

Curso Técnico Superior Profissional em: Tecnologias e Programação de Sistemas de Informação

2º Ano/1º Semestre

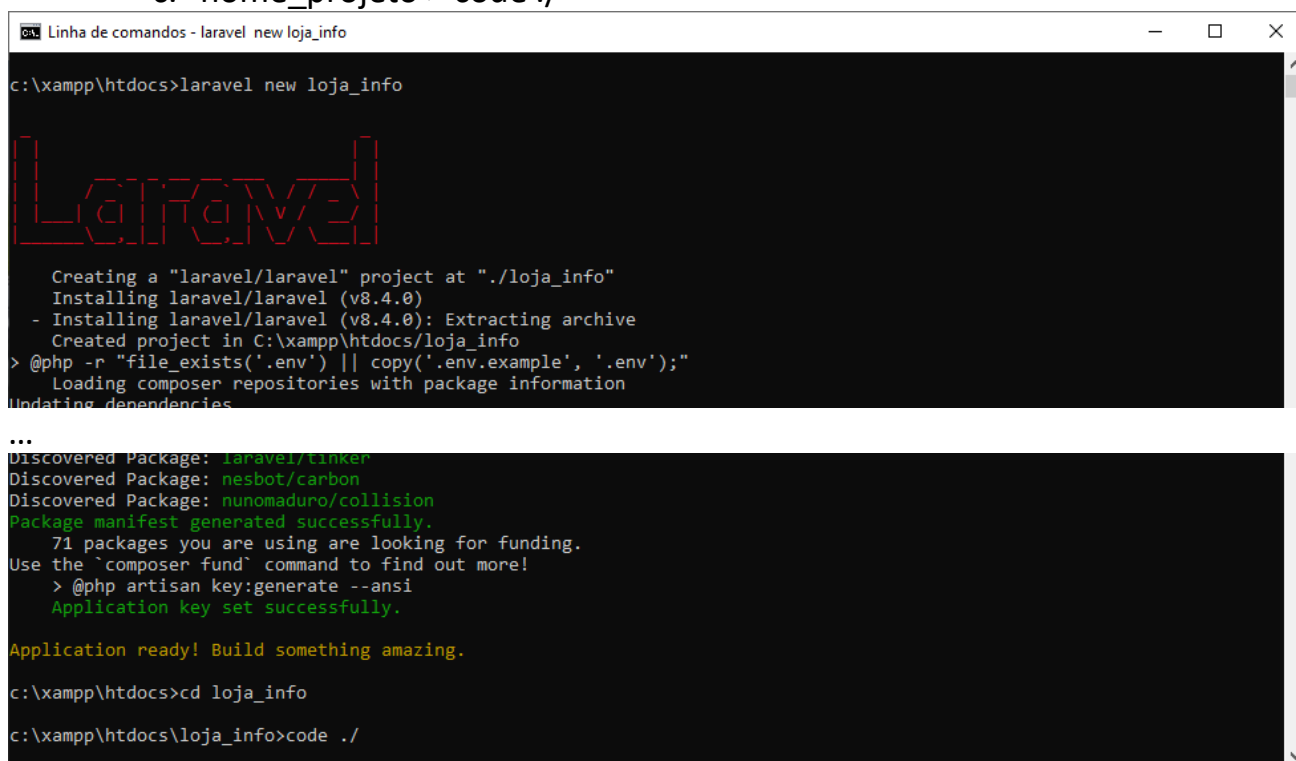
Unidade Curricular: Aplicações Centradas em Redes

Docente: Michael Silva / Hugo Perdigão

INTRODUÇÃO ÀS LINGUAGENS PARA O DESENVOLVIMENTO DE APLICAÇÕES CENTRADAS EM REDES

Laravel - Exemplo

- 1) Criar o projeto e abrir no VSCode:
 - a. > Laravel new nome_projeto
 - b. > cd nome_projeto
 - c. nome_projeto > code ./



```
Linha de comandos - laravel new loja_info

c:\xampp\htdocs>laravel new loja_info

Laravel

Creating a "laravel/laravel" project at "./loja_info"
Installing laravel/laravel (v8.4.0)
- Installing laravel/laravel (v8.4.0): Extracting archive
Created project in C:\xampp\htdocs\loja_info
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies
...
Discovered Package: laravel/tinker
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Package manifest generated successfully.
71 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
> @php artisan key:generate --ansi
Application key set successfully.

Application ready! Build something amazing.

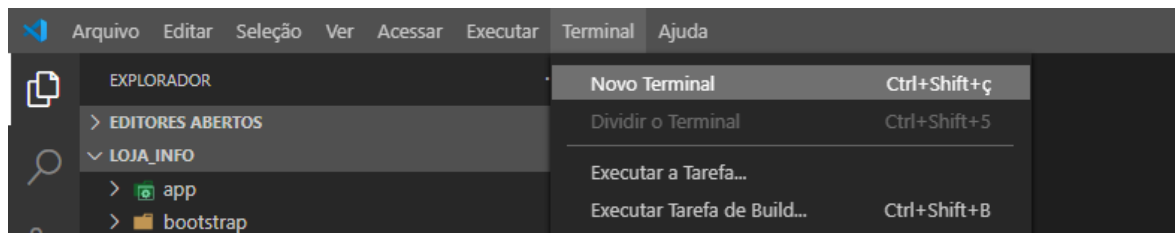
c:\xampp\htdocs>cd loja_info

c:\xampp\htdocs\loja_info>code ./
```

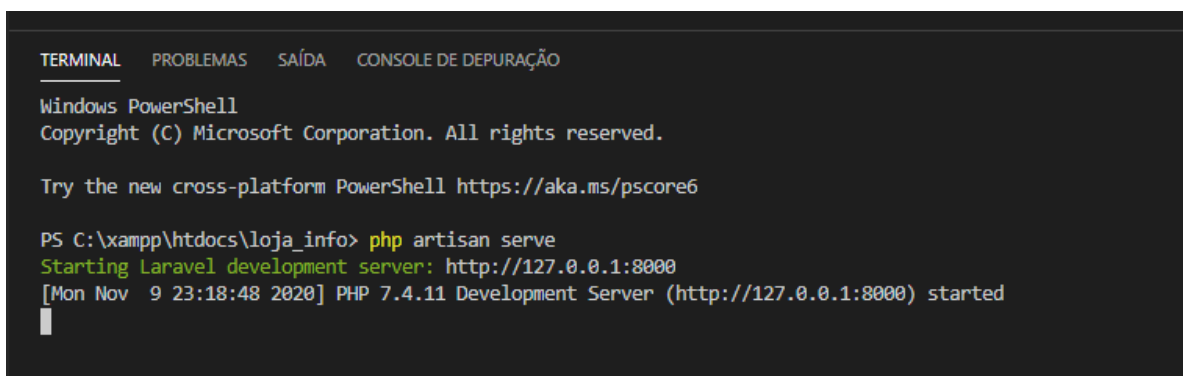
Cofinanciado por:

2) Iniciar o servidor web

a. Abrir um terminal no VSCode:

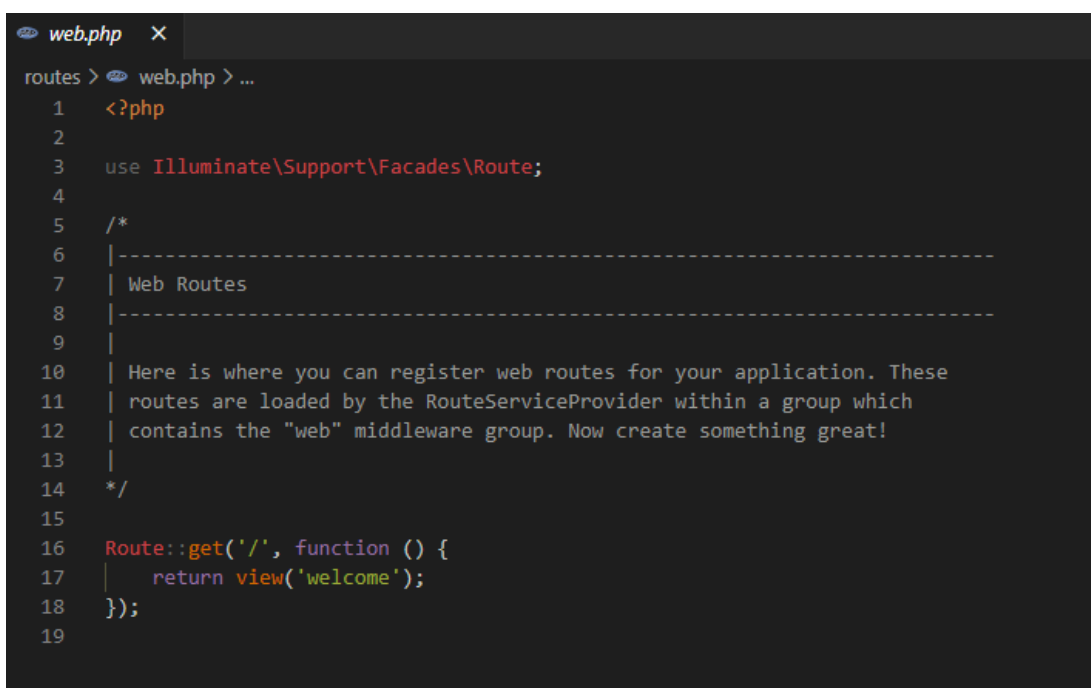


b. Executar `php artisan serve` (click na ligação para abrir no browser).



3) Para criar routes

a. Abrir o ficheiro `/routes/web.php`



Cofinanciado por:



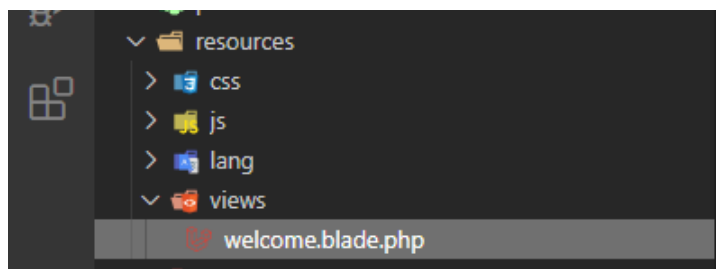
- b. Já está especificada a Route '/', vamos especificar duas novas Routes '/produtos' e '/detalhes'.

```
web.php
routes > web.php > ...
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4
5  /*
6  |-----
7  | Web Routes
8  |-----
9  |
10 | Here is where you can register web routes for your application. These
11 | routes are loaded by the RouteServiceProvider within a group which
12 | contains the "web" middleware group. Now create something great!
13 |
14 */
15
16 Route::get('/', function ()
17 {
18     return view('welcome');
19 });
20
21 Route::get('/produtos', function ()
22 {
23     return view('produtos');
24 });
25
26 Route::get('/detalhes', function ()
27 {
28     return view('detalhes');
29 });
30
```

NOTA: Não basta criar as Routes, estas não vão funcionar se não existirem views associadas

4) Para criar Views

- a. As views devem ficar em /resources/views (Já está criada a view 'welcome')



- b. As views podem ser apenas HTML Puro, vamos começar por criar apenas as views um título (h1) em cada uma com o nome da página.

Cofinanciado por:

i. welcome.blade.php (eliminamos todo o conteúdo do body)

```
resources > views > welcome.blade.php > ...
1 <!DOCTYPE html>
2 <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
3   <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1">
6
7     <title>Laravel</title>
8
9     <!-- Fonts -->
10    <link href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&display=sw
11
12    <!-- Styles -->
13    <style>
14      /*! normalize.css v8.0.1 | MIT License | github.com/necolas/normalize.css */html{li
15    </style>
16
17    <style>
18      body {
19        font-family: 'Nunito';
20      }
21    </style>
22  </head>
23  <body class="antialiased">
24    <h1>Loja de Informática</h1>
25  </body>
26 </html>
27
```

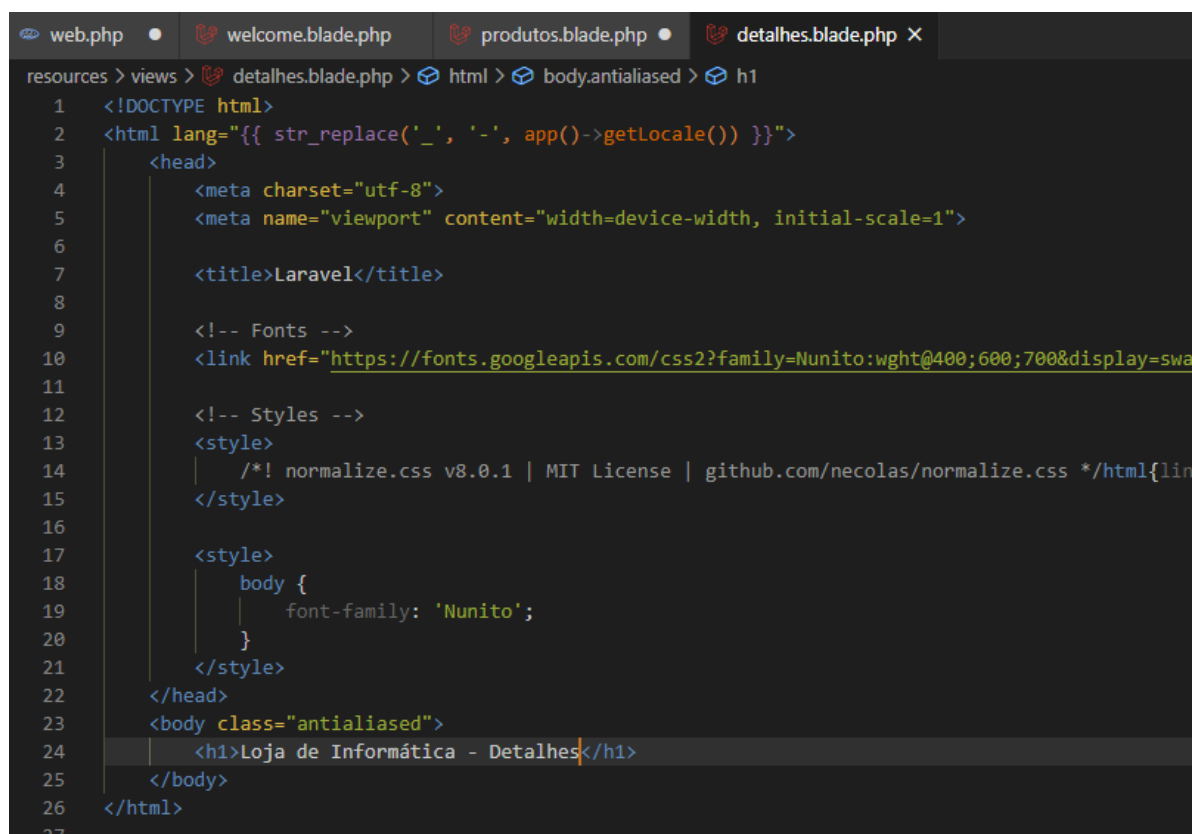
ii. produtos.blade.php (copiamos a view welcome e alteramos o título)

```
resources > views > produtos.blade.php > html > body.antialiased > h1
1 <!DOCTYPE html>
2 <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
3   <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1">
6
7     <title>Laravel</title>
8
9     <!-- Fonts -->
10    <link href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&display=sw
11
12    <!-- Styles -->
13    <style>
14      /*! normalize.css v8.0.1 | MIT License | github.com/necolas/normalize.css */html{li
15    </style>
16
17    <style>
18      body {
19        font-family: 'Nunito';
20      }
21    </style>
22  </head>
23  <body class="antialiased">
24    <h1>Loja de Informática - Produtos</h1>
25  </body>
26 </html>
27
```

Cofinanciado por:

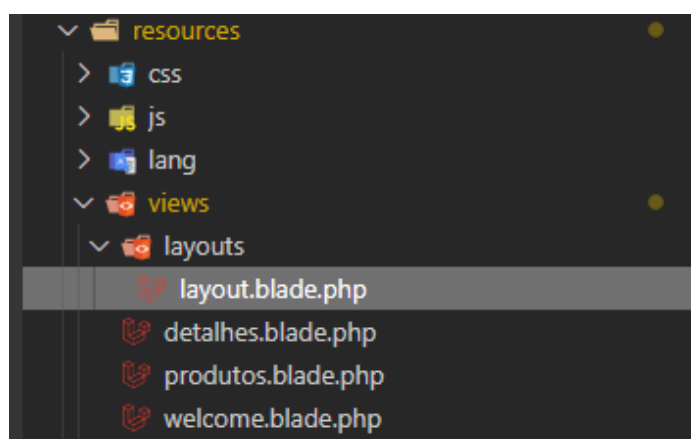


iii. detalhes.blade.php



```
resources > views > detalhes.blade.php > html > body.antialiased > h1
1 <!DOCTYPE html>
2 <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
3     <head>
4         <meta charset="utf-8">
5         <meta name="viewport" content="width=device-width, initial-scale=1">
6
7         <title>Laravel</title>
8
9         <!-- Fonts -->
10        <link href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&display=swa
11
12        <!-- Styles -->
13        <style>
14            /*! normalize.css v8.0.1 | MIT License | github.com/necolas/normalize.css */html{lin
15        </style>
16
17        <style>
18            body {
19                font-family: 'Nunito';
20            }
21        </style>
22    </head>
23    <body class="antialiased">
24        <h1>Loja de Informática - Detalhes</h1>
25    </body>
26 </html>
27
```

- c. Para garantir que todas as páginas ficam com o mesmo aspeto, podemos usar um layout.
 - i. Criar uma pasta 'layouts' dentro da 'views'.
 - ii. Copiar o ficheiro 'welcome.blade.php' para dentro da pasta
 - iii. Mudar o nome do ficheiro para 'layout.blade.php'



Cofinanciado por:



- iv. No template do layout criar um cabeçalho e um rodapé (comuns a todas as páginas)
- v. Criar uma secção com a diretiva @Yield que deve ser declarada nas restantes views.

```
resources > views > layouts > layout.blade.php > html > body
1  <!DOCTYPE html>
2  <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
3      <head>
4          <meta charset="utf-8">
5          <meta name="viewport" content="width=device-width, initial-scale=1">
6
7          <title>Laravel</title>
8
9          <!-- Fonts -->
10         <link href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&display=swa
11
12         <!-- Styles -->
13         <style>
14             /*! normalize.css v8.0.1 | MIT License | github.com/necolas/normalize.css */html{lin
15         </style>
16
17         <style>
18             body {
19                 font-family: 'Nunito';
20             }
21         </style>
22     </head>
23     <body>
24         <header>
25             Loja de Informática
26         </header>
27         <div id="contentArea">
28             @yield('content')
29         </div>
30         <footer>
31
32         </footer>
33     </body>
34 </html>
```

Cofinanciado por:



- vi. Depois alteramos as Views para herdarem todo o layout, declarando apenas o conteúdo que deve ir na secção 'content'

```
resources > views > welcome.blade.php > ...
1  @extends('layouts.layout')
2
3  @section('content')
4      <h1>Loja de Informática</h1>
5  @endsection
6
```

```
resources > views > produtos.blade.php > ...
1  @extends('layouts.layout')
2
3  @section('content')
4      <h1>Loja de Informática - Produtos</h1>
5  @endsection
6
```

```
resources > views > detalhes.blade.php > h1
1  @extends('layouts.layout')
2
3  @section('content')
4      <h1>Loja de Informática - Detalhes</h1>
5  @endsection
6
```

5) Para passar informação entre páginas.

NOTA: Vamos criar um array para simular uma tabela da base de dados.

- a. Podemos passar variáveis para uma view, num array, neste caso, a chave 'produtos' vai levar um array com informações sobre os produtos.

```
21 Route::get('/produtos', function ()
22 {
23     $arrayProdutos = [
24         ["id" => 1, "nome" => "PC Gamer", "desc" => "PC Gamer Top", "img" => "/img/produtos/1.jpg", "preco" => 1000 ],
25         ["id" => 2, "nome" => "PC Workstation", "desc" => "PC para trabalho", "img" => "/img/produtos/2.jpg", "preco" => 600 ],
26         ["id" => 3, "nome" => "PC Bonzinho", "desc" => "PC Bonzinho, desenrasca", "img" => "/img/produtos/3.jpg", "preco" => 400 ],
27         ["id" => 4, "nome" => "PC Rasca", "desc" => "PC mesmo muito fraco", "img" => "/img/produtos/4.jpg", "preco" => 250 ]
28     ];
29
30     return view('produtos', ['produtos' => $arrayProdutos]);
31 });
```

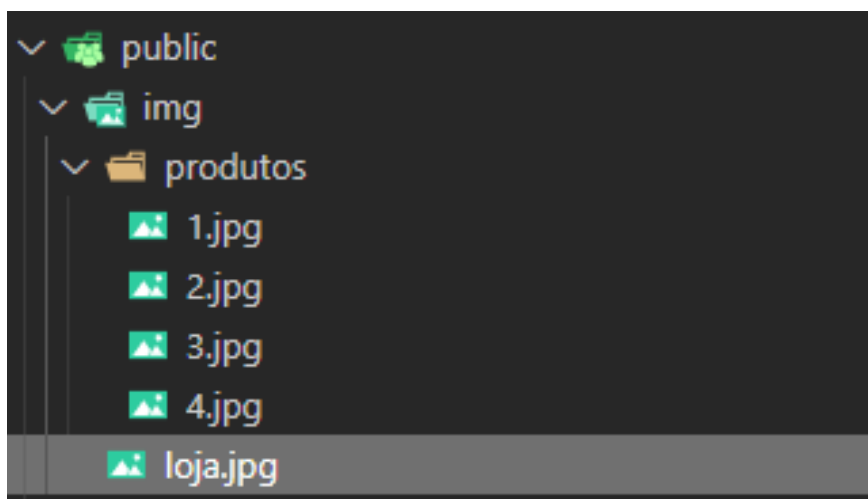
Cofinanciado por:



Na View (produtos) é criada uma variável com o nome que foi enviado para esta ('produtos' na view é a variável \$produtos). O Blade, permite-nos utilizar um ciclo foreach diretamente no HTML tal e qual como no php.

```
web.php  produtos.blade.php X
resources > views > produtos.blade.php > ...
1  @extends('layouts.layout')
2
3  @section('content')
4      <h1>Loja de Informática - Produtos</h1>
5
6      @foreach ($produtos as $produto)
7          <div class="produto">
8              <a href="/detalhes?id={{ $produto['id'] }}">
9                  
10                 <h2>{{ $produto['nome'] }}</h2>
11             </a>
12         </div>
13     @endforeach
14 @endsection
15
```

NOTA: para a página ficar melhor vamos transferir algumas imagens da Net (uma para a página principal e uma para cada produto)



Cofinanciado por:



Aproveitamos para colocar uma imagem e link na página inicial

```
web.php welcome.blade.php X produtos.blade.php
resources > views > welcome.blade.php > ...
1  @extends('layouts.layout')
2
3  @section('content')
4      <h1>Loja de Informática</h1>
5      <div class="intro">
6          
7
8          <a href="/produtos">Ver Produtos</a>
9      </div>
10 @endsection
11
```

- b. Na view dos produtos, cada produto tem uma ligação para a página detalhes, passando o id por GET. Vamos alterar a nossa route para aceitar este valor e pesquisar no array e retornar o produto correspondente

NOTA: apenas para efeitos de teste, apesar de não ser uma boa abordagem, vamos duplicar o array para poder efetuar a pesquisa

```
33 Route::get('/detalhes', function ()
34 {
35     //recebe parametro GET/POST
36     $prodId = request('id');
37     //Array de produtos
38     $arrayProdutos = [
39         ["id" => 1, "nome" => "PC Gamer", "desc" => "PC Gamer Top", "img" => "/img/produtos/1.jpg", "preco" => 1000 ],
40         ["id" => 2, "nome" => "PC Workstation", "desc" => "PC para trabalho", "img" => "/img/produtos/2.jpg", "preco" => 600 ],
41         ["id" => 3, "nome" => "PC Bonzinho", "desc" => "PC Bonzinho, desenrasca", "img" => "/img/produtos/3.jpg", "preco" => 400 ],
42         ["id" => 4, "nome" => "PC Rasca", "desc" => "PC mesmo muito fraco", "img" => "/img/produtos/4.jpg", "preco" => 250 ]
43     ];
44     //pesquisa produto no array
45     $produtoSelecioneado = NULL;
46     foreach($arrayProdutos as $linhaProduto)
47     {
48         if($linhaProduto['id'] == $prodId)
49         {
50             $produtoSelecioneado = $linhaProduto;
51         }
52     }
53
54     return view('detalhes', ['produto' => $produtoSelecioneado]);
55 });
```

Cofinanciado por:



6) Para utilizar um wildcard no URL

- a. vamos alterar a Route, em vez de detalhes com um parâmetro GET, vamos passar um id do produto através do URL /produtos/X onde X é o ID associado.

```
33 Route::get('/produtos/{id}', function ()
```

- b. Também alterar os links associados na view 'produtos'

```
6     @foreach ($produtos as $produto)
7         <div class="produto">
8             <a href="/produtos/{{ $produto['id'] }}">
9                 
10                <h2>{{ $produto['nome'] }}</h2>
11            </a>
12        </div>
13    @endforeach
```

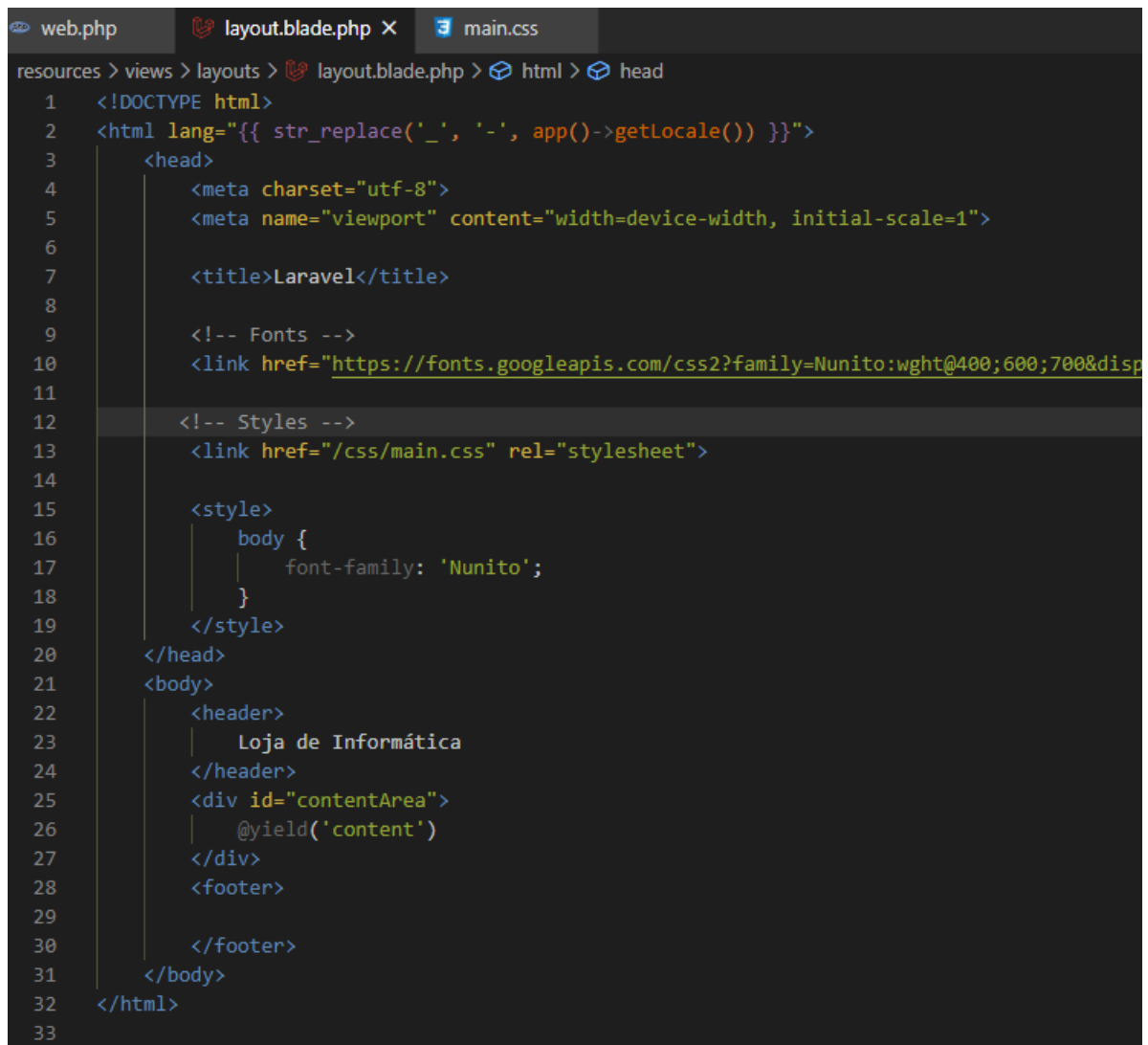
7) Na view detalhes, vamos mostrar os detalhes do produto

```
web.php • 2020_11_10_141106_create_products_table.php • detalhes.blade.php
resources > views > detalhes.blade.php > ...
1  @extends('layouts.layout')
2
3  @section('content')
4      <h1>Loja de Informática - Detalhes</h1>
5      <div class="detalhes">
6          @if (isset($produto))
7              
8              <h2>{{ $produto['nome'] }}</h2>
9              <p>{{ $produto['nome'] }}<br/>
10             € {{ $produto['preco'] }}</p>
11          @else
12              <h1>O produto não existe</h1>
13          @endif
14          <a href="/produtos">voltar aos produtos</a>
15      </div>
16  @endsection
17
```

Cofinanciado por:



- 8) Para visualizar um pouco melhor, vamos aplicar alguns estilos criando uma css e incluindo no layout.



```
web.php layout.blade.php X main.css
resources > views > layouts > layout.blade.php > html > head
1 <!DOCTYPE html>
2 <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
3   <head>
4     <meta charset="utf-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1">
6
7     <title>Laravel</title>
8
9     <!-- Fonts -->
10    <link href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&disp
11
12    <!-- Styles -->
13    <link href="/css/main.css" rel="stylesheet">
14
15    <style>
16      body {
17        font-family: 'Nunito';
18      }
19    </style>
20  </head>
21  <body>
22    <header>
23      Loja de Informática
24    </header>
25    <div id="contentArea">
26      @yield('content')
27    </div>
28    <footer>
29
30    </footer>
31  </body>
32 </html>
33
```

Cofinanciado por:



a. Estilos a copiar para a css (/public/css/main.css):

```
header
{
  z-index: 9999;
  position: fixed;
  top:0;
  left: 0;
  right: 0;
  height: 10vh;
  background-color:brown;
}

footer
{
  z-index: 9999;
  text-align: center;
  line-height: 10vh;
  position: fixed;
  bottom: 0;
  left: 0;
  right: 0;
  height: 10vh;
  background-color: brown;
}

#contentArea
{
  text-align: center;
  position: fixed;
  top: 10vh;
  bottom: 10vh;
  left:0;
  right: 0;
  overflow-y: auto;
}

h1
{
  text-align: center;
  font-size: 5vh;
  line-height: 10vh;
}

.produto
{
```

Cofinanciado por:



```

display: inline-block;
width: 15vw;
vertical-align: top;
}

.produto img
{
width: 15vw;
}

```

7) Vamos agora criar um Controller para centralizar toda a lógica relativa aos produtos

a. No terminal

```

TERMINAL  PROBLEMAS  2  SAÍDA  CONSOLE DE DEPURAÇÃO

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\xampp\htdocs\loja_info> php artisan make:controller ProdutosController
Controller created successfully.
PS C:\xampp\htdocs\loja_info>

```

b. Vamos passar toda a lógica relativa aos produtos das Routes para o controller

i. Criando o array (simulação de BD como variável provada do controller) e criando as funções index() e show():

```

web.php  produtos.blade.php  layout.blade.php  main.css  ProdutosController.php X
app > Http > Controllers > ProdutosController.php > ProdutosController > index
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class ProdutosController extends Controller
8  {
9      private $arrayProdutos = [
10         ["id" => 1, "nome" => "PC Gamer", "desc" => "PC Gamer Top", "img" => "/img/produtos/1.jpg", "pr
11         ["id" => 2, "nome" => "PC Workstation", "desc" => "PC para trabalho", "img" => "/img/produtos/
12         ["id" => 3, "nome" => "PC Bonzinho", "desc" => "PC Bonzinho, desenrasca", "img" => "/img/produ
13         ["id" => 4, "nome" => "PC Rasca", "desc" => "PC mesmo muito fraco", "img" => "/img/produtos/4.
14     ];
15
16     public function index()
17     {
18         return view('produtos', ['produtos' => $this->arrayProdutos]);
19     }
20
21     public function show($id)
22     {
23         $produtoSelecioneado = NULL;
24         foreach($this->arrayProdutos as $linhaProduto)
25         {
26             if($linhaProduto['id'] == $id)
27             {
28                 $produtoSelecioneado = $linhaProduto;
29             }
30         }
31
32         return view('detalhes', ['produto' => $produtoSelecioneado]);
33     }
34 }

```

Cofinanciado por:

- c. Temos também que atualizar as Routes de forma a utilizar as funções do controller.

```
web.php x produtos.blade.php layout.blade.php main.css Proc
routes > web.php > ...
2
3 use App\Http\Controllers\ProdutosController;
4 use Illuminate\Support\Facades\Route;
5
6 /*
7 |-----
8 | Web Routes
9 |-----
10 |
11 | Here is where you can register web routes for your application. These
12 | routes are loaded by the RouteServiceProvider within a group which
13 | contains the "web" middleware group. Now create something great!
14 |
15 */
16
17 Route::get('/', function ()
18 {
19     return view('welcome');
20 });
21
22 Route::get('/produtos', [ProdutosController::class, 'index']);
23 Route::get('/produtos/{id}', [ProdutosController::class, 'show']);
24
```

Bom Trabalho!

Cofinanciado por:

