# Build an EF and ASP.NET Core 3.0 App HOL

## Lab 11

This lab walks you through creating a new Area in the AutoLot.Web application. Prior to starting this lab, you must have completed Lab 10.

# Part 1: Create the Folder Structure

1) Create a new folder in the AutoLot.Web project named `Areas` and, in this folder, create a new folder named Admin. In the Admin folder create a new folder named Controllers and another folder named Views.
   **NOTE:** Visual Studio can scaffold this for you, but that's about all it does.

# Part 2: Move the Shared views

1) Move the partial views named _ViewImports.cshtml and _ViewStart.cshtml from the main Views to the project level. This will apply them to the area views as well.

# Part 3: Add the MakesController

1) Add a new file named MakesController to the Areas\Admin\Controllers folder. Add the following namespaces to the top of the file:

```
using Microsoft.AspNetCore.Mvc;
using AutoLot.Models.Entities;
using AutoLot.Dal.Repos.Interfaces;
```

2) Make the class public and inherit from Controller. Add the Area attribute and the Route attribute to the top of the class:

```
[Area("Admin")]
[Route("Admin/[controller]/[action]")]
public class MakesController : Controller
{
}
```

All files copyright Phil Japikse (http://www.skimedic.com/blog)

3) Add the following code into the class:

```
private readonly IMakeRepo _repo;

public MakesController(IMakeRepo repo)
{
    _repo = repo;
}

// GET: Admin/Makes
[Route("/Admin")]
[Route("/Admin/[controller]")]
[Route("/Admin/[controller]/[action]")]
public IActionResult Index()
{
  return View(_repo.GetAll());
}

// GET: Admin/Makes/Create
public IActionResult Create()
{
  return View();
}

// POST: Admin/Makes/Create
// To protect from overposting attacks, enable the specific properties you want to bind to, for
// more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Create([Bind("Name,Id,TimeStamp")] Make make)
{
  if (!ModelState.IsValid) return View(make);
  _repo.Add(make);
  return RedirectToAction(nameof(Index));
}

// GET: Admin/Makes/Details/5
public IActionResult Details(int? id)
{
  if (id == null) return BadRequest();
  var make = _repo.Find(id.Value);
  if (make == null) return NotFound();
  return View(make);
}

// GET: Admin/Makes/Edit/5
public IActionResult Edit(int? id)
{
  if (id == null) return BadRequest();
  var make = _repo.Find(id.Value);
  if (make == null) return NotFound();
  return View(make);
}
```

```
// POST: Admin/Makes/Edit/5
// To protect from overposting attacks, enable the specific properties you want to bind to, for
// more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Edit(int id, [Bind("Name,Id,TimeStamp")] Make make)
{
  if (id != make.Id) return NotFound();
  _repo.Update(make);
  return RedirectToAction(nameof(Index));
}
// GET: Admin/Makes/Delete/5
public IActionResult Delete(int? id)
{
  if (id == null) return BadRequest();
  var make = _repo.Find(id.Value);
  if (make == null) return NotFound();
  return View(make);
}

// POST: Admin/Makes/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public IActionResult Delete(int id, Make make)
{
  if (id == null) return BadRequest();
    _repo.Delete(make);
    return RedirectToAction(nameof(Index));
}
```

# Part 4: Add the Views

## Step 1: Add the Index view

1) Add a new folder named Makes in the Areas\Admin\Views folder. Add a new view named `Index.cshtml` into the new folder.
2) Update the code to match the following:

```
@model IEnumerable<AutoLot.Models.Entities.Make>
@{
    ViewData["Title"] = "Index";
}

<h1>Index</h1>
<p><a asp-action="Create">Create New</a></p>
<table class="table">
  <thead>
  <tr>
    <th>@Html.DisplayNameFor(model => model.Name) </th>
    <th></th>
  </tr>
  </thead>
  <tbody>
  @foreach (var item in Model)
  {
    <tr>
      <td>@Html.DisplayFor(modelItem => item.Name) </td>
      <td>
        <a asp-action="Edit" asp-route-id="@item.Id">Edit</a> |
        <a asp-action="Details" asp-route-id="@item.Id">Details</a> |
        <a asp-action="Delete" asp-route-id="@item.Id">Delete</a>
      </td>
    </tr>
  }
  </tbody>
</table>
```

## Step 2: Add the Details view

1) Add a new view named `Details.cshtml` into the new folder.
2) Update the code to match the following:

```
@model AutoLot.Models.Entities.Make
@{ ViewData["Title"] = "Details"; }
<h1>Details</h1>
<div>
  <h4>Make</h4>
  <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>
</div>
<div>
  <a asp-action="Edit" asp-route-id="@Model.Id">Edit</a> |
  <a asp-action="Index">Back to List</a>
</div>
```

## Step 3: Add the Create view

1) Add a new view named `Create.cshtml` into the new folder.
2) Update the code to match the following:

```
@model AutoLot.Models.Entities.Make
@{ ViewData["Title"] = "Create"; }
<h1>Create</h1>
<h4>Make</h4>
<hr/>
<div class="row">
  <div class="col-md-4">
    <form asp-action="Create">
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>
      <div class="form-group">
        <label asp-for="Name" class="form-control"></label>
        <input asp-for="Name" class="form-control"/>
        <span asp-validation-for="Name" class="text-danger"></span>
      </div>
      <div class="form-group">
        <input type="submit" value="Create" class="btn btn-primary"/>
      </div>
    </form>
  </div>
</div>

<div><a asp-action="Index">Back to List</a></div>

@section Scripts {
  @{ await Html.RenderPartialAsync("_ValidationScriptsPartial"); }
}
```

## Step 4: Add the Edit view

1) Add a new view named `Edit.cshtml` into the new folder.
2) Update the code to match the following:

```
@model AutoLot.Models.Entities.Make

@{ ViewData["Title"] = "Edit"; }
<h1>Edit</h1>
<h4>Make</h4>
<hr/>
<div class="row">
  <div class="col-md-4">
    <form asp-action="Edit">
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>
      <div class="form-group">
        <label asp-for="Name" class="control-label"></label>
        <input asp-for="Name" class="form-control"/>
        <span asp-validation-for="Name" class="text-danger"></span>
      </div>
      <input type="hidden" asp-for="Id"/>
      <div class="form-group">
        <input type="submit" value="Save" class="btn btn-primary"/>
      </div>
    </form>
  </div>
</div>

<div>
    <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
    @{ await Html.RenderPartialAsync("_ValidationScriptsPartial"); }
}
```

### Step 5: Add the Delete view

1) Add a new view named `Delete.cshtml` into the new folder.
2) Update the code to match the following:

```
@model AutoLot.Models.Entities.Make
@{ ViewData["Title"] = "Delete"; }
<h1>Delete</h1>
<h3>Are you sure you want to delete this?</h3>
<div>
  <h4>Make</h4>
  <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>
    <dt class="col-sm-2">@Html.DisplayNameFor(model => model.TimeStamp)</dt>
    <dd class="col-sm-10">@Html.DisplayFor(model => model.TimeStamp)</dd>
  </dl>
  <form asp-action="Delete">
    <input type="hidden" asp-for="Id"/>
    <input type="hidden" asp-for="TimeStamp"/>
    <input type="submit" value="Delete" class="btn btn-danger"/> |
    <a asp-action="Index">Back to List</a>
  </form>
</div>
```

# Part 5: Update the Main Menu

### Step 1: Add the Index view

1) Open the _Menu.cshtml file located in \Views\Shared\Partials folder. Update the menu to add a link to the Makes Admin screens:

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

# Summary

The lab created the admin area and finished this workshop.