ASP.NET Core MVC Applications

By

Narasimha Rao T

Microsoft.Net FSD Trainer

Professional Development Trainer

tnrao.trainer@gmail.com

1. MVC Pattern Overview

What is MVC?

MVC stands for **Model–View–Controller**, a software design pattern used to separate application logic into three interconnected components:

- **Model** → Manages data and business logic.
- **View** → Defines UI representation.
- **Controller** → Handles requests and coordinates between Model & View.

Advantages of MVC

- Separation of concerns.
- Easier to test and maintain.
- Parallel development (UI team, business logic team, data team).
- Reusability of components.

2. ASP.NET Core MVC Applications

ASP.NET Core MVC is a **framework** for building **web applications** and **APIs** using the MVC pattern.

Key Features:

- Built-in **Routing**.
- Strongly-typed **Views** with **Razor**.
- **Model binding** & **validation**.
- **Dependency Injection** support.
- Cross-platform execution.

3. Working with Controllers

A **Controller**:

- Is a C# class derived from `Controller` or `ControllerBase`.
- Handles incoming HTTP requests.
- Uses **Action Methods** to process requests and return results.

Example:

```
public class HomeController : Controller
{
    public IActionResult Index()
    {
        return View();
    }
}
```

4. Details About Controller

- Stored in the **Controllers** folder (by convention).
- Named with the suffix **`Controller`** (e.g., `ProductController`).
- Uses **Routing** to map URLs to actions.
- Can have filters for authentication, caching, logging.

5. Action Methods

- Public methods in a controller that handle requests.
- Return an **Action Result** or raw data.
- Action method names are part of the URL (unless routing overrides).

Example:

```
public IActionResult About()
{
    return View();
}
```

Rules:

- Must be **public**.
- Cannot be **static**.
- Cannot be **overloaded** only by parameter type.

6. Different Types of Action Results

ASP.NET Core MVC supports **multiple return types** for actions.

Action Result Type	Description
ViewResult	Returns an HTML view.
JsonResult	Returns JSON data.
ContentResult	Returns plain text.
FileResult	Returns a file for download.
RedirectResult	Redirects to a URL.
RedirectToActionResult	Redirects to another action method.
StatusCodeResult	Returns an HTTP status code.

Example:

```
public IActionResult GetJson()
{
    return Json(new { Name = "ASP.NET Core", Version = "8.0" });
}
```

7. View Development

What is a View?

- A `.cshtml` file containing HTML + C# code.
- Displays data to the user.
- Stored in the **Views** folder (by convention).

Flow:

Controller → Passes data (Model) → View → Renders HTML.

8. Razor Engine

- **Razor** is the view engine for ASP.NET Core.
- Allows mixing **C#** and **HTML** in `.cshtml` files.
- Uses `@` syntax to switch between HTML and C#.

Example:

```
<h1>Hello @Model.Name</h1>
<p>Today is @DateTime.Now.ToLongDateString()</p>
```

9. Razor Programming

Razor Syntax Basics:

- Variables: `@varName`
- C# blocks: `@{ ... }`
- Loops:

```
@foreach (var item in Model)
{
    <li>@item</li>
}
```

- Conditional:

```
@if (Model.Count > 0)
{
    <p>Items found.</p>
}
```

10. Adding Views

Steps to Create a View

1. Create a `.cshtml` file inside the **Views** folder.
2. Name it the same as the action method.
3. Use `return View()` in the controller.
4. Optionally pass a **Model** to the view.

Example:

```
public IActionResult ProductDetails()
{
    var product = new Product { Id = 1, Name = "Laptop" };
    return View(product);
}
```

`Views/Product/ProductDetails.cshtml`:

```
@model Product
<h1>@Model.Name</h1>
```

upGrad ENTERPRISE

1. What is middleware in ASP.NET Core?
2. How is middleware executed in the request pipeline?
3. How do you create custom middleware in ASP.NET Core?
4. Give some examples of built-in middleware components in ASP.NET Core.
5. How is data passed from a controller to a view in ASP.NET Core?
6. What is the difference between ViewData, ViewBag, TempData, and a strongly typed view?
7. What is the difference between IActionResult and ActionResult<T>?
8. How do you return JSON data from a controller in ASP.NET Core?

Q & A

Narasimha Rao T
tnrao.trainer@gmail.com