CODE TEST

Descrição do Projeto

O objetivo deste code test é avaliar os conhecimentos do candidato para uma posição de desenvolvedor full-stack. O candidato será responsável por criar uma tabela de usuários. O projeto deve ser desenvolvido utilizando as seguintes tecnologias e conceitos:

- **Nest.JS**: O servidor backend deverá ser desenvolvido utilizando Nest.js (Framework de Node.js https://docs.nestjs.com/)
- **React ou Vue**: O frontend será desenvolvido utilizando React.js ou Vue.js para criar a interface do usuário interativa.
- **Gerenciamento de Estado**: Um mecanismo de gerenciamento de estado como ContextAPI, Redux, MobX ou VueX deve ser utilizado para controlar o estado da aplicação.
- **Testes Unitários**: Devem ser escritos testes unitários no backend para garantir a qualidade e robustez do código desenvolvido.
- **Typescript**: O código deve ser escrito em TypeScript para aproveitar os benefícios do sistema de tipos estáticos.

Requisitos Obrigatórios do Projeto

O projeto deve atender aos seguintes requisitos:

1. Criação de uma tabela para exibir os dados de usuários.

2. A tabela deve ter as seguintes colunas:
    - Nome (string)
    - Idade (number)
    - Email (string)
    - Avatar (string)
    - Ações (botões para editar e excluir o usuário)

3. Os dados dos usuários devem ser obtidos por meio de uma API RESTful, buscando em um banco de dados MongoDB (pode-se utilizar um free cluster disponível no Mongo Atlas). Os dados podem ser mockados.

4. Ao clicar no botão "Editar" de um usuário, um modal deve ser exibido com um formulário para editar os dados do usuário.

5. O modal deve conter os campos:
    - Nome (string)
    - Idade (number)
    - Email (string)
    - Avatar (string)

6. O formulário de edição deve permitir a validação dos campos e exibir mensagens de erro em caso de valores inválidos.

7. Ao salvar as alterações no formulário, os dados do usuário devem ser enviados para a API e a tabela deve ser atualizada com os novos dados através de gerenciamento de estado.

8. Ao excluir ou adicionar um usuário deve-se realizar uma animação na tabela, representando a exclusão ou inclusão desse novo elemento.

9. O projeto deve ser responsivo e se adaptar a diferentes tamanhos de tela.

10. Devem ser escritos testes unitários para as principais funcionalidades do projeto.

Requisito Bônus
Além dos requisitos mencionados anteriormente, o seguinte requisito bônus pode ser implementado:

1. **Autenticação**: Adicione um sistema de autenticação à aplicação, de modo que apenas usuários autenticados possam visualizar e editar os dados dos usuários. Utilize um mecanismo de autenticação, como tokens JWT (JSON Web Tokens), para proteger as rotas relevantes da API.

Esse requisito bônus demonstrará habilidades adicionais na implementação de recursos de segurança e será considerado como um diferencial na avaliação do code test.

Certifique-se de fornecer instruções adicionais sobre como testar e interagir com esse requisito bônus, se implementado.

Instruções Adicionais

- Você é livre para utilizar bibliotecas e frameworks de componentes adicionais que julgar necessárias para implementar o projeto.

- Subir commits separadamente, por funcionalidade, arquivo, etc. Inclusive com a correta descrição semântica do conteúdo do commit.

- Esperamos que você crie um código limpo, bem organizado e seguindo as boas práticas de desenvolvimento.

- Dê preferência ao uso de componentes reutilizáveis e à separação clara de responsabilidades entre as partes do código.

- Certifique-se de que o código esteja bem documentado e adicione comentários sempre que necessário.

- Ao concluir o projeto, forneça instruções claras de como executá-lo localmente e como executar os testes unitários.

Avaliaremos o projeto em relação à sua qualidade de código, organização, escolha de bibliotecas e ferramentas, estrutura do projeto, testes unitários e aderência aos requisitos mencionados acima.

Se você tiver alguma dúvida, não hesite em perguntar. Boa sorte!