03. [2023-03-06] S1L03 Session 03

Roadmap toward working with the ASP.NET Model-View-Controller (MVC) Pattern.

Overview Notes

You can follow the following sequence as an overview. This is an “exploration” guideline and not a step-by-step guide.

1) Creating an MVC project.

- Focussing on community edition versus the professional edition.
    - Licensing.
    - Features.
    - Team collaborations.
    - Target platforms.
    - Extensions.
- Similarities between the different versions of ASP MVC.
    - Architecture.
    - Routing.
    - Razor views.
    - HTML helpers.
    - Testability.
- Start with a new project.
- Search ASP.NET MVC search options.
    - 2 are C# and 2 are Visual Basic.
    - Create new project (ASP.NET Web Application (.NET Framework)) and not the (.NET Core) or Core web applications.
    - Give it a decent name such as S1L03 with solution name S1L03.
    - Folder references.
    - Authentication options.

2) Loading of NuGet packages.

- Microsoft Package Manager.
- For example, jQuery added.
- Bootstrap added (HTML & CSS framework originally from Twitter) added as NuGet package.

3) Look at NuGet packages.

- Right click on References "Manage NuGet package".
- Do not just update update carefully and selectively.
- Be careful about the "prelease options".
- Update all can cause issues.
- Pre-release still in development Do not enable.
- Keep it simple Use defaults when starting.

4) Microsoft general name for web-based applications.

- MVC >> Model-View-Controller. MVC code is compiled and run on the server.
- User do not get C# code separation of concerns. MVC is a User Interface centric pattern.
- Program logic should be mostly in the class library and attach it to the MVC application.

5) Overview of MVC.

- Controllers links the views with the data.

6) HomeController.

  - HomeController Sort of empty with a sample to get you started.
  - It starts with the default working pattern that needs to be expanded.
  - Take note of the naming conventions.

7) Index.cshtml.

- The .cshtml file is a file extension used in ASP.NET MVC (Model-View-Controller) web applications.
- It stands for "C# Server Page with HTML".
- It starts with general boilerplate code.
- Boilerplate code repetitive code that does very little.
- Boilerplate code important so as to show the starting point / structure.
  - Literally from the plate on a boiler it is descriptive details about the boiler that is on a plate.
  - It is repetitive but is important to keep the boiler shut (keep the process going).
- The view is part of the HTML page.

8) Model Create a model.

- In the Model View Controller (MVC) architectural pattern, the "Model" represents the data and business logic.
- When we start there is no model A model is a class with properties.
- Create model and Add Class called PersonModel.
- Type “prop” and then press tab twice (“tab tab”) etc.
- Add decorators (look at how we add decorators in the example).
- The decorators (also known as attributes) are special annotations that can be applied to model class properties.
- Decorators provide additional information about how they should be handled by the framework.
  - [Required]: Specifies that the property is required and must have a value.
  - [StringLength]: Specifies the maximum and/or minimum length of a string property.
  - [Range]: Specifies the allowable range of values for numeric properties.
  - [RegularExpression]: Specifies a regular expression pattern that the property value must match.
  - [Display]: Specifies the display name for a property.
  - [DataType]: Specifies the data type of a property.
  - [DisplayName]: Specifies the display name for a property.
  - [DisplayFormat]: Specifies the format for displaying a property's value.
  - [Bind]: Specifies which properties of a model should be included or excluded from model binding.
  - [AllowAnonymous]: Specifies that a controller or action should be accessible to anonymous users.

9) Go to the Shared folder to the _Layout.cshtml file.

- In ASP.NET MVC, the _Layout.cshtml page is a view file that provides a common layout or template.
- It is like a master page that provides a consistent look and feel for the entire application.
- Contains the additional HTML file details.
- Look at the Render().
- Look at the Razor syntax mixing HTML and C#.
- The At indicates we are going to look at HTML and C#.

10) Run it to see what it looks like.

- Look at URL and ViewBag.
- The ViewBag is a dynamic property that is used to pass data from the controller to the view.
- The controller creates a new instance of the ViewBag property and assigns values to it.
- The values can then be accessed in the view, where they are used to render the HTML responses.
- Can make modifications to HTML and not the C#.

11) App_Start folder (Routing).

- The RouteConfig.cs file is a configuration file that defines the URL routing rules for an application.
- The routing rules determine how incoming requests are mapped to controller actions and views.

12) The bundles App_Start.

- You find the scripts in the footer of _Layout.cshtml.
- Then in the BundleConfig.cs we can add styles and script bundles.
- The BundleConfig.cs file in the App_Start folder is used to configure bundling of CSS and JavaScript files.

13) Content Site.css.

- This is your site or page specific CSS that you would like to add.

14) @RenderSection.

- Where you load all the custom JavaScript.
- Sequence of script execution is important.
- Any script in the View snippet (for example Index.cshtml) will only be rendered when you get to @Render.

15) Extra details we are leaving alone for now.

- _ViewStart.cshtml Where your default layout would be located if you would like to change it but for now leave it alone.
- Global.asax where the application actually starts.
- Where routes and bundles are registered. You can add for example a global login option if required.

16) Add PeopleController.

- On Controllers right click and indicate Add Controller.
- Choose to add empty framework.
- Choose empty Look how default is highlighted (helper) Name controller with "something" with controller at the end.
- Scaffold the controller for me.
- Go to the PeopleController.

17) Look in the folders we have a people folder all of a sudden.

- Create matching cshtml file Right click Add View.
- Use the layout page use the _Layout.cshtml file Leve it empty. It will link to the file.
- Can add different layouts for users, guests, etc depending on the user. By default, it uses the _Layout.cshtml.

18) Add a list of people to the Model.

- The list is our fake data When we do databases, we will get the data from a database or Entity Framework.
- Right click on ListPeople and add the View.
    - Template we choose is "List" Gives you a structure to work from.
    - Model Class is "PersonModel".
    - If model is not in the list, close all the code windows and build the project.

19) ListPeople.cshtml.

- @model IEnumerable<S1L03.Models.PersonModel> Interface Enumerable List.
- Now you have a list template with CRUD links. These links do not work.
- It is part of the pattern and should be updated / changed to ensure eventual functionality.
- Run it so that you can see how it looks (data from model).

20) Go to PersonModel and add decorators.

- using System.ComponentModel.DataAnnotations; // Add to project.

21) What are 404 errors in MVC?

- A 404 error is a reasonably common error.
- A 404 error is an HTTP error code that indicates that the requested resource was not found on the server.
- The server could not find the URL that was requested.
- Usually, the reference or link on the page is incorrectly referenced.
- Can also happen when a function's code is wrong or server is unavailable.

22) To Publish.

- Right click on S1L03 Publish and choose the publishing option. We are not doing this now.