

PHP COM MYSQL(PDO)

Cadastro de produto com foto

André Luis de Souza Silva

PHP COM MYSQL

Cadastro de produto com foto

André Luis de Souza Silva

Todos os direitos reservados. Não é permitida a reprodução total ou parcial da presente obra sem prévia autorização do autor (Lei 9610/98). Nota: Apesar do zelo e dedicação na edição dessa obra, poderão ocorrer erros de digitação, impressão ou dúvida conceitual. Em qualquer dos casos citados, solicito que entre em contato para esclarecimento. e-mail do autor: andre_souzasilva@hotmail.com

S586p	Silva, André - 1982 PHP COM MYSQL(PDO): cadastro de produto com foto / André Luis de Souza Silva – São Joaquim da Barra – 2023. ISBN: 978-85-919198-2-6 1. 1. PHP 2. Programação de web 3. CSS 4. HTML CDU 620.5
-------	---

Sumário

Objetivo.....	10
1 Referencial	5
1.1 HTML	5
1.2 Bootstrap	6
1.3 SQL	7
1.4 PHP.....	10
2 Desenvolvendo com Bootstrap	12
3 Desenvolvendo cadastro de usuário.....	25
3.1 Criando bando de dados.....	25
3.2 Desenvolvendo “conexao.php”	27
3.3 Inserindo informações no banco de dados.....	30
3.4 Consultando produtos no bando de dados.....	37
3.5 Consultar um produto, e enviar os dados para o formulário.....	40
3.6 Alterar dados do produto no banco de dados.....	46

3.7 Excluir dados do produto no banco de dados	49
3.8 Mostando os produtos na index	53

*Dedico essa obra a minha família, que cedeu o seu
precioso tempo para que este pudesse ser concluído, pela
compreensão e dedicação.*

Objetivo

O objetivo desse livro é ajudar os iniciantes em linguagem de programação PHP a desenvolver um site de cadastro de produto.

No conteúdo do livro você vai aprender a como criar uma tabela utilizando o SGBD MySQL, e desenvolver um site dinâmico em PHP, com um cadastro de produto contendo inserção, alteração e exclusão dos dados

Espero que o conteúdo desse livro ajude você a iniciar seus conhecimentos nessa poderosa linguagem de programação.

1 Referencial

1.1 HTML

HTML (abreviação para a expressão inglesa *HyperText Markup Language*, que significa *Linguagem de Marcação de Hipertexto*) é uma linguagem de marcação utilizada para produzir páginas na Web. Documentos HTML podem ser interpretados por navegadores. A tecnologia é fruto da junção entre os padrões HyTime e SGML.

HyTime é um padrão para a representação estruturada de hipermídia e conteúdo baseado em tempo. Um documento é visto como um conjunto de eventos concorrentes dependentes de tempo (como áudio, vídeo, etc.), conectados por hiperligações. O padrão é independente de outros padrões de processamento de texto em geral.

SGML é um padrão de formatação de textos. Não foi desenvolvido para hipertexto, mas tornou-se conveniente para transformar documentos em hiper-objetos e para descrever as ligações. (pt.wikipedia.org/wiki/HTML)

1.2 Bootstrap

O Bootstrap é uma solução criada em 2010 por dois desenvolvedores do Twitter fissurados em CSS, @mdo e @fat. A intenção era criar e disponibilizar uma solução para otimizar a produção de layouts responsivos para web. O projeto era utilizado internamente no Twitter, mas como era open source e estava disponibilizado no GitHub, a popularidade do framework explodiu e hoje é praticamente inviável um desenvolvedor web não saber utilizar o mesmo.

Apesar do Bootstrap ajudar muito no desenvolvimento web, ele não vai fazer todo o seu trabalho. Com ele você pode reproduzir qualquer layout presente na web, desde que tenha o conhecimento necessário em CSS para ajustes no código. (RENE 2015)

Um layout com design responsivo é aquele que tem em suas linhas de código, parâmetros para melhorar a forma como o layout vai ser exibido em diferentes resoluções de

tela, que basicamente se resume em smartphones, tablets, notebooks, desktops e televisores smart.

Imagina o trabalho que desenvolvedores teriam se fossem desenvolver uma aplicação para cada dispositivo (já não basta as diferentes plataformas no mercado), com base na resolução de tela. A imagem abaixo descreve bem esse drama, pense em cada tela como um dispositivo, do Apple Watch a um monitor LED Full HD. (RENE 2015)

1.3 SQL

Structured Query Language, ou **Linguagem de Consulta Estruturada** ou **SQL**, é a linguagem de pesquisa declarativa padrão para banco de dados relacional (base de dados relacional). Muitas das características originais do SQL foram inspiradas na álgebra relacional.

O **SQL** foi desenvolvido originalmente no início dos anos 70 nos laboratórios da IM em San Jose, dentro do projeto System R, que tinha por objetivo demonstrar a viabilidade da implementação do modelo relacional proposto por E. F. Codd. O nome original da linguagem era *SEQUEL*,

acrônimo para "*Structured English Query Language*" (Linguagem de Consulta Estruturada, em Inglês), vindo daí o facto de, até hoje, a sigla, em inglês, ser comumente pronunciada "síquel" ao invés de "és-kiú-él", letra a letra. No entanto, em português, a pronúncia mais corrente é a letra a letra: "ésse-quê-éle".

A linguagem é um grande padrão de banco de dados. Isto decorre da sua simplicidade e facilidade de uso. Ela se diferencia de outras linguagens de consulta a banco de dados no sentido em que uma consulta SQL especifica a forma do resultado e não o caminho para chegar a ele. Ela é uma linguagem declarativa em oposição a outras linguagens procedurais. Isto reduz o ciclo de aprendizado daqueles que se iniciam na linguagem.

Embora o SQL tenha sido originalmente criado pela IBM, rapidamente surgiram vários "dialectos" desenvolvidos por outros produtores. Essa expansão levou à necessidade de ser criado e adaptado um padrão para a linguagem. Esta tarefa foi realizada pela American National Standards Institute (ANSI) em 1986 e ISO em 1987.

O SQL foi revisto em 1992 e a esta versão foi dado o nome de SQL-92. Foi revisto novamente em 1999 e 2003 para se tornar SQL:1999 (SQL3) e SQL:2003, respectivamente. O SQL:1999 usa expressões regulares de emparelhamento, *queries* recursivas e gatilhos (*triggers*). Também foi feita uma adição controversa de tipos não-escalados e algumas características de orientação a objeto. O SQL:2003 introduz características relacionadas ao XML, sequências padronizadas e colunas com valores de auto-generalização (inclusive colunas-identidade).

Tal como dito anteriormente, embora padronizado pela ANSI e ISO, possui muitas variações e extensões produzidos pelos diferentes fabricantes de sistemas gerenciadores de bases de dados. Tipicamente a linguagem pode ser migrada de plataforma para plataforma sem mudanças estruturais principais.

Outra aproximação é permitir para código de idioma procedural ser embutido e interagir com o banco de dados. Por exemplo, o Oracle e outros incluem Java na base de dados, enquanto o PostgreSQL permite que funções sejam

escritas em Perl, Tcl, ou C, entre outras linguagens.
(pt.wikipedia.org/wiki/sql)

1.4 PHP

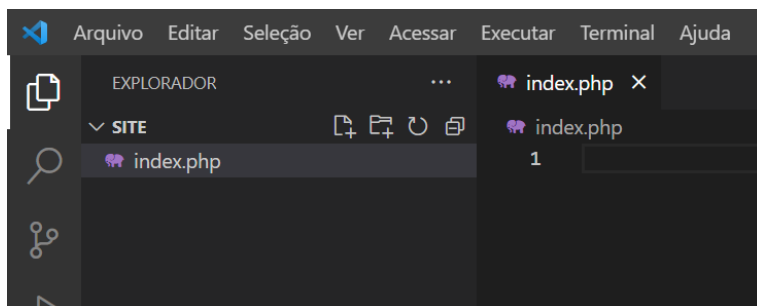
PHP (um acrônimo Recursivo para "***PHP: Hypertext Preprocessor***", originalmente ***Personal Home Page***) é uma linguagem interpretada livre, usada originalmente apenas para o desenvolvimento de aplicações presentes e atuantes no lado do servidor, capazes de gerar conteúdo dinâmico na World Wide Web. Figura entre as primeiras linguagens passíveis de inserção em documentos HTML, dispensando em muitos casos o uso de arquivos externos para eventuais processamentos de dados. O código é interpretado no lado do servidor pelo módulo PHP, que também gera a página web a ser visualizada no lado do cliente. A linguagem evoluiu, passou a oferecer funcionalidades em linha de comando, e além disso, ganhou características adicionais, que possibilitaram usos adicionais do PHP, não relacionados a web sites. É possível instalar o PHP na maioria dos sistemas

operacionais, gratuitamente. Concorrente direto da tecnologia ASP pertencente à Microsoft, o PHP é utilizado em aplicações como o MediaWiki, Facebook, Drupal, Joomla, WordPress, Magento e o Oscommerce. Criado por Rasmus Lerdorf em 1995, o PHP tem a produção de sua implementação principal — referência formal da linguagem, mantida por uma organização chamada The PHP Group. O PHP é software livre, licenciado sob a PHP License, uma licença incompatível com a GNU General Public License (GPL) devido a restrições no uso do termo PHP. (pt.wikipedia.org/wiki/PHP)

2 Desenvolvendo com Bootstrap

Criar uma pasta site e abrir no editor de código fonte, neste livro será usado o VSCODE.

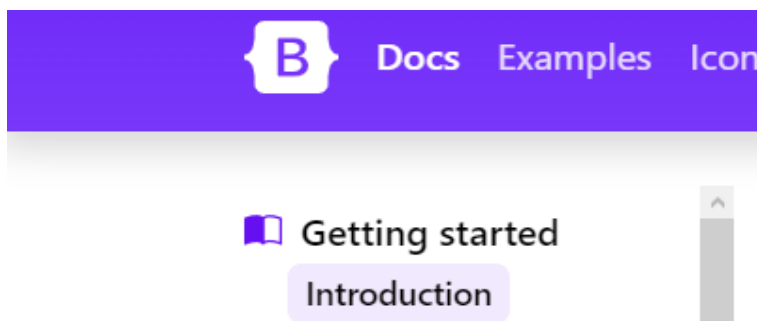
Criar um arquivo “index.php”



Abra o site <https://getbootstrap.com>, e clique em docs.



Clique em Introduction.



Incluir CSS e JS do Bootstrap, copiando o trecho de código HTML para o arquivo index.php.

2. **Include Bootstrap's CSS and JS.** Place the `<link>` tag in the `<head>` for our CSS, and the `<script>` tag for our JavaScript bundle (including Popper for positioning dropdowns, poppers, and tooltips) before the closing `</body>`. Learn more about our [CDN links](#).

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap demo</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
  </head>
  <body>
    <h1>Hello, world!</h1>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js" rel="script">
  </body>
</html>
```

Para testar será usado o servidor embutido do PHP, para isso abra o CMD na pasta site e digite o comando "php -S localhost:8080."

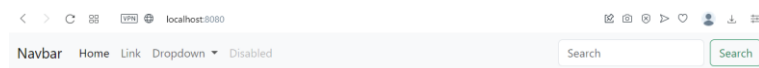

```
C:\Windows\System32\cmd.e  X  +  v
Microsoft Windows [versão 10.0.22621.1265]
(c) Microsoft Corporation. Todos os direitos reservados.
C:\site>php -S localhost:8080|
```

Abra o navegador e digite “localhost:8080”



Localizar o menu “NavBar”, e copiar o código “html” que está em “Supported content”, para dentro do arquivo “index.php”, entre as tags <body>, apague a linha do “Hello, word!”.

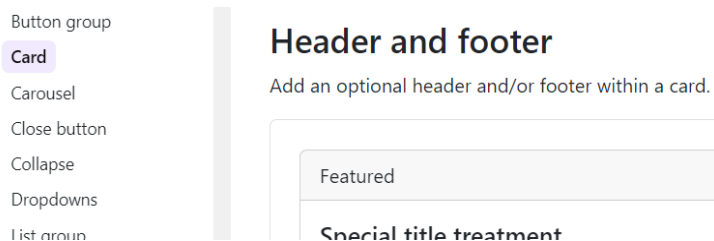
Para testar digitar “localhost:8080” no navegador.



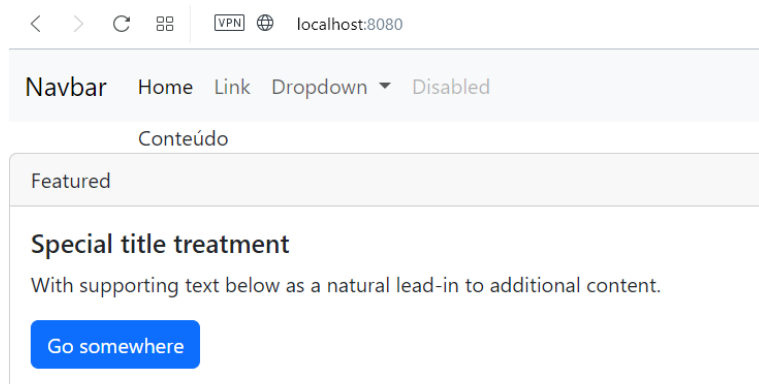
Digite o código embaixo da tag “</nav>”

```
44     </div>
45   </nav>
46   <div class="container">
47     Conteúdo
48   </div>
49   <script src="https://cdn.jsdelivr.net/npm/
50 </body>
51 </html>
```

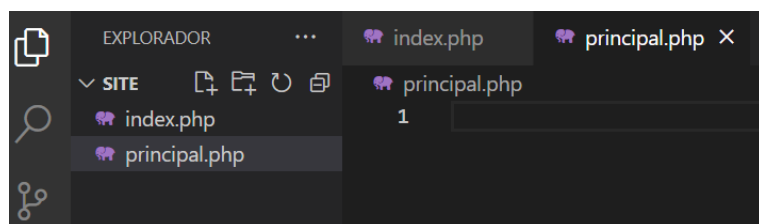
Localizar no menu do bootstrap a opção “Cards”, e copiar o código “html” que está em “Header and footer”, para dentro do arquivo “index.php”, embaixo da ultima tag “</div>”.



Para testar digitar “localhost:8080” no navegador.



Criar um arquivo “principal.php” dentro da pasta site.



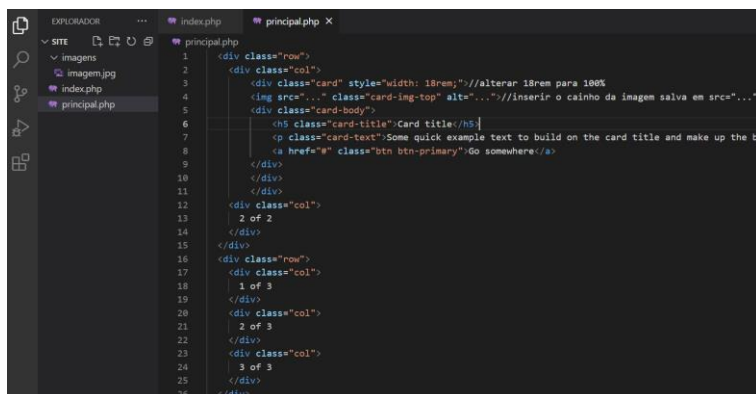
Localizar no menu do bootstrap a opção “Grid” e copiar o código “html” que está dentro de “Equal-width”, para dentro do arquivo “principal.php”.

```
principal.php
1  <div class="container text-center">//apagar essa linha
2      <div class="row">
3          <div class="col">
4              1 of 2
5          </div>
6          <div class="col">
7              2 of 2
8          </div>
9      </div>
10     <div class="row">
11         <div class="col">
12             1 of 3
13         </div>
14         <div class="col">
15             2 of 3
16         </div>
17         <div class="col">
18             3 of 3
19         </div>
20     </div>
21 </div>//apagar essa linha
```

Apagar as linhas 1 e 21.

Localizar no menu do bootstrap a opção “Card” e copiar o código “html” que está dentro de “Example”, para dentro do arquivo “principal.php”, dentro da primeira “<div class=“col””, substituindo o texto 1 of 2, crie uma pasta imagem dentro da pasta site e salve uma imagem qualquer com o nome de “imagem.jpg”.

Na imagem abaixo mostra algumas alterações que devem ser feitas, a primeira na linha 3, alterar 18rem para 100%, a segunda e inserir o caminho da imagem salva em `src="..."` na linha 4.



```
1 <div class="row">
2   <div class="col">
3     <div class="card" style="width: 18rem;"/><!--alterar 18rem para 100%
4     <!--inserir o caminho da imagem salva em src="..."
5     <div class="card-body">
6       <h5 class="card-title">Card title</h5>
7       <p class="card-text">Some quick example text to build on the card title and make up the bulk of the
8       <a href="#" class="btn btn-primary">Go somewhere</a>
9     </div>
10  </div>
11 </div>
12 <div class="col">
13   2 of 2
14 </div>
15 </div>
16 <div class="row">
17   <div class="col">
18     1 of 3
19   </div>
20   <div class="col">
21     2 of 3
22   </div>
23   <div class="col">
24     3 of 3
25   </div>
26 </div>
```

Após as alterações o código deve ficar com a imagem abaixo.



```
1 <div class="row">
2   <div class="col">
3     <div class="card" style="width: 100%;">
4     
5     <div class="card-body">
6       <h5 class="card-title">Card title</h5>
7       <p class="card-text">Some quick example text to build on the ca
8       <a href="#" class="btn btn-primary">Go somewhere</a>
9     </div>
10  </div>
11 </div>
12 <div class="col">
13   2 of 2
14 </div>
```

Copiar o conteúdo que está dentro da "<div class='col'>" e colar dentro das outras "divs" com a mesma classe. Abrir o arquivo "index.php" e inserir o código.

```
45     </nav>
46     <div class="container">
47         <?php
48             $link = @$_GET['link'];
49             $pag[1]='principal.php';
50
51             if(!empty($link)){
52                 if(file_exists($pag[$link])){
53                     include $pag[$link];
54                 }
55             }else{
56                 trim(include 'principal.php');
57             }
58         ?>
59     </div>
60     <div class="card">
61         <div class="card-header">
62             Featured
```

Esse código é responsável para deixar dinâmico o conteúdo que será dinâmico na página "index.php".

Linha 48 – foi criado uma variável que recebe um valor passado pela super global "\$_GET".

Linha 49 – foi declarado um array que será associado com as páginas do site.

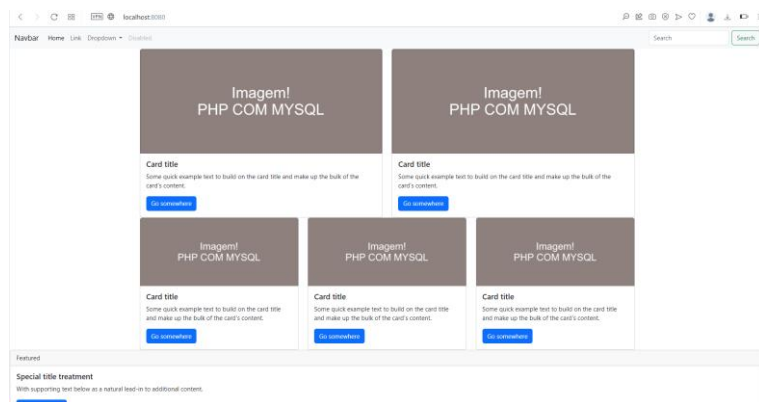
Linha 51 – teste se a variável está diferente de vazio.

Linha 52 – teste se existe um array associado com o número que foi passado.

Linha 53 – se existir mostra a página relacionada.

Linha 56 – se a condição não for verdadeira, abre a página principal dentro da “index.php”.

Para testar digitar “localhost:8080” no navegador.



Criar um arquivo “cadProduto.php” dentro da pasta site e digite o código abaixo.

```

1 <h1>Cadastro de Produto</h1>
2 <form name="form1" action="" method="post" enctype="multipart/form-data">
3   <div class="mb-3">
4     <label>Descrição</label>
5     <input type="text" name="descricao" class="form-control" value="">
6   </div>
7   <div class="mb-3">
8     <label>Valor de custo</label>
9     <input type="text" name="custo" class="form-control" value="">
10  </div>
11  <div class="mb-3">
12    <label>Fornecedor</label>
13    <input type="text" name="fornecedor" class="form-control" value="">
14  </div>
15  <div class="mb-3">
16    <label>Foto</label>
17    <input type="file" name="foto" class="form-control">
18  </div>
19  <input type="submit" class="btn btn-primary" value="Enviar">
20 </form>
```

Acrescentar o código dentro do arquivo "index.php".

```

34 </nav>
35 <div class="container">
36   <?php
37     $link = @$_GET['link'];
38     $pag[1]='principal.php';
39     $pag[2]='cadProduto.php';
40
41
42     if(!empty($link)){
43       if(file_exists($pag[$link])){
44         include $pag[$link];
45       }
46     }else{
47       trim(include 'principal.php');
48     }
49   <?>
```

Apagar os trechos de código marcados com vermelho


```
10 <nav class="navbar navbar-expand-lg bg-body-tertiary">
11 <div class="container-fluid">
12 <a class="navbar-brand" href="#">Navbar</a>
13 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedConte
14 <span class="navbar-toggler-icon"></span>
15 </button>
16 <div class="collapse navbar-collapse" id="navbarSupportedContent">
17 <ul class="navbar-nav me-auto mb-2 mb-lg-0">
18 <li class="nav-item">
19 <a class="nav-link active" aria-current="page" href="#">Home</a>
20 </li>
21 <li class="nav-item">
22 <a class="nav-link" href="#">Link</a>
23 </li>
24 <li class="nav-item dropdown">
25 <a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="fal
26 Dropdown
27 </a>
28 <ul class="dropdown-menu">
29 <li><a class="dropdown-item" href="#">Action</a></li>
30 <li><a class="dropdown-item" href="#">Another action</a></li>
31 <li><hr class="dropdown-divider"></li>
32 <li><a class="dropdown-item" href="#">Something else here</a></li>
33 </ul>
34 </li>
35 <li class="nav-item">
36 <a class="nav-link disabled">Disabled</a>
37 </li>
38 </ul>
39 <div class="d-flex" role="search">
```

Alterar os trechos de código marcados com amarelo e acrescentar os que estão marcados com vermelho.

```
<body>
<nav class="navbar navbar-expand-lg bg-body-tertiary">
  <div class="container-fluid">
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
        <li class="nav-item">
          <a class="nav-link active" aria-current="page" href="index.php?link=1">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="index.php?link=2">Cadastro de Produto</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="index.php?link=3">Área Restrita</a>
        </li>
      </ul>
    </div>
    <form class="d-flex" role="search">
```

Alterar o trecho de código do rodapé.

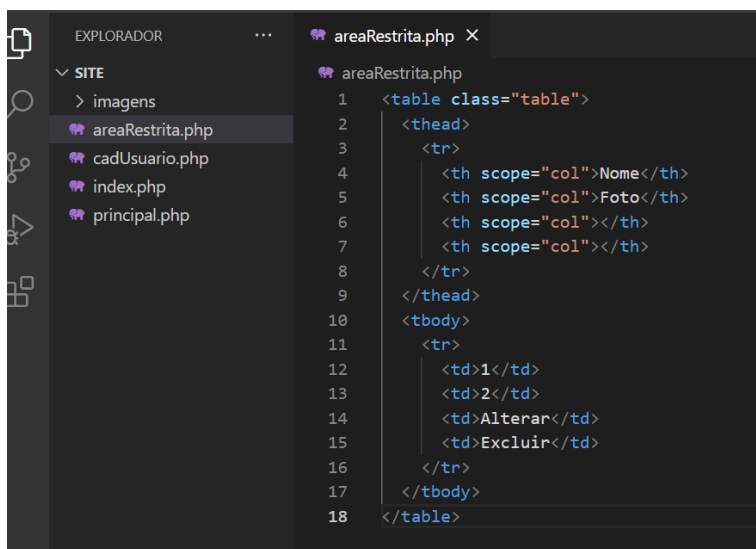
```
<div class="card">
  <div class="card-header">

</div>
<div class="card-body">
  <h5 class="card-title">Cadastro de Produto com PHP e Mysql</h5>
  <p class="card-text">Site para desenvolvimento de cadastro de produto</p>
  <a href="index.php?link=1" class="btn btn-primary">HOME</a>
</div>
</div>
```

Acrescentar os códigos no arquivo "index.php"

```
33     </nav>
34     <div class="container">
35         <?php
36             $link = @$_GET['link'];
37             $pag[1]='principal.php';
38             $pag[2]='cadUsuario.php';
39             $pag[3]='areaRestrita.php';
40
41
42             if(!empty($link)){
43                 if(file_exists($pag[$link])){
44                     include $pag[$link];
45                 }
46             }else{
47                 trim(include 'principal.php');
48             }
49         ?>
50     </div>
51     <div class="card">
```

Criar arquivo "areaRestrita.php" dentro da pasta site e digitar o código.



The image shows a code editor interface. On the left, the 'EXPLORADOR' (Explorer) panel shows a file structure for a 'SITE' with a subdirectory 'imagens' and four PHP files: 'areaRestrita.php', 'cadUsuario.php', 'index.php', and 'principal.php'. The 'areaRestrita.php' file is selected. The main editor area displays the code for 'areaRestrita.php', which is an HTML table with 4 columns and 2 rows. The columns are 'Nome', 'Foto', and an unnamed column. The rows contain the numbers '1', '2', 'Alterar', and 'Excluir'.

```
1 <table class="table">
2   <thead>
3     <tr>
4       <th scope="col">Nome</th>
5       <th scope="col">Foto</th>
6       <th scope="col"></th>
7     </tr>
8   </thead>
9   <tbody>
10    <tr>
11      <td>1</td>
12      <td>2</td>
13      <td>Alterar</td>
14      <td>Excluir</td>
15    </tr>
16  </tbody>
17 </table>
```

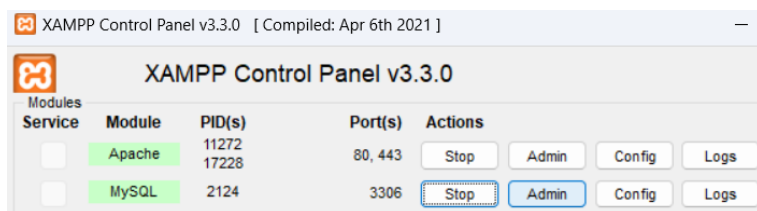
Para testar digitar "localhost:8080" no navegador.

Esse layout foi desenvolvido de uma forma simples para dar seguimento com o estudo do PHP, serão feitas mais alterações nos próximos capítulos.

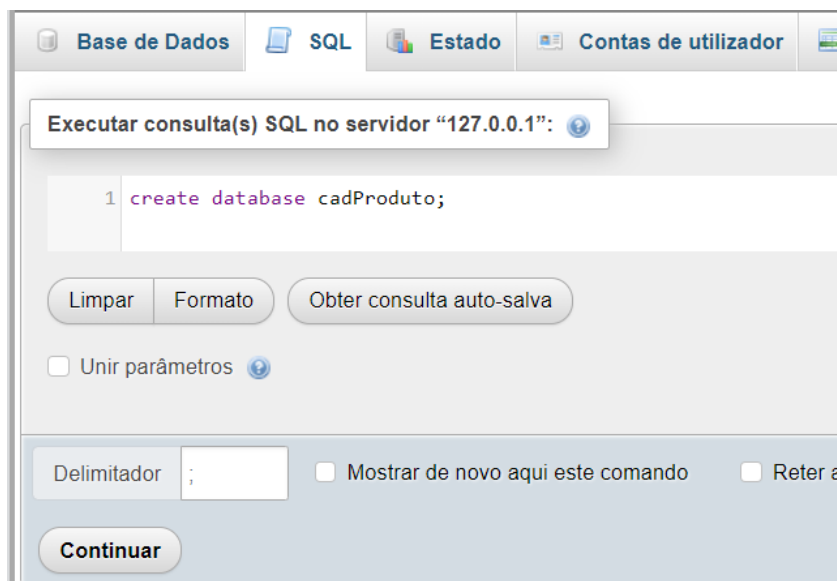
3 Desenvolvendo cadastro de usuário

3.1 Criando bando de dados

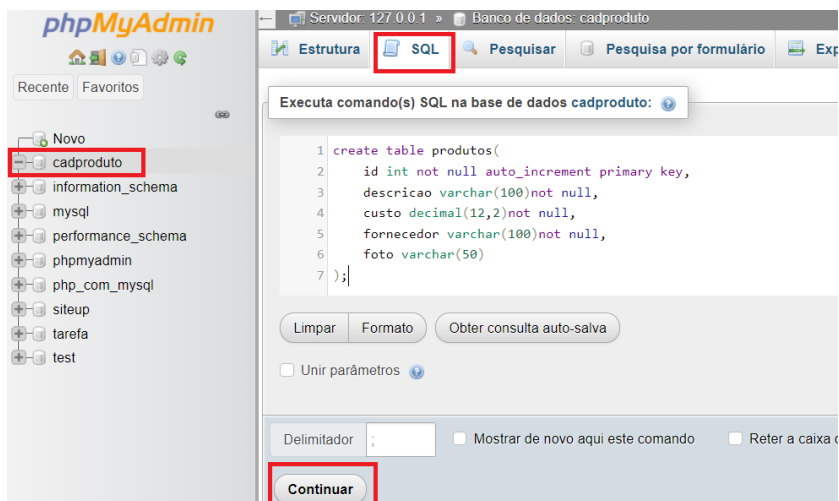
Abrir o xampp XAMPP Control e iniciar o serviço do APACHE e MYSQL. Clique no botão admin do MYSQL para abrir o phpmyadmin.



Clicar na aba sql e digite o código para criar a base de dados, clique em continuar para finalizar.



Selecionar a base de dados que foi criada na lateral esquerda, clique na aba sql e digite o código para criar a tabela produtos, para finalizar aperte continuar.



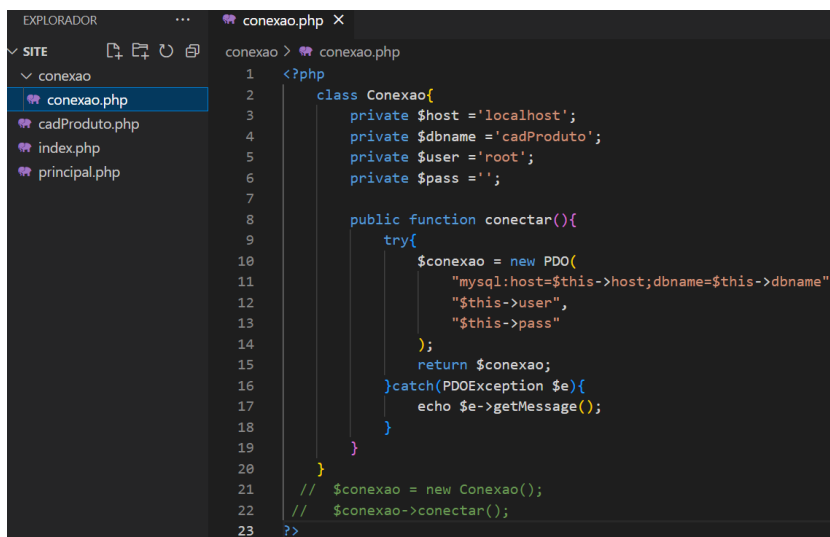
3.2 Desenvolvendo “conexao.php”

Criar uma pasta conexão dentro da pasta site, em seguida criar um arquivo “conexao.php” dentro da pasta conexão, e digite o código.

Linha 1 – foi criado a classe “Conexao”.

Linhas 3 a 6 – foi declarado os atributos do objeto com escopo privado.

Linha 8 – foi criado a função “conectar”



```
1 <?php
2 class Conexao{
3     private $host ='localhost';
4     private $dbname ='cadProduto';
5     private $user ='root';
6     private $pass ='';
7
8     public function conectar(){
9         try{
10             $conexao = new PDO(
11                 "mysql:host=$this->host;dbname=$this->dbname",
12                 "$this->user",
13                 "$this->pass"
14             );
15             return $conexao;
16         }catch(PDOException $e){
17             echo $e->getMessage();
18         }
19     }
20 }
21 // $conexao = new Conexao();
22 // $conexao->conectar();
23 ?>
```

Linha 9 – comando “try catch” para tratamento de erros.

Linha 10 – foi instanciado um objeto do tipo PDO, que recebe por parâmetro os valores para conexão.

Linha 15 – retorna a conexão para ser usada em outro momento.

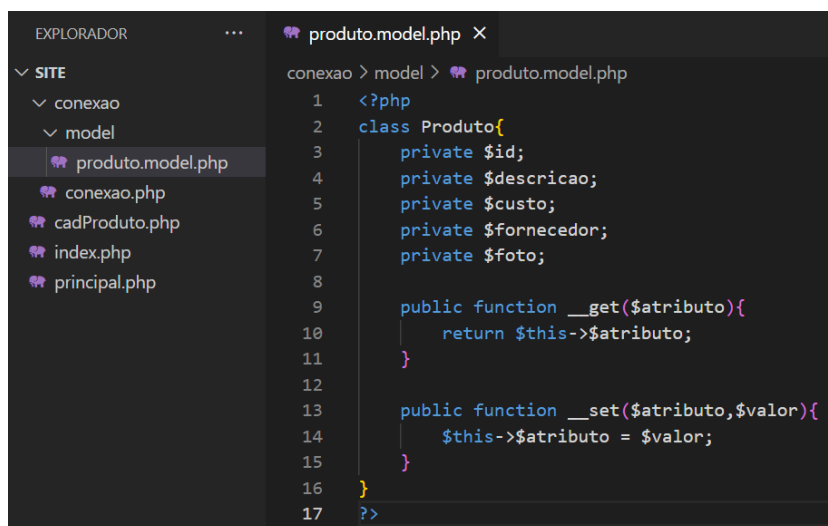
Linha 17 – se ocorrer algum erro com a conexão, mostra na tela.

Linhas 21 e 22 – foi criado um objeto “conexao” do tipo “Usuario” para realizar teste.

Para testar tire as barras das linhas 21 e 22, digitar “localhost:8080/conexao/conexao.php” no navegador, se retornar uma página em branco deu tudo certo.

3.3 Inserindo informações no banco de dados

Criar uma pasta “model” dentro da pasta site, em seguida criar um arquivo “produto.model.php” dentro da pasta model e digitar o código.



The screenshot shows a code editor with a dark theme. On the left, the 'EXPLORADOR' (Explorer) panel shows a file tree with 'SITE' as the root. Under 'SITE', there is a folder 'conexao' and a folder 'model'. Inside 'model', the file 'produto.model.php' is selected. The main editor area shows the code for 'produto.model.php' with line numbers 1 through 17. The code defines a 'Produto' class with private attributes and two public methods: '__get' and '__set'.

```

1  <?php
2  class Produto{
3      private $id;
4      private $descricao;
5      private $custo;
6      private $fornecedor;
7      private $foto;
8
9      public function __get($atributo){
10         return $this->$atributo;
11     }
12
13     public function __set($atributo,$valor){
14         $this->$atributo = $valor;
15     }
16 }
17 ?>
    
```

Linha 2 – foi criado a classe “Produto”, que será o modelo do objeto.

Linhas 3 à 7 – foram criados os atributos do objeto, com escopo privado.

Linha 9 – foi criado o método assessor “__get”, para recuperar os dados salvos nos atributos.

Linha 13 – foi criado o método assessor “__set”, para guardar as informações nos atributos.

Criar uma pasta “service” dentro da pasta site, em seguida criar um arquivo “produto.service.php”, dentro da pasta “service”, e digitar o código.

```
service > produto.service.php
1  <?php
2  class ProdutoService{
3      private $produto;
4      private $conexao;
5
6      public function __construct(Produto $produto, Conexao $conexao){
7          $this->conexao= $conexao->conectar();
8          $this->produto= $produto;
9      }
10
11     public function inserir(){
12         $query = 'insert into produtos (descricao, custo, fornecedor, foto)
13             values(?, ?, ?, ?)';
14
15         $stmt = $this->conexao->prepare($query);
16         $stmt->bindValue(1,$this->produto->__get('descricao'));
17         $stmt->bindValue(2,$this->produto->__get('custo'));
18         $stmt->bindValue(3,$this->produto->__get('fornecedor'));
19         $stmt->bindValue(4,$this->produto->__get('foto'));
20
21         if($stmt->execute()){
22             $diretorio = 'fotoProduto/';
23             move_uploaded_file($_FILES['foto']['tmp_name'],
24                 $diretorio.$this->produto->__get('foto'));
25         }
26     }
27 }
```

Linha 2 – foi criado a classe “ProdutoService”.

Linhas 3 e 4 – foram criados os atributos.

Linha 6 – foi criado o método construtor, que recebe por parâmetro os objetos “conexao” e “produto”.

Linha 11 – foi criado a função inserir.

Linha 12 – foi criado uma variável query que recebe uma “string”.

Linha 15 – foi criado uma variável “stmt” que recebe o objeto “conexao” que é do tipo “PDO”, junto com o método “prepare” com o parâmetro “query”.

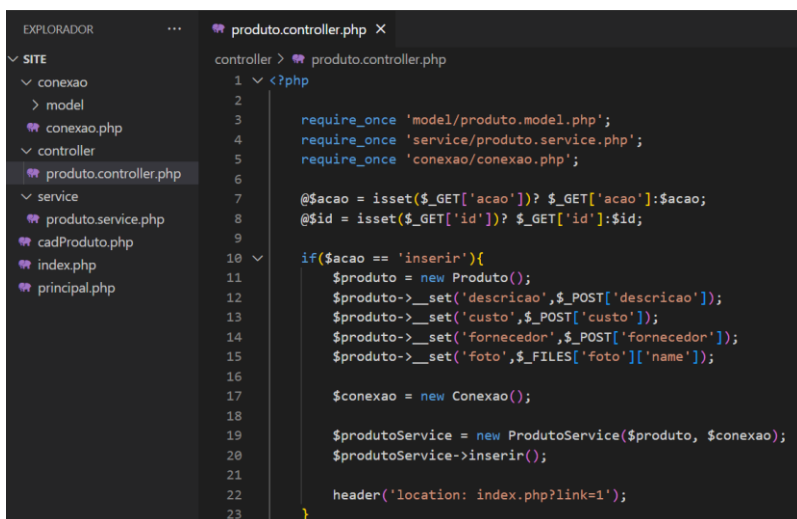
Linhas 16 à 19 – com a variável “stmt” acesso o “bindValue” para tratar a “query” e associar com as interrogações os valores que estão sendo buscado com o método “__get” na classe “Produto”.

Linha 21 – foi criado uma condição que se o “execute” acontecer entra no bloco de instrução, que atribui na variável “diretório” o texto “fotoProduto/”, que se junta com o nome da foto que foi enviado na hora do cadastro, assim, formando o caminho para salvar a foto dentro da pasta. O método “execute” roda a query no banco de dados.

Crie uma pasta “controller” dentro da pasta site, em seguida crie um arquivo “produto.controller.php” e digite o código.

Linhas 3 a 5 – foi feita a importação dos arquivos que serão usados.

Linhas 7 e 8 – foram criadas duas variáveis que recebem por “get” ou por declaração de variável, assim garante qual ação será realizada e quando necessário qual usuário será modificado pelo número do id.



```

1 <?php
2
3 require_once 'model/produto.model.php';
4 require_once 'service/produto.service.php';
5 require_once 'conexao/conexao.php';
6
7 @$acao = isset($_GET['acao'])? $_GET['acao']:$acao;
8 @$id = isset($_GET['id'])? $_GET['id']:$id;
9
10 if($acao == 'inserir'){
11     $produto = new Produto();
12     $produto->__set('descricao',$_POST['descricao']);
13     $produto->__set('custo',$_POST['custo']);
14     $produto->__set('fornecedor',$_POST['fornecedor']);
15     $produto->__set('foto',$_FILES['foto']['name']);
16
17     $conexao = new Conexao();
18
19     $produtoService = new ProdutoService($produto, $conexao);
20     $produtoService->inserir();
21
22     header('location: index.php?link=1');
23 }
    
```

Linha 10 – foi feita uma condição para comparar a ação recebida.

Linha 11 – foi instanciado o objeto do tipo “Produto”.

Linhas 12 à 15 – com o objeto “produto” através do método “__set” será salvo nos atributos da classe “Produto” os dados do produto passado pelo formulário de cadastro.

Linha 17 – foi instanciado o objeto do tipo “Conexao”.

Linha 19 – foi instanciado o objeto do tipo “ProdutoService”, e passado como parâmetro os objetos criados “produto” e “conexao”.

Linha 20 – com o objeto “produtoService” é chamado o método “inserir”.

Linha 22 – redireciona para página “index.php”.

Abrir o arquivo “cadProduto.php” e modifique como mostra a figura.

```
cadProduto.php
1 <h1>Cadastro de Produto</h1>
2 <form name="form1" action="index.php?link=4&acao=inserir" method="post" enctype="multipart/form-data">
3   <div class="mb-3">
4     <label>Descrição</label>
5     <input type="text" name="descricao" class="form-control" value="">
6   </div>
7   <div class="mb-3">
8     <label>Valor de custo</label>
9     <input type="text" name="custo" class="form-control" value="">
10  </div>
11  <div class="mb-3">
12    <label>Fornecedor</label>
13    <input type="text" name="fornecedor" class="form-control" value="">
14  </div>
15  <div class="mb-3">
16    <label>Foto</label>
17    <input type="file" name="foto" class="form-control" >
18  </div>
19  <input type="submit" class="btn btn-primary" value="Enviar">
20 </form>
```

Criar um arquivo “produto.controller.php” dentro da pasta site e digitar o código.

```
produto.controller.php
1  <?php
2  |  require_once 'controller/produto.controller.php';
3  ?>
```

Linha 2 – quando esse arquivo for chamado, inclui o “produto.controller” da pasta controller.

Inserir no array da página “index.php” o arquivo “produto.controller.php”.

```
33  <?php
34  $link = @$_GET['link'];
35  $pag[1]='principal.php';
36  $pag[2]='cadProduto.php';
37  $pag[3]='areaRestrita.php';
38  $pag[4]='produtoController.php';
39
40
41  if(!empty($link)){
42  |  if(file_exists($pag[$link])){
43  |  |  include $pag[$link];
```

Criar dentro da pasta site uma pasta fotoProduto, para salvar as fotos que serão inseridas pelo cadastro de produto.

Para testar digite "localhost:8080" no navegador e clique em cadastro de usuário, digite as informações do produto com foto e clique em enviar.

Se não retornou nenhum erro o site será redirecionado para a página "index.php".

Abra o "PHPmyAdmin", clique no banco de dados php_com_mysql, clique na tabela "produtos" e verifique se os dados foram inseridos.

Abra a pasta fotoProduto e verifique se a foto foi salva.

	id	descricao	custo	fornecedor	foto
<input type="checkbox"/>	3	Computador Gamer Completo RGB Intel Core i5 8GB HD...	4500.00	infotec	computador1.jpg
Com os selecionados: <input type="checkbox"/> Marcar todos <input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Apagar <input type="checkbox"/> Exportar					

C:\site\fotoProduto



computador1.jpg

3.4 Consultando produtos no bando de dados

Abrir o arquivo “produto.service.php” e digitar o código, função recuperar.

```
26     }  
27     public function recuperar(){  
28         $query = 'select id, descricao, custo, fornecedor, foto  
29             from produtos';  
30         $stmt=$this->conexao->prepare($query);  
31         $stmt->execute();  
32         return $stmt->fetchAll(PDO::FETCH_OBJ);  
33     }  
34 }
```

Linha 27 – foi criado a função para recuperar todos os produtos.

Linha 28 – foi declarado uma variável para receber um texto, na linguagem sql.

Linha 30 - foi criado uma variável “stmt” que recebe o objeto “conexao” que é do tipo “PDO”, junto com o método “prepare” com o parâmetro “query”.

Linha 31 – executa o código sql no bando de dados.

Linha 32 – retorna em formato de objeto todos os produtos cadastrados no banco de dados.

Abrir o arquivo “produto.controller.php” e digitar o código referente a recuperar.

```
22     header('location: index.php?link=1');
23 }
24 if($acao == 'recuperar'){
25     $produto = new Produto();
26     $conexao = new Conexao();
27
28     $produtoService = new ProdutoService($produto, $conexao);
29     $produto= $produtoService->recuperar();
30 }
```

Linha 24 – foi realizado uma condição se ação for igual a recuperar.

Linha 25 – foi instanciado o objeto “produto”.

Linha 26 – foi instanciado o objeto “conexao”.

Linha – 28 foi instanciado o objeto do tipo “ProdutoService”, e passado como parâmetro os objetos criados “produto” e “conexao”.

Linha 29 – com o objeto “produtoService” é chamado o método “recupera”, as informações que serão retornadas serão salvas dentro da variável “produto”.

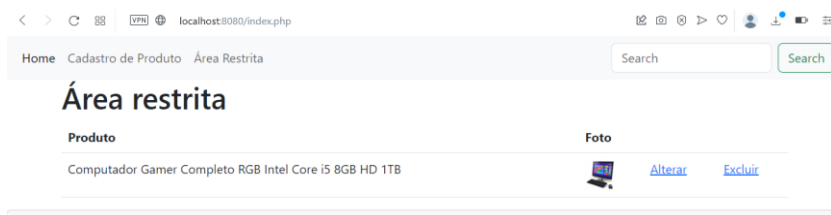
Abrir o arquivo “areaRestrita.php”, e faça as alterações necessárias.

```

1  <?php
2  $acao = 'recuperar';
3  require_once 'produto.controller.php';
4  ?>
5  <h1>Área restrita</h1>
6  <table class="table">
7    <thead>
8      <tr>
9        <th scope="col">Produto</th>
10       <th scope="col">Foto</th>
11       <th scope="col"></th>
12       <th scope="col"></th>
13     </tr>
14   </thead>
15   <?php foreach($produto as $indice => $produto){ ?>
16     <tbody>
17       <tr>
18         <td><?= $produto->descricao;></td>
19         <td></td>
20         <td><a href="index.php?link=2&metodo=alterar&id=<?= $produto->id;>">Alterar</a></td>
21         <td><a href="index.php?link=2&metodo=excluir&id=<?= $produto->id;>">Excluir</a></td>
22       </tr>
23     </tbody>
24   <?php } ?>
25 </table>

```

Para testar, digitar "localhost:8080/" e clique no menu área restrita.



Na área restrita vai aparecer todos os produtos cadastrados com sua respectiva foto, e com as opções de alterar e excluir.

3.5 Consultar um produto, e enviar os dados para o formulário.

Abrir o arquivo “produto.service.php” e digitar o código, função recuperarProduto.

```
33     }  
34     public function recuperarProduto($id){  
35         $query = 'select id, descricao, custo, fornecedor, foto  
36             from produtos  
37             where id = ?';  
38         $stmt = $this->conexao->prepare($query);  
39         $stmt->bindValue(1,$id);  
40         $stmt->execute();  
41         return $stmt->fetchALL(PDO::FETCH_OBJ);  
42     }  
43 }
```

Linha 34 – foi criado a função para recuperar um produto, referente ao “\$id” que foi passado.

Linha 35 – foi declarado uma variável para receber um texto, na linguagem sql que consulta o usuário referente ao id.

Linha 38 - foi criado uma variável “stmt” que recebe o objeto “conexao” que é do tipo “PDO”, junto com o método “prepare” com o parâmetro “query”.

Linha 39 – com a variável “\$smtp” utilizo a função “bindValue” para associar o “\$id” que foi passado com a primeira interrogação (?) do código SQL.

Linha 40 – executa o código sql no bando de dados.

Linha 41 – retorna em formato de objeto o usuário cadastrado no banco de dados com o id passado.

Abrir o arquivo “produto.controller.php” e digitar o código referente a recuperarProduto.

```
29     $produto= $produtoService->recuperar();  
30 }  
31 if($acao == 'recuperarProduto'){  
32     $produto = new Produto();  
33     $conexao = new Conexao();  
34  
35     $produtoService = new ProdutoService($produto, $conexao);  
36     $produto= $produtoService->recuperar($id);  
37 }
```

Linha 31 – foi realizado uma condição se ação for igual a recuperarProduto.

Linha 32 – foi instanciado o objeto “produto”.

Linha 33 – foi instanciado o objeto “conexao”.

Linha – 35 foi instanciado o objeto do tipo “ProdutoService”, e passado como parâmetro os objetos criados “produto” e “conexao”.

Linha 36 – com o objeto “produtoService” é chamado o método “recuperaProduto” passando o parâmetro “\$id”, as informações que serão retornadas serão salvas dentro da variável “produto”.

Abrir o arquivo “cadProduto.php”, e faça as alterações necessárias.

```
cadProduto.php
1  <?php
2      if(isset($_GET['metodo'])){
3          $metodo = $_GET['metodo'];
4          $acao = 'recuperarProduto';
5          $id = $_GET['id'];
6          require_once 'produto.controller.php';
7          foreach($produto as $index => $produto){
8              $id = $produto->id;
9              $descricao = $produto->descricao;
10             $custo = $produto->custo;
11             $fornecedor = $produto->fornecedor;
12             $foto = $produto->foto;
13             $_SESSION['foto']=$produto->foto;
14         }
15     }
16 ?>
17 <h1>Cadastro de Produto</h1>
```

Linha 2 – Foi criado um condição, que testa se a super global “\$_GET[‘metodo’]” estiver carregada o fluxo continua.

Linha 3 – foi criado uma variável que recebe o valor de “\$_GET[‘metodo’]”.

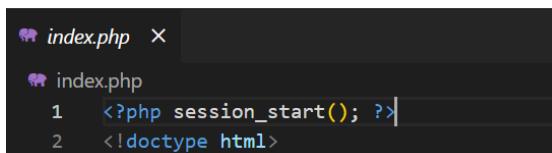
Linha 4 – foi criado uma variável “\$acao” que recebe “recuperarProduto”.

Linha 5 – Foi criado uma variável “\$id” que o valor de “\$_GET[‘id’]”

Linha 6 – foi feito uma requisição do arquivo “produto.controller.php” para dentro desta página.

Linhas 7 à 23 – foi criado um comando de repetição para guardar dentro de cada variável as informações retornada do produto.

Linha 12 – foi criado uma “\$_SESSION[‘foto’]” para guardar o registo da foto do usuário, que será utilizada quando for apagar a foto do usuário da pasta, para a “SESSION” funcionar digite na página “index.php” o seguinte código



```
index.php x
index.php
1 <?php session_start(); ?>
2 <!doctype html>
```

Modifique a propriedade “action” da tag “<form>” no “cadProduto.php”.

```

18 <form name="form1" action="index.php?link=4&acao=<?php if(isset($metodo)){echo 'inserir';}
19     elseif($metodo == 'alterar'){echo 'alterar';}else{echo 'excluir';} ?>"
20     method="post" enctype="multipart/form-data">

```

Linha 18 – será direcionado para a página de “produto.controller.php” que é referente ao link 4, e será passado a ação que pode mudar dependendo do que receber a variável “\$metodo”, e a ação vai determinar se insere, altera ou apaga os dados do banco de dados.

Inserir no formulário “cadProduto.php” o que está marcado com amarelo.

```

20     method="post" enctype="multipart/form-data">
21     <div class="mb-3">
22         <label>Descrição</label>
23         <input type="text" name="descricao" class="form-control" value="<?php if(isset($descricao)){echo $descricao;}else{echo ' ';} ?>"
24     </div>
25     <div class="mb-3">
26         <label>Valor de custo</label>
27         <input type="text" name="custo" class="form-control" value="<?php if(isset($custo)){echo $custo;}else{echo ' ';} ?>"
28     </div>
29     <div class="mb-3">
30         <label>Fornecedor</label>
31         <input type="text" name="fornecedor" class="form-control" value="<?php if(isset($fornecedor)){echo $fornecedor;}else{echo ' ';} ?>"
32     </div>
33     <div class="mb-3">
34         <label>Foto</label>
35         <input type="file" name="foto" class="form-control" >
36     </div>
37     <?php
38     if(isset($foto)){
39         echo ;
40     }
41     <input type="hidden" name="id" value="<?php if(isset($id)){echo $id;}else{echo ' ';} ?>"
42     <input type="submit" class="btn btn-primary" value="<?php if(isset($metodo)){echo 'inserir';}
43     elseif($metodo == 'alterar'){echo 'alterar';}else{echo 'excluir';} ?>"
44 </form>

```

Linhas 23, 27 e 31 – foi inserido dentro da propriedade “value” uma checagem se a respectivas variáveis existe,

se sim guarda o valor na propriedade, se não, guarda o valor vazio.

Linhas 37 a 41 – mostra a foto do Produto.

Linha 42 – foi criado um campo “hidden”, que não fica visível na página, para salvar o id do produto.

Linha 43 – dependendo do valor da variável “\$metodo” o valor do botão muda.

Para testar, digitar “localhost:8080/” e clique no menu área restrita, em seguida clique em alterar ou excluir, vai abrir o formulário com as informações do produto cadastrado.

[Home](#) [Cadastro de Produto](#) [Área Restrita](#)

Cadastro de Produto

Descrição

Computador Gamer Completo RGB Intel Core i5 8GB HD 1TB

Valor de custo

4500.00


Fornecedor

infotec

Foto

Escolher arquivo

Nenhum arquivo escolhido



alterar

3.6 Alterar dados do produto no banco de dados

Abra o arquivo “produto.service.php” e digite o código da função alterar.

```

43     public function alterar(){
44         $query = 'update produtos
45             set descricao = ?, custo= ?, fornecedor=?, foto=?
46             where id = ?';
47         $stmt = $this->conexao->prepare($query);
48         $stmt->bindValue(1,$this->produto->__get('descricao'));
49         $stmt->bindValue(2,$this->produto->__get('custo'));
50         $stmt->bindValue(3,$this->produto->__get('fornecedor'));
51         $stmt->bindValue(4,$this->produto->__get('foto'));
52         $stmt->bindValue(5,$this->produto->__get('id'));
53         if($stmt->execute()){
54             if($_SESSION['foto']!=$this->produto->__get('foto')){
55                 unlink('fotoProduto\\' . $_SESSION['foto']);
56                 $diretorio = 'fotoProduto/';
57                 move_uploaded_file($_FILES['foto']['tmp_name'],
58                     $diretorio.$this->produto->__get('foto'));
59             }
60         }
61     }

```

Linha 43 – foi criado a função alterar.

Linha 44 – foi criado uma variável query que recebe uma “string”.

Linha 15 – foi criado uma variável “stmt” que recebe o objeto “conexao” que é do tipo “PDO”, junto com o método “prepare” com o parâmetro “query”.

Linhas 47 a 52 – com a variável “stmt” acesso o “bindValue” para tratar a “query” e associar com as interrogações os valores que estão sendo buscado com o método “__get” na classe “Produto”.

Linha 21 – foi criado uma condição que se o “execute” acontecer entra no bloco de instrução, que testa se a “\$_SESSION[‘foto’]” é diferente do que esta salvo na classe “Produto”, se for diferente vai apagar a foto que esta dentro da pasta fotoProduto, e atribui na variável “diretório” o texto “fotoProduto/”, que se junta com o nome da foto que foi enviado na hora do cadastro, assim, formando o caminho para salvar a foto dentro da pasta.

Abra o arquivo “produto.controller.php” e digite o código referente a alterar.

```
37     }
38     if($acao == 'alterar'){
39         $produto = new Produto();
40         $produto->__set('descricao',$_POST['descricao']);
41         $produto->__set('custo',$_POST['custo']);
42         $produto->__set('fornecedor',$_POST['fornecedor']);
43         if($_FILES['foto']['name'] != ''){
44             $produto->__set('foto',$_FILES['foto']['name']);
45         }else{
46             $produto->__set('foto',$_SESSION['foto']);
47         }
48         $produto->__set('id',$_POST['id']);
49
50         $conexao = new Conexao();
51
52         $produtoService = new ProdutoService($produto, $conexao);
53         $produtoService->alterar();
54
55         header('location: index.php?link=1');
56     }
```

Linha 38 – foi feita uma condição para comparar a ação recebida.

Linha 39 – foi instanciado o objeto do tipo “Produto”.

Linhas 40 a 48 – com o objeto “produto” através do método “__set” será salvo nos atributos da classe “Produto” os dados do produto passados pelo formulário de cadastro, foi feita uma condição que testa se a foto vai ser alterada também.

Linha 50 – foi instanciado o objeto do tipo “Conexao”.

Linha 52 – foi instanciado o objeto do tipo “ProdutoService”, e passado como parâmetro os objetos criados “produto” e “conexão”.

Linha 53 – com o objeto “produtoService” é chamado o método “alterar”.

Linha 55 – redireciona para página “index.php”.

Para testar, digitar “localhost:8080/” e clique no menu área restrita, em seguida clique em alterar ou excluir, vai abrir o formulário com as informações do produto cadastrado, altere as informações e a imagem, verifique se a foto foi alterada na pasta “fotoProduto”.

3.7 Excluir dados do produto no banco de dados

Abra o arquivo “produto.service.php” e digite o código da função excluir.

```
61     }
62     public function excluir(){
63         $query = 'delete from produtos where id = ?';
64         $stmt = $this->conexao->prepare($query);
65         $stmt->bindValue(1,$this->produto->__get('id'));
66         if($stmt->execute()){
67             unlink('fotoProduto\\' . $_SESSION['foto']);
68         }
69     }
70 }
```

Linha 62 – foi criado a função excluir.

Linha 63 – foi criado uma variável query que recebe uma “string”.

Linha 64 – foi criado uma variável “stmt” que recebe o objeto “conexao” que é do tipo “PDO”, junto com o método “prepare” com o parâmetro “query”.

Linha 65 – com a variável “stmt” acesso o “bindValue” para tratar a “query” e associar com as interrogações os valores que estão sendo buscado com o método “__get” na classe “Produto”.

Linha 21 – foi criado uma condição que se o “execute” acontecer paga a foto que está dentro da pasta fotoProduto.

Abra o arquivo “produto.controller.php” e digite o código referente a excluir.

```
56     }
57     if($acao == 'excluir'){
58         $produto = new Produto();
59         $produto->__set('id',$_POST['id']);
60
61         $conexao = new Conexao();
62
63         $produtoService = new ProdutoService($produto, $conexao);
64         $produtoService->excluir();
65
66         header('location: index.php?link=1');
67     }
```

Linha 57 – foi feita uma condição para comparar a ação recebida.

Linha 58 – foi instanciado o objeto do tipo “Produto”.

Linha 59 – com o objeto “produto” através do método “__set” será salvo nos atributos da classe “Produto” os dados do produto passado pelo formulário de cadastro.

Linha 61 – foi instanciado o objeto do tipo “Conexao”.

Linha 63 – foi instanciado o objeto do tipo “ProdutoService”, e passado como parâmetro os objetos criados “produto” e “conexao”.

Linha 64 – com o objeto “produtoService” é chamado o método “excluir”.

Linha 66 – redireciona para página “index.php”.

Para testar, digitar “localhost:8080/” e clique no menu cadastrar produto e realize o cadastro com foto e clique em inserir.

< > ↺ 🍷 | VPN 🌐 localhost:8080/index.php

Home Cadastro de Produto Área Restrita

Cadastro de Produto

Descrição

Computador Gamer Completo RGB Intel Core i5 8GB HD 1TB

Valor de custo

4500.00

Fornecedor

infotec

Foto

Escolher arquivo computador1.jpg

inserir

Clique em no menu área restrita para visualizar os dados do produto cadastrado.

Faça o teste para alterar e excluir o registro do produto.

3.8 Mostando os produtos na index

Abra o arquivo “produto.service.php” e digite o código da função pesquisar.

```
70     public function pesquisar(){
71         $query = 'select id, descricao, custo, fornecedor, foto
72                 from produtos limit 5';
73         $stmt=$this->conexao->prepare($query);
74         $stmt->execute();
75         return $stmt->fetchALL(PDO::FETCH_OBJ);
76     }
77 }
```

Linha 70 – foi criado a função para recuperar 5 produtos do banco de dados

Linha 71 – foi declarado uma variável para receber um texto, na linguagem sql.

Linha 73 - foi criado uma variável “stmt” que recebe o objeto “conexao” que é do tipo “PDO”, junto com o método “prepare” com o parâmetro “query”.

Linha 74 – executa o código sql no bando de dados.

Linha 75 – retorna em formato de objeto os 5 primeiros produtos cadastrados no banco de dados.

Abrir o arquivo “produto.controller.php” e digitar o código referente a recuperar.

```
67     }
68     if($acao == 'pesquisar'){
69         $produto = new Produto();
70         $conexao = new Conexao();
71
72         $produtoService = new ProdutoService($produto, $conexao);
73         $produto= $produtoService->pesquisar();
74     }
```

Linha 68 – foi realizado uma condição se ação for igual a pesquisar.

Linha 69 – foi instanciado o objeto “produto”.

Linha 70 – foi instanciado o objeto “conexao”.

Linha – 72 foi instanciado o objeto do tipo “ProdutoService”, e passado como parâmetro os objetos criados “produto” e “conexao”.

Linha 73 – com o objeto “produtoService” é chamado o método “pesquisar”, as informações que serão retornadas serão salvas dentro da variável “produto”.

Abrir o arquivo “principal.php”, e faça as alterações necessárias.

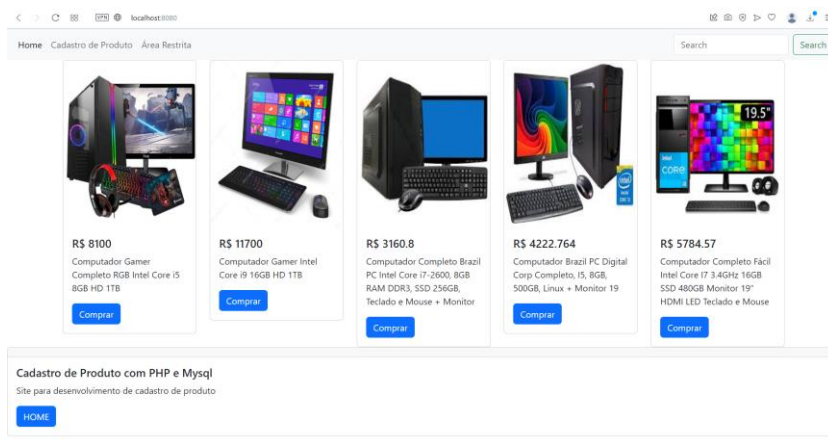
```
principal.php X
principal.php
1 <div class="row">
2   <div class="col">
3     <div class="card" style="width: 100%;">
4       
5       <div class="card-body">
6         <h5 class="card-title">Card title</h5>
7         <p class="card-text">Some quick example text to build on the card title and make up the bulk of the ca
8         <a href="#" class="btn btn-primary">Go somewhere</a>
9       </div>
10    </div>
11  </div>
12 </div>
```

Apagar o código da linha 12 até 53, e modifique o código como a imagem abaixo.

```
principal.php X
principal.php
1 <?php
2   $acao = 'pesquisar';
3   require_once 'produto.controller.php';
4 >
5 <div class="row">
6   <?php foreach($produto as $indice =>$produto) { ?>
7     <div class="col">
8       <div class="card" style="width: 100%;">
9         
10        <div class="card-body">
11          <h5 class="card-title">R$ <?= $produto->custo * 1.8 ;?></h5>
12          <p class="card-text"><?= $produto->descricao;?></p>
13          <a href="#" class="btn btn-primary">Comprar</a>
14        </div>
15      </div>
16    </div>
17  <?php } ?>
18 </div>
```

Para testar, digitar "localhost:8080/" os produto vão aparecer.

PHP COM MYSQL – Cadastro de usuário com foto, por André Luis de Souza Silva



O cadastro de produto está pronto, com esse conhecimento você pode realizar outros cadastros e desenvolver um site completo.