

Programação .NET

Professor Ricardo Frohlich da Silva

Desenvolver uma aplicação web full-stack ("server-based UI") em ASP.NET Core usando Razor Pages

- Criar uma aplicação Razor Pages com ASP.NET
- Entender a anatomia de uma aplicação ASP.NET Web
- Criar uma aplicação Web com um componente usando Razor
- Adicionar dados a uma página utilizando Page Model
- Criar URLs para Razor Pages com roteamento
- Manipular formulários utilizando Model Binding
- Realizar a validação dos dados do lado do servidor
- Realizar a validação dos dados do lado do cliente
- Escrever arquivos em ASP.NET usando Razor
- Ler arquivos em ASP.NET usando Razor

Componentes

- A linguagem C#
- Plataforma/Framework: ASP.NET
 - ASP.NET é um framework da Microsoft para construção de aplicações web e serviços. Ele fornece um ambiente de execução para aplicações web escritas em diferentes linguagens, incluindo C#.
- ASP.NET Core:
 - ASP.NET Core é uma versão mais recente e modular do ASP.NET. Ele é de código aberto, multiplataforma e otimizado para desempenho. ASP.NET Core permite o desenvolvimento de aplicações web modernas e escaláveis.

Componentes

- IDE (Ambiente de Desenvolvimento Integrado): Visual Studio:
 - O Visual Studio é a IDE preferida para desenvolvimento de aplicações C# e ASP.NET. Ele oferece ferramentas poderosas para edição de código, depuração, testes e publicação.
- Banco de Dados: Entity Framework Core:
 - Entity Framework Core é um ORM (Object-Relational Mapping) que simplifica o acesso a bancos de dados relacionais. Ele permite que você interaja com o banco de dados usando objetos C# em vez de SQL puro.
- Frontend: HTML, CSS, JavaScript:
 - Desenvolver aplicações web envolve a criação de interfaces de usuário interativas. HTML é usado para estruturar a página, CSS para estilos e layout, e JavaScript para interatividade do lado do cliente.

Componentes

- ASP.NET Razor:
 - ASP.NET Razor é uma sintaxe de marcação que combina código C# com HTML. É frequentemente usado para criar páginas dinâmicas em ASP.NET.
- Model-View-Controller (MVC) ou Razor Pages:
 - O padrão de arquitetura MVC (Model-View-Controller) é comumente usado em ASP.NET. Ele divide a aplicação em três componentes: Model (lógica de negócios), View (interface do usuário) e Controller (lógica de controle). Além disso, há o modelo alternativo chamado Razor Pages, que simplifica a estruturação da aplicação.
- Web API: ASP.NET Web API:
 - Para construir serviços web (APIs RESTful), a ASP.NET Web API é uma escolha comum. Ela permite a criação de APIs que podem ser consumidas por clientes web, móveis ou outras aplicações.

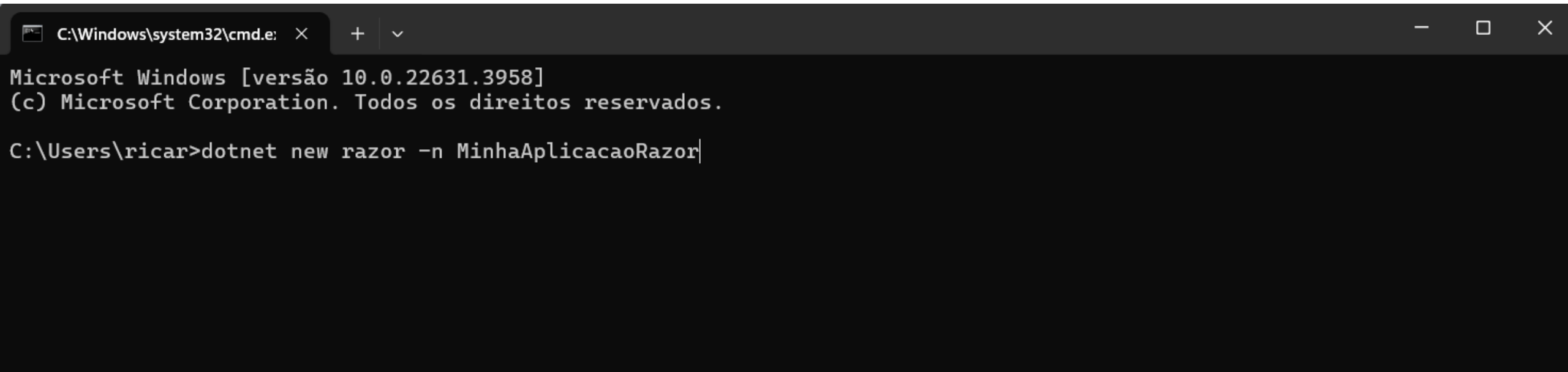
Componentes

- Middleware e Pipeline de Solicitação
 - ASP.NET Core utiliza um conceito de middleware e pipeline de solicitação para processar solicitações HTTP. Diversos middlewares podem ser configurados para executar tarefas específicas, como autenticação, autorização e logging.
- Git
 - O controle de versão é crucial no desenvolvimento de software. O Git é amplamente utilizado para controle de versão, permitindo colaboração eficiente e rastreamento de alterações.
- Implantação e Hospedagem: Azure, AWS, Heroku, etc
 - Para implantar e hospedar sua aplicação web, existem várias opções, incluindo serviços em nuvem como Azure e AWS, além de plataformas como Heroku.

Componentes

- Testes Automatizados: Xunit, NUnit, MSTest
 - A escrita de testes automatizados é uma prática recomendada. Diversos frameworks de teste, como Xunit, NUnit e MSTest, são compatíveis com o ecossistema C#.
- Segurança: ASP.NET Identity, OAuth, JWT
 - Para gerenciamento de identidade e autenticação, o ASP.NET Identity é comumente usado. Além disso, OAuth e JWT (JSON Web Tokens) são frequentemente utilizados para autenticação em APIs.

Desenvolvendo uma aplicação Razor Pages com ASP.NET



```
C:\Windows\system32\cmd.e: X + v
Microsoft Windows [versão 10.0.22631.3958]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\ricar>dotnet new razor -n MinhaAplicacaoRazor|
```


Desenvolvendo uma aplicação Razor Pages com ASP.NET

```
C:\Windows\system32\cmd.e: X + v
C:\Users\ricar>dotnet new razor -n MinhaAplicacaoRazor

Bem-vindo(a) ao .NET 8.0.
-----
Versão do SDK: 8.0.303

Telemetria
-----
As ferramentas do .NET coletam dados de uso para ajudar-nos a aprimorar sua experiência. Eles são coletados pela Microsoft e compartilhados com a comunidade. Você pode recusar a telemetria definindo a variável de ambiente DOTNET_CLI_TELEMETRY_OPTOUT como '1' ou 'true' usando seu shell favorito.

Leia mais sobre a telemetria das Ferramentas da CLI do .NET: https://aka.ms/dotnet-cli-telemetry

-----
Instalou um certificado de desenvolvimento ASP.NET Core HTTPS.
Para confiar no certificado, execute 'dotnet dev-certs https --trust'
Saiba mais HTTPS: https://aka.ms/dotnet-https

-----
Escreva seu primeiro aplicativo: https://aka.ms/dotnet-hello-world
Descubra o que há de novo: https://aka.ms/dotnet-whats-new
Explorar a documentação: https://aka.ms/dotnet-docs
Relate problemas e encontre a fonte no GitHub: https://github.com/dotnet/core
Use 'dotnet --help' para ver os comandos disponíveis ou visite: https://aka.ms/dotnet-cli
-----
O modelo "Aplicativo Web ASP.NET Core" foi criado com êxito.
Este modelo contém tecnologias de outras pessoas além da Microsoft, consulte https://aka.ms/aspnetcore/8.0-third-party-notices para obter detalhes.

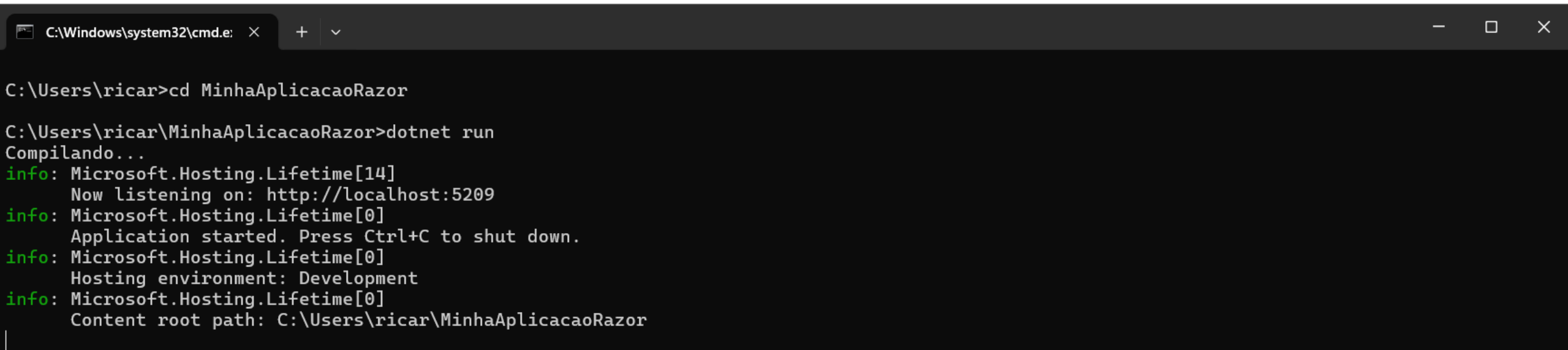
Processando ações pós-criação...
Restaurando C:\Users\ricar\MinhaAplicacaoRazor\MinhaAplicacaoRazor.csproj:
  Determining projects to restore...
  C:\Users\ricar\MinhaAplicacaoRazor\MinhaAplicacaoRazor.csproj restaurado (em 89 ms).
A restauração foi bem-sucedida.

C:\Users\ricar>
```

Desenvolvendo uma aplicação Razor Pages com ASP.NET

Nome	Data de modificação	Tipo	Tamanho
obj	30/07/2024 17:16	Pasta de arquivos	
Pages	30/07/2024 17:16	Pasta de arquivos	
Properties	30/07/2024 17:16	Pasta de arquivos	
wwwroot	30/07/2024 17:16	Pasta de arquivos	
appsettings.Development.json	30/07/2024 17:16	Arquivo JSON	1 KB
appsettings.json	30/07/2024 17:16	Arquivo JSON	1 KB
MinhaAplicacaoRazor.csproj	30/07/2024 17:16	C# Project File	1 KB
Program.cs	30/07/2024 17:16	C# Source File	1 KB

Desenvolvendo uma aplicação Razor Pages com ASP.NET



```
C:\Windows\system32\cmd.e: X + v

C:\Users\ricar>cd MinhaAplicacaoRazor

C:\Users\ricar\MinhaAplicacaoRazor>dotnet run
Compilando...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5209
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\ricar\MinhaAplicacaoRazor
```

Desenvolvendo uma aplicação Razor Pages com ASP.NET



MinhaAplicacaoRazor Home Privacy

Welcome

Learn about [building Web apps with ASP.NET Core](#).

Anatomia de uma Aplicação ASP.NET Web

- Uma aplicação ASP.NET Web é composta por vários arquivos e pastas que trabalham juntos para criar uma aplicação funcional.
- Compreender a estrutura desses arquivos é crucial para o desenvolvimento eficiente.

Anatomia de uma Aplicação ASP.NET Web

- Pages: Contém arquivos .cshtml e seus arquivos de modelo de página correspondentes .cshtml.cs.
- wwwroot: Contém arquivos estáticos, como CSS, JavaScript e imagens.
- appsettings.json: Arquivo de configuração para a aplicação.
- Startup.cs: Configura os serviços e o pipeline de solicitação HTTP da aplicação.

Startup.cs nas versões mais atuais

- Na versão mais recente do ASP.NET Core, o arquivo Startup.cs foi substituído pelo uso de um arquivo Program.cs simplificado.
- Isso faz parte de uma iniciativa para simplificar a inicialização e configuração de aplicativos ASP.NET Core.

Privacy.cshtml Index.cshtml

C# PrimeiroRazorApp

```
1 @page
2 @model PrivacyModel
3 @{
4     ViewData["Title"] = "Política de privacidade";
5 }
6 <h1>@ViewData["Title"]</h1>
7
8 <p>Use esta página para detlhar a política de privacidade do seu site.</p>
9
```

Gerenciador de Soluções

Pesquisar em Gerenciador de Soluções (Ctrl 🔍)

Solução 'PrimeiroRazorApp' (1 de 1 projeto)

PrimeiroRazorApp

Connected Services

Dependências

Analísadores

Estruturas

Properties

wwwroot

Pages

Shared

_ViewImports.cshtml

_ViewStart.cshtml

Error.cshtml

Index.cshtml

Privacy.cshtml

appsettings.json

C# Program.cs


```
Privacy.cshtml  Index.cshtml X
PrimeiroRazorApp
1  @page
2  @model IndexModel
3  @{
4      ViewData["Title"] = "Home page";
5  }
6
7  <div class="text-center">
8      <h1 class="display-4">Bem vindo</h1>
9      <p>Leia sobre <a href="https://learn.microsoft.com/aspnet/core">construindo Web apps com ASP.NET Core</p>
10 </div>
11
```

Gerenciador de Soluções

Pesquisar em Gerenciador de Soluções (Ctrl F)

- Solução 'PrimeiroRazorApp' (1 de 1 projeto)
 - PrimeiroRazorApp
 - Connected Services
 - Dependências
 - Analísadores
 - Estruturas
 - Properties
 - wwwroot
 - Pages
 - Shared
 - _ViewImports.cshtml
 - _ViewStart.cshtml
 - Error.cshtml
 - Index.cshtml
 - Privacy.cshtml
 - appsettings.json
 - C# Program.cs

Index.cshtml X Index.cshtml.cs X

PrimeiroRazorApp PrimeiroRazorApp.Pages.IndexModel _logger

```
1 using Microsoft.AspNetCore.Mvc;
2 using Microsoft.AspNetCore.Mvc.RazorPages;
3
4 namespace PrimeiroRazorApp.Pages;
5
6 public class IndexModel : PageModel
7 {
8     private readonly ILogger<IndexModel> _logger;
9
10    public IndexModel(ILogger<IndexModel> logger)
11    {
12        _logger = logger;
13    }
14
15    public void OnGet()
16    {
17    }
18 }
19
20
```

115 % Não foi encontrado nenhum problema Ln: 1 Car: 1 SPC CRLF

Gerenciador de Soluções

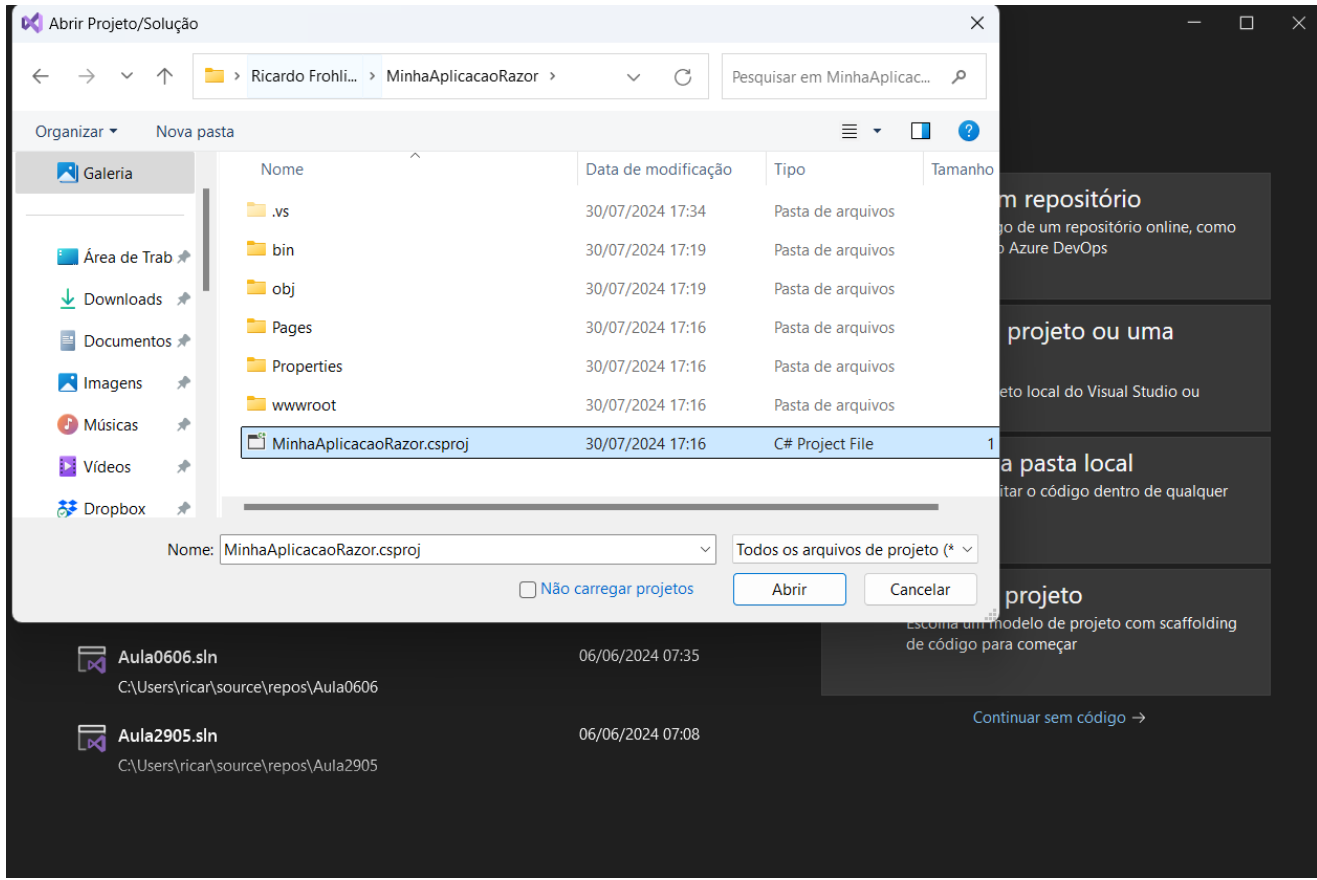
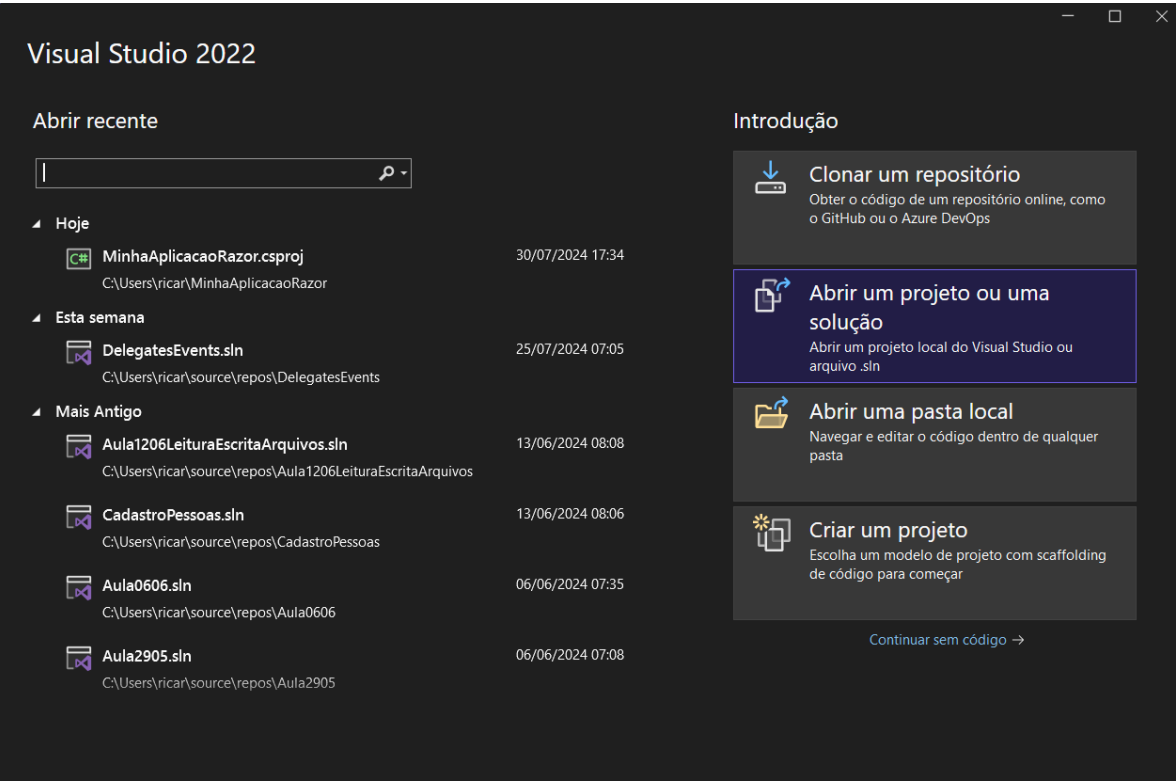
Pesquisar em Gerenciador de Soluções (Ctrl F)

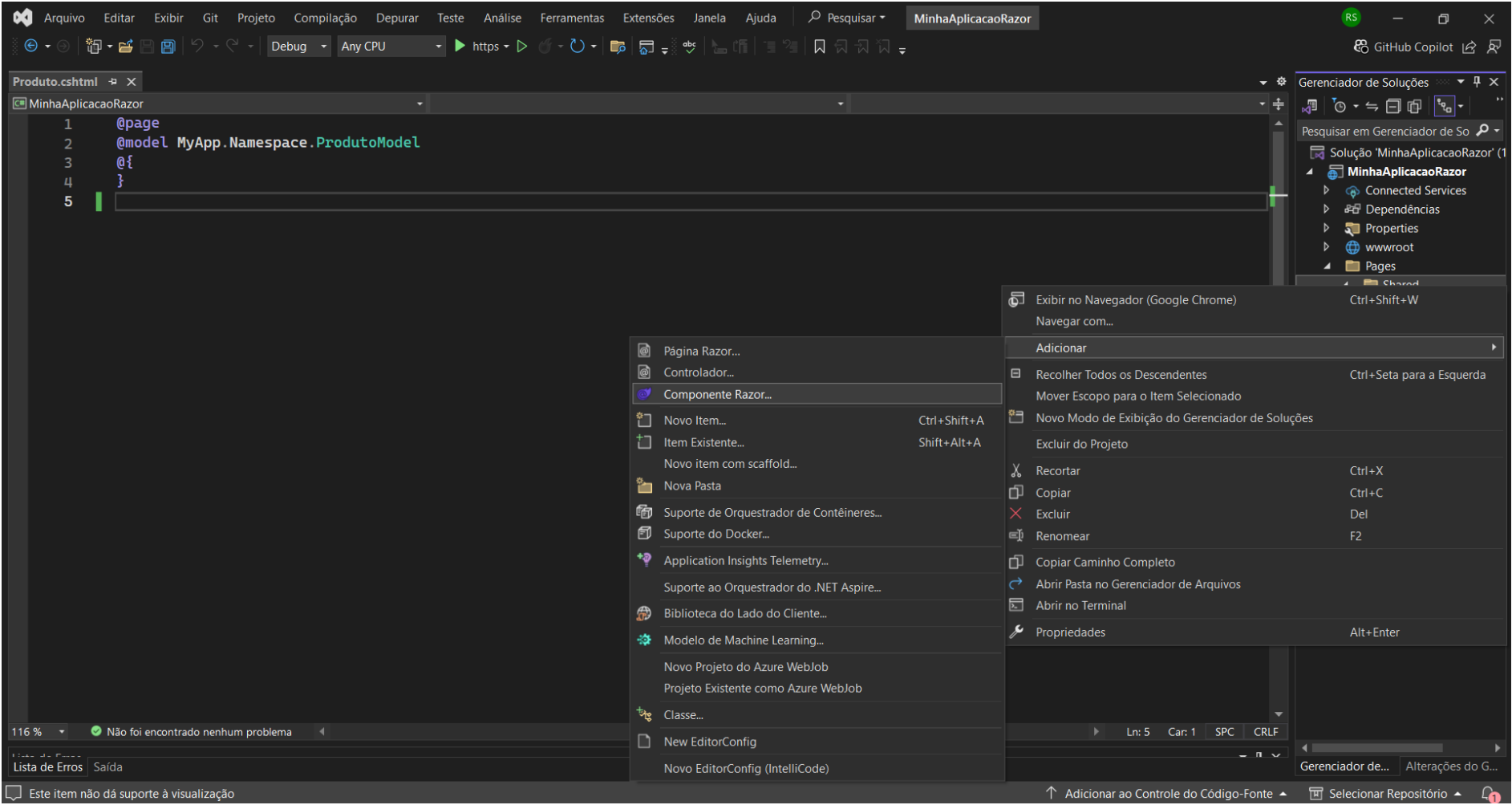
Solução 'PrimeiroRazorApp' (1 de 1 projeto)

- PrimeiroRazorApp
 - Connected Services
 - Dependências
 - Analísadores
 - Estruturas
 - Properties
 - wwwroot
 - Pages
 - Shared
 - _Layout.cshtml
 - _ValidationScriptsPartial.cshtml
 - _ViewImports.cshtml
 - _ViewStart.cshtml
 - Error.cshtml
 - C# Error.cshtml.cs
 - Index.cshtml
 - C# Index.cshtml.cs
 - Privacy.cshtml
 - C# Privacy.cshtml.cs
 - appsettings.json
 - C# Program.cs

```
C:\Users\ricar\MinhaAplicacaoRazor>dotnet new page -n Produto
O modelo "Página Razor" foi criado com êxito.
```

```
C:\Users\ricar\MinhaAplicacaoRazor>|
```



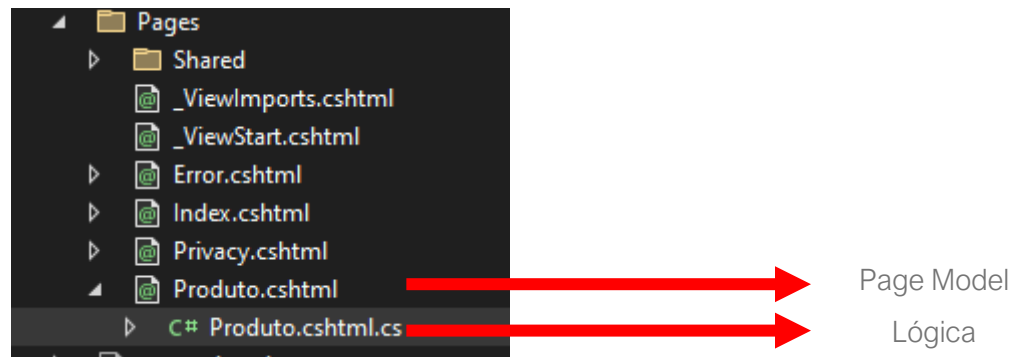


```
4  namespace PrimeiroRazorApp.Pages
5  {
6      public class ProdutoModel : PageModel
7      {
8          public string Descricao { get; set; }
9          public decimal Preco { get; set; }
10
11         public void OnGet()
12         {
13             //Lógica para a requisição GET
14             Descricao = "Coca cola";
15             Preco = 8.99m;
16         }
17     }
18 }
```

```
1  @page
2  @model PrimeiroRazorApp.Pages.ProdutoModel
3  @{
4      ViewData["Title"] = "Cadastro de produtos";
5      ViewData["Teste"] = "Estou aqui testando esta forma de dados";
6  }
7
8  <h2>@ViewData["Title"]</h2>
9  <p>@ViewData["Teste"]</p>
10 <p>Estou testando os dados da minha página</p>
11 <p>Nome do produto: @Model.Descricao</p>
12 <p>Preço: @Model.Preco</p>
```

Adicionar Dados a uma Página Utilizando Page Model

- O Page Model em Razor Pages é a classe de modelo que acompanha cada página Razor.
- Ele encapsula a lógica da página, permitindo a manipulação de dados e interação com o backend.





Razor APP Home Privacidade

Cadastro de produtos

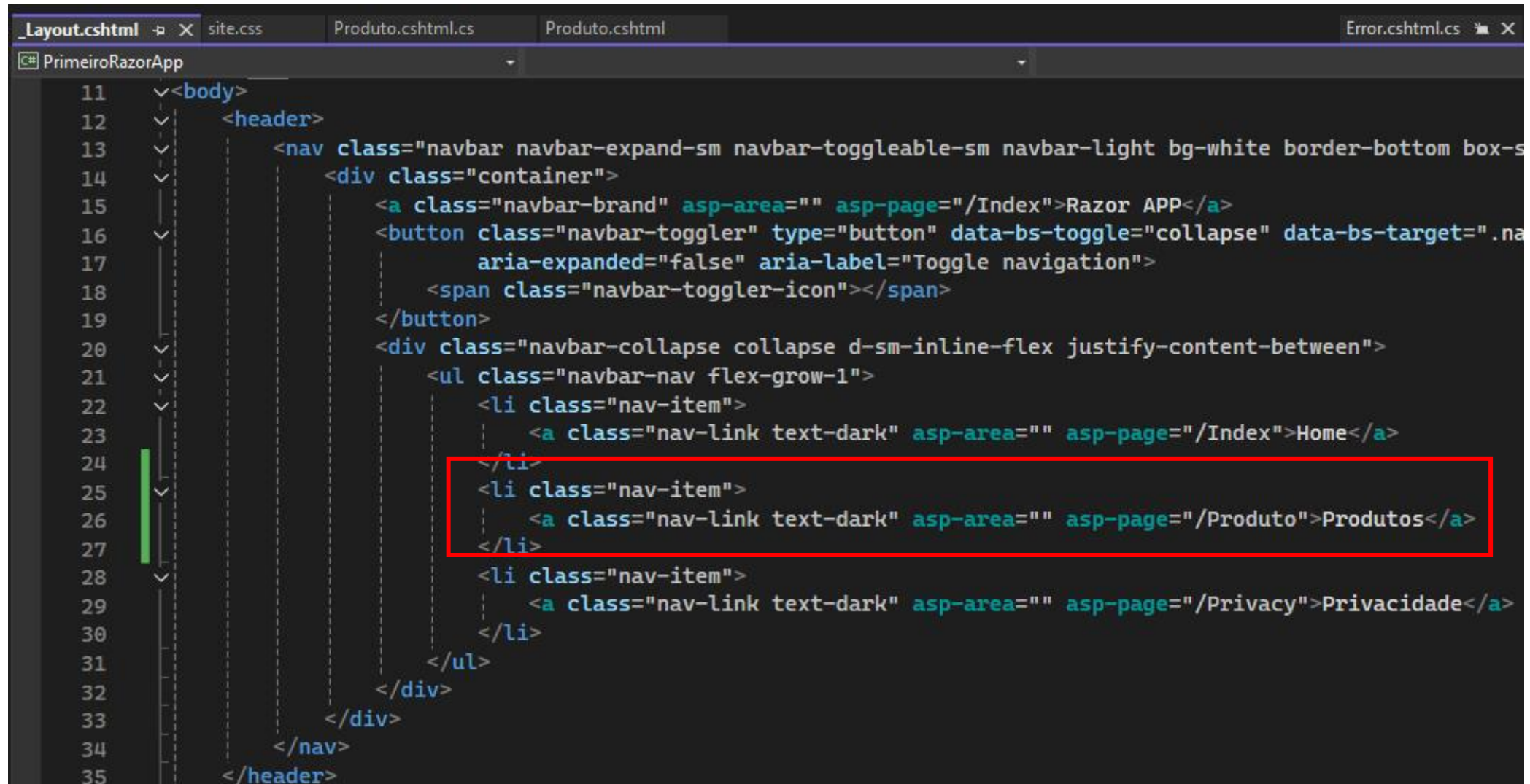
Estou aqui testando esta forma de dados

Estou testando os dados da minha página

Nome do produto: Coca cola

Preço: 8,99

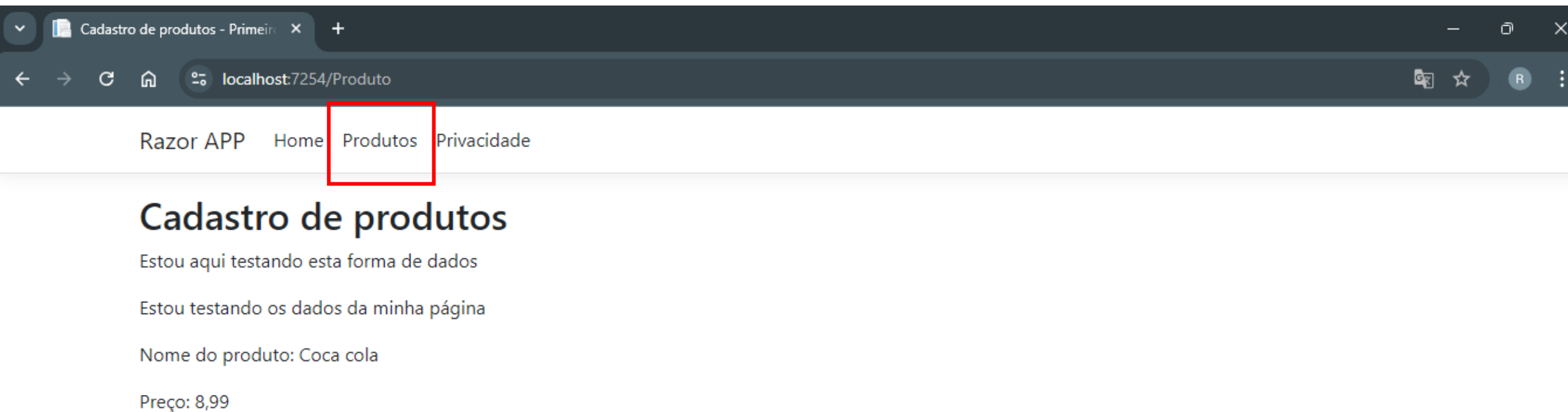
Adicionando um item no NavBar no arquivo _Layout.cshtml



The screenshot shows the Visual Studio IDE with the file explorer on the left displaying the project structure: `PrimeiroRazorApp`. The main editor window shows the `_Layout.cshtml` file. The code is a Razor view that defines the layout of the application, including a header section with a navigation bar. The navigation bar is a Bootstrap navbar with a toggle button and a list of links. A new link, `Produtos`, has been added to the list, highlighted by a red rectangle. The code is as follows:

```
11 <body>
12 <header>
13 <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-s
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=".na
17 <span class="navbar-toggler-icon"></span>
18 </button>
19 <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
20 <ul class="navbar-nav flex-grow-1">
21 <li class="nav-item">
22 <a class="nav-link text-dark" asp-area="" asp-page="/Index">Home</a>
23 </li>
24 <li class="nav-item">
25 <a class="nav-link text-dark" asp-area="" asp-page="/Produto">Produtos</a>
26 </li>
27 <li class="nav-item">
28 <a class="nav-link text-dark" asp-area="" asp-page="/Privacy">Privacidade</a>
29 </li>
30 </ul>
31 </div>
32 </div>
33 </nav>
34 </header>
35
```

Item adicionado no NavBar



Produto.cshtml.cs

```
public class ProdutoModel : PageModel
{
    public List<Produto> Produtos { get; set; }

    public void OnGet()
    {
        //Lógica para a requisição GET
        Produtos = new List<Produto>
        {
            new Produto { Descricao = "Coca cola", Preco = 8.99m },
            new Produto { Descricao = "Pepsi cola", Preco = 6.99m },
            new Produto { Descricao = "Feijão", Preco = 3.49m },
            new Produto { Descricao = "Arroz", Preco = 4.97m },
            new Produto { Descricao = "Carne moída", Preco = 23.50m }
        };
    }
}

public class Produto
{
    public string Descricao { get; set; }
    public decimal Preco { get; set; }
}
```

Produto.cshtml

```
1  @page
2  @model PrimeiroRazorApp.Pages.ProdutoModel
3  @{
4      ViewData["Title"] = "Cadastro de produtos";
5  }
6
7  <h2>@ViewData["Title"]</h2>
8  <table>
9      <thead>
10         <tr>
11             <th>Nome do produto</th>
12             <th>Preço</th>
13         </tr>
14     </thead>
15     <tbody>
16         @foreach(var produto in Model.Produtos){
17             <tr>
18                 <td>@produto.Descricao</td>
19                 <td>@produto.Preco.ToString("C")</td>
20             </tr>
21         }
22     </tbody>
23 </table>
```

- A estrutura do projeto está assim.
- E para acessar cada arquivo .cshtml, basta acessar o /NomeDoArquivo, por exemplo,

