.NET 10 App Dev Hands-On Lab

Razor Pages Lab 8 –Areas

This walks you through creating an area for Make maintenance. You must have completed Razor Page Lab 7 before starting this lab.

Note: Adjust any directory separators to your OS (e.g. \ for Windows, / for Mac/Linux)

Part 1: Create the Area, Templates, and ViewStart/ViewImports

- Create a new folder named `Areas` in the `AutoLot.Web` project. In that folder, add a new folder named Admin, then a folder named Pages under Admin, and under Admin, a new folder named `Makes`. Under the `Makes` directory, add a `DisplayTemplates` directory and an `EditorTemplates` directory.
  NOTE: The MVC Area scaffolding doesn't correctly set up Areas for Razor Pages, so you must do these steps by hand.
- Create a new view named `Make.cshtml` in the `DisplayTemplates` directory. Update the markup to the following:

```
@model Make
<hr/>
<dl class="row">
  <dt class="col-sm-2">
    @Html.DisplayNameFor(model => model.Name)
  </dt>
  <dd class="col-sm-10">
    @Html.DisplayFor(model => model.Name)
  </dd>
</dl>
```

- Create a new view named `Make.cshtml` in the `EditorTemplates` directory. Update the markup to the following:

```
@model Make
<div>
  <label asp-for="Name" class="col-form-label"></label>
  <input asp-for="Name" />
  <span asp-validation-for="Name" class="text-danger"></span>
</div>
```

- Either copy the `AutoLot.Web\Pages\_ViewImports.cshtml` and `AutoLot.Web\Pages\_ViewStart.cshtml` files into the `Areas\Admin\Pages` folder or move them to the project root.

Part 2: The Makes Pages

Step 1: Create the Index Page

- Add a new Razor Page named `Index.cshtml` to the `Areas\Admin\Pages\Makes` folder and update the code behind to the following:

```
namespace AutoLot.Web.Areas.Admin.Pages.Makes;
public class IndexModel(IAppLogging appLogging, IMakeRepo repo) : PageModel
{
  [ViewData]
  public string Title => "Makes";
  public IEnumerable<Make> MakeRecords { get; set; }
  public void OnGet() => MakeRecords = repo.GetAllIgnoreQueryFilters();
}
```

- Update the markup to the following:

```
@page
@model AutoLot.Web.Areas.Admin.Pages.Makes.IndexModel
<h1>Vehicle Makes</h1>
<p><item-create></item-create></p>
<table class="table">
  <thead>
    <tr>
      <th>@Html.DisplayNameFor(model => ((List<Make>)model.MakeRecords)[0].Name)</th>
      <th></th>
    </tr>
  </thead>
  <tbody>
  @foreach (var item in Model.MakeRecords) {
    <tr>
      <td>@Html.DisplayFor(modelItem => item.Name)</td>
      <td>
        <item-edit item-id="@item.Id"></item-edit> |
        <item-details item-id="@item.Id"></item-details> |
        <item-delete item-id="@item.Id"></item-delete>
      </td>
    </tr>
  }
  </tbody>
</table>
```

Step 2: Create the Details Page

- Add a new Razor Page named `Details.cshtml` to the `Areas\Admin\Pages\Makes` folder and update the code behind to the following:

```
namespace AutoLot.Web.Areas.Admin.Pages.Makes;
public class DetailsModel(IAppLogging appLogging, IMakeRepo makeRepo)
  : BasePageModel<Make>(appLogging, makeRepo, "Details")
{
  public void OnGet(int? id) => GetOneEntity(id);
}
```

- Update the markup to the following:

```
@page "{id?}"
@model AutoLot.Web.Areas.Admin.Pages.Makes.DetailsModel
<h1>Details for @Model.Entity.Name</h1>
@if (!string.IsNullOrEmpty(Model.Error))
{
  <div class="alert alert-danger" role="alert">@Model.Error</div>
}
else
{
  @Html.DisplayFor(m => m.Entity)
  <div>
    <item-edit item-id="@Model.Entity.Id"></item-edit> |
    <item-delete item-id="@Model.Entity.Id"></item-delete> |
    <item-list></item-list>
  </div>
}
```


Step 3: Create the Create Page

- Add a new Razor Page named `Create.cshtml` to the `Areas\Admin\Pages\Makes` folder and update the code behind to the following:

```
namespace AutoLot.Web.Areas.Admin.Pages.Makes;
public class CreateModel(IAppLogging appLogging, IMakeRepo makeRepo)
  : BasePageModel<Make>(appLogging, makeRepo, "Create")
{
  public void OnGet() => Entity = new Make();
  public IActionResult OnPost() => SaveOne(BaseRepoInstance.Add);
}
```

- Update the markup to the following:

```
@page
@model AutoLot.Web.Areas.Admin.Pages.Makes.CreateModel
<h1>Create a New Make</h1>
<hr/>
@if (!string.IsNullOrEmpty(Model.Error))
{
  <div class="alert alert-danger" role="alert">@Model.Error</div>
}
else
{
  <form asp-page="Create">
    <div class="row">
      <div class="col-md-4">
        <div asp-validation-summary="ModelOnly" class="text-danger"></div>
        @Html.EditorFor(x => x.Entity, new { LookupValues = Model.LookupValues })
      </div>
    </div>
    <div class="d-flex flex-row mt-3">
       <button type="submit" class="btn btn-success">Create
       <i class="fa-solid fa-plus"></i></button>  |  
       <item-list></item-list>
    </div>
  </form>
  @section Scripts {
    <partial name="_ValidationScriptsPartial"/>
  }
}
```

Step 4: Create the Edit Page

- Add a new Razor Page named `Edit.cshtml` to the `Areas\Admin\Pages\Makes` folder and update the code behind to the following:

```
namespace AutoLot.Web.Areas.Admin.Pages.Makes;
public class EditModel(IAppLogging appLogging, IMakeRepo repo)
  : BasePageModel<Make>(appLogging, repo, "Edit")
{
  public void OnGet(int? id) => GetOneEntity(id);
  public IActionResult OnPost() => SaveOne(BaseRepoInstance.Update);
}
```

- Update the markup to the following:

```
@page "{id?}"
@model AutoLot.Web.Areas.Admin.Pages.Makes.EditModel
<h1>Edit @Model.Entity.Name</h1>
<hr/>
@if (!string.IsNullOrEmpty(Model.Error))
{
  <div class="alert alert-danger" role="alert">@Model.Error</div>
}
else
{
  <form asp-page="Edit" asp-route-id="@Model.Entity.Id">
    <div class="row">
      <div class="col-md-4">
        <div asp-validation-summary="ModelOnly"></div>
        @Html.EditorFor(x => x.Entity)
        <input type="hidden" asp-for="Entity.Id"/>
        <input type="hidden" asp-for="Entity.TimeStamp"/>
      </div>
    </div>
    <div class="d-flex flex-row mt-3">
      <button type="submit" class="btn btn-primary">Save
        <i class="fa-solid fa-save"></i></button>  |  
      <item-list></item-list>
    </div>
  </form>
}
@section Scripts {
  @{ await Html.RenderPartialAsync("_ValidationScriptsPartial"); }
}
```

Step 5: Create the Delete Page

- Add a new Razor Page named `Delete.cshtml` to the `Areas\Admin\Pages\Makes` folder and update the code behind to the following:

```
namespace AutoLot.Web.Areas.Admin.Pages.Makes;
public class DeleteModel(IAppLogging appLogging, IMakeRepo repo)
  : BasePageModel<Make>(appLogging, repo, "Delete")
{
  public void OnGet(int? id) => GetOneEntity(id);
  public IActionResult OnPost(int id) => DeleteOne(id);
}
```

- Update the markup to the following:

```
@page "{id?}"
@model AutoLot.Web.Areas.Admin.Pages.Makes.DeleteModel
<h1>Delete @Model.Entity.Name</h1>
@if (!string.IsNullOrEmpty(Model.Error))
{
  <div class="alert alert-danger" role="alert">@Model.Error</div>
}
else
{
  <h3>Are you sure you want to delete this car?</h3>
  <div>
    @Html.DisplayFor(c=>c.Entity)
    <form asp-page="Delete" asp-route-id="@Model.Entity.Id">
      <input type="hidden" asp-for="Entity.Id"/>
      <input type="hidden" asp-for="Entity.TimeStamp"/>
      <button type="submit" class="btn btn-danger">Delete
         <i class="fa-solid fa-trash"></i></button>  |  
      <item-list></item-list>
    </form>
  </div>
}
```

Part 3: Update the Menu

- Add the following to the `_Menu.cshtml` partial:

```
<li class="nav-item">
  <a class="nav-link text-dark" asp-area="Admin" asp-page="/Makes/Index" title="Makes Admin">
    Makes Admin <i class="fa-solid fa-cog"></i>
  </a>
</li>
```

Summary

This lab created the area for Make maintenance and completed the `AutoLot.Web` application.

Next Steps

The next labs will add the data services to abstract away the calls to the repositories.