FlyingPlane ADR

Seth Dijkstra, Jesse Taylor

1. Hybrid

Using React Native, the FlyingPlane app uses a hybrid approach, granting the desired cross-platform functionality while keeping a native app feel. This choice will give a consistent user experience across both iOS and Android platforms, which is essential for user retention and satisfaction.

2. UI Toolkit

After evaluating the available UI toolkits, we have chosen React Native Elements for its comprehensive set of pre-built components, ease of customization, and strong community support. This toolkit will accelerate our UI development process and ensure a consistent design language throughout the app.

3. Navigation Strategy

We have opted for React Navigation to manage the navigation stack of our application. Its simple API, active maintenance, and community support make it a reliable choice for handling our app’s navigation requirements.

4. Hardware

Our game will employ the device's Speaker to deliver audio feedback and enhance the gameplay experience. Additionally, Touch Input will be the primary mode of user interaction within the game, facilitating intuitive control and engagement.

5. Data Storage

Considering the nature of our game, local encrypted storage is selected to securely manage user data and game state. This approach aligns with our privacy policy and ensures data integrity.

6. Additional Frameworks or Technology Stacks

Given the scope and simplicity of our project, our game is designed to run solely on the device without requiring external server interactions. All necessary game assets and data are bundled with the app, eliminating the need for additional frameworks or technology stacks for HTTP requests or real-time synchronization. This approach aligns with our goal to keep the architecture straightforward and the game easy to manage and play.