Tables/Searching Lab

Lab 1: Product and Category Pages

Right mouse-click on the **\Views** folder and create a new folder named **CategoryMaintenance**

Right mouse-click on the CategoryMaintenance folder and create a new view named **CategoryMaintenance.cshtml**.

Open the new view and make it look like the following:

```
@{
  ViewData["Title"] = "Category Maintenance";
}

<h2>Category Maintenance</h2>
```

Right mouse-click on the **\Views** folder and create a new folder named **ProductMaintenance**

Right mouse-click on the **ProductMaintenance** folder and create a new view named **ProductMaintenance.cshtml**.

Open the new view and make it look like the following:

```
@{
  ViewData["Title"] = "Product Maintenance";
}

<h2>Product Maintenance</h2>
```

Right mouse-click on the **\Controllers** folder and create an empty controller named **CategoryMaintenanceController.cs**

Make the code look like the following:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Logging;

namespace AdvWorks.Controllers
{
  public class CategoryMaintenanceController : Controller
  {
    public
CategoryMaintenanceController(ILogger<CategoryMaintenanceController>
logger)
    {
      _logger = logger;
    }

    private readonly ILogger<CategoryMaintenanceController> _logger;

    public IActionResult CategoryMaintenance()
    {
      return View();
    }
  }
}
```

Right mouse-click on the **\Controllers** folder and create an empty controller named **ProductMaintenanceController.cs**

Make the code look like the following:

Copyright © 2022 by Paul D. Sheriff
All rights reserved worldwide. Reproduction is strictly prohibited.

```
using AdvWorks.Common;
using AdvWorks.EntityLayer;
using AdvWorks.ViewModelLayer;
using Microsoft.AspNetCore.Mvc;

namespace AdvWorks.Controllers
{
  public class ProductMaintenanceController : Controller
  {
    public
ProductMaintenanceController(ILogger<ProductMaintenanceController>
logger,
      IRepository<Product, ProductSearch> repo)
    {
      _logger = logger;
      _repo = repo;
    }

    private readonly ILogger<ProductMaintenanceController> _logger;
    private readonly IRepository<Product, ProductSearch> _repo;

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

Open the **\Views\Home\Index.cshtml** file and make it look like the following:

```
@{
  ViewData["Title"] = "Home Page";
}

<div class="text-center">
  <h1 class="display-4">Welcome to Adventure Works</h1>
  <p>This site helps you work with the data in the AdventureWorksLT
database.</p>
</div>

<div class="row justify-content-center">
  <div class="list-group col-6">
    <a asp-action="ProductMaintenance"
       asp-controller="ProductMaintenance" class="list-group-item">
      Product Maintenance
    </a>
    <a asp-action="CategoryMaintenance"
       asp-controller="CategoryMaintenance" class="list-group-item">
      Category Maintenance
    </a>
  </div>
</div>
```


Copyright © 2022 by Paul D. Sheriff
All rights reserved. Reproduction is strictly prohibited.

Try it Out

Run the application and make sure you can get to both these new pages

Lab 2: Display Product Table

Open the **ProductMaintenanceController.cs** file

Modify the **ProductMaintenance**() method to look like the following:

```
[HttpGet]
public IActionResult ProductMaintenance()
{
  // Create view model passing in repository
  ProductViewModel vm = new(_repo);

  // Call method to load products
  vm.Search();

  return View(vm);
}
```

Create a new partial view in the **\Views\ProductMaintenance** folder named **_List.cshtml**

Add the following code to this file

Copyright © 2022 by Paul D. Sheriff
All rights reserved worldwide. Reproduction is strictly prohibited.

```
@using AdvWorks.EntityLayer
@model AdvWorks.ViewModelLayer.ProductViewModel

<table class="table table-bordered table-hover table-striped">
  <thead>
    <tr>
      <th>Product Name</th>
      <th>Product Number</th>
      <th class="text-right">Cost</th>
      <th class="text-right">Price</th>
    </tr>
  </thead>
  <tbody>
    @foreach (Product item in Model.Products) {
      <tr>
        <td>@Html.DisplayFor(m => item.Name)</td>
        <td>@Html.DisplayFor(m => item.ProductNumber)</td>
        <td class="text-right">@Html.DisplayFor(m =>
item.StandardCost)</td>
        <td class="text-right">@Html.DisplayFor(m =>
item.ListPrice)</td>
      </tr>
    }
  </tbody>
</table>
```

Open the **ProductMaintenance.cshtml** file and make it look like the following:

```
@{
  ViewData["Title"] = "Product Maintenance";
}

<h2>Product Maintenance</h2>

<partial name="_List" />
```

Try it Out

Run the application and view the product table you just created

Lab 3: Go to Detail

Right mouse-click on the **\Views\ProductMaintenance** folder and create a new View named **_Detail.cshtml**.

Copyright © 2022 by Paul D. Sheriff
All rights reserved. Reproduction is strictly prohibited.

```
@model AdvWorks.ViewModelLayer.ProductViewModel

<form>
  <input type="hidden" asp-for="SelectedProduct.ProductID" />
  <input type="hidden" asp-for="IsAdding" />

  <div class="form-group">
    <label asp-for="SelectedProduct.Name"></label>
    <input class="form-control" asp-for="SelectedProduct.Name" />
  </div>
  <div class="form-group">
    <label asp-for="SelectedProduct.ProductNumber"></label>
    <input class="form-control" asp-
for="SelectedProduct.ProductNumber" />
  </div>
  <div class="form-group">
    <label asp-for="SelectedProduct.Color"></label>
    <input class="form-control" asp-for="SelectedProduct.Color" />
  </div>
  <div class="form-group">
    <label asp-for="SelectedProduct.StandardCost"></label>
    <input class="form-control" asp-
for="SelectedProduct.StandardCost" />
  </div>
  <div class="form-group">
    <label asp-for="SelectedProduct.ListPrice"></label>
    <input class="form-control" asp-for="SelectedProduct.ListPrice"
/>
  </div>
</form>
```

Open the **ProductMaintenance.cshtml** file and make it look like the following:

```
@{
  ViewData["Title"] = "Product Maintenance";
}

<h2>Product Maintenance</h2>

@if (Model.IsDetailVisible) {
  <partial name="_Detail" />
}
else {
  <partial name="_List" />
}
```

Add the Edit Button

Open the **_List.cshtml** file

Add a new <th> in the <thead> as the first <th> in the <tr>

Copyright © 2022 by Paul D. Sheriff
All rights reserved worldwide. Reproduction is strictly prohibited.

```
<th>Actions</th>
```

Add a new <td> in the <tbody> as the first <td> in the <tr>

```
<td>
  <a asp-controller="ProductMaintenance"
     asp-action="ProductEdit"
     asp-route-id="@item.ProductID"
     class="btn btn-primary">
    Edit
  </a>
</td>
```

Open the **ProductMaintenanceController.cs** file and add a new method

```
[HttpGet]
public IActionResult ProductEdit(int id)
{
// Create view model passing in repository
  ProductViewModel vm = new(_repo);

  // Call method to load a product
  vm.Get(id);

  return View("ProductMaintenance", vm);
}
```

Try it Out

Run the application and view the product table.

Click on any Edit button and view the detail page for that product.

Lab 4: Add Search Form

Right mouse-click on the **\Views\ProductMaintenance** folder and create a new View named **_Search.cshtml**.

Copyright © 2022 by Paul D. Sheriff
All rights reserved. Reproduction is strictly prohibited.

```
@model AdvWorks.ViewModelLayer.ProductViewModel

<form method="get" asp-action="ProductSearch">
  <div class="card">
    <div class="card-header bg-primary text-light">
      <h5 class="card-title">Search for Products</h5>
    </div>
    <div class="card-body">
      <div class="form-group">
        <label asp-for="SearchEntity.Name"></label>
        <input asp-for="SearchEntity.Name" class="form-control" />
      </div>
      <div class="form-group">
        <label asp-for="SearchEntity.ListPrice"></label>
        <input type="number" asp-for="SearchEntity.ListPrice"
class="form-control" />
      </div>
    </div>
    <div class="card-footer bg-primary text-light">
      <button class="btn btn-success">Search</button>
    </div>
  </div>
</form>
```

Open the **ProductMaintenance.cshtml** file and add the <partial> tag shown in bold

```
@model AdvWorks.ViewModelLayer.ProductViewModel

@{
  ViewData["Title"] = "Product Maintenance";
}

<h2>Product Maintenance</h2>

@if (Model.IsDetailVisible) {
  <partial name="_Detail" />
}
else {
  <partial name="_Search" />
  <partial name="_List" />
}
```

Try it Out

Run the application and display the **search** and the table

NOTE: The Search button does not yet work

Copyright © 2022 by Paul D. Sheriff
All rights reserved worldwide. Reproduction is strictly prohibited.

Lab 5: Enable Searching

In the AdvWorks.DataLayer project open the **\RepositoryClasses\ProductRespository.cs** file and modify the AddWhereClause() method.

```
public IQueryable<Product> AddWhereClause(IQueryable<Product> query,
ProductSearch search) {
  // Perform Searching
  if (!string.IsNullOrEmpty(search.Name)) {
    query = query.Where(row => row.Name.StartsWith(search.Name));
  }
  if (search.ListPrice.HasValue) {
    query = query.Where(row => row.ListPrice >= search.ListPrice);
  }

  return query;
}
```

Open the **ProductMaintenanceController.cs** file and add a new GET method

```
[HttpGet]
public IActionResult ProductSearch(ProductViewModel vm)
{
  vm.Repository = _repo;

  // Call method to search for products
  vm.Search();

  return View("ProductMaintenance", vm);
}
```

Try it Out

Run the application and put the letter "C" into the Product Name search box

Click the Search button and you should see the records filtered.

Lab 6: Add a Reset Button

Open the **_Search.cshtml** file and immediately after the Search button, add a hyperlink to call the ProductMaintenance() method

Copyright © 2022 by Paul D. Sheriff
All rights reserved. Reproduction is strictly prohibited.

```
<a class="btn btn-primary" asp-action="ProductMaintenance">
  Reset
</a>
```

Try it Out

Run the application and put the letter "C" into the Product Name search box

Click the Search button and you should see the records filtered.

Click the **Reset** button.

Copyright © 2022 by Paul D. Sheriff
All rights reserved worldwide. Reproduction is strictly prohibited.