

MOVIES NOTES



Filipe Ramos, Sarah Tat
Projet Transversal I - Printemps 2024

Introduction.....	2
Description du projet et sa problématique.....	2
Analyse des objectifs.....	3
1. Modèle des objectifs.....	3
2. Modèle et liste des exigences.....	4
3. Use Case.....	5
Conception.....	5
1. Schéma conceptuel (entités-relation) de la BD.....	5
2. Schéma logique de la BD.....	6
Modèle de l'algorithme.....	7
Données de test.....	8
Identification.....	8
Récupération.....	8
Implémentation client/serveur.....	9
Démonstration.....	10
Outils utilisés.....	11
Conclusion.....	11

Introduction

Dans le cadre du cours Projet Transversal I, nous avons choisi de développer un site web pour les amoureux de cinéma, inspiré par IMDb. Notre projet vise à implémenter un site permettant non seulement de référencer une multitude de films de divers genres et époques, mais également d'offrir aux utilisateurs la possibilité de les noter et de les commenter. Ce projet a pour objectif de créer une ressource exhaustive et interactive qui facilite l'échange d'opinions, enrichit la culture cinématographique des utilisateurs et constitue une référence pour le choix de films à découvrir.

Notre but est de bâtir une communauté de passionnés de cinéma, où chaque membre peut contribuer activement au contenu et à l'évaluation des films. Ce rapport détaillera les phases de conception, de développement et de mise en œuvre du site.

Description du projet et sa problématique

Le projet MOVIES NOTES vise à offrir aux utilisateurs la possibilité de rechercher, noter et commenter une large sélection de films.

L'objectif principal est de créer une base de données exhaustive et interactive de films, permettant aux utilisateurs de se créer des listes de ces derniers. Par exemple, un utilisateur pourrait se créer une liste de films à voir, une autre de ses films préférés avec les notes données et éventuellement des commentaires.

La problématique centrale de ce projet réside dans la conception et la mise en œuvre d'une solution technologique capable de soutenir une interaction riche et dynamique entre les utilisateurs et la base de données de films. Il s'agit de permettre aux utilisateurs de facilement trouver des informations sur les films, de les noter et de les commenter, tout en garantissant une expérience utilisateur fluide et engageante. Le défi inclut également la gestion de la sécurité des données des utilisateurs, la performance du système face à une charge élevée d'interactions et surtout une base de données conséquente en volume.

Analyse des objectifs

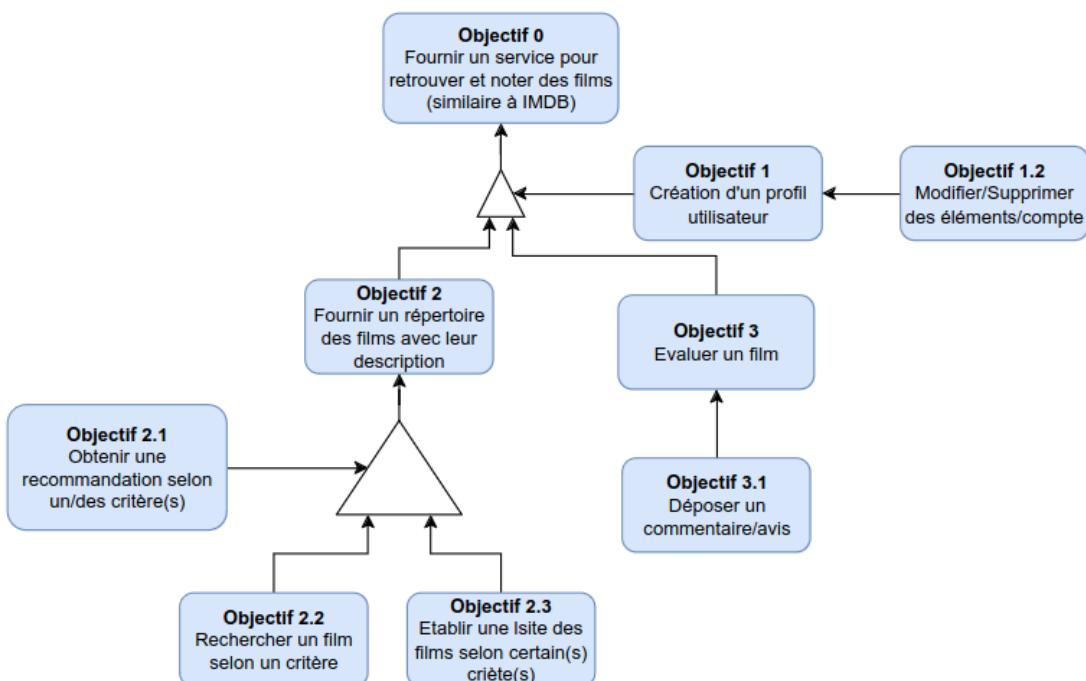
1. Modèle des objectifs

Pour ce projet, nous avons analysé divers objectifs. Tout d'abord, il est important que le service puisse permettre aux utilisateurs de retrouver et noter des films de manière similaire au service IMDB. Ils pourront ainsi rechercher un film spécifique facilement selon certaines données (titre, catégorie, réalisateur, etc.). Ils pourront également déposer un avis et noter sur une échelle de 5 étoiles leur appréciation.

Puis, il est fondamental que les films soient correctement répertoriés avec leur description (titre, acteur, réalisateur, année, catégorie, etc.). Les utilisateurs pourront ainsi retrouver leur film favoris en quelques clics. Ils pourront également obtenir une liste de recommandation selon certains critères.

Finalement, il est également essentiel de pouvoir créer une liste des films vus ou à voir, et de les rendre public afin de les partager avec d'autres utilisateurs qui pourraient avoir les mêmes intérêts et goûts. Cela permettrait à différents utilisateurs de connaître de nouveaux films et de partager leur engouement pour une certaine catégorie de films.

Voici donc le modèle des objectifs :



2. Modèle et liste des exigences

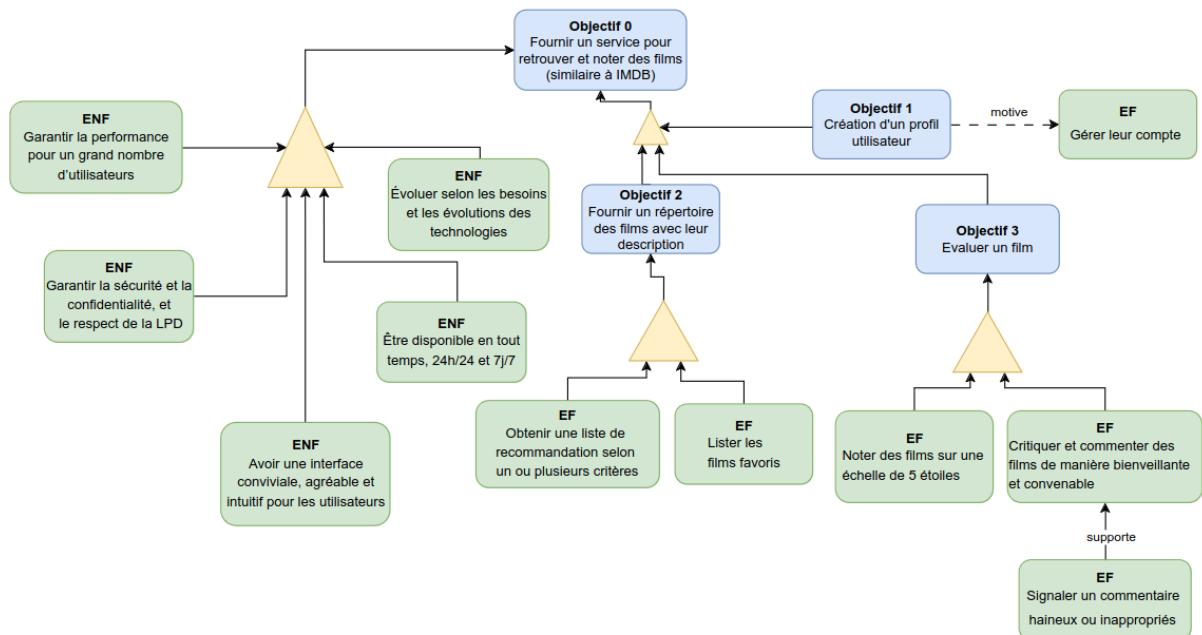
En ce qui concerne les besoins fonctionnels, le service doit ainsi permettre aux utilisateurs de :

- Noter des films sur une échelle de 5 étoiles
 - Critiquer et commenter des films de manière bienveillante et convenable
 - Lister leurs films favoris
 - Obtenir une liste de recommandation selon un ou plusieurs critères
 - Gérer leur compte
 - Signaler un commentaire haineux ou inapproprié

Pour les besoins non-fonctionnels, le service doit être capable de :

- Garantir la performance pour un grand nombre d'utilisateurs
 - Garantir la sécurité et la confidentialité, et le respect de la LPD
 - Avoir une interface conviviale, agréable et intuitif pour les utilisateurs
 - Être disponible en tout temps, 24h/24 et 7j/7
 - Évoluer selon les besoins et les évolutions des technologies

Voici le modèle des exigences :



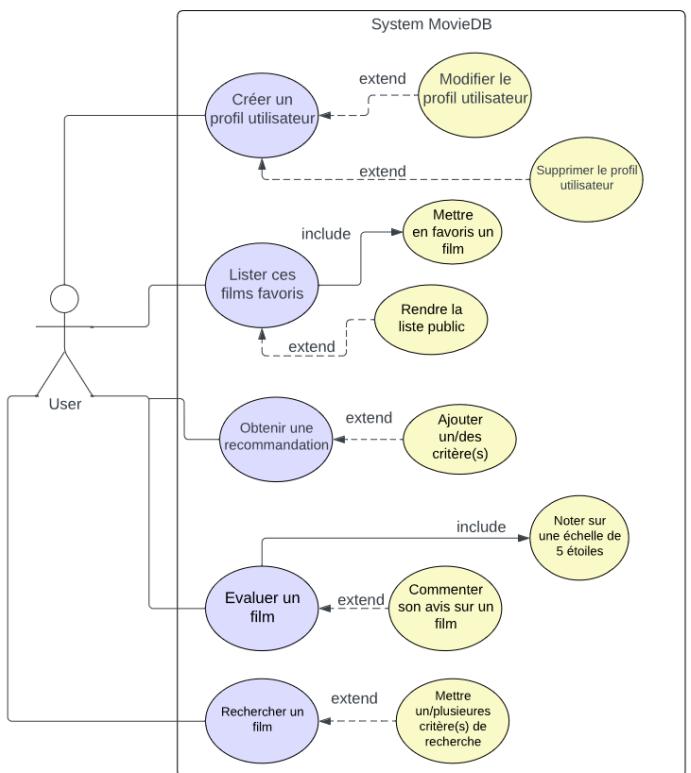
3. Use Case

Le diagramme des cas d'utilisation nous permet de visualiser les différentes actions qu'un utilisateur peut réaliser sur notre site.

Pour notre projet, chaque utilisateur a donc accès à quatre fonctionnalités principales :

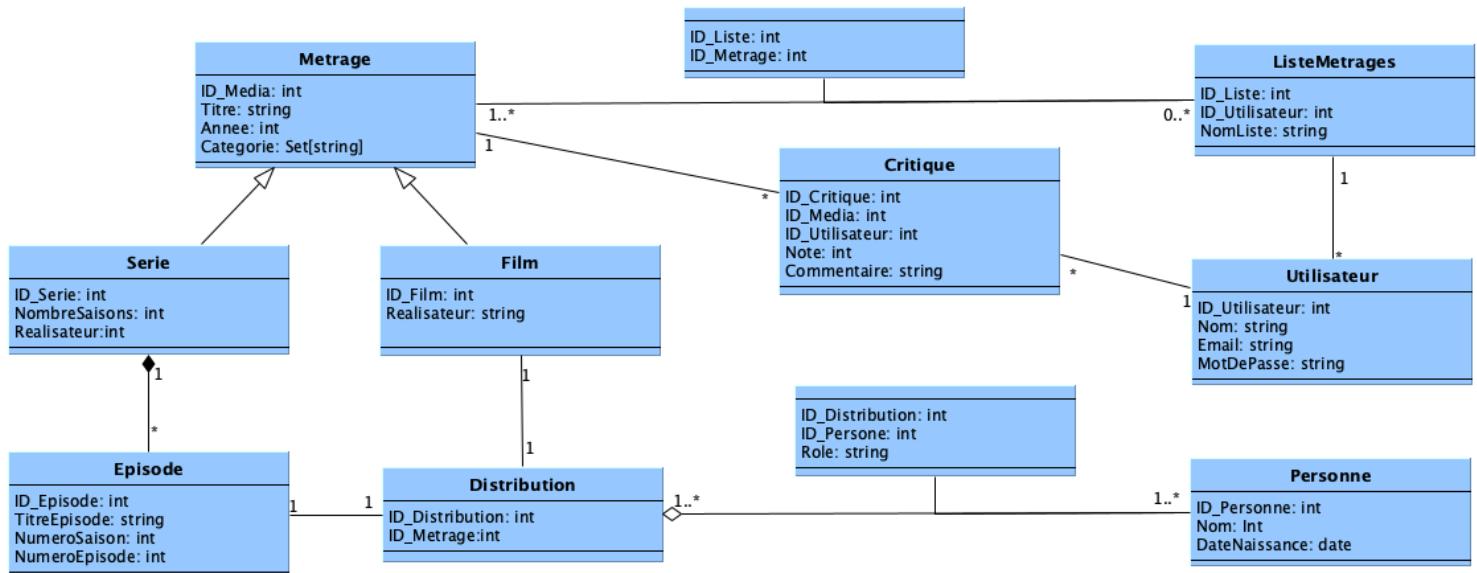
1. Créer un profil utilisateur
2. Lister ces films favoris
3. Obtenir une recommandation
4. Evaluer un film
5. Rechercher un film

Chaque fonctionnalité contient d'autres sous-fonctionnalités qui sont soit obligatoire pour réaliser la fonctionnalité de base, soit optionnelle et dépend donc du but de l'utilisateur.

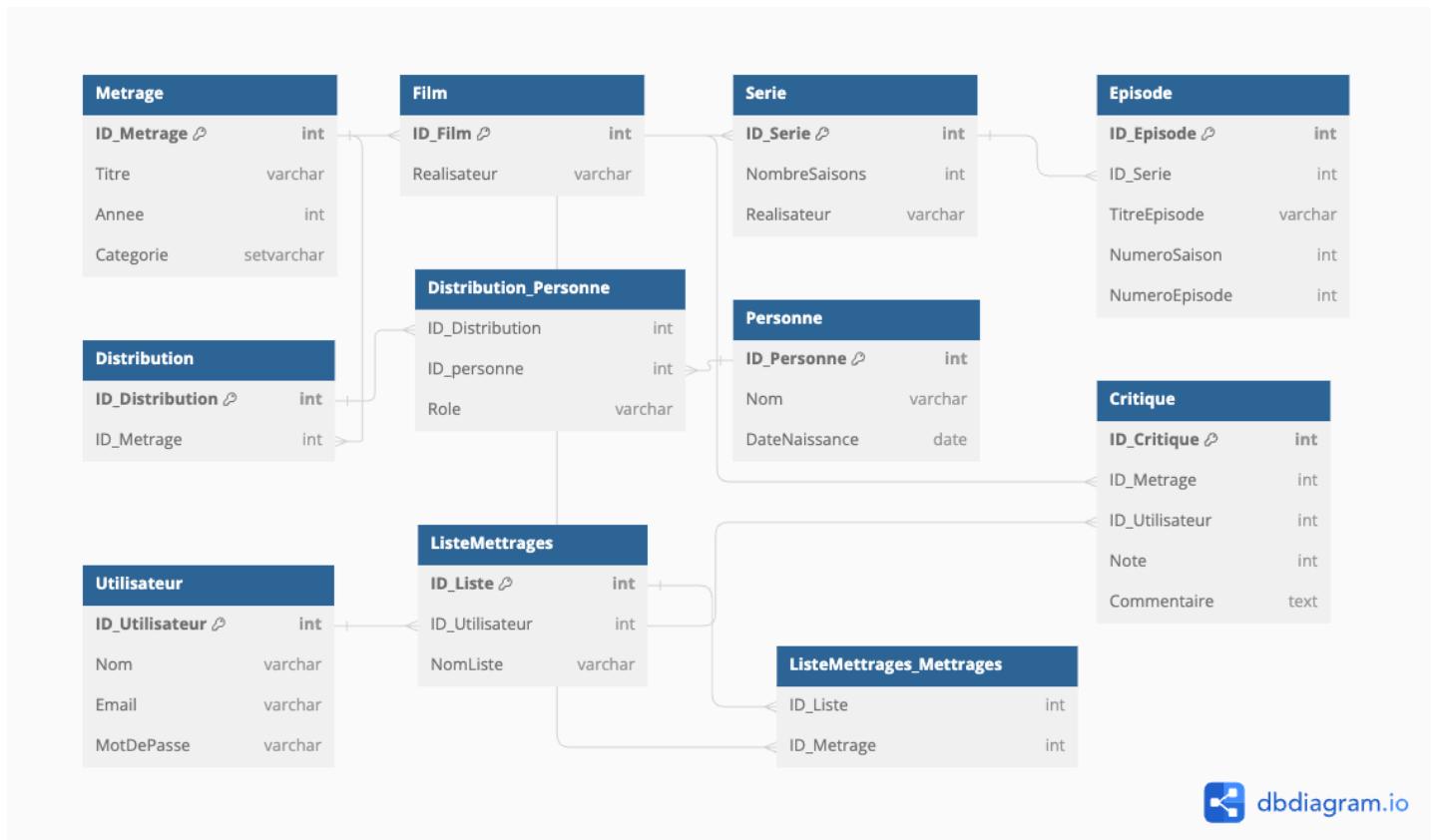


Conception

1. Schéma conceptuel (entités-relation) de la BD



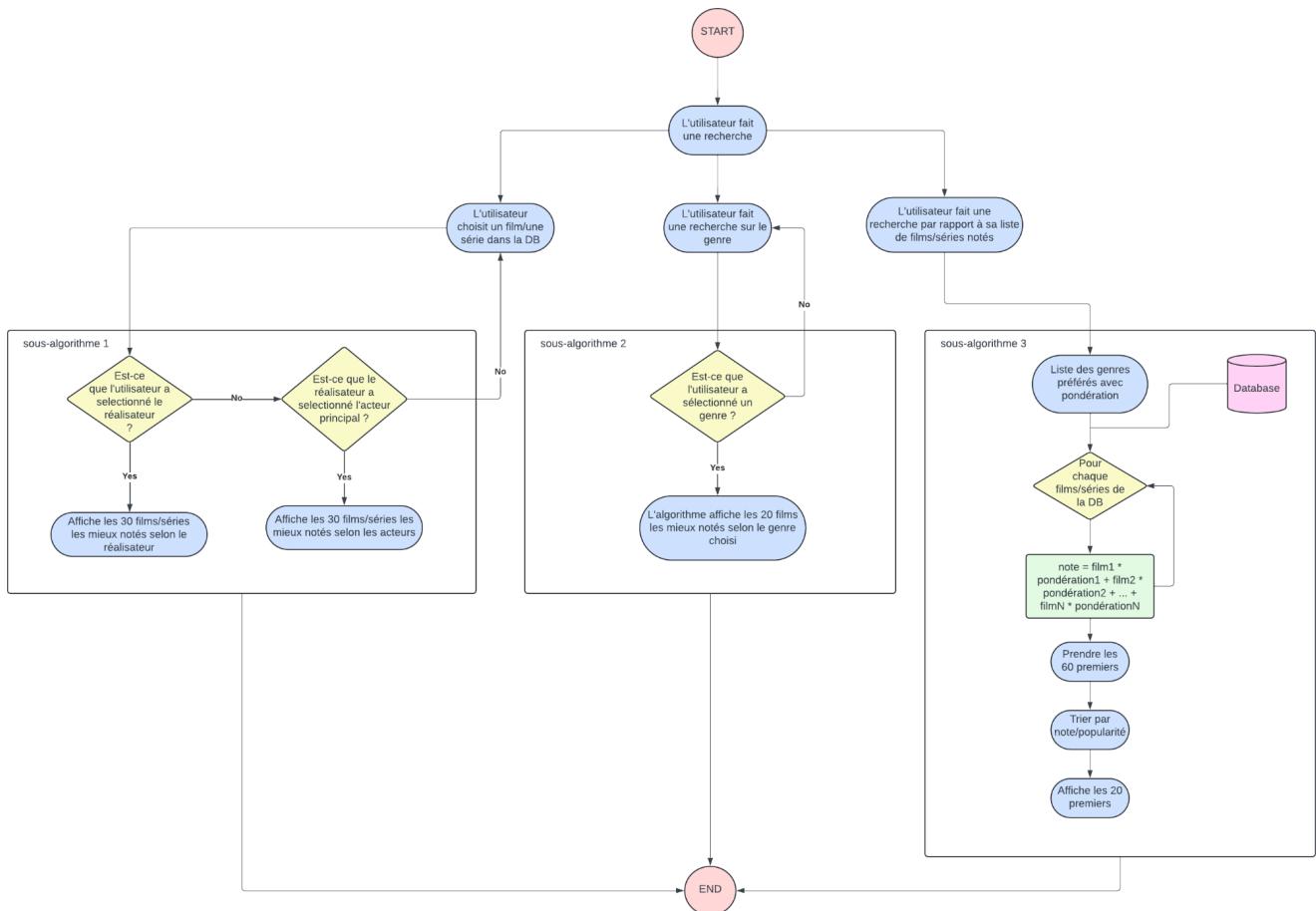
2. Schéma logique de la BD



Modèle de l'algorithme

En ce qui concerne le modèle de l'algorithme, nous avons choisi de diviser l'algorithme en trois sous-algorithmes. Le premier concerne le choix d'un film ou d'une série de l'utilisateur. L'utilisateur peut ensuite choisir s'il souhaite une liste de recommandation en fonction du réalisateur ou de l'acteur principal du film/série choisi(e). Pour le second, la liste de recommandation part sur le genre choisi par l'utilisateur. Ces deux algorithmes n'impliquent pas une complexité particulière, c'est pourquoi il est possible que ces deux algorithmes évoluent durant l'avancée du projet.

Pour le dernier, elle sera calculée en fonction de la liste des films notés/vus par l'utilisateur. Une pondération selon le genre sera ajoutée afin de calculer et de trouver les meilleurs films à recommandés. Nous avons implémenté un exemple de cet sous-algorithme - *model.py*. Toutefois, elle n'est ni liée à la base de données, ni à la liste des utilisateurs. Nous avons simplement mis une liste quelconque de genre et une pondération aléatoire afin d'avoir une idée plus concrète.



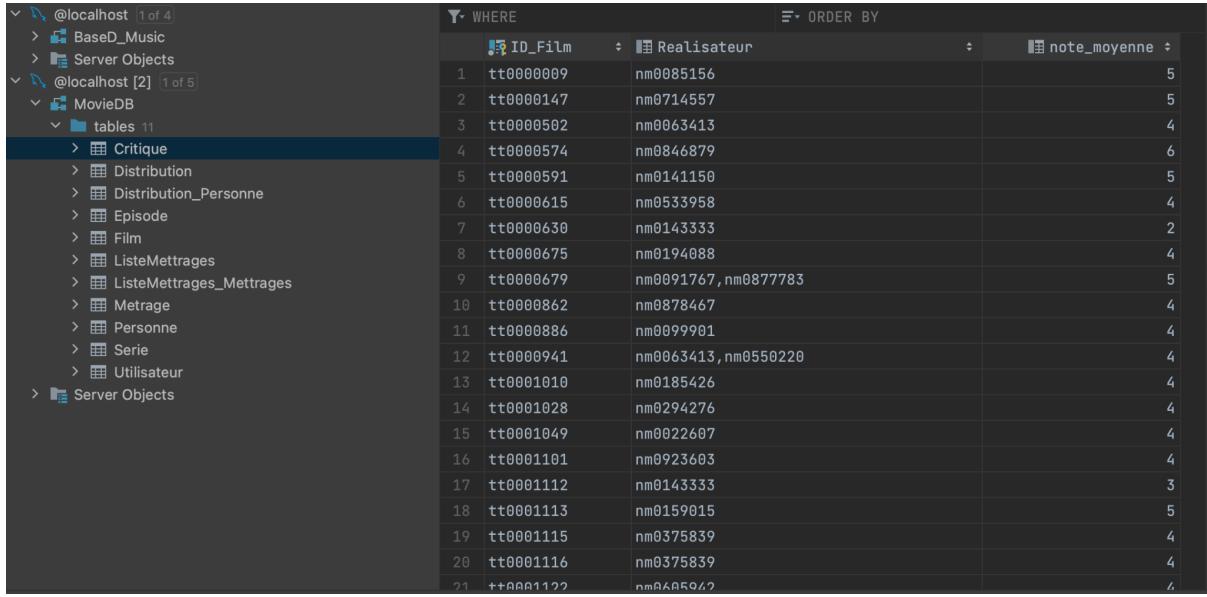
Données de test

Identification

Pour tester notre application, nous pensons utiliser la base de données non commerciale de iMDb: <https://developer.imdb.com/non-commercial-datasets/> que nous devrons nettoyer et adapter à notre système.

Récupération

Afin de pouvoir travailler avec ces données nous avons dû faire un premier tri pour réduire leur taille. En effet, une fois décompressés, les fichiers tsv (tab separated values) de IMdb faisaient environ 6Go et il était impossible pour nous de travailler avec nos machines sur une telle masse de données. Nous avons donc dû travailler directement sur les données en tsv avant de pouvoir les importer dans une base sql. Pour ce faire, nous avons utilisé la bibliothèque Pandas (github.com/pandas-dev/pandas) une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Nous avons d'abord fait un premier script de filtrage pour supprimer une partie des données inutiles puis un second de "mapping" pour que ces dernières correspondent à notre structure. Au vu de la masse de données, ce travail de récupération a été assez long car il a fallu, non seulement se familiariser avec pandas mais aussi apprendre la patience car chaque essai était chronophage et une petite erreur faisait que l'on se retrouvait avec des films très peu connus et très anciens. Nous avons même commencé à regarder du côté de la bibliothèque beautifulsoup4 pour faire du scrapping sur le site senscritique.com mais cette erreur étant corrigée, nous devrions pouvoir travailler avec ce dataframe. Pour les utilisateurs nous avons généré une liste de 500 utilisateurs avec <https://generatedata.com/generator> afin de remplir notre base de données. Enfin, le prochain défi sera de créer des listes de favoris cohérents pour pouvoir tester l'algorithme de proposition car si nous créons des listes de favoris hasardeuses, il n'y aura aucun moyen de vérifier que les goûts correspondent bien.



	ID_Film	Realisateur	note_moyenne
1	tt000009	nm0085156	5
2	tt000047	nm0714557	5
3	tt0000502	nm0063413	4
4	tt0000574	nm0846879	6
5	tt0000591	nm0141150	5
6	tt0000615	nm0533958	4
7	tt0000630	nm0143333	2
8	tt0000675	nm0194088	4
9	tt0000679	nm0091767, nm0877783	5
10	tt0000862	nm0878467	4
11	tt0000886	nm0099901	4
12	tt0000941	nm0063413, nm0550220	4
13	tt0001010	nm0185426	4
14	tt0001028	nm0294276	4
15	tt0001049	nm0022607	4
16	tt0001101	nm0923603	4
17	tt0001112	nm0143333	3
18	tt0001113	nm0159015	5
19	tt0001115	nm0375839	4
20	tt0001116	nm0375839	4
21	tt0001122	nm0605942	4

Implémentation client/serveur

Pour commencer l'implémentation du client/serveur, nous avons commencé par un script simple en python. Nous avons utilisé le module socket qui permet au client de se connecter à un serveur. Le module sys est utilisé dans le script du client afin d'accéder aux arguments donnés en entrée. Afin de pouvoir gérer plusieurs clients (utilisateurs), nous devons ajouter une boucle. Cette partie de l'implémentation n'a toutefois pas encore commencé et ne serait implémenté que dans le cas où nous utiliserions une grande base de données hébergée sur un espace de stockage externe. Il est possible que selon l'avancée du projet, nous décidions d'opter plutôt pour une structure conteneurisée, avec un conteneur docker pour notre base de données et un second pour notre application flask. Nous les ferons ensuite communiquer entre eux avec docker-compose localement.

```
client.py x ...  
g4 > client.py > ...  
1 import socket  
2 import sys  
3  
4 def start_client(host='127.0.0.1', port=12345, message="Hello!"): # Création d'un socket  
5     client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
6  
7     # Connexion au serveur  
8     try:  
9         client_socket.connect((host, port))  
10        print(f"Connected to server at {host}:{port}")  
11  
12        # Envoyer un message au serveur  
13        client_socket.sendall(message.encode())  
14    except Exception as e:  
15        print(f"Failed to connect to server: {e}")  
16    finally:  
17        # Fermer le socket  
18        client_socket.close()  
19        print("Connection closed")  
20  
21 if __name__ == "__main__":  
22     if len(sys.argv) < 3:  
23         print("Usage: python3 client.py [host] [port]")  
24     else:  
25         host = sys.argv[1]  
26         port = int(sys.argv[2])  
27         start_client(host, port)  
28  
29
```

```
serveur.py x ...  
g4 > serveur.py > ...  
1 import socket  
2  
3 def start_server(host='127.0.0.1'): # création socket  
4     server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
5  
6     # lier du socket à une adresse IP et un port dynamique  
7     server_socket.bind((host, 0)) # 0 signifie "choisir un port disponible"  
8  
9     # Obtenir le port assigné  
10    port = server_socket.getsockname()[1]  
11  
12    # Le serveur écoute les connexions entrantes  
13    server_socket.listen()  
14    print(f"Server listening on {host}:{port}")  
15  
16    # Accepter une connexion entrante  
17    client_socket, client_address = server_socket.accept()  
18    print(f"Connected to {client_address}")  
19  
20    # Recevoir des données du client  
21    message = client_socket.recv(1024).decode()  
22    print(f"Received from client: {message}")  
23  
24    # Fermer la connexion  
25    client_socket.close()  
26    server_socket.close()  
27    print("Server closed")  
28  
29 if __name__ == "__main__":  
30     start_server()  
31  
32
```

Démonstration

En ce qui concerne la démonstration, nous pouvons simplement vous présenter le concept de l'algorithme que nous allons proposer. Voici donc le résultat pour ce concept :

Ce résultat liste les films selon les genres préférés d'un utilisateur. Il serait intéressant d'ajouter dans l'algorithme une sélection de genre minimum afin de bien pouvoir recommander l'utilisateur.

```
● sarah@sarah:~/Desktop/mdb/g4$ python3 model.py
Interstellar ['Sci-Fi', 'Adventure', 'Drama'] 80
star wars ['Sci-Fi', 'Action', 'Adventure'] 75
Inception ['Sci-Fi', 'Thriller'] 60
LOTR ['Fantasy', 'Action', 'Adventure'] 55
Harry Potter ['Fantasy', 'Adventure'] 50
Matrix ['Sci-Fi', 'Action'] 45
Avengers ['Action', 'Adventure'] 35
○ sarah@sarah:~/Desktop/mdb/g4$ █
```

Outils utilisés

En ce qui concerne les outils utilisés pour ce début de projet, nous avons utilisé le site Lucidchart pour faire le schéma de l'use case, le site draw.io pour le modèle des objectifs, dbdiagram.io pour le schéma logique et yEd pour le schéma conceptuel. Nous avons également utilisé le site <https://datasets.imdbws.com/> pour tout ce qui concerne la base de données.

En termes de communication, nous utilisons principalement whatsapp et telegram. Nous allons également utiliser Gitlab pour déposer notre travail au fur et à mesure de l'avancée du projet et finalement nous essayerons de containeriser notre travail avec Docker.

Conclusion

Le premier checkpoint était une étape essentielle pour la compréhension et le développement de notre projet : la création d'un site web de référence pour les films, inspiré du site IMDb. Nous avons défini dans ce rapport les bases nécessaires pour le développement du projet ainsi que la problématique imaginée de prime abord.

Pour ce second checkpoint, nous nous sommes concentrés sur la base de données et le modèle de l'algorithme. Ce dernier est composé de trois sous-algorithmes. Nous avons commencé par implémenter un concept pour un de ces trois sous-algorithmes. Cependant, il a été très difficile pour nous d'imaginer un tel algorithme sans savoir quelles données allaient pouvoir être exploitées et voulant peut-être trop en faire, nous avons perdu un temps précieux avec la base de données qui n'est toujours pas complète. Nous travaillons dessus et avons bon espoir de terminer dans les délais.

En conclusion, ce second checkpoint nous permet ainsi de visualiser l'ensemble du projet en termes d'implémentation et de prendre un peu de recul sur ce qu'il reste à faire. Nous commencerons donc notre implémentation par les trois différents sous-algorithmes et ajouterons également une petite interface simple pour les utilisateurs. Nous finaliserons également la base de données afin de nous concentrer pleinement sur l'implémentation.