Project Knowledge Transfer (KT) - Driver App

1. Project Overview
This project is a React Native application designed for drivers. It facilitates various operations such as onboarding, attendance marking, trip management, wallet transactions, and document uploads.

Tech Stack
- **Framework**: React Native (0.72+)
- **Language**: TypeScript
- **State Management**: Redux Toolkit & RTK Query
- **Navigation**: React Navigation (v6/v7)
- **Networking**: Axios / RTK Query
- **UI Components**: Custom components with StyleSheet
- **Native Modules**: `react-native-image-picker`, `react-native-permissions`, `react-native-geolocation-service`, etc.

---

2. Project Structure
The source code is located in the `src/` directory. Here is a high-level breakdown:

Directory	Description
`src/Screens`	Contains all the screen components (pages) of the application. Organized by feature (e.g., `Auth`, `Home`, `DriverOnboarding`, `Wallet`).
`src/Redux`	State management logic. Includes `slices` (reducers), `services` (RTK Query APIs), and `store` configuration.
`src/Navigator`	Navigation configuration. `index.tsx` defines the Stack and Tab navigators and authentication flow logic.
`src/Components`	Reusable UI components used across multiple screens (e.g., Buttons, Inputs, Modals).
`src/Utills`	Helper functions and utilities (e.g., `imageUpload.ts`, `locationPermission.ts`, date formatters).
`src/Hooks`	Custom React hooks for shared logic (e.g., `useImageUpload`, `useLocation`).
`src/assets`	Static assets like images, fonts, and localization files.
`src/Configuration`	App configuration constants, environment variables, and API endpoints.
`android/`	Native Android project files (Gradle, Manifest).

---

3. Key Workflows & Code Flow

3.1 Authentication & Navigation
- **Entry Point**: `src/App.tsx` initializes the Redux Provider and the main Navigator.
- **Navigation Logic**: `src/Navigator/index.tsx` handles the switching between Auth stacks (Login) and App stacks (Home) based on the user's token presence in Redux state.

3.2 State Management (Redux)
- **Store**: Configured in `src/Redux/store.ts`.
- **Slices**: Located in `src/Redux/slices/`. Used for synchronous state (e.g., `authSlice` for storing user details).
- **Services**: Located in `src/Redux/services/`. Uses RTK Query for API calls (e.g., `auth.ts` for login APIs).
- **Usage**:
```typescript
// accessing state
```

```
const user = useAppSelector(state => state.auth.user);
// dispatching actions
dispatch(setUserData(newUser));
// calling APIs
const [login, { isLoading }] = useLoginMutation();
```

3.3 Driver Onboarding & Image Upload
- **Screen**: `src/Screens/DriverOnboarding/index.tsx`
- **Architecture**: The onboarding form is split into modular components:
- `PersonalInfoSection`
- `DocumentsSection`
- `BankDetailsSection`
- **Image Upload Flow**:
1. User taps an upload button.
2. `useImageUpload` hook is triggered.
3. **Permission Check**: `src/Utills/imageUpload.ts` checks permissions.
- *Android 13+*: Skips permission check (uses System Photo Picker).
- *Android 12-*: Checks `READ_EXTERNAL_STORAGE`.
4. **Picker**: `react-native-image-picker` is launched.
5. **Upload**: Image is uploaded to AWS S3 via `uploadImageToServer` utility.

3.4 Location Services
- **Utility**: `src/Utills/locationPermission.ts`
- **Permission Flow**:
- The app strictly uses **Foreground Location** (`ACCESS_FINE_LOCATION`).
- **Background Location**: `ACCESS_BACKGROUND_LOCATION` is commented out in `AndroidManifest.xml` to comply with Google Play policies unless strictly required and justified.

---

4. Google Play Store Compliance (Critical)
Recent updates have been made to ensure the app complies with Google Play policies.

4.1 Permissions (`AndroidManifest.xml`)
- **READ_MEDIA_IMAGES**: Removed. We now use the System Photo Picker for Android 13+, which does not require this permission.
- **ACCESS_BACKGROUND_LOCATION**: Commented out to avoid rejection for "Invalid use of location in background".
- **CAMERA**: Retained as it is necessary for capturing documents.

4.2 Handling "Permission Denied"
- **Logic**: Use `src/Utills/imageUpload.ts`.
- **Fix**: The code now correctly identifies Android 13+ and does NOT request `READ_EXTERNAL_STORAGE` or `READ_MEDIA_IMAGES`, preventing falsy "Permission Denied" errors and crashes.

---

5. Development Guidelines

Adding a New Screen
1. Create the screen component in `src/Screens/NewFeature/index.tsx`.

2. Add the screen name to `src/Constants/screens.ts` (or equivalent).
3. Register the screen in `src/Navigator/index.tsx` within the appropriate Stack.

API Integration
1. Define the API endpoint in `src/Redux/services/yourService.ts` using RTK Query `builder.mutation` or `builder.query`.
2. Export the auto-generated hook (e.g., `useSubmitDataMutation`).
3. Use the hook in your component.

Debugging
- **Redux**: Use Flipper or React Native Debugger to inspect state changes.
- **Network**: Use `react-native-network-logger` (if installed) or Flipper Network plugin.
- **Logs**: Console logs are generally stripped in production; use a logging service for critical errors.

---

6. Setup & Run
1. **Install Dependencies**: `npm install` or `yarn install`
2. **iOS Setup**: `cd ios && pod install`
3. **Run Android**: `npm run android`
4. **Run iOS**: `npm run ios`
