UTFPR

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DIO CEZAR GO

LARAVEL

Parte 5

VUE E LARAVEL

- ao utilizar o *Vue.js* com o *Laravel*, estaremos delegando a responsabilidade de renderização das informações vindas do *Back-End* para o *Vue.js*
- para isso, precisamos desacoplar as chamadas das views;
- agora, não chamamos mais as views que retornam arquivos do *Blade*;

VUE E LARAVEL

- neste cenário → toda *Controller* deve consultar uma *Model* e retornar um JSON;
- note que, neste exemplo, o acesso está liberado, sem a necessidade de qualquer autenticação;
- opcionalmente pode-se definir suas rotas em *api.php* em *routes*
    - nesse arquivo também é possível exigir a autenticação de quem está fazendo o *request*;

CONTROLLER JSON

EXEMPLO DE RETORNO DE JSON PELA CONTROLLER

```
<?php

namespace App\Http\Controllers\Front;

use App\Post;
use App\Comment;
use App\Http\Controllers\Controller as Controller;

use Illuminate\Http\Request;

class FeedController extends Controller{
    public function getPosts(){
           $posts = Post::all();
           return response()->json(["posts" => $posts->toArray()]);
    }
}
```

VUE E LARAVEL - PREPARAÇÃO

- agora devemos de fato ativar toda a estrutura para compilação dos arquivos do *Vue.JS*;
- na versão 5.5 do *Laravel*, um exemplo já está pronto para ser usado, bem como as configurações do *webpack*, então basta utilizar o comando:

`npm install` ← instala todas as dependências de *JavaScript* do projeto;

`npm run dev` ← compila os arquivos e colocar na pasta public

`npm run watch` ← observa os arquivos e faz a compilação se houver alteração;

VUE E LARAVEL - CONFIGURAÇÃO

- o arquivo *package.json* possui todas as dependências utilizadas pela estrutura, bem como uma lista de comandos que podem ser utilizados;
- o *Laravel* utiliza o *laravel-mix* para configurar uma *api* para compilação do *webpack*;
- seu arquivo de configuração é bem simples e funciona de forma semelhante ao *Gulp*;
- as configurações podem ser feitas em *webpack.mix.js*

EXEMPLO LARAVEL MIX

EXEMPLO LARAVEL MIX

```
let mix = require('laravel-mix');

mix.js('resources/assets/js/app.js', 'public/js')
   .sass('resources/assets/sass/app.scss', 'public/css');
```

VUE E LARAVEL - RESOURCES

- note que na pasta *resources* temos a pasta *assets*;
- são esses *assets* que serão observados e compilados pelo sistema;
- note também que a estrutura também irá compilar arquivos *sass*;
- em *js* devem ser organizados nossos componentes *Vue.js*;

VUE E LARAVEL - APP.JS

- é o arquivo principal a ser utilizado para incorporação dos seus componentes *Vue.js*;
- note que este arquivo importa *bootstrap;*
- é uma configuração inicial de tudo que será utilizado em conjunto com o *Vue.js*:
    - *lodash* ← biblioteca com uma série de bibliotecas úteis para *JavaScript*;
    - *jquery*;
    - *axios* ← utilizado para fazer o request dos *JSONs*;

VUE E LARAVEL - APP.JS

- ainda no *app.js* importamos o *Vue.js*:

  ```
  window.Vue = require('vue');
  ```

- e podemos efetuar a importação de nossos componentes, que ficam na pasta *components*;

  ```
  import CompFooter from './components/CompFooter.vue';
  ```

- e finalmente declarar nosso aplicativo;

EXEMPLO APP.JS

EXEMPLO APP.JS

```
require('./bootstrap');
window.Vue = require('vue');

import CompFooter from './components/CompFooter.vue';

const vueApp = new Vue({
    el: '#vue-app',
    components: { CompFooter }
});
```

VUE E LARAVEL - COMPONENTE VUE.JS

- como vimos, podemos criar um componente para o *footer* em questão;
- *CompFooter.vue*;
- neste componente podemos definir:
  - *template* → será o html a ser exibido no lugar em que o template for chamado;
  - *script* → será a estrutura javascript a ser utilizadas, aqui serão definidos comandos;
  - *style* → serão os CSS personalizados para este componente;

EXEMPLO COMPFOOTER.VUE

EXEMPLO COMPFOOTER.VUE - PARTE 1

```
<template>
      <div>
         <footer>
             <div class="container">
                 <div class="row">
                     <div class="col-lg-12">
                         <small class="text-center">{{text}}</small>
                     </div>
                 </div>
             </div>
         </footer>
     </div>
</template>

...
```

EXEMPLO COMPFOOTER.VUE

EXEMPLO COMPFOOTER.VUE - PARTE 2

```
<script>
export default {
    data() {
        return {
            text : 'Todos os direitos reservados'
        }
    }
}
</script>

...
```

EXEMPLO COMPFOOTER.VUE

EXEMPLO COMPFOOTER.VUE - PARTE 2

```
<style>
.text-center {
    text-align: center;
    width: 100%;
    position: relative;
    float: left;
}
</style>
```

VUE E LARAVEL - VIEW

- em nossa *view* precisamos:
  - importar o *JavaScript* compilado pelo *webpack*;
  - definir a *tag* que irá renderizar o componente;

EXEMPLO VIEW

EXEMPLO VIEW

```
<!DOCTYPE html>
<html lang="{{ app()->getLocale() }}">
<head>
    ...
</head>
<body>
    <div id="vue-app">
        @yield('content')
        <comp-footer></comp-footer>
    </div>
    <script src="{{ asset('js/app.js') }}"></script>
</body>
</html>
```

VUE E LARAVEL - DISCUSSÃO

- e você vai deixar todos os seus componentes em um único arquivo JS?
- quando isso é viável?
- qual o tamanho da sua aplicação?
- quantos componentes?
- qual o tamanho do seu app.js?
- alternativas?
    - criação de componentes separados e incluídos mediante a necessidade;

EXEMPLO

- um estudo de exemplo de *Laravel + Vue.js*
- https://goo.gl/qXnCEn

THAT'S ALL FOLKS

MATERIAIS COMPLEMENTARES

- https://goo.gl/gbyDMm
- https://goo.gl/rMXoTH
- https://goo.gl/Wzezup