AI23
Rapport projet A23

Petrel Lucas
Zizouni Maher
Mougoue Ruben
Anière Quentin

25 décembre 2023

Table des matières

1 Présentation du projet

1.1 Contexte

Ce projet a été réalisé dans le cadre de l'UV AI23 / NF16 à l'UTC. Il consiste en la conception et la réalisation d'une base de données pour une bibliothèque. Les technologies utilisées sont le langage SQL, le SGBD PostgreSQL et le langage Python pour l'application.

1.2 Organisation du projet

Le projet par 4 personnes et a été divisé en 5 jalons. Chaque jalon est déposé sur le dépôt Gitlab de l'UTC.

1.3 Note de clarification et première version du MCD

Le premier jalon a consisté à clarifier le sujet et à réaliser une première version du MCD. Cette version a été réalisée en groupe et a été déposée sur le dépôt Gitlab. Nous avons utilisé *plantuml* pour le MCD, il a l'avantage d'être au format texte et donc facilement modifiable par plusieurs personnes avec un outil de versionning comme Git.
La première version du MCD est disponible en annexe.

1.4 Correction du MCD et MLD

Suite aux commentaires reçus sur le premier MCD, nous avons réalisé une seconde version du MCD. Les corrections concernaient principalement sur un problème de de relation (composition au lieu d'agrégation) ainsi que le rajout de la date de retour d'un prêt.
Nous avons ensuite transformé le MCD en MLD. Nous avons rencontré quelques difficultés au niveau du choix du type d'héritage pour les tables, nous avons finalement choisi l'héritage par classe mère.
La deuxième version du MCD est disponible en annexe.

1.5 Code SQL

1.5.1 Création des tables

Nous avons converti le MLD en code SQL pour créer les tables. Nous avons pas rencontré de difficultés particulières pour cette partie, appart le fait que nous avons du créer les tables dans un ordre précis pour éviter les problèmes de dépendances.

1.5.2 Création des vues

Nous avons créé les vues en nous basant sur les requêtes SQL demandées dans le sujet. Nous avons pas rencontré de difficultés particulières pour cette partie.
Les vues permettent de simplifier l'utilisation de la base de données pour l'application Python et d'améliorer les performances.

1.5.3 Insertion des données

Nous avons inséré les données dans les tables. Nous avons pas rencontré de difficultés particulières pour cette partie si ce n'est que nous avons du insérer les données dans un ordre précis pour éviter les problèmes de dépendances (comme lors de la création des tables).

1.6 Application Python

Nous avons réalisé une application Python pour utiliser la base de données. Nous avons utilisé la librarie *psycopg2* pour communiquer avec la base de données. Nous avons choisi le format web pour notre application, avec la librairie *flask*. Cette application est documentée dans la prochaine section.

2 Documentation de l'application

2.1 Archictecture

Nous avons choisi de faire une application web pour utiliser la base de données car c'est le format le plus simple à utiliser et le plus pratique pour ce type d'application. Nous avons utilisé la librairie *flask* pour le serveur web (backend) et *psycopg2* pour communiquer avec la base de données.

Pour la partie interface, c'est tout simplement du HTML/CSS/JS. Nous avons utilisé *tailwindcss* pour du style CSS déjà prêt. Les deux communiquent avec une API REST. Nous avons utilisé l'outil *swagger* pour documenter l'API.

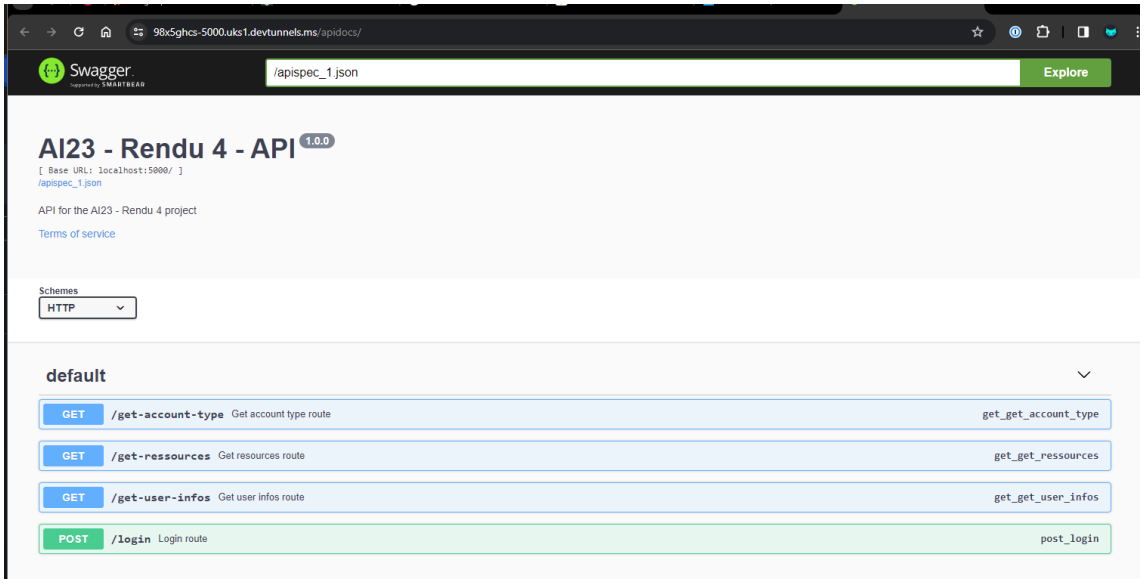

FIGURE 1 – Documentation de l'API

2.2 Partie backend

Le backend est composé de 2 fichiers :

- *app.py* : le fichier principal qui contient le serveur web
- *database.py* : le fichier qui contient les fonctions pour communiquer avec la base de données

2.2.1 app.py

Ce fichier contient le serveur web. Il associe une route (URL) à une fonction. Cette fonction va communiquer avec la base de données et renvoyer le résultat au format JSON.

Nous avons utilisé la librairie *flask* pour le serveur web. Nous avons utilisé la librairie *psycopg2* pour communiquer avec la base de données. Nous avons utilisé la librairie *flask-cors* pour permettre à l'application web d'accéder à l'API.

Nous avons utilisé la librairie *flasgger* pour documenter l'API. Cette librairie permet de générer une documentation au format *swagger* à partir des commentaires dans le code.

Nous avons utilisé la librairie *python-dotenv* pour charger les variables d'environnement depuis un fichier *.env*.

2.2.2 database.py

Ce fichier contient les fonctions pour communiquer avec la base de données. Nous avons utilisé la librairie *psycopg2* pour communiquer avec la base de données.

Nous avons créé une fonction pour chaque requête SQL. L'intêret d'avoir un ficher dédié est de pouvoir changer d'interface facilement (par exemple passer d'une interface web à une interface en ligne de commande).

2.3 Partie frontend

Le frontend est composé de 2 fichiers HTML :

- *login.html* : la page qui permet de se connecter à l'application
- *dashboard.html* : la page d'accueil de l'application

Nous avons également trois fichiers HTML, des vues qui sont intégrés dans la page *dashboard.html* avec un *iframe* :

- *catalog.html* : la page qui affiche toute les ressources de la bibliothèque et qui permet d'emprunter.
- *adherents.html* : la page qui permet de gérer les membres

- *creer_adh.html* : la page qui permet de créer un nouveau membre.

Nous avons utilisé la librairie *tailwindcss* pour le style CSS. Cela permet de gagner du temps avec des classes CSS déjà prêtes. La communication avec le backend se fait avec l'API REST. Nous avons utilisé la fonction standard *fetch* pour communiquer avec l'API.

L'authefication se fait avec un token JWT (JSON Web Token). Ce token est généré lors de la connexion et est stocké dans le local storage du navigateur. Il est ensuite envoyé dans le header de chaque requête. Cela permet de vérifier que l'utilisateur est bien connecté a chaque requête.

Des captures d'écran de l'application sont disponibles en annexe.

3 Annexes

3.1 MCD (Version 1)

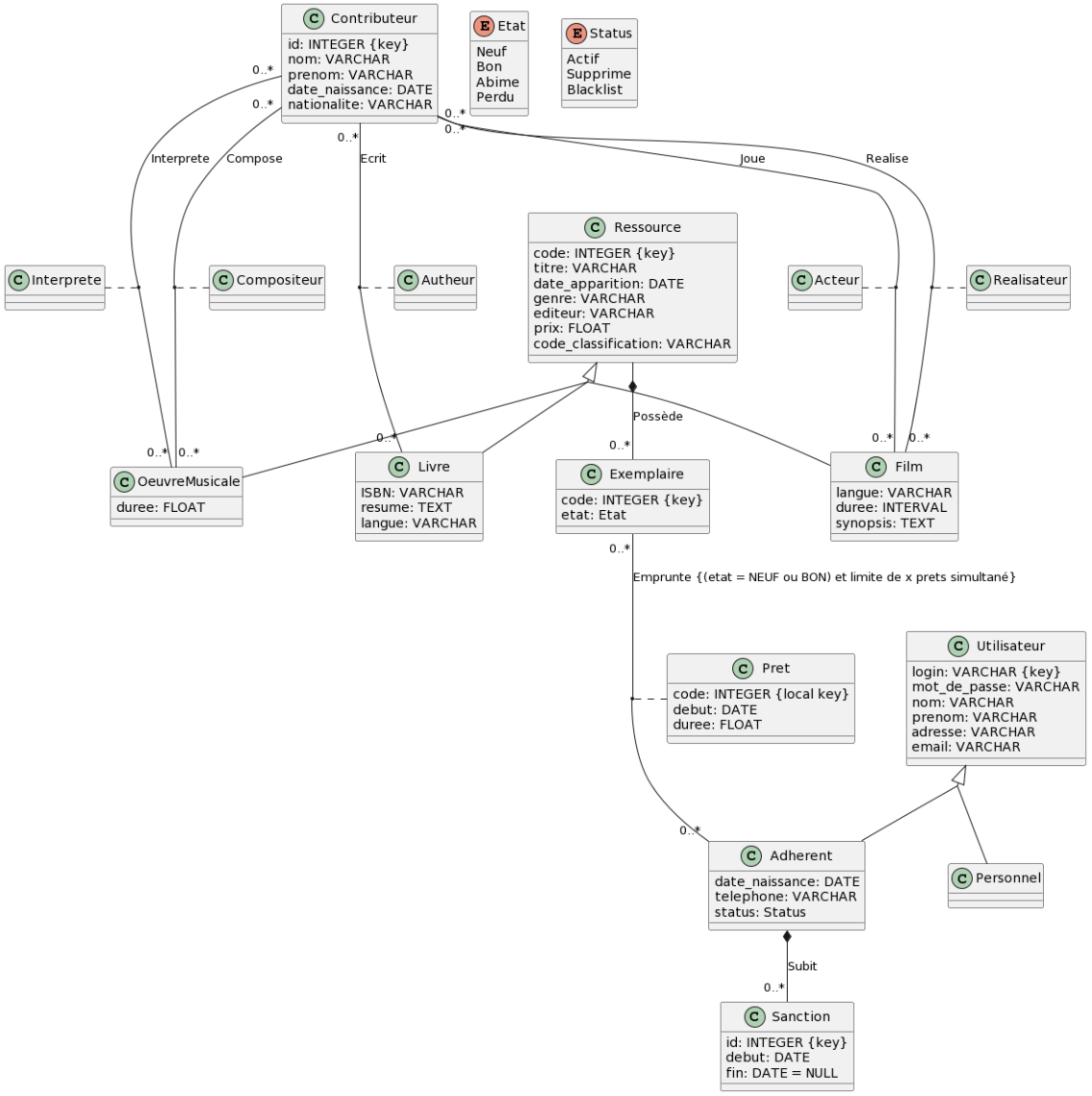

Figure 2 – Première version du MCD

3.2 MCD (Version 2)

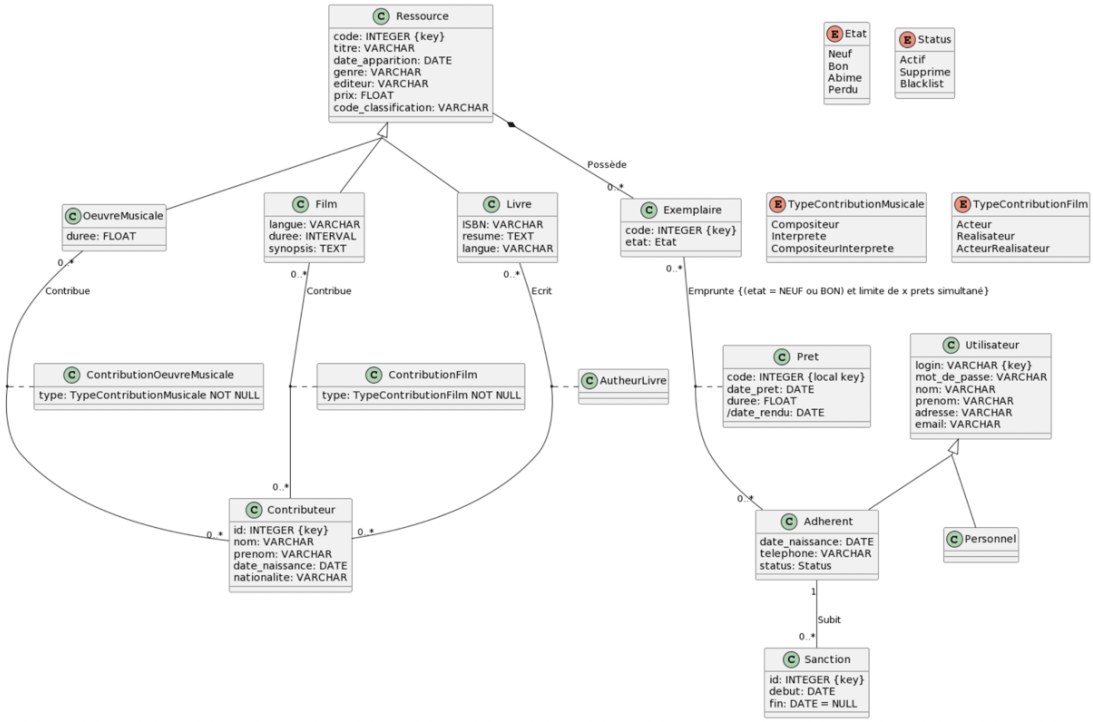

FIGURE 3 – Deuxième version du MCD

3.3 Captures d'écran de l'application

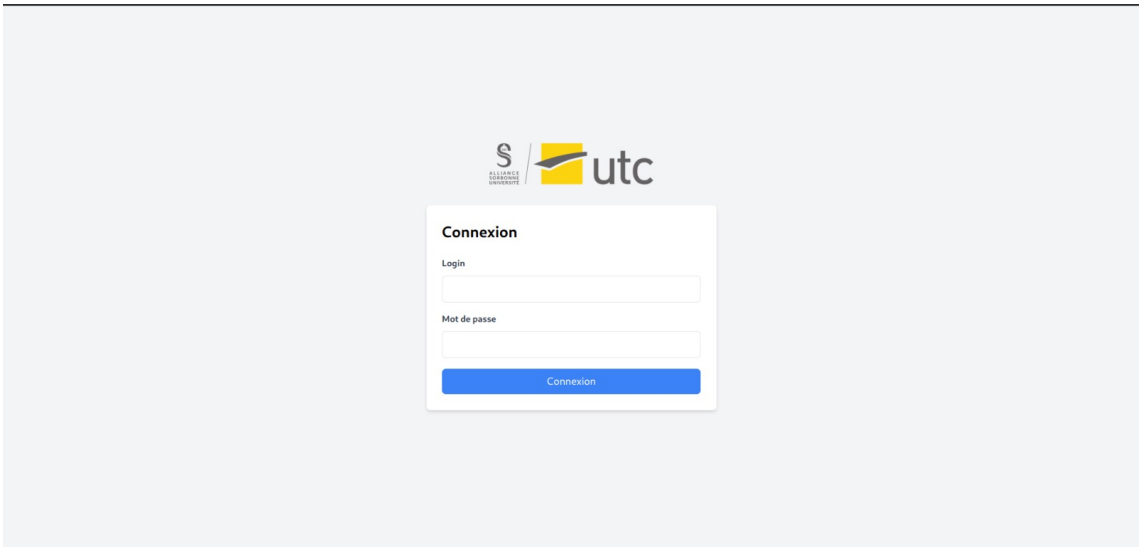

FIGURE 4 – Page de connexion

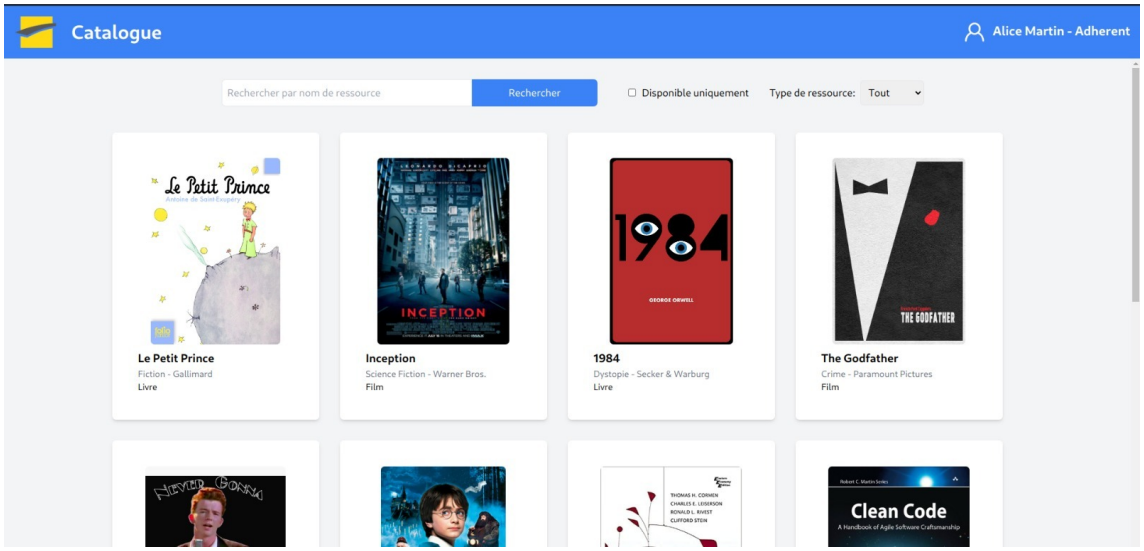

FIGURE 5 – Page d'accueil

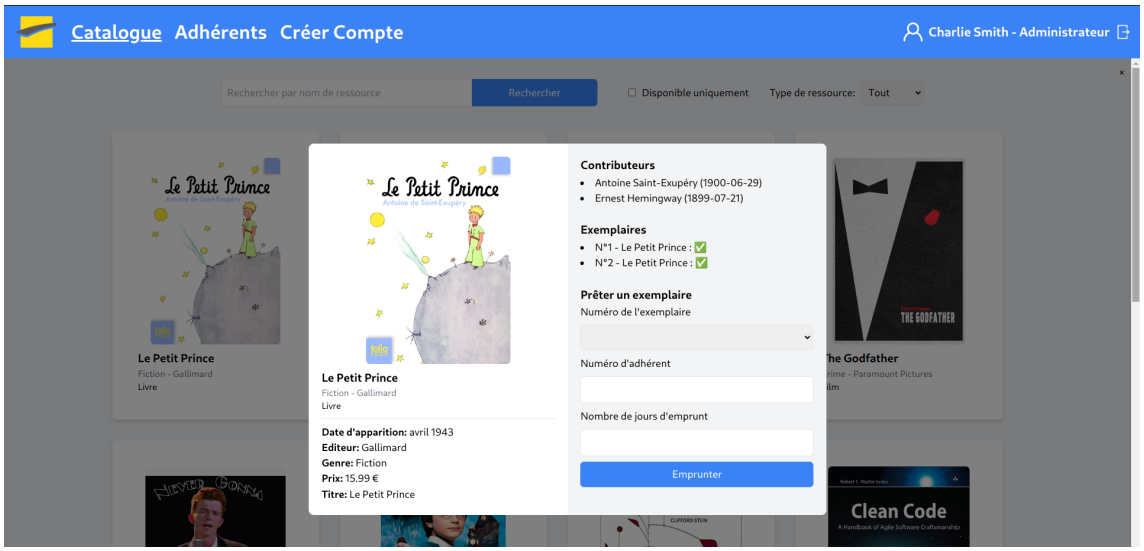

FIGURE 6 – Page d'emprunt