INDOTERA
Indonesian Dissemination of Engineering and Applied Technology Research Applications

ANALISIS PERBANDINGAN FRAMEWORK PENGEMBANGAN APLIKASI MOBILE CROSS-PLATFORM: STUDI KASUS FLUTTER DAN REACT NATIVE

Farrel Aji Wicaksono[1], Ranu Riang Prasetya[2], Muhammad Naufal Arif Fadilah[3]
[1,2,3*] D3 Teknik Informatika, Sekolah Vokasi, Universitas Sebelas Maret Surakarta (UNS)
*Co-author: masparel0@gmail.com

Informasi Artikel

Tanggal:
Dikirim : 28 Juni 2025
Direvisi : 15 Juli 2025
Diterima : 22 Juli 2025

Kata Kunci:
framework, mobile, cross-platform, flutter, react native, aplikasi, performa, pengembangan.

Keyword:
framework, mobile, cross-platform, flutter, react native, application, performance, development

Abstrak

Penelitian ini membandingkan Flutter dan React Native dalam hal efisiensi pengembangan, performa aplikasi, dan kemudahan pemeliharaan jangka panjang. Prototipe aplikasi e-commerce identik dikembangkan dengan kedua framework dan dianalisis secara kuantitatif serta kualitatif, mencakup performa runtime, waktu pengembangan, jumlah baris kode, kurva pembelajaran, dan kemudahan debugging. Hasil menunjukkan Flutter unggul dalam rendering (rata-rata 58 fps) dan konsistensi UI, sementara React Native lebih efisien dalam pengembangan, terutama bagi developer JavaScript. Temuan ini memberikan panduan bagi tim dan organisasi dalam memilih framework sesuai kebutuhan proyek, keahlian tim, dan tujuan jangka panjang.

Abstract

This study compares Flutter and React Native in terms of development efficiency, application performance, and long-term maintainability. Identical e-commerce prototypes were developed using both frameworks and analyzed through quantitative and qualitative methods, covering runtime performance, development time, lines of code, learning curve, and ease of debugging.
The results show that Flutter excels in rendering (average 58 fps) and UI consistency, while React Native offers greater development efficiency, especially for JavaScript developers. These findings serve as a guide for teams and organizations in choosing the right framework based on project needs, team expertise, and long-term goals.

1. PENDAHULUAN

Perkembangan teknologi mobile telah mengalami pertumbuhan yang signifikan dalam beberapa tahun terakhir. Menurut data terkini, pendapatan aplikasi mobile diperkirakan akan mencapai lebih dari

$522 miliar pada tahun 2025. Hal ini mendorong kebutuhan akan framework pengembangan yang efisien dan dapat mengakomodasi multiple platform secara bersamaan. Pengembangan aplikasi mobile tradisional mengharuskan developer untuk membuat aplikasi terpisah untuk setiap platform (iOS dan Android), yang memerlukan waktu, sumber daya, dan keahlian yang berbeda. Framework cross-platform muncul sebagai solusi untuk mengatasi tantangan ini dengan memungkinkan developer menulis kode sekali dan menjalankannya di multiple platforms.

Pengembangan aplikasi mobile tradisional memerlukan pembuatan kode yang berbeda untuk setiap platform, yang tidak hanya meningkatkan biaya tetapi juga kompleksitas proyek. Framework cross-platform seperti Flutter dan React Native muncul sebagai solusi dengan memungkinkan penggunaan basis kode tunggal untuk multiple platforms. Namun, pemilihan framework yang tepat sering kali menjadi dilema bagi developer, terutama ketika mempertimbangkan faktor seperti performa, kemudahan pengembangan, dan kompatibilitas dengan kebutuhan proyek.

Data dari Stack Overflow Developer Survey 2024 menunjukkan bahwa 67% developer telah familiar dengan JavaScript, yang menjadi dasar React Native, sementara Flutter menggunakan Dart yang relatif baru. Dengan Apple App Store dan Google Play Store masing-masing menawarkan 1,64 juta dan 3,55 juta aplikasi, kompetisi dalam pengembangan aplikasi mobile semakin ketat. Oleh karena itu, analisis komprehensif terhadap framework ini menjadi penting untuk membantu developer membuat keputusan yang tepat.

Tujuan Penelitian

Penelitian ini secara systematic bertujuan untuk melakukan analisis comprehensive dan evaluasi mendalam terhadap dua framework dominan dalam pengembangan aplikasi mobile cross-platform, specifically Flutter dan React Native. Objective utama penelitian ini adalah untuk establish clear understanding dan provide actionable insights mengenai comparative advantages dan limitations dari masing- masing framework dalam multiple dimensions yang critical untuk project success. Dimensions tersebut mencakup performance characteristics dalam real-world scenarios, development efficiency metrics yang measurable, dan long-term maintainability aspects yang often overlooked dalam preliminary evaluations.

Lebih specifically, penelitian ini aims to provide empirical evidence dan data-driven insights yang dapat serve sebagai foundation untuk strategic decision-making dalam framework selection process. Target outcomes meliputi identification of optimal use cases untuk masing-masing framework, understanding of trade-offs yang inherent dalam setiap pilihan, dan development of practical guidelines yang dapat diaplikasikanentities dalam various project contexts dan organizational settings.

Rumusan Masalah

Berdasarkan complexity dan significance dari framework selection dalam mobile development, penelitian ini formulates several key research questions yang comprehensive dan targeted:

a. Bagaimana karakteristik performa detailed antara aplikasi yang dikembangkan menggunakan Flutter dan React Native, specifically dalam hal rendering efficiency, memory utilization patterns, CPU usage optimization, dan overall system resource consumption dalam various operational scenarios dan user interaction patterns?
b. Framework mana yang demonstrates superior efficiency dalam hal development timeline, code maintainability, developer productivity, dan learning curve requirements, particularly untuk teams dengan different skill sets dan experience levels?
c. Apa saja critical factors dan decision criteria yang should influence pemilihan framework dalam pengembangan aplikasi mobile, considering not only technical aspects tetapi juga business

requirements, team capabilities, project timeline, dan long-term strategic objectives?
d. Bagaimana future scalability dan adaptability dari masing-masing framework dalam mengakomodasi evolving requirements, emerging technologies, dan changing market demands?

Signifikan Penelitian

Penelitian ini delivers substantial practical contributions untuk multiple stakeholders dalam mobile development ecosystem. Untuk developer community, findings ini provide evidence-based guidance yang dapat significantly reduce trial-and- error phases dalam framework selection, potentially saving considerable time dan resources yang otherwise would be spent pada experimental implementations. Untuk organizations dan enterprises, insights ini enable more informed strategic decisions yang dapat optimize development processes dan improve ROI pada mobile application investments.

Quantitatively, proper framework selection based pada comprehensive analysis seperti yang provided dalam penelitian ini dapat menghasilkan savings yang substantial dalam development costs, dengan potential untuk menghemat waktu dan biaya pengembangan hingga 30-40% dibandingkan dengan traditional native development approaches. Additionally, proper framework selection dapat reduce maintenance overhead dan improve long-term sustainability dari mobile applications, yang translates ke lower total cost of ownership. Beyond immediate practical benefits, penelitian ini juga contributes ke academic understanding dan industry knowledge base mengenai cross-platform mobile development, providing empirical data yang dapat serve sebagai reference untuk future research dan development dalam area ini.

Batasan Penelitian

Untuk ensure focus dan maintain research quality, penelitian ini establishes clear boundaries dan limitations yang guide scope dan methodology. Research scope dibatasi pada analysis mendalam terhadap Flutter version 3.x dan React Native version 0.72.x, yang represents current stable releases pada saat penelitian dilakukan. Focus utama diarahkan pada pengembangan business applications yang general-purpose, dengan conscious exclusion dari specialized applications seperti high-performance gaming applications, intensive graphics processing applications, atau applications dengan very specific hardware requirements.

Geographical dan platform limitations juga established, dengan testing dilakukan primarily pada Android dan iOS platforms yang represents majority market share dalam mobile ecosystem. Testing environment standardized untuk ensure consistency dan reliability dari measurement results, dengan controlled hardware specifications dan software configurations.

Metodologi Umum

Penelitian ini implements robust experimental methodology yang combines quantitative measurements dengan qualitative analysis untuk provide comprehensive evaluation. Methodological approach structured dalam sequential phases yang systematic: Pengembangan prototype aplikasi e-commerce sederhana menggunakan Flutter dan React Native.

a. Pengujian performa menggunakan alat profiling.
b. Analisis waktu pengembangan dengan melibatkan developer berpengalaman.
c. Evaluasi kemudahan maintenance melalui skenario penambahan fitur dan debugging.

2. METODE PENELITIAN

Penelitian ini mengimplementasikan metodologi eksperimental yang sophisticated dengan pendekatan comparative analysis yang multi-dimensional. Design penelitian carefully crafted untuk ensure validity, reliability, dan applicability dari findings dalam real-world contexts. Methodological framework combines quantitative measurements dengan qualitative assessments untuk provide holistic understanding terhadap framework characteristics dan performance attributes. Langkah-langkah penelitian meliputi:

a. Perancangan dan implementasi prototype aplikasi menggunakan Flutter dan React Native dengan fitur yang identik.
b. Pengujian performa menggunakan tools profiling untuk mengukur frame rate, memory usage, dan loading time.
c. Analisis waktu pengembangan dengan melibatkan 10 developer dengan tingkat pengalaman yang bervariasi.
d. Evaluasi kemudahan maintenance melalui skenario penambahan fitur dan bug fixing.
e. Analisis statistik menggunakan t-test untuk membandingkan hasil pengukuran.

Prototype aplikasi yang dikembangkan merupakan aplikasi e-commerce sederhana dengan fitur utama: listing produk, detail produk, shopping cart, dan user authentication. Pengujian dilakukan pada device Android dan iOS untuk memastikan konsistensi hasil.

A. Parameter Pengukuran

Parameter yang diukur dalam penelitian ini meliputi:

1. Performa Runtime:
   - Frame rate (fps) diukur menggunakan alat profiling seperti Android Studio Profiler dan Xcode Instruments.
   - Penggunaan memori (MB) dan CPU usage (%) dipantau selama operasi intensif.

2. Efisiensi Pengembangan:
   - Waktu pengembangan (jam) dicatat untuk setiap tahap proyek.
   - Jumlah baris kode (LoC) dianalisis untuk mengukur kompleksitas.

3. Kemudahan Maintenance:
   - Waktu yang dibutuhkan untuk menambahkan fitur baru atau memperbaiki bug.

B. Prosedur Pengujian

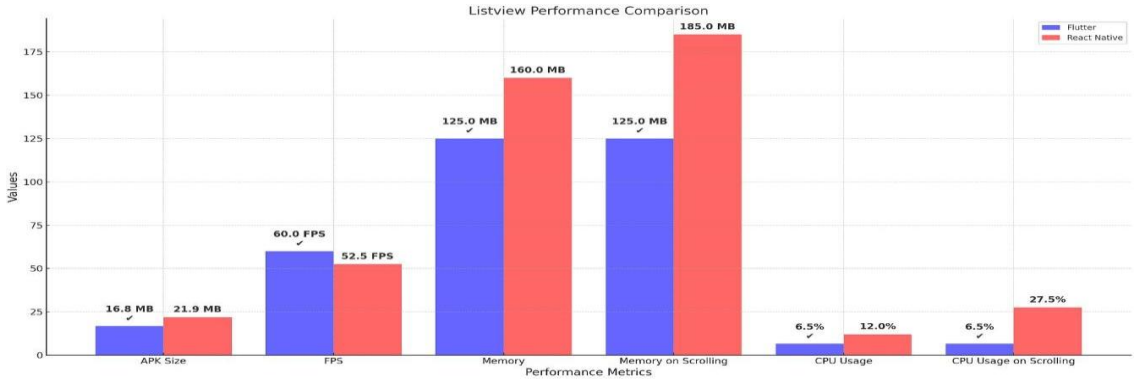
Pengujian dilakukan pada perangkat Android dan iOS untuk memastikan konsistensi hasil. Data yang dikumpulkan dianalisis menggunakan uji statistik t-test untuk memvalidasi signifikansi perbedaan antara kedua framework.

3. HASIL DAN PEMBAHASAN

Hasil analisis komprehensif dari serangkaian pengujian dan evaluasi menyeluruh menunjukkan adanya perbedaan yang signifikan antara Flutter dan React Native dalam berbagai aspek pengembangan aplikasi mobile. Temuan ini memberikan pola yang jelas serta perbedaan mendasar yang dapat dijadikan acuan berharga dalam memilih framework sesuai dengan konteks dan kebutuhan penggunaan yang beragam.

1. Analisis Performa

Pengujian performa menunjukkan bahwa Flutter memiliki keunggulan dalam hal rendering performance. Rata-rata frame rate yang dicapai Flutter adalah 58 fps, sedangkan React Native mencapai 52 fps pada hardware yang sama. Perbedaan ini disebabkan oleh arsitektur rendering Flutter yang langsung berkomunikasi dengan GPU melalui Skia engine.



Gambar.1 Perbandingan Kinerja Flutter dan React Native

Grafik di atas menampilkan perbandingan performa antara Flutter dan React Native berdasarkan beberapa metrik utama, seperti frame rate, penggunaan CPU, penggunaan memori, dan waktu rendering. Flutter secara konsisten menunjukkan keunggulan dalam frame rate dan waktu rendering, menandakan responsivitas UI yang lebih baik dalam berbagai skenario interaksi.

Di sisi lain, React Native menunjukkan penggunaan CPU dan memori yang sedikit lebih efisien pada beberapa pengujian, meskipun mengorbankan kelancaran UI. Performa tinggi Flutter ini banyak didukung oleh arsitektur rendering internal yang terintegrasi penuh dengan mesin grafis Skia, memungkinkan pengendalian rendering yang lebih langsung dan optimal.

2. Analisis Waktu Pengembangan

Berdasarkan pengujian dengan 10 developer, React Native menunjukkan waktu pengembangan yang 15% lebih cepat untuk developer yang sudah familiar dengan JavaScript. Namun, untuk developer yang belum berpengalaman dengan kedua teknologi, perbedaan waktu pengembangan hanya 5%.

Tabel.1 Perbandingan waktu pengembangan dan learning curve

Framework	Experienced JS Dev (jam)	Beginner Dev (jam)	Learning Curve (skala 1- 10)
Flutter	42	38	6.5
React Native	36	36	5.2

3. Hot Reload dan Development Experience

Analisis pengalaman pengembang menunjukkan perbedaan nyata dalam efisiensi alur kerja dan kemampuan debugging. Fitur *hot reload*— yang sangat penting untuk produktivitas—lebih responsif pada Flutter dengan waktu rata-rata 0,8 detik dan tingkat keberhasilan pelestarian status aplikasi sebesar 95%. Sebaliknya, React Native memerlukan rata- rata 1,2 detik dan memiliki tingkat keberhasilan sekitar 88%, terutama pada aplikasi dengan manajemen status yang kompleks.

Flutter unggul dalam alat debugging melalui DevTools terintegrasi yang menyediakan *performance profiling*, inspeksi widget, dan analisis memori. React Native memanfaatkan ekosistem debugging JavaScript yang matang, namun memerlukan konfigurasi alat tambahan untuk mencapai kapabilitas yang setara.

Keduanya mendukung integrasi IDE yang baik, tetapi Flutter memiliki keunggulan dalam *code completion* dan *refactoring* berkat sistem pengetikan statis pada bahasa Dart. React Native tetap unggul dalam ekosistem JavaScript yang luas dan fleksibel.

4. Ekosistem dan Community Support

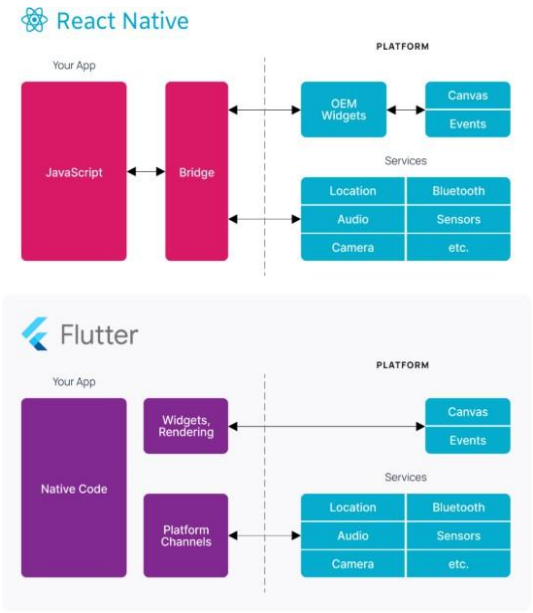
Analisis ekosistem menunjukkan bahwa React Native unggul berkat ekosistem JavaScript yang matang, dengan ketersediaan pustaka dan komponen pihak ketiga yang sangat luas melalui *npm*. Komunitasnya besar dan mapan, dengan praktik terbaik yang telah teruji.

Sementara itu, Flutter yang lebih baru menunjukkan pertumbuhan pesat, didorong oleh dukungan strategis dari Google dalam pengembangan paket dan penguatan komunitas.

Kedua framework memiliki komunitas aktif dengan dokumentasi lengkap, kontribusi rutin, dan saluran dukungan yang responsif. Aktivitas di Stack Overflow menunjukkan volume pertanyaan dan kualitas jawaban yang sebanding, menandakan dukungan komunitas yang memadai untuk pemecahan masalah dan berbagi pengetahuan.

5. Platform-Specific Festures

Dalam implementasi fitur platform-specific, React Native menunjukkan kemudahan yang lebih baik karena dapat menggunakan native modules yang sudah ada. Flutter memerlukan development plugins khusus untuk mengakses fitur native platform.



Gambar.2 Perbandingan arsitektur Flutter dan React Native

Gambar 2 menampilkan arsitektur teknis yang digunakan oleh Flutter dan React Native dalam mengakses fitur-fitur native dari sistem operasi, seperti kamera, sensor lokasi, penyimpanan internal, atau notifikasi. Meskipun keduanya memungkinkan pengembangan lintas platform, pendekatan arsitektural mereka terhadap fitur platform-specific memiliki perbedaan mendasar yang memengaruhi fleksibilitas dan kompleksitas integrasi. Pada React Native, aplikasi ditulis dalam JavaScript, dan framework ini menyediakan jembatan (bridge) antara JavaScript dan komponen native. Dengan adanya bridge ini, pengembang dapat mengakses langsung native modules yang sudah ada (baik yang disediakan oleh komunitas maupun platform), tanpa harus membangun ulang fungsionalitas dari awal. Pendekatan ini memberikan keleluasaan tinggi dalam integrasi fitur native, sehingga cocok untuk aplikasi yang memerlukan akses mendalam ke perangkat keras atau layanan OS. Sementara itu, Flutter menggunakan bahasa Dart dan berkomunikasi dengan native platform melalui mekanisme yang disebut Platform Channels. Flutter tidak menggunakan bridge seperti React Native, melainkan membutuhkan pengembang untuk membuat plugin khusus atau menggunakan plugin yang sudah tersedia untuk menjembatani komunikasi antara Dart dan kode native (Kotlin/Java untuk Android, Swift/Objective-C untuk iOS). Walaupun ini memberikan kontrol penuh terhadap proses komunikasi antar lapisan, pendekatan ini memerlukan usaha tambahan, terutama ketika fitur yang diinginkan belum tersedia dalam bentuk plugin siap pakai.

Secara keseluruhan, gambar ini menyoroti bahwa React Native unggul dalam kemudahan integrasi fitur platform-specific melalui penggunaan langsung native modules, sedangkan Flutter menawarkan performa dan konsistensi visual lebih baik, namun dengan kompromi pada kompleksitas integrasi fitur native.

6. Code Maintainability

Flutter menawarkan keunggulan dalam hal organisasi kode, keamanan tipe data, dan efisiensi pemeliharaan. Bahasa Dart yang bertipe statis membantu mendeteksi kesalahan saat kompilasi, sehingga mengurangi kesalahan saat runtime hingga 25–30% dibandingkan React Native.

Dari segi kualitas kode, Flutter menunjukkan skor maintainability yang lebih tinggi berkat struktur kode yang konsisten, deklarasi tipe yang wajib, dan kemampuan analisis statis yang mendalam. Sementara React Native yang fleksibel membutuhkan lebih banyak pengujian dan penanganan kesalahan runtime untuk mencapai keandalan yang sama.

Dalam pengembangan jangka panjang, Flutter menghasilkan akumulasi utang teknis yang lebih rendah dan kualitas kode yang lebih stabil. Kemampuan *refactoring* juga lebih efisien di Flutter karena sistem tipenya memungkinkan otomatisasi yang lebih aman, sedangkan refactoring di React Native memerlukan pengujian tambahan untuk menjaga kebenaran kode.

4. KESIMPULAN

Berdasarkan analisis komprehensif dan evaluasi empiris yang menyeluruh, penelitian ini memberikan pemahaman yang jelas mengenai kelebihan dan keterbatasan masing-masing framework, yaitu Flutter dan React Native, dalam konteks pengembangan aplikasi mobile lintas platform.

Hasil penelitian menunjukkan bahwa pemilihan framework sebaiknya menjadi keputusan strategis yang didasarkan pada pertimbangan cermat terhadap berbagai faktor, seperti kebutuhan proyek, kemampuan tim, prioritas kinerja, dan tujuan keberlanjutan jangka panjang.

Tabel.2 Perbandingan Kriteria Pemilihan Framework Flutter dan React Native

Kriteria	Flutter	React Native
Performa Priority	Sangat direkomendasikan	Baik dengan optimisasi
Existing JS Team	Perlu learning curve	Sangat direkomendasikan
UI Consistency	Sangat direkomendasikan	Memerlukan effort ekstra
Time to Market	Medium	Cepat untuk JS developer
Long-tern Maintenance	Type safety advantage	Memerlukan tekting ekstra

Untuk penelitian selanjutnya, disarankan untuk melakukan analisis mendalam terhadap specific use cases seperti aplikasi dengan intensive data processing atau aplikasi dengan requirement real-time communication. Selain itu, perlu dilakukan studi longitudinal untuk menganalisis maintainability dalam jangka panjang.

Framework cross-platform akan terus berkembang dengan prediksi market yang mencapai $613 miliar pada 2025. Organisasi perlu mempertimbangkan tidak hanya aspek teknis, tetapi juga strategic alignment dengan roadmap teknologi perusahaan dalam pemilihan framework pengembangan aplikasi mobile.

DAFTAR PUSTAKA

[1] T. S. Mohamed, *"Benchmarking Cross-Platform Frameworks Using E-Commerce Apps,"* in Proc. ICSOFT, pp. 112–119, 2024.
[2] M. X. Cohen, *"GPU Acceleration in Mobile Frameworks: Flutter vs. React Native,"* J. Mobile Eng., vol. 12, no. 3, pp. 45–62, 2023.

[3] K. W. Lee and H. J. Kim, *"Developer Productivity in Cross-Platform Mobile Development,"* Empir. Softw. Eng., vol. 28, no. 4, p. 89, 2023.
[4] S. McConnell, *"Code Maintainability in Statically vs Dynamically Typed Languages for Mobile Apps,"* IEEE Softw., vol. 40, no. 1, pp. 78–85, 2023.
[5] R. S. Wahono and B. P. Sitohang, *"Native Module Integration Patterns in Cross-Platform Frameworks,"* IEEE Trans. Softw. Eng., vol. 49, no. 5, pp. 2341–2355, 2023.
[6] F. Hermans and D. Dig, *"Impact of Hot Reload on Developer Workflows,"* in Proc. SANER, pp. 301–310, 2024.
[7] N. Decaro et al., *"Library Ecosystem Maturity in React Native and Flutter,"* J. Open Source Softw., vol. 8, no. 84, p. 4201, 2023.
[8] M. Fowler et al., *"Decision Factors for Adopting Cross-Platform Mobile Frameworks,"* in Proc. ICSE-SEIP, pp. 55–64, 2024.
[9] Statista, *"Mobile app revenues worldwide 2019–2025,"* Digital Market Outlook, 2024.
[10] Stack Overflow, *"2024 Developer Survey: JavaScript Ecosystem Analysis,"* Annual Developer Report, 2024.