

A Página Carrossel

Transcrição

Vamos dar início à criação da solução para o nosso cliente Casa do Código. Para isso, precisaremos criar uma aplicação com um aspecto profissional, portanto, será necessário um bom layout. Sendo assim, utilizaremos o design responsivo, ou seja, a aplicação terá que rodar bem tanto no desktop, quanto em tablet e celular.

Entretanto, este curso não tem o intuito de ensinar a desenhar layouts, isso fica a cargo do web designer. Ele nos fornecerá o conteúdo para montarmos a página, já finalizado. Caso queira se aprofundar nisso, na Alura, temos o curso de [Bootstrap: criação de uma single-page responsiva](https://www.alura.com.br/curso-online-bootstrap-criacao-single-page-responsiva) (<https://www.alura.com.br/curso-online-bootstrap-criacao-single-page-responsiva>), desenvolvido pelo instrutor Yuri Padilha.

Continuando, temos algumas páginas e imagens fornecidas pela web designer, para montarmos nossa aplicação. Uma delas é o catálogo, também chamado de carrossel, que exibe os produtos da casa do código, no caso os livros, que ficarão passando pela tela, emulando um carrossel.

A próxima página é a de carrinho. Conforme o cliente escolhe um produto, ele é direcionado para a página de carrinho, para que possa visualizar a compra atual. Feito isso, pode retornar ao carrossel para procurar mais produtos, até finalizar a compra. Neste ponto, o cliente completa um cadastro e então é direcionado à página "resumo do pedido".

Todas as páginas já foram montadas pela web designer, que forneceu os arquivos que serão inseridos na solução. Retornaremos ao Visual Studio, onde temos uma página chamada "Recursos", com as subpastas "css", "dados", "logos", "produtos" e "views-com-bootstrap".

Copiaremos estes arquivos para o nosso projeto, em pastas específicas. Precisamos ter cuidado com o destino dos arquivos; caso contrário, podemos ter um problema em nossa aplicação. Por exemplo, na pasta "css" temos o arquivo `site.css`, correspondente à folha de estilos, ou como desenhar a tela, cores, posicionamento, entre outras informações de layout.

Copiaremos este arquivo e, com cuidado, localizaremos dentro do projeto `CasaDoCodigo` uma pasta chamada "wwwroot" e, dentro dela, a pasta "css", onde colaremos o arquivo que acabamos de copiar. O Visual Studio pedirá uma confirmação. Basta clicarmos em "Sim".

Além disso, temos uma pasta chamada "logos", que contém o logotipo da Casa do Código. Copiaremos o arquivo `logo.png`, localizaremos a pasta "wwwroot > images" e colaremos o arquivo nela.

A seguir, abriremos a pasta "produtos" e notaremos que ela contém uma série de imagens dos produtos disponíveis na loja. Em "wwwroot", dentro da pasta "images", criaremos, uma subpasta chamada "produtos".

O nome da pasta precisa, necessariamente, estar em letras minúsculas.

Selecionaremos os arquivos de imagens que vimos anteriormente, que totalizam 65 arquivos. Os copiaremos e colaremos na pasta "produtos", que acabamos de criar.

Ainda temos uma pasta chamada "views-com-bootstrap", com os arquivos `Cadastro.cshtml`, `Carrinho.cshtml`, `Carrossel.cshtml` e `Resumo.cshtml`. Primeiro, localizaremos no projeto `CasaDoCodigo` a pasta "Views". Nela, encontramos as subpastas "Home" e "Shared". Criaremos, neste mesmo nível hierárquico, uma nova pasta chamada "Pedido".

Copiaremos os quatro arquivos `.cshtml` , `Cadastro.cshtml` , `Carrinho.cshtml` , `Carrossel.cshtml` , `Resumo.cshtml` , e colaremos nesta página "Pedido" que acabamos de criar.

Por fim, copiaremos o arquivo `_Layout.cshtml` , e colaremos na pasta "CasaDoCodigo > Views > Shared". Ao fazermos isso, o Visual Studio pedirá uma confirmação. Basta confirmarmos. Rodaremos a aplicação, com a tecla de atalho "F5".

Caímos novamente em nossa página inicial, mas queremos acessar as páginas novas que foram criadas. Para isso, acessaremos a URL `localhost:58246/pedido/carrossel` , mas quando fazemos isso o ASP.NET Core MVC não consegue encontrar a página. Por quê?

Isso aconteceu porque temos que passar primeiro pelo intermediário — que é o controller — para acessarmos uma view. Só que, neste caso, não temos o controller de pedido, pois ele ainda não foi criado.

Retornaremos ao Visual Studio, pararemos a aplicação e, dentro da página "Controllers", faremos esta adição. Criaremos uma nova classe, nesta pasta, chamada `PedidoController` , que herdará da classe base `Controller` :

```
namespace CasaDoCodigo.Controllers
{
    public class PedidoController : Controller
    {
    }
}
```

Para buscarmos a referência, basta utilizarmos a atalho "Ctrl + ." sobre a palavra `Controller` . A seguir, criaremos os métodos, que aqui correspondem às *actions*, ou seja, **cadastro**, **carrinho**, **carrossel** e **resumo**. Esse método retornará o resultado de uma *action*, ou seja, um `ActionResult` . No bloco, teremos a respectiva view:

```
namespace CasaDoCodigo.Controllers
{
    public class PedidoController : Controller
    {
        public IActionResult Carrossel()
        {
            return View();
        }
    }
}
```

Rodaremos a aplicação novamente, com o atalho "F5". Acessaremos o mesmo endereço, `localhost:58246/pedido/carrossel` e, agora sim, temos a visualização do nosso carrossel, com os produtos da Casa Do Código, no caso, os livros que são oferecidos aos clientes.