Wells Fargo Mobile Banking Platform

Role: Senior iOS / Cross-Platform Mobile Engineer
Tech Stack: Swift 5.x, SwiftUI, UIKit, Combine, Objective-C, React Native (TypeScript), Flutter (Dart), React 18, Angular 15, Keychain, OAuth2, MFA, PCI-DSS, Alamofire, Realm, GraphQL, REST, Fastlane, Jenkins

---

Overview

Led the modernization and expansion of Wells Fargo's enterprise-grade **mobile banking ecosystem**, powering secure real-time financial operations for **tens of millions of customers** across iOS, Android, and web.

The initiative involved a full-stack mobile transformation: migrating legacy Objective-C to Swift 5+, designing a scalable mobile architecture, extending features to **React Native cross-platform apps**, and ensuring strict compliance with **banking-grade security and PCI-DSS standards**.

The platform delivered highly secure transactions, instant account updates, biometric authentication, personalized financial insights, and performance-optimized mobile workflows across the entire Wells Fargo digital ecosystem.

Team Configuration

- **1 Product Manager** – Feature roadmap, security & compliance acceptance, KPI tracking.
- **3 iOS Engineers (including me)** – Swift migration, feature development, SwiftUI/Combine architecture.
- **2 Cross-Platform Engineers** – React Native & Flutter banking modules.
- **1 Backend Engineer** – API design, GraphQL schemas, SSL/TLS security hardening.
- **1 Security Engineer** – PCI-DSS, MFA, Keychain, OAuth2 compliance.
- **1 QA Engineer** – Automated testing, regression suites, device labs.
- **1 DevOps Engineer** – Fastlane, CI/CD pipelines, release automation.

Process: Agile (2-week sprints), code reviews via GitHub, nightly automated builds via Fastlane + Jenkins, weekly release trains, and cross-team architecture syncs.

Key Engineering Contributions

1. Mobile Architecture Modernization (Swift 5 + SwiftUI + Combine)

- Migrated large legacy Objective-C codebase to **Swift 5+**, improving safety, testability, and long-term maintainability.
- Rebuilt major flows (account overview, transfers, bill pay) using **SwiftUI**, **Combine**, and **MVVM** for reactive data binding.
- Implemented modular architecture enabling **feature isolation**, rapid iteration, and cross-team parallel development.
- Introduced dependency injection and protocol-driven patterns, reducing code duplication by **~35%**.

---

2. Cross-Platform Expansion (React Native + TypeScript)

- Extended core native banking features to **React Native**, enabling a unified codebase for iOS and Android.
- Built secure JS bridges for Keychain, analytics tracking, and encrypted API access.
- Delivered shared components for transactions, statements, and authentication, decreasing dual-platform effort by **~40%**.
- Supported web dashboards using **React 18** and admin tooling in **Angular 15**.

---

3. Security & Compliance Engineering (PCI-DSS / MFA / Biometrics)

- Implemented **Face ID / Touch ID** biometric authentication with fallback passcode flows.
- Developed secure Keychain-backed encrypted credential storage with rotation policies.
- Integrated **multi-factor authentication (MFA)** using OAuth2 and time-based OTP.
- Partnered with security engineers to enforce **PCI-DSS** compliance across data flows, session handling, and network requests.

---

4. Networking & Data Layer Engineering

- Built secure, TLS-pinned networking layers using **Alamofire** for REST and GraphQL API consumption.
- Designed robust offline support and caching via **Realm** for account data, transaction history, and profile details.
- Implemented live account updates using real-time API polling + incremental sync logic.
- Hardened network stack with SSL/TLS enforcement, cert pinning, and retry patterns for banking-grade reliability.

---

5. High-Performance Mobile Banking Features

- Optimized check deposit camera pipeline (real-time edge detection, reduced capture latency).
- Rewrote transaction history UI with diffable data sources, improving rendering performance by **~45%**.
- Used **Xcode Instruments** for:
  - memory profiling
  - leak detection
  - network usage optimization
  - GPU/frame rendering profiling
- Reduced overall app launch time by **~30%** through startup refactoring and lazy-loading strategies.

---

6. CI/CD & Release Automation (Fastlane + Jenkins)

- Designed automated build and test pipelines using **Fastlane**, **Jenkins**, and provisioning automation.
- Set up nightly builds, regression test triggers, linting workflows, and TestFlight deployment.
- Implemented automated screenshot generation and versioning for App Store compliance.
- Reduced release cycle time from **2 weeks → 3 days** through optimized CI/CD workflows.

Impact

- ↑ **40% improvement** in app responsiveness and load times through Swift migration and performance optimization.
- ↓ **50% reduction** in platform-specific Android/iOS code through React Native feature unification.
- ↑ **25% increase** in mobile user engagement due to smoother flows and personalized financial insights.
- ↓ **35% reduction** in crash rates after architecture modernization and memory leak auditing.
- Delivered a **bank-grade secure, scalable, and highly performant** mobile platform used daily by millions of consumers.

Key Responsibilities

- Modernizing enterprise iOS architecture using Swift 5, SwiftUI, Combine, and MVVM.
- Building secure banking features with biometrics, MFA, Keychain, and PCI-DSS compliance.
- Developing cross-platform modules using React Native + TypeScript.
- Architecting secure networking layers using Alamofire, REST, and GraphQL.
- Implementing CI/CD automation via Fastlane + Jenkins for scalable, repeatable releases.
- Collaborating with product, security, DevOps, and backend teams across a distributed enterprise environment.

Relevance to macOS Engineering (Renewed Vision)

This project demonstrates the exact skills Renewed Vision seeks:

- Expertise in **Objective-C → Swift migration**, parallel to their legacy modernization.
- Deep experience in **SwiftUI, Combine, modular MVVM**, ideal for macOS AppKit/SwiftUI hybrid apps.

- Strong background in **secure, real-time, high-performance applications**, directly relevant to Renewed Vision’s media-heavy, low-latency products.
- Proficiency integrating **REST/GraphQL**, working with complex architectures, and optimizing UI rendering.
- CI/CD, profiling, debugging, and cross-team collaboration at scale.