Date: 17/02/2025 - 21/02/2025

SPRINT 3 BRIEF 2

Brief 2: Gestion des stocks d’un site e-commerce via une page admin

Contexte professionel

404.js souhaite développer un **système de gestion de stock** pour une plateforme e-commerce. Cette nouvelle fonctionnalité permet aux administrateurs de gérer facilement les produits disponibles sur le site via une **interface admin**.

L’objectif est de créer une **page admin dynamique** où l’administrateur peut ajouter, modifier, supprimer et consulter les produits du catalogue, avec toutes les informations nécessaires telles que les images, le titre, la description, le prix et le stock disponible.

L’API REST associée gérera le stockage des produits et assurera la communication entre le frontend et la base de données.

Conception de l’API REST

Le backend sera développé avec **Node.js** et **Express.js**, et communiquera avec une base de données **MongoDB** via **Mongoose**.

L’API devra permettre :

- L’ajout de nouveaux produits avec image, titre, description, prix et stock.
- La modification des informations d’un produit existant.
- La suppression d’un produit.
- L’affichage de tous les produits disponibles.

Validation des données :

- Vérifier que tous les champs obligatoires sont remplis.
- Valider que le prix est un nombre positif.
- S’assurer que le stock est une valeur numérique valide.

Mise à jour du Frontend

1. **Page Admin - Gestion des Produits** :
    - Un formulaire permettant d'**ajouter un produit** avec image, titre, description, prix et stock.
    - Une **liste des produits disponibles**, affichant les informations et permettant la modification ou suppression d'un produit.
2. **Connexion avec l'API REST** :
    - Intégration de `fetch` ou `axios` pour la gestion des requêtes.
    - Affichage en **temps réel** des produits disponibles après ajout/suppression/modification.

Modélisation UML

Avant le développement, une **modélisation UML** du système doit être réalisée pour structurer l'architecture de l'application. Elle devra inclure :

- **Diagramme de classes** : représentation des entités produit et leurs relations.
- **Diagramme de séquence** : interactions entre l'admin et l'API REST lors de l'ajout ou de la mise à jour d'un produit.
- **Diagramme de cas d'utilisation** : description des actions possibles sur la page admin.

Technologies utilisées

- **Frontend** : React JS, Tailwind CSS
- **Backend** : Node.js, Express.js, Mongoose
- **Base de données** : MongoDB.
- **Outils** : Postman (pour tester l'API), GitHub, Draw.io ou un autre outil UML

Modalités pédagogiques

- **Travail** : individuel
- **Durée** : 5 jours (17/02/2025 - 21/02/2025)
- **Évaluation** : Présentation du projet, démonstration de la gestion des produits, revue de code et questions techniques.

Livrables

1. **Code source du projet** (GitHub).
2. **Lien de planification** (Trello, Jira, etc.).
3. **Diagrammes UML** (Classes, Séquence, Cas d'utilisation).
4. **Présentation du projet** (Canva, PowerPoint ou équivalent).

Critères de performance

1. **Modélisation UML** :
    - Diagrammes UML clairs et bien structurés.
2. **Fonctionnalité de l'API** :
    - L'API permet l'ajout, la modification, la suppression et l'affichage des produits.
3. **Frontend mis à jour** :
    - Le formulaire permet l'ajout de produits avec validation.
    - La liste des produits s'affiche dynamiquement et peut être modifiée/supprimée.
4. **Qualité du code** :
    - Structure propre et utilisation des bonnes pratiques Express/Mongoose.
5. **Expérience utilisateur** :
    - Interface fluide et responsive.
    - Mise à jour en temps réel des produits.

les compétences techniques visées : C1N2, C4N2, C3N2, C5N2, C6N2, C7N2.
les compétences transversales visées : CT1N2, CT4N2,CT6N2.