

## BÀI THỰC HÀNH 2

### Layout

- Sử dụng web template làm layout cho ứng dụng
- Tách layout thành các partial để sử dụng một cách linh hoạt
- **@RenderBody():** render view
- **@RenderSection:** render section
- Sử dụng **ViewComponent** để hiển thị **Strongly Typed PartialView**

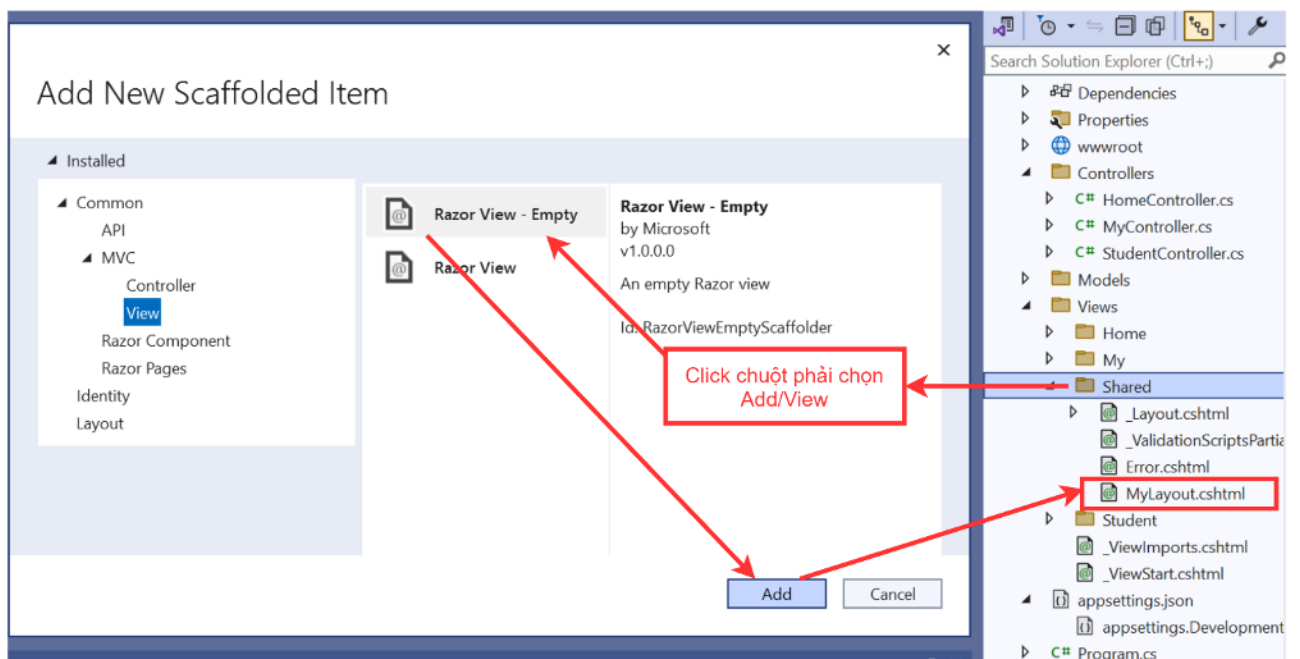
### Hướng dẫn

#### 1. Sử dụng web template làm layout

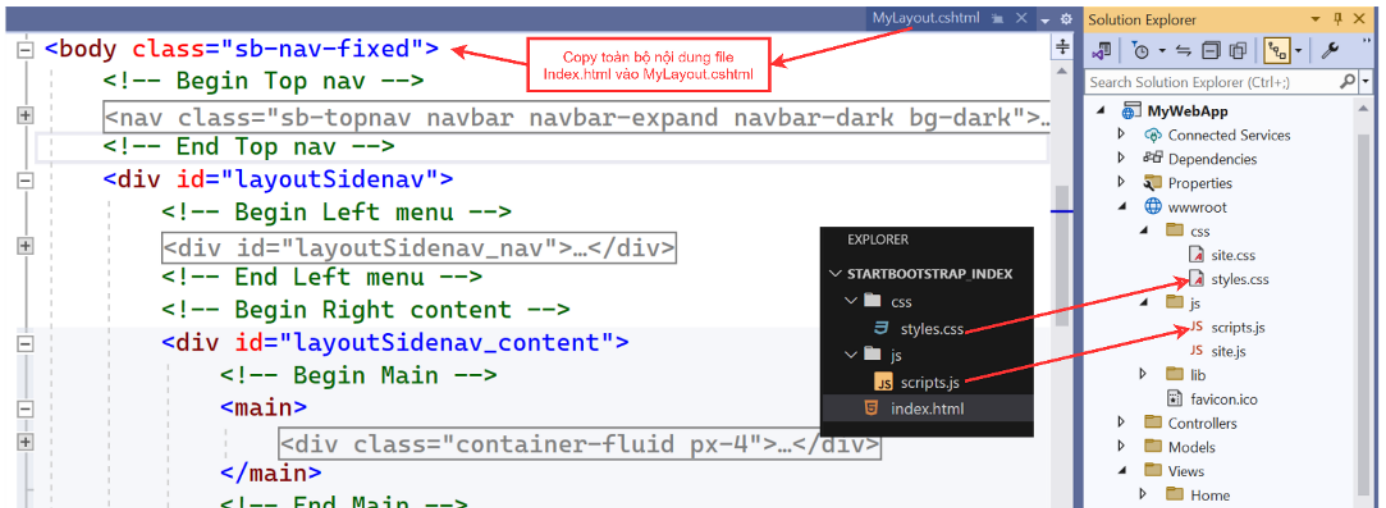
Trong thư mục web template thường sẽ có:

- File `index.html` là trang template chính
- Thư mục `con` chứa các file style sheet (`.css`) dùng trong giao diện
- Thư mục `con` js chứa các file java script (`.js`) dùng trong giao diện

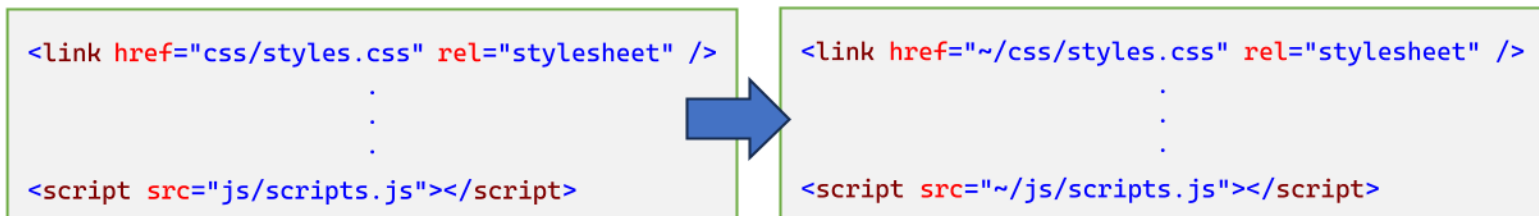
Trong thư mục `View/Shared`, click chuột phải chọn `Add/View` rồi chọn `Razor View Empty`, đặt tên file là `MyLayout.cshtml` như hình vẽ



- Copy toàn bộ nội dung file Index.html vào file MyLayout.cshtml
- Copy file css/style.css vào wwwroot/css
- Copy file js/script.js vào wwwroot/js

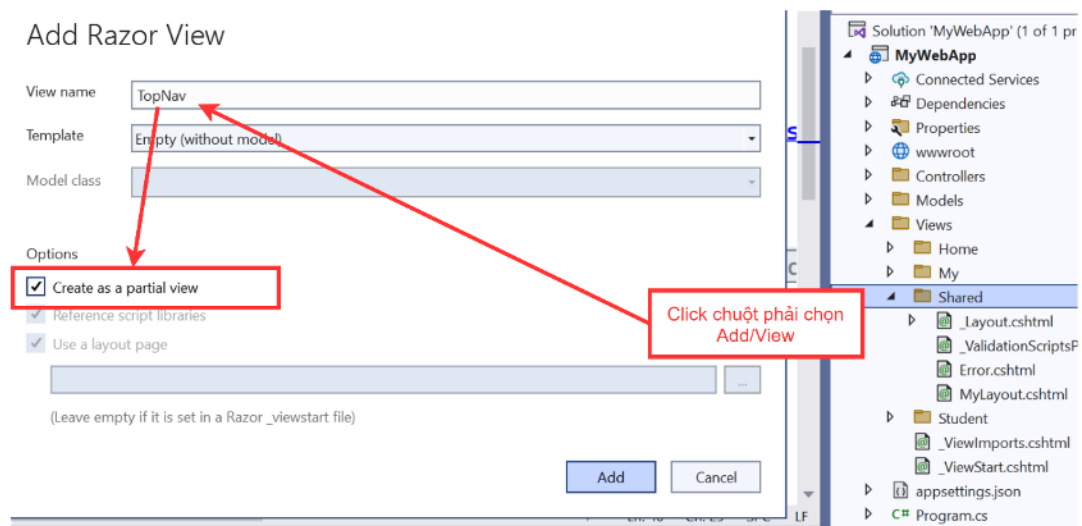


Trong file MyLayout.cshtml, sửa lại đường dẫn đến các file .css và .js trong các thẻ link, script



## 2. Tách các phần trong layout thành các partial view

Tạo partial view và đặt tên là TopNav.cshtml



Trong MyLayout.cshtml, đánh dấu toàn bộ phần Top nav chọn cut

```
<body class="sb-nav-fixed">
```

```
<!-- Begin Top nav -->
```

```
<nav class="sb-topnav">
```

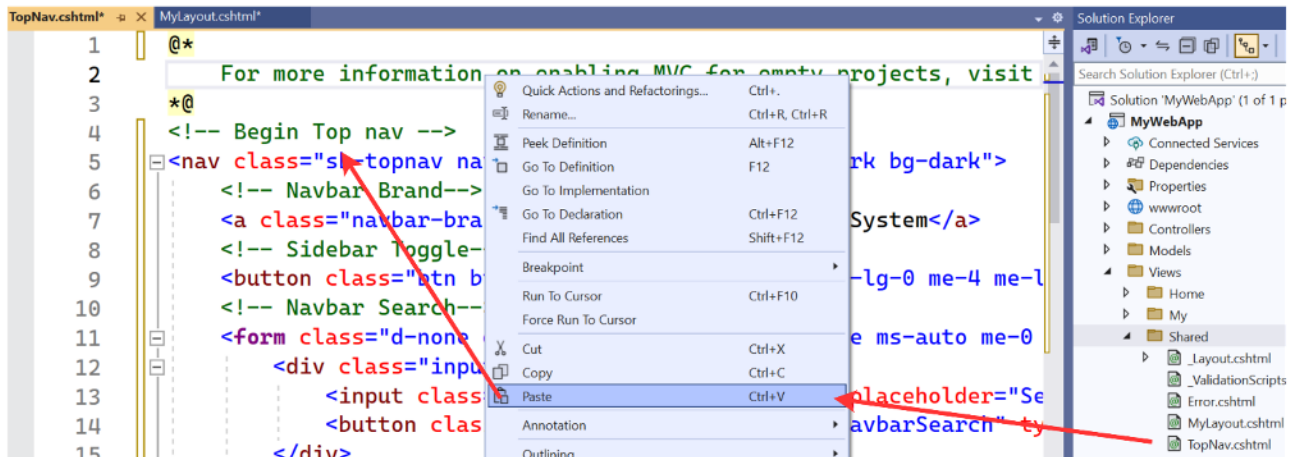
```
<!-- End Top nav -->
```

```
<div id="layoutSide">
```

```
<!-- Begin Left
```

Go To Controller	Ctrl+M,
Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Del

Paste vào file TopNav.cshtml



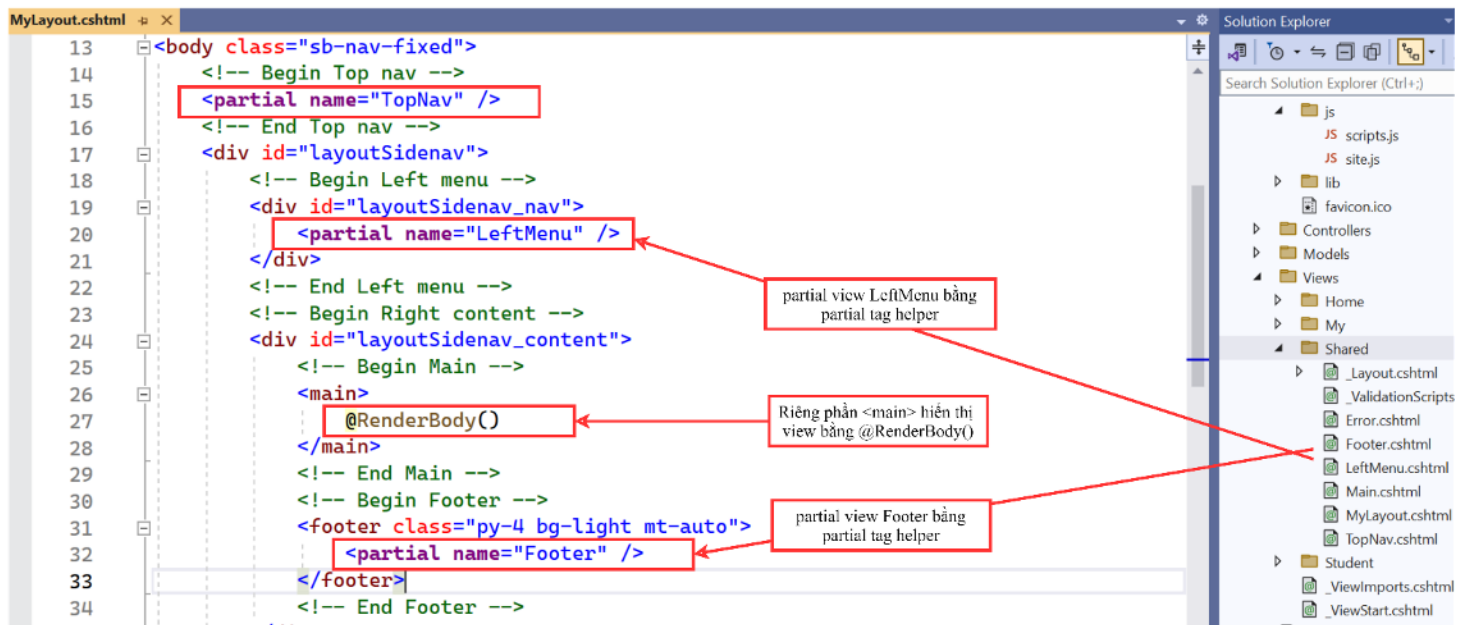
Trong file MyLayout.cshtml, hiển thị partial view TopNav bằng partial tag helper như sau



Làm tương tự để

- Tạo Partial View LeftMenu.cshtml và hiển thị lại trên MyLayout.cshtml bằng partial tag helper
- Tạo Partial View Footer.cshtml và hiển thị lại trên MyLayout.cshtml bằng partial tag helper

Riêng phần main hiển thị view bằng @RenderBody()



### 3. Cấu hình Layout

Trong view nào muốn sử dụng MyLayout thì cấu hình layout theo cú pháp

@{

Layout = "~/Views/Shared/MyLayout.cshtml";

}

```

@{
    ViewData["Title"] = "Student List";
    Layout = "~/Views/Shared/MyLayout.cshtml";
}
@model IEnumerable<Student>
<a asp-action="Create" class="btn btn-primary">Create Student</a>
<table class="table table-striped" >
    ...
</table>

```

Kết quả chạy Student/Index

localhost:7255/Student/Index

Management System								
Create Student								
Mã	Họ tên	Ngày sinh	Ngành	Giới tính	Hệ ĐT	Địa chỉ	Email	
101	Hải Nam	1/1/0001 12:00:00 AM	CNTT	Nam	Chính quy	A1-2018	nam@g.com	
102	Minh Tú	1/1/0001 12:00:00 AM	Kinh tế	Nữ	Chính quy	A1-2019	tu@g.com	
103	Hoàng Phong	1/1/0001 12:00:00 AM	Công trình	Nam	Phi chính quy	A1-2020	phong@g.com	
104	Xuân Mai	1/1/0001 12:00:00 AM	Điện - Điện tử	Nữ	Phi chính quy	A1-2021	mai@g.com	

### 4. Sử dụng RenderSection

Trong view nào muốn tạo Section sử dụng cú pháp

```
@section [Tên Section]{
    //Đoạn code bên trong section
}
```

Ví dụ : Trong View Student/Index.cshtml tạo section Scripts như sau

```
@section Scripts{
    <script>
        var table = document.querySelector("table");
        table.onclick = function onClick() {
            table.style = "background-color: yellow;"
        }
    </script>
}
```

Muốn render section này lên Layout thì trong MyLayout.cshtml gọi @RenderSection theo cú pháp:

```
@RenderSection("Scripts", required:false);
```

Trong đó required: false có nghĩa là không bắt buộc trong view phải có Section Scripts

```
MyLayout.cshtml
<!-- Begin Main -->
<main>
    @RenderBody()
</main>
<!-- End Main -->
<!-- Begin Footer -->
<footer class="py-4 bg-light mt-auto">
    <partial name="Footer" />
</footer>
<!-- End Footer -->
</div>
<!-- End Right content -->
</div>
<script src="~/js/scripts.js"></script>
@RenderSection("Scripts", required:false);
</body>
```

## 5. Sử dụng ViewComponent để hiển thị Strongly Typed PartialView

Trong ví dụ trên, các PartialView: TopNav, LeftMenu, Footer cố định, không được ràng buộc với bất cứ một cấu trúc dữ liệu nào.

Nhiều trường hợp layout được cấu thành từ các PartialView được ràng buộc với một Model nào đó gọi là các Strongly Typed PartialView. Đây là các trường hợp phải dùng tới ViewComponent

Ví dụ: LeftMenu là một Strongly Typed PartialView với Model là một collection các MenuItem (có thể được load từ cơ sở dữ liệu). Trong ví dụ này ta giả lập một collection các MenuItem được khởi tạo trong ViewComponent.

← → ↻ localhost:7255/Student/Index

Management System

Search for...

CREATE STUDENT

Mã	Họ tên	Ngày sinh	Ngành	Giới tính	Hệ ĐT	Địa chỉ	Email
101	Hải Nam	1/1/0001 12:00:00 AM	CNTT	Nam	Chính quy	A1-2018	nam@g.com
102	Minh Tú	1/1/0001 12:00:00 AM	Kinh tế	Nữ	Chính quy	A1-2019	tu@g.com
103	Hoàng Phong	1/1/0001 12:00:00 AM	Công trình	Nam	Phi chính quy	A1-2020	phong@g.com
104	Xuân Mai	1/1/0001 12:00:00 AM	Điện - Điện tử	Nữ	Phi chính quy	A1-2021	mai@g.com

CORE

- Dashboard

ADDONS

- Branches
- Students
- Subjects
- Courses

### 5.1. Trong thư mục Models, tạo class MenuItem

```
namespace MyWebApp.Models
{
    public class MenuItem
    {
        public int Id { get; set; } //Item id
        public string Name { get; set; } //Item name
        public string Link { get; set; } //Item label
    }
}
```

### 5.2. Tạo ViewComponent theo các nguyên tắc sau:

- Đặt tên class kết thúc bởi hậu tố ViewComponent
- Thường đặt trong thư mục ViewComponent
- Kế thừa từ lớp ViewComponent hoặc đặt thuộc tính [ViewComponent]
- Nếu muốn hiển thị View theo cơ chế đồng bộ thì định nghĩa phương thức Invoke theo cú pháp

```
public IActionResult Invoke([tham số nếu có]) {
    ...
}
```

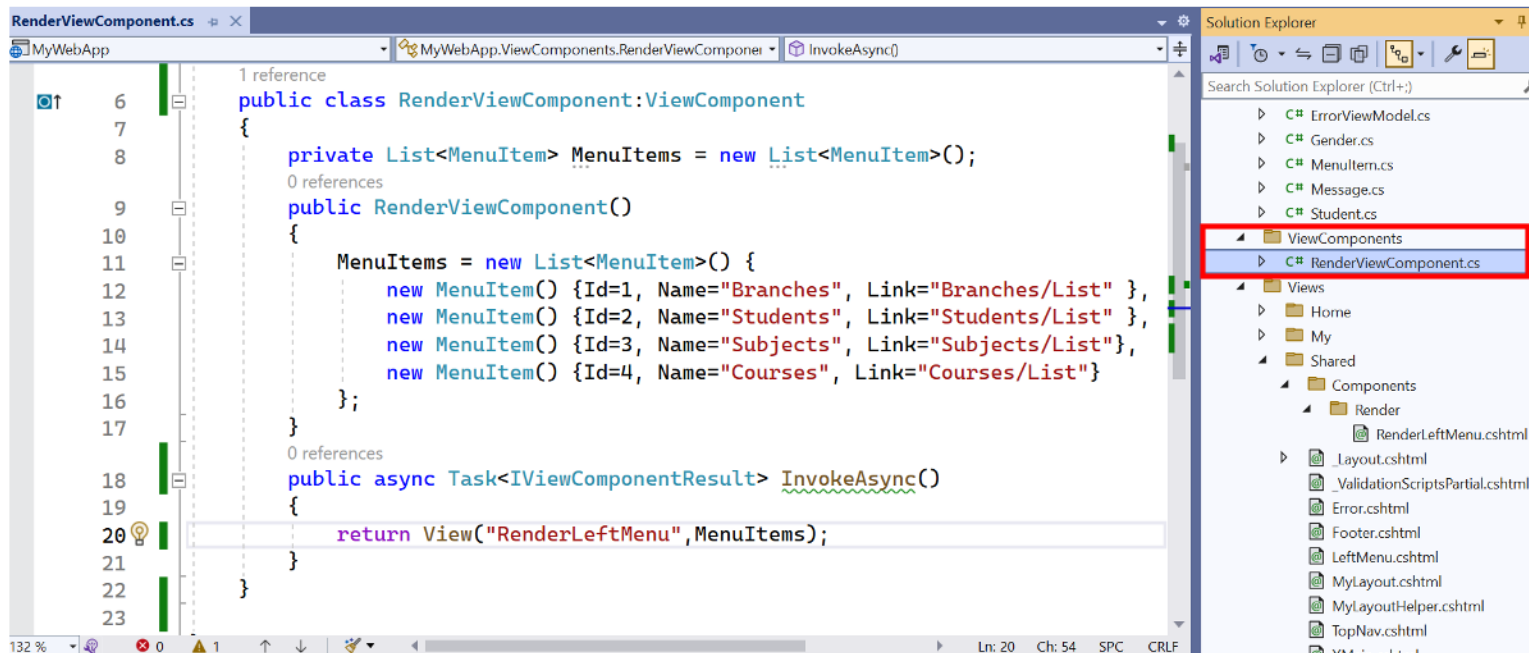
- Nếu muốn hiển thị View theo cơ chế bất đồng bộ thì định nghĩa phương thức InvokeAsync theo cú pháp

```
public async Task<IViewComponentResult> InvokeAsync([tham số nếu có])
{
    ...
}
```

- Nếu trả về View từ các phương thức Invoke/InvokeAsync của ViewComponent thì View phải được đặt trong thư mục Shared/Components/[tên ViewComponent]



Áp dụng các nguyên tắc trên. Trong thư mục dự án, tạo thư mục con ViewComponents (cùng cấp với thư mục Models, Views, Controllers). Trong đó, tạo class RenderViewComponent.cs như sau



The screenshot shows the Visual Studio IDE. The main editor displays the `RenderViewComponent.cs` file. The code defines a class `RenderViewComponent` that inherits from `ViewComponent`. It has a private `List<MenuItem>` named `MenuItems` and a public constructor that initializes it with four `MenuItem` objects. There is also an `InvokeAsync()` method that returns the view `RenderLeftMenu` with the `MenuItems` list. The Solution Explorer on the right shows the project structure, with the `ViewComponents` folder and `RenderViewComponent.cs` file highlighted.

```
1 reference
6 public class RenderViewComponent:ViewComponent
7 {
8     private List<MenuItem> MenuItems = new List<MenuItem>();
9     public RenderViewComponent()
10    {
11        MenuItems = new List<MenuItem>() {
12            new MenuItem() {Id=1, Name="Branches", Link="Branches/List" },
13            new MenuItem() {Id=2, Name="Students", Link="Students/List" },
14            new MenuItem() {Id=3, Name="Subjects", Link="Subjects/List"},
15            new MenuItem() {Id=4, Name="Courses", Link="Courses/List"}
16        };
17    }
18    0 references
19    public async Task<IViewComponentResult> InvokeAsync()
20    {
21        return View("RenderLeftMenu", MenuItems);
22    }
23 }
```

### 5.3. Tạo View được gọi bởi ViewComponent

Trong thư mục Shared, tạo thư mục Components. Trong thư mục Components, tạo thư mục con Render (trùng tên với ViewComponent đã tạo ở trên là `RenderViewComponent.cs`).

Trong thư mục Render tạo Razor View - empty có tên là `RenderLeftMenu.cshtml`

Copy toàn bộ nội dung file `LeftMenu.cshtml` sang `RenderLeftMenu.cshtml`

Định kiểu cho `RenderLeftMenu.cshtml` là `IEnumerable<MenuItem>`

Render danh sách các `MenuItem` bằng Razor



The screenshot shows the Visual Studio IDE. The main editor displays the `RenderLeftMenu.cshtml` file. The code is a Razor view that takes an `IEnumerable<MenuItem>` as a model. It uses HTML tags to create a sidebar navigation menu. A `@foreach` loop iterates over the `Model` to generate links for each `MenuItem`. The Solution Explorer on the right shows the project structure, with the `RenderLeftMenu.cshtml` file highlighted.

```
1 @model IEnumerable<MenuItem>
2 <div id="layoutSidenav_nav">
3     <nav class="sb-sidenav accordion sb-sidenav-dark" id="sidenavAccordion">
4         <div class="sb-sidenav-menu">
5             <div class="nav">
6                 <div class="sb-sidenav-menu-heading">Core</div>
7                 <a class="nav-link" href="index.html">
8                     <div class="sb-nav-link-icon"><i class="fas fa-tachometer-alt">
9                         Dashboard
10                    </i>
11                </a>
12                <div class="sb-sidenav-menu-heading">Addons</div>
13                @foreach(var i in Model)
14                {
15                    <a class="nav-link" href="@i.Link">
16                        <div class="sb-nav-link-icon"><i class="fas fa-chart-area">
17                            @i.Name
18                        </i>
19                    </a>
20                }
21            </div>
22        </div>
23        <div class="sb-sidenav-footer">
```

#### 5.4. Tạo Layout sử dụng ViewComponent

Trong thư mục Shared, tạo Razor View – empty tên là MyLayoutHelper.cshtml

Copy toàn bộ nội dung file MyLayout.cshtml sang MyLayoutHelper.cshtml

Gọi ViewComponent: Thay thế dòng `<partial name="LeftMenu" />`

Bằng dòng `@await Component.InvokeAsync("Render");`

Cấu hình lại Layout cho Student/Index thành MyLayoutHelper và test thử.

