Création d'une API REST pour un service de matchmaking de jeux

Objectif

Le but de ce TP est de développer une API REST qui permet aux utilisateurs de se connecter, de recevoir des succès de jeux, et de trouver des matchs sur des serveurs de jeux. Le système doit utiliser une authentification JWT pour les utilisateurs et un système de matchmaking basé sur des statistiques simples. Vous utiliserez Docker Compose pour lancer l'environnement de développement, incluant la base de données PostgreSQL, un serveur REDIS, et votre API Web.

Fonctionnalités requises (20 points)

1. **Authentification et gestion des utilisateurs :**
    - Les utilisateurs sont définis en base de données avec un UUID, username, email, password, et salt.
    - Implémenter une authentification JWT permettant aux utilisateurs de se connecter à l'API.
2. **Gestion des succès de jeux :**
    - Les succès de jeux sont définis en base de données par un nom, une description, une image, et un UUID.
    - Les utilisateurs authentifiés peuvent recevoir des informations sur les succès de jeux.
    - Seuls un dedicated game server peut grant un achievement à un joueur.
3. **Matchmaking :**
    - Le processus de matchmaking utilise les statistiques du joueur (kill/death ratio moyen et niveau (Bronze, Argent, Or, Platinum, Diamond).
    - Les niveaux des joueurs sont configurables dans la base de données.
    - Un serveur s'enregistre auprès de l'API en démarrant, enregistrant sa session dans une base de données Redis. La session comprend l'adresse du serveur, les joueurs présents, et d'autres informations pertinentes.
4. **Docker Compose :**
    - Mettre en place un Docker Compose pour lancer PostgreSQL, REDIS, et l'API Web.
    - S'assurer que l'API peut communiquer avec PostgreSQL et REDIS, mais que ces deux services ne peuvent pas communiquer directement entre eux.
5. **Intégration dans votre jeu**

Points bonus pour les meilleurs d'entre vous :p

1. **Gestion d'un système d'amis (10pts):**
    - Permettre aux joueurs de s'ajouter en amis et de voir les informations de base des parties auxquelles ils jouent (nom de la map, niveau du joueur).
2. **Caching Redis (10pts)**
    - Mettre en place du caching pour les informations des utilisateurs et des succès de jeux en utilisant REDIS.
3. **Déploiement Kubernetes (10pts):**
    - Convertir la configuration Docker Compose en un déploiement Kubernetes, qui sera exécuté sur une instance de microK8S.

Consignes

- Respectez les bonnes pratiques de développement, notamment en ce qui concerne la sécurité des données et des communications.
- Documentez votre code et votre architecture, en expliquant les choix technologiques et architecturaux.
- Testez votre API pour vous assurer de sa fiabilité et de sa performance.
- Si ça ne compile pas c'est 0
- À rendre pour le 1er Mai à 23H59
- Le rendu doit se faire au moyen d'un lien Git **PUBLIC**
- Toute documentation doit être dans le README ou le Wiki du Git

Bonne chance!