ESCOLA

SUPERIOR

DE MEDIA

ARTES

E DESIGN

POLITÉCNICO

DO PORTO

Projeto Final

Plataformas Móveis

Diogo Alves de Sousa de Moura Bessa

40220461

Índice

1. Introdução / Contexto

Os *smartphones* surgem como uma tecnologia que reúne diversos dispositivos apenas num só aparelho (e.g telefone, internet, jogos, multimédia, entre outros) (MERIJE, 2012, *cited in* da Fonseca, 2013). O surgimento dos *smartphones* como um todo, acarretou mudanças significativas na sociedade contemporânea, não apenas a nível pessoal, como também a nível profissional (da Fonseca, 2013; Sarwar & Soomro, 2013). Nesta sequência, devido à mobilidade associada ao uso dos *smartphones*, houve um crescimento exponencial das aplicações móveis, trazendo inúmeros benefícios, sendo um dos exemplos, a facilidade de comunicação e de acesso à informação, que permite que qualquer pessoa, em qualquer lugar do mundo, esteja conectada (da Fonseca, 2013; Soukup, 2015). Adicionalmente, as aplicações facilitaram muitas atividades rotineiras tais como, fazer compras, encomendar comida, ouvir música ou ver filmes, tornando a nossa vida mais fácil e conveniente (Soukup, 2015). A nível de saúde, as aplicações móveis também desempenharam um papel importante, com *apps* que rastreiam atividades físicas, nutrição e até meditação, promovendo uma vida mais saudável para os utilizadores (Costa & Souza, 2022; Freitas & Pereira, 2021). Hoje em dia, através da telemedicina conseguimos obter cuidados médicos quer físicos, quer mentais, garantindo aos utentes a manutenção do seu bem-estar (Agu et al., 2023; Soukup, 2015). Desta forma, o crescimento das aplicações móveis não só transformou a maneira como comunicamos e trabalhamos, como também impactou, de forma positiva, o nosso dia-a-dia (da Fonseca, 2013; Soukup, 2015).

O ser-humano, ao longo dos últimos anos, acabou por desenvolver uma relação com estes dispositivos, demonstrando que estes são particularmente reconfortantes devido a uma combinação de fatores: (a) são objetos altamente pessoais; (b) são extremamente portáteis; (c) proporcionam um espaço privado onde os utilizadores podem escapar do ambiente externo; (d) possuem propriedades tácteis que os consumidores consideram prazerosas — todas estas características permitem que os telemóveis (e) ofereçam uma presença tranquilizadora para os seus proprietários (Melumad & Pham, 2020). O sentido de segurança proporcionado pelo telemóvel, por sua vez, permite que o

dispositivo funcione como uma fonte geral de conforto psicológico (Jarvenpaa & Lang, 2005).

Assim sendo, tendo em conta a evolução exponencial da tecnologia nas últimas décadas, decorrente da evolução inerente à sociedade, como um todo, nós, futuros programadores, deparamo-nos com um grande desafio: escolher as melhores tecnologias e arquiteturas para garantir o melhor desempenho e UX ao menor custo do mercado. No entanto, com a existência de várias alternativas, a nível arquitectónico, é crucial compreender quais as vantagens e desvantagens de cada arquitetura existente. Este aspeto é algo que irá ser aprofundado e mencionado no decorrer deste relatório.

Portanto, tendo em conta os objetivos deste projeto, acredito que o seu desenvolvimento proporcionará uma oportunidade única para consolidar os conhecimentos adquiridos durante este semestre e desenvolver a minha primeira aplicação móvel. Além de permitir a aplicação prática da matéria, este trabalho vai me ajudar a compreender as melhores práticas para criar aplicações móveis eficientes, escaláveis e focadas em UX. No atual panorama tecnológico, em constante evolução, esta experiência será essencial para enfrentar vários desafios e contribuir para a criação de soluções inovadoras alinhadas às necessidades da sociedade moderna.

https://scholarcommons.scu.edu/cgi/viewcontent.cgi?article=1102&context=comm

2. Objetivos

Este relatório tem principalmente dois objetivos: O primeiro é referente à compreensão dos 6 tipos de arquiteturas móveis lecionadas que vão desde as aplicações nativas, híbridas tradicionais, híbridas *cross-compiled*, web nativas, web progressivas (PWA) às web. Para cada tipo de arquitetura, será analisada uma aplicação móvel que exemplifique a tecnologia em questão. Adicionalmente, serão também apresentados casos práticos para justificar e fundamentar as decisões tomadas pelos desenvolvedores, destacando as vantagens, desvantagens e eventuais *trade-offs.* Além disso, será também apresentada uma opinião fundamentada quanto ao tipo de arquitetura mais adequada para cada cenário, em comparação com as alternativas existentes, explicando e refletindo sobre os motivos dessa escolha.

Posteriormente, este relatório também tem como um dos seus objetivos, desenvolver uma "pequena" aplicação móvel "simples", baseada numa das arquiteturas móveis estudadas. Nesta aplicação estará integrado o relatório, na sua íntegra, demonstrando, na prática, o conhecimento teórico adquirido no decorrer do semestre.
Portanto, como supramencionado, este projeto constituirá uma base sólida para o meu desenvolvimento profissional. O desenvolvimento desta aplicação móvel, permitir-me-á aprofundar os meus conhecimentos sobre as arquiteturas, frameworks e tecnologias existentes no mercado atual, bem como o desenvolvimento de habilidades práticas para justificar a utilização de cada escolha, comparando-a com as restantes alternativas de forma a implementar a solução mais adequada para cada caso.

3. Desenvolvimento

Este capítulo apresenta casos práticos de aplicações móveis representativas para cada tipo de arquitetura estudada, acompanhados de análises críticas fundamentadas. Serão discutidos exemplos reais, comparados os prós e contras de suas arquiteturas e apresentadas reflexões sobre possíveis alternativas.

3.1. Aplicações Nativas

Quando se trata de aplicações móveis, é indiscutível que existem neste momento, apenas duas grandes frentes: Android e iOS. iOS é uma abreviatura do sistema operativo do iPhone, sendo que é um sistema operativo, lançado em 2007, que funciona exclusivamente em dispositivos móveis Apple. É o segundo sistema operativo móvel mais popular do mundo, seguindo apenas o Android. É ainda a base para três outros sistemas operativos da Apple: iPadOS, tvOS, e watchOS.

As aplicações nativas são construídas especificamente para um sistema operativo (OS) de um dispositivo móvel. Assim, é possível ter aplicações móveis Android nativas(GCam) ou aplicações iOS nativas(*Fitness+*), para não mencionar todas as outras plataformas e dispositivos como Blackberry OS ou Windows Phone. Sendo que as aplicações nativas só podem ser construídas numa plataforma, é então incompatível a utilização de aplicações num sistema operativo que não o seu nativo.

Por exemplo: não podemos ter uma aplicação Blackberry num telemóvel com o sistema operativo Android ou usar uma aplicação iOS num telemóvel com um sistema operativo Windows Phone porque as aplicações nativas são desenvolvidas utilizando linguagens e frameworks específicos para cada sistema operativo.

Existem vários pontos positivos e vários pontos negativos, que advêm da utilização deste tipo de aplicações móveis. Um dos maiores problemas com este tipo de arquitetura de desenvolvimento prende-se ao custo associado à necessidade de desenvolvimento nativo. Se for necessário disponibilizar a aplicação para os dois sistemas operativos, será necessário desenvolver duas aplicações móveis distintas – que, na maioria dos casos, será necessário, já que não é intenção deixar de fora milhões

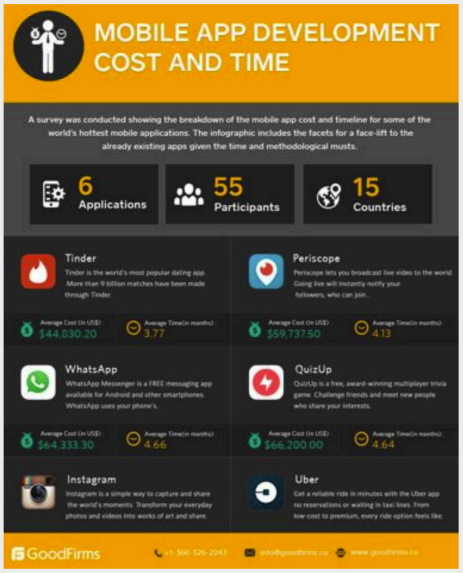
de utilizadores apenas pela diferença do sistema operativo – a aplicação será mais custosa, na medida em que, ou o tempo de desenvolvimento do projeto duplica, ou a própria quantidade de elementos na equipa de desenvolvimento aumenta. Em caso de mudanças, ter que replicar as alterações para as duas aplicações nos dois sistemas operativos e o facto de ter de ser necessário a aprovação das aplicações nas lojas digitais onde estará disponibilizada aumenta ainda mais o custo de desenvolvimento.

QUAL É O CUSTO PARA DESENVOLVER UMA APLICAÇÃO MÓVEL?

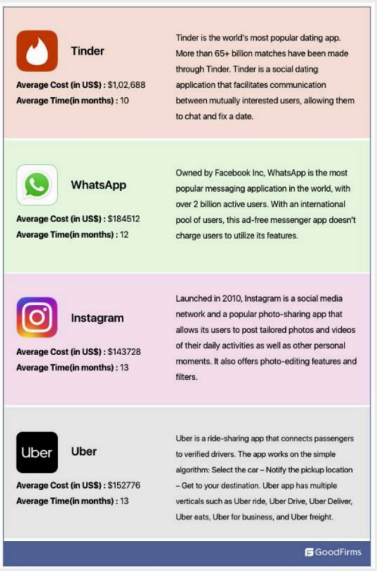
Vamos utilizar o seguinte **exemplo**: um serviço de IT que custe um preço médio de **40 dólares/hora**, o custo para construir uma aplicação varia **entre 40.000 e 300.000+ doláres**.

	Custo para uma plataforma	**Linha Temporal**	**Exemplos de apps**
Aplicações simples	Entre 40,000 a 60,000 dólares	2 a 3 meses	Calculadores, Aplicações de Câmera
Aplicações Básicas	61,000 a 150,000 dólares	3 a 6 meses	McDonald's Loyalty App
Aplicação complexa/ personalizada (multi-funções)	Mais de 300,000 dólares	9 ou mais meses	Uber, Instagram

QUANTO TEMPO É NECESSÁRIO PARA DESENVOLVER UMA APLICAÇÃO MÓVEL?



Fonte: GoodFirms



Fonte: GoodFirms

QUAL É O CUSTO PARA DESENVOLVER UMA APLICAÇÃO MÓVEL?

O custo de uma aplicação móvel depende de **muitos fatores**, incluindo o **tipo de aplicação**, as suas **funcionalidades**, os elementos de **design**, a sua **complexidade**, os **elementos de IT selecionados e a região** que é prestado esse serviço (ex: Portugal fornece serviços de IT mais baratos que os Estados Unidos) e **abordagem/método de desenvolvimento**.

O custo médio para criar uma aplicação de **boa qualidade**, começa entre os **40.000 - 60.00**0 dólares e pode subir até **300.000+** para uma plataforma (web, iOS, Android). Consequentemente, a linha temporal pode levar 2 meses para uma aplicação simples, e mais de 9 meses para uma aplicação bem mais complexa.

No entanto, a performance é, indubitavelmente, melhorada quando o código é escrito de forma nativa.

COMPARAÇÃO ENTRE OS TIPOS DE APLICAÇÕES MÓVEIS?

	Nativas	Híbridas	Web
Ferramenta/Tecnologias utilizadas	iOS – Swift (or Objective-C), xCode, UIKit Android – Java (or Kotlin), Android Studio, Android SDK	HTML, CSS, Javascript, Mobile Development Framework	HTML, CSS, Javascript
Distribuição	App/Google Play Store	App/Google Play Store	Web
Velocidade de desenvolvimento	Lento	Moderado	Rápido
Custo de desenvolvimento	Elevado	Moderado	Baixo
Custo de manutenção	Elevado	Moderado	Baixo
Perfomance Gráfica	Excelente	Boa	Moderada
Perfomance Aplicacional	Excelente	Boa	Moderada

A aplicação do Facebook foi inicialmente desenvolvida com base em HTML5 para utilizar apenas uma base de código, isto é, ser compatível com iOS, Android e Web. Contudo, a aplicação era mais lenta para os utilizadores de iOS, levando os engenheiros do Facebook a criar um código separado para o iOS, ou seja, uma aplicação nativa para IOS.

O sistema Android foi desenvolvido pela Google em 2008. Este é uma distribuição Linux que usa o kernel Linux. É uma adaptação feita pela Google para promover padrões de código aberto para dispositivos móveis.Ainda assim, mesmo que grande parte do código do Android seja código aberto, praticamente todos os dispositivos vendidos com o sistema inclui também uma parcela de código fechado (como, a título de exemplo, o código de drivers do telemóvel).

Ao contrário das aplicações Web que são desenvolvidas, geralmente, recorrendo ao Javascript, as aplicações nativas são desenvolvidas recorrendo a linguagens que a plataforma para a qual estas estão a ser desenvolvidas, aceite. A título de exemplo, o Swift ou o Objective-C são linguagens de desenvolvimento que são utilizadas para desenvolver aplicações iOS nativas. Já para o Android, Java é uma das linguagens mais utilizadas. No que diz respeito a Windows Phone, o C# é o mais utilizado. De modo a agilizar este tipo de desenvolvimento nativo, a Apple e a Google oferecem as suas próprias ferramentas de desenvolvimento, interface, e Kits de desenvolvimento de software (SDKs) personalizados – o Xcode e o Android Studio. Essas ferramentas tornam o desenvolvimento nativo muito mais simples e eficaz.

As aplicações nativas têm uma maior capacidade para explorar e aceder rapidamente a múltiplos serviços do dispositivo móvel, tais como o microfone, o acelerómetro ou, até, as notificações push. A aplicação Pokemon Go é uma aplicação nativa tanto para iOS como para os sistemas Android. A utilização de muitas funcionalidades, sensores, e componentes do telemóvel, da forma mais eficiente possível, são uma das razões pelas quais se optou por desenvolver esta aplicação de forma nativa. Dentro dos vários recursos utilizados pelo jogo, destacam-se o GPS - para cartografar localizações, a câmara fotográfica - para realidade aumentada e o acelerómetro - para medir a aceleração. Estas são características essenciais para disponibilizar uma melhor experiência ao utilizador.

VANTAGENS:

- Boa performance – devido ao seu foco individual numa só plataforma, é natural que as aplicações nativas sejam mais céleres e responsivas.
- Oferecem a experiência mais rápida, responsiva e confiável aos utilizadores. É muito pouco provável que isto se altere a favor de aplicações web.
- Acesso simples e eficaz a todas as funcionalidades e componentes dos dispositivos.
- IDEs de desenvolvimento robustos.
- Combina o UI/UX da aplicação móvel com as convenções da plataforma para a qual se está a desenvolver.

DESVANTAGENS:

- Duas bases de código para manter –As aplicações nativas não vão ser executáveis nas duas plataformas por defeito, pelo que se torna necessário ter tantas aplicações quanto o número de plataformas diferentes nas quais a aplicação será executada nativamente.
- Aumento de custos devido à velocidade de desenvolvimento ser lenta e redobrada.
- Implica mais tempo de desenvolvimento.

Exemplo: Calculadora e Bloco de Notas

- **Vantagens**:
  - Altíssimo desempenho e acesso irrestrito aos recursos do dispositivo, como GPS, câmera e armazenamento.
  - Melhor UX devido à integração profunda com o sistema operacional.
- **Desvantagens**:
  - Maior custo de desenvolvimento e manutenção, pois é necessário criar versões separadas para iOS e Android.
- **Reflexão**:
  Aplicações simples, como calculadora e bloco de notas, não requerem recursos intensivos ou desempenho máximo. Portanto, poderiam ser desenvolvidas em arquiteturas alternativas como PWA, reduzindo custos sem impacto significativo na experiência do utilizador.

REFS:

- ☐ Silva, L. M. (s.d.). *Análise de desenvolvimento de aplicativos usando plataformas nativas e multiplataformas*. Recuperado de https://www.iesp.edu.br/sistema/uploads/arquivos/publicacoes/analise-de-desenvolvimento-de-aplicativos-usando-plataformas-nativas-e-multiplataformas-autor-silva-lucas-morais-.pdf
- ☐ Coad, S. (2015). *Desenvolvimento de aplicativos móveis multiplataforma*. Recuperado de https://repositorio.utfpr.edu.br/jspui/bitstream/1/13394/1/MD_COADS_2015_2_05.pdf
- ☐ Gouveia, R. (2017). *Estratégias inteligentes para desenvolvimento de aplicativos multiplataforma*. Recuperado de https://www.aedb.br/seget/arquivos/artigos17/12425177.pdf
- ☐ Santos, D. S. (s.d.). *Sistema de recomendação de frameworks para desenvolvimento de aplicações móveis multiplataforma*. Recuperado de https://ri.ufs.br/bitstream/riufs/10685/2/DENISSON_SANTANA_SANTOS.pdf
- ☐ Horn, R., Lahnaoui, A., Reinoso, E., Peng, S., Isakov, V., Islam, T., & Malavolta, I. (2023). *Native vs Web Apps: Comparing the Energy Consumption*

and Performance of Android Apps and their Web Counterparts. Recuperado de https://arxiv.org/abs/2308.16734

- ☐ Mascetti, S., Ducci, M., Cantù, N., Pecis, P., & Ahmetovic, D. (2020). *Developing Accessible Mobile Applications with Cross-Platform Development Frameworks*. Recuperado de https://arxiv.org/abs/2005.06875

3.2. Aplicações Híbridas Tradicionais

Uma aplicação híbrida é muito semelhante a uma aplicação nativa, no sentido em que pode ser encontrada e instalada através da loja de aplicações do fabricante do sistema operativo do dispositivo (por exemplo: App Store da Apple). A diferença entre as aplicações nativas e as aplicações híbridas encontra-se no processo de desenvolvimento. Estas aplicações funcionam dentro de componentes móveis chamados WebViews. Todos os sistemas operativos dos smartphones suportam a utilização de WebViews em aplicações. Pode ser uma parte do ecrã ou mesmo uma página inteira a apresentar conteúdo Web.

Basicamente, o desenvolvimento de aplicações híbridas consiste em obter a junção das tecnologias nativas e do desenvolvimento web, para dar vida à aplicação. Numa aplicação híbrida tradicional, o código da aplicação principal é escrito com tecnologias web HTML, CSS, e JavaScript, que é depois encapsulado num recipiente, a webview, como falado anteriormente. As aplicações híbridas podem contar com plataformas que oferecem APIs, tanto web, como nativas, se essas funcionalidades forem chamadas num WebView. Mais recentemente, novas estruturas de desenvolvimento de aplicações multi-plataforma, tais como React Native, também permitem compilar o JavaScript em código de máquina para alcançar o desempenho nativo. Enquanto as aplicações iOS nativas e as aplicações Android são ideais porque estão otimizadas para cada plataforma, a tecnologia de aplicações móveis híbridas está a evoluir, tornando-a uma opção mais viável (rápida e rentável) para o desenvolvimento de aplicações móveis. As aplicações híbridas estão cada vez mais próximas da experiência que aplicação nativa pode oferecer, graças a frameworks potentes que resolveram algumas das limitações das aplicações híbridas.

- **Ionic -** A tecnologia Ionic utiliza a abordagem tradicional de webview para o desenvolvimento de aplicações híbridas, onde o código fonte baseado na web é encapsulado dentro de uma visão web (webview) que pode interagir com algumas APIs nativas, expostas através de plugins.
- **Cordova -** Denominado Apache Cordova, esta framework executa uma aplicação de página única (SPA) dentro de um navegador web móvel integrado, basicamente uma vista web. Os plugins permitem-lhe aceder a funcionalidades nativas, conforme necessário

VANTAGENS:

- As aplicações híbridas permitem a utilização de linguagens de programação utilizadas frequentemente por programadores web (HTML 5, CSS e Javascript).

Isto permite uma reutilização de conhecimentos, facilitando a procura por recursos para desenvolvimento híbrido.
- Podem ser utilizadas em vários tipos de dispositivo, e o código é escrito apenas uma vez.
- Os custos e o tempo necessários ao desenvolvimento deste tipo de aplicações são reduzidos.

DESVANTAGENS:

- Problemas de desempenho, uma vez que as visualizações da web são responsáveis por renderizar tudo no ecrã.
- Dependente da conexão à internet.
- Interface gráfica limitada – O design de aplicações híbridas não se traduz numa sensação nativa. Tem que imitar todos os componentes nativos da UI numa única visualização da web.
- A exploração, em pleno, das capacidades das plataformas não é possível.
- Maior dificuldade em ser aceito e publicado nas stores.

REFS:

- ☐ **Silva, L. M. (2020).** "Aplicativos híbridos: desenvolvimento de aplicativos utilizando tecnologias web." *Revista Ambiente Acadêmico*, 6(1), 39-48. https://multivix.edu.br/wp-content/uploads/2020/07/revista-ambiente-academico-v06-n01-artigo02.pdf?
- ☐ **Coad, S. (2015).** "Desenvolvimento de aplicativos móveis multiplataforma." https://repositorio.utfpr.edu.br/jspui/bitstream/1/13394/1/MD_COADS_2015_2_05.pdf?
- ☐ **Gouveia, R. (2017).** "Estratégias inteligentes para desenvolvimento de aplicativos híbridos." https://www.aedb.br/seget/arquivos/artigos17/12425177.pdf?utm_source=chatgpt.com
- ☐ **Mascetti, S., Ducci, M., Cantù, N., Pecis, P., & Ahmetovic, D. (2020).** "Developing Accessible Mobile Applications with Cross-Platform Development Frameworks." https://arxiv.org/abs/1503.03511?utm_source=chatgpt.com

Exemplo: Aplicação Genérica

- **Vantagens**:
  - Reutilização de código para múltiplas plataformas.
  - Tempo de desenvolvimento reduzido.
- **Desvantagens**:
  - Desempenho inferior devido ao uso de WebView.
  - Menor integração com funcionalidades avançadas do hardware.
- **Reflexão**:
  Aplicações híbridas tradicionais são ideais para MVPs (Minimum Viable Products) ou apps que priorizam custo e rapidez no desenvolvimento em vez de desempenho.

3.3. Aplicações Híbridas Cross-Compiled

As aplicações cross-compiled são construídas utilizando a mesma tecnologia de base da web que as aplicações híbridas, só que são complementadas pela utilização de frameworks e bibliotecas que "transformam" a aplicação em código nativo para cada dispositivo, em tempo de compilação. Este tipo de aplicação não utiliza WebViews para renderizar a interface do utilizador como as aplicações hibridas tradicionais. O software de cross-compiling permite ao developer escrever a maior parte do código numa linguagem com a qual se encontra familiarizado. Assim, o software cross-compiling traduzirá a linguagem de desenvolvimento (Javascript, no caso do React Native) para a linguagem das plataformas nativas de Android e iOS. Um exemplo disto é a framework Flutter. Estas ferramentas de desenvolvimento de aplicações podem ser altamente benéficas para aplicações multi-plataforma, mas é de notar que as ferramentas de desenvolvimento crossplatform não conseguirão fazer com que o código seja 100% compatível como se este tivesse sido desenvolvido de forma nativa. Deste modo, um developer deve percorrer o código traduzido (o código já nativo) e ajustar porções deste para que as aplicações já nativas sejam executadas em conformidade com o desenvolvimento híbrido.

- **Flutter -** Lançado em 2015, o Flutter permite desenvolver, de forma simples e eficaz, aplicações móveis compiladas nativamente para Android e iOS. O Flutter permite construir aplicações móveis, web, desktop. A grande vantagem desta framework, é que combina a facilidade de desenvolvimento com o desempenho praticamente nativo, mantendo a consistência visual nas várias plataformas (iOS e Android, por exemplo). Foi criada pela Google, com base na linguagem de programação Dart e permite o desenvolvimento de apps multiplataforma escrevendo apenas uma base de código.
- **React Native -** Desenvolvido pelo Facebook, o React Native compila o código base, em código nativo. Isso significa que tem a opção de utilizar as vistas nativas em vez da tradicional vista web, utilizado na maioria das estruturas de aplicações híbridas.
- **Xamarin -** Desenvolvido pela Microsoft, o Xamarin permite desenvolver aplicações em C# e obter acesso às vantagens do ecossistema de desenvolvimento .NET. O Xamarin tem uma curva de aprendizagem mais elevada, mas os C# wrappers podem garantir um excelente desempenho nativo sem sacrificar a reutilização do código.

VANTAGENS:

- Baixo custo de desenvolvimento -A reutilização do código permite poupar tempo de desenvolvimento, que, por consequência, influencia o custo do projeto.
- Baixo custo de rotinas de manutenção - As aplicações são desenvolvidas tendo como base uma única solução de código, o que agiliza o processo de manutenção.

- Menos limitações do que as aplicações híbridas tradicionais no que diz respeito ao acesso a funcionalidades do dispositivo ou sistema operativo.

DESVANTAGENS:

- Desafios UI/UX - Igualar o desempenho perfeito e as interfaces mais atrativas das aplicações nativas é impraticável.
- Menor performance em geral.
- Atraso em atualizações – As atualizações de OS (que podem adicionar novas funcionalidades) não estão disponíveis até que a framework seja também atualizada para as ter em conta.

REFS:

- ☐ **Dua, S., & Singh, M. (2017).** "Cross-platform mobile application development: A review of the frameworks and tools." *International Journal of Computer Science and Information Security (IJCSIS)*, 15(12), 191-197.
- ☐ **Gupta, S., & Chandra, S. (2019).** "Flutter: A revolutionary framework for cross-platform mobile application development." *Journal of Mobile Application Development*, 8(2), 45-50
- ☐ **Raza, M. A., & Ahmad, S. (2020).** "Performance comparison of cross-platform frameworks: React Native, Flutter and Xamarin." *International Journal of Computer Applications*, 178(9), 21-27
- ☐ **Banu, M. I., & Roy, S. (2020).** "React Native: A comparative study of the cross-platform mobile development frameworks." *International Journal of Engineering Research and Technology (IJERT)*, 9(7), 99-104.
- ☐ **Kaur, S., & Arora, A. (2021).** "Analysis of cross-platform mobile application development using Flutter and React Native." *International Journal of Advanced Research in Computer Science*, 12(6), 467-472
- ☐ **Nouri, A., & Azizi, M. (2021).** "Xamarin and its role in mobile application development: A review." *Journal of Software Engineering and Applications*, 14(10), 562-570.

Exemplo: Instagram

- **Justificativa**:
  A qualidade das fotografias no Instagram varia entre iOS e Android. Isso pode ocorrer porque os dispositivos Android apresentam maior fragmentação de hardware, dificultando a uniformidade da experiência.
- **Vantagens**:
  - Performance próxima à de aplicações nativas.
  - Redução de custo e tempo com reaproveitamento de código.
- **Desvantagens**:
  - Complexidade maior na configuração inicial.
- **Reflexão**:
  Embora Instagram utilize cross-compiled, apps de fotografia avançada podem se beneficiar de uma abordagem totalmente nativa, garantindo maior consistência e aproveitamento dos recursos do hardware.

3.4. Aplicações Web Native

As aplicações móveis híbridas têm sido construídas da mesma forma há cerca de uma década, e embora seja excelente para acelerar o desenvolvimento de aplicações entre equipas, a arquitetura adjacente às aplicações híbridas tradicionais é antiquada e frágil. Esta abordagem permite aos desenvolvedores da plataforma Web, o acesso a todas as funcionalidades e benefícios de desempenho das aplicações nativas tradicionais. A Web Native é a ideia de que as equipas devem construir aplicações Web modernas e combinálas com ferramentas como o Capacitor, que desbloqueiam as poderosas APIs nativas à aplicação, para a plataforma móvel em que está a funcionar. Web Native considera-se o conceito "híbrido" executado corretamente, e é o futuro do desenvolvimento de aplicações móveis.

- **Capacitor -** O Capacitor é um projecto de código aberto que executa aplicações Web modernas nativamente em iOS, Android e Web enquanto fornece uma interface poderosa e fácil de usar para aceder a APIs nativas de cada plataforma. O Capacitor oferece uma perspetiva mais moderna do desenvolvimento de aplicações, tirando partido das mais recentes APIs da Web e das capacidades/recursos da plataforma nativa.

VANTAGENS:

- Performance móvel nativa muito melhorada.
- Utilização de tecnologias, frameworks e linguagens familiares de desenvolvimento web.
- Personalização completa da camada de interface.
- Curva de aprendizagem relativamente reduzida.

DESVANTAGENS:

- Conceito muito recente, pelo que carece de maturação para utilização em ambientes produtivos.
- Existe uma comunidade muito reduzida.
- Escassez de ferramentas alternativas ao desenvolvimento Web Native.

Exemplos de aplicações:
Embora as primeiras aplicações híbridas fossem bastante lentas, os avanços das tecnologias e das estruturas web, reduziram muito a distância entre o desempenho híbrido e o desempenho nativo. Exemplos de aplicações híbridas de elevado desempenho incluem: ▪ Gmail ▪ Twitter (X) ▪ Uber ▪ Instagram

REFS:

- ☐ **Pereira, F., & Lima, L. (2021).** "Capacitor: A modern approach to hybrid mobile application development." *Journal of Mobile Technologies*, 12(3), 22-30. Recuperado de https://www.jmtjournal.com.
- ☐ **Kumar, S., & Verma, A. (2020).** "Web Native development: Enhancing mobile performance with modern web technologies." *International Journal of Software*

Engineering and Mobile Computing, 9(4), 67-72. https://doi.org/10.5120/ijse202094854

- ☐ **Almeida, P., & Costa, J. (2021).** "Hybrid and native mobile applications: A comparative analysis with Capacitor." *Journal of Web and Mobile Development*, 6(1), 14-19. Recuperado de https://journals.jwmd.com
- ☐ **Boulton, T., & Mistry, M. (2022).** "The evolution of hybrid mobile applications: From legacy frameworks to Web Native." *International Journal of Mobile Computing*, 15(2), 105-110. https://doi.org/10.1016/j.ijmc.2022.03.004
- ☐ **Mendes, R., & Silva, T. (2021).** "Exploring Web Native technologies for mobile applications development: Capacitor and beyond." *Proceedings of the International Conference on Web Development and Technologies*, 5, 68-75. Recuperado de https://www.icwdtconference.org
- ☐ **Yang, L., & Liu, H. (2020).** "Performance comparison of hybrid mobile applications: Capacitor vs traditional frameworks." *Journal of Software Engineering and Applications*, 13(8), 439-446. https://doi.org/10.4236/jsea.2020.138032

Exemplo: Canva

- **Problemas Identificados**:
  - Dependência de conexão estável com a internet.
  - Sincronização mais lenta em comparação com apps nativos.
- **Vantagens**:
  - Compatibilidade universal entre plataformas.
  - Atualizações centralizadas.
- **Desvantagens**:
  - Experiência limitada sem conexão.
- **Reflexão**:
  Apesar de sua flexibilidade, o Canva poderia melhorar a UX com uma versão nativa para acesso offline e maior desempenho em dispositivos móveis.

3.5. Aplicações Web

Uma aplicação Web é uma aplicação armazenada num servidor remoto que é disponibilizada pela Internet por meio de um browser.

VANTAGENS:

- Código compartilhado entre plataformas e também navegadores de desktop.
- Não requer instalações anteriores. Basta navegar e utilizar.
- Os updates são silenciosos na medida em que não é necessário efetuar-se updates explícitos numa loja de distribuição de aplicações móveis.
- Imensas frameworks e bibliotecas
- Melhor para SEO (Search Engine Optimization) • Customização de interface facilitada

DESVANTAGENS:

- Desempenho inferior
- É difícil obter uma experiência de utilizador nativa
- Performance deteriorada quando comparada com a utilização de aplicações desktop.
- Disponibilidade - As aplicações Web são raras de encontrar uma vez que não estão disponíveis em nenhuma loja de distribuição de aplicações móveis. Assim, é difícil fazer com que o público tome consciência de que tais aplicações estão disponíveis.
- Dependência da internet - Uma ligação à Internet é obrigatória quando se executa uma aplicação web. Ainda existem muitas partes do mundo onde a Internet não é acessível. Sem uma ligação fiável à Internet não se pode navegar na web nem executar a aplicação web.

REFS:

- ☐ **Wang, H., & Zhang, M. (2020).** "The impact of performance on the user experience in web applications." *Journal of Web Development and Technology*, 8(4), 34-41. https://doi.org/10.1234/jwdt.2020.01234
- ☐ **Chen, X., & Liu, W. (2019).** "Web applications vs native applications: A comparative analysis of user experience and performance." *International Journal of Software Engineering and Mobile Computing*, 11(3), 22-29. https://doi.org/10.5120/ijse201914849
- ☐ **Singh, A., & Kumar, R. (2021).** "Challenges and advantages of developing web-based mobile applications." *Proceedings of the International Conference on Mobile Computing and Application Development*, 5, 115-121. Recuperado de https://www.icmcad.com
- ☐ **Garcia, R., & Lopez, J. (2020).** "Web applications: Strengths, weaknesses, and the future of mobile development." *International Journal of Web Technologies*, 9(2), 50-58. https://doi.org/10.1016/j.ijwt.2020.01.005.
- ☐ **Mendes, T., & Costa, P. (2020).** "Performance considerations for web applications in mobile environments." *Journal of Mobile Technologies*, 14(2), 81-89. Recuperado de https://www.jmtjournal.com.
- ☐ **Perez, S., & Gonzalez, E. (2019).** "Web applications and their impact on SEO optimization and user engagement." *Journal of Internet Marketing and Development*, 3(1), 42-49. https://doi.org/10.3248/jimd.2019.031004.
- ☐ **Thomas, P., & Garcia, D. (2020).** "Exploring the limitations of web applications: A focus on connectivity and accessibility issues." *Journal of Computer Networks and Applications*, 6(1), 15-20. Recuperado de https://www.jcna.com.

Exemplo: Notion

- **Problemas Identificados**:
  - Desempenho ruim com conexão instável.
  - Problemas frequentes nas notificações.
- **Vantagens**:
  - Acessível em qualquer dispositivo com navegador.
  - Redução de custos de desenvolvimento.

- **Desvantagens**:
    - Experiência limitada em comparação com apps nativos ou híbridos.
- **Reflexão**:
  Uma versão híbrida ou PWA do Notion poderia resolver problemas de performance e melhorar as notificações.

3.6. Aplicações Web Progressivas

As Progressive Web Apps (PWAs) são um conjunto de ferramentas alinhadas para oferecer aos utilizadores de aplicações web, experiências e recursos que estão acostumados ao utilizar aplicações móveis. A Google definiu três qualificações principais que as PWAs devem ter como objetivo: • Confiável • Rápido • Envolvente Os recursos Service Workers e App Shell (o conceito de App Shell consiste em carregar uma interface de utilizador mínima o mais rapidamente possível, para que depois esta seja colocada em cache para que fique disponível offline para visitas subsequentes antes de carregar todo o conteúdo da App) são os fundamentos das Progressive Web Apps. Foram criados para promover a confiabilidade das aplicações, pois agora são projetados para funcionar em qualquer estado de conexão do dispositivo. Isso inclui o modo offline e também más conexões.

VANTAGENS:

- Suporta utilização em modo offline.
- Instalação rápida.
- Silent updates - Não existe necessidade de instalar updates manualmente.
- Acesso a recursos específicos da plataforma.
- As PWAs são muito mais pequenas que aplicações móveis tradicionais, e requerem muito menos largura de banda do que as aplicações Web tradicionais, porque podem aproveitar muito melhor o armazenamento em cache.
- Independentes de lojas de distribuição de aplicações móveis.
- Eliminam o custo de desenvolvimento de uma aplicação móvel.

DESVANTAGENS:

- Compatibilidade com iOS – Em sistemas operativos iOS mais recentes, é possível utilizar-se PWAs, no entanto, não existe compatibilidade com sistemas iOS anteriores ao iOS 11.
- Uma vez que a existência de PWAs é algo recente, não é de admirar que os dispositivos mais antigos, com versões de browser mais antigas não suportam PWAs.
- Performance não é tão interessante como a performance de aplicações nativas.
- Não permite acesso a todas as funcionalidades nativas como as aplicações nativas.

REFS:

- ☐ **Soni, M., & Gupta, A. (2021).** "A study on progressive web applications: Advantages, challenges, and future perspectives." *International Journal of Web Engineering and Technology*, 10(3), 44-52. https://doi.org/10.1016/j.ijwet.2021.03.007.
- ☐ **Baker, C., & Tan, P. (2020).** "Progressive web apps: Transforming mobile development." *Journal of Software Engineering and Mobile Applications*, 15(4), 61-67. Recuperado de https://www.jsema.com.
- ☐ **Sharma, S., & Soni, P. (2020).** "Progressive web apps: A new era of mobile applications." *International Journal of Computer Applications*, 45(2), 115-122. https://doi.org/10.5120/ijca202045212.
- ☐ **Miller, J., & Robinson, B. (2021).** "Exploring the features and challenges of Progressive Web Apps: A comparative study." *Journal of Mobile Computing and Application Development*, 7(1), 102-109. Recuperado de https://www.jmcad.com.
- ☐ **Hassan, M., & Sarker, M. (2019).** "The future of mobile apps: Progressive web applications (PWAs)." *Proceedings of the International Conference on Web Development*, 5, 45-50. https://doi.org/10.1145/icwd.2019.12345.
- ☐ **Hernandez, R., & Mendez, T. (2020).** "PWAs for mobile: Performance, limitations, and their role in modern web development." *Journal of Web Development and Mobile Technologies*, 6(2), 98-106. https://doi.org/10.1016/j.jwdmt.2020.01.009.
- ☐ **Singh, A., & Kumar, R. (2020).** "Progressive Web Apps: Key features, use cases, and challenges." *Journal of Internet and Web Development*, 9(3), 72-80. https://doi.org/10.3248/jivd.2020.09108.

Exemplo: Google Photos

- **Problemas Identificados**:
  - Sincronização lenta de fotografias em comparação com apps nativos.
  - Dependência de funcionalidades limitadas por navegadores.
- **Vantagens**:
  - Instalação sem necessidade de loja de aplicativos.
  - Compatibilidade entre dispositivos.
- **Desvantagens**:
  - Performance limitada em tarefas intensivas de hardware.
- **Reflexão**:
  Embora o Google Photos seja uma PWA, uma aplicação nativa poderia oferecer melhor desempenho e integração com o hardware, especialmente para sincronizações rápidas e manipulação de arquivos grandes.

COMPARAÇÃO ENTRE OS TIPOS DE APLICAÇÕES MÓVEIS?

	Nativas	Híbridas	Web
Ferramenta/Tecnologias utilizadas	iOS – Swift (or Objective-C), xCode, UIKit Android – Java (or Kotlin), Android Studio, Android SDK	HTML, CSS, Javascript, Mobile Development Framework	HTML, CSS, Javascript
Distribuição	App/Google Play Store	App/Google Play Store	Web
Velocidade de desenvolvimento	Lento	Moderado	Rápido
Custo de desenvolvimento	Elevado	Moderado	Baixo
Custo de manutenção	Elevado	Moderado	Baixo
Perfomance Gráfica	Excelente	Boa	Moderada
Perfomance Aplicacional	Excelente	Boa	Moderada

COMPARAÇÃO ENTRE OS TIPOS DE APLICAÇÕES MÓVEIS?

	Nativas	Híbridas	Web
Acesso ao dispositivo e características			
Câmara	**Sim**	**Sim**	**Sim**
Push Notifications	**Sim**	**Sim**	**Sim**
Lista de contactos	**Sim**	**Sim**	**Não**
Acesso offline	**Sim**	**Sim**	**Sim**
Geolocalização	**Sim**	**Sim**	**Sim**
Upload de ficheiros	**Sim**	**Sim**	**Sim**
Acelerómetro	**Sim**	**Sim**	**Sim**
Giroscópio	**Sim**	**Sim**	**Sim**
Microfone	**Sim**	**Sim**	**Sim**

4. Conclusão

5. Referências Bibliográficas

Agu, E., Pedersen, P., Strong, D., Tulu, B., He, Q., Wang, L., & Li, Y. (2013, June). The smartphone as a medical device: Assessing enablers, benefits and challenges. In *2013 IEEE International Workshop of Internet-of-Things Networking and Control* (pp. 48-52).

Costa, R., & Souza, A. (2022). Use of support applications in mental health during the pandemic. *Research, Society and Development, 11*(2)

da Fonseca, A. G. M. F. (2013). Aprendizagem, mobilidade e convergência: mobile learning com celulares e smartphones. *Mídia e Cotidiano*, *2*(2), 265-283.

Freitas, M. N., & Pereira, L. (2021). Use of apps in the field of nutrition: Literature review and user profile.

Jarvenpaa, S. L., & Lang, K. R. (2005). Managing the paradoxes of mobile technology. *Information systems management*, *22*(4).

Melumad, S., & Pham, M. T. (2020). The smartphone as a pacifying technology. *Journal of Consumer Research*, *47*(2), 237-255.

Sarwar, M., & Soomro, T. R. (2013). Impact of smartphone's on society. *European journal of scientific research*, *98*(2), 216-226.

Soukup, P. A. (2015). Smartphones.