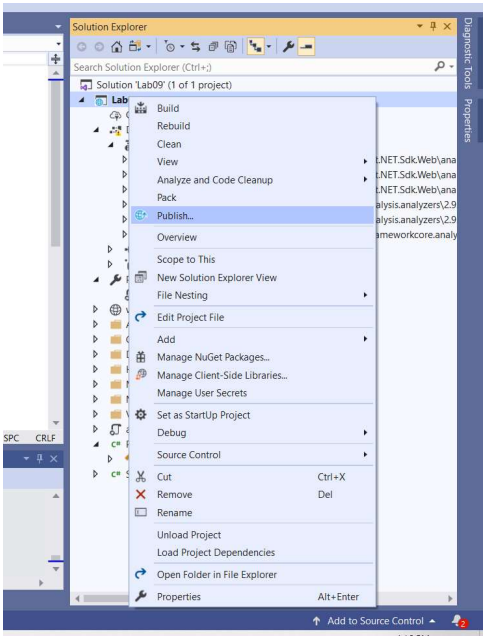
Bước 1: Download ASP.NET Core Runtime 3.1.1 (Hosting bundle): https://dotnet.microsoft.com/download/dotnet-core/thank-you/runtime-aspnetcore-3.1.1-windows-hosting-bundle-installer

Bước 2: Cài đặt IIS

Control Panel → Programs and Features → Turn Windows Features on or off → Internet Infomation Services (Cấu hình như hình dưới)

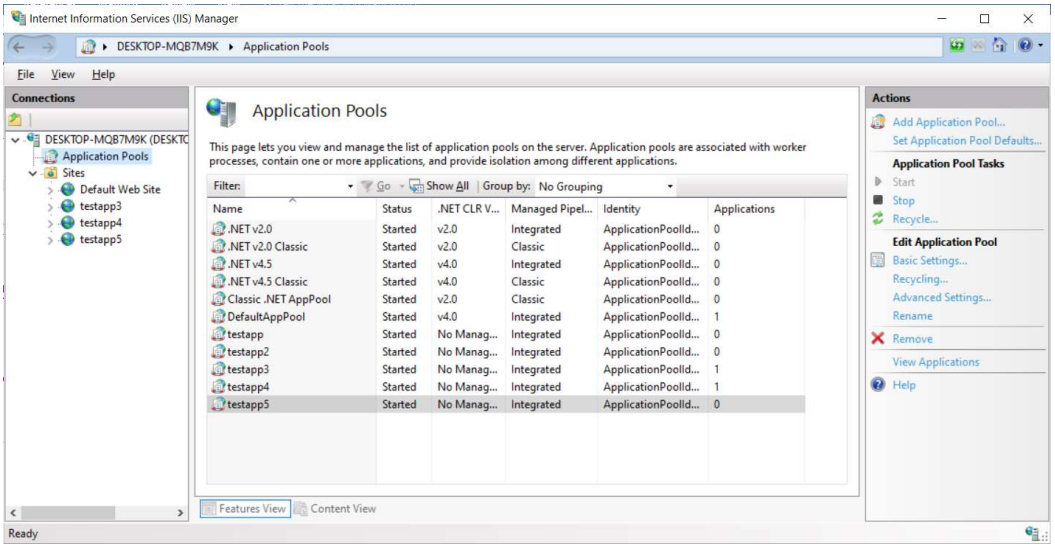


Bước 3: Publish ứng dụng vào thư mục chỉ định (tùy chọn)



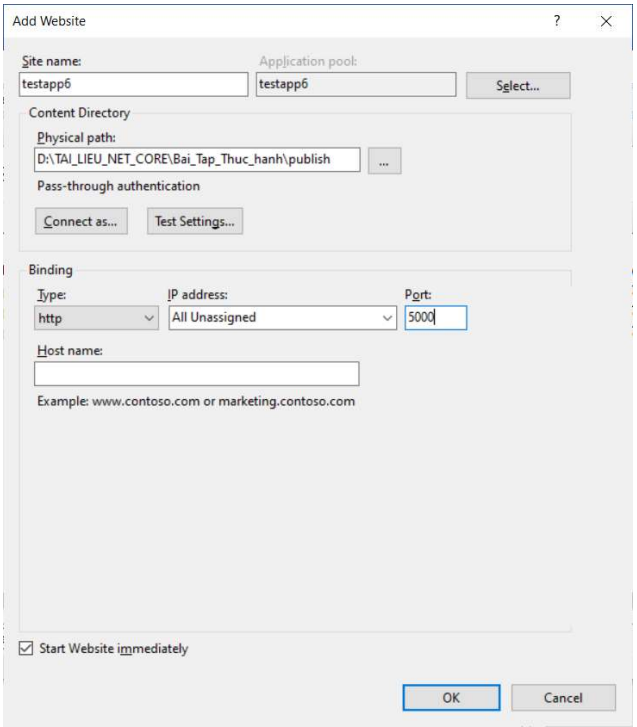
Bước 5: Thiết lập cấu hình IIS:

Control pannel → Administrative tool → Internet Infomation services(IIS)



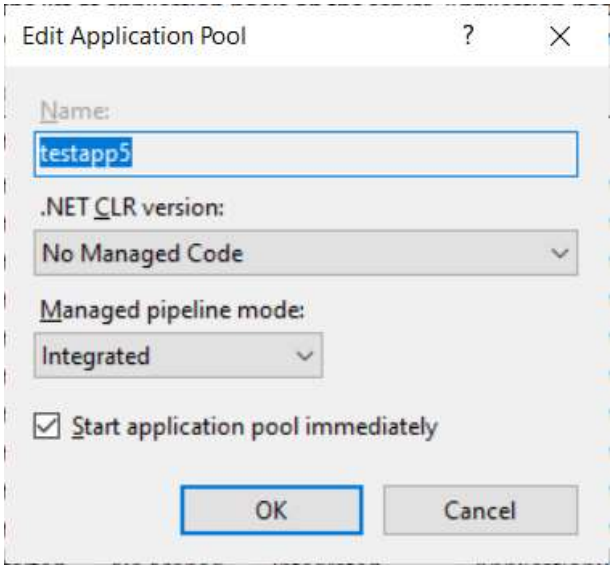
Bước 5 .1: add site

- Right Click Sites
- thêm các thông tin như hình bên
- Chú ý đường dẫn đến thư mục Publish
- Tinh chỉnh Port : 5000



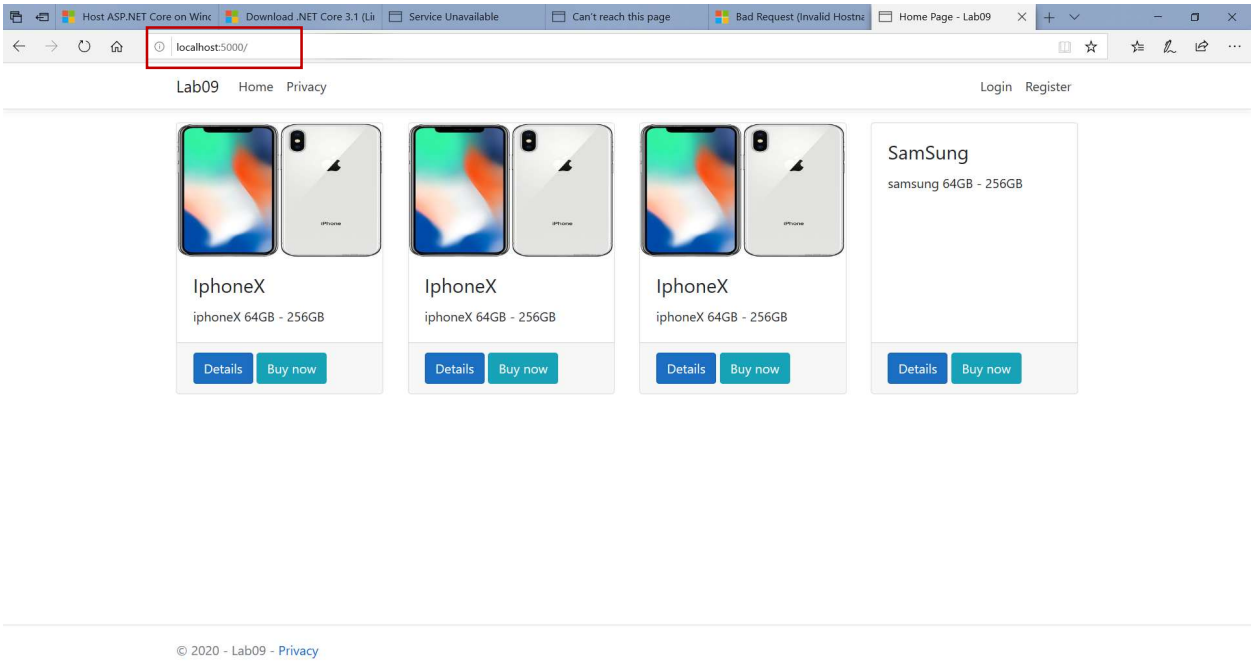
Bước 5.2: Thay cập nhật Application Pools

- Chon Application Pools
- Right clich vào tên site mới tạo → Basic settings
- Tinh chỉnh như hình bên



Bước 6: Kiểm tra

Right Click vào site vừa tạo trong Sites →Manage Website → Browse

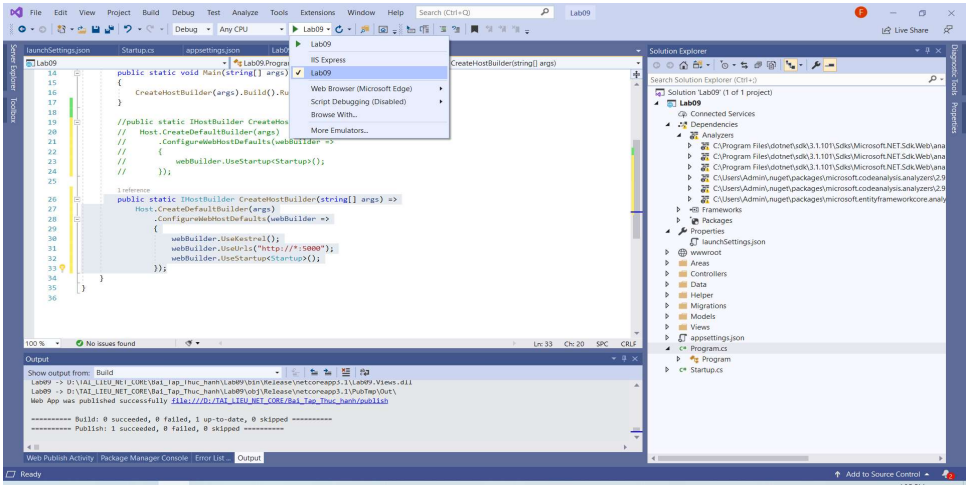


Cách 2: Dùng Hosting URL

Bước 1: Tinh chỉnh cấu hình trong Program.cs

```
public static IHostBuilder CreateHostBuilder(string[] args) =>
        Host.CreateDefaultBuilder(args)
            .ConfigureWebHostDefaults(webBuilder =>
            {
                webBuilder.UseKestrel();
                webBuilder.UseUrls("http://*:5000");
                webBuilder.UseStartup<Startup>();
            });
```

Bước 2: Run chương trình theo cấu hình sau:



Bước 3: Kiểm tra