EC03 – Développement back-end avec BDD relationnelle + NoSQL

Contexte général

Dans le cadre du projet **FoodSafe**, cette épreuve consiste à développer une partie du back-end dédiée à la **gestion des utilisateurs et de leurs profils alimentaires**, en manipulant à la fois une base **relationnelle** et une base **NoSQL**.

Le socle technique est fourni : authentification de base déjà en place, environnement Docker configuré, bases de données initialisées (PostgreSQL + MongoDB), et documentation fonctionnelle. L'objectif est de mettre en œuvre des opérations ciblées sur ces deux types de bases.

L'évaluation porte autant sur l'écriture de code que sur la capacité à justifier les choix techniques et à documenter les arbitrages.

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)
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, 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

- Respecter les bonnes pratiques de **séparation des responsabilités** (MVC)
- Sécuriser l'accès aux données sensibles : variables d'environnement, headers HTTP, gestion des rôles
- Utiliser **des migrations** ou scripts pour la base relationnelle (PostgreSQL)
- Fournir des **dumps de données cohérents** et testables (profils utilisateurs + fiches produit)
- Documenter clairement les **choix techniques** dans le README
- Fournir des **scripts automatisés** simples pour lancer, tester, sauvegarder et restaurer

Exemples d'éléments attendus

- Une entité `User` et `AllergyProfile` persistées dans PostgreSQL avec migrations
- Une collection `Products` et `SearchHistory` dans MongoDB
- Une route protégée (JWT) listant les produits compatibles avec un profil utilisateur
- Un fichier `.env` clair, des logs pertinents, un `docker-compose.yml` fonctionnel
- Des captures de terminal montrant les commandes utilisées (migrations, requêtes, sauvegardes)

Nom de dossier attendu

- une archive ZIP nommée `EC03_NomPrenom.zip` contenant :
  - Le dossier `src` du projet

- Le fichier `README.md` avec explications techniques
- Le dump SQL (`/data/foodsafe.sql`) et le dump Mongo (`/data/mongo.json`)
- Les scripts `backup.sh` et `restore.sh`

Rappel pédagogique

Cette épreuve permet d'évaluer la **capacité à manipuler des architectures back-end réalistes**, en incluant :

- La **programmation orientée objet** et l'usage d'un framework MVC
- L'exploitation de **deux types de bases de données complémentaires** (relationnelle pour les utilisateurs, NoSQL pour les produits et historiques)
- La maîtrise de l'**environnement de développement** (Docker, terminal)
- La capacité à **justifier techniquement des arbitrages** entre SQL et NoSQL

Elle introduit une logique métier complète autour de la gestion des utilisateurs et prépare à la mise en place d'une API sécurisée (EC04).