PREPARAÇÃO DO AMBIENTE DE DESENVOLVIMENTO

2.1 Tecnologias de frontend

ReactJS: biblioteca JavaScript de código aberto mantida pelo Facebook e por uma comunidade ativa, com mais de 200 mil estrelas no GitHub. Permite a construção de interfaces de usuário através de um modelo declarativo baseado em componentes, facilitando:

- Reutilização de código e padronização de componentes.
- Gerenciamento de estado com React Hooks (useState, useEffect, useReducer), além de possibilidade de integração com bibliotecas como Zustand e Redux.
- Integração nativa com TypeScript, permitindo maior segurança e detecção de erros em tempo de desenvolvimento.
- Ecossistema maduro, com plugins e extensões para testes (Jest, React Testing Library) e estilização (Styled Components, Emotion).

Vite: ferramenta moderna de build e desenvolvimento para projetos frontend, projetada para velocidade e simplicidade:

- Build extremamente rápido com bundler baseado em ESBuild e suporte a módulos ES nativos.
- Hot Module Replacement (HMR) quase instantâneo, proporcionando feedback em tempo real.
- Suporte nativo a TypeScript, JSX, CSS Modules, PostCSS e Sass.
- Arquitetura enxuta, ideal para projetos que desejam controle total sobre o roteamento e a estrutura.
- Integração facilitada com bibliotecas de estilização como Tailwind CSS ou Styled Components.

Justificativa:

A adoção de ReactJS com Vite garante um fluxo de trabalho eficiente e leve,

desde a prototipação até a entrega em produção. ReactJS proporciona a base de componentes e gerenciamento de estado, enquanto o Vite acelera o desenvolvimento com build ultrarrápido e configuração simplificada. Essa combinação resulta em maior produtividade do time e melhor experiência do usuário final.

2.2 Tecnologias de backend

Node.js (JavaScript/TypeScript): plataforma de runtime construída sobre o motor V8 do Chrome, com modelo de I/O não bloqueante e baseado em eventos, ideal para alta concorrência. Principais características:

- Suporte a JavaScript e TypeScript, promovendo tipagem estática e maior robustez no desenvolvimento.
- Gerenciamento de pacotes via NPM/Yarn, com ecossistema rico de bibliotecas.
- Performance otimizada para operações de rede e I/O, graças ao loop de eventos e threads worker (Worker Threads).
- Ferramentas de depuração e monitoramento, como PM2 e Node Clinic.
- Integração simples com ferramentas de testes (Jest, Mocha) e linting (ESLint).

Express: framework minimalista para Node.js, ideal para construção rápida de APIs RESTful e serviços web:

- Middleware flexível e roteamento customizável.
- Integração fácil com MongoDB usando drivers nativos ou ORMs como Mongoose.
- Amplo suporte da comunidade e documentação madura.
- Ideal para arquiteturas leves e microserviços.

Justificativa:
A escolha de Node.js com Express permite a criação de serviços escaláveis com

baixa latência, promovendo integração nativa com o frontend em JavaScript e facilitando a construção de APIs performáticas e manuteníveis.

2.3 Banco de dados

MongoDB: banco NoSQL orientado a documentos, para esquemas dinâmicos e armazenagem de dados sem estrutura fixa:

- Modelo de dados flexível baseado em documentos BSON.
- Alta performance para leitura e escrita.
- Escalabilidade horizontal por meio de sharding.
- Ferramentas gráficas como MongoDB Compass para visualização e manipulação de dados.
- Compatibilidade com ORMs como Mongoose para modelagem de dados com validação e middlewares.

Justificativa:
A utilização do MongoDB permite maior flexibilidade na modelagem dos dados, adaptando-se facilmente às mudanças no escopo do projeto. É uma solução adequada para aplicações com estrutura de dados variável e que demandam alta performance em grandes volumes de dados.

2.4 IDEs e ferramentas de suporte

- Visual Studio Code: editor leve com marketplace rico de extensões (ESLint, Prettier, GitLens, Tailwind CSS IntelliSense, Docker). Suporte a debugging integrado para Node.js e attach remoto. Integração nativa com Git e ambientes Docker. Live Share para colaboração em tempo real.
- MongoDB Compass: visualização de dados e consultas no banco MongoDB.
- Postman / Insomnia: criação e organização de coleções de chamadas de API. Testes automatizados com scripts pré- e pós-requisição. Suporte a variáveis de ambiente e geração de documentação compartilhável.

- Docker & Docker Compose: conteinerização de serviços (backend, banco de dados) para isolar dependências. Definição de ambientes via YAML. Facilita onboarding e padroniza builds e deploys.

2.5 Conclusão

Este planejamento de preparação do ambiente de desenvolvimento estabelece uma base sólida e escalável para o projeto, articulando escolhas coerentes em todas as camadas de tecnologia. No frontend, a união de ReactJS e Vite oferece agilidade na prototipação, performance otimizada e flexibilidade estrutural. No backend, a combinação de Node.js e Express equilibra alta concorrência e simplicidade na construção de APIs. A escolha do MongoDB como banco de dados orientado a documentos atende à necessidade de estruturas de dados flexíveis e escaláveis. Por fim, o conjunto de IDEs e ferramentas — VS Code, MongoDB Compass, Postman/Insomnia e Docker — promove produtividade, padronização e robustez no fluxo de trabalho, reduzindo gargalos e acelerando o desenvolvimento contínuo e colaborativo.