**Architectural Decision Record**

I have decided to develop a native mobile application for both iOS and Android platforms.

**Choosing Between Native, Web, or Hybrid App Development:**

After careful consideration, I have decided to focus on developing native applications for several compelling reasons. I firmly believe that native apps provide unmatched user experiences and performance across all platforms. in alignment with my commitment to broad accessibility, native apps are the best choice for efficiently reaching both iOS and Android users. Developing native applications allows to seamlessly align my development efforts with my objectives, ensuring a consistent and high-quality user experience irrespective of user platform.

**UI Framework Selection:**

I picked React Native for our app's interface because it works well on both iOS and Android, giving users a familiar feel. It's great for developing quickly and supports lots of features, so we can make our app do more without slowing it down. Overall, React Native is perfect for meeting our project's UI needs because it's fast and works smoothly on all devices.

**Backend Language Selection:**

I decided to choose JavaScript as it is a good choice for the backend because it works well with Android frameworks. It's known for protecting against fraud, which is important for apps handling payments. JavaScript also makes it easier to build apps that work on different devices, unlike languages like C, which is harder to use across various other platforms.

**Permissions:**

The app needs to use your device's storage and location. It uses your location to fill in addresses automatically when you're shopping, which can be handy. You can still type in an address if you prefer. The storage access is for keeping track of your orders and making sure features work when you're offline. For example, it saves your favorite products so you can still buy them even without internet and completes your orders once you're back online.

**Data Storage:**

To ensure the app stays connected to the server and works offline, we'll use local device storage alongside syncing technology. SQLite is a simple database that works on different devices and will manage things like order history and user data. To sync data with servers, we'll use Firebase Realtime Database or Cloud Firestore, both of which work smoothly with React Native.

**Additional frameworks:**

React Native.

JavaScript for backend language.

Tailwind CSS for styling.

SQLite for DBMS.

Firebase Realtime Database or Cloud Firestore.

Google Analytics for Mobile Apps or Firebase Analytics for analytics.

Top of Form

Bottom of Form