



## CineFeels

Plateforme de Recommandation de Films

Basée sur l'Analyse Émotionnelle par IA

**Aouami Salma**

Student ID : 127443

**EL JABBAR Mohamed  
Houssam**

Student ID : 120248

**El Gharss Mohammed  
Amin**

Student ID : 118563

**AFRIAD Abdeslam**

Student ID : 120056

Encadrant :

Ayoub RABEH

Module :

AI Project

Année Académique : 2025 – 2026

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Contexte et Motivation . . . . .	3
1.2	Problématique . . . . .	3
1.3	Solution : CineFeels . . . . .	3
1.3.1	Innovation . . . . .	3
1.4	Objectifs . . . . .	3
<b>2</b>	<b>Architecture du Système</b>	<b>4</b>
2.1	Vue d'Ensemble . . . . .	4
2.2	Stack Technologique . . . . .	6
2.3	Structure du Projet . . . . .	6
2.4	Flux de Données . . . . .	7
2.5	Diagrammes UML . . . . .	7
2.5.1	Diagramme de Classes . . . . .	7
2.5.2	Diagramme de Séquence . . . . .	8
2.5.3	Diagramme de Cas d'Utilisation . . . . .	9
<b>3</b>	<b>Système d'Analyse Émotionnelle</b>	<b>9</b>
3.1	Modèle BERT . . . . .	9
3.1.1	Modèle Utilisé . . . . .	9
3.2	Les 10 Émotions CineFeels . . . . .	10
3.2.1	Émotions de Base (BERT) . . . . .	10
3.2.2	Émotions Composites . . . . .	10
3.3	Implémentation . . . . .	10
3.4	Mapping Genres → Émotions . . . . .	11
<b>4</b>	<b>Algorithme de Recommandation</b>	<b>11</b>
4.1	Principe . . . . .	11
4.2	Formule Mathématique . . . . .	12
4.2.1	Moyenne Pondérée . . . . .	12
4.3	Exemple de Calcul . . . . .	12
4.4	Implémentation . . . . .	12
<b>5</b>	<b>Modèle de Données</b>	<b>13</b>
5.1	MongoDB - Films . . . . .	13
5.2	Neo4j - Utilisateurs . . . . .	15

5.2.1 Requêtes Cypher . . . . .	16
5.2.2 Liaison entre MongoDB et Neo4j . . . . .	17
<b>6 API REST</b>	<b>17</b>
6.1 Endpoints Principaux . . . . .	18
6.2 Exemple de Requête . . . . .	18
6.3 Sécurité . . . . .	19
<b>7 Interface Utilisateur</b>	<b>19</b>
7.1 Pages Principales . . . . .	19
7.2 Fonctionnalités Clés . . . . .	19
7.2.1 Sélection d'Émotions . . . . .	20
7.2.2 Comparaison de Films . . . . .	21
7.3 Design . . . . .	21
<b>8 Défis Techniques</b>	<b>21</b>
8.1 Problèmes Rencontrés . . . . .	22
8.2 Exemple de Correction . . . . .	22
<b>9 Résultats et Évaluation</b>	<b>22</b>
9.1 Performances . . . . .	22
9.2 Tests Utilisateurs . . . . .	23
<b>10 Conclusion</b>	<b>23</b>
10.1 Bilan du Projet . . . . .	23
10.2 Compétences Acquises . . . . .	23
10.3 Mot de Fin . . . . .	23

# 1 Introduction

## Contexte et Motivation

Dans l'ère du streaming numérique, les utilisateurs sont confrontés à un paradoxe : une surabondance de choix qui mène souvent à l'indécision. Les plateformes comme Netflix recommandent des films basés sur l'historique de visionnage, mais ignorent un facteur crucial : **l'état émotionnel actuel de l'utilisateur.**

## Problématique

**Question centrale :** Comment recommander des films qui correspondent non pas à l'historique de visionnage, mais aux émotions que l'utilisateur souhaite ressentir à un moment donné ?

## Solution : CineFeels

**CineFeels** est une plateforme web qui place les émotions au centre du processus de recommandation. Au lieu de demander "Quel genre voulez-vous?", nous posons la question : "**Que voulez-vous ressentir?**"

## Innovation

- Analyse émotionnelle par IA (modèle BERT)
- Profil émotionnel multi-dimensionnel (10 émotions)
- Interface interactive avec sliders
- Algorithme de similarité mathématique
- Personnalisation avec historique utilisateur

## Objectifs

1. Développer une API REST performante (FastAPI)
2. Implémenter l'authentification sécurisée (JWT)
3. Intégrer l'analyse émotionnelle (BERT)
4. Créer une architecture hybride (MongoDB + Neo4j)
5. Développer une interface moderne et intuitive

## 2 Architecture du Système

### Vue d'Ensemble

CineFeels suit une architecture **Client-Serveur** avec séparation claire entre frontend et backend.

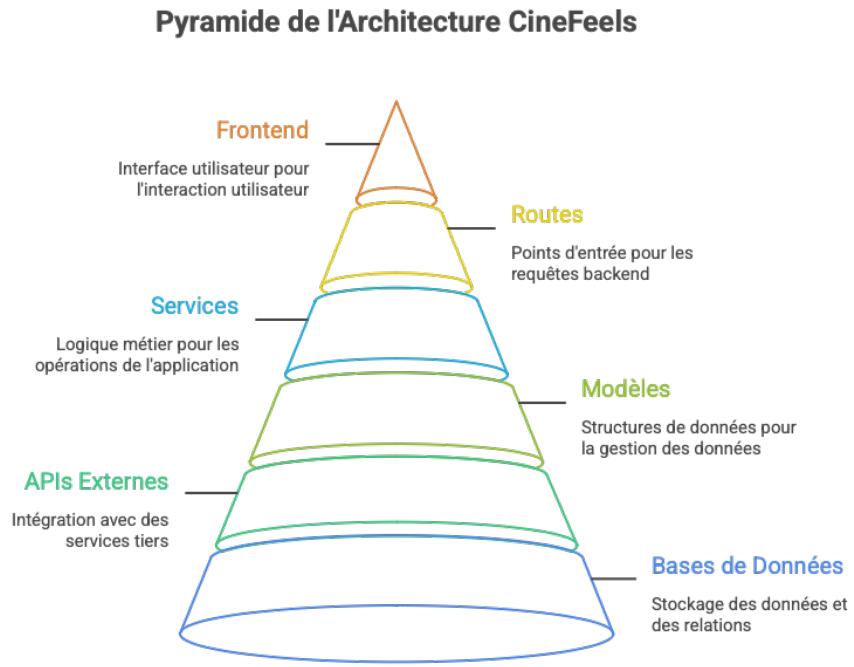
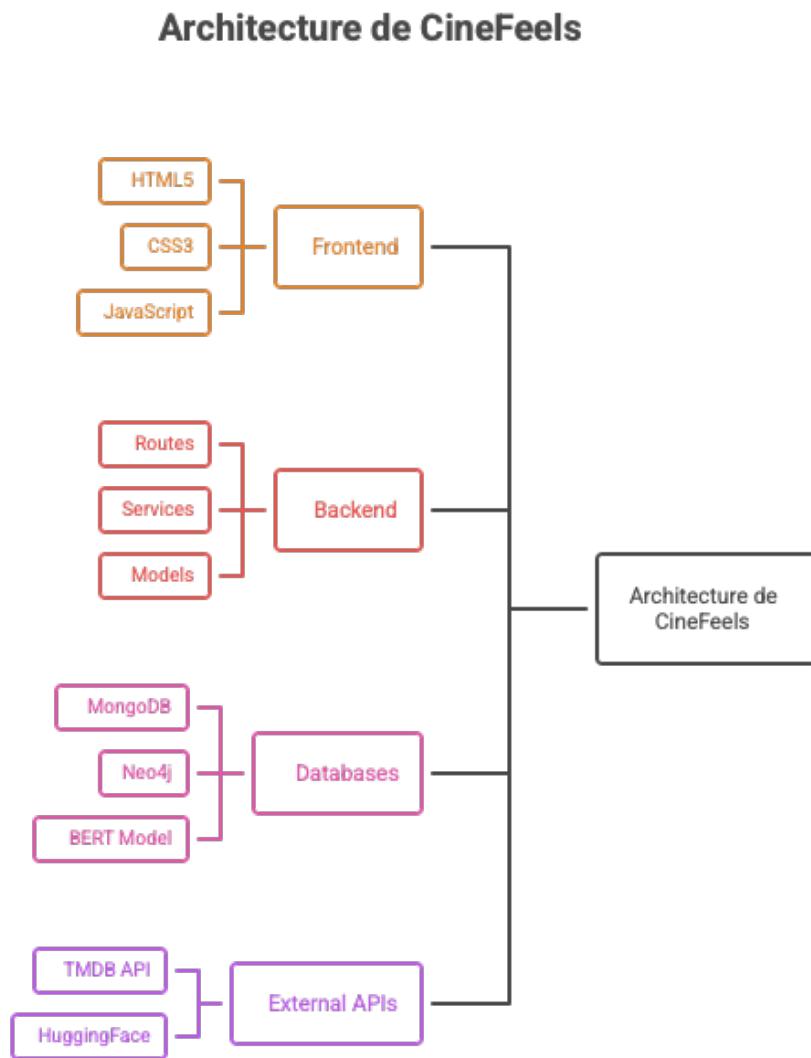


FIGURE 2.1 – Pyramide de l'architecture CineFeels



Made with Napkin

FIGURE 2.2 – Architecture détaillée par composants

## Stack Technologique

Composant	Technologie	Rôle
blue !20Backend		
Framework	FastAPI 0.100+	API REST asynchrone
Serveur	Uvicorn 0.23+	Serveur HTTP ASGI
Auth	python-jose 3.3+	Tokens JWT
Password	passlib[bcrypt]	Hashing sécurisé
green !20Bases de Données		
Films	MongoDB Atlas	Base NoSQL (cloud)
Utilisateurs	Neo4j Aura	Base graphe (cloud)
orange !20Intelligence Artificielle		
Modèle IA	BERT (DistilRoBERTa)	Analyse émotionnelle
Framework ML	Transformers 4.30+	HuggingFace
Calculs	NumPy 1.24+	Similarité
yellow !20Frontend		
Interface	HTML5 / CSS3 / JS	Interface utilisateur
Graphiques	Chart.js 4.0+	Visualisation radar

TABLE 2.1 – Stack technologique de CineFeels

## Structure du Projet

```

1 CineFeels/
2 |-- backend/
3 |   |-- main.py           # Point d'entree FastAPI
4 |   |-- api/routes/      # Endpoints REST
5 |   |-- services/         # Logique metier
6 |   |-- models/           # Modeles de donnees
7 |   |-- config/           # Configuration
8 |
9 |-- frontend_html/
10 |  |-- index.html        # Structure (~800 lignes)
11 |  |-- style.css          # Styles (~1400 lignes)
12 |  |-- script.js          # Logique (~1300 lignes)

```

Listing 2.1 – Arborescence du projet

## Flux de Données

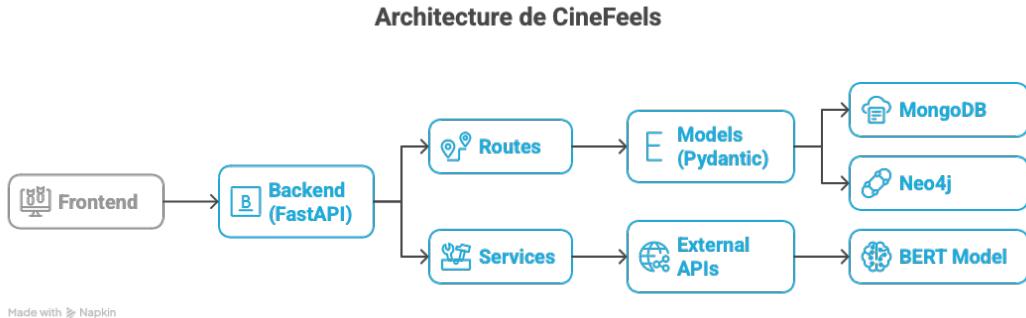


FIGURE 2.3 – Flux de communication entre composants

## Diagrammes UML

### Diagramme de Classes

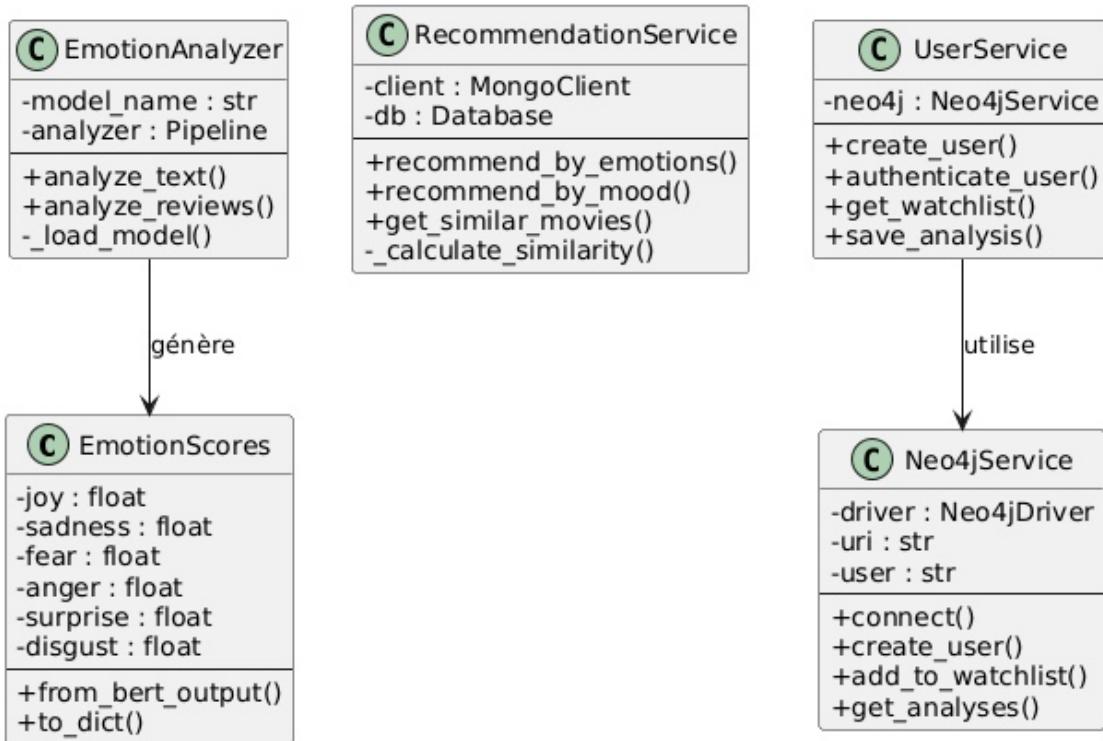


FIGURE 2.4 – Diagramme de classes

## Diagramme de Séquence

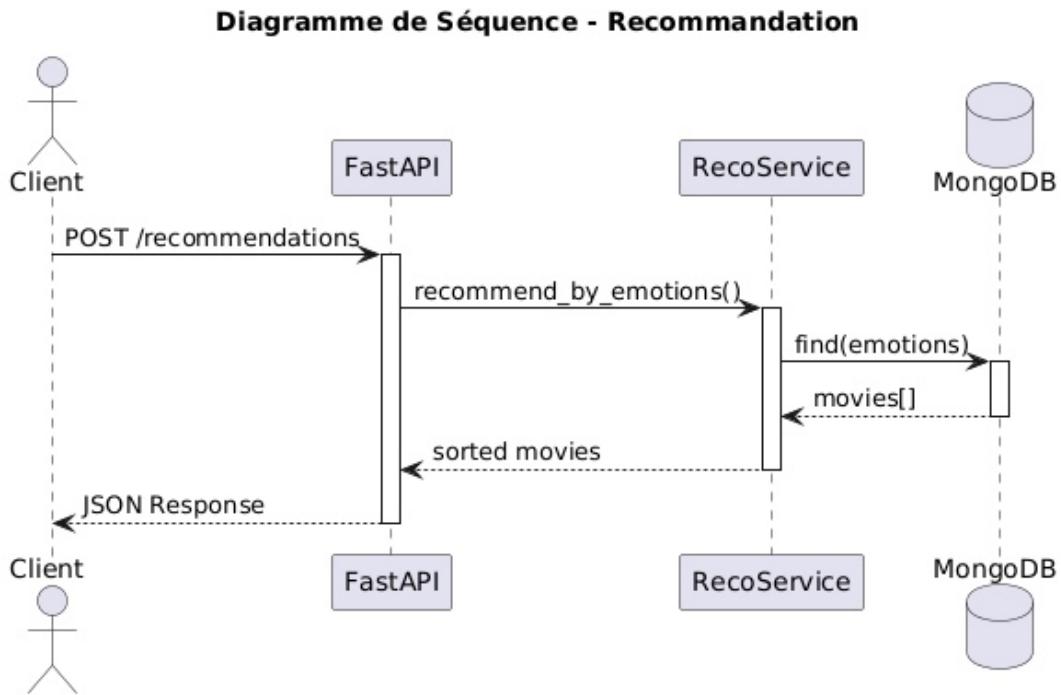


FIGURE 2.5 – Diagramme de séquence

## Diagramme de Cas d'Utilisation

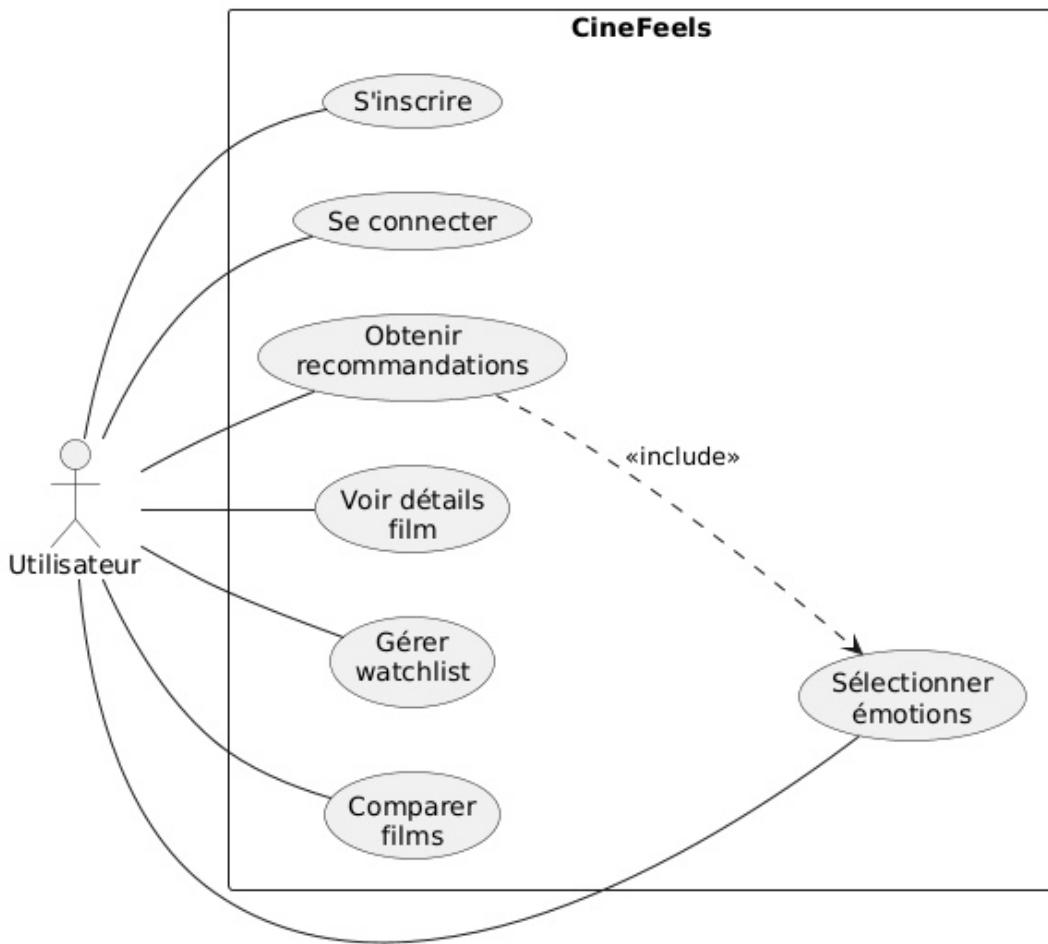


FIGURE 2.6 – Diagramme de cas d'utilisation

## 3 Système d'Analyse Émotionnelle

### Modèle BERT

BERT (Bidirectional Encoder Representations from Transformers) est un modèle de langage pré-entraîné qui révolutionne le NLP grâce à son analyse contextuelle bidirectionnelle.

### Modèle Utilisé

Nous utilisons j-hartmann/emotion-english-distilroberta-base :

- Optimisé pour la classification d'émotions
- 6 émotions de base détectées
- Équilibre entre précision et rapidité

## Les 10 Émotions CineFeels

### Émotions de Base (BERT)

Icône	Émotion	Description
	Joy	Bonheur, satisfaction
	Sadness	Mélancolie, tristesse
	Fear	Peur, anxiété
	Anger	Colère, frustration
	Surprise	Étonnement, choc
	Disgust	Dégoût, répulsion

TABLE 3.1 – 6 émotions de base (BERT)

### Émotions Composites

Icône	Émotion	Composition
	Thrill	Fear + Surprise
	Romance	Joy + romantisme
	Humor	Joy + comédie
	Inspiration	Joy + Surprise

TABLE 3.2 – 4 émotions CineFeels

## Implémentation

```

1 from transformers import pipeline
2
3 class EmotionAnalyzer:
4     def __init__(self):
5         self.analyzer = pipeline(
6             "text-classification",
7             model="j-hartmann/emotion-english-distilroberta-base",
8             return_all_scores=True

```

```

9      )
10
11     def analyze_text(self, text: str) -> dict:
12         """Analyse un texte et retourne les scores d'émotions"""
13         results = self.analyzer(text)[0]
14
15         emotions = {}
16         for result in results:
17             emotion = result['label'].lower()
18             emotions[emotion] = round(result['score'], 2)
19
20         # Ajouter emotions composites
21         emotions['thrill'] = (
22             emotions.get('fear', 0) * 0.6 +
23             emotions.get('surprise', 0) * 0.4
24         )
25
26     return emotions

```

Listing 3.1 – Service d’analyse émotionnelle

## Mapping Genres → Émotions

Lorsque l’analyse BERT n’est pas disponible, les émotions sont générées depuis les genres :

Genre	Profil Émotionnel
Action	thrill : 0.8, fear : 0.4, surprise : 0.6
Comedy	joy : 0.9, humor : 0.9
Horror	fear : 0.9, disgust : 0.5, thrill : 0.8
Drama	sadness : 0.6, joy : 0.3
Romance	romance : 0.9, joy : 0.7

TABLE 3.3 – Mapping simplifié genres-émotions

## 4 Algorithme de Recommandation

### Principe

L’algorithme calcule un **score de similarité** entre le profil émotionnel demandé et chaque film.

## Formule Mathématique

### Moyenne Pondérée

$$S = \frac{\sum_{i=1}^n w_i \times e_i^{(film)}}{\sum_{i=1}^n w_i} \quad (4.1)$$

Où :

- $S$  = Score de similarité (0 à 1)
- $w_i$  = Poids de l'émotion  $i$  (utilisateur)
- $e_i^{(film)}$  = Score de l'émotion  $i$  (film)
- $n$  = Nombre d'émotions

## Exemple de Calcul

**Utilisateur** : joy : 0.8, fear : 0.2

**Film "Zootopia 2"** : joy : 0.67, fear : 0.21

**Calcul** :

$$\begin{aligned} w_{joy} &= 0.8 / (0.8 + 0.2) = 0.8 \\ w_{fear} &= 0.2 / (0.8 + 0.2) = 0.2 \\ S &= 0.8 \times 0.67 + 0.2 \times 0.21 \\ &= 0.536 + 0.042 = \mathbf{0.578 (57.8\%)} \end{aligned}$$

## Implémentation

```

1 import numpy as np
2
3 def calculate_similarity(user_emotions: dict, movie: dict) -> float:
4     """Calcule la similarité entre profils émotionnels"""
5     movie_emotions = movie.get("emotions", {})
6
7     # Construire les vecteurs
8     user_vector = [user_emotions.get(e, 0.0) for e in user_emotions]
9     movie_vector = [movie_emotions.get(e, 0.0) for e in user_emotions]
10

```

```
11 # Convertir en arrays
12 user_array = np.array(user_vector)
13 movie_array = np.array(movie_vector)
14
15 # Normaliser les poids
16 weights = user_array / np.sum(user_array)
17
18 # Moyenne ponderee
19 similarity = np.sum(weights * movie_array)
20
21 return float(min(max(similarity, 0.0), 1.0))
```

Listing 4.1 – Algorithme de similarité

## 5 Modèle de Données

### MongoDB - Films

Structure JSON d'un film :

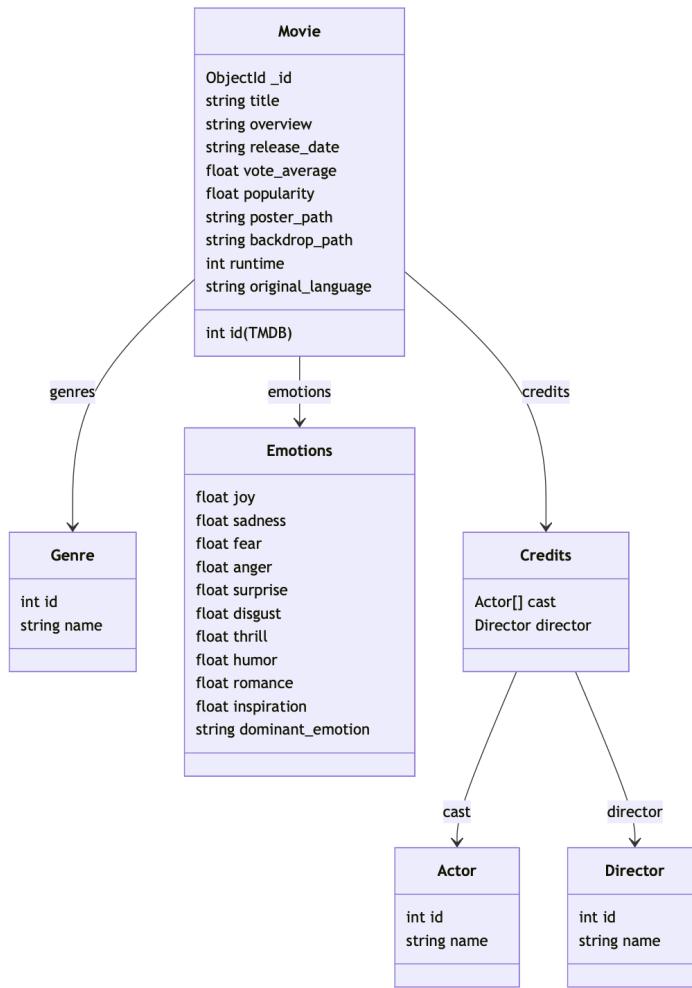


FIGURE 5.1 – Schéma document MongoDB pour les films

```

1 {
2   "id": 238,
3   "title": "The Godfather",
4   "overview": "Spanning the years 1945 to 1955...",
5   "release_date": "1972-03-14",
6   "vote_average": 8.7,
7   "genres": [
8     {"id": 18, "name": "Drama"},
9     {"id": 80, "name": "Crime"}
10  ],
11   "emotions": {
12     "joy": 0.25,
13     "sadness": 0.57,
14     "fear": 0.38,

```

```
15 "anger": 0.0,  
16 "surprise": 0.47,  
17 "disgust": 0.0,  
18 "dominant_emotion": "sadness"  
19 }  
20 }
```

Listing 5.1 – Document film dans MongoDB

## Neo4j - Utilisateurs

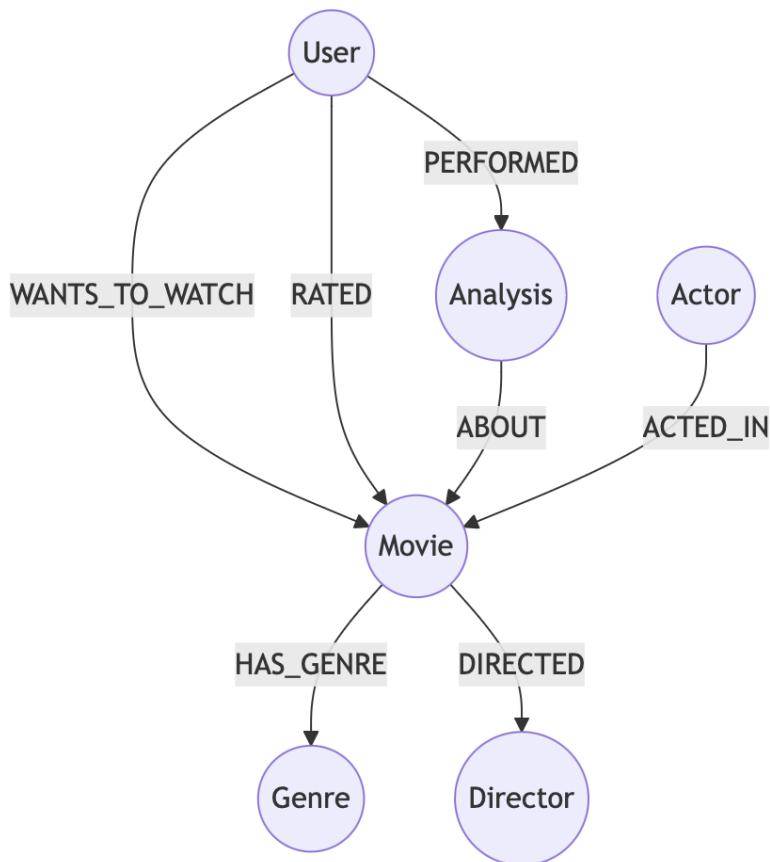


FIGURE 5.2 – Modèle de données graphe Neo4j

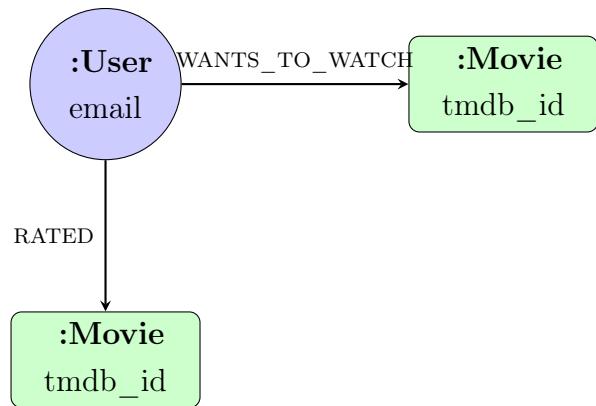


FIGURE 5.3 – Modèle de graphe Neo4j

## Requêtes Cypher

```

1 -- Creer un utilisateur
2 CREATE (u:User {
3   email: $email,
4   username: $username,
5   hashed_password: $password
6 })
7
8 -- Ajouter a la watchlist
9 MATCH (u:User {email: $email})
10 MERGE (m:Movie {tmdb_id: $movie_id})
11 MERGE (u)-[r:WANTS_TO_WATCH]->(m)

```

Listing 5.2 – Exemples Cypher

## Liaison entre MongoDB et Neo4j

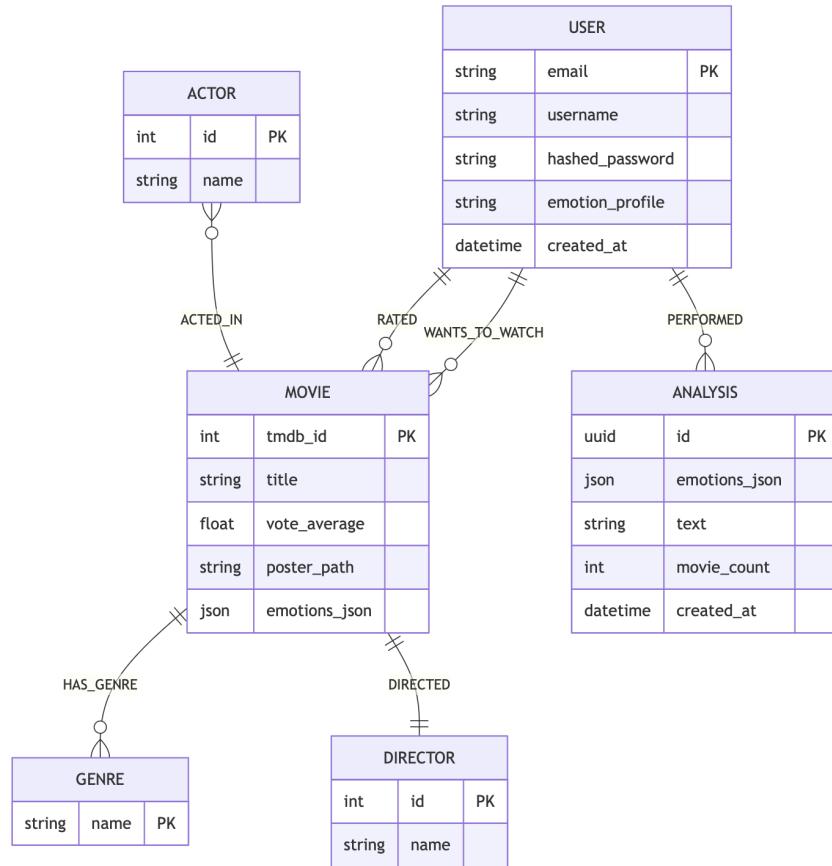


FIGURE 5.4 – Clé de liaison TMDB entre MongoDB et Neo4j

## 6 API REST

## Endpoints Principaux

Méthode	Endpoint	Auth
<b>blue !20Authentification</b>		
POST	/api/v1/auth/register	
POST	/api/v1/auth/login	
GET	/api/v1/auth/me	
<b>green !20Films</b>		
GET	/api/v1/movies	
GET	/api/v1/movies/{id}	
GET	/api/v1/movies/{id}/emotions	
<b>orange !20Recommandations</b>		
POST	/api/v1/recommendations	
GET	/api/v1/recommendations/similar/{id}	
<b>yellow !20Données Utilisateur</b>		
GET	/api/v1/user/watchlist	
POST	/api/v1/user/watchlist	
GET	/api/v1/user/analyses	

TABLE 6.1 – Endpoints de l'API CineFeels

## Exemple de Requête

```

1 // Requete
2 {
3     "emotions": {
4         "joy": 0.8,
5         "thrill": 0.6
6     },
7     "limit": 10
8 }
9
10 // Reponse
11 {
12     "movies": [
13         {
14             "id": 1084242,
15             "title": "Zootopia 2",
16             "similarity_score": 0.72,
17             "poster_path": "/...",
18             "vote_average": 7.5

```

```
19     }
20   ]
21 }
```

Listing 6.1 – POST /recommendations

## Sécurité

- **JWT** : Tokens avec expiration (30 min)
- **Bcrypt** : Hashing des mots de passe
- **CORS** : Configuration des origines autorisées
- **Validation** : Pydantic pour toutes les entrées

# 7 Interface Utilisateur

## Pages Principales

1. **Landing Page** : Présentation de CineFeels
2. **Authentification** : Inscription / Connexion
3. **Dashboard** : Sélection d'émotions avec sliders
4. **Résultats** : Liste de films recommandés
5. **Détails Film** : Informations complètes
6. **Watchlist** : Films sauvegardés
7. **Comparaison** : Comparer 2-4 films

## Fonctionnalités Clés

## Sélection d'Émotions

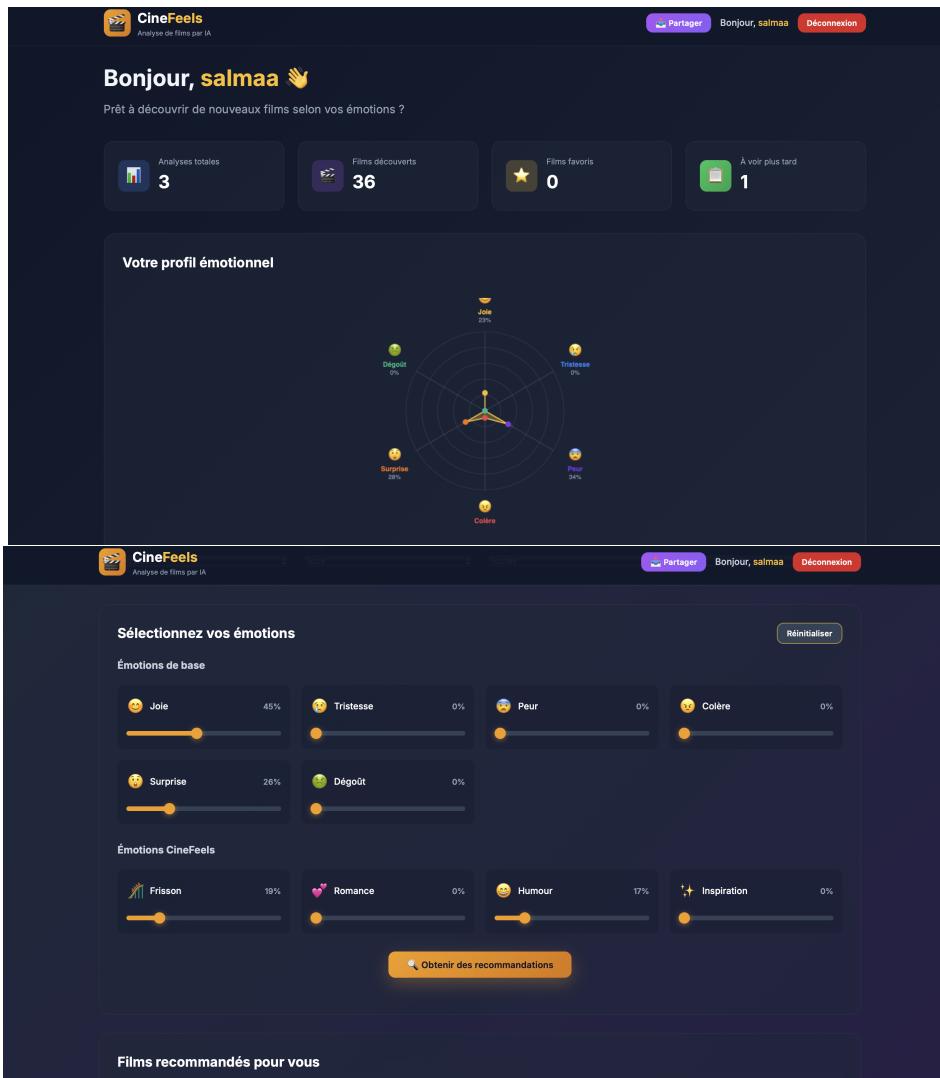


FIGURE 7.1 – Interface avec 10 sliders

- 10 sliders (0-100%) pour chaque émotion
- Graphique radar dynamique (Chart.js)
- Mise à jour en temps réel

## Comparaison de Films

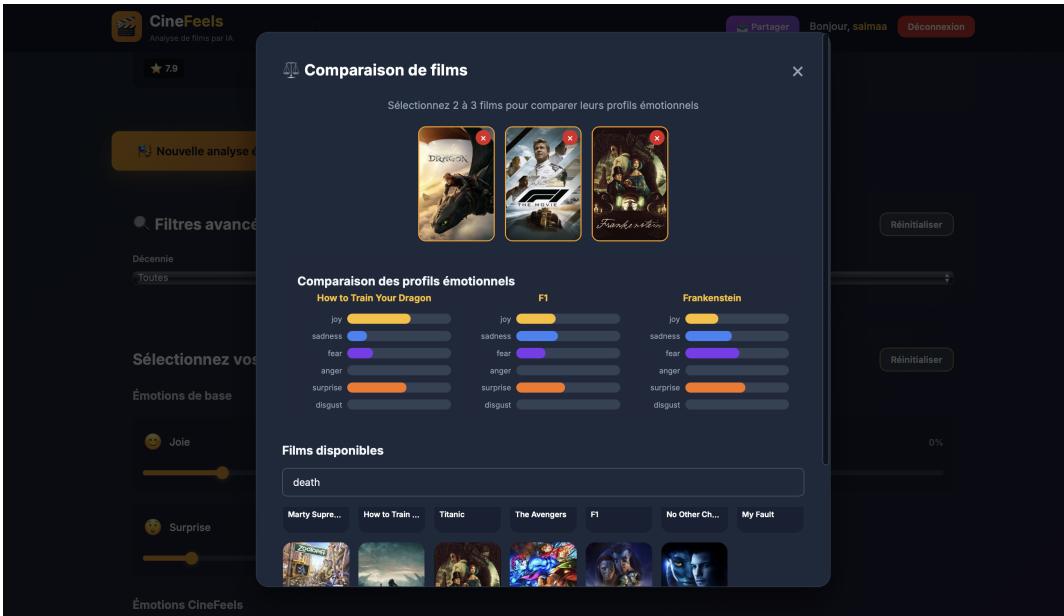


FIGURE 7.2 – Barres de comparaison

Visualisation côte-à-côte des profils émotionnels de 2-4 films avec barres horizontales.

## Design

- **Thème** : Sombre (navy) avec accents dorés
- **Responsive** : Adapté mobile/tablette/desktop
- **Animations** : Transitions CSS fluides
- **Accessibilité** : Contrastes WCAG AA

## 8 Défis Techniques

## Problèmes Rencontrés

Problème	Solution
Neo4j CypherTypeError (DateType)	Fonction <code>convert_neo4j_types()</code> pour convertir en ISO string
Neo4j Maps imbriqués non supportés	Stockage en JSON string ( <code>emotions_json</code> )
Cosine similarity peu discriminante	Changement vers moyenne pondérée
<code>anger/disgust</code> manquants	Ajout automatique avec valeur 0.0
Barres comparaison vides	Retour systématique des 6 émotions de base

TABLE 8.1 – Problèmes résolus durant le développement

## Exemple de Correction

```

1 @router.get("/{movie_id}/emotions")
2 async def get_movie_emotions(movie_id: int):
3     # S'assurer que toutes les emotions de base existent
4     base_emotions = ["joy", "sadness", "fear",
5                      "anger", "surprise", "disgust"]
6
7     emotions = movie.get("emotions", {}).copy()
8     for emotion in base_emotions:
9         if emotion not in emotions:
10             emotions[emotion] = 0.0
11
12     return {"movie_id": movie_id, "emotions": emotions}

```

Listing 8.1 – Correction émotions manquantes

# 9 Résultats et Évaluation

## Performances

- **Temps de recommandation** : 50-100 ms (500 films)
- **Analyse BERT** : 200-300 ms par texte
- **Chargement page** : < 1 seconde

- **API** : Asynchrone, support de 100+ requêtes/sec

## Tests Utilisateurs

Retours positifs sur :

- Intuitivité de l'interface
- Pertinence des recommandations
- Originalité du concept émotionnel

# 10 Conclusion

## Bilan du Projet

CineFeels réussit à créer une plateforme de recommandation innovante en plaçant les émotions au centre de l'expérience utilisateur. Le projet intègre avec succès :

- Une API REST performante et sécurisée
- Un modèle d'IA (BERT) pour l'analyse émotionnelle
- Une architecture hybride (MongoDB + Neo4j)
- Une interface moderne et intuitive
- Un algorithme de recommandation mathématiquement fondé

## Compétences Acquises

- Développement d'API REST asynchrones (FastAPI)
- Intégration de modèles d'IA pré-entraînés
- Gestion de bases de données NoSQL et graphe
- Sécurité web (JWT, hashing, CORS)
- Frontend moderne (CSS3, JavaScript ES6+)
- Travail d'équipe et gestion de projet

## Mot de Fin

Ce projet démontre le potentiel de l'intelligence artificielle pour créer des expériences utilisateur plus personnalisées et émotionnellement intelligentes. CineFeels ouvre la voie à

une nouvelle génération de systèmes de recommandation qui comprennent non seulement ce que nous regardons, mais aussi ce que nous ressentons.