



Marcos Rodrigo Momo, M.Sc.  
e-mail: marcos.momo@ifsc.edu.br

## Programação para Internet II



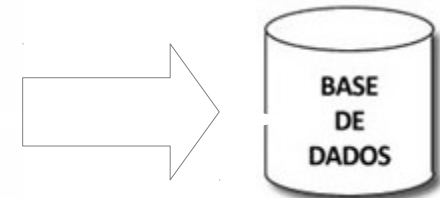
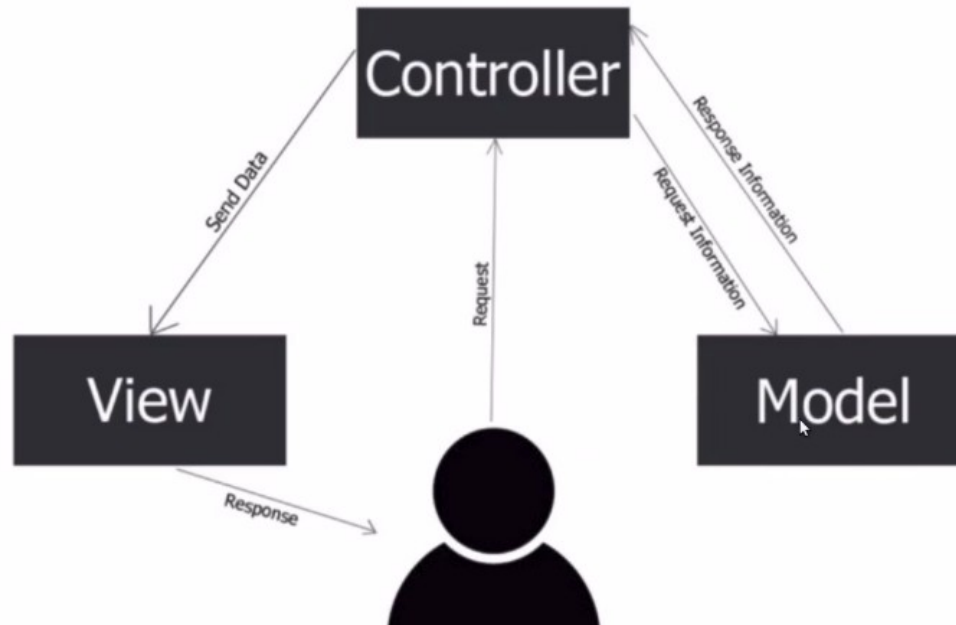
**INSTITUTO  
FEDERAL**  
Santa Catarina

Câmpus  
Gaspar

Gaspar, maio 2021.

# Aula anterior

## Model-View-Controller



Criar tabela  
Criar campo  
Fazer relacionamentos  
Fazer consultas

# Roteiro aula

## Projeto de Cadastro

- Rotas
- Views/Formulários
- Construtores
- Modelos
- Layout/Componentes/Estilização
- Persistência no banco

# Roteiro Estudo de Caso

Home Produtos Categorias

## Cadastro de produtos

Aqui você cadastra todos os seus produtos. Só não se esqueça de cadastrar previamente as categorias

Cadastre seus produtos

## Cadastro de Categorias

Cadastre as categorias dos seus produtos

Cadastre suas categorias

# Criar o projeto e preparar o ambiente

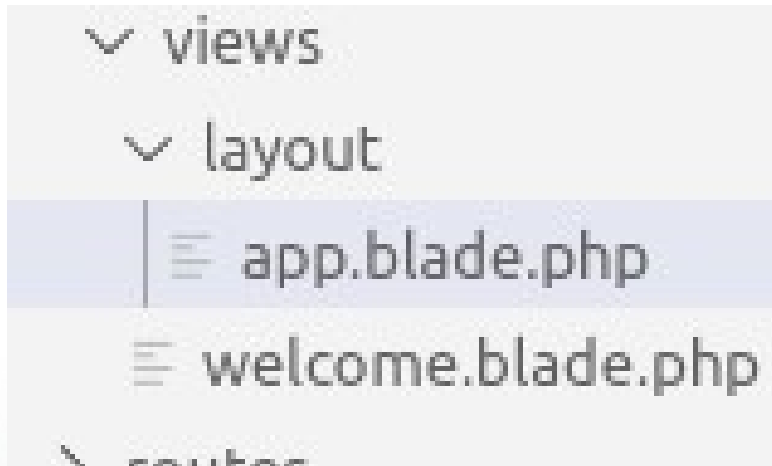
- Via terminal

laravel new cadastro

npm install (baixar dependências)

# Criar um layout para o projeto

- Criar uma pasta dentro de View
- Criar um arquivo .blade



# App.blade.php

```
<html>
  <head>
    <link href="{{ asset('css/app.css') }}" rel="stylesheet">
    <title>Cadastro de Produtos</title>
    <meta name="csrf-token" content="{{ csrf_token() }}">

  </head>
  <body>
    <div class="container">
      <main role="main">
        @hasSection('body')
          @yield('body')
        @endif
      </main>
    </div>

    <script src="{{ asset('js/app.js') }}" type="text/javascript"></script>
  </body>
</html>
```

# Criar o index

- Na pasta view
- Criar um arquivo index.blade.php

```
▼ views
  ▼ layout
    ≡ app.blade.php
    ≡ index.blade.php
    ≡ welcome.blade.php
```



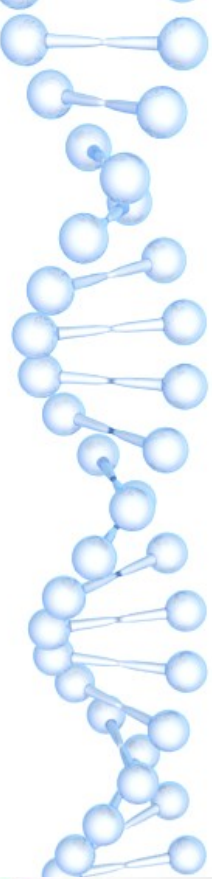
# Testar o index

```
@extends('layout.app', ["current" => "home"])
```

```
@section('body')
```

```
<h1>Testando o index....</h1>
```

```
@endsection
```

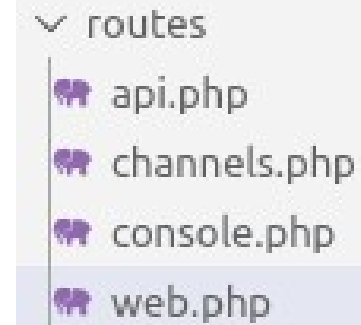


# Testar o index

- Mudar a rota em routes>web.php

```
Route::get('/', function () {  
    return view('index');  
});
```

Executar o projeto: php artisan serve  
Acessar localhost:8000



```
▼ routes  
api.php  
channels.php  
console.php  
web.php
```

# Implementando o index

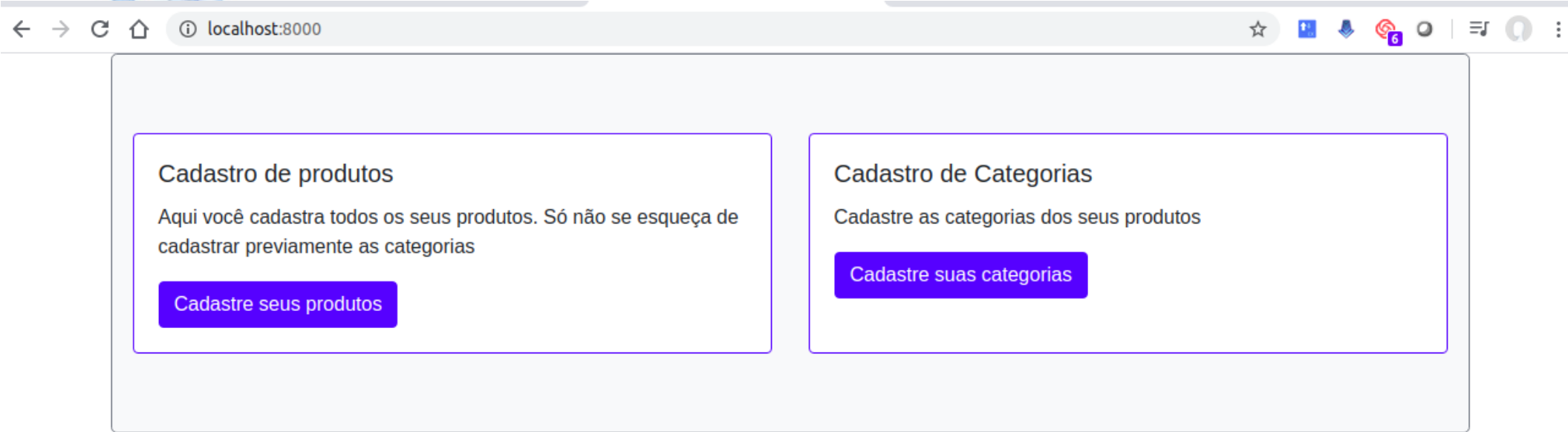
Fazer o teste

```
• @extends('layout.app')
•
• @section('body')
•
• <div class="jumbotron bg-light border border-secondary">
•   <div class="row">
•     <div class="card-deck">
•       <div class="card border border-primary">
•         <div class="card-body">
•           <h5 class="card-title">Cadastro de produtos</h5>
•           <p class="card-text">
•             Aqui você cadastra todos os seus produtos.
•             Só não se esqueça de cadastrar previamente as categorias
•           </p>
•           <a href="/produtos" class="btn btn-primary">Cadastre seus produtos</a>
•         </div>
•       </div>
•       <div class="card border border-primary">
•         <div class="card-body">
•           <h5 class="card-title">Cadastro de Categorias</h5>
•           <p class="card-text">
•             Cadastre as categorias dos seus produtos
•           </p>
•           <a href="/categorias" class="btn btn-primary">Cadastre suas categorias</a>
•         </div>
•       </div>
•     </div>
•   </div>
• </div>
• @endsection
```

# Configurar o arquivo css

- Verifique na pasta public/css/app.css
- Caso não tenha criado, substituir o arquivo do SIGAA

# Teste



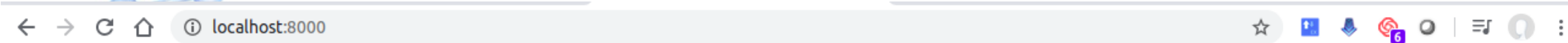
# Configurando o padding no arquivo app.css

- No arquivo app.css

```
body {  
    padding: 20px;  
}  
  
.navbar {  
    margin-bottom: 20px;  
}
```

Fazer o teste

# Teste



## Cadastro de produtos

Aqui você cadastra todos os seus produtos. Só não se esqueça de cadastrar previamente as categorias

Cadastre seus produtos

## Cadastro de Categorias

Cadastre as categorias dos seus produtos

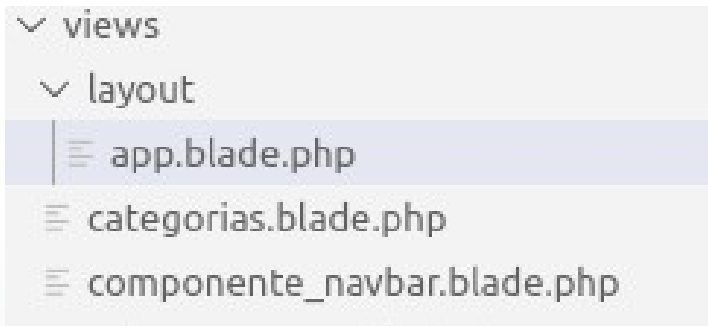
Cadastre suas categorias



# Implementar o navbar utilizando component

Home Produtos Categorias

- Vamos criar um componente dentro da pasta view
  - Componente.blade.php





# Implementado o navbar

<https://getbootstrap.com/docs/4.4/components/navbar/>

```
<nav class="navbar navbar-expand-lg navbar-dark bg-dark rounded">
  <button class="navbar-toggler" type="button" data-toggle="collapse"
    data-target="#navbar" aria-controls="navbar" aria-expanded="false" aria-
    label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbar">
    <ul class="navbar-nav mr-auto">
      <li @if($current=="home") class="nav-item active" @else class="nav-item" @endif>
        <a class="nav-link" href="/">Home </a>
      </li>
      <li @if($current=="produtos") class="nav-item active" @else class="nav-item"
@endif>
        <a class="nav-link" href="/produtos">Produtos</a>
      </li>
      <li @if($current=="categorias") class="nav-item active" @else class="nav-item"
@endif>
        <a class="nav-link" href="/categorias">Categorias </a>
      </li>
    </ul>
  </div>
</nav>
```

# Inserir o componente navbar no app.blade.php

@component('componentes.com  
ponent\_navbar', [ "current" =>  
\$current ?? " " ])

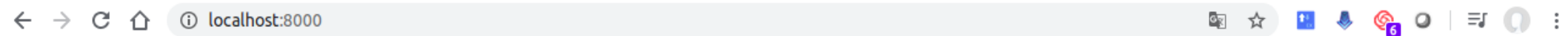
@endcomponent

Fazer o teste

```
<html>
  <head>
    <link href="{{ asset('css/app.css') }}" rel="stylesheet">
    <title>Cadastro de Produtos</title>
    <meta name="csrf-token" content="{{ csrf_token() }}">
  </head>
  <body>
    <div class="container">
      @component('componentes.component_navbar', [ "current" => $current ?? '' ])
      @endcomponent
      <main role="main">
        @hasSection('body')
          @yield('body')
        @endif
      </main>
    </div>

    <script src="{{ asset('js/app.js') }}" type="text/javascript"></script>
  </body>
</html>
```

# Teste



[Home](#) [Produtos](#) [Categorias](#)

## Cadastro de produtos

Aqui você cadastra todos os seus produtos. Só não se esqueça de cadastrar previamente as categorias

[Cadastre seus produtos](#)

## Cadastro de Categorias

Cadastre as categorias dos seus produtos

[Cadastre suas categorias](#)

# Criar base de dados

- Arquivo .env

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=vendas
DB_USERNAME=user
DB_PASSWORD=user
```



# Criar os modelos *migrations* e configurar as bases de dados

- Criar duas tabelas de categoria e produto

```
php artisan make:model Categoria -m
```

```
php artisan make:model Produto -m
```

- Esse comando cria o modelo e o *migration*

# Criar campos nas tabelas

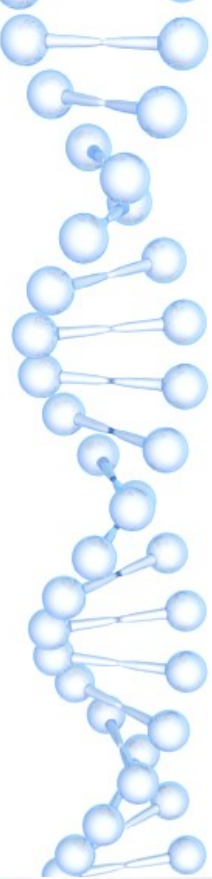
- Fazer a edição das classes para novos atributos

```
public function up()
{
    Schema::create('categorias', function (Blueprint $table) {
        $table->increments('id');
        $table->string('nome');
        $table->timestamps();
    });
}

public function up()
{
    Schema::create('produtos', function (Blueprint $table) {
        $table->increments('id');
        $table->string('nome');
        $table->integer('estoque');
        $table->float('preco');
        $table->integer('categoria_id')->unsigned();
        $table->foreign('categoria_id')->references('id')->on('categorias');
        $table->timestamps();
    });
}
```

# Executar o *migrate*

- Para atualizar tabelas, executar o migrate  
php artisan migrate



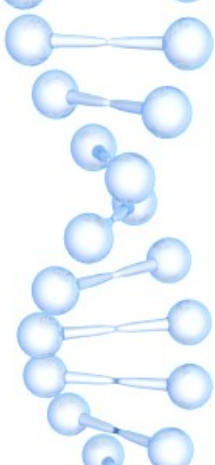
# Criando controladores

- Devemos agora criar as rotas

```
php artisan make:controller ControladorProdutos - -resource
```

```
php artisan make:controller ControladorCategorias - -resource
```
- Verificar os arquivos criados em `app/http/controllers`





# Configurar as rotas no arquivo web.php

- Todas as requisições que chegaram tanto para produto como para categoria, devem ser redirecionadas

```
use Illuminate\Support\Facades\Route;  
use App\Http\Controllers\ControladorCategorias;  
use App\Http\Controllers\ControladorProdutos;
```

```
Route::resource('/produtos', ControladorProdutos::class);  
Route::resource('/categorias', ControladorCategorias::class);
```

# No arquivo de controlador de categoria

- Configurar o index

```
public function index()  
{  
    return view('categorias');  
}
```

# No arquivo de controlador de produto

- Configurar o index

```
public function index()  
{  
    return view('produtos');  
}
```

# Criar as views produtos e categorias

- Em resource/view
  - produtos.blade.php
  - categorias.blade.php

Fazer o teste

```
@extends('layout.app')
```

```
@section('body')
```

```
<h4>Página de Categorias...</h4>
```

```
@endsection
```

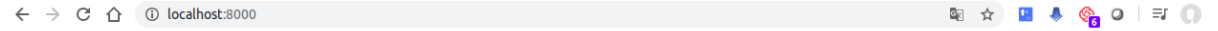
```
@extends('layout.app')
```

```
@section('body')
```

```
<h4>Página de Produtos...</h4>
```

```
@endsection
```

# Teste



Home Produtos Categorias

## Cadastro de produtos

Aqui você cadastra todos os seus produtos. Só não se esqueça de cadastrar previamente as categorias

Cadastre seus produtos

## Cadastro de Categorias

Cadastre as categorias dos seus produtos

Cadastre suas categorias

Home Produtos Categorias

Página de Categorias...

Home Produtos Categorias

Página de Produtos...

# Configurando 'active' do do NavBar

- Utilizar a variável current, nas páginas html de produto e de categoria

```
@extends('layout.app', ["current" => "categorias" ])
```

```
@section('body')  
    <h4>Página de categorias...</h4>  
@endsection
```

```
@extends('layout.app', ["current" => "produtos" ])
```

```
@section('body')  
    <h4>Página de produtos...</h4>  
@endsection
```



# Configurando 'active' do do NavBar

- Utilizar a variável current, nas páginas html de NavBar

```
<nav class="navbar navbar-expand-lg navbar-dark bg-dark rounded">
  <button class="navbar-toggler" type="button" data-toggle="collapse"
    | data-target="#navbar" aria-controls="navbar" aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbar">
    <ul class="navbar-nav mr-auto">
      <li @if($current=="home") class="nav-item active" @else class="nav-item" @endif>
        <a class="nav-link" href="/">Home </a>
      </li>
      <li @if($current=="produto") class="nav-item active" @else class="nav-item" @endif>
        <a class="nav-link" href="/produto">Produtos</a>
      </li>
      <li @if($current=="categoria") class="nav-item active" @else class="nav-item" @endif>
        <a class="nav-link" href="/categoria">Categorias </a>
      </li>
    </ul>
  </div>
</nav>
```

# Configurando 'active' do do NavBar

- Utilizar a variável current, nas páginas html de Index

```
@extends('layout.app', ["current" => "home"])
```



# Configurando 'active' do do NavBar

- Utilizar a variável current, nas páginas html em app

```
<body>  
  <div class="container">  
    @component('componente_navbar', [ "current" => $current ] )  
  @endcomponent
```

Fazer o teste

# Teste do active

[Home](#) [Produtos](#) [Categorias](#)

## Cadastro de produtos

Aqui você cadastra todos os seus produtos. Só não se esqueça de cadastrar previamente as categorias

[Cadastre seus produtos](#)

## Cadastro de Categorias

Cadastre as categorias dos seus produtos

[Cadastre suas categorias](#)

[Home](#) [Produtos](#) [Categorias](#)

Página de produtos...

# Criando formulário de cadastro de categorias

- Criando formulário
  - Vamos acrescentar as rotas no web.php

```
Route::resource('/produtos', ControladorProdutos::class);  
Route::resource('/categorias', ControladorCategorias::class);
```

```
GET|HEAD | categorias/create
```

```
categorias.create
```

```
App\Http\Controllers\ControladorCategorias@create | web
```

- Implementar o método create da classe controller, que vai retornar uma view (form)

```
public function create()  
{  
    return view('novacategoria');  
}
```

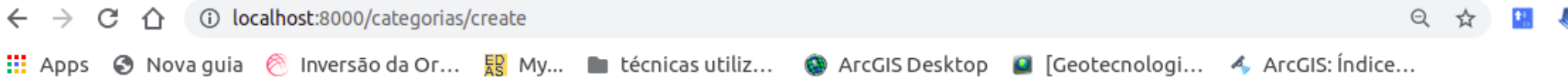


# Criar a view do método create de controller categoria

- Na pasta view, criar um novo arquivo, novacategoria.blade.php

```
@extends('layout.app' ,["current" => "categoria"])  
  
@section('body')  
    <h4>Formulários para nova categoria...</h4>  
@endsection
```

Fazer o teste



Home Produtos Categorias

Formulário para nova categoria...


# Criar a view do método create de controller categoria (completa)

```
@extends('layout.app', ["current" => "categorias"])

@section('body')

<div class="card border">
  <div class="card-body">
    <form action="/categorias" method="POST">
      @csrf
      <div class="form-group">
        <label for="nomeCategoria">Nome da Categoria</label>
        <input type="text" class="form-control" name="nomeCategoria"
          id="nomeCategoria" placeholder="Categoria">
      </div>
      <button type="submit" class="btn btn-primary btn-sm">Salvar</button>
      <button type="cancel" class="btn btn-danger btn-sm">Cancelar</button>
    </form>
  </div>
</div>
@endsection
```

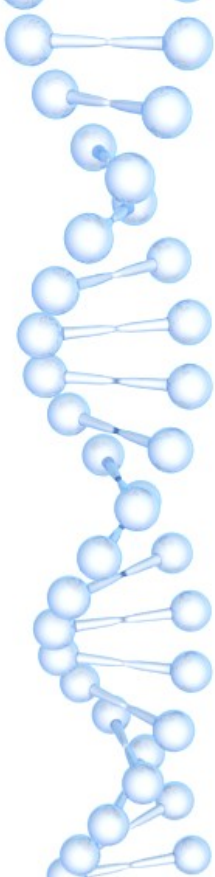
# Configurar o botão no index para nova categoria



```
cadastre as categorias dos seus produtos
</p>
<a href="/categorias/create" class="btn btn-primary">Cadastre suas categorias</a>
</div>
</div>
</div>
```

# Persistência....

- Próxima aula sábado 29/05 assíncrona ....



# Configurar o método store para persistir no BD os dados do formulário

- Configurar action no formulário de cadastro de categorias

```
@extends('layout.app', ["current" => "categorias"])

@section('body')

<div class="card border">
  <div class="card-body">
    <form action="{{ route('categorias.store') }}" method="POST">
      @csrf
      <div class="form-group">
        <label for="nomeCategoria">Nome da Categoria</label>
        <input type="text" class="form-control" name="nomeCategoria"
              id="nomeCategoria" placeholder="Categoria">
      </div>
        <button type="submit" class="btn btn-primary btn-sm">Salvar</button>
        <button type="cancel" class="btn btn-danger btn-sm">Cancelar</button>
      </form>
    </div>
  </div>
</section>
```



# Configurar o método store para persistir no BD os dados do formulário

- Configurar o método store do controller categoria
- Importar categoria pelo use

```
use Illuminate\Http\Request;  
use App\Models\Categoria;
```

```
public function store(Request $request)  
{  
    $cat = new Categoria();  
    $cat->nome = $request->input('nomeCategoria');  
    $cat->save();  
    return redirect('/categorias');  
}
```

Fazer o teste  
Ver BD

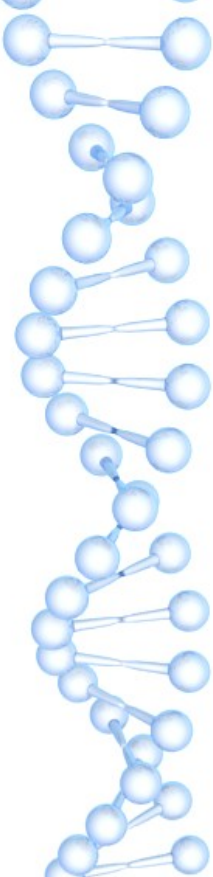
# Listar categorias

- No index de controller categoria, criar um variável do tipo array para armazenar todas as categorias

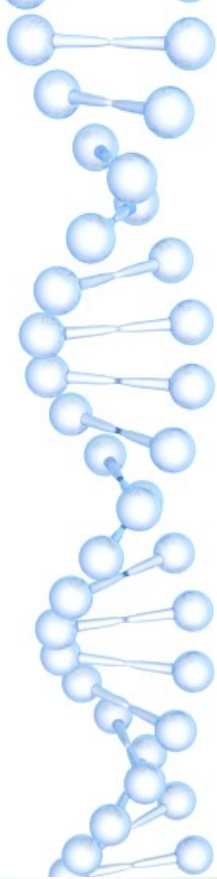
```
public function index()
{
    $cats = Categoria::all();
    return view('categorias', compact('cats'));
}
```

# Configura a view categorias

- Vamos utilizar componentes do bootstrap para listar todas as categorias



# view categorias



```
@extends('layout.app', ["current" => "categorias"])
```

```
@section('body')
```

```
<div class="card border">
```

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

```
    <h5 class="card-title">Cadastro de Categorias</h5>
```

```
@if(count($cats) > 0)
```

```
  <table class="table table-ordered table-hover">
```

```
    <thead>
```

```
      <tr>
```

```
        <th>Código</th>
```

```
        <th>Nome da Categoria</th>
```

```
        <th>Ações</th>
```

```
      </tr>
```

```
    </thead>
```

```
    <tbody>
```

```
      @foreach($cats as $cat)
```

```
        <tr>
```

```
          <td>{{ $cat->id }}</td>
```

```
          <td>{{ $cat->nome }}</td>
```

```
          <td>
```

```
            <a href="#" class="btn btn-sm btn-primary">Editar</a>
```

```
            <a href="#" class="btn btn-sm btn-danger">Apagar</a>
```

```
          </td>
```

```
        </tr>
```

```
      @endforeach
```

```
    </tbody>
```

```
  </table>
```

```
@endif
```

```
@endsection
```

# Redirect

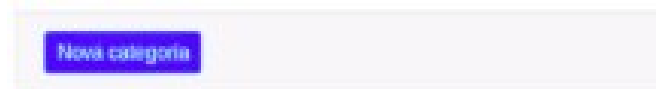
- Ao salvar uma categoria, deve fazer um redirecionamento para a view categorias
  - Na função store do controller categoria, configurar o redirect

```
public function store(Request $request)
{
    $cat = new Categoria();
    $cat->nome = $request->input('nomeCategoria');
    $cat->save();
    return redirect('/categorias');
}
```
  - Ao persistir um elemento, ele redireciona para categoria

# Criando botão para chamar o formulário

- Na view categoria, criar um botão para criar nova categoria

```
<div class="card-footer">  
  <a href="/categorias/novo" class="btn btn-sm btn-primary" role="button">Nova categoria</a>  
</div>
```



# Apagar categoria

- Configurar a rota para apagar uma categoria

egoria}	categorias.update	App\Http\Controllers\ControladorCategorias@update
egoria}	categorias.destroy	App\Http\Controllers\ControladorCategorias@destro
egoria/edit	categorias edit	App\Http\Controllers\ControladorCategorias@edit

- Vamos chamar a função destroy do controller categoria, passando um id como parâmetro para apagar uma categoria

```
public function destroy($id)
{
    $cat = Categoria::find($id);
    if (isset($cat)) {
        $cat->delete();
    }
    return redirect('/categorias');
}
```

Fazer o teste

# Configurar a rota no botão apagar do formulário categorias

```
<form action="{{ route('categorias.destroy', $cat['id']) }}" method="POST">
    @csrf
    @method('DELETE')
    <input type="submit" class="btn btn-sm btn-danger"
value="Apagar">
</form>
```

```

        @foreach($cats as $cat)
            <tr>
                <td>{{ $cat->id }}</td>
                <td>{{ $cat->nome }}</td>
                <td>
                    <a href="#" class="btn btn-sm btn-primary">Editar</a>
                    <td>
                        <form action="{{ route('categorias.destroy', $cat['id']) }}" method="POST">
                            @csrf
                            @method('DELETE')
                            <input type="submit" class="btn btn-sm btn-danger" value="Apagar">
                        </form>
                    </td>
                </tr>
            </td>
        @endforeach
```



# Editar categoria

- Implementar a rota para editar uma categoria

categorias/{categoria}/edit	categorias.edit	App\Http\Controllers\ControladorCategorias@edit	web
produtos	produtos.index	App\Http\Controllers\ControladorProdutos@index	web

- Vamos chamar a função edit do controller categoria, passando um id como parâmetro para editar uma categoria

```
foreach($cats as $cat)
    <tr>
        <td>{{ $cat->id }}</td>
        <td>{{ $cat->nome }}</td>
        <td>
            <a href="{{ route('categorias.edit', $cat['id']) }}" class="btn btn-sm btn-primary">Editar</a>
        <td>
            <form action="{{ route('categorias.destroy', $cat['id']) }}" method="POST">
                @csrf
                @method('DELETE')
                <input type="submit" class="btn btn-sm btn-danger" value="Apagar">
            </form>
        <td>
    </tr>
endforeach
```

# Editar categoria

```
public function edit($id)
{
    $cat = Categoria::find($id);
    if(isset($cat)) {
        return view('editarcategoria', compact('cat'));
    }
    return redirect('/categorias');
}
```

- Vamos criar a view editcategoria
  - Na pasta resource/view

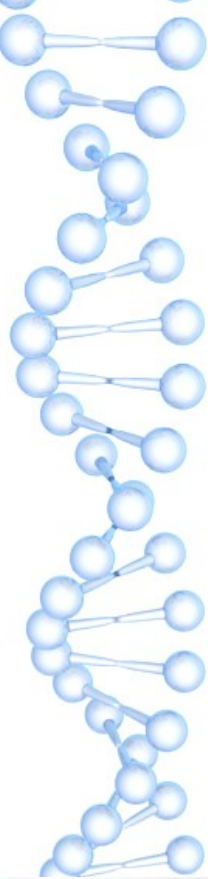
# Criar view editcategoria

```
@extends('layout.app', ["current" => "categorias"])

@section('body')

<div class="card border">
  <div class="card-body">
    <!--Configurar o actions para chamar o método update-->
    <form action="{{ route('categorias.update', $cat['id']) }}" method="POST">
      @csrf
      @method('PUT')
      <div class="form-group">
        <label for="nomeCategoria">Nome da Categoria</label>
        <input type="text" class="form-control" name="nomeCategoria"
          id="nomeCategoria" placeholder="Categoria">
      </div>
      <button type="submit" class="btn btn-primary btn-sm">Salvar</button>
      <button type="cancel" class="btn btn-danger btn-sm">Cancel</button>
    </form>
  </div>
</div>

@endsection
```



# Criar rota para atualizar categoria

- Criar a rota para chamar a função update do controller categoria

PUT|PATCH | categorias/{categoria} | categorias.update | App\Http\Controllers\ControladorCategorias@update | web

- Implementar a função update do controller categoria

```
public function update(Request $request, $id)
{
    $cat = Categoria::find($id);
    if(isset($cat)) {
        $cat->nome = $request->input('nomeCategoria');
        $cat->save();
    }
    return redirect('/categorias');
}
```

Fazer o teste

# Concluindo

- Testar as quatro operação de cadastro de categoria
  - Listar, inserir, atualizar e apagar
- Repetir a mesma operação para produto
  - Um produto pertence a uma categoria