Tipo : Guía de Laboratorio
Capítulo : Aplicando técnicas en una aplicación ASP.NET MVC
Duración : 45 minutos

I. OBJETIVO

Crear mantenimiento para Customer.

II. REQUISITOS

Los siguientes elementos de software son necesarios para la realización del laboratorio:

- Windows 10 (como mínimo Windows 8)
- Visual Studio 2015 (como mínimo Visual Studio 2013)

III. EJECUCIÓN DEL LABORATORIO

- Ejercicio Nº 5.2: Crear mantenimiento para Customer

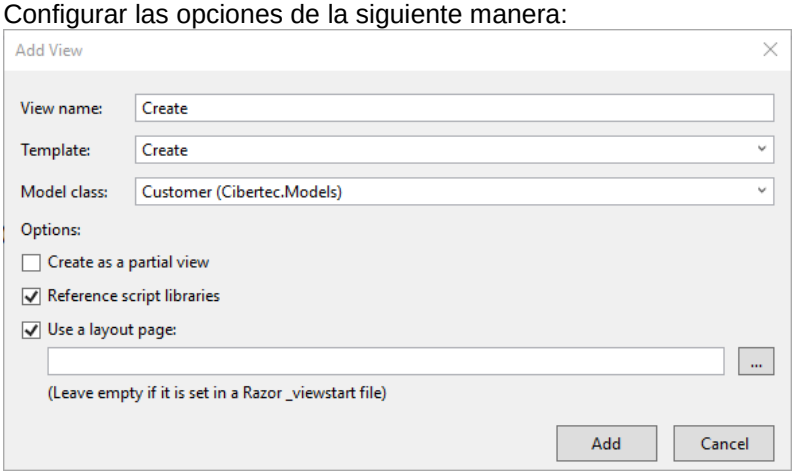
1. Abrir la solución Cibertec.

2. En el proyecto "**Cibertec.Mvc**" abrimos el controlador Customer y agregamos los siguientes métodos:

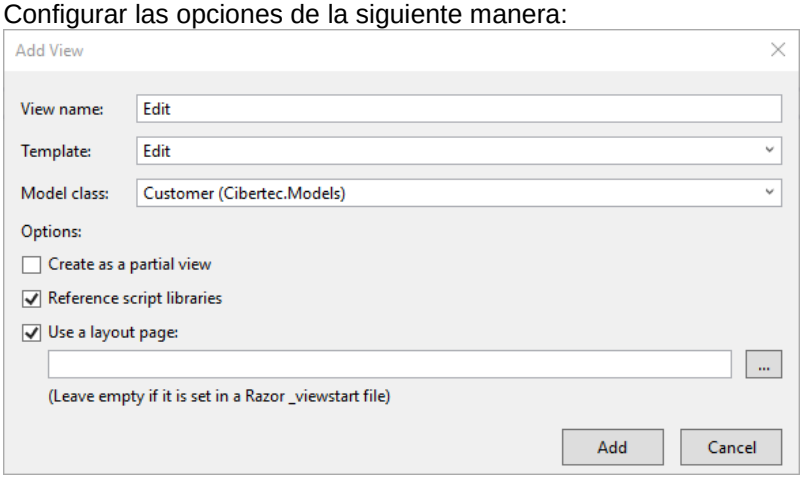
```
public ActionResult Create()
 {
     return View();
 }

 [HttpPost]
 public ActionResult Create(Customer customer)
 {
     if (ModelState.IsValid)
     {
         _unit.Customers.Insert(customer);
         RedirectToAction("Index");
     }
     return View(customer);
 }

 public ActionResult Edit(int id)
 {
     return View(_unit.Customers.GetById(id));
 }

 [HttpPost]
 public ActionResult Edit(Customer customer)
 {
     if (ModelState.IsValid)
     {
         _unit.Customers.Update(customer);
         RedirectToAction("Index");
     }
     return View(customer);
 }
```

```
        public ActionResult Delete(int id)
        {
            return View(_unit.Customers.GetById(id));
        }

        [HttpPost]
        public ActionResult Delete(Customer customer)
        {
            if (_unit.Customers.Delete(customer)) return
RedirectToAction("Index");
            return View(customer);
        }
```

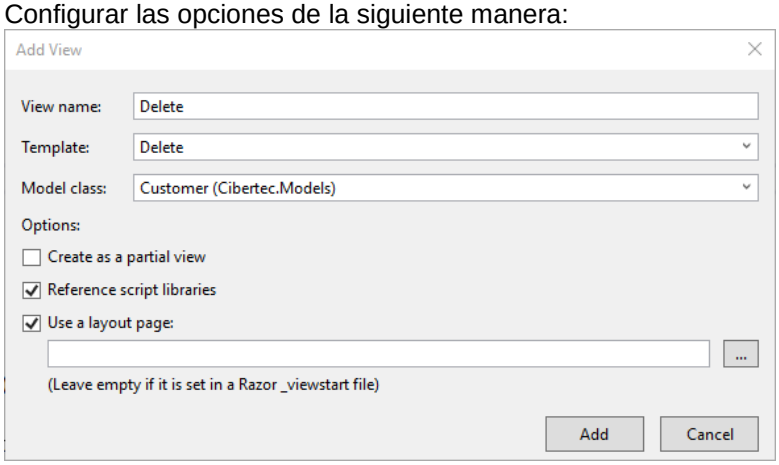
3. Con ayuda del asistente scaffolding de MVC, procedemos a crear el método **Crear**.
   (1) Hacer clic derecho en el método "**Create**" y hacer clic en "**Add View**"
   (2) Configurar las opciones de la siguiente manera:



4. Con ayuda del asistente scaffolding de MVC, procedemos a crear el método **Edit**.
   (1) Hacer clic derecho en el método "**Edit**" y hacer clic en "**Add View**"
   (2) Configurar las opciones de la siguiente manera:



5. Con ayuda del asistente scaffolding de MVC, procedemos a crear el método **Delete**.
   (1) Hacer clic derecho en el método "**Delete**" y hacer clic en "**Add View**"
   (2) Configurar las opciones de la siguiente manera:



6. Modificamos el Layout para facilitar la navegación en nuestra aplicación. Abrimos el "**_Layout.cshtml**" ubicado en la carpeta "**Views\Shared**"

```
<div class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
                <li>@Html.ActionLink("Home", "Index", "Home")</li>
                <li>@Html.ActionLink("Customer", "Index", "Customer")</li>
        </ul>
</div>
```

   Debe de quedar como la siguiente imagen:

```
<div class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
        <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            @Html.ActionLink("Application name", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })
        </div>
        <div class="navbar-collapse collapse">
            <ul class="nav navbar-nav">
                <li>@Html.ActionLink("Home", "Index", "Home")</li>
                <li>@Html.ActionLink("Customer", "Index", "Customer")</li>
            </ul>
        </div>
    </div>
</div>
<div class="container body-content">
    @RenderBody()
    <hr />
    <footer>
        <p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>
    </footer>
</div>
```

7. Procedemos a validar que todo funcione adecuadamente.

IV. EVALUACIÓN

1. ¿Cuáles son las carpetas principales en una aplicación MVC?

   Las carpetas principales son: Controllers, Views, Models y wwwroot

2. ¿Qué pasa si al nombre del controlador no se le coloca el prefijo Controller?

   Por convención todos los archivos que van a ser controladores deben tener el prefijo Controller al final del nombre para que puedan ser reconocidos por Asp.NET como tales.

3. ¿Cuál es el tipo de datos de los métodos del controller?

   El tipo base de todos los controladores es IActionResult.

4. ¿Qué es una vista Layout?

   Es la vista maestra, en ella se establece la estructura base (html) de todas las vistas que tendrá la aplicación web. Algunas vistas pueden no utilizar la vista base.

5. Los modelos, ¿pueden ser librería de clases?

   Sí, los modelos son un concepto que involucra entidades, acceso a datos, reglas de negocio y pueden ser implementadas en librería de clases (dll).

6. ¿Qué son las vistas parciales?

   Al igual que las vistas normales, estas permiten ser reutilizadas en ciertas partes de la aplicación.