Relação de tecnologias

Ambiente e ferramentas

- Runtime e gerenciador: Node.js 20+ e pnpm, com scripts padrão dev, build, start e lint expostos no package.json e destacados no README.
- IDE sugerida: Visual Studio Code com extensões de TypeScript e Tailwind, alinhada ao uso de tipagem estrita, alias @/* e utilitários Tailwind configurados no projeto.

Framework, arquitetura e layout

- Next.js 14 (App Router) + React 18 com componentes de servidor e cliente; o layout global injeta cabeçalho fixo e o widget de IA em todas as páginas.
- Middleware de sessão garante sincronização dos cookies Supabase em qualquer rota da aplicação.
- Supabase: clients de servidor/navegador encapsulados, com verificação de variáveis e compartilhamento de cookies para SSR/CSR.

Persistência e APIs internas

- Camada de dados para projetos e gamificação que consulta Supabase, com fallback para mocks quando o backend não está configurado; inclui CRUD de projetos e RPC de gamificação.
- Rotas Next API:
    - /api/projects e /api/projects/[id] expõem leitura/criação/atualização/remoção com tratamento de erros e metadados de origem.
    - /api/export/excel gera planilha via xlsx.
    - /api/requests mantém protótipo em memória para registro de demandas.
    - /api/gamification consulta progresso e chama a RPC complete_stage_and_update_progress.
    - /api/ai/chat integra com OpenRouter, incluindo prompt contextualizado com dados mocks.

UI, formulários e estado

- Páginas principais: home pública, dashboard com métricas/insights, catálogo filtrável de projetos, detalhe de projeto com participantes, central de gamificação com progresso e trilhas, formulário de solicitações e página de login.
- Estado e formulários: hooks React (useState, useEffect, useMemo) para filtros, submissões e feedback de erro/sucesso (ex.: solicitação de solução).

- Chat flotante com controle de abertura, fila de mensagens e desativação automática se a chave OpenRouter estiver ausente.

Visualização e exportação de dados

- Recharts para gráficos de barras e pizza em dashboards.
- XLSX para exportação do portfólio (download client-side + geração server-side).

Estilização e design system

- Tailwind CSS com extensões customizadas (cores Eurofarma, gradientes, utilitários globais como page-shell, glass-panel, btn-primary).
- Ícones via lucide-react e ativos estáticos em public/icons para badges e trilhas.

Integrações de terceiros

- Supabase (auth, tabelas projects, innovation_trails, badges, user_gamification_progress, RPC de estágios).
- OpenRouter para chat IA, condicionado à variável OPENROUTER_API_KEY no backend.
- Fallback de dados mockados garante experiência offline/local para portfólio e dashboards.

Qualidade e build

- ESLint herda regras next/core-web-vitals/typescript; TypeScript estrito evita emissões e usa moduleResolution: bundler.
- Estilo e lint rodáveis em CI/CD via pnpm lint; README sugere deploy em Vercel (configuração de pipeline deve alinhar lint/build antes da publicação).