The source code so long, past top 3 code and I attached the like source for whole code

HomController

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using OnlineShop.Data;
using OnlineShop.Models;
using OnlineShop.Utility;
using X.PagedList;

namespace OnlineShop.Controllers
{
    [Area("Customer")]

    public class HomeController : Controller
    {
        private ApplicationDbContext _db;

        public HomeController(ApplicationDbContext db)
        {
            _db = db;
        }


        public IActionResult Index(int? page)
        {
            return
View(_db.Products.Include(c=>c.ProductTypes).Include(c=>c.SpecialTag).ToList().
ToPagedList(page??1,9));
        }

        public IActionResult Privacy()
        {
            return View();
        }

        [ResponseCache(Duration = 0, Location = ResponseCacheLocation.None,
NoStore = true)]
        public IActionResult Error()
        {
            return View(new ErrorViewModel { RequestId = Activity.Current?.Id
?? HttpContext.TraceIdentifier });
        }

        //GET product detail acation method

        public ActionResult Detail(int? id)
        {

            if(id==null)
            {
                return NotFound();
            }
```

```
            var product = _db.Products.Include(c =>
c.ProductTypes).FirstOrDefault(c => c.Id == id);
            if(product==null)
            {
                return NotFound();
            }
            return View(product);
        }

        //POST product detail acation method
        [HttpPost]
        [ActionName("Detail")]
        public ActionResult ProductDetail(int? id)
        {
            List<Products>products=new List<Products>();
            if (id == null)
            {
                return NotFound();
            }

            var product = _db.Products.Include(c =>
c.ProductTypes).FirstOrDefault(c => c.Id == id);
            if (product == null)
            {
                return NotFound();
            }

            products = HttpContext.Session.Get<List<Products>>("products");
            if(products==null)
            {
                products=new List<Products>();
            }
            products.Add(product);
            HttpContext.Session.Set("products", products);
            return RedirectToAction(nameof(Index));
        }
        //GET Remove action methdo
        [ActionName("Remove")]
        public IActionResult RemoveToCart(int? id)
        {
            List<Products> products =
HttpContext.Session.Get<List<Products>>("products");
            if (products != null)
            {
                var product = products.FirstOrDefault(c => c.Id == id);
                if (product != null)
                {
                    products.Remove(product);
                    HttpContext.Session.Set("products", products);
                }
            }
            return RedirectToAction(nameof(Index));
        }

        [HttpPost]

        public IActionResult Remove(int? id)
        {
            List<Products> products =
HttpContext.Session.Get<List<Products>>("products");
            if(products!=null)
```

```
            {
                var product = products.FirstOrDefault(c => c.Id == id);
                if(product!=null)
                {
                    products.Remove(product);
                    HttpContext.Session.Set("products", products);
                }
            }
            return RedirectToAction(nameof(Index));
        }

        //GET product Cart action method

        public IActionResult Cart()
        {
            List<Products> products =
HttpContext.Session.Get<List<Products>>("products");
            if(products==null)
            {
                products=new List<Products>();
            }
            return View(products);
        }

    }
}
```

LoginPartial

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

<ul class="navbar-nav">
@if (SignInManager.IsSignedIn(User))
{
    <li class="nav-item">
        <a  class="nav-link text-dark" asp-area="Identity" asp-
page="/Account/Manage/Index" title="Manage"> Hello @User.Identity.Name!</a>
    </li>
    <li class="nav-item">
        <form  class="form-inline" asp-area="Identity" asp-
page="/Account/Logout" asp-route-returnUrl="@Url.Action("Index", "Home", new {
area = "" })">
            <button  type="submit" class="nav-link btn btn-link text-
dark">Logout</button>
        </form>
    </li>
}
else
{
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="Customer" asp-controller="User"
asp-action="Create">Register</a>
    </li>
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="Identity" asp-
page="/Account/Login">Login</a>
    </li>
}
</ul>
```

Startup

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Identity.UI;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.HttpsPolicy;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using OnlineShop.Data;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using OnlineShop.Models;

namespace OnlineShop
{
    public class Startup
    {
        public Startup(IConfiguration configuration)
        {
            Configuration = configuration;
        }

        public IConfiguration Configuration { get; }

        // This method gets called by the runtime. Use this method to add
services to the container.
        public void ConfigureServices(IServiceCollection services)
        {
            services.Configure<CookiePolicyOptions>(options =>
            {
                // This lambda determines whether user consent for non-
essential cookies is needed for a given request.
                options.CheckConsentNeeded = context => true;
                options.MinimumSameSitePolicy = SameSiteMode.None;
            });

            services.AddSession(options =>
            {
                // Set a short timeout for easy testing.
                options.IdleTimeout = TimeSpan.FromMinutes(30);
                //options.Cookie.HttpOnly = true;
                // Make the session cookie essential
                options.Cookie.IsEssential = true;
            });

            services.AddDbContext<ApplicationDbContext>(options =>
                options.UseSqlServer(
                    Configuration.GetConnectionString("DefaultConnection")));
            services.AddIdentity<IdentityUser,IdentityRole>()
                .AddDefaultUI(UIFramework.Bootstrap4)
                .AddEntityFrameworkStores<ApplicationDbContext>();


services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_2);
        }
```

```
        // This method gets called by the runtime. Use this method to configure
the HTTP request pipeline.
        public void Configure(IApplicationBuilder app, IHostingEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
                app.UseDatabaseErrorPage();
            }
            else
            {
                app.UseExceptionHandler("/Home/Error");
                // The default HSTS value is 30 days. You may want to change
this for production scenarios, see https://aka.ms/aspnetcore-hsts.
                app.UseHsts();
            }

            app.UseHttpsRedirection();
            app.UseStaticFiles();
            app.UseCookiePolicy();
            app.UseSession();
            app.UseAuthentication();

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