ANÁLISE COMPARATIVA DO DESENVOLVIMENTO DE APLICATIVOS MOBILE: HÍBRIDO VS. NATIVO

Bruno Rosso Da Rocha[1]
Guilherme De Noni Marchioro[2]
Lucas Resendes Ferreira[3]
Leandro Antunes Ugioni[4]
Thyerri Fernandes Mezzari[5]

Resumo: O crescimento acelerado do uso de smartphones impulsionou pessoas e empresas inovadoras a desenvolverem aplicativos móveis, a fim de atender às demandas e necessidades dos clientes de forma mais prática e ágil. Quando uma empresa decide criar um aplicativo para atender ao mercado, ela se depara com a necessidade de escolher as tecnologias a serem adotadas. Na camada técnica, o desenvolvimento de soluções móveis pode ser dividido em duas abordagens principais: nativa e híbrida, cada uma com suas características, vantagens e desvantagens. Este trabalho tem como objetivo realizar uma análise comparativa entre dois aplicativos, desenvolvidos respectivamente com as abordagens de desenvolvimento nativo e híbrido, abordando aspectos financeiros, de manutenção e desempenho, com o intuito de esclarecer as principais diferenças entre essas duas formas de desenvolvimento de aplicativos móveis. A partir dessa análise, foi possível identificar cenários em que uma abordagem se mostrou mais vantajosa que a outra, permitindo, assim, oferecer uma base sólida para a escolha mais estratégica e eficiente da tecnologia a ser utilizada no desenvolvimento de aplicativos móveis.

Palavras-chaves: Mobile. Aplicativos Nativos. Aplicativos Híbridos. Software. Smartphone.

1 INTRODUÇÃO

O ano de 2007 marcou um grande avanço no setor de telecomunicações com o lançamento do primeiro dispositivo móvel da Apple, o iPhone. Esse evento revolucionou o mercado, impulsionando concorrentes a desenvolverem seus próprios sistemas operacionais para smartphones. Nesse contexto de inovação, surgiram as plataformas Android e iOS, que rapidamente se consolidaram como as principais plataformas móveis da atualidade.

De acordo com dados divulgados pela Agência Nacional de Telecomunicações

[1] Graduando em Engenharia de Computação, 2025/2. E-mail: brunoschibroonn@gmail.com
[2] Graduando em Engenharia de Computação, 2025/2. E-mail: guilherme.marchioro@outlook.com
[3] Professor do Centro Universitário UniSATC.E-mail: lucas.ferreira@satc.edu.br
[4] Professor do Centro Universitário UniSATC. E-mail: leandro.ugioni@satc.edu.br
[5] Professor do Centro Universitário UniSATC. E-mail: thyerri.mezzari@satc.edu.br

(Anatel) em janeiro de 2025, o Brasil conta com aproximadamente 263,3 milhões de celulares em uso, resultando em uma densidade de 120,94 celulares para cada 100 habitantes. Esse crescimento expressivo evidencia a crescente demanda por soluções tecnológicas que atendam às necessidades dos usuários de forma eficiente.

Além do aumento no número de dispositivos móveis, a economia digital tem se expandido rapidamente. Segundo a pesquisa *State of Mobile 2025*, da Sensor Tower, a receita anual de aplicativos, excluindo o setor de games, cresceu 1.877% entre 2014 e 2024, passando de 3,5 bilhões para 69,2 bilhões de dólares. Esse cenário reflete o impacto do aumento da conectividade e da popularização dos smartphones, especialmente após a pandemia, que impulsionou a adoção de soluções digitais.

De acordo com uma pesquisa realizada pela Appdome, 90,3% dos consumidores brasileiros preferem utilizar aplicativos móveis em vez de plataformas web ou desktop. Essa porcentagem é 18% superior à média global de 76,5%. Diante desse panorama, empresas e startups passaram a priorizar o desenvolvimento de aplicativos móveis, buscando oferecer um acesso mais prático e ágil a seus produtos e serviços. O mobile tornou-se uma plataforma essencial para a interação com os consumidores, permitindo uma experiência mais eficiente, personalizada e acessível.

No lado técnico, durante o desenvolvimento de aplicativos mobile, existem duas abordagens principais: o desenvolvimento nativo e o híbrido, cada uma com suas próprias características e particularidades. Segundo REIS (2019), as aplicações nativas são desenvolvidas especificamente para uma plataforma, seja iOS ou Android, e, consequentemente, são executadas apenas no ambiente que foram planejadas e instruídas com suas respectivas linguagens de código (uma indicada para cada sistema operacional). Esse tipo de desenvolvimento exige a adaptação do código-base caso haja a necessidade de portar a aplicação para outra plataforma, pois cada sistema operacional possui sua própria linguagem de programação e ambiente de desenvolvimento integrado (IDE). Como linguagens nativas, destacam-se Swift e Objective-C para iOS, e Java e Kotlin para Android.

Ainda de acordo com Reis (2019), um dos fatores mais relevantes no desenvolvimento nativo é o desempenho. Como os aplicativos são construídos diretamente para o sistema operacional, sua execução ocorre de maneira mais otimizada, permitindo seguir as diretrizes da plataforma e aprimorar a experiência do usuário. Além disso, outra vantagem do desenvolvimento nativo é a possibilidade de

utilizar plenamente os recursos integrados do dispositivo, sem a necessidade de uma camada intermediária para acessar funcionalidades específicas do hardware.

Por outro lado, as aplicações híbridas possibilitam o desenvolvimento e criação de aplicativos com um único código-fonte base, permitindo sua implementação em múltiplas plataformas sem a necessidade de criar versões separadas para cada sistema operacional. Essa abordagem combina elementos do desenvolvimento mobile e web, sendo construída com tecnologias como HTML5, CSS e JavaScript. Além disso, frameworks como React Native, Flutter e Ionic se destacam como as soluções mais utilizadas para a criação de aplicativos híbridos, proporcionando maior eficiência no processo de desenvolvimento e manutenção.

De acordo com a pesquisa realizada pelo Stack Overflow em 2024, o JavaScript, linguagem base para frameworks híbridos como React Native e Ionic, continua sendo a mais popular entre os desenvolvedores, superando linguagens nativas como Java, Kotlin, Swift e Objective-C. Esse dado evidencia o crescimento das soluções híbridas e reforça a importância do questionamento: quais as principais diferenças entre o desenvolvimento de aplicativos nativos e híbridos nos aspectos financeiros, de manutenção e de performance?

Diante da crescente competitividade do mercado e da diversidade de soluções tecnológicas disponíveis, torna-se fundamental que as equipes de desenvolvimento mobile escolham a abordagem mais alinhada às necessidades específicas de cada aplicação, considerando fatores como desempenho, custo, prazo e público-alvo (Reis, 2019).

O objetivo deste artigo é esclarecer as diferenças entre as abordagens de desenvolvimento mobile nativo e híbrido, por meio da análise de estudos e pesquisas recentes, dados de mercado sobre a adoção de ambas as tecnologias, além da criação prática de duas aplicações, uma utilizando cada abordagem. A análise proposta é especialmente relevante para empresas, startups e desenvolvedores que estão planejando construir um aplicativo ou um software como serviço (SaaS) e precisam tomar decisões estratégicas quanto às tecnologias e metodologias mais adequadas para o desenvolvimento eficiente e sustentável de suas soluções.

Além de contribuir para o avanço do mercado de soluções mobile, o artigo também se propõe a expandir o repertório de informações sobre o tema na comunidade científica, servindo como referência para pesquisas acadêmicas futuras.

Para os acadêmicos, além de reforçar o aprendizado prático de conceitos mobile estudados ao longo da graduação, o estudo proporciona uma compreensão mais profunda sobre as implicações de cada abordagem tecnológica. Criar aplicações que funcionem em diferentes plataformas, com o menor custo possível, alta performance e baixa taxa de erros, é uma tarefa complexa que exige conhecimento técnico aprofundado, planejamento e dedicação (Silva; Prado, 2019).

Por fim, este artigo permitirá que empresas e startups tomem decisões mais assertivas na escolha entre desenvolvimento híbrido e nativo, considerando suas condições, equipe e recursos técnicos. Em vez de perder tempo na tomada de decisão, as organizações poderão concentrar seus esforços e investimentos em outras áreas essenciais do desenvolvimento do produto, acelerando sua chegada ao mercado e aumentando suas chances de sucesso.

2 PROCEDIMENTO EXPERIMENTAL

Neste capítulo, são apresentados os procedimentos e etapas definidos para fundamentar a análise comparativa entre abordagens de desenvolvimento mobile. A metodologia adotada consistiu na criação de dois aplicativos de lista de tarefas (To-Do List), sendo um desenvolvido de forma nativa (Kotlin) e o outro por meio de abordagem híbrida (React Native), com o objetivo de avaliar e comparar aspectos como construção, performance, custo e manutenção. Essa abordagem possibilitou uma análise prática e fundamentada, alinhada à definição de método como um conjunto de atividades organizadas e lógicas para gerar conhecimentos válidos e apoiar decisões (Marconi; Lakatos, 2022).

A pesquisa científica apresentada neste artigo foi classificada, quanto à sua natureza, como aplicada, pois teve como objetivo gerar conhecimentos com aplicação prática voltados à solução de problemas específicos no contexto do desenvolvimento de aplicativos móveis. No que se refere à forma de abordagem, tratou-se de uma pesquisa quantitativa, uma vez que envolveu a análise e comparação de dados mensuráveis relacionados ao desempenho das abordagens de desenvolvimento nativo e híbrido.

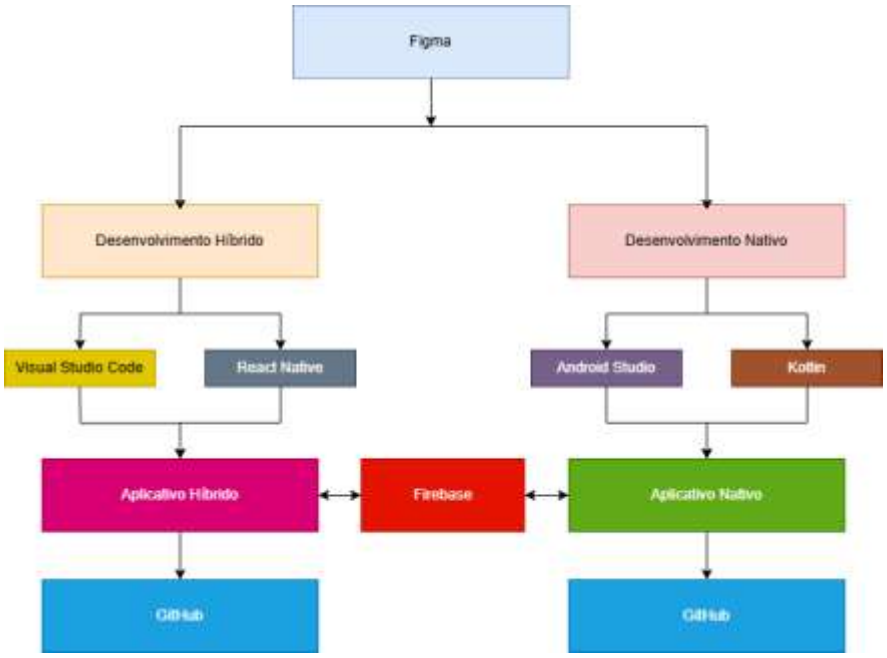
Em relação aos objetivos, a pesquisa foi caracterizada como exploratória, pois buscou compreender e aprofundar o conhecimento sobre as potencialidades e

limitações das tecnologias abordadas, além de propor insights para o aprimoramento de práticas no desenvolvimento mobile. Por fim, os procedimentos técnicos adotados foram de natureza experimental, pois incluíram o desenvolvimento de dois aplicativos, um nativo e outro híbrido, seguidos por uma análise comparativa que fundamentou os resultados obtidos.

2.1 MATERIAIS

Para a projeção, o desenvolvimento e os testes dos aplicativos To-Do List, desenvolvidos nas abordagens híbrida e nativa, diversas ferramentas foram utilizadas e estão mapeadas no fluxograma apresentado na Figura 1.

Figura 1: Tecnologias empregadas no desenvolvimento.



Fonte: Dos autores (2025).

2.1.1 Figma

A interface e as telas dos aplicativos foram elaboradas no Figma, uma ferramenta de design colaborativo amplamente utilizada para a criação de protótipos e interfaces com foco em usabilidade e experiência do usuário (UX/UI). Ambos os aplicativos seguiram o mesmo padrão visual e de navegação, garantindo uma

experiência consistente independentemente da abordagem de desenvolvimento adotada.

2.1.2 Android Studio

O desenvolvimento do aplicativo nativo foi realizado utilizando o Android Studio, o ambiente de desenvolvimento integrado (IDE) amplamente reconhecido por suas ferramentas e recursos voltados ao desenvolvimento de aplicativos nativos na plataforma Android.

Esse ambiente de desenvolvimento inclui um emulador integrado que permite simular dispositivos Android. Essa funcionalidade facilitou significativamente o processo de teste e validação das funcionalidades desenvolvidas, garantindo maior eficiência e precisão no desenvolvimento. A versão do Android Studio utilizada foi a Koala Feature Drop 2024.1.2, e o aplicativo foi compilado utilizando a versão 35 do Android SDK.

2.1.3 Kotlin

Como principal linguagem de programação para o desenvolvimento nativo, foi utilizado o Kotlin, uma linguagem moderna, fortemente tipada e projetada para funcionar com Java.

O Kotlin destaca-se por sua capacidade de otimizar o processo de desenvolvimento, proporcionando maior eficiência e segurança junto ao Android Studio e é o mais indicado pelo Google, atual detentora do sistema operacional Android.

2.1.4 Visual Studio Code

O desenvolvimento do aplicativo híbrido foi conduzido utilizando o Visual Studio Code, um editor de código amplamente utilizado devido à sua extensibilidade e suporte a diversas linguagens de programação.

Embora não seja uma IDE completa, o Visual Studio Code pode ser

configurado com extensões específicas que proporcionam funcionalidades avançadas, como suporte ao React Native e integração com emuladores de dispositivos Android, viabilizando o processo de teste e validação do aplicativo híbrido.

2.1.5 React Native

Para o desenvolvimento híbrido, utilizou-se o React Native, um framework multiplataforma baseado em React.js que permite criar aplicativos para Android e iOS a partir de uma única base de código, utilizando JavaScript ou TypeScript. A versão do React Native empregada no projeto foi a 0.74.5.

2.1.6 Expo

Junto ao React Native, foi utilizado o Expo, uma ferramenta que simplificou o processo de desenvolvimento, testes e publicação de aplicativos híbridos. O Expo oferece um ambiente de desenvolvimento mais acessível e produtivo, dispensando configurações complexas e permitindo testes rápidos em dispositivos físicos ou emuladores, o que contribui para uma maior agilidade no ciclo de desenvolvimento. Neste projeto, foi utilizada a versão 51.0.28 do Expo.

2.1.7 Emulador Android

Para a realização dos testes práticos dos aplicativos, foi utilizado um emulador Android do modelo Pixel 4, a fim de avaliar o comportamento das aplicações em diferentes ambientes e condições de uso.

2.1.8 GitHub

Além das linguagens de programação, frameworks e ambientes de desenvolvimento, o controle e o versionamento do código foram realizados por meio do GitHub. A plataforma foi utilizada para armazenar o código-fonte de ambos os aplicativos, organizados em repositórios distintos, permitindo o acompanhamento das

alterações realizadas e a colaboração eficiente no desenvolvimento.

2.1.9 Firebase

Foi utilizada a plataforma Firebase na versão 10.14.1 como API externa, na qual possibilitou uma estrutura de back-end completa. O Cloud Firestore foi empregado como banco de dados, permitindo sincronização em tempo real, além de outros recursos como autenticação de usuários.

2.2 MÉTODOS

Com o objetivo de realizar uma avaliação detalhada e uma comparação precisa entre os dois aplicativos de To-Do List desenvolvidos, foram definidas métricas-chave, como construção, performance, custo e manutenção.

2.2.1 Métrica de construção

A avaliação da construção do aplicativo incluiu a análise do tempo total de desenvolvimento e a complexidade do código gerado. Esses fatores foram considerados para entender a eficiência no desenvolvimento e os desafios enfrentados durante a implementação das funcionalidades.

2.2.2 Métrica de performance

Foi realizada uma avaliação do tempo de carregamento inicial de ambos os aplicativos, além de uma análise do desempenho e complexidade na integração com a API externa Firebase, verificando a eficiência e a estabilidade da comunicação entre o aplicativo e o banco de dados Cloud Firestore em tempo real.

2.2.3 Métrica de custo

Análise do custo de desenvolvimento de ambos os aplicativos, considerando o número de desenvolvedores e a remuneração da equipe.

2.2.4 Métrica de manutenção

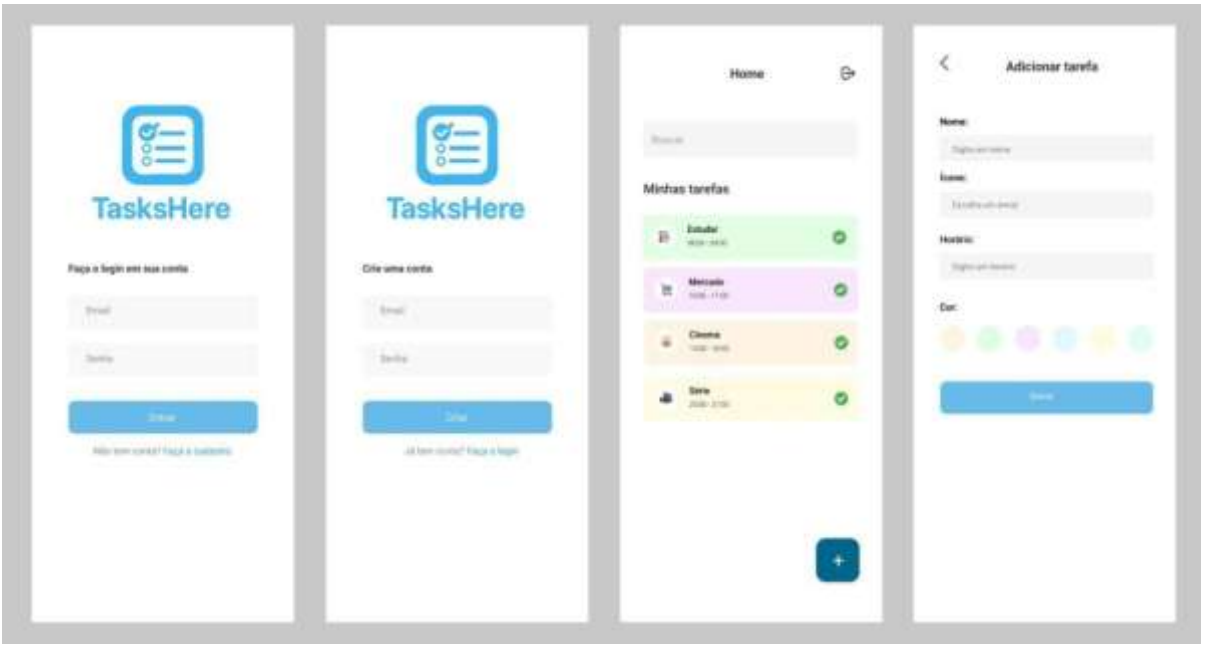
Realizou-se uma avaliação da flexibilidade do código em termos de manutenção, considerando a facilidade de implementação de ajustes e melhorias.

3 DESENVOLVIMENTO DOS APLICATIVOS

Para realizar a análise comparativa entre as abordagens de desenvolvimento mobile nativa e híbrida, foram desenvolvidos dois projetos de aplicativos com a exata mesma proposta e base visual: uma aplicação de lista de tarefas (To-Do List).

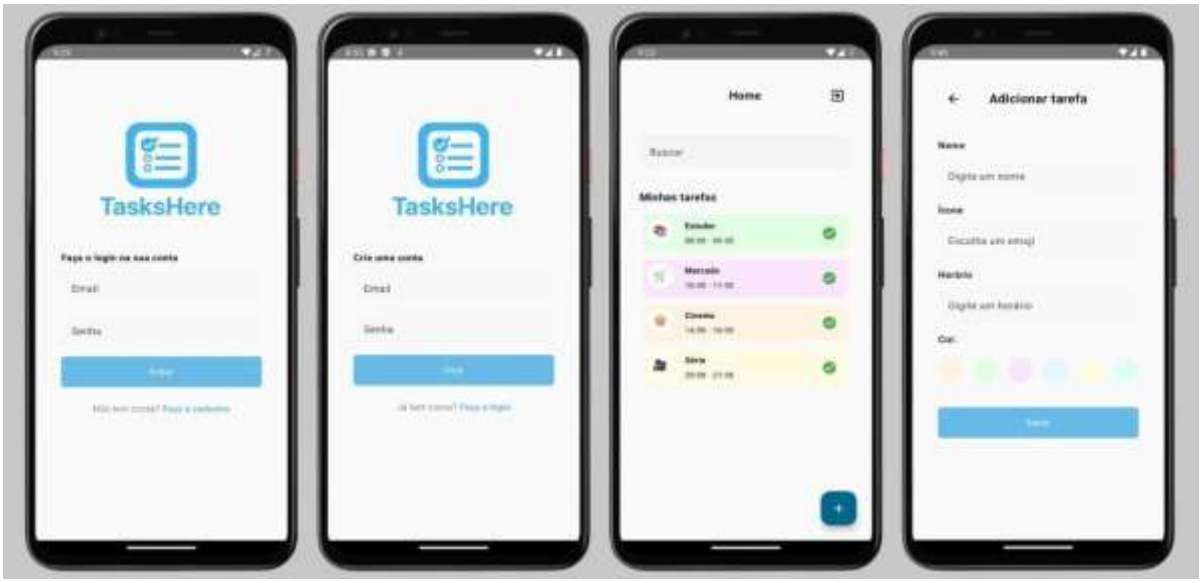
Ambos os aplicativos foram construídos com base em um protótipo elaborado no Figma, conforme apresentado na Figura 2. A versão nativa foi implementada utilizando a linguagem Kotlin, por meio do Android Studio, o design das telas foi apresentado na Figura 3. Já a versão híbrida foi desenvolvida com o framework React Native, conforme demonstrado na Figura 4.

Figura 2: Protótipo construído no Figma.



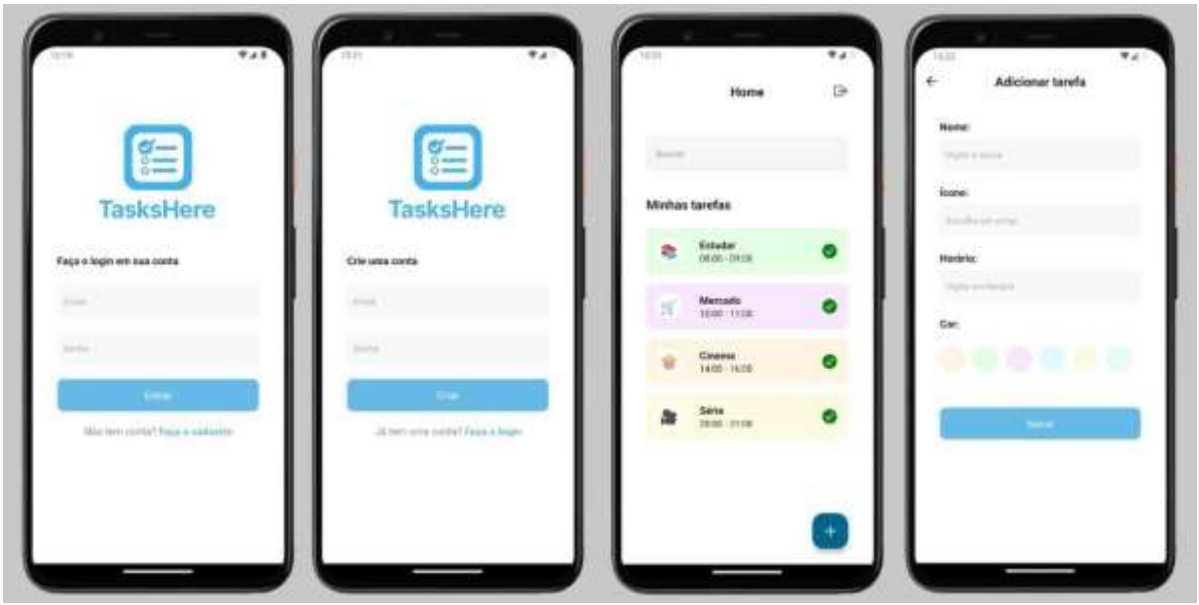
Fonte: Dos autores (2025).

Figura 3: Aplicativo construído com Kotlin.



Fonte: Dos autores (2025).

Figura 4: Aplicativo construído com React Native.



Fonte: Dos autores (2025).

Ao analisar as telas do protótipo e dos aplicativos desenvolvidos, é possível perceber que ambos os aplicativos mantêm a identidade visual proposta no protótipo, com pequenas variações na renderização de componentes, principalmente nos botões e campos de input, decorrentes das particularidades de cada tecnologia utilizada.

O código do aplicativo em Kotlin está disponível no GitHub: https://github.com/marchiorog/tasksHereAppKotlin. Enquanto que a versão construída em React Native pode ser acessada pelo através do link:

https://github.com/marchiorog/tasksHereAppReactNative.

4 RESULTADOS E DISCUSSÕES

Nesta seção, são apresentados os resultados obtidos a partir da construção das duas versões do mesmo protótipo de aplicativo. A análise comparativa considera avaliação de construção, performance, custo e manutenção, de cada projeto.

4.1 AVALIAÇÃO DE CONSTRUÇÃO

Em relação à construção dos aplicativos, foi feito um mapeamento das horas dedicadas a cada aplicação, conforme mostrado na Tabela 1. O desenvolvimento foi realizado pelos autores, acadêmicos de nível júnior.

Tabela 1: Comparativo de horas de desenvolvimento entre tecnologias.

Etapas	React Native	Kotlin
Protótipo: construção com Figma	1	1
Front-End: componentização e design	9	15
Back-End: integração com Firebase	3	5
Cloud Firestore: configuração	1	1
Total (horas)	14	22

Fonte: Dos autores (2025).

Observou-se que o tempo total necessário para a implementação do aplicativo utilizando a abordagem híbrida com React Native foi de aproximadamente 14 horas, enquanto a versão nativa desenvolvida em Kotlin demandou cerca de 22 horas. Apesar de ambas as abordagens resultarem em aplicativos funcionais equivalentes, o desenvolvimento com React Native demonstrou maior eficiência em termos de tempo.

Essa diferença está relacionada à reutilização de componentes, à simplicidade na configuração de recursos e à integração facilitada com bibliotecas externas, como

as de navegação e serviços do Firebase. Por outro lado, o desenvolvimento com Kotlin exigiu maior atenção à estruturação do código e à configuração detalhada de dependências, além de um esforço adicional para integrar serviços externos. A construção de interfaces nativas, a linguagem mais robusta e a manipulação direta dos elementos da plataforma Android também contribuíram para um processo de desenvolvimento mais longo e tecnicamente mais exigente.

4.2 AVALIAÇÃO DE PERFORMANCE

A análise do tempo de carregamento inicial revelou que o aplicativo desenvolvido em React Native em build de desenvolvimento apresentou uma média de 3,2 segundos para estar completamente carregado, enquanto a versão nativa em Kotlin atingiu um tempo médio de 2,8 segundos. Embora a diferença seja pequena, a aplicação nativa demonstrou uma inicialização mais rápida em função da otimização inerente à integração direta com o sistema Android.

Referente a integração com a API externa do Firebase, ambos os aplicativos demonstraram eficiência e estabilidade na comunicação com o banco de dados Cloud Firestore durante as operações em tempo real. O aplicativo híbrido, no entanto, apresentou maior facilidade e agilidade durante a implementação, devido à disponibilidade de bibliotecas bem integradas e de fácil configuração no ambiente React Native. Já a versão em Kotlin exigiu mais configurações manuais e ajustes específicos, o que demandou um maior esforço e tempo de desenvolvimento.

Uma das principais desvantagens do desenvolvimento híbrido é que, embora tenha como objetivo a compatibilidade entre diferentes plataformas, na prática, muitas soluções híbridas apresentam limitações de desempenho ou compatibilidade, o que pode restringir sua execução plena em ambas. Isso acaba exigindo adaptações específicas, aumentando o tempo de desenvolvimento, os custos e a complexidade para disponibilizar o aplicativo em múltiplas plataformas (Silva; Prado, 2019).

De modo geral, a integração com o Firebase foi bem-sucedida em ambas as abordagens, assegurando a funcionalidade essencial de sincronização em tempo real para a aplicação de lista de tarefas.

4.3 AVALIAÇÃO DE MANUTENÇÃO

A avaliação da flexibilidade do código em termos de manutenção indicou que o aplicativo desenvolvido em React Native apresentou maior facilidade na implementação de ajustes e melhorias, devido à arquitetura modular e à reutilização de componentes proporcionada pelo framework. Modificações como inclusão de novas funcionalidades ou correções de bugs puderam ser realizadas de forma mais rápida e com menor impacto no restante do código, além de atender ambas as plataformas.

Por outro lado, a versão nativa em Kotlin, embora ofereça maior integração com o ecossistema Android, demonstrou maior complexidade para manutenção, exigindo um conhecimento mais aprofundado da estrutura da plataforma Android e maior cuidado na alteração de componentes interligados.

4.4 AVALIAÇÃO DE CUSTO

A avaliação de custo dos aplicativos foi realizada com base em uma pesquisa acerca dos valores relacionados à contratação de desenvolvedores. Os dados foram obtidos a partir de plataformas como Glassdoor e LinkedIn, que forneceram informações pertinentes para a comparação entre as abordagens de desenvolvimento nativo e híbrido.

É importante considerar que diferentes níveis de desenvolvedores entregam o trabalho em prazos distintos, de acordo com seu conhecimento e experiência. Para esta pesquisa, foram considerados apenas profissionais com conhecimento prévio nas linguagens e ferramentas necessárias para o desenvolvimento dos aplicativos, optando-se por utilizar como referência desenvolvedores de nível pleno.

Com base em dados coletados no site Glassdoor em março de 2025, a faixa salarial geral para desenvolvedores no Brasil varia entre R$3.000 e R$6.000 mensais. Ao refinar a busca para “Desenvolvedor iOS pleno”, foram identificadas 189 vagas, com salários médios entre R$6.000 e R$9.000 mensais, conforme atualização de março de 2024. A pesquisa por “Desenvolvedor Android pleno” revelou 417 vagas nacionais, com faixa salarial semelhante, segundo dados de fevereiro de 2024. No LinkedIn, foram encontradas 609 vagas para desenvolvedores iOS pleno e 713 para Android pleno, em buscas realizadas em março de 2025, embora esta plataforma não

divulgue informações salariais.

Para profissionais que atuam com tecnologias híbridas, como React Native, o cenário apresenta variações tanto na remuneração quanto na disponibilidade de vagas. Na busca por “Desenvolvedor React Native pleno” no Glassdoor, foram identificadas 437 oportunidades, com salários entre R$3.000 e R$7.000 mensais. No LinkedIn, esse número sobe para 1.263 ofertas em todo o país.

Tabela 2: Faixa salarial do desenvolvedor por Glassdoor e Linkedin.

Desenvolvedor pleno	Faixa salarial	N° Vagas (Glassdoor)	N° Vagas (Linkedin)
IOS	R$6.000 a R$9.000	189	609
Android	R$6.000 a R$9.000	417	713
React Native	R$3.000 a R$7.000	437	1262

Fonte: Glassdoor/Linkedin.

Esses dados evidenciam que, para empresas iniciantes ou com recursos limitados, a disponibilidade e os custos associados à contratação de desenvolvedores podem influenciar significativamente a escolha da abordagem de desenvolvimento, especialmente quando o objetivo é testar o mercado com o menor investimento inicial possível.

O custo dobrado na contratação de dois desenvolvedores, um para iOS e outro para Android, pode ser justificado não apenas pela necessidade de profissionais distintos, mas também pelo tempo demandado caso um único profissional domine ambas as plataformas. Nesse cenário, o desenvolvimento pode levar o dobro do tempo, impactando diretamente o custo final do projeto.

5 CONSIDERAÇÕES FINAIS

A partir da análise comparativa entre as abordagens de desenvolvimento híbrido, utilizando React Native com foco na plataforma Android, e nativo, com Kotlin, foi possível responder à pergunta-problema inicial sobre qual abordagem se mostra mais adequada para o desenvolvimento de aplicativos. O objetivo geral deste artigo, que consistia em avaliar as vantagens, desafios e custos de cada abordagem para

auxiliar na tomada de decisão, foi plenamente alcançado por meio da análise técnica, financeira e de manutenção realizada entre dois aplicativos.

O desenvolvimento híbrido com React Native destacou-se pela agilidade na implementação, menor complexidade na configuração inicial e facilidade de integração com bibliotecas externas como Firebase. O uso de uma única base de código para múltiplas plataformas mostrou-se vantajoso, principalmente para equipes reduzidas, ao reduzir significativamente o tempo de desenvolvimento e os custos com mão de obra. Por outro lado, o desenvolvimento nativo baseado na linguagem Kotlin, apesar do maior esforço técnico, custos mais elevados e tempo mais extenso de desenvolvimento, apresentou desempenho levemente superior, quando analisado em conjunto com o ecossistema Android, evidenciando seu potencial para aplicações que demandam integração profunda com recursos específicos do sistema operacional.

No que se refere à manutenção e flexibilidade do código, o React Native novamente se mostrou mais vantajoso, oferecendo maior rapidez na aplicação de correções e melhorias, o que é fundamental para ambientes em constante evolução. A curva de aprendizado mais suave e a ampla comunidade de suporte também são fatores que fortalecem o uso dessa tecnologia em contextos dinâmicos e com recursos limitados. Já a abordagem nativa demonstrou ser mais exigente, pois exige dois desenvolvedores distintos para desenvolver ambas as plataformas.

A avaliação de custos também favoreceu a abordagem híbrida. A necessidade de apenas um desenvolvedor, os salários médios mais acessíveis e a possibilidade de economizar com infraestrutura e equipamentos especialmente no caso do desenvolvimento para iOS tornam o React Native uma escolha financeiramente mais viável para a maioria dos projetos iniciais.

Sugere-se que pesquisas futuras explorem outras tecnologias híbridas, como o Flutter, e também nativas, como o Swift para a plataforma Apple, além de investigações mais empíricas que envolvam estudos de caso, testes de usabilidade e análise de desempenho em aplicações reais. A recomendação para considerar o iOS entre os estudos futuros justifica-se pelo fato de os autores não terem, até o momento, acesso a um ambiente de desenvolvimento adequado que atenda aos padrões da Apple, o que limitou a análise durante este estudo. Também seria relevante analisar o impacto do uso dessas tecnologias no ciclo de vida do software em projetos de maior escala e em diferentes setores.

Dessa forma, conclui-se que o desenvolvimento híbrido é a alternativa mais indicada para empresas que estão iniciando no mercado ou desenvolvendo um produto mínimo viável (MVP), pois proporciona um bom equilíbrio entre eficiência, desempenho e custo-benefício. Já o desenvolvimento nativo pode ser mais apropriado em fases posteriores, quando o aplicativo já estiver consolidado e for necessário explorar ao máximo os recursos de cada plataforma.

Portanto, a escolha entre desenvolvimento nativo e híbrido deve levar em consideração o contexto do projeto, os recursos disponíveis e os objetivos de negócio. O estudo demonstrou que, embora a abordagem nativa ofereça vantagens técnicas pontuais, o desenvolvimento híbrido apresenta uma solução mais prática e acessível para a maioria das iniciativas de desenvolvimento de software.

REFERÊNCIAS

AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES (ANATEL)**. Painéis de Dados**. Disponível em: https://informacoes.anatel.gov.br/paineis/acessos. Acesso em: 22 mar. 2025.

SENSOR TOWER. **State of Mobile 2025**. 2025. Disponível em: <https://e.infogram.com/_/8KyvK4LlLM6MDMLOmLCn?src=embed> Acesso em: 22 mar. 2025.

APPDOME. ***Mobile Consumer Cyber Security Survey 2024*.** 2024. Disponível em: https://www.appdome.com/mobile-consumer-cyber-security-survey-2024/. Acesso em: 22 mar. 2025.

MINHA OPERADORA. **Uso de aplicativos no Brasil supera a média global, revela pesquisa.** 2023. Disponível em: https://www.minhaoperadora.com.br/2023/06/uso-de-aplicativos-brasil-supera-a-media-global-revela-pesquisa.html. Acesso em: 22 mar. 2025.

REIS, Antônio C. S. dos. **Um estudo comparativo entre modelos de desenvolvimento de aplicações móveis.** 2019. 39 f. Monografia (Bacharel em Engenharia de Software) – Universidade Federal do Ceará, Quixadá, 2019. Disponível em:<https://repositorio.ufc.br/bitstream/riufc/49707/1/2019_tcc_acsdosreis.pdf>. Acesso em: 22 mar. 2025.

STACK OVERFLOW. **Survey 2024**: Technology. Disponível em: https://survey.stackoverflow.co/2024/technology/. Acesso em: 22 mar. 2025.

MARCONI, Marina de Andrade; LAKATOS, Eva Maria. **Fundamentos de metodologia científica***.* 9. ed. São Paulo: Atlas, 2022.

CORAZZA, Paulo Victor. **Um aplicativo multiplataforma desenvolvido com flutter e NoSQL para o cálculo da probabilidade de apendicite**. 2018.
SILVA, Fransérgio Aparecido de Souza; PRADO, Ely Fernando do. **Análise teórica sobre o desenvolvimento de aplicativos nativos, híbridos e webapps**. Disponível em: https://ric-cps.eastus2.cloudapp.azure.com/handle/123456789/5044. Acesso em: 5 abr. 2025.

SALES, Daniela Ferreira. **Desenvolvimento de aplicativos móveis: o mercado das aplicações nativas e híbridas**. 2023. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Redes de Computadores) – Instituto Federal de Educação, Ciência e Tecnologia do Amapá, Macapá, 2023. Disponível em: https://repositorio.ifap.edu.br/jspui/handle/prefix/790. Acesso em: 23 abr. 2025.

GLASSDOOR. **Vagas de emprego – Glassdoor Brasil**. Disponível em: https://www.glassdoor.com.br/Job/index.htm. Acesso em: 23 abr. 2025.

LINKEDIN CORPORATION. **LinkedIn**. Disponível em: https://www.linkedin.com. Acesso em: 23 abr. 2025.