Github App Documentation

Getting Started

Initial setup, run the following commands:

1. yarn install
2. cd ios && yarn pod install

To run on iOS, run the following command:

1. yarn iOS

Source Structure

- **Components**: This directory contains reusable UI elements. It's subdivided into two subdirectories:
  - **base**: This subdirectory hosts fundamental, low-level components that are used across various parts of the application. Examples might include buttons, input fields, or custom text components.
  - **composed**: This subdirectory contains higher-level components that are assembled from one or more base components. These components encapsulate more specific functionality or complex UI layouts.
- **navigator**: This directory hosts the configuration for react-navigation in the application. It includes the setup for a single stack navigator, responsible for managing transitions between different screens in a stack-like manner.
- **screens**: This directory encompasses all the screen components of the application. Each screen corresponds to a distinct view or state of the application that a user can interact with.
- **services**: This directory hosts the core network communication functionalities:
  - **module**: This subdirectory contains individual files for different API endpoints. Each file encapsulates the API calls related to a specific module or feature of the application.
  - **api.ts**: This is the base API setup file which configures the basic attributes and functionalities for making network requests, such as the base URL, request headers, or error handling.

- **store**: This directory contains the setup for the Redux store. It further includes subdirectories for each slice of the Redux store, allowing for an organized and modular state management structure.
- **styles**: This directory hosts global styles used throughout the application. This can include theme colors, common layout values, typography settings, and more. By consolidating these values in a single location, it facilitates consistency in appearance across the application.

Third-party Libraries Rational

- **react-native-masked-view/masked-view**: This library is used for creating views with a masked layout. This is particularly useful for creating visually intriguing designs and interfaces within the React Native environment
- **react-navigation/native**: This library serves as the foundation for navigating between screens in a React Native app. It provides components and hooks that help with navigation and managing app state.
- **react-navigation/stack**: This package is used for stack navigation in a React Native application. It allows for creating a stack of screens where one can navigate forward and backward.
- **reduxjs/toolkit**: Redux Toolkit is the official, opinionated, batteries-included toolset for efficient Redux development. It is used for managing application state in a predictable manner.
- l**odash**: Lodash is a JavaScript utility library that provides helpful methods for manipulation and combination of arrays, objects, and other data types. This can help make code more readable and less verbose.
- r**eact-native-mmkv**: This library is a super fast key-value storage library that is used to store data persistently in a React Native application. Required for redux-persist setup.
- **react-native-safe-area-context**: This library provides a way to access device safe area insets and to apply them to your layout so your app can respect safe area margins, avoiding notches and sensor clusters on modern devices.
- **react-native-screens**: This library improves performance by managing the presentation of screens in a React Native app at the native level, which can make navigating between different screens much smoother.

- **react-redux**: This package enables the connection of the Redux store with React components. It provides hooks and higher-order components to access the state and dispatch actions.

- r**edux-persist:** This library is used to persist and rehydrate a redux store. This helps to persist the Redux store to the local storage, so the state remains even after the app is closed or refreshed.