

**Travaux de session**

**Présenté À**

**Monsieur Ismaël Biskry**

**POUR LE COURS**

**SMI1002 - Base de données 2**

**Par**

**Compaoré Yann Djamel**

**Mohamed Camara**

**Code d’accès**

**User Name : SMI1002\_033**

**Mot de passe :** **49jahm68**

Introduction

Ce rapport vise à présenter le travail réalisé dans le cadre du projet de gestion de location de films. L'objectif principal de ce projet est de développer un système informatisé permettant aux clients de louer des films et de gérer les transactions associées de manière efficace et cohérente.

Contexte du Projet

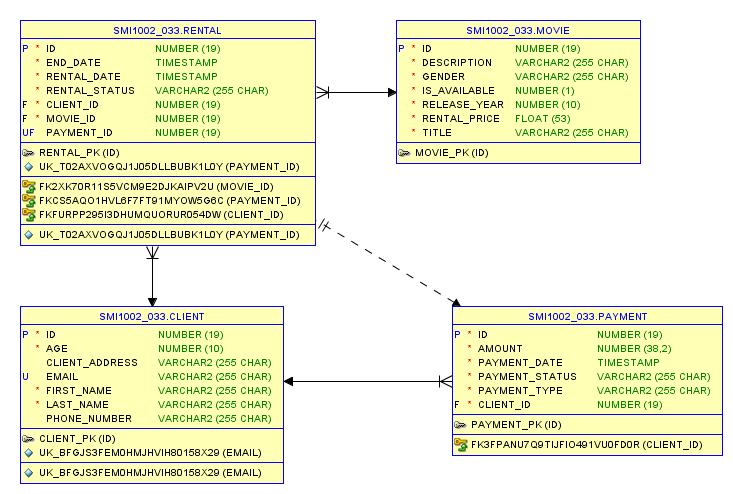
La problématique initiale était de concevoir une solution qui permettrait aux utilisateurs de rechercher, réserver et retourner des films tout en assurant la gestion des paiements et des critiques associées. Pour répondre à cette problématique, une analyse approfondie des besoins a été réalisée, conduisant à la création d'un diagramme de classe détaillé et à la mise en place de règles de gestion spécifiques.

NB : Exécuter le programme et entrer l’url suivant dans le navigateur : <http://localhost:8080/>

Modèle conceptuel Entité-relation

Il comprend les tables suivantes :

* **Movie** : Stocke les détails des films tels que le titre, la description, l'année de sortie, le genre, le prix de location, et l'indicateur de disponibilité.
* **Client** : Contient les informations des clients comme le prénom, le nom, l'âge, l'email, le numéro de téléphone et l'adresse.
* **Payment** : Enregistre les paiements effectués, y compris le montant, la date, le statut du paiement, le type de paiement (location ou frais de retard), et la référence au client qui a effectué le paiement.
* **Rental** : Gère les locations de films, avec des champs pour l'ID du film, l'ID du client, les dates de location et de retour, le statut de location et la référence au paiement correspondant.



Règles de Gestion et Cas Particuliers

1. **Disponibilité des Films** : Un film ne peut être loué que s'il est marqué comme disponible dans le système.
2. **Durée de la Location** : Un film ne peut pas être loué pour plus de 7 jours consécutifs.
3. **Paiements** : Les paiements doivent être effectués avant de pouvoir louer un film.
4. **Retours en Retard** : Si un film n'est pas retourné à temps, des frais de retard peuvent être appliqués.
5. **Restrictions d'Âge** : Les utilisateurs doivent être majeurs avant de pouvoir louer des films.

Procédures Stockées

Dans le cadre du projet, plusieurs procédures stockées ont été développées pour gérer les opérations CRUD sur les tables Client, Movie, Rental, Payment, PaymentType.

1. **AddMovie** : Ajoute un nouveau film à la base de données.
2. **UpdateMovie** : Met à jour les informations d'un film existant.
3. **DeleteMovie** : Supprime un film de la base de données.
4. **AddClient** : Ajoute un nouveau client à la base de données.
5. **UpdateClient** : Met à jour les informations d'un client existant.
6. **DeleteClient** : Supprime un client de la base de données.
7. **AddPayment** : Ajoute un nouveau paiement à la base de données.
8. **UpdatePayment** : Met à jour les informations d'un paiement existant.
9. **DeletePayment** : Supprime un paiement de la base de données.
10. **AddRental** : Ajoute une nouvelle location à la base de données.
11. **UpdateRental** : Met à jour les informations d'une location existante.
12. **DeleteRental** : Supprime une location de la base de données.

Optimisation des Requêtes de Sélection

Deux requêtes de sélection ont été optimisées pour améliorer les performances du système. Par exemple, une procédure de filtrage des locations par ID de film a été optimisée pour réduire le temps de réponse et la charge sur la base de données.

Stratégie pour les Accès Concurrentiels

Pour gérer les accès concurrentiels, des mécanismes de verrouillage ont été mis en place au niveau des transactions pour éviter les problèmes de concurrence lors de l'accès et de la modification des données.

Journalisation des Transactions

Un fichier de journal (Trace) a été mis en place pour enregistrer toutes les transactions effectuées sur la base de données, y compris les opérations d'insertion, de modification et de suppression. Cela permet de suivre et d'analyser les activités et les modifications apportées aux données.

**Schémas d’optimisation**

* **Exemple de Requête 1: Sélection de Films Disponibles sql**

SELECT title, release\_year, rental\_price

FROM MOVIE

WHERE is\_available = 1;

Optimisation:

Schéma d'Optimisation pour Requête 1:

Créer un index sur is\_available si la sélectivité est élevée.

Analyser les plans d'exécution pour confirmer que l'index est utilisé.

* **Exemple de Requête 2: Sélection de Locations pour un Client Spécifique Sql**

SELECT r.rental\_date, m.title, r.rental\_status

FROM RENTAL r

JOIN MOVIE m ON r.movie\_id = m.id

WHERE r.client\_id = 12345;

Optimisation:

Un index sur RENTAL.client\_id pourrait accélérer cette requête car il filtre sur cette colonne.

Schéma d'Optimisation pour Requête 2:

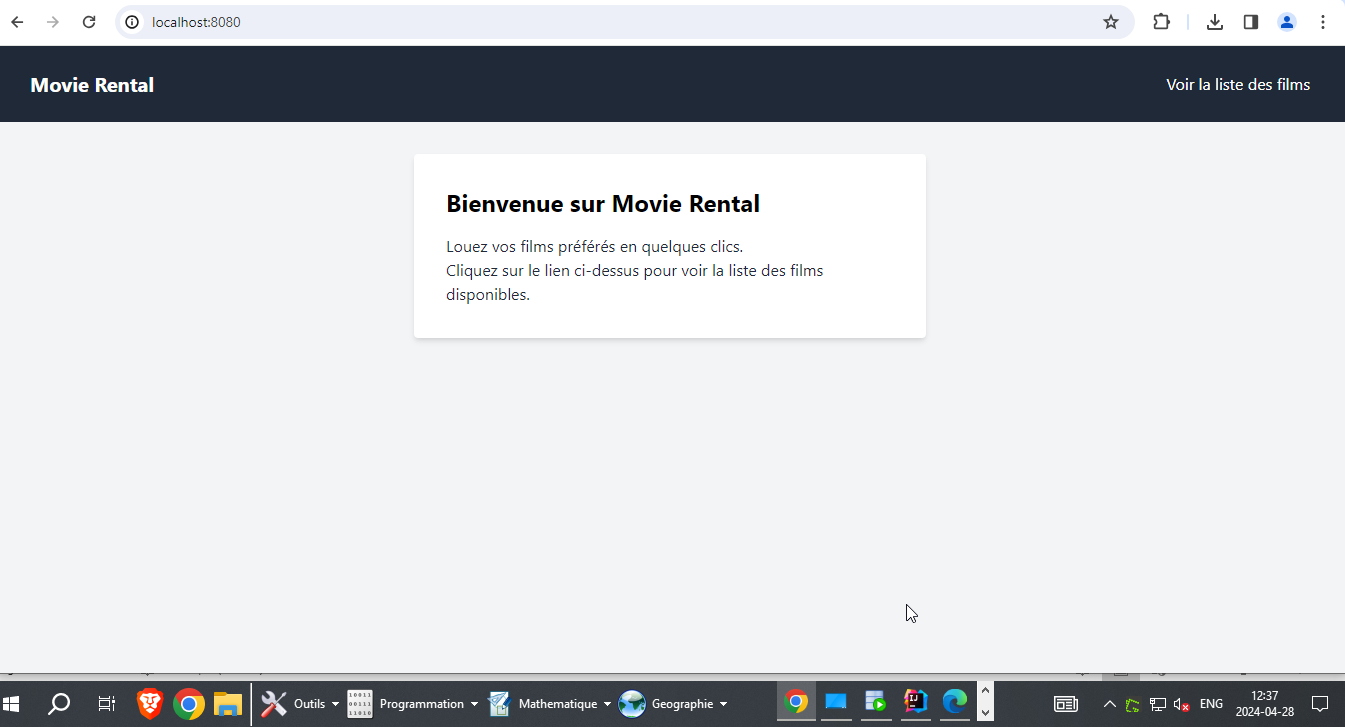
Index sur RENTAL.client\_id pour le filtrage rapide des locations par client.

Confirmation des index sur MOVIE.id et RENTAL.movie\_id pour optimiser le JOIN.

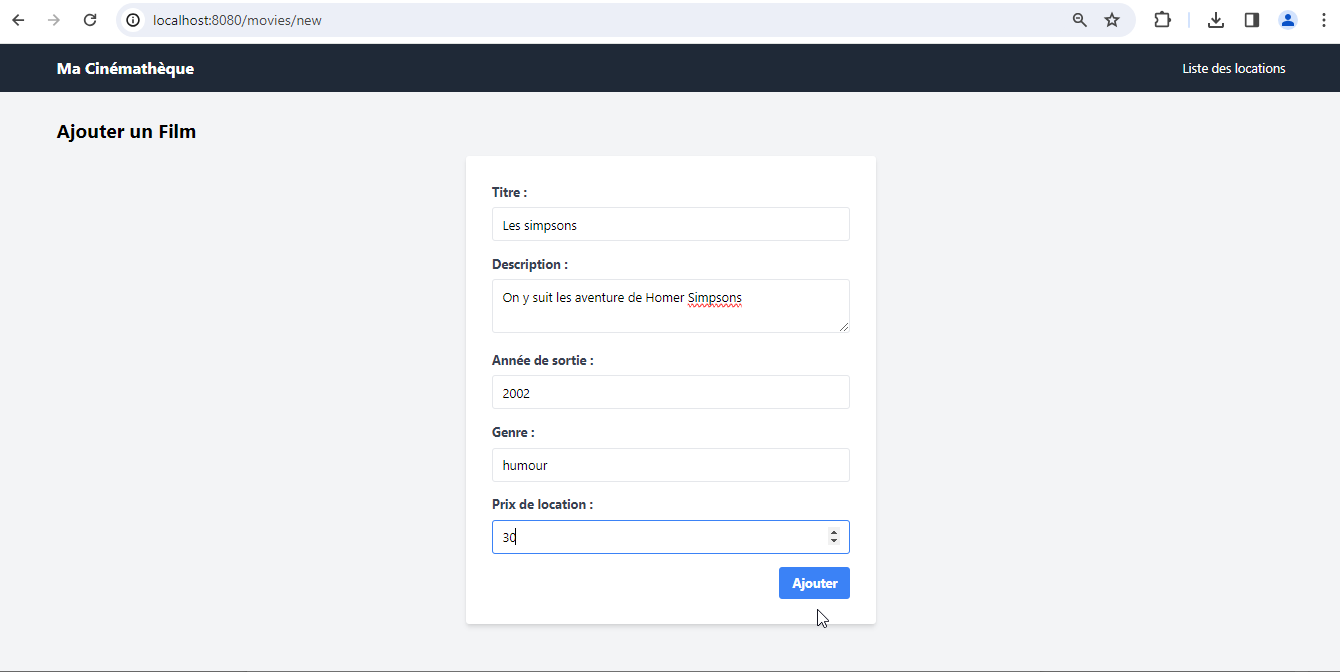
Examiner l'utilisation d'un index composite si les requêtes sont fréquentes avec la même combinaison de colonnes.

Test d’utilisation

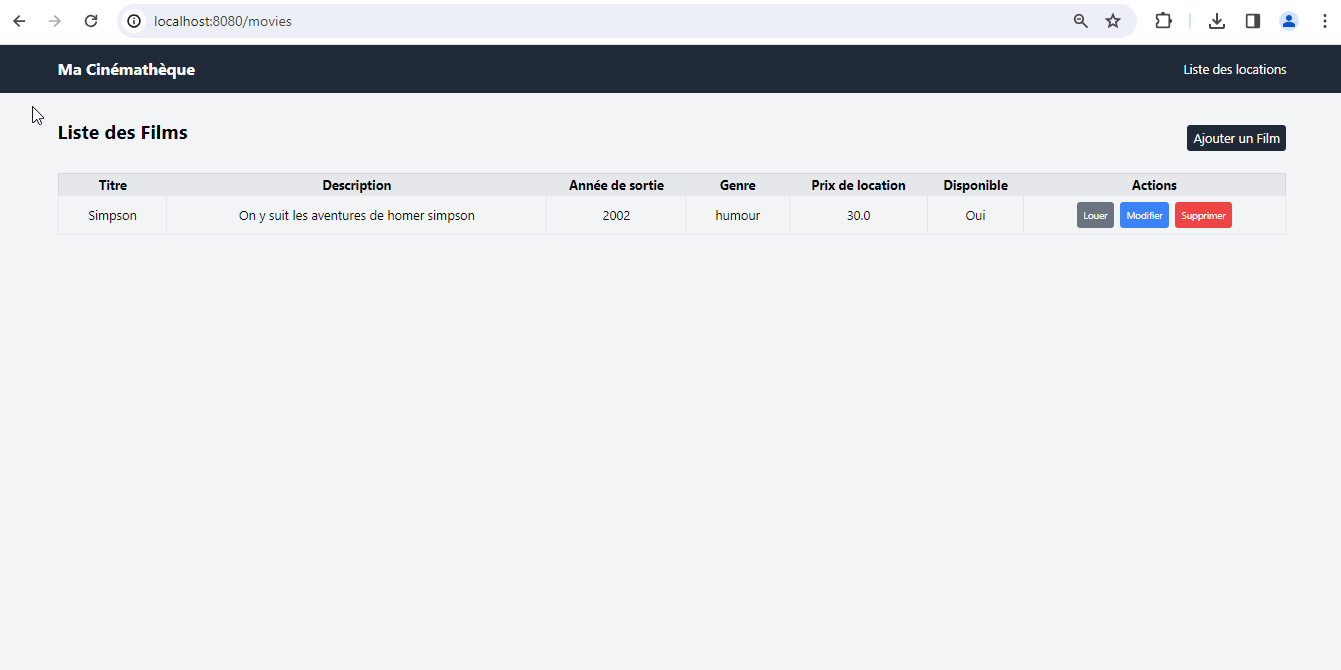
Interface home



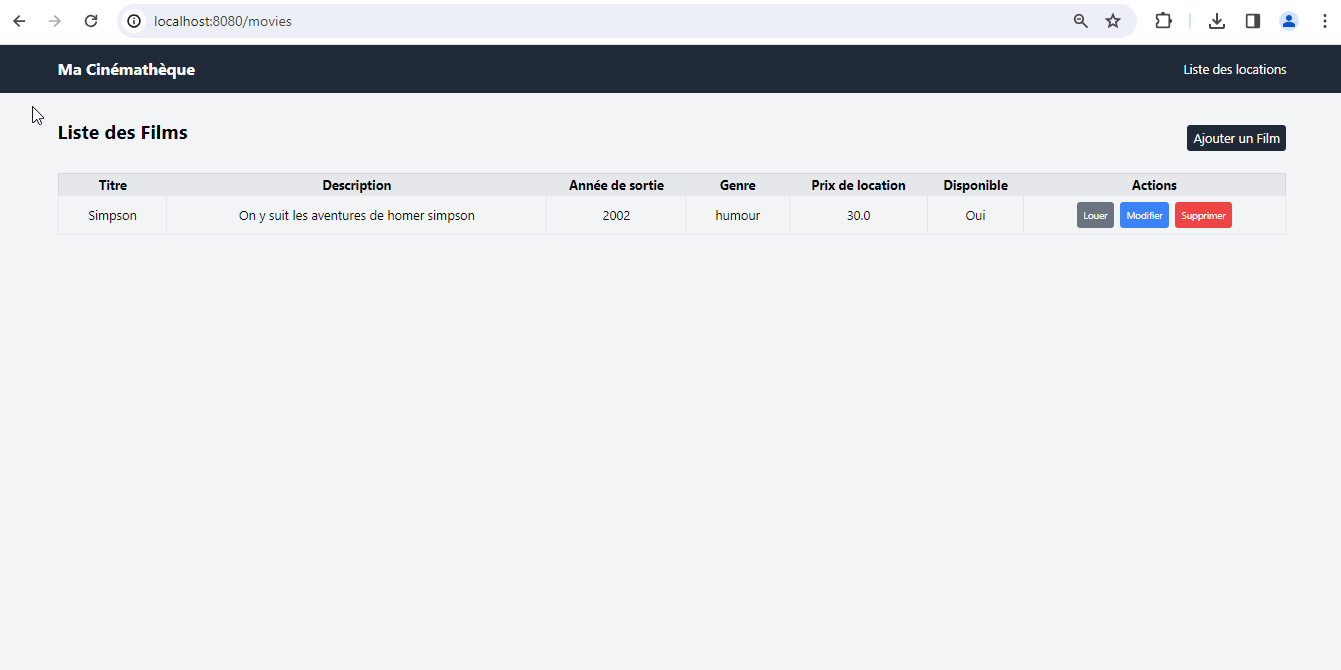
Ajouter film

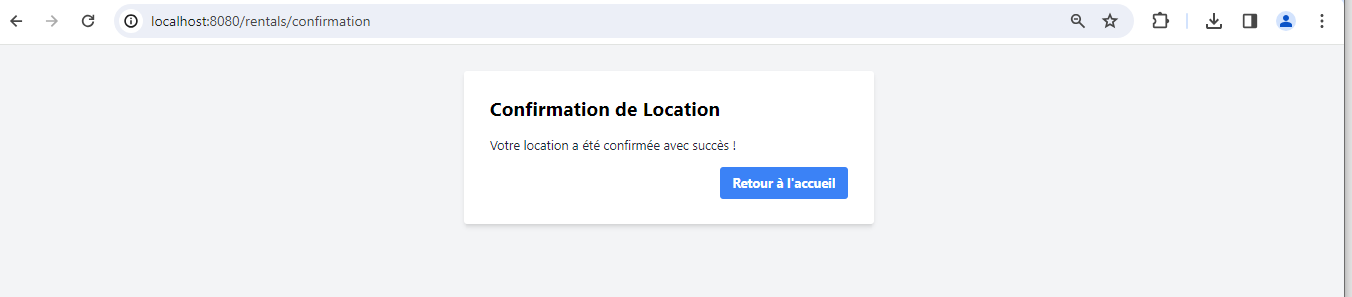


Liste des films

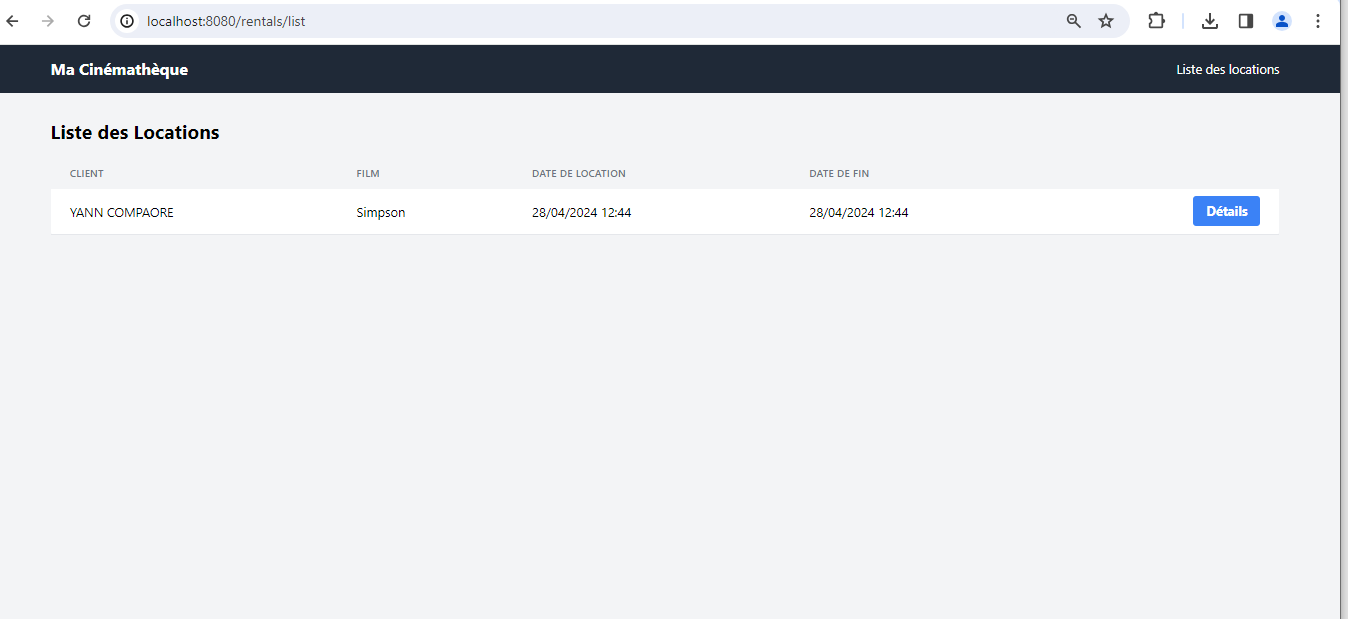


Louer film





Liste des locations



Supprimer film





Conclusion

En conclusion, ce projet a permis de mettre en place un système complet de gestion de location de films, intégrant des fonctionnalités de réservation, de paiement, de critique et de gestion des transactions. Les différentes composantes du système ont été conçues et optimisées pour assurer une performance optimale, une gestion efficace des données et une expérience utilisateur fluide.