

## 2. Documento: 02

Este documento descreve a arquitetura técnica, a organização do código e o fluxo de dados do Site Institucional, desenvolvido em **React JS com Vite**.

### 2.1. Estrutura de Pastas e Componentização

O projeto segue uma estrutura de pastas lógica e modular, facilitando a manutenção e o desenvolvimento:

- /src: Código principal da aplicação.
  - /src/components: Componentes **reutilizáveis** (Ex: Header.jsx, Footer.jsx, Motivos.jsx).
  - /src/pages: Componentes de **rotas** (Ex: App.jsx como Home, Planos.jsx).
  - /src/assets: Arquivos estáticos (imagens, SVGs).
  - /src/services: Módulos de integração com APIs (Ex: api.js, geminiServices.js).
  - /src/context: Para gerenciamento de estado global (Ex: plantaAnaliseContext.js).
  - /src/hooks: Para lógica customizada (Ex: useTheme.js).
  - /src/styles: Arquivos Sass para customização (variables.scss, main.scss).

### 2.2. Componentes Críticos Reutilizáveis

Componente	Responsabilidade Principal	Tecnologias Chave
Header.jsx	Navegação principal, <i>routing</i> (React Router) e <i>switch</i> de tema.	React Router, Context API (para tema).
Footer.jsx	Links de rodapé e informações de contato.	Mantém a consistência visual do tema.
Motivos.jsx	Exibição de <i>cards</i> de benefícios (reutilizável).	Componentização modular, Ícones Bootstrap.
Depoimentos.jsx	Renderiza depoimentos de usuários.	Estrutura de <i>grid</i> responsivo.

### 2.3. Gerenciamento de Estado

O gerenciamento de estado do Site Institucional é mantido simples, sem a necessidade de bibliotecas complexas (Ex: Redux/Zustand), devido à natureza do site ser predominantemente estática e informativa.

- **Estado Global:** Utiliza o **React Context API** (`useTheme.js`) para gerenciar a alternância entre o tema **claro/escuro** (Dark Mode).
- **Persistência:** O estado do tema é armazenado no **localStorage** para que o usuário mantenha sua preferência de visualização em futuras visitas.
- **Autenticação:** A gestão simples de *tokens* também é realizada através do **localStorage**.

### 2.4. Fluxo de Dados e Integração com a API

O site utiliza a biblioteca **Axios** para se conectar com a API do Sistema de Gestão:

- **Base URL:** O *frontend* aponta para a URL base <https://enercheck.onrender.com>.
- **Serviços Chave:**
  - **getUserByToken():** Funções no `api.js` buscam dados de usuários autenticados (se estiverem logados).
  - **postProjetoAnalisar():** Envio de projetos para análise de IA (Embora este seja um fluxo mais comum no Mobile/Gestão, o *service* está presente).
- **Segurança:** A integração utiliza **Interceptors do Axios** para gerenciar a inclusão automática do *token Bearer* no cabeçalho das requisições e para lidar com o *refresh* de *tokens*, garantindo a segurança nas chamadas para buscar dados (ex: estatísticas agregadas de análises realizadas).