06

DESENVOLVIMENTO DE APLICAÇÃO PARTE 1

Criando um CRUD em Laravel

03

```
  INFO  Running migrations.

  0001_01_01_000000_create_users_table
  0001_01_01_000001_create_cache_table
  0001_01_01_000002_create_jobs_table .


C:\>cd ProjLaravel

C:\ProjLaravel>code .
```

Criando um CRUD em Laravel

Depois de criar a pasta ProjLaravel e instalar o Laravel. Digite no CMD:

- cd LaravelEng

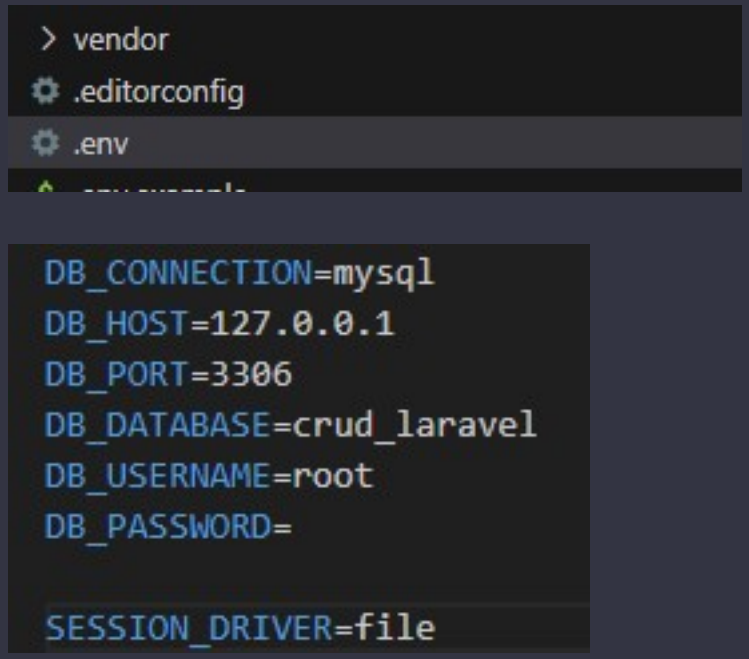
- code .

Para abrir o projeto no VSCode.

Criando um CRUD em Laravel

03

Criando o banco de dados e tabelas



1 - Localize o arquivo .env

2 - Abra o Arquivo .env e edite:

3 - DB_CONNECTION=mysql

4 - DB_DATABASE =bd_laravel

5 – SESSION_DRIVE=file

6 - Depois salve as alterações.

Criando um CRUD em Laravel

03

Criando o banco de dados e tabelas

E sem seguida, em seu navegador de preferência, acesse localhost/phpmyadmin, clique em Novo e digite o nome do seu novo banco de dados.

Atenção: Por padrão o Laravel utiliza o Agrupamento utf8_unicode_ci.

Criando um CRUD em Laravel

03

Criando o banco de dados e tabelas

Com o banco de dados devidamente configurado chegou a hora de criarmos as primeiras tabelas. O Laravel já possui algumas migrations (arquivos de instrução para criação de tabelas) nativas, para acessá-las basta encontrar, a partir da raiz do projeto, o diretório database/migrations.

```
0001_01_01_000000_create_users_table
0001_01_01_000001_create_cache_table
0001_01_01_000002_create_jobs_table .
```

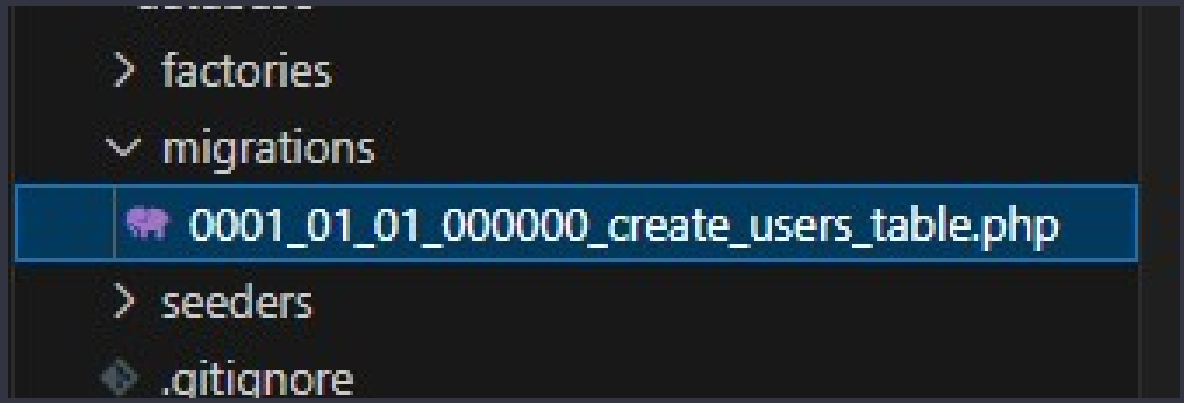
Criando um CRUD em Laravel

03

Criando o banco de dados e tabelas

Localize o arquivo:

0001_01_01_000000_create_users_table.php



Criando um CRUD em Laravel

03

```php
return new class extends Migration
{
    /**
     * Run the migrations.
     */

    protected $table = "usuarios";

    public function up(): void
    {
        Schema::create('usuarios', function (Blueprint $table) {
            $table->id();
            $table->string('nome');
            $table->string('email')->unique();
            $table->string('senha');
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('usuarios');
    }
};
```

co de dados e tabelas

1 – Edite o arquivo _create_users_table.php

2 - Na tabela usuarios, após criada, contará com os campos nome, email, e senha.

3 - No seu terminal, na raiz do projeto, digite o seguinte comando:

php artisan migrate

Criando um CRUD em Laravel

03

Criando o banco de dados e tabelas

#	Nome	Tipo
1	id	bigint(20)
2	nome	varchar(255)
3	email	varchar(255)
4	senha	varchar(255)

Perceba que uma tabela migrations foi criada. É um recurso nativo do Laravel que serve para registrar quais migrations já foram executadas para prevenir a tentativa de criação de uma tabela já existente, afinal de contas no dia a dia nós criaremos várias migrações que não serão necessariamente executadas ao mesmo tempo.

Criando um CRUD em Laravel

03

Criando uma migração

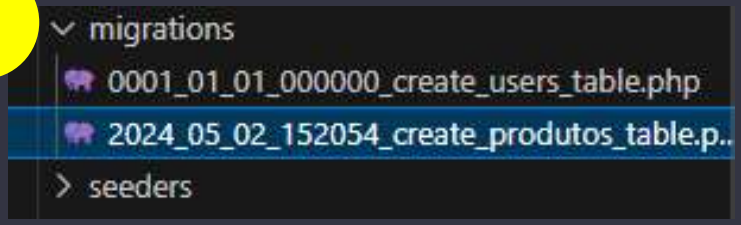
- No seu terminal, na pasta raiz do projeto, digite o seguinte comando:

php artisan make:model Produto –m

1

```
PS C:\ProjLaravel> php artisan make:model Produto -m

  INFO  Model [C:\ProjLaravel\app\Models\Produto.php] created successfully.

  INFO  Migration [C:\ProjLaravel\database\migrations/2024_05_02_152054_create_produtos_table.php] created successfully.
```

2



Criando um CRUD em Laravel

03

Criando uma migração

database > migrations > 2020_03_22_234841_create_produtos_table.php > ...

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateProdutosTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('produtos', function (Blueprint $table) {
            $table->id();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('produtos');
    }
}
```

Agora precisamos dizer para essa nova migração que criamos quais campos queremos na nossa tabela. Como você já deve ter percebido estamos criando uma tabela para armazenar dados sobre um determinado produto, então vamos nos preocupar em guardar o nome, custo, preço e quantidade.

Criando um CRUD em Laravel

03

Criando uma migração

```php
return new class extends Migration
{
    /**
     * Run the migrations.
     */
    protected $table = "produtos";
    public function up(): void
    {
        Schema::create('produtos', function (Blueprint $table) {
            $table->id();
            $table->string('nome');
            $table->decimal('custo', 19,2);
            $table->decimal('preco', 19,2);
            $table->integer('quantidade');
            $table->timestamps();
        });
    }
    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('produtos');
    }
};
```

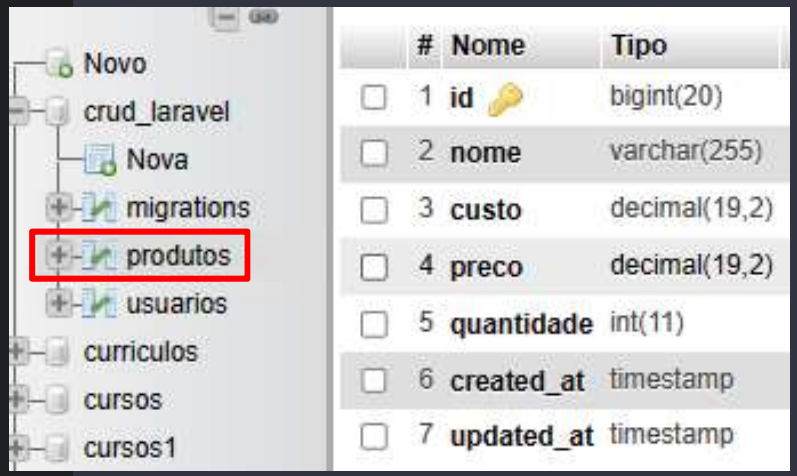
Como você já deve ter percebido estamos criando uma tabela para armazenar dados sobre um determinado produto, então vamos nos preocupar em guardar o nome, custo, preço e quantidade. Altere o seu arquivo de migração para que fique igual ao meu.

No seu terminal na pasta raiz do projeto digite novamente: **php artisan migrate**

Criando um CRUD em Laravel

03

Criando uma migração



Pronto, nossa tabela de produtos está criada.

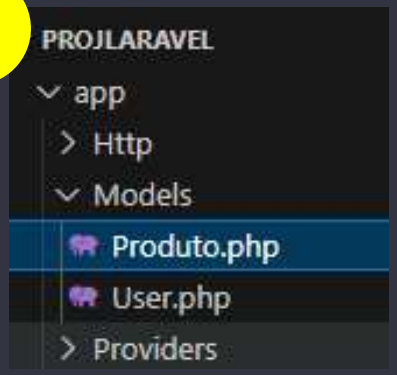
Por que o comando é make:model?
Model é um “template” de uma tabela existente dentro do framework, com a Model conseguimos Criar, Visualizar, Editar e Deletar com muita facilidade.

Criando um CRUD em Laravel

03

Configurando a Model

1



2

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Produto extends Model
{
    use HasFactory;
}
```

Na raiz do projeto encontre a pasta app, e em seguida o arquivo **Produto.php** e abra no seu editor de código preferido.

Agora vamos configurar algumas informações necessárias para começarmos a efetivamente desenvolver o tão esperado CRUD.

Criando um CRUD em Laravel

03

Configurando a Model

- A variável **fillable** literalmente informa a nossa model que esses campos que estão dentro do array serão preenchidos e/ou editados por nosso código em algum momento. Por enquanto é só isso que precisamos fazer na model!

2024_05_02_152054_create_produtos_table.php | Produto.php | .env

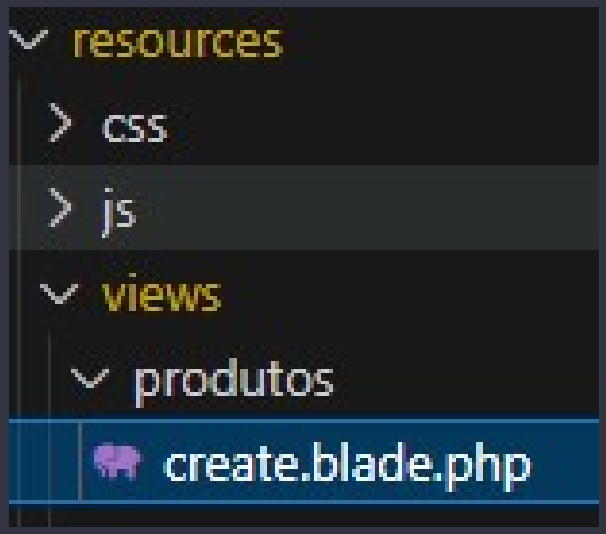
app > Models > Produto.php > ...

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Produto extends Model
{
    protected $fillable = ['nome', 'custo', 'preco', 'quantidade'];
}
```

Criando um CRUD em Laravel

03

Create (Criação de registro)



O Laravel por padrão armazena e reconhece seus arquivos de páginas html em resources/views, localize esse diretório:

Crie uma pasta chamada produtos, em seguida, dentro dessa nova pasta, crie um arquivo chamado create.blade.php.

Criando um CRUD em Laravel

03

Create (Criação de registro)

- Edite o seu novo arquivo, **create.blade.php**, para que fique desta forma:

resources > views > Produtos > create.blade.php > html > body > form > button

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Cadastrar Produtos</title>
7  </head>
8  <body>
9      <form action="">
10         <label for="">Nome</label><br>
11         <input type="text" name="nome"><br>
12         <label for="">Custo</label><br>
13         <input type="text" name="custo"><br>
14         <label for="">Preço</label><br>
15         <input type="text" name="preco"><br>
16         <label for="">Quantidade</label><br>
17         <input type="text" name="quantidade"><br>
18         <button>Salvar</button>
19     </form>
20 </body>
21 </html>
```

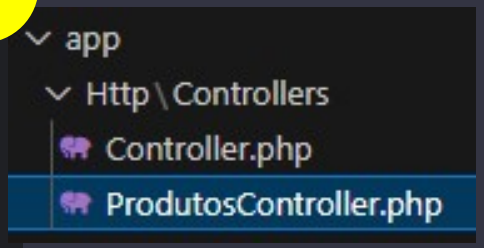
Blade facilita a administração de html, layouts e redirecionamentos

Criando um CRUD em Laravel

03

Create (Criação de registro)

1



2

```
p > Controllers > ProdutosController.php > ...
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class ProdutosController extends Controller
{
    //
}
```

Bom, temos o formulário criado, mas como nós podemos acessá-lo?

Em seu terminal, na pasta raiz do projeto, digite o seguinte comando:

php artisan make:controller ProdutosController

No diretório app\Http\Controllers, localize o arquivo ProdutosController.php

Criando um CRUD em Laravel

03

Create (Criação de registro)

app > Http > Controllers > ProdutosController.php > Produto

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class ProdutosController extends Controller
{
    public function create()
    {
        return view('produtos.create');
    }
}

```

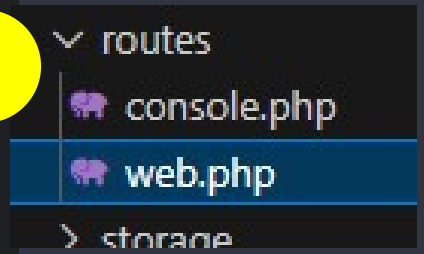
Crie uma função chamada create, e nela retorne a função view, essa função do Laravel percorre o diretório resources/views e tenta encontrar a o arquivo que você especificou como parâmetro e o carrega no navegador.

Criando um CRUD em Laravel

03

Create (Criação de registro)

1



2

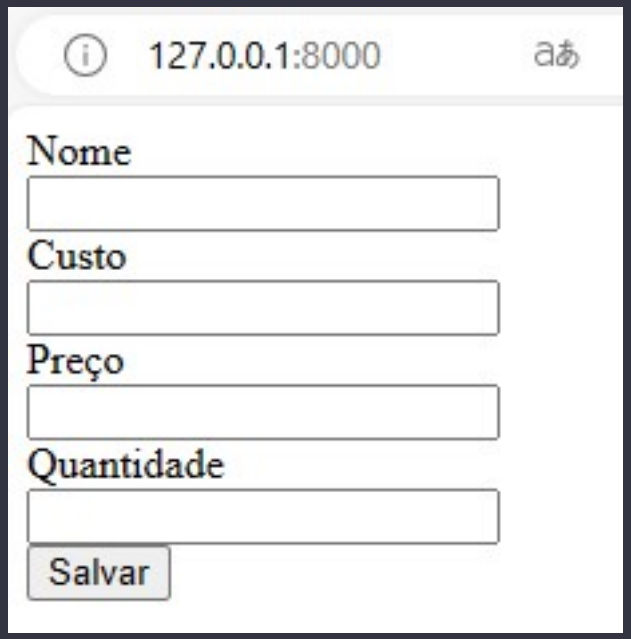
```
routes > web.php > ...
1   <?php
2
3   use Illuminate\Support\Facades\Route;
4
5   Route::get('/', function () {
6       return view ('/produtos/create');
7   });
8
```

Chegou a hora de verificarmos se a nossa função está funcionando corretamente e se a nossa página com o formulário será carregada. Na pasta raiz do seu projeto localize o diretório routes e em seguida o arquivo web.php, e abra no seu editor de código.

Criando um CRUD em Laravel

03

Create (Criação de registro)



Chegou a hora de testar, vamos ligar o servidor do Laravel e acessar pelo navegador para ver nosso formulário carregando. Para isso vá no seu terminal, na pasta raiz do projeto, e digite o seguinte comando:

php artisan serve

Criando um CRUD em Laravel

03

Create (Criação de registro)

p > Http > Controllers > ProdutosController.php > Produto

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class ProdutosController extends Controller
{
    public function create()
    {
        return view('produtos.create');
    }

    public function store(Request $request)
    {
        dd($request->all());
    }
}
```

Agora chegou a hora de fazermos com que o seu envio crie um novo produto na nossa tabela do banco de dados.

Vamos continuar os nosso trabalhos! No ProdutosController, crie uma função chamada store.

Criando um CRUD em Laravel

03

Create (Criação de registro)

No arquivo de rota web.php cria uma nova rota, dessa vez utilizando o verbo post e a função store do nosso controller!

web.php > ...

```php
<?php

use Illuminate\Support\Facades\Route;

Route::get('/', function () {
    return view ('/produtos/create');

});

Route::get('/produtos/create', 'App\Http\Controllers\ProdutosController@create');
Route::post('/produtos/create', 'App\Http\Controllers\ProdutosController@store')->name('registrar_produto');
```

Criando um CRUD em Laravel

03

Create (Criação de registro)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scal
    <title>Cadastrar Produtos</title>
</head>
<body>
    <form action="{{ route('registrar_produto')}}" method="POST">
    @csrf
        <label for="">Nome</label><br>
        <input type="text" name="nome"><br>
        <label for="">Custo</label><br>
        <input type="text" name="custo"><br>
        <label for="">Preço</label><br>
        <input type="text" name="preco"><br>
        <label for="">Quantidade</label><br>
        <input type="text" name="quantidade"><br>
        <button>Salvar</button>
    </form>
</body>
</html>
```

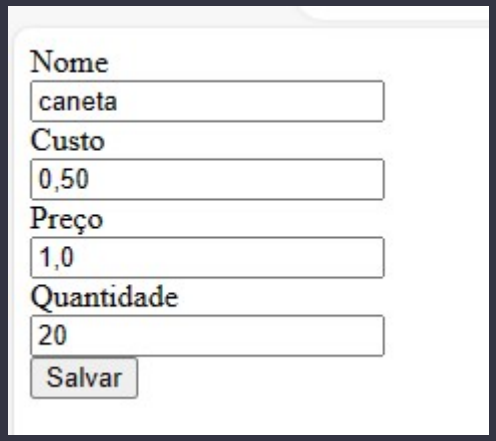
Mais uma vantagem em se utilizar a extensão blade.php juntamente com o Laravel é de poder usar funções nativas do framework. Basta utilizarmos os mustaches, que são essas chaves duplas, que o arquivo blade entenderá que se trata de um código PHP. Nesse caso estamos chamando a função route() e dizendo qual é o nome da rota que queremos utilizar.

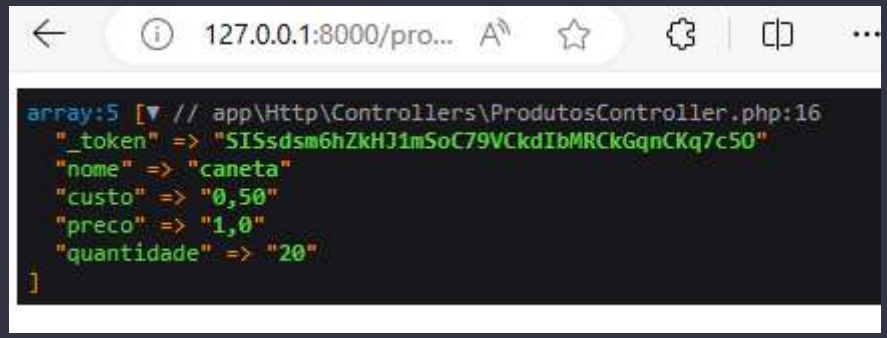
Criando um CRUD em Laravel

03

Create (Criação de registro)

- Vejam que após clicar em Salvar nossa rota foi chamada e caímos de fato na função store do ProdutosController. Perceba que além das informações de nome, custo, preço e quantidade também temos o token de validação!





Criando um CRUD em Laravel

03

Http > Controllers > ProdutosController.php > ...

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Produto;

class ProdutosController extends Controller
{
    public function create()
    {
        return view('produtos.create');
    }

    public function store(Request $request)
    {
     Produto::create([
            'nome' => $request->nome,
            'custo' => $request->custo,
            'preco' => $request->preco,
            'quantidade' => $request->quantidade
        ]);

        return "Produto Criado com Sucesso!";
    }
}
```

Create (Criação de registro)

Excelente, temos nossas informações chegando corretamente na nossa função, agora vamos de fato salvar no banco de dados!

Vá na sua função store e deixe ele como a minha.

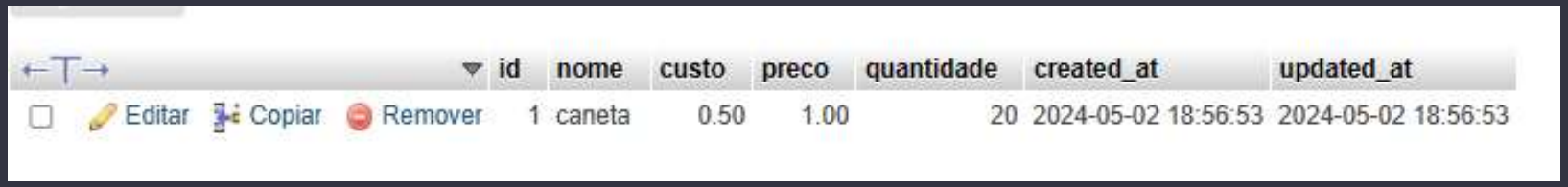
Não se esqueça de chamar a classe use App\Models\Produto, como eu fiz na linha 6!

Criando um CRUD em Laravel

03

Create (Criação de registro)

Vamos verificar lá no banco de dados se o produto foi realmente criado! Acesse localhost/phpmyadmin, entre no banco que você criou e veja e na tabela produtos está presente o registro que acabamos de criar.



		id	nome	custo	preco	quantidade	created_at	updated_at
☐	Editar Copiar Remover	1	caneta	0.50	1.00	20	2024-05-02 18:56:53	2024-05-02 18:56:53

Criando um CRUD em Laravel

03

Read (Leitura de um Registro)

No seu arquivo de rotas web.php crie uma nova rota do tipo get, dessa vez com um parâmetro, o id. Esse id será passado diretamente na rota, para que possamos realizar a busca no banco de dados e encontrar o registro.

```
});

Route::get('/produtos/create', 'App\Http\Controllers\ProdutosController@create');
Route::post('/produtos/create', 'App\Http\Controllers\ProdutosController@store')->name('registrar_produto');
Route::get('/produtos/ver/{id}', 'App\Http\Controllers\ProdutosController@show');
```

Criando um CRUD em Laravel

03

Read (Leitura de um Registro)

- Agora precisamos criar em nosso ProdutosController a função show(), que receberá o id informado na rota e fará a busca no banco de dados. Após obter o registro, a função retornará uma view com um formulário preenchido com as informações do produto. A sua função show() deve ficar igual a minha.

```
        return "Produto Criado com Sucesso!";
    }

    public function show($id)
    {
        $produto = Produto::findorFail($id);
        return view('produtos.show', ['produto'=>$produto]);
    }
}
```

Criando um CRUD em Laravel

03

Read (Leitura de um Registro)

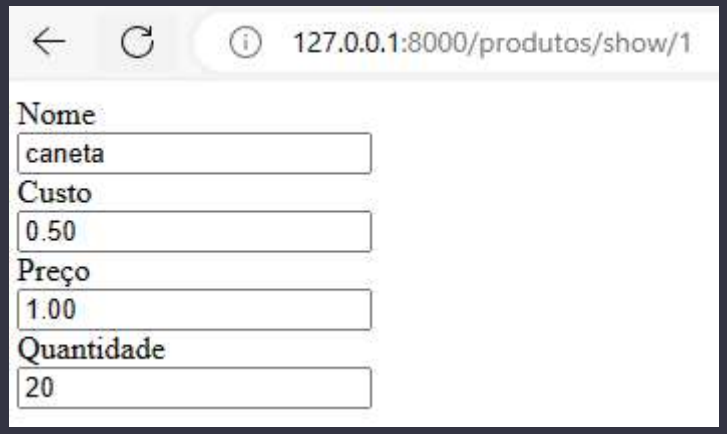
- Agora chegou a hora de criarmos de fato a view, em resources/views/produtos crie um arquivo chamado **show.blade.php** e o deixe igual ao meu:

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ver Produtos</title>
</head>
<body>
        <label for="">Nome</label><br>
        <input type="text" name="nome" value='{{$produto->nome}}'><br>
        <label for="">Custo</label><br>
        <input type="text" name="custo" value="{{$produto->custo}}"><br>
        <label for="">Preço</label><br>
        <input type="text" name="preco" value="{{$produto->preco}}"><br>
        <label for="">Quantidade</label><br>
        <input type="text" name="quantidade" value="{{$produto->quantidade}}"><br>
</body>
</html>
```

Criando um CRUD em Laravel

03

Read (Leitura de um Registro)



E acesse, em seu navegador, localhost:8000/produto/show/1 (o número 1 é o id do produto, você pode criar vários produtos e testar com todos os ids).