

# Documentação Geral do Projeto StoreTchê

## Introdução

Este projeto consiste em uma aplicação Frontend desenvolvida em React que simula um sistema básico de e-commerce. A aplicação permite visualizar produtos cadastrados, acessar os detalhes de cada item, adicionar produtos ao carrinho de compras, gerenciar quantidades respeitando o estoque disponível e cadastrar novos produtos.

Para simulação do backend e persistência de dados, foi utilizado o JSON Server, permitindo o consumo de uma API REST local de forma simples. O projeto tem como objetivo aplicar conceitos fundamentais de React, como componentização, estado global, consumo de APIs, rotas e validações básicas.

- **Uso do Context API:**

A Context API foi utilizada para gerenciar o estado global do carrinho de compras, evitando o repasse excessivo de propriedades entre componentes. O contexto do carrinho foi criado utilizando “createContext” e envolve toda a aplicação por meio de um Provider.

Dentro desse contexto são armazenados os produtos adicionados ao carrinho, a quantidade de cada item e o valor total da compra. Além dos dados, o contexto centraliza as principais funções de manipulação do carrinho, como adicionar produtos, aumentar e diminuir a quantidade e remover itens.

Esse contexto é consumido em diferentes partes da aplicação, como na página de listagem de produtos, página de detalhes do produto e página do carrinho, garantindo que qualquer alteração no carrinho reflita imediatamente em toda a interface.

- **Controle de Estoque e Persistência do Carrinho**

O controle de estoque é realizado no Frontend, verificando a quantidade disponível de cada produto antes de permitir sua adição ao carrinho. Quando o limite do estoque é atingido, o botão de adicionar é automaticamente desabilitado, evitando inconsistências.

O carrinho de compras é persistido utilizando o “localStorage”. Dessa forma, mesmo ao atualizar a página, os produtos adicionados permanecem armazenados, proporcionando uma melhor experiência ao usuário.

As quantidades dos produtos no carrinho influenciam diretamente o cálculo do estoque disponível exibido nas páginas de detalhes e no carrinho, bem como nos valores de subtotal e total, que são recalculados dinamicamente conforme o estado do carrinho é alterado.

- **Consumo da API com JSON Server**

O Json Server foi utilizado para simular uma API REST local, com os dados sendo armazenados em um arquivo "db.json". Esse arquivo contém a lista de produtos cadastrados no sistema.

A aplicação realiza requisições do tipo GET para listar os produtos e buscar os dados de um item específico, além de requisições POST para cadastrar novos produtos. O consumo da API inclui tratamento de estados de carregamento e mensagens de erro, garantindo que o usuário seja informado caso ocorra alguma falha durante a comunicação com o servidor.

- **Estrutura Geral do Projeto**

O projeto foi organizado de forma modular, separando responsabilidades em diferentes pastas. Os componentes reutilizáveis são mantidos separados das páginas principais da aplicação. A lógica de estado global está centralizada na pasta de contextos, enquanto os serviços de consumo da API estão abstraídos em arquivos específicos.

As páginas criadas incluem a página inicial com listagem de produtos, a página de detalhes do produto, onde é possível visualizar informações completas e adicionar itens ao carrinho, a página do carrinho de compras, que permite gerenciar quantidades e visualizar o total da compra, e a página de cadastro de produtos.

A navegação entre essas páginas é realizada por meio de rotas, permitindo que o usuário transite facilmente entre as funcionalidades da aplicação.

- **Funcionalidades Implementadas**

Entre as funcionalidades efetivamente implementadas, destacam-se a listagem de produtos consumidos da API, a visualização de detalhes de cada produto, o cadastro de novos produtos e o gerenciamento completo do carrinho de compras.

O formulário de cadastro valida campos obrigatórios e realiza a conversão correta dos valores numéricos. Caso o usuário não informe uma imagem para o produto, o sistema atribui automaticamente uma imagem padrão armazenada na pasta de assets, garantindo consistência visual.

O carrinho permite adicionar produtos, aumentar ou diminuir quantidades com validação de estoque, remover itens e visualizar o subtotal de cada produto e o valor total da compra, que são atualizados dinamicamente conforme o estado do carrinho é alterado.

- **Execução do Projeto**

Para executar o projeto, é necessário instalar as dependências utilizando o gerenciador de pacotes, iniciar o Json Server a partir do arquivo “db.json” e, em seguida, rodar a aplicação React em ambiente de desenvolvimento. Com isso, a aplicação estará pronta para uso, consumindo a API local e permitindo o teste de todas as funcionalidades implementadas.