Clean Code & Clean Architecture
Projet Pédagogique
5ème année Ingénierie du Web

Introduction :

Triumph Motorcycles cherche à améliorer la gestion des motos utilisées par des entreprises partenaires (livreurs, coursiers, services de location, etc.) et par ses concessionnaires. Le projet consiste à développer une **plateforme de gestion de flotte** permettant à ces partenaires de suivre et optimiser l'utilisation des motos, tout en se concentrant sur des aspects comme la gestion des entretiens, le suivi du cycle de vie des motos, et d'autres aspects liés à la gestion efficace d'une flotte de véhicules.

Fonctionalités (15 points) :

Gestion des entretiens préventifs et curatifs :

- **Planification des entretiens** : Possibilité de définir des intervalles d'entretien en fonction du du modèle de moto choisi par un client (ex. 10 000 km pour une Street Triple ou tous les ans, 16 000 km pour une Tiger Sport 660, etc).
- **Rappels automatiques** : Notifications envoyées aux gestionnaires et aux clients lorsqu'un entretien est dû, avec possibilité de renseigner manuellement le kilométrage par le client.
- **Suivi des entretiens réalisés** : Historique détaillé des entretiens effectués, incluant les pièces changées, les coûts associés et les recommandations des techniciens, à la fois côté gestionnaire et client.
- **Gestion des pannes & garanties** : Enregistrement des pannes et garanties, gestion des réparations et suivi des actions correctives avec historisation.

Gestion des pièces détachées et des stocks :

- **Suivi des pièces détachées** : Gestion du stock de pièces détachées utilisées pour la maintenance (filtre à huile, pneus, freins, etc.).
- **Alerte de stock bas** : Notifications envoyées lorsqu'une pièce atteint un seuil critique de disponibilité.

- **Historique des commandes de pièces** : Suivi des commandes de pièces, coûts, délais de livraison et quantité restante.

Suivi des essais :

- **Profil des conducteurs** : Gestion des conducteurs utilisant les motos, avec des informations sur leur permis, leur expérience, et leur historique de conduite.
- **Essais moto** : Gestion des motos assignées à chaque conducteur avec suivi des dates et de la durée de l'utilisation pour les essais moto.
- **Historique des incidents** : Enregistrement des incidents liés à chaque conducteur (accidents, infractions, etc.).

Contraintes techniques :

1. **Langage** : Développement en **TypeScript** (backend et frontend).
2. **Clean Architecture** :
    - Séparation stricte des couches : **Domain (Entities)**, **Application (Use Cases)**, **Interface** (API/Interface utilisateur), et **Infrastructure** (base de données, frameworks, etc.).
    - Chaque couche doit être indépendante des frameworks spécifiques pour faciliter la maintenance.
    - Proposer 2 adaptateurs (in-memory, SQL, NoSQL, etc) pour les bases de données et 2 frameworks backend (Nest.js, Express, Fastify, etc).
3. **Clean Code** :
    - Respect des principes de Clean Code vu en cours.
    - Les pratiques supplémentaires et documentées sous la forme d'œuvre et d'ouvrages sont aussi à prendre en compte (livres de Bob Martin, etc).

Bonus :

1. **CQRS :**
    a. Utiliser des commandes pour les requêtes
    b. Utiliser des queries pour les demandes
    c. Permet de préparer l'Event-Sourcing
2. **Event-Sourcing :**
    a. Utiliser l'Event-Sourcing avec comme objectif le retour dans le temps des événements passés
    b. Utilisation de microservices bienvenue
3. **Framework Frontend :**
    a. Utilisation de plusieurs frameworks frontend
    b. Angular, React & Solid.js à privilégier
    c. Lister les avantages et inconvénients de chacun