Documentation Projet Movie Search DB :

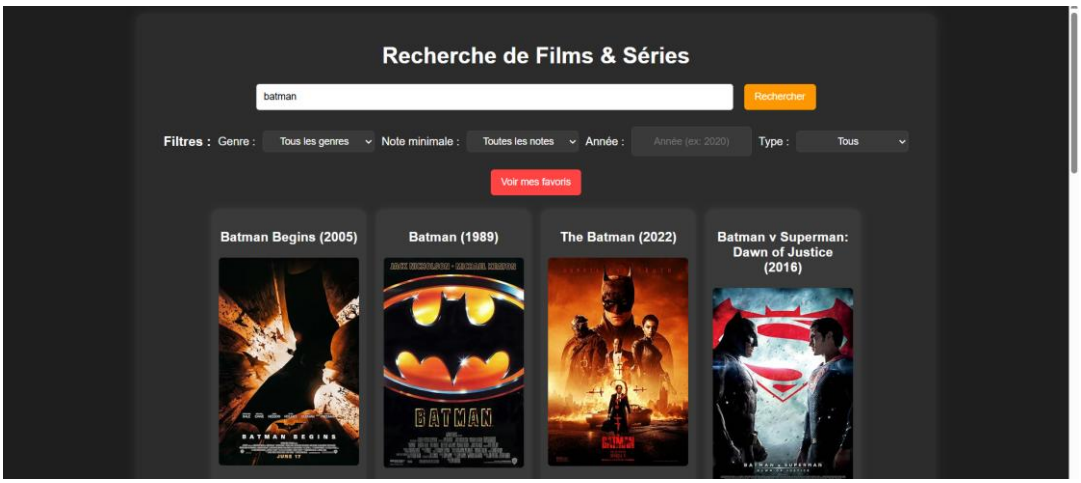
1) Présentation rapide

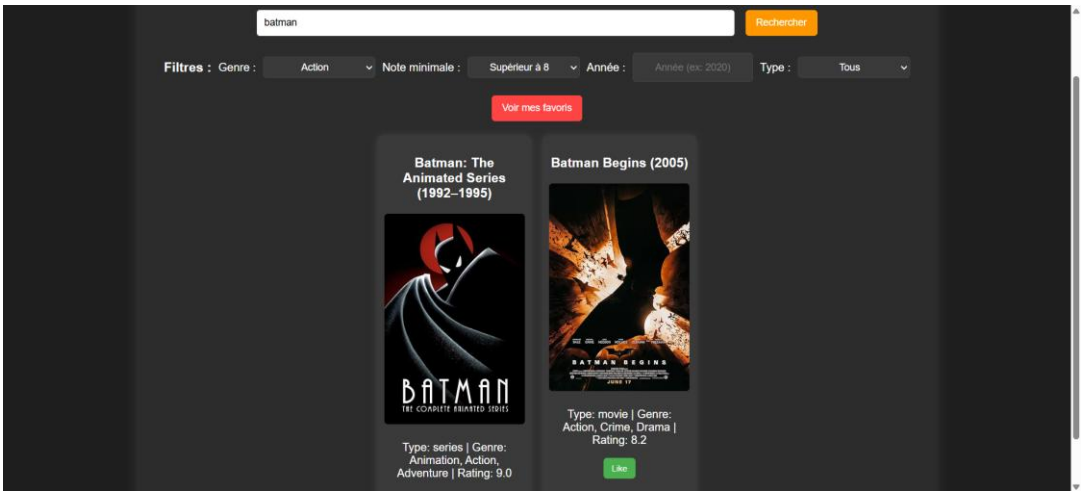
- Nom du projet : Movie Search DB (Recherche de Films & Séries)
- Objectif : Créer une interface dynamique permettant de rechercher des films ou des séries en temps réel via une API externe, de les filtrer et de gérer une liste de favoris.
- Contexte : Entraînement personnel / Mini-projet de développement Front-end.

2) Fonctionnalités

L'application offre une expérience utilisateur fluide pour explorer la base de données cinématographique OMDb :

- Recherche textuelle : Recherche par titre de film ou série.
- Système de filtrage avancé : Filtrage par genre, note minimale (IMDb), année de sortie et type (Film ou Série).
- Gestion des favoris : Possibilité de "Liker" un contenu pour l'ajouter à une section dédiée.
- Persistance des données : Les favoris sont sauvegardés même après la fermeture du navigateur.
- Public ciblé : Amateurs de cinéma souhaitant organiser leurs prochaines séances.





3) Partie technique simplifiée

- Langages utilisés : HTML, CSS, JavaScript.
- API externe : Utilisation de l'API REST OMDb API (Open Movie Database).
- Base de données : Non (utilisation du LocalStorage du navigateur pour simuler une base de données locale).
- Structure du projet : Architecture "Single Page Application" (SPA) simple dans un dossier unique, avec une séparation claire des responsabilités (Structure HTML / Design CSS / Logique JS).

4) Difficultés / Ce que j'ai appris

Ce projet m'a permis de relever des défis techniques spécifiques :

- Consommation d'API REST : Utilisation de fetch() pour l'interrogation de services tiers (OMDb API) et gestion des réponses asynchrones.
- Optimisation des performances : Implémentation de Promise.all() pour paralléliser la récupération des détails (genres, notes) de plusieurs films simultanément, garantissant une interface réactive.
- Manipulation dynamique du DOM : Génération de l'interface utilisateur (cartes de films, modale de favoris) via JavaScript sans rechargement de page.
- Persistance de données (Client-side) : Utilisation du LocalStorage pour conserver les préférences utilisateur (favoris) entre deux sessions.
- Logique de filtrage multi-critères : Développement d'algorithmes de tri croisant plusieurs variables (année, note, genre) en temps réel.