Build an EF and ASP.NET Core 2.1 App HOL

Welcome to the Build an Entity Framework Core and ASP.NET Core 2.1 Application in a Day Hands On Lab. This lab walks you through creating a View Component and cleans up some miscellaneous partial views.

Prior to starting this lab, you must have completed Lab 5.

All labs and files are available at https://github.com/skimedic/dotnetcore_hol.

Part 1: Create the View Component Server-Side Code

1) Create a new folder in the MVC project named ViewComponents.
2) Add a new class named Menu.cs.
3) Add the following using statements:

```
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.ViewComponents;
using SpyStore_HOL.DAL.Repos.Interfaces;
```

4) Make the class public, inherit from ViewComponent, and add the InvokeAsync method:

```
public class Menu : ViewComponent
{
  public async Task<IViewComponentResult> InvokeAsync()
  {
  }
}
```

5) Add a constructor that takes an instance of the ICategoryRepo and a private variable to hold the instance. This will be added into the class by the DI container:

```
private readonly ICategoryRepo _categoryRepo;
public Menu(ICategoryRepo categoryRepo)
{
  _categoryRepo = categoryRepo;
}
```

6) Implement the InvokeAsync method. This method uses the ICategoryRepo instance to get the list of Categories. If the call is successful, then the methods returns the MenuView partial View (to be created soon). If the call fails, the method returns a ContentViewComponentResult with an error message:

```
public async Task<IViewComponentResult> InvokeAsync()
{
  return await Task.Run<IViewComponentResult>(() =>
  {
    var cats = _categoryRepo.GetAll();
    if (cats == null)
    {
      return new ContentViewComponentResult("There was an error getting the categories");
    }
    return View("MenuView", cats);
  });
}
```

Part 2: Update the ViewImports and Create the Partial View

Step 1: Update the _ViewImports.cshtml File

1) Open the _ViewImports.cshtml file in the Views folder. This file is loaded before any Views at or below the level of this file in the directory tree. This enables a central place to include all of the usings for the Views. Confirm the using statements match the following:

```
@using SpyStore_HOL.MVC
@using SpyStore_HOL.Models.Entities
@using SpyStore_HOL.Models.ViewModels
@using SpyStore_HOL.Models.ViewModels.Base
@using System.Collections.Generic
@using Microsoft.AspNetCore.Mvc.Rendering
@using SpyStore_HOL.MVC.ViewModels
```

2) In order to use the ViewComponent as a Tag Helper, the assembly must be registered in the _ViewImports.cshtml file. Add the following to the end of the file:

```
@addTagHelper *, SpyStore_HOL.MVC
```

Step 1: Create the MenuView partial view

1) Add a new folder named Components under the Views\Shared folder.
2) Add a new folder named Menu under the Components folder.
3) Add a new partial view named MenuView.cshtml in the new folder.
4) Update the code to match the following:

```
@model IEnumerable<Category>
<li><a asp-controller="Products" asp-action="Featured">Featured</a></li>
@foreach (var item in Model)
{
  <li><a asp-controller="Products" asp-action="ProductList" asp-route-
id="@item.Id">@item.CategoryName</a></li>
}
```

Part 3: Update the Main Layout

1) Open the _Layout.cshtml file in Views\Shared folder and replace the menu items with the following:

```
<vc:menu></vc:menu>
```

2) Add the call to render the LoginView partial view to the menu div, as follows:

```
<div class="navbar-collapse collapse">
  <ul class="nav navbar-nav">
    <vc:menu></vc:menu>
  </ul>
  @Html.Partial("LoginView")
</div>
```

Summary

The lab created the View Component and its view as well as added the LoginPartial view.

Next steps

In the next part of this tutorial series, you will create a custom tag helper