educación IT
CODE YOUR FUTURE

Programación Web .NET Core

Módulo 4 - Laboratorio adicional

Para poder realizar este laboratorio, se recomienda...

- Revisar contenidos previos.
- Instalar los laboratorios y demostraciones: ***Allfiles***.
- Tener instalado **SQL Server.**

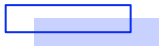
Introducción al laboratorio

Escenario

Trabajas como desarrollador junior en **AdventureWorks**.

Un desarrollador senior te ha pedido que investigues la posibilidad de crear una aplicación **CoreMVC** basada en **webASP.NET** para los clientes de su organización, similar a la que ha visto en Internet. Dicha aplicación promoverá una comunidad de ciclistas que utilizan equipos de **AdventureWorks**, y los miembros de la comunidad podrán compartir sus experiencias.

Esta iniciativa pretende aumentar la popularidad de **AdventureWorksCycles**, y así aumentar sus ventas. Se te ha pedido que comiences la planificación de la solicitud. También se te ha pedido que examines el modelo **Razor Pages**.



Objetivos

En este laboratorio crearás una sencilla aplicación **Páginas de Razor** y explorarás su estructura.

Las principales tareas para este ejercicio son:

1. Crear una aplicación web **Razor Pages.**
2. Explorar la estructura de la aplicación.
3. Adicionar de funcionalidades sencillas.
4. Ejecutar la aplicación.

Tiempo estimado: **30 minutos**.

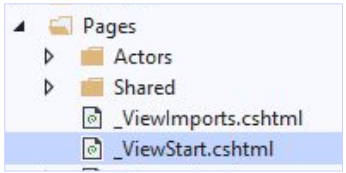
Ejercicio 1: Creación de una aplicación web Razor Pages

1. Iniciar **Visual Studio 2019** y crear una nueva **aplicación web ASP.NET Core** con la siguiente información:
    - Nombre: **ActorsRazorPages**.
    - Ubicación: **Allfiles/Mod01/Labfiles_01_ActorsRazorPages**.
    - Nombre de la solución: **ActorsRazorPages**.
    - Crear directorio para la solución: **True**.
    - Plantilla de proyecto: **Aplicación web**.
    - Habilitar compatibilidad con docker: **False**.
    - Configurar para **HTTPS**: **False**.
    - Establezca la versión **ASP.NET Core** en **2.1**.

2. Ejecutar la aplicación **sin depurar**.
3. En la ventana **Microsoft Edge,** ver la página **Contacto.**

Ejercicio 2: Explorar la estructura de la aplicación

1. En el archivo ***_ViewStart.cshtml***, revisar el valor de **Layout es "_Layout".**
2. En la carpeta ***Shared*** buscar la página ***_Layout.cshtml***



Ver si en el elemento HEAD está el vínculo a la página ***./css/site.css.***

Nota: Este es el archivo de hoja de estilos CSS que se aplica en *_Layout.cshtml.*

3. Localizar, abrir y revisar rápidamente el archivo **.css sitio.**
4. En la página ***Contact.cshtml***, observar que no hay vínculos a los archivos .css.

Ejercicio 3: Añadir funcionalidades simples

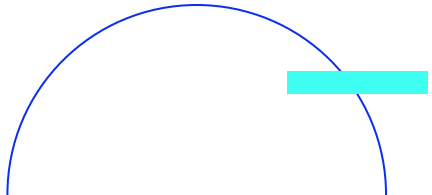
1. Crear una nueva **página de Razor** con la siguiente información:
   - Nombre: **TestPage**.
   - Carpeta: **Pages**.
   - Generar clase PageModel: **True**.
   - Crear como vista parcial: **False**.
   - Bibliotecas de scripts de referencia: **True**.
   - Usar una página de diseño: **True**.
2. Eliminar el contenido de la página ***TestPage.cshtml***.
3. En la página ***TestPage.cshtml***:
   a. Guardar la clave y valor en la propiedad **ViewData**:
      - Clave: **Title.**
      - Valor : **"Test Page".**
   b. Agregar un elemento **H1** con la siguiente información:
      - Contenido: **@ViewData["Title"].**

c. Agregar un elemento **H2** con la siguiente información:
   - Contenido: **Esta es una página de prueba.**

```
@{
ViewData[“Title”] = “Test Page”;
}
<h1> @ViewData[“Title”] </h1>
<h2> This is a Test Page </h2>
```

4. En la página ***_Layout.cshtml***, en el elemento `UL`, agregar un elemento `LI`.
5. En el nuevo elemento `LI`, agregar un elemento `A` con la siguiente información:
   - Contenido: **Test Page**
   - asp-page: **/TestPage**

```
<li>
      <a asp-page=“/TestPage”> Test Page </a>
</li>
```



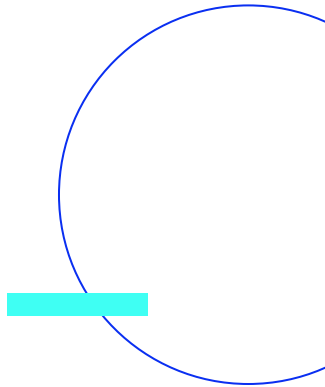
6. Crear una nueva carpeta con la siguiente información:
    - Nombre de la carpeta: **Models**
7. En la carpeta ***Models*** crear un nuevo modelo con la siguiente información:
    - Nombre: **Actor**
8. Al modelo ***Actor,*** agregarle las propiedades de la tabla a la derecha.

Propiedades en el modelo Actor			
Alcance	**Nombre**	**Tipo**	**Acceso**
público	Id	int	Leer y escribir
público	FirstName	cadena	Leer y escribir
público	LastName	cadena	Leer y escribir
público	KnownFor	cadena	Leer y escribir
público	OscarWinner	bool	Leer y escribir
público	ImageName	cadena	Leer y escribir

9. En la carpeta ***Models*** crear una nueva interfaz pública:
    - Nombre: **IData**

Propiedad			
Tipo		**Nombre**	**Acceso**
Lista<Actor>		ActorsList	Leer y escribir
Métodos			
	Tipo retorno	**Nombre**	**Parámetros**
Función	List<Actor>	ActorsInitializeData	Ninguno

10. Copiar el archivo ***Data.cs*** con la siguiente información:
    - Ubicación de la **fuente:**
      Allfiles\Mod01\Labfiles**01_ActorsRazorPage**
    - Ubicación de **destino**:
      Allfiles\Mod01\Labfiles**Labfiles_01_ActorsRazorPages**
      \ActorsRazorPages\Models
11. Copiar la carpeta ***Images*** en el proyecto
    01_ActorsRazorPages, con la siguiente información:
    - Ubicación de la **fuente:**
      Allfiles\Mod01\Labfiles**01_ActorsRazorPages**
    - Ubicación de **destino**:
      Allfiles\Mod01\Labfiles\ **Labfiles_01_ActorsRazorPages**
      \ActorsRazorPages\wwwroot



12. Buscar la carpeta ***Pages*** y crear dentro una nueva carpeta ***Actors.***
13. En la carpeta ***Pages/Actors*** agregar una nueva **Página de Razor** con la siguiente información:
    - Nombre: **Index**
14. En el archivo ***Index.cshtml.cs,*** agregar una instrucción **`using`** para el siguiente espacio de nombres:
    - **ActorsRazorPages.Models**

- Programar en ***Index.cshtml.cs:***

Campo			
Alcance	**Tipo Clase**	**Nombre**	
privado	IData	_data	
Constructor			
Alcance	**Nombre**	**Parámetro**	**Contenido**
público	IndexModel	Tipo: IData Nombre: data	_data = data;
Propiedades			
Alcance	**Nombre**	**Tipo Clase**	**Acceso**
público	Actors	IList<Actor>	Leer y escribir
Método Void			
Alcance	**Nombre**	Contenido	
público	OnGet()	inicializar la propiedad **Actors** mediante el método **ActorsInitializeData** del campo **_data**	

15. Eliminar el contenido de la página ***Index.cshtml***.
16. Copiar el contenido del archivo de texto ***Indexcshtml.txt*** en la página ***Index.cshtml*** considerando:

    Ubicación de la **fuente:**
    Allfiles\Mod01\Labfiles\01_ActorsRazorPages\Pages

17. En la clase ***Startup,*** agregar un **`using`** para el siguiente espacio de nombres:

    - **ActorsRazorPages.Models**

18. En el método **ConfigureServices,** llamar al método **AddSingleton** del parámetro **services** con la siguiente información:
    - Interfaz: **IData**.
    - Implementación: **Datos**.

```
public void
ConfigureServices(IServiceCollection services)
{
        services.AddSingleton<IData, Data>();
```

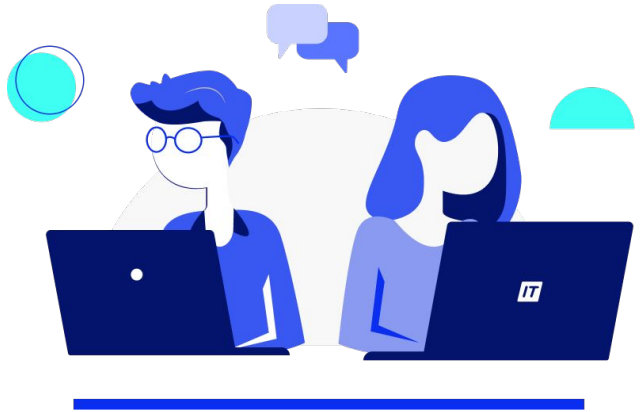
19. En la página ***_Layout.cshtml,*** en el elemento **UL,** agregar un elemento **LI.**

20. En el nuevo elemento **LI**, agregar un elemento **A** con la siguiente información:
    - Contenido: **Actores**
    - asp-page: **/Actors/Index**

```
<ul class="nav navbar-nav">
      <li> <a asp-page="/Index"> Home </a> </li>
      <li> <a asp-page="/About"> About </a> </li>
      <li> <a asp-page="/Contact"> Contact </a> </li>
      <li> <a asp-page="/TestPage"> Test Page </a> </li>
      <li> <a asp-page="/Actors/Index"> Actors </a> </li>
</ul>
```

Ejercicio 4: Ejecutar la aplicación

1. Guardar todos los cambios.
2. Ejecutar la aplicación **sin depurar**.
3. En Microsoft Edge, en la barra de navegación, haz clic en ***Página de prueba*** para ver su contenido.
4. En la ventana ***Página de prueba***, en la barra de navegación, haz clic en ***Actores*** para ver su contenido.
5. Comprobar que el diseño y los estilos del sitio se han aplicado a todas las páginas.



En la sección de **Descargas** encontrarás los recursos necesarios para realizar los ejercicios y su resolución para que verifiques cómo te fue.

educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!