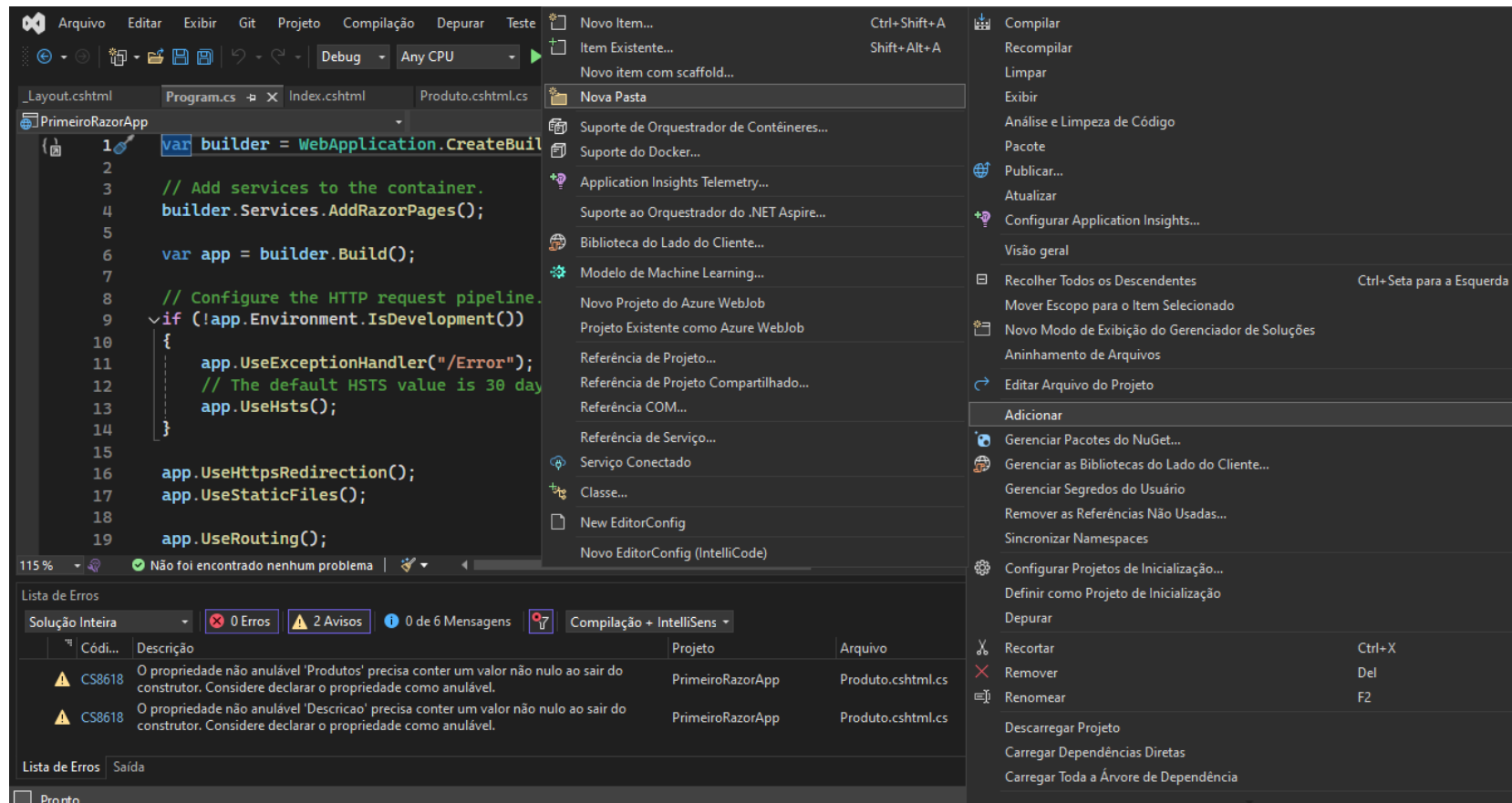


Programação .NET

Professor Ricardo Frohlich da Silva

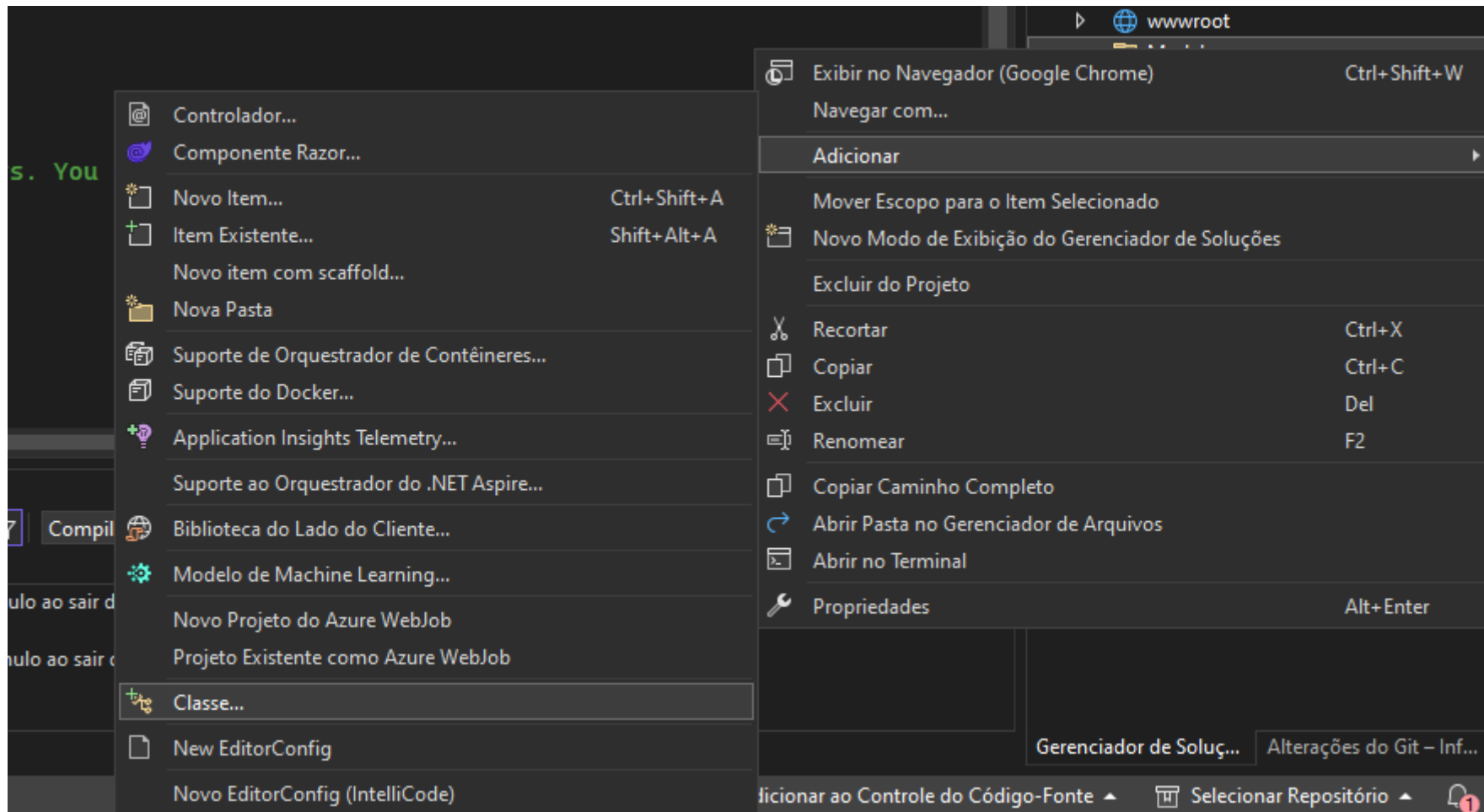
Criando um cadastro de Usuario

- Crie uma pasta chamada Models



Criando um cadastro de Usuario

- Crie uma classe chamada Usuario dentro da pasta Models



Criando um cadastro de Usuario

- Crie uma classe chamada Usuario dentro da pasta Models

```
public class Usuario
{
    public int Id { get; set; } // Identificador único
    public string Nome { get; set; }
    public string Senha { get; set; }
    public string Email { get; set; }
}
```

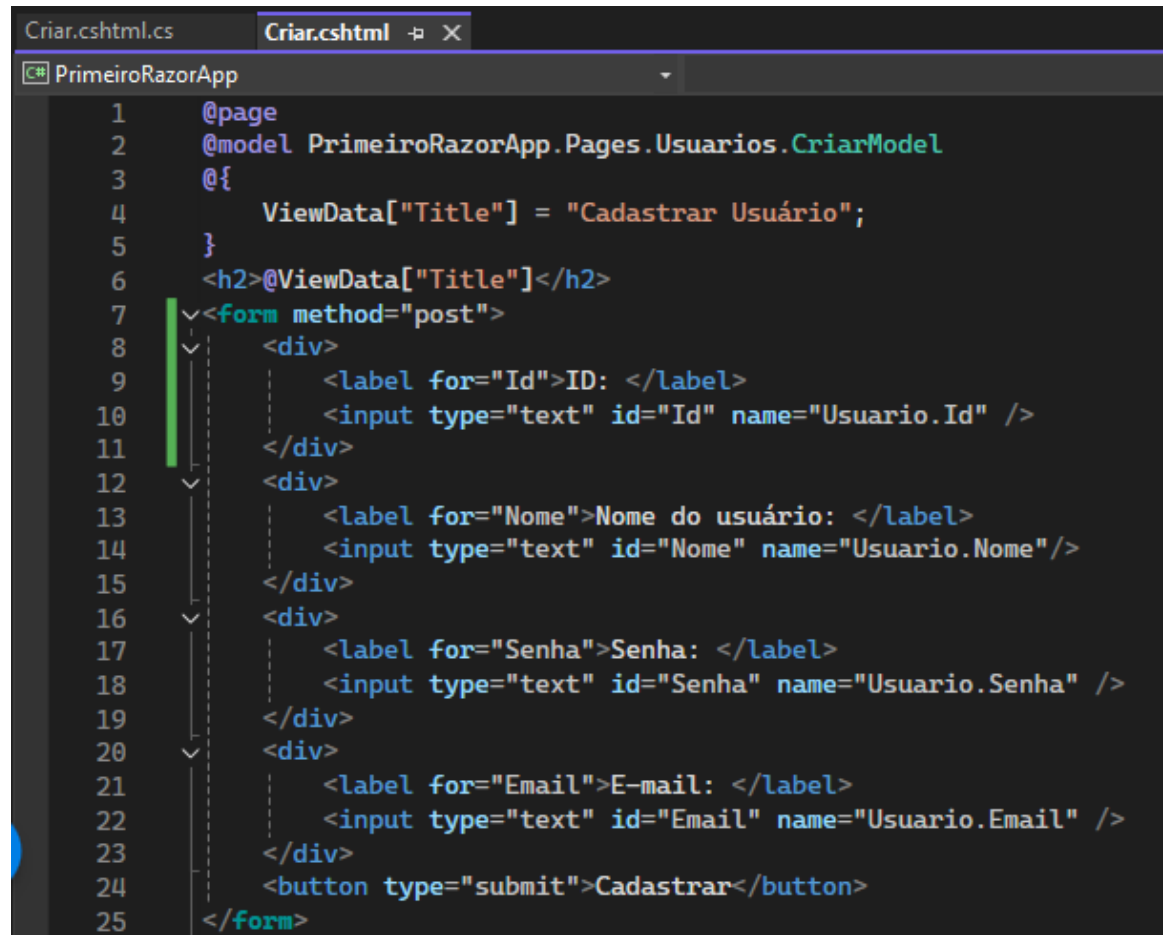
Criando um Usuário

```
Criar.cshtml.cs  Criar.cshtml
PrimeiroRazorApp  PrimeiroRazorApp.Pages.Usuarios.CriarModel  usuario

7 public class CriarModel : PageModel
8 {
9     [BindProperty] // vincula os dados do formulário HTML às propriedades do modelo da página
10    //quando eu envio um formulário pelo método POST, o ASP.NET verifica os valores e campos
11    public Usuario usuario { get; set; } //Um objeto pra ser instanciado e armazenado no arquivo
12    public void OnGet() // método que executa quando é chamado o método Get, ou seja, quando acesso a página
13    {
14    }
15    public IActionResult OnPost() // método que executa quando é feito o método Post
16    {
17        //Verifica se os dados enviados no formulário são válidos conforme as regras do modelo
18        if (!ModelState.IsValid)
19        {
20            return Page();
21        }
22        else
23        {
24            //Vou fazer o armazenamento destes dados dentro do meu arquivo texto
25            using (var writer = new StreamWriter("usuarios.txt", true))
26            {
27                writer.WriteLine(usuario.Id+";"+usuario.Nome + ";" + usuario.Senha + ";" + usuario.Email);
28                return RedirectToPage("/Usuarios/Index");
29            }
30        }
31    }
32 }
33 }
```

Criando um Usuário

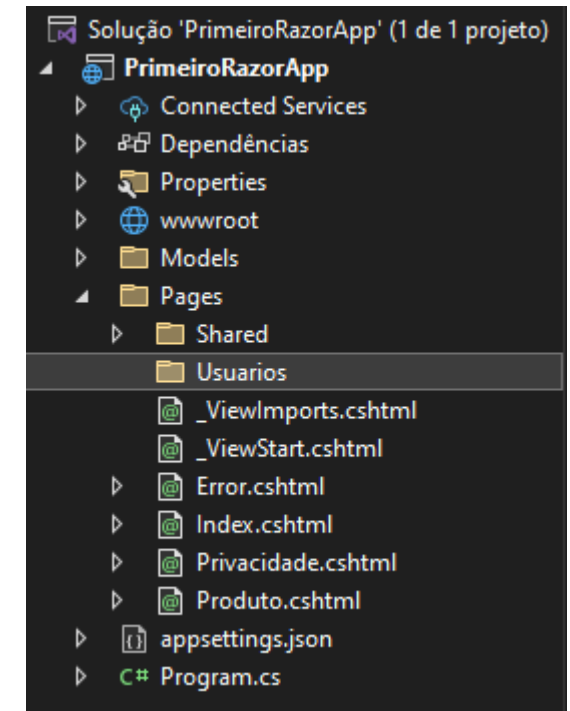
- Para detalhar e editar um cadastro, nós precisaremos de um identificador único



```
1 @page
2 @model PrimeiroRazorApp.Pages.Usuarios.CriarModel
3 @{
4     ViewData["Title"] = "Cadastrar Usuário";
5 }
6 <h2>@ViewData["Title"]</h2>
7 <form method="post">
8     <div>
9         <label for="Id">ID: </label>
10        <input type="text" id="Id" name="Usuario.Id" />
11    </div>
12    <div>
13        <label for="Nome">Nome do usuário: </label>
14        <input type="text" id="Nome" name="Usuario.Nome"/>
15    </div>
16    <div>
17        <label for="Senha">Senha: </label>
18        <input type="text" id="Senha" name="Usuario.Senha" />
19    </div>
20    <div>
21        <label for="Email">E-mail: </label>
22        <input type="text" id="Email" name="Usuario.Email" />
23    </div>
24    <button type="submit">Cadastrar</button>
25 </form>
```

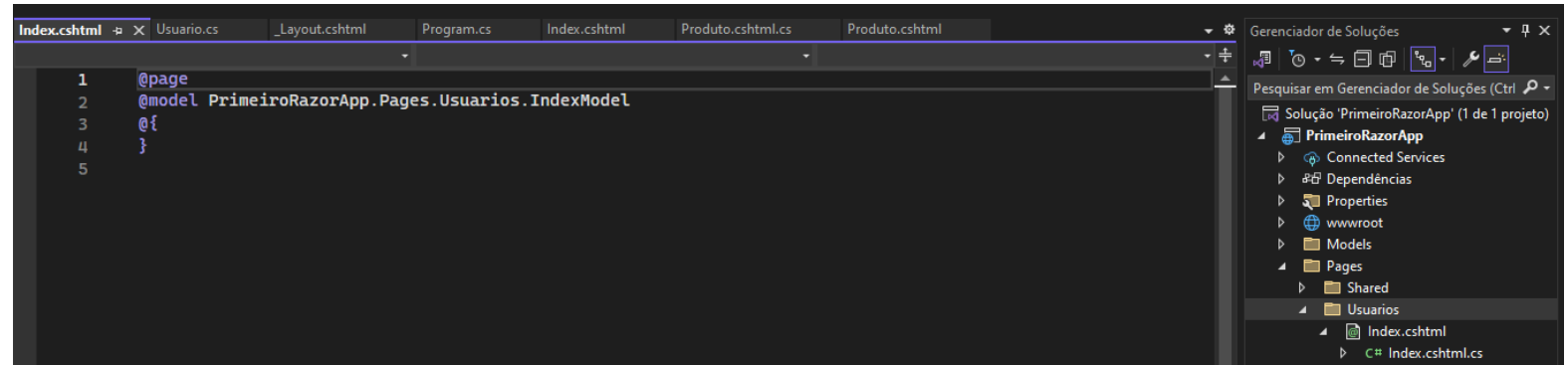
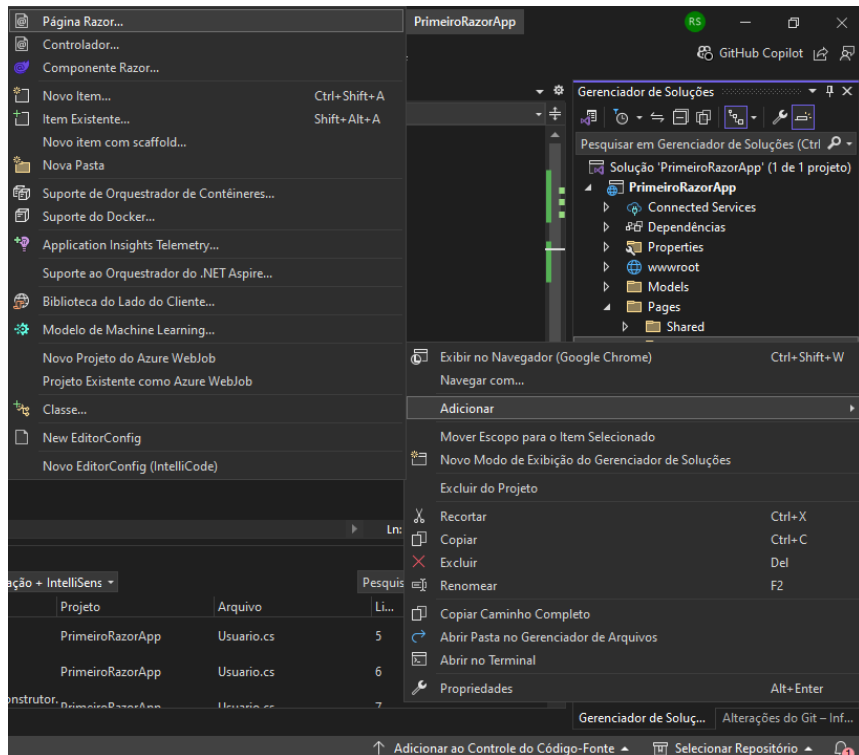
Criando um cadastro de Usuario

- Crie uma pasta chamada Usuarios dentro de Pages
- Aqui ficará todos os arquivos relacionados para cadastrar um usuário



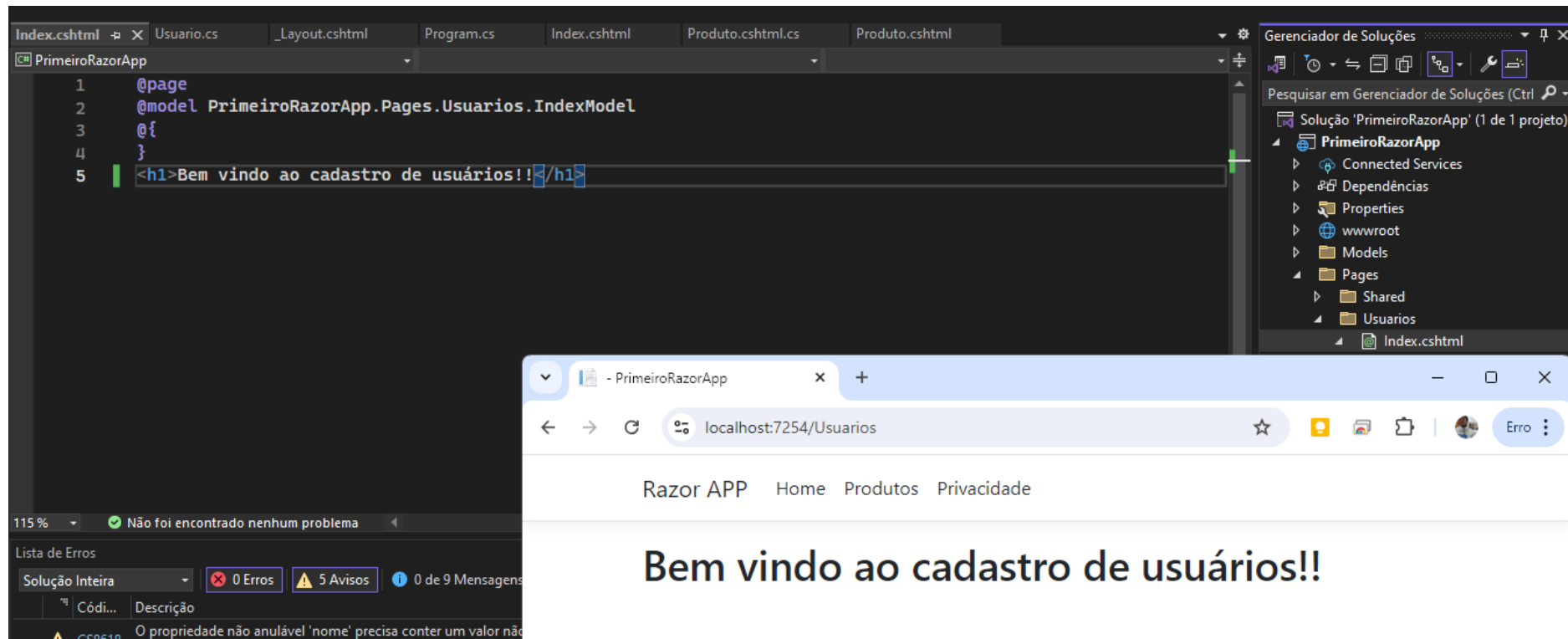
Criando um cadastro de Usuario

- Dentro da pasta Usuarios, crie uma Página Razor vazia chamada Index.cshtml



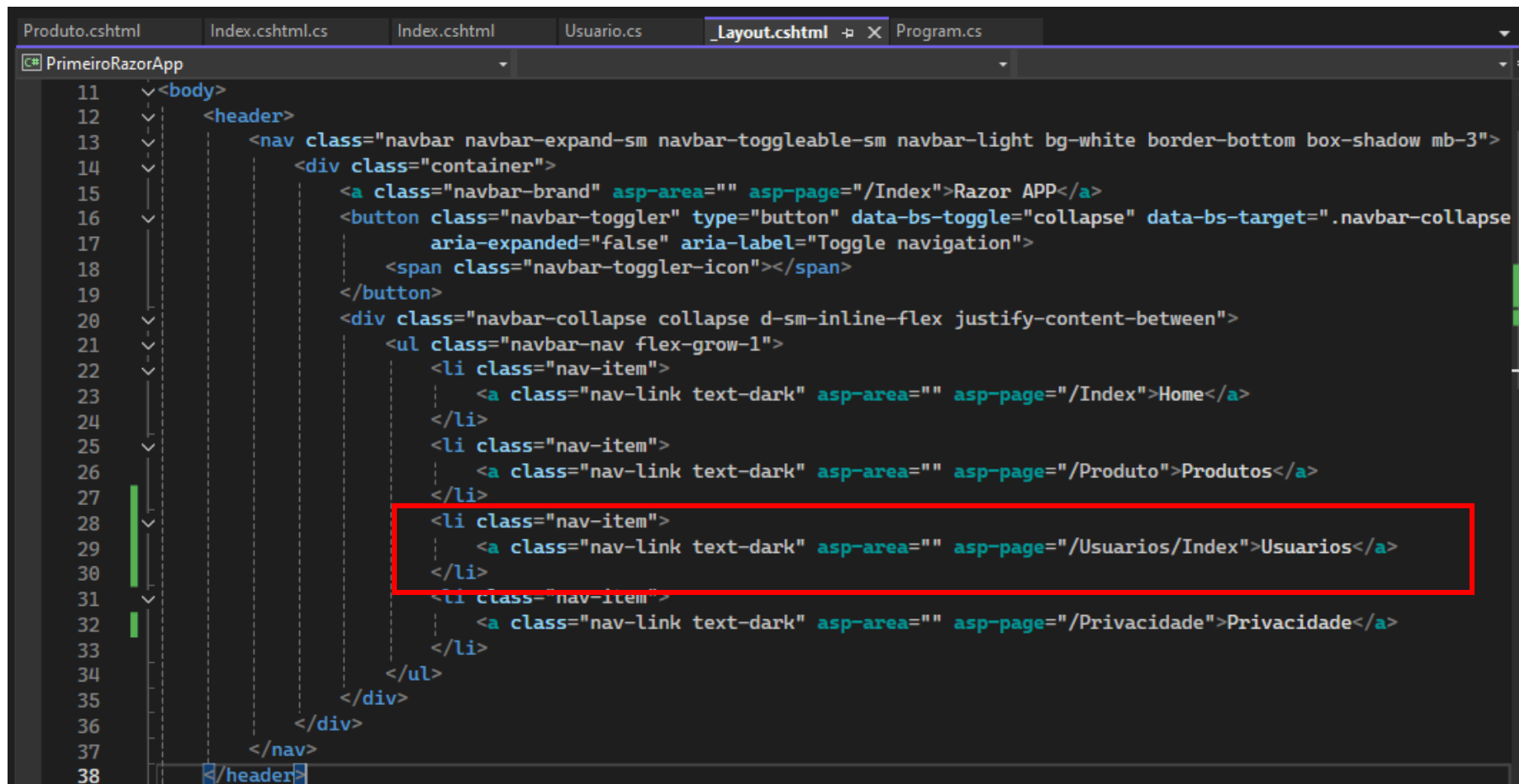
Criando um cadastro de Usuario

- Neste momento, já conseguimos acessar a página criada com o roteamento gerado



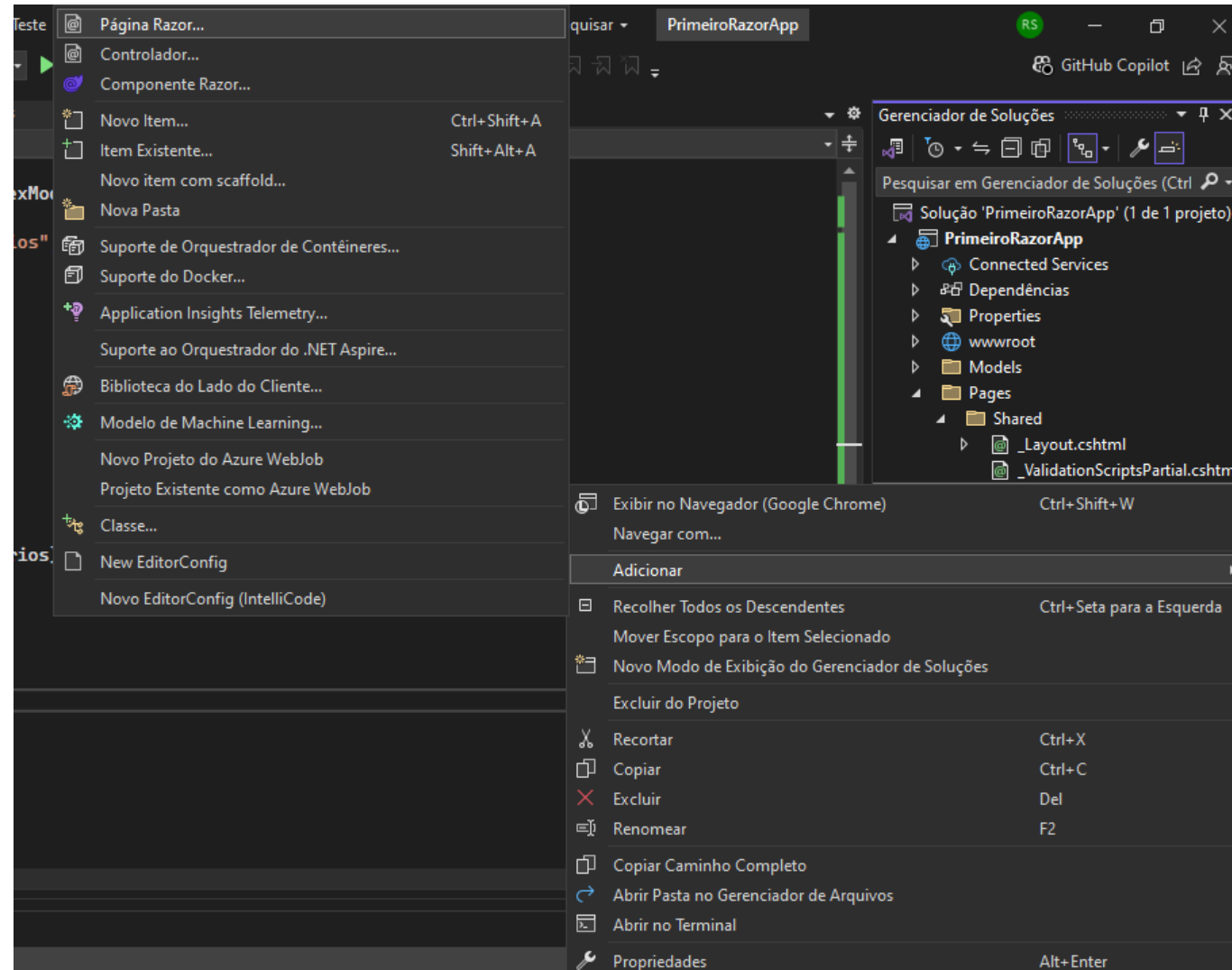
Criando um cadastro de Usuario

- No _Layout.cshtml:

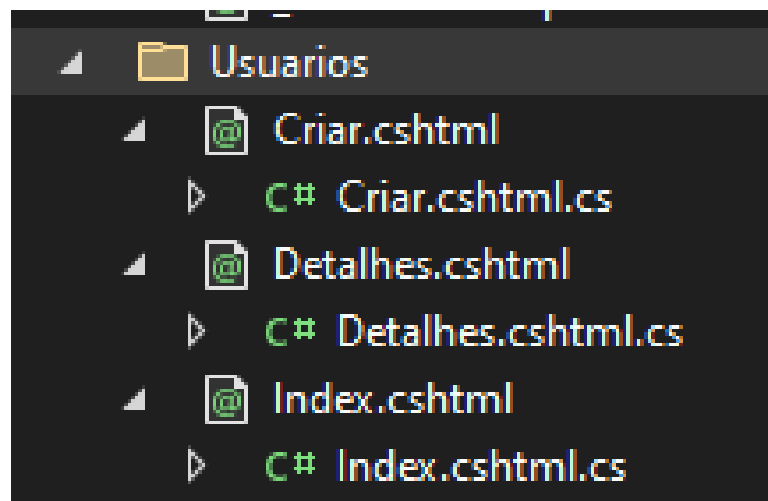


```
11 <body>
12 <header>
13 <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3">
14 <div class="container">
15 <a class="navbar-brand" asp-area="" asp-page="/Index">Razor APP</a>
16 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse"
17     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-page="/Index">Home</a>
24 </li>
25 <li class="nav-item">
26     <a class="nav-link text-dark" asp-area="" asp-page="/Produto">Produtos</a>
27 </li>
28 <li class="nav-item">
29     <a class="nav-link text-dark" asp-area="" asp-page="/Usuarios/Index">Usuarios</a>
30 </li>
31 <li class="nav-item">
32     <a class="nav-link text-dark" asp-area="" asp-page="/Privacidade">Privacidade</a>
33 </li>
34 </ul>
35 </div>
36 </div>
37 </nav>
38 </header>
```

Criando um Usuário



Criando um Usuário



Agora, vamos mudar o nosso Index.cshtml para que leia do arquivo

```
public class IndexModel : PageModel
{
    public List<Usuario> Usuarios { get; set; }

    public void OnGet()
    {
        Usuarios = new List<Usuario>(); //instanciei a lista onde carregarei os usuários do arquivo

        if (System.IO.File.Exists("usuarios.txt")) //Verifico se existe o arquivo salvo
        {
            //estou lendo as linhas do arquivo e armazeno num array de Strings
            var linhas = System.IO.File.ReadAllLines("usuarios.txt");

            foreach (var linha in linhas) //percorre as linhas e a cada registro, atribuo em linha
            {
                var dados = linha.Split(';'); // divido a linha em partes conforme o ;
                //dados[0] -> ID
                //dados[1] -> Nome
                //dados[2] -> Senha
                //Dados[3] -> Email

                var usuario = new Usuario()
                {
                    Id = int.Parse(dados[0]),
                    Nome = dados[1],
                    Senha = dados[2],
                    Email = dados[3]
                };
                //Adicionar o usuário à lista que eu instanciei lá no início
                Usuarios.Add(usuario);
            }
        }
    }
}
```

Detalhando e editando um usuário

- Para detalhar e editar um cadastro, nós precisaremos de um identificador único

```
public class IndexModel : PageModel
{
    public List<Usuario> Usuarios { get; set; }

    public void OnGet()
    {
        Usuarios = new List<Usuario>(); //instanciei a lista onde carregarei os usuários do arquivo

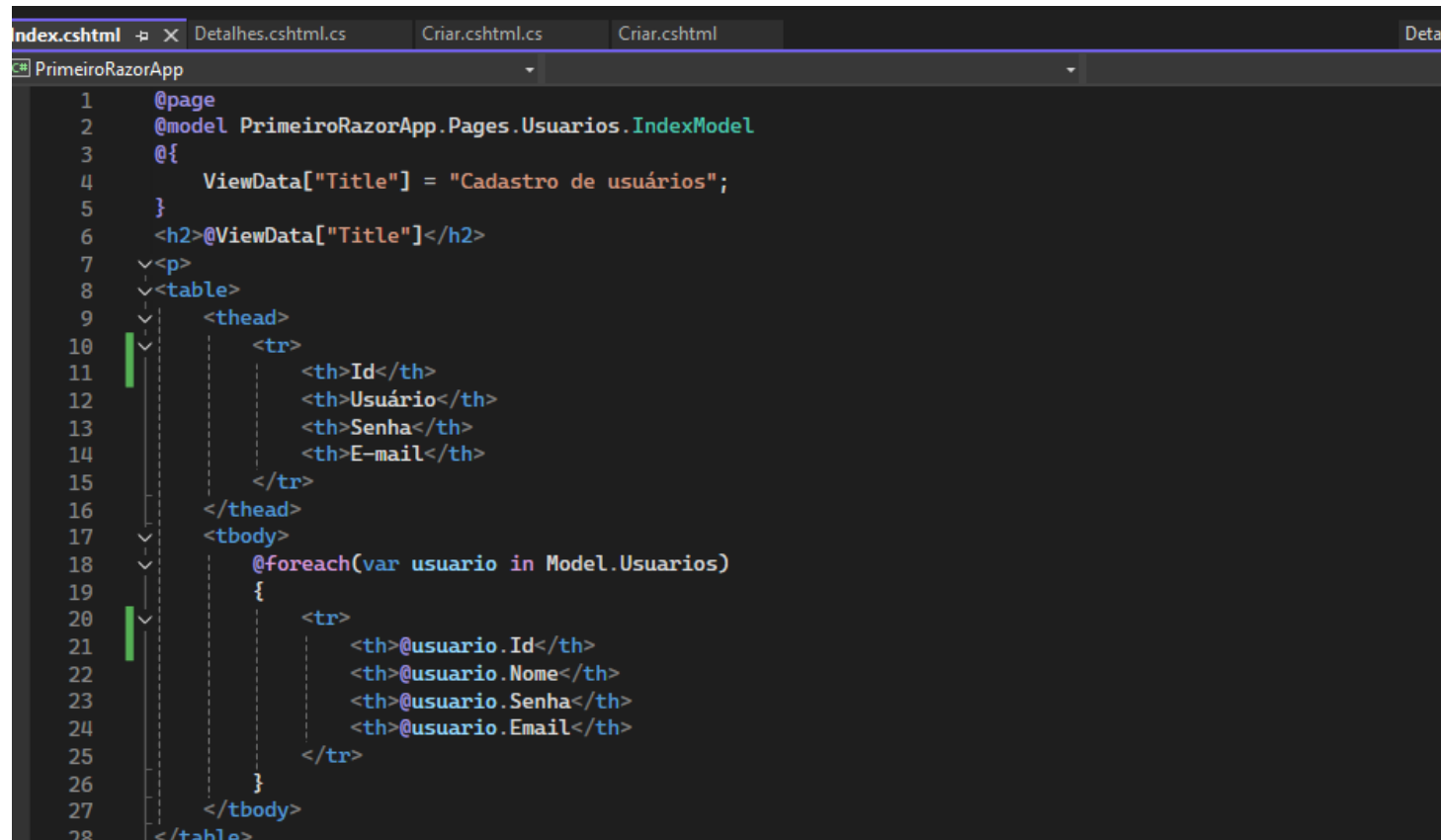
        if (System.IO.File.Exists("usuarios.txt")) //Verifico se existe o arquivo salvo
        {
            //estou lendo as linhas do arquivo e armazeno num array de Strings
            var linhas = System.IO.File.ReadAllLines("usuarios.txt");

            foreach (var linha in linhas) //percorre as linhas e a cada registro, atribuo em linha
            {
                var dados = linha.Split(';'); // divido a linha em partes conforme o ;
                //dados[0] -> ID
                //dados[1] -> Nome
                //dados[2] -> Senha
                //Dados[3] -> Email

                var usuario = new Usuario()
                {
                    Id = int.Parse(dados[0]),
                    Nome = dados[1],
                    Senha = dados[2],
                    Email = dados[3]
                };
                //Adicionar o usuário à lista que eu instanciei lá no início
                Usuarios.Add(usuario);
            }
        }
    }
}
```

Detalhando e editando um usuário

- Para detalhar e editar um cadastro, nós precisaremos de um identificador único



The screenshot shows a code editor with a file named `index.cshtml` open. The code is a Razor view for a user management application. It starts with a `@page` directive and a `@model` directive pointing to `PrimeiroRazorApp.Pages.Usuarios.IndexModel`. The view sets the page title to "Cadastro de usuários" and displays a table of users. The table has a header with columns for Id, Usuário, Senha, and E-mail. The body of the table is populated using a `@foreach` loop over the `Model.Usuarios` collection. Each row in the table displays the user's Id, Name (`Nome`), Password (`Senha`), and Email (`Email`).

```
1  @page
2  @model PrimeiroRazorApp.Pages.Usuarios.IndexModel
3  @{
4      ViewData["Title"] = "Cadastro de usuários";
5  }
6  <h2>@ViewData["Title"]</h2>
7  <p>
8  <table>
9      <thead>
10         <tr>
11             <th>Id</th>
12             <th>Usuário</th>
13             <th>Senha</th>
14             <th>E-mail</th>
15         </tr>
16     </thead>
17     <tbody>
18         @foreach (var usuario in Model.Usuarios)
19         {
20             <tr>
21                 <th>@usuario.Id</th>
22                 <th>@usuario.Nome</th>
23                 <th>@usuario.Senha</th>
24                 <th>@usuario.Email</th>
25             </tr>
26         }
27     </tbody>
28 </table>
```

Detalhando e editando um usuário

```
public class DetalhesModel : PageModel
{
    public Usuario Usuario { get; set; }

    public IActionResult OnGet(int id)
    {
        //Carregar a lista de usuários
        var usuarios = CarregarUsuarios();

        //Buscar o usuário -> LINQ
        Usuario = usuarios.FirstOrDefault(u => u.Id == id);

        if (Usuario == null)
        {
            return RedirectToPage("/Usuario/Index");
        }
        return Page();
    }
}
```

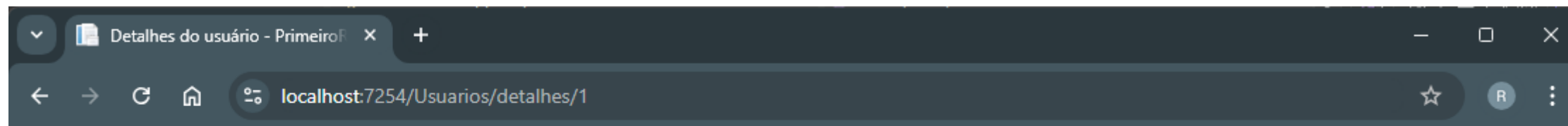
```
public List<Usuario> CarregarUsuarios()
{
    var Usuarios = new List<Usuario>();
    if (System.IO.File.Exists("usuarios.txt"))
    {
        var linhas = System.IO.File.ReadAllLines("usuarios.txt");

        foreach (var linha in linhas)
        {
            var dados = linha.Split(';');
            var usuario = new Usuario()
            {
                Id = int.Parse(dados[0]),
                Nome = dados[1],
                Senha = dados[2],
                Email = dados[3]
            };
            Usuarios.Add(usuario);
        }
    }
    return Usuarios;
}
```


Detalhando e editando um usuário

```
Detalhes.cshtml.cs  Detalhes.cshtml  X
C# PrimeiroRazorApp
1  @page "{id:int}"
2  @model PrimeiroRazorApp.Pages.Usuarios.DetalhesModel
3  @{
4      ViewData["Title"] = "Detalhes do usuário";
5  }
6  <h2>@ViewData["Title"]</h2>
7
8  <div>
9      <label>Id: </label>
10     <span>@Model.Usuario.Id</span>
11 </div>
12 <div>
13     <label>Nome: </label>
14     <span>@Model.Usuario.Nome</span>
15 </div>
16 <div>
17     <label>E-mail: </label>
18     <span>@Model.Usuario.Email</span>
19 </div>
20
21 <a asp-page="/Usuarios/Index">Voltar</a>
```

Detalhando e editando um usuário



[Razor APP](#) [Home](#) [Produtos](#) [Usuarios](#) [Privacidade](#)

Detalhes do usuário

Id: 1

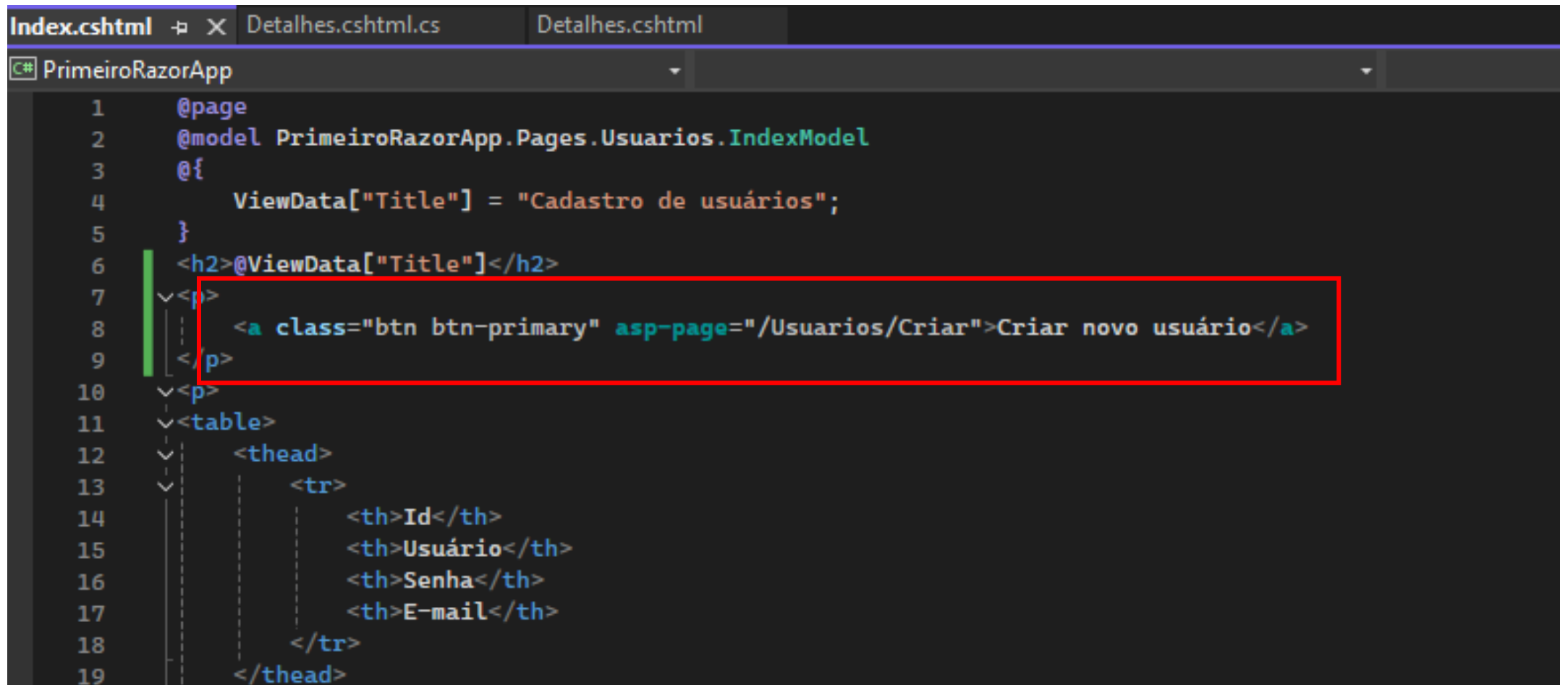
Nome: Ricardo

E-mail: ricardo@frohlich.inf.br

[Voltar](#)

Alterando o Index.cshtml

- Colocando um botão para criar um novo usuário



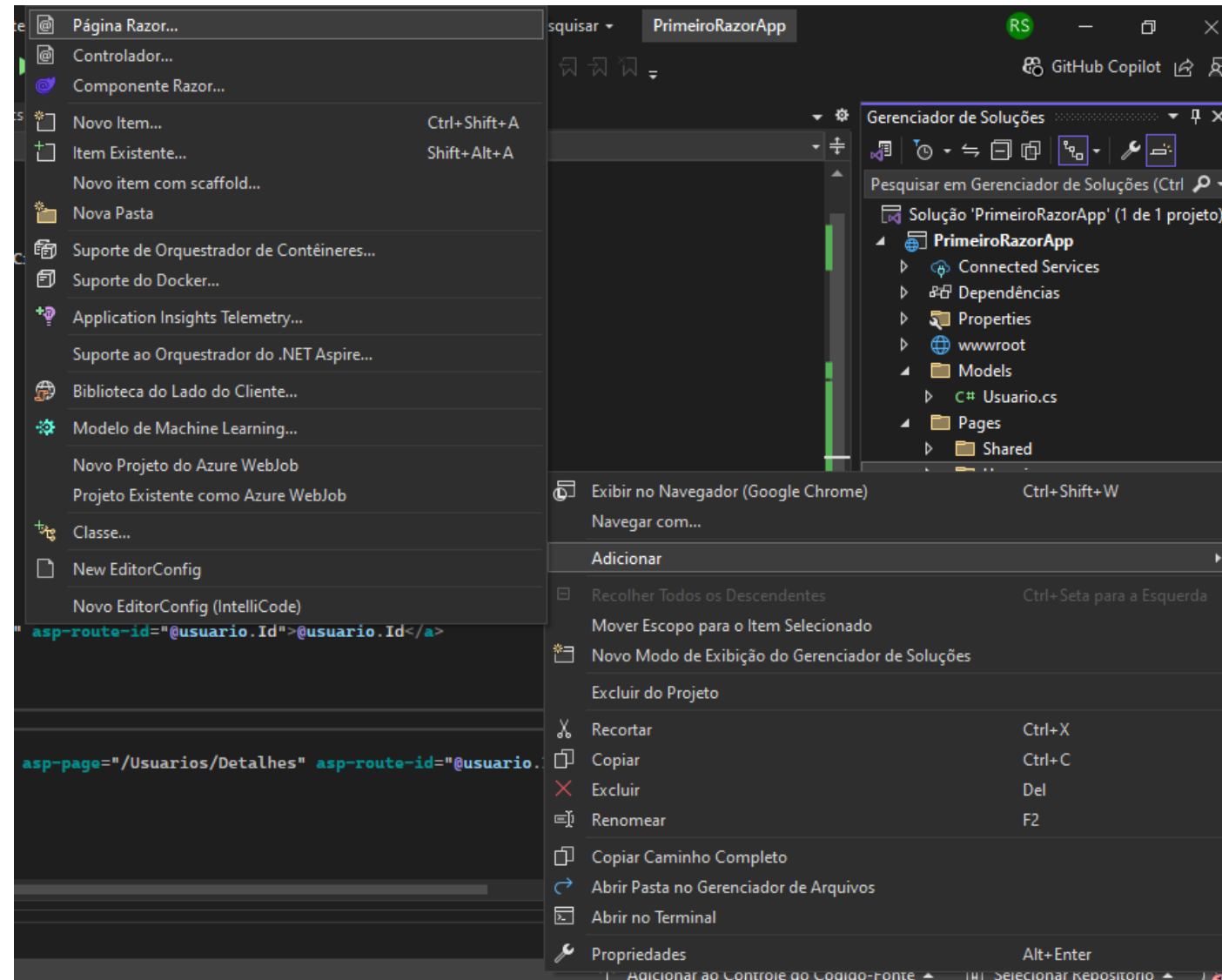
```
Index.cshtml  X  Detalhes.cshtml.cs  Detalhes.cshtml
C# PrimeiroRazorApp
1  @page
2  @model PrimeiroRazorApp.Pages.Usuarios.IndexModel
3  @{
4      ViewData["Title"] = "Cadastro de usuários";
5  }
6  <h2>@ViewData["Title"]</h2>
7  <p>
8      <a class="btn btn-primary" asp-page="/Usuarios/Criar">Criar novo usuário</a>
9  </p>
10 <p>
11 <table>
12 <thead>
13 <tr>
14 <th>Id</th>
15 <th>Usuário</th>
16 <th>Senha</th>
17 <th>E-mail</th>
18 </tr>
19 </thead>
```

Alterando o Index.cshtml

- Acessando detalhes do Usuário

```
6 <h2>@ViewData["Title"]</h2>
7 <p>
8   <a class="btn btn-primary" asp-page="/Usuarios/Criar">Criar novo usuário</a>
9 </p>
10 <p>
11 <table>
12 <thead>
13 <tr>
14 <th>Id</th>
15 <th>Usuário</th>
16 <th>Senha</th>
17 <th>E-mail</th>
18 </tr>
19 </thead>
20 <tbody>
21 @foreach(var usuario in Model.Usuarios)
22 {
23 <tr>
24 <td>
25   <a asp-page="/Usuarios/Detalhes" asp-route-id="@usuario.Id">@usuario.Id</a>
26 </td>
27 <td>@usuario.Nome</td>
28 <td>@usuario.Senha</td>
29 <td>@usuario.Email</td>
30 <td>
31   <a class="btn btn-outline-info" asp-page="/Usuarios/Detalhes" asp-route-id="@usuario.Id">Detalhar</a>
32 </td>
33 </tr>
34 }
35 </tbody>
36 </table>
```

Criando uma Razor Pages para Editar



No Editar.cshtml

```
Criar.cshtml.cs  Criar.cshtml  Editar.cshtml.cs  Editar.cshtml  Index.cshtml
C# PrimeiroRazorApp
1  @page "{id:int}"
2  @model PrimeiroRazorApp.Pages.Usuarios.EditarModel
3  @{
4      ViewData["Title"] = "Editar Usuário";
5  }
6
7  <h2>@ViewData["Title"]</h2>
8
9  <form method="post">
10     <input type="hidden" asp-for="Usuario.Id" />
11     <div class="form-group">
12         <label asp-for="Usuario.Nome">Nome</label>
13         <input asp-for="Usuario.Nome" class="form-control" />
14     </div>
15     <div class="form-group">
16         <label asp-for="Usuario.Senha">Senha</label>
17         <input asp-for="Usuario.Senha" type="password" class="form-control" />
18     </div>
19     <div class="form-group">
20         <label asp-for="Usuario.Email">E-mail</label>
21         <input asp-for="Usuario.Email" class="form-control" />
22     </div>
23     <button type="submit" class="btn btn-primary">Salvar</button>
24     <a asp-page="/Usuarios/Index" class="btn btn-secondary">Cancelar</a>
25 </form>
```

Adicionando um botão no Index para editar um usuário

```
Editar.cshtml.cs  Editar.cshtml  Detalhes.cshtml  Index.cshtml  Detalhes.cshtml.cs  Criar.cshtml  Criar.cshtml.cs
C# PrimeiroRazorApp
13      <tr>
14          <th>Id</th>
15          <th>Usuário</th>
16          <th>Senha</th>
17          <th>E-mail</th>
18      </tr>
19  </thead>
20  <tbody>
21      @foreach(var usuario in Model.Usuarios)
22      {
23          <tr>
24              <td>
25                  <a asp-page="/Usuarios/Detalhes" asp-route-id="@usuario.Id">@usuario.Id</a>
26              </td>
27              <td>@usuario.Nome</td>
28              <td>@usuario.Senha</td>
29              <td>@usuario.Email</td>
30              <td>
31                  <a class="btn btn-outline-info" asp-page="/Usuarios/Detalhes" asp-route-id="@usuario.Id">Detalhar</a>
32              </td>
33              <td>
34                  <a class="btn btn-outline-info" asp-page="/Usuarios/Editar" asp-route-id="@usuario.Id">Editar</a>
35              </td>
36          </tr>
37      }
38  </tbody>
39  </table>
```

No Editar.cshtml.cs

```
40 public List<Usuario> CarregarUsuarios()
41 {
42     var Usuarios = new List<Usuario>();
43     if (System.IO.File.Exists("usuarios.txt"))
44     {
45         var linhas = System.IO.File.ReadAllLines("usuarios.txt");
46
47         foreach (var linha in linhas)
48         {
49             var dados = linha.Split(';');
50             var usuario = new Usuario()
51             {
52                 Id = int.Parse(dados[0]),
53                 Nome = dados[1],
54                 Senha = dados[2],
55                 Email = dados[3]
56             };
57             Usuarios.Add(usuario);
58         }
59     }
60     return Usuarios;
61 }
62 }
```


No Editar.cshtml.cs

```
5 namespace PrimeiroRazorApp.Pages.Usuarios
6 {
7     public class EditarModel : PageModel
8     {
9         [BindProperty]
10         public Usuario Usuario { get; set; }
11         public void OnGet(int id)
12         {
13             var usuarios = CarregarUsuarios();
14             Usuario = usuarios.FirstOrDefault(u => u.Id == id);
15         }
16
17
18         public IActionResult OnPost()
19         {
20             if (!ModelState.IsValid)
21             {
22                 return Page();
23             }
24             var linhas = System.IO.File.ReadAllLines("usuarios.txt").ToList();
25
26             for (int i = 0; i < linhas.Count; i++)
27             {
28                 var dados = linhas[i].Split(';');
29                 if (int.Parse(dados[0]) == Usuario.Id)
30                 {
31                     linhas[i] = Usuario.Id + ";" + Usuario.Nome + ";" + Usuario.Senha + ";" + Usuario.Email;
32                     break;
33                 }
34             }
35             System.IO.File.WriteAllLines("usuarios.txt", linhas);
36             return RedirectToPage("/Usuarios/Index");
37         }
38     }
}
```

Manipular Formulários Utilizando Model Binding

```
1  @page "{id:int}"
2  @model PrimeiroRazorApp.Pages.Usuarios.EditarModel
3  @{
4      ViewData["Title"] = "Editar Usuário";
5  }
6
7  <h2>@ViewData["Title"]</h2>
8
9  <form method="post">
10     <input type="hidden" asp-for="Usuario.Id" />
11     <div class="form-group">
12         <label asp-for="Usuario.Nome">Nome</label>
13         <input asp-for="Usuario.Nome" class="form-control" />
14     </div>
15     <div class="form-group">
16         <label asp-for="Usuario.Senha">Senha</label>
17         <input asp-for="Usuario.Senha" type="password" class="form-control" />
18     </div>
19     <div class="form-group">
20         <label asp-for="Usuario.Email">E-mail</label>
21         <input asp-for="Usuario.Email" class="form-control" />
22     </div>
23     <button type="submit" class="btn btn-primary">Salvar</button>
24     <a asp-page="/Usuarios/Index" class="btn btn-secondary">Cancelar</a>
25 </form>
```

Manipular formulários utilizando Model Binding

- Use asp-for no formulário Razor para vincular os campos do formulário às propriedades do modelo:

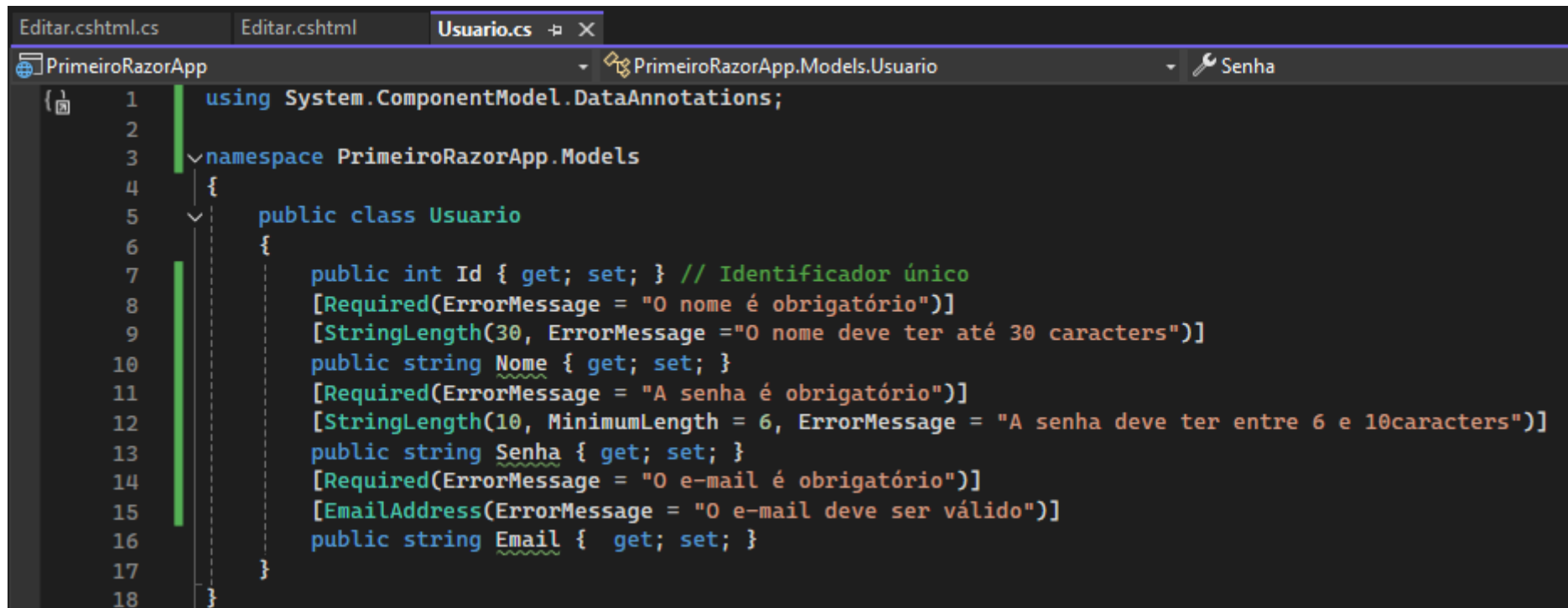
```
1  @page "{id:int}"
2  @model PrimeiroRazorApp.Pages.Usuarios.EditarModel
3  @{
4      ViewData["Title"] = "Editar Usuário";
5  }
6
7  <h2>@ViewData["Title"]</h2>
8
9  <form method="post">
10     <input type="hidden" asp-for="Usuario.Id" />
11     <div class="form-group">
12         <label asp-for="Usuario.Nome">Nome</label>
13         <input asp-for="Usuario.Nome" class="form-control" />
14     </div>
15     <div class="form-group">
16         <label asp-for="Usuario.Senha">Senha</label>
17         <input asp-for="Usuario.Senha" type="password" class="form-control" />
18     </div>
19     <div class="form-group">
20         <label asp-for="Usuario.Email">E-mail</label>
21         <input asp-for="Usuario.Email" class="form-control" />
22     </div>
23     <button type="submit" class="btn btn-primary">Salvar</button>
24     <a asp-page="/Usuarios/Index" class="btn btn-secondary">Cancelar</a>
25 </form>
```

Manipular formulários utilizando Model Binding

- No método OnPost, os dados enviados pelo formulário são automaticamente vinculados ao modelo graças à anotação [BindProperty].

```
namespace PrimeiroRazorApp.Pages.Usuarios
{
    public class EditarModel : PageModel
    {
        [BindProperty]
        public Usuario Usuario { get; set; }
        public void OnGet(int id)
```

Validação dos Dados do Lado do Servidor



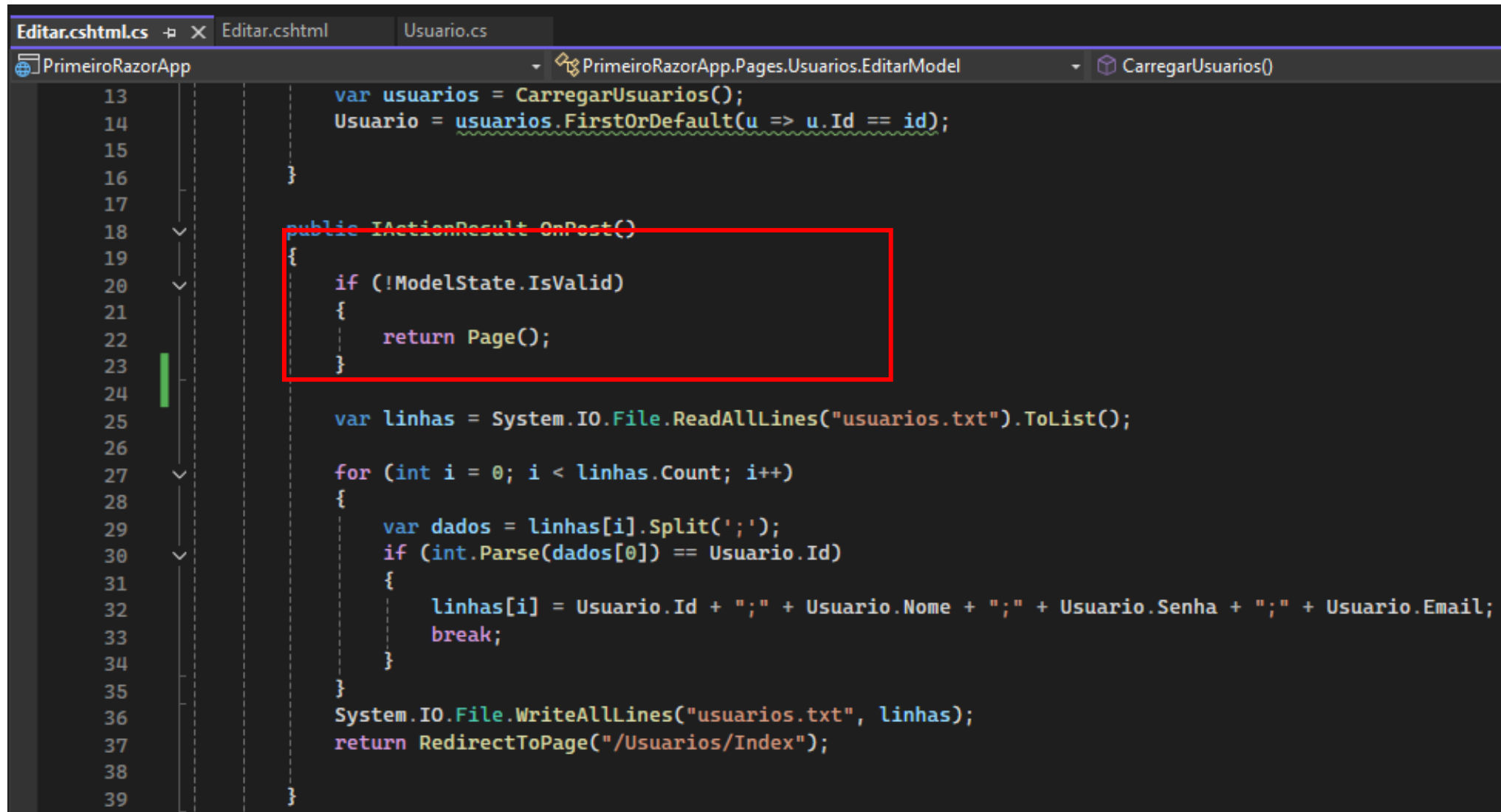
The image shows a code editor window with the file 'Usuario.cs' open. The code defines a 'Usuario' class within the 'PrimeiroRazorApp.Models' namespace. It includes several validation attributes from 'System.ComponentModel.DataAnnotations' to enforce data integrity on the server side. The attributes are applied to the 'Nome' and 'Senha' properties, ensuring they are required, have specific lengths, and the email is valid.

```
1 using System.ComponentModel.DataAnnotations;
2
3 namespace PrimeiroRazorApp.Models
4 {
5     public class Usuario
6     {
7         public int Id { get; set; } // Identificador único
8         [Required(ErrorMessage = "O nome é obrigatório")]
9         [StringLength(30, ErrorMessage = "O nome deve ter até 30 caracteres")]
10        public string Nome { get; set; }
11        [Required(ErrorMessage = "A senha é obrigatório")]
12        [StringLength(10, MinimumLength = 6, ErrorMessage = "A senha deve ter entre 6 e 10caracters")]
13        public string Senha { get; set; }
14        [Required(ErrorMessage = "O e-mail é obrigatório")]
15        [EmailAddress(ErrorMessage = "O e-mail deve ser válido")]
16        public string Email { get; set; }
17    }
18 }
```

Validação dos Dados do Lado do Servidor

```
Editar.cshtml.cs  Editar.cshtml  X  Usuario.cs
PrimeiroRazorApp
1  @page "{id:int}"
2  @model PrimeiroRazorApp.Pages.Usuarios.EditarModel
3  @{
4      ViewData["Title"] = "Editar Usuário";
5  }
6
7  <h2>@ViewData["Title"]</h2>
8
9  <form method="post">
10     <input type="hidden" asp-for="Usuario.Id" />
11     <div class="form-group">
12         <label asp-for="Usuario.Nome">Nome</label>
13         <input asp-for="Usuario.Nome" class="form-control" />
14         <span asp-validation-for="Usuario.Nome" class="text-danger"></span>
15     </div>
16     <div class="form-group">
17         <label asp-for="Usuario.Senha">Senha</label>
18         <input asp-for="Usuario.Senha" type="password" class="form-control" />
19         <span asp-validation-for="Usuario.Senha" class="text-danger"></span>
20     </div>
21     <div class="form-group">
22         <label asp-for="Usuario.Email">E-mail</label>
23         <input asp-for="Usuario.Email" class="form-control" />
24         <span asp-validation-for="Usuario.Email" class="text-danger"></span>
25     </div>
26     <button type="submit" class="btn btn-primary">Salvar</button>
27     <a asp-page="/Usuarios/Index" class="btn btn-secondary">Cancelar</a>
28 </form>
```

Validação dos Dados do Lado do Servidor



```
13      var usuarios = CarregarUsuarios();
14      Usuario = usuarios.FirstOrDefault(u => u.Id == id);
15
16  }
17
18  public IActionResult OnPost()
19  {
20      if (!ModelState.IsValid)
21      {
22          return Page();
23      }
24
25      var linhas = System.IO.File.ReadAllLines("usuarios.txt").ToList();
26
27      for (int i = 0; i < linhas.Count; i++)
28      {
29          var dados = linhas[i].Split(';');
30          if (int.Parse(dados[0]) == Usuario.Id)
31          {
32              linhas[i] = Usuario.Id + ";" + Usuario.Nome + ";" + Usuario.Senha + ";" + Usuario.Email;
33              break;
34          }
35      }
36      System.IO.File.WriteAllLines("usuarios.txt", linhas);
37      return RedirectToPage("/Usuarios/Index");
38  }
39  }
```

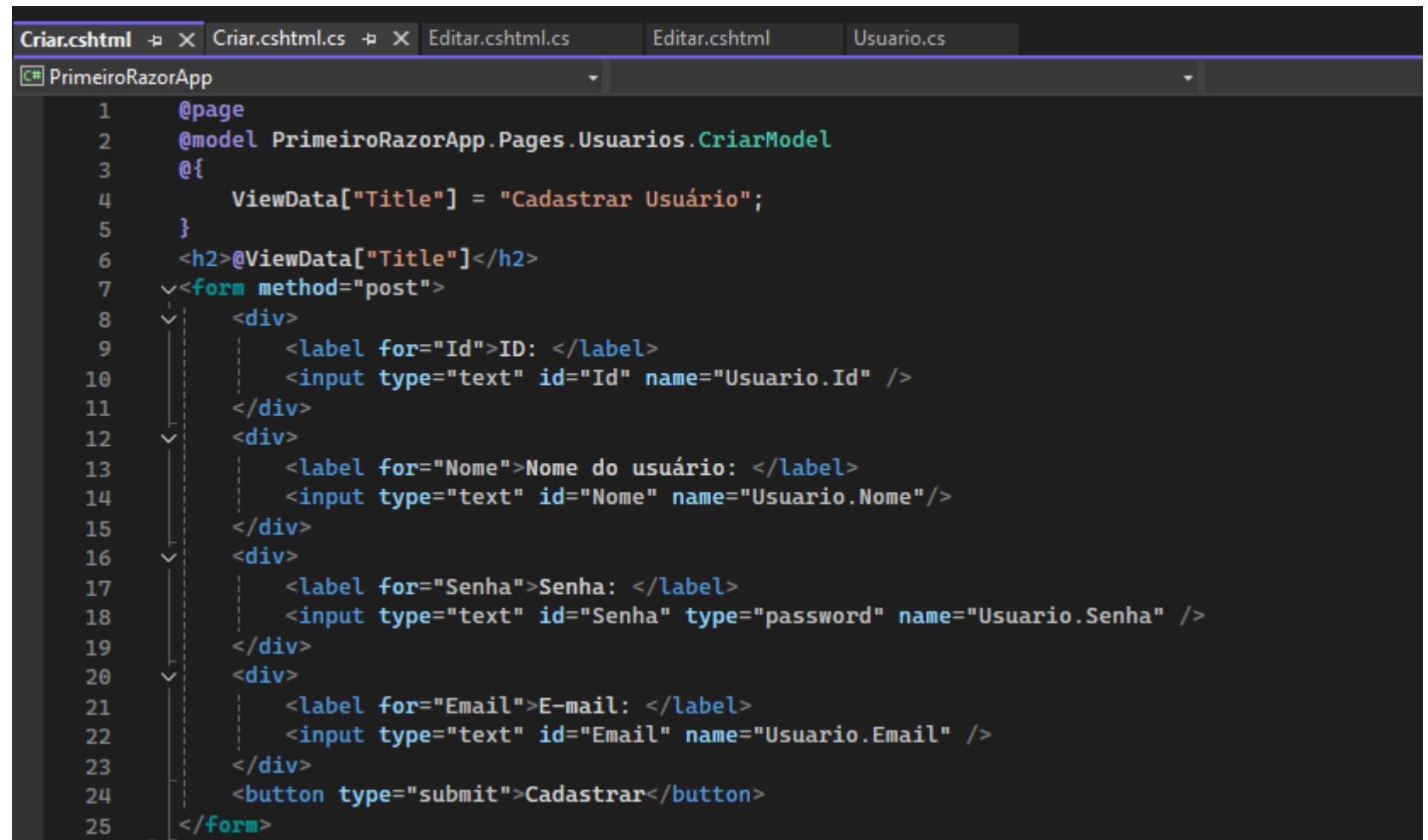
Validação dos Dados do Lado do Servidor

- No Editar.cshtml está tudo validado, mas e no criar?
 - Testem...

Validação dos Dados do Lado do Servidor

- No Criar.cshtml agora está assim:

- O que falta?



The screenshot shows a code editor with the following tabs: Criar.cshtml, Criar.cshtml.cs, Editar.cshtml.cs, Editar.cshtml, and Usuario.cs. The active file is Criar.cshtml, which contains the following code:

```
1 @page
2 @model PrimeiroRazorApp.Pages.Usuarios.CriarModel
3 @{
4     ViewData["Title"] = "Cadastrar Usuário";
5 }
6 <h2>@ViewData["Title"]</h2>
7 <form method="post">
8     <div>
9         <label for="Id">ID: </label>
10        <input type="text" id="Id" name="Usuario.Id" />
11    </div>
12    <div>
13        <label for="Nome">Nome do usuário: </label>
14        <input type="text" id="Nome" name="Usuario.Nome" />
15    </div>
16    <div>
17        <label for="Senha">Senha: </label>
18        <input type="text" id="Senha" type="password" name="Usuario.Senha" />
19    </div>
20    <div>
21        <label for="Email">E-mail: </label>
22        <input type="text" id="Email" name="Usuario.Email" />
23    </div>
24    <button type="submit">Cadastrar</button>
25 </form>
```

Validação dos Dados do Lado do Servidor

- Falta usar o ModelBinding e a validação do modelo:

```
Editar.cshtml  Criar.cshtml  Criar.cshtml.cs
PrimeiroRazorApp
1  @page
2  @model PrimeiroRazorApp.Pages.Usuarios.CriarModel
3  @{
4      ViewData["Title"] = "Cadastrar Usuário";
5  }
6  <h2>@ViewData["Title"]</h2>
7  <form method="post">
8      <div>
9          <label asp-for="usuario.Id">ID: </label>
10         <input asp-for="usuario.Id" class="form-control" />
11         <span asp-validation-for="usuario.Id" class="text-danger"></span>
12     </div>
13     <div class="form-group">
14         <label asp-for="usuario.Nome">Nome</label>
15         <input asp-for="usuario.Nome" class="form-control" />
16         <span asp-validation-for="usuario.Nome" class="text-danger"></span>
17     </div>
18     <div class="form-group">
19         <label asp-for="usuario.Senha">Senha</label>
20         <input asp-for="usuario.Senha" type="password" class="form-control" />
21         <span asp-validation-for="usuario.Senha" class="text-danger"></span>
22     </div>
23     <div class="form-group">
24         <label asp-for="usuario.Email">E-mail</label>
25         <input asp-for="usuario.Email" class="form-control" />
26         <span asp-validation-for="usuario.Email" class="text-danger"></span>
27     </div>
28     <button type="submit">Cadastrar</button>
29 </form>
```

Validação dos Dados do Lado do Servidor

- Resultado:

[Razor APP](#) [Home](#) [Produtos](#) [Usuarios](#) [Privacidade](#)

Cadastrar Usuário

ID:

The value " " is invalid.

Nome

O nome é obrigatório

Senha

A senha é obrigatório

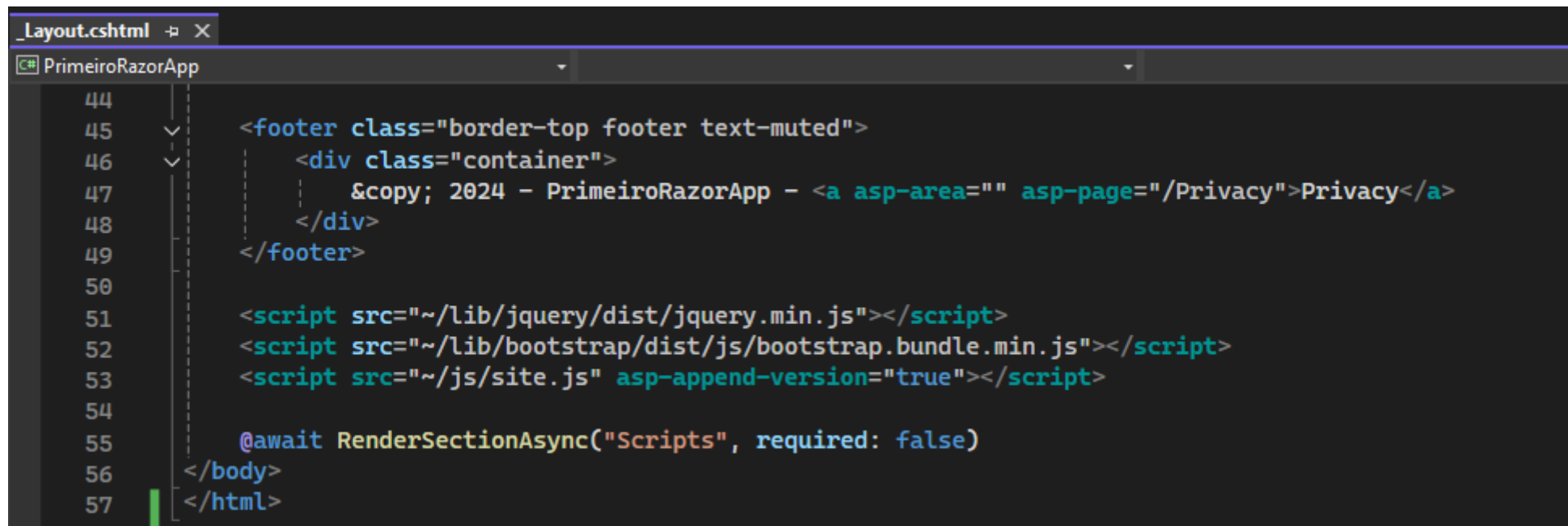
E-mail

O e-mail é obrigatório

Cadastrar

Validação dos Dados do Lado do Cliente

- ASP.NET Core Razor Pages já inclui as bibliotecas necessárias para a validação do lado do cliente (jquery.validate e jquery.validate.unobtrusive).
- Certifique-se de que estão referenciadas no seu layout:



```
44
45 <footer class="border-top footer text-muted">
46     <div class="container">
47         &copy; 2024 - PrimeiroRazorApp - <a asp-area="" asp-page="/Privacy">Privacy</a>
48     </div>
49 </footer>
50
51 <script src="~/lib/jquery/dist/jquery.min.js"></script>
52 <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
53 <script src="~/js/site.js" asp-append-version="true"></script>
54
55 @await RenderSectionAsync("Scripts", required: false)
56 </body>
57 </html>
```

Validação dos Dados do Lado do Cliente

- O Razor Pages automaticamente gera os atributos de validação HTML5 necessários, como `required`, `data-val`, `data-val-required`, etc., com base nas Data Annotations.
- Quando o usuário tenta submeter o formulário, a validação do lado do cliente será acionada, evitando o envio se houver erros.

Criar URLs para Razor Pages com Roteamento

- O roteamento em ASP.NET Core permite a criação de URLs amigáveis e legíveis para as páginas da aplicação.
- Isso melhora a navegabilidade e a experiência do usuário.

Criar URLs para Razor Pages com Roteamento

```
Program.cs  x Produto.cshtml.cs  ProdutoComponent.razor  Produto.cshtml
MinhaAplicacaoRazor
1  var builder = WebApplication.CreateBuilder(args);
2
3  // Add services to the container.
4  builder.Services.AddRazorPages();
5
6  var app = builder.Build();
7
8  // Configure the HTTP request pipeline.
9  if (!app.Environment.IsDevelopment())
10 {
11     app.UseExceptionHandler("/Error");
12     // The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts
13     app.UseHsts();
14 }
15
16 app.UseHttpsRedirection();
17 app.UseStaticFiles();
18
19 app.UseRouting();
20
21 app.UseAuthorization();
22
23 app.MapRazorPages();
24
25 app.Run();
```

Layouts em Razor Pages

- Layouts em Razor Pages são como "moldes" para suas páginas web. Imagine que você está criando várias páginas para um site, e todas elas têm a mesma estrutura básica, como um cabeçalho com o logo, um menu de navegação no topo, e um rodapé com informações de contato.
- Em vez de repetir esse código em cada página, você cria um Layout que contém essa estrutura comum.

Layouts em Razor Pages

- Um layout é definido como um arquivo Razor (.cshtml) separado, normalmente localizado na pasta Pages/Shared com o nome `_Layout.cshtml`.
- Para aplicar um layout a uma página Razor, você define a propriedade `Layout` no topo do arquivo da página ou no arquivo `_ViewStart.cshtml`, que é executado antes de cada página:

Partials em Razor Pages

- Partials são fragmentos de páginas Razor que podem ser reutilizados em várias páginas ou dentro de um layout.
- Servem para componentes de interface do usuário que aparecem em várias páginas, como formulários, listas, ou seções de uma página.