**Section in a Layout View**  
  
A layout page in ASP.NET Core MVC can also include a section. A section can be optional or mandatory. It provides a way to organize where certain page elements should be placed.

**Example**   
You have a custom JavaScript files that is only needed by a few views in your application. It's a good practice to place the script files at the bottom of the page before the closing </body> tag.   
  
If the custom script file is required by all the views then we could place it in the Layout page as shown below.

<html>

<head>

    <meta name="viewport" content="width=device-width" />

    <title>@ViewBag.Title</title>

</head>

<body>

    <div>

        @RenderBody()

    </div>

    <script src="~/js/CustomScript.js"></script>

</body>

</html>

In our case, we do not need the custom JavaScript file in every view. Let's assume we only need it in the Details view but not in the other views. This is one example where we could make use of a **section.**   
  
**Rendering the Section**  
  
In the layout page, call RenderSection() method at the location where you want the section content to be rendered. In our case, we want the script file to be included just before the closing </body> tag. This is the reason we have placed the call to @RenderSection() method just before the closing </body> tag.    
  
RenderSection() method has 2 parameters. The first parameter specifies the name of the section. The second parameter parameter specifies if the section is required or optional.

<html>

<head>

    <meta name="viewport" content="width=device-width" />

    <title>@ViewBag.Title</title>

</head>

<body>

    <div>

        @RenderBody()

    </div>

    @RenderSection("Scripts", required: false)

</body>

</html>

If required is set to true and if a content view does not include the section, we get the following error.   
  
InvalidOperationException: The layout page '/Views/Shared/\_Layout.cshtml' cannot find the section 'Scripts' in the content page '/Views/Home/Index.cshtml'.   
  
**Making the layout section optional**   
  
There are 2 options to mark a layout section as optional   
  
**Option 1 :**Set required parameter of RenderSection() method to false

@RenderSection("Scripts", required: false)

**Option 2 :** Use IsSectionDefined() method

@if (IsSectionDefined("Scripts"))  
{  
    @RenderSection("Scripts", required: false)  
}

**Providing Section Content**  
  
Every view that intends to provide content for the section, must include a section with the same name. We use @section directive to include the section and provide the content as shown below.   
  
In our case, we want the details view to include <script> tag in the "Scripts" section in the layout page. To achieve this we include "Scripts" section in Details.cshtml as shown below.

@model EmployeeManagement.ViewModels.HomeDetailsViewModel

@{

    Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h3>@Model.PageTitle</h3>

<div>

    Name : @Model.Employee.Name

</div>

<div>

    Email : @Model.Employee.Email

</div>

<div>

    Department : @Model.Employee.Department

</div>

@section Scripts {

    <script src="~/js/CustomScript.js"></script>

}

**Testing**  
  
With these changes in place, if we navigate to /Home/Details we see the <script> tag included just before the closing </body> tag. If we navigate to /Home/Index we do not see the <script> tag.