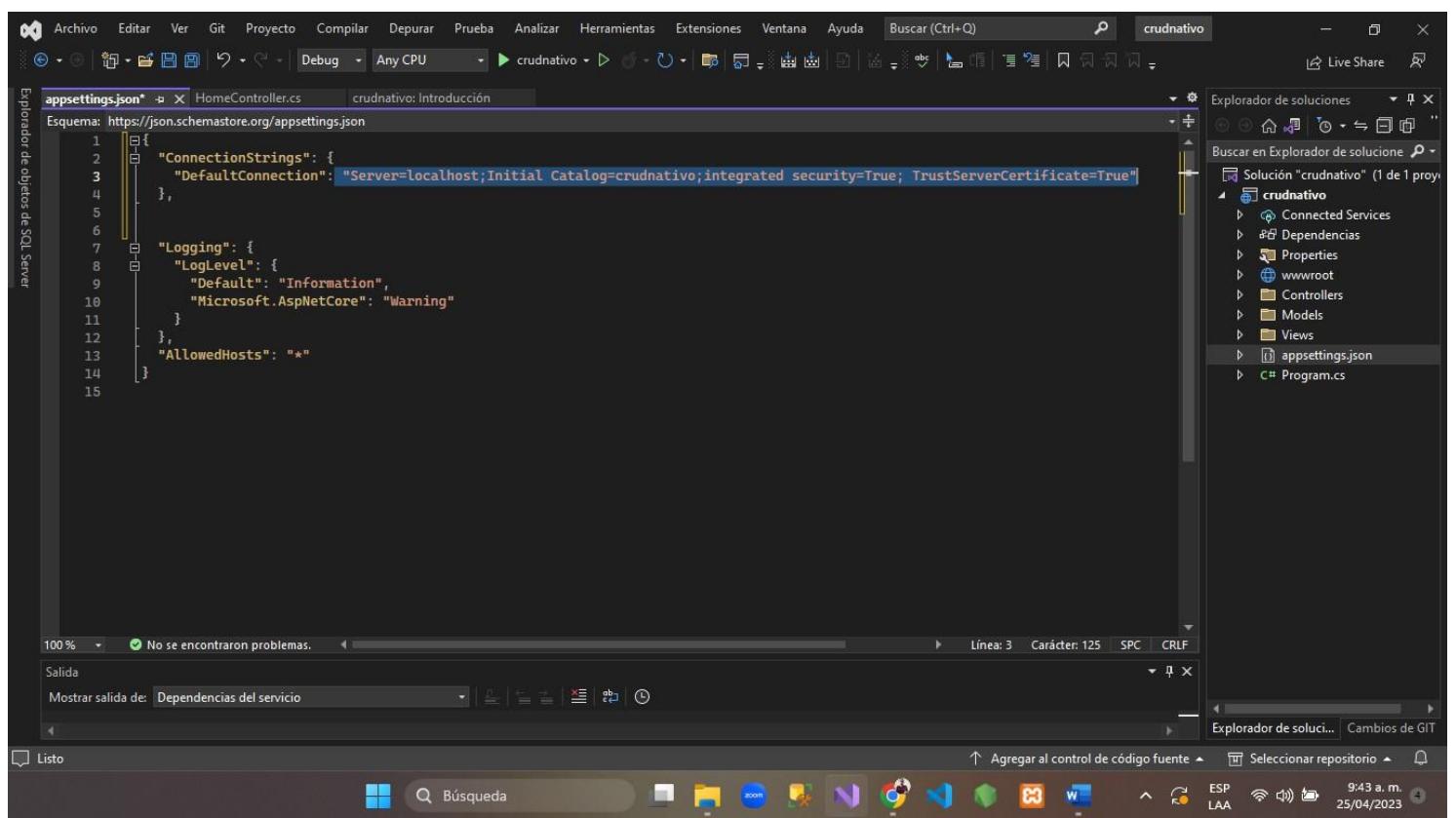


CRUD NATIVO

1. Creamos un nuevo proyecto ASP.Net MVC
2. Creamos la ruta cadena de conexión con sql server

"Server=localhost;Initial Catalog=crudnativo;integrated security=True;
TrustServerCertificate=True"



3. Creamos el modelo: clic derecho aregar clase Libro, definimos los campos de la tabla también podemos utilizar datanotations (validaciones)

The screenshot shows the Microsoft Visual Studio code editor displaying the `Libro.cs` file. The class definition is as follows:

```
namespace crudnativo.Models
{
    public class Libro
    {
        [Key] //Llave primaria
        public int Id { get; set; }

        [Required(ErrorMessage = "Campo Obligatorio")]
        [StringLength(50, ErrorMessage = "El campo debe ser minimo 3 caracteres y maximo 50 caracteres", MinimumLength = 3)]
        public string Titulo { get; set; }

        [Required(ErrorMessage = "Campo Obligatorio")]
        [StringLength(50, ErrorMessage = "El campo debe ser minimo 3 caracteres y maximo 50 caracteres", MinimumLength = 3)]
        public string Descripcion { get; set; }

        [Required(ErrorMessage = "Campo Obligatorio")]
        [DataType(DataType.Date)]
        public DateTime FechaLanzamiento { get; set; }

        [Required(ErrorMessage = "Campo Obligatorio")]
        [StringLength(50, ErrorMessage = "El campo debe ser minimo 3 caracteres y maximo 50 caracteres", MinimumLength = 3)]
        public string Autor { get; set; }

        [Required(ErrorMessage = "Campo Obligatorio")]
        [StringLength(50, ErrorMessage = "El campo debe ser minimo 3 caracteres y maximo 50 caracteres", MinimumLength = 3)]
        public string Precio { get; set; }
    }
}
```

4. Vamos a crear la primera migración. Vamos a program comentamos la línea de código para ahora darle la solución

```

1  using Microsoft.Extensions.Options;
2
3  var builder = WebApplication.CreateBuilder(args);
4
5  // Add services to the container.
6  builder.Services.AddControllersWithViews();
7
8  //configuracion cadena de conexión
9
10 //builder.Services.AddDbContext<ApplicationDbContext>(options =>
11 //options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));
12
13
14 var app = builder.Build();
15
16 // Configure the HTTP request pipeline.
17 if (app.Environment.IsDevelopment())
18 {
19     app.UseExceptionHandler("/Home/Error");
20     // The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.
21     app.UseHsts();
22 }
23
24 app.UseHttpsRedirection();
25 app.UseStaticFiles();
26
27 app.UseRouting();
28
29 app.UseAuthorization();
30
31 app.MapControllerRoute(
    
```

Salida

Mostrar salida de: Dependencias del servicio

Elementos guardados

91 %

No se encontraron problemas.

Línea: 10 Carácter: 1 SPC CRLF

Explorador de soluciones Explorador de soluciones

Buscar en Explorador de soluciones

Solución "crudnativo" (1 de 1 proyecto)

crudnativo

- Connected Services
- Dependencias
- Properties
- wwwroot
- Controllers
- Models
- C# ErrorViewModel.cs
- C# Libro.cs
- Views
- appsettings.json
- C# Program.cs

Explorador de soluciones Cambios de GIT

Agregar al control de código fuente Seleccionar repositorio

ESP LAA 10:37 a. m. 25/04/2023

5. Creamos una nueva carpeta llamada data para crear una clase llamada ApplicationDbContext y crear el método y constructor

```

1  using crudnativo.Models; // llamamos el modelo
2
3  namespace crudnativo.Data
4  {
5      public class ApplicationDbContext : DbContext // hereda de la clase
6      {
7          public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options) // creamos el constructor
8          {
9          }
10
11
12
13
14
15      public DbSet<Libro> Libro { get; set; }
16
17
18
    
```

Salida

Mostrar salida de: Dependencias del servicio

100 %

Línea: 2 Carácter: 46 SPC MIXTO

Explorador de soluciones Explorador de soluciones

Buscar en Explorador de soluciones

Solución "crudnativo" (1 de 1 proyecto)

crudnativo

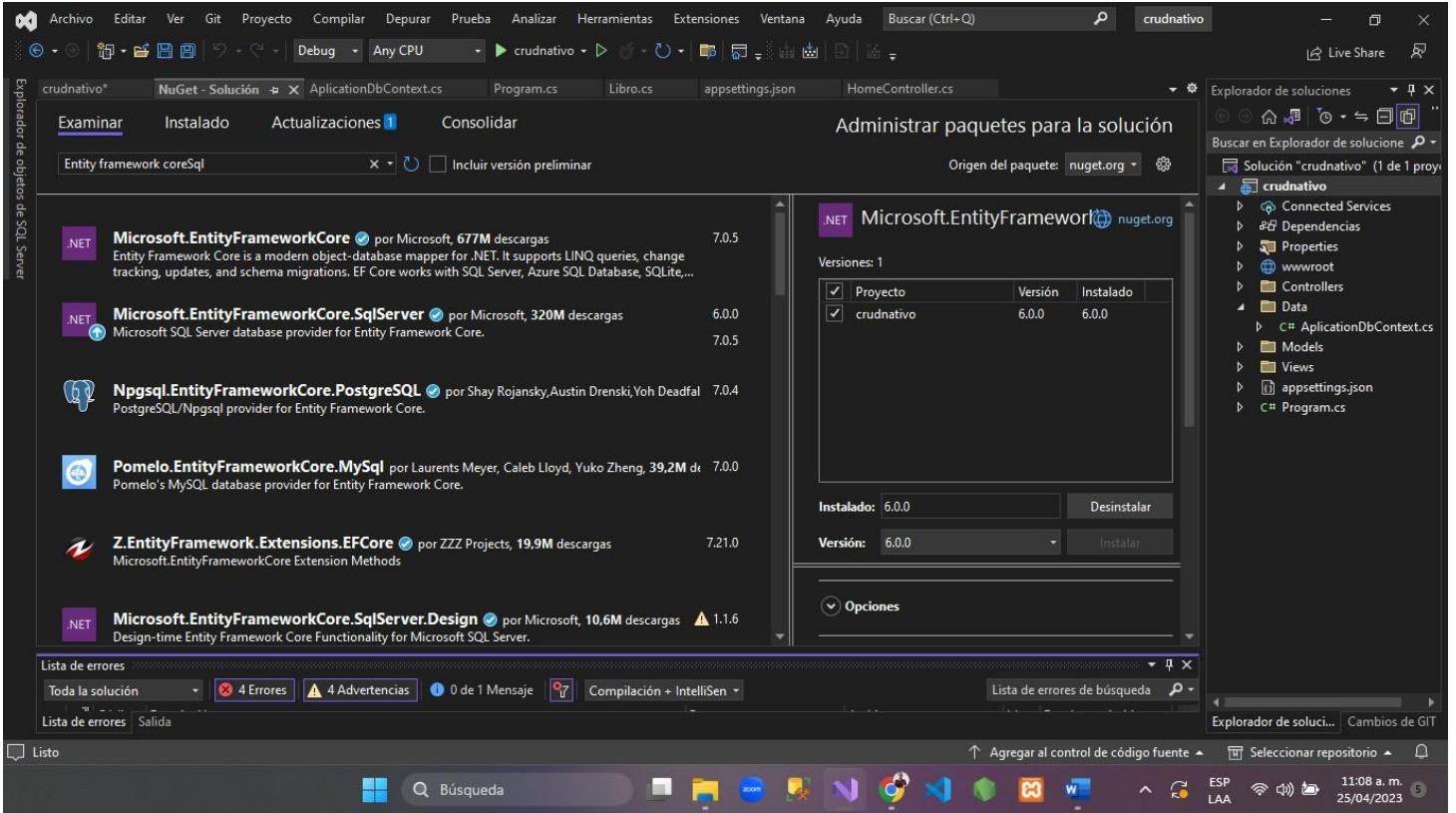
- Connected Services
- Dependencias
- Properties
- wwwroot
- Controllers
- Data
- C# ApplicationDbContext.cs
- Models
- Views
- appsettings.json
- C# Program.cs

Explorador de soluciones Cambios de GIT

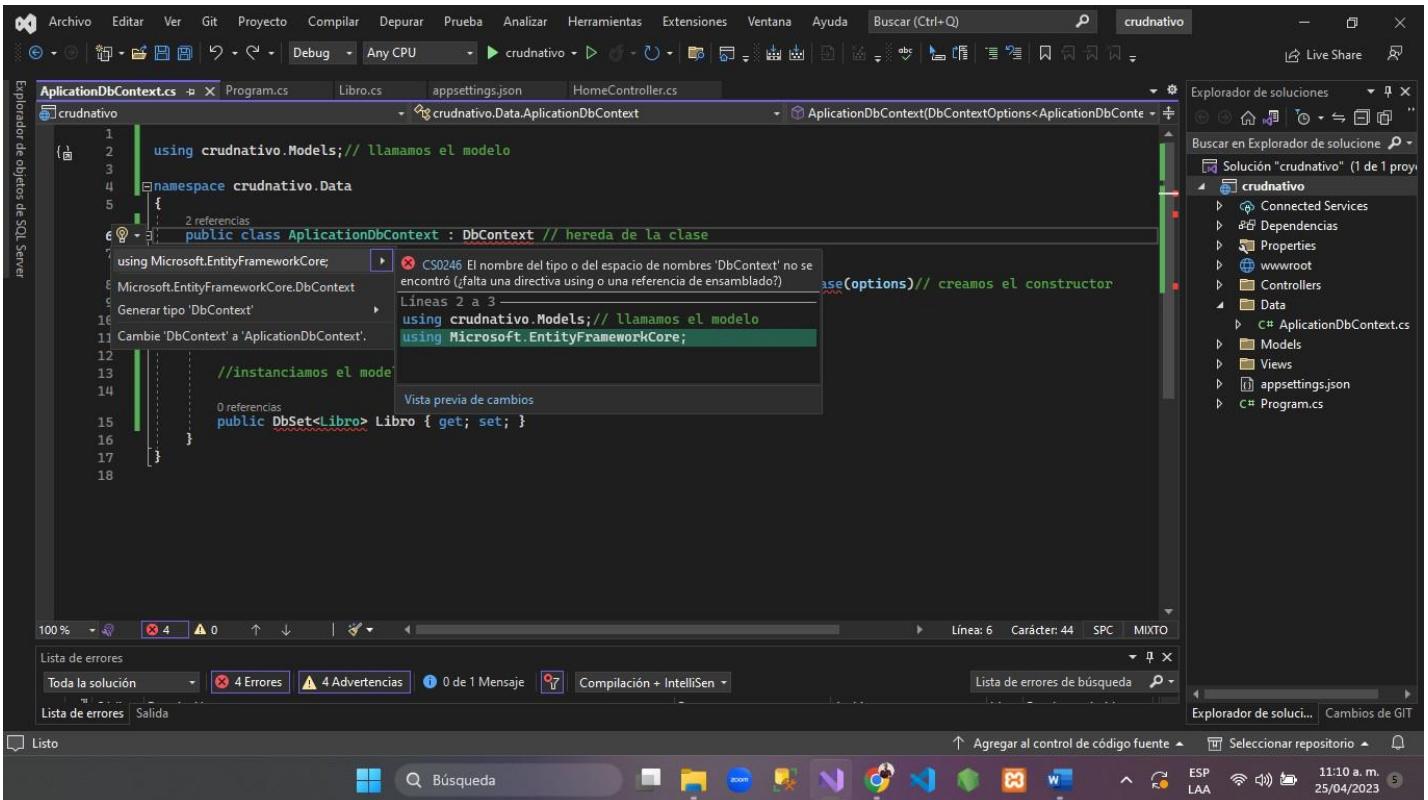
Agregar al control de código fuente Seleccionar repositorio

ESP LAA 11:01 a. m. 25/04/2023

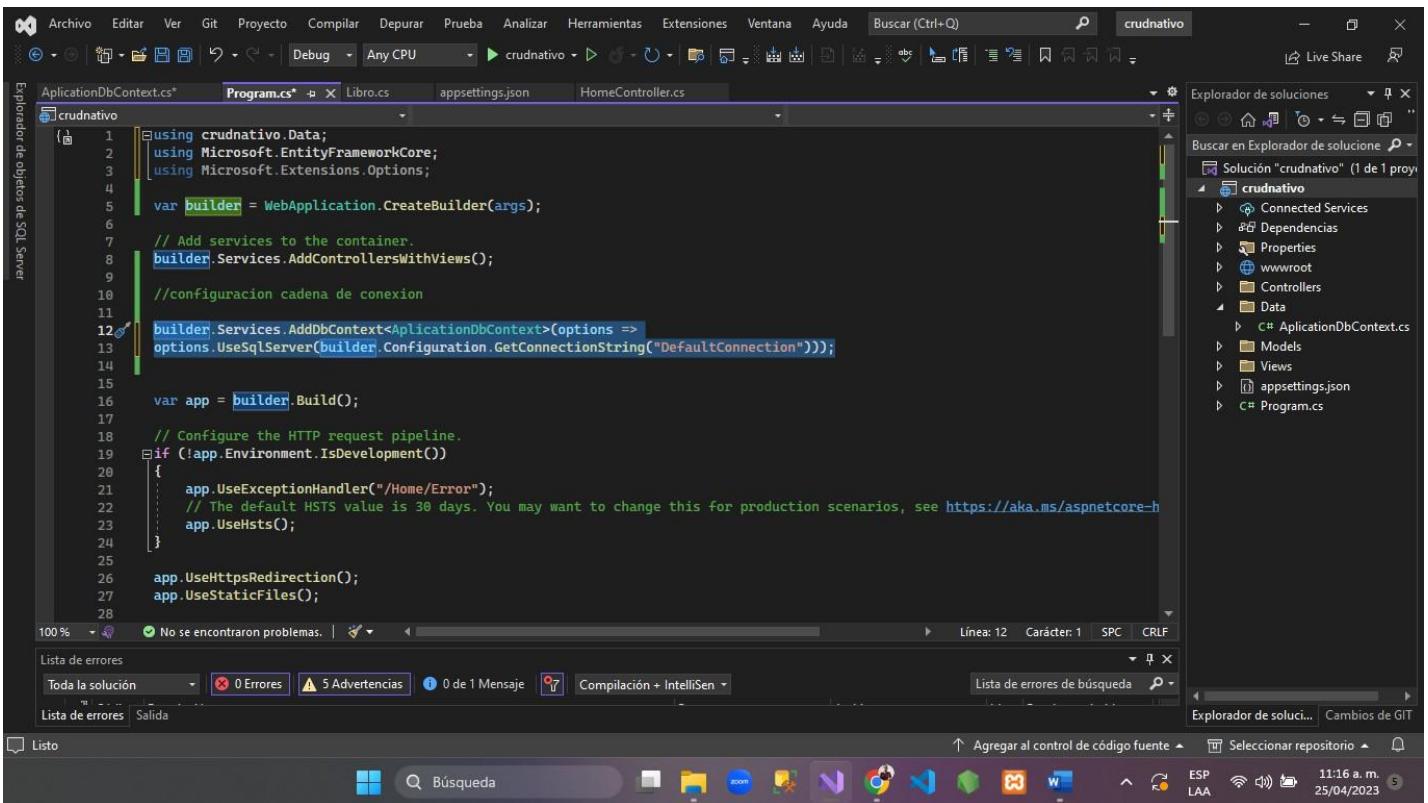
6. Ahora tenemos que instalar las librerías nuget para solucionar los errores. Vamos a herramientas admin de paquetes – administración de paquetes nuguest de la solución, instalamos Entity framework coreSql, (tener en cuenta la version)



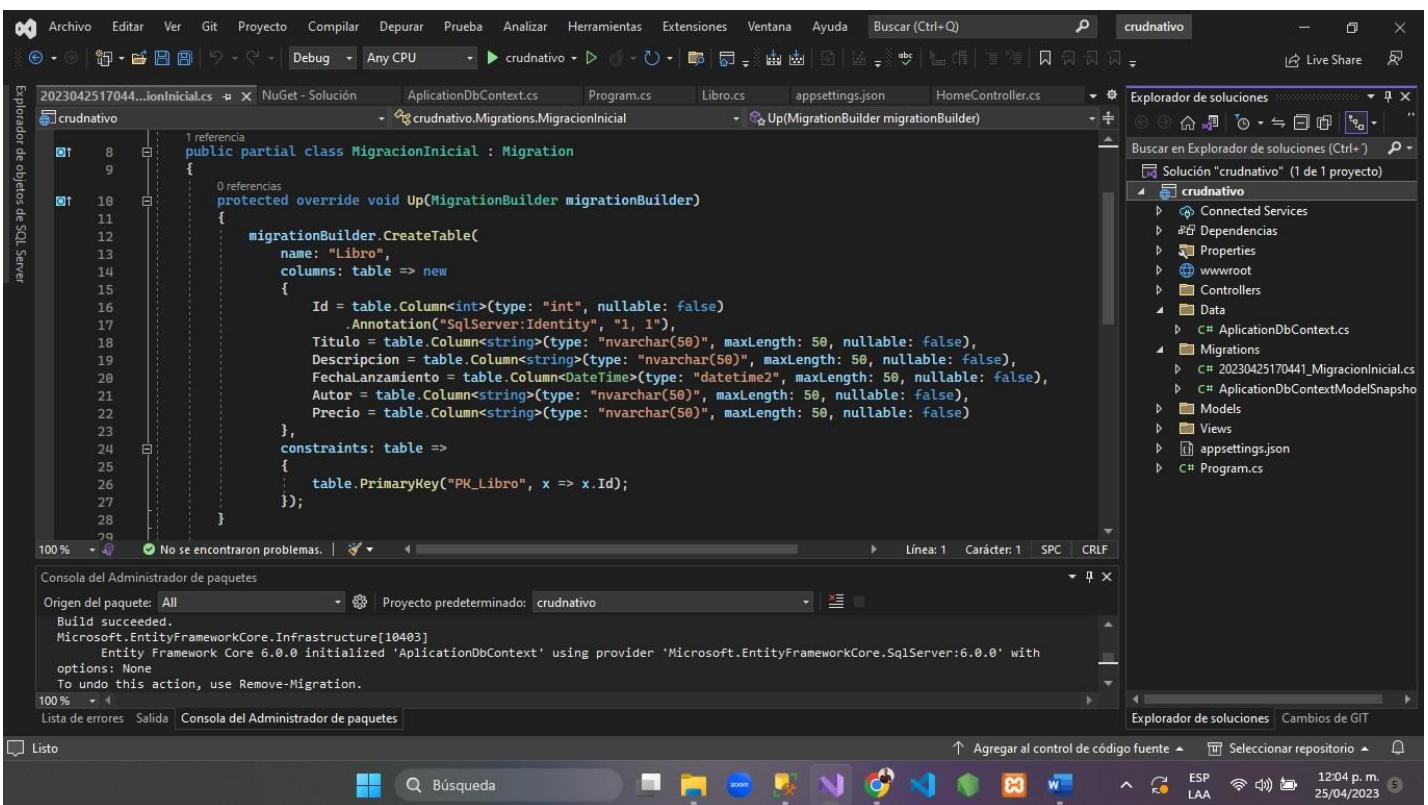
7. Vamos a la clase ApplicationDbContext y agregamos using para utilizar la librería y solucionar errores



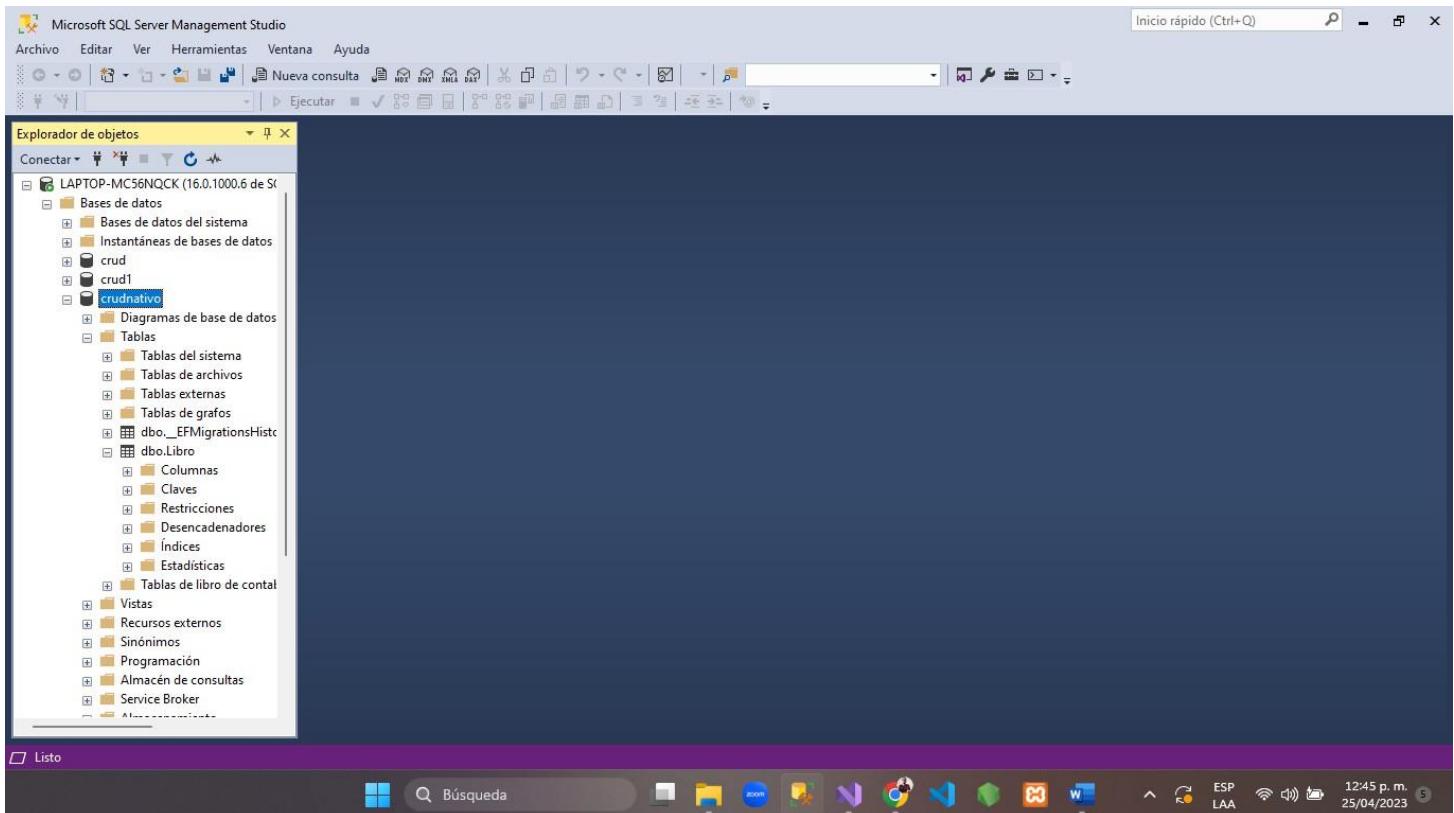
8. Vamos a program y también agregamos la librería que acabamos de instalar para solucionar el error



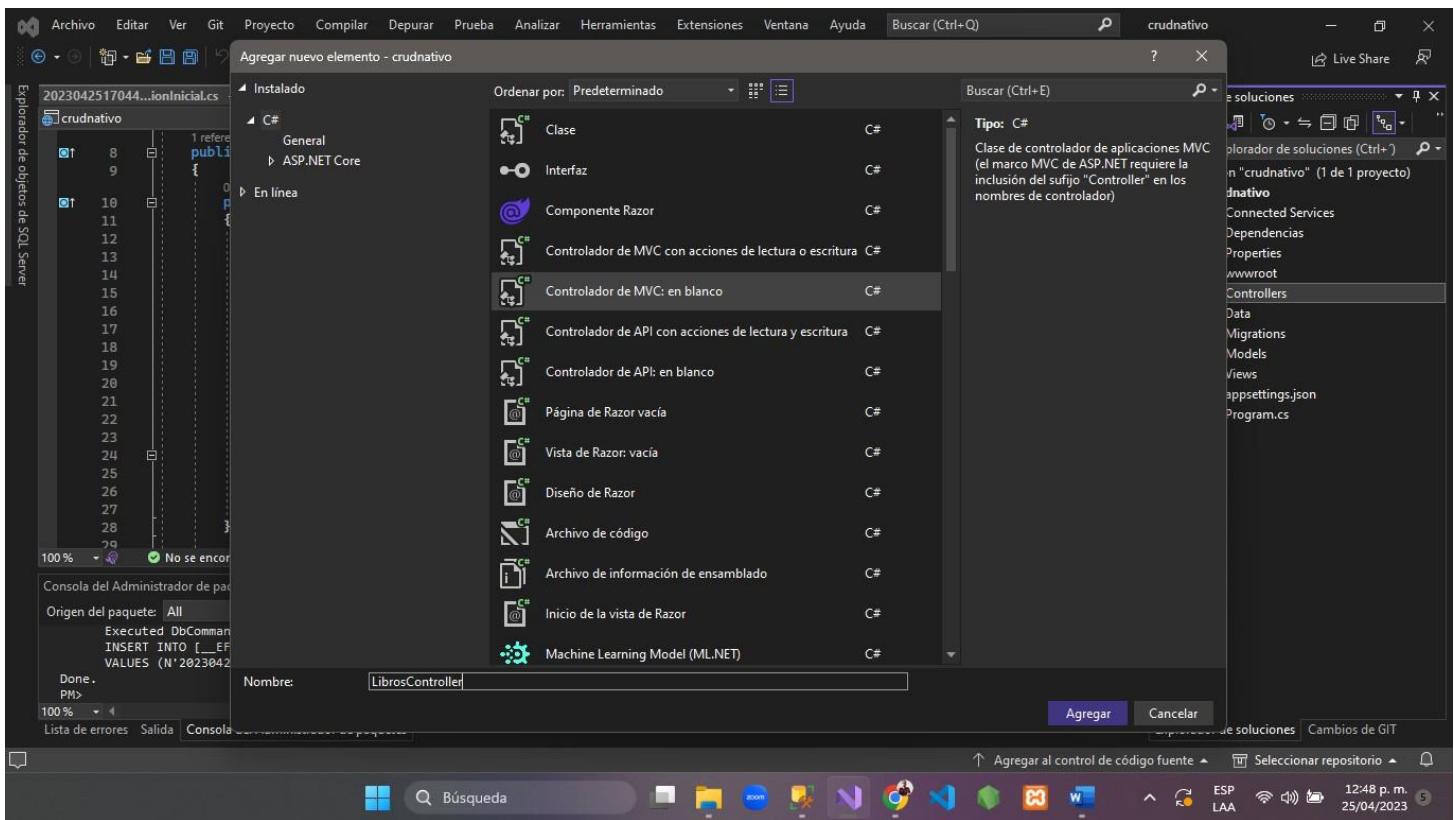
9. Vamos a instalar la librería entity framework core. Tools, luego hacemos la migración, vamos a herramientas administrador de paquetes nuget – consola y hacemos la migración con el siguiente comando add-migration migracionInicial se nos crea una carpeta migrations y ahí se crea un archivo migración con todas las tablas



10. Ahora vamos a actualizar la bd con update-database



11. Ahora vamos a crear el controlador. Dentro de la carpeta controlador clic derecho agregar controlador y seleccionamos controlador MVC en blanco lo llamamos LibrosController



The screenshot shows the Visual Studio IDE interface. The top menu bar includes Archivo, Editar, Ver, Git, Proyecto, Compilar, Depurar, Prueba, Analizar, Herramientas, Extensiones, Ventana, Ayuda, Buscar (Ctrl+Q), and crudnativo. The toolbar below has icons for file operations like New, Open, Save, Print, and others. The main window displays the code for LibrosController.cs:

```
namespace crudnativo.Controllers
{
    public class LibrosController : Controller
    {
        //invocamos para acceder a la base de datos
        private readonly ApplicationDbContext _context;

        //Creamos el constructor
        public LibrosController(ApplicationDbContext context)
        {
            _context = context;
        }

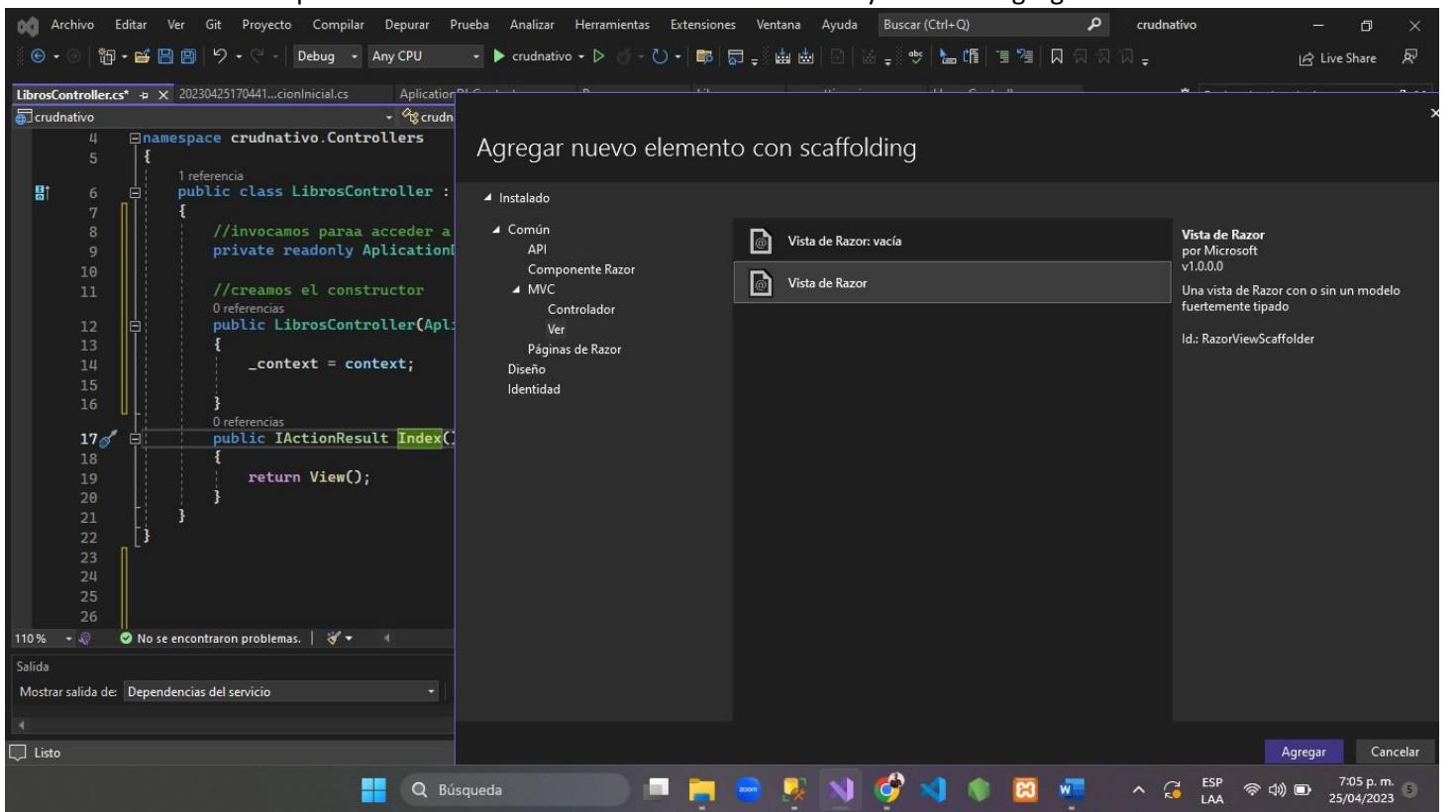
        //Mostramos los libros
        public IActionResult Index()
        {
            return View();
        }
    }
}
```

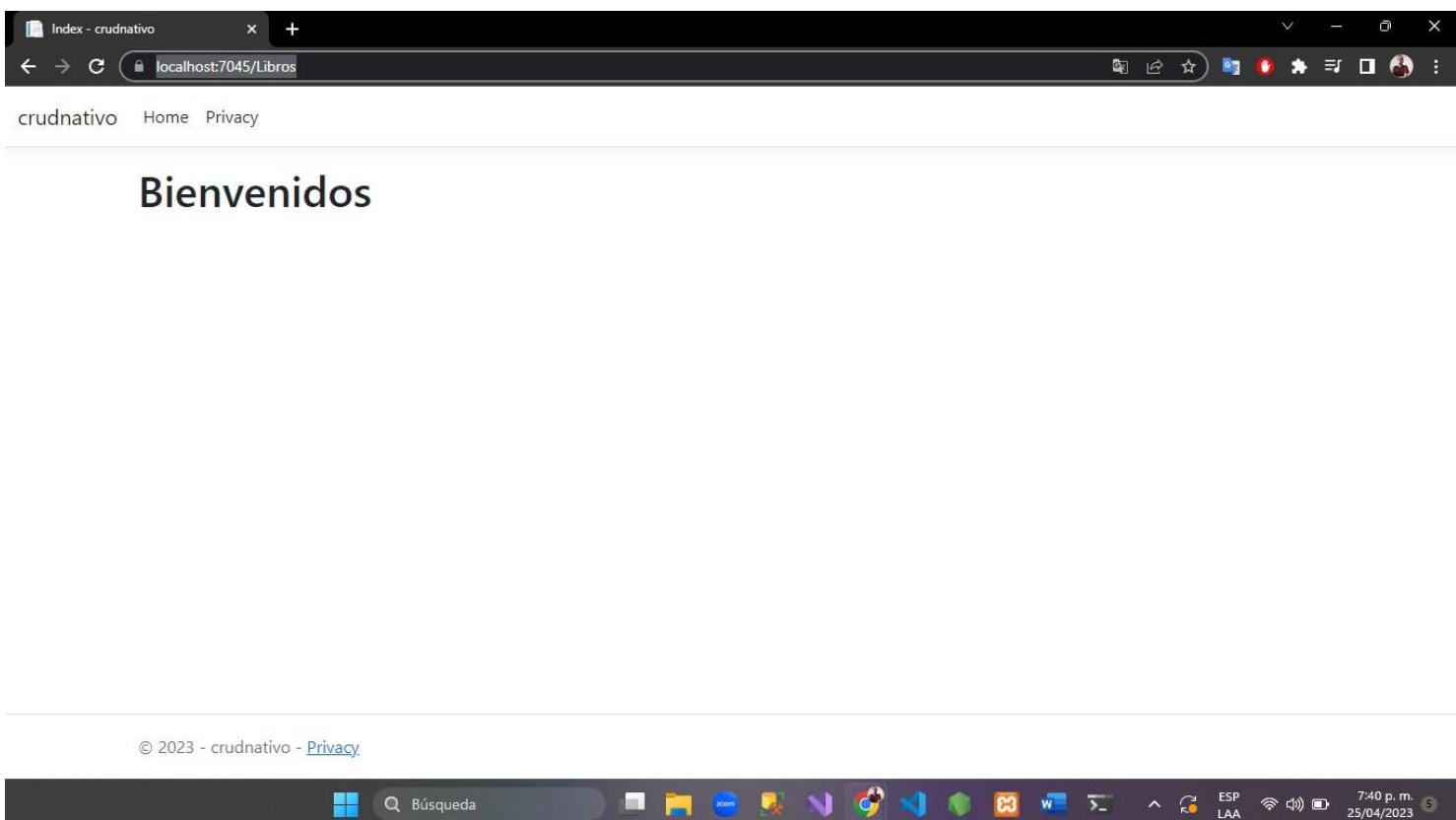
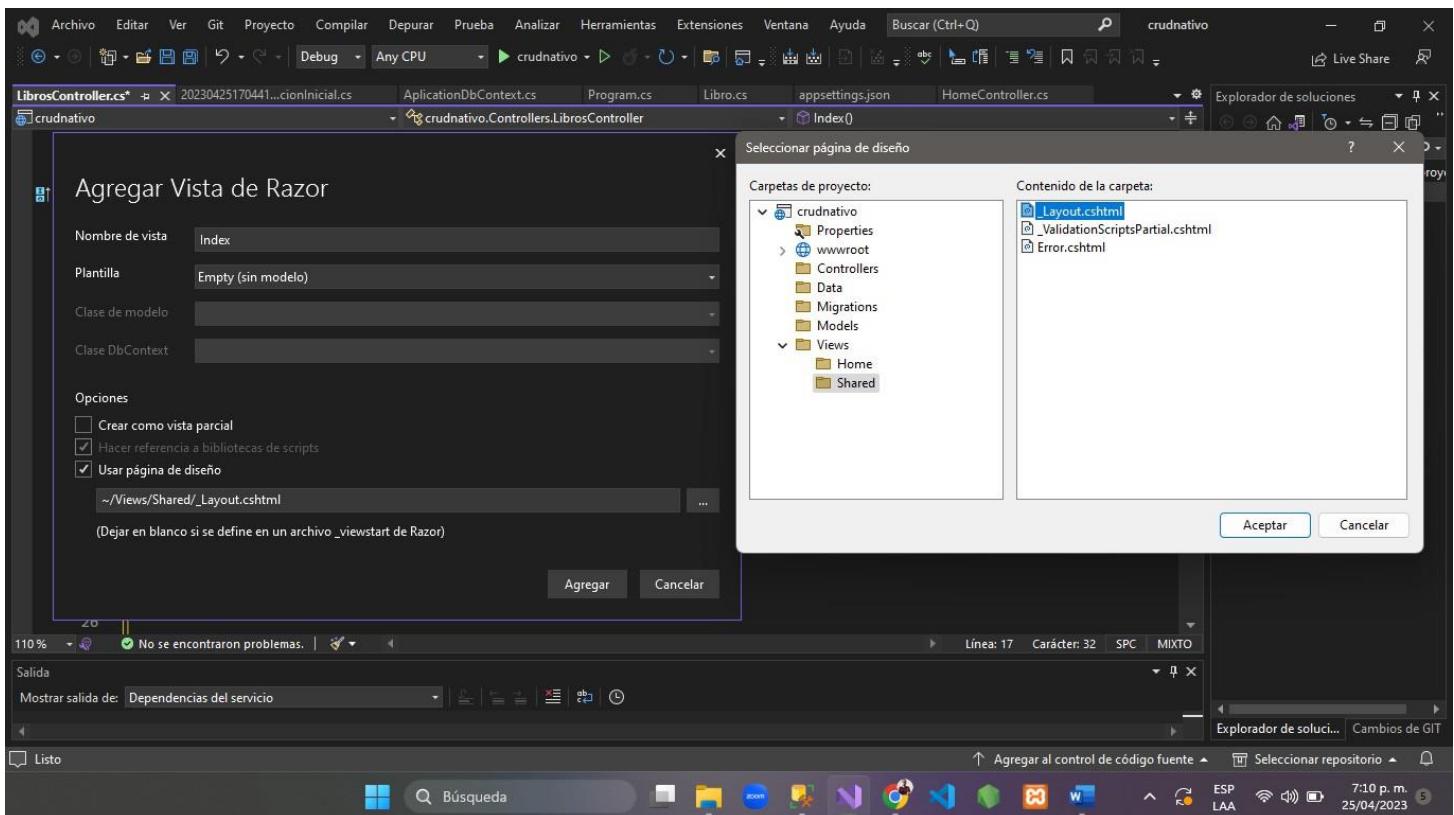
The code editor shows line numbers from 4 to 26. The status bar at the bottom indicates "Linea: 14" and "Carácter: 32". To the right is the Solution Explorer showing the project structure:

- Solución "crudnativo" (1 de 1 proyecto)
 - crudnativo
 - Connected Services
 - Dependencias
 - Properties
 - wwwroot
 - Controllers
 - HomeController.cs
 - LibrosController.cs
 - Data
 - Migrations
 - Models
 - Views
 - appsettings.json
 - Program.cs

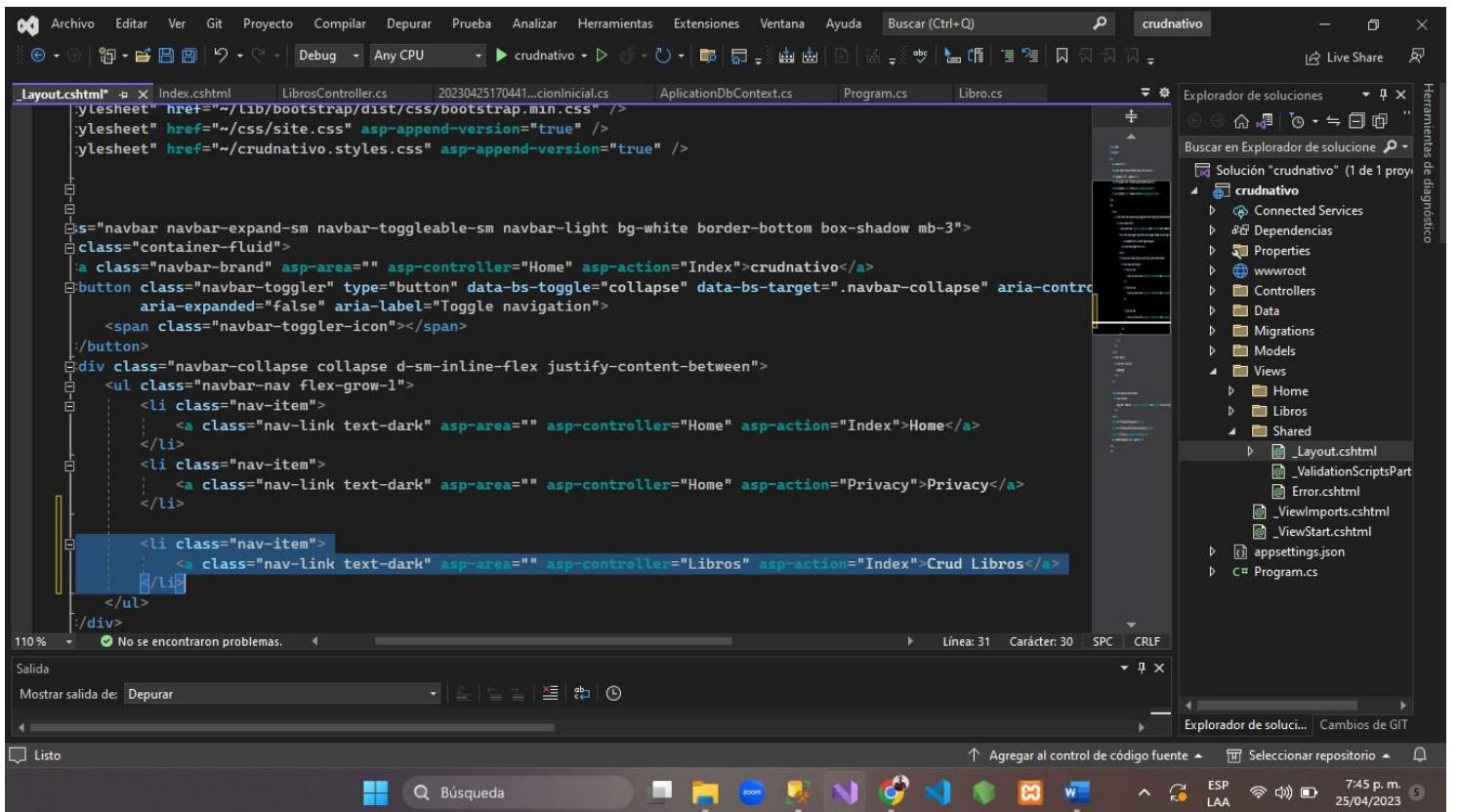
Below the Solution Explorer is the Task List, which is currently empty. The bottom right corner shows the date and time: 7/03 p.m. 25/04/2023.

12. Vamos a crear una vista para el index. Damos clic derecho sobre el index y le damos agregar vista:





13. Podemos crear un botón en el menú enseguida de Provacy, vamos views -shared - _layout. Y agregamos el código html para el botón



The screenshot shows the Visual Studio IDE interface. The top menu bar includes Archivo, Editar, Ver, Git, Proyecto, Compilar, Depurar, Prueba, Analizar, Herramientas, Extensiones, Ventana, Ayuda, and Buscar (Ctrl+Q). The title bar says "crudnativo". The left sidebar has tabs for Index.cshtml, LibrosController.cs, 20230425170441...cionInicial.cs, ApplicationDbContext.cs, Program.cs, and Libro.cs. The main code editor window displays the _Layout.cshtml file with the following code:

```
<head>
    <link href="/lib/bootstrap/dist/css/bootstrap.min.css" rel="stylesheet"/>
    <link href="/css/site.css" asp-append-version="true" />
    <link href="/css/site.css" asp-append-version="true" />

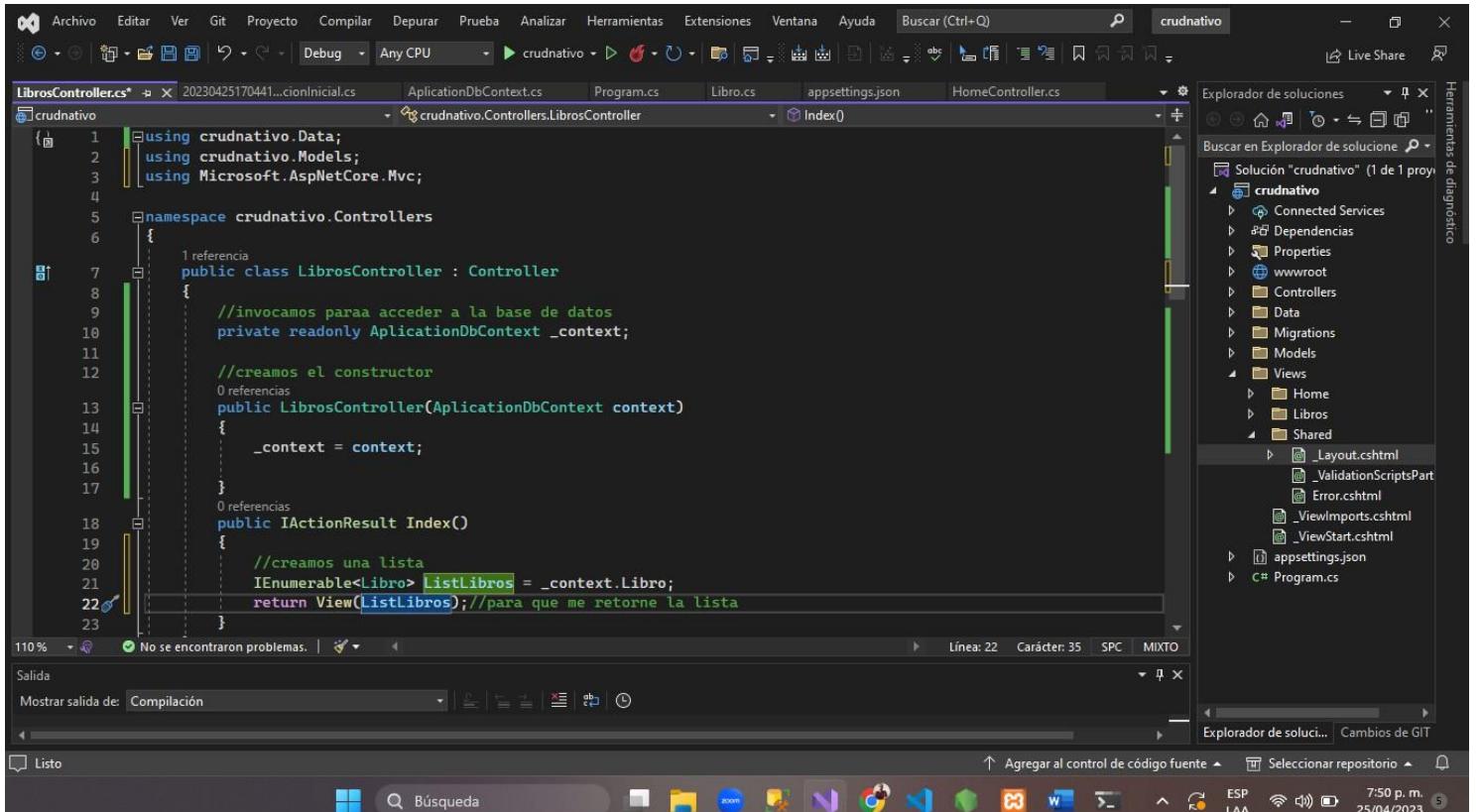
```

```
<s>=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">crudnativo</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" 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="Libros" asp-action="Index">Crud Libros</a>
            </li>
        </ul>
    </div>
</div>
```

The "Libros" menu item is highlighted in blue. The status bar at the bottom shows "Línea: 31" and "Carácter: 30". The bottom right corner shows "7:45 p. m." and "25/04/2023".

14. CREAMOS EL CONTROLADOR

Vamos al controlador libroController,

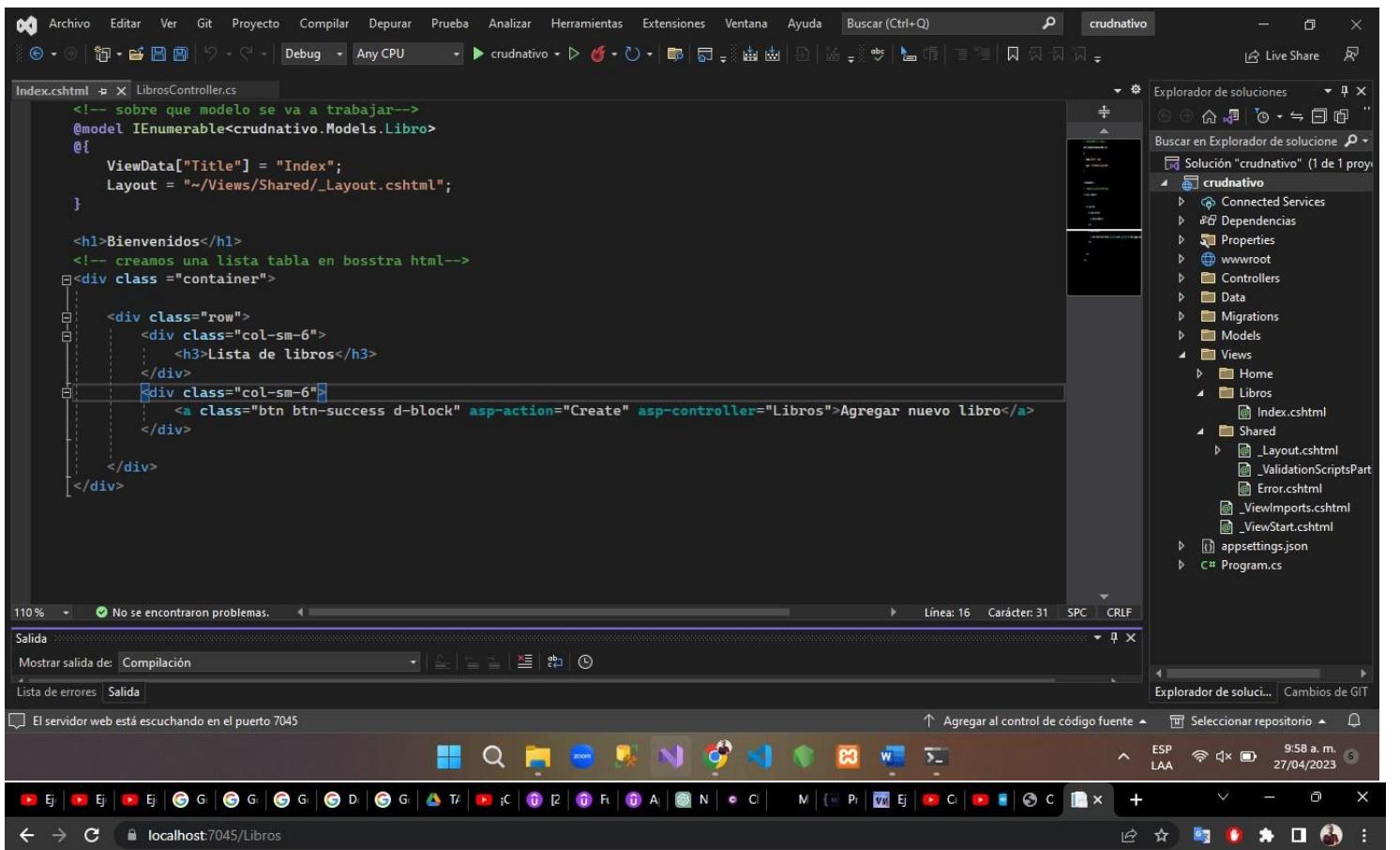


The screenshot shows the Visual Studio IDE interface. The top menu bar includes Archivo, Editar, Ver, Git, Proyecto, Compilar, Depurar, Prueba, Analizar, Herramientas, Extensiones, Ventana, Ayuda, and Buscar (Ctrl+Q). The title bar says "crudnativo". The left sidebar has tabs for LibrosController.cs, 20230425170441...cionInicial.cs, ApplicationDbContext.cs, Program.cs, Libro.cs, appsettings.json, and HomeController.cs. The main code editor window displays the LibrosController.cs file with the following code:

```
1  using crudnativo.Data;
2  using crudnativo.Models;
3  using Microsoft.AspNetCore.Mvc;
4
5  namespace crudnativo.Controllers
6  {
7      public class LibrosController : Controller
8      {
9          //invocamos para acceder a la base de datos
10         private readonly ApplicationDbContext _context;
11
12         //creamos el constructor
13         public LibrosController(ApplicationDbContext context)
14         {
15             _context = context;
16         }
17
18         //creamos una lista
19         public IActionResult Index()
20         {
21             IEnumerable<Libro> listLibros = _context.Libro;
22             return View(listLibros); //para que me retorne la lista
23         }
24     }
25 }
```

The status bar at the bottom shows "Línea: 22" and "Carácter: 35". The bottom right corner shows "7:45 p. m." and "25/04/2023".

15. Vamos a la vista y construimklos la vista con bosstrap para crear un botón q nos permitira agregar un nuevo libro



crudnativo Home Privacy Crud Libros

Bienvenidos

Lista de libros

Agregar nuevo libro

© 2023 - crudnativo - [Privacy](#)



16. Vamos a crear donde se carga los registro y hacemos validación para saber si hay registro o no.

```
<div class="row">
    <!-- para validar si hay registros-->
    @if (Model.Count() > 0)
    {
        <table class="table table-bordered table-striped">
            <thead>
                <tr>
                    <td>@Html.DisplayNameFor(m => m.Id)</td><!-- para acceder al titulo-->
                    <td>@Html.DisplayNameFor(m => m.Titulo)</td>
                    <td>@Html.DisplayNameFor(m => m.Descripcion)</td>
                    <td>@Html.DisplayNameFor(m => m.FechaLanzamiento)</td>
                    <td>@Html.DisplayNameFor(m => m.Autor)</td>
                    <td>@Html.DisplayNameFor(m => m.Precio)</td>
                    <td>Acciones</td>
                </tr>
            </thead>
            <tbody>
                <!-- para que se recorran los registros-->
                @foreach(var item in Model)
                {
                    <tr>
                        <td>@item.Id</td><!-- para acceder al titulo-->
                        <td>@item.Titulo</td>
                        <td>@item.Descripcion</td>
                        <td>@item.FechaLanzamiento</td>
                        <td>@item.Autor</td>
                        <td>@item.Precio</td>
                        <td>
                            <a asp-controller="Libros" asp-action="Edit" asp-route-id="@item.Id" class="btn btn-warning">Editar</a><!-- apunte al controlador-->
                            <a asp-controller="Libros" asp-action="Delete" asp-route-id="@item.Id" class="btn btn-warning">Eliminar</a><!-- apunte al controlador-->
                        </td>
                    </tr>
                }
            </tbody>
        </table>
    }

```

```
<div class="row">
    <!-- para validar si hay registros-->
    @if (Model.Count() > 0)
    {
        <table class="table table-bordered table-striped">
            <thead>
                <tr>
                    <td>@item.Id</td><!-- para acceder al titulo-->
                    <td>@item.Titulo</td>
                    <td>@item.Descripcion</td>
                    <td>@item.FechaLanzamiento</td>
                    <td>@item.Autor</td>
                    <td>@item.Precio</td>
                    <td>
                            <a asp-controller="Libros" asp-action="Edit" asp-route-id="@item.Id" class="btn btn-warning">Editar</a><!-- apunte al controlador-->
                            <a asp-controller="Libros" asp-action="Delete" asp-route-id="@item.Id" class="btn btn-warning">Eliminar</a><!-- apunte al controlador-->
                    </td>
                </tr>
            </thead>
            <tbody>
                <!-- para que se recorran los registros-->
                @foreach(var item in Model)
                {
                    <tr>
                        <td>@item.Id</td><!-- para acceder al titulo-->
                        <td>@item.Titulo</td>
                        <td>@item.Descripcion</td>
                        <td>@item.FechaLanzamiento</td>
                        <td>@item.Autor</td>
                        <td>@item.Precio</td>
                        <td>
                            <a asp-controller="Libros" asp-action="Edit" asp-route-id="@item.Id" class="btn btn-warning">Editar</a><!-- apunte al controlador-->
                            <a asp-controller="Libros" asp-action="Delete" asp-route-id="@item.Id" class="btn btn-warning">Eliminar</a><!-- apunte al controlador-->
                        </td>
                    </tr>
                }
            </tbody>
        </table>
    }
    else
    {
        <p>No existen registros</p>
    }

```

17. Vamos a crear un nuevo registro, vamos al controlador

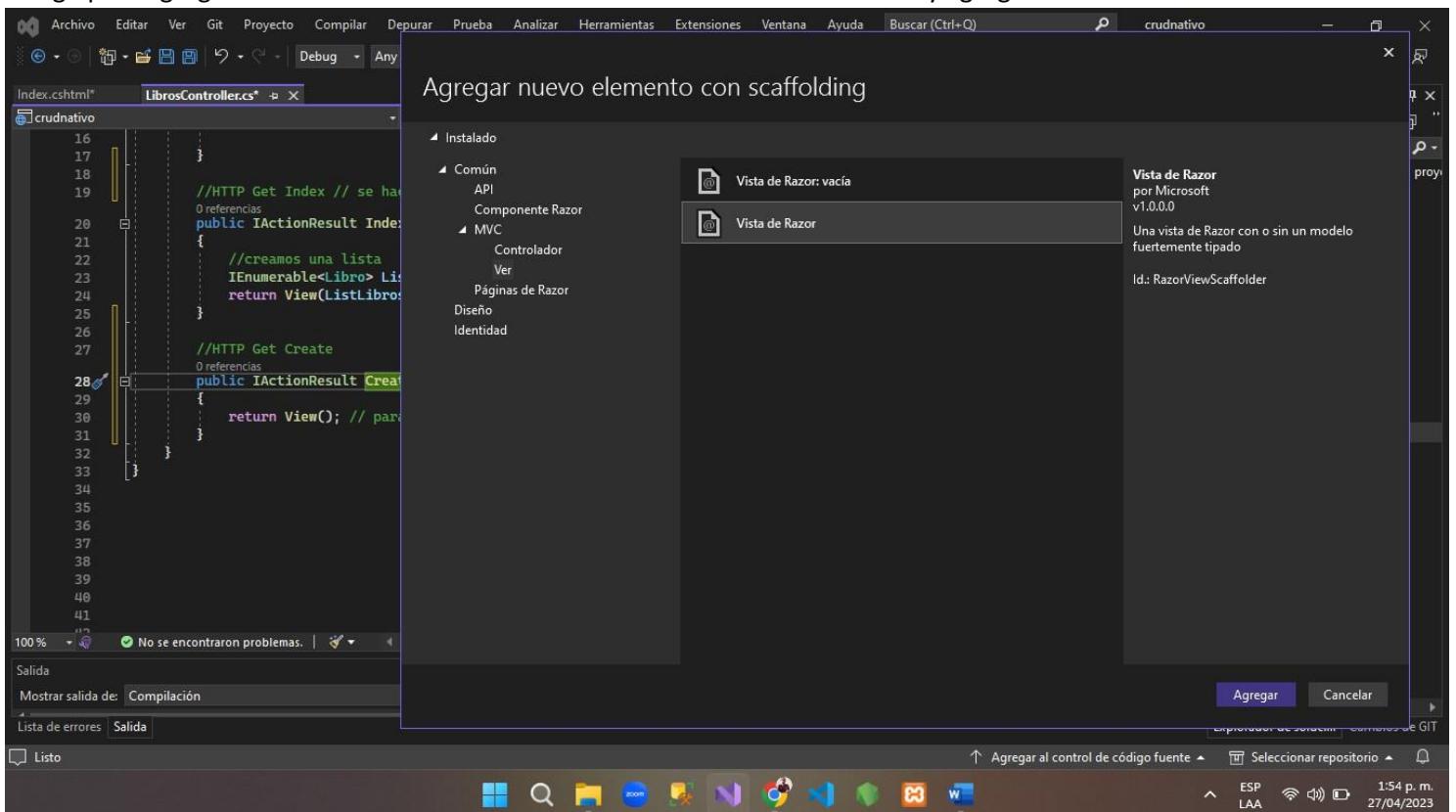
The screenshot shows the Visual Studio interface. In the top navigation bar, the project name "crudnativo" is selected. The code editor displays the file "LibrosController.cs" with the following code:

```
16
17
18
19     //HTTP Get Index // se hace peticion get y post
19     public IActionResult Index()
20     {
21         //Creamos una lista
22         IEnumerable<Libro> ListLibros = _context.Libro;
23         return View(ListLibros); //para que me retorne la lista
24     }
25
26
27     //HTTP Get Create
28     public IActionResult Create()
29     {
30         return View(); // para que me muestre el formulario
31     }
32
33 }
```

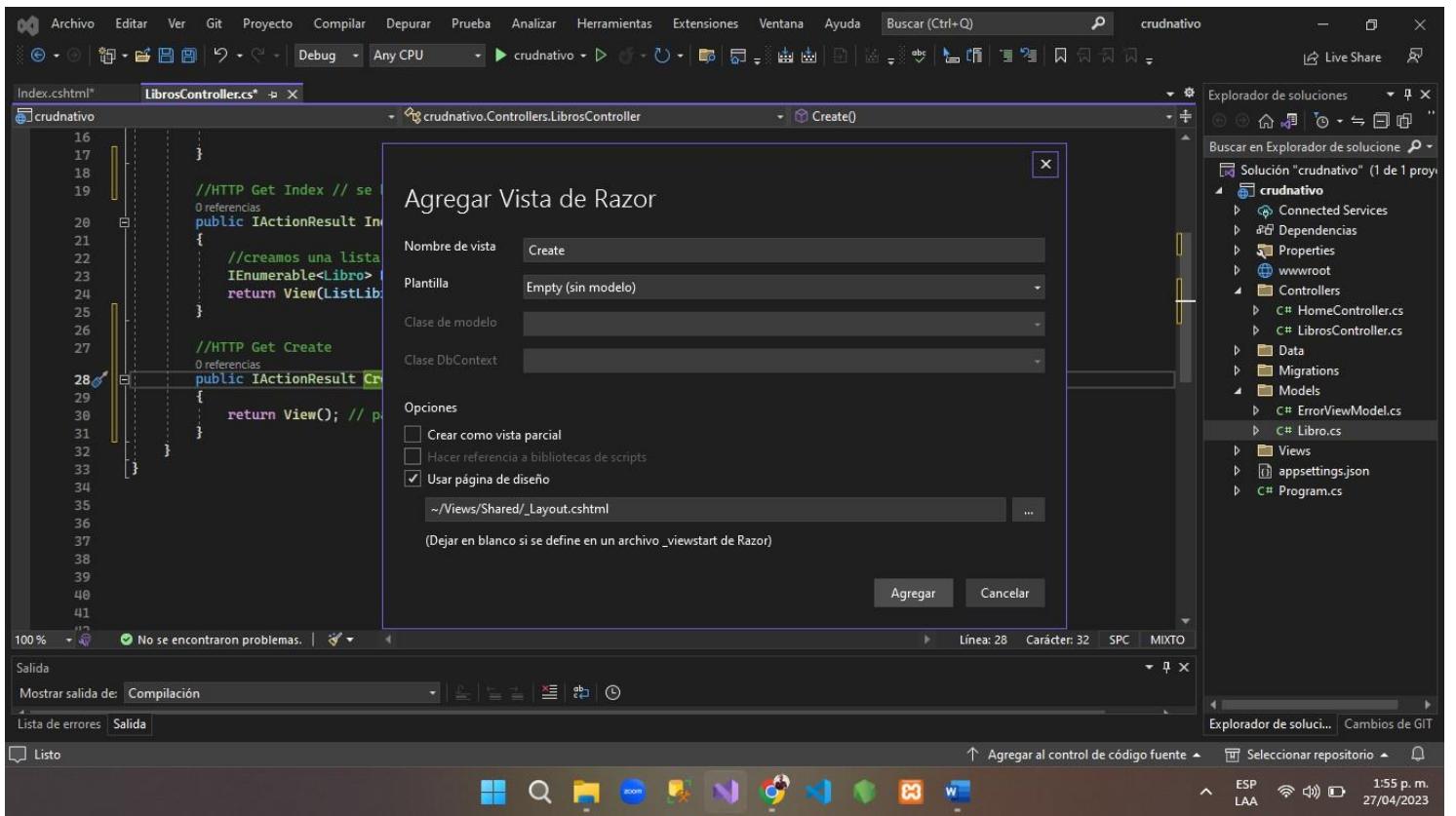
The Solution Explorer on the right shows the project structure:

- Solución "crudnativo" (1 de 1 proyecto)
 - crudnativo
 - Connected Services
 - Dependencias
 - Properties
 - wwwroot
 - Controllers
 - HomeController.cs
 - LibrosController.cs
 - Data
 - Migrations
 - Models
 - ErrorViewModel.cs
 - Libro.cs
 - Views
 - appsettings.json
 - Program.cs

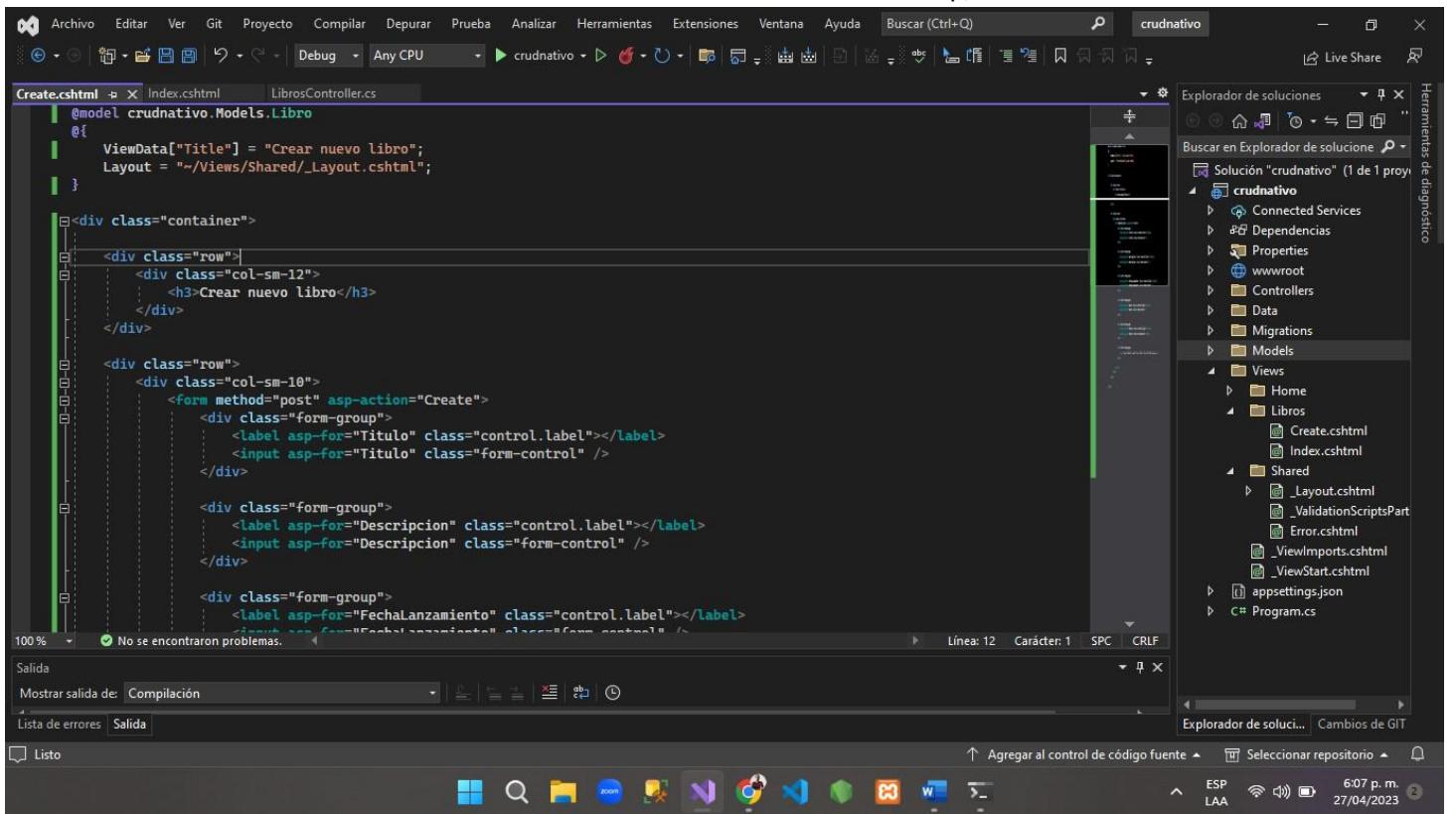
18. Luego para agregar la vista damos clic derecho sobre create – seleccionar vista y agregar razor vacia



19. Le damos agregar sin modelo, como se muestra en la imagen



20. Vamos a crear el formulario en esta nueva vista Libros se realiza en html bootstrap,



The screenshot shows the Microsoft Visual Studio interface. The top menu bar includes Archivo, Editar, Ver, Git, Proyecto, Compilar, Depurar, Prueba, Analizar, Herramientas, Extensiones, Ventana, Ayuda, and Buscar (Ctrl+Q). The title bar says "crudnativo". The left sidebar has "Create.cshtml" and "Index.cshtml" under "LibrosController.cs". The main code editor displays the following ASP.NET Core view code:

```
<div class="form-group">
    <label asp-for="FechaLanzamiento" class="control-label"></label>
    <input asp-for="FechaLanzamiento" class="form-control" />
</div>

<div class="form-group">
    <label asp-for="Autor" class="control-label"></label>
    <input asp-for="Autor" class="form-control" />
</div>

<div class="form-group">
    <label asp-for="Precio" class="control-label"></label>
    <input asp-for="Precio" class="form-control" /><br>
</div>

<div class="form-group">
    <input type="submit" value="Crear libro" class="btn btn-primary" />
</div>
</form>
</div>
</div>
```

The Solution Explorer on the right shows the project structure for "crudnativo":

- Solución "crudnativo" (1 de 1 proyecto)
 - Connected Services
 - Dependencias
 - Properties
 - wwwroot
 - Controllers
 - Data
 - Migrations
 - Models
 - Views
 - Home
 - Libros
 - Create.cshtml
 - Index.cshtml
 - Shared
 - _Layout.cshtml
 - _ValidationScriptsPartial.cshtml
 - Error.cshtml
 - _ViewImports.cshtml
 - _ViewStart.cshtml
 - appsettings.json
 - C# Program.cs

21. Vamos a crear un método quien recibe el create, para poder crear un nuevo libro libroController.

The screenshot shows the Microsoft Visual Studio interface. The top menu bar includes Archivo, Editar, Ver, Git, Proyecto, Compilar, Depurar, Prueba, Analizar, Herramientas, Extensiones, Ventana, Ayuda, and Buscar (Ctrl+Q). The title bar displays "crudnativo". The left sidebar shows the "Create.cshtml", "Index.cshtml", and "LibrosController.cs" files. The main code editor window contains the following C# code for the LibrosController:

```
22     //Creamos una lista
23     IEnumerable<Libro> ListLibros = _context.Libro;
24     return View(ListLibros); //para que me retorne la lista
25
26
27     //HTTP Get Create
28     public IActionResult Create()
29     {
30         return View(); // para que me muestre el formulario
31     }
32
33     //HTTP Get post
34     [HttpPost]
35     public IActionResult Create(Libro libro)//recibe un parametro libro-----
36     {
37         //validar el modelo
38         if (ModelState.IsValid)//valida el modelo
39         {
40             _context.Libro.Add(libro);//si el modelo es valido
41             _context.SaveChanges();//se guarda los cambios
42
43             return RedirectToAction("index"); // para retornal al index
44         }
45
46         return View();
47     }

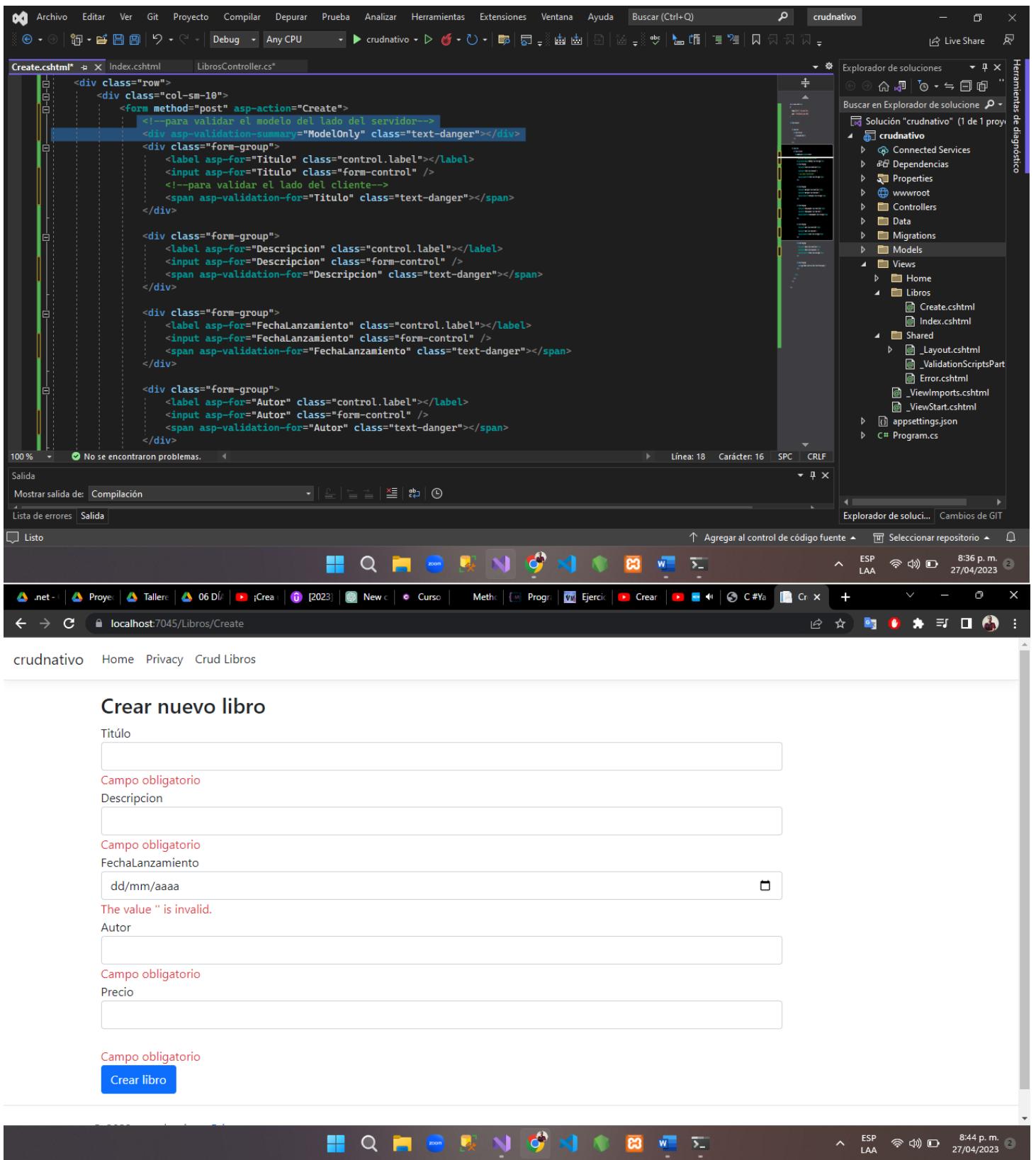
```

The Solution Explorer on the right lists the project structure:

- Solución "crudnativo" (1 de 1 proyecto)
 - crudnativo
 - Connected Services
 - Dependencias
 - Properties
 - wwwroot
 - Controllers
 - Data
 - ApplicationDbContext.cs
 - Migrations
 - Models
 - ErrorViewModel.cs
 - Libro.cs
 - Views
 - Home
 - Libros
 - Create.cshtml
 - Index.cshtml
 - Shared
 - _Layout.cshtml
 - _ValidationScriptsPartial.cshtml
 - Error.cshtml
 - _ViewImports.cshtml
 - _ViewStart.cshtml
 - appsettings.json
 - Program.cs

The status bar at the bottom shows "Línea: 22, Carácter: 10, SPC, MIXTO".

22. Vamos nuevamente a la vista créate y agregamos las validaciones



23. Creamos una validación del lado del cliente, vamos a vista y agregamos un javascript para llamar la vista parcial vamos a views shared – ValidationScript

The screenshot shows the Microsoft Visual Studio IDE interface. In the top navigation bar, the following items are visible: Archivo, Editar, Ver, Git, Proyecto, Compilar, Depurar, Prueba, Analizar, Herramientas, Extensiones, Ventana, Ayuda, and Buscar (Ctrl+Q). The title bar displays the project name "crudnativo".

The main code editor window contains the file "Create.cshtml" with the following code:

```
<!-- creamos el formulario -->
<form>
    <div>
        <!-- creamos el script para validar -->
        @section Scripts{
            <!-- llamamos una vista parcial -->
            @if <partial name="_ValidationScriptsPartial" />{ }
        }
    </div>
</form>
```

The "Explorador de soluciones" (Solution Explorer) panel on the right shows the project structure for "crudnativo".

```
Explorador de soluciones
Solución "crudnativo" (1 de 1 proyecto)
crudnativo
    Connected Services
    Dependencias
    Properties
    wwwroot
    Controllers
    Data
    Migrations
    Models
    Views
        Home
        Libros
        Shared
            _Layout.cshtml
            _ValidationScriptsPartial.cshtml
            Error.cshtml
            _ViewImports.cshtml
            _ViewStart.cshtml
    appsettings.json
    Program.cs
```

The "Salida" (Output) window shows the message "No se encontraron problemas." (No problems found).

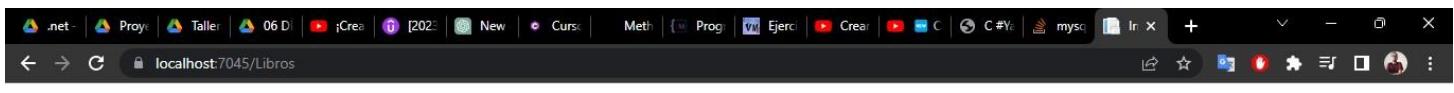
The browser taskbar at the bottom shows the URL "localhost:7045/Libros/Create".

The browser window displays the "Crear nuevo libro" (Create new book) form. The fields are:

- Titulo: Campo obligatorio (Title: Required field)
- Descripcion: Campo obligatorio (Description: Required field)
- FechaLanzamiento: dd/mm/aaaa (Release Date: dd/mm/aaaa)
- Autor: Campo obligatorio (Author: Required field)
- Precio: Campo obligatorio (Price: Required field)

A blue "Crear libro" (Create book) button is located at the bottom of the form.

24. Vamos creamos un nuevo libro en el formulario



crudnativo Home Privacy Crud Libros

Bienvenidos

Lista de libros

Id	Titulo	Descripcion	Fecha de lanzamiento	Autor	Precio	Acciones
1	prueba	probando codigo	21/04/2023 10:13:00 p. m.	admin	5000	<button>Editar</button> <button>Eliminar</button>
2	prueba	probando codigo	21/04/2023 10:13:00 p. m.	admin	5000	<button>Editar</button> <button>Eliminar</button>
3	libro1	asbfajfbjk	14/04/2023 10:27:00 p. m.	administradoe	499999	<button>Editar</button> <button>Eliminar</button>
4	pruebamas	seguir probando	4/04/2023 12:00:00 a. m.	administrador	8900000	<button>Editar</button> <button>Eliminar</button>

© 2023 - crudnativo - [Privacy](#)



25. Podemos agregar una variable mensaje para que aparezca después de registrar un dato

```
25
26
27
28     //HTTP Get Create
29     public IActionResult Create()
30     {
31         return View(); // para que me muestre el formulario
32     }
33
34     //HTTP Get post
35     [HttpPost]
36     public IActionResult Create(Libro libro)//recibe un parametro libro-----
37     {
38         //validar el modelo
39         if (ModelState.IsValid)//valida el modelo
40         {
41             _context.Libro.Add(libro);//si el modelo es valido
42             _context.SaveChanges();//se guarda los cambios
43
44             TempData["mensaje"] = "Libro registrado con exito";// se crea una variable
45
46             return RedirectToAction("index"); // para retornal al index
47
48         }
49     }
50 }
```

The code in the screenshot shows the implementation of the `Create` action method in the `LibrosController.cs`. It handles both GET and POST requests. For a GET request, it returns the `Create` view. For a POST request, it validates the `Libro` model. If valid, it adds the book to the database using `_context.Libro.Add(libro)` and saves changes with `_context.SaveChanges()`. It then sets a `TempData` message `TempData["mensaje"] = "Libro registrado con exito"` and redirects to the `index` action.

Luego nos dirigimos al index para agregar una condición

Screenshot of the Visual Studio IDE showing the code editor, solution explorer, and browser interface for a .NET application named "crudnativo".

Code Editor:

```

<h1>Bienvenidos</h1>
<!-- creamos una lista tabla en bosstra html-->
<div class="container">
    <!-- para mostrar un mensaje -->
    <if ( TempData["mensaje"] != null )>
        <div class="alert alert-warning alert-dismissible fade show" role="alert">
            @TempData["mensaje"]
            <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                <span aria-hidden="true">&times;Editar</a> <a href="#" class="text-danger" data-id="1" data-action="Delete">Eliminar</a></td>
                </tr>
                <tr>
                    <td>2</td>
                    <td>prueba</td>
                    <td>probando codigo</td>
                    <td>21/04/2023 10:13:00 p. m.</td>
                    <td>admin</td>
                    <td>5000</td>
                    <td><a href="#" class="text-warning" data-id="2" data-action="Edit">Editar</a> <a href="#" class="text-danger" data-id="2" data-action="Delete">Eliminar</a></td>
                </tr>
                <tr>
                    <td>3</td>
                    <td>libro1</td>
                    <td>asbfajfbjk</td>
                    <td>14/04/2023 10:27:00 p. m.</td>
                    <td>administradore</td>
                    <td>499999</td>
                    <td><a href="#" class="text-warning" data-id="3" data-action="Edit">Editar</a> <a href="#" class="text-danger" data-id="3" data-action="Delete">Eliminar</a></td>
                </tr>
                <tr>
                    <td>4</td>
                    <td>pruebamas</td>
                    <td>seguir probando</td>
                    <td>4/04/2023 12:00:00 a. m.</td>
                    <td>administrador</td>
                    <td>8900000</td>
                    <td><a href="#" class="text-warning" data-id="4" data-action="Edit">Editar</a> <a href="#" class="text-danger" data-id="4" data-action="Delete">Eliminar</a></td>
                </tr>
                <tr>
                    <td>5</td>
                    <td>libro3</td>
                    <td>libro3</td>
                    <td>20/04/2023 12:00:00 a. m.</td>
                    <td>carlos</td>
                    <td>4567</td>
                    <td><a href="#" class="text-warning" data-id="5" data-action="Edit">Editar</a> <a href="#" class="text-danger" data-id="5" data-action="Delete">Eliminar</a></td>
                </tr>
                <tr>
                    <td>6</td>
                    <td>libro 5</td>
                    <td>tomo 5</td>
                    <td>3/04/2023 12:00:00 a. m.</td>
                    <td>juan</td>
                    <td>64738</td>
                    <td><a href="#" class="text-warning" data-id="6" data-action="Edit">Editar</a> <a href="#" class="text-danger" data-id="6" data-action="Delete">Eliminar</a></td>
                </tr>
                <tr>
                    <td>7</td>
                    <td>nada</td>
                    <td>iqual</td>
                    <td>12/04/2023 12:00:00 a. m.</td>
                    <td>admin</td>
                    <td>45678</td>
                    <td><a href="#" class="text-warning" data-id="7" data-action="Edit">Editar</a> <a href="#" class="text-danger" data-id="7" data-action="Delete">Eliminar</a></td>
                </tr>
            </tbody>
        </table>
    </if>
</div>

```

Solution Explorer:

- Solución "crudnativo" (1 de 1 proyecto)
 - Connected Services
 - Dependencias
 - Properties
 - wwwroot
 - Controllers
 - Data
 - Migrations
 - Models
 - C# ErrorViewModel.cs
 - C# Libro.cs
 - Views
 - appsettings.json
 - C# Program.cs

Browser:

localhost:7045/Libros

The browser shows the application's home page with the title "Bienvenidos". A success message "Libro registrado con éxito" is displayed. The "Lista de libros" table lists seven books with columns: Id, Título, Descripción, Fecha de lanzamiento, Autor, Precio, and Acciones (Edit and Delete buttons). The "Agregar nuevo libro" button is visible at the top right.

26. Vamos a EDITAR el registro, como tenemos un Id vamos al controlador y copiamos el método **HTTPf1 Get Create y lo cambiamos por Edit**

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the `LibrosController.cs` file under the `crudnativo` project. The code implements methods for creating and editing books. The `Index.cshtml` and `Create.cshtml` files are also visible in the solution. The Solution Explorer on the right shows the project structure with files like `HomeController.cs`, `LibrosController.cs`, `appsettings.json`, and `Program.cs`. The status bar at the bottom indicates the current time as 7:45 p.m. on 30/04/2023.

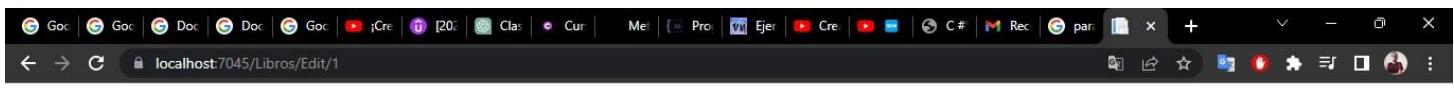
```
43     TempData["mensaje"] = "Libro registrado con éxito"; // se crea una variable
44
45     return RedirectToAction("Index"); // para retornar al index
46
47 }
48
49     return View();
50 }
51 //HTTP Get Edit
52 public IActionResult Edit(int? id)//recibe un entero que puede ser null
53 {
54     if (id == null || id == 0)
55     {
56         return NotFound(); //cuando no encuentra la pagina o error 404
57     }
58
59     //obtener el libro
60     var libro = _context.Libro.Find(id); //utilizamos el contexto con el metodo find para allar el id
61     if(libro == null)
62     {
63         return NotFound();
64     }
65     return View(libro); // para que me muestre el formulario
66 }
67 }
68 }
```

27. Vamos a crear la vista podemos copiar sobre créate y pegar y cambiamos el nombre

The screenshot shows the Visual Studio IDE interface with the following details:

- Top Bar:** Archivo, Editar, Ver, Git, Proyecto, Compilar, Depurar, Prueba, Analizar, Herramientas, Extensiones, Ventana, Ayuda, Buscar (Ctrl+Q).
- Solution Explorer:** Shows the solution "crudnativo" with one project "crudnativo". The "Libros" folder under "Views" contains "Create.cshtml", "Edit.cshtml", and "Index.cshtml". Other files include "Home", "Shared", "ViewImports.cshtml", "ViewStart.cshtml", "appsettings.json", and "Program.cs".
- Code Editor:** The "Edit.cshtml" file is open, displaying the code for creating a new book. It includes a title section, a form with validation for the "Titulo" field, and another group for the "Descripcion" field.
- Status Bar:** Shows "100%" completion, "No se encontraron problemas.", Linea: 1, Carácter: 1, SPC, CRLF.
- Bottom Navigation:** Includes "Explorador de soluciones", "Cambiros de GIT", "Listo", "Salida", and various system icons.

Modificamos el viewData por Editar, también remplazamos créate por edit



crudnativo Home Privacy Crud Libros

Editar libro

Titúlo

prueba

Descripción

probando código

Fecha de lanzamiento

21/04/2023



Autor

admin

Precio

5000

Editar libro

© 2023 - crudnativo - [Privacy](#)

ESP LAA 7:56 p. m. 30/04/2023

28. Necesitamos el método que reciba al dar clic y realice la operación. Vamos a libros controller copiamos y pegamos el Http get post

```
65     } // para que me muestre el formulario
66
67     //HTTP Get post Edit
68     [HttpPost]
69     public IActionResult Edit(Libro libro) //recibe un parametro libro-----
70     {
71         //validar el modelo
72         if (ModelState.IsValid)//valida el modelo
73         {
74             _context.Libro.Update(libro); //si el modelo es valido colocamos Update
75             _context.SaveChanges(); //se guarda los cambios
76
77             TempData["mensaje"] = "Libro actualizado con éxito"; // se crea una variable
78
79             return RedirectToAction("Index"); // para retornal al index
80         }
81
82         return View();
83     }
84 }
```

Explorador de soluciones

- Solución "crudnativo" (1 de 1 proyecto)
 - crudnativo
 - Connected Services
 - Dependencias
 - Properties
 - wwwroot
 - Controllers
 - Data
 - Migrations
 - Models
 - Views
 - Home
 - Libros
 - Create.cshtml
 - Edit.cshtml
 - Index.cshtml
- Shared
 - _ViewImports.cshtml
 - _ViewStart.cshtml

- appsettings.json
- C# Program.cs

Salida

Mostrar salida de: Compilación

Lista de errores Salida

Lista de errores

Agregar al control de código fuente Seleccionar repositorio

8:03 p. m. 30/04/2023

29. Creamos la función ELIMINAR. , Creamos el delete en el controlador, copiamos el `//HTTPf1 Get Edit` que vamos a utilizar para el delete.

```

84 }
85 }
86 }
87 //HTTP Get Delete
88 public IActionResult Delete(int? id) //recibe un entero que puede ser null
89 {
90     if (id == null || id == 0)
91     {
92         return NotFound(); //cuando no encuentra la pagina o error 404
93     }
94 
95     //obtener el libro
96     var libro = _context.Libro.Find(id); //utilizamos el contexto con el metodo find para allar el id
97     if (libro == null)
98     {
99         return NotFound();
100    }
101    return View(libro); // para que me muestre el formulario
102 }
103 }
104 }
105 }
106 }
107 }
108 }
109 }
110 }

```

100% No se encontraron problemas. | Línea: 87 Carácter: 36 SPC CRLF

Salida

Mostrar salida de: Compilación

Lista de errores Salida

Explorador de soluciones Cambios de GIT

Agregar al control de código fuente Seleccionar repositorio

ESP LAA 8:09 p. m. 30/04/2023

30. Para crear la vista podemos copiar el edit y pegar y remplazamos el nombre por delete, y remplazamos en la vista el editar por eliminar, agregamos disabled, para no poder editar el campo

```

ViewData["title"] = "Eliminar libro";
Layout = "~/Views/Shared/_Layout.cshtml";
}

<div class="container">
    <div class="row">
        <div class="col-sm-12">
            <h3>Eliminar libro</h3>
        </div>
    </div>

    <div class="row">
        <div class="col-sm-10">
            <!--cambiamos el nombre del metodo-->
            <form method="post" asp-action="DeleteLibro">

                <!--mandamos internamente como campo oculto el id-->
                <input type="hidden" asp-for="Id" />

                <!--para validar el modelo del lado del servidor-->
                <div asp-validation-summary="ModelOnly" class="text-danger"></div>
                <div class="form-group">
                    <label asp-for="Titulo" class="control-label"></label>
                    <input asp-for="Titulo" class="form-control" disabled />
                    <!--para validar el lado del cliente-->
                    <span asp-validation-for="Titulo" class="text-danger"></span>
                </div>
            </form>
        </div>
    </div>
</div>

```

100% No se encontraron problemas. | Línea: 27 Carácter: 65 SPC CRLF

Salida

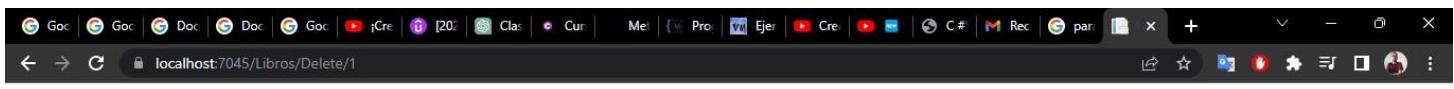
Mostrar salida de: Compilación

Lista de errores Salida

Explorador de soluciones Cambios de GIT

Agregar al control de código fuente Seleccionar repositorio

ESP LAA 8:32 p. m. 30/04/2023



crudnativo Home Privacy Crud Libros

Eliminar libro

Titulo

prueba098765

Descripcion

probando codigo

Fecha de lanzamiento

21/04/2023

Autor

admin12345

Precio

2345678

Eliminar libro

© 2023 - crudnativo - [Privacy](#)

31. Ahora vamos a darle funcionalidad en el botón eliminar “libro”, vamos al controlador y copiamos el método http post edit.

```
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
```

```
        return View(libro); // para que me muestre el formulario
    }

    //HTTP post Delete
    [HttpPost]
    [ValidateAntiForgeryToken]// en los post una proteccion en los formularios para evitar bot
    public IActionResult DeleteLibro(int? id)//recibe un parametro libro-----
    {
        //obtener el libro por id
        var libro = _context.Libro.Find(id);

        if (libro == null)
        {

            return NotFound();
        }

        _context.Libro.Remove(libro);//si el modelo es valido/ colocamos Update
        _context.SaveChanges(); //se guarda los cambios
        TempData["mensaje"] = "Libro eliminado con exito";// se crea una variable
        return RedirectToAction("Index"); // para retornar al index
    }
}
```

The screenshot shows the Visual Studio interface with the code editor open to the `Delete.cshtml` file. The code implements a POST method for deleting a book. It first checks if the book exists. If it does, it removes it from the database and saves changes. A success message is stored in the TempData dictionary and the user is redirected back to the index page. The Solution Explorer on the right shows the project structure, including controllers like `LibrosController.cs` and views like `Create.cshtml`, `Delete.cshtml`, `Edit.cshtml`, and `Index.cshtml`.

