

Actividad evaluable UT8

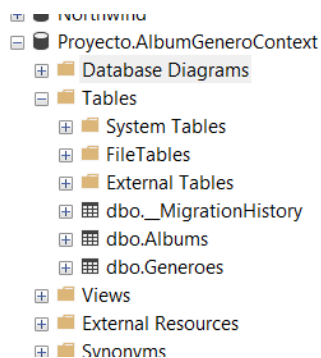
I. Requerimiento 1 (ASP.Net Core MVC):

1. Descripción

A través de ASP-Net Core MVC crearemos una aplicación web que pueda listar, modificar y borrar registros de una base de datos.

2. Base de datos

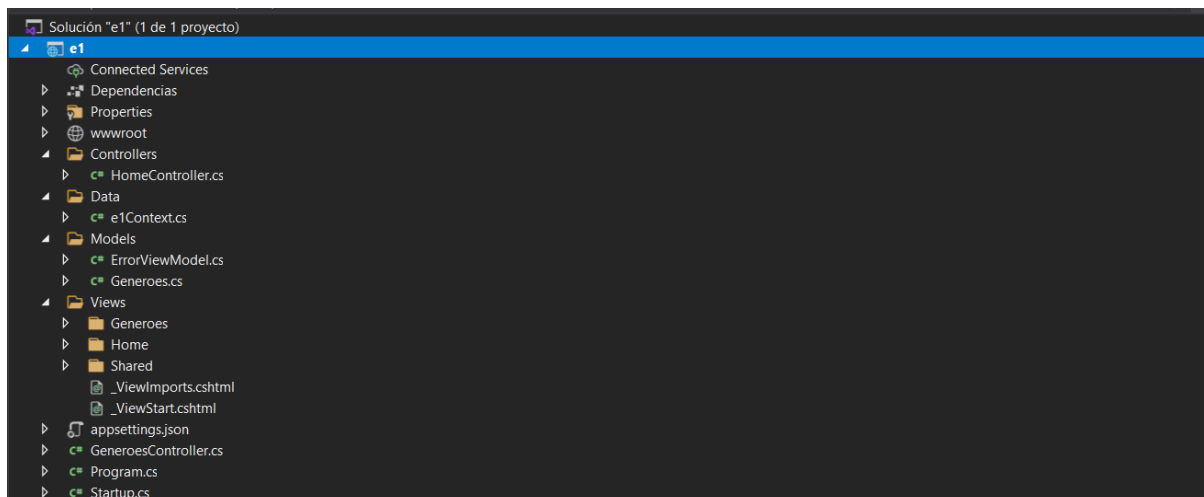
Se utiliza una base de datos existente.



Generoes	
Id	
Name	
Description	

3. Código:

Estructura de la solución:



Algunos modificaciones del código fueron las mostradas a continuación:

Se creó un modelo de la base de datos que representa la tabla *Generoes*:

```
namespace e1.Models
{
    18 referencias
    public class Generoes
    {
        11 referencias
        public int Id { get; set; }
        12 referencias
        public string Name { get; set; }
        12 referencias
        public string Description { get; set; }
    }
}
```

En *_Layout.cshtml* se agregó un elemento para poder ver un listado de los géneros

```
</button>
<div class="navbar-collapse collapse d-sm-inline-flex flex-sm-row-reverse">
    <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="Generoes" asp-action="Index">Generoes</a>
        </li>
    </ul>
</div>
```

En *appsettings.json* se modificó la cadena de conexión

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "e1Context": "Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=Proyecto.AlbumGeneroContext;Integrated Security=True;Connect Timeout=30;Encrypt=False;"
  }
}
```

4-Resultados:

Listado de los géneros:

e1	Home	Privacy	Generoes
<h2>Index</h2> <p>Create New</p>			
Name	Description		
rock	genero rock es cool	Edit	Details Delete
metal	genero metal es cool	Edit	Details Delete
balada	genero balada es cool	Edit	Details Delete
pop	genero pop es cool	Edit	Details Delete
reggae	genero reggae es fine	Edit	Details Delete

Editar género:

e1

Home

Privacy

Generoes

Edit

Generoes

Name

rock

Description

genero rock es cool

Save

[Back to List](#)

Agregar género

e1

Home

Privacy

Generoes

Create

Generoes

Name

Blues

Description

Genero blues es goosebumps

Create

[Back to List](#)

Eliminar género

reggae	genero reggae es fine	Edit Details Delete
--------	-----------------------	---

Resultado de eliminación

Index

[Create New](#)

Name	Description	
rock	genero rock es cool	Edit Details Delete
metal	genero metal es cool	Edit Details Delete
balada	genero balada es cool	Edit Details Delete
pop	genero pop es cool	Edit Details Delete

II. Requerimiento 2 (ASP.Net Razor Pages):

1. Descripción

Se desarrolló los 2 primeros puntos de este [tutorial](#)

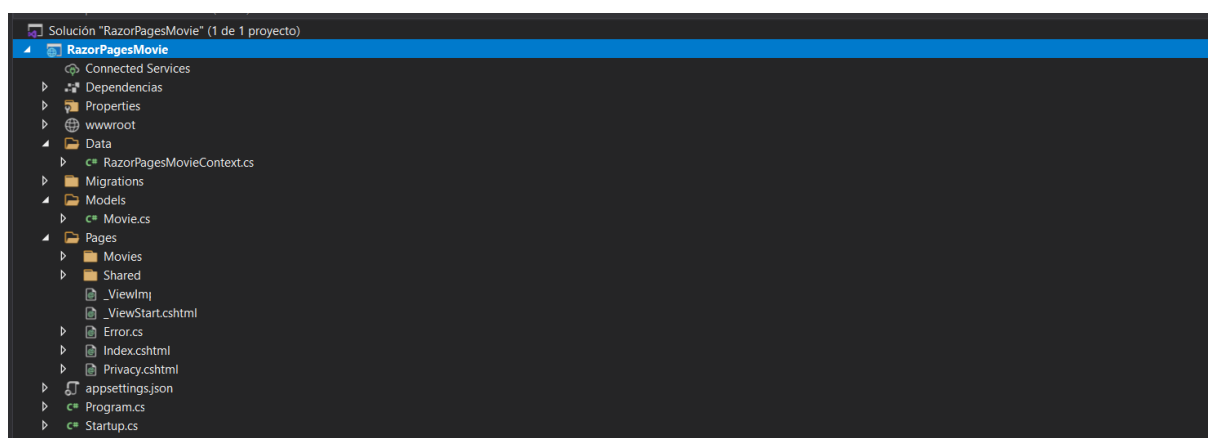
2. Base de datos

A través de las clases del modelo generamos una bases de datos nueva

3. Código:

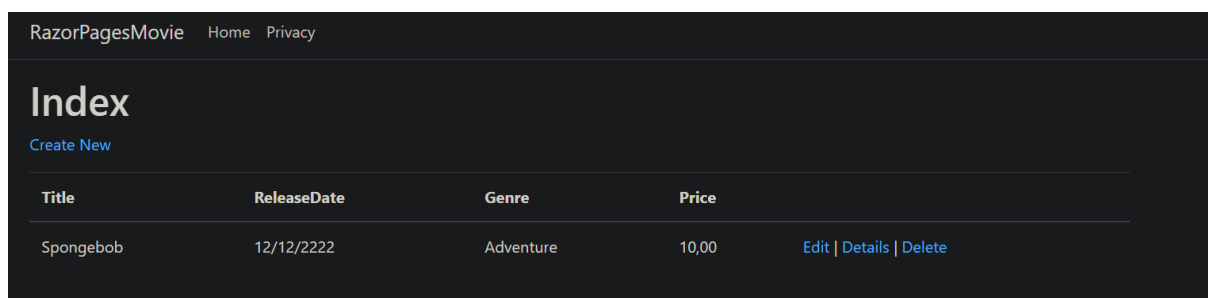
Es el mismo que el presentado en el tutorial de Microsoft.

Estructura de la solución:



4-Resultados:

Al acceder a <http://localhost:port/movies> en el navegador veremos:



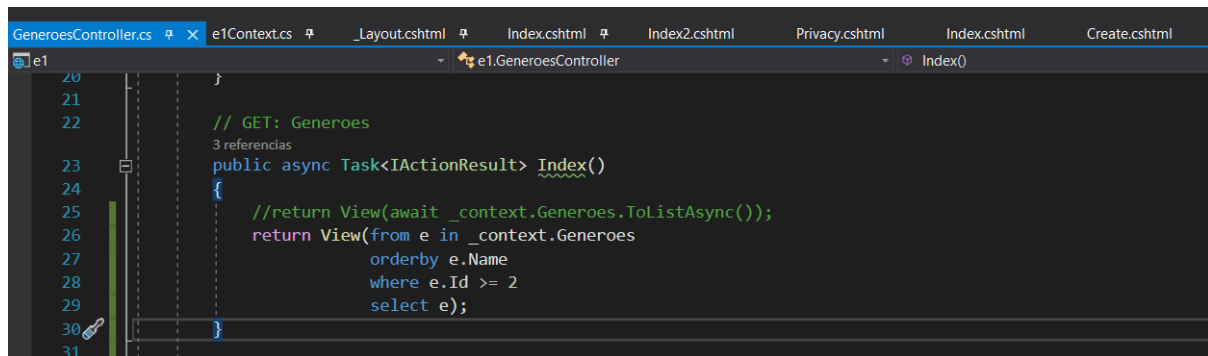
Podremos agregar, crear y modificar registros de la tabla *Movies*.

III. Requerimiento 3 (Ampliación):

Se mostrara la lista de géneros cuyo id es mayor o igual a 2, ordenado por su nombre.

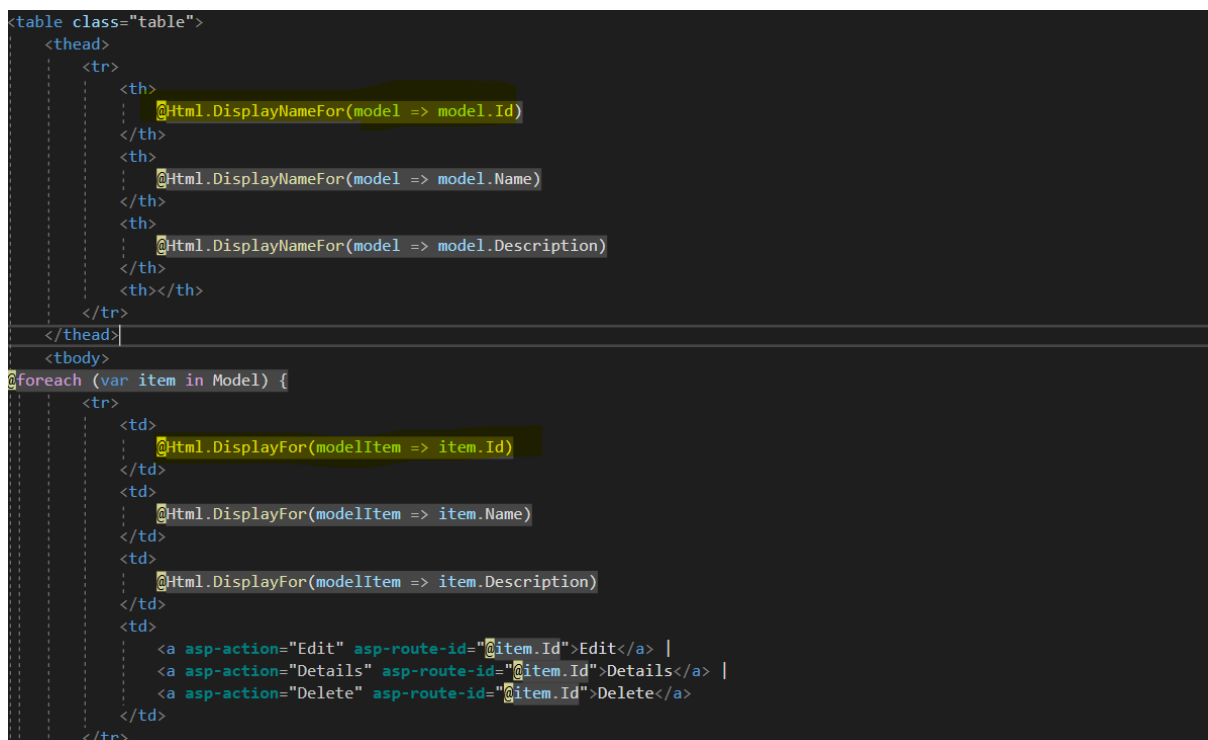
1. Código:

Modificamos el método Index() del controlador para que devuelva el resultado de la consulta previamente descrita.



```
20 }
21
22 // GET: Generoes
23 3 referencias
24 public async Task<IActionResult> Index()
25 {
26     //return View(await _context.Generoes.ToListAsync());
27     return View(from e in _context.Generoes
28                 orderby e.Name
29                 where e.Id >= 2
30                 select e);
31 }
```

Modificamos el archivo index.cshtml para agregar a la vista el campo id



```
<table class="table">
  <thead>
    <tr>
      <th>
        @Html.DisplayNameFor(model => model.Id)
      </th>
      <th>
        @Html.DisplayNameFor(model => model.Name)
      </th>
      <th>
        @Html.DisplayNameFor(model => model.Description)
      </th>
    </tr>
  </thead>
  <tbody>
    <foreach (var item in Model) {
      <tr>
        <td>
          @Html.DisplayFor(modelItem => item.Id)
        </td>
        <td>
          @Html.DisplayFor(modelItem => item.Name)
        </td>
        <td>
          @Html.DisplayFor(modelItem => item.Description)
        </td>
        <td>
          <a asp-action="Edit" asp-route-id="@item.Id">Edit</a> |
          <a asp-action="Details" asp-route-id="@item.Id">Details</a> |
          <a asp-action="Delete" asp-route-id="@item.Id">Delete</a>
        </td>
      </tr>
    }
  </tbody>
</table>
```

2-Resultados:

Ahora notaremos que solo se mostraran los registros cuyo id es mayor o igual que 2 ordenados por nombre.

e1 Home Privacy Generoes

Index

[Create New](#)

Id	Name	Description	
3	balada	genero balada es cool	Edit Details Delete
2	metal	genero metal es cool	Edit Details Delete
4	pop	genero pop es cool	Edit Details Delete