03

Salvando item de pedido

Transcrição

Agora que estamos rodando a aplicação, veremos se a *action* de carrinho está conseguindo obter o novo pedido. Adicionaremos um produto qualquer, cairemos na *action* de carrinho, que por sua vez chamará o método `GetPedido()`. Ele criará um pedido, caso seja necessário, e chegaremos em `SaveChanges()`, onde o novo pedido será salvo.

Ao pressionarmos a tecla "F5", voltaremos à *action* do carrinho com o pedido criado em `return View(pedido.Itens)`, o qual possui `id 4010`. Apertaremos "F5" novamente para atualizarmos a página, e o pedido será obtido mais uma vez, porém ele é recriado desnecessariamente; o certo seria mantermos o mesmo pedido por meio da sessão.

O problema é que não estamos gravando o `id` do pedido na sessão, portanto pausaremos a aplicação e, logo após o `contexto.SaveChanges()`, responsável pela gravação no banco de dados e obtenção do novo `id` para o pedido, pegaremos o `PedidoId` e o salvaremos na sessão usando o método `SetPedidoId()`.

Com o código abaixo, gravaremos na sessão e garantimos que, a cada navegação, sempre reaproveitaremos o mesmo `id`.

```
if (pedido == null)
{
    pedido = new Pedido();
    dbSet.Add(pedido);
    contexto.SaveChanges();
    SetPedidoId(pedido.Id);
}
```

Vamos voltar a `PedidoController.cs` e modificar a *action* para passarmos o código do produto selecionado no carrossel como argumento. Adicionaremos `codigo` ao nosso carrinho, mas antes disto precisaremos verificar se ele foi preenchido ou não, usando uma condição:

```
public IActionResult Carrinho(string codigo)
{
    if (!string.IsNullOrEmpty(codigo))
    {
        pedidoRepository.AddItem(codigo);
    }

    Pedido pedido = pedidoRepository.GetPedido();
    return View(pedido.Itens);
}
```

O método `AddItem()` ainda não existe, então vamos criá-lo; deixaremos o mouse sobre ele, usaremos "Ctrl + ." e escolheremos "Gerar método 'IPedidoRepository.AddItem'". Em seguida, navegaremos ao novo método pressionando "F12" no teclado.

Em `PedidoRepository.cs`, temos a interface `IPedidoRepository`, e então criaremos uma implementação concreta deste método na classe `PedidoRepository`, clicando no nome da interface (`IPedidoRepository`), usando "Ctrl + ." e "Implementar interface". Então, faremos as verificações, começando pela existência ou não do código.

Declararemos uma variável de `produto` para obtermos o produto que possui o código correspondente no DbSet de produtos. Em seguida, faremos uma consulta com o filtro `Where()` e chamaremos o método `SingleOrDefault()` para verificarmos se existe algum produto com o código.

Caso não exista, lançaremos uma exceção, caso contrário pegaremos o pedido com uma variável local, `GetPedido()`, e em seguida verificaremos se o item já existe para este pedido em específico, com uma consulta simples no DbSet de `ItemPedido`.

Se o `itemPedido` não for encontrado, é necessário adicioná-lo ao carrinho. Sendo assim, criaremos outro `if`, com outro método para o adicionarmos. No entanto, para isto precisaremos criar uma instância de `itemPedido`, e no construtor passaremos como parâmetros `pedido`, `produto`, a quantidade, que inicialmente é `1`, e o preço unitário, que vem de `produto.Preco`.

Chamaremos o `contexto.SaveChanges()` para podermos gravar isso no banco de dados. O código ficará da seguinte maneira:

```
public void AddItem(string codigo)
{
    var produto = contexto.Set<Produto>()
                        .Where(p => p.Codigo == codigo)
                        .SingleOrDefault();

    if (produto == null)
    {
        throw new ArgumentException("Produto não encontrado");
    }

    var pedido = GetPedido();

    var itemPedido = contexto.Set<ItemPedido>()
                        .Where(i => i.Produto.Codigo == codigo
                            && i.Pedido.Id == pedido.Id)
                        .SingleOrDefault();

    if (itemPedido == null)
    {
        itemPedido = new ItemPedido(pedido, produto, 1, produto.Preco);
        contexto.Set<ItemPedido>()
                        .Add(itemPedido);

        contexto.SaveChanges();
    }
}
```