**YouCode Simplon**

**Rapport sur le projet de Recommandation de Films Jay-Z Entertainment : Intégration de Big Data et IA***Réalisé par : AZEMOUR Amine le 10/12/2023*

Table des matières

Objectif**3**

Configuration de l'Environnement **4**

Prétraitement des Données avec Apache Spark**5**

Construction du Modèle de Recommandation **6**

Visualisation avec Kibana **7**

Développement de l'API Flask **9**

Conformité au RGPD et la Gouvernance des Données **11**

Conformité au RGPD et la Gouvernance des Données **12**

**Objectif**

L'objectif initial de ce projet est de créer un système de recommandation de films en temps réel tout en permettant la visualisation des données associées. Le projet se base sur l'examen de l'ensemble de données MovieLens, qui comprend des informations sur les évaluations de films, les genres, les détails des films, et les profils d'utilisateurs.

Ressources

Les données MovieLens se composent de quatre fichiers distincts:

1. data:

- Contient des informations sur les évaluations de films par les utilisateurs.

- Chaque ligne représente une évaluation avec les colonnes suivantes : UserID | MovieID | Rating | Timestamp.

2. genre:

- Contient la correspondance entre les genres et leur identifiant.

- Chaque ligne a le format suivant : GenreName|GenreID.

3. item:

- Comprend des détails sur les films, tels que le titre, la date de sortie, et les genres.

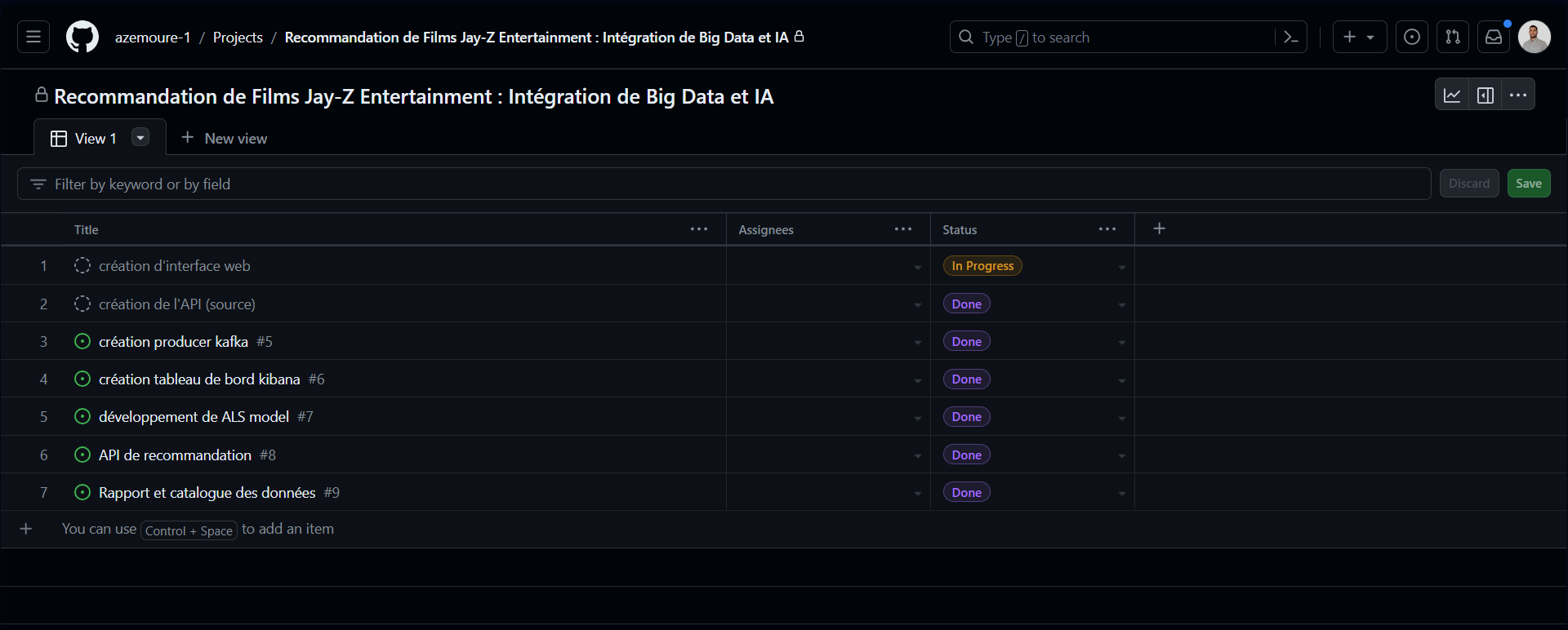
- Chaque ligne contient les informations suivantes : MovieID | Title | ReleaseDate | ... | Genre1 | Genre2 | ... | GenreN.

4. user:

- Contient les profils des utilisateurs avec des détails tels que l'âge, le sexe, le métier et le code postal.

- Chaque ligne est structurée comme suit : UserID | Age | Gender | Occupation | ZipCode.

Planification :



**Configuration de l'Environnement**

Installation

Nous avons mis en place l'environnement nécessaire en installant les outils clés sur notre machine locale Windows. Voici les étapes pour chaque composant :

Apache Spark

1. Nous avons téléchargé Apache Spark depuis le site officiel : [Apache Spark Downloads](https://spark.apache.org/downloads.html).

2. Ensuite, nous avons décompressé l'archive dans un répertoire de notre choix.

3. Pour la configuration, nous avons défini les variables d'environnement suivantes :

- JAVA\_HOME : Chemin vers l'installation de Java.

- SPARK\_HOME : Chemin vers le répertoire Spark.

4. Enfin, nous avons ajouté le répertoire bin de Spark à la variable d'environnement PATH.

Elasticsearch

1. Nous avons téléchargé Elasticsearch depuis le site officiel : [Elasticsearch Downloads](https://www.elastic.co/downloads/elasticsearch).

2. Après avoir décompressé l'archive, nous avons exécuté Elasticsearch en utilisant le script d'exécution approprié pour notre système d'exploitation.

Kibana

1. Nous avons téléchargé Kibana depuis le site officiel : [Kibana Downloads](https://www.elastic.co/downloads/kibana).

2. Après décompression, nous avons modifié le fichier de configuration de Kibana pour spécifier l'emplacement d'Elasticsearch, si nécessaire.

3. Ensuite, nous avons exécuté Kibana en utilisant le script d'exécution approprié pour notre système d'exploitation.

Flask

1. Pour Flask, nous avons vérifié que Python est installé sur notre machine.

2. En utilisant la commande :

```bash

pip install Flask

```

nous avons installé Flask.

**Prétraitement des Données avec Apache Spark**

Chargement des Données MovieLens dans Spark

Nous avons créé un processus de chargement des données MovieLens dans Apache Spark, en utilisant le module de streaming Kafka pour capturer les informations en temps réel. Trois fils d'exécution distincts ont été développés pour traiter différentes facettes des données : les données des films, les données des utilisateurs et les évaluations des utilisateurs.

**1. consumer\_movie\_data.py**

Dans ce script, nous avons configuré un consommateur Kafka pour les données des films. Les données JSON sont extraites, transformées et chargées dans un DataFrame Spark. Nous avons effectué plusieurs transformations, notamment la gestion des genres, la conversion des dates, la création de colonnes supplémentaires pour la visualisation, et enfin, l'écriture des données dans Elasticsearch.

**2. consumer\_user\_data.py**

Ce script gère les données des utilisateurs provenant du flux Kafka. Les données JSON sont parsées, et nous avons ajouté une colonne `age\_group` en fonction de l'âge des utilisateurs. Les données ont été filtrées pour exclure les enregistrements avec des valeurs manquantes, puis écrites dans Elasticsearch après la création d'un index avec le mappage approprié.

**3. user\_ratings\_data.py**

Dans ce script, nous avons pris en charge les évaluations des utilisateurs provenant du flux Kafka. Les données JSON ont été transformées, les valeurs manquantes ont été filtrées, et le type de la colonne "timestamp" a été modifié pour une meilleure interprétation. Les données ont ensuite été écrites dans Elasticsearch avec un index spécifique.

Ces scripts s'exécutent en continu, capturant les données en temps réel et les traitant avant de les charger dans Elasticsearch. Le processus est configuré pour s'adapter à des données dynamiques et garantit la cohérence entre les différentes sources d'informations.

**Construction du Modèle de Recommandation**

**1. Importation des Données**

Nous avons utilisé pandas pour importer les données à partir de trois fichiers : "data", "genre", et "item". Le fichier "data" contient les évaluations des utilisateurs pour les films, "genre" contient les genres des films, et "item" contient des détails sur chaque film, y compris son titre, sa date de sortie, son URL IMDb, et ses genres.

**2. Nettoyage des Données**

*Gestion des Genres*

Nous avons extrait les genres des films à partir des colonnes correspondantes dans le fichier "item" et avons créé une nouvelle colonne "genre" en les concaténant. Nous avons ensuite éliminé les colonnes individuelles des genres pour simplifier le DataFrame.

*Fusion des Données*

Nous avons fusionné les données des évaluations des utilisateurs avec les détails des films basés sur la colonne "movie\_id" pour créer un ensemble de données final.

*Traitement des Titres*

Nous avons supprimé les années de sortie des titres des films pour les rendre plus cohérents.

**3. Développement du Modèle**

Nous avons utilisé le module ALS (Alternating Least Squares) de Spark MLlib pour construire un modèle de recommandation collaborative. Ce modèle est basé sur la factorisation matricielle et permet de prédire les évaluations d'un utilisateur pour un film donné.

**4. Entraînement et Évaluation du Modèle**

Nous avons divisé les données en ensembles d'entraînement et de test. En utilisant l'ALS, nous avons entraîné le modèle sur l'ensemble d'entraînement. Pour évaluer ses performances, nous avons utilisé le RMSE (Root Mean Squared Error) sur l'ensemble de test. Un RMSE de 1.07 a été obtenu, indiquant une précision relativement raisonnable du modèle.

Nous avons également généré des recommandations pour tous les utilisateurs et affiché les meilleures recommandations pour le premier utilisateur.

**Interprétation du RMSE**

Le RMSE mesure la différence moyenne entre les valeurs prédites par le modèle et les valeurs réelles. Un RMSE de 1.07 signifie que, en moyenne, les prédictions du modèle peuvent différer d'environ 1.07 unité de la véritable note d'évaluation. Un RMSE plus bas indique une meilleure précision du modèle.

**Fonction de Recommandation Personnalisée**

Nous avons développé une fonction de recommandation personnalisée qui prend le titre d'un film en entrée et renvoie une liste de films recommandés en fonction du modèle ALS. Par exemple, en entrant "Toy Story (1995)", la fonction pourrait recommander des films similaires tels que "In the Name of the Father", "Leave It to Beaver", "Heavenly Creatures", etc.

Le modèle peut être utilisé pour suggérer des films à un utilisateur en fonction de ses préférences passées, améliorant ainsi l'expérience globale de l'utilisateur.

**Visualisation avec Kibana**

1. Création de Tableaux de Bord

**Distribution des Évaluations Moyennes :**

- Type de Visualisation : Histogramme

- Index Elasticsearch : "user\_ratings\_data"

- Objectif : Comprendre la répartition des évaluations moyennes attribuées par les utilisateurs. Identifiez les films les mieux et les moins bien notés.

**Genres de Films Préférés :**

- Type de Visualisation : Diagramme à barres empilées (Stacked Bar Chart)

- Index Elasticsearch : "movie\_data"

- Objectif : Identifier les genres de films les plus populaires parmi les utilisateurs. Analysez la distribution des évaluations pour chaque genre.

**Activité des Utilisateurs au Fil du Temps :**

- Type de Visualisation : Graphique de ligne ou Histogramme en aires

- Index Elasticsearch : "user\_ratings\_data"

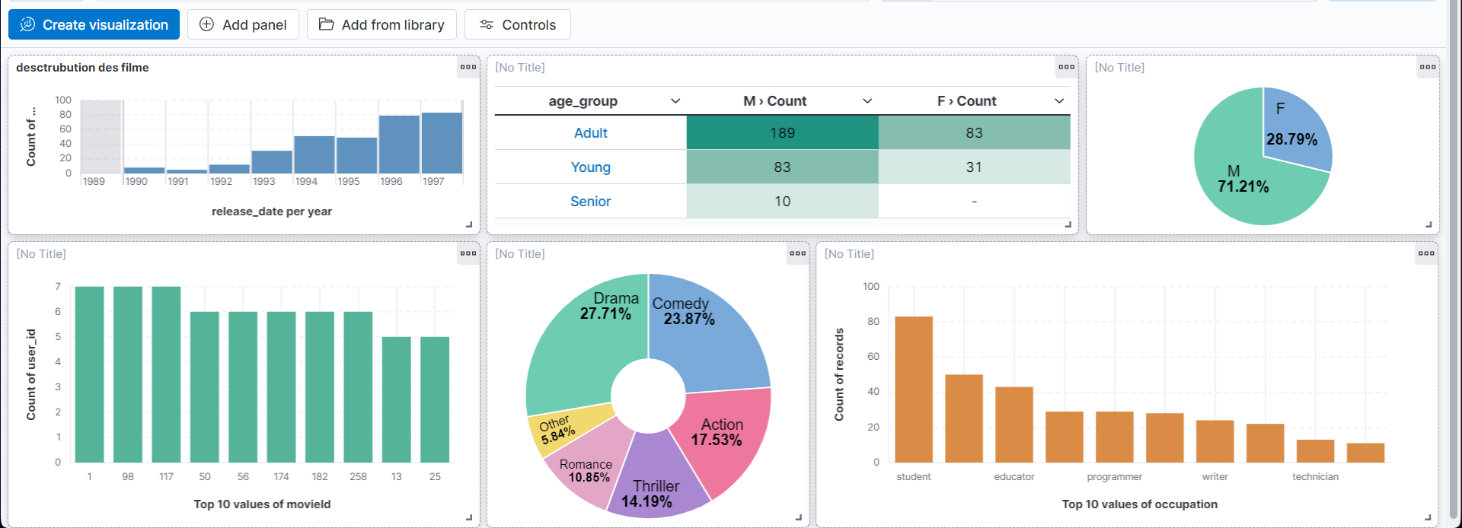
- Objectif : Suivre l'évolution de l'activité des utilisateurs au fil du temps. Identifiez les périodes de pic d'activité.

**Répartition des Âges des Utilisateurs :**

- Type de Visualisation : Histogramme

- Index Elasticsearch : "user\_data"

- Objectif : Analyser la distribution des tranches d'âge parmi les utilisateurs. Comprendre la démographie de l'audience.



2. Interprétation des Données

**Distribution des Évaluations Moyennes :**

Une majorité d'utilisateurs semble attribuer des évaluations moyennes, mais il existe également un nombre significatif d'évaluations élevées, suggérant une appréciation générale des films.

**Genres de Films Préférés :**

Les genres de comédie et de drame semblent être les plus populaires, avec un nombre élevé d'évaluations. Cela peut indiquer une préférence générale pour ces genres parmi les utilisateurs.

**Activité des Utilisateurs au Fil du Temps :**

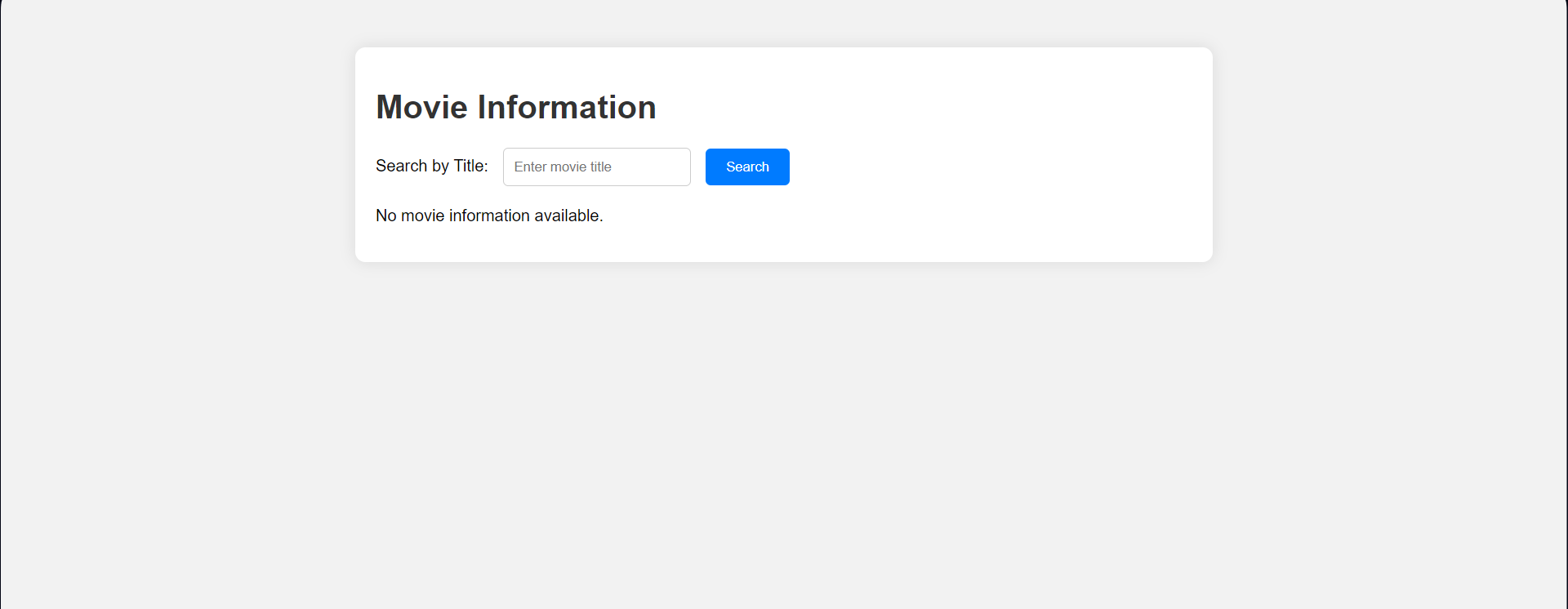
Le graphique de l'activité des utilisateurs au fil du temps peut révéler des tendances saisonnières ou des périodes de l'année où l'activité est plus intense. Cela peut être utilisé pour planifier des promotions ou des événements spéciaux.

**Répartition des Âges des Utilisateurs :**

Analyser la répartition des tranches d'âge peut aider à adapter le contenu et les recommandations pour mieux correspondre aux groupes démographiques. Cela peut également influencer les décisions de marketing.

En interprétant ces visualisations, l'équipe peut prendre des décisions informées pour améliorer l'expérience utilisateur, ajuster les recommandations, et personnaliser le contenu pour répondre aux besoins de l'audience.

**Développement de l'API Flask**



Le développement de l'API Flask constitue une étape cruciale pour offrir une interface conviviale permettant aux utilisateurs d'interagir avec le système de recommandation de films en temps réel. Cette API sera conçue pour recevoir le titre d'un film en entrée, rechercher les utilisateurs qui ont interagi avec ce film, et enfin, utiliser les identifiants d'utilisateur ainsi obtenus pour générer des recommandations personnalisées grâce au modèle ALS.

**Recevoir le Titre du Film**

L'API Flask sera configurée pour recevoir des requêtes contenant le titre du film en tant que paramètre. Cette première étape est cruciale pour identifier le film spécifique pour lequel l'utilisateur souhaite obtenir des recommandations.

**Recherche des Utilisateurs Associés**

Suite à la réception du titre du film, l'API effectuera une recherche dans la base de données pour identifier les utilisateurs qui ont interagi avec ce film. Ces interactions peuvent inclure des évaluations, des vues, ou d'autres formes d'engagement en fonction de la nature des données disponibles.

**Utilisation des ID Utilisateur pour les Recommandations**

Les identifiants d'utilisateurs ainsi identifiés seront ensuite utilisés comme entrée pour le modèle ALS. Ce modèle générera des recommandations personnalisées basées sur les préférences passées de ces utilisateurs, créant ainsi une expérience de recommandation individualisée.

**Gestion des Erreurs et Retours**

L'API sera conçue pour gérer divers scénarios, notamment la réception d'un titre de film non trouvé, l'absence d'interactions passées pour ce film, ou d'autres situations exceptionnelles. Des retours clairs et des codes d'état HTTP appropriés seront fournis pour garantir une expérience utilisateur cohérente.

**Sécurité et Confidentialité**

Des mécanismes de sécurité robustes seront mis en place pour protéger les données utilisateur et garantir la confidentialité des informations. Cela peut inclure l'utilisation de protocoles d'authentification, le cryptage des communications, et la gestion appropriée des autorisations.

**Documentation de l'API**

Une documentation détaillée de l'API sera fournie, décrivant les points de terminaison, les paramètres acceptés, les codes de réponse, et d'autres informations pertinentes. Cela facilitera l'intégration de l'API par d'autres services ou applications.

En somme, le développement de l'API Flask sera orchestré avec une approche centrée sur l'utilisateur, offrant une expérience fluide et personnalisée tout en veillant à la sécurité et à la confidentialité des données.

**Conformité au RGPD et la Gouvernance des Données**

1. Introduction

Le présent rapport vise à garantir la conformité au Règlement Général sur la Protection des Données (RGPD) ainsi que la mise en place de politiques de gouvernance des données pour le projet de recommandation de films en temps réel. Ces mesures assurent la protection des données personnelles, la qualité des données, et la transparence dans la gestion des informations.

2. Conformité au RGPD

**2.1 Consentement des Utilisateurs**

Les utilisateurs seront informés clairement et de manière transparente sur la collecte, le traitement, et l'utilisation de leurs données. Un mécanisme de consentement explicite sera mis en place, permettant aux utilisateurs de donner leur accord avant toute collecte de données. Les préférences de consentement seront stockées de manière sécurisée, et les utilisateurs auront la possibilité de les modifier à tout moment.

**2.2 Protection des Données Personnelles**

Les données personnelles seront traitées de manière sécurisée et conforme aux normes de sécurité définies par le RGPD. Des mesures de sécurité, y compris le chiffrement des données en transit et au repos, seront implémentées. L'accès aux données sera strictement contrôlé, et des audits réguliers seront effectués pour garantir la sécurité des informations.

3. Gouvernance des Données

**3.1 Politiques de Gouvernance des Données**

Des politiques de gouvernance des données seront établies pour garantir la qualité, la sécurité, et l'intégrité des données. Cela inclut la définition de responsabilités claires pour la gestion des données, la mise en place de procédures de contrôle de la qualité des données, et la création de mécanismes pour résoudre les problèmes de qualité des données de manière proactive.

**3.2 Catalogue de Données**

Un catalogue de données détaillé sera développé et maintenu. Ce catalogue fournira une traçabilité complète des sources de données, des définitions de variables, et des métadonnées associées. Il servira de référence centralisée pour assurer une compréhension approfondie des données utilisées dans le projet.

**3.3 Transparence et Responsabilité**

La transparence sera maintenue tout au long du cycle de vie des données. Les utilisateurs seront informés de la finalité de la collecte des données, et toute modification apportée à la politique de confidentialité sera clairement communiquée. La responsabilité quant à la gestion des données sera clairement définie, et des formations régulières seront dispensées pour assurer la conformité continue.

**Conclusion**

En suivant les recommandations fournies dans ce rapport, le projet de recommandation de films en temps réel s'engage à respecter les normes élevées de conformité au RGPD et à mettre en œuvre des pratiques robustes de gouvernance des données. Ceci garantit la protection des droits des utilisateurs, la qualité des données, et la transparence dans la gestion des informations. Ce rapport servira de guide pour la mise en œuvre réussie du projet, renforçant ainsi la confiance des utilisateurs et des parties prenantes.