<table class="table table-striped table-bordered">

<?php

foreach($produtos as $produto) :

?>

<tr>

<td><?= $produto['nome'] ?></td>

<td><?= $produto['preco'] ?></td>

<td>

<a href="remove-produto.php" class="text-danger">remover</a>

</td>

</tr>

<?php

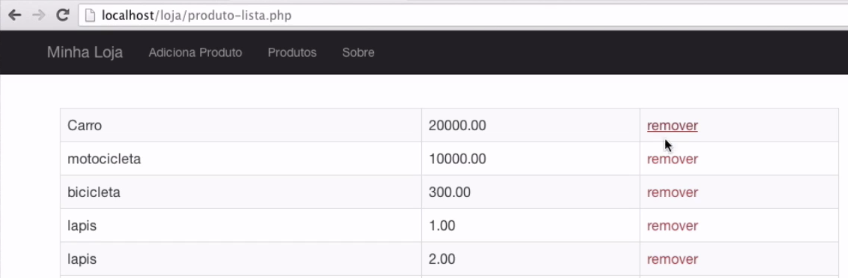
endforeach

?>

</table>

O link está aparecendo certo na listagem. Mas ainda precisamos passar uma informação para esse link: qual produto eu quero remover? A melhor maneira de fazer isso é passando o ID do produto; afinal o ID identifica exclusivamente aquele produto em relação aos outros. Vamos passar o parâmetro pela própria URL, igual fizemos no primeiro capítulo:

<a href="remove-produto.php?id=<?=$produto['id']?>" class="text-danger">remover</a>



Agora, veja no browser, que os links são diferentes para cada produto. Com o link pronto, precisamos implementar o remove-produto.php. O corpo dele é igual o corpo das outras páginas, onde precisamos incluir as várias sub-páginas que criamos:

<?php

include("cabecalho.php");

include("conecta.php");

include("banco-produto.php");

include("rodape.php");

?>

Vamos criar a query que deleta um produto. Já sabemos onde colocar, no banco-produto.php. Ele receberá a conexão e o id a ser deletado:

function removeProduto($conexao, $id) {

$query = "delete from produtos where id = {$id}";

return mysqli\_query($conexao, $query);

}

De volta ao remove-produto.php, vamos agora invocar esse método criado. Basta pegar o id que veio como parâmetro, e passar para o método:

<?php

include("cabecalho.php");

include("conecta.php");

include("banco-produto.php");

$id = $\_GET['id'];

removeProduto($conexao, $id);

?>

<p class="text-success">Produto <?=$id?> removido!</p>

<?php

include("rodape.php");

?>

Legal. A remoção funciona. Mas a parte chata é que toda vez que deletamos, vamos pra página de sucesso, e precisamos voltar para a listagem. E o pior, quando o usuário usa o botão "back" do navegador, ele não atualiza a página; ele mostra a página antiga, que ele armazenou.

Vamos fazer então que, quando clicar em remover, o usuário seja automaticamente redirecionado para a listagem. Para fazer isso, precisamos entender um pouco mais sobre o HTTP. Geralmente toda resposta contém um corpo em HTML, que é o que o browser mostra pro usuário final. Mas, podemos responder outras coisas, e elas não precisam obrigatoriamente ter HTML. Podemos falar pro browser que o que ele deve fazer na verdade é acessar um outro endereço. E é assim que fazemos esse redirecionamento. Ao invés de devolver HTML, devolveremos essa instrução pro navegador. Ele, ao receber, faz uma nova requisição, sem que o usuário perceba.

Por padrão, quando uma requisição é bem sucedida, o servidor devolve um código (um status) avisando o navegador que está tudo certo. Quando tudo dá certo, o código é 200. Quando você tenta acessar um endereço que não existe, o código é 404 (Page Not Found). Erro no servidor? 500. Redirecionamento? 302. Existem vários outros códigos de status, mas esses são os mais importantes nesse momento.

Em código, vamos fazer o PHP devolver 302 para o browser:

<?php

include("conecta.php");

include("banco-produto.php");

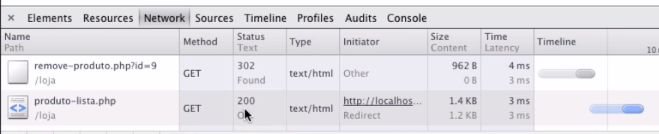
$id = $\_GET['id'];

removeProduto($conexao, $id);

header("Location: produto-lista.php");

?>

Veja que colocamos um header, cuja chave é Location e o valor é a página que queremos que o browser redirecione automaticamente. Podemos ver que ele voltou 302, olhando a aba Network do Chrome Developer Tools. Lá, vemos que a requisição para o remove-produto.php devolve 302. Na mesma hora, o browser faz uma outra requisição, para a listagem.



Para garantir que a linha última executada é o header, podemos pedir para o PHP parar de processar ali mesmo, usando o método die():

<?php

include("conecta.php");

include("banco-produto.php");

$id = $\_GET['id'];

removeProduto($conexao, $id);

header("Location: produto-lista.php");

die();

Mas só redirecionar para a listagem não é suficiente. Precisamos exibir uma mensagem amigável na própria listagem. Vamos colocar a mensagem na listagem, lá em produto-lista.php:

<p class="alert-success">Produto apagado com sucesso.</p>

Mas essa mensagem não pode aparecer sempre; mas sim quando veio de uma deleção. Precisamos avisar a listagem então, que o comando anterior foi o de deleção. Vamos passar um parâmetro para a página, informando isso então, no próprio header:

header("Location: produto-lista.php?removido=true");

Agora, basta fazer o if na listagem:

<?php if(array\_key\_exists("removido", $\_GET) && $\_GET['removido']=='true') { ?>

<p class="alert-success">Produto apagado com sucesso.</p>

<?php } ?>

Agora sim, a mensagem aparece na hora certa! Nossa aplicação está evoluindo.

Agora faça o teste, passe o "true" sem aspas (passando o booleando true). Tudo continua funcionando. Por que? Por que o == do PHP consegue fazer a conversão dos tipos de maneira automática e descobre que eles são iguais. Se não quisermos que o PHP faça conversão de tipos (pois isso pode ser problemático em alguns casos), podemos fazer uso dos ===.