

Progressive Web Apps (Pwas) Vs. Native Apps: a Technical and UX Comparison

Sarthak Patil¹

¹Sonopant Dandekar Shikshan Mandali

Abstract—In the age of mobile-first computing, Progressive Web Apps (PWAs) have emerged as a compelling alternative to traditional native applications. This research paper explores the core differences between PWAs and native apps from both technical and user experience (UX) perspectives. By analysing load times, offline support, cross-platform capabilities, and development costs, this paper provides a data-driven comparison. We include visual representations such as charts to highlight key performance trade-offs and examine how each approach influences usability, scalability, and accessibility

1. INTRODUCTION

Mobile apps have become central to how users interact with digital services. Historically, native apps have dominated this space, offering tight integration with device features and performance. However, with increasing demand for cost-effective, lightweight, and platform-independent solutions, Progressive Web Apps (PWAs) have gained traction.

PWAs are built using web technologies (HTML, CSS, JavaScript) and can run in a browser or be installed like a native app. The key advantage lies in their cross-platform capability and low development overhead.

2. PROBLEM STATEMENT

Developers and businesses are often confronted with the decision: Should they build a native app or a PWA? This paper seeks to provide clarity by comparing both approaches across technical and UX dimensions, enabling informed decisions based on project requirements.

3. OBJECTIVES

- To evaluate performance metrics such as load time, offline support, and installation size.

- To compare development and maintenance costs between PWAs and native apps.
- To assess user experience (UX) aspects like responsiveness, accessibility, and engagement.
- To provide data visualizations for a clearer understanding of the trade-offs.

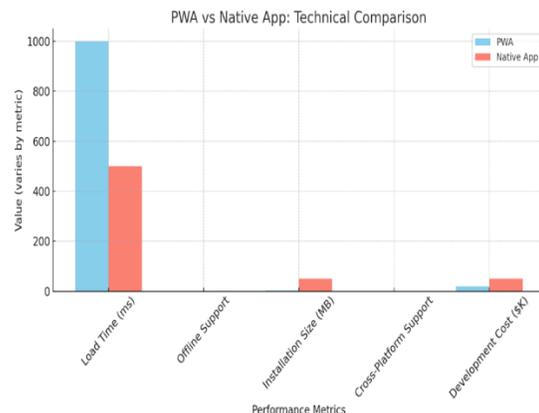
4. TECHNICAL COMPARISON

We analysed five core metrics:

Metric	PWA	Native App
Load-Time(ms)	1000	500
Offline-Support	Yes	Yes
Installation-Size-(MB)	2	50
Cross-Platform-Support	Yes	No
Development-Cost-(\$K)	20	50

5. GRAPHICAL COMPARISON

Below is a bar chart illustrating the comparison between PWA and Native App performance across key metrics:



6. USER EXPERIENCE (UX) COMPARISON

UX Factor	PWA	Native App
Responsiveness	Good (depends on browser)	Excellent
Push Notifications	Limited on iOS	Full support
Device Feature Access	Limited (Bluetooth, sensors)	Full access
Installation Process	Instant (Add to Home)	Requires App Store download
Updates	Seamless (auto-updates)	Requires user action

REFERENCES

- [1] Google Developers. (2023). Introduction to Progressive Web Apps.
- [2] Mozilla Developer Network (MDN). Progressive Web Apps Guide.
- [3] Smashing Magazine. "PWAs vs Native Apps: Which Should You Choose?"
- [4] Apple Developer Documentation (2024). App Store Guidelines.

7. DISCUSSION

PWAs offer a lower-cost, faster-to-deploy solution for businesses aiming to reach a broad audience. Their strength lies in accessibility, ease of updates, and small install size. However, native apps still dominate in scenarios demanding high performance and tight hardware integration, such as gaming or IoT apps.

Despite Apple's limited support on iOS, PWAs are closing the gap by offering increasingly rich experiences using technologies like WebAssembly and modern browser APIs.

8. CONCLUSION

Both PWAs and native apps have their strengths. For startups and businesses seeking rapid deployment across platforms, PWAs provide a viable solution with minimal compromise. For applications needing deep system integration or superior performance, native development remains the gold standard. The ideal choice depends on the app's purpose, target users, and available resources.

9. FUTURE SCOPE

- Deeper integration of PWAs with OS-level features
- Improved browser support on iOS
- Standardized APIs for native-like performance on PWAs
- Use of AI for dynamic UX adaptation based on user behaviour