INTERFACE TECNOLÓGICA

ANÁLISE COMPARATIVA ENTRE FRAMEWORKS FRONTEND BASEADOS EM JAVASCRIPT PARA APLICAÇÕES WEB

COMPARISON ANALYSIS BETWEEN FRONTEND JAVASCRIPT BASED FRAMEWORKS FOR WEB APPLICATIONS

Haline Kelly Ferreira – halineferreira@gmail.com
Jederson Donizete Zuchi – jederson.zuchi@fatectq.edu.br
Faculdade de Tecnologia de Taquaritinga (FATEC) – SP – Brasil
DOI: 10.31510/infa.v15i2.502

RESUMO

Diferente de alguns anos atrás, quando desenvolvedores Web utilizavam apenas JavaScript puro para dar vida a seus projetos, o desenvolvimento *frontend* atual possui diversas opções de frameworks para melhor estruturar o desenvolvimento do projeto. Porém, essa vasta quantidade de opções pode gerar muitas dúvidas na hora de escolher, e isso pode afetar o tempo e produtividade do desenvolvimento. Este artigo tem como objetivo apontar os principais frameworks JavaScript do mercado e mostrar suas características mais marcantes, a fim de auxiliar os desenvolvedores de software na escolha daquele que mais se adequa às necessidades e expectativas de seu projeto. Para isso, após uma breve introdução de cada framework, alguns itens foram analisados. O primeiro deles é a arquitetura de cada um, seguido pelo fator documentação e suporte da comunidade, que influencia diretamente na evolução do aprendizado do desenvolvedor e auxílio em seus momentos de dificuldade. Em seguida, mostra-se a compatibilidade de cada um dos frameworks com os principais navegadores do mercado atual. Então, apresenta-se o tamanho do pacote de arquivos, que pode influenciar no tempo de carregamento da aplicação, que, se muito lento, pode fazer com que o usuário final desista do acesso. O item seguinte é o tempo de renderização, que, assim como o tamanho do pacote de arquivos, pode afetar a usabilidade do usuário final no sistema. Por fim, é possível colher informações mais assertivas dos mesmos para identificar aquele que mais atende aos principais requisitos do projeto a ser desenvolvido.

Palavras-chave: Framework. Análise Comparativa. Angular. Vue. React.

ABSTRACT

Web development in today's industry is vastly different than it was years ago. In the past, Web developers used only pure JavaScript to bring life to their projects. Current *frontend* development has several framework options to better structure project development. However, the vast number of options can raise many questions during development and can also affect development time and productivity. This article aims to point out the main JavaScript frameworks on the market and show their main characteristics in order to help

software developers choose the one that best suits their project needs and expectations. To achieve this, each framework is briefly introduced, and some aspects are analyzed. The first aspect is the architecture of each of them, followed by the documentation and community support. These aspects directly influence the evolution of the developer's learning and help and help when they encounter challenges. Next, it shows the compatibility of each framework with the main browsers of today's market. Then, the size of the file package is presented, which can influence the application load time, which is important because if it is too slow, it can make the end user to give up access. The next item is the rendering time, which, like the size of the file package, can affect the usability of the end user's experience in the system. Then, the reader can gather more assertive information about framework options to identify which one best matches the project requirements.

Keywords: Framework. Comparison Analysis. Angular. Vue. React.

1 INTRODUÇÃO

Todos os dias os desenvolvedores de software, assim como outras pessoas, precisam fazer diversas escolhas, seja na vida pessoal ou profissional e, nem sempre têm as orientações necessárias sobre como proceder para tomar a melhor decisão. Na área de desenvolvimento de software, existem diversas opções de linguagens de programação disponíveis e ainda, os desenvolvedores, se deparam com novos frameworks e ferramentas a cada dia. De acordo com Mariano (2017), para os desenvolvedores Web, é importante não apenas utilizar o framework que atende às necessidades de seu projeto, mas que também provê um código de alta qualidade e boa performance. E segundo Mizuta (2017), a escolha de um framework traz suas vantagens e desvantagens, por isso é a análise é importante.

Mariano (2017) ainda afirma que um típico framework JavaScript deve abstrair ou generalizar as mais longas e complexas operações, para promover suporte e compatibilidade entre navegadores, assegurando e permitindo o rápido desenvolvimento de software. Apesar da vasta lista de opções, na maioria das vezes, a escolha de um framework pode gerar muitas dúvidas para os profissionais, pois, segundo Gizas (2012), além da alta qualidade de código e boa performance, também devem ser levados em consideração a manutenibilidade e suporte ativo da comunidade.

De acordo com Mariano (2017), muitas vezes, os profissionais optam por utilizar mais de um framework, especialmente para o desenvolvimento de aplicações mais complexas ou de larga escala. A principal vantagem disso é reutilizar o código, permitindo que as organizações concentrem mais tempo e atenção em projetar um aplicativo Web escalável, escolhendo o

framework apropriado, que pode ser abraçado por todos os desenvolvedores dentro de uma empresa. Por outro lado, se uma estrutura menos adequada for escolhida, isso pode afetar muito o desenvolvimento de uma aplicação, provocando uma reação em cadeia, em que a qualidade da aplicação é reduzida e os prazos são perdidos.

Para este estudo, foram comparados os principais frameworks JavaScript do mercado, Angular, Vue e React, a fim de auxiliar os desenvolvedores Web na definição de qual deles é mais adequado ao seu projeto.

2 HISTÓRIA E EVOLUÇÃO DO JAVASCRIPT

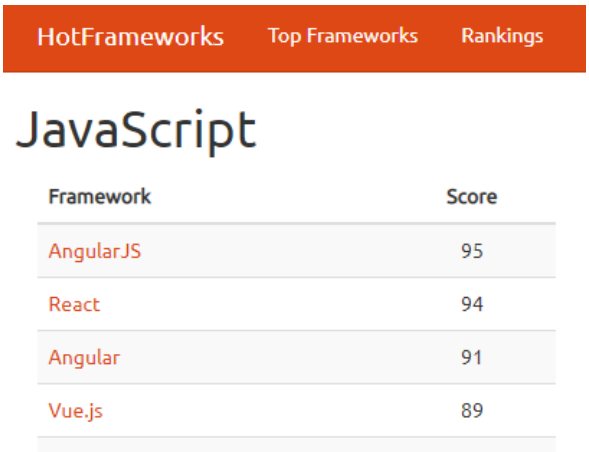
Há alguns anos, as páginas da internet eram estáticas, com intenção de apenas exibir seu conteúdo aos usuários, que somente clicavam em um link e esperavam uma nova página ser carregada (MCFARLAND, 2014). Em um segundo momento, o usuário passou a também enviar informações e, tempos depois, com o surgimento da linguagem JavaScript, a interação entre o sistema e o usuário final passou a ser mais dinâmica e interativa, com respostas instantâneas e páginas responsivas com efeitos visuais dinâmicos. Para Elliott (2014), "JavaScript é a mais importante linguagem de programação da Terra. Quase todo mundo com um computador ou um smartphone tem todas as ferramentas que precisa para executar um programa JavaScript que ele mesmo criou". Portanto, é possível criar um programa com poucos recursos e alguns conhecimentos básicos, contudo, aprofundando-se na linguagem, muitas outras possibilidades aparecem.

O JavaScript foi criado no ano de 1995 por Brendan Eich em apenas 10 dias enquanto o mesmo trabalhava na Netscape (MCFARLAND, 2014). Ele possibilitou uma revolução da maneira que a Web trabalhava, pois, ela deixava de ser um grupo de documentos interligados para ser um verdadeiro ambiente de imersão e interação (BARLETT, 2016). A princípio, era vista como uma linguagem de programação de hobby, usada apenas para acrescentar efeitos menos úteis, como informações que se deslocam com a barra de rolagem, ou até mesmo borboletas animadas que seguem o cursor do mouse (MCFARLAND, 2014). Era também, muitas vezes, evitada devido à dificuldade de desenvolver o mesmo efeito para diferentes *browsers*, mas, com a evolução da linguagem, foram definidos padrões que facilitam o desenvolvimento de programas JavaScript, fazendo com que muitos desenvolvedores aderissem à linguagem e essa se tornasse uma das mais populares atualmente.

2.1 Frameworks e as dificuldades de seleção

Um framework é uma aplicação escrita previamente, oferecendo um conjunto de bibliotecas e uma série de componentes, tornando o desenvolvimento de aplicações Web mais produtivo, flexível, manutenível e testável (BRANAS, 2014). Acerca dos frameworks Javascript, Mizuta (2017) afirma que "uma das principais vantagens é podermos utilizar uma solução pronta e testada para problemas comuns, ou seja, não precisamos reinventar a roda para cada projeto". Um framework descreve como uma aplicação deve ser construída, permitindo que o código fique mais organizado, reforçando a escalabilidade e flexibilidade de uma aplicação. Ele oferece a descrição de uma arquitetura, porém, é ainda mais que um *template*, pois possui *helpers*, construtores, etc. Atualmente, existem diversos frameworks JavaScript tanto para o desenvolvimento *frontend* quanto para o *backend*, e diversos fatores podem influenciar a escolha (GIZAS, 2012).

Figura 1 - Principais frameworks JavaScript



Fonte: HOTFRAMEWORKS (2018)

Como pode-se observar na Figura 1, o site HotFrameworks (que cria rankings de frameworks de diversas linguagens baseando-se na popularidade dos projetos desenvolvidos utilizando os mesmos no GitHub, e no número de questões os referenciando no Stack Overflow) mostra que os frameworks JavaScript mais populares são: AngularJS (primeira versão do Angular), React, Angular e Vue. Suas versões mais atuais no momento em que o

presente artigo é escrito serão analisadas neste estudo: Angular 6.2.2, React 16.5.1 e Vue 2.5.17.

2.1.1 Angular

De acordo com Branas (2014), o AngularJS foi criado por Miško Hevery e Adam Abron em 2009 e é um framework JavaScript *open source* (código aberto), *client-side* (do lado do cliente) que promove uma alta produtividade na experiência do desenvolvimento Web. Ele permite utilizar sintaxe HTML para utilizar os componentes da aplicação de forma clara e sucinta (TUTORIALS PONT, 2014). Quando Hevery passou a trabalhar em um projeto da Google, percebeu que haviam mais de 17 mil linhas de código, quando decidiu reescrevê-lo, implantando seu framework. Três semanas depois, o projeto foi entregue com 1.500 linhas e, atualmente, é usado em mais de 100 projetos da empresa.

O framework passou por uma grande modificação no lançamento de sua versão 2, que foi, na verdade, uma reformulação que o time do Angular decidiu fazer para ter que o framework tivesse as vantagens de muitas funcionalidades novas, se tornando inclusive, compatível com as versões mais recentes do TypeScript. Apesar das melhorias, o fato desta versão ser totalmente diferente das antecessoras fez com que muitos profissionais tivessem que aprender a utilizar um novo framework e questionar se o mesmo aconteceria nas versões futuras. Isso os fez pensar com mais atenção em outras opções de frameworks, mais estáveis. (BOOTH, 2017).

2.1.2 Vue

Segundo sua documentação (2018), Vue é uma lib/framework JavaScript reativo, para o desenvolvimento de componentes que, por sua vez, são códigos que podem ser reaproveitados em sua aplicação.

Resumidamente, a função da lib reativa Vue é observar um objeto JavaScript e refletir qualquer mudança do seu estado no DOM do HTML. (LACERDA, 2017).

Filipova (2016) afirma que a ideia do Vue surgiu enquanto Evan You trabalhava para a Google Creative Labs, com o propósito de desenvolver um protótipo rapidamente oferecendo uma maneira fácil e flexível de *data binding* reativo e componentes reusáveis.

Para Kyriakidis (2016), o Vue tem um ótimo ecosistema de *plugins* e ferramentas que estende seus serviços básicos, podendo-se incluir rapidamente em qualquer projeto, grande ou pequeno, com poucas linhas de código. É rápido, leve e o futuro do desenvolvimento frontend.

2.1.3 React

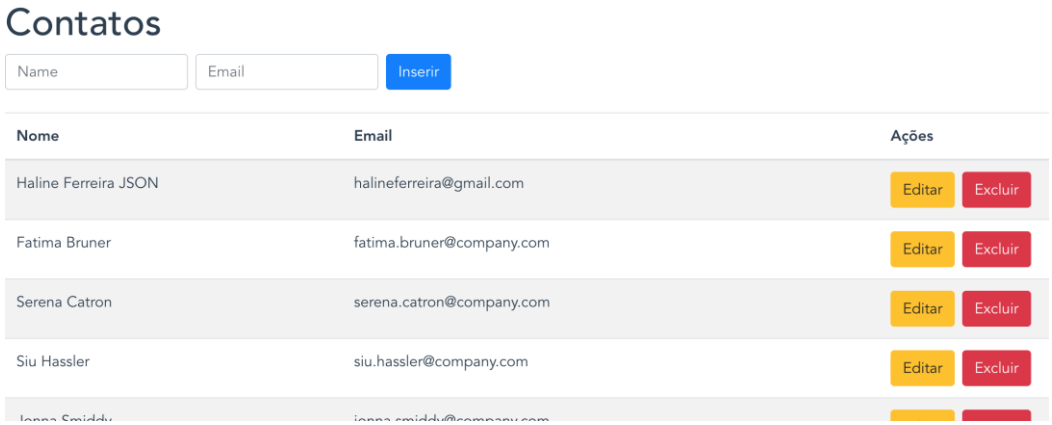
Desenvolvido por Jordan Walke, um engenheiro de software do Facebook, o React é hoje mantido por esta empresa, juntamente com o Instagram e uma comunidade de desenvolvedores individuais (MARIANO, 2017). Segundo a sua documentação (2018), ele é, na verdade, uma biblioteca de UI (*User Interface*), representando apenas a camada *view* do *Model View Controler* (MVC). Apesar disso, torna-se muito performático por atualizar o DOM retornando apenas o novo elemento. Por ser apenas uma biblioteca declarativa (faz com que o desenvolvedor foque mais no resultado do que na forma como ele é atingido) e não uma solução completa, muitas vezes é necessário utilizá-lo juntamente com outras bibliotecas ou frameworks para se chegar no resultado desejado, o que não é o caso desse estudo.

"React enfatiza a programação funcional sobre programação orientada a objetos. Essa mudança de pensamento pode levar a benefícios em áreas como testabilidade e desempenho" (BRANKS, 2017).

3 PROCEDIMENTOS METODOLÓGICOS (OU MATERIAIS E MÉTODOS)

Para a realização deste trabalho, foram desenvolvidas pequenas aplicações semelhantes, *Single Page Application* (SPA, ou aplicação de página única), utilizando cada um dos frameworks em sua versão *Command Line Interface* (CLI). Para cada uma delas, foi criado um componente com um simples formulário de contato contendo dois campos (nome e email) e uma listagem de 100 registos prévios vindos de um arquivo JSON e dos dados registrados através do formulário. Nessa listagem, é possível editar e excluir os registros. Para a estilização da página, foi utilizada a versão específica do Bootstrap de cada framework. A captura de tela da aplicação pode ser observada na Figura 2.

Figura 2 - Captura de tela da aplicação desenvolvida.



Fonte: **elaborada pelos autores (2018).**

Os códigos-fonte das aplicações podem ser encontrados nos seguintes endereços de internet para reprodução:

- **Angular**: https://github.com/halineferreira/simple-crud-angular
- **Vue**: https://github.com/halineferreira/crud-vuejs
- **React**: https://github.com/halineferreira/simple-crud-reactjs

3.1 Ambiente de Comparação

O ambiente usado para a comparação de performance foi um MacBook Pro com as seguintes características:

Sistema Operacional: macOS High Sierra (Version 10.13.6)
Processor: 3 GHz Intel Core i7
Armazenamento: Apple SSD 128GB
Memória RAM: 16 GB 1600 MHz DDR3
Network: Ethernet LAN Connection: 20Mbps Download / 3Mbps Upload
Versão do navegador: Google Chrome versão 69.0.3497.100

3.2 Fatores analisados

- Arquitetura;
- Documentação e comunidade, verificando-se os principais recursos para interação dos membros obtenção de suporte e esclarecimento de dúvidas;
- Compatibilidade, analisando-se os navegadores compatíveis com cada framework;
- Tamanho do pacote de arquivos, comparando-se o tamanho do arquivo JavaScript gerado; e,
- Tempo de renderização, medindo-se o tempo necessário para o carregamento da página;

4 RESULTADOS E DISCUSSÃO

4.1 Arquitetura

Para a experiência do presente estudo, foi desenvolvida uma SPA com cada um dos frameworks. Porém, "ecada framework possui seu próprio ecossistema e soluções para resolver problemas do *client-side*, como roteamento, gerenciamento de status da aplicação, comunicação com *backend*, etc" (SVIATOSLAV, 2018). Embora as estruturas completas não tenham sido utilizadas para o desenvolvimento deste projeto, é importante conhecê-las.

O Angular não deixa um padrão explícito em sua documentação, porém, em sua comunidade, profissionais afirmam que seja *Model-View-Whatever* (MVW), ou seja, pode-se utilizar a estrutura que for mais conveniente ao desenvolvedor.

Em sua documentação (2018), é afirmado que o design do Vue é parcialmente inspirado no padrão *Model-View-ViewModel* (MVVM) e, como convenção, é utilizada a nomenclatura *ViewModel*. “Ele conecta a Visualização e o Modelo por meio de ligações de dados bidirecionais. Manipulações DOM reais e formatação de saída são abstraídas em diretivas e filtros”.

Mariano (2017) afirma que o React representa apenas o V (de View) da arquitetura MVC. “Ele faz uso de um *Document Object Model* (DOM) virtual que é usado para a nova renderização eficiente do DOM”.

4.2 Documentação e comunidade

Ao escolher um framework, biblioteca ou qualquer tipo de ferramenta para o desenvolvimento, é importante saber se haverá algum tipo de suporte caso haja algum tipo de dificuldade. A documentação dos frameworks detalha suas características e como devem ser usados. Caso haja algum problema, é importante poder entrar em contato com profissionais experientes para sanar dúvidas específicas, para isso, existem as comunidades.

Os três frameworks analisados possuem uma extensa documentação em seu website. Diferentemente dos outros, o Vue possui suporte em diversos idiomas, inclusive em português do Brasil. Já os outros frameworks oferecem suporte oficial apenas em inglês.

Como pode-se observar no Quadro 1 e de acordo com os websites dos frameworks em questão, todos possuem seu código disponível no GitHub, uma rede social específica para desenvolvedores. E, segundo pesquisa do site Stack Overflow (2018), todos também estão presentes na comunidade. Todos os frameworks possuem algum tipo de chat para discussões: o Angular está presente no Gitter, já o Vue e o React, estão no Discord e ainda possuem fóruns de discussão integrados em seus websites. Além das opções divulgadas no site de cada framework, existem diversos recursos de comunidade não oficiais disponíveis na internet.

Quadro 1 - Recursos de comunicação dos frameworks JavaScript

	Angular	**Vue**	**React**
Fórum integrado	Não	Sim	Sim
Chat	Gitter	Discord	Discord
Github	Sim	Sim	Sim
StackOverflow	Sim	Sim	Sim

Fonte: Elaborado pelos autores (2018)

4.3 Compatibilidade de navegadores

De acordo com suas documentações oficiais, Vue e React suportam quaisquer *browsers* capazes de interpretar EcmaScript 5, ou seja, os mais recentes do mercado e algumas de suas versões antecessoras.

Já o Angular disponibiliza, também em sua documentação, uma lista com os *browsers* suportados, que também inclui os navegadores contemporâneos, mas também, alguns em versões mais antigas, como, por exemplo, o Internet Explorer 9.

4.4 Tamanho do pacote de arquivos

Segundo Mizuta (2017), para que um site seja carregado, o navegador deve baixar os arquivos necessários, como os de extensão .html, .css, .js, além de fontes, imagens, etc. Com o uso de um framework, há ainda mais arquivos a serem carregados, aumentando também o consumo de rede, o que pode ser prejudicial, principalmente para usuários de dispositivos móveis que têm o uso de internet limitado. Outro fator em que o tamanho do pacote pode interferir é o tempo de renderização. Dependendo da velocidade de internet do usuário, o navegador pode aguardar a renderização dos dados por um tempo determinado, ao esperar por um tempo prolongado, o navegador pode ficar inoperante, fazendo com que o usuário perca o interesse no conteúdo não carregado.

Para analisar o tamanho dos arquivos JavaScript, foi gerada a versão de *build* de cada projeto e o tamanho do arquivo de extensão .js foi verificado através das propriedades do arquivo, disponibilizadas pelo sistema operacional, e observa-se na Tabela 1:

Tabela 1 - Tamanho do arquivo JavaScript gerado

Framework	Tamanho do aplicativo gerado
Angular	16 KB
Vue	11 KB
React	6 KB

Fonte: elaborada pelos autores (2018)

A partir das informações da Tabela 1, nota-se que o Angular gera o maior arquivo, com 16 KB, seguido pelo Vue, com 11 KB e, por último, o React, que gera o arquivo de menor tamanho, 6 KB.

4.5 Tempo de renderização

O tempo de renderização foi analisado através do recurso do painel Network, do Chrome DevTools, em que há um item chamado Load, que, segundo a documentação da ferramenta (2018) "é acionada quando uma página é totalmente carregada", o que inclui os elementos DOM e JavaScript. Por haver variações, foram colhidas 100 amostras de tempo para cada framework e a média dos resultados foram calculadas, como pode-se observar na Tabela 2:

Tabela 2 - Tempo de renderização da página

Framework	Load
Angular	1.17 s
Vue	767 ms
React	788 ms

Fonte: elaborada pelos autores (2018)

Por meio dos testes com as aplicações desenvolvidas para a pesquisa, pode-se notar que o Vue foi o mais performático em relação ao tempo médio de renderização, com 767 ms para carregar completamente a página. Já o React, foi o segundo mais rápido, com média de 788 ms, e, por último, o Angular, com 1.17 s para carregamento da mesma.

5 LIMITAÇÕES

Dentre as principais limitações desta pesquisa, destaca-se a escassez de artigos acadêmicos sobre o assunto, o que aumenta a dificuldade de encontrar fundamentos confiáveis para citação.

Outro ponto é a pequena quantidade de frameworks testados dentre muitos existentes. Seria interessante, o desenvolvimento de um projeto maior, com diversos outros, inclusive, usando a integração com o *backend* e persistência em banco de dados.

6 CONCLUSÃO

Durante o desenvolvimento da pesquisa, por se tratar de frameworks de mesma linguagem base, foi possível observar grandes similaridades no código de cada um e, também, suas particularidades, o que é acarretado em vantagens e desvantagens em índices pontuais, sendo que cada um dos frameworks destaca-se em diferentes pontos.

Todos eles, possuem uma vasta comunidade e são compatíveis com os principais navegadores de internet do mercado atual. No quesito tamanho, o React traz vantagens diante dos concorrentes abordados, porém, não é o mais vantajoso quando se trata de velocidade de carregamento da página, sendo o Vue, o mais veloz, seguido do React, com pouca diferença no tempo de carregamento. Tratando-se dos dois últimos fatores analisados, apesar de ser um framework bastante completo, o Angular foi o que menos se destacou.

Portanto, pode-se concluir que cada framework pode trazer determinada vantagem ao projeto, e cabe ao desenvolvedor definir qual índice é o mais importante para sua aplicação.

REFERÊNCIAS

BARTLETT, J. **New Programmers Start Here**: An introduction to computer programming using JavaScript. Burlington, United States: Bartlett Publishing, 2016.

BOOTH J. D. **Angular 2 Succinctly**. Morrisville, United States of America: Syncfusion, 2017.

BRANAS, R. **AngularJS essentials**: Design and construct reusable, maintainable, and modular web applications with AngularJS. Birmingham, United Kingdom: Packt Publishing, 2014.

BRANKS A.; PORCELLO E. **Learning React:** Functional Web Development with React and Redux. Sebastopol, United States of America: O'Reilly, 2017

ELLIOTT E. **Programming JavaScript applications**: Robust Web Architecture with Node, HTML5, and a Modern JS Library. Sebastopol, United States of America: O'Reilly, 2014.

FILIPOVA O. **Learning Vue.js 2**. Birmingham, United Kingdom: Packt Publishing. 2016.

GIZAS, A. B. et al. **Comparative evaluation of JavaScript frameworks.** Lyon, France: HPCLab, 2012.

HOTFRAMEWORKS. **JavaScript.** 2017. Disponível em: <https://hotframeworks.com/languages/javascript>. Acesso em: 18 de set. 2018.

KYRIAKIDIS A.; MANIATIS K.; YOU E. **The Majesty of Vue.js**. Leanpub, 2016.

LACERDA, M. **No mar de libs e frameworks: conhecendo o Vue.js – Parte I**. [s.l.]: BrazilJS, 2017. Disponível em: <https://braziljs.org/blog/no-mar-de-libs-e-frameworks-conhecendo-o-vue-js-parte>. Acesso em: 22 out. 2017.

MARIANO, C. L. **Benchmarking JavaScript frameworks**. Tese (M.Sc. in Computing - Advanced Software Development). Dublin Institute of Technology. Dublim, Irlanda: 2017.

MCFARLAND, D. S. **JavaScript & jQuery**: The Missing Manual. 3. ed. Sebastopol, United States of America: O'Reilly, 2014.

MIZUTA, M. **Frameworks e bibliotecas JavaScript: quando e qual usar.** Elo7, 2017. Disponível em: <https://engenharia.elo7.com.br/frameworks-js/>. Acesso em: 20 de fev. 2018.

TUTORIALS POINT. **Learn AngularJS: Web application framework**. 2014. Disponível em: <https://www.tutorialspoint.com/angularjs/angularjs_tutorial.pdf>. Acesso em: 22 out. 2017.

SVIATOSLAV, A. **The Best JS Frameworks for Front End**. Ruby Garage, 2018. Disponível em: <https://rubygarage.org/blog/best-javascript-frameworks-for-front-end>. Acesso em: 18 de set. 2018.