

# VueJS: Uma Abordagem Sobre a Ferramenta

Beatriz Michelson Reichert<sup>1</sup>, Geremias Correa<sup>1</sup>

<sup>1</sup> Departamento de Ciência da Computação  
Universidade do Estado de Santa Catarina (UDESC) – Joinville, SC – Brasil

beatrizreichert99@gmail.com, gere.c@hotmail.com

**Resumo.** *A framework VueJS vem cada vez ganhando mais espaço, seja pela simplicidade, pelo objetivo da ferramenta ser útil para todos que utilizam a linguagem JavaScript ou mesmo porque se trata de uma ferramenta de acesso e edição livres. Este trabalho aborda um pouco dela, desde seu surgimento até suas funcionalidades, seu ecossistema, seus recursos e mesmos comparações e complementos para com outras ferramentas do gênero.*

## 1. Introdução

O VueJS – que pronuncia-se como a palavra *view*, do inglês –, é uma *framework* de código aberto do Javascript para construções de interfaces destinadas ao usuário, que é a camada do usuário, além de também aplicações do tipo *single-page*. Ela parte de implementações de bibliotecas para uso, existindo também todo um ecossistema que dá suporte a tais biblioteca e ajuda na aplicação destas mesmo que em escala [You 2013].

Neste trabalho pretende-se abordar tal *framework*, assim como uma abordagem histórica da mesma, suas funcionalidades, ferramentas e aplicações úteis, além de comparações e relações com outras plataformas de uso.

O VueJS já se desenvolveu muito e ainda tem muito o que se desenvolver, pois tem um potencial gigante tanto para soluções e melhorias quanto pelo público alvo que recebe diretamente tal tecnologia, por conta disso se considera esse um trabalho extremamente válido de ter seu conteúdo abordado, a fim de estudá-lo, analisá-lo e poder tirar conclusões do mesmo, como o quão útil é, como funciona, possíveis melhorias e, quem sabe, vir a utilizá-lo no futuro.

## 2. História do VueJS

O VueJS surge em desenvolvimento em julho de 2013 e com lançamento oficial fevereiro de 2014, 6 anos atrás, criado por Evan You, como uma plataforma de código aberto para desenvolver bibliotecas focadas na interface com o usuário na linguagem JavaScript. Evan já havia trabalho para a Google na criação do AngularJS, pensando, após isso, então em criar algo que extraísse apenas a parte que ele realmente achasse legal e bonito e construísse algo leve e focado nisso, que é no caso então essa relação com a interface apresentado ao usuário em aplicações *single-page*.

A *framework* está licenciada pelo MIT e sua versão, que está sempre em constante atualização, atualmente é a 2.6.11, de 13 de dezembro de 2019. Ela está disponibilizada gratuitamente no GitHub, cujo também permite colaboração livre do público, contando também hoje com, por exemplo, mais de 160 mil *stargazers* – espécie de seguidores – na versão mais acompanhada no site [You 2013].

### 3. Funcionalidades

A *framework* permite e é composta por diversas funcionalidade diferentes, onde cada uma possui sua peculiariedade e importância dentro da plataforma.

#### 3.1. Componentes

De forma geral, estes são pedaços de código do tipo Html, CSS ou JavaScript, ao qual podem ser encapsulados em algum tipo de biblioteca e então importá-los para uso dentro de algum outro código, dando a ideia do que conhecemos como *templates*. Juntos estes podem compor interfaces mais agradáveis e padronizadas e, portanto, reaproveitáveis. Como é uma das características dos componentes de programação Web, existe a possibilidade do uso de tags de marcação Html personalizadas, de uso simples e objetivo, facilitando o uso também para o caso do VueJS [You 2013].

Existem diversos componentes eficazes prontos e que podem mesmo serem desenvolvidos por qualquer usuários e adaptados às bibliotecas. Existem métodos para tal, por exemplo, para registro de um novo componente você precisa primeiro registrá-lo através de um construtor – ou seja, instanciá-lo – e então atribuí-lo como *.extend()* [Vuejs.org 2020].

#### 3.2. Modelos

O VueJS possui e permite a criação e utilização de *templates*, ou seja, implementações de código para posterior utilização. Ela permite isso através de uma linguagem baseada e válida em HTML, permitindo também vincular o DOM renderizado aos dados da instância realizada no Vue, o que minimiza a quantidade de manipulações na transição e permite a utilização destes *templates*. Hoje, para tal, é utilizado o ReactDOM, que trabalha em conjunto ao VueJS no Javascript.

A partir da utilização do JSX (ou então de um DOM virtual para renderização, o que é mais complexo), extensor de arquivo do JavaScript, os usuários podem selecionar escrever diretamente nas *render functions*, ou então apenas selecionar as sintaxes *template* disponíveis para uso [You 2016].

#### 3.3. Reatividade

Programação reativa aborda um paradigma declarativo de programação em conjunto de um alto fluxo de dados e propagação de mudanças. O sistema reativo do Vue é extremamente discreto utiliza objetos do JavaScript e otimização de renderização em looping, através de reatividade das dependências do que está sendo renderizado [Vuejs.org 2020].

#### 3.4. Transições

O Vue permite uma variedade de aplicações em questão de efeitos de transição, os quais são usados nos comandos *update*, *insert* e *delete* do DOM, utilizando-se de ferramentas como Animate.css e Velocity.js, que realizam o processo de *wrapper* nele, que se aproximaria da ideia de embrulhar ou encapsular o elemento.

Além disso, podemos dizer que o sistema de transição do Vue permite diversas maneiras simples e interativas de animação de entrada, saída, assim como listas. Para isso se usam características como cálculos numéricos, a disposição das cores, posicionamentos dos nós SVG – que tratam dos vetores gráficos escaláveis, ou *scalated vectorial graphics* –, além do tamanho e outras propriedades dos elementos.

### 3.5. Encaminhamento

A *framework* dispõe de uma biblioteca específica e otimizada para realizar o *routing*, ou seja, o encaminhamento, transmissão e integração dos dados para as fontes de destino das *single-page applications*, chamada vue-router. De forma geral, isso na prática é representado por quando se coloca links/caminhos dentro da página para serem levados para outros lugares, para isso ser feito de forma otimizada e sem problemas é utilizado um roteamento através do vue-router.

Caso seja desejado optar por outras formas de realizar o roteamento – para não ter que adicionar outra biblioteca para compilação no código, por exemplo –, também é possível, tais como através do Page.js, Director ou por roteamento dinâmico, apesar que principalmente esta última seja indicada para roteamento com rotas bem simples.

### 3.6. API

Uma API nada mais é que uma interface de uma aplicação desenvolvida a partir de um programa, ou seja, um software. Esta API para funcionar corretamente na *framework* VueJS precisa possuir três partes: *props* ou acessórios, eventos e *slots* ou espaços. O primeiro trata de permitir o ambiente externo alimentar dados ao componente da API, o segundo quanto a permitir o componentes disparar ações ao ambiente externo, enquanto o terceiro busca permitir que o ambiente externo possa inserir conteúdo dentro da estrutura de visualização do componente.

Para uma API funcionar, ela também trabalha diretamente e em perfeita sintonia com as outras funcionalidades citadas anteriormente, como o roteamento, os componentes, os modelos, assim como os outros. A API seria a forma bem executada de se realizar que se tem dentro do VueJS com o seu objetivo final, apresentar uma aplicação de qualidade com uma interface que represente o objetivo do sistema requisitado.

## 4. Ecossistema

Existem algumas ferramentas e bibliotecas fundamentais que um bom desenvolvedor do Vue pode usar. Fazendo uso delas você pode aumentar sua produtividade, facilitando o desenvolvimento do projeto [Suassuna 2020].

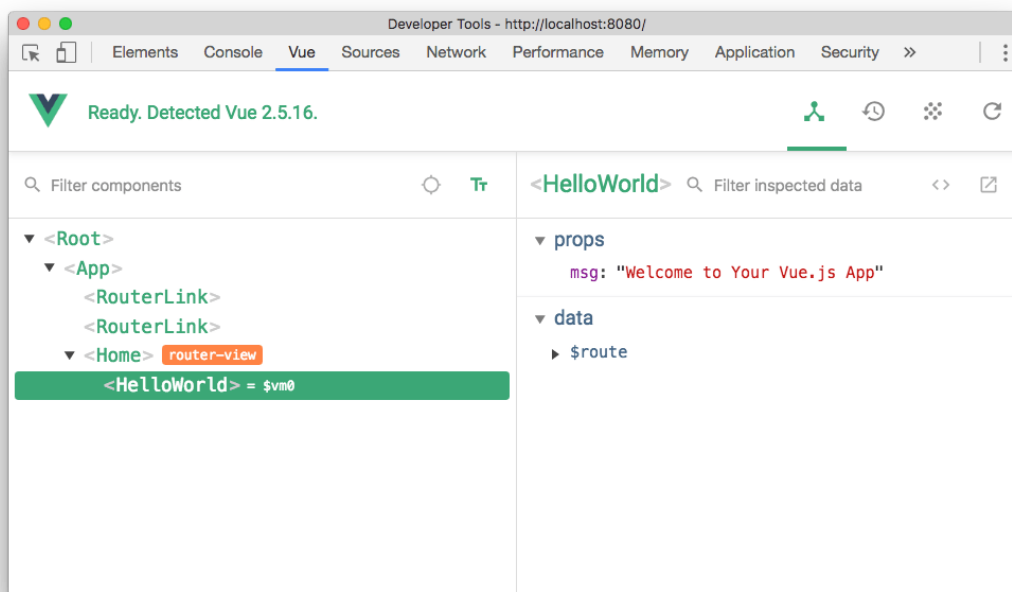
### 4.1. Ferramentas

Aqui serão citadas três ferramentas originais do Vue: Devtools, Vue CLI e Vue Loader.

#### 4.1.1. Devtools

Extensão Devtools do navegador para depuração de aplicativos Vue.js. É um método para depurar, disponível para o Chrome e Firefox. Com ele você pode ativar a edição em tempo real das propriedades e ver as mudanças refletidas imediatamente. Outro benefício é a possibilidade de ver o histórico de alterações do Vuex [Vuejs.org 2020].

Ao abrir o painel do Vue Devtools, você pode navegar na árvore de componentes. Podemos observar na figura 1 que quando é escolhido um componente da lista à esquerda, o painel da direita mostra os objetos e dados que ele contém [Copes 2018].



**Figure 1. Painel Vue Devtools**

Fonte: [Copes, 2018]

#### 4.1.2. Vue CLI

É um conjunto completo de ferramentas para um rápido desenvolvimento em Vue. Ele possui suporte para o uso de Babel, TypeScript, ESLint, PostCSS, PWA, Unit Testing e End-to-end Testing. Permite que o usuário crie, desenvolva e gerencie seus projetos através de uma interface gráfica [Vuejs.org 2020].

Ele é extensível, ou seja, o sistema de plug-ins permite que a comunidade construa e compartilhe soluções reutilizáveis para necessidades comuns. O CLI do Vue é totalmente configurável sem a necessidade de ejeção, isso permite que seu projeto fique atualizado a longo prazo [Vuejs.org 2020].

#### 4.1.3. Vue Loader

O vue-loader é um carregador para *webpack* que permite criar componentes do Vue em um formato SFC (*Single-File Components*). A combinação de *webpack* e vue-loader fornece um fluxo de trabalho front-end moderno, flexível e extremamente importante para a criação de aplicativos Vue.js [Vuejs.org 2020].

*Webpack* é um empacotador de módulo. Ele pega um grupo de arquivos, trata cada um como um módulo, descobre as dependências entre eles e empacota em ativos estáticos prontos para implantação [Vuejs.org 2020].

O vue-loader fornece alguns recursos bastante usados, como permitir usar outros *webpack loaders* para cada parte de um componente do Vue, por exemplo, Sass para

<style> e Pug para <template>. Permite customizar blocos em um arquivo .vue que pode ter cadeias de *loaders* personalizadas aplicadas a eles. Permite tratar os ativos estáticos mencionados em <style> e <template> como dependências do módulo e lidar com eles usando *webpack loaders*. Também simula CSS com escopo definido para cada componente [Vuejs.org 2020].

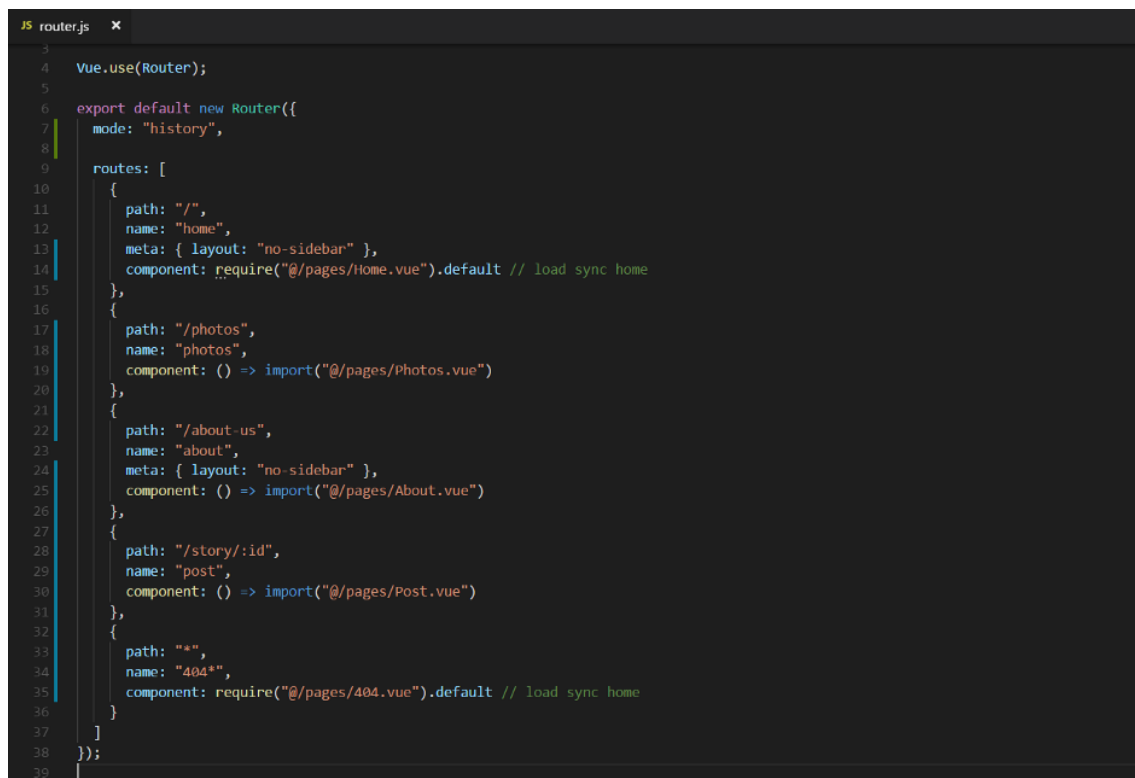
## 4.2. Bibliotecas

Aqui serão citadas três bibliotecas originais do Vue: Vue Router, Vuex e Vue Server Renderer.

### 4.2.1. Vue Router

Vue Router é o "roteador" oficial do Vue.js. Segundo [Vuejs.org 2020], ele facilita a criação de aplicativos *Single Page* e possui características como: roteiro aninhado/visualizar mapeamento, configuração modular de roteador baseada em componente, parâmetros de rota/consulta, permite ver os efeitos de transição, possui controle de navegação refinado, links com classes CSS ativas automáticas, comportamento de rolagem personalizável, entre outros.

Na figura 2 podemos observar um exemplo de aplicação do Vue Router, nesta imagem estão sendo adicionadas páginas de uma pasta específica aos *routers*.



```
JS router.js
3
4 Vue.use(Router);
5
6 export default new Router({
7   mode: "history",
8
9   routes: [
10    {
11      path: "/",
12      name: "home",
13      meta: { layout: "no-sidebar" },
14      component: require("@/pages/Home.vue").default // load sync home
15    },
16    {
17      path: "/photos",
18      name: "photos",
19      component: () => import("@/pages/Photos.vue")
20    },
21    {
22      path: "/about-us",
23      name: "about",
24      meta: { layout: "no-sidebar" },
25      component: () => import("@/pages/About.vue")
26    },
27    {
28      path: "/story/:id",
29      name: "post",
30      component: () => import("@/pages/Post.vue")
31    },
32    {
33      path: "**",
34      name: "404*",
35      component: require("@/pages/404.vue").default // load sync home
36    }
37  ]
38 });
39
```

Figure 2. Páginas sendo adicionadas aos *routers*

Fonte: [Itnext 2018]

#### 4.2.2. Vuex

Vuex é um padrão de gerenciamento de estado + biblioteca para aplicativos Vue.js. Ele serve como um armazenamento centralizado para todos os componentes em uma aplicação, onde o estado só pode ser alterado de forma previsível. Também pode ser integrado à extensão Devtools do Vue, fornecendo recursos avançados [Vuejs.org 2020].

O Vuex Store é um objeto importante do Vuex e é bastante similar a uma instância do Vue. Segundo [Suassuna 2020], ele é dividido em quatro partes:

- *state*: é um objeto que armazena os dados do aplicativo.
- *mutations*: é um objeto que contém métodos que afetam diretamente o estado
- *actions*: é um objeto que contém métodos usados para acionar mutações e executar código assíncrono
- *getters*: é um objeto que contém métodos usados para abstrair o acesso ao estado



```
const store = new Vuex.Store({
  state: {
    ...
  },
  mutations: {
    ...
  },
  actions: {
    ...
  },
  getters: {
    ...
  }
})
```

O diagrama mostra um código JavaScript para criar um novo Vuex.Store. O objeto de configuração contém quatro propriedades principais: state, mutations, actions e getters. Cada propriedade é seguida por um ícone circular: state (caixa azul), mutations (engrenagem vermelha), actions (seta amarela) e getters (lupa cinza).

Figure 3. Exemplo da estrutura do Vuex.Store

Fonte: [Vuejs.org 2020]

#### 4.2.3. Vue Server Renderer

Vimos que, por padrão, os componentes do Vue produzem e manipulam o DOM no navegador como saída. Porém, também é possível renderizar os mesmos componentes em sequências de caracteres HTML no servidor, enviá-los diretamente para o navegador e então "hidratar" a marcação estática em um aplicativo interativo no cliente [Vuejs.org 2020].

A "hidratação" refere-se ao processo do lado do cliente durante o qual o Vue assume o HTML estático enviado pelo servidor e o transforma em DOM dinâmico que pode reagir às alterações de dados do lado do cliente [Vuejs.org 2020].

O Vue Server Renderer, comparado com uma aplicação *Single-Page*, possui um tempo de conteúdo mais rápido, especialmente em internet lenta ou dispositivos lentos.

Porém, o Server Renderer possui restrições de desenvolvimento, requer um ambiente em que um servidor Node.js possa ser executado e exige mais CPU [Vuejs.org 2020].

## 5. Data Bind

O Data Bind é um recurso do Vue.js bastante importante, pode ser classificado como unidirecional ou bidirecional, ele permite ligar as tags HTML aos dados definidos no JavaScript. Feita essa ligação, o elemento da interface passará a ser atualizado sempre que os objetos no script sofrerem mudanças. Chamamos essa relação de *one-way binding*. Quando essa atualização acontece nos dois sentidos podemos dizer que temos um *two-way binding* [Rodrigues 2017].

### 5.1. Unidirecional

Primeiro é preciso declarar a propriedade **data** no objeto **Vue** e definir nela as demais propriedades que serão ligadas às tags HTML [Rodrigues, 2017]. Em seguida, para mostrar seu valor na tela usamos `{{titulo}}`, conforme a linha 2 da figura 4 [Rodrigues 2017].

Segundo [Rodrigues 2017], para que o Data Bind funcione, as tags com as quais desejamos ligar os dados devem estar no interior da div, esta foi iniciada na linha 1 da figura 4. Ao abrir esse documento no *browser* temos como resultado a figura 5.

```
1 <div id = "app">
2   <h1>{{titulo}}</h1>
3 </div>
4
5 <script src = "https://unpkg.com/vue"></script>
6 <script>
7   var app = new Vue({
8     el: '#app',
9     data : {
10      titulo : 'Disciplina de Desenvolvimento de Aplicações na Web'
11    }
12  });
13 </script>
```

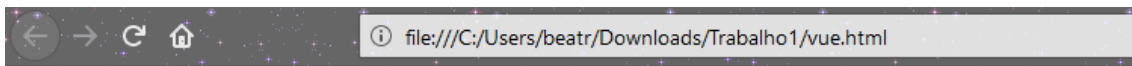
Figure 4. Exemplo de Data bind unidirecional

Fonte: Dos autores

### 5.2. Bidirecional

Segundo [Rodrigues 2017], no Data Bind bidirecional podemos fazer alterações nos dados a partir da view. Para isso, devemos fazer essa ligação por meio da diretiva **v-model**. Será necessário inserir um input dentro da div app, observamos essa modificação na linha 3 da figura 6.

A partir desse input poderemos modificar o título da página e ver as mudanças acontecerem simultaneamente, sem a necessidade de alterar o código fonte do projeto. Notamos o resultado desta modificação na figura 7.



## Disciplina de Desenvolvimento de Aplicações na Web

**Figure 5. Conteúdo exibido via data bind**

Fonte: Dos autores

```
1 <div id = "app">
2   <h1>{{titulo}}</h1>
3   <input type="text" v-model="titulo">
4 </div>
5
6 <script src = "https://unpkg.com/vue"></script>
7 <script>
8   var app = new Vue({
9     el: '#app',
10    data : {
11      titulo : 'Disciplina de Desenvolvimento de Aplicações na Web'
12    }
13  });
14 </script>
```

**Figure 6. Exemplo de data bind bidirecional**

Fonte: Dos autores

## 6. Comparações

Segundo [Vuejs.org 2020], Vue e React possuem algumas similaridades, por exemplo:

- utilizam a abordagem de DOM virtual
- provêm componentes visuais reativos e combináveis
- mantêm o foco na biblioteca principal, com preocupações como roteamento e gerenciamento de estado global tratadas por bibliotecas companheiras

Porém, observamos que em questão de ecossistema mais completo e um maior número de renderizadores, o React superou o Vue. Porque ele deixa para a comunidade a preocupação de manter as bibliotecas de gerenciamento de estado e roteamento sempre atualizadas em relação à biblioteca principal. Criando um ecossistema fragmentado, e consideravelmente mais rico do que o do Vue [Vuejs.org 2020].

Segundo [Vuejs.org 2020], ao falar em esforço e otimização o Vue aparenta ser melhor, porque ele tira do desenvolvedor a responsabilidade de conhecimento de toda a gama de otimizações de desempenho, permitindo ao desenvolvedor focar em construir a aplicação.





# Desenvolvimento de Aplicações na Web

Desenvolvimento de Apli

**Figure 7. Conteúdo exibido via Data Bind**

Fonte: Dos autores

Em se tratando de escalabilidade, ambos possuem soluções de roteamento robustas. Porém, no Vue temos o Vuex, que é uma solução de gerenciamento de estado inspirada em Elm, já citada anteriormente. Ele se integra profundamente com o Vue, e acredita-se que oferece uma experiência de desenvolvimento superior [Vuejs.org 2020].

Um ponto importante é a curva de aprendizado. Para usar o React é necessário saber sobre JSX, ES2015+ e processos de transpilação. Enquanto que no Vue, isso não será necessário, e um desenvolvedor leva menos de um dia para aprender o suficiente para criar aplicações triviais [Vuejs.org 2020].

## 7. Conclusão

Era tido o objetivo aqui de abordar alguma linguagem ou ferramentas referente a disciplina de ODAW, a fim de agregar conhecimento aos alunos quanto as tecnologias existentes para tal. Decidimos optar pela *framework* Vue.js, bastante em alta no momento. Realizar o mesmo foi uma experiência satisfatória e serviu de grande aprendizados para ambos os alunos da equipe, tal qual pode vir, sim, a ser útil no futuro.

Foram abordados, de forma breve, a história, as funcionalidades e o ecossistema da *framework* do Vue.js. Foi feita também uma comparação com a biblioteca React, apontando pontos positivos e negativos de ambas as ferramentas. Entre as funcionalidades, foi abordado sobre os componentes, modelos, reatividade, transições, encaminhamento e API. De forma breve, discutimos sobre ferramentas e bibliotecas originais da *framework*, são elas: Devtools, Vue CLI, Vue Loader, Vue Router, Vuex e Vue Server Renderer. Com base em um exemplo encontrado, foi feito um teste prático sobre Data Bind unidirecional e bidirecional, apresentando seus resultados no artigo.

Não tínhamos nenhum conhecimento sobre essa *framework*, mas com o que desenvolvemos aqui conseguimos ter uma boa base, mesmo que abordando de forma breve alguns assuntos, pois o conteúdo é bem denso. É uma ferramenta de uso fácil e prático, que pode vir a ser usada pelos autores em trabalhos futuros, devido a sua simplicidade, aprendizagem rápida e comentários positivos de usuários.

## 8. Referências

Copes, F. (2018). The Vue.js DevTools. Disponível em: <<https://flaviocopes.com/vue-devtools/>>. Acesso em: 20 abr. 2020.

Darklymnx (2018). Anyway, Here's How to Create a Multiple Layout System with Vue and VUE-Router. ITNEXT. Disponível em: <<https://itnext.io/anyway-heres-how-to-create-a-multiple-layout-system-with-vue-and-vue-router-b379baa91a05>>. Acesso em: 20 abr. 2020.

Rodrigues, J. (2017). Vue.js: Utilizando data bind unidirecional e bidirecional. DEVMEDIA. Disponível em: <<https://www.devmedia.com.br/vue-js-utilizando-data-bind-unidirecional-e-bidirecional/38026>>. Acesso em: 20 abr. 2020.

Suassuna, A. (2020). 10 Principais Ferramentas e Bibliotecas do VueJS. TipsCode. Disponível em: <<https://www.tipscode.com.br/10-principais-ferramentas-bibliotecas-do-vuejs/>>. Acesso em: 20 abr. 2020.

Vuejs.org (2020). The Progressive JavaScript Framework. Disponível em: <<https://vuejs.org/>>. Acesso em: 19 abr. 2020.

You, E. (2013). Vuejs, GitHub repository. Disponível em: <<https://github.com/vuejs/vue>>. Acesso em: 19 abr. 2020.

You, E. (2016). Vue 2.0 is Here!. The Vue Point. Disponível em: <<https://medium.com/the-vue-point/vue-2-0-is-here-ef1f26acf4b877d9>>. Acesso em 19 abr. 2020.