Desafio: *Desenvolvimento de uma Aplicação de Gerenciamento de Tarefas*

Objetivo:

O objetivo primário deste teste é avaliar a capacidade do candidato de construir uma aplicação SPA utilizando NextJS, incluindo mas não se limitando a hábilidade de criar componentes reutilizaveis, implementar roteamento e integrar APIs no Front-End. Para além disto, o teste permite avaliar competencias básicas de back-end, noções básicas de segurança e lógica de desenvolvimento.

O candidato será avaliado nos termos dos critérios de avaliação, bem como a partir da estrutura lógica de desenvolvimento.

Descrição Geral:

Desenvolva uma aplicação completa de gerenciamento de tarefas (Task Manager) que permita a criação usuários e SQUADS, permitindo criar, editar, excluir e visualizar tarefas, bem como filtrar informações de maneira eficiente. A aplicação deve incluir um frontend em React com NextJS, um backend em NodeJS, gerenciamento de estado com Redux, e um sistema de autenticação e autorização utilizando JWT.

Requisitos funcionais da aplicação:

1. Deverá possuir a funcionalidade de criação de usuários com perfís de: **Gerente** e **Funcionário**

2. Deverá possuir a funcionalidade de criação de **SQUADS** permitindo que usuários do tipo **Gerente** criem, editem e excluam SQUADS.

3. Deverá ter a funcionalidade de gerenciamento de **SQUADS**, permitindo que usuários do tipo **Gerente** adicionem usuários do tipo **Funcionários** aos SQUADS.

4. Deverá ter a funcionalidade de filtragem de tarefas por funcionário e squad. Usuários do tipo Funcionário **não deverão** visualizar tarefas de outros Funcionários.

Parte 1 - Front-End com NextJS/React

Descrição:

1. Crie uma aplicação SPA usando React/NextJS

2. Implemente paginas para listagem, criação, edição e exclusão de tarefas.

3. Cada tarefa deve ter um título, descrição, data de vencimento, pessoa responsável e status (pendente, em andamento, concluída)

4. Deve ser possível alterar a pessoa responsável e anexar evidencias de conclusão da tarefa.

5. A aplicação deve ter visualizações diferenciadas de acordo com o contexto de autorização.

Parte 2 - Back-end com NodeJS

Descrição:

1. Desenvolva uma API RESTful usando NodeJS que implemente as operações CRUD para gerenciamento das tarefas que serão consumidas pelo front-end.

2. A API deve incluir endpoints para criação, leitura, atualização e exclusão das tarefas.

3. Integre um banco de dados SQL para armazenar as tarefas.

Parte 3 - Gerenciamento de Estado com Redux

Descrição:

1. Utilize Redux para gerenciar o estado global da aplicação.

2. Implemente ações e reducers para gerenciar as operações de tarefas (criação, edição, exclusão e listagem).

Parte 4 - Autenticação e Autorização.

Descrição:

1. Implemente um sistema de autenticação e autorização utilizando JWT

2. Crie endpoints para registros de usuários

3. Proteja as rotas de gerenciamento de tarefas para que apenas usuários autenticados possam acessar.

Entrega Final:

Repositório Github com o código da aplicação, incluindo o front-end, back-end e um dump do banco de dados.

Documentação explicando como configurar e executar a aplicação localmente.

Critérios de avaliação:

1. Funcionalidade completa da aplicação.
2. Qualidade e clareza de código.
3. Estrutura e organização do projeto.
4. Uso apropriado de tecnologias e frameworks mencionados.
5. Implementação de autenticação e autorização.
6. Responsividade e usabilidade da interface de usuário.

Critérios de "desempate":

1. Clareza e qualidade do código javascript.
2. Avaliação da usabilidade/experiencia de usuário da aplicação.
3. Implementação de testes e cobertura de testes.
4. Documentação da aplicação