**Teste Frontend**

**Objetivo:**

Desenvolver uma aplicação web para gerenciamento de produtos, com foco em utilização de **Next**, gerenciamento de estado, consumo de APIs, estilização com **Tailwind CSS** e criação de testes.

**Descrição do desafio:**

Você deve criar uma aplicação de gerenciamento de produtos com as seguintes funcionalidades:

**Funcionalidades obrigatórias:**

1. Listar produtos com as seguintes informações:
   * Nome do produto
   * Categoria
   * Preço
   * Descrição
   * Imagem (URL da imagem)
2. Criar um formulário para cadastro de novos produtos:
   * Campos: Nome, Preço, Descrição e URL da Imagem.
   * O produto cadastrado deve ser exibido na lista.
3. Implementar um filtro para buscar produtos pelo nome, faixa de preço.
4. Implementar ordenação dos resultados.

**Requisitos técnicos:**

* Usar NextJs, Typescript.
* Gerenciar o estado global.
* Consumir uma API fictícia (mock com JSONPlaceholder ou outra biblioteca como msw).
* Estilizar o layout com **Tailwind CSS**.

**Extras (diferenciais):**

* Implementar paginação para a lista de produtos.
* Tornar o layout responsivo.
* Escrever documentação simples explicando as escolhas feitas no projeto.
* 1 teste de snapshot de tela.

PS: A beleza da tela não será critério de avaliação, mas de desempate.

**Teste Backend**

**Objetivo:**

Desenvolver uma API RESTful para gerenciamento de pedidos e produtos, com foco em boas práticas (SOLID), organização de código em classes, e manipulação de banco de dados.

**Descrição do desafio:**

Você deve criar uma API para um sistema de gerenciamento de pedidos e produtos.

**Funcionalidades obrigatórias:**

1. **Produtos:**
   * Criar, listar, editar e deletar produtos.
   * Cada produto deve conter os seguintes campos:
     + id (autogerado)
     + nome
     + categoria
     + descrição
     + preço
     + quantidade\_estoque
2. **Pedidos:**
   * Criar e listar pedidos.
   * Cada pedido deve conter os seguintes campos:
     + id (autogerado)
     + produtos (lista de produtos no pedido, incluindo quantidade de cada um).
     + total\_pedido (valor total do pedido).
     + status ("Pendente", "Concluído" ou "Cancelado").
   * Ao criar um pedido:
     + Verificar se a quantidade de cada produto está disponível no estoque.
     + Atualizar o estoque caso o pedido seja concluído.

**Requisitos técnicos:**

* Usar **Node.js** com **NestJS**.
* Organizar o código seguindo boas práticas:
  + Separar em camadas (controller, service, repository).
  + Aplicar princípios do **SOLID**.
* Utilizar banco de dados relacional ou não relacional
* Implementar pelo menos um middleware (ex.: log de requisições).
* Escrever pelo menos:
  + 2 testes unitários.

**Extras (diferenciais):**

* Documentar a API com Swagger ou Postman.
* Adicionar autenticação simples com JWT.
* Implementar Docker para subir o ambiente.

**Entrega do Teste**

* Prazo: 5 a 7 dias úteis.
* Entrega: Subir o código em um repositório Git (público ou privado) e compartilhar o link.

**Critérios de Avaliação:**

1. **Organização do código**: Boa estruturação e padrões adotados.
2. **Cumprimento dos requisitos**: Atender às funcionalidades descritas.
3. **Qualidade do código**: Leitura, reutilização e boas práticas (SOLID, DRY, etc.).
4. **Testes**: Abrangência e eficiência dos testes implementados.
5. **Extras entregues** (se houver).

**Dica:**

Documente bem o repositório com instruções de como rodar o projeto e rodar os testes. Isso será um diferencial!