

Practice 305 - It's Magic

1. Crea un nuevo modelo para los países.

- En tu carpeta **Models** del proyecto **webacademy.UI** crea una nueva clase con el nombre **CountryDto.cs**.

```
namespace webacademy.Models
{
    public class CountryDto
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Code { get; set; }
    }
}
```

2. Instala las herramientas necesarias para poder utilizar el generador de código

- Es necesario que instalemos la herramienta aspnet-codegenerator, para ello abre tu terminal y ejecuta el siguiente comando:

```
dotnet tool install -g dotnet-aspnet-codegenerator
```

```
TERMINAL  PROBLEMS  3  OUTPUT  DEBUG CONSOLE

PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> dotnet tool install -g dotnet-aspnet-codegenerator
Puede invocar la herramienta con el comando siguiente: dotnet-aspnet-codegenerator
La herramienta "dotnet-aspnet-codegenerator" (versión '5.0.2') se instaló correctamente.
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui>
```

- Ejecuta el comando `aspnet-codegenerator -h`, para comprobar que la herramienta se instaló de manera correcta y conocer las opciones que nos brinda el comando

```
dotnet aspnet-codegenerator -h
```

```
TERMINAL  PROBLEMS  3  OUTPUT  DEBUG CONSOLE

PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> dotnet aspnet-codegenerator -h

Usage: aspnet-codegenerator [arguments] [options]

Arguments:
  generator  Name of the generator. Check available generators below.

Options:
  -p|--project          Path to .csproj file in the project.
  -n|--nuget-package-dir
  -c|--configuration    Configuration for the project (Possible values: Debug/ Release)
  -tfm|--target-framework Target Framework to use. (Short folder name of the tfm. eg. net46)
  -b|--build-base-path
  --no-build


Available generators:
  area      : Generates an MVC Area.
  controller: Generates a controller.
  identity  : Generates identity files.
  razorpage : Generates RazorPage(s).
  view      : Generates a view.

RunTime 00:00:02.39
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui>
```

3. Crea un nuevo controlador utilizando las herramientas de scaffold.

- Abre la terminal e Instala el paquete nuget `Microsoft.VisualStudio.Web.CodeGeneration.Design`.

```
dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design
```



```
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design
Determinando los proyectos que se van a restaurar...
Writing C:\Users\jchuc\AppData\Local\Temp\tmp92D4.tmp
info : Agregando PackageReference para el paquete "Microsoft.VisualStudio.Web.CodeGeneration.Design" al proyecto "C:\Home\3A\Curso.MVC.Core\webacademy.ui\webacademy.ui.csproj".
info : GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.visualstudio.web.codegeneration.design/index.json
info : OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.visualstudio.web.codegeneration.design/index.json 331 ms
info : Restaurando paquetes para C:\Home\3A\Curso.MVC.Core\webacademy.ui\webacademy.ui.csproj...
info : El paquete "Microsoft.VisualStudio.Web.CodeGeneration.Design" es compatible con todos los marcos de trabajo especificados del proyecto "C:\Home\3A\Curso.MVC.Core\webacademy.ui\webacademy.ui.csproj".
info : Se actualizó PackageReference para la versión "5.0.2" del paquete "Microsoft.VisualStudio.Web.CodeGeneration.Design" en el archivo "C:\Home\3A\Curso.MVC.Core\webacademy.ui\webacademy.ui.csproj".
info : Ejecutando restauración...
info : Escribiendo el archivo de recursos en el disco. Ruta de acceso: C:\Home\3A\Curso.MVC.Core\webacademy.ui\obj\project.assets.json
log : Se ha restaurado C:\Home\3A\Curso.MVC.Core\webacademy.ui\webacademy.ui.csproj (en 16.03 sec).
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui>
```

- Ejecuta el siguiente comando para consultar las opciones de creación de un controlador:

```
dotnet aspnet-codegenerator controller -h
```



```
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> dotnet aspnet-codegenerator controller -h

Usage: aspnet-codegenerator [arguments] [options]

Arguments:
  generator  Name of the generator. Check available generators below.

Options:
  -p|--project          Path to .csproj file in the project.
  -n|--nuget-package-dir
  -c|--configuration    Configuration for the project (Possible values: Debug/ Release)
  -tfm|--target-framework Target Framework to use. (Short folder name of the tfm. eg. net46)
  -b|--build-base-path
  --no-build

Selected Code Generator: controller

Generator Options:
  --controllerName|-name          : Name of the controller
  --useAsyncActions|-async        : Switch to indicate whether to generate async controller actions
  --noViews|-nv                  : Switch to indicate whether to generate CRUD views
  --restWithNoViews|-api         : Specify this switch to generate a Controller with REST style API, noViews is assumed and any view related options are ignored
  --readWriteActions|-actions     : Specify this switch to generate Controller with read/write actions when a Model class is not used
  --model|-m                    : Model class to use
  --dbContext|-dc               : DbContext class to use
  --referenceScriptLibraries|-scripts : Switch to specify whether to reference script libraries in the generated views
  --layout|-l                   : Custom Layout page to use
  --useDefaultLayout|-udl        : Switch to specify that default layout should be used for the views
  --force|-f                     : Use this option to overwrite existing files
  --relativeFolderPath|-outDir    : Specify the relative output folder path from project where the file needs to be generated, if not specified, file will be generated in the project folder
  --controllerNamespace|-namespace : Specify the name of the namespace to use for the generated controller
  --useSqlite|-sqlite            : Flag to specify if DbContext should use SQLite instead of SQL Server.

RunTime 00:00:02.26
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui>
```

- Ahora que conocemos las opciones de creación de controladores puedes crear un controlador basado en las entidades de tu datacontext para ello ejecuta el siguiente comando:

```
dotnet aspnet-codegenerator controller -name <<NameController>> -m <<Model>> -dc <<DbContext>> -async -nv -actions -outDir Controllers
```

```
TERMINAL  PROBLEMS 3  OUTPUT  DEBUG CONSOLE  powershell + v ^ x

PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> dotnet aspnet-codegenerator controller -name CountryController -m Country -dc ApplicationDbContext -async -nv -actions -outDir Controllers
Building project ...
Finding the generator 'controller'...
Running the generator 'controller'...
Attempting to compile the application in memory.
Attempting to figure out the EntityFramework metadata for the model and DbContext: 'Country'
Added Controller : '\Controllers\CountryController.cs'.
RunTime 00:00:12.44
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> █
```

- Una vez que el comando se ejecuta, puedes ver una nueva clase en la carpeta **Controllers** con el nombre **CountryController.cs** y con un resultado similar al siguiente:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Rendering;
using Microsoft.EntityFrameworkCore;
using webacademy.Data;
using webacademy.Entities;

namespace webacademy.ui.Controllers
{
    public class CountryController : Controller
    {
        private readonly ApplicationDbContext _context;

        public CountryController(ApplicationDbContext context)
        {
            _context = context;
        }

        // GET: Country
        public async Task<IActionResult> Index()
        {
            return View(await _context.Countries.ToListAsync());
        }

        // GET: Country/Details/5
        public async Task<IActionResult> Details(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var country = await _context.Countries
                .FirstOrDefaultAsync(m => m.Id == id);
            if (country == null)
            {
                return NotFound();
            }

            return View(country);
        }
    }
}
```

```

// GET: Country/Create
public IActionResult Create()
{
    return View();
}

// POST: Country/Create
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create([Bind("Id,Name,Code,IsDeleted,CreateDate,ModifiedDate")] Country country)
{
    if (ModelState.IsValid)
    {
        _context.Add(country);
        await _context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    return View(country);
}

// GET: Country/Edit/5
public async Task<IActionResult> Edit(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var country = await _context.Countries.FindAsync(id);
    if (country == null)
    {
        return NotFound();
    }
    return View(country);
}

// POST: Country/Edit/5
// To protect from overposting attacks, enable the specific properties you want to bind to.
// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Edit(int id, [Bind("Id,Name,Code,IsDeleted,CreateDate,ModifiedDate")] Country country)
{
    if (id != country.Id)
    {
        return NotFound();
    }

    if (ModelState.IsValid)
    {
        try
        {
            _context.Update(country);
            await _context.SaveChangesAsync();
        }
    }
}

```

```

        catch (DbUpdateConcurrencyException)
        {
            if (!CountryExists(country.Id))
            {
                return NotFound();
            }
            else
            {
                throw;
            }
        }
        return RedirectToAction(nameof(Index));
    }
    return View(country);
}

// GET: Country/Delete/5
public async Task<IActionResult> Delete(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var country = await _context.Countries
        .FirstOrDefaultAsync(m => m.Id == id);
    if (country == null)
    {
        return NotFound();
    }

    return View(country);
}

// POST: Country/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public async Task<IActionResult> DeleteConfirmed(int id)
{
    var country = await _context.Countries.FindAsync(id);
    _context.Countries.Remove(country);
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}

private bool CountryExists(int id)
{
    return _context.Countries.Any(e => e.Id == id);
}
}

```

4. Crea una nueva vista con las herramientas de scaffold.
 - Ejecuta el siguiente comando para conocer las opciones de creación de las vistas:

```
dotnet aspnet-codegenerator view -h
```

```
TERMINAL  PROBLEMS 3  OUTPUT  DEBUG CONSOLE
powershell + ^ x

actions -outDir Controllers
Building project ...
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> dotnet aspnet-codegenerator view -h

Usage: aspnet-codegenerator [arguments] [options]

Arguments:
  generator  Name of the generator. Check available generators below.

Options:
  -p|--project          Path to .csproj file in the project.
  -n|--nuget-package-dir
  -c|--configuration    Configuration for the project (Possible values: Debug/ Release)
  -tfm|--target-framework Target Framework to use. (Short folder name of the tfm. eg. net46)
  -b|--build-base-path
  --no-build

Selected Code Generator: view

Generator Arguments:
  viewName      : Name of the view
  templateName  : The view template to use, supported view templates: 'Empty|Create|Edit|Delete|Details|List'

Generator Options:
  --model|--m          : Model class to use
  --dataContext|--dc    : DbContext class to use
  --referenceScriptLibraries|--scripts : Switch to specify whether to reference script libraries in the generated views
  --layout|--l          : Custom Layout page to use
  --useDefaultLayout|--udl : Switch to specify that default layout should be used for the views
  --force|--f           : Use this option to overwrite existing files
  --relativeFolderPath|--outDir : Specify the relative output folder path from project where the file needs to be generated, if not specified, file will be generated in the project folder
  --controllerNamespace|--namespace : Specify the name of the namespace to use for the generated controller
  --partialView|--partial : Generate a partial view, other layout options (-l and -udl) are ignored if this is specified
  --useSqlite|--sqlite   : Flag to specify if DbContext should use SQLite instead of SQL Server.

RunTime 00:00:02.25
```

- Ejecuta el siguiente comando para crear una vista de consulta

```
dotnet aspnet-codegenerator view <<Name>> List -m <<Model>> -dc <<DataContext>> -scripts -l Views/<<Layout.cshtml>> -outDir Views/<<Model>>
```

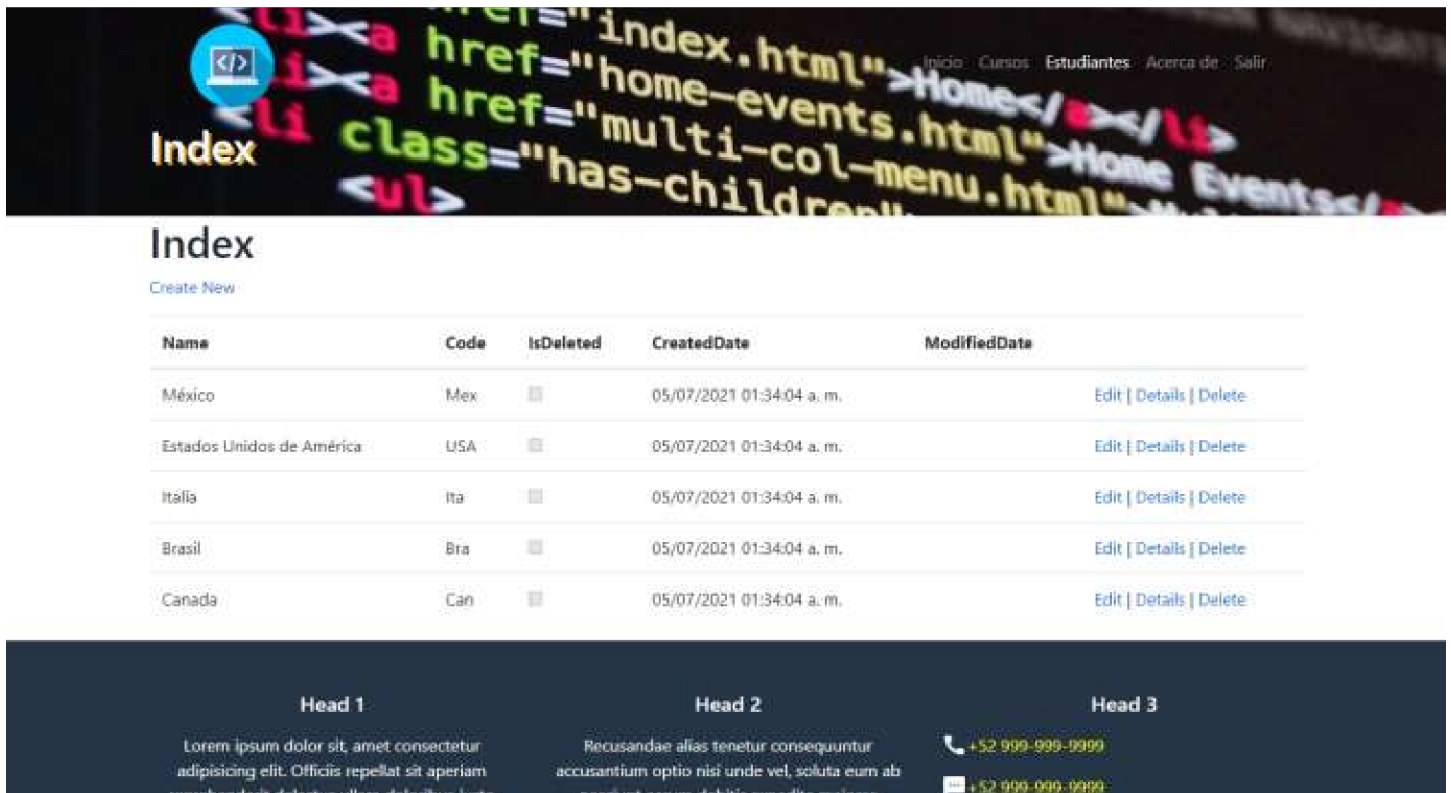
```
TERMINAL  PROBLEMS 3  OUTPUT  DEBUG CONSOLE
powershell + ^ x

PS C:\Home\3A\Curso.MVC.Core\webacademy.ui> dotnet aspnet-codegenerator view Index List -m Country -dc ApplicationDbContext -scripts -l Views/Template/_adminLayout.cshtml -outDir Views/Country
Building project ...
Finding the generator 'view'...
Running the generator 'view'...
Attempting to compile the application in memory.
Attempting to figure out the EntityFramework metadata for the model and DbContext: 'Country'
Added View : \Views\Countr\Index.cshtml
RunTime 00:00:11.63
PS C:\Home\3A\Curso.MVC.Core\webacademy.ui>
```

- Abre el archivo `Index.cshtml` que se encuentra en la carpeta `Views > Country`, en la sección del layout agregar el símbolo “~” para indicar la ruta relativa en la que se encuentra el Layout.

5. Prueba tu aplicación

- Después de ejecutar tu aplicación ingresa a la siguiente dirección: <https://localhost:5001/Country>; deberías ver un resultado similar al siguiente:



The screenshot displays a web application interface. At the top, there is a navigation bar with links: Inicio, Cursos, Estudiantes, Acerca de, and Salir. Below the navigation bar, the word "Index" is prominently displayed. Underneath "Index", there is a "Create New" link. The main content area features a table with the following columns: Name, Code, IsDeleted, CreatedDate, and ModifiedDate. The table contains five rows of data representing different countries. At the bottom of the page, there is a dark blue footer with three columns labeled Head 1, Head 2, and Head 3. Head 1 contains placeholder text, Head 2 contains placeholder text, and Head 3 contains a phone icon and a WhatsApp icon, both with the number +52 999-999-9999.

Name	Code	IsDeleted	CreatedDate	ModifiedDate
México	Mex	<input type="checkbox"/>	05/07/2021 01:34:04 a. m.	Edit Details Delete
Estados Unidos de América	USA	<input type="checkbox"/>	05/07/2021 01:34:04 a. m.	Edit Details Delete
Italia	Ita	<input type="checkbox"/>	05/07/2021 01:34:04 a. m.	Edit Details Delete
Brasil	Bra	<input type="checkbox"/>	05/07/2021 01:34:04 a. m.	Edit Details Delete
Canada	Can	<input type="checkbox"/>	05/07/2021 01:34:04 a. m.	Edit Details Delete

Head 1

Lorem ipsum dolor sit, amet consectetur adipiscing elit. Officiis repellat sit aperiam reprehenderit delectus ullam delectibus justo

Head 2

Recusandae alias tenetur consequuntur accusantium optio nisi unde vel, soluta eum ab persequuntur, debitis expedita exasperat

Head 3

+52 999-999-9999
+52 999-999-9999

Bibliografía

- <https://docs.microsoft.com/en-us/ef/core/get-started/overview/first-app?tabs=netcore-cli>
- <https://docs.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-5.0#how-to-download-a-sample>
- <https://docs.microsoft.com/en-us/aspnet/core/fundamentals/tools/dotnet-aspnet-codegenerator?view=aspnetcore-5.0>
- <https://www.nuget.org/packages/Microsoft.VisualStudio.Web.CodeGeneration.Design/6.0.0-preview.5.21321.1>