**Alterando produtos**

Vamos agora fazer o formulário de alteração de um produto. Para fazer essa funcionalidade, precisamos usar tudo que aprendemos até aqui, exibir formulário, criar um link, executar a consulta de atualização, mostrar a mensagem de sucesso, e etc.

Vamos começar adicionando mais um botão na listagem, que dará a opção para o usuário "alterar" o produto:

<tr>

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

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

<td><?= substr($produto['descricao'],0,40) ?></td>

<td><a href="produto-altera-formulario.php">alterar</a>

<td>

<form action="remove-produto?id=<?=$produto['id']?>" method="post">

<button class="btn btn-danger">remover</button>

</form>

</td>

</tr>

O link já apareceu, mas ainda está bem feio. Vamos melhorar o layout desse link:

<td><a class="btn btn-primary" href="produto-altera-formulario.php">alterar</a>

Agora, precisamos passar o ID para esse link, igual fizemos na exclusão. Vamos passar isso pela URL, já que estamos fazendo uma requisição do tipo GET:

<td><a class="btn btn-primary" href="produto-altera-formulario.php?id=<?=$produto['id']?>">alterar</a>

Ótimo. Com o link funcionando, vamos criar o formulário produto-altera-formulario.php. Ele é bem parecido com o formulário de adição, então vamos copiar e colar a estrutura principal:

<?php include("cabecalho.php");

include("conecta.php");

include("banco-categoria.php");

$categorias = listaCategorias($conexao);

?>

<h1>Formulário de produto</h1>

<form action="adiciona-produto.php" method="post">

<table class="table">

<tr>

<td>Nome</td>

<td> <input class="form-control" type="text" name="nome"></td>

</tr>

<tr>

<td>Preço</td>

<td><input class="form-control" type="number" name="preco"></td>

</tr>

<tr>

<td>Descrição</td>

<td><textarea class="form-control" name="descricao"></textarea></td>

</tr>

<tr>

<td></td>

<td><input type="checkbox" name="usado" value="true"> Usado

</tr>

<tr>

<td>Categoria</td>

<td>

<select name="categoria\_id" class="form-control">

<?php foreach($categorias as $categoria) : ?>

<option value="<?=$categoria['id']?>">

<?=$categoria['nome']?>

</option>

<?php endforeach ?>

</select>

</td>

</tr>

<tr>

<td>

<button class="btn btn-primary" type="submit">Cadastrar</button>

</td>

</tr>

</table>

</form>

<?php include("rodape.php"); ?>

Apesar de parecidos, eles tem pequenas diferenças, que é o que vamos atacar agora. Vamos mudar o título:

<h1>Alterando produto</h1>

Vamos mudar a ação do formulário:

<form action="altera-produto.php" method="post">

Vamos mudar o nome do botão:

<button class="btn btn-primary" type="submit">Alterar</button>

Se entrarmos no browser, a página funciona. Mas tem algo estranho: em um formulário de alteração, geralmente o formulário vem com as informações atuais daquele produto, para que o usuário final apenas altere os valores que desejar.

Para isso funcionar, a primeira coisa que precisamos fazer é buscar o produto no banco de dados, para começar a exibir nos campos do formulário. Então, já que temos o ID do produto que foi passado pela URL, no começo da página, fazemos:

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

$produto = buscaProduto($conexao, $id);

// aqui continua normal

$categorias = listaCategorias($conexao);

Com essa variável produto, que veio do banco de dados, podemos exibir os dados no começo do formulário.

<tr>

<td>Nome</td>

<td> <input class="form-control" type="text" name="nome" value="<?=$produto['nome']?>"></td>

</tr>

<tr>

<td>Preço</td>

<td><input class="form-control" type="number" name="preco" value="<?=$produto['preco']?>"></td>

</tr>

<tr>

<td>Descrição</td>

<td><textarea class="form-control" name="descricao"><?=$produto['descricao']?></textarea></td>

</tr>

Antes de continuarmos, vamos escrever a função buscaProduto no nosso banco-produto.php. Basta fazermos a query e retornar o fetch do primeiro elemento (afinal, só tem um):

function buscaProduto($conexao, $id) {

$query = "select \* from produtos where id = {$id}";

$resultado = mysqli\_query($conexao, $query);

return mysqli\_fetch\_assoc($resultado);

}

Não esqueça também de incluir esse arquivo no formulário de alteração:

include("banco-produto.php");

Se abrirmos agora a página no browser, alguns valores já aparecem. Precisamos agora preencher o resto. No checkbox, caso o produto seja usado, precisamos escrever checked=checked. Vamos então criar uma variável que, caso o produto seja usado, ele preencha com o valor que precisamos:

<?php

$usado = $produto['usado'] ? "checked='checked'" : "";

?>

<tr>

<td></td>

<td><input type="checkbox" name="usado" <?=$usado?> value="true"> Usado

</tr>

Ótimo. Isso está funcionando! Repare no ?, essa é uma maneira de fazer um "if" numa única linha. Se a condição for verdadeira, ele retorna o primeiro valor; se for falsa, retorna o segundo valor.

Vamos agora trabalhar na categoria. Precisamos selecionar a categoria certa. A solução é análoga: no option certo, precisamos escrever selected="selected". Para isso, vamos verificar se o id da categoria do produto é igual a categoria que está dentro do loop:

<tr>

<td>Categoria</td>

<td>

<select name="categoria\_id" class="form-control">

<?php

foreach($categorias as $categoria) :

$essaEhACategoria = $produto['categoria\_id'] == $categoria['id'];

$selecao = $essaEhACategoria ? "selected='selected'" : "";

?>

<option value="<?=$categoria['id']?>" <?=$selecao?>>

<?=$categoria['nome']?>

</option>

<?php endforeach ?>

</select>

</td>

</tr>

Pronto. Nosso formulário agora já traz todos os dados atuais do produto para o usuário apenas alterar. Precisamos agora fazer a lógica de alteração, ou seja, o arquivo altera-produto.php. Ele é parecido com a regra de adicionar. Ele pega os valores do formulário e invoca a função de alterar (que ainda precisamos escrever). Vamos, por enquanto, só alterar o nome da função para alteraProduto:

<?php include("cabecalho.php");

include("conecta.php");

include("banco-produto.php");

$nome = $\_POST['nome'];

$preco = $\_POST['preco'];

$descricao = $\_POST['descricao'];

$categoria\_id = $\_POST['categoria\_id'];

if(array\_key\_exists('usado', $\_POST)) {

$usado = "true";

} else {

$usado = "false";

}

if(alteraProduto($conexao, $nome, $preco, $descricao, $categoria\_id, $usado)) { ?>

<p class="text-success">O produto <?= $nome ?>, <?= $preco ?> foi alterado.</p>

<?php } else {

$msg = mysqli\_error($conexao);

?>

<p class="text-danger">O produto <?= $nome ?> não foi alterado: <?= $msg?></p>

<?php

}

?>

<?php include("rodape.php"); ?>

Lá no banco-produto.php, vamos fazer algo parecido com o de inserir. Mas dessa vez, usaremos um UPDATE. Repare também que na atualização, precisamos passar o id, para alterar somente o de um produto:

function alteraProduto($conexao, $id, $nome, $preco, $descricao, $categoria\_id, $usado) {

$query = "update produtos set nome = '{$nome}', preco = {$preco}, descricao = '{$descricao}',

categoria\_id= {$categoria\_id}, usado = {$usado} where id = '{$id}'";

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

}

Mas veja que se recebemos o id do produto aqui, esse id precisa ser fornecido pelo altera-produto.php:

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

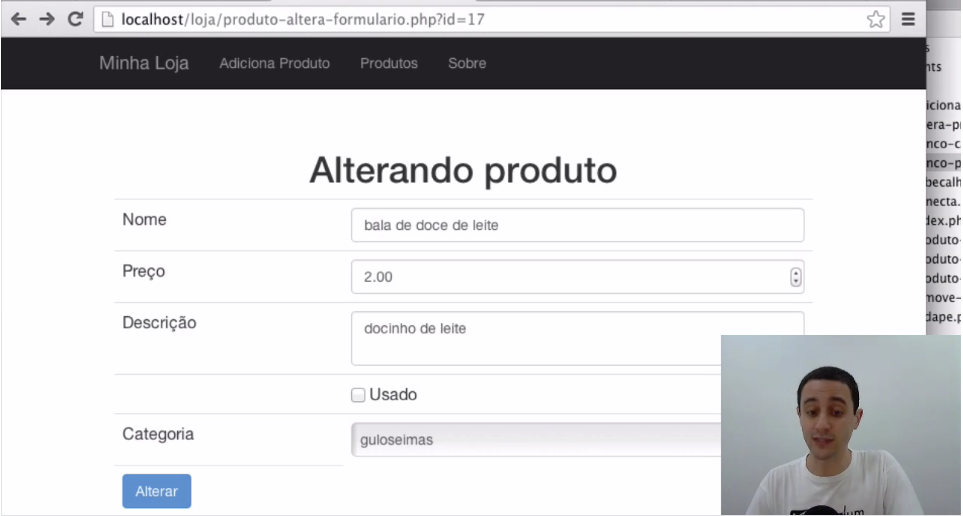
// ...

if(alteraProduto($conexao, $id, $nome, $preco, $descricao, $categoria\_id, $usado)) {

E já que pegamos ele com o $\_POST, precisamos também enviá-lo pelo formulário. Vamos colocar um campo hidden, igual usamos nos capítulos anteriores, afinal o usuário não precisa ver essa informação:

<input type="hidden" name="id" value="<?=$produto['id']?>" />

Agora sim, a alteração funciona.



Veja que não há muita diferença para a inserção, a não ser o ID, que precisamos para alterar o produto específico.