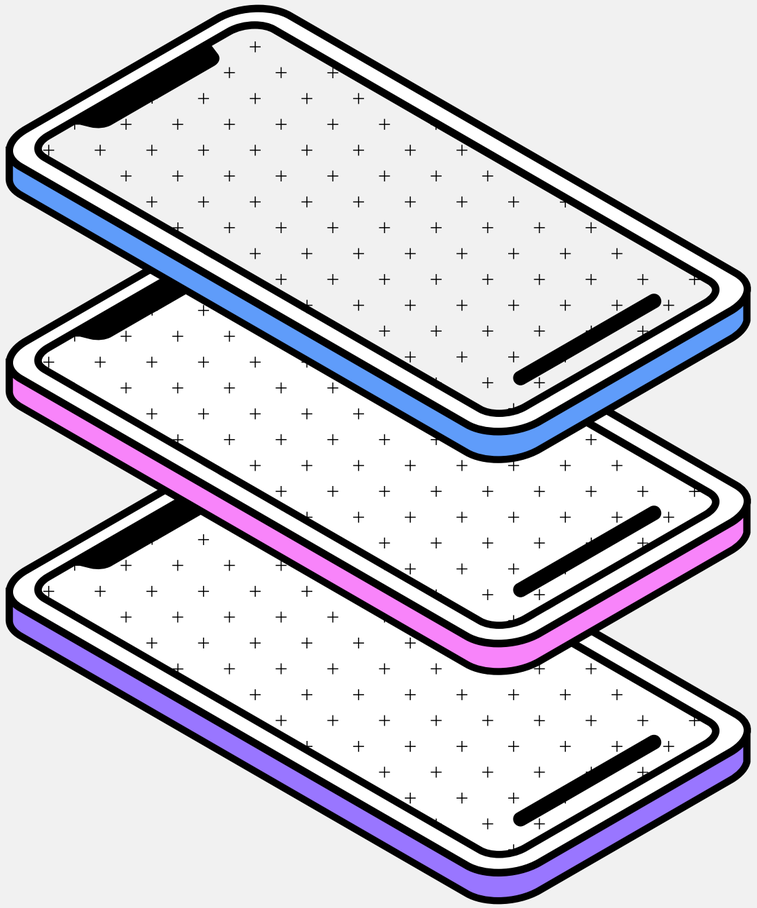
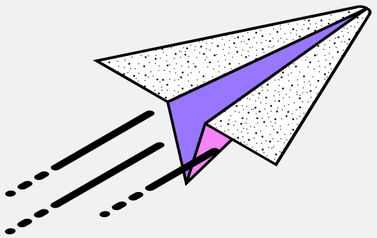
REACT NATIVE VS FLUTTER VS XAMARIN

Trabalho 3

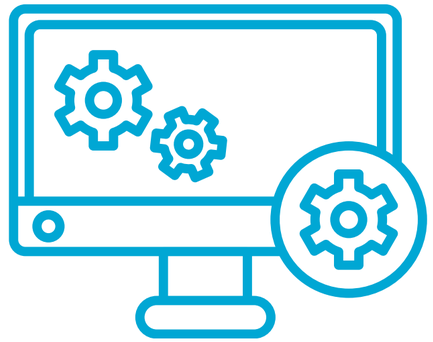
Ana Carolina, Emelly Yasmin, Gabriela Cristina, Gabrielly, Marco Gitti, Nabila Sampaio e Thaysa Vitória.



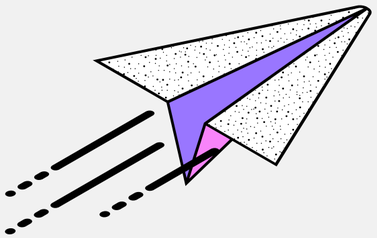


Contextualização do Desenvolvimento Multiplataforma

Desenvolvimento de Software Multiplataforma:



- Criação de aplicativos para várias plataformas, sem reescrita.
- Funcional em computadores, smartphones e tablets.
- Maior flexibilidade e alcance para os desenvolvedores.
- Economia de tempo, recursos e energia.
- Ferramentas e tecnologias para consistência em diferentes sistemas.
- Eficiência no desenvolvimento e manutenção.
- Base de código única compartilhada entre sistemas.
- Facilita implementação de atualizações e correções.

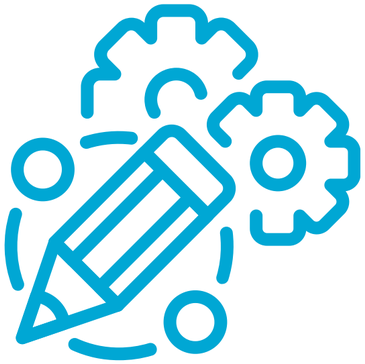


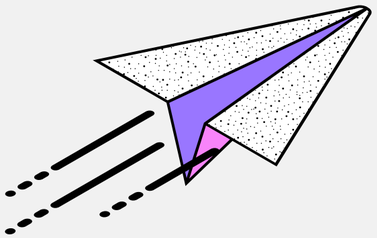
Justificativa e Objetivo do Estudo

- Comparação de Frameworks Multiplataforma para Aplicações Móveis
- **Objetivo:** Analisar React Native, Flutter e Xamarin.
- **Foco:** Características, desempenho, facilidade de uso e suporte.
- **Experiência do Usuário:** Consistência em diversas plataformas.

- **Estudo de Caso**

- **Metodologia:** Análise prática das vantagens e desvantagens.
- **Resultados:** Contribuição para uma compreensão abrangente das opções no mercado.





Visão Geral do Desenvolvimento Multiplataforma

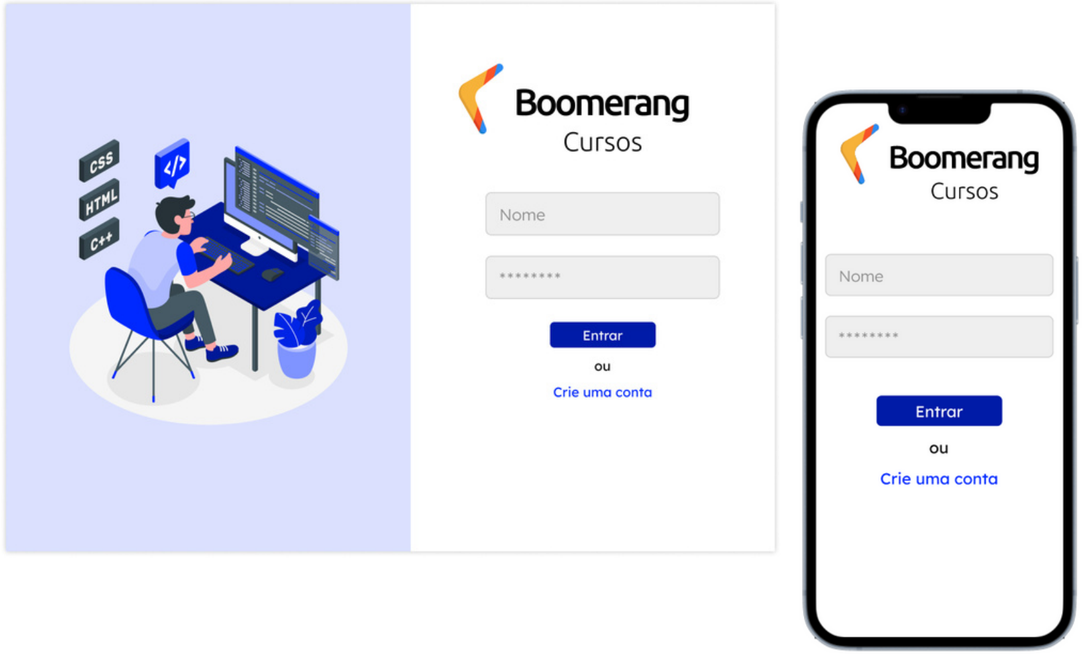
Desenvolvimento Nativo

- Desenvolvimento de aplicativos utilizando as linguagens de programação e frameworks nativos de cada plataforma, proporcionando um alto desempenho e integração total com as características da plataforma, mas requer habilidades específicas para cada plataforma e envolve mais tempo e esforço para desenvolver e manter múltiplas versões do aplicativo.

Fonte: Desenvolvido pelos autores.

Desenvolvimento Cross-Platform

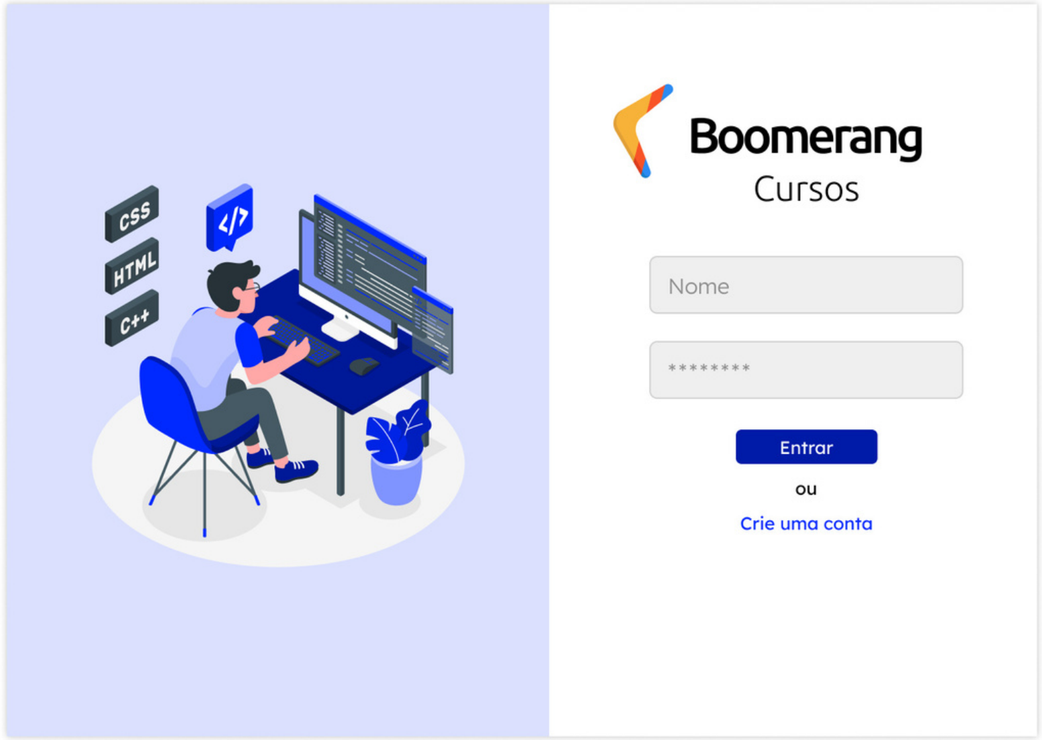
- Utilização de frameworks e tecnologias que permitem escrever código uma vez e executá-lo em múltiplas plataformas. Isso pode ser feito através de tecnologias como Xamarin (C#), React Native (JavaScript), Flutter (Dart).



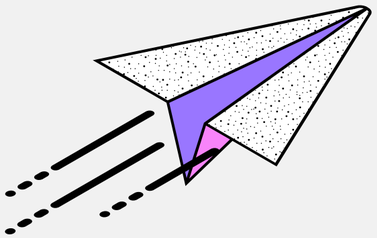
Fonte: Desenvolvido pelos autores.

Desenvolvimento Web

- Construção de aplicativos como páginas da web responsivas ou aplicativos da web progressivos que podem ser acessados através de navegadores em diferentes plataformas. Isso oferece uma abordagem altamente portátil, mas pode ter limitações de desempenho e acesso a recursos do dispositivo.



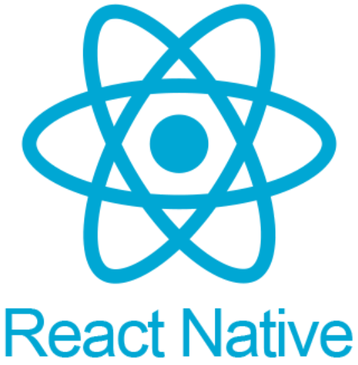
Fonte: Desenvolvido pelos autores.



Descrição dos Frameworks

React Native

- Framework de código aberto (open-source) desenvolvido pela Meta (Facebook).
- É capaz de criar aplicativos para iOS e Android, utilizando JavaScript e React, além de oferecer uma abordagem de desenvolvimento mais eficiente.
- O framwork também recebeu, ao longo dos anos, várias atualizações de novas funcionalidades e APIs, incluindo APIs do Google e do Google Maps.
- inclui melhorias no suporte ao TypeScript, proporcionando uma experiência de desenvolvimento mais robusta e tipada.



Xamarin

- Utiliza linguagem C# para maior portabilidade com outros dispositivos.

- Adota padrões MVC (Model-View-Controller) para separação clara entre as partes (camadas) do aplicativo.

- Criação de Interfaces de Usuário Nativas, permitindo a personalização de estilos de interface.

- Com o Visual Studio, .NET, C# e Xamarin Studio, temos uma combinação poderosa e precisa para desenvolver aplicativos, de forma rápida e eficaz.

Flutter

- Framework de código aberto (open-source), desenvolvido pela Google, de código aberto. Utiliza a linguagem de programação Dart, também desenvolvida pela Google.

- Conectividade com APIs e integrações com diversos serviços.

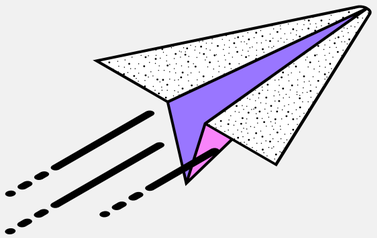
- Interface de Usuário atrativa e alta performance de qualidade.

- **Opções de Estados do Flutter:**

- **setState:** Notificar o framework de que um estado interno foi modificado/atualizado.

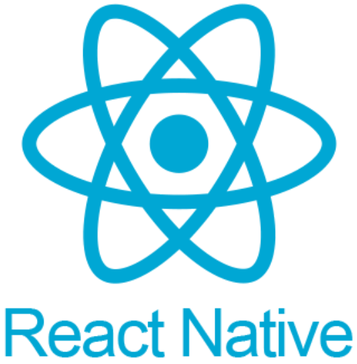
- **Provider:** Padrão de Gerenciamento de Estado que visa fornecer um compartilhamento de dados entre Widgets.

- **Redux:** Arquitetura de Gerenciamento de Estado utilizada em aplicativos Flutter.



Métodos de Avaliação de Desempenho

React Native



Profiler do React:

Permite analisar o desempenho da renderização e identificar componentes que podem estar causando gargalos de desempenho.

Chrome DevTools:

Inclui ferramentas para análise de desempenho, como o "Performance" e o "Memory" tabs, que podem ser usadas para identificar problemas de desempenho.

Benchmarking

Desenvolva benchmarks específicos para as partes críticas da sua aplicação e execute-os regularmente para medir o desempenho ao longo do tempo e identificar regressões.

Flutter

Flutter Performance Monitor:

Ferramenta oferecida pelo flutter, que permite monitorar métricas de desempenho em tempo real, como FPS (frames por segundo), uso de CPU e memória.

Testes de Carga:

testes de carga para simular o comportamento do aplicativo em condições de uso intensivo e identificar possíveis problemas de desempenho, como vazamentos de memória ou lentidão na resposta do servidor.

Perfil de Desempenho do Dart DevTools:

inclui um profiler de desempenho que pode ser usado para identificar gargalos de desempenho no código Dart de um aplicativo Flutter.

Xamarin

Xamarin Profiler:

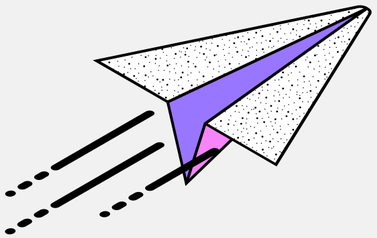
Permite analisar o desempenho do aplicativo Xamarin em tempo real. Ele pode ser usado para identificar gargalos de desempenho, vazamentos de memória e otimizar o uso de recursos do dispositivo.

Testes de Carga:

Usado para simular o comportamento do aplicativo em condições de uso intensivo e identificar possíveis problemas de desempenho, como vazamentos de memória ou lentidão na resposta do servidor.

Análise de Bundle:

Ferramentas para analisar o tamanho e a composição do pacote de aplicativos Xamarin e identificar oportunidades de otimização,como a remoção de dependências não utilizadas



Métodos de Avaliação de Usabilidade

Testes de Usabilidade:

Teste com usuários para identificar problemas de usabilidade. Isso pode ser feito através de entrevistas, observação direta do usuário interagindo com a aplicação

A/B Testing:

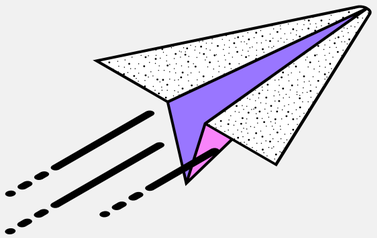
Experimentos A/B para comparar diferentes designs ou funcionalidades e determinar qual gera uma melhor experiência para o usuário.

Análise de Mapa de Calor e Gravação de Sessão:

Ferramentas como Hotjar ou Crazy Egg para analisar mapas de calor e gravações de sessões de usuários interagindo com a aplicação e identificar padrões de comportamento e áreas problemáticas.

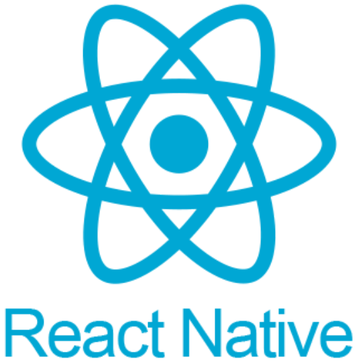
Análise Heurística:

Princípios de design e heurísticas de usabilidade para avaliar a interface do usuário. Existem várias listas de heurísticas disponíveis, como as heurísticas de Nielsen.



Avaliação de Desempenho

React Native



Profiler do React:

Permite analisar o desempenho da renderização e identificar componentes que podem estar causando gargalos de desempenho.

Chrome DevTools:

Inclui ferramentas para análise de desempenho, como o "Performance" e o "Memory" tabs, que podem ser usadas para identificar problemas de desempenho.

Benchmarking

Desenvolva benchmarks específicos para as partes críticas da sua aplicação e execute-os regularmente para medir o desempenho ao longo do tempo e identificar regressões.

Flutter

Flutter Performance Monitor:

Ferramenta oferecida pelo flutter, que permite monitorar métricas de desempenho em tempo real, como FPS (frames por segundo), uso de CPU e memória.

Testes de Carga:

testes de carga para simular o comportamento do aplicativo em condições de uso intensivo e identificar possíveis problemas de desempenho, como vazamentos de memória ou lentidão na resposta do servidor.

Perfil de Desempenho do Dart DevTools:

inclui um profiler de desempenho que pode ser usado para identificar gargalos de desempenho no código Dart de um aplicativo Flutter.

Xamarin

Xamarin Profiler:

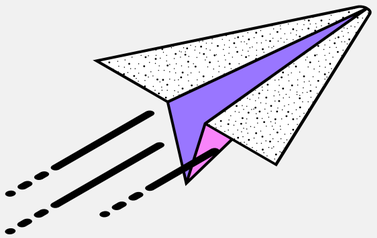
Permite analisar o desempenho do aplicativo Xamarin em tempo real. Ele pode ser usado para identificar gargalos de desempenho, vazamentos de memória e otimizar o uso de recursos do dispositivo.

Testes de Carga:

Usado para simular o comportamento do aplicativo em condições de uso intensivo e identificar possíveis problemas de desempenho, como vazamentos de memória ou lentidão na resposta do servidor.

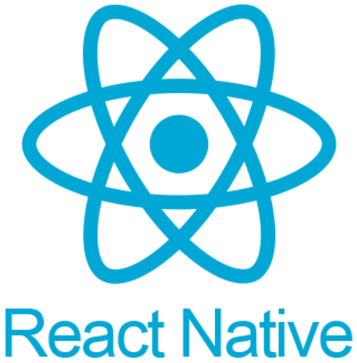
Análise de Bundle:

Ferramentas para analisar o tamanho e a composição do pacote de aplicativos Xamarin e identificar oportunidades de otimização,como a remoção de dependências não utilizadas



Vantagem, Desvantagens e Implicações Gerais

React Native



Vantagens:

- Grande comunidade e suporte.
- Boa performance com componentes nativos.
- Desenvolvimento rápido e reaproveitamento de código.

Desvantagens:

- Dependência de módulos nativos.
- Curva de aprendizado para não familiarizados com React.

Implicações Gerais:

- Rápido desenvolvimento para apps de média complexidade.

Flutter

Vantagens:

- Hot-reloading para desenvolvimento rápido.
- Interface consistente em diferentes plataformas.
- Compilação para código nativo, alta performance.

Desvantagens:

- Tamanho do aplicativo pode ser maior.
- Menos módulos prontos em comparação com outros.

Implicações Gerais:

- Ideal para interfaces complexas e visualmente impressionantes.

Xamarin

Vantagens:

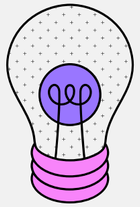
- Uso de uma linguagem (C#) em todo o aplicativo.
- Integração com .NET e acesso total às APIs nativas.

Desvantagens:

- Comunidade menor comparada a outras soluções.
- Curva de aprendizado para não familiarizados com C#.

Implicações Gerais:

- Indicado para empresas com experiência em .NET e C#.

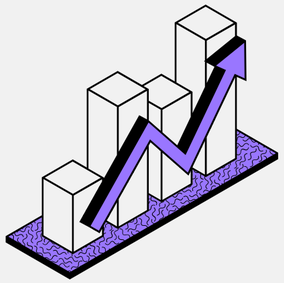


Estudos de comparação de cada Framework

Popularidade de cada Framework

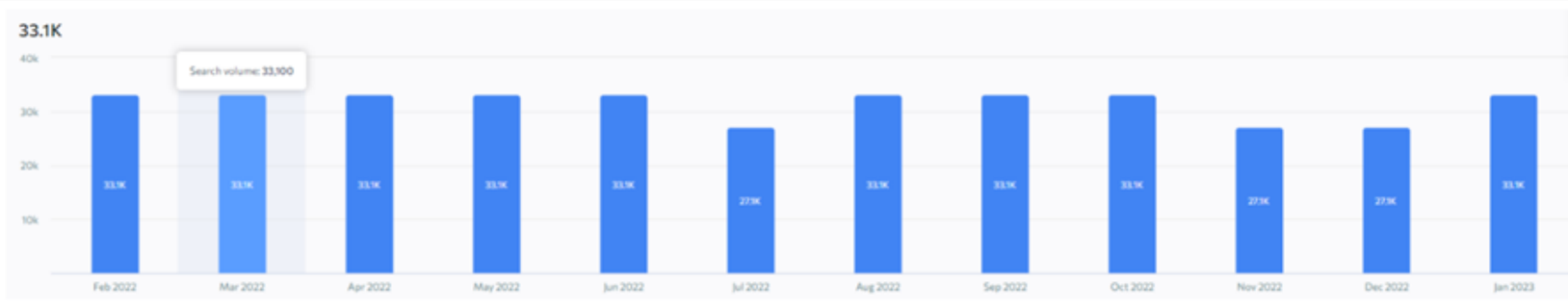
Plataforma	**React Native**	**Flutter**	**Xamarin**
GitHub	110 mil estrelas.	154 mil estrelas.	5,7 mil estrelas.
Reddit	104 mil usuários na comunidade.	109 mil usuários na comunidade.	5,7 mil usuários na comunidade.
StackOverflow	131.539 perguntas marcadas com [react-native].	162.447 perguntas marcadas com [flutter].	50.346 perguntas marcadas com [xamarin].
X (Antigo Twitter)	177,6 mil seguidores.	232,9 mil seguidores.	70 mil seguidores.

Fonte: https://themobilereality.com/blog/xamarin-vs-flutter-vs-react-native



Quantidades de pesquisas realizadas no Google sobre cada Framework (EUA)

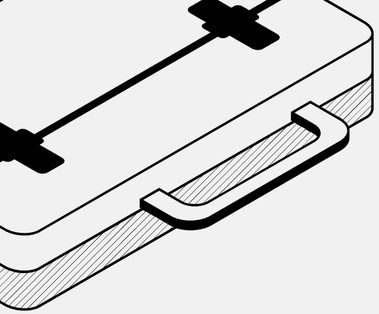
React Native (Fevereiro de 2022 - Dezembro 2022)



Flutter (Março de 2022 - Fevereiro 2023)

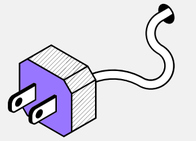
Xamarin (Setembro de 2019 - Agosto 2020)

Fonte: https://themobilereality.com/blog/xamarin-vs-flutter-vs-react-native



React Native vs Flutter

Quando escolher:



React Native

Se você já usa o React Native para desenvolver aplicativos web e quer uma maneira parecida de criar aplicativos móveis, especialmente se você valoriza a rapidez no desenvolvimento. Além disso, se é importante integrar muitas bibliotecas de terceiros de forma eficiente e se seu público inclui dispositivos mais antigos, o React Native pode ser uma escolha ideal para o seu projeto.

Flutter

Se você quer criar um aplicativo com bom desempenho e que seja pequeno, especialmente para dispositivos mais recentes, e prefere usar uma estrutura que está constantemente sendo atualizada, o Flutter pode ser a melhor opção para o seu projeto.

Fonte: https://themobilereality.com/blog/xamarin-vs-flutter-vs-react-native

Estudo de Caso Tela de Login

Comparação Geral de cada Framework

	React Native	Flutter	Xamarin
Quantidade de Código	É relativamente conciso devido ao uso de hooks e componentes funcionais.	É comparativamente mais curto, graças à sua abordagem declarativa e uso de widgets.	É mais extenso devido à natureza mais verbosa da linguagem C# e do XAML.
Vantagens	Código feito em Java (popular), vários desenvolvedores,e bibliotecas, desenvolvimento rápido por causa de recursos	Código feito em Dart (eficiente), bom desempenho por possuir seus próprios mecanismos, desenvolvimento rápido.	Código feito em C# (popular), vários desenvolvedores, acesso às APIs nativas das plataformas, desempenho sólido.
Desvantagens	Ponte nativa afeta desempenho em aplicativos complexos e gerenciamento de estado pode se tornar complicado em aplicativos maiores.	Ecossistema ainda em crescimento, aprender Dart para quem não está familiarizado , arquitetura de widgets.	Mais código para escrever (XAML, C#), aprender XAML e C# para quem não está familiarizado.

Diferenças

Lógica de Programação

A lógica para a validação de login é semelhante em todos.

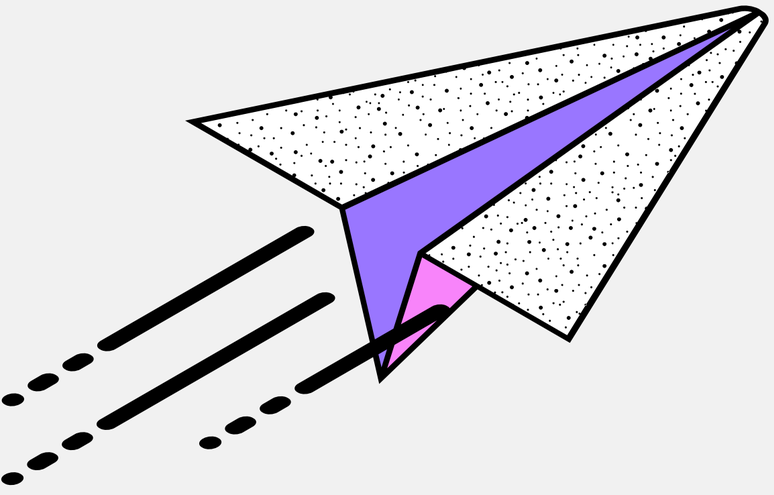
Estrutura de Código

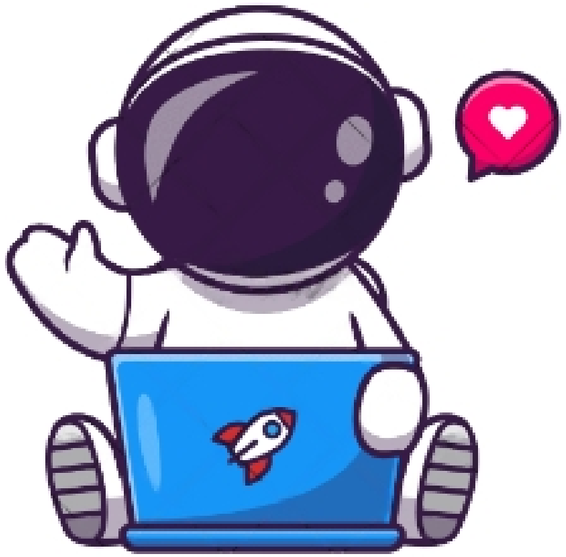
React Native usa hooks de estado, Xamarin utiliza propriedades e eventos, e Flutter possui seu próprio sistema de gerenciamento de estado.

Estilização

A maneira como os componentes de interface do usuário são definidos e estilizados também difere, o que resulta em diferentes abordagens na criação da interface de login.

Conclusão





Qual a melhor linguagem de programação para se criar um aplicação?