Progressive Web Apps: Análise comparativa de frameworks JavaScript para desenvolvimento

Guilherme Gehring* Anubis Graciela de Moraes Rossetto †

2021

Resumo

Este artigo apresenta uma análise comparativa de frameworks JavaScript para desenvolvimento de Progressive Web Apps (PWA), envolvendo as bibliotecas Vue.js e React. Para isso, o trabalho discorre sobre o desenvolvimento de uma mesma aplicação em ambos os frameworks, incorporando serviços e ferramentas de geolocalização, armazenamento de dados e hospedagem das aplicações. Além do desenvolvimento das aplicações em cada framework, é apresentada uma avaliação das aplicações utilizando a ferramenta Google Lighthouse, que analisa aplicativos e páginas da web, coletando métricas de desempenho e ideias sobre as práticas do desenvolvimento, verificando aspectos de desempenho e atendimento dos elementos que devem estar presentes em um PWA.

Palavras-chave: Progressive Web App. React. Vue.js. Frameworks JavaScript.

1 Introdução

Aproximadamente metade dos usuários de dispositivos mobile não instala nenhuma solução em um mês, enquanto o usuário final médio baixa duas. Por falar em quem baixa aplicativos ao longo do mês, a média de downloads por pessoa é de 3,5 (SUSHKO, 2017). Deste modo a construção de softwares nativos, que fazem com que o usuário tenha que buscá-lo em uma loja e realizar o download e a instalação para então poder utilizá-lo, podem dificultar a divulgação e a utilização da aplicação.

*<guilhermegehring.pf178@academico.ifsul.edu.br>
†<anubis.rossetto@passofundo.ifsul.edu.br>

Visando isso, Progressive Web App (PWA) refere-se a uma metodologia de desenvolvimento de software diferenciada em comparação com aplicativos instaláveis e aplicações web tradicionais. Trata-se de um método para a construção de páginas web que propõe uma solução de maneira híbrida, onde pode-se utilizar dos recursos disponibilizados pelos dispositivos móveis, como o envio de notificações, acesso a câmera e galeria, funcionamento offline, entre outros, enquanto possui as facilidades do desenvolvimento de uma página web que rodará em um navegador.

Segundo Russell e Berriman (2015), "esses aplicativos não são empacotados e implantados em lojas, são apenas sites que tomaram todas as vitaminas certas". Esta abordagem visa uma aplicação web que mantém as características da web enquanto entrega uma experiência de usuário próxima à dos aplicativos nativos.

O desenvolvimento de aplicações nativas para Android e iOS é um processo com alto custo, os aplicativos são construídos para o sistema operacional do dispositivo e, portanto, precisam ser codificados em diferentes linguagens de programação diferentes sistemas operacionais, como Java e Swift (MUMAN, 2021).

Assim, o desenvolvimento empregando a abordagem PWA vem ganhando destaque no campo de aplicações móveis, dado que a ideia principal dos PWAs é aprimorar os aplicativos da web de forma gradual, adicionando novos recursos a aplicativos da web existentes ou novos com base no suporte do navegador.

Atualmente existem diversos frameworks JavaScript para desenvolvimento de aplicações web que possuem as especificações necessárias para o desenvolvimento com a abordagem PWA, dentre os quais podemos citar o Angular[1], o Vue[2], o React[3] e o Ember[4]. Dois que tem recebido destaque na comunidade de desenvolvedores são o React e o Vue.js. Eles são tecnologias JavaScript que possuem uma curva de aprendizado baixa, apresentam bom desempenho, tem comunidades grandes e ativas e muito úteis para aplicações de pequena e grande escala (ROGOJAN, 2019).

A escolha do framework é algo importante pois pode impactar no esforço de desenvolvimento necessário para criar a aplicação e na qualidade e desempenho percebidos da aplicação gerada. Assim, quando um desenvolvedor define por um framework, considera que tal opção maximizará a reutilização da base de código do projeto e minimizará o esforço total necessário para suportar múltiplas plataformas, bem como será capaz de produzir um produto final responsivo, atraente, fácil de usar e consistente entre plataformas (GONSALVES, 2018).

Assim, este trabalho partiu da seguinte questão de pesquisa: no desenvolvimento de PWA's, qual o impacto da escolha de um framework JavaScript no desempenho e na qualidade percebida da solução final? Pretende-se, portanto, avaliar os frameworks Vue e React no desenvolvimento de PWAs, a partir do desenvolvimento de uma aplicação em ambas bibliotecas JavaScript e aferir os resultados com a ferramenta Lighthouse.

O artigo está organizado da seguinte maneira: A seção 2 apresenta as tecnolo-

[1] https://angular.io/
[2] https://vuejs.org/
[3] https://reactjs.org/
[4] https://emberjs.com/

gias utilizadas no desenvolvimento do trabalho. A seção 3 detalha o desenvolvimento. A seção 4 apresenta os resultados obtidos a partir da avaliação dos frameworks por meio da ferramenta Lighthouse. Por fim, a última seção contém as considerações finais apresentando sugestões de trabalhos futuros.

2 Tecnologias utilizadas no desenvolvimento

Nesta seção é apresentado o embasamento teórico sobre o qual o trabalho se fundamenta, assim como as tecnologias utilizadas pelo mesmo. Desta forma, são abordados conceitos e características de PWAs, os frameworks Vue e React e os serviços da plataforma Firebase.

2.1 Progressive Web App (PWA)

Progressive Web App trata-se de uma nova metodologia para a construção de páginas web que propõe uma solução de maneira híbrida, onde pode-se utilizar dos recursos disponibilizados pelos dispositivos móveis enquanto mantém os recursos oferecidos pela maioria dos navegadores.

Para auxiliar na construção desta nova classe de aplicativos, Russell e Berriman (2015) enumeram algumas características, que são:

- Responsivo: deve funcionar igualmente em qualquer tamanho de tela;
- Independente de conectividade: progressivamente aprimorado com Service Workers para que seja possível operar offline;
- Interações semelhantes a aplicativos: adote um modelo de aplicativo Shell + Conteúdo para criar navegações e interações atraentes e que simulem as de um aplicativo nativo;
- Atualizado: Sempre atualizado de maneira transparente graças ao processo de atualização do Service Worker;
- Seguro: Servido pelo protocolo Transport Layer Security (TLS) (um requisito do Service Worker) para evitar espionagem e manter a integridade das informações;
- Detectáveis: São identificáveis como “aplicativos” graças aos manifestos e ao escopo de registro do Service Worker, permitindo que os mecanismos de pesquisa os encontrem;
- Reengajável: Pode acessar as User Interfaces (UIs) de reengajamento do sistema operacional; por exemplo, notificações push;
- Instalável: Na tela inicial por meio de prompts fornecidos pelo navegador, permitindo que os usuários ”mantenham” os aplicativos que considerem mais úteis sem o incômodo de uma loja de aplicativos;

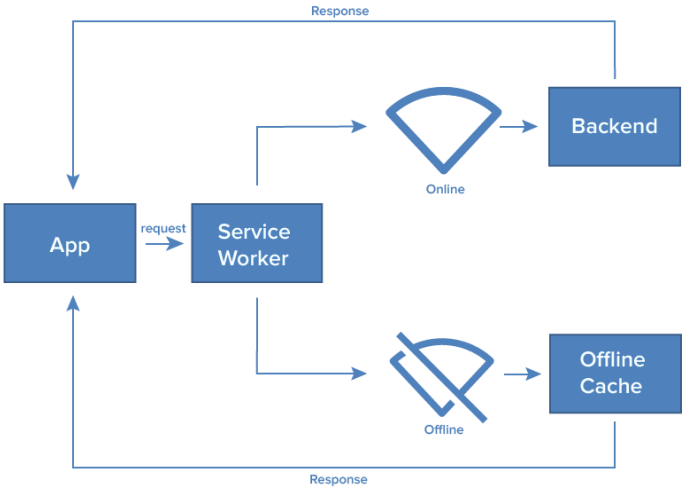
- Linkável: O que significa que são sem atrito, sem instalação e fáceis de compartilhar. Deve aproveitar-se do poder social das Uniform Resource Locators (URLs).

Existem alguns requisitos básicos para o PWA, que muitas vezes são similares aos definidos como padrões na web. Para criar um PWA é preciso manter os padrões do arquivo manifest.json na raiz do sistema. Esse arquivo aplica o nome, cor do browser e modo de execução, além de outras informações relevantes para o funcionamento (PATEL, 2019).

Além do arquivo de manifesto, outro componente básico para a construção deste tipo de aplicações se trata do Service Worker. Um Service Worker é um script que o navegador executa em background, separado de uma página da web, abrindo a porta para recursos que não precisam de uma página da web ou interação do usuário. Hoje, eles já incluem recursos como notificações push e sincronização em segundo plano. No futuro, os Service Workers podem oferecer suporte a outras coisas, como sincronização periódica ou geofencing[5]. O principal recurso é a capacidade de interceptar e manipular solicitações de rede, incluindo o gerenciamento programado de um cache de respostas (GAUNT, 2019).

A Figura 1 mostra o comportamento de um Progressive Web App fazendo uma requisição, utilizando dos recursos do Service Worker, que é capaz de buscar os dados no back-end da aplicação e quando estiver off-line, buscar pelos dados armazenados no cache do dispositivo.

Figura 1 – Comportamento de um PWA



Fonte: (VU, 2019)

[5] Geofencing: É um serviço baseado em localização que usa de posicionamento global ou identificação de radiofrequência para definir um limite virtual que aciona uma ação quando um dispositivo entra ou sai em uma área pré-definida.

2.2 Vue.js

Vue é um framework progressivo para a construção de interfaces de usuário (VUE.JS, 2020). 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. Por outro lado, Vue também é perfeitamente capaz de fornecer Single-Page Applications avançadas quando usado em conjunto com ferramentas modernas e bibliotecas de apoio (VUE.JS, 2020).

Um conceito importante para o desenvolvimento com a tecnologia é o seu sistema de componentes. Cada componente pode ser visto como uma instância do projeto, deste modo o framework proporciona a construção de aplicações de larga escala compostas por pequenos componentes interligados, fazendo com que a interface de uma aplicação possa ser abstraída em uma árvore de componentes.

Para os componentes, o Vue usa arquivos .vue proprietários, como arquivos de modelo, semelhante ao modo como o React usa os arquivos .jsx. Esses arquivos combinam elementos HyperText Markup Language (HTML) e funcionalidades dinâmicas de JavaScript. Os componentes são geralmente construídos em um formato pequeno, independente e reutilizável e, em seguida, combinados em módulos maiores e coesos. Os componentes modulares são geralmente definidos em uma estrutura de árvore de arquivos e pastas. Os componentes do Vue são semelhantes aos elementos personalizados, que fazem parte das especificações dos componentes da web e foram modelados a partir deles (LEVLIN, 2020).

2.3 React.js

React.js é uma biblioteca JavaScript de código aberto criada pelo Facebook em maio de 2013 e é usada para construir interfaces de usuário. Uma vantagem do React é que ele usa um estilo declarativo de programação em vez de um estilo imperativo. Enquanto o primeiro especifica ao compilador o que fazer, o último também deve especificar como fazê-lo. Assim, a programação com React resulta em menos código (THAKKAR, 2020). Além disso, segundo os apontamentos do repositório do framework (REACT, 2013), trata-se de uma biblioteca para construção de interfaces de usuário:

- Declarativa: React torna fácil criar UIs interativas. Possível projetar visualizações simples para cada estado em seu aplicativo, e o React atualizará e renderizará com eficiência apenas os componentes certos quando os dados forem alterados. Visualizações declarativas tornam o código mais previsível, mais simples de entender e mais fácil de depurar;

- Com base em componentes: constrói componentes encapsulados que gerenciam seu próprio estado e os compõe para criar interfaces de usuário complexas. Como a lógica do componente é escrita em JavaScript em vez de modelos, pode-se facilmente passar dados por meio de seu aplicativo e manter o estado fora do Document Object Model (DOM);

- Aprenda uma vez, escreva em qualquer lugar: não faz suposições sobre o resto de sua pilha de tecnologia, portanto, pode-se desenvolver novos recursos no React sem reescrever o código existente. O React também pode renderizar no servidor usando Node e alimentar aplicativos móveis usando React Native.

2.4 Firebase

O Firebase é a plataforma de desenvolvimento de aplicativos móveis do Google que dá suporte para a criação, melhoramentos e expansão de aplicativos (SILVA, 2020). Trata-se de uma plataforma de desenvolvimento com diversos produtos voltados na criação de aplicativos móveis e páginas web rapidamente, sem precisar gerenciar a infraestrutura, possuindo funcionalidades como: análises, bancos de dados, mensagens e relatórios de erros.

2.4.1 Firebase Cloud Firestore

O Cloud Firestore é um banco de dados flexível e escalonável para desenvolvimento de dispositivos móveis, web e servidores a partir do Firebase e do Google Cloud Platform. Ele mantém seus dados em sincronia em aplicativos cliente por meio de listeners em tempo real. Além disso, oferece suporte off-line para dispositivos móveis e web para que consiga-se criar aplicativos responsivos que funcionem independentemente da latência da rede ou da conectividade com a Internet (FIREBASE, 2019). Deste modo poderá ser analisado o comportamento da aplicação quando não tiver conexão e quando se reconectar com a Internet.

O Firestore armazena dados NoSQL, orientado a documentos e com a sintaxe JSON. Deste modo o Firestore tem uma estrutura de dados que são divididos em coleções(collections) e documentos(documents) (FERNANDES, 2017).

3 Desenvolvimento das aplicações

Com o objetivo de alcançar os resultados visados para o projeto, foi desenvolvida uma mesma aplicação PWA em ambos os frameworks (Vue.js e React). Além disso, foram utilizadas as Application Programming Interfaces (APIs) da plataforma Google Maps para a realização das operações de geolocalização presentes no aplicativo, e também o banco de dados Cloud Firestore, disponibilizado pela plataforma de desenvolvimento Firebase, com o intuito de fazer a persistência dos dados da aplicação.

Nesta seção são apresentados mais detalhes sobre a aplicação proposta, as APIs e ferramentas utilizadas junto aos frameworks para o desenvolvimento, bem como o detalhamento das aplicações desenvolvidas em cada framework.

3.1 Propósito da Aplicação

A aplicação tem por objetivo buscar pessoas próximas que tenham assuntos de interesse em comum. Para tanto, o usuário da aplicação faz um cadastro e defini assuntos de interesse. Dessa forma, todos os usuários da aplicação podem efetuar

buscas por pessoas que estejam em um perímetro e tenham interesse em determinado assunto.

3.2 Requisitos Funcionais

Para a construção do PWA foram elencados os seguintes requisitos funcionais:

- Cadastrar usuário: A aplicação deve permitir o cadastro dos usuários;
- Realizar Login: A aplicação só estará disponível após login do usuário (pode ser por e-mail e senha ou por conta do Google ou Facebook;
- Escolher Interesses: O usuário pode escolher quais tipos de assunto são de seu interesse, além de poder cadastrar novos, caso não encontre o buscado;
- Buscar: A aplicação utiliza das informações cadastradas para buscar por pessoas com interesses semelhantes e que estejam no raio de alcance informado pelo usuário;
- Delimitar alcance: O usuário pode delimitar o alcance em que a aplicação realizará a busca.

3.3 Requisitos Não-Funcionais

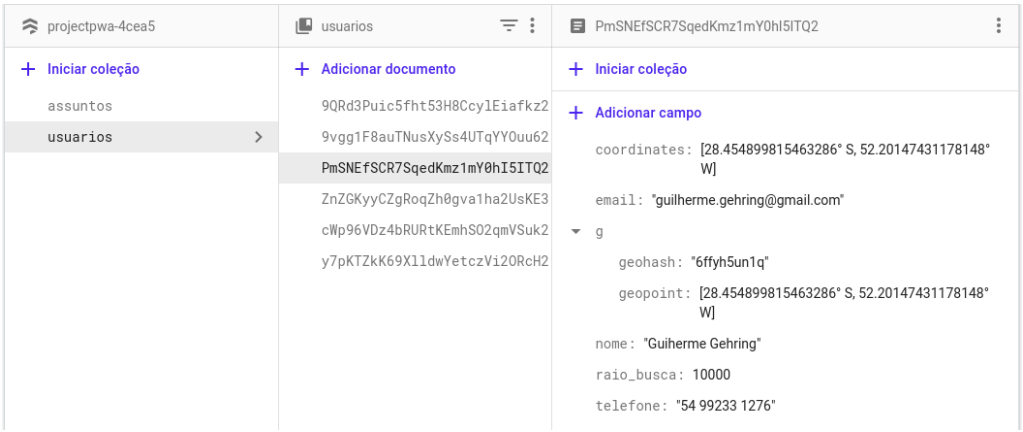
Para a construção da aplicação foram elencados os seguintes requisitos não-funcionais:

- PWA: A aplicação possui as características de um Progressive Web App;
- Armazenamento: A aplicação utiliza a plataforma Cloud Firestore para armazenar os dados dos usuários;
- Login: A aplicação permite o cadastro de usuário utilizando a conta do Google ou Facebook;
- Geolocalização: A aplicação utiliza do recurso de geolocalização do dispositivo para auxiliar na busca de usuários.

3.4 Persistência dos dados

Para o armazenamento dos dados cadastrados foi utilizado o Cloud Firestore. As informações são mantidas nas coleções de *assuntos* e *usuarios*, onde os assuntos ficam contidos em documentos, com a data de criação, o título, a descrição e um array com os identificadores dos usuários. Na coleção de usuários, os documentos possuem os campos nome, email, raio de busca, telefone, coordenadas de geolocalização, um array com as informações de localização e o hash utilizado na busca. A Figura 2 mostra uma estrutura de dados construída na plataforma, onde está uma coleção denominada "usuarios" que possui os documentos e estes por sua vez possuem um conjunto de dados e um valor único que serve como identificador.

Figura 2 – Banco de dados gerado com Firestore



Fonte: Do autor, 2021

3.5 Google Maps Platform

A Plataforma Google Maps é uma Interface de programação das APIs do Google Maps onde por meio desta ferramenta é possível implementar mapas e funcionalidades atreladas ao Google em suas aplicações (BRANDÃO, 2019). Para a sua utilização em páginas web, a plataforma disponibiliza suas funcionalidades através de suas APIs que são oferecidas em três divisões de produtos:

- Maps: Oferece à aplicação mapas com imagens ou elementos interativos. Por meio deste, pode-se definir pela utilização de marcadores para representar pontos de interesse;
- Routes: Fornece opções de rotas para o deslocamento até o local desejado e calcula os tempos de deslocamento atuais ou futuros com base no trânsito em tempo real;
- Places: Ferramenta que permite a busca por locais cadastrados e suas descrições e avaliações por usuários.

Para o desenvolvimento da aplicação foi utilizada a API Maps Javascript, biblioteca que disponibiliza os recursos para a visualização do mapa, assim como a adição de elementos gráficos que podem ser utilizados como marcadores de pontos de interesse, caixas de informações e delimitadores de área. Com esta biblioteca, a aplicação consegue mostrar um mapa, utilizando das coordenadas dos usuários para a criação de marcadores e um círculo representando a àrea na qual o usuário logado delimitou para a busca de pessoas próximas.

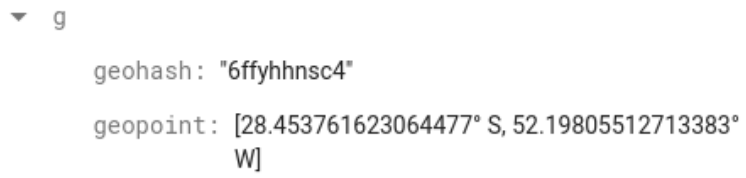
3.6 GeoFirestore

GeoFirestore é uma biblioteca de código aberto que estende a biblioteca Firestore para armazenar e consultar documentos com base em sua localização

geográfica, onde seu principal benefício é a possibilidade de recuperar apenas aqueles documentos dentro de uma determinada área geográfica em tempo real (SOLATI, 2018).

A figura 3 exemplifica o funcionamento da biblioteca, onde com a latitude e longitude desejada é capaz de gerar um hash em que cada caractere representa uma subseção referente ao valor ao lado, assim quanto mais caracteres possuir, maior será a precisão da localização.

Figura 3 – Exemplo de informação gerada pela biblioteca GeoFirestore



Fonte: Do autor, 2021

Utilizando desta biblioteca, a aplicação consegue calcular a distância entre usuários, para verificar quais estão dentro do raio de busca informado pelo usuário logado.

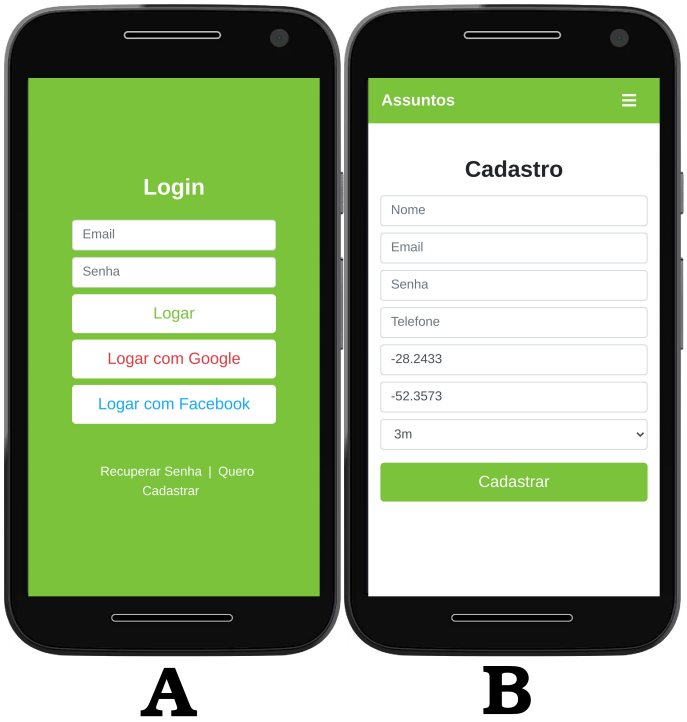
3.7 Características das aplicações

Visando o objetivo de realizar a comparação dos frameworks, foram desenvolvidos os aplicativos em ambos, utilizando dos padrões e dos plugins dispostos para cada um, sendo a proposta da aplicação buscar por pessoas próximas que tenham assuntos de interesse em comum. No desenvolvimento das aplicações buscou-se seguir em ambos os frameworks o mesmo padrão visual de construção das telas, bem como em termos de funcionalidades. A seguir são apresentadas as funcionalidades das aplicações desenvolvidas.

3.7.1 Login e cadastro de usuários

Para realizar o acesso ao aplicativo, foram utilizados os recursos do Firebase Authentication e Firestore, com isso, as aplicações podem realizar a autenticação do usuário utilizando um email e senha, uma conta vinculada ao Google ou uma conta vinculada ao Facebook, além de salvar as demais informações do usuário que são utilizadas pela aplicação. A Figura 4A mostra a tela de login, com as opções de acessar com e-mail e senha, com o Google ou com o Facebook, além das opções de recuperação de senha e cadastro de novos usuários. A Figura 4B mostra a tela de cadastro onde são informados o nome, email, senha, telefone e raio de busca. Ao acessar o cadastro, a aplicação utiliza das informações do dispositivo para adicionar a latitude e a longitude de maneira automática, estas informações podem ser editadas pelo usuário. Destas informações, todas são obrigatórias para o cadastro, com exceção do número de telefone, que é utilizada apenas como uma informação adicional para facilitar a comunicação de usuários.

Figura 4 – Telas de login e de cadastro referentes à aplicação React



Fonte: Do autor, 2021

3.7.2 Tela inicial e assuntos

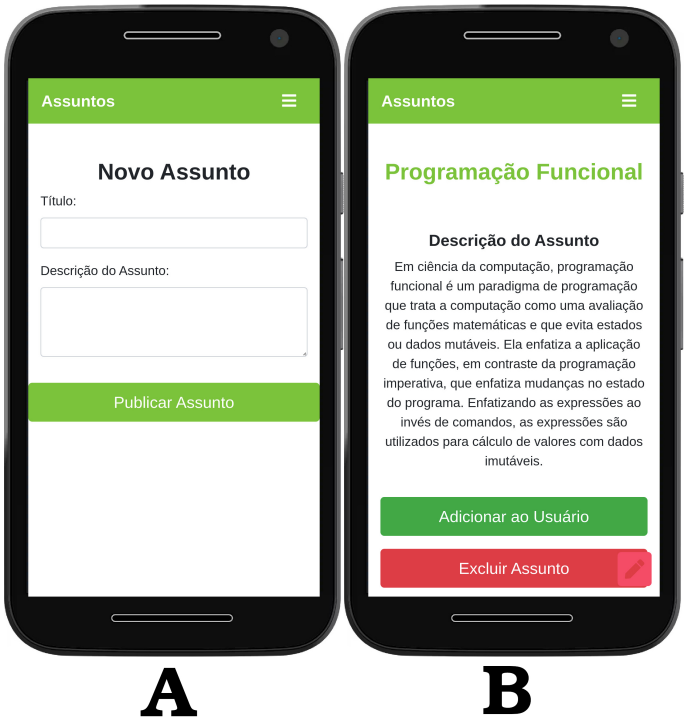
Tanto para a tela inicial, quanto para a listagem e cadastro de assuntos, foram utilizados os recursos do Firebase Firestore, para armazenar os dados relativos a *assuntos*, como o título, a descrição e a lista dos identificadores dos usuários que marcaram determinado assunto como de seu interesse. A Figura 5A mostra a tela referente a página inicial e a Figura 5B a listagem de interesse do usuário, ambas utilizam da mesma estilização e se diferem no título apresentado e no fato que a tela inicial apresenta apenas os assuntos que foram escolhidos pelo usuário.

Figura 5 – Tela inicial e de listagem de assuntos referentes à aplicação React

Fonte: Do autor, 2021

A Figura 6A demonstra a tela de criação e edição de um assunto e a Figura 6B mostra a visualização do assunto selecionado para poder sinalizar interesse pelo mesmo.

Figura 6 – Tela de visualização e edição de assuntos referentes à aplicação Vue



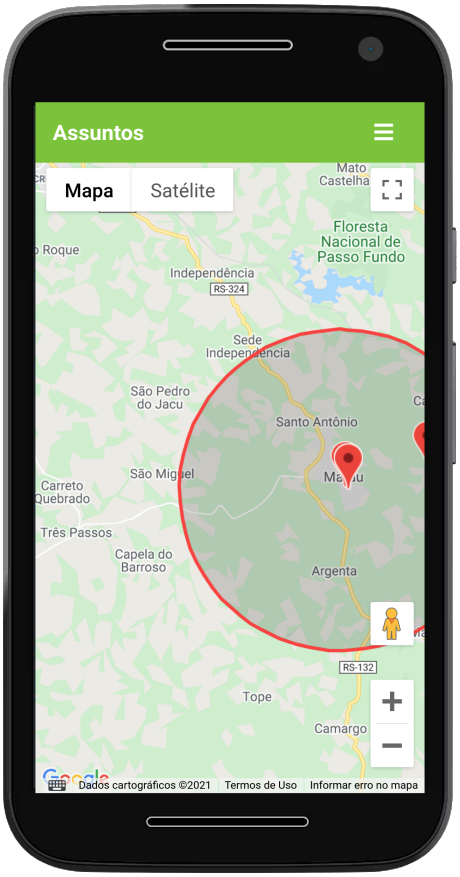
Fonte: Do autor, 2021

3.7.3 Mapa de busca

Referente ao mapa de busca de usuários, ambos os aplicativos utilizam os recursos do Firebase Firestore, do Google Maps e GeoFirestore. Já para a criação do mapa e a exibição dos marcadores de usuários, cada framework utiliza de um plugin próprio, onde tando o *react-google-maps* quanto o *vue2-google-maps* possuem funcionalidades parecidas, tendo suas principais diferenças no código escrito, cada um utilizando dos padrões disponibilizados pelo framework.

A Figura 7 mostra a tela onde são utilizadas as coordenadas do usuário e o raio de busca cadastrado para formar uma circunferência, demonstrando a região delimitada para a busca.

Figura 7 – Tela de busca de usuários referente a aplicação Vue



Fonte: Do autor, 2021

4 Avaliação dos aplicativos

Nesta seção são apresentadas as avaliações dos PWAs a partir da ferramenta Lighthouse, bem como algumas considerações sobre os frameworks utilizados no desenvolvimento. Para a avaliação, os aplicativos foram hospedados na plataforma Firebase Hosting e submetidos a ferramenta Lighthouse que gerou os relatórios para análise.

O Lighthouse é uma ferramenta automatizada que analisa aplicativos e páginas da web, coletando métricas de desempenho e ideias sobre as práticas do desenvolvimento (LIGHTHOUSE, 2019). A ferramenta foi utilizada na avaliação dos PWAs construídos nos frameworks. Com tal ferramenta é possível verificar aspectos de desempenho e atendimento dos elementos que devem estar presentes em um PWA. Ele pode ser instalado como uma extensão do navegador e é capaz de executar uma série de revisões na aplicação com relação aos seguintes tópicos:

- Performance: Avalia o desempenho e busca por oportunidades para acelerar o carregamento da aplicação;

- Acessibilidade: Avalia se todos os usuários podem acessar o conteúdo e realizar a navegação com eficácia;
- Boas práticas: Avalia a integridade do código, de acordo com as práticas recomendadas pelo Google;
- Search Engine Optimization (SEO): Avalia a maneira em que os mecanismos de busca podem encontrar o conteúdo disponibilizado.

Além disso, o Lighthouse possui a função de validar os aspectos de um Progressive Web App, desta maneira, caso esta metodologia esteja sendo utilizada, a ferramenta é capaz de validar os seguintes aspectos:

- O manifesto do aplicativo da web e o service worker atendem aos requisitos de instalabilidade;
- O tráfego de informações utiliza de https;
- O aplicativo possui uma tela inicial personalizada;
- O aplicativo possui uma cor de tema para a barra de endereço;
- O conteúdo está dimensionado corretamente para a janela de visualização;
- O aplicativo possui uma tag <meta name = "viewport"> com largura ou escala inicial;
- O aplicativo fornece um ícone padrão para dispositivos iOS;
- O manifesto tem um ícone mascarável para que o mesmo não fique distorcido.

Quando a ferramenta finaliza o teste, um relatório é gerado com dicas úteis para implementar, juntamente com uma pontuação que representa seu desempenho atual (CHEMOUL, 2020).

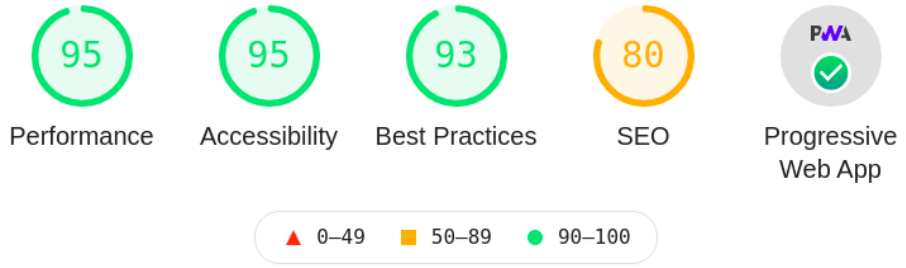
4.1 Resultados sobre o aplicativo React

Para realizar a avaliação com o Lighthouse, é recomendado que o aplicativo seja aberto utilizando de uma aba anônima, assim evitando que dados armazenados possam afetar no desempenho.

Seguindo esta orientação, para a aplicação desenvolvida em React, a ferramenta retornou as pontuações apresentadas na Figura 8, onde as pontuações de performance, acessibilidade e boas práticas atingiram pontuações superiores a 90 pontos, além de uma avaliação positiva sobre os requisitos de um Progressive Web App.

No entanto o resultado referente ao SEO foi menor que os outros. Segundo a análise, o documento não possui uma meta tag de descrição (que pode ser utilizada nos motores de busca) e não há um arquivo "robots.txt" válido, assim os rastreadores podem não conseguir entender como deseja-se que o site seja rastreado ou indexado.

Figura 8 – Resultado referente ao PWA desenvolvido com o framework React

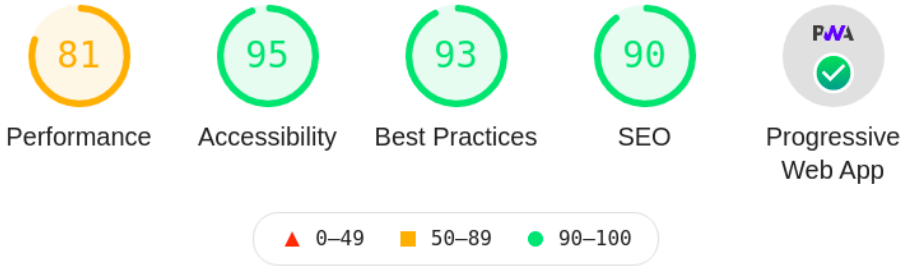


Fonte: Do autor, 2021

4.2 Resultados sobre a aplicação Vue

A avaliação do PWA construído com o framework Vue também foi submetido ao Lighthouse, seguindo a orientação sobre o uso da navegação anônima, que retornou as pontuações apresentadas na Figura 9.

Figura 9 – Resultado referente ao PWA desenvolvido com o framework Vue



Fonte: Do autor, 2021

De acordo com as pontuações obtidas pela ferramenta, a acessibilidade, as boas práticas e o SEO atingiram pontuações superiores a 90 pontos, além de uma avaliação positiva sobre os requisitos de um Progressive Web App. No entanto, o resultado referente à performance foi inferior aos outros listados.

Segundo os apontamentos da ferramenta, algumas medidas que podem ser tomadas são:

- Remover arquivos JavaScript não utilizados. Entretanto, a análise aponta a remoção de arquivos gerados automaticamente durante a build da aplicação;
- Verificar se os textos permanecem visíveis durante o carregamento da página;
- Aumentar o tempo do cache, o que pode acelerar as visitas repetidas ao PWA.

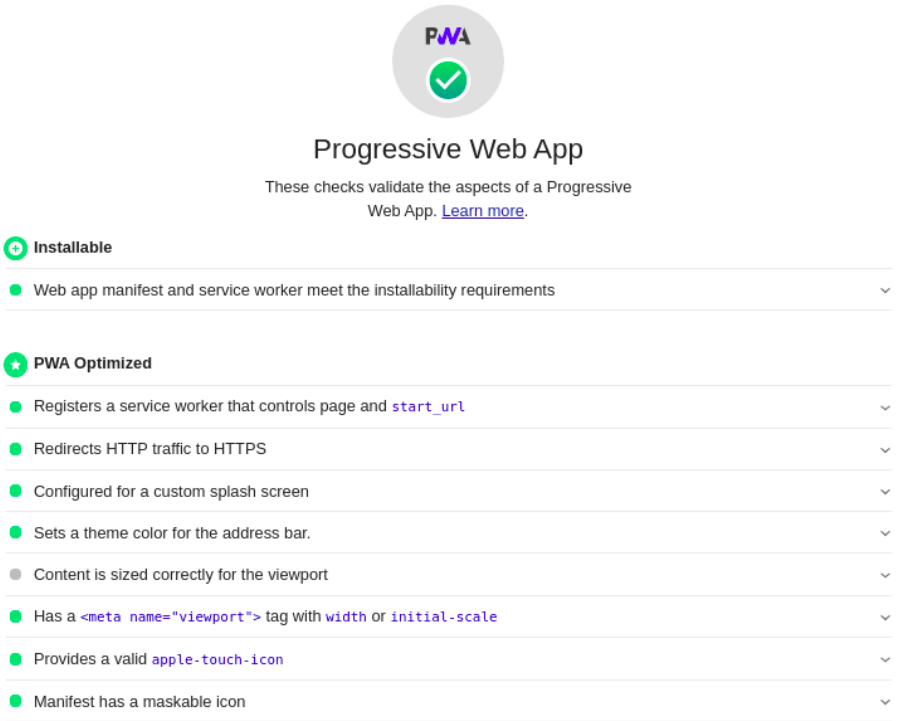
4.3 Considerações sobre os frameworks

Com a avaliação realizada, foi possível observar resultados semelhantes nos tópicos de Acessibilidade, Boas Práticas e Características de PWA. Contudo a análise mostrou diferenças em outros pontos, onde a aplicação construída em React atingiu uma pontuação superior no quesito performance quando comparada com o resultado da aplicação Vue. Já a aplicação construída em Vue.js atingiu uma pontuação superior no quesito SEO em relação ao app React.

Com as notas apresentadas pela ferramenta Lighthouse, pode-se apontar que a biblioteca React é mais recomendada onde o desempenho seja um ponto de maior importância para o produto final. Por outro lado, caso o objetivo buscado vise obter uma maior facilidade de otimização para mecanismos de busca, o framework Vue.js pode ser mais indicado.

Referentes à análise sobre as características de um PWA que é vista na Figura 10, ambas as aplicações alcançaram a mesma avaliação, sem conter nenhuma consideração negativa, no entanto, a ferramenta não conseguiu verificar se o conteúdo está dimensionado corretamente para a janela de visualização, por este motivo, não foi possível retornar nenhum tipo de sugestão, o que tornou este ponto inconclusivo.

Figura 10 – Resultado referente à avaliação do PWA construído em React, realizado pelo Lighthouse



Fonte: Do autor, 2021

Considerações finais

Considerando que o desenvolvimento de PWAs ainda é algo recente e que possui a tendência de crescimento, o trabalho teve como propósito a avaliação dos frameworks Vue.js e React, a partir da construção de uma aplicação com conceitos de PWA. Estes aplicativos possibilitam ao usuário cadastrar assuntos de interesse e buscar por pessoas próximas que tenham assuntos em comum. Para auxiliar no desenvovimento e na comparação, foram utilizadas as plataformas Google Firebase e Google Maps.

Utilizando das respectivas bibliotecas Javascript em conjunto com as demais ferramentas e padrões para a construção de um Progressive Web App, foi possível verificar a maneira como a escolha de um framework pode impactar no desempenho e na qualidade do produto final.

Com o auxílio da ferramenta de análise Lighthouse, pode-se verificar que ambas as aplicações cumpriram com os requisitos de um PWA e alcançaram as pontuações 95/100 em Acessibilidade e 93/100 em Boas Práticas. Com isso, as principais diferenças ficaram referentes a Performance e ao SEO, onde o primeiro obteve um resultado maior para a aplicação React, enquanto o segundo apontou um resultado maior para a aplicação Vue.js. Desta maneira, a escolha de um framework ideal para a construção de um PWA dependerá da aplicação pretendida e quais dos pontos necessitarão de maior atenção para a construção da mesma.

Além destes pontos avaliados, outros aspectos podem ser levados em consideração na escolha do framework, entre alguns deles podem estar a facilidade de se buscar por plugins, atividade da comunidade para manter o framework e a maneira como os dados são tratados pela aplicação. Sobre o desenvolvimento dos aplicativos, devido ao maior crescimento da comunidade, foi observado que a biblioteca React possui alguns plugins mais atualizados pela comunidade.

Outro ponto observado que difere entre as aplicações, está relacionado com o fato de como cada uma trabalha com a reatividade dos dados. A reatividade dos dados baseia-se na capacidade de uma variável sinalizar que foi alterada. Construindo uma cadeia de reações, um fluxo no código, é possível que mudando uma variável a mudança se propague tornando a aplicação reactiva. Neste quesito, o Vue consegue alterar os dados e a camada visual se atualiza, algo que o React acaba utilizando de extensões para poder obter o mesmo resultado. Este apontamento, em conjunto com a semelhança da sua sintaxe de templates baseada em HTML, contribuem para que a curva de aprendizado do framework seja reduzida em comparação com o React.

Com isso, para o desenvolvimento de melhorias futuras, propõem-se a avaliação detalhada dos PWAs, visando corrigir os apontamentos feitos pelo Lighthouse e, consequentemente, melhorar os pontos aos quais cada framework possuiu menor pontuação na comparação. Além disso, referente as características de um PWA, pode-se buscar uma verificação mais detalhada sobre a análise da janela de visualização, para obter uma conclusão sobre este aspecto.

Progressive Web Apps: Comparative analysis of JavaScript frameworks for development

Guilherme Gehring[‖] Anubis Graciela de Moraes Rossetto [**]

2021

Abstract

This paper presents a comparative analysis of JavaScript frameworks for Progressive Web Apps (PWA) development involving the Vue.js and React libraries. For this, the work discusses the development of the same application in both frameworks, incorporating geolocation services and tools, data storage and application hosting. In addition to the development of applications in each framework, an evaluation of the applications is presented using the Google Lighthouse tool, which analyzes applications and web pages, collecting performance metrics and ideas about development practices, checking aspects performance and fulfillment of the elements that must be present in a PWA.

Palavras-chave: Progressive Web App. React. Vue.js. JavaScript Frameworks.

Referências

BRANDÃO, B. *Google Maps Platform*. 2019. Disponível em: <https://maplink.global/blog/o-que-e-google-maps-platform/>. Acesso em: 28 nov 2020. Citado na página 8.

[‖]<guilhermegehring.pf178@academico.ifsul.edu.br>
[**]<anubis.rossetto@passofundo.ifsul.edu.br>

CHEMOUL, P. *A Complete Guide to Google Lighthouse*. 2020. Disponível em: <https://seoexpertpatrick.com/google-lighthouse/>. Acesso em: 26 nov 2020. Citado na página 14.

FERNANDES, R. P. *O que é Cloud Firestore?* 2017. Disponível em: <https://medium.com/android-dev-moz/firestore-8d225062ffe8>. Acesso em: 16 nov 2020. Citado na página 6.

FIREBASE, G. D. *Cloud Firestore*. 2019. Disponível em: <https://firebase.google.com/docs/firestore>. Acesso em: 19 nov 2020. Citado na página 6.

GAUNT, M. *Service Workers: an Introduction*. 2019. Disponível em: <https://developers.google.com/web/fundamentals/primers/service-workers>. Acesso em: 18 nov 2020. Citado na página 4.

GONSALVES, M. Evaluating the mobile development frameworks apache cordova and flutter and their impact on the development process and application characteristics. 2018. Citado na página 2.

LEVLIN, M. Dom benchmark comparison of the front-end javascript frameworks react, angular, vue, and svelte. Åbo Akademi, 2020. Citado na página 5.

LIGHTHOUSE, G. C. *Lighthouse*. 2019. Disponível em: <https://github.com/GoogleChrome/lighthouse>. Acesso em: 26 nov 2020. Citado na página 13.

MUMAN, J. Progressive web apps: An optimistic approach to traditional application development. *International Journal of Scientific Development and Research*, p. 143–146, 2021. Citado na página 2.

PATEL, N. *PWA (Progressive Web App): O Que É e Como Criar o Seu (+3 Exemplos)*. 2019. Disponível em: <https://neilpatel.com/br/blog/pwa-o-que-e>. Acesso em: 18 nov 2020. Citado na página 4.

REACT, F. *React*. 2013. Disponível em: <https://github.com/facebook/react>. Acesso em: 17 nov 2020. Citado na página 5.

ROGOJAN, B. *Vue vs React: Which is the Better Framework?* 2019. Disponível em: <https://buttercms.com/blog/vue-vs-react-which-is-the-better-framework>. Acesso em: 01 dez 2020. Citado na página 2.

RUSSELL, A.; BERRIMAN, F. *Progressive Web Apps: Escaping Tabs Without Losing Our Soul*. 2015. Disponível em: <https://infrequently.org/2015/06/progressive-apps-escaping-tabs-without-losing-our-soul/>. Acesso em: 04 nov 2020. Citado 2 vezes nas páginas 2 e 3.

SILVA, E. *Firebase: o que é e quando usar no desenvolvimento mobile?* 2020. Disponível em: <https://blog.geekhunter.com.br/firebase-o-que-e-e-quando-usar-no-desenvolvimento-mobile>. Acesso em: 18 out 2020. Citado na página 6.

SOLATI, M. *geofirestore*. 2018. Disponível em: <https://geofirestore.com>. Acesso em: 14 set 2021. Citado na página 9.

SUSHKO, D. *Eight Reasons to Consider Progressive Web App Development*. 2017. Disponível em: <https://da-14.com/blog/eight-reasons-consider-progressive-web-app-development>. Acesso em: 01 dez 2020. Citado na página 1.

THAKKAR, M. *Building React Apps with Server-Side Rendering*. [S.l.]: Apress, Berkeley, CA, 2020. Citado na página 5.

VU, L. *PWA vs SPA: Alike but Different*. 2019. Disponível em: <https://www.simicart.com/blog/pwa-vs-spa/>. Acesso em: 15 nov 2020. Citado na página 4.

VUE.JS. *Introdução - Vue.js*. 2020. Disponível em: <https://br.vuejs.org/v2/guide/index.html>. Acesso em: 04 nov 2020. Citado na página 5.