**Podemos inserir e retornar variáveis/chaves as definições pré-determinadas fixas nas rotas.**

*Route*::get('/', *function* () {

    $nome = "Wesley";

    $idade = 19;

    return view('welcome', ['nome' => $nome, 'idade' => $idade, 'profissao' => "Programador"]);

});

**Modo de utilizar na página. Se usa a chave entre {{}} declarada na rota com ‘’.**

<p>

            {{$nome}}

        </p>

**Blade é a template engine do Laravel.**

**Condicionais / Diretivas do Blade.**

@if($nome == "Pedro")

        <p>O nome é Pedro</p>

        @elseif($nome == "Wesley")

        <p>O nome é {{$nome}} e ele tem {{ $idade }} anos e sua profissão é {{ $profissao }}.</p>

        @else

        <p>O nome não é Pedro</p>

        @endif

**Laço for com array declarado na rota.**

$arr = [10, 20, 30, 40, 50];

'arr' => $arr]);

**Irá mostrar, os valores e índice do array.**

        @for($i = 0; $i < count($arr); $i++)

            <p>{{$arr[$i]}} - {{$i}}</p>

        @endfor

**Marcação de código php no HTML com Blade.**

@php

            $name = "Wesley";

            echo $name;

        @endphp

**Não usado na aula, porém funciona.**

<?php

            $name = "Wesley";

            echo $name;

        ?>

**Comentários no Blade não é renderizado na inspeção.**

<!--Comentário com HTML é renderizado na inspeção-->

{{--Comentário do Blade não é renderizado na inspeção--}}

**Foreach e suas maneiras de renderizar os índices e valores na tela.**

 @foreach($nomes as $indice => $nome)

            <p>{{$indice}} - {{$nome}}</p>

            <p>{{$loop->index}}</p>

        @endforeach

**Antes, lógica do projeto.**

*Route*::get('/produtos', *function* () {

    $busca = request('search');

    return view('products', ['busca' => $busca]);

});

**Depois, passando pro controller.**

*Route*::get('/produtos', [*EventController*::class, 'produtos']);

**Php artisan migrate – criar tabelas no banco de dados.**

**php artisan migrate**

**Migration table created successfully.**

**Migrating: 2014\_10\_12\_000000\_create\_users\_table**

**Migrated: 2014\_10\_12\_000000\_create\_users\_table (62.77ms)**

**Migrating: 2014\_10\_12\_100000\_create\_password\_resets\_table**

**Migrated: 2014\_10\_12\_100000\_create\_password\_resets\_table (27.36ms)**

**Migrating: 2019\_08\_19\_000000\_create\_failed\_jobs\_table**

**Migrated: 2019\_08\_19\_000000\_create\_failed\_jobs\_table (27.32ms)**

**Migrating: 2019\_12\_14\_000001\_create\_personal\_access\_tokens\_table**

**Migrated: 2019\_12\_14\_000001\_create\_personal\_access\_tokens\_table (38.50ms)**

**php artisan make:migration create\_products\_table – Criar tabela com nome específico**

**php artisan migrate:fresh – Adiciona as inserções de elementos na tabela que está no banco de dados.**

public *function* up()

    {

*Schema*::create('products', *function* (*Blueprint* $table) {

            $table->id();

            $table->string('name');

            $table->integer('qty');

            $table->text('description');

            $table->timestamps();

        });

    }

**php artisan migrate:status – olhar os status das migrações.**

**php artisan migrate:reset – desfazer as migrations, rollback.**

**php artisan migrate:refresh – irá fazer o rollback e o migrate respectivamete.**

**php artisan migrate:fresh – reseta as tabelas, e refaz tudo novamente.**

**php artisan make:model Event – criar um model.**

**Exemplos de prática.**

<h1>Teste do vermelho</h1>

    <p><img src="/img/banner.jpg" alt="Banner"></p>

    @for($i = 0; $i < count($arr); $i++)

        <p>{{$arr[$i]}} - {{$i}}</p>

        @if($i == 2)

        <p style="color: green; background-color: black;">O i é 2</p>

        @endif

    @endfor

    @foreach($nomes as $indice => $nome)

        <p>{{$indice}} - {{$nome}}</p>

        <p>{{$loop->index}}</p>

    @endforeach

    @php

        $name = "Wesley em PHP";

        echo $name;

    @endphp