Table of Contents

[Section 13: Advance Concepts 2](#_Toc145500194)

[Authorization 2](#_Toc145500195)

[Session in .NET Core 3](#_Toc145500196)

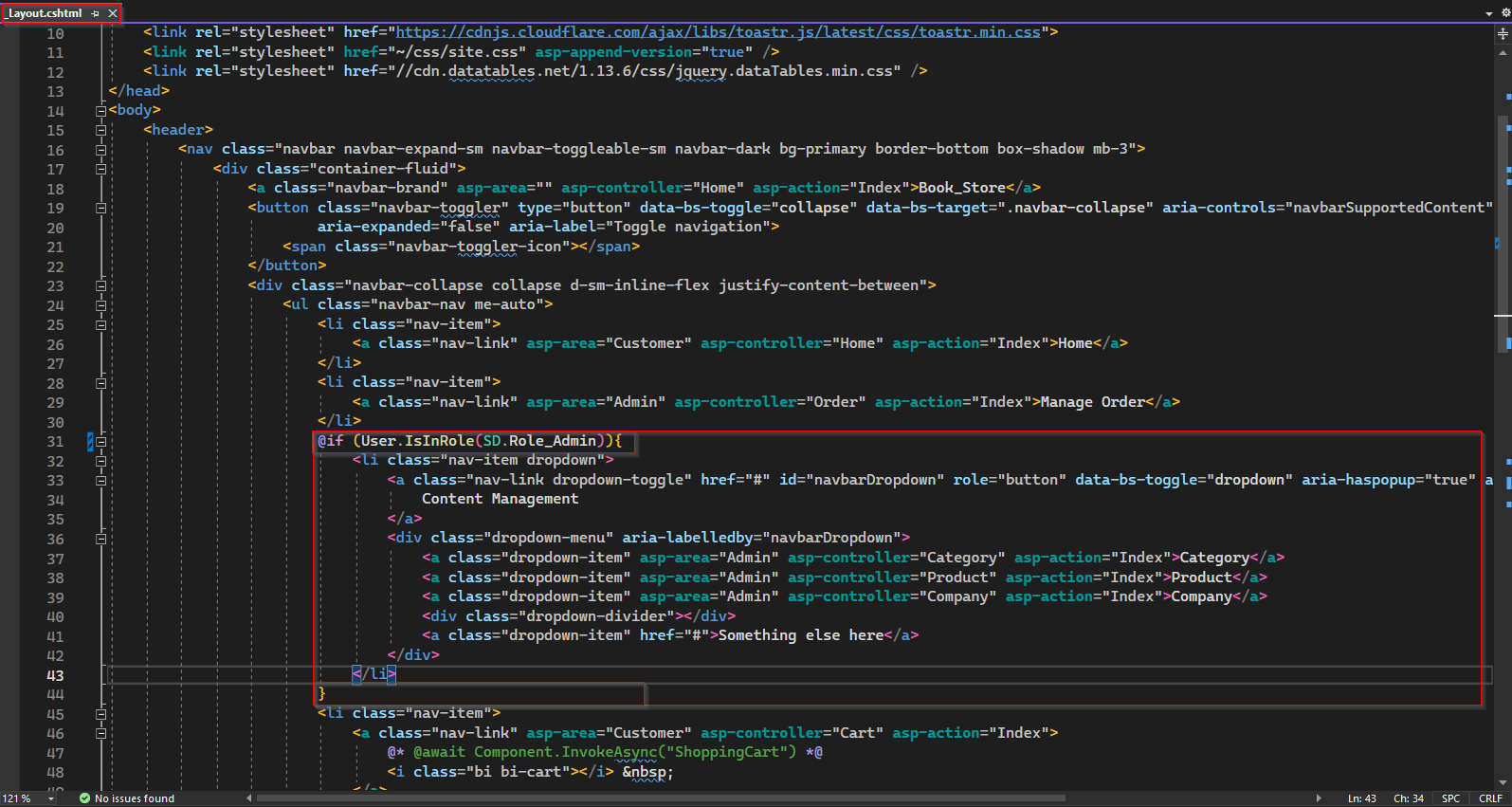
[Configure Session in .NET Core 3](#_Toc145500197)

[Set and Access Session 4](#_Toc145500198)

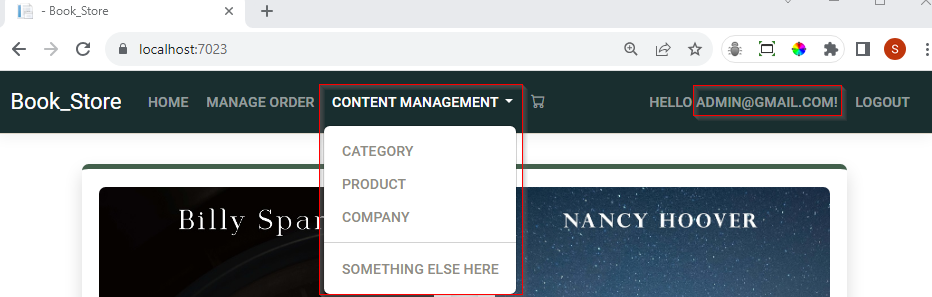
[View Component 5](#_Toc145500199)

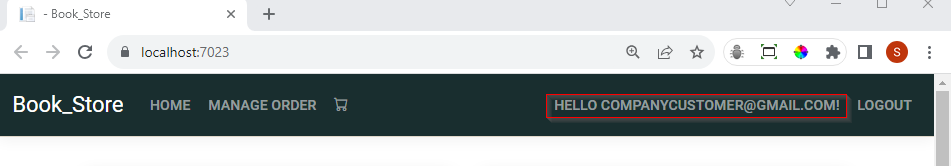
# Section 13: Advance Concepts

## Authorization



To provide CRUD operation on Category, Product, Company, we can place "Content Management" drop down list of navigation bar inside if condition with statement verifying if user has role "Admin". Hence the "Content Management" will only be visible in user interface if user is under the role of "Admin".



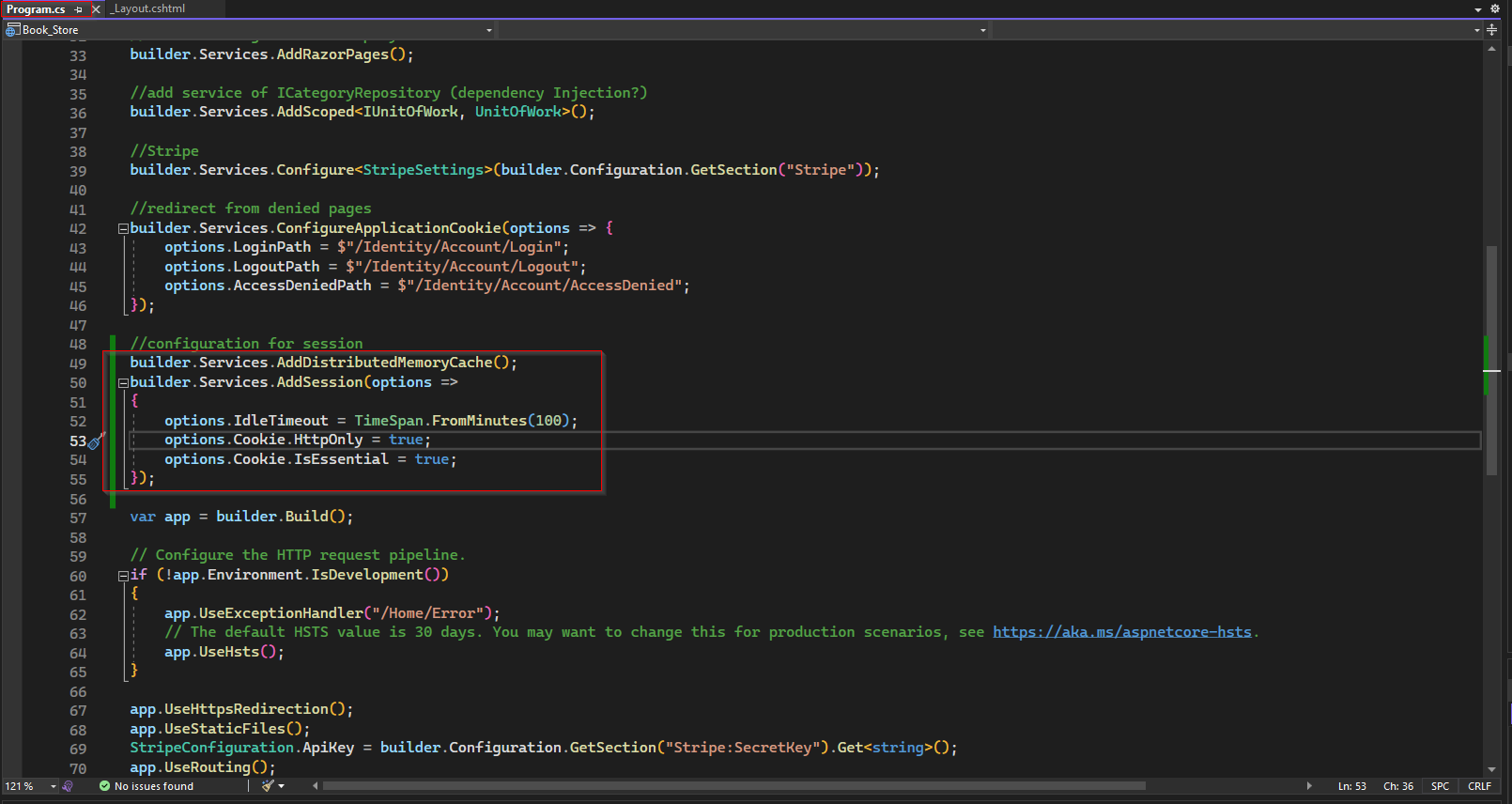


## Session in .NET Core

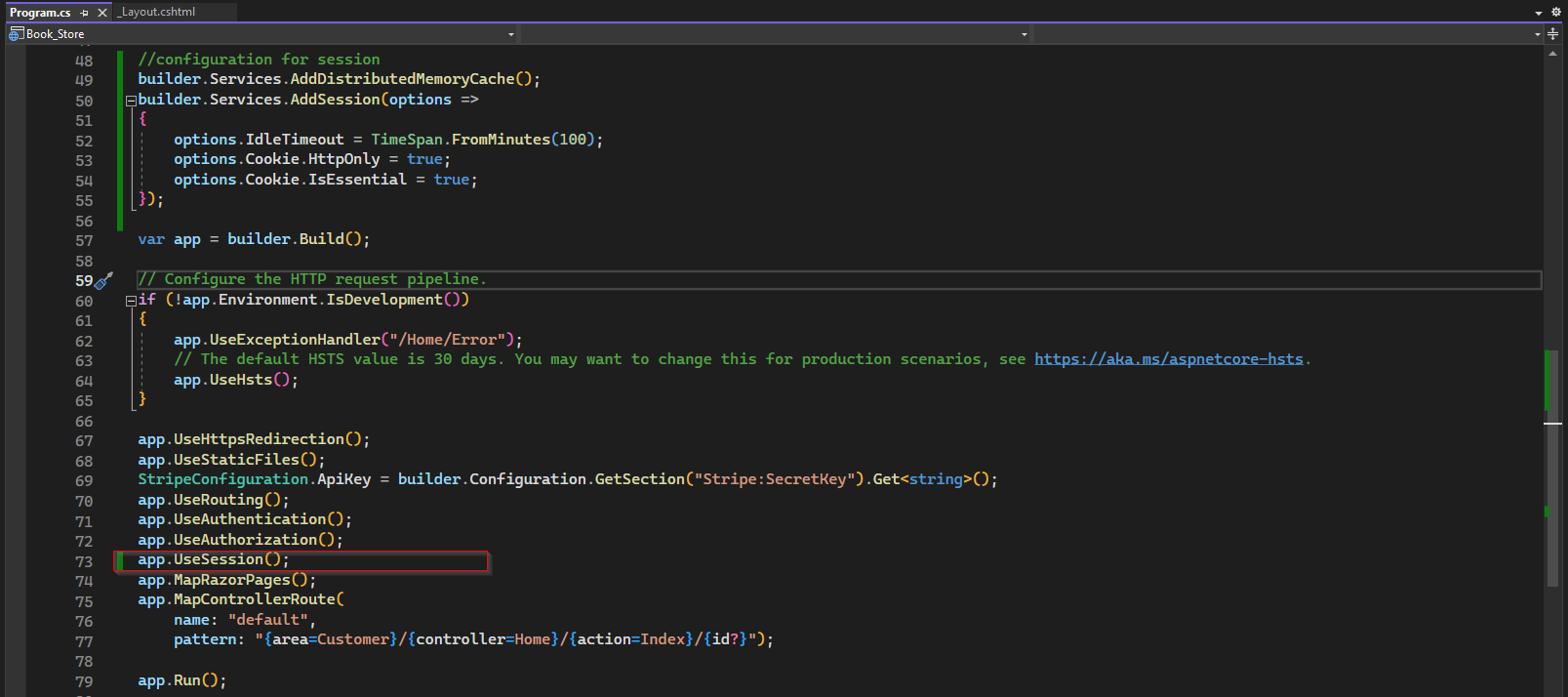
In real life e-commerce applications when a user adds or update items to the shopping cart, we can see count of distinct item along with shopping cart icon which is use as navigation button to redirect to shopping cart page. In this application, to achieve similar functionality we can use we can use session variable. But in a default .NET Core project, it is not added by default. Hence to use session in, we need to do some configuration.

### Configure Session in .NET Core

To configure session in our application we need to add following two service in program.cs file.



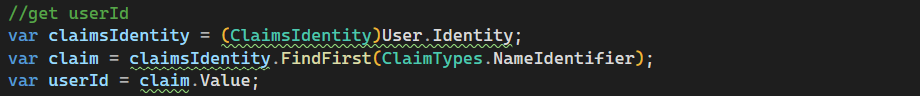
Then, after UseAuthorization , we need to add UseSession middleware as shown below.

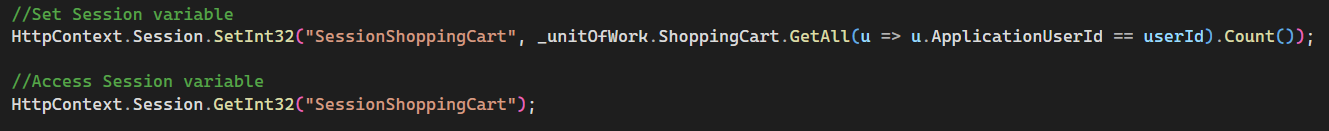


Now this application is configured to use session.

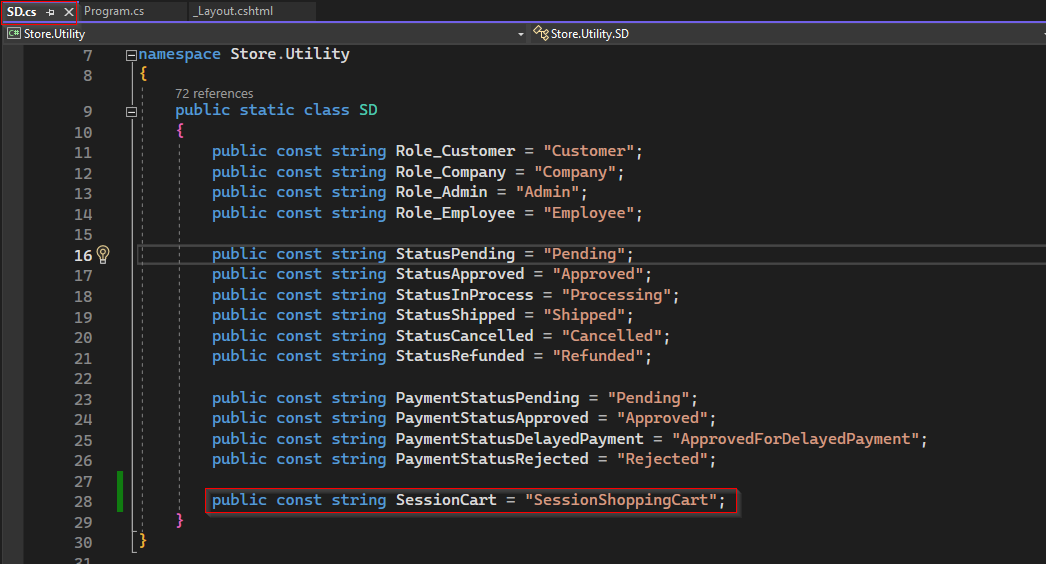
### Set and Access Session

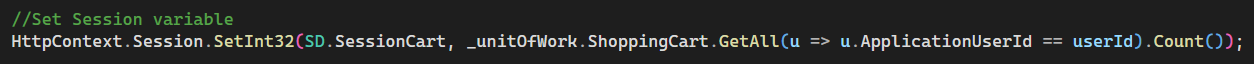
To set session variable, we use HttpContext and add a key value pair. Example session variable for number of distinct items in cart of current user.





We need to set, update, and access the session variable in several places, instead of using magic string "SessionShoppingCart", to avoid spelling mistake, we can add a constant and use it.

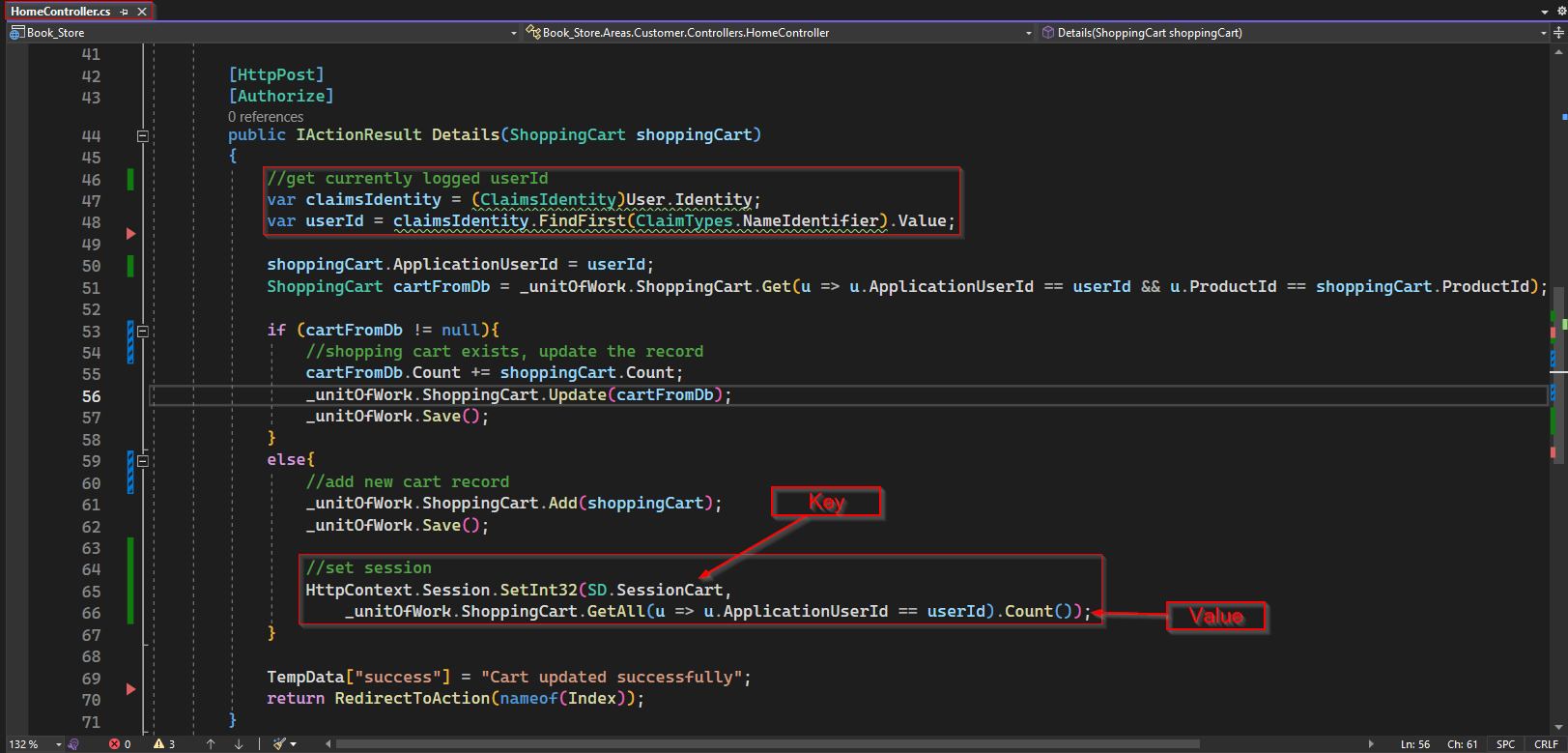




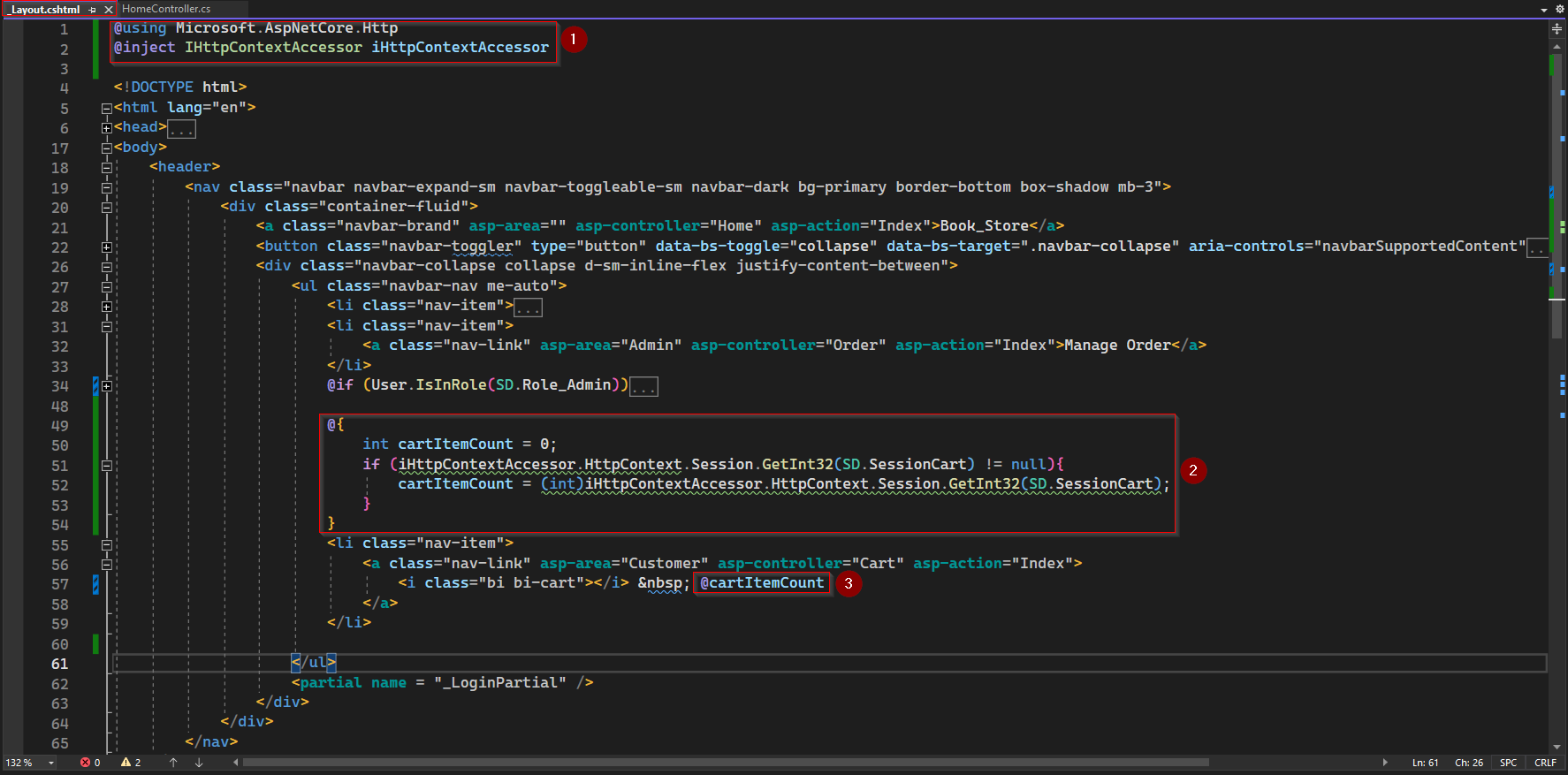


To display distinct cart items for current user. Basically, we first need to get number of distinct items for current user from database and set session variable each time when user logged in and update its value when user add item on cart or modify count or remove items from cart, and clear session variable when the user logged out.

#### Set Session while adding item on cart:

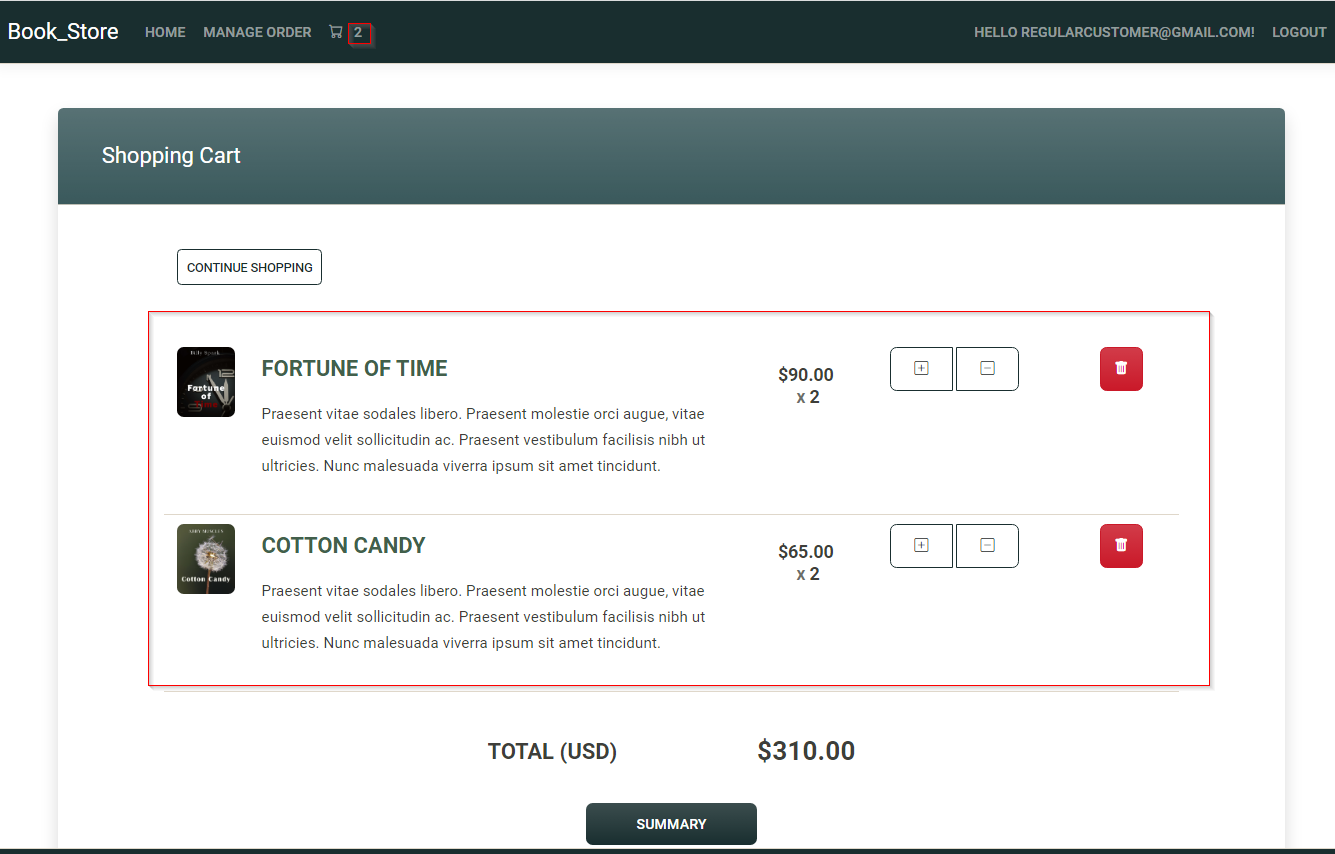


#### Access session variable on \_Layout page:

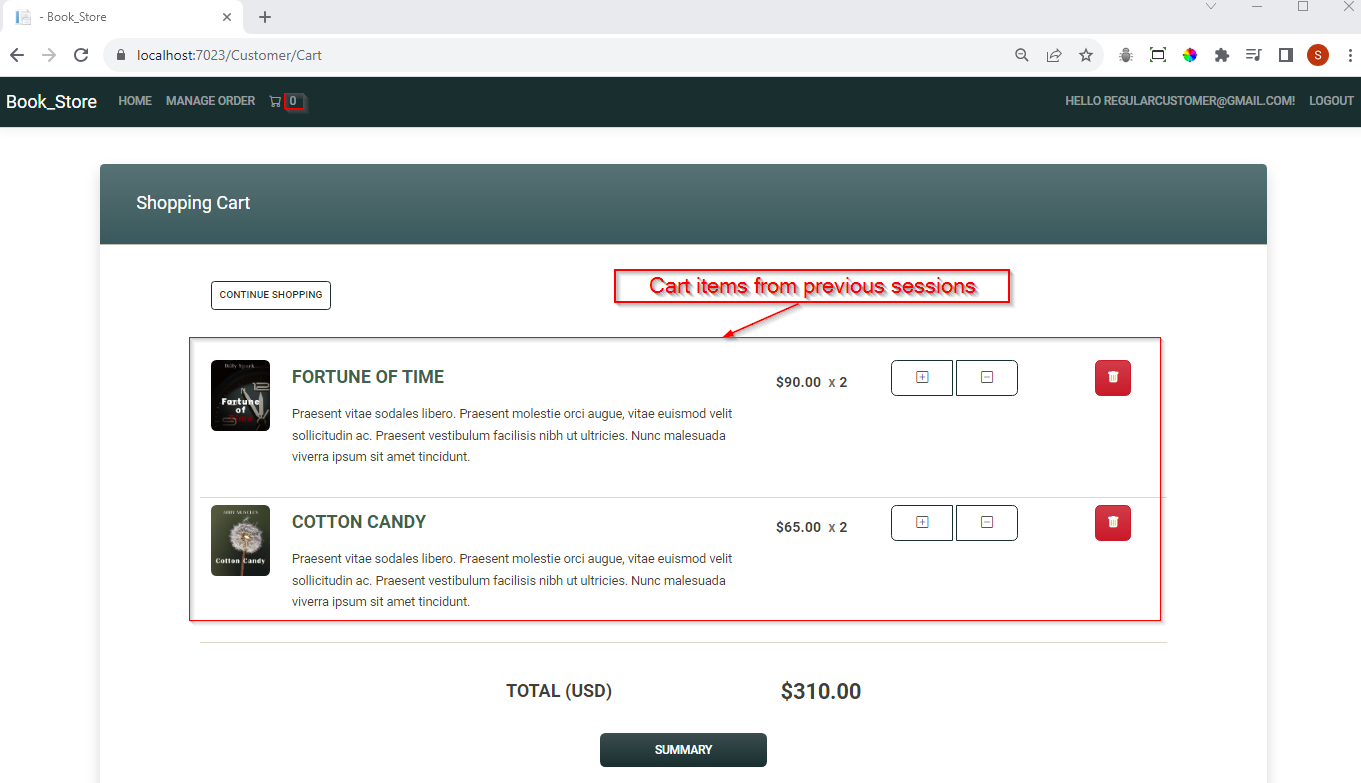


To access HttpContext in a view we need to explicitly inject that in the page. But before injecting iHttpContextAccessor, we first have to add a using statement for Microsoft.ASPNetCore.Http.

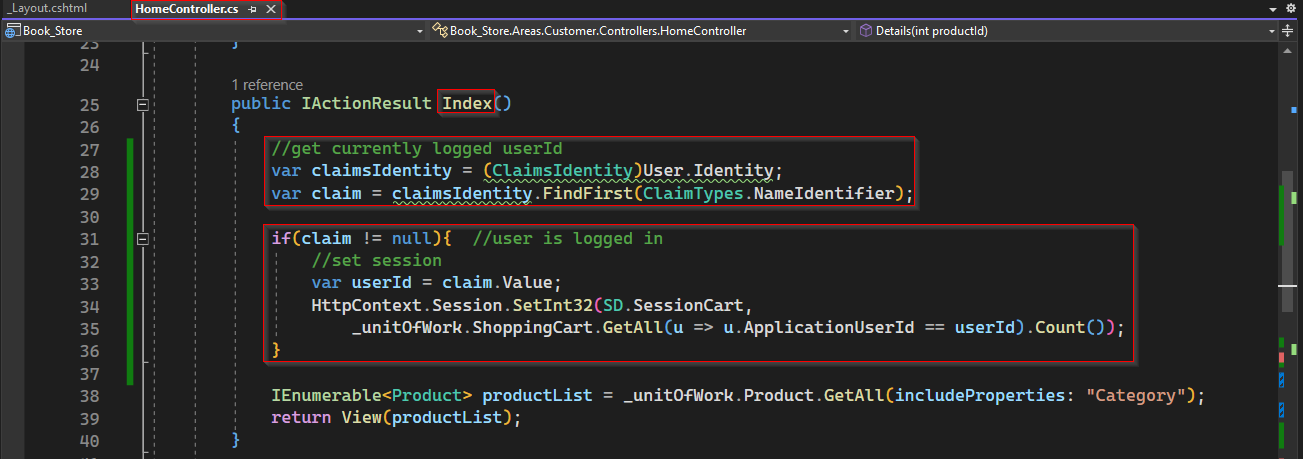
Using that context accessor now we can access the shopping cart session. In above screenshot, we have declared a variable cartItemCount of type integer and assigned default value zero to it. Then we have if statement to check is session have value for key SD.SessionCart and if yes, it will update the cartItemCount. In any case cartItemCount will be displayed along with cart icon on navigation bar.



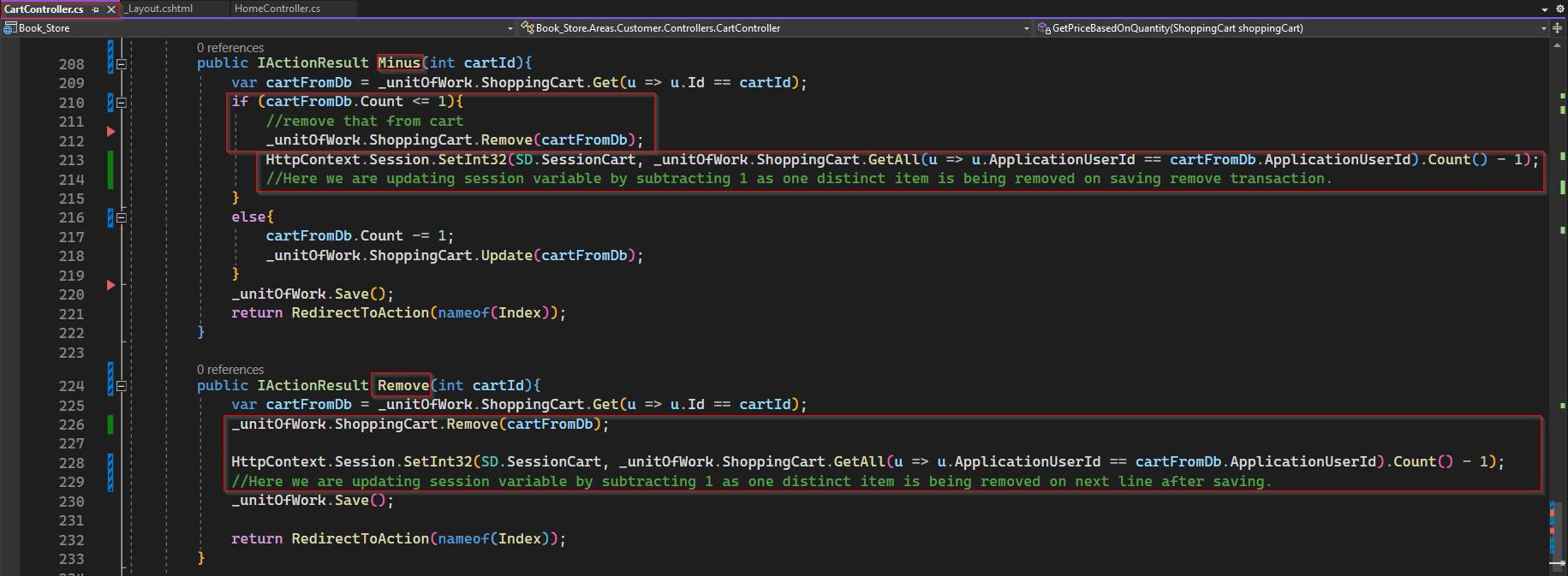
With above code application will display cart item when any user add item to cart, but it will not display item which was added on previous session or before login.



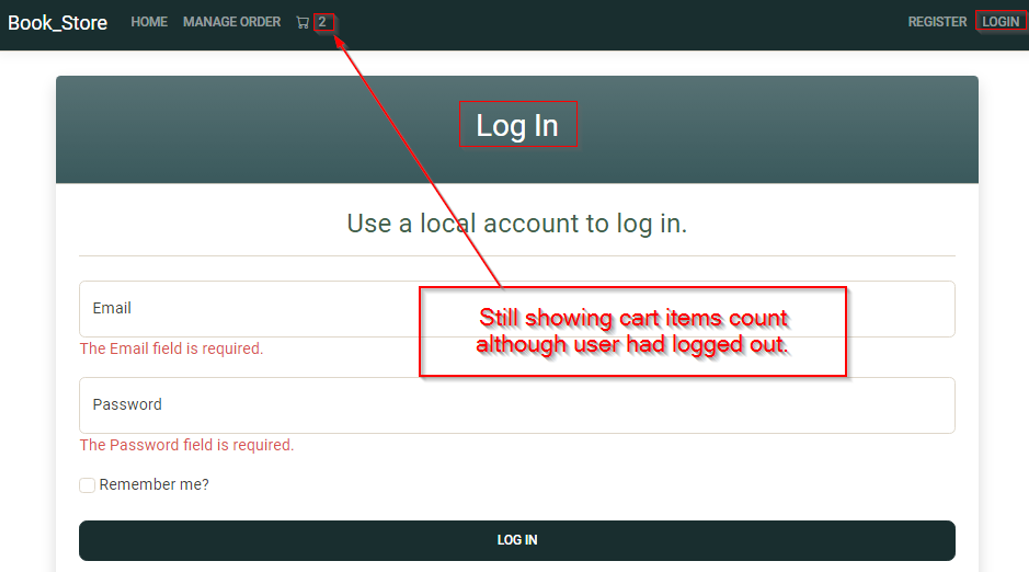
To correct above bug, we can set session while loading home page that mean on Action method Home/Index.



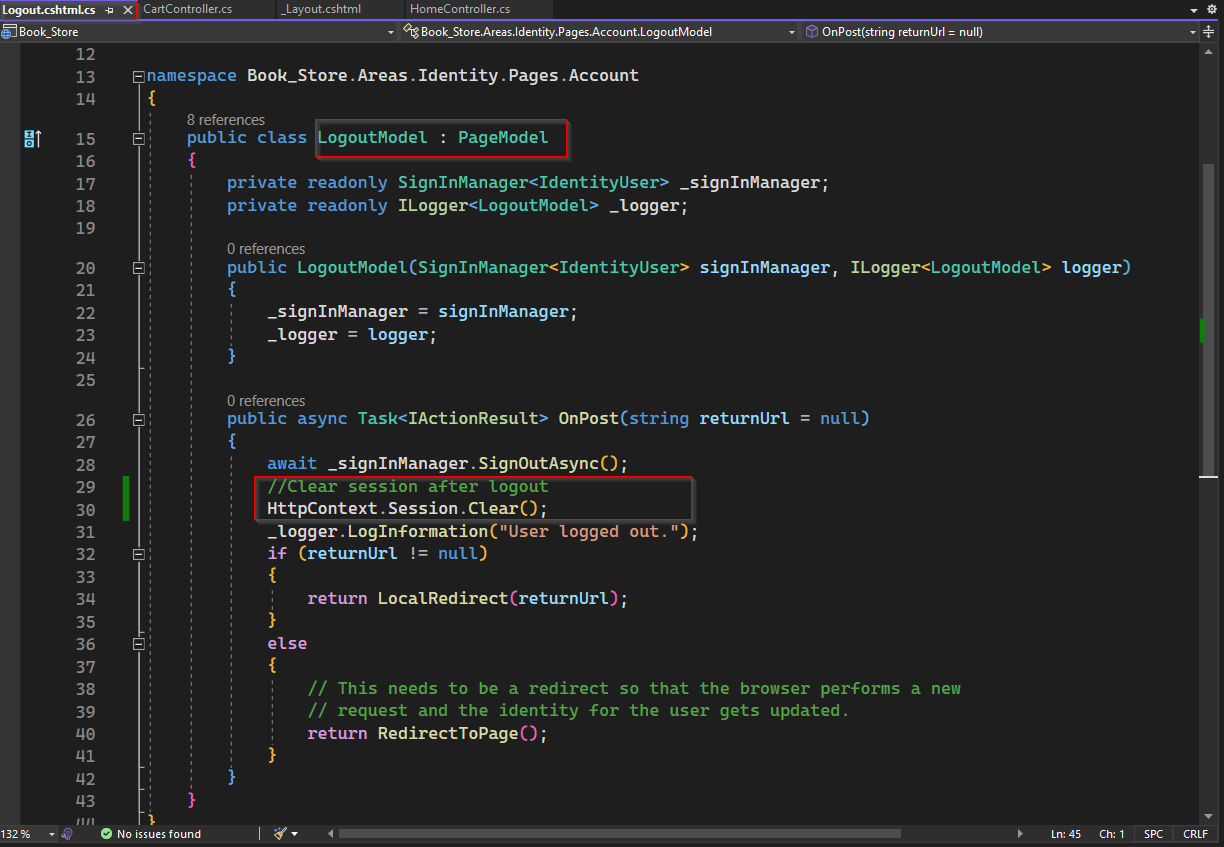
Next issue this functionality is whenever user delete items from the cart it will not update the value displayed. To solve the issue, we need to update session variable whenever any cart item is removed.



Next issue for item count is whenever an user logged out with some item on cart, the item count will not erased.



To fix the issue, whenever user is logout, we can set the session variable to zero or instead, simply clear the session, which will remove all session variable. To do so we need to modify Logout.cshtml.cs which is the page model of Razor page Identity/Pages/Account.Logout.cshtml as shown below.



Now finally, count of distinct item on navigation bar seem working as expected.

Instead of going through all these hacky ways, it would be more straight forward if whenever \_Layout.cshtml loads it would automatically check the cart items count for current user and display it. But \_Layout.cshtml is a simple view file which do not have page model or code behind where we can write backend code as in Razor page, it is not possible with it. But to achieve same functionality we can using view component.

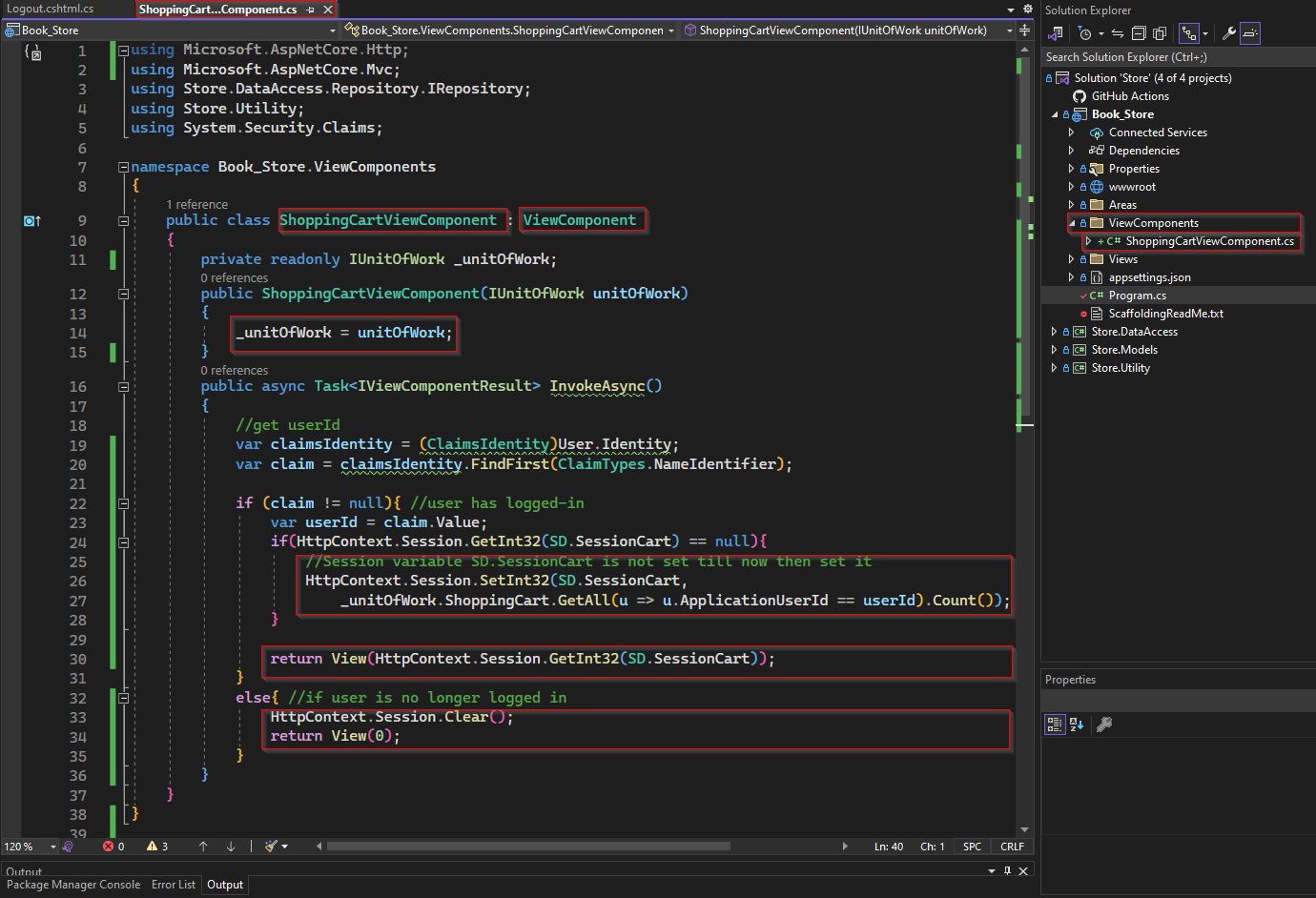
## View Component

Now, view component is like a partial view. But the difference between them is, if you are working with partial view, you do not have a page model. While with view component will have a page model or a code behind file where you can write backend code for your UI.

It is one of the advanced concepts in MVC, to use view component we have do configuration after that is done, everything will come together.

To get started with view components, inside main web project, we need to create a folder and rename it as ViewComponents. Inside that folder, we will be adding our page model of our view components, which will basically be some Class file having name followed by special keyword "ViewComponent.cs" which must inherit the base class "ViewComponent" of Microsoft.AspNetCore.Mvc.

Now, this will be the backend file for our view component, so we need to think about what we want to do here. Basically, as we did earlier, we have to go to the shopping cart database and get the shopping cart for a logged-in user as shown below.



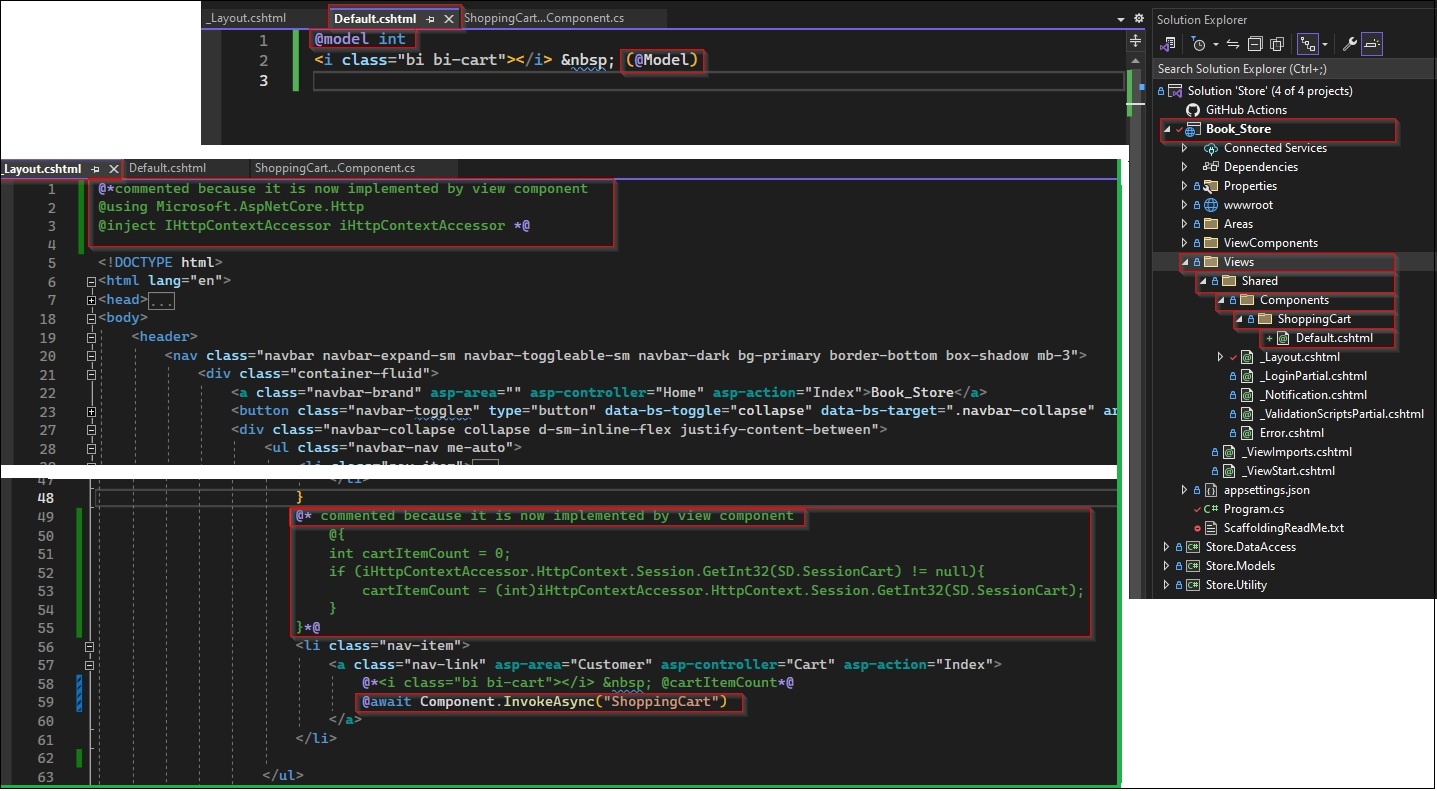
Inside above snippet of code, as we need to get data from database unitOfWork has been injected using dependency injection.

Then there is an async method handle the backend functionality of view component to be created. This method will return async Task, it can call as IActionResult, but as we know it will be returning view component, so return type has been explicitly declare as ViewComponentResult.

Now we need to create view for above page model. For that we need to create a folder inside "Views/Shared" under main web project and rename it as "Component". Now inside "Component", create a folder with same name of above page model i.e., "ShoppingCart.cshtml" and inside this folder we must create our view with name "Default.cshtml".

In Default.cshtml, we just need to add a cart icon and the populate value pass by invokeAsync method of view component.

And on \_Layout now de don’t need anymore IHttpContextAccessor and the if statement to access session variable. We just need to use view component replacing the cart icon.



**Note:**

* **Here naming file and folder is crucial and should be strictly followed.**
* **After implementing view component, if application couldn’t run successfully and some error encountered then clean solution and build it.**

Once view component is implementing as above,

* we can delete or comment all code we added to Logout.cshtml.cs, \_Layout.cshtml and action method Home/Index to display distinct cart item count on navigation bar.
* But we still need the code we added to action methods to add item to cart and update or remove items from cart.

