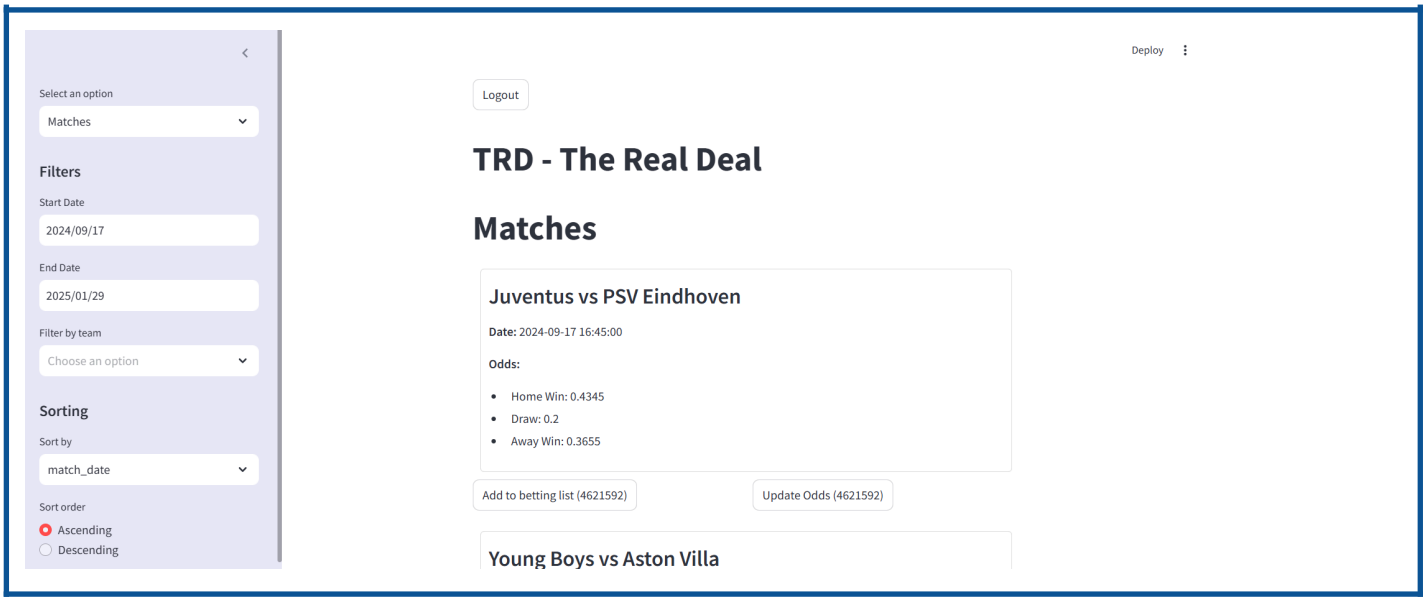
THE REAL DEAL

Application en architecture orientée service



Olha ALIEINIK - Doriane BEDIER - Oleksandra KUKSA

M2 SID - 2024/2025

Sommaire

1 - Contexte du projet

The Real Deal (TRD) est un **site spécialisé dans les jeux d'argent en ligne**, notamment dans les paris hippiques et dans le casino, et souhaite étendre ses activités aux paris sportifs. Nous avons développé une application en architecture micro service afin de permettre aux utilisateurs de parier sur des matchs de foot au sein de notre site.
Voici une vue d'ensemble du projet, donnée par le biais de diagrammes montrant l'architecture de l'application et son fonctionnement.
Vous pouvez accéder à une démonstration vidéo du site via **ce lien** (menant vers un dossier google drive)

2 - Architecture du système

2.1 - Vue d'ensemble

Ce schéma représente l'architecture globale de notre application en **micro-services.**

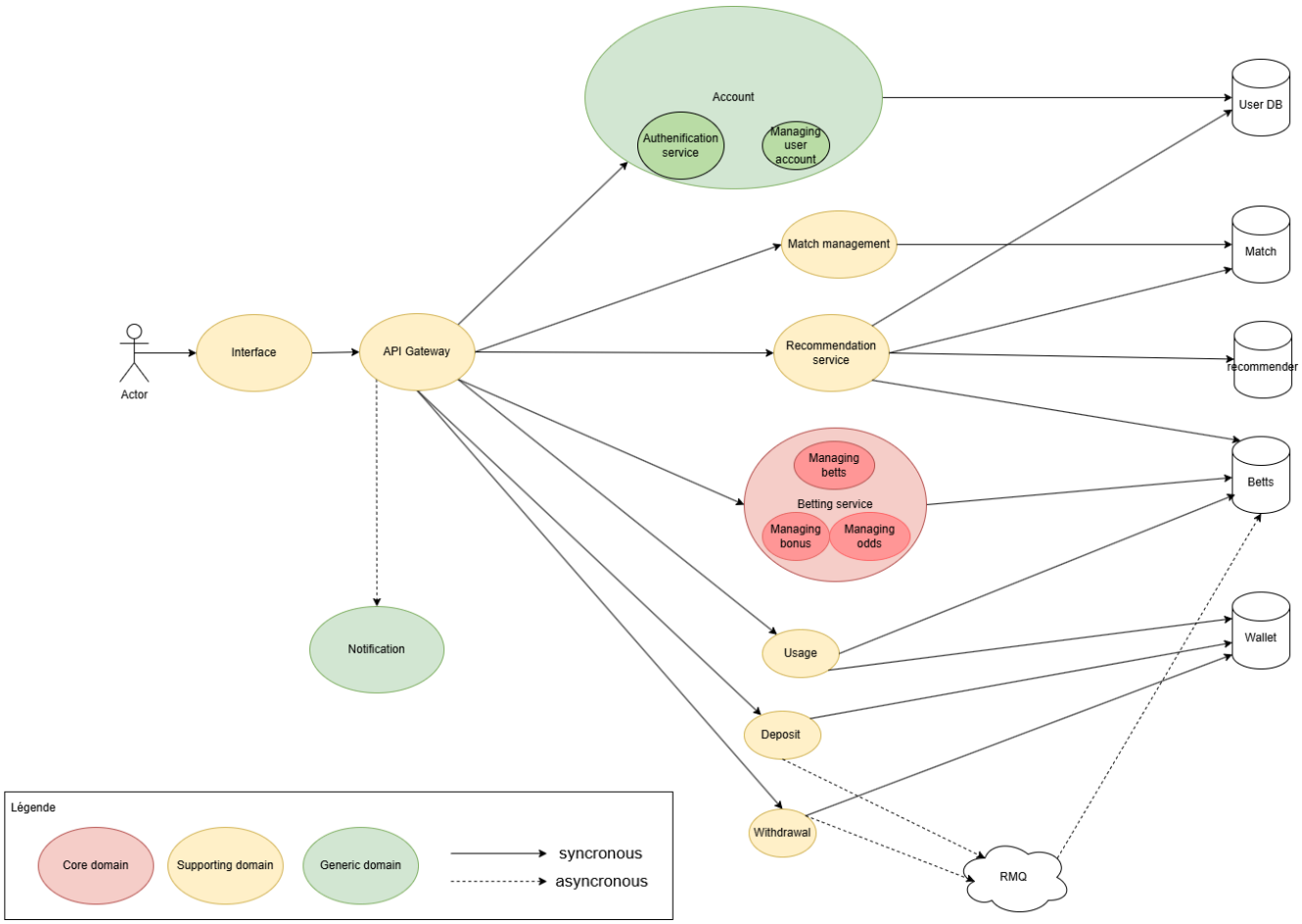


Figure 1 : Architecture de l'application The Real Deal

L'utilisateur communique avec l'application seulement via l'interface, implémentée dans notre service "Streamlit-app", et qui fait office de gateway.
L'application a été développée en Python, l'interface est faite avec streamlit. Chaque service et base de données ont été créés au sein d'un conteneur Docker différent.

2.2 - Fonctionnalités

Nous avons les fonctionnalités suivantes disponible pour un utilisateur du site :

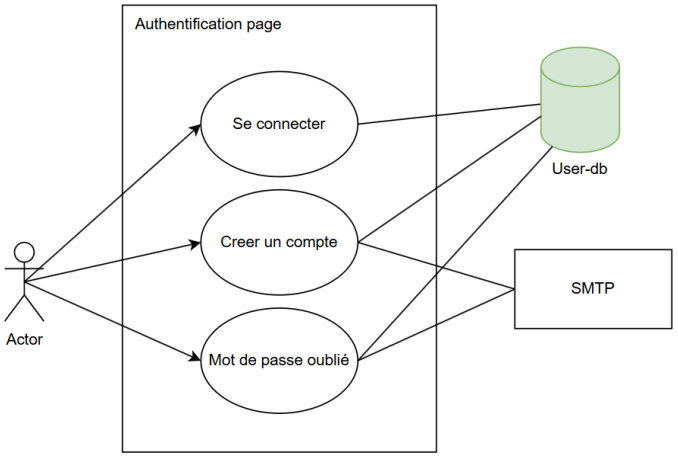


Figure 2 : Diagramme d'utilisation de TRD - page Authentification

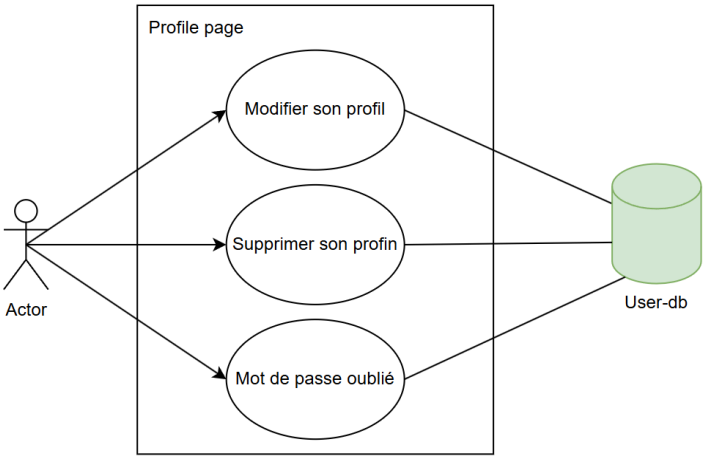


Figure 3 : Diagramme d'utilisation de TRD - page Profile

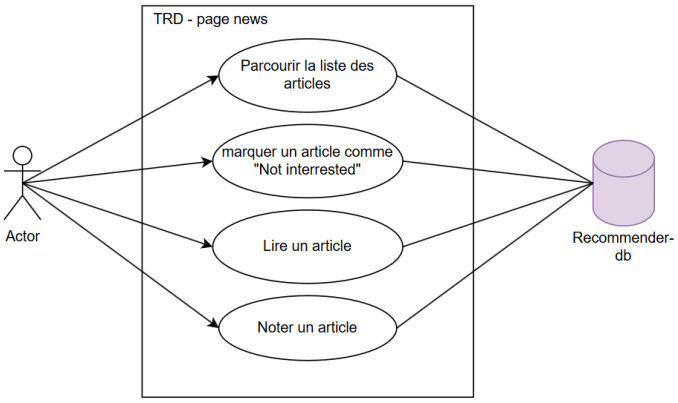


Figure 4 : Diagramme d'utilisation de TRD - page News

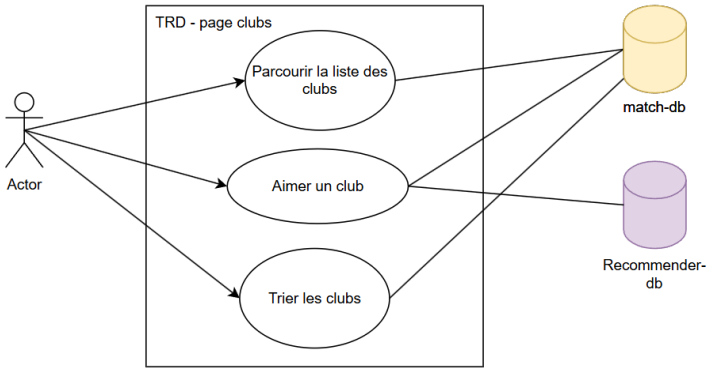


Figure 5 : Diagramme d'utilisation de TRD - page Clubs

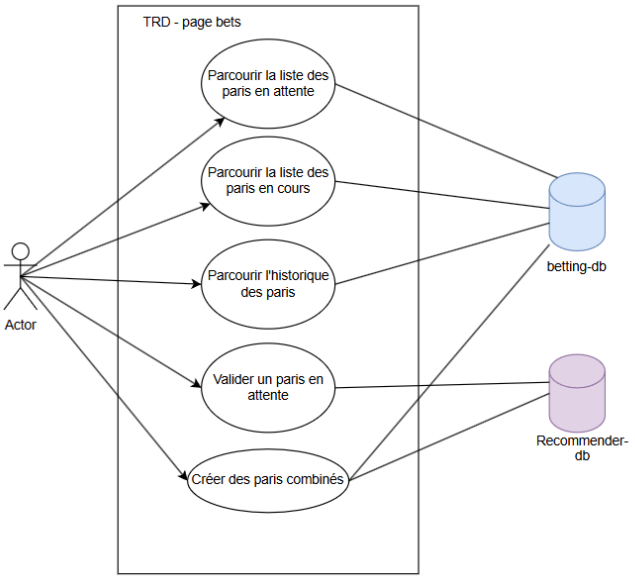


Figure 6 : Diagramme d'utilisation de TRD - page Bets

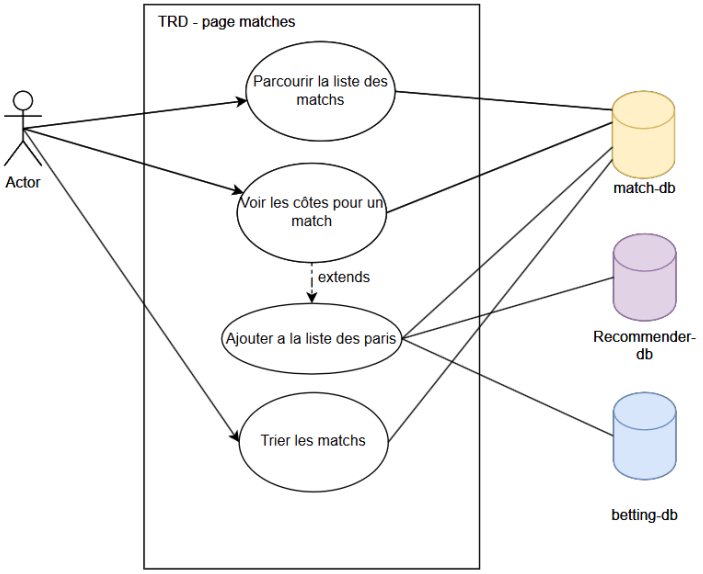


Figure 7 : Diagramme d'utilisation de TRD - page Matches

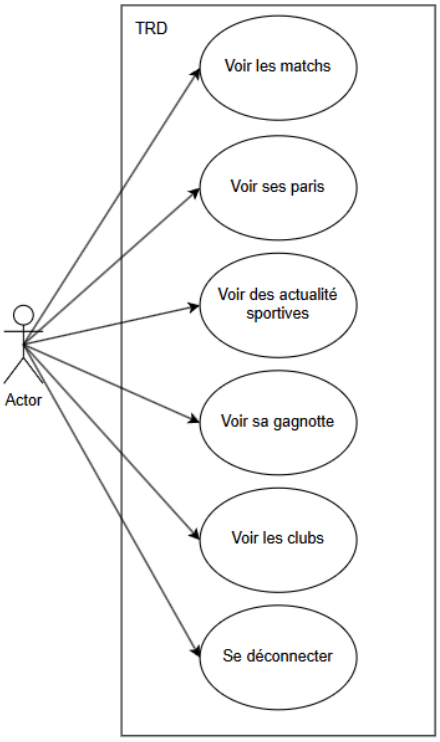


Figure 8 : Diagramme d'utilisation de TRD - toutes les pages

2.3 - Exemple d’utilisation

Ce diagramme de séquence vous présente une utilisation classique du site : un utilisateur (connecté) fait un pari. Il sera aussi présenté en un peu plus gros en annexe

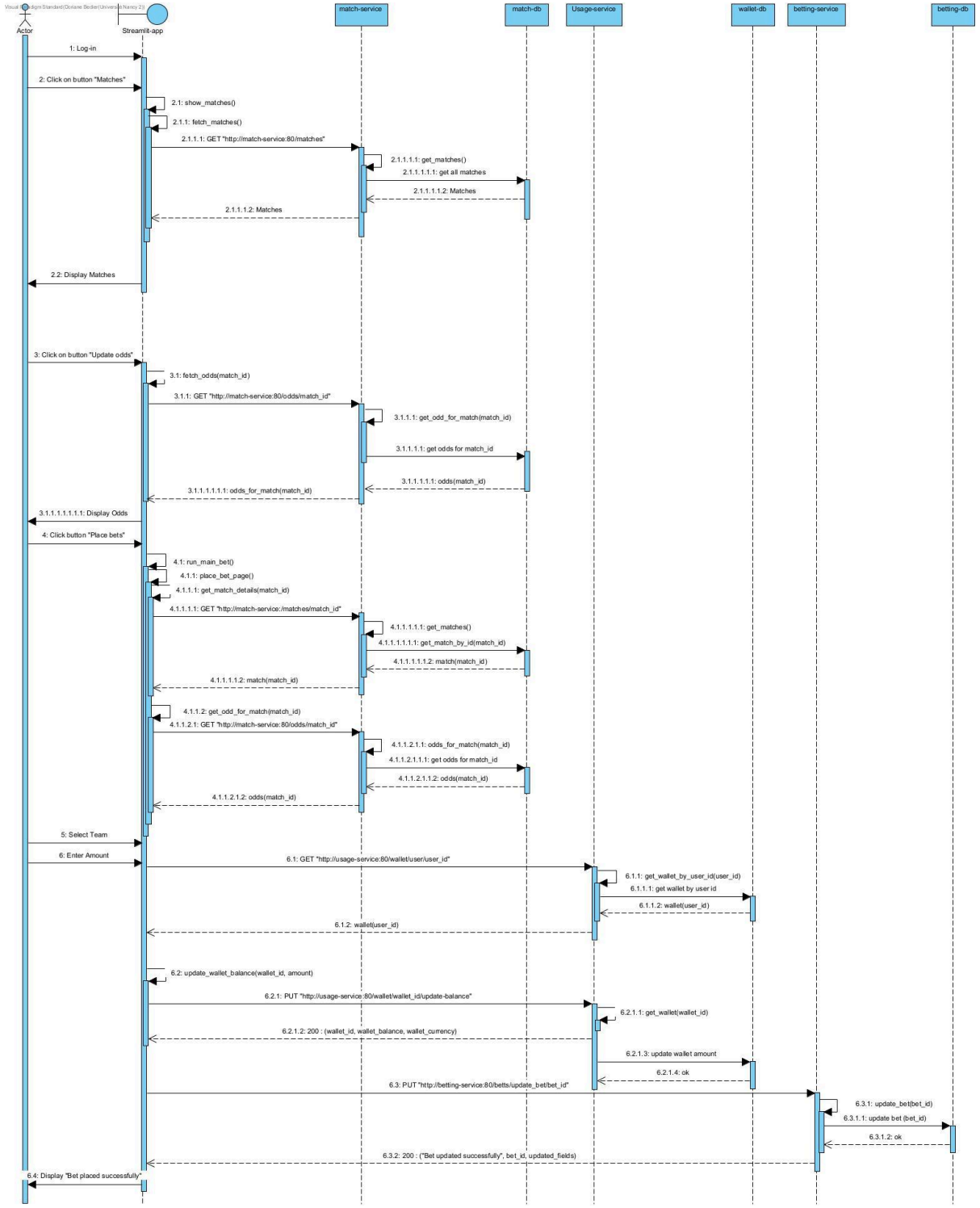


Figure 9 : Diagramme de séquence : Faire un pari (pour un utilisateur connecté)

Ce diagramme nous permet de voir les interactions entre les services dans un cas concret.
Le point de liaison entre tous ces services est Streamlit-app, qui fait office de gateway, et qui permet une communication entre tous.

2.4 - Sources des données

Au sein de notre application sont utilisées des données publiques et des données internes, nous permettant de faire fonctionner notre site.

Données publiques :

Les données publiques incluent les articles sportifs collectés à partir d'API publiques fournissant des nouvelles et informations liées aux événements sportifs.
Nous avons récupéré toutes les données concernant le football sur ce site :
https://rapidapi.com/Creativesdev/api/free-api-live-football-data/playground/apiendpoint_65078b41-9d2e-45bc-ad61-0216ba285ff2

Cette API met à disposition de nombreux endpoints permettant d'accéder à différentes ressources.

Nous avons utilisé pour récupérer les articles les endpoints suivant :

- /football -get-all-leagues
- /football-get-all-team?{league_id}
- /football-get-team-news?{team_id}

Nous avons placé les articles ainsi obtenus d'abord dans un CSV, puis nous l'avons chargé dans la table sport_news de recommender-db.

En ce qui concerne les matchs, nous avons utilisé les données trouvées sur le Discord du projet, et nous les avons placées dans la table "matches" de match-db.

Données internes :

Les données internes sont issues des interactions des utilisateurs avec le site. Elles comprennent les clics sur les articles, les paris effectués, ainsi que les évaluations et consultations des articles.
Lors de l'interaction de l'utilisateur avec l'interface de notre site, nous récupérons les actions qu'il effectue et nous les plaçons dans la base de données recommender-db, dans les tables user_action et feedback.
Dans la table user_action, on stocke les clubs likés par l'utilisateur. Dans la table feedback, on stocke les retours de l'utilisateur vis à vis des articles. On peut ainsi savoir si un utilisateur n'est pas intéressé par des articles qui lui sont recommandés ou la note qu'il leur attribue.

2.5 - Schéma de base de données

Dans notre application en architecture micro-service, nous avons mis en place cinq bases de données :

- recommender-db
- user-db
- betting-db
- match-db
- wallet-db

Ces bases de données ne sont pas directement connectées entre elles, mais communiquent par le biais des différents services. Dans le diagramme d'entité-relation que nous mettons ci-dessous, nous avons fait le choix de mettre en relation des bases et tables qui ne sont en réalité pas connectées pour montrer quelles données chacunes fournit aux autres, indirectement. Ces relations symboliques sont en pointillé dans le diagramme.

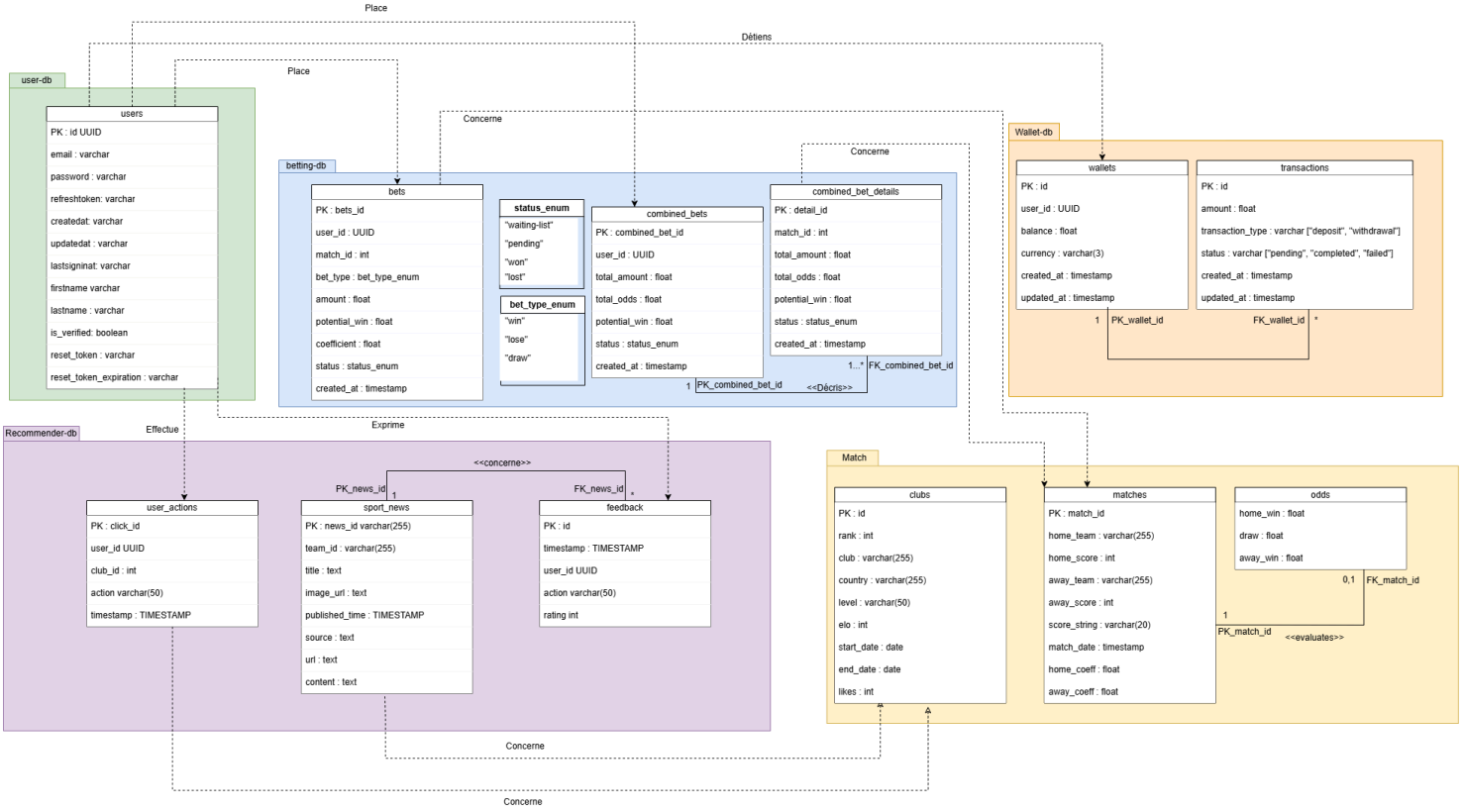


Figure 10 : Diagramme du modèle de données de TRD

3 - Quelques remarques

Nous avons fait le choix pour ce projet de ne pas travailler en temps réel : nos données datent de décembre 2024. Les articles présents dans le site ne reflètent pas l'actualité sportive. Les matchs sont tous déjà terminés.

L'âge des utilisateurs n’est pas vérifié, il est tout de même important de rappeler que les jeux d'argent sont interdits aux mineurs. En revanche, la vérification d’une adresse mail est obligatoire pour créer son compte.

4 - Synthèse des fonctionnalités effectuées parmi les fonctionnalités demandées

- ☑ front-end

---

1. Gestion des utilisateurs

- ☑ Authentification et sécurisation avec **JWT** (Token sécurisé).
- ☑ Inscription rapide sans vérification d’identité
- ☑ Gestion de la **cagnotte utilisateur** (argent déposé puis utilisé pour parier).
- ☑ **Historique des transactions financières** (dépôts, gains, retraits).
- ☑ **État de la cagnotte** visible pour l’utilisateur.
- ☑ **Blocage des paris** si la cagnotte est vide.
- ☑ **Récupération** de son mot de passe
- ☐ Différents niveaux d’accès : **Joueurs** et **Bookmakers** (rôles distincts).
- ☐ Obligation d’être **âgé de 18 ans** pour s’inscrire.

---

2. Système de paris

- ☑ **Paris simples** : Victoire, Match Nul, Défaite
- ☑ **Paris combinés** : Produit de plusieurs cotes
- ☑ Une fois un pari placé, l’utilisateur garde la **cote au moment du pari**, même si elle évolue.
- ☐ **Impact de la popularité des paris** sur la cote (prise en compte du nombre de paris).
- ☑ **Annulation** d’un pari
- ☐ **Annulation de pari possible seulement dans la première minute** après validation.
- ☑ **Mise** à définir.
- ☐ **Limitation des paris simultanés** : max **10 paris simples par minute** (optionnel).
- ☑ **Interdiction de parier 2 fois sur la même équipe ou le même match dans un combiné**.
- ☐ Gestion des **matchs annulés** :
    - ☐ Si définitivement annulé, suppression du match du pari combiné et recalcul de la cote.
    - ☐ Si reporté dans **moins de 48h**, il reste valide.

---

3. Gestion des cotes et gains

- ☐ **Fixation des cotes par les bookmakers**.
- ☑ **Affichage des cotes en temps réel** (simulation si besoin).
- ☑ Calcul des **gains potentiels** pour un pari en fonction de la mise et des cotes.
- ☑ Gestion des **paris gagnants/perdants** et versement des gains dans la cagnotte.
- ☑ **Système de bonus** pour les paris combinés

---

4. Gestion des matchs et des résultats

- ☑ **Récupération des informations des matchs via une API externe** (à trouver)
- ☑ **Affichage d'un agenda des matchs**.
- ☐ **Vérification de l'heure réelle du match** pour s'assurer que les paris soient fermés à temps.
- ☐ **Résultats affichés uniquement à la fin du match** (pas de paris en direct).

---

5. Notifications et suivi

- ☑ Vérifier son adresse mail
- ☑ Mot de passe oublié
- ☑ Changer son mot de passe
- ☐ **Notifications pour les joueurs** :
  - ☐ Confirmation de paris placés.
  - ☐ Résultats des paris gagnés/perdus.
  - ☐ Changements de statut des matchs (annulé, reporté).
- ☐ **Notifications pour les bookmakers** :
  - ☐ Informations sur l'état des paris et ajustement des cotes.
- ☐ **Historique des paris** (paris **gagnés, perdus, en cours, bloqués**)

---

6. Recommandations

- ☑ **Recommandation** d'articles traitant de **l'actualité sportive**

---

7. Sécurité et contraintes

- ☐ **Gestion sécurisée des transactions** et de l'identification avec **JWT**.
- ☐ **Distinction entre joueurs et bookmakers** avec permissions adaptées.
- ☐ **Protection contre les abus** :

- [ ] Limite de mises et de cotes à définir.
- [ ] Blocage des comptes frauduleux si nécessaire.

---

5 - Bilan

Ce projet nous a permis d'apprendre et de développer une application complète en microservices, en mettant en pratique de nombreuses compétences techniques et organisationnelles. Grâce à l'utilisation de Python pour le back-end et Streamlit pour l'interface utilisateur, nous avons pu concevoir une plateforme fonctionnelle et interactive.

Au-delà de l'aspect technique, ce projet nous a aussi donné l'opportunité de réimplémenter des notions de gestion de projet vues auparavant, notamment la méthode agile, nous permettant ainsi d'adapter notre travail en fonction des défis rencontrés et des retours du groupe.

Nous avons cependant dû faire face à plusieurs difficultés, notamment :

- Des problèmes avec Docker, qui ont heureusement été résolus grâce à un cours dispensé par des camarades, aidant ainsi une partie du groupe à mieux comprendre son fonctionnement.
- Des problèmes de performance sur certains ordinateurs, ralentissant parfois l'exécution des services et la fluidité du développement.

Même si toutes les fonctionnalités demandées par la société The Real Deal n'ont pas été complètement implémentées, et malgré ces défis, nous sommes très satisfaits du travail accompli. Ce projet nous a permis de gagner en expérience sur le développement en microservices, de renforcer nos compétences techniques et d'apprendre à surmonter des obstacles en équipe.

5 - Table des illustrations

6 - Annexe

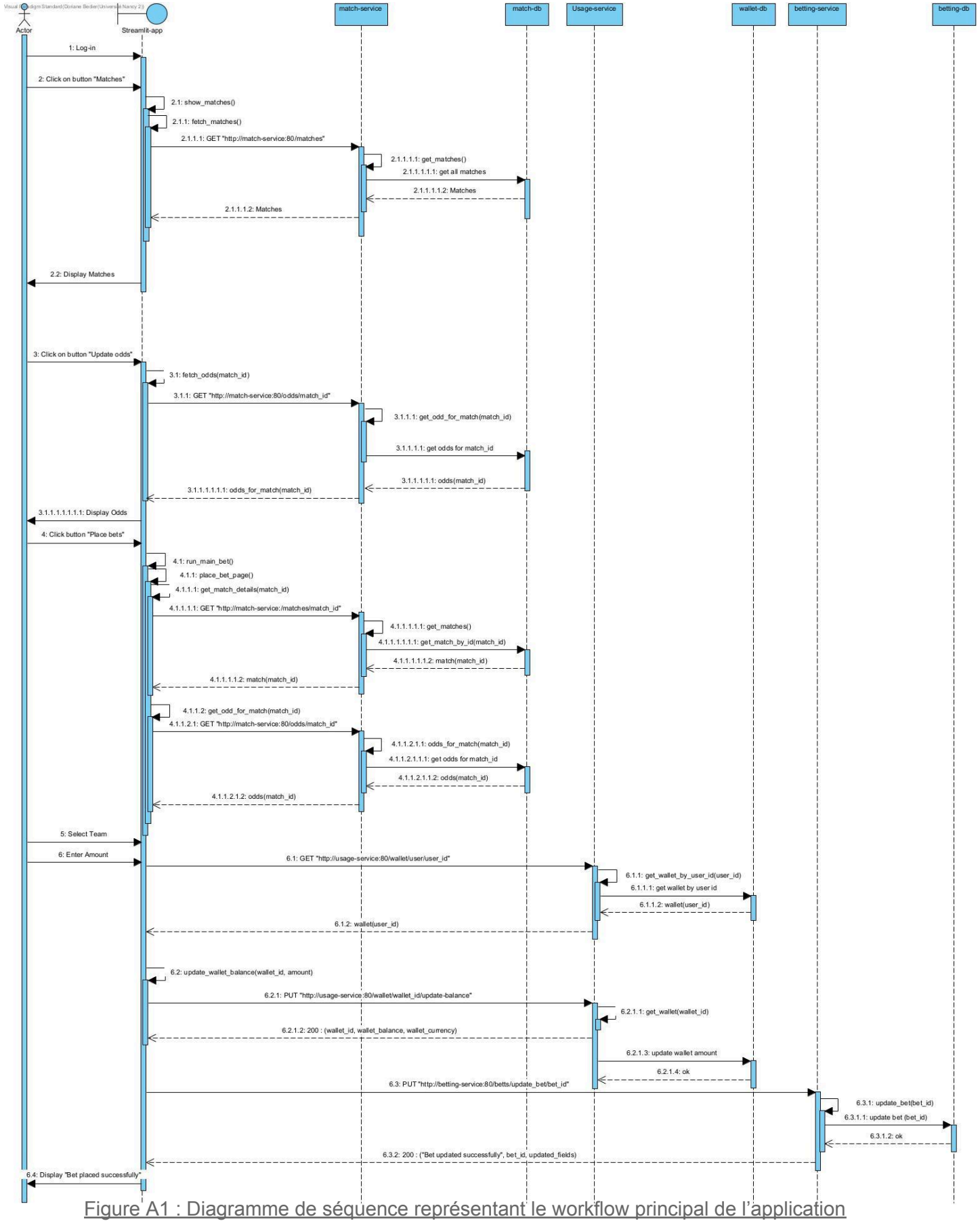


Figure A1 : Diagramme de séquence représentant le workflow principal de l'application

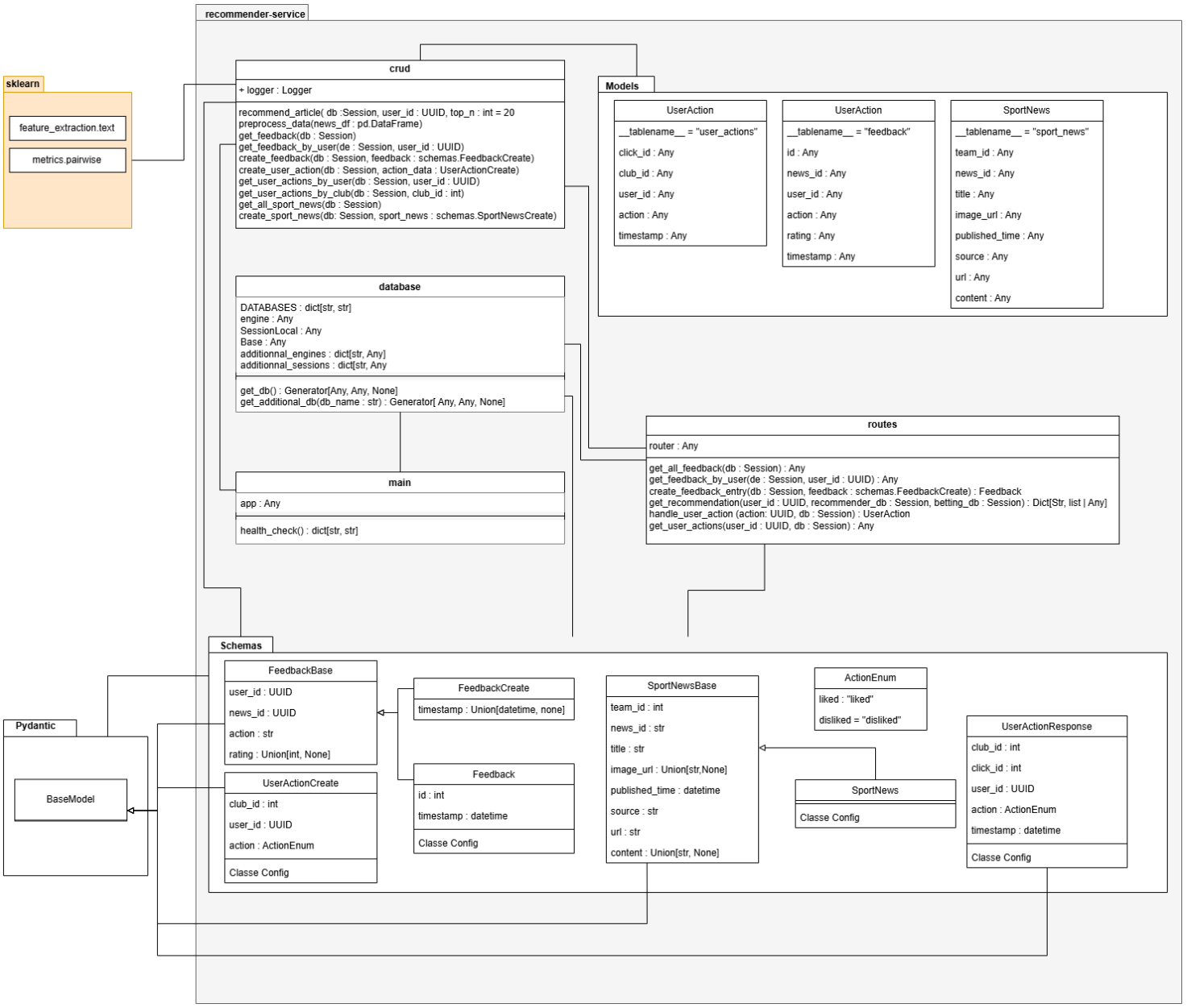


Figure A2 : Diagramme de classe de Recommender-service

Ce diagramme de classe d'un des services de notre application est un exemple concret de comment nous avons implémenté la plupart de nos services.

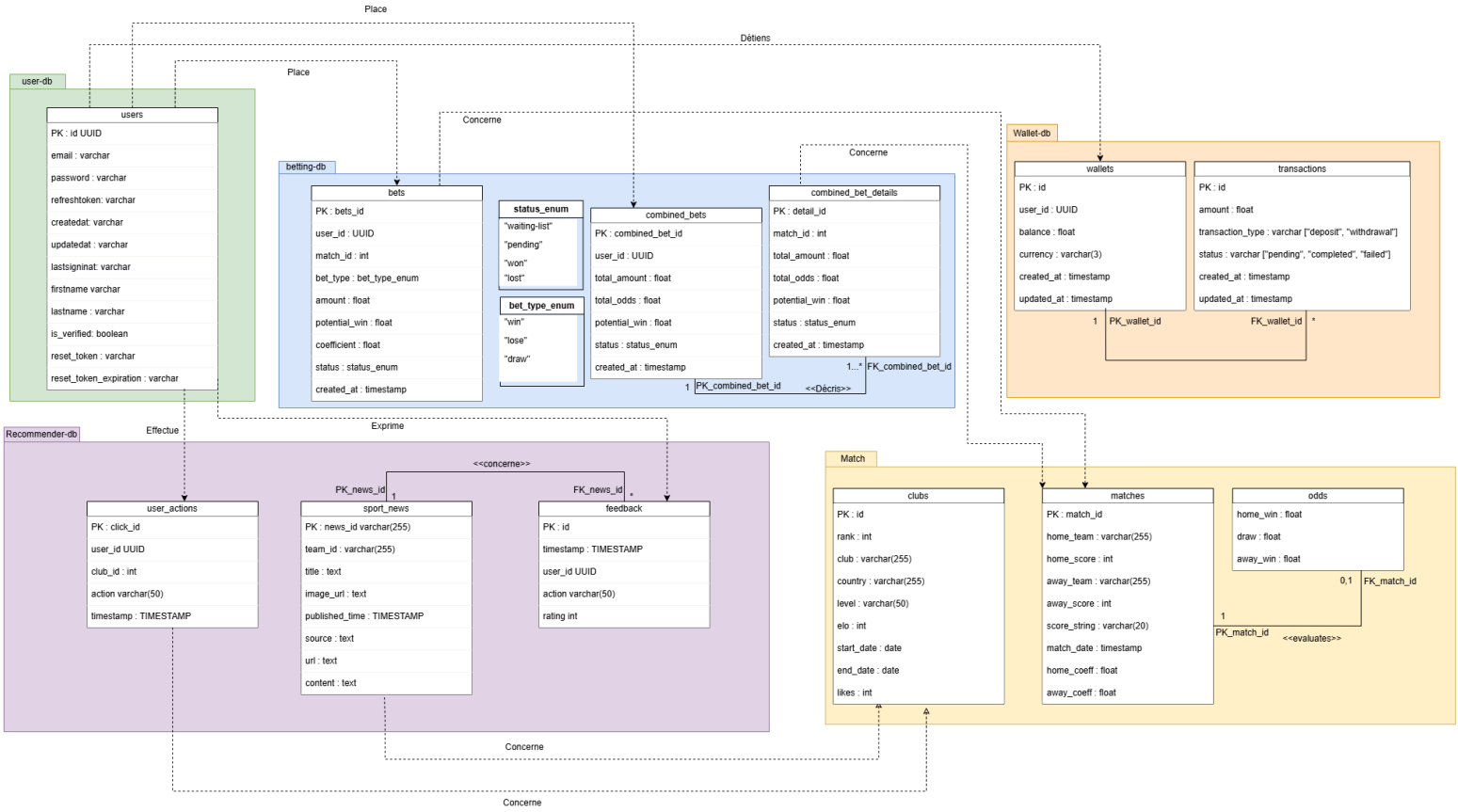


Figure A3 :schémas du modèle de données de TRD