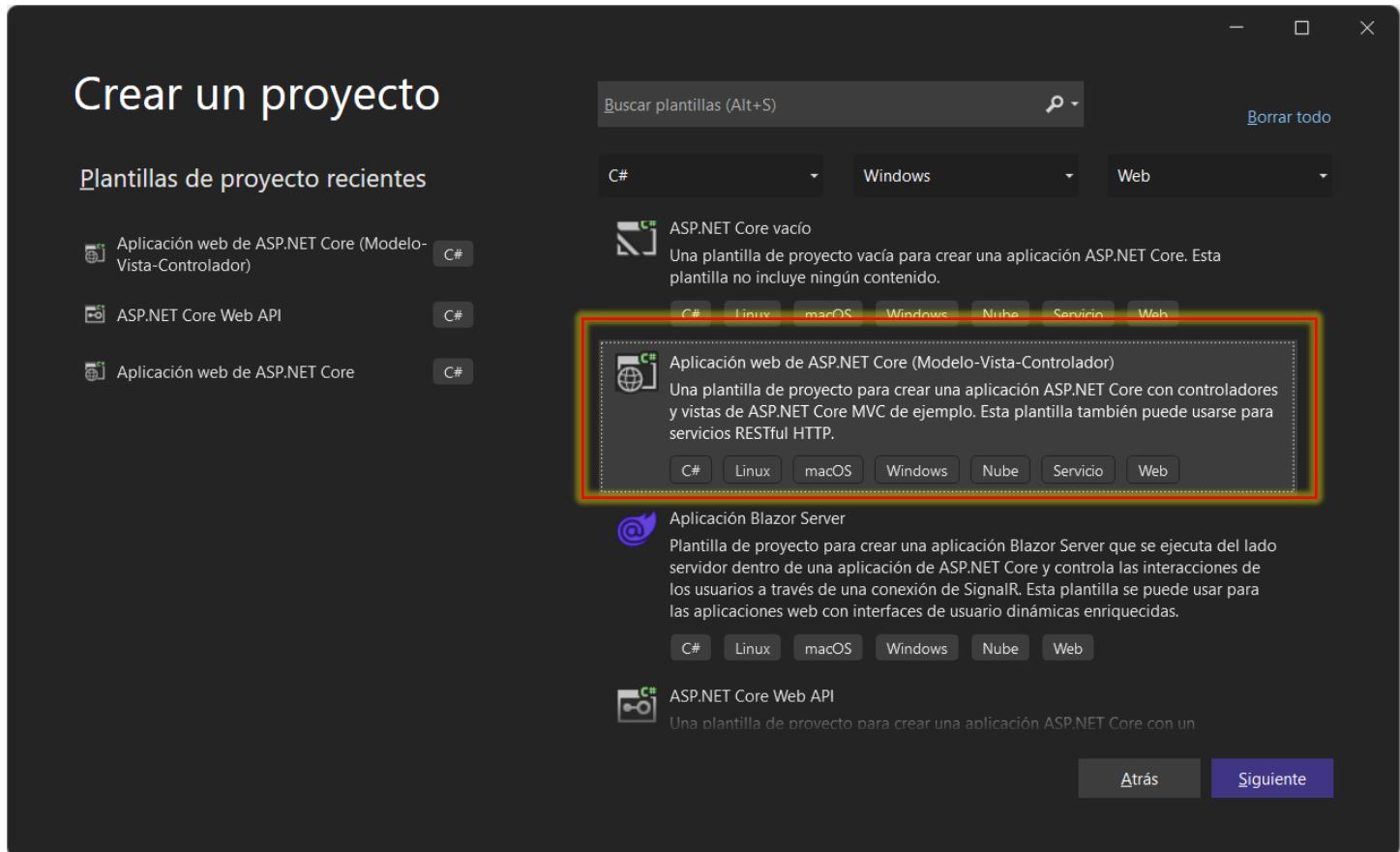


ASP. NET con MVC Entity Framework

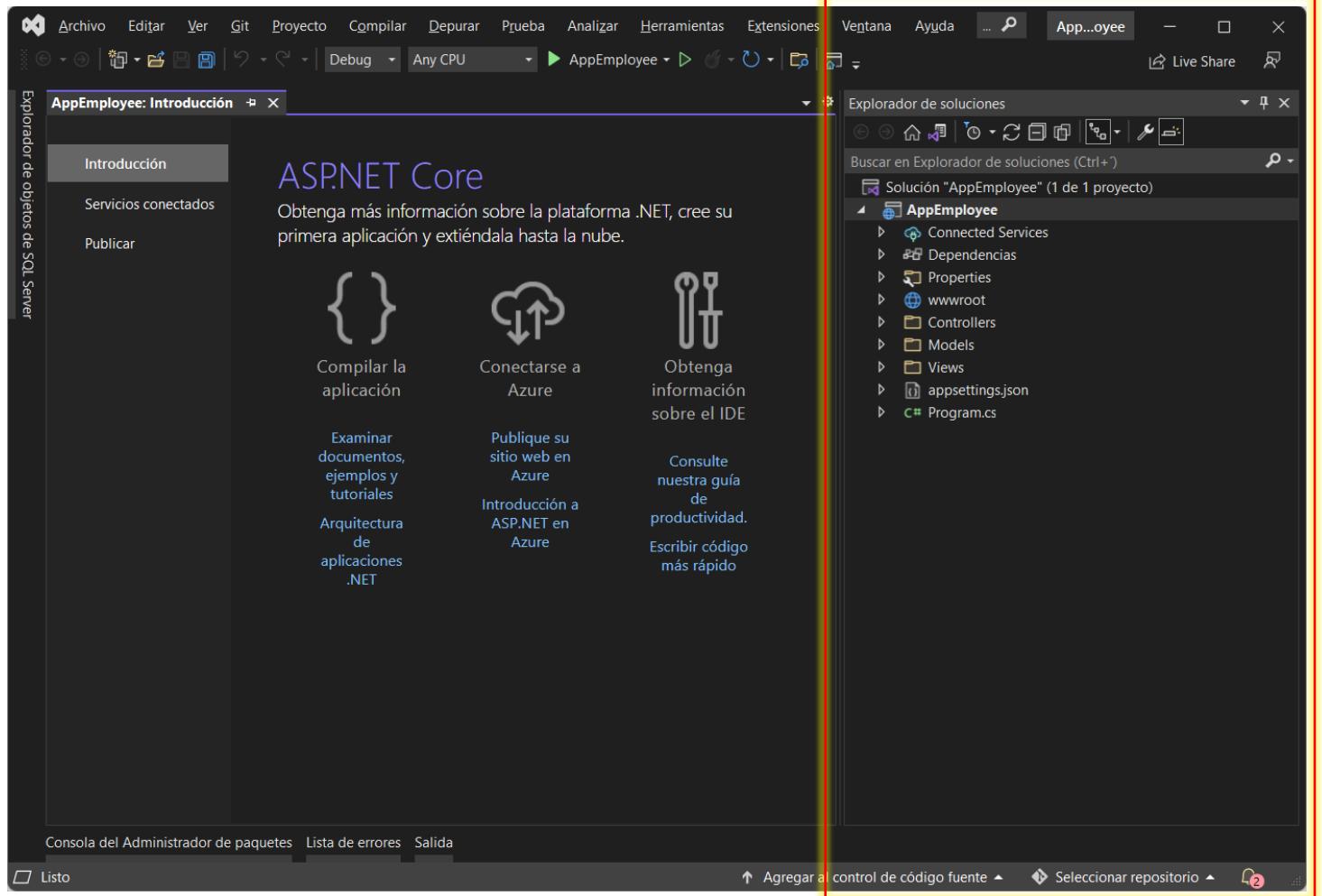
0. Este documento es creado solamente con fines académicos, el objetivo es que sea una ayuda adicional para la realización de las prácticas en clases, no sustituye la explicación, clase magistral y dialogo didáctico de la profesora para una mejor comprensión.

1. Crear un nuevo proyecto **Aplicación web ASP.NET Core (MVC)** con el nombre AppEmpleados.

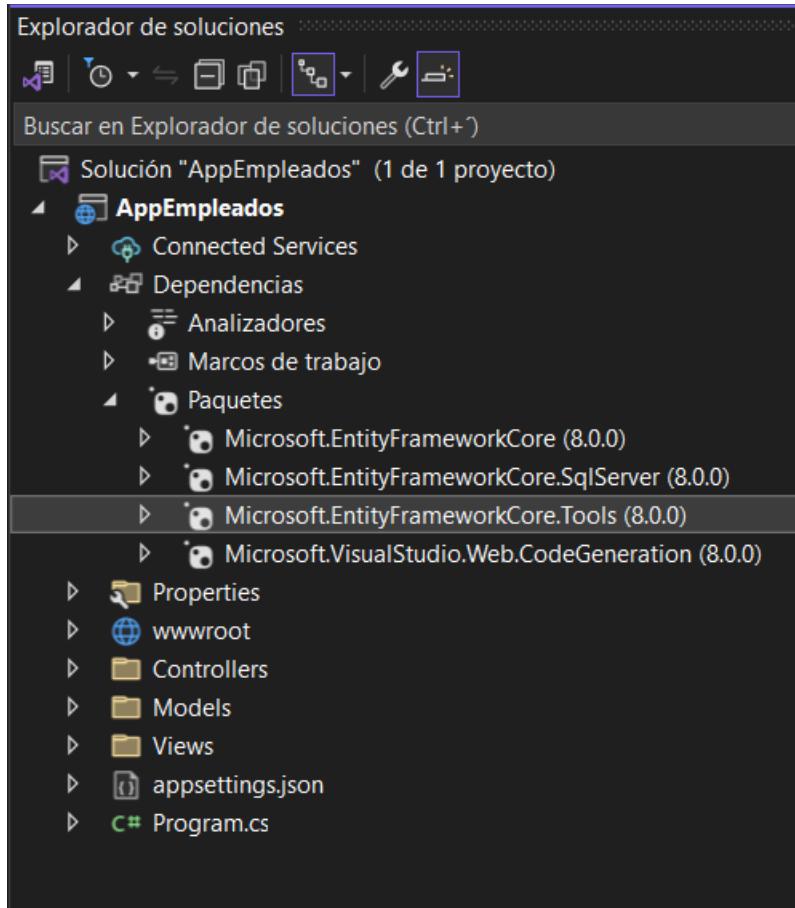


En los siguientes pasos dejar la configuración por defecto.

2. Proyecto creado, verificar que este visible el explorador de soluciones



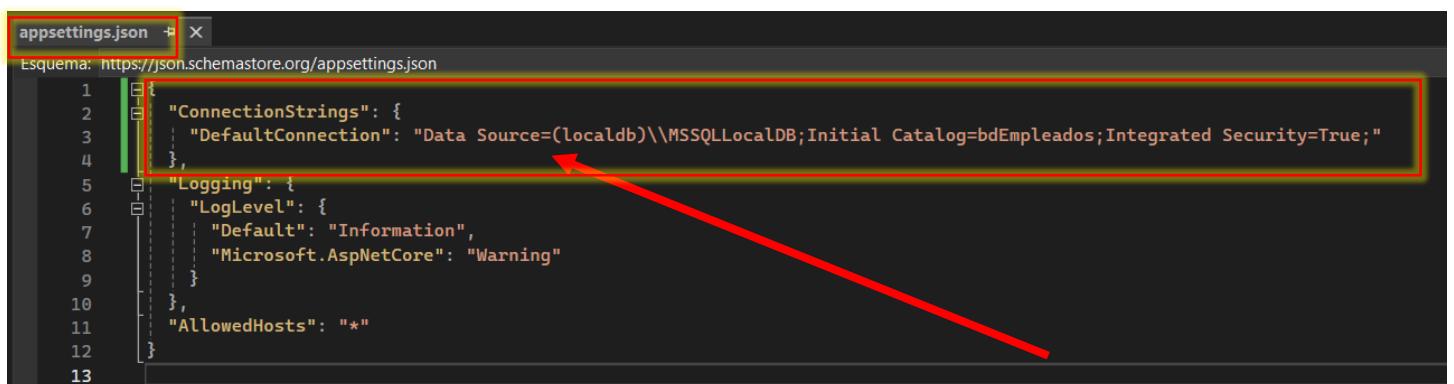
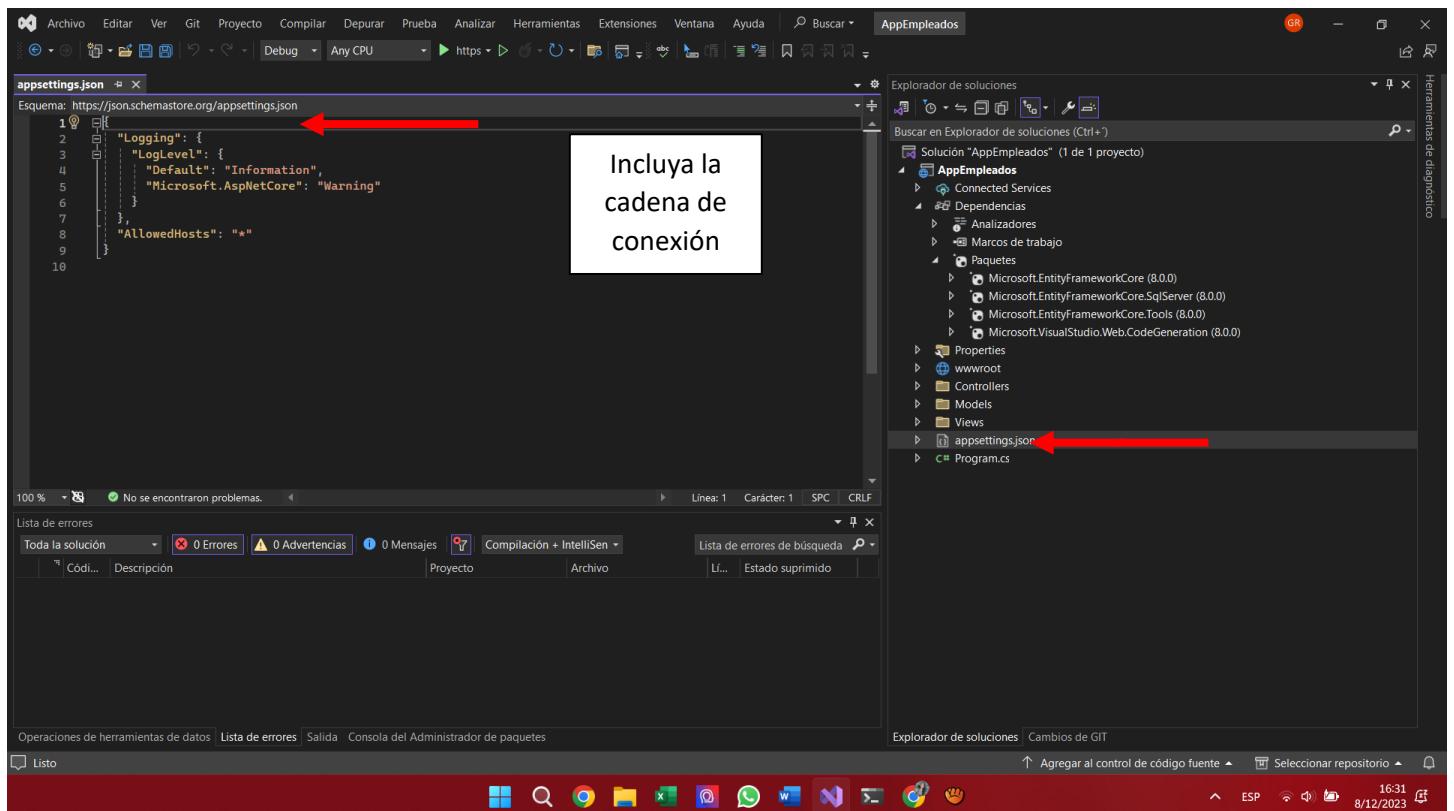
3. Instalar los siguientes paquetes, verificar que la versión se la misma en todos los paquetes.



Paquetes

Microsoft.EntityFrameworkCore
Microsoft.EntityFrameworkCore.SqlServer
Microsoft.EntityFrameworkCore.Tools
Microsoft.VisualStudio.Web.CodeGeneration

4. Agregar la conexión a la base de datos a utilizar, ir al archivo appsetting.json e incluir el siguiente fragmento de código
5. Modificar el nombre de la base de datos en el archivo **appsettings.json**. Ponerle el nombre **bdEmpleados**.



```

{
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=bdEmpleados;Integrated Security=True;"
  }

  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*"
}

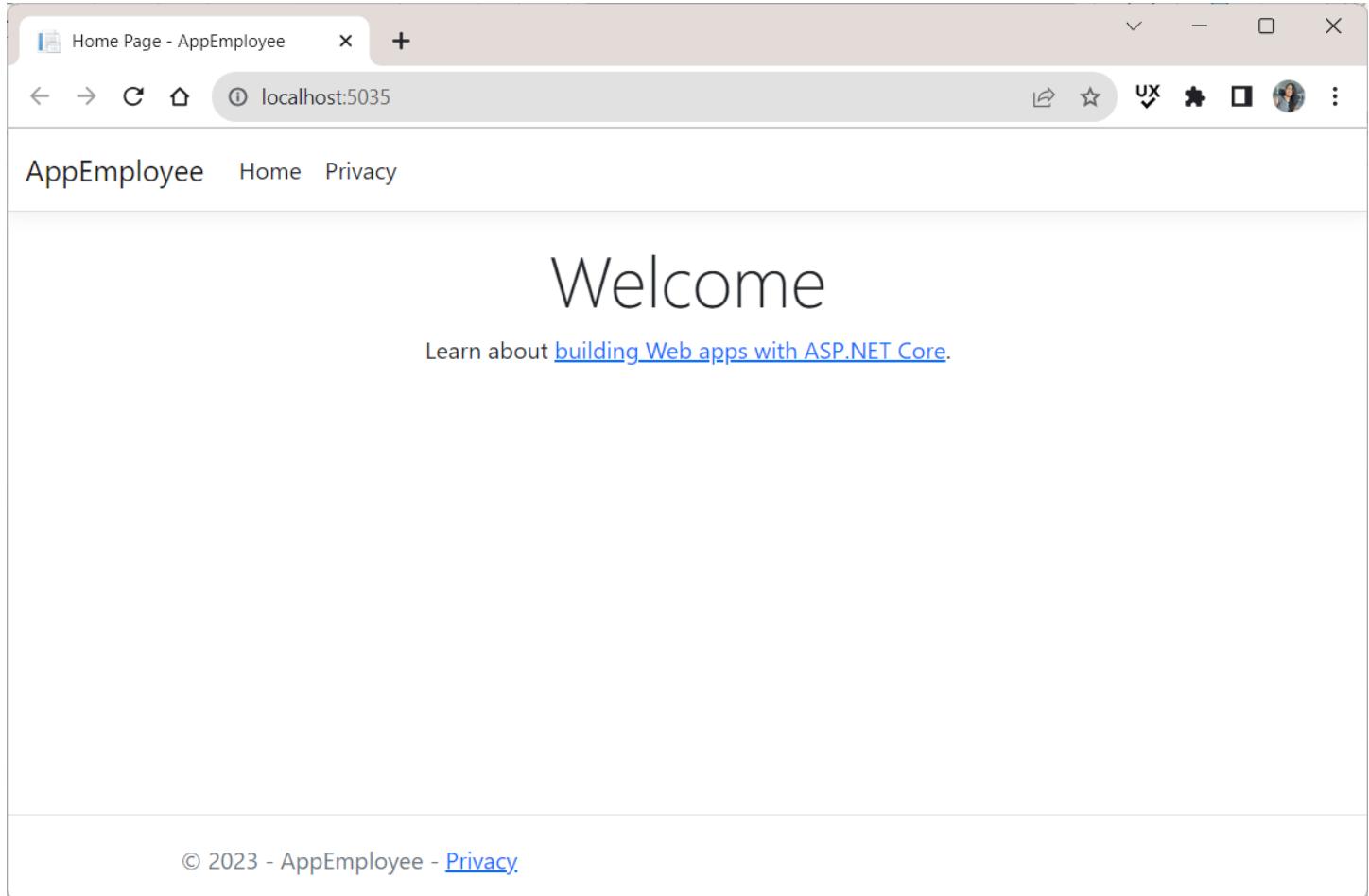
```

Anatomía de una cadena de conexión:

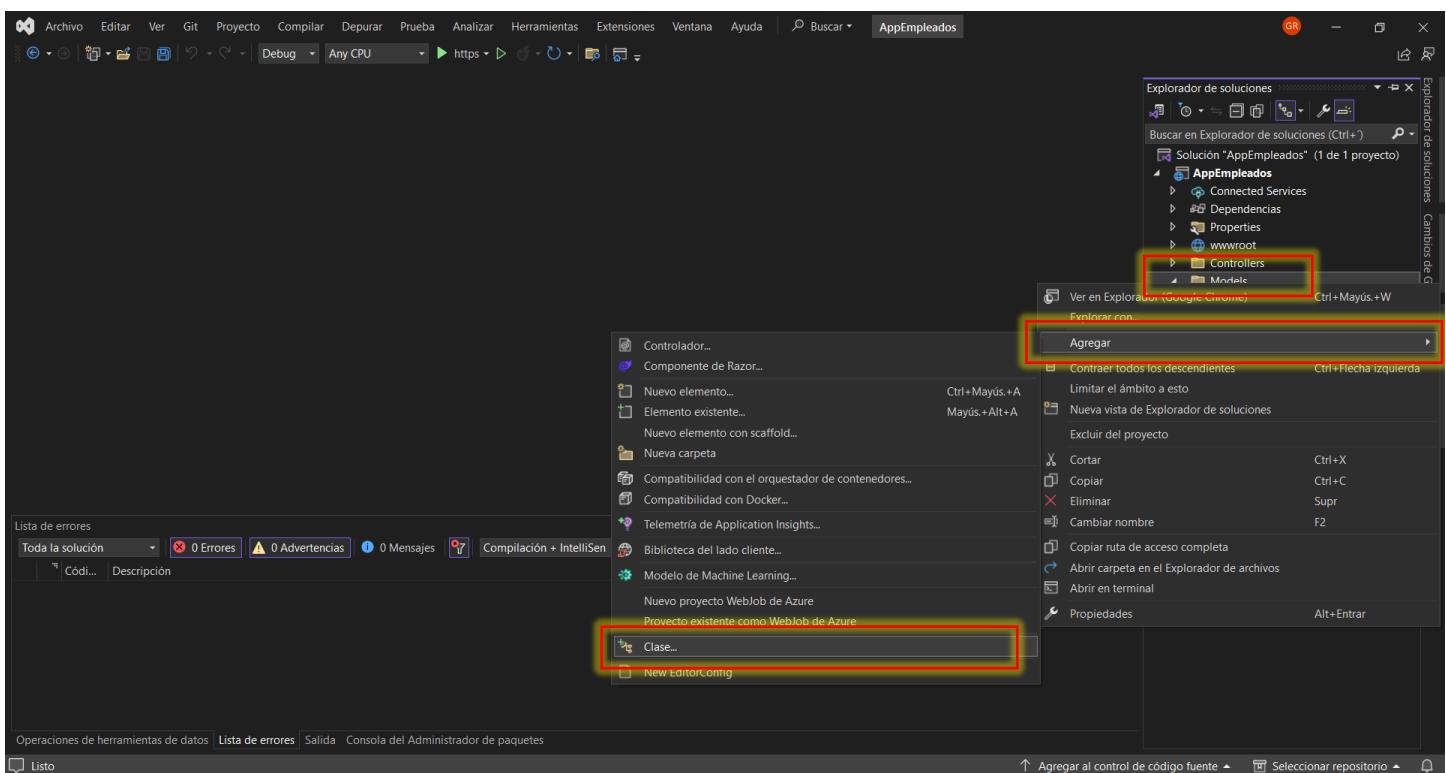
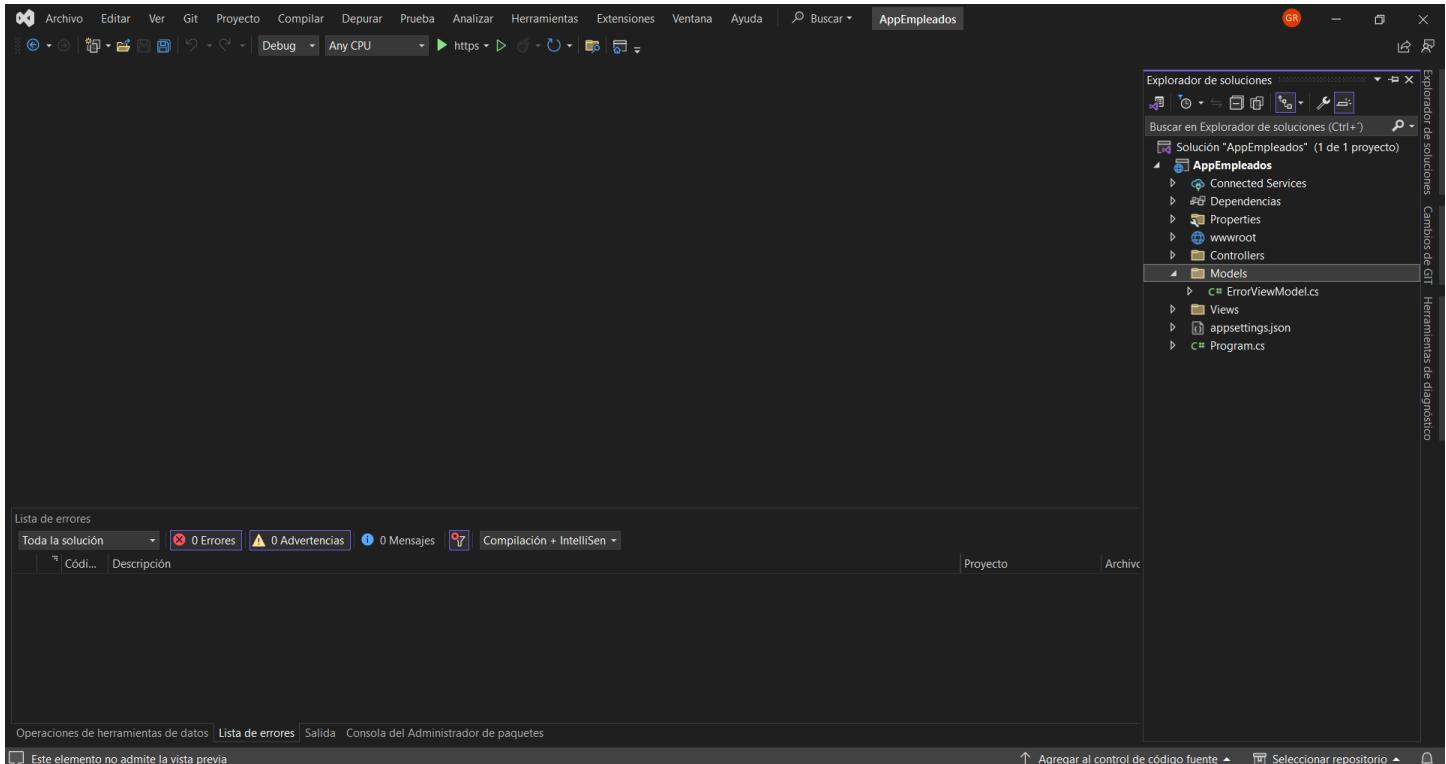
"Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=bdEmpleados;Integrated Security=True;"

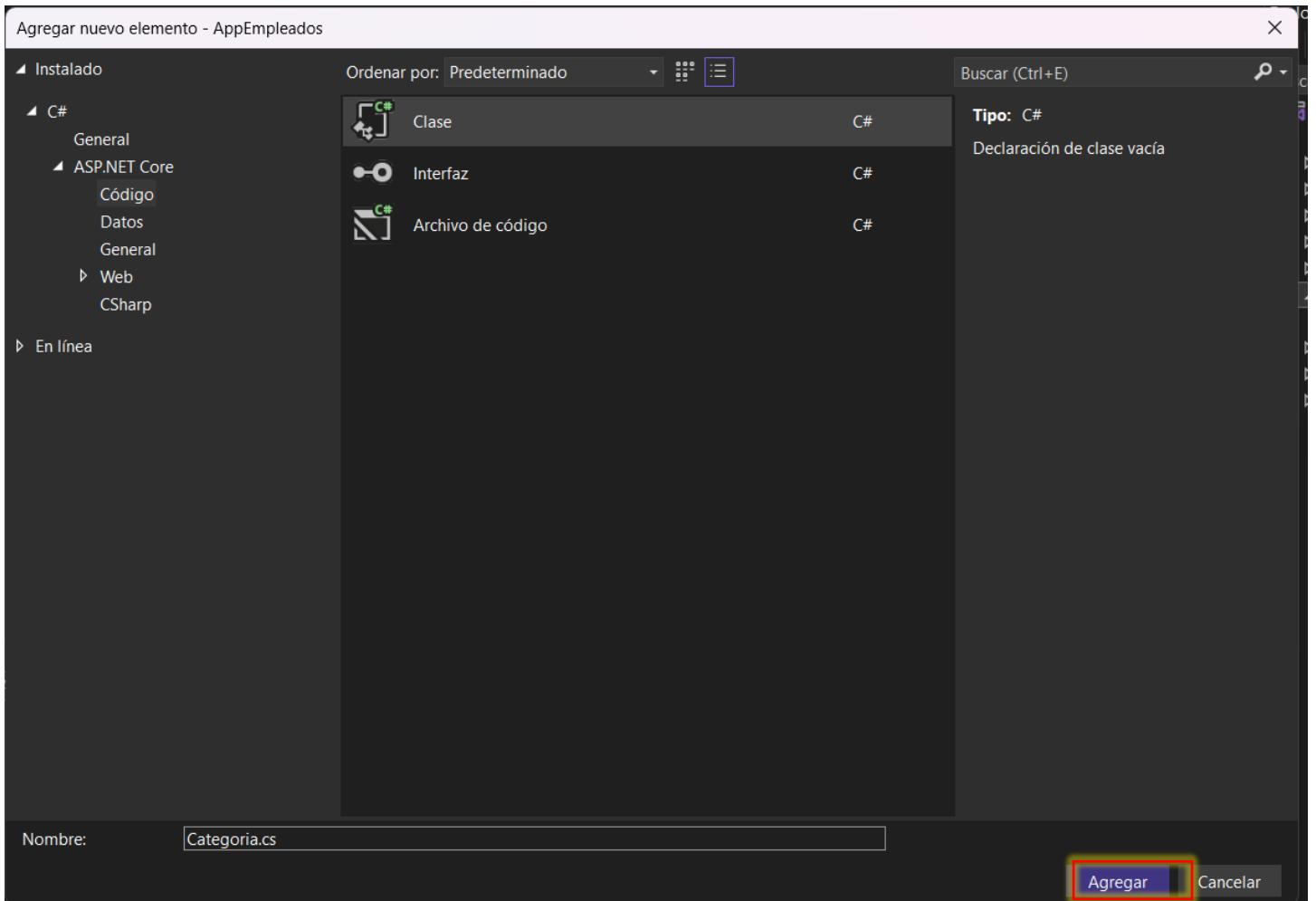
Nombre del servidor Nombre de la bd Otras configuraciones

6. Correr la aplicación, explorar las diferentes vistas.



7. Incluir los modelos **Categoría** y **Empleado** en la carpeta Models





Categoría.cs

AppEmpleados

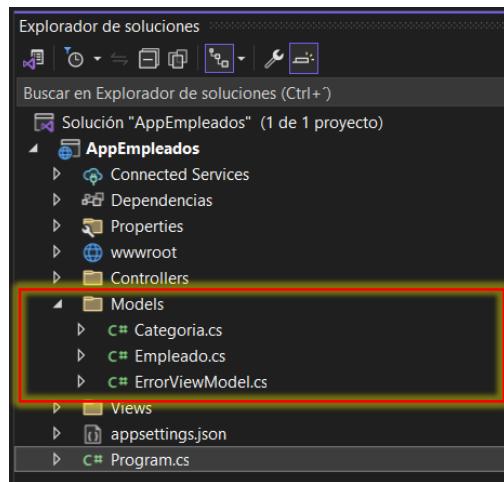
AppEmpleados.Models.Categ

```
namespace AppEmpleados.Models
{
    public class Categoría
    {
        public int CategoríaId { get; set; }
        public string Nombre { get; set; }
        public List<Empleado> Empleados { get; set; }
    }
}
```

The screenshot shows the 'Categoria.cs' code editor in Visual Studio. The code defines a 'Categoría' class with properties 'CategoríaId', 'Nombre', and a list of 'Empleados'. A red box highlights the 'Empleados' property.

```
public class Categoría
{
    public int CategoríaId { get; set; }
    public string Descripción { get; set; }
    public virtual List<Empleado> Empleados { get; set; }
}
```

Repita el proceso y cree la clase Empleado

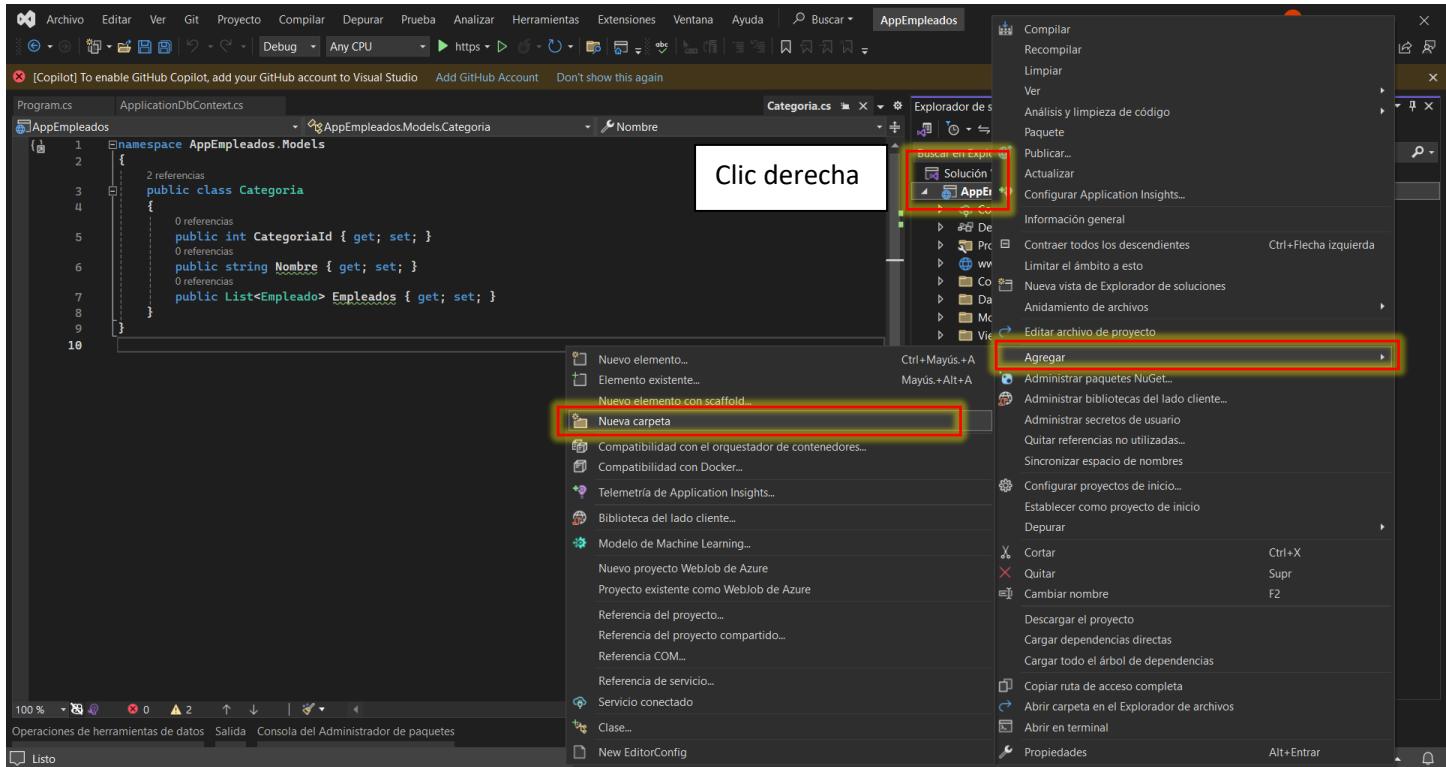


Empleado.cs

```
{ 1  using System.ComponentModel.DataAnnotations;
 2
 3  namespace AppEmpleados.Models
 4  {
 5      public class Empleado
 6      {
 7          public int EmpleadoId { get; set; }
 8          public string Nombre { get; set; }
 9          public string Apellidos { get; set; }
10          //Anotacion para que solo muestre la fecha sin la hora
11          [DataType(DataType.Date)]
12          public DateTime FechaNacimiento { get; set; }
13          //Clave foranea
14          public int CategoriaId { get; set; }
15          //Propiedad de navegacion
16          public Categoria Categoria { get; set; }
17      }
18  }
```

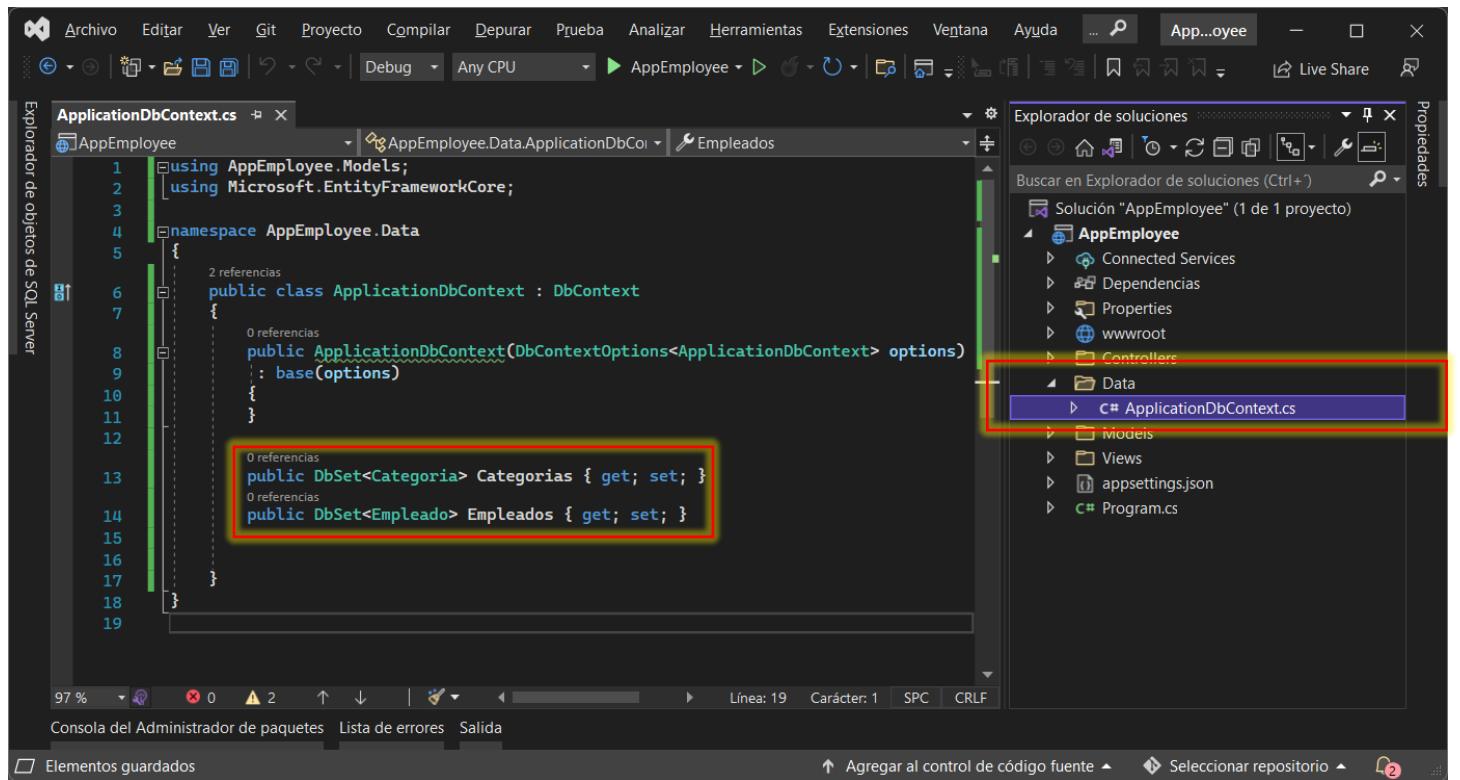
```
public class Empleado
{
    public int EmpleadoId { get; set; }
    public string Nombre { get; set; }
    public string Apellidos { get; set; }
    //Anotacion para que solo muestre la fecha sin la hora
    [DataType(DataType.Date)]
    public DateTime FechaNacimiento { get; set; }
    //Clave foranea
    public int CategoriaId { get; set; }
    //Propiedad de navegacion
    public Categoria Categoria { get; set; }
}
```

8. Crear una carpeta con el nombre Data



9. Dentro de la carpeta, crear una Clase con el nombre ApplicationDbContext

10. Incluir dentro de la clase el fragmento de código que permite la creación de la tabla categorías y Empleados en la base de datos SQL-Server, a través de los comandos add-migration y update-database.



```

public class ApplicationDbContext : DbContext
{
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
        : base(options)
    {
    }

    public DbSet<Categoria> Categorias { get; set; }
    public DbSet<Empleado> Empleados { get; set; }

}

}

```

11. Incluir en la clase **ApplicationDbContext** el fragmento de código que permite configurar cuales modelos (clases) se convertirán en tablas en la base de datos, en nuestro caso **Categoría y Empleado**,

12. Configuración de la fuente de datos en Program.cs

```

Program.cs
1  using AppEmpleados.Models;
2  using Microsoft.EntityFrameworkCore;
3
4  var builder = WebApplication.CreateBuilder(args);
5
6  //Configurar la base de datos
7  builder.Services.AddDbContext<ApplicationDbContext>(opciones =>
8      opciones.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));
9
10 // Add services to the container.
11 builder.Services.AddControllersWithViews();
12
13 var app = builder.Build();
14
15 // Configure the HTTP request pipeline.
16 if (!app.Environment.IsDevelopment())
17 {
18     app.UseExceptionHandler("/Home/Error");
19 }
20 app.UseStaticFiles();
21
22 app.UseRouting();
23
24 app.UseAuthorization();
25
26 app.MapControllerRoute(
27     name: "default",
28     pattern: "{controller=Home}/{action=Index}/{id?}");
29
30 app.Run();
31

```

Explorador de soluciones

- Solución "AppEmpleados" (1 de 1 proyecto)
 - Solution Items
 - .editorconfig
 - AppEmpleados
 - Connected Services
 - Dependencias
 - Properties
 - wwwroot
 - Controllers
 - Models
 - ApplicationContext.cs
 - Categoria.cs
 - Empleado.cs
 - ErrorViewModel.cs
 - Views
 - .editorconfig
 - appsettings.json
 - Program.cs

```

//Configuramos la conexión a sql server local
builder.Services.AddDbContext<ApplicationDbContext>(opciones =>
    opciones.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

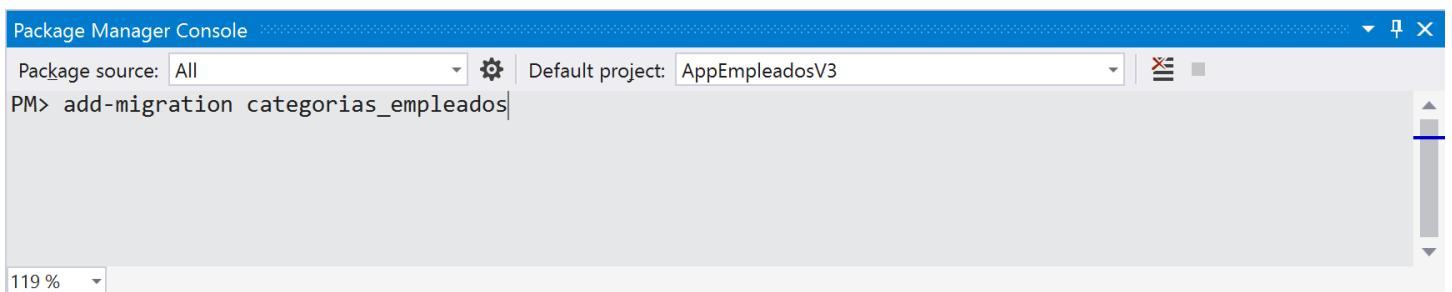
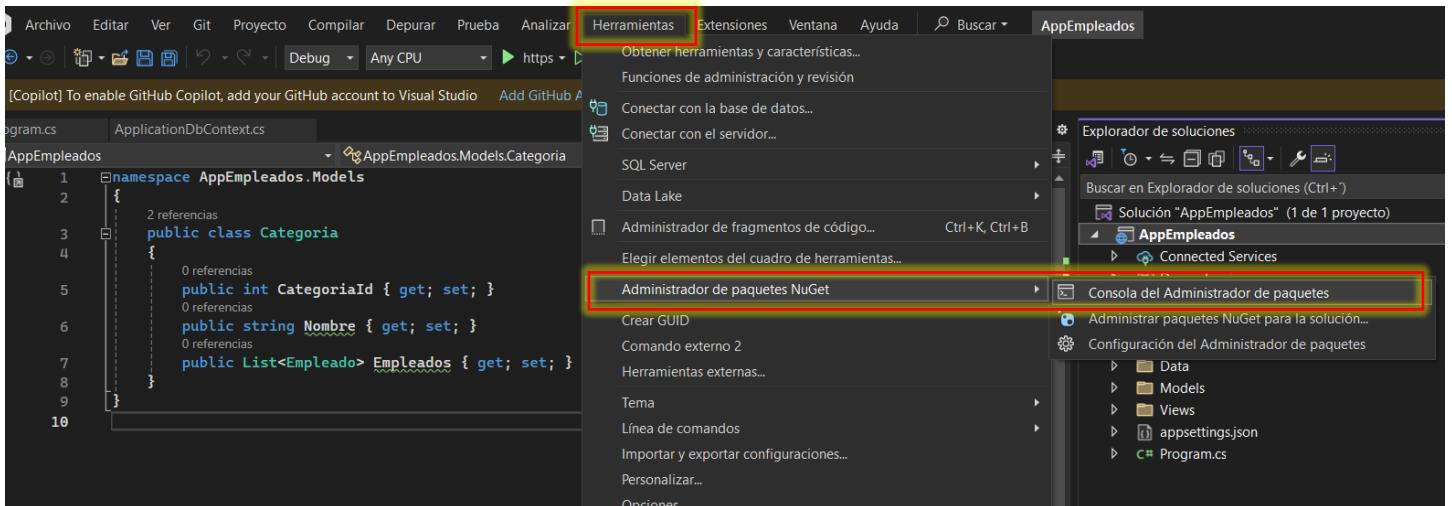
```

13. Compilar la aplicación, verificar que no hay errores

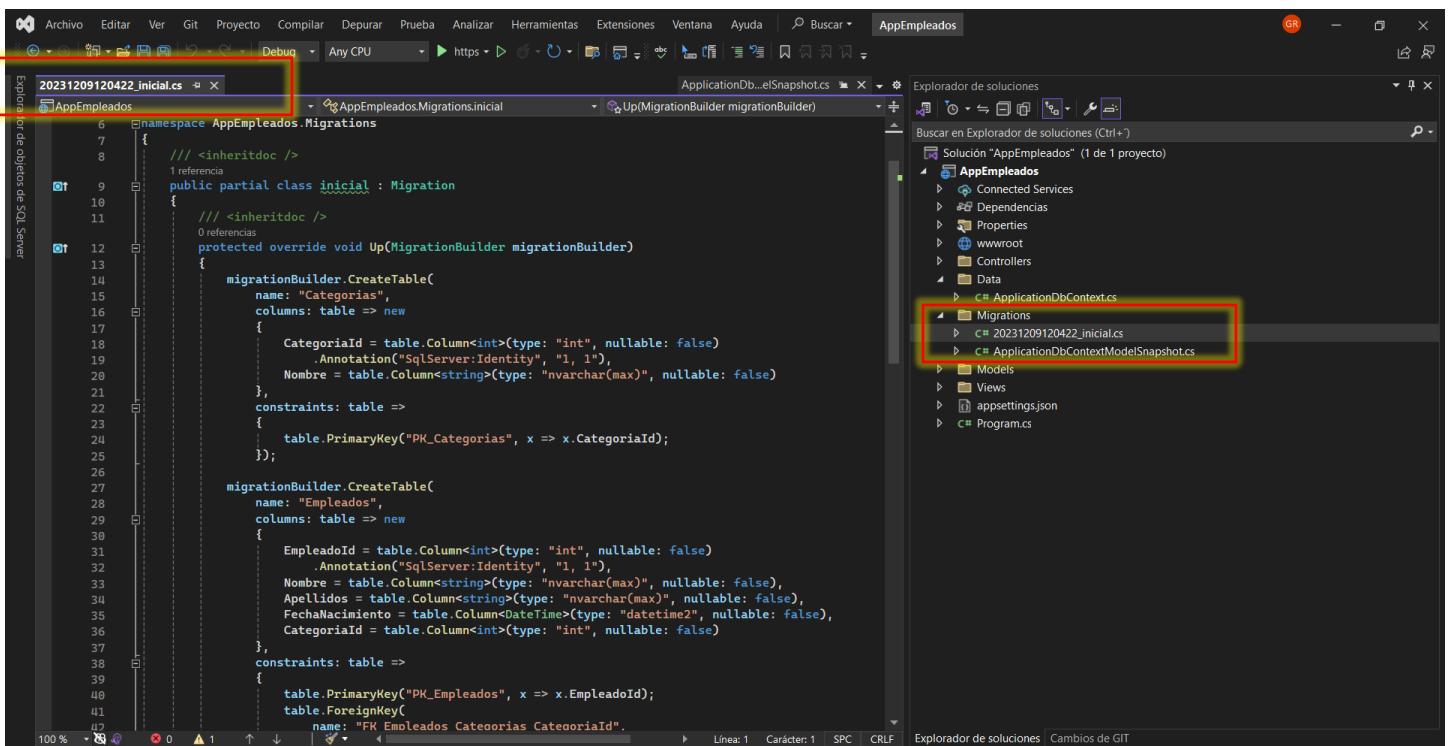
14. Ejecutar el comando Agregar la migración, este comando se ejecuta en Package Manager Console.

Debe escribir el comando **add-migration nombre**

Este comando permite crear el código en EntityFramework que será convertido en el futuro en las tablas de las base de datos, de acuerdo a la configuración de las clases.

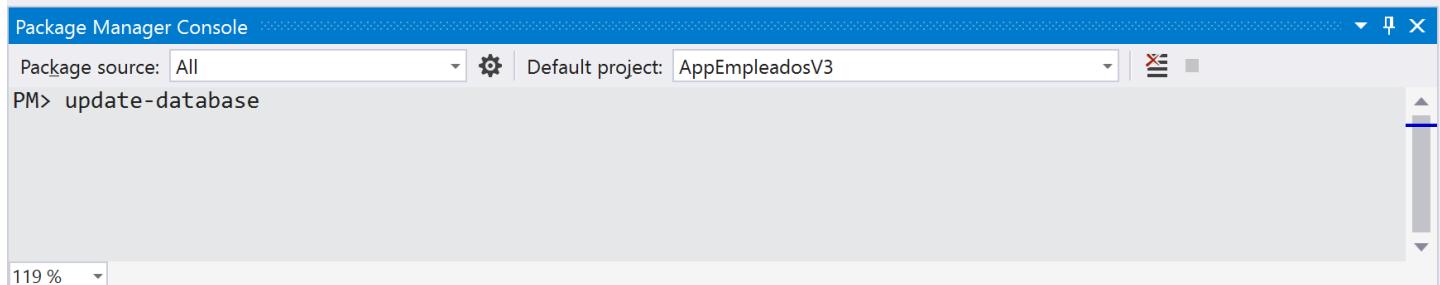


Una vez ejecutado el comando, puede visualizar en el árbol de carpetas del proyecto, una nueva carpeta con el nombre **Migrations** y dentro de ella el archivo en cs que contiene el código que permite migrar los modelos a tablas.

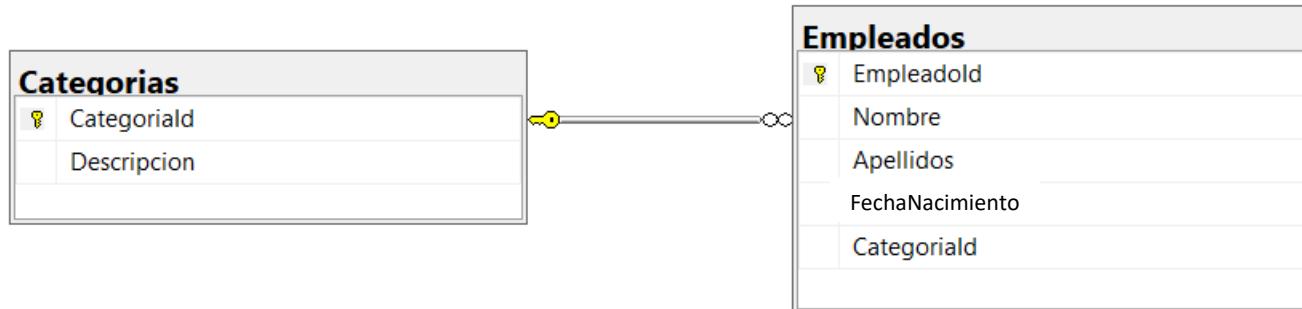


15. Actualizar la base de datos.

Ejecute el comando **update-database** en **Package Manager Console**, este comando lee el código generado en la carpeta Migrations y procede a la creación de la base de datos y tablas de acuerdo a la configuración de los modelos.



16. Revisar en la base de datos que se hayan creado las tablas.

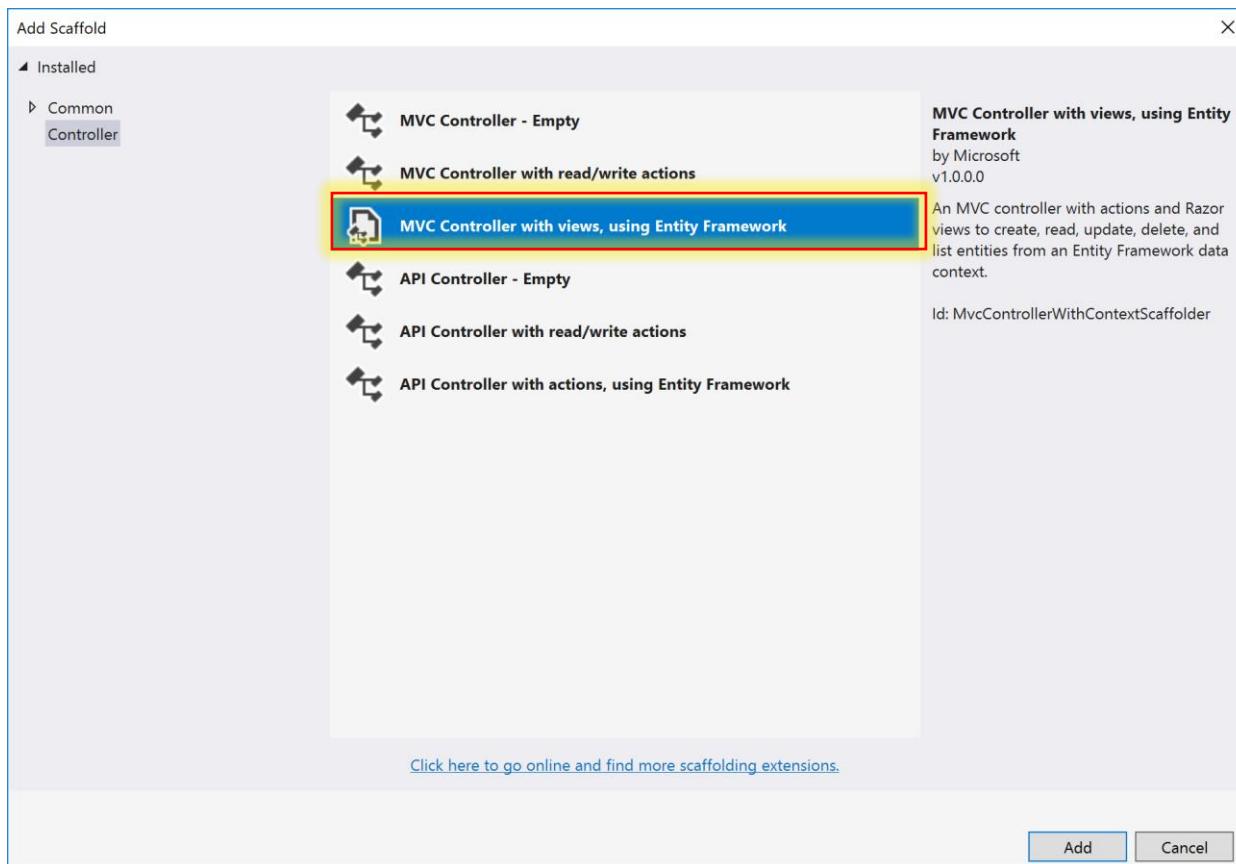
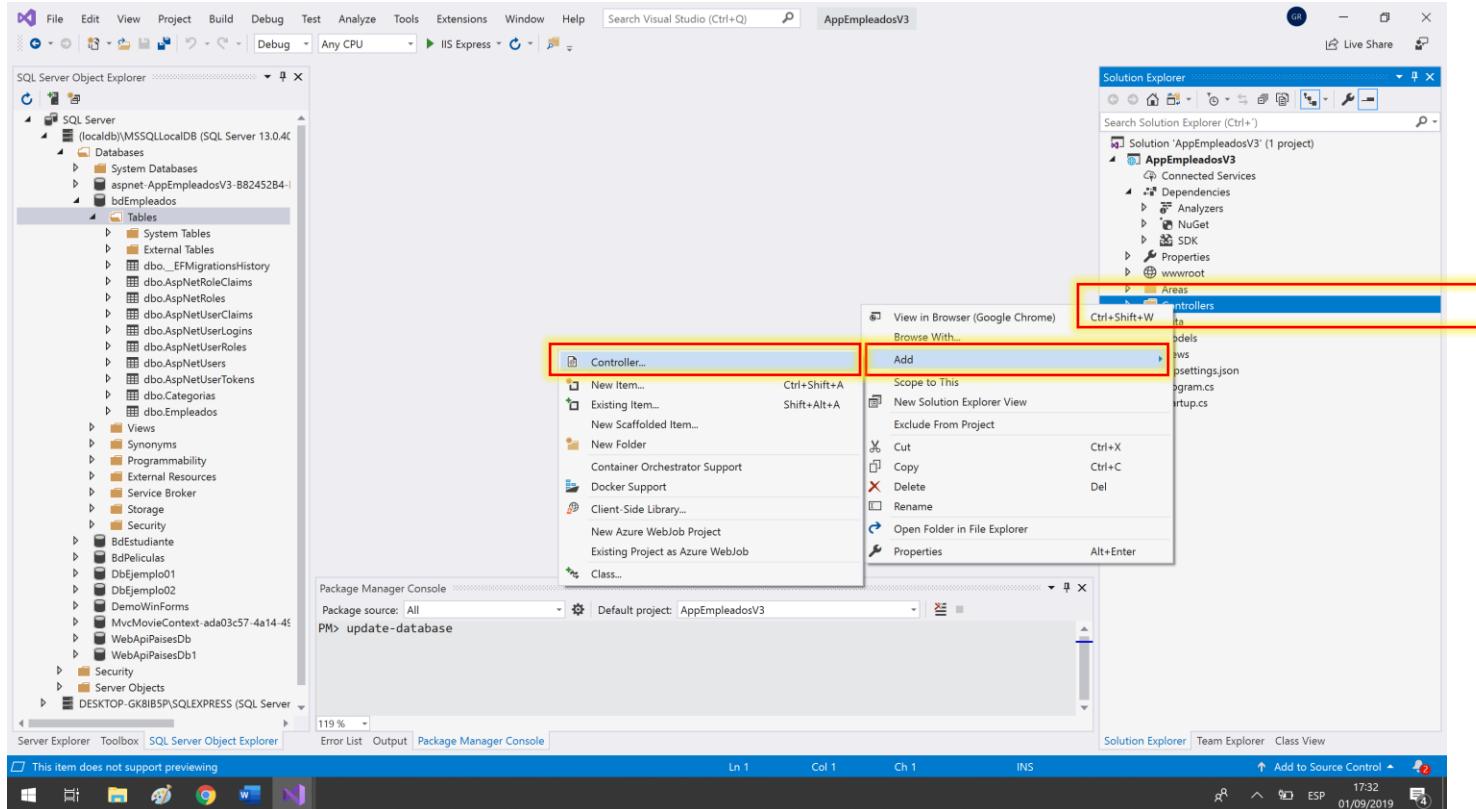


La definición de los modelos (clases) correctamente, es muy importante porque define el código generado en las migraciones para la creación de tablas, los campos y propiedades de los campos y muy importante las relaciones entre los modelos. Una muy buena definición de modelos y sus relaciones puede ahorrarnos muchos dolores de cabeza

```
public class Categoria
{
    0 referencias
    public int CategoriaId { get; set; }
    0 referencias
    public string Nombre { get; set; }
    0 referencias
    public List<Empleado> Empleados { get; set; }
}
```

```
public class Empleado
{
    0 referencias
    public int EmpleadoId { get; set; }
    0 referencias
    public string Nombre { get; set; }
    0 referencias
    public string Apellidos { get; set; }
    //Anotacion para que solo muestre la fecha sin la hora
    [DataType(DataType.Date)]
    0 referencias
    public DateTime FechaNacimiento { get; set; }
    //Clave foranea
    0 referencias
    public int CategoriaId { get; set; }
    //Propiedad de navegacion
    0 referencias
    public Categoria Categoria { get; set; }
}
```

17. Agregar un controlador de tipo Controlador de MVC con vistas que usa Entity Framework que gestione la tabla Categoría utilizando la clase de datos de contexto **ApplicationDbContext**



Add MVC Controller with views, using Entity Framework

X

Model class:

Data context class: +

Views:

Generate views

Reference script libraries

Use a layout page:

...

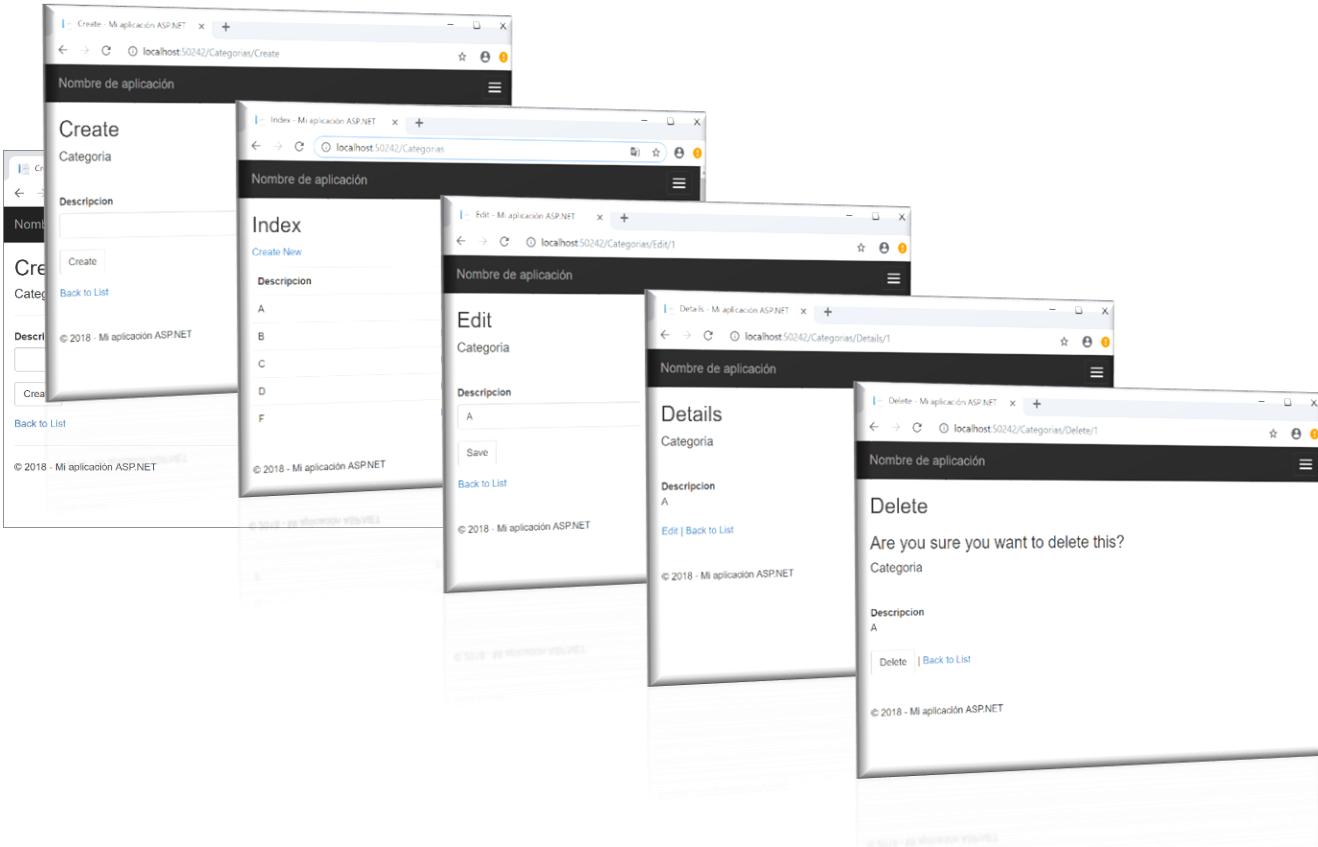
(Leave empty if it is set in a Razor _viewstart file)

Controller name:

Add

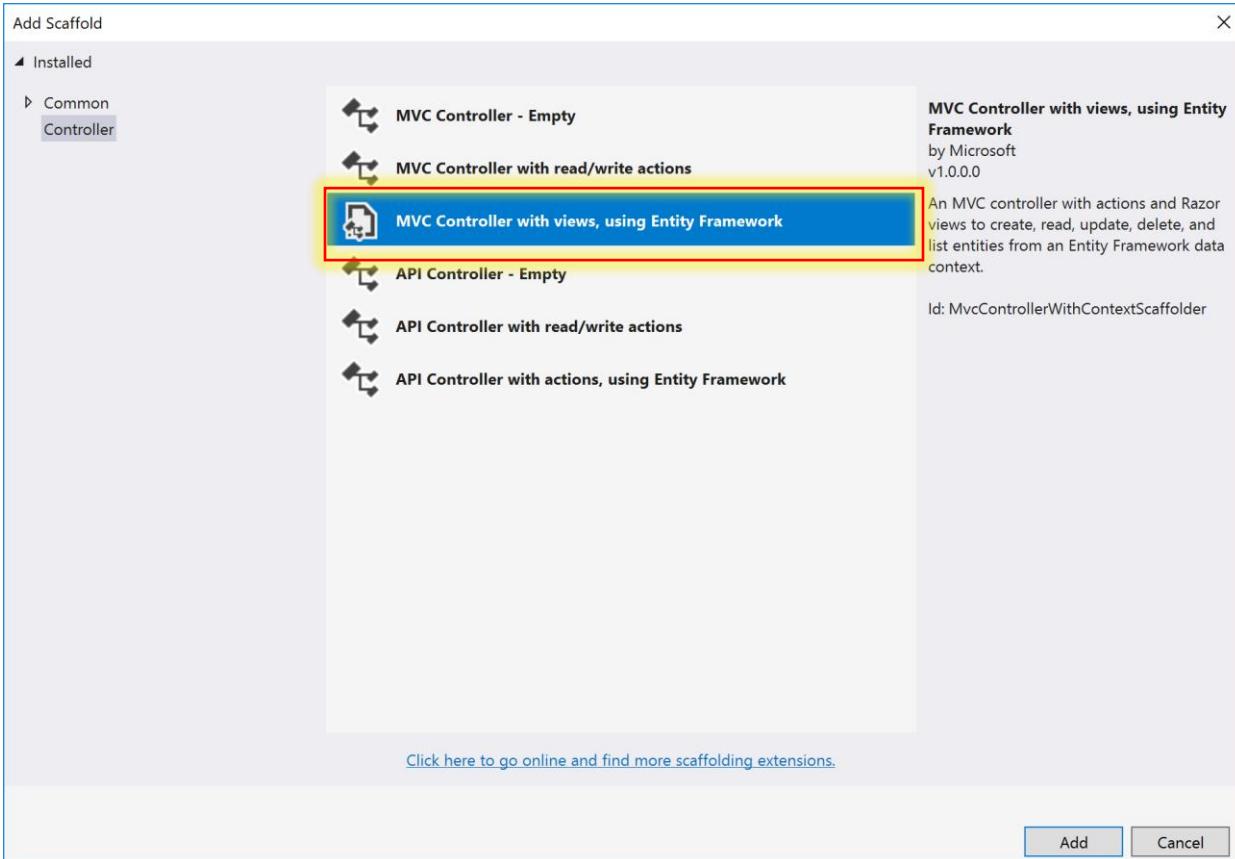
Cancel

18. Corremos y probamos todas las funcionalidades



Entity framework Core es una librería creada por Microsoft que nos permite acceder a las bases de datos, principalmente relacionales, de una forma muy simple y amigable, ya que nos permite utilizar código en vez de SQL, tiene mucha aceptación en el mercado tiene funcionalidades muy potentes como por ejemplo, soporte para la concurrencia, soporte para transacciones, caché, también se utilizan **entidades** como representaciones de las tablas, nos permite elegir el tipo de estructura o de desarrollo que queremos hacer, ya sea a través de code First o de Database first; sobre ambos escenarios veremos ejemplos en el Curso, y permite construir un front end aceptable basado en los modelos.

19. Agregar un controlador de tipo Controlador de MVC con vistas que usa Entity Framework que gestione la tabla **Empleado** utilizando la clase de datos de contexto **ApplicationDbContext**



Add MVC Controller with views, using Entity Framework

X

Model class: Empleado (AppEmpleadosV3.Models)

Data context class: ApplicationDbContext (AppEmpleadosV3.Data) +

Views:

Generate views
 Reference script libraries
 Use a layout page:
 ...

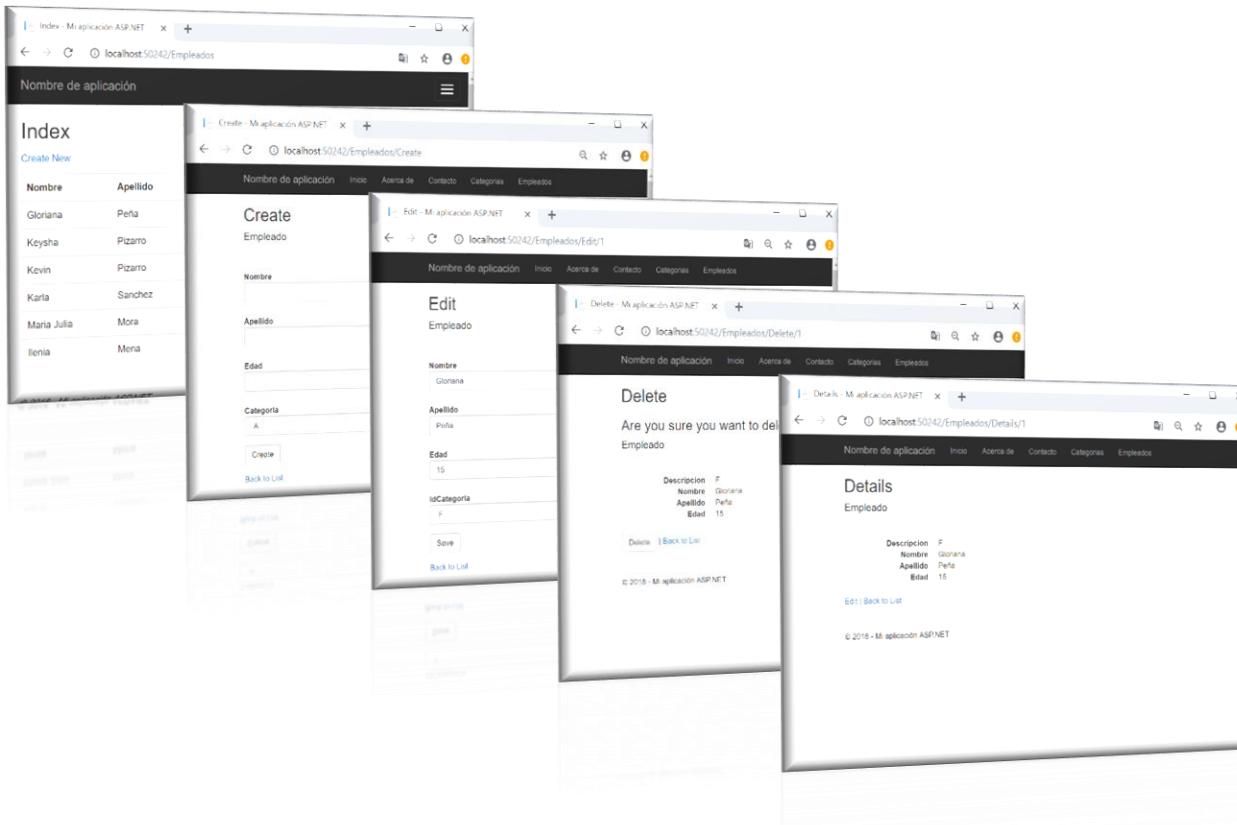
(Leave empty if it is set in a Razor _viewstart file)

Controller name: EmpleadoController Corregir el nombre del controlador

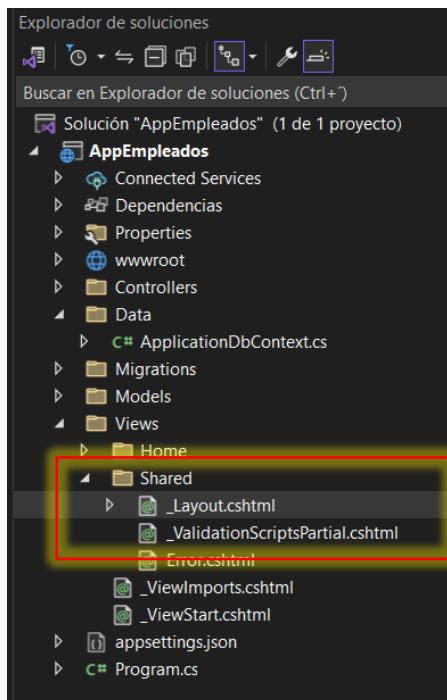
Add Cancel

20. Corremos y probamos todas las funcionalidades

21. Agregar nuevos registros y comprobar que el CRUD funciona correctamente.



22. Agregar los hipervínculos para tener acceso a los nuevos controladores en el menú



```
19     </head>
20     <body>
21         <header>
22             <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3">
23                 <div class="container">
24                     <a class="navbar-brand" asp-area="" asp-controller="Home" asp-action="Index">AppEmpleadosV3</a>
25                     <button class="navbar-toggler" type="button" data-toggle="collapse" data-target=".navbar-collapse" aria-controls="navbarsExample3">
26                         aria-expanded="false" aria-label="Toggle navigation">
27                         <span class="navbar-toggler-icon"></span>
28                     </button>
29                     <div class="navbar-collapse collapse d-sm-inline-flex flex-sm-row-reverse">
30                         <partial name="_LoginPartial" />
31                         <ul class="navbar-nav flex-grow-1">
32                             <li class="nav-item">
33                                 <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>
34                             </li>
35                             <li class="nav-item">
36                                 <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
37                             </li>
38                             <li class="nav-item">
39                                 <a class="nav-link text-dark" asp-area="" asp-controller="Categorias" asp-action="Index">Categorías</a>
40                             </li>
41                             <li class="nav-item">
42                                 <a class="nav-link text-dark" asp-area="" asp-controller="Empleados" asp-action="Index">Empleados</a>
43                             </li>
44                         </ul>
45                     </div>
46                 </div>
47             </nav>
48         </header>
49         <div class="container">
50             <partial name="_CookieConsentPartial" />
51             <main role="main" class="pb-3">
52                 @RenderBody()
53             </main>
54         </div>
55
56         <footer class="border-top footer text-muted">
57             <div class="container">
58                 &copy; 2019 - AppEmpleadosV3 - <a asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
59             </div>

```

23. A partir de la versión C#8, hay que especificar explícitamente las variables que van a permitir valores nulos. Para omitir estas advertencias modifique el archivo de propiedades del proyecto cambiando la propiedad Nullable por disable

The screenshot shows the Visual Studio IDE interface. On the left, the 'Explorador de objetos de SQL Server' (SQL Server Object Explorer) is visible. The main area displays the contents of the 'AppEmployee.csproj' file:

```
<Project Sdk="Microsoft.NET.Sdk.Web">
<PropertyGroup>
    <TargetFramework>net6.0</TargetFramework>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
</PropertyGroup>
<ItemGroup>
    <PackageReference Include="Microsoft.EntityFrameworkCore" Version="6.0.0" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="6.0.0" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Relational" Version="6.0.0" />
    <PrivateAssets>all</PrivateAssets>
    <IncludeAssets>runtime; build; native; contentfiles; analyzers</IncludeAssets>
</PackageReference>
    <PackageReference Include="Microsoft.VisualStudio.Web.CodeGenerators.Mvc" Version="6.0.0" />
    <PackageReference Include="Microsoft.VisualStudio.Web.CodeGenerators.NuGet" Version="6.0.0" />
    <PackageReference Include="Swashbuckle.AspNetCore" Version="6.0.0" />
</ItemGroup>
</Project>
```

The 'Explorador de soluciones' (Solution Explorer) on the right shows the project structure for 'AppEmployee'. It includes items like 'Connected Services', 'Dependencias', 'Properties', 'wwwroot', 'Controllers', 'Data' (containing 'ApplicationDbContext.cs'), 'Migrations', 'Models' (containing 'Categoria.cs', 'Empleado.cs', 'ErrorViewModel.cs'), 'Views' (containing 'Categories', 'Empleados', 'Home', 'Shared' with 'Layout.cshtml'), and 'Layout.cshtml'.

Gestionar (CRUD) información tanto de Categorías como de Empleados.

Categorías

The screenshot shows a web browser window displaying the 'Index' page for 'Categorías'. The URL in the address bar is 'localhost:7127/Categorias'. The page header includes links for 'AppEmpleados', 'Home', 'Privacy', 'Categorías', and 'Empleados'. The main content area has a title 'Index' and a link 'Create New'. Below is a table with three rows, each representing a category:

Nombre	
A	Edit Details Delete
B	Edit Details Delete
C	Edit Details Delete

Empleados

AppEmpleados Home Privacy Categorías Empleados

Index

[Create New](#)

Nombre	Apellidos	FechaNacimiento	Categoría	
GLORIANA	PEÑA RAMIREZ	9/12/2023	2	Edit Details Delete
KEVIN	PIZARRO PEÑA	17/12/2023	1	Edit Details Delete
KEYSHA	PIZARRO PENA	10/12/2023	1	Edit Details Delete

Realizar los siguientes cambios para mejorar el diseño visual de los datos.

Cambios de diseño en Categoría

Lista de categorías

[Nueva categoría](#)

Nombre

A	Editar Ver Borrar
B	Editar Ver Borrar
C	Editar Ver Borrar

Editar la información de categoría

Ingrese los nuevos datos

Nombre

[Guardar](#)

[Volver a la lista](#)

Detalles de la categoría

Categoría

Nombre

A

[Editar](#) | [Volver a la lista](#)

Eliminar una categoría

Está seguro de querer borrar esta categoría?

Categoría a eliminar

Nombre

A

[Eliminar](#) | [Volver a la lista](#)

Cambios de diseño en Empleados

Vista index

Lista de empleados			
Nombre	Apellidos	Fecha de nacimiento	Categoría
GLORIANA	PEÑA RAMIREZ	9/12/2023	B
KEVIN	PIZARRO PEÑA	17/12/2023	A
KEYSHA	PIZARRO PENA	10/12/2023	A

Vista Crear

Empleado nuevo

Ingrese los datos del nuevo empleado

Nombre

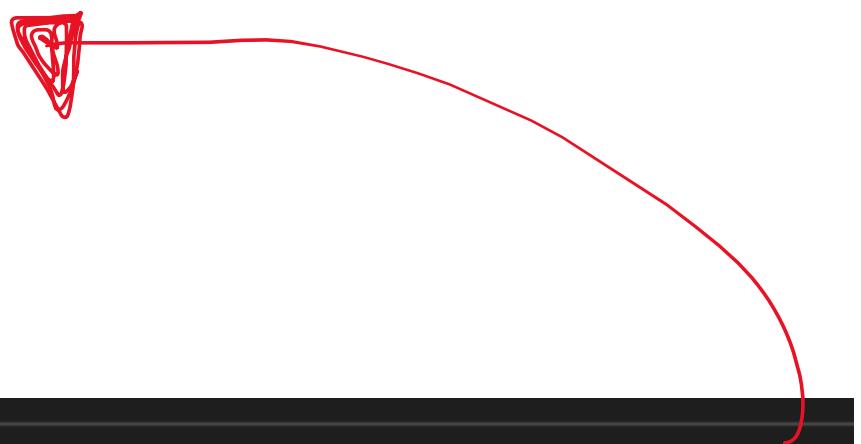
Apellidos

Fecha de Nacimiento
 dd/mm/aaaa

Categoría

A
A
B
C
Crear

[Volver a la lista](#)



```
wwwroot
└─ Controllers
    └─ C# CategoríasController.cs
    └─ C# EmpleadosController.cs
    └─ C# HomeController.cs
```

```
// GET: Empleados/Create
0 referencias
public IActionResult Create()
{
    ViewData["CategoriaId"] = new SelectList(_context.Categorias, "CategoriaId", "Nombre");
    return View();
}
```

Vista Borrar

Borrar

Está seguro de eliminar el empleado?

Empleado

Nombre	GLORIANA
Apellidos	PEÑA RAMIREZ
Fecha de nacimiento	9/12/2023
Categoría	B

[Borrar](#) | [Volver a la lista](#)

Vista Details

Detalles del empleado

Nombre	GLORIANA
Apellidos	PEÑA RAMIREZ
Fecha de Nacimiento:	9/12/2023
Categoría	B

[Editar](#) | [Volver a la lista](#)

DataTables. Tablas avanzadas instantáneamente.

<https://datatables.net>

<https://datatables.net/download/index>

DataTables es un complemento para la biblioteca jQuery Javascript. Es una herramienta altamente flexible, basada en los cimientos de la mejora progresiva, que agrega características avanzadas a cualquier tabla HTML.

Características aplicadas:

- Paginación. Anterior, siguiente y navegación por la página.
- Búsqueda instantánea. Filtrar los resultados por búsqueda de texto.
- Ordenamiento multi-columna. Ordenar los datos por varias columnas a la vez.
- Usa casi cualquier fuente de datos. DOM, Javascript, Ajax y procesamiento del lado del servidor.
- Fácilmente se le pueden aplicar temas. Creador de temas DataTables, Bootstrap 3/4, Foundation y Semantic UI
- Amplia variedad de extensiones. Editor, Botones, Responsive y más.
- Las tablas se adaptan al tamaño de la ventana gráfica.
- Totalmente internacionalizable. Traduce fácilmente las Tablas de datos a múltiples idiomas.
- Software libre de código abierto.

1. Descargar el archivo <https://datatables.net/download/index>

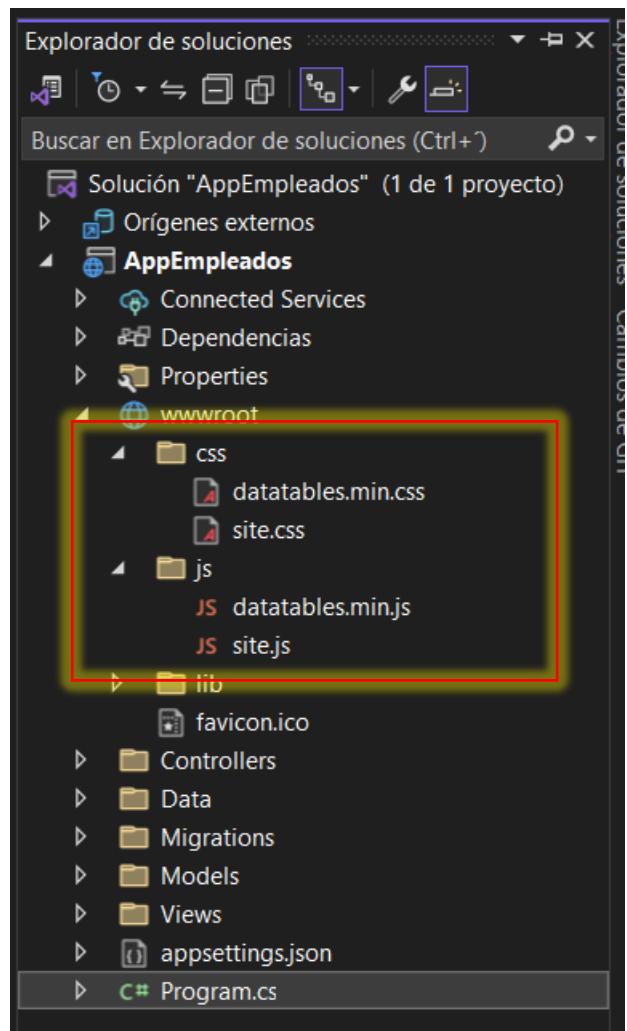
The screenshot shows a web browser window with the URL <https://datatables.net/download/index> in the address bar. The page title is "Step 3. Pick a download method". Below the title, there are tabs for "CDN", "Download" (which is selected), "NPM", "Yarn", and "Bower". A note states: "The files required for the `<link>` and `<script>` tags shown above can be downloaded using the button below. The package is downloaded as a zip file which should be unzipped and uploaded to your web-server." Two checkboxes are checked: "Minify" and "Concatenate". Below these checkboxes is a code editor showing the following HTML code:

```
1 <link rel="stylesheet" type="text/css" href="DataTables/datatables.min.css"/>
2
3 <script type="text/javascript" src="DataTables/datatables.min.js"></script>
```

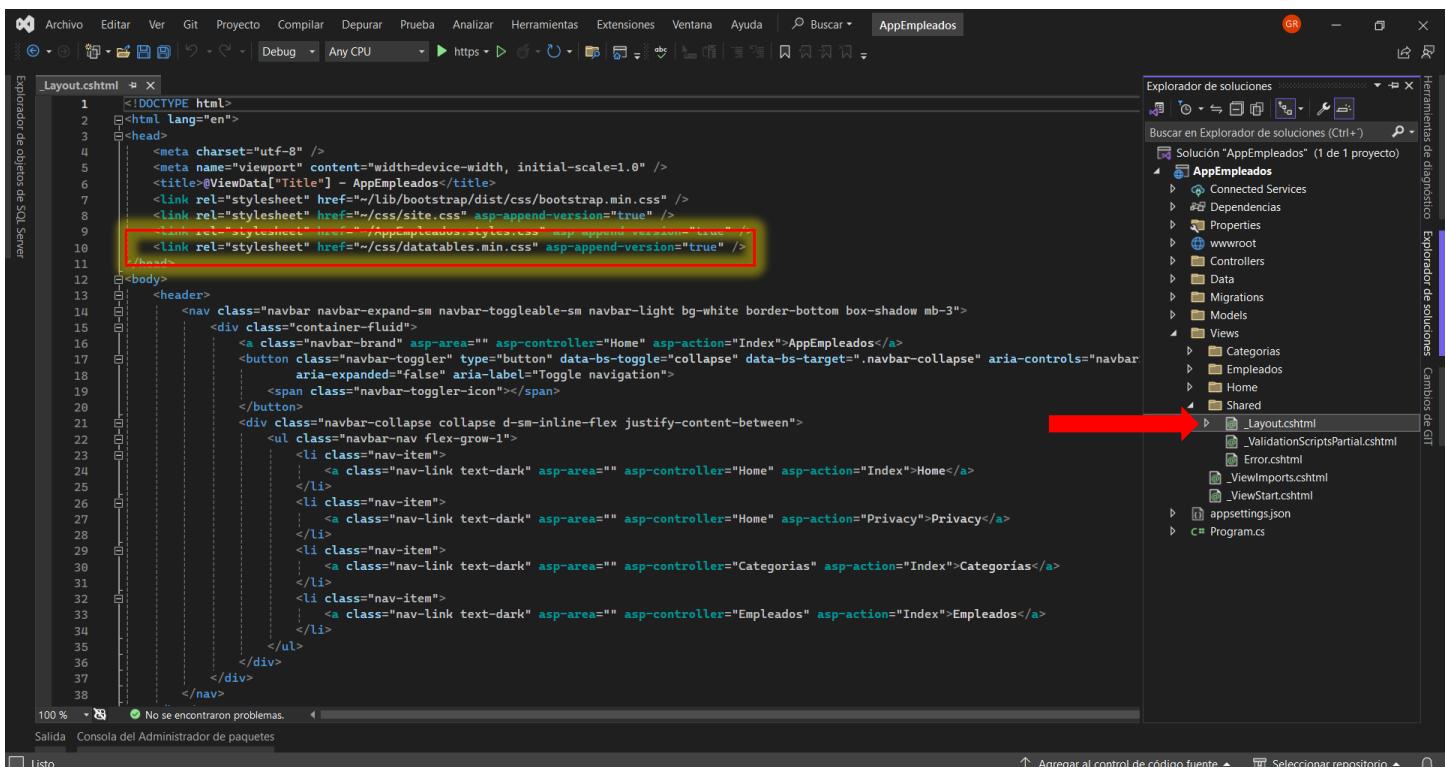
A large red box highlights the "Download files" button at the bottom of the code editor area. At the bottom right of the page, there is a logo for "DataTables" and the text: "DataTables designed and created by SpryMedia Ltd. © 2007-2019 MIT licensed. Privacy policy. Supporters. SpryMedia Ltd is registered in Scotland, company no. SC456502."

2. Agregar el archivo

- a) *jquery.dataTables.min.js* a la carpeta Scripts
- b) *jquery.dataTables.min.css* a la carpeta Content



3. Agregar en el archivo **_Layout.cshtml** las referencias a los nuevos archivos agregados tanto en la carpeta **css** como en la carpeta **js**.

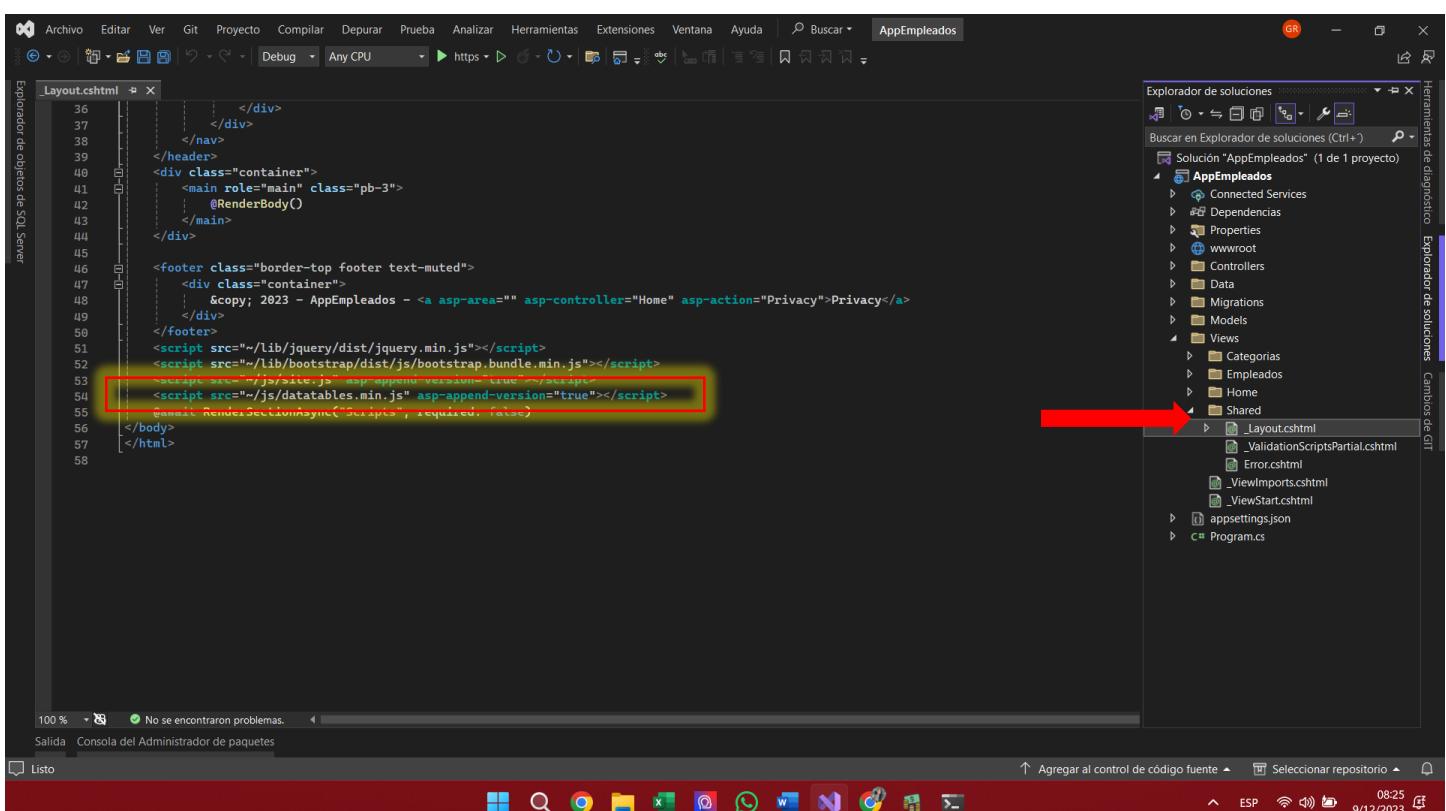


```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8" />
5   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6   <title>@ViewData["Title"] - AppEmpleados</title>
7   <link rel="stylesheet" href="/lib/bootstrap/dist/css/bootstrap.min.css" />
8   <link rel="stylesheet" href="/css/site.css" asp-append-version="true" />
9   <link rel="stylesheet" href="/~/AppEmpleados.styles.css" asp-append-version="true" />
10  <link rel="stylesheet" href="/~/css/datatables.min.css" asp-append-version="true" />
11 </head>
12 <body>
13   <header>
14     <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3">
15       <div class="container-fluid">
16         <a class="navbar-brand" asp-area="" asp-controller="Home" asp-action="Index">AppEmpleados</a>
17         <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
18           <span class="navbar-toggler-icon"></span>
19         </button>
20         <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
21           <ul class="navbar-nav flex-grow-1">
22             <li class="nav-item">
23               <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>
24             </li>
25             <li class="nav-item">
26               <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
27             </li>
28             <li class="nav-item">
29               <a class="nav-link text-dark" asp-area="" asp-controller="Categorias" asp-action="Index">Categorias</a>
30             </li>
31             <li class="nav-item">
32               <a class="nav-link text-dark" asp-area="" asp-controller="Empleados" asp-action="Index">Empleados</a>
33             </li>
34           </ul>
35         </div>
36       </div>
37     </nav>
38   </header>
39   <div class="container">
40     <main role="main" class="pb-3">
41       @RenderBody()
42     </main>
43   </div>
44
45   <footer class="border-top footer text-muted">
46     <div class="container">
47       &copy; 2023 - AppEmpleados - <a asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
48     </div>
49   </footer>
50   <script src="/lib/jquery/dist/jquery.min.js"></script>
51   <script src="/lib/bootstrap/dist/js/bootstrap.bundle.min.js" asp-append-version="true"></script>
52   <script src="/js/site.js" asp-append-version="true"></script>
53   <script src="/js/datatables.min.js" asp-append-version="true"></script>
54   @await RenderSectionAsync("Scripts", required: false)
55 </body>
56 </html>

```

`<link rel="stylesheet" href="/~/css/datatables.min.css" asp-append-version="true" />`



```

36   </div>
37 </div>
38 </header>
39 <div class="container">
40   <main role="main" class="pb-3">
41     @RenderBody()
42   </main>
43 </div>
44
45   <footer class="border-top footer text-muted">
46     <div class="container">
47       &copy; 2023 - AppEmpleados - <a asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
48     </div>
49   </footer>
50   <script src="/lib/jquery/dist/jquery.min.js"></script>
51   <script src="/lib/bootstrap/dist/js/bootstrap.bundle.min.js" asp-append-version="true"></script>
52   <script src="/js/site.js" asp-append-version="true"></script>
53   <script src="/js/datatables.min.js" asp-append-version="true"></script>
54   @await RenderSectionAsync("Scripts", required: false)
55 </body>
56 </html>

```

`<script src="/js/datatables.min.js" asp-append-version="true"></script>`

4. Modificar la vista Index de Categoría, lo resaltado en color amarillo

```
@model IEnumerable<AppEmpleados.Models.Categoría>
{
    ViewData["Title"] = "Categoría";
}
<h1>Lista de categorías</h1>

<p>
    <a asp-action="Create">Nueva categoría</a>
</p>
<table class="table" id="tblCategoria">
    <thead>
        <tr>
            <th>
                @Html.DisplayNameFor(model => model.Nombre)
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
@foreach (var item in Model) {
    <tr>
        <td>
            @Html.DisplayFor(modelItem => item.Nombre)
        </td>
        <td>
            <a asp-action="Edit" asp-route-id="@item.CategoríaId">Editar</a> |
            <a asp-action="Details" asp-route-id="@item.CategoríaId">Ver</a> |
            <a asp-action="Delete" asp-route-id="@item.CategoríaId">Borrar</a>
        </td>
    </tr>
}
    </tbody>
</table>

@section scripts {
    <script>
        let table = new DataTable('#tblCategoria', {
            "language": {
                "decimal": "",
                "emptyTable": "No hay registros",
                "info": "Mostrando _START_ a _END_ de _TOTAL_ Entradas",
                "infoEmpty": "Mostrando 0 to 0 of 0 Entradas",
                "infoFiltered": "(Filtrado de _MAX_ total entradas)",
                "infoPostFix": "",
                "thousands": ",",
                "lengthMenu": "Mostrar _MENU_ Entradas",
                "loadingRecords": "Cargando...",
                "processing": "Procesando...",
                "search": "Buscar:",
                "zeroRecords": "Sin resultados encontrados",
                "paginate": {
                    "first": "Primero",
                    "last": "Último",
                    "next": "Siguiente",
                    "previous": "Anterior"
                }
            },
            "width": "100%"
        });
    </script>
}
```

5. Observamos los cambios con la librería ya aplicada.

- a. Paginación
- b. Orden de las columnas
- c. Búsqueda por campos
- d. Cantidad de registros

AppEmpleados Home Privacy Categorías Empleados

Lista de categorías

Nueva categoría

Mostrar 10 Entradas



Buscar:

Nombre

A

Editar | Ver | Borrar

B

Editar | Ver | Borrar

C

Editar | Ver | Borrar

Mostrando 1 a 3 de 3 Entradas

Anterior 1 Siguiente

Mejorar el diseño con colores en los enlaces en la vista Índex de Categoría

```
Index.cshtml ✎ X
1  @model IEnumerable<AppEmpleados.Models.Categoría>
2
3  @{
4      ViewData["Title"] = "Categoría";
5  }
6
7  <h1>Lista de categorías</h1>
8
9  <p>
10     <a asp-action="Create" class="btn btn-success">Nueva categoría </a>
11 </p>
12 <table class="table" id="tblCategoria">
13     <thead>
14         <tr>
15             <th>
16                 @Html.DisplayNameFor(model => model.Nombre)
17             </th>
18             <th></th>
19         </tr>
20     </thead>
21     <tbody>
22     @foreach (var item in Model) {
23         <tr>
24             <td>
25                 @Html.DisplayFor(modelItem => item.Nombre)
26             </td>
27             <td>
28                 <a asp-action="Edit" asp-route-id="@item.CategoríaId" class="btn btn-warning">Editar</a> |
29                 <a asp-action="Details" asp-route-id="@item.CategoríaId" class="btn btn-secondary">Ver</a> |
30                 <a asp-action="Delete" asp-route-id="@item.CategoríaId" class="btn btn-danger">Borrar</a>
31             </td>
32         </tr>
33     }
34     </tbody>
35 </table>
```

Resultado

The screenshot shows a web application interface titled 'Lista de categorías'. At the top, there is a navigation bar with links: 'AppEmpleados', 'Home', 'Privacy', 'Categorías', and 'Empleados'. Below the navigation, a green button labeled 'Nueva categoría' is visible. The main content area has a search bar with 'Mostrar 10 Entradas' and a 'Buscar:' input field. A table lists three categories: 'A', 'B', and 'C'. Each category row contains three buttons: 'Editar' (yellow), 'Ver' (grey), and 'Borrar' (red). Below the table, a message says 'Mostrando 1 a 3 de 3 Entradas' and includes navigation buttons for 'Anterior' and 'Siguiente'.

Mejorar el diseño con iconos en los enlaces en la vista Índex de Categoría

Agregar la referencia a los iconos

The screenshot shows the 'Layout.cshtml' file in a code editor. The line of code highlighted with a red box is:

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.2/font/bootstrap-icons.min.css">
```

This line adds the Bootstrap icons font to the application's CSS resources.

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.2/font/bootstrap-icons.min.css">
```

Agregar a los enlaces los iconos

```
@model IEnumerable<AppEmpleados.Models.Categoría>

 @{
    ViewData["Title"] = "Categoría";
}



# Lista de categorías



Crear nueva categoría



| @Html.DisplayNameFor(model => model.Nombre) |                                                                                                                                                                                                                                                                                                                                                                                                |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| @Html.DisplayFor(modelItem => item.Nombre)  | <a asp-action="Edit" asp-route-id="@item.CategoríaId" class="btn btn-warning"><i class="bi bi-pencil-fill"> Editar</i></a>   <a asp-action="Details" asp-route-id="@item.CategoríaId" class="btn btn-secondary"><i class="bi bi-info-circle-fill"> Ver</i></a>   <a asp-action="Delete" asp-route-id="@item.CategoríaId" class="btn btn-danger"><i class="bi bi-x-circle-fill"> Borrar</i></a> |


```

Resultado

The screenshot shows the 'Listado de categorías' page. At the top, there is a green button labeled '+ Crear nueva categoría'. Below it, there are filters for 'Mostrar' (set to 10) and 'Entradas' (dropdown menu), and a 'Buscar:' input field. The main area displays a table with three rows, each representing a category:

Nombre	
A	<i class="bi bi-pencil-fill"> Editar</i> <i class="bi bi-info-circle-fill"> Ver</i> <i class="bi bi-x-circle-fill"> Borrar</i>
B	<i class="bi bi-pencil-fill"> Editar</i> <i class="bi bi-info-circle-fill"> Ver</i> <i class="bi bi-x-circle-fill"> Borrar</i>
C	<i class="bi bi-pencil-fill"> Editar</i> <i class="bi bi-info-circle-fill"> Ver</i> <i class="bi bi-x-circle-fill"> Borrar</i>

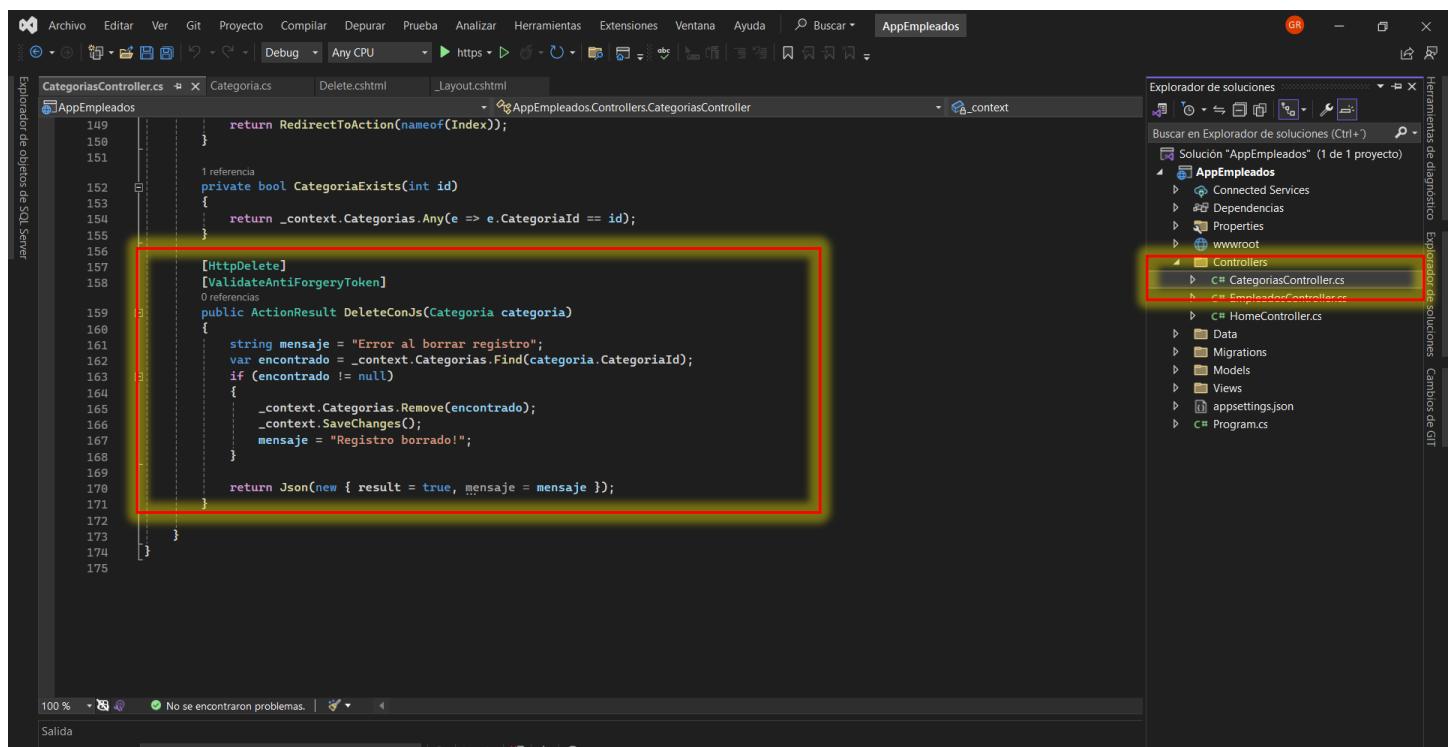
At the bottom, there is a footer with the text 'Mostrando 1 a 3 de 3 Entradas' and navigation buttons for 'Anterior', 'Siguiente', and a page number '1'.

Utilizando JavaScript para la llamada de los métodos de acción Delete

Agregar el método **DeleteConAjax** en el Controlador **Categoría**

```
[HttpDelete]
[ValidateAntiForgeryToken]
public ActionResult DeleteConJs(Categoría categoria)
{
    string mensaje = "Error al borrar registro";
    var encontrado = _context.Categorías.Find(categoría.CategoríaId);
    if (encontrado != null)
    {
        _context.Categorías.Remove(encontrado);
        _context.SaveChanges();
        mensaje = "Registro borrado!";
    }

    return Json(new { result = true, mensaje });
}
```



1. Hacer modificaciones en la vista **Delete** de Categorías

```
@model AppEmpleados.Models.Categoría
{
    ViewData["Title"] = "Categoría eliminar";
}

<h1>Eliminar una categoría</h1>

<h3>Está seguro de querer borrar esta categoría?</h3>
<div>
    <h4>Categoría a eliminar</h4>
    <hr />
    <dl class="row">
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Nombre)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Nombre)
        </dd>
    </dl>

    <form asp-action="Delete" id="formulario">
        <input type="hidden" asp-for="CategoriaId"/>
        <input type="button" id="btnDeleteJs" value="Eliminar" class="btn btn-danger" /> |
        <a asp-action="Index">Volver a la lista</a>
    </form>
</div>

@section Scripts {
    @{
        await Html.RenderPartialAsync("_ValidationScriptsPartial");
    }
    <script>
        async function fetchData() {

            const dataform = new FormData(document.querySelector('#formulario'));
            const response = await fetch('/Categorias/DeleteConJs',
                {
                    method: 'DELETE',
                    body: dataform
                });
            const result = await response.json();
            return result;
        }

        async function deleteCategoria() {

            try {
                var response = await fetchData();
                if (response.result) {
                    console.log(response.result);
                    Swal.fire({
                        title: "El registro fue borrado exitosamente!",
                        icon: "info",
                        timer: 2000,
                        timerProgressBar: false,
                        didOpen: () => {
                            Swal.showLoading();
                            timerInterval = setInterval(() => {
                            }, 100);
                        }
                    })
                }
            } catch (error) {
                console.error(error);
            }
        }
    </script>
}
```

```
        },
        willClose: () => {
            clearInterval(timerInterval);
        }
    }).then((result) => {
        window.location.href = '@Url.Action("Index", "Categorias")'
    });

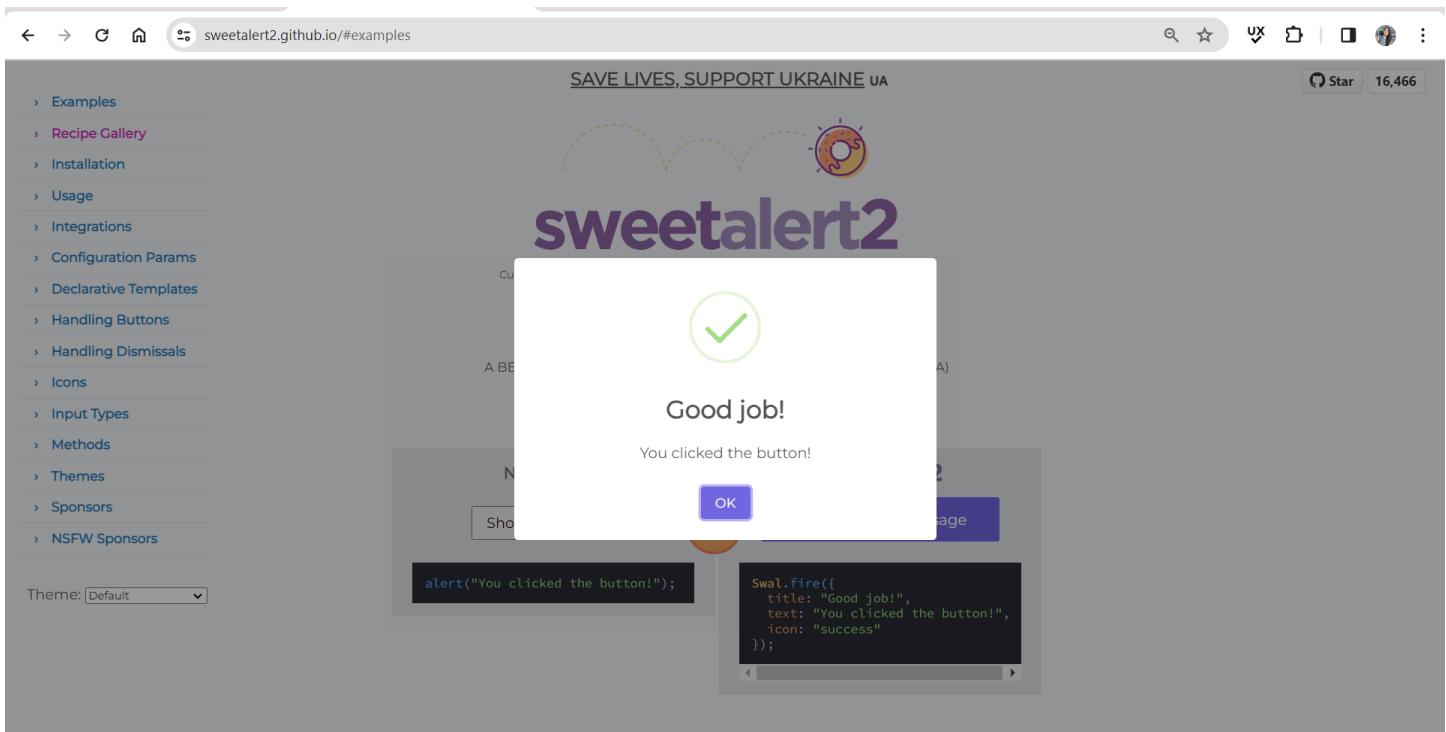
} else {
    Swal.fire({
        position: 'top-end',
        icon: 'error',
        title: response.mensaje,
        showConfirmButton: false,
        timer: 1500
    });
}
} catch (error) {
    console.log(error);
}

}

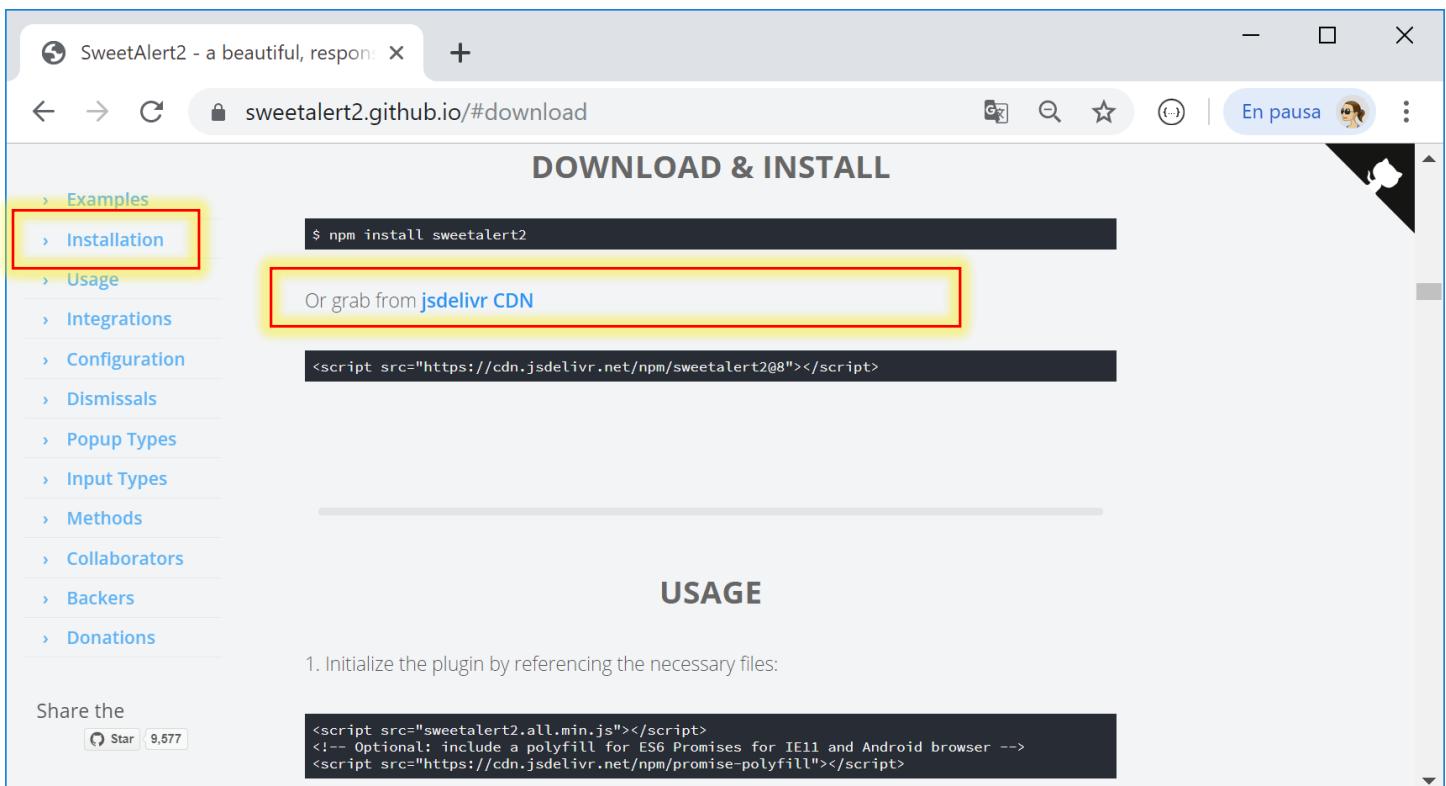
document.addEventListener("DOMContentLoaded", () => {
    document.querySelector("#btnDeleteJs").addEventListener('click', () => {
        Swal.fire({
            title: "Está seguro de eliminar el registro?",
            text: "Tome en cuenta que es un proceso irreversible",
            icon: "warning",
            showCancelButton: true,
            confirmButtonColor: "#3085d6",
            cancelButtonColor: "#d33",
            confirmButtonText: "Si, borrar!"
        }).then((result) => {
            if (result.isConfirmed) {
                deleteCategoria();
            }
        });
    });
});
</script>
}
```

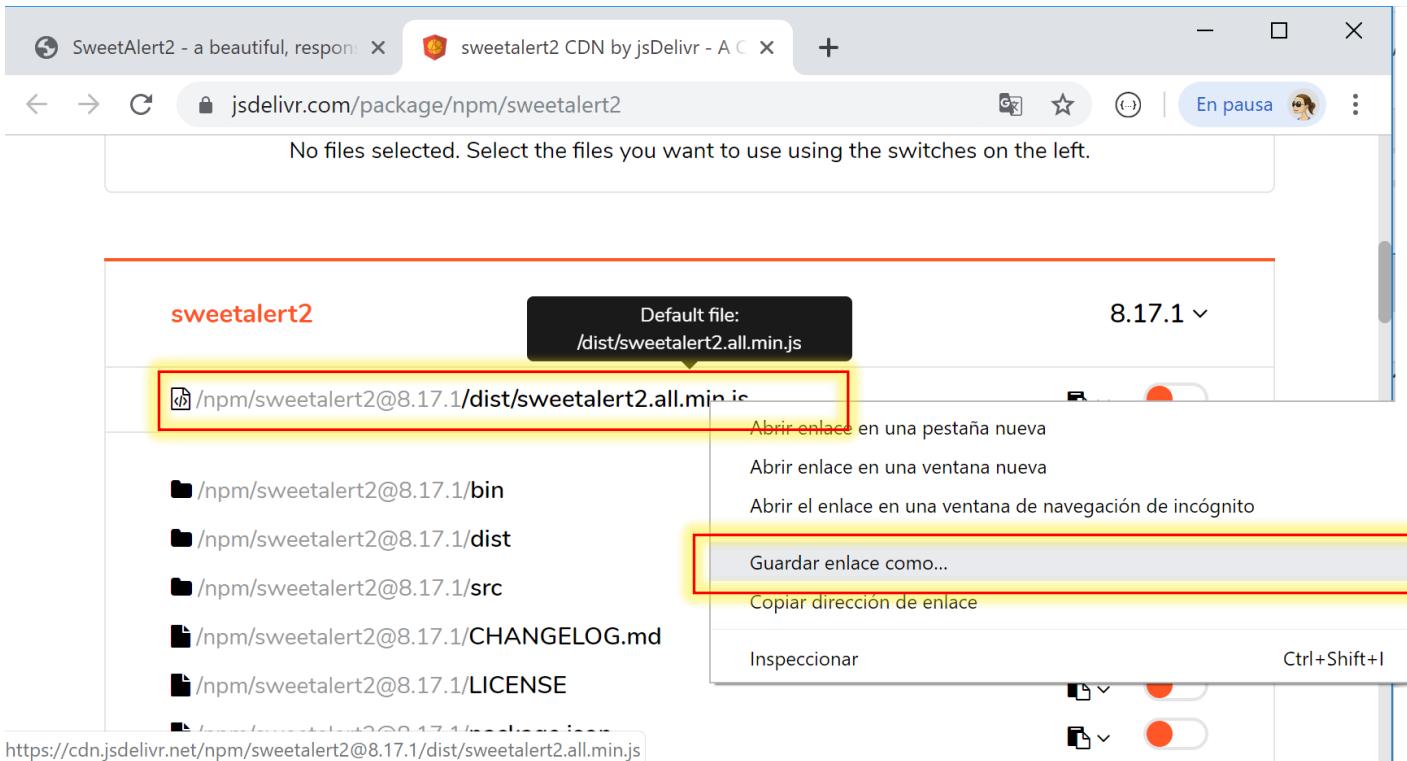
Un reemplazo hermoso, sensible, personalizable, accesible (wai-aria) para los cuadros emergentes de javascript, cero dependencias

<https://sweetalert2.github.io/#usage>



<https://sweetalert2.github.io/#download>





6. Agregar la referencia al nuevo archivo agregado en el archivo _Layout.cshtml.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - AppEmpleados</title>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/AppEmpleados.styles.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/css/datatables.min.css" asp-append-version="true" />
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.11.2/font/bootstrap-icons.min.css">
    <link rel="stylesheet" href="~/css/sweetalert2.min.css" asp-append-version="true" />
</head>
<body>
    <header>
        <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white
border-bottom box-shadow mb-3">
            <div class="container-fluid">
                <a class="navbar-brand" asp-area="" asp-controller="Home" asp-
action="Index">AppEmpleados</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-
bs-target=".navbar-collapse" aria-controls="navbarSupportedContent"
                    aria-expanded="false" aria-label="Toggle navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="navbar-collapse collapse d-sm-inline-flex justify-content-
between">
                    <ul class="navbar-nav flex-grow-1">
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Home"
asp-action="Index">Home</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Home"
asp-action="Privacy">Privacy</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-
controller="Categorias" asp-action="Index">Categorías</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-
controller="Empleados" asp-action="Index">Empleados</a>
                        </li>
                    </ul>
                </div>
            </div>
        </nav>
    </header>
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer class="border-top footer text-muted">
        <div class="container">
            &copy; 2023 - AppEmpleados - <a asp-area="" asp-controller="Home" asp-
action="Privacy">Privacy</a>
        </div>
    </footer>

```

```
        </div>
    </footer>
<script src="~/lib/jquery/dist/jquery.min.js"></script>
<script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
<script src="~/js/site.js" asp-append-version="true"></script>
<script src="~/js/datatables.min.js" asp-append-version="true"></script>
<script src="~/js/sweetalert2.all.min.js" asp-append-version="true"></script>
@await RenderSectionAsync("Scripts", required: false)
</body>
</html>
```

Evaluación

Esta guía corresponde a las dos primeras prácticas en clases correspondientes a 5% cada una

ITEM	Porcentaje
01 Primer App MVC con vistas Entity Framework	5%
02 MVC 2 Tablas relacionadas + Bootstrap	5%