Projeto FatecHorário App – Um Documentário de Desenvolvimento

Este documento guia você através da jornada de criação do FatecHorário App, explorando cada decisão arquitetural, técnica e de UX adotada até o momento, bem como os planos e próximos passos.

1. Visão Geral Inicial

- Carregar dados de forma simples (JSON local inicialmente)
- Ser responsivo e performático
- Permitir navegação fluida entre telas
- Seguir padrões de código fáceis de manter e escalar

Decidimos usar:

- React Native (Expo) para rapidez de prototipação
- TypeScript para segurança de tipo
- Tailwind CSS (via NativeWind) para estilos ágeis
- React Navigation para fluxo de telas
- JSON local como fonte de dados inicial

2. Passo a Passo de Implementação

2.1. Definição dos Modelos de Dados

Em src/types/index.ts, definimos as interfaces:

```
interface GradeData { shifts: Shift[] }
interface Shift { name: string; courses: Course[] }
interface Course { name: string; semesters: Semester[] }
interface Semester { number: number; days: Day[] }
interface Day { day: string; classes: ClassItem[] }
interface ClassItem { time: string; subject: string; professor: string; location: string }
```

2.2. Service Layer – gradeService.ts

Centralizamos toda a lógica de acesso aos dados:

- getShifts() → Shift[]
- getCourses(shiftName) → Course[]
- getSemesters(shiftName, courseName) → Semester[]
- getDays(shiftName, courseName, semester) → Day[]
- getClasses(shiftName, courseName, semester, day) → ClassItem[]

2.3. Presentation Layer – Telas e Componentes Reutilizáveis

Criamos componentes genéricos:

- HeaderAnimated: cabeçalho com animação de escala e fade-in
- OptionList: lista de botões dentro de um BlurView

E telas sob medida:

- HomeScreen: lista de turnos (getShifts)

- CourseListScreen: lista de cursos de um turno (getCourses)
- ScheduleScreen: lista de semestres de um curso (getSemesters)
- DayListScreen: lista de dias de um semestre (getDays)

2.4. Routing Layer – Navegação Tipada

Em AppNavigator.tsx, definimos:

```
export type RootStackParamList = {
Home: undefined
CourseList: { shiftName: string }
Schedule: { shiftName: string; courseName: string }
DayList: { shiftName: string; courseName: string; semesterNumber: number }
ClassList: { shiftName: string; courseName: string; semesterNumber: number; dayName: string }
};
```

3. Princípios e Padrões Aplicados

- Separation of Concerns: UI isolada de lógica de dados
- Single Responsibility Principle: cada função/arquivo faz apenas uma coisa
- Law of Demeter: telas não percorrem o JSON diretamente, usam getters do service
- DRY: componentes HeaderAnimated e OptionList reaproveitados
- Typed Navigation: segurança total na passagem de params

4. Próximos Passos e Planos

- ClassListScreen: exibir lista de aulas de um dia (getClasses)
- Dark Mode: tema claro/escuro com Tailwind e Context API
- Notificações Locais: lembretes de início de aula (expo-notifications)
- Exportar para Calendário: integração com react-native-calendar-events
- Integração com API: trocar JSON estático por endpoint REST da Fatec
- Testes: unitários para service e componentes, E2E com Detox
- Animações Avançadas: transições compartilhadas, Lottie para confete
- CI/CD: GitHub Actions para lint, testes e build automáticos