FACULDADE DE TECNOLOGIA DE SÃO PAULO
VICTÓRIO DE OLIVEIRA CAPUCÇO

Vantagens e desvantagens no desenvolvimento de aplicações com frameworks e bibliotecas: Uso de Vue, Bulma e Buefy

SÃO PAULO
2021

FACULDADE DE TECNOLOGIA DE SÃO PAULO
VICTÓRIO DE OLIVEIRA CAPUCÇO

Vantagens e desvantagens no desenvolvimento de aplicações com frameworks e bibliotecas: Uso de Vue, Bulma e Buefy

Trabalho submetido como exigência parcial para a obtenção do Grau de Tecnólogo em Análise e Desenvolvimento de Sistemas Orientador:
Professor Mestre Sergio Luiz Banin

SÃO PAULO
2021

FACULDADE DE TECNOLOGIA DE SÃO PAULO

VICTÓRIO DE OLIVEIRA CAPUCÇO

Vantagens e desvantagens no desenvolvimento de aplicações com frameworks e bibliotecas: Uso de Vue, Bulma e Buefy

Trabalho submetido como exigência parcial para a obtenção do Grau de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Parecer do Professor Orientador:

_____

_____

_____

Conceito/Nota Final: _____

Atesto o conteúdo contido na postagem do ambiente TEAMS pelo aluno e assinada por mim para avaliação do TCC.

Orientador: Professor Mestre Sergio Luiz Banin

SÃO PAULO, ____ de _____ de 2021

Assinatura do Orientador

Assinatura do aluno.

Dedico este trabalho a todos que estiveram presentes em minha vida, positiva ou negativamente, em especial à minha mãe Eunice, minha irmã Carol e minha namorada Ana Beatriz.

Agradecimentos

Meus sinceros agradecimentos ao mestre Sergio Banin, por ter me ensinado a ser um profissional melhor e por me orientar neste trabalho, ao Centro Paula Souza por permitir que eu me tornasse o desenvolvedor que sou hoje, a todos os professores, em especial ao Irineu, Ciscato, Milton Rocha, Kurata e Alexandre Solis e a todos os outros funcionários da Fatec São Paulo e Etec Parque da Juventude.

Resumo

O uso de frameworks e bibliotecas em sistemas é muito comum na atualidade, trazendo inúmeras vantagens, e claro, algumas desvantagens. O objetivo deste trabalho é apresentar os principais conceitos que envolvem Vue, Bulma e Buefy, além de, em dados momentos, compará-los com tecnologias concorrentes, como React e Angular. Sendo assim, serão abordadas as características, seus pontos fortes, exemplos de empresas que fazem uso e opiniões gerais da comunidade.

A ideia central deste trabalho não é definir qual ferramenta é melhor, já que isso depende de inúmeras variáveis, tendo a ver com a ferramenta em si ou não. É possível avaliar os riscos e benefícios de se usar determinada tecnologia ao longo do texto, tendo uma breve análise na conclusão.

Palavras-chave: Vantagens e desvantagens de frameworks e bibliotecas; Vue; Bulma; Buefy.

Abstract

The use of frameworks and libraries in systems is very common nowadays, bringing advantages and some disadvantages. The purpose of this Final Paper is to present the main concepts involving Vue, Bulma and Buefy, comparing them with competing technologies, such as React and Angular at certain times. To characterize the technologies we are going to address the strengths, examples of companies that use them and general community opinions.

The main idea of this work is not to define which tool is better, as it depends on variables, having a relationship with the tool itself or not. It is possible to assess the risks and benefits of using the technology provided throughout the text, having a brief analysis at the conclusion.

Keywords: Advantages and disadvantages of frameworks and libraries; Vue; Bulma; Buefy.

SUMÁRIO

INTRODUÇÃO **10**

CONCEITOS BÁSICOS **11**
O que é Javascript? 11
O que é um framework? 13
O que é uma biblioteca? 14

AS TECNOLOGIAS **15**
Vue Js 15
Principais características 15
Exemplos de seus pontos fortes 19
Empresas que fazem uso 24
Opiniões da comunidade 25
Bulma 28
Principais características 28
Exemplos de seus pontos fortes 32
Empresas que fazem uso 37
Opiniões da comunidade 38
Buefy 40
Principais características 40
Exemplos de seus pontos fortes 43
Opiniões da comunidade 46

VANTAGENS NA UTILIZAÇÃO DE FRAMEWORKS E BIBLIOTECAS **47**
Comunidade dedicada 47
Resolução de problemas 47
Criação de novas funcionalidades 49
Documentação própria 50

DESVANTAGENS NA UTILIZAÇÃO DE FRAMEWORKS E BIBLIOTECAS **51**
Desvantagens funcionais 51
Comunidade pequena ou pouco ativa 51
Problemas estruturais na ferramenta 53
Desvantagens não funcionais 54
Risco da tecnologia ser descontinuada 54
Falta de mão de obra qualificada 55
Curva de aprendizagem 57

CONCLUSÃO **59**

REFERÊNCIAS **61**

1. INTRODUÇÃO

Com o uso cada vez maior da internet, diversos serviços, produtos e informações podem ser obtidos de forma virtual. Com isso, ao passar dos anos diversas tecnologias que visam a melhoria dos sites vão surgindo, algumas com paradigmas totalmente novos que geram grandes desafios para os desenvolvedores.

Talvez um dos maiores desafios para um profissional da área seja se adaptar e aprender cada vez mais sobre frameworks e bibliotecas, aprendendo sobre suas características e aplicações, além de saber dosar seu uso em determinadas empresas.

A ideia deste trabalho é apresentar Vue, Bulma e Buefy, 3 tecnologias modernas de construção de websites, mostrando suas principais características, vantagens e desvantagens. Durante todo o texto vários outros conceitos serão abordados para melhor embasamento, assim como outras tecnologias concorrentes também serão exemplificadas.

A razão pela qual Bulma, Buefy e Vue foram escolhidos juntos é o fato de que ambos trabalham muito bem em conjunto, pois foram feitos para trabalhar de forma simultânea. Já o motivo de mostrar os frameworks, bibliotecas e suas vantagens e desvantagens é o fato de que não são muito conhecidos (com exceção do Vue), e têm muito a agregar, quando usados com cautela.

Este trabalho foi escrito baseado em referências bibliográficas, usando principalmente as documentações oficiais de cada ferramenta, e a razão disso é justamente o fato de se tratarem de tecnologias novas, não tendo muitos livros e artigos a respeito.

É possível navegar facilmente pelo conteúdo selecionando o capítulo desejado. Em conceitos básicos, iremos abordar as ferramentas base que sustentam os frameworks e bibliotecas, enquanto que nas Tecnologias já trataremos das mesmas, com seus principais conceitos. Após citar muitas vantagens e desvantagens ao longo deste capítulo, teremos também outros dois específicos para mostrar as que todos têm em comum. Por fim, a conclusão traz uma coleção de informações resumidas, apresentando um pouco da opinião do autor sobre o tema.

2. CONCEITOS BÁSICOS

2.1. O que é Javascript?

Segundo a Mozilla (organização sem fins lucrativos, que mantém o navegador Firefox), Javascript é uma linguagem de programação que permite exibir mais que informação estática nas páginas web, atualizando dados em tempo real, mapas, gráficos, etc. Por ser uma linguagem de programação, ele permite utilizar-se de variáveis, funções, eventos, classes e uma das principais coisas: Integração com API'S (Application Programming Interfaces).

Ainda segundo a organização, API's são conjuntos de código que permitem a implementação de programas que seriam muito difíceis (ou impossíveis) de desenvolver. É possível dividi-las em

- API's de navegadores: São as que já vem implementadas nos *browsers* e permitem obter dados do ambiente em que o computador se encontra, como por exemplo a geolocalização do usuário (necessário autorização por parte do mesmo), criação de gráficos 2D e 3D, exibição de multimídia (áudio e vídeo), além de interagir com HTML e CSS (definição logo abaixo), podendo modificar a estrutura da página e a exibição de conteúdo.
- API's de terceiro: São obtidas a partir de outras organizações (empresas ou pessoas) e permitem retorno de informações e dados específicos. Um exemplo é a API do IBGE, para obter estados e municípios, além dos ceps (com as respectivas ruas e bairros).

Foi citado logo acima duas tecnologias específicas, *HTML* e *CSS*. Segundo Flatschart (p. 5 a 9), HTML (*HyperText Markup Language)* é uma linguagem de marcação, que remonta à época em que revisores profissionais de texto faziam marcações específicas, para que houvesse maior facilidade em reconhecer tais anotações no final da escrita, tanto que é muito utilizada para comunicação de editores, gráficas e autores na indústria moderna editorial. Ainda segundo o autor, esta é a principal linguagem utilizada atualmente na web, e se utiliza de *tags* (< e >) para indicar seus comandos.

O W3C (Consórcio *World Wide Web)* é o responsável por padronizar a web, desenvolvendo "protocolos e diretrizes que garantam seu crescimento de longo prazo" (Site oficial do consórcio).

Já com relação ao CSS (*Cascading Style Sheets)*, segundo a organização Mozilla, é uma linguagem utilizada para estilizar um documento *HTML*, descrevendo como os elementos multimídia serão exibidos. Assim como o *HTML, também é padronizado pela W3C.*

2.2. O que é um framework?

Baseado no que diz Lima e Lezana (2004, p. 178), um framework serve para traduzir temas complexos, de forma que possam ser estudados e analisados. É comumente utilizado para compartilhar ideias e descobertas com a comunidade, auxiliar no progresso de ferramentas, técnicas e procedimentos, e muito mais.

Esta foi uma definição genérica. Trazendo para o lado da computação encontramos definições mais específicas, como por exemplo no site HostGator (empresa de provedores e hospedagem web dedicada): “podemos dizer que é uma espécie de template que conta com diversas funcionalidades que podem ser utilizados pelo desenvolvedor em seus projetos. Ele conta com **ferramentas, sistemas, componentes e guias** que agilizam o processo de criação de soluções, sendo, portanto, um artifício essencial na vida de um profissional de TI.”

Ainda segundo a empresa, a principal finalidade de um framework é “resolver problemas recorrentes com uma abordagem genérica”, tornando desnecessário a reescrita de códigos, e obtendo muito mais produtividade na resolução de problemas. Além disso, o uso dessa tecnologia permite um desenvolvimento mais rápido e seguro de projetos, permitindo atender qualquer necessidade profissional, contanto que a escolha seja bem feita.

Em se tratando de desenvolvimento de aplicações web, há 2 frameworks muito conhecidos a se citar: Angular (desenvolvido pelo Google) e Vue (código aberto). Durante esse texto iremos abordar apenas o último, além do Bulma, “uma estrutura de código aberto gratuita que fornece componentes de front-end prontos para uso que você pode combinar facilmente para construir interfaces da web responsivas.” (Trecho extraído do site oficial da ferramenta).

2.3. O que é uma biblioteca?

Para Hanashiro, uma biblioteca é um conjunto de códigos que servem para resolver determinados problemas. Quando iniciamos na programação utilizamos e muitas vezes não nos damos conta, como por exemplo na linguagem C, com a biblioteca math.h, que segundo o professor Feofiloff, “ contém várias constantes e funções matemáticas”. Ou seja, é uma biblioteca nativa da linguagem.

Trazendo para um contexto mais atual, temos a biblioteca para JavaScript React (desenvolvido pelo Google), muitas vezes confundido com um framework, devido ao seu grande uso e aplicabilidade. Segundo o site oficial desta tecnologia, “React faz com que a criação de UIs interativas seja uma tarefa fácil”.

Além do React, temos também o JQuery, que segundo o site oficial é uma “uma biblioteca JavaScript rápida, pequena e rica em recursos.”. De acordo com o site DevMedia (maior Plataforma para desenvolvedores de software em língua portuguesa), essa biblioteca foi criada em 2006 por John Resig e é usada por 77% dos 10 mil sites mais visitados do mundo, sendo a biblioteca JavaScript mais popular.

Durante este trabalho iremos abordar a biblioteca Buefy, que segundo o repositório oficial no Github, é “uma biblioteca leve de componentes de IU responsivos para Vue.js com base na estrutura e design Bulma.”

3. AS TECNOLOGIAS

3.1. Vue Js

3.1.1. Principais características

Segundo Filipova, Vue.Js foi criado por Evan You. A ideia da criação deste framework nasceu enquanto Evan trabalhava no Google Creative Labs, devido a sempre precisar fazer código repetido, além de diversos outros aspectos negativos. Antes de começar o desenvolvimento, You tentou pesquisar e não encontrou nenhuma ferramenta que se adequasse ao que precisava. Portanto, decidiu criá-la.

“Vue é um framework acessível, versátil e performático que ajuda a criar um código mais manutenível e simples de testar” (Pollack, 201-?). Ainda segundo Pollack, Vue é um framework progressivo, ou seja, se você já possuir uma aplicação (que contenha outras tecnologias) pode plugá-lo em apenas um pequeno trecho, podendo aproveitar de todas as facilidades que o Vue propõe neste espaço.

O autor propõe que as bibliotecas oficiais e o próprio ecossistema do Vue possuem vantagens para criar regras de negócio, pois é possível criar componentes reutilizáveis, cada um contendo seu próprio HTML, CSS e JavaScript, o que permite melhor organização e consequentemente aplicação de regras próprias.

O guia oficial deste framework, tem definições parecidas, porém, mais objetivas: “Ao contrário de outros frameworks monolíticos, Vue foi projetado desde sua concepção para ser adotável incrementalmente. A biblioteca principal é focada exclusivamente na camada visual (view layer), sendo fácil adotar e integrar com outras bibliotecas ou projetos existentes.”

Para Filipova, ao utilizar Vue.js não é necessário criar modelos ou coleções específicas, seguir uma sintaxe especial e nem mesmo instalar uma infinidade de dependências. E este pensamento está certo, pois de acordo com o guia oficial, para iniciar os estudos não é necessário instalar nada, basta colocar apenas uma linha no HTML:

```
<script src="https://cdn.jsdelivr.net/npm/vue@2"></script>
```

Em se tratando de projetos, não é utilizado a terminologia acima, e sim o Vue-Cli, um conjunto de ferramentas padrão para o desenvolvimento com Vue, rico

em recursos, extensível e com interface gráfica ao usuário (Seção Vue-Cli - Vue.Js, 201-?).

Ao utilizar Vue.js pela primeira vez, é muito comum se deparar com conceitos novos e formas diferentes de desenvolver. Mas primeiro precisamos entender o que é o DOM (*Document Object Model)*. Segundo a organização Mozilla (Seção Tecnologia Web para desenvolvedores) “é uma interface de programação para os documentos HTML e XML”, na qual “representa o documento com nós e objetos, dessa forma, as linguagens de programação podem se conectar à página.”.

Entendido do que se trata o DOM, com Vue.Js podemos manipulá-lo mais facilmente. Segundo o guia oficial do Vue, podemos renderizar dados facilmente no DOM devido a um sistema no núcleo do framework. É possível entender melhor com o exemplo de código abaixo extraído do mesmo artigo:

HTML:

```
<div id="app">
  {{ message }}
</div>
```

JavaScript:

```
var app = new Vue({
  el: '#app',
  data: {
    message: 'Olá Vue!'
  }
})
```

O que será exibido na tela é a mensagem “Olá Vue!”. É totalmente possível fazer isso apenas com HTML e JavaScript, porém, com Vue se torna mais fácil tornar dinâmicas essas informações exibidas.

Segundo Caldeira, diretivas são basicamente elementos HTML que são adicionados dentro de modelos. Tendo em vista esta definição, podemos explorar as diretivas do Vue. Ainda segundo o guia educativo oficial do do Framework, as

diretivas no Vue começam com -v. Para entender melhor, temos um exemplo muito bom do site:

HTML:

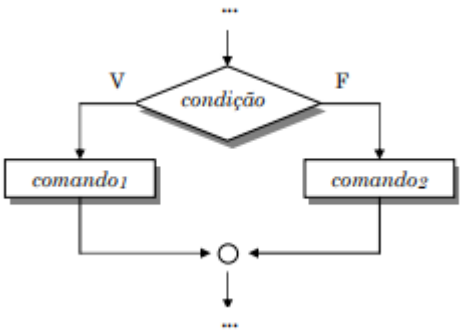
```
<div id="app-3">
  <p v-if="seen">Agora você me viu</p>
</div>
```

JavaScript:

```
var app3 = new Vue({
  el: '#app-3',
  data: {
    seen: true
  }
})
```

No código acima, temos algo muito interessante: exibição condicional. “A estrutura condicional ou de decisão simples serve para escolher um entre dois comandos alternativos“ (Pereira, 20--?). Ainda segundo Pereira, essa escolha pode ser feita na programação com os comandos *if* e *else*, sendo que o primeiro indica o que queremos, e o segundo o que não está incluso no que queremos. Fica mais simples de entender com a figura 1 abaixo, extraída da mesma apostila do autor:

figura 1: A estrutura de decisão simples



Sabendo disso, fica simples entender o exemplo do guia do Vue. Segundo Rocha, variáveis são valores que podem ser alterados durante a execução de uma aplicação, representando uma posição de memória que armazena dados,

identificado por um nome. Vemos no exemplo citado que data é uma variável (do tipo objeto), tendo como propriedade *seen*, a qual tem o valor *true*

Levando em consideração que *true* traduzido literalmente para o português é “verdadeiro”, vemos que a propriedade *seen* no código é verdadeira, ao mesmo tempo em que o texto “Agora você me viu” está sujeito ao *if*, ele só será exibido se for verdadeiro.

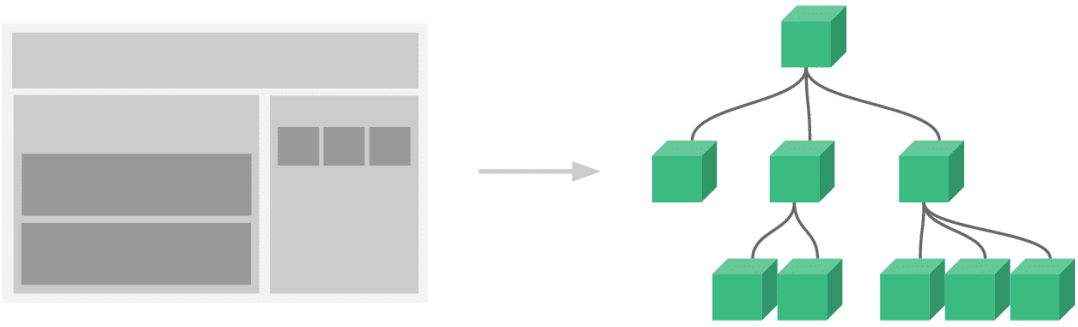
Portanto, caso em algum momento da execução queiramos que esse texto não exista, basta trocar o valor para *false*, que traduzido literalmente significa “falso”. É importante ressaltar que há uma diferença entre algo não existir e não ser exibido em um site. Para o segundo caso temos a propriedade *hidden* do CSS.

3.1.2. Exemplos de seus pontos fortes

Um dos principais pontos fortes do Vue é seu sistema de componentes, já citados neste texto. É possível obter uma definição muito esclarecedora na introdução do guia oficial do Vue: “uma abstração que proporciona a construção de aplicações de larga escala compostas por pequenos componentes, auto-contidos e frequentemente reutilizáveis” (VueJs Org, 201?).

Interessante observar que funciona de forma parecida com uma árvore, onde componentes podem chamar outros componentes, igual na figura 2 abaixo, extraída dessa mesma fonte:

Figura 2: Árvore de componentes



Portanto, caso seja necessário desenvolver um site no qual tenha um menu de navegação (o que é muito comum) não será necessário criá-lo em todas as páginas, basta apenas criar um componente com o conteúdo desejado e chamá-lo, seja nas páginas, ou em um *layout* padrão, evitando assim duplicação de código e possíveis dores de cabeça em futuras alterações. Para isso é muito simples, basta adaptar o exemplo do guia oficial do Vue, como podemos ver abaixo:

JavaScript:

```
// Define um novo componente chamado menu
Vue.component(menu', {
  template: '<li>Isso é um menu</li>'
})
```

```
var app = new Vue(...)
```

HTML:

```
<ol>
  <!-- Cria uma instância do componente menu -->
  <menu></menu>
</ol>
```

Como podemos ver acima, chamamos o menu a partir de uma *tag*, podendo incrementá-lo a vontade a partir daí. Porém, ainda segundo o guia, o exemplo acima não é a forma mais performática de criar um componente. Primeiro precisamos entender a hierarquia existente. De forma resumida, quem chama o componente é chamado de Pai, e o componente chamado é conhecido por Filho.

Tendo em vista esses conceitos. podemos apresentar uma forma mais adequada de utilizar o nosso menu:

JavaScript

```
Vue.component(menu', {
  props: [‘secao’],
  template: '<li>{{ secao.nome }}</li>'
})

var app7 = new Vue({
  el: '#app-7',
  data: {
    secoesMenu: [
      { id: 0, nome: 'Produtos' },
      { id: 1, nome: 'Serviços' },
      { id: 2, nome: 'Sobre' }
    ]
  }
})
```

HTML:

```
<div id="app-7">
  <ol>
    <menu
      v-for="secao in secoesMenu"
      v-bind:secao="secao"
      v-bind:key="secao.id"
    ></menu>
  </ol>
</div>
```

O código acima resultará no seguinte resultado:

1. Produtos
2. Serviços
3. Sobre

Alguns pontos nos códigos acima ainda não foram explicados, como por exemplo “*prop*”, que basicamente se trata de um atributo personalizado, no qual passamos dados para um componente. Além disso, surgiu uma tag nova, chamada *v-for*. Basicamente se trata de um laço de repetição

A instrução *for* é uma estrutura de repetição com um contador, que “ tem seu funcionamento controlado por uma variável que conta o número de vezes que o comando é executado“ (Pereira, p.26). Tendo esse conceito em vista, podemos afirmar que o *v-for* no *HTML* apresentado acima apresenta todas as seções contidas na variável “secoesMenu”, pois o mesmo se trata de um laço de repetição, nativo do VueJs.

Outro ponto interessante do Vue, é a possibilidade de interpolar textos facilmente. Segundo o guia oficial de Síntaxe de Templates do VueJs, levando em consideração que “texto” se refere a uma variável, podemos exibi-la utilizando de *Mustache* (chaves duplas), da seguinte forma para o usuário:

```
<p>Mensagem: {{ texto }}</p>
```

O que pode ter dentro da variável “texto”? Qualquer dado, seja um conjunto de caracteres, números, objetos, matrizes, etc. Além disso, ainda segundo a documentação, podemos fazer não apenas interpolar dados simples, mas também

tags, através da diretiva *v-html*. Digamos que é necessário exibir um texto em negrito, a *tag* necessária para isso é a <b>, portanto, vamos criar uma string e exibi-la na página:

JavaScript:

```
textoPersonalizado: '<b>Este texto está em negrito</b>'
```

HTML:

```
<p v-html="textoPersonalizado"></p>
```

Observação importante: Como estamos renderizando dinamicamente o *HTML*, precisamos ser cautelosos e nunca utilizar um conteúdo passado pelo usuário para esse fim, visto que, não temos controle sobre o que é informado.

Agora digamos que é necessário apresentar um formulário ao usuário, onde temos alguns campos. Para termos em mãos as respostas informadas em um campo de texto simples, precisamos de um *input* (do próprio *HTML*) e do *v-model*. Segundo o guia oficial de Interligações em Formulários do Vue, utilizamos *v-model* para criar interligações de mão dupla, ou seja, esta diretiva busca a maneira correta de atualizar dados de acordo com o tipo de entrada, isto é, texto, número, etc.

HTML:

```
<input v-model="message">
<p>A mensagem é: {{ message }}</p>
```

O exemplo acima foi extraído também do guia oficial. Nele será exibido um campo ao usuário para digitação, onde "*message*" pode ser alterado tanto pelo digitador, quanto em qualquer outra parte do código. Um exemplo prático de como poderíamos fazer uma ferramenta simples com esses conceitos, é criando uma função na qual tratamos o dado recebido pelo usuário, retornando ao mesmo o texto digitado em letras maiúsculas. Para isso, usaríamos o método nativo do JavaScript chamado "*toUpperCase()*".

Para realizar a tarefa acima, podemos utilizar os métodos do Vue. Porém, além dos métodos, temos algo parecido, chamado "Dados Computados". Na seção

de mesmo nome no guia oficial do Vue, conseguimos obter uma definição satisfatória: “Um dado computado somente será reavaliado quando alguma de suas dependências for alterada”.

Portanto, somente se mudarmos uma dependência envolvida o dado computado será retornado. Para convertermos todo o texto que o usuário digitar para letras maiúsculas, precisamos montar a nossa página da seguinte:

HTML:

```
<input v-model="mensagem”>
<p>A mensagem é: {{ mensagemMaiuscula }}</p>
```

JavaScript

```
var vm = new Vue({
  el: '#example',
  data: {
    mensagem: ''
  },
  computed: {
    mensagemMaiuscula: function () {
      return this.mensagem.toUpperCase()
    }
  }
})
```

No exemplo acima, toda vez que o usuário digitar algo no campo, será exibido embaixo o mesmo texto, porém, em caixa alta. Há muitas funções impressionantes do Vue para explorar, esses foram apenas alguns exemplos que um iniciante iria se deparar ao começar a aprender. Além disso, é totalmente possível aprender apenas com a documentação oficial, visto que a mesma é muito completa e em boa parte, traduzida para o português.

3.1.3. Empresas que fazem uso

De acordo com o site oficial do Vue Js (traduzido para o português por meio do próprio site), há várias empresas brasileiras fazendo uso do Vue, como por exemplo O Boticário, Cuponomia, Buser, PagSeguro, etc.

Porém, essas são umas das mais famosas usando apenas o Vue, se formos levar em consideração empresas que fazem uso de bibliotecas relacionadas o número é muito maior. Um exemplo disso, é a Olx, que utiliza o Vuex. Segundo o site oficial desta biblioteca, Vuex é “um **padrão de gerenciamento de estado + biblioteca** para aplicações Vue.js. Ele serve como um store centralizado para todos os componentes em uma aplicação, com regras garantindo que o estado só possa ser mutado de forma previsível”.

Um outro exemplo disso é o iMasters, um site que “congrega uma comunidade segmentada formada por mais de 450 mil desenvolvedores e profissionais de web, com publicação diária de conteúdo” (trecho extraído do site oficial). Este, utiliza de Vue Router, “um projeto de código aberto licenciado pelo MIT” (trecho extraído do projeto no Github), um Router que auxilia no mapeamento de rotas (site oficial do Vue, na página do Vue Router).

Além disso, deve-se levar em consideração empresas que não constam neste site e utilizam Vue, como é o caso da Melhortaxa, “o maior marketplace especializado em crédito imobiliário do Brasil, com uma plataforma online que permite comparar instantaneamente as ofertas das mais importantes instituições financeiras do país.“ (extraído do perfil oficial no Linkedin).

É possível saber que este site utiliza o Vue graças à extensão do Navegador Chrome chamada Wappalyzer, “um utilitário de plataforma cruzada que revela as tecnologias usadas em sites.” (descrição extraída da Chrome Web Store).

3.1.4. Opiniões da comunidade

Pelo fato do Vue ser popular no Brasil, vamos ver as opiniões em português. Começando pelo iMasters (já citado anteriormente), para Reis, quando comparado com React, ambos têm as mesmas características e fazem as mesmas coisas, porém, o Vue é menor e mais rápido.

Além disso, ainda segundo o autor, aprender React nem sempre é uma tarefa fácil. Assim como o Angular, em ambas as tecnologias acaba sendo necessário utilizar de extensões, como por exemplo o JSX, e isso gera um aprendizado maior, pois não necessariamente essas bibliotecas foram feitas para React, gerando um estudo a mais.

Estudar sempre é algo positivo, porém, leva tempo, algo que muitas empresas lutam contra na entrega de projetos, então principalmente para novos integrantes na equipe, geraria um gasto a mais de recurso. Para o autor, “Integrar novos desenvolvedores em um projeto feito com Vue.js é simples, e não exige muito tempo, rapidamente o novo membro da equipe estará produzindo.”

Ainda segundo Reis, Vue não é apenas mais um framework Javascript, e o que diferencia dos concorrentes é seu nascimento. Diferente dos outros, esta ferramenta é da comunidade para a comunidade, ou seja, não foi criada e nem é mantida por uma grande organização e nem mesmo surgiu com a necessidade de atender demandas específicas de uma empresa, basicamente se trata de um projeto pessoal, no qual fez muito sucesso e fez com que o criador possa se dedicar totalmente ao seu desenvolvimento, graças às arrecadações (doações de empresas e afins).

Já na opinião de Frias, “O Vue.js foi criado com base no template do Angular, porém usando algumas das coisas do React, como a componentização e controle do DOM”, “une o melhor dos dois mundos”. Para o autor, o Vue é extremamente leve e possui um *template engine* muito fácil e intuitivo, não havendo dificuldade de aprender caso já tenha algum conhecimento de Angular ou React.

Mas para o autor Vue js também tem suas desvantagens, sendo a principal o fato de ser ainda muito recente entre os grandes frameworks. Além disso, pelo fato de não ter empresas gigantes por trás como Google e Facebook (responsáveis por Angular e React, respectivamente) acaba sendo “menos seguro” para as empresas.

No entanto, Frias reafirma sua opinião sobre este framework: “Será só uma questão de tempo até que o Vue esteja no mesmo patamar de popularidade”.

Mostrando outra opinião positiva sobre Vue, Ferreira alega que começar um projeto com poucos componentes e sem utilizar ferramentas adicionais é um grande atrativo, e assim como em uma das opiniões logo acima, o fato do react utilizar JSX é um ponto a menos para o React em uma comparação de frameworks. Ainda segundo o autor, a documentação mantida pela comunidade “É sem dúvida uma das melhores documentações que já utilizamos”..

No entanto, para o autor, “O lado ruim de ter uma comunidade vibrante é que existem soluções de terceiros para praticamente qualquer problema, por menor ou por mais simples que seja a solução”. Isso ocorre justamente por ter uma comunidade muito ativa, fazendo com que coisas simples sejam desenvolvidas a fim de gerar maior produtividade na criação de sistemas.

Porém, na opinião do autor, as muitas soluções feitas por terceiros “limitavam bastante as possibilidades de uso dos componentes”, visto que as vezes as mesmas traziam ferramentas indesejadas ou não faziam exatamente o que se precisava em um projeto, sendo necessário contornar o problema, o que é muito custoso.

Observando os relatos acima, assim como a comunidade Vue no Brasil, as opiniões são muito parecidas. Boa parte parece gostar ou ter vontade de utilizar, sendo que os problemas relatados estão relacionados à aceitação no mercado e bibliotecas de terceiros. Para encontrar opiniões contrárias foi necessário recorrer a uma opinião estrangeira, escrita em inglês.

Segundo Wolff, a lista de diretivas em Vue são muito pequenas se comparadas ao Angular, ao mesmo tempo em que possui uma complexidade adicional, necessária para usar a ferramenta com eficiência. Para ele, “Isso é ainda mais exacerbado pela flexibilidade que o Vue.js confere às diretivas, que por si só vem com complexidade adicional.”.

Ainda segundo o autor, há duplicidade de códigos em alguns momentos, como por exemplo no uso de componentes. Ao mesmo tempo em que se importa, deve-se informar ao compilador de modelos quais componentes está utilizando. É possível entender melhor com a figura 3, cujo código foi extraído do mesmo artigo.

Figura 3: Exemplo de duplicidade de código em Vue

```
// Import your components as you normally would with ES Modules
import ComponentA from './ComponentA';
import ComponentC from './ComponentC';

export default {
  components: {
    // Register them with the template compiler
    ComponentA,
    ComponentC,
  },
  // Then finally use them in your template
  template: `
        <ComponentA />
    `,
};
```

Por fim, Wolff não se adequa muito bem às regras de manipuladores de eventos do Vue, pelo fato de se utilizar de *strings* para isso. Para o autor, é bom pelo fato do Vue detectar qualquer erro bobo de digitação que possa ocorrer, mas por opinião pessoal, não gosta muito. Na figura 4 podemos ver um exemplo de código que demonstra os eventos personalizados, extraído diretamente do mesmo artigo.

Figura 4: Exemplo de eventos personalizados

```
<button @click="greet">Greet</button>
```

É importante ressaltar que Wolff é um desenvolvedor React e muitas das desvantagens citadas por ele estão relacionadas a preferências na hora de desenvolver, e não necessariamente sobre pontos fracos do Vue ou o que pode ser feito com esta tecnologia.

3.2. Bulma

3.2.1. Principais características

De acordo com a seção *overview* (visão global em português) do site oficial do Bulma, a estrutura deste framework é composta pelas seguintes características:

- Necessário apenas um arquivo CSS para utilizá-lo
- É possível escrever o HTML da maneira que preferir, já que se trata apenas de uma coleção de classes CSS
- Sintaxe de fácil entendimento
- Modularização, ou seja, é possível utilizar apenas o que for necessário
- É um framework responsivo, que utiliza do conceito ***mobile-first***

Antes de falar um pouco sobre o Bulma, é preciso definir o termo citado acima. *Mobile-First* é um conceito, criado por Luke Wroblewski. No blog pessoal do autor, ele destaca o livro que foi escrito a partir do conceito, de mesmo nome, no qual a ideia principal é o desenvolvimento de aplicações começar primeiro pela versão para celular.

Isso porque atualmente o uso de websites pelos dispositivos móveis é muito grande, e é importante que o usuário tenha uma boa experiência de uso, o que em muitos casos não ocorria devido ao fato das aplicações serem focadas em *desktop*, e adaptadas ao *mobile*.

Voltando ao framework, começaremos pela primeira característica. Segundo a seção de início da documentação oficial, é possível, assim como o Vue, desenvolver sem muitas dores de cabeça, apenas referenciando um link no código. A diferença principal é que há 2 formas de fazer, uma via CSS e a outra via HTML, como podemos ver abaixo:

CSS:

```
@import "https://cdn.jsdelivr.net/npm/bulma@0.9.3/css/bulma.min.css";
```

HTML:

```
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bulma@0.9.3/css/bulma.min.css">
```

Além disso, para projetos mais robustos é o ideal adicionar o Bulma via gerenciador de pacotes. Usando NPM, iremos obter a biblioteca SASS do Bulma da seguinte forma:

```
npm install bulma
```

Sabendo dessas informações, surgem duas dúvidas: O que é NPM? O que é SASS?. NPM é uma empresa fundada em 2014, que tem grande participação no ecossistema JavaScript, atuando de modo a dar suporte a milhares de desenvolvedores no mundo, permitindo instalar e publicar pacotes (Site oficial do NPM).

Já com relação ao SASS, segundo seu site oficial, se trata da linguagem de extensão CSS de nível profissional mais madura, estável e poderosa do mundo. Para o W3Schools, cada dia os códigos CSS estão mais complexos e difíceis de se manter, e justamente para isso o SASS serve, permitindo importar, usar variáveis, funções integradas, etc.

Utilizando as informações da seção de Classes CSS do site oficial, podemos concluir que no final, tudo se resume a apenas um arquivo no Bulma, e que o seu HTML não interfere no resultado. É possível estilizar campo a campo, de apenas uma página, assim como definir um padrão central a ser seguido.

Se a intenção for definir um padrão, deve-se colocar as informações dentro do arquivo generic.sass do projeto. Um exemplo prático é caso queiramos que todos os campos de texto tenham altura de 45 pixels. Para isso colocaríamos o seguinte trecho no arquivo citado acima:

```
.input {
        height: 45px;
}
```

Partindo para outra característica muito interessante do Bulma, podemos observar na seção Sintaxe de Modificadores, do site oficial, as classes que

modificam o conteúdo de maneira rápida e prática. O padrão é muito intuitivo, com o prefixo “is”.

Um exemplo simples é modificar a cor de um botão, para a cor azul. Se fossemos usar CSS, teríamos que definir uma classe e utilizar do atributo color, mas com o Bulma, basta fazermos o seguinte:

```
<button class="button is-primary">
  Button
</button>
```

Se por acaso desejar que o botão fique carregando, basta adicionar a classe “is-loading”, e o mesmo vale para caso queira desabilitá-lo: “is-disabled”. Mas o que exatamente significa o “is-primary”?. Isso nos leva a outra característica do bulma: As cores.

É possível utilizar as cores pré-definidas do bulma, assim como fazer as suas próprias. Na seção de Cores da documentação oficial, nos deparamos com 10 cores básicas: *White, Black, Light, Dark, Link, Info, Success, Warning* e *Danger.* Todas são muito intuitivas, como é o caso da última, em um tom de vermelho, para chamar a atenção do usuário para um erro, por exemplo. É possível editar essas cores padrões ou criar outras no arquivo *map*, por meio de variáveis iniciadas com cifrão. *Warning* por exemplo, é representado pela variável $yellow, com valor hsl(48, 100%, 67%). (hsl é um padrão de cores, alternativo ao RGB).

Por fim, é interessante ressaltar a capacidade de responsividade deste framework. O que muitas vezes é uma dor de cabeça para o desenvolvedor, pode ser em alguns casos facilmente contornado com a seção Responsividade da documentação oficial. Utilizando colunas (que veremos no próximo tópico), podemos tornar um trecho de código responsivo utilizando poucas palavras, além de mantê-lo de fácil entendimento e manutenção, como por exemplo: *mobile, desktop, widescreen* e *fullhd.*

E ainda podemos adicionar a palavra *only* (apenas, em português), que permite aplicar modificações apenas na resolução desejada, como *tablet-only, desktop-only, widescreen-only*, ou a palavra *until* (até, em português), obtendo *until-widescreen* e *until-fullhd*. Os nomes são sugestivos, e no primeiro caso aplicaríamos as classes apenas nas resoluções desejadas. Já no segundo caso,

englobaria todas as telas até a mencionada. Exemplos práticos desse assunto serão tratados no próximo tópico, junto com outros conceitos.

3.2.2. Exemplos de seus pontos fortes

Para falar sobre o primeiro ponto forte, é necessário definir um conceito chamado *Flexbox.* Segundo a seção *Basic Concepts of Flexbox* da Mozilla, *Flexbox* “foi projetado tanto como um modelo de layout unidimensional quanto como um método capaz de organizar espacialmente os elementos em uma interface, além de possuir capacidades avançadas de alinhamento”. No Bulma, chamamos a *Flexbox* de Colunas (ou Columns).

Como o próprio nome sugere, conseguimos criar colunas nas páginas web. Segundo a seção *Columns* da documentação oficial do Bulma, podemos facilmente criar nossas colunas utilizando apenas as classes do framework, sendo que temos um elemento pai chamado *columns*, e seus filhos, chamados de *column*. O exemplo abaixo, extraído da mesma página, ilustra bem os conceitos:

```
<div class="columns">
  <div class="column">
    First column
  </div>
  <div class="column">
    Second column
  </div>
  <div class="column">
    Third column
  </div>
  <div class="column">
    Fourth column
  </div>
</div>
```

No exemplo acima, teríamos 4 colunas de tamanhos iguais. Porém, na subseção *Sizes*, descobrimos que é possível determinar o tamanho de cada coluna, basta adicionar “is-N” à classe, onde N é um número no intervalo de 1 a 12. De forma resumida, o comprimento horizontal da tela é dividido em 12 partes iguais, e portanto, podemos assim moldar nosso conteúdo da maneira que for mais conveniente. Caso precisemos de 2 colunas, com a primeira ocupando aproximadamente 75% e a segunda o restante (25%), o código ficaria da seguinte forma:

```
<div class="columns">
  <div class="column is-9">
    Coluna 1, ocupando 75% do comprimento horizontal
  </div>
  <div class="column is-3">
    Coluna 2, ocupando 25% do comprimento horizontal
  </div>
</div>
```

Caso seja necessário haver, por exemplo, um espaço vazio antes do conteúdo, esta mesma seção da documentação pode nos ajudar. Basta adicionar “is-offset-N”, onde N, assim como no conceito de colunas, é um número de 1 a 12. Se o conteúdo deve aparecer apenas após 25% do comprimento inicial horizontal da tela, basta adicionar a classe “is-offset-3”.

No final do tópico 3.2.1 vimos de forma resumida o conceito de responsividade. Na documentação do Bulma, há uma subseção específica para a responsividade das colunas, denominada *responsiveness*. Para facilitar a vida do desenvolvedor, este framework possibilita ferramentas para uma melhor experiência do usuário quando o mesmo estiver usando um celular. Tudo começa adicionando a classe “is-mobile” no *columns* (importante frisar que é no *columns* e não no *column*).

Vamos a um exemplo real: No computador, o usuário deverá visualizar duas colunas lado a lado, ambas com o mesmo tamanho, e no celular cada coluna deva ocupar 100%, com a segunda logo abaixo. O código ficará da seguinte forma:

```
<div class="columns is-mobile is-multiline">
  <div class="column is-6-desktop is-12-mobile">
    Coluna 1, ocupando 100% no celular e 50% no computador
  </div>
  <div class="column is-6-desktop is-12-mobile">
    Coluna 2, ocupando 100% no celular e 50% no computador
  </div>
</div>
```

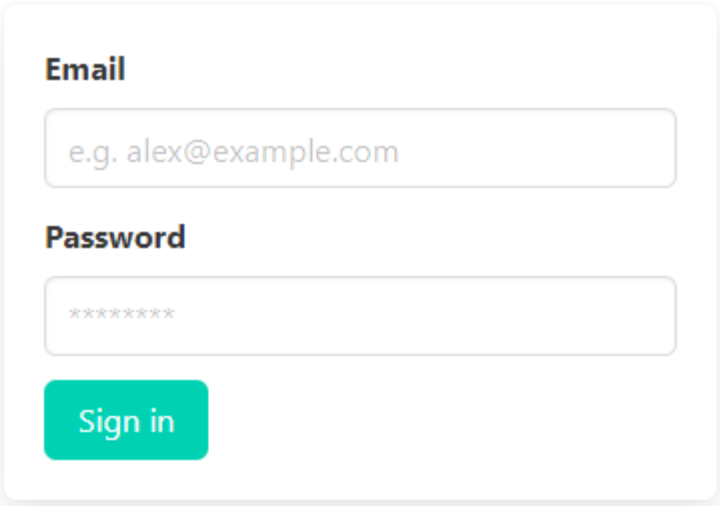
No exemplo acima temos várias novidades. Ainda segundo a seção *responsiveness*, sempre que precisarmos definir o tamanho de uma coluna para determinado tipo de tela, basta identificarmos na própria definição de comprimento. Há 5 tipos de tela diferentes: *mobile* (celular), *tablet*, *desktop* (computador), *widescreen* e *fullhd*.

Em se tratando de elementos, temos uma seção cheia de conteúdo. Uma subseção muito útil, denominada *Box*, nos ensina a criar um container, já com sombra, preenchimento (chamado de *padding* no CSS, é um“espaço” em todas as direções, que permite uma melhor visualização do conteúdo) e um fundo branco de padrão. O mais incrível: basta adicionar uma classe:

```
<div class="box">
  I'm in a box.
</div>
```

Exemplo mais elaborado, extraído diretamente da mesma subseção:

Figura 5: Formulário criado com a classe box - Bulma framework.



Para encerrar essa demonstração de pontos fortes do Bulma, vamos falar sobre a subseção de botões (presente na seção elementos) da documentação oficial do Bulma. Seja para confirmar o envio de mensagens, finalizar uma compra ou preencher um formulário, os botões são muito utilizados nos websites e aplicativos, e é totalmente possível criá-los usando HTML e CSS.

No entanto, o Bulma facilita nossa vida com classes pré-prontas, que estilizam de maneira fácil e rápida os botões. Para utilizar o Bulma nos botões, basta colocar a classe *button*. A partir de um botão simples (branco), podemos adicionar uma cor, como por exemplo preto, utilizando a classe *is-black*. Com isso, além da cor de fundo do botão passar a ser preta, o bulma muda a letra para branco automaticamente. E com relação ao tamanho do botão? Basta informar a classe *is-large* para obter o maior tamanho disponível.

Há diversas demonstrações de cores, tamanhos, atividades que o Bulma possibilita. Mas melhor do que explicar, é exibir o resultado final. Abaixo, é possível ver os exemplos originais, extraídos ainda da subseção de botões da documentação:

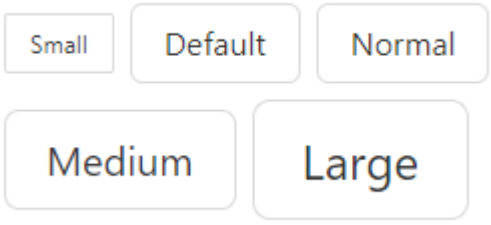
```
<div class="buttons">
  <button class="button is-primary">Primary</button>
  <button class="button is-link">Link</button>
</div>

<div class="buttons">
  <button class="button is-info">Info</button>
  <button class="button is-success">Success</button>
  <button class="button is-warning">Warning</button>
  <button class="button is-danger">Danger</button>
</div>
```

Figura 6: Exemplos de cores dos botões - Bulma framework.


```
<button class="button is-small">Small</button>
<button class="button">Default</button>
<button class="button is-normal">Normal</button>
<button class="button is-medium">Medium</button>
<button class="button is-large">Large</button>
```

figura 7: Exemplos de tamanhos dos botões - Bulma framework.



Estes foram pequenos exemplos do que o Bulma é capaz. Foi feito um breve resumo com as principais ferramentas que um iniciante usaria. Além de diversas outras funcionalidades, há também um aprofundamento de conceitos muito maior do que o demonstrado aqui.

3.2.3. Empresas que fazem uso

Segundo a seção *Expo* do site oficial do Bulma, há diversos sites utilizando seu framework. O primeiro (e talvez mais conhecido) a ser listado é o Signal, no qual o site oficial alega ser um aplicativo de mensagens simples, poderoso e seguro, com criptografia de ponta a ponta, gratuito e sem anúncios.

Com uma consulta simples ao código fonte do site, podemos confirmar o uso, havendo diversas classes como *columns, container* e *section* (as duas últimas não tratamos neste trabalho, mas informações estão disponíveis no site oficial para uso).

Além deste site, ainda segundo a seção *Expo*, o Formal Founder também é feito utilizando Bulma. Segundo seu site oficial, o Formal Founder é uma ferramenta gratuita de pesquisa que verifica a disponibilidade de nomes comerciais, o que pode ser útil para empresários definirem um nome para seus empreendimentos (que não estejam sendo usados e nem sejam patenteados).

Verificando o código fonte, verifica-se a utilização de *column, section, container,* assim como outras classes não vistas neste texto, como *has-text-centered* (que centraliza o texto), *title* (para títulos), *is-bold* (para tornar o texto negrito) e muitos outros.

Outro site já citado anteriormente, a Melhortaxa, também utiliza de Bulma. Classes como *columns,* column, *container, section, footer* (para criar rodapés) e muitas outras são utilizadas. Inclusive, no cabeçalho do site é utilizada a classe *has-background-primary*, que traz a cor de fundo *primary* (um tom azulado):

Figura 8: Uso de classe do framework Bulma na página inicial do site Melhortaxa.

3.2.4. Opiniões da comunidade

O site DevMountain rasga elogios em seu artigo intitulado “Bulma CSS: o que é e por que é uma estrutura que os desenvolvedores adoram” (traduzido do inglês). Na opinião do site, este framework é fácil de aprender, além de ser responsivo, usar apenas CSS, ser modular, bem documentado, compatível com inúmeros navegadores e está em constante evolução.

O blog ainda ressalta: Não são todas as características do framework que irão lhe agradar, mas como se trata de uma ferramenta de código aberto, é totalmente possível fazer suas próprias alterações e deixar da maneira que achar melhor, sem gastar com licenças.

Para Khan (2019), Bulma foi um divisor de águas. Em meio a frameworks já muito utilizados no mercado e o CSS puro, ele o encontrou e citou as seguintes vantagens: sem estilo padrão, poderosa Flexbox Grid, tamanho pequeno, reutilizável e sem Javascript. Com relação à última vantagem citada, é possível determinar que é uma vantagem e uma desvantagem, depende do ponto de vista.

Ao mesmo tempo em que o fato do Bulma ser focado no CSS e não utilizar JavaScript é algo bom, ele também limita o uso e manipulação de elementos. Se por acaso fosse necessário uma classe ser aplicada apenas em uma determinada circunstância (valor de uma variável, por exemplo), seria necessário utilizar JavaScript, o que torna o código mais custoso (em tempo de desenvolvimento principalmente). Porém, alguns desses problemas são solucionados com o uso de Vue e Buefy (que será visto no próximo tópico).

Para Pattakos (2021), em uma lista dos 9 melhores frameworks de 2021, o Bulma ocupa a terceira colocação. Na opinião do autor, este framework é uma boa alternativa ao *Bootstrap*, tendo como principais características positivas o design estético bonito e moderno, ser amigável ao desenvolvedor, fácil de personalizar e não possuir JavaScript.

No entanto, Pattakos enxerga um possível defeito na primeira característica positiva: sites parecidos. O que atualmente ocorre com o *Bootstrap* é ter muitos sites com o mesmo design, devido ao seu grande uso pelos desenvolvedores. Na opinião do autor, isso pode ocorrer com o Bulma também, quando estiver mais popular.

Por fim, o autor acha o Bulma menos completo comparado ao seu grande competidor, o *Bootstrap*. Ainda faltam funcionalidades que visam acessibilidade e recursos de nível empresarial.

3.3. Buefy

3.3.1. Principais características

De acordo com a seção *Documentation* do site oficial desta ferramenta, o Buefy é uma biblioteca de componentes de interface ao usuário feita baseada em Vue e Bulma, tendo como objetivo ser simples e leve.

Começando pela sua instalação, é tão simples quanto o Vue, podendo ser instalado via NPM, com o comando “npm install buefy”, sendo importado no projeto da seguinte forma:

```
import Vue from 'vue'
import Buefy from 'buefy'

Vue.use(Buefy)
```

Observando o seu funcionamento, vemos que os componentes são chamados a partir de tags, em alguns casos possuem os mesmos nomes dos elementos originais do HTML, com a diferença de possuir um “b-” como prefixo. Mas afinal, de que componentes estamos falando?

Boa parte dos elementos que usamos em aplicativos para a web o Buefy possui, como por exemplo botões, caixas de texto, menus de navegação, modais, tabelas, slides, caixas de seleção e muito mais.

Um dos primeiros componentes a serem listados na documentação, e um dos mais fáceis de ser assimilado, é o botão. Na subseção do mesmo, vemos a forma de declará-lo, assim como suas classes para estilização. Muitas dessas classes são do Bulma, que vimos no tópico anterior, e portanto, não vamos nos aprofundar. Na linha abaixo podemos ver um exemplo simples baseado na documentação:

```
<b-button type="is-danger is-light">Botao vermelho</b-button>
<b-button disabled>Desabilitado</b-button>
```

No exemplo acima, serão exibidos dois botões: o primeiro com fundo vermelho claro e letra vermelha escuro, contendo o texto “Botão vermelho”; já o segundo, é um botão simples, com fundo branco e letra escura, contendo o texto “desabilitado”, além de, claro, estar desabilitado para cliques do usuário.

Já se quisermos criar um campo de texto, além de criarmos o campo em si, precisaremos de um componente extra englobando-o, chamado *field*. Não que não seja possível criar o campo sem ele, se trata de um componente “para adicionar funcionalidade aos controles e para anexar / agrupar componentes e elementos juntos” (Seção Field, Buefy).

Além disso, é possível definir um identificador do componente a partir do *field*, que chamamos de *label*. Sabendo dessas informações, vamos supor que precisamos criar um campo de texto para que o usuário informe uma senha de no mínimo 8 caracteres e no máximo 10. Baseado na subseção *Input* da documentação oficial, ficaria da seguinte forma:

```
<b-field label="Senha">
    <b-input
        minlength="10"
        maxlength="100"
        password-reveal
        size="is-large"
    ></b-input>
</b-field>
```

Obs: *password-reveal* se trata do botão, em formato de olho, para visualizar a senha digitada e *size* o tamanho do campo, sendo o informado o maior. Existem 4 tamanhos disponíveis: is-small (30 pixels de altura), default (não precisa ser informado, 40 pixels de altura), is-medium (50 pixels de altura) e is-large (60 pixels de altura).

Podemos observar que as propriedades são definidas de forma simples, sem a necessidade do uso de JavaScript puro e CSS. Além disso, em cada componente da documentação, no fim da página, temos as principais propriedades, eventos, métodos e mais, tudo muito bem explicado. A figura 9 abaixo é um retrato tirado da página de *Input*, mencionada acima:

Figura 9: Documentação facilitada dos componentes do Buefy

API

Properties | Events | Methods

Name	Description	Type	Values	Default
`v-model`	Binding value	String, Number	—	—
`lazy`	Makes the binding lazy. Note: `v-model.lazy` won't work	Boolean	—	`false`
`type`	Input type, like native	String	Any native input type, and `textarea`	`text`
`size`	Vertical size of input, optional	String	`is-small` , `is-medium` , `is-large`	—
`expanded`	Makes input full width when inside a grouped or addon field	Boolean	—	`false`
`password-reveal`	Add the reveal password functionality	Boolean	—	`false`

3.3.2. Exemplos de seus pontos fortes

Vimos no tópico acima a utilização de botões, as facilidades que trazem seu fácil aprendizado. Porém, se é uma biblioteca baseada no framework Bulma, geralmente utilizada nos projetos em conjunto com ele, por qual razão usaríamos o *b-button* se basta informar a classe *button* em um botão normal?

A razão é simples, e foi citada ao longo do texto: O Bulma não utiliza JavaScript, e portanto, não consegue armazenar valores dos campos. O Buefy, por ser baseado no Bulma e no Vue, trabalha com maestria em conjunto com ambos. No caso do botão, por exemplo, sempre que o usuário clicar nele, podemos chamar uma função do Vue, através do evento @click.

Se quisermos exibir ao usuário a quantidade de cliques em um botão, podemos fazer, baseado na subseção *button*, da seguinte forma:

```
<template>
    <section>
        <b-button @click="addOne">{{ contador }}</b-button>
    </section>
</template>

<script>
    export default {
       data(): {
            return {
               contador: 0
            }
       }
        methods: {
            addOne() {
                this.contador += 1
            }
        }
    }
</script>
```

Na figura 9 apresentada no tópico anterior, vemos que há uma aba de eventos, é nela que verificamos os eventos do componente. Se observarmos a aba de eventos do componente *input*, veremos que há 6 eventos. Gostaria de destacar 3 para exemplificar:

- ***Input:*** É chamado um evento do Vue quando o valor do campo é alterado (ao digitar um caractere).
- ***Focus:*** É chamado um evento do Vue quando o campo está em foco (útil para disparar ações, como por exemplo carregar uma lista de informações).
- ***Blur*****:** É chamado um evento do Vue quando o campo perde o foco (útil para efetuar uma busca quando o usuário clica fora do campo, por exemplo).

Para terminar de exemplificar o Buefy, vamos tratar de menu de navegação, já que praticamente todos os sites possuem. Em vez de montar um do zero, tendo que criar inúmeras div's e se preocupar com CSS, basta utilizarmos o modelo existente desta biblioteca.

De forma resumida, segundo a subseção *Navbar* da documentação oficial, um menu de navegação é composto por 3 elementos principais: *Brand* (utilizado para mostrar o logo), início (links úteis ao lado do logo) e fim (geralmente utilizado para informações como login, cadastro e sobre). Na figura 10 abaixo, podemos ver um exemplo extraído diretamente da documentação:

Figura 10: Exemplo de menu de navegação utilizando o componente do Buefy.

Como podemos ver acima, há o logo do Buefy (*brand*), links para *home, Documentation* e *info* (*start*) e *Sign up* e *Log in* (*end*). No código abaixo, temos um exemplo de aplicação prática, não tão detalhado como o da documentação oficial, mas com uma base para melhor entendimento.

```
<template>
    <b-navbar>
        <template #brand>
            <b-navbar-item>
                <img src="link">
            </b-navbar-item>
        </template>
        <template #start>
```

```
            <b-navbar-item href="#">
                Início
            </b-navbar-item>
        </template>

        <template #end>
            <b-navbar-item tag="div">
                <a class="button is-primary">
                    <strong>Entrar</strong>
                </a>
            </b-navbar-item>
        </template>
    </b-navbar>
</template>
```

Explicando um pouco do código acima, cada “seção” do menu de navegação é apresentada a partir de um template, sendo que em cada seção podemos ter diversos itens, identificados pela tag *b-navbar-item*.

3.3.3. Opiniões da comunidade

Seguindo a lógica das outras tecnologias, aqui deveriam ser apresentadas as empresas que fazem uso de Buefy. No entanto, por se tratar de uma biblioteca, é muito difícil obter dados confiáveis acerca disso, visto que um *b-button* por exemplo se apresenta como um *button* de HTML no código fonte, e suas classes são do Bulma. Há uma página específica do Buefy contendo sites que o utilizam, porém, grande parte não possui grande relevância.

Apenas uma pequena parte dos projetos utilizando Buefy aparecem lá, pois é preciso que o utilizador informe que usa. Isso não significa que essa biblioteca seja pouco utilizada, muito pelo contrário, já que segundo o gerenciador de pacotes NPM, do dia 02/11/2021 até o dia 08/11/2021 ela foi baixada 56.577 vezes, como consta na figura 11 abaixo:

Figura 11: Captura de tela no site NPM - Quantidade de downloads da biblioteca Buefy.

Para Samuel (2020), a biblioteca usada por mais de 200 mil desenvolvedores em todo o mundo torna o desenvolvimento de websites simples, além de trazer economia de tempo e manter o código sempre limpo e moderno.

4. VANTAGENS NA UTILIZAÇÃO DE FRAMEWORKS E BIBLIOTECAS

Durante todo este trabalho foram listadas diversas vantagens no uso de frameworks, em meio a explicações técnicas de suas características. Devido a isso, neste tópico serão abordadas qualidades mais generalistas das tecnologias, que se aplicam a todas as abordadas neste trabalho.

4.1. Comunidade dedicada

4.1.1. Resolução de problemas

De acordo com informações obtidas no repositório do React no Github (2021), esta biblioteca possui 777 erros corrigidos. O primeiro é datado em 28 de junho de 2013, solicitado por Newman (2013). Atualmente há 120 solicitações para que o React verifique erros (Github, 2021).

Já na biblioteca Buefy, de acordo com seu repositório oficial no Github (2021) há 142 *bugs* corrigidos, sendo o primeiro datado em 10 de abril de 2017, no qual Heckenberg (2017) solicitou a correção de um erro na exibição de um ícone. Atualmente, há 5 solicitações para que a equipe do Buefy verifique erros relatados (Github, 2021), mostrando a diferença enorme do tamanho da comunidade, quando comparado ao React.

Quando verificamos a quantidade de erros corrigidos pela equipe do Angular percebemos o quão maior é sua comunidade, pois de acordo com o repositório do Angular no Github (2021), já foram corrigidas 6293 solicitações desde o dia 3 de fevereiro de 2015, onde Berchet (2015) solicitou a revisão de um detector de mudanças no código do projeto. Atualmente, há 587 pedidos para verificação de possíveis erros existentes no framework (Github, 2021).

Já no repositório oficial do Vue no Github (2021) foram corrigidas 336 solicitações de *bugs* desde o dia 21 de outubro de 2013, quando You (2013), o próprio criador do Vue, verificou que algumas mudanças eram necessárias para valores aninhados. Segundo o Github (2021), atualmente há 42 solicitações para verificação de possíveis erros no Vue.

Por fim, com o menor dos números dentre os avaliados temos o Bulma, no qual segundo o repositório oficial no Github (2021), 38 erros catalogados já foram corrigidos desde 11 de julho de 2016, quando Maher (2016) solicitou alterações no

layout de colunas. Atualmente, há 4 solicitações para verificação de possíveis erros existentes neste framework CSS (Github, 2021).

4.1.2. Criação de novas funcionalidades

De acordo com informações obtidas no repositório oficial do React no Github (2021), esta biblioteca possui 270 recursos novos solicitados pela plataforma, sendo o primeiro em 9 de novembro de 2013, por Plievone, quando o mesmo solicitou a Especificação de eventos de ponteiro de suporte (Plievone, 2013). Atualmente, há 50 solicitações de novos recursos em aberto (Github, 2021).

Já na biblioteca Buefy, vemos que mais recursos novos solicitados foram atendidos e implementados pela equipe, pois segundo o seu repositório no Github (2021), 409 ocorreram, sendo o primeiro datado em 10 de abril de 2017, quando Molenaar (2017) solicitou que fosse disponibilizado o JavaScript compilado do projeto. Atualmente, há 44 solicitações de novos recursos em aberto (Github, 2021).

No repositório oficial do Angular no Github (2021), o número de solicitações para novos recursos no framework implementadas é muito alto: 3075. A primeira ocorreu em 16 de junho de 2015, quando Kirov (2015) solicitou uma melhoria no suporte a variáveis CSS. Segundo o Github (2021), atualmente, há incríveis 745 solicitações de novos recursos para o framework em aberto.

Já quando verificamos o repositório oficial do Vue no Github (2021), a quantidade de implementações em solicitações para novos recursos é de 292, sendo a primeira em 9 de setembro de 2016, onde Sergii (2016) solicitou que fosse permitido que a renderização do lado do servidor renderize tags de script embutidas. Há 65 solicitações de novos recursos para o Vue atualmente (Github, 2021).

Por fim, para o framework CSS Bulma, segundo o repositório oficial no Github, 84 novos recursos foram solicitados e realizados, sendo que o primeiro é datado em 28 de janeiro de 2016, quando Kadlot (2016) pediu para que fosse possível utilizar uma quantidade de colunas diferentes, mesmo que em *breakpoints* diferentes. De acordo com o Github (2021), atualmente há 14 solicitações de novos recursos para este framework, sendo a mais antiga ainda de janeiro de 2016 e a mais recente de agosto de 2018.

4.2. Documentação própria

Praticamente toda a referência técnica utilizada neste trabalho foi obtida a partir dos respectivos sites oficiais, de forma gratuita. Este é um grande ponto positivo, já que compreendendo aspectos básicos da tecnologia é possível avançar nos estudos sem a obrigatoriedade de fazer cursos, o que gera menor custo tanto ao desenvolvedor quanto às empresas.

Começando pelo Vue, de acordo com o Guia do site oficial, há 9 tópicos de estudo, englobando mais de 40 subtópicos (Vue Guide). Além da enorme quantidade de material escrito, o site oficial recomenda 2 plataformas para ensino em vídeo do framework: Vue School e Vue Mastery.

Já no Bulma, o número de tópicos é um pouco menor: Segundo a documentação oficial deste framework, são 9. Já com relação à estudos em vídeo, há apenas aplicações mais práticas, como por exemplo a criação de um blog responsivo, dividida em 4 aulas, ou a criação de uma apresentação de slides, dividida também em 4 aulas.

Com a biblioteca Buefy, de acordo com sua documentação oficial, temos 3 grandes classificadores para o conteúdo: Instalação, Layout e Componentes. O primeiro possui 3 tópicos; o segundo apenas 1; e o terceiro possui mais de 20, dividido em diversos subtópicos. São neles que são listados os botões, menus, campos de textos e etc.

5. DESVANTAGENS NA UTILIZAÇÃO DE FRAMEWORKS E BIBLIOTECAS

Durante toda essa pesquisa foram listadas as desvantagens existentes em cada framework, porém, este tópico tem como função destacar aspectos generalistas, a fim de mostrar efeitos negativos no uso de frameworks em geral.

5.1. Desvantagens funcionais

5.1.1. Comunidade pequena ou pouco ativa

Quando se faz o uso de uma tecnologia de código aberto, isso dá ao desenvolvedor a possibilidade de entrar em contato com a equipe que desenvolve as ferramentas, propondo correção de erros ou possíveis melhorias. Algumas tecnologias utilizam o Github como canal para esses contatos.

No entanto, dependendo da popularidade do framework, o retorno do contato pode ser demorado, assim como as correções dos problemas. Um exemplo é uma solicitação de correção de erro feita por Gotlieb (2020) no Github da biblioteca Buefy, onde o mesmo solicitou a correção de um erro, no qual conforme um campo do tipo data era preenchido e em seguida apagado, a *label* que descrevia a data logo abaixo não era atualizada para um valor vazio.

Gotlieb realizou a solicitação no dia 29 de janeiro de 2020, e sua solicitação foi catalogada no mesmo dia. No entanto, a correção para seu problema só foi efetuada em 27 de março do mesmo ano, e disponibilizada em 29 de março. Ou seja, exatos 2 meses de espera.

Já com relação ao framework Bulma, Nihlén (2020) solicitou que uma correção fosse feita no repositório oficial no Github. Tratava-se de um problema ao utilizar uma classe para desabilitar botões, que o deixavam com letras ilegíveis. O problema foi catalogado no mesmo dia em que o autor solicitou correção, 27 de abril de 2020, e foi corrigido no dia 25 de junho, ou seja, uma espera de quase 2 meses.

As demoras relatadas acima podem ter fatores diferentes para ocorrer. Mas quando pegamos um framework mais popularizado, como é o caso do Angular, vemos que as solicitações de melhorias e erros costumam ser mais rápidas. Um exemplo é uma requisição feita por Piotrowicz (2021), na qual certos elementos se assemelhavam a botões, mas não o eram de fato. Por se tratar de um

desenvolvedor que contribui com o projeto, ele mesmo alterou e solicitou que as intervenções fossem aceitas.

Do começo do processo (26 de setembro) até a revisão (feita por outro desenvolvedor), aprovação e disponibilização da correção (6 de outubro) foram 10 dias, um prazo muito menor que os relatados acima, para Buefy e Bulma.

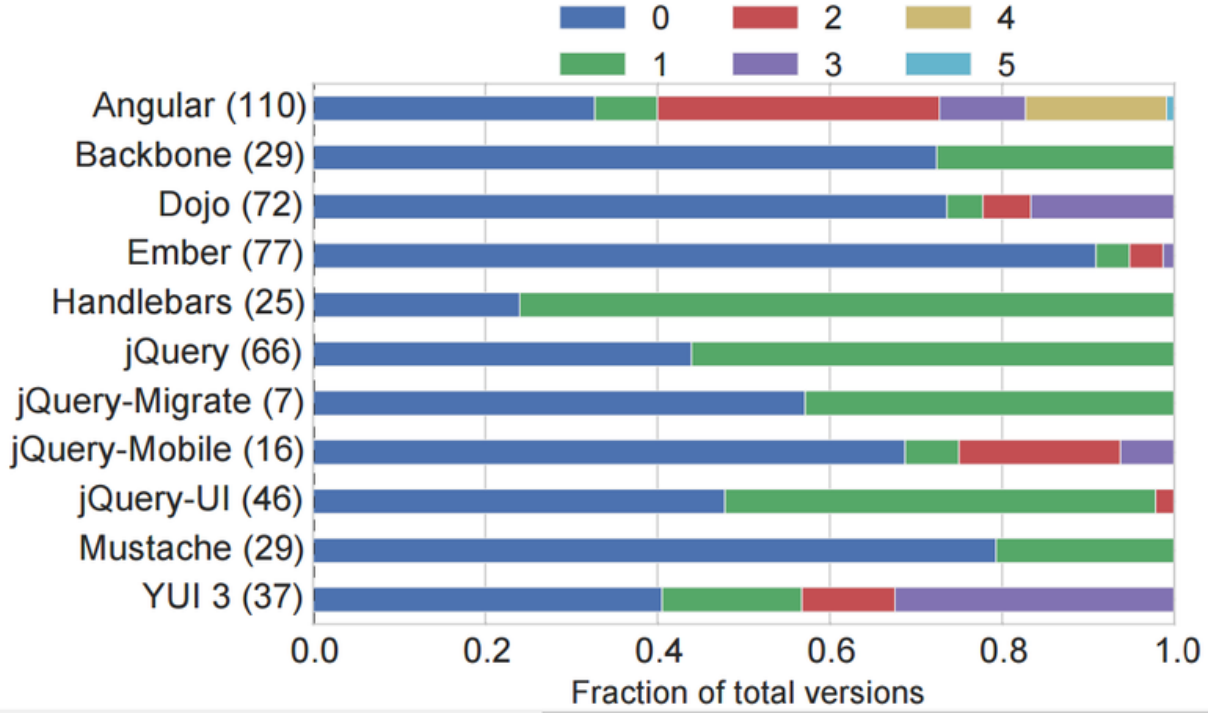
5.1.2. Problemas estruturais na ferramenta

Segundo o site TecMundo (2021), duas vulnerabilidades foram encontradas no plugin “Gutenberg Template Library & Redux Framework”, utilizado por mais de 1 milhão de sites construídos a partir do Wordpress. Essa vulnerabilidade permite obter informações confidenciais sobre o site, além do controle de extensões.

Ainda falando sobre Wordpress, uma matéria do Tecmundo (2020) revelou que milhares de sites foram hackeados devido a falhas em plugins. Ainda segundo a matéria, essas falhas possibilitaram o redirecionamento para páginas fraudulentas, que visam roubar dados, fazer downloads falsos, sorteios falsos e etc.

Não falamos durante este trabalho a respeito de plugins, então trazendo para bibliotecas, Tung (2017) afirma que em uma pesquisa feita pela Northeastern University, de 133 mil sites avaliados cerca de 37% possuía ao menos uma biblioteca vulnerável. O autor, com base nos estudos da mesma universidade, publicou a figura 12 abaixo, onde demonstra a “fração das versões da biblioteca JavaScript com vulnerabilidades conhecidas distintas, cada uma representada por cores, do total de versões da biblioteca entre colchetes.”:

Figura 12: Fração das versões da biblioteca JavaScript com vulnerabilidades conhecidas.



5.2. Desvantagens não funcionais

5.2.1. Risco da tecnologia ser descontinuada

Segundo Pedro (2021), o Windows Phone nasceu em 2010, entrando como concorrente com o IOS e o Android, tendo evoluído, conquistado uma pequena fatia de mercado, até que por diversos fatores, chegou ao seu fim. Todo o tempo investido que empresas de desenvolvimento fizeram para publicar aplicativos na loja oficial agora já não possuem uso.

Estamos falando de uma empresa gigante como a Microsoft, que desenvolveu um sistema operacional que não deu certo. Dadas as devidas proporções (já que um sistema operacional é muito diferente de um framework), o mesmo pode ocorrer com empresas que fazem uso de ferramentas de terceiros.

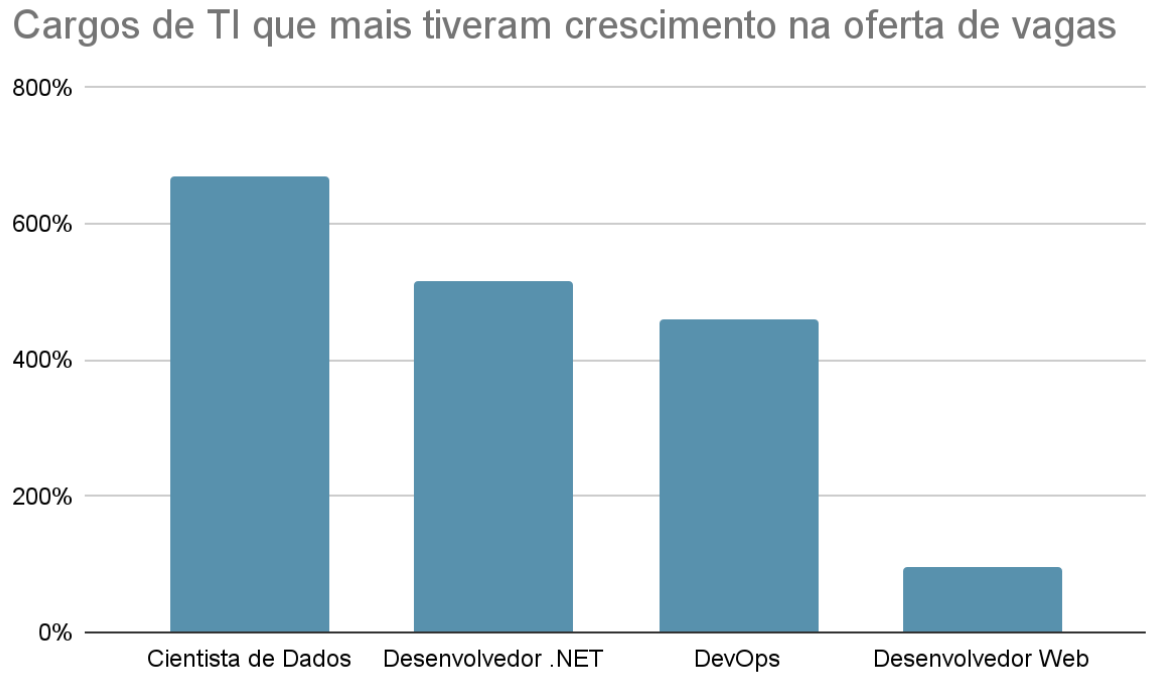
Trazendo um exemplo mais próximo que pode ocorrer, podemos citar a biblioteca Moment Js, que segundo seu repositório oficial no Github, se trata de “
Uma biblioteca de datas JavaScript para analisar, validar, manipular e formatar datas”.

Segundo a seção *Project Status* da documentação oficial do Moment, atualmente esta biblioteca está em desuso. A principal razão é a evolução da web em geral, já que a ferramenta foi criada em 2011, e de lá pra cá seu núcleo e idéia central se mantiveram. Mudanças poderiam ser feitas a fim de modernizar a biblioteca, porém, de acordo com a equipe, seria necessário basicamente refazer um novo projeto.

Ainda segundo a seção, a biblioteca atingiu uma incrível marca de mais de 12 milhões de downloads semanais em setembro de 2020, e a equipe se sente com o dever cumprido, pela ferramenta ter sido usada por diversos sites ao redor do mundo. Por fim, é recomendada a não utilização desta biblioteca, pelo próprio site oficial.

5.2.2. Falta de mão de obra qualificada

Sempre há o risco de não encontrar profissionais adequados para ocupar determinados cargos, porém, parece que ao longo dos últimos anos a procura por profissionais de TI aumentou consideravelmente, é o que diz Sena (2021). Segundo a matéria, de 2021 para 2021 houve aumento de cerca de 671% nas vagas para cientista de dados, 517% para desenvolvedor .NET, 460% para devOps e 97% para desenvolvedor web. O gráfico abaixo ilustra esta informação



Fonte: Exame

Mesmo não se tratando de dados de falta de mão de obra para frameworks específicos, podemos pensar que a situação não deve ser muito diferente, por se tratar da mesma área. Se para o framework .NET, lançado em 2002, a procura aumentou 460%, qual o grau de dificuldade para encontrar um profissional que entenda de Vue, Bulma e Buefy?

É possível neste caso contratar um colaborador que entenda de tecnologias semelhantes, como por exemplo, React ou Angular, mas isso nos leva ao tópico seguinte: Quanto tempo demora para o profissional se adaptar?

Mas antes de partir para o próximo tópico, um outro ponto chama a atenção: O valor do profissional. Um desenvolvedor Sênior no Brasil tem média salarial de 11 mil reais por mês (Glassdoor, 2021). Já nos Estados Unidos, um desenvolvedor

sênior tem média salarial de 117 mil dólares por ano, o que equivale a 9,75 mil dólares ao mês (Glassdoor, 2021).

No entanto, conforme convertemos o câmbio, há uma disparidade muito grande. De acordo com o Ipeadata, no dia 12/11/2021, 1 dólar valia R$ 5,4193. Ou seja, em uma conversão direta, um desenvolvedor sênior nos EUA ganha quase 53 mil reais, o que equivale a quase 5 vezes o salário da mesma função no Brasil. Com a escassez de profissionais no país e o salário mais alto no exterior, ter um profissional qualificado em TI poderá custar cada vez mais caro.

5.2.3. Curva de aprendizagem

Este é um ponto muito interessante a se levar em consideração na criação ou modificação de um projeto, já que colaboradores vêm e vão, e pode ser que os funcionários precisem de um tempo de adaptação às novas tecnologias. No entanto, vamos levar em consideração que uma empresa deseja um estagiário, e que o mesmo não possui experiência prévia, precisando também de um aprofundamento nos conceitos básicos.

Antes de partir diretamente para o framework Vue, o ideal é que se tenha uma boa base em HTML, CSS e JavaScript. Levando em consideração essas tecnologias, podemos usar como base de ensino os cursos da Fundação Bradesco. A plataforma possui cursos gratuitos dessas 3 ferramentas em sua Escola Virtual, na qual há 2 cursos de HTML, 2 de CSS e 1 de JavaScript. Os dois primeiros totalizam 17 horas, os seguintes 50 e o último 20.

Ou seja, caso o novo colaborador precise reforçar conceitos seguindo a didática do Bradesco, levará até 87 horas. Obviamente os valores são os sugeridos, e os alunos podem terminar antes do prazo. Com todo o conteúdo básico estudado, podemos avançar para o estudo do Vue. Em um curso deste framework criado por Schwarzmüller et al (2021) na plataforma de cursos online Udemy, a carga horária apenas das aulas gravadas é de 43 horas, sem contar os projetos e lições.

Mesmo com os dados informados acima, um estagiário (ou um desenvolvedor em qualquer nível) pode criar tarefas enquanto aprende os conceitos, portanto, este tempo todo demandado não seria o tempo que a empresa ficaria sem o colaborador. Porém, podemos ver que o aprendizado de uma nova tecnologia leva tempo, principalmente para assimilação do conteúdo.

Ainda na plataforma Udemy, em um curso criado por Fraga (2021), intitulado “React Js do zero ao avançado na prática", a carga horária apenas das aulas em vídeo totalizam 22,5 horas. Já em um curso também da Udemy, contendo Angular e Typescript, criado por Troquatte (2021), é necessário assistir 15,5 horas de aulas gravadas.

Todos os cursos, por mais que afirmem ter conteúdo do básico ao avançado, são para melhor entendimento do desenvolvedor. O que realmente trará expertise no

assunto é a prática, e portanto, o tempo para assimilação do conteúdo pode ser um pouco maior.

A documentação oficial do Vue possui uma seção específica para tratar de Curva de Aprendizado. Segundo esta página, em uma comparação entre Angular, React e Vue, o primeiro é o mais complexo a se aprender, devido ao fato de ser projetado para construir apenas aplicações grandes e complexas. Já o segundo, antes de começar a compreendê-lo, é necessário ter conhecimento de JSX e ES 2015 (ou superior). Por fim, o último parece ter a melhor curva de aprendizado, já que basta ter noções de HTML, CSS e Javascript.

6. CONCLUSÃO

A presente pesquisa trouxe os principais conceitos acerca de Vue, Bulma e Buefy, 3 tecnologias modernas para o desenvolvimento de aplicações Web, além de citar e contextualizar outras semelhantes. Durante todo o trabalho foram citadas vantagens e desvantagens no uso de cada tecnologia, tendo no final uma visão mais generalista.

Para uma melhor elucidação dos conceitos, foram apresentados conceitos básicos no desenvolvimento de sistemas para a internet, como linguagens de programação, marcação e suas principais siglas, para que o leitor consiga entender melhor a origem dos conceitos posteriormente apresentados.

Tendo sido apresentados os principais conceitos, foi então apresentado um contexto geral de cada uma das tecnologias, citando suas principais características, seus pontos fortes, empresas que fazem uso e algumas opiniões de usuários na internet. Nesta parte, muitas outras tecnologias foram citadas, como por exemplo React e Angular.

Avançando um pouco mais, são observadas sucintamente, de forma generalista, as vantagens no uso de frameworks e bibliotecas. A razão pela qual o tópico foi curto é o fato de que ao longo de toda a apresentação das tecnologias as vantagens foram citadas de forma individual. Sendo assim, aspectos que podem ser observados em todas as tecnologias foram abordados, com exemplos de seu funcionamento.

Foi possível observar de forma clara a grande diferença da quantidade de usuários usando determinada tecnologia, quando comparamos Angular com Bulma, por exemplo. Enquanto o primeiro tem milhares de solicitações de novos recursos, o segundo está na casa das centenas. É interessante observar que mesmo parecendo tecnologias estáveis, há muito ainda a ser construído e aprimorado, tendo milhares de cérebros pensando juntos em melhorias.

Outro fato muito interessante sobre os frameworks são suas documentações muito completas e esclarecedoras. Sabendo ler em inglês, o desenvolvedor tem acesso a muito conteúdo de forma gratuita. Mas como nem todos aprendem bem apenas lendo, sempre há muito conteúdo (pago e gratuito) na internet, sendo muito maior para React e Angular. Para Vue, temos uma quantidade reduzida de conteúdo

em português e para Buefy e Bulma é muito difícil encontrar na nossa língua materna.

Já no tópico das desvantagens, vemos que mesmo aquilo que nos ajuda no dia a dia possui seus contras, sendo que um deles é possível ver até mesmo no tópico de vantagens, quando abordamos a comunidade. Bulma e Buefy têm comunidades menores quando comparadas a Angular e React, e também por isso, resoluções de problemas demoram mais a ocorrer. Outra desvantagem apresentada é o risco, tanto para o desenvolvedor quanto para as empresas, da tecnologia ser descontinuada. Se ocorreu com o sistema mobile da Microsoft e com a grande biblioteca Moment Js, pode ocorrer com qualquer outra. Portanto, usar os frameworks e bibliotecas com sabedoria é o ponto, fazendo os estudos necessários de benefícios e riscos para tomar a decisão mais acertada

Por fim, a curva de aprendizagem é uma desvantagem que ocorre com muitas coisas no dia a dia, já que o mundo está em frequente mudança de paradigmas e tecnologias. O mesmo vale para a falta de mão de obra, que ocorre não somente de desenvolvedores especializados em determinado framework, mas para muitas outras áreas até mesmo fora do TI. A diferença é o fato de que quanto mais específico e restrito for o framework, mais difícil pode ser fazer a reposição da mão de obra. Portanto, outro ponto a ser levado em consideração na escolha das tecnologias.

Portanto, conclui-se que o uso de frameworks e bibliotecas trazem grandes vantagens na hora de desenvolver softwares, mas também são necessários cuidados, principalmente no momento da adoção de uma nova tecnologia, pois as desvantagens podem gerar atrasos em entregas, aumento de custos e até mesmo interrupção do projeto.

7. REFERÊNCIAS

About NPM. **NPMJS**. About. Disponível em: <https://www.npmjs.com/about>. Acesso em: 31 de out. 2021.

ABOUT W3C. **W3C org.** Consortium. Disponível em: <https://www.w3.org/Consortium/>. Acesso em: 03 de out. 2021.

Angular, Angular. **Github**. Issues, type: bug/fix. Disponível em: <https://github.com/angular/angular/labels/type%3A%20bug%2Ffix>. Acesso em: 21 de nov. 2021.

Angular, Angular. **Github**. Issues, feature. Disponível em: <https://github.com/angular/angular/labels/feature>. Acesso em: 21 de nov. 2021.

Basic Concepts of Flexbox. **Mozilla org**. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox>. Acesso em: 11 de nov. 2021.

BERCHET, Victor. Support UTF in Angular expressions. **Github**. Angular, Issues. Disponível em: <https://github.com/angular/angular/issues/517>. Acesso em: 21 de nov. 2021.

Buefy. **Github**. Disponível em: <https://github.com/buefy/buefy>. Acesso em: 03 de out. 2021.

Buefy, Buefy. **Github**. Issues, bug. Disponível em: <https://github.com/buefy/buefy/issues?page=1&q=label%3Abug+is%3Aclosed>. Acesso em: 21 de nov. 2021.

Buefy, Buefy. **Github.** Issues, feature request. Disponível em: <https://github.com/buefy/buefy/issues?q=is%3Aissue+label%3A"feature+request"+is%3Aclosed>. Acesso em: 21 de nov. 2021.

Buefy. **NPMJS**. Packages. Disponível em: <https://www.npmjs.com/package/buefy>. Acesso em: 16 de nov. 2021.

Bulma, jgthms. **Github**. Issues, bug. Disponível em: <https://github.com/jgthms/bulma/issues?page=1&q=is%3Aissue+label%3Abug+is%3Aclosed>. Acesso em: 21 de nov. 2021.

Bulma, jgthms. **Github**. Issues, feature. Disponível em: <https://github.com/jgthms/bulma/issues?q=is%3Aopen+is%3Aissue+label%3Afeature>. Acesso em: 21 de nov. 2021.

Bulma: the modern CSS framework that just works. **Bulma io.** Disponível em: <https://bulma.io>. Acesso em: 03 de out. 2021.

Bulma CSS: What It Is and Why It's a Framework That Developers Love. **DevMountain**. Disponível em: <https://blog.devmountain.com/why-bulma-css-could-be-your-new-favorite-framework/>. Acesso em 11 de nov. 2021.

Button. **Buefy**. Disponível em: <https://buefy.org/documentation/button>. Acesso em: 15 de nov. 2021.

CALDEIRA, Diego. Aprenda Diretivas em 7 Exemplos Práticos— Diretivas Vue.JS 2.0. **Medium**. 17 de abr. 2019. Disponível em: <https://medium.com/vue-js-o-manual-definitivo/aprenda-diretivas-em-7-exemplos-práticos-diretivas-vue-js-2-0-a1688bb1839e>. Acesso em: 10 de out. 2021.

Colors. **Bulma**. Disponível em: <https://bulma.io/documentation/overview/colors/>. Acesso em: 31 de out. 2021.

Columns. **Bulma**. Disponível em: <https://bulma.io/documentation/columns/>. Acesso em: 11 de nov. 2021.

CSS Classes. **Bulma**. Disponível em: <https://bulma.io/documentation/overview/classes/>. Acesso em: 31 de out. 2021.

CSS with superpowers. **SASS LANG**. Disponível em: <https://sass-lang.com>. Acesso em: 31 de out. 2021.

Curva de aprendizado. **VueJs org.** Guia. Disponível em: <https://br.vuejs.org/v2/guide/comparison.html#Curva-de-Aprendizado>. Acesso em: 16 de nov. de 2021.

Dados Computados e Observadores. **VueJs org.** Guia. Disponível em: <https://br.vuejs.org/v2/guide/computed.html>. Acesso em: 15 de out. de 2021.

DevMedia. Educação à distância. Perfil no LinkedIn. 69.522 seguidores. Disponível em: <https://br.linkedin.com/company/devmedia>. Acesso em: 03 de out. 2021.

Documentation. **Buefy**. Disponível em: <https://buefy.org/documentation/>. Acesso em 15 de nov. 2021.

Elements. **Bulma**. Disponível em: <https://bulma.io/documentation/elements/>. Acesso em: 11 de nov. 2021.

Empresas que usam Vue.js no Brasil. **empresas-usando-vuejs**. Disponível em: <https://empresas-usando-vuejs.netlify.app>. Acesso em: 10 de out. 2021.

FEOFILOFF, Paulo. **IME USP.** Algorítmos. Disponível em: <https://www.ime.usp.br/~pf/algoritmos/apend/math.h.html>. Acesso em: 03 de out. 2021.

Field. **Buefy**. Disponível em: <https://buefy.org/documentation/field>. Acesso em: 15 de nov. 2021.

FILIPOVA, Olga. **Learning Vue.js 2**. Birmingham: Packt Publishing Ltd, 2016.

FLATSCHART, Fábio. **HTML 5: embarque imediato**. Rio de Janeiro: Brasport, 2011.

FRAGA, Matheus. React Js do zero ao avançado na prática. **Udemy.** 2021. Disponível em: <https://www.udemy.com/course/curso-reactjs/>. Acesso em: 16 de nov. 2021.

Framework: o que é, quais utilizar e como eles funcionam!. **Hostgator.** Blog. Disponível em: <https://www.hostgator.com.br/blog/frameworks-na-programacao/>. Acesso em: 03 de out. 2021.

FRIAS, Thiago. React vs Vue vs Angular: qual escolher? **Geekhunter**. Blog. 13 de abr. 2020. Disponível em: <https://blog.geekhunter.com.br/react-vs-vue-vs-angular-qual-escolher/>. Acesso em: 10 de out. 2021.

GOTLIEB, Aaron. DateTimePicker does not support v-model binding with empty value/empty date. **Github**. Buefy, Issues. Disponível em: <https://github.com/buefy/buefy/issues/3168>. Acesso em: 21 de nov. 2021.

HANASHIRO, Akira. Qual a diferença entre Framework e Biblioteca?. **TreinaWeb.** Blog. Disponível em: <https://www.treinaweb.com.br/blog/qual-a-diferenca-entre-framework-e-biblioteca>. Acesso em 03 de out. 2021.

HECKENBERG, Stew. select addon appears squashed. **Github**. Buefy, Issues. Disponível em: <https://github.com/buefy/buefy/issues/2>. Acesso em: 21 de nov. 2021.

Input. **Buefy**. Disponível em: <https://buefy.org/documentation/input>. Acesso em: 15 de nov. 2021.

Interligações em Formulários. **VueJs org.** Guia. Disponível em: <https://br.vuejs.org/v2/guide/forms.html>. Acesso em: 15 de out. de 2021.

Introdução. **VueJs org.** Guia. Disponível em: <https://br.vuejs.org/v2/guide/index.html>. Acesso em: 14 de out. de 2021.

Introdução ao DOM. **Mozilla org**. Tecnologia Web para desenvolvedores. Disponível em:

<https://developer.mozilla.org/pt-BR/docs/Web/API/Document_Object_Model/Introduction>. Acesso em: 14 de out. de 2021.

JQuery Tutorial. **DevMedia**. Artigo. Disponível em: <https://www.devmedia.com.br/jquery-tutorial/27299>. Acesso em: 03 de out. 2021.

KADLOT, Peter. Add ability to use different number of columns on different breakpoints. **Github**. Bulma, Issues. Disponível em: <https://github.com/jgthms/bulma/issues/7>. Acesso em: 21 de nov. 2021.

KIROV, Rado. Adding support for CSS variable shimming. **Github**. Angular, Issues. Disponível em: <https://github.com/angular/angular/issues/2567>. Acesso em: 21 de nov. 2021.

KHAN, Aashir Aamir. Bulma - The Most Underrated Framework of the CSS Framework Era. **Dev.to**, Sukkur, 15 de set. 2019. Disponível em: <https://dev.to/justaashir/bulma-the-most-underrated-framework-of-the-css-framework-era-2gj8>. Acesso em: 11 de nov. 2021.

LIMA, Edson Pinheiro de; LEZANA, Álvaro Guilhermo Rojas. DESENVOLVENDO UM FRAMEWORK PARA ESTUDAR A AÇÃO ORGANIZACIONAL: DAS COMPETÊNCIAS AO MODELO ORGANIZACIONAL. **Gestão e Produção**, Curitiba; Florianópolis, v.12, n.2, p.177-190, mai.-ago. 2005. Disponível em: https://www.scielo.br/j/gp/a/7b3dyswQthxzYFwQxNL86nk/?format=pdf&lang=pt Acesso em 03 de out. 2021.

MAHER, Jason. columns layout incorrect with svg child elements. **Github**. Bulma, Issues. Disponível em: <https://github.com/jgthms/bulma/issues/268>. Acesso em: 21 de nov. 2021.

Melhortaxa. Serviços financeiros. Perfil no LinkedIn. 1.581 seguidores. Disponível em: <https://br.linkedin.com/company/melhortaxa>. Acesso em: 10 de out. 2021.

Milhares de sites Wordpress foram hackeados por falha de plugins. **Tecmundo**. 24 de jan. 2020. Segurança. Disponível em: <https://www.tecmundo.com.br/seguranca/149630-milhares-sites-wordpress-hackeados-falha-plugins.htm>. Acesso em: 16 de nov. 2021.

Modifiers syntax. **Bulma**. Disponível em: <https://bulma.io/documentation/overview/modifiers/>. Acesso em: 31 de out. 2021.

MOLENAAR, Rody. Serve Builds. **Github**. Buefy, Issues. Disponível em: <https://github.com/buefy/buefy/issues/5>. Acesso em: 21 de nov. 2021.

Moment.Js. **GitHub**. Disponível em: <https://github.com/moment/moment/>. Acesso em: 16 de nov. 2021.

Moment.Js Documentation. **MomentJs**. Project Status. Disponível em: <https://momentjs.com/docs/#/-project-status/>. Acesso em: 16 de nov. 2021.

National Business Name Availability Check. **Formal Founder**. Disponível em: <https://www.formalfounder.com>. Acesso em: 11 de nov. 2021.

Navbar. **Buefy**. Disponível em: <https://buefy.org/documentation/navbar>. Acesso em: 15 de nov. 2021.

NEWMAN, Ben. Address EMFILE (too many open files) errors for clean grunt test runs. **Github**. React, Issues. Disponível em: <https://github.com/facebook/react/issues/137>. Acesso em: 21 de nov. 2021.

NIHLÉN, Malcolm. Fieldset light buttons disabled. **Github**. Bulma, Issues. Disponível em: <https://github.com/jgthms/bulma/issues/2926>. Acesso em: 21 de nov. 2021.

O Framework JavaScript Progressivo. **VueJs org**. Disponível em: <https://br.vuejs.org/index.html>. Acesso em: 10 de out. 2021.

O que é JavaScript?. **Mozilla org.** Developer. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/What_is_JavaScript>. Acesso em: 03 de out. de 2021.

O que é Vuex?. **VueJs org**. Disponível em: <https://vuex.vuejs.org/ptbr/>. Acesso em: 10 de out. 2021.

Overview. **Bulma**. Disponível em: <https://bulma.io/documentation/overview/>. Acesso em: 31 de out. 2021.

PATTAKOS, Aris. 9 Best CSS Frameworks in 2021. **aThemes**, Jan. 2021. Disponível em: <https://athemes.com/collections/best-css-frameworks/>. Acesso em: 11 de nov. 2021.

PEREIRA, Silvio do Lago. Linguagem C. Disponível em: <https://www.ime.usp.br/~slago/slago-C.pdf>. Acesso em: 14 de out. 2021.

PIOTROWICZ, Dario. fix(docs-infra): convert button-like elements to actual buttons. **Github**. Angular, Issues. Disponível em: <https://github.com/angular/angular/pull/43601>. Acesso em: 21 de nov. 2021.

PLIEVONE. Support Pointer events specification. **Github**. React, Issues. Disponível em: <https://github.com/facebook/react/issues/499>. Acesso em: 21 de nov. 2021.

Plugin do WordPress deixa 1 milhão de sites vulneráveis a hackers. **Tecmundo**. 02 de set. 2021. Segurança. Disponível em: <https://www.tecmundo.com.br/seguranca/224268-plugin-wordpress-deixa-1-milhao-sites-vulneraveis-hackers.htm>. Acesso em: 16 de nov. 2021.

POLLACK, Gregg. WHY VUE.JS. **VueJs org**, 201?. Disponível em: <https://vuejs.org>. Acesso em: 14 de out. 2021.

Quanto ganha um Developer Senior? Estados Unidos da América. **Glassdoor**. Salários. Disponível em: <https://www.glassdoor.com.br/Salários/estados-unidos-da-américa-senior-developer-salário-SRCH_IL.0,25_IN1_KO26,42.htm>. Acesso em: 16 de nov. 2021.

Quanto ganha um Developer Senior? Brasil. **Glassdoor**. Salários. Disponível em: <https://www.glassdoor.com.br/Salaries/senior-developer-salary-SRCH_KO0,16.htm?countryRedirect=true>. Acesso em: 16 de nov. 2021.

React, Facebook. **Github**. Issues, Type: Bug. Disponível em: <https://github.com/facebook/react/issues?page=1&q=label%3A"Type%3A+Bug"+is%3Aclosed>. Acesso em: 21 de nov. 2021.
React Uma biblioteca JavaScript para criar interfaces de usuário. **React org.** Disponível em: <https://pt-br.reactjs.org>. Acesso em: 03 de out. 2021.

React, Facebook. **Github**. Issues, Type: Feature Request. Disponível em: <https://github.com/facebook/react/issues?q=label%3A"Type%3A+Feature+Request"+is%3Aclosed>. Acesso em: 21 de nov. 2021.

REIS, Vinicius. Por que Vue.js e não React?. **iMasters**, 26 de jul. 2017. Back-End. Disponível em: <https://imasters.com.br/back-end/por-que-vue-js-e-nao-react>. Acesso em: 10 de out. 2021.

Responsiveness. **Bulma**. Disponível em: <https://bulma.io/documentation/overview/responsiveness/>. Acesso em: 31 de out. 2021.

ROCHA, Milton Silva da. 1- Estrutura sequencial: “Entrada Processamento Saída”, 2018. 12 slides. Disponível em: <https://miltonrocha.eng.br/wp-content/uploads/2018/01/2018-programacao-de-computadores.pdf>. Acesso em: 15 de out. 2021.

SAMUEL, Ogundipe. Why you should use Buefy and Vue.js for your next business website. **LogRocket**. Blog, 4 de ago. 2021. Disponível em: <https://blog.logrocket.com/buefy-vue-js-next-business-website/>. Acesso em: 16 de nov. 2021.

Sass Introduction. **W3Schools**. SASS. Disponível em: <https://www.w3schools.com/sass/sass_intro.php>. Acesso em: 31 de out. 2021.

SENA, Victor. Vagas em tecnologia sobrevivem à crise e disparam 600% em SP. **Exame**. 25 de jan. 2021. Carreira. Disponível em:

<https://exame.com/carreira/vagas-em-tecnologia-sobrevivem-a-crise-e-disparam-600-em-sp/>. Acesso em: 16 de nov. 2021.

SERGII. Allow SSR to render unescaped inlined script tags. **Github**. Vue, Issues. Disponível em: <https://github.com/vuejs/vue/issues/3650>. Acesso em: 21 de nov. 2021.

Signal. **Signal.** Disponível em <https://signal.org/pt_BR/>. Acesso em 11 de nov. 2021.

Sintaxe de Templates. **VueJs org.** Guia. Disponível em: <https://br.vuejs.org/v2/guide/syntax.html>. Acesso em: 15 de out. de 2021.

Sobre o iMasters. **iMasters**. Disponível em: <https://imasters.com.br/p/sobre-o-imasters>. Acesso em: 10 de out. 2021.

Start. **Bulma**. Disponível em: <https://bulma.io/documentation/overview/start/>. Acesso em: 31 de out. 2021.

SCHWARZMÜLLER , Maximilian; LEITAO, Leonardo Moura; COD3R. Curso Vue JS 2 - O Guia Completo (incl. Vue Router & Vuex). **Udemy**. 2021. Disponível em: <https://www.udemy.com/course/vue-js-completo/>. Acesso em: 16 de nov. 2021.

Taxa de câmbio comercial para compra: real (R$) / dólar americano (US$) - média. **Ipeadata**. Disponível em: <http://www.ipeadata.gov.br/ExibeSerie.aspx?serid=38590&module=M>. Acesso em: 16 de nov. 2021.

Tecnologia. **Escola Virtual Bradesco**. Disponível em: <https://www.ev.org.br/areas-de-interesse/tecnologia>. Acesso em: 16 de nov. 2021.

The Bulma Expo. **Bulma**. Disponível em: <https://bulma.io/expo/>. Acesso em: 11 de nov. 2021.

TROQUATTE, Dener. Curso de Angular 2 + Typescript do Básico ao Avançado. **Udemy**. Disponível em: <https://www.udemy.com/course/curso-de-angular/>. Acesso em: 16 de nov. 2021.

TUNG, Liam. An insecure mess: How flawed JavaScript is turning web into a hacker's playground. **ZDNet**. Security. Disponível em: <https://www.zdnet.com/article/an-insecure-mess-how-flawed-javascript-is-turning-web-into-a-hackers-playground/>. Acesso em: 16 de nov. 2021.

Videos. **Bulma**. Disponível em: <https://bulma.io/videos/>. Acesso em: 21 de nov. 2021.

Você e a Melhortaxa. **Melhortaxa**. Disponível em: <https://www.melhortaxa.com.br/>. Acesso em: 11 de nov. 2021.

Vue-router. **Github**. VueJs. Disponível em: <https://github.com/vuejs/vue-router>. Acesso em: 10 de out. 2021.

Vue, Vuejs. **Github**. Issues, bug. Disponível em: <https://github.com/vuejs/vue/issues?q=is%3Aopen+is%3Aissue+label%3Abug>. Acesso em: 21 de nov. 2021.

Vue, Vuejs. **Github**. Issues, feature request. Disponível em: <https://github.com/vuejs/vue/labels/feature%20request>. Acesso em: 21 de nov. 2021.

Wappalyzer. **Google**. Disponível em: <https://chrome.google.com/webstore/detail/wappalyzer/gppongmhjkpfnbhagpmjfkannfbllamg?hl=pt-BR>. Acesso em: 10 de out. 2021.

Whats is JQuery?. **JQuery org.** Disponível em: <https://jquery.com>. Acesso em: 03 de out. 2021.

WOLFF, Harry. Things I Don't Like about Vue.js (as a React engineer). **Hs Wolff.** Blog. 26 de out. 2020. Disponível em: <https://hswolff.com/blog/things-i-dont-like-about-vuejs-as-a-react-engineer/>. Acesso em: 10 de out. 2021.

WROBLEWSKI, Luke. Mobile First. **Lukew**. [S.l.], out. 2011. Disponível em: <https://www.lukew.com/resources/mobile_first.asp>. Acesso em: 31 de out. 2021.

YOU, Evan. dependency tracking for nested values. **Github**. Vue, Issues. Disponível em: <https://github.com/vuejs/vue/issues/25>. Acesso em: 21 de nov. 2021.