.NET 10 App Dev Hands-On Lab

MVC Lab 1 – MVC Project

This lab guides you through creating an ASP.NET Core Web Application using the MVC pattern project and adding or updating NuGet packages. Before starting this lab, you must have completed EF Lab 6.

Part 1: Creating the Project

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

```
dotnet new mvc -h
```

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

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

- Add the project to the solution and project references:

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

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

```
dotnet add AutoLot.Mvc package LigerShark.WebOptimizer.Core -v '[3.*,4)'
dotnet add AutoLot.Mvc package Microsoft.Web.LibraryManager.Build -v '[3.*,4.0)'
dotnet add AutoLot.Mvc package Microsoft.EntityFrameworkCore.SqlServer -v '[10.*,11.0)'
dotnet add AutoLot.Mvc package Microsoft.EntityFrameworkCore.Design -v '[10.*,11.0)'
dotnet add AutoLot.Mvc 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.Mvc project to set nullable reference types to disable and remove the
nullable indicator on the ErrorViewModel.cs file in the AutoLot.Mvc project.
```

Manual

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

```
<Nullable>disable</Nullable>
```

- Remove the nullable reference indicator on the `ErrorViewModel.cs` class::

```
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 added an ASP.NET Core Web Application project to the solution, along with the required NuGet packages and references.

Next steps

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