Test Technique MPEL DIGITAL - Développeur Web

Objectif

Créer une application complète comprenant un backend et un frontend permettant la gestion d'une fiche produit. Le candidat devra implémenter un CRUD de produits avec une architecture bien pensée.

Contexte

Vous devez développer une application web qui permet de gérer des produits. Chaque produit possède les informations suivantes :

- **Nom** (string, obligatoire)
- **Description** (string, facultatif)
- **Prix** (number, obligatoire, supérieur à 0)
- **Stock** (number, obligatoire, supérieur ou égal à 0)
- **Image** (URL, facultatif)

L'application doit permettre :

- **Créer un produit**
- **Lire la liste des produits**
- **Mettre à jour un produit**
- **Supprimer un produit**

Stack technique

Backend

- **Langage** : Aux choix, Typescript sera un plus.
- **Framework** : Express.js ou NestJS.
- **Base de données** : Au choix (PostgreSQL, MongoDB, MySQL, SQLite, etc.).
- **Gestion des erreurs et logs** : Recommandé.

Frontend

- **Framework** : React.js ou Next.js (en TypeScript sera un plus).
- **Style** : TailwindCSS.
- **Gestion des requêtes API** : Fetch ou React Query (React Query est un plus).
- **Utilisation de Server Components (Next.js uniquement)** : Sera un plus.
- **Utilisation de Custom Hooks** : Sera un plus.

Critères d'évaluation

1. Backend

- **Architecture du projet** : séparation des couches (routes, services, contrôleurs, etc.).
- **Sécurité** : validation des données, gestion des erreurs, gestion des autorisations si applicable.

- **Efficacité** : performances, pagination pour la liste des produits si applicable.
- **Documentation** : README clair et concis.
- **Qualité du code** : respect des bonnes pratiques TypeScript et Express.js/NestJS.

2. Frontend

- **Architecture du projet** : séparation des composants, custom hooks si utilisés.
- **Expérience utilisateur (UX)** : interface claire.
- **Gestion des données** : bon usage des API, utilisation de React Query ou Fetch.
- **Performances** : gestion des re-render, optimisation de la récupération des données.
- **Qualité du code** : respect des bonnes pratiques TypeScript, lisibilité et maintenabilité.

Instructions

- Créez votre repo et envoyez le directement par mail à adrien@elyamaje.com.
- Codez l'application en suivant les consignes ci-dessus.
- Fournissez un fichier **README** expliquant comment lancer votre projet (backend et frontend).
- Présentez votre projet et les choix techniques dans une courte **documentation.**
- Envoyez votre projet sous **5 jours.**

Bonne chance ! 🚀