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

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

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

**Date:** 2024-02-09

**Change Log**

Every significant alteration made to this Architectural Decision Record is documented in this section.

Version 1.0 - 2024-02-09 - First Creation: Detailed the choice to create a hybrid mobile application employing globalization and offline capability to improve the in-store experience. Choosing a strategy for app development, UI framework, backend language, picture storage, and payment.

**Referenced Use Case**

**Use Case:** Enhancing the In-Store Experience Through a Hybrid Mobile App

A buyer wants to make a fast purchase when they walk into a retail establishment. They use their smartphone to browse products, see specials, and find items within the store thanks to the retail company's app installed on it. Once the user has chosen a product, they may access product details, customer reviews, and related products by using the app to scan the barcode. The consumer puts the item in their shopping basket and checks out with the app, saving their selected payment method.

- Offline Functionality: The app's offline browsing features, which were decided upon in the ADR, enable customers to access product information and reviews even in parts of the shop with inadequate internet connectivity.

- Globalization: The purchasing experience is made simple and personalized by the app, which constantly adapts content to the user's preferred language and currency.

- Payment Gateway Integration: Customers can select their chosen payment option thanks to Stripe and PayPal's seamless integration, which guarantees a quick and safe checkout procedure.

- Push Notifications: After completing the transaction, the customer gets a push notice at the store's collection point with an anticipated pick-up time and an order confirmation.

**Context**

Through a smartphone app that facilitates product browsing, ordering, checking purchase histories, tracking shipment status, and participation in loyalty programs, the retail corporation hopes to increase customer involvement. 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.

**Proposed Design**

- 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.

**Considerations**

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**

- 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.

**References:**

<https://docs.edgexfoundry.org/2.3/design/adr/template/>