TP : API Microservices pour un Réseau social

Contexte :

Vous êtes LE/LA développeur.se back-end au sein d'une jeune startup ambitieuse, décidée à créer un réseau social révolutionnaire et à défier l'empereur Melon Must, et son fameux Y, en imposant ses propres règles. Votre mission est de concevoir une **API scalable**, en utilisant une architecture **microservices, et MongoDB**, qui supportera les fonctionnalités principales de ce nouveau réseau social.

Cette API devra être composée de trois microservices distincts et indépendants, chacun gérant une partie spécifique de l'application :

1. **Service d'Authentification et de Gestion des Utilisateurs** :

    - Gestion des **utilisateurs** avec des informations telles que **userName**, **password**, et un système de **rappel de mot de passe**. (au cas où l'utilisateur l'oublie)
    - Chaque utilisateur doit être unique et le mot de passe doit être **crypté** et **sécurisé**. Il est formellement interdit de stocker des mots de passe en clair dans la base de données. (RGPD)
    - Le service devra permettre à un utilisateur de se **connecter**, de **créer un compte** et de **récupérer son mot de passe**.

2. **Service de Gestion des Posts** :

    - Chaque utilisateur pourra créer des **posts** contenant un **texte** ou un **contenu** quelconque.
    - Chaque post doit également avoir un compteur de **likes** afin de mesurer son interaction avec les autres utilisateurs.
    - Le service devra gérer l'ajout, la mise à jour, et la suppression des posts.

3. **Service des Likes** :

    - Les utilisateurs pourront aimer des **posts**.
    - Un **like** est lié à un **post spécifique** et à un **utilisateur**. Chaque like devra être associé à un post spécifique.
    - Le service devra permettre l'ajout, la suppression et la gestion des likes sur les posts.

Objectifs du TP :

- **Microservices indépendants** : Chaque service (authentification, gestion des posts, gestion des likes) doit être indépendant et répondre à une partie spécifique de l'application.
- **Sécurisation des données** : Assurez-vous que les mots de passe des utilisateurs soient stockés de manière sécurisée (en utilisant des algorithmes de hachage comme bcrypt ou argon2) et ne jamais être stockés en clair dans la base de données.
- **Scalabilité** : Concevoir une API capable de s'adapter à la montée en charge, en utilisant une architecture flexible.
- **Gestion des relations entre les services** : Même si chaque microservice est indépendant, vous devrez établir des liens entre eux. Par exemple, un like doit être rattaché à un post spécifique et à un utilisateur.

Créer un fichier .env avec toutes les routes de votre API :

exemple : POSTS = “mylink:3000”
USERS = “myUsersLink:3000”