.NET 10 App Dev Hands-On Lab

RP Lab 1 – The Razor Pages Project

This lab walks you through creating the ASP.NET Core Web Application (Razor Pages) project and adding/updating the NuGet packages. Before starting this lab, you must have completed EF Lab 6.

Part 1: Create the Project

- The `razor` template is highly configurable. Options can be explored by `using -h` (help):

```
dotnet new razor -h
```

- Create the ASP.NET Core web application project using the model-view-controller pattern:

```
[Windows]
dotnet new razor -lang c# -n AutoLot.Web -au none -o .\AutoLot.Web -f net10.0
```

- Add the project to the solution and project references:

```
dotnet sln AutoLot.sln add AutoLot.Web
dotnet add AutoLot.Web reference AutoLot.Models
dotnet add AutoLot.Web reference AutoLot.Dal
dotnet add AutoLot.Web reference AutoLot.Services
```

- Add the required NuGet packages to the project (if using a regular command prompt, remove the single quotes):

```
dotnet add AutoLot.Web package LigerShark.WebOptimizer.Core -v [3.*,4)
dotnet add AutoLot.Web package Microsoft.Web.LibraryManager.Build -v [3.*,4.0)
dotnet add AutoLot.Web package Microsoft.EntityFrameworkCore.SqlServer -v [10.*,11.0)
dotnet add AutoLot.Web package Microsoft.EntityFrameworkCore.Design -v [10.*,11.0)
dotnet add AutoLot.Web package Microsoft.VisualStudio.Threading.Analyzers -v [17.*,19.0)
dotnet add AutoLot.Mvc package Microsoft.Build
```

Part 2: Disable Nullable Reference Types

Coplot Agent Mode

```
Prompt: Update the AutoLot.Web project file to set nullable reference types to disable and remove
the nullable indicator on the ErrorModel.cs file in the AutoLot.Web project.
```

Manual

- Open the new project file (`*.csproj`) and update the `PropertyGroup` to the following (change is in bold):

```
<PropertyGroup>
  <TargetFramework>net10.0</TargetFramework>
  <ImplicitUsings>enable</ImplicitUsings>
  <Nullable>disable</Nullable>
</PropertyGroup>
```

- Remove the nullable reference indicator on the `Error.cshtml.cs` page in the `Pages` folder:

```
public string RequestId { get; set; }
```

Part 3: Adjust the launchsettings.json file

Open the `launchsettings.json` file (in the `project's Properties directory`) and move the `HTTPS` profile to the top.

NOTE: In current versions of ASP.NET Core, the IIS profile doesn't exist until you launch your app using IIS in Visual Studio.

Summary

This lab created the ASP.NET Core Web Application project and added the NuGet packages and the appropriate references.

Next steps

You will start building the services layer in the next part of this tutorial series.