**Architectural Decision Record (ADR) for Retail Company Mobile App**

**Title:** Development of a Hybrid Mobile App for Retail Company

**Status:** Accepted

**Deciders:** CodeWavers { Simardeep , jashanjot singh, Jotsaroop Singh, Rudra kainth}

**Date:** 2024-02-09

**Context**

The retail company aims to enhance customer engagement through a mobile app that allows product browsing, purchasing, order history viewing, shipping status tracking, and loyalty program participation. To achieve global reach, the app will include push notifications, offline mode support, payment gateway integration, analytics on user behavior, image optimization, and internationalization. These features are crucial for providing a seamless in-store experience, improving customer service, and accommodating diverse languages and cultural preferences.

**Decision Drivers**

- The necessity for offline browsing and order history viewing.

- The need for real-time updates via push notifications.

- Secure and flexible payment gateway integration.

- Analytics to understand and improve user engagement.

- Efficient image storage and optimization.

- Support for multiple languages and cultures.

**Considered Options**

1. App Development Approach: Native vs. Hybrid vs. Web

2. UI Framework: Ionic vs. React Native

3. Backend Language: Node.js

4. Permissions: System-level permissions for device features

5. Data Storage: Local vs. Cloud (AWS S3)

6. Additional Frameworks/Technology Stacks: Firebase, Stripe, PayPal, i18next

**Decision Outcome**

- App Development Approach: Hybrid, using React Native, to leverage cross-platform compatibility and development speed, enabling a unified user experience across iOS and Android with a single codebase.

- UI Framework: React Native was chosen for its extensive component library and native-like performance, facilitating rapid development and a responsive UI.

- Backend Language: Node.js, selected for its efficiency in handling asynchronous processes, interfacing with databases, and integrating with push notification providers.

- Permissions: System-level permissions will be managed to ensure the app has access to necessary device features (e.g., notifications, storage) while respecting user privacy.

- Data Storage: AWS S3 for reliable, scalable, and secure management of images and other assets. Local storage will be used for caching and offline data access.

- Additional Frameworks/Technology Stacks:

- Firebase for push notifications and analytics, providing insights into user behavior and enabling effective communication with users.

- Stripe and PayPal for payment processing, offering secure and versatile payment options.

- i18next for internationalization, supporting global user accessibility by accommodating various languages and cultural norms.

**Positive Consequences**

- Unified development and maintenance process for both iOS and Android platforms.

- Enhanced user experience with native-like performance.

- Secure, scalable backend infrastructure.

- Efficient image management and optimization.

- Simplified and secure payment transactions.

- Valuable analytics for continuous improvement.

- Effective use of push notifications.

- Accessibility for a global audience through localization.

**Negative Consequences**

- Potential complexities and costs associated with managing hybrid app development.

- Reliance on third-party services for critical functionalities.

Links

[Link to the detailed ADR document hosted on GitHub]

This template structures the architectural decisions made for the retail company's mobile app development project, providing a comprehensive overview of the choices and their implications. It is designed to guide the project team and stakeholders through the rationale behind these decisions, ensuring alignment with business objectives and user needs.