

Micro Api CRUD Produtos

Caroline Vitoria Paneco
28/11/2024

1	VISÃO GERAL	3
2	PROTÓTIPOS DE TELA.....	4
2.1	TELA DE CADASTRO DE PRODUTO.....	4
2.2	TELA DE LISTAGEM DE PRODUTOS.....	4
2.3	TELA DE EDIÇÃO DE PRODUTO.....	5
3	CONCEITOS.....	6
3.1	API CRUD	6
3.2	REACT.JS	6
3.3	ANT DESIGN.....	6
3.4	ASP.NET CORE.....	6
4	ARQUITETURA DO SISTEMA	7
4.1	FRONT-END (REACT.JS)	7
4.2	BACK-END (ASP.NET CORE).....	7
5	FLUXO DE PROCESSOS	8
5.1	CRIAÇÃO DE PRODUTO:.....	8
5.2	EXIBIÇÃO DE PRODUTOS:.....	8
5.3	EDIÇÃO DE PRODUTO:	8
5.4	EXCLUSÃO DE PRODUTO:.....	8
6	TECNOLOGIAS USADAS	9
7	INSTRUÇÕES DE INSTALAÇÃO	10
7.1	FRONT-END	10
7.1.1	<i>Clonando o repositório:</i>	<i>10</i>
7.1.2	<i>Instalando as dependências:</i>	<i>10</i>
7.1.3	<i>Rodando a aplicação:</i>	<i>10</i>
7.2	BACK-END	10
7.2.1	<i>Clonando o repositório:</i>	<i>10</i>
7.2.2	<i>Instalando as dependências (se necessário):</i>	<i>10</i>
7.2.3	<i>Rodando a API:.....</i>	<i>10</i>
8	MELHORIAS FUTURAS	11
9	CONCLUSÃO	12

1 VISÃO GERAL

O **CRUD API** é uma aplicação que visa fornecer um gerenciamento completo de produtos para uma empresa, com uma interface simples e intuitiva. Este projeto foi desenvolvido com o objetivo de oferecer uma solução de controle de estoque, permitindo o cadastro, visualização, edição e remoção de produtos. A API do sistema foi construída utilizando **ASP.NET Core**, e o front-end foi desenvolvido com **React.js**, utilizando **Ant Design** para componentes de UI.

2 PROTÓTIPOS DE TELA

Aqui estão os protótipos de tela da aplicação, mostrando a interface de usuário para os principais fluxos do sistema.

2.1 Tela de Cadastro de Produto

Descrição: Esta tela permite o cadastro de novos produtos no sistema. O formulário inclui campos como nome do produto, preço de custo, preço de venda e quantidade.

Cadastro de Produtos

Nome

Nome do Produto

Preço de Custo

Preço de Custo

Preço de Venda

Preço de Venda

Quantidade

Quantidade

Adicionar Produto

2.2 Tela de Listagem de Produtos

Descrição: A tela de listagem exibe todos os produtos cadastrados no sistema. Nela, o usuário pode editar ou excluir produtos. A tabela é responsiva e usa componentes do **Ant Design** para exibição e interação.

ID	Nome	Preço Custo	Preço Venda	Quantidade	Ações
39	Mouse	20	40	5	<div>EditarDeletar</div>
40	Teclado sem fio	30	45	10	<div>EditarDeletar</div>
41	Headset	35	50	15	<div>EditarDeletar</div>
42	Mousepad	5	15	30	<div>EditarDeletar</div>
43	Porta copos	15	35	10	<div>EditarDeletar</div>

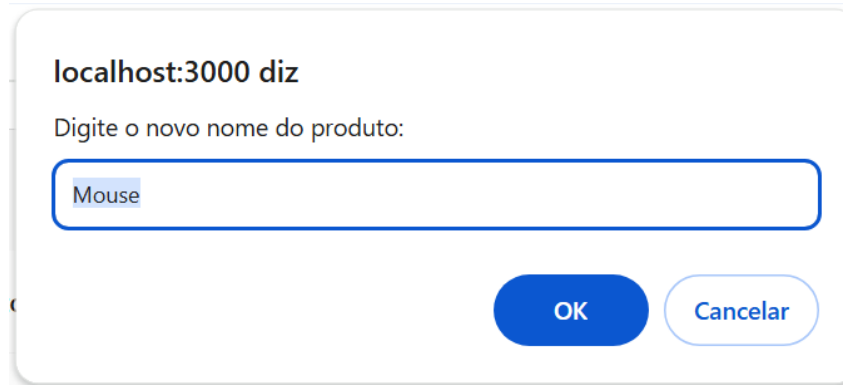
<

1

>

2.3 Tela de Edição de Produto

Descrição: Ao clicar em um produto na tabela, o usuário é levado a esta tela para editar os detalhes do produto, como nome, preço e quantidade.



The image shows a modal dialog box for editing a product. At the top, it displays the text "localhost:3000 diz". Below this, a label reads "Digite o novo nome do produto:". A text input field contains the word "Mouse". At the bottom right, there are two buttons: a solid blue "OK" button and a white "Cancelar" button with a blue border.

3 CONCEITOS

3.1 API CRUD

O termo **CRUD** refere-se às quatro operações básicas de persistência de dados:

- **Create** (Criar)
- **Read** (Ler)
- **Update** (Atualizar)
- **Delete** (Deletar)

Essas operações são aplicadas em nosso sistema de gerenciamento de produtos, onde podemos criar, listar, atualizar e excluir produtos do estoque.

3.2 React.js

O React é uma biblioteca JavaScript para a criação de interfaces de usuário. Sua principal característica é a capacidade de construir componentes reutilizáveis, o que torna o desenvolvimento mais modular e eficiente.

3.3 Ant Design

O **Ant Design** é uma biblioteca de componentes React que fornece uma coleção de componentes prontos para uso, altamente configuráveis e com design sofisticado. Ele facilita a criação de interfaces de usuário de alta qualidade com uma boa experiência visual.

3.4 ASP.NET Core

O ASP.NET Core é um framework de desenvolvimento web de código aberto da Microsoft. Ele permite a criação de APIs RESTful de alto desempenho que podem ser utilizadas por qualquer cliente HTTP. Neste projeto, ele foi utilizado para criar a API que manipula os dados dos produtos.

4 ARQUITETURA DO SISTEMA

4.1 Front-end (React.js)

O front-end da aplicação foi desenvolvido com **React.js** e utiliza a biblioteca **Ant Design** para a criação dos componentes de UI. O fluxo da interface se concentra na interação do usuário com as funções CRUD para gerenciar produtos.

- **Componente ProdutoForm:** Este componente é responsável por permitir que os usuários adicionem novos produtos ao sistema. Ele coleta informações sobre o produto, como nome, preço de custo, preço de venda e quantidade.
- **Componente ProdutoTable:** Exibe todos os produtos cadastrados na tabela. O usuário pode editar ou excluir produtos diretamente nesta tabela.
- **Componente AdicionarProduto:** Gerencia a exibição do formulário de criação de um novo produto. Ele lida com a interação do usuário ao adicionar um produto.

4.2 Back-end (ASP.NET Core)

O back-end é responsável por processar as requisições do cliente (front-end) e interagir com o banco de dados para realizar as operações CRUD.

- **Controladores:** São responsáveis por gerenciar as requisições HTTP (GET, POST, PUT, DELETE) e manipular os dados do produto.
- **Modelos de Dados:** Representam a estrutura dos dados que serão armazenados no banco de dados. Cada produto tem propriedades como nome, preço de custo, preço de venda e quantidade.
- **Banco de Dados:** Utiliza o **Entity Framework** para acessar o banco de dados, e as operações são executadas através de consultas LINQ.

5 FLUXO DE PROCESSOS

5.1 Criação de Produto:

- O usuário acessa o formulário de cadastro.
- Preenche as informações do produto (nome, preço de custo, preço de venda, quantidade).
- Ao submeter o formulário, uma requisição POST é enviada para a API, criando um novo produto no banco de dados.

5.2 Exibição de Produtos:

- A tabela de produtos é carregada com todos os produtos cadastrados.
- Ao acessar a página inicial, a aplicação faz uma requisição GET à API para buscar os produtos armazenados no banco de dados.

5.3 Edição de Produto:

- O usuário pode editar os produtos diretamente na tabela.
- Ao selecionar a opção de editar, o produto selecionado é carregado no formulário.
- Após as alterações, uma requisição PUT é enviada para a API para atualizar os dados no banco.

5.4 Exclusão de Produto:

- O usuário pode excluir um produto diretamente da tabela.
- Ao clicar na opção de excluir, uma requisição DELETE é enviada para a API para remover o produto do banco de dados.

6 TECNOLOGIAS USADAS

Front-end

- **React.js:** Biblioteca JavaScript para criar interfaces de usuário dinâmicas.
- **Ant Design:** Biblioteca de componentes UI para React, que fornece um conjunto de componentes prontos para uso.
- **Axios:** Biblioteca para realizar requisições HTTP para a API.

Back-end

- **ASP.NET Core:** Framework para construir APIs RESTful.
- **Entity Framework:** ORM para interação com o banco de dados.
- **SQL Server:** Banco de dados utilizado para armazenar as informações dos produtos.

7 INSTRUÇÕES DE INSTALAÇÃO

7.1 Front-end

7.1.1 Clonando o repositório:

```
git clone https://github.com/CarolPaneco/CRUD-API.git  
cd produto-frontend
```

7.1.2 Instalando as dependências:

```
npm install
```

7.1.3 Rodando a aplicação:

```
npm start
```

A aplicação será executada no navegador com a interface do front-end, normalmente em <http://localhost:3000>.

7.2 Back-end

7.2.1 Clonando o repositório:

```
git clone https://github.com/CarolPaneco/CRUD-API.git  
cd SistemaConsultaEstoque
```

7.2.2 Instalando as dependências (se necessário):

- Usando o Visual Studio, abra o projeto e certifique-se de que todas as dependências estão instaladas.

7.2.3 Rodando a API:

- No Visual Studio, pressione **F5** para executar o back-end.

A API estará rodando no endereço <https://localhost:7254>.

8 MELHORIAS FUTURAS

- **Autenticação de Usuário:** Adicionar funcionalidade de login e registro de usuários.
- **Validação de Dados:** Implementar validações no front-end e back-end para garantir a integridade dos dados.
- **Teste de UI:** Criar testes de interface para garantir que a experiência do usuário seja consistente.
- **Estilização Avançada:** Melhorar a estética da interface para torná-la mais atrativa.

9 CONCLUSÃO

Este projeto oferece uma base sólida para um sistema de gerenciamento de estoque, com uma interface moderna e funcional. A API RESTful garante um gerenciamento eficiente dos dados dos produtos, enquanto o front-end oferece uma experiência intuitiva e fácil de usar. O código está pronto para ser expandido e adaptado a diferentes necessidades, como autenticação de usuários ou integração com outros sistemas.