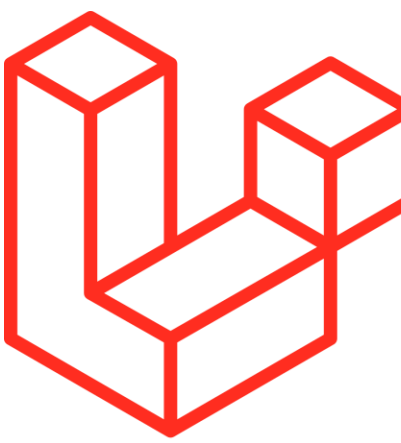


INTRODUÇÃO AO LARAVEL

CONTINUANDO O PRIMEIRO PROJETO - CRUD (READ, UPDATE E DELETE)

PROF. ESP. JEFERSON ROBERTO DE LIMA - JEFERSON.LIMA17@ETEC.SP.GOV.BR

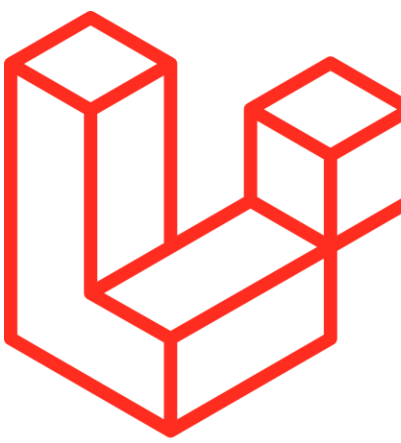
INTRODUÇÃO AO LARAVEL



CONTINUANDO O NOSSO PROJETO LARAVEL

- Hoje vamos continuar o nosso projeto Laravel com banco de dados.
- Vamos resgatar o nosso projeto do GitHub acessando o CMD e digitando o seguinte comando:
 - `cd c:/xampp/htdocs/`
- Para realizarmos o download de um projeto no GitHub será necessário digitar o seguinte comando no cmd:
 - `git clone https://github.com/jefersonrl/biblioteca-laravel`
- Após a realização desse comando daremos início a configuração das dependências do nosso projeto.

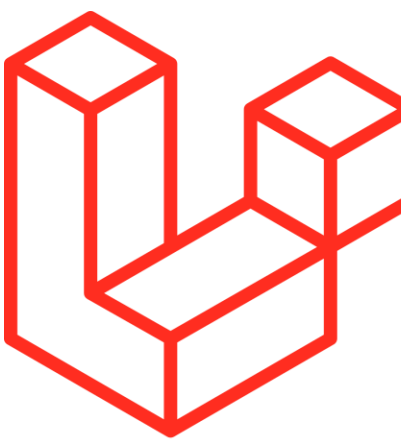
INTRODUÇÃO AO LARAVEL



CONTINUANDO O NOSSO PROJETO LARAVEL

- Acessando o diretório que acabamos de fazer download do GitHub, executaremos o seguinte comando no CMD:
 - `composer install`
- Ao executar o comando acima o composer acessa o arquivo `composer.json` verifica as bibliotecas utilizadas e realiza os respectivos downloads no diretório `vendor`.

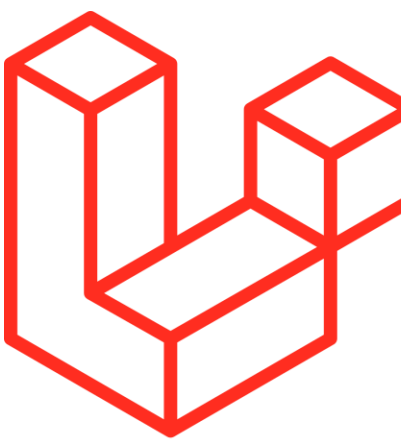
INTRODUÇÃO AO LARAVEL



CONTINUANDO O NOSSO PROJETO LARAVEL

- Ao realizar o clone do projeto do GitHub o mesmo deixa como exemplo o arquivo `.env.example` para informar a conexão com a base de dados.
- Nesse momento você deve renomear o arquivo para `.env` e definir os dados de acesso ao banco.
- Com o XAMPP aberto e as modalidades Apache e MySQL habilitada, acesse o phpmyadmin e crie a base de dados `biblioteca_laravel`.

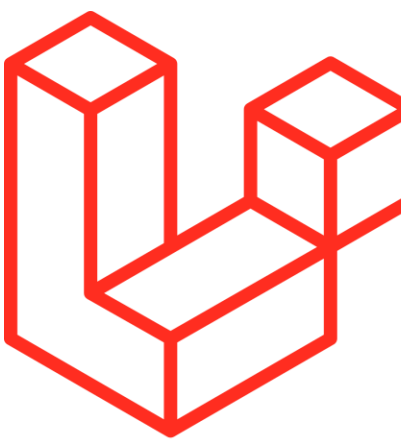
INTRODUÇÃO AO LARAVEL



CONTINUANDO O NOSSO PROJETO LARAVEL

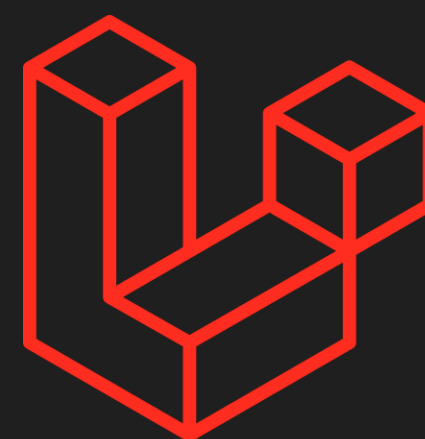
- Antes de iniciarmos o nosso projeto via CMD, é importante gerarmos uma Key de acesso a base através do comando:
 - `php artisan key:generate`
- Em seguida executaremos um comando para limpar todas as propriedades armazenadas no cache:
 - `php artisan config:cache`

INTRODUÇÃO AO LARAVEL



TESTANDO A NOSSA APLICAÇÃO

- Para testarmos o funcionamento da aplicação vamos acessar o terminal e digitar o seguinte comando:
 - `php artisan serve`
- E vamos acessar o browser com o seguinte endereço:
 - <http://127.0.0.1:8000>



PROBLEMAS

SAÍDA

CONSOLE DE DEPURAÇÃO

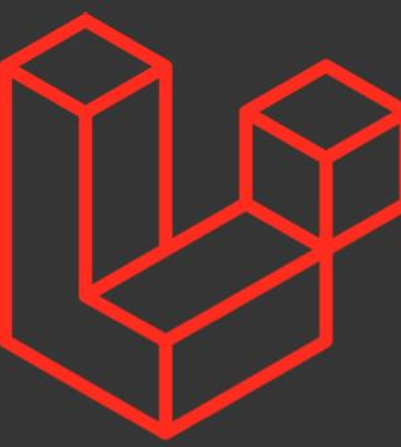
TERMINAL

Windows PowerShell

Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Experimente a nova plataforma cruzada PowerShell <https://aka.ms/pscore6>

PS C:\xampp\htdocs\biblioteca_laravel> **php** artisan serve



← → ↻ ⓘ 127.0.0.1:8000

Produtos

Nome:

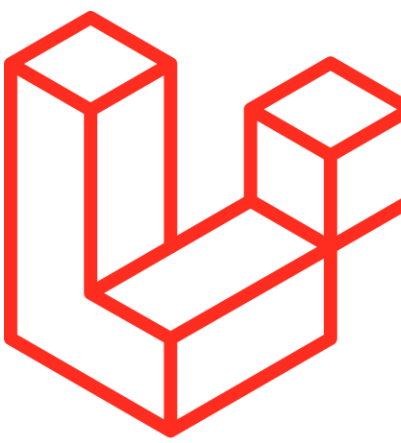
Valor:

Quantidade:

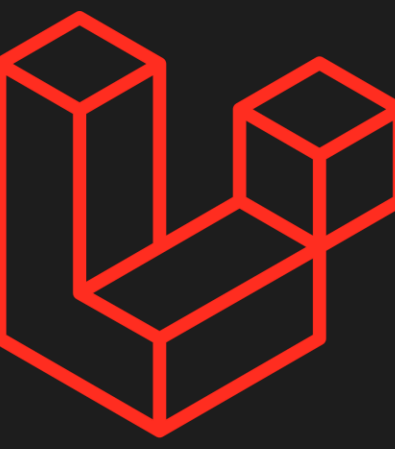
Cadastrar

INTRODUÇÃO AO LARAVEL

CRIANDO O READ DO PROJETO LARAVEL



- A partir de agora vamos criar uma rota para tela de listagem dos dados armazenados em nosso banco.
- No diretório routes vamos editar o arquivo web.php

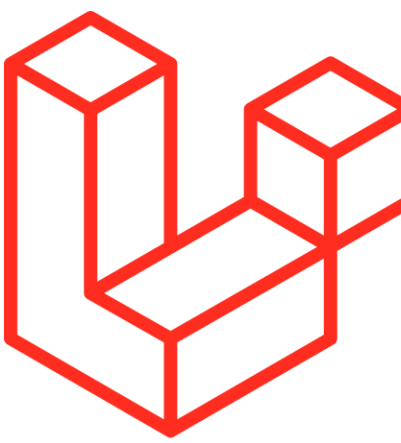


biblioteca_laravel > routes > web.php

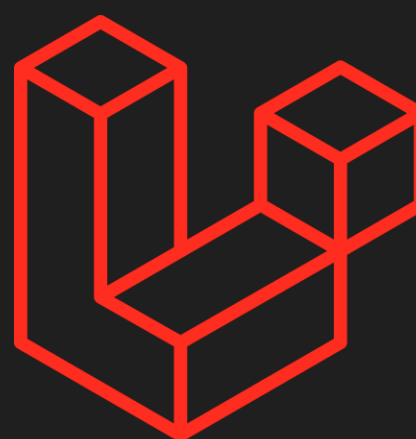
```
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4
5  /*
6  |-----
7  | Web Routes
8  |-----
9  |
10 | Here is where you can register web routes for your application. These
11 | routes are loaded by the RouteServiceProvider within a group which
12 | contains the "web" middleware group. Now create something great!
13 |
14 */
15
16 use App\Models\Produto;
17 use Illuminate\Http\Request;
18
19 Route::get('/', function () {
20     return view('inicio');
21 });
22
23 Route::post('/cadastrar-produto', function(Request $request){
24     //dd($request->all());
25
26     Produto::create([
27         'nome' => $request->nome,
28         'valor' => $request->valor,
29         'estoque' => $request->estoque
30     ]);
31
32     echo "Produto criado com sucesso!";
33 });
34
35 Route::get('/listar-produto/{id}', function($id){
36     //dd(Produto::find($id)); //debug and die
37     $produto = Produto::find($id);
38     return view('listar', ['produto' => $produto]);
39 });
40
```


INTRODUÇÃO AO LARAVEL

CRIANDO O READ DO PROJETO LARAVEL



- Com a rota criada vamos desenvolver a tela que receberá os dados do banco.
- No diretório resources/views vamos criar o arquivo listar.blade.php e aplicaremos os seguintes comandos.

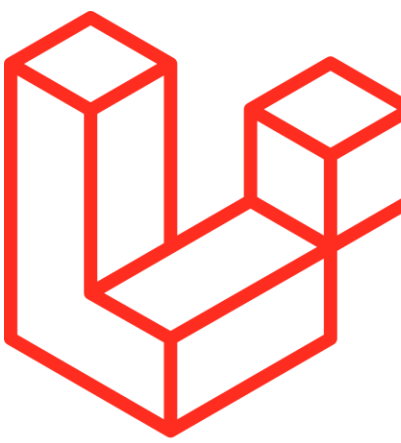


biblioteca_laravel > resources > views >  listar.blade.php

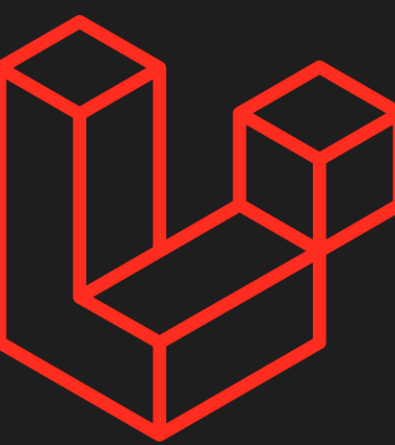
```
1  <!DOCTYPE html>
2  <html lang="pt">
3      <head>
4          <meta charset="utf-8">
5          <meta name="viewport" content="width=device-width, initial-scale=1">
6          <title>Produtos</title>
7      </head>
8      <body>
9          <h1>Produtos</h1>
10
11          <label for="lblNome">Nome:</label>
12          <input type="text" name="nome" value="{{ $produto->nome }}">
13          <br><br>
14          <label for="lblValor">Valor:</label>
15          <input type="text" name="valor" value="{{ $produto->valor }}">
16          <br><br>
17          <label for="lblQuantidade">Quantidade:</label>
18          <input type="text" name="estoque" value="{{ $produto->estoque }}">
19          <br><br>
20      </body>
21  </html>
22
```

INTRODUÇÃO AO LARAVEL

CRIANDO O UPDATE DO PROJETO LARAVEL



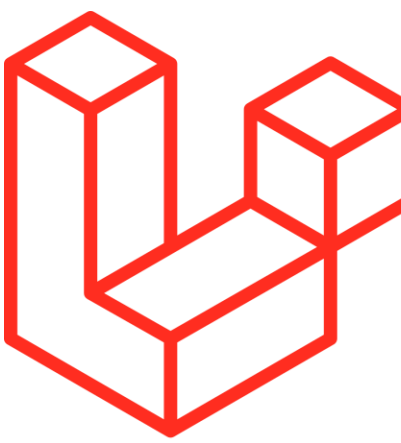
- A partir de agora vamos criar a rota para tela de edição dos dados armazenados em nosso banco.
- No diretório routes vamos editar o arquivo web.php



biblioteca_laravel > routes > web.php

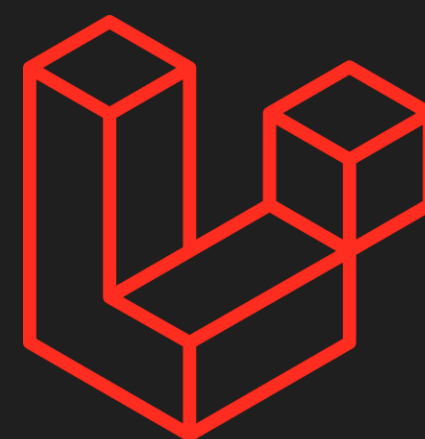
```
28         'valor' => $request->valor,
29         'estoque' => $request->estoque
30     ]);
31
32     echo "Produto criado com sucesso!";
33 });
34
35 Route::get('/listar-produto/{id}', function($id){
36     //dd(Produto::find($id)); //debug and die
37     $produto = Produto::find($id);
38     return view('listar', ['produto' => $produto]);
39 });
40
41 Route::get('/editar-produto/{id}', function($id){
42     //dd(Produto::find($id)); //debug and die
43     $produto = Produto::find($id);
44     return view('editar', ['produto' => $produto]);
45 });
46
47 Route::post('/editar-produto/{id}', function(Request $request, $id){
48     //dd($request->all());
49     $produto = Produto::find($id);
50
51     $produto->update([
52         'nome' => $request->nome,
53         'valor' => $request->valor,
54         'estoque' => $request->estoque
55     ]);
56
57     echo "Produto editado com sucesso!";
58 });
59
```

INTRODUÇÃO AO LARAVEL



CRIANDO O UPDATE DO PROJETO LARAVEL

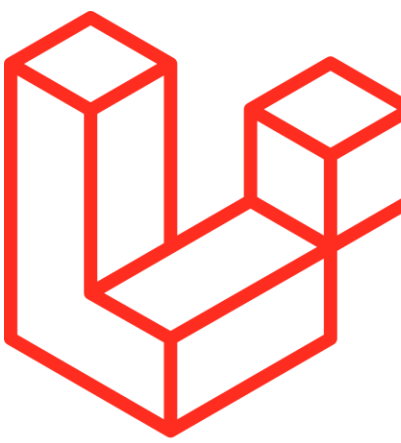
- Agora vamos criar a tela para editar os dados armazenados em nosso banco.
- No diretório resources/views vamos criar o arquivo editar.blade.php e aplicaremos os seguintes comandos.



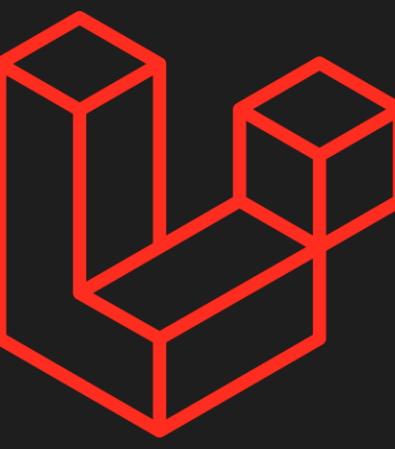
```
1  <!DOCTYPE html>
2  <html lang="pt">
3      <head>
4          <meta charset="utf-8">
5          <meta name="viewport" content="width=device-width, initial-scale=1">
6          <title>Produtos</title>
7      </head>
8      <body>
9          <h1>Produtos</h1>
10
11         <form action="/editar-produto/{{ $produto->id }}" method="POST">
12             @csrf
13             <label for="lblNome">Nome:</label>
14             <input type="text" name="nome" value="{{ $produto->nome }}">
15             <br><br>
16             <label for="lblValor">Valor:</label>
17             <input type="text" name="valor" value="{{ $produto->valor }}">
18             <br><br>
19             <label for="lblQuantidade">Quantidade:</label>
20             <input type="text" name="estoque" value="{{ $produto->estoque }}">
21             <br><br>
22             <button>Cadastrar</button>
23         </form>
24     </body>
25 </html>
26
```

INTRODUÇÃO AO LARAVEL

CRIANDO O DELETE DO PROJETO LARAVEL

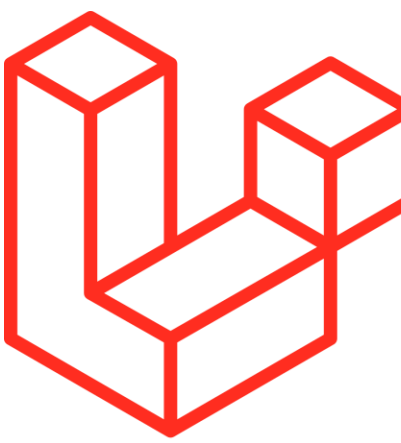


- A partir de agora vamos criar a rota para tela de exclusão dos dados armazenados em nosso banco.
- No diretório routes vamos editar o arquivo web.php



```
biblioteca_laravel > routes > web.php
28         'valor' => $request->valor,
29         'estoque' => $request->estoque
30     ]);
31
32     echo "Produto criado com sucesso!";
33 });
34
35 Route::get('/listar-produto/{id}', function($id){
36     //dd(Produto::find($id)); //debug and die
37     $produto = Produto::find($id);
38     return view('listar', ['produto' => $produto]);
39 });
40
41 Route::get('/editar-produto/{id}', function($id){
42     //dd(Produto::find($id)); //debug and die
43     $produto = Produto::find($id);
44     return view('editar', ['produto' => $produto]);
45 });
46
47 Route::post('/editar-produto/{id}', function(Request $request, $id){
48     //dd($request->all());
49     $produto = Produto::find($id);
50
51     $produto->update([
52         'nome' => $request->nome,
53         'valor' => $request->valor,
54         'estoque' => $request->estoque
55     ]);
56
57     echo "Produto editado com sucesso!";
58 });
59
60 Route::get('/excluir-produto/{id}', function($id){
61     //dd($request->all());
62     $produto = Produto::find($id);
63     $produto->delete();
64
65     echo "Produto excluido com sucesso!";
66 });
```

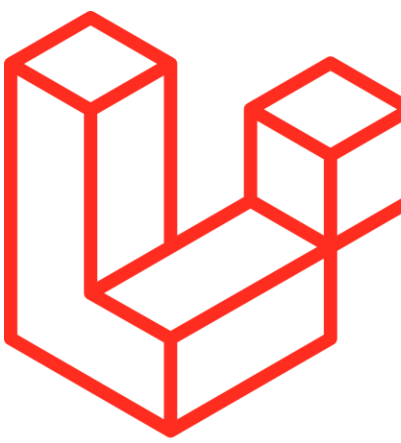
INTRODUÇÃO AO LARAVEL



TESTANDO A APLICAÇÃO NO BROWSER

- Com a finalização de todas as etapas anteriores podemos testar a aplicação através dos endereços de URLs:
- 127.0.0.1:8000/
- 127.0.0.1:8000/listar-produto/1
- 127.0.0.1:8000/editar-produto/1
- 127.0.0.1:8000/excluir-produto/1

INTRODUÇÃO AO LARAVEL



NOVA IMPLEMENTAÇÃO

- Através dos conceitos anteriores vamos criar uma tela principal para receber todos os links:
 - Cadastrar
 - Listar
 - Atualizar
 - Deletar
- Cada link deverá ser encaminhado para a rota corresponde ao nosso projeto Laravel, deixando a nossa aplicação com uma opção de navegação adequada.