EC03 – Rattrapage : Développement back-end “Gestion des produits et alertes” avec BDD relationnelle + NoSQL

Contexte général

Dans le cadre du projet **FoodSafe**, l’épreuve de rattrapage consiste à développer une partie du back-end dédiée à la **gestion des produits alimentaires et des alertes d’allergènes**.

L’objectif est de manipuler simultanément une base **relationnelle** (PostgreSQL) et une base **NoSQL** (MongoDB), afin de gérer deux dimensions complémentaires :

- les **produits et leurs métadonnées** (nom, marque, composition, allergènes connus) stockés en SQL,
- les **alertes et signalements utilisateurs** (ex. : “mauvais étiquetage”, “réaction signalée”) stockés en NoSQL.

Le socle technique est fourni : environnement Docker configuré, authentification de base, bases de données initialisées et documentation fonctionnelle.
L’évaluation porte à la fois sur la qualité du code, la cohérence des choix techniques et la rigueur de la documentation.

Livrables attendus

Livrable	Détail
Code source	Développement en POO / MVC avec BDD relationnelle ET NoSQL
Dump SQL et JSON	Extraits de données (PostgreSQL et MongoDB)

Livrable	Détail
Scripts de sauvegarde	`backup.sh` , `restore.sh` pour les deux bases
README	Justification des choix techniques, commandes terminal, arbitrages

Modalités d’évaluation

- **Type d’épreuve** : Mise en situation reconstituée sur ordinateur
- **Durée** : 4h
- **Nature** : Épreuve individuelle, sans oral
- **Critères évalués** :
  - C7.1 : Configuration environnement de dev (Docker, terminal)
  - C8.1 : Programmation orientée objet, architecture MVC
  - C9 : Sécurisation et optimisation du serveur
  - C10 : Utilisation de base de données relationnelle (PostgreSQL)
  - C11 : Utilisation de base de données NoSQL (MongoDB)
  - C12 : Justification des choix BDD (SQL vs NoSQL)
  - C13 : Mise en place de la sauvegarde et restauration

Recommandations

- Structurer le code selon le **pattern MVC** et appliquer une **POO propre et modulaire**
- Sécuriser les routes d’administration et de création d’alertes via **JWT ou middleware d’autorisation**
- Fournir des **migrations SQL** pour la base relationnelle (produits, marques, allergènes)
- Créer des **collections MongoDB cohérentes** pour les alertes et retours utilisateurs
- Documenter clairement les **choix techniques** et les **commandes d’exécution**
- Produire des **scripts fonctionnels** pour la sauvegarde et la restauration des données

Exemples d’éléments attendus

- Une entité `Product` persistée dans PostgreSQL avec ses relations (`Brand` , `Allergen`)

- Une collection `Alerts` dans MongoDB pour stocker les signalements utilisateurs
- Une route sécurisée (`GET /products/alerts`) combinant les données des deux bases
- Des fichiers `.env` , `docker-compose.yml` et scripts shell (`backup.sh` , `restore.sh`) fonctionnels
- Des exemples de requêtes et logs illustrant les opérations SQL/NoSQL

Nom de dossier attendu

- Une archive ZIP nommée `EC03R_NomPrenom.zip` contenant :
  - Le dossier `src` du projet
  - Le fichier `README.md` détaillant l'architecture et les choix techniques
  - Le dump SQL (`/data/foodsafe_products.sql`) et le dump Mongo (`/data/alerts.json`)
  - Les scripts `backup.sh` et `restore.sh`

Rappel pédagogique

Cette épreuve de rattrapage vise à valider les mêmes compétences que l'EC03 initiale :

- Maîtrise du **développement back-end en POO et MVC**
- Utilisation complémentaire de **bases SQL et NoSQL** pour modéliser des données réelles
- Capacité à **sécuriser, documenter et justifier** ses choix techniques
- Connaissance pratique de l'**environnement Docker et des scripts système**

Le scénario change (gestion des produits et alertes au lieu des profils utilisateurs), mais le niveau d'exigence et les objectifs pédagogiques restent identiques.