3. ESCOLHA E JUSTIFICATIVA DAS TECNOLOGIAS UTILIZADAS

3.1 Frontend: React.js com Vite

A construção da interface do sistema foi realizada utilizando o framework React.js, com o auxílio da ferramenta de build Vite. A escolha do React.js justifica-se pela sua capacidade de criação de interfaces de usuário de forma modularizada e reutilizável, promovendo a escalabilidade e facilitando a manutenção do código. A utilização do Virtual DOM proporciona alta performance nas renderizações e atualizações dinâmicas dos componentes. O Vite foi selecionado como ferramenta de empacotamento (bundler) por oferecer alta velocidade na inicialização do projeto e atualizações em tempo real (hot reload), otimizando o tempo de desenvolvimento.

3.2 Backend: Node.js com Express

Para o desenvolvimento da API RESTful do sistema, adotou-se o ambiente de execução Node.js, juntamente com o framework Express. A escolha fundamenta-se na alta capacidade do Node.js de lidar com operações assíncronas, ideal para aplicações que necessitam de comunicação constante entre frontend e backend. O Express, por sua vez, proporciona uma estrutura simplificada e flexível para a criação de rotas, middlewares e tratamento de requisições HTTP, agilizando o desenvolvimento e organização da API.

3.3 Banco de Dados: PostgreSQL

O sistema de gerenciamento de banco de dados escolhido foi o PostgreSQL. A decisão deve-se à robustez, confiabilidade e suporte a operações complexas oferecidas por esse banco de dados relacional. O PostgreSQL ainda apresenta suporte a extensões geográficas (como PostGIS), que futuramente poderão ser incorporadas ao sistema para aprimorar o monitoramento de coordenadas geográficas das árvores cadastradas.

3.4 Ferramentas de Apoio

Ferramenta	Justificativa
DBeaver	Utilizado como cliente gráfico para facilitar a visualização, a criação e o gerenciamento de estruturas no banco de dados PostgreSQL.
React Icons	Aplicado na composição visual do sistema, com o objetivo de tornar os formulários mais intuitivos e melhorar

	a experiência do usuário (UX).
localStorage (Frontend)	Implementado para armazenar informações do usuário logado, permitindo o preenchimento automático de campos como nome do registrante e ID do usuário.
Context API (React)	Utilizada para o gerenciamento global do estado de autenticação e informações do usuário, evitando a propagação excessiva de propriedades entre componentes.

3.5 Estilização e Interface do Usuário

Optou-se pela utilização de CSS customizado no frontend para garantir uma identidade visual leve, moderna e coerente. O layout foi desenvolvido com foco em:
- Acessibilidade;
- Responsividade para diferentes tamanhos de tela;
- Facilidade de navegação;
- Clareza nas informações apresentadas.

3.6 Considerações Finais

As tecnologias selecionadas visaram atender aos seguintes objetivos principais:
- Desempenho: Garantir uma aplicação ágil e eficiente, tanto em nível de cliente quanto de servidor.
- Escalabilidade: Permitir a evolução e expansão do sistema, com facilidade na adição de novos módulos e funcionalidades.
- Manutenção: Promover a organização do projeto para facilitar correções e melhorias futuras, além de tornar o sistema acessível para novos desenvolvedores.