# Progressive Web Apps (PWA) vs Native Apps: A Technical Comparison

## Overview

With the rise of mobile usage, developers often face a critical choice: **build a native app** or develop a **Progressive Web App (PWA)**. Both approaches offer distinct advantages depending on goals like reach, performance, and platform-specific features.

## What is a Native App?

A **native app** is built specifically for a platform (e.g., Android, iOS) using platform-specific tools:

- Android: Java/Kotlin with Android SDK
- iOS: Swift/Objective-C with Xcode

### Native App Features

- Full access to hardware APIs (Bluetooth, NFC, camera, GPS)
- Optimized UI performance
- Tightly integrated with OS features (e.g., Siri, system notifications)

## When to Use PWA

- You want **cross-platform reach** without writing multiple codebases.
- The app is **content-centric** (news, e-commerce, blog, etc.).
- You don’t need deep hardware or OS integration.
- Fast iteration and updates are key (e.g., startups, MVPs).

### Examples:

- Twitter Lite
- Pinterest PWA
- Starbucks PWA

## Hybrid Alternatives

- **React Native**, **Flutter**, and **Capacitor**: Offer native performance with single-codebase flexibility.
- Useful when you need more than a PWA but want to avoid writing platform-specific code.

---

## Conclusion

Both PWAs and native apps serve their purpose based on context:

- **PWAs excel in accessibility, development speed, and distribution.**
- **Native apps shine in performance, UX consistency, and hardware integration.**

A strategic mix may even make sense—using a PWA as a public-facing front and a native app for advanced use cases.