

# Projeto Café na Taverna

**Equipe de Desenvolvimento:** Matheus Castilhos, Mario da Fonseca, Ryan Kologeski.

## 1. Visão Geral

Esta documentação detalha a arquitetura, estrutura de arquivos e o fluxo de navegação do front-end do projeto **Café na Taverna**. O sistema é dividido entre uma área pública (consumo de conteúdo) e uma área administrativa (gestão de episódios).

## 2. Tecnologias Principais

- **React:** Biblioteca base para construção da interface.
- **TypeScript:** Tipagem estática para maior segurança e manutenção.
- **Vite:** Ferramenta de build e servidor de desenvolvimento.
- **React Router:** Gerenciamento de rotas e navegação SPA.

## 3. Estrutura de Diretórios

Plaintext

```
cafenatavernaweb/
├── src/
│   ├── main.tsx      # Ponto de entrada (Renderização e Router)
│   ├── App.tsx       # Definição central de rotas
│   ├── components/   # Componentes reutilizáveis
│   │   └── Layout.tsx # Estrutura visual global (Público)
│   ├── pages/        # Telas da interface pública
│   │   ├── Home.tsx
│   │   ├── Episodes.tsx
│   │   ├── EpisodePage.tsx
│   │   └── Player.tsx
│   ├── admin/        # Módulos restritos à administração
│   │   ├── AdminLayout.tsx
│   │   ├── AdminDashboard.tsx
│   │   └── AdminEpisodes.tsx
│   └── vite.config.ts # Configurações do ambiente
```

## 4. Arquitetura de Rotas e Páginas

## 4.1 Área Pública

As páginas públicas utilizam o `Layout.tsx`, que centraliza o Header e o Footer.

- **Home:** Introdução ao projeto e destaque de conteúdos.
- **Episodes:** Listagem dinâmica consumindo dados da API.
- **EpisodePage:** Visualização detalhada de um episódio específico.
- **Player:** Interface dedicada para reprodução de áudio e controle de progresso.
- **Sobre:** Conteúdo institucional.

## 4.2 Área Administrativa (`/admin`)

Protegida por layout próprio (`AdminLayout.tsx`), focado em produtividade e gestão.

- **Dashboard:** Visão geral de métricas (ex: total de episódios).
  - **Episodes Management:** Interface CRUD (Create, Read, Update, Delete) para gestão do catálogo de podcasts.
- 

# 5. Fluxos de Navegação

### Fluxo do Usuário

1. Acessa a **Home**.
2. Navega para a lista de **Episódios**.
3. Seleciona um episódio para ver detalhes (**EpisodePage**).
4. Inicia a reprodução no **Player**.

### Fluxo Administrativo

1. Autenticação (Login).
  2. Acesso ao **Dashboard**.
  3. Navegação para **Gestão de Episódios**.
  4. Execução de Criação, Edição ou Exclusão de registros.
- 

# 6. Integração com API

O front-end atua como consumidor de serviços REST. As principais responsabilidades de integração são:

- **Busca (GET):** Listagem e detalhes de episódios.
- **Persistência (POST/PUT):** Envio de dados de formulário e upload de arquivos de áudio/imagem.
- **Exclusão (DELETE):** Remoção de registros via área administrativa.

---

## 7. Manutenção

Para garantir a escalabilidade do projeto:

- Novos componentes devem ser criados na pasta `/components` se forem reutilizáveis.
- Novas páginas devem ser registradas centralmente no arquivo `App.tsx`.
- A tipagem via TypeScript deve ser respeitada em todas as novas implementações para evitar regressões.