ANÁLISE DE DESENVOLVIMENTO DE APLICATIVOS USANDO PLATAFORMAS NATIVAS E MULTIPLATAFORMAS

Lucas Morais Silva[1]
Messias Rafael Batista[2]

RESUMO

Introduzir o contexto geral sobre uma análise de desenvolvimento de aplicativos é o objetivo geral deste trabalho, assim, uma definição baseada na teoria bibliográfica foi determinada com base na utilização de plataformas nativas e multiplataformas. A revisão bibliográfica narrativa foi a metodologia utilizada para desenvolver o tema, e foi feita por meio de métodos qualitativos e descritivos. Para selecionar os autores citados, foram pesquisados artigos científicos nas bases de dados *Scielo* e *Google* Acadêmico, bem como livros etc., A lista bibliográfica foi desenvolvida com base em critérios analíticos para o título e uma breve leitura do resumo de cada trabalho. A divisão do período foi determinada entre os últimos 10 anos. Os referenciais usados como base para a criação da discussão acerca de plataformas nativas e multiplataformas apresentam muito conhecimento sobre o assunto. Dessa forma, a pesquisa é rica, responde à pergunta de pesquisa originalmente definida.

Palavras-chaves: Plataformas nativas. Multiplataformas. Vantagens e desvantagens.

ABSTRACT

Introducing the general context of an application development analysis is the general objective of this work, thus, a definition based on bibliographic theory was determined based on the use of native platforms and cross-platforms. The narrative bibliographic review was the methodology used to develop the theme, and it was carried out using qualitative and descriptive methods. To select the cited authors, scientific articles were searched in the Scielo and Google Scholar databases, as well os books etc. The bibliographic list was developed based on analytical criteria for the title and a brief reading of the abstract of each work. The division of the period was determined between the last 10 years. The references used as a basis for creating the discussion about native platforms and cross-platforms present a lot of knowledge on the subject. In this way, the search is rich, it answers the originally defined research question.

Keywords: Native platforms. cross-platform. Advantages and disadvantages.

1 INTRODUÇÃO

O mercado de dispositivos móveis está crescendo, com 3,2 bilhões de usuários em todo o mundo, e mais de 80% do tempo do dispositivo é gasto em aplicativos. Portanto, o mercado de aplicativos também deve crescer: o número de downloads de aplicativos móveis (BERA; MINE; LOPES, 2015).

Atualmente, os sistemas operacionais *Android* e iOS ocupam conjuntamente 98,82% do mercado de dispositivos móveis (celulares e *tablets*), tornando-se o foco do desenvolvimento de aplicativos. No entanto, esses sistemas possuem peculiaridades próprias, além de linguagens

[1] Graduande do Curso de.... E-mail: ....
[2] Graduande do Curso de.... E-mail: ....

de programação nativas e diferentes kits de desenvolvimento de software (SDKs), que exigem que os aplicativos desenvolvidos para esses dois sistemas sejam acessíveis à maioria do público,

separadamente é necessário desenvolver duas versões do mesmo aplicativo em linguagens diferentes, dobrando o esforço, custo e/ou tempo de desenvolvimento (SILVIA; SOUSA, 2019).

Além do desenvolvimento inicial, manter dois projetos iguais também exige mais esforço do que apenas um projeto. Como solução para esse problema, surgiram *frameworks* e ferramentas de desenvolvimento multiplataforma, também conhecidos como híbridos. Essas ferramentas são projetadas para usar uma única base de código para todas as plataformas de destino, reduzindo o esforço necessário para desenvolver aplicativos (FERREIRA; MELO, 2017).

Introduzir o contexto geral sobre uma análise de desenvolvimento de aplicativos é o objetivo geral deste trabalho, assim, uma definição baseada na teoria bibliográfica foi determinada com base na utilização de plataformas nativas e multiplataformas.

Para atingir esse objetivo geral e demonstrar o domínio do assunto, foi desenvolvido os seguintes objetivos específicos:

- Evidenciar uma contextualização geral acerca das plataformas de desenvolvimento mobile nativos e multiplataformas;
- Classificar as diferenças das plataformas nativas e multiplataformas;
- Determinar as vantagens e desvantagens das plataformas nativas;
- Caracterizar as vantagens e desvantagens das multiplataformas.

A partir destes objetivos determinados, e considerando o que precisa ser apresentado no desenvolvimento do trabalho para atingir essas classificações, pode-se determinar que o problema de pesquisa é: como se dá a análise de desenvolvimento de aplicativos usando plataformas nativas e multiplataformas?

A análise apresentada enriquece os resultados da pesquisa contemporânea ao argumentar sobre a análise de desenvolvimento de aplicativos usando plataformas nativas e multiplataformas, levando em conta o maior referencial científico sobre o tema. Dadas as lacunas levantadas e a resolução dessas lacunas, apresentam benefícios tanto a nível académico como profissional e social como um todo. Pois, dessa forma, além das respostas às questões colocadas, atribui referências a novas questões e a base para o surgimento de experimentos. Em

seguida, contribui para a sociedade e atribui melhorias aos processos comumente utilizados por grandes públicos. Além disso, contribui para o campo acadêmico, pois enriquece ainda mais o acervo científico de pesquisas sobre esse tema e levanta questões que precisam ser discutidas em pesquisas futuras.

A metodologia utilizada para desenvolver o tema, foi feita por meio de métodos qualitativos e descritivos. Para selecionar os autores citados, foram pesquisados artigos científicos nas bases de dados *Scielo* e *Google* Acadêmico, bem como livros etc.

Segundo Lakatos e Marconi (2017), materiais complementares publicados por fontes confiáveis, assim como as próprias revisões bibliográficas, possuem alto grau de confiabilidade e atestação do que suas fontes cobrem, o que fornece uma base para o uso de dados e relatórios, e legitimidade as pessoas que o usam.

A lista bibliográfica foi desenvolvida com base em critérios analíticos para o título e uma breve leitura do resumo de cada trabalho. A divisão do período foi determinada entre os últimos 10 anos.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção foi detalhado a análise do desenvolvimento usando plataformas nativas e multiplataformas, mostrando as particularidades de cada aplicação e quais as suas vantagens e desvantagens.

2.1 PLATAFORMAS DE DESENVOLVIMENTO *MOBILE* NATIVOS E MULTIPLATAFORMAS.

WebApps são aplicativos desenvolvidos para a *web* utilizando as tecnologias HTML5, CSS3 e *Javascript*. Esses aplicativos podem ter funcionalidade e comportamento semelhantes aos aplicativos desenvolvidos localmente, mesmo que estejam vinculados à *web*. A principal vantagem dessa solução é que os aplicativos podem ser projetados independentemente do tipo de dispositivo móvel e plataforma a ser utilizada (EL-KASSAS *et al*., 2015).

O CSS é parte essencial na criação de páginas da *web*, pois é responsável pela apresentação visual dos elementos HTML, ou seja, pela formatação das informações criadas pelo HTML. Essas informações podem ser representadas por elementos como texto, imagens, áudio, vídeo ou outros elementos criados. Junto com o HTML5 veio uma nova versão do padrão CSS, CSS3 (AHTI; HYRYNSALMI; NEVALAINEN, 2016).

Esta versão apresenta novos recursos, como suporte de fonte integrado, novos seletores, divisão de elementos em colunas em layouts e melhorias visuais, como transparência, sombras, cantos arredondados, animações, gradientes e transições. Muitas melhorias visuais já existem na criação da página, mas seu uso tornou-se mais trabalhoso para os desenvolvedores devido à falta de recursos nativos da linguagem, incluindo a necessidade de elementos HTML adicionais. Com a liberação local desses recursos, o código se torna mais acessível aos desenvolvedores, melhorando a manutenibilidade, reduzindo alterações na renderização de imagens e carregamentos de página mais rápidos (BOTELLA; ESCRIBANO; PENALVER, 2016).

Ao desenvolver aplicativos *web*, é importante avaliar a velocidade de um aplicativo multiplataforma em comparação com um aplicativo nativo.

Os aplicativos *web* não são viáveis para aplicativos que precisam processar imagens e jogos, como jogos 3D por exemplo, porque seu tempo de resposta precisar ser curto.

Ao usar um aplicativo nativo, a interface sempre terá a mesma apresentação e comportamento. Em *WebApps*, porém, é preciso levar em conta que as

funcionalidades podem variar, dependendo do navegador utilizado. Em *WebApps*, o acesso aos recursos do dispositivo é restrito. Por exemplo, recursos como câmeras, acelerômetros, agendas de endereços e geolocalização não são acessíveis em aplicações *web* (BOTELLA; ESCRIBANO; PENALVER, 2016).

Como solução alternativa, a maioria dos navegadores possui APIs disponíveis para funcionalidade de geolocalização. Além dessa API, o *World Wide Web Consortium* (W3C), que mantém os padrões da *web*, mantém um grupo que cria APIs para acessar outros recursos locais nos dispositivos (SHAKSHUKI *et al.*, 2013).

O aplicativo da *web* também não fornece suporte offline completo. Embora o HTML5 tenha a capacidade de funcionar *offline*, nem todos os navegadores oferecem suporte a essa opção. Essa solução também conta com a conexão com servidores de aplicação, onde o acesso pode ocorrer de forma errática ou ficar indisponível (AHTI, HYRYNSALMI; NEVALAINEN, 2016).

O desenvolvimento multiplataforma é a prática de desenvolver softwares que podem ser usados em diferentes plataformas, mesmo com arquiteturas e APIs nativas completamente diferentes. Os aplicativos multiplataforma usam elementos nativos para fornecer a melhor experiência ao usuário, independentemente do dispositivo usado. Ou seja, eles compartilham a

mesma base de código em todas as plataformas, mas podem ter suas próprias diferenças (BRITO *et al*., 2018).

Com o surgimento de várias plataformas tecnológicas, surgiram oportunidades de mercado significativas. Manter contato próximo com seus usuários finais pode aumentar muito suas chances de consumir seus produtos e serviços, aumentando assim sua lealdade. No entanto, ter software em múltiplas plataformas não é simples e vários aspectos devem ser considerados, como as tecnologias suportadas pelas plataformas relevantes e a forma como os usuários interagem com elas (CUMMAUDO; VASA; GRUNDY, 2019).

Poucos aplicativos desenvolvidos para uma plataforma podem ser executados em uma plataforma diferente sem alterações ou novos desenvolvimentos. Esse é um dos fatores que muitos desenvolvedores de software enfrentam ao criar aplicativos que podem ser executados em vários dispositivos. *frameworks* como *Flutter* e *React Native* encontraram uma solução e um dos fatores que impulsionam a popularidade do desenvolvimento multiplataforma (BERA; MINE; LOPES, 2015).

Essas técnicas ajudam a economizar tempo e dinheiro. Vários autores sugeriram algumas vantagens e desvantagens. O desenvolvimento multiplataforma usa uma linguagem comum, evitando a necessidade de especialização em linguagens específicas do sistema. Além disso, não há necessidade de fazer um estudo detalhado de cada plataforma. As ferramentas de plataforma cruzada usam apenas suas próprias APIs e recursos (RODRÍGUEZ; BALDRICH, 2015).

O desenvolvimento multiplataforma tem como sua maior vantagem poder proporcionar um alcance maior do aplicativo para o público em um tempo menor em relação aos nativos. Existem muitas plataformas para criação dos projetos, porém, cada plataforma carrega consigo suas funções e características. Um exemplo disso é o mundialmente conhecido *Facebook* que foi criado com a ferramenta *React Native*, onde primeiramente suporta o IOS e posteriormente o *Android*.

No entanto, os componentes no *React Native* são simulados nativamente usando o JavaScript Core como o link entre o JSX e a linguagem. Essa conexão omite uma camada de aplicativo que permite executar APIs de renderização Java e Objective-C. A estrutura possui três threads principais: a *Shadow Queue* (onde o layout é manipulado), o *thread* principal (onde ocorre todo o processo de renderização da interface do usuário) e o *thread JavaScript* que executa o *script* (AHTI; HYRYNSALMI; NEVALAINEN, 2016).

2.2 AS DIFERENÇAS DAS PLATAFORMAS NATIVAS E MULTIPLATAFORMAS

Os avanços na tecnologia começam a forçar empresas e desenvolvedores a buscarem maior agilidade na entrega à medida que novos produtos são lançados. Em resposta a esta potencial problema no mercado atual, encontrar métodos de trabalho e ferramentas que facilitem este processo tornou-se o foco da empresa.

Como resultado, surgiu uma forma de desenvolvimento que possibilita a entrega de software para todo o público-alvo de forma mais ágil, ou seja, desenvolvimento multiplataforma (BERRNARDES; MIYAKE, 2016).

Quando se fala sobre desenvolvimento multiplataforma, se quer dizer desenvolver um único código-fonte que pode ser compilado em código nativo para muitos sistemas operacionais diferentes. Isso significa que o mesmo código pode ser usado em diferentes sistemas, como *Windows*, *Android*, iOS e *Cloud*. O objetivo de qualquer empresa de desenvolvimento mobile é atingir o maior número possível de usuários disponibilizando o mesmo aplicativo para diferentes plataformas (PAPAJORGJI, 2015).

O desenvolvimento multiplataforma refere-se ao desenvolvimento de software que pode ser usado em diferentes plataformas, sejam dispositivos móveis como *Android* e iOS, ou *tablets*,

smart TVs etc. As soluções multiplataforma possibilitam a implementação de aplicativos que podem ser executados em diferentes plataformas (AHMAD *et al*., 2018).

Com o desenvolvimento da tecnologia multiplataforma, diversos *frameworks* surgiram no mercado para auxiliar os desenvolvedores. Abaixo está uma descrição de alguns dos principais *frameworks* do mercado (BIØRN-HANSEN; GRØNLI; GHINEA, 2018).

• *Cordova/PhoneGap*: Permite a criação de aplicativos híbridos para plataformas iOS, *Android*, *Blackberry*, *Symbian*, Bada e *Windows Phone*. Por meio do componente *webview* nativo de cada plataforma, o *PhoneGap* pode executar aplicativos desenvolvidos usando tecnologias da web. Ele fornece várias APIs *JavaScript* que permitem o uso de recursos nativos do dispositivo, como uma bússola para escrever e ler arquivos em unidades de armazenamento e geolocalização (SYDOW, 2020);

• *Ionic*: Assim como o *PhoneGap* suporta o desenvolvimento de aplicações híbridas com as mesmas funcionalidades, ele utiliza o *framework* Angular por padrão, com suporte ngCordova para criação de aplicações *JavaScript*;

• *Sencha Touch*: Baseado nas bibliotecas *JavaScript ExtJS*, *jQTouch* e Raphaël, o *Sencha Touch* é um dos *frameworks* mais usados. Permite criar aplicações web para plataformas *Android*, iOS e *Blackberry*, possui IDE própria, *Sencha Architect* e extensão *Sencha Animator* para criação de animações com CSS3;

• *Mono*: *MonoTouch* e *Mono* para *Android* são dois produtos da *Xamarim*. Essas duas ferramentas possibilitam a criação de aplicativos nativos para as plataformas iOS e *Android* utilizando a plataforma .NET (mais precisamente a linguagem C#) (SMUTNÝ, 2012).

• *Titanium Mobile*: Criado pela *Appcelerator* e usando seu próprio IDE, *Titanium Studio*, o *Titanium Mobile* suporta a criação de aplicativos web e híbridos para plataformas iOS, *Android* e *Blackberry*. Além de usar JavaScript como linguagem principal, também permite que PHP, *Python* e *Ruby* sejam usados no desenvolvimento de aplicativos. Também possui sua própria *API JavaScript* para utilizar a funcionalidade nativa fornecida pela plataforma (URSINO; CAPANNA, 2015).

• *Adobe AIR*: É a versão desktop do *Adobe Flash Player*, agora compatível com as plataformas móveis iOS, *Android* e *Blackberry*. *Adobe Flex* é um *framework* que permite a criação de aplicativos AIR usando *Action Script* e MXML (MATOS; SILVA, 2016).

2.3 VANTAGENS E DESVANTAGENS DAS PLATAFORMAS NATIVAS

Um aplicativo criado em uma plataforma nativa é projetado para ser executado em uma plataforma específica. Os arquivos resultantes da compilação do aplicativo devem ser instalados diretamente no sistema operacional, como apresentação, processamento e armazenamento de dados.

Metodologias de desenvolvimento de software e escolha de uma sobre a outra quando indeciso. A razão para tais dúvidas surge porque cada um deles tem vantagens e desvantagens. Isso faz parte do caráter deles. Portanto, é importante entender suas propriedades (MARDAN, 2015)

Conhecendo-os, se pode extrair o melhor método de cada método. Mas afinal, o que é uma metodologia de desenvolvimento mobile? As metodologias de desenvolvimento de aplicativos para dispositivos móveis nada mais são do que os procedimentos que serão utilizados na criação do aplicativo e o produto será a aplicação do método em questão (AHTI; HYRYNSALMI; NEVALAINEN, 2016).

Por exemplo, para fazer um aplicativo usando uma abordagem híbrida, deve-se usar um pacote de aplicativos e, como resultado desse processo, obterá um aplicativo híbrido (BERA; MINE; LOPES, 2015).

Dado que um determinado aplicativo nativo é executado em uma plataforma específica, uma das primeiras decisões de design é escolher em qual plataforma ele será executado. Em alguns casos, as empresas envolvidas decidiram cobrir as principais plataformas, mas nem sempre isso é possível. Pois além do alto custo, pode não ser necessária uma certa participação de mercado (BERNARDES; MIYAKE, 2016).

Quando se diz que é caro, não se está falando apenas do fato de que o mesmo aplicativo deve ser desenvolvido para diferentes plataformas, mas também da curva de aprendizado de cada plataforma. Se o trabalho for para um único desenvolvedor, significa que ele terá que aprender cada plataforma, e aprender apenas uma plataforma significa muito tempo (BIØRN-HANSEN; GRØNLI; GHINEA, 2018).

Embora esse seja um método mais caro, muitos fatores mantêm as empresas optando por esse método. Por exemplo, com aplicativos nativos. É possível a manipulação de dados off-line, ou seja, armazenados em um banco de dados no próprio dispositivo, o que permite que o software local continue funcionando mesmo onde não há acesso à internet (AHMAD *et al*., 2018).

Embora isso também seja possível para web baseada em HTML5, essa ideia de manipulação de dados offline é mais madura em métodos nativos. Em aplicativos nativos, o hardware presente no dispositivo, como telefone, câmera, microfone, *bluetooth* e acelerômetro, pode ser bem aproveitado, o que pode se tornar mais útil, fácil e interagir com esses tipos de aplicativos (AHTI; HYRYNSALMI; NEVALAINEN, 2016).

Com isso, os desenvolvedores têm em mãos um leque de possibilidades para poderem fazer diversos tipos de aplicações sem grandes restrições de hardware. Dependendo da situação, pode ser necessária a utilização desses recursos, para os quais essa abordagem se torna uma forte candidata. Afinal, embora outros métodos suportem algumas funcionalidades, ainda existem muitas limitações a esse respeito (HEITKöTTER; HANSCHKE; MAJCHRZAK, 2012).

2.4 VANTAGENS E DESVANTAGENS DAS MULTIPLATAFORMAS

O desenvolvimento de plataforma híbrida consiste em fazer com que o código seja executado no sistema operacional principal. Nativo, por outro lado, significa construir aplicativos para apenas um tipo de sistema (MARDAN, 2015).

Uma abordagem de desenvolvimento de plataforma híbrida oferece aos desenvolvedores de aplicativos móveis a oportunidade de implantar aplicativos em várias plataformas móveis a partir de uma única base de código. Eles permitem que se escreva código uma vez e execute-o em qualquer lugar. Sem esses métodos, os desenvolvedores teriam que desenvolver e manter bases de código separadas para cada plataforma em que desejam oferecer seus produtos. Essa abordagem é chamada de desenvolvimento nativo (AHMAD *et al.*, 2018).

Atualmente, o mercado de telefonia móvel é dominado pelas plataformas *Android* do *Google* e iOS da *Apple*. O desenvolvimento nativo do *Android* usando

Java e/ou *Kotlin* são duas linguagens de programação, *Oracle* para Java e *JetBrains* para *Kotlin*, e o *Android Studio* IDE pode ser usado para desenvolver aplicativos Android ao mesmo tempo. (BIØRN-HANSEN; GRØNLI; GHINEA, 2018).

Como exemplo podemos adicionar arquivos *Kotlin* ao desenvolvimento e após isso converter o código Java para *Kotlin*. Com isso é possível utilizar as ferramentas disponíveis na plataforma do *Android Studio* com código *Kotlin*.

O *React Native* foi criado em 2015, pertence a uma biblioteca *JavaScript*. A sua criação conta com o desenvolvimento de algumas empresas como a *Microsoft*. Ao iniciar um projeto de um aplicativo, é necessário começar primeiro visualizando o público-alvo para a aplicação. Após isso balancear os prós e contras de cada plataforma e seu custo para o desenvolvimento

do projeto. Essa pesquisa precisa ser bem-feita porque o custo é variável e isso implica no valor final do projeto, e na sua manutenção futura.

3 MODELO DA APLICAÇÃO

Neste capítulo foi apresentado o desenvolvimento de uma aplicação mobile para exemplificar o desenvolvimento de um aplicativo.

O aplicativo foi desenvolvido para cadastrar usuários e poder autenticá-los, a sua finalidade é realizar o cadastro de pessoas e salvar os dados em um banco de dados, posteriormente os dados foram usados para que cada usuário possa fazer um login no aplicativo onde será exibido as suas informações.

Primeiramente foi desenvolvido e especificado, quais ferramentas foram usadas, em seguida foi apresentada a aplicação em funcionamento de forma geral como também de forma específica, como por exemplo as validações que o sistema tem.

3.1 FERRAMENTAS UTILIZADAS NO PROJETO

O aplicativo foi desenvolvido usando a plataforma *Android Studio*, a escolha da plataforma se deu por causa da praticidade que a plataforma oferece, juntamente com a plataforma foi usado a linguagem Java. Para salvar os dados dos cadastros dos usuários foi usado o *Firebase*, porque após fazer uma pesquisa ele se encaixava melhor no projeto, já que ele oferece tudo é um único lugar, como o banco de dados, o sistema para a autenticação dos usuários e como por exemplo um servidor de comunicação em tempo real.

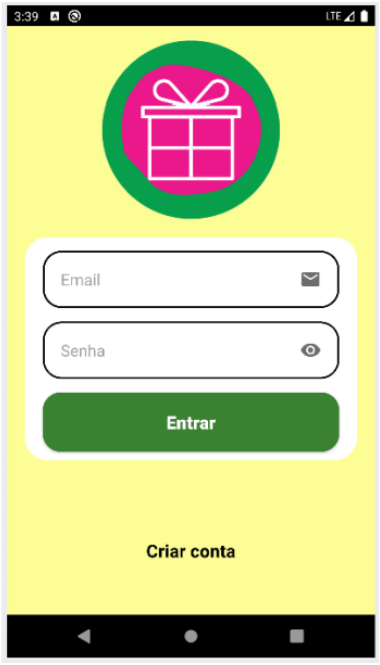
3.2 FUNCIONAMENTO DA APLICAÇÃO

Nesta seção foi apresentado as funcionalidades do aplicativo de forma geral, tanto quanto as validações que o sistema faz ao realizar um cadastro de usuário e o seu *login*.

3.2.1 Primeira tela

Na figura 1 foi apresentada a primeira tela do aplicativo exibe duas funções, em que o usuário pode interagir. Que são elas a fazer o login e criar conta, caso o usuário novo não esteja cadastrado. Na função de login será necessário informar o e-mail e a senha e clicar no botão entrar, caso os dados estejam corretos o aplicativo para a página de perfil do usuário. Caso o usuário clique em criar conta será direcionado para a página de cadastro.

Figura 1. Primeira tela

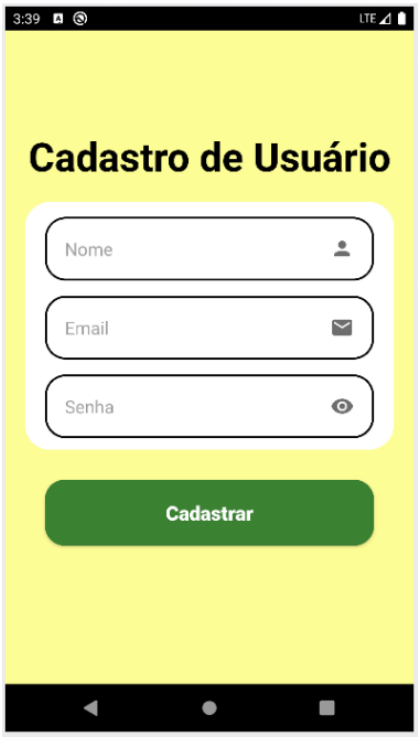


Fonte. Autor da pesquisa (2022)

3.2.2 Tela de cadastro

Na figura 2, é possível realizar o cadastro de novos usuários. Para que o cadastro seja realizado será necessário que sejam informados os campos que aparecem na tela. Ao preencher todos os campos e clicar no botão cadastrar se todos os campos estiverem corretos, o sistema vai exibir a mensagem de cadastrado com sucesso, caso falte algum campo cadastrado exibirá uma mensagem informando ao usuário.

Figura 2. Tela de cadastro

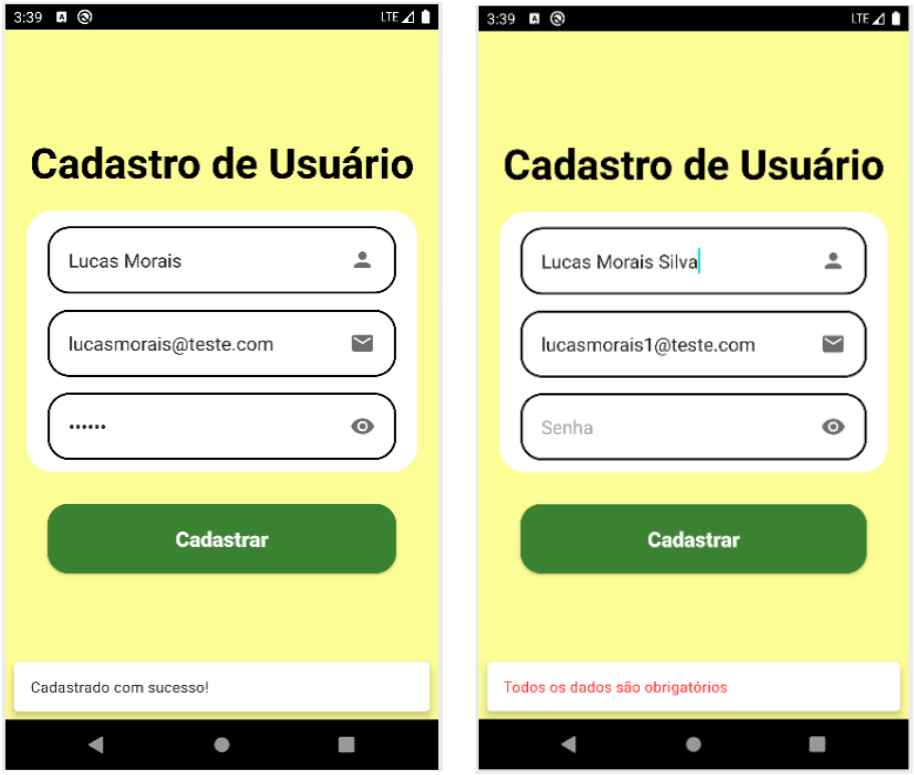


Fonte. Autor da pesquisa (2022)

3.2.3 Validação de campos no cadastro

Após o usuário clicar no botão de cadastro, se todos os campos estiverem preenchidos corretamente o sistema vai exibir a mensagem que o cadastro foi feito corretamente, de acordo com a figura 3.

Figura 3. Validação de campos no cadastro



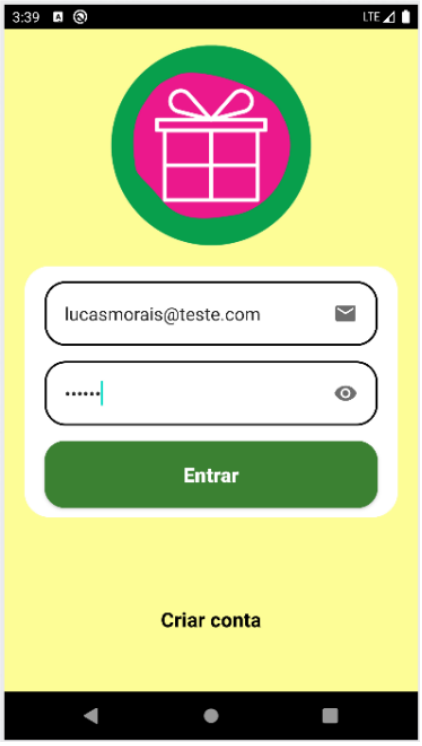
Fonte. Autor da pesquisa (2022)

Se o usuário clicar em cadastrar sem que todos os campos estejam devidamente preenchidos, o aplicativo vai exibir uma mensagem informando que todos os campos são obrigatórios.

3.2.4 Realizando login

Após o usuário realizar o seu cadastro, poderá voltar a tela inicial (figura 4) e realizar o login no aplicativo, para o login é apenas necessário o e-mail e senha. Informando os campos e depois clicando no botão entrar caso os dados estejam corretos o aplicativo vai para a página minha conta.

Figura 4. Tela de login

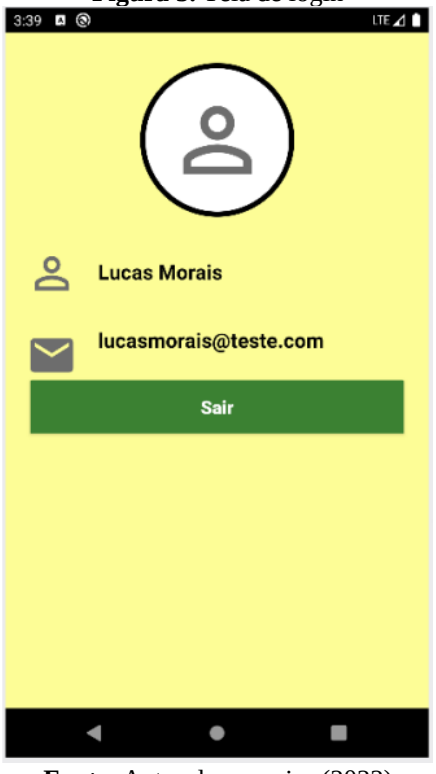


Fonte. Autor da pesquisa (2022)

3.2.5 Tela minha conta

Ao realizar o login o usuário será direcionado para a tela da conta onde serão exibidos o nome de usuário e o e-mail que foram cadastrados e a ação que a tela tem é a de sair que vai fazer o aplicativo voltar para a tela inicial (Figura 5).

Figura 5. Tela de login

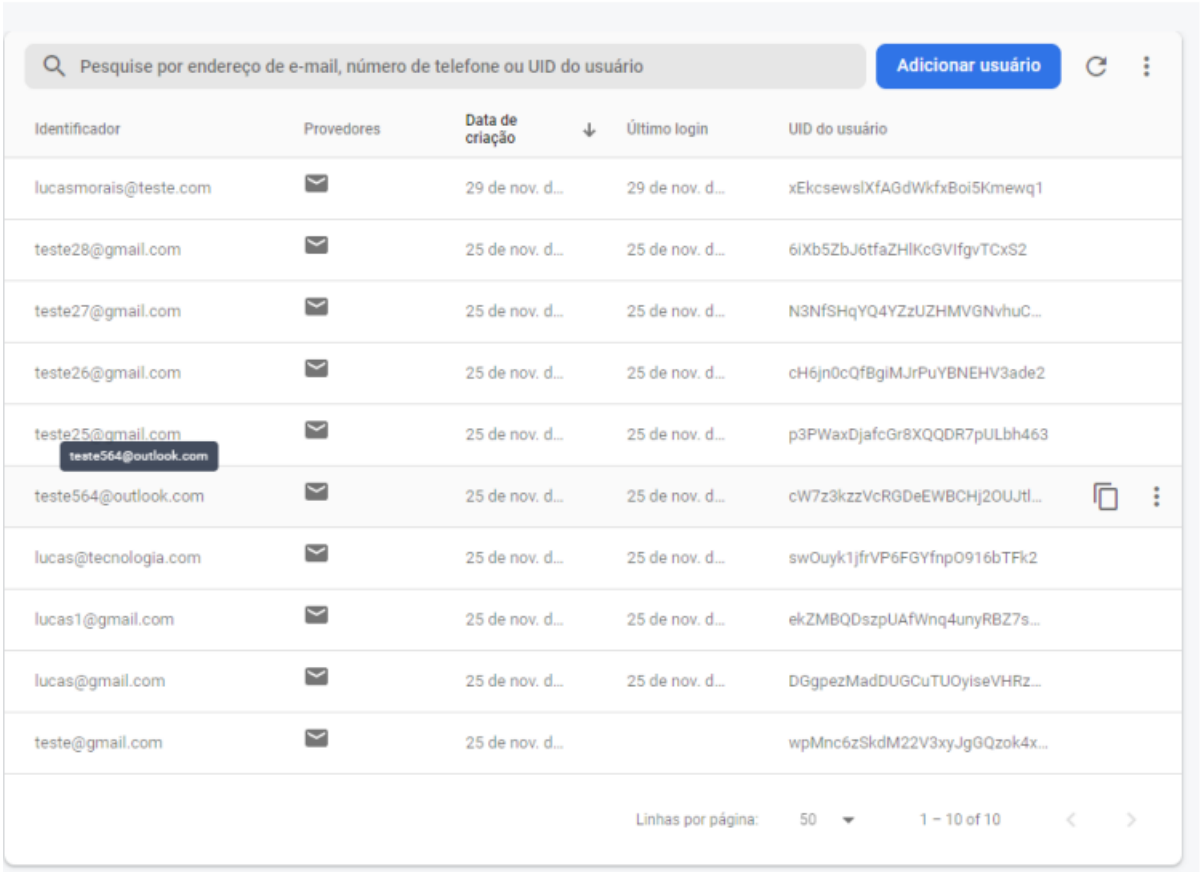


Fonte. Autor da pesquisa (2022)

3.2.6 Banco de dados

Como descrito anteriormente foi usado o *Firebase*, para usar o banco de dados (figura 5) e a autenticação dos usuários. Ao realizar um cadastro os dados informados pelo usuário são salvos no *Firebase*, onde os dados serão usados para fazer a autenticação no momento do login. Algumas funções que se pode fazer pelo *Firebase* é excluir o usuário, redefinir sua senha ou desativar a conta.

Figura 5. Base de dados



Fonte. Autor da pesquisa (2022)

5 CONSIDERAÇÕES FINAIS

Os referenciais usados como base para a criação da discussão acerca de plataformas nativas e multiplataformas apresentam muito conhecimento sobre o assunto. Dessa forma, a pesquisa é rica, responde à pergunta de pesquisa originalmente definida e facilita uma definição clara das considerações finais.

A conclusão final é que ao contrário dos aplicativos nativos feitos para uma plataforma específica, os aplicativos multiplataforma, como o nome sugere, atendem às necessidades de várias plataformas ao mesmo tempo. Eles funcionam na maioria dos dispositivos, pois usam formatos e estilos como sites móveis (HTML, CSS e *JavaScript*). De qualquer forma, no entanto, ainda há necessidade de continuar as pesquisas sobre esse tema para enriquecer o que toda a produção científica já trouxe e tirar melhores conclusões sobre o assunto.

REFERÊNCIAS

AHMAD, A. *et al*. An empirical study of investigating mobile applications development challenges. **IEEE Access**, v. 6, p. 17711-17728, 2018.

AHTI, V.; HYRYNSALMI, S.; NEVALAINEN, O. An evaluation framework for cross-platform mobile app development tools: A case analysis of adobe phonegap framework. In: **Proceedings of the 17th International Conference on Computer Systems and Technologies 2016**. 2016. p. 41-48.

BERA, M. H. G; MINE, A. F; LOPES, L. F. B. MEAN Stack: Desenvolvendo Aplicações Web Utilizando Tecnologias Baseadas em JavaScript. In: **III Seminário Empresarial e III Jornada de TI**, Maringá/PR. 2015.

BERNARDES, T. F.; MIYAKE, M. Y. Cross-platform mobile development approaches: A systematic review. **IEEE Latin America Transactions**, v. 14, n. 4, p. 1892-1898, 2016.

BIØRN-HANSEN, A.; GRØNLI, T.; GHINEA, G. A survey and taxonomy of core concepts and research challenges in cross-platform mobile development. **ACM Computing Surveys (CSUR)**, v. 51, n. 5, p. 1-34, 2018.

BOTELLA, F.; ESCRIBANO, P.; PEÑALVER, A.. Selecting the best mobile framework for developing web and hybrid mobile apps. In: **Proceedings of the XVII International Conference on Human Computer Interaction**. 2016. p. 1-4.

BRITO, H. *et al*. JavaScript in mobile applications: React native vs ionic vs NativeScript vs native development. In: **2018 13th Iberian conference on information systems and technologies (CISTI)**. IEEE, 2018. p. 1-6.

CUMMAUDO, A.; VASA, R.; GRUNDY, J.. What should I document? A preliminary systematic mapping study into API documentation knowledge. In: **2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)**. IEEE, 2019. p. 1-6.

EL-KASSAS, W. S. *et al.* Taxonomy of cross-platform mobile applications development approaches. **Ain Shams Engineering Journal**, v. 8, n. 2, p. 163-190, 2017.

FERREIRA, W; M. Desenvolvimento de aplicações multiplataforma usando Javascript: Uma análise prática. Morrinhos/GO, Brasil. **XIII Encontro Anual de Computação** - EnAComp 2017 – UFG. 2017.

EITKÖTTER, H.; HANSCHKE, S.; MAJCHRZAK, T. A. Evaluating cross-platform development approaches for mobile applications. In: **International Conference on Web Information Systems and Technologies**. Springer, Berlin, Heidelberg, 2012. p. 120-138.

LAKATOS, E.M.; MARCONI, M.A. **Fundamentos de metodologia científica**. São Paulo: Atlas, 2017.

MARDAN, A. **Full Stack JavaScript**. 2ª ed. New York/NY: Apress Media, 2015.

MATOS, B. R. D.; SILVA, J. G. B. **Estudo comparativo entre o desenvolvimento de aplicativos móveis utilizando abordagem nativa e multiplataforma**. Brasília. Estudo comparativo. 2016.

PAPAJORGJI, P. **Automated Enterprise Systems for Maximizing Business Performance**. 1 edition. ed. Hershey, PA: IGI Global, 2015.

RODRíGUEZ, A. M.; BALDRICH, R. **Diseño e implementación de una aplicación multidispositivo en un entorno HTML5**. 2015.

SHAKSHUKI, E. M.; *et al.* Component based Framework to Create Mobile Crossplatform Applications. **Procedia Computer Science**, 2013

SILVIA, D; SOUSA C. Construção de app com react native. **Revista Tecnologias em Proje**ção, 2019.

SMUTNÝ, P. Mobile development tools and cross-platform solutions In: **13th International Carpathian Control Conference** (ICCC). 2012.

SYDOW, L. **The State of Mobile in 2020**: The Key Stats You Need to Know. [S. l.], 15 jan. 2020.

URSINO, D.; CAPANNA, C. A. **AngularJS - un framework di frontiera per la realizzazione di siti Web**. 2015.