**Criando Funções no PHP**

No capítulo passado, vimos como fazer uma inserção no MySql. O próximo passo será mostrar a lista de produtos. Mas antes disso, vamos melhorar um pouco o que já escrevemos até então. Afinal, é fundamental que o programador escreva código que seja fácil de ser lido e mantido por outros desenvolvedores.

Nosso código atual, da página que adiciona produtos, é esse:

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

<?php

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

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

$conexao = mysqli\_connect("localhost", "root", "");

if(mysqli\_query($conexao, $query)) {

?>

<p class="alert-success">Produto <?= $nome; ?>, <?= $preco; ?> adicionado com sucesso!</p>

<?php

} else {

<?

<p class="alert-danger">O produto <? = $nome; ?> não foi adicionado</p>

<?php

}

?>

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

Esse if que verifica se deu sucesso está difícil de ler. Vamos extrair para uma variável:

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

if($resultadoDaInsercao) {

// codigo continua

Já está melhor. Mais fácil de ser lido. Mas vamos tentar extrair essa função de inserir produto para um método, com um nome mais semântico.

function insereProduto($nome, $preco) {

$query = "insert into produtos (nome, preco) values ('{$nome}', ${preco})";

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

}

A função está quase pronta. Para que funcione bem, ela precisa receber a conexão também (veja a variávelconexao). A função precisa também retornar o resultado:

function insereProduto($conexao, $nome, $preco) {

$query = "insert into produtos (nome, preco) values ('{$nome}', ${preco})";

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

return $resultadoDaInsercao;

}

Para usá-la, fica fácil:

if(insereProduto($conexao, $nome, $preco)) {

// continua aqui

}

Fica fácil reutilizar agora. Basta invocar a função sempre que quisermos inserir um produto. O código fica muito mais fácil de ser mantido.

Vamos agora mudar a cor da mensagem de sucesso. Hoje, a mensagem e seu fundo ficam verdes. Vamos fazer com que só o texto fique verde. Para isso, basta trocar a classe alert-success e alert-danger para text-success etext-danger. Não deixe de estudar as classes do Bootstrap, pois ele realmente nos ajuda a melhorar a interface do nosso site.

Outro ponto que a interface ainda peca é na navegabilidade. Da página principal não há como chegar na página do formulário; do formulário não há como chegar na página principal.

Vamos mexer isso no cabecalho.php, criando um div novo. Veja que usamos as várias classes do Bootstrap, que começam com nav\*. Essas classes nos ajudam a criar menus:

<html>

<head>

<title>Minha loja</title>

<meta charset="utf-8">

<link href="css/bootstrap.css" rel="stylesheet" />

<link href="css/loja.css" rel="stylesheet" />

</head>

<body>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<a href="index.php" class="navbar-brand">Minha Loja</a>

</div>

<div>

<ul class="nav navbar-nav">

<li><a href="produto-formulario.php">Adiciona Produto</a></li>

<li><a href="sobre.php">Sobre</a></li>

</ul>

</div>

</div><!-- container acaba aqui -->

</div>

<div class="container">

<div class="principal">

Muito melhor, nosso site já está mais navegável.

Vamos agora melhorar o formulário, que ainda não está bonito. Para isso, vamos fazer uso de tabelas:

<html>

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

<table>

<tr>

<td>Nome</td>

<td><input type="text" name="nome" /></td>

</tr>

<tr>

<td>Preço</td>

<td><input type="number" name="preco" /></td>

</tr>

<tr>

<td><input type="submit" value="Cadastrar" /></td>

</tr>

</table>

</form>

</html>

Já está melhor. O estilo de tabelas do Bootstrap são amigáveis. Podemos também por a classe form-control nas caixas de texto, que é o estilo próprio do Bootstrap para esse tipo de elemento. No botão, podemos por os estilosbtn btn-primary.

Por fim, vamos melhorar a mensagem de erro. Hoje, quando algo falha, apenas exibimos uma mensagem avisando o usuário, mas não explicamos bem qual foi o erro. Vamos resolver isso, exibindo o erro que o próprio MySql nos envia. Para isso, basta usar mysqli\_error:

<?php

if(insereProduto($conexao, $nome, $preco)) {

?>

<p class="alert-success">Produto <?= $nome; ?>, <?= $preco; ?> adicionado com sucesso!</p>

<?php

} else {

$msg = mysqli\_error($conexao);

?>

<p class="alert-danger">O produto <? = $nome; ?> não foi adicionado: <?= $msg ?></p>

<?php

}

?>