

**DOSSIER PROJET**

**Développeur Web & Web Mobile**

**Emmanuel SHLIMON**

 **La Plateforme**

Session d'examen 2025

## Sommaire

1. Introduction
2. Présentation des différents projets
3. Mon rôle
4. CP1 – Installer et configurer son environnement de travail
5. CP2 – Maquetter des interfaces utilisateur web ou web mobile
6. CP3 – Réaliser des interfaces utilisateur statiques web ou web mobile
7. CP4 – Développer la partie dynamique des interfaces utilisateur web ou web mobile
8. CP5 – Mettre en place une base de données relationnelle
9. CP6 – Développer des composants d'accès aux données SQL et NoSQL
10. CP7 – Développer des composants métier côté serveur
11. CP8 – Documenter le déploiement d'une application dynamique web
12. Conclusion

## Introduction

Pendant ma formation au titre de Développeur Web et Web Mobile à La Plateforme, j'ai eu l'occasion de travailler sur plusieurs projets concrets, seul ou en groupe. Chaque projet m'a permis de me confronter à des cas réels et de mettre en pratique ce que j'apprenais au fur et à mesure.

Ce dossier retrace ce parcours, en montrant comment j'ai validé les différentes compétences pour ce titre, à travers des réalisations variées : un site CV en HTML/CSS, un site dynamique connecté à une base de données, une maquette collaborative sur Figma, ou encore un projet orienté NoSQL.

Ce n'est pas un simple résumé de code ou de tâches, mais plutôt une façon de montrer comment j'ai progressé, ce que j'ai compris, et comment je me suis adapté aux exigences du métier.

# Présentation des différents projets

## 1. Site CV

### Contexte du projet :

Il s'agit d'un site personnel en HTML et CSS, conçu pour présenter mon profil, mon parcours et mes compétences. Le design est volontairement simple mais structuré, avec une image de fond en arrière-plan pour apporter du relief visuel. Le contenu est organisé en sections bien distinctes (présentation, compétences, contact...).

### Objectifs du projet :

Ce projet avait pour but de recréer mon CV sous forme de site web statique, en utilisant uniquement HTML et CSS. Il visait également à me faire découvrir les bases du **responsive design**, en adaptant le site à différents formats d'écran à l'aide de **media queries**.

### Durée / planning :

Le développement initial s'est déroulé sur une semaine, mais j'ai repris plusieurs fois le projet au cours de la formation pour l'améliorer au fur et à mesure de ma montée en compétences.

### Technologies utilisées :

- HTML
- CSS (media queries, animations simples)

### Difficultés rencontrées :

La gestion du responsive a été un vrai défi au départ. Il m'a fallu un peu de temps pour comprendre comment organiser mon CSS avec des breakpoints adaptés, et comment rendre chaque section lisible sur mobile. J'ai aussi découvert que la cohérence des unités (%, em, rem, etc.) jouait un rôle clé.

### Ce que je referais autrement :

Avec le recul, j'améliorerais complètement la partie design. Ce projet date du tout début de ma formation, et je vois aujourd'hui de nombreuses façons de le rendre plus moderne et visuellement cohérent. J'aurais

notamment utilisé **Figma** pour faire une maquette avant de coder, afin d'avoir une vraie vision d'ensemble du rendu final.

### **Ce que j'ai appris :**

Ce projet m'a permis de consolider mes bases en intégration web. J'ai notamment compris l'importance d'une structure HTML bien pensée, et j'ai découvert le fonctionnement des **media queries** ainsi que les bases du **responsive design**. J'ai aussi expérimenté plusieurs effets CSS (transitions, animations simples) pour améliorer l'interactivité.

## **2. Site Animatech**

### **Contexte du projet :**

Animatech est un site web dynamique développé avec PHP et MySQL, qui a pour objectif de référencer exclusivement des films d'animation, à la différence des sites classiques généralistes.

L'utilisateur peut consulter une base de données, filtrer les films par genre, accéder à une fiche détaillée, se connecter, commenter avec une note, personnaliser son profil, ou encore créer une liste de favoris.

Ce projet m'a permis de découvrir concrètement les bases du développement backend, la séparation des couches (logique métier, données, vues), ainsi que la mise en place d'une vraie expérience utilisateur.

## Cahier des charges - Projet Animatech

### Objectifs du projet :

Créer une plateforme communautaire dédiée aux films d'animation.

Les fonctionnalités clés sont :

- Navigation dans un catalogue dynamique
- Gestion de comptes (connexion, inscription, modification du profil)
- Ajout de films en favoris
- Ajout de commentaires + notation
- Partage et consultation de profils publics

Ce projet devait offrir une dimension sociale en plus du simple affichage, avec une interface fluide et personnalisée.

- Code source du site web
- Base de données exportée (fichier .sql)
- Documentation d'installation et d'utilisation
- Captures d'écran des principales fonctionnalités
- MCD, MLD, MPD

### Livrables

- Sécurité : protection contre les injections SQL, hash des mots de passe, vérification des sessions
- Performances : structure de BDD optimisée, indexation
- Expérience utilisateur : interface fluide, design épuré
- Limitation de la stack (pas de framework) pour mieux comprendre les rouages techniques

### Contraintes

- Utilisation de PHP natif pour le backend
- Base de données MySQL relationnelle
- Communication asynchrone (AJAX) pour les favoris
- API TMDB pour enrichir le contenu filmographique
- Hébergement distant via Plesk / FTP
- Responsive design avec HTML/CSS/Media Queries

### **Objectifs techniques**

- Permettre aux utilisateurs de créer un compte, se connecter et modifier leur profil
- Accéder à un catalogue dynamique de films d'animation
- Ajouter des films en favoris
- Noter et commenter les films
- Répondre à des commentaires (threading)
- Voir les profils publics des utilisateurs

### **Objectifs fonctionnels**

Animatech est un site communautaire qui centralise les films d'animation. Il vise à offrir aux passionnés une plateforme dédiée où ils peuvent consulter un catalogue, commenter les films, les ajouter en favoris et partager leur profil. Le projet a été conçu dans le cadre de la formation DWWM pour mettre en œuvre une application PHP / MySQL complète.

### **Durée / planning :**

Le projet initial a été développé sur un mois. Cependant, j'y suis revenu plusieurs fois tout au long de l'année, à mesure que j'apprenais de nouvelles compétences.

Cela m'a permis d'améliorer la qualité du code, d'optimiser les requêtes SQL, de rendre l'affichage plus ergonomique, et d'intégrer de nouvelles fonctionnalités. Ce projet représente ainsi très bien ma progression pendant toute la formation.

### Technologies utilisées :

- HTML / CSS
- JavaScript (animations et interactions front)
- PHP natif
- MySQL
- phpMyAdmin
- GitHub Desktop, VS Code

### Difficultés rencontrées :

C'était mon premier projet avec une base de données relationnelle, et au départ rien ne fonctionnait comme prévu.

Mais j'ai rapidement pris goût à ce projet, car il m'a permis de comprendre l'importance d'une **logique métier bien pensée**, ainsi que les interactions complexes entre back-end, base de données et affichage utilisateur.

Ce fut aussi un vrai challenge de sécuriser les formulaires, d'organiser les routes serveur et de structurer correctement mes requêtes SQL.

### Ce que je referais autrement :

Avec le recul, j'aurais développé ce projet avec le framework **Symfony**.

Même si PHP natif m'a permis de comprendre les fondements, le projet a atteint un niveau de complexité qui aurait bénéficié d'un cadre plus structurant.

Symfony m'aurait apporté :

- Une architecture MVC claire
- Des composants prêts à l'emploi (authentification, sécurité, formulaires...)
- Une meilleure protection des données (CSRF, injections SQL...)
- La gestion ORM avec Doctrine pour la base de données
- Une meilleure maintenabilité du code



### Ce que j'ai appris :

Ce projet m'a permis d'acquérir de nombreuses compétences :

- La logique **CRUD**
- La sécurisation des **formulaires**
- La **gestion des sessions utilisateurs**
- La **connexion et manipulation SQL**
- Des notions avancées de **JavaScript** pour enrichir l'expérience utilisateur

J'ai également gagné en autonomie et en rigueur, car c'était la première fois que je développais une application dynamique complète de bout en bout.

## 3. Projet MongoDB (NoSQL)

### Contexte du projet :

Afin d'expérimenter l'approche **NoSQL**, j'ai conçu un mini-projet nommé **FlashNotes**, une application simplifiée de prise de notes, pensée pour être légère, rapide et accessible.

Chaque utilisateur peut créer, lire ou supprimer des notes, qui contiennent un **texte**, une **date** et un **tag** (ex. : "perso", "travail", "idée").

Les données sont stockées dans une base **MongoDB**, sous forme de documents indépendants, organisés par utilisateur.

Ce projet m'a surtout permis de comprendre la logique propre aux bases NoSQL et de faire le lien entre la **structure des documents**, les **requêtes Mongo**, et le fonctionnement de l'application côté serveur.

## Objectifs du projet :

- Comprendre le fonctionnement d'une base de données **orientée documents**
- Appliquer les opérations CRUD avec MongoDB
- Simuler une interface de gestion de notes simple
- Apprendre à structurer les données sans schéma rigide

## Durée / planning :

Ce projet a été imaginé et préparé sur plusieurs jours, avec une partie interface simulée via maquettes statiques, et une structure backend théorique.

Il a servi de support à l'apprentissage des commandes MongoDB et de la logique NoSQL.

## Technologies utilisées :

- MongoDB
- Node.js (simulé côté serveur pour les routes API)
- JSON (pour représenter les données)
- Figma et captures d'écran fictives pour l'interface

## Difficultés rencontrées :

La principale difficulté a été de **désapprendre la logique relationnelle** pour comprendre le fonctionnement "souple" et flexible de MongoDB.

J'ai aussi dû revoir ma manière de penser les relations entre utilisateurs et notes, en intégrant le fait qu'une base NoSQL privilégie **l'agrégation** et **l'autonomie des documents**.

## Ce que je referais autrement :

Si je devais aller plus loin avec ce projet, je le connecterais à un vrai front développé en **React** ou **Vue.js**, pour tester une API Node.js réelle avec

MongoDB.

Je pourrais aussi implémenter un système de **connexion sécurisée**, et ajouter des fonctionnalités comme la modification ou le tri des notes, ou encore la synchronisation multi-appareils.

### Ce que j'ai appris :

Grâce à FlashNotes, j'ai découvert :

- La structure flexible des documents MongoDB
- Les requêtes spécifiques au NoSQL (find, insertOne, deleteMany...)
- La manière d'organiser des données imbriquées (sous-documents, tableaux...)
- Les avantages et cas d'usage d'une base NoSQL face à un modèle relationnel

Ce projet m'a permis de poser des bases solides pour intégrer MongoDB dans de futurs projets, notamment dans une architecture de type **API REST + frontend JS**.

## 4. Maquette Figma : Les Repas de Lili

### Contexte du projet :

Dans le cadre d'un **travail de groupe pour l'Atelier**, j'ai participé à la conception de la maquette d'un site web appelé **Les Repas de Lili**.

Le projet avait une dimension solidaire : il visait à créer une plateforme qui met en relation des **restaurateurs** et l'**association Les Repas de Lili**, afin de proposer des repas à **1 euro** destinés aux personnes démunies.

L'idée était de permettre aux restaurateurs partenaires de proposer des plats accessibles, que l'association redistribue ensuite selon les besoins.

### Objectifs du projet :

- Concevoir une maquette complète et responsive sur Figma
- Travailler en équipe sur une **arborescence claire**, des **wireframes**, et les **écrans finaux**
- Intégrer les contraintes UX/UI pour une interface accessible et intuitive
- Répondre à un besoin concret de lien entre acteurs sociaux et restaurateurs solidaires

### **Durée / planning :**

Le projet s'est déroulé sur environ 1 mois, avec des points réguliers en équipe.

Nous avons défini les rôles de chacun, partagé les maquettes via Figma, et validé collectivement chaque étape (arborescence, zoning, responsive, charte graphique finale).

### **Technologies utilisées :**

- Figma (pour toute la phase de maquettage)
- Méthodologie collaborative (partage, commentaires, itérations)
- Outils complémentaires : Trello pour l'organisation des tâches, Discord pour la communication

### **Difficultés rencontrées :**

Le principal défi a été l'absence de charte graphique fournie au début du projet.

Nous avons donc dû travailler les premiers écrans **en noir et blanc**, en nous concentrant uniquement sur la structure et l'ergonomie, dans l'attente de validations côté client.

Cela nous a obligés à nous adapter, à rester flexibles, et à mettre en avant la **clarté fonctionnelle** plutôt que le style visuel.

### Ce que je referais autrement :

Avec plus de temps, nous aurions pu travailler une version plus poussée de la charte graphique, incluant un choix typographique plus cohérent, une palette de couleurs validée dès le début, et des composants UI réutilisables. J'aurais aussi anticipé davantage les **comportements utilisateurs mobiles** pour améliorer la version responsive.

### Ce que j'ai appris :

- Comprendre le fonctionnement d'un outil de maquettage collaboratif comme **Figma**
- Apprendre à concevoir un parcours utilisateur structuré (UX)
- M'initier aux bases de l'UI design (cohérence visuelle, hiérarchie d'information)
- Travailler en équipe de manière fluide et organisée

C'était une expérience très formatrice qui m'a montré l'importance de l'étape de maquettage en amont du développement web.

## Mon rôle

Je m'appelle Emmanuel, j'ai 22 ans et je vis à Marseille.

Depuis mon enfance, j'ai toujours baigné dans l'univers du numérique, ce qui m'a naturellement orienté vers un métier en lien avec ce domaine.

C'est ainsi que j'ai intégré, en début d'année 2024, la formation des **Petits Débrouillards**, qui propose une initiation aux différents métiers du numérique.

C'est au cours de cette expérience que j'ai découvert **La Plateforme** et la formation de **Développeur Web & Web Mobile** qu'elle proposait. J'y ai vu

une véritable opportunité de me lancer concrètement dans ce domaine qui m'attire depuis longtemps.

Ce qui me plaît particulièrement dans ce métier, c'est l'aspect évolutif et stimulant du développement : il faut sans cesse se tenir à jour, découvrir de nouvelles technologies, apprendre en continu et ne jamais se reposer sur ses acquis. C'est un domaine exigeant, mais passionnant, dans lequel je me projette pleinement.

Au fil des projets réalisés pendant la formation, mon rôle a évolué selon les contextes et les objectifs pédagogiques. J'ai pu travailler en autonomie sur certaines parties, mais aussi collaborer avec d'autres apprenants sur des projets de groupe. Cela m'a permis de toucher à l'ensemble du processus de développement web, tout en développant des compétences en gestion de projet, en communication et en adaptation.

Sur les projets individuels, comme le **site CV** ou le site **Animatech (Cinétech)**, j'ai été entièrement responsable de la conception, du développement et de l'organisation du code. J'ai dû faire mes propres choix techniques, structurer mes fichiers, corriger mes erreurs et gérer l'ensemble du cycle de développement. Cela m'a permis de gagner en rigueur, notamment sur la lisibilité du code, la réutilisabilité des composants et le respect des bonnes pratiques (comme la séparation du HTML, CSS, JS ou la gestion sécurisée des formulaires en PHP).

Dans le cadre du projet **Les Repas de Lili**, réalisé en groupe, j'ai participé à la création de la maquette sur Figma.

Nous avons réparti les tâches entre nous, organisé plusieurs points d'échange pour valider les décisions d'UX/UI, et utilisé des outils collaboratifs pour travailler efficacement à distance. Cette expérience m'a permis de mieux comprendre les contraintes liées au travail en équipe, l'importance de la clarté dans la communication, et la nécessité de bien documenter les choix graphiques et fonctionnels.

Enfin, même dans le projet fictif **FlashNotes** (MongoDB), j'ai joué le rôle d'un développeur qui devait concevoir une structure de données adaptée au NoSQL. Ce projet m'a servi à approfondir les logiques documentaires et les requêtes spécifiques à MongoDB, que je n'avais jamais manipulé avant cette formation.

Dans tous les cas, j'ai toujours cherché à progresser, à poser des questions quand je bloquais, et à comprendre ce que je faisais, pas juste à appliquer des recettes. Mon rôle a donc été à la fois technique et évolutif, avec l'objectif constant de devenir plus autonome et plus efficace dans mes choix de développeur.

## CP1 – Installer et configurer son environnement de travail

### Contexte

Dès le début de la formation, il a été essentiel de mettre en place un environnement de développement stable, fonctionnel et adapté aux technologies que nous allions utiliser tout au long des projets. L'objectif était de pouvoir coder, tester, versionner et déployer nos applications web dans des conditions proches de celles rencontrées en entreprise.

Bien sûr cela n'a pas été facile j'ai eu des problèmes à faire fonctionner mon PHP sur mon éditeur au début car le chemin d'environnement était mal défini.

La synchronisation avec GitHub m'a demandé un réel temps d'adaptation, notamment sur la gestion des conflits lors des merges. J'avais aussi du mal avec les commit dont je n'avais pas saisi l'importance au départ et qui par la suite se sont avérés très utiles et indispensable

J'ai donc dû installer, configurer et parfois dépanner plusieurs outils pour travailler efficacement sur mes projets individuels et collaboratifs.

## Outils utilisés

**Visual Studio Code (VS Code)** : éditeur de code principal.

**Laragon** : environnement de développement local pour PHP et MySQL. Plus simple à configurer que XAMPP ou WAMP, et plus rapide dans son exécution. J'ai notamment configuré mes projets pour qu'ils se lancent directement dans le dossier www de Laragon.

**Git et GitHub Desktop** : pour la gestion de version. J'ai appris à versionner mes projets, créer des branches, faire des commits réguliers et pousser mon code sur GitHub.

**Node.js & npm** : installés pour certains projets front-end nécessitant des outils comme *Live Sass Compiler* ou des frameworks JS.

**MongoDB Compass** : interface graphique pour manipuler les bases de données NoSQL dans le cadre du projet fictif *FlashNotes*.

**Figma** : pour les projets de maquettage, notamment *Les Repas de Lili*, en collaboration avec d'autres élèves.



## CP2 – Maquetter des interfaces utilisateur web ou web mobile

### Contexte

La compétence de maquettage a été abordée principalement dans le cadre d'un **travail de groupe** sur le projet *Les Repas de Lili*.

Il s'agit d'un site imaginé pour faire le pont entre des **restaurants partenaires** et une **association**, avec un objectif fort : permettre aux restaurateurs de vendre des repas à **1 euro seulement**, afin que ceux-ci puissent ensuite être redistribués aux personnes démunies grâce à l'association *Les Repas de Lili*.

Avant même d'écrire une ligne de code, il fallait réfléchir à l'**arborescence du site**, à l'organisation des écrans, et à la manière dont les utilisateurs allaient naviguer sur la plateforme. C'est là que le maquettage a joué un rôle essentiel.

### Technologies utilisées

Nous avons utilisé **Figma**, un outil de design collaboratif, pour construire la maquette.

C'était la première fois que je travaillais en équipe sur un outil de ce type. J'ai découvert plusieurs aspects importants :

- Le fonctionnement des **frames** et **composants**
- L'utilisation des **grilles et contraintes responsives**
- Le partage et la coédition en temps réel

Le projet a été pensé dès le départ en **version mobile-first**, avec une adaptation responsive ensuite sur tablette et desktop.

## Réalisation

Voici les principales étapes du travail effectué :

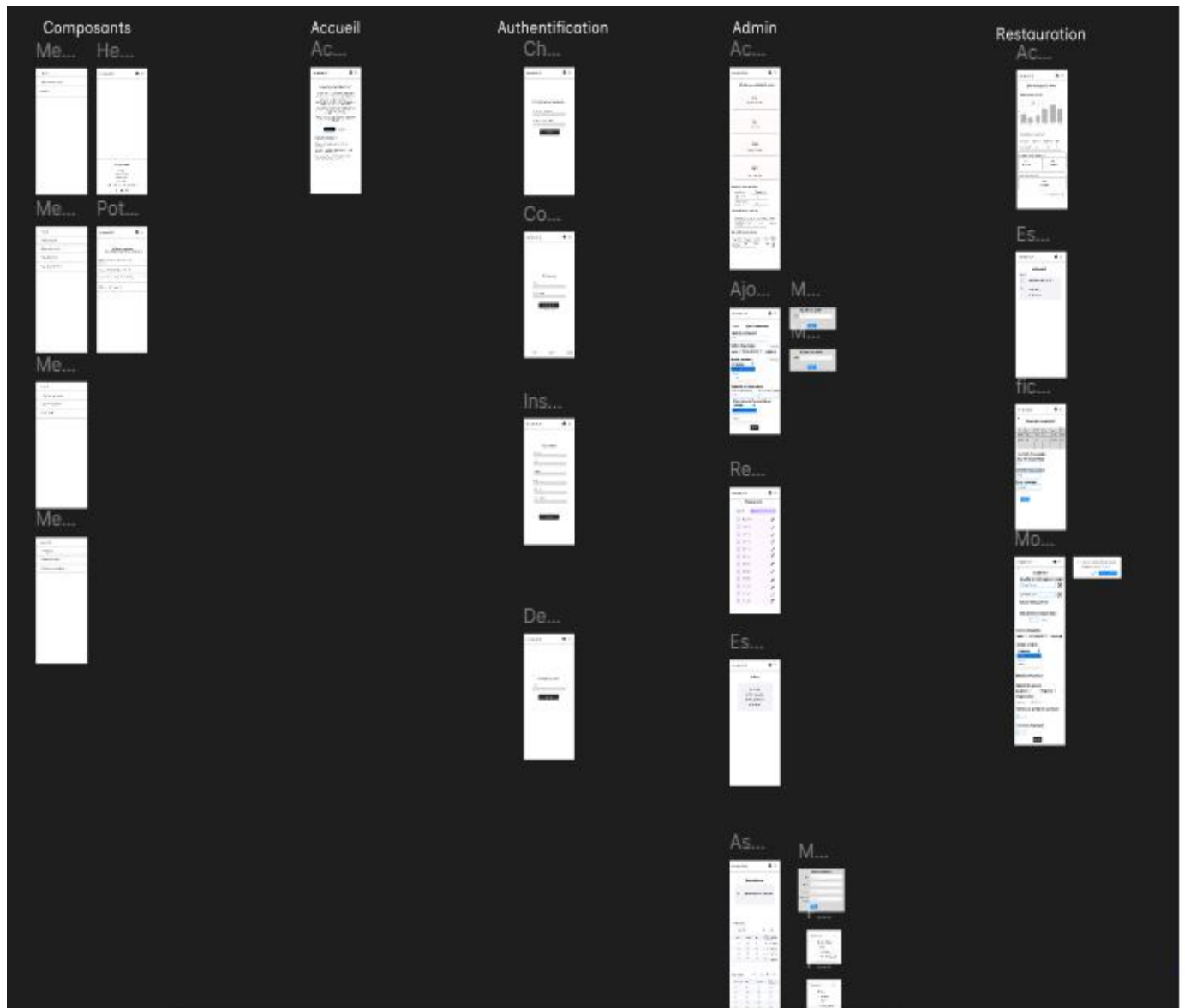
- **Brainstorming en groupe** sur les besoins fonctionnels du site
- **Création de l'arborescence** générale du site (pages principales, parcours utilisateur)
- Répartition des écrans à maquetter entre les membres de l'équipe
- Réalisation des **wireframes basse fidélité** pour valider les idées rapidement
- Création des **maquettes finales**, en intégrant une charte graphique simple et claire (typographie lisible, couleurs cohérentes avec l'identité du projet)
- **Vérification de la cohérence UX**, notamment sur les parcours de commande, de connexion et de gestion des repas

J'ai personnellement travaillé sur l'interface d'accueil, la page de connexion, et une partie de l'espace client.

Ce projet m'a permis de mieux comprendre l'importance du maquettage : on gagne énormément de temps en développement quand on sait exactement à quoi chaque écran doit ressembler.

J'ai aussi appris à **travailler avec des contraintes**, à prendre en compte la logique utilisateur, et à argumenter certains choix graphiques en équipe. Le fait de devoir faire des compromis ou revoir certaines idées m'a montré l'importance de l'écoute et de la clarté dans la communication technique.

## Aperçu global des maquettes Figma (version mobile).



## CP3 – Réaliser des interfaces utilisateur statiques web ou web mobile

### Contexte

Cette compétence a été mise en œuvre dans le cadre de la création de mon site CV. L'objectif était de réaliser une page personnelle attractive et responsive, qui permette de présenter mon profil, mes compétences et mes projets dans une interface propre, moderne et intuitive. Il s'agissait du tout premier projet de type "site vitrine" que j'ai développé en autonomie, en mobilisant les bases du HTML et du CSS, ainsi que des outils de structuration et de mise en page.

### Technologies utilisées

- **HTML5** : pour structurer le contenu sémantique de la page (titres, sections, liens, formulaires).
- **CSS3** : pour la mise en forme du site, l'organisation des sections, la gestion des couleurs, polices et animations.
- **Media Queries** : pour l'adaptation responsive aux différents formats (mobile, tablette, desktop).
- **Flexbox** : pour gérer l'agencement des blocs de contenu de manière fluide et réactive.
- **Google Fonts** : pour une typographie moderne et lisible.

## Illustrations

### Version desktop du site CV.



## Version mobile responsive du site CV.



## CP4 – Développer la partie dynamique des interfaces utilisateur web ou web mobile

### Contexte


Cette compétence a été mise en œuvre dans le cadre du projet **Animatech**, un site web permettant de consulter, commenter et mettre en favoris des films d'animation.

Le projet avait pour but d'intégrer une vraie couche dynamique à un site vitrine, en exploitant la communication entre le client (navigateur) et le serveur, et en gérant des données utilisateur (authentification, favoris, avis...).

Le tout a été développé sans framework, en pur PHP, pour bien comprendre les mécanismes internes d'un site dynamique.

### Fonctionnalités de l'application

**Création de compte** : formulaire d'inscription avec contrôle des champs et vérification en base.



**INSCRIPTION**

Nom d'utilisateur

Email

Mot de passe

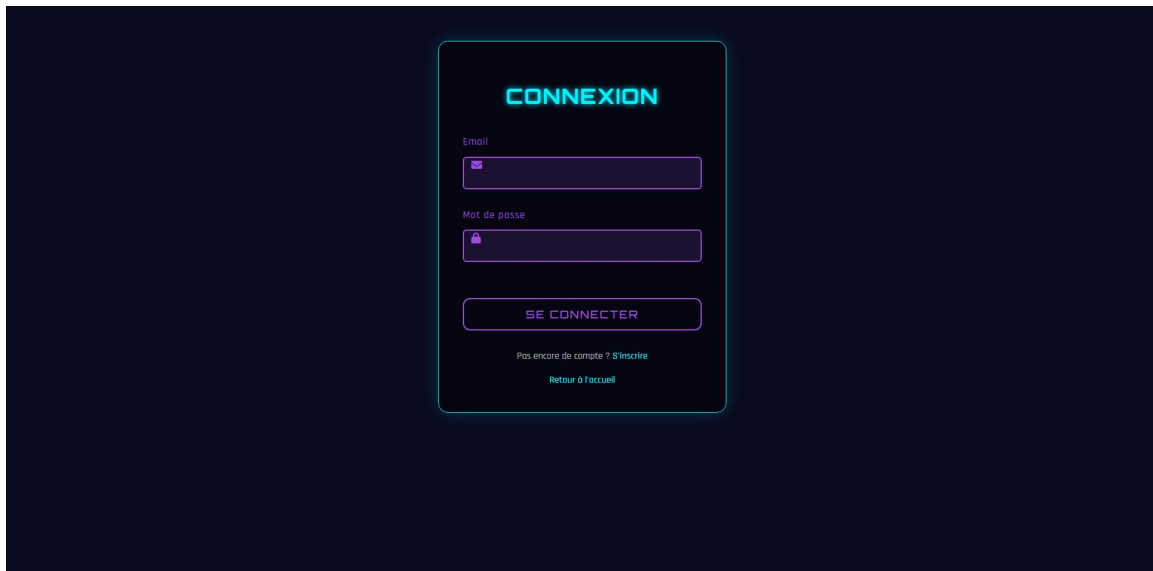
Le mot de passe doit contenir au moins 6 caractères

**S'INSCRIRE**

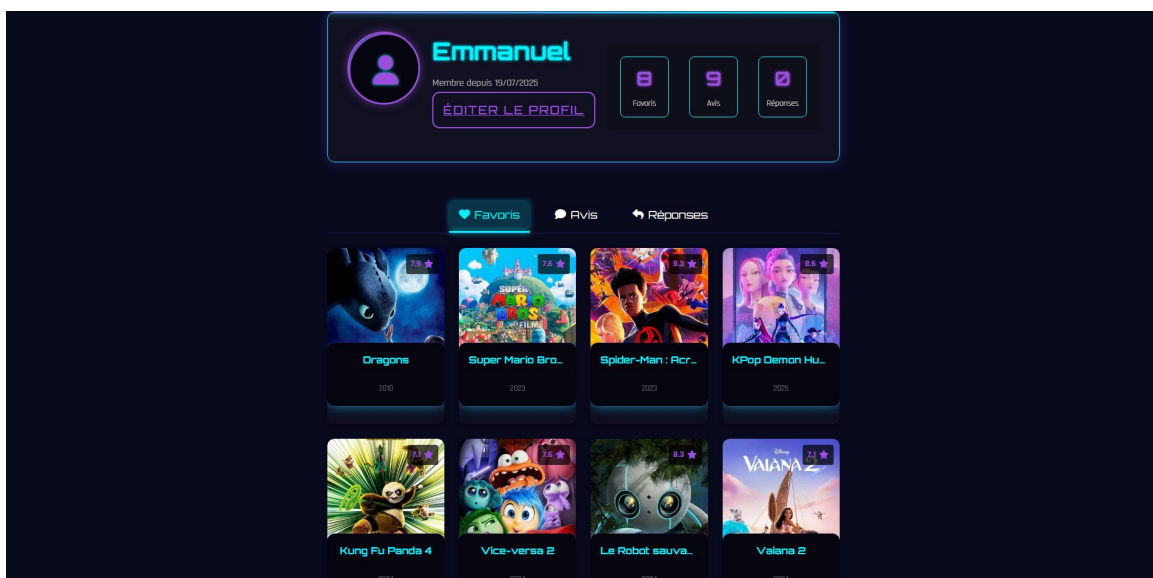
Déjà un compte ? [Se connecter](#)

[Retour à l'accueil](#)

**Connexion / déconnexion sécurisée** : gestion des sessions utilisateurs.

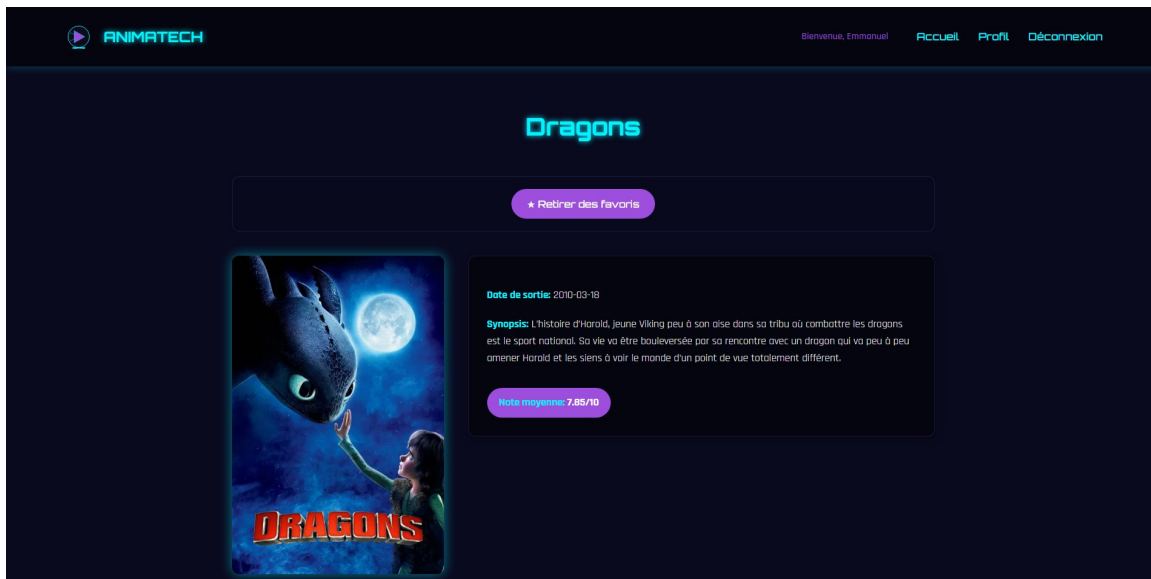


**Page profil** : chaque utilisateur peut consulter les films ajoutés en favoris.

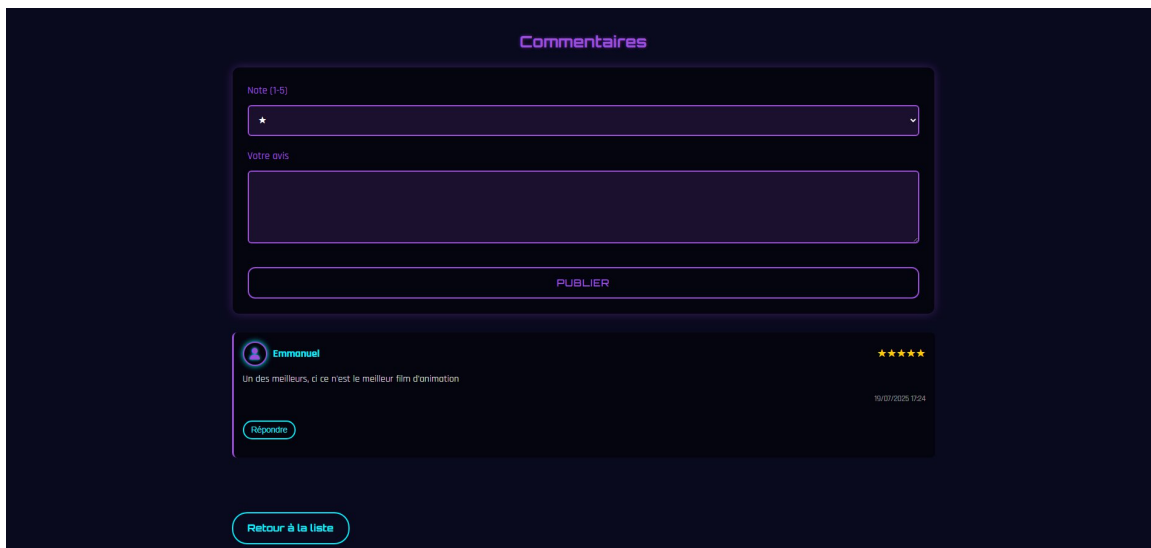




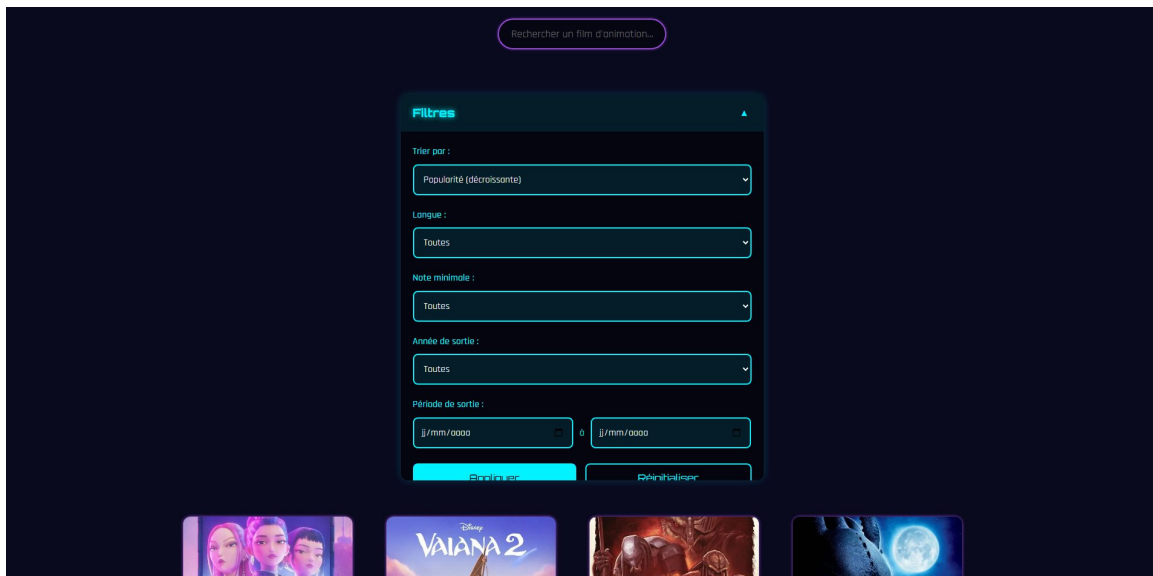
**Ajout / suppression de favoris** : clic interactif pour stocker des films en favoris.



**Commentaires** : zone de commentaire sur chaque fiche film, reliée à la base de données.



**Tri par genres & pagination** : possibilité de filtrer les films et de naviguer par pages.



**Affichage dynamique des films** : les films sont chargés dynamiquement depuis l'API TMDB, et les données peuvent être enrichies localement en base.

```
# Configuration de l'API TMDB
TMDB_API_KEY=eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOi
TMDB_API_URL=https://api.themoviedb.org/3/
```

## Technologies utilisées

**PHP (sans framework)** : pour le traitement des formulaires, des sessions, et l'interaction avec la base.

**MySQL** : pour stocker les utilisateurs, favoris, et commentaires.

**HTML / CSS** : pour structurer les pages.

**JavaScript (léger)** : pour améliorer l'interaction utilisateur (ex : favoris en AJAX, animation de chargement).

**API TMDB** : pour récupérer dynamiquement les affiches et résumés des films.

**Sessions PHP** : pour gérer l'authentification et la personnalisation du contenu.

## Illustration page d'accueil



ANIMATECH

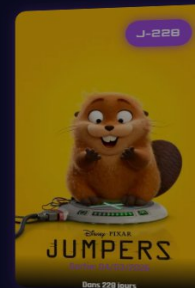
Bienvenue, Emmanuel

[Accueil](#)

[Profil](#)

[Déconnexion](#)

### FILMS ANIMATION LES PLUS ATTENDUS



Rechercher un film d'animation...

## CP5 – Mettre en place une base de données relationnelle

### Contexte

Dans le cadre du projet **Animatech**, il a été nécessaire de mettre en place une base de données relationnelle robuste, capable de gérer les utilisateurs, leurs actions (favoris, commentaires), et les informations complémentaires aux films récupérés via l'API TMDb.

L'objectif était de concevoir un schéma relationnel cohérent, puis de l'implémenter dans un environnement de développement local à l'aide de **phpMyAdmin**, tout en respectant les bonnes pratiques de normalisation et de structuration des tables.

### Modèle conceptuel des données

Le **Modèle Conceptuel des Données** représente les différentes entités manipulées dans le projet *Animatech* ainsi que leurs relations logiques, sans se soucier des aspects techniques de mise en œuvre. Il sert de base pour concevoir une base de données cohérente, normalisée et adaptée aux besoins fonctionnels.

Dans ce projet, plusieurs entités principales ont été identifiées :

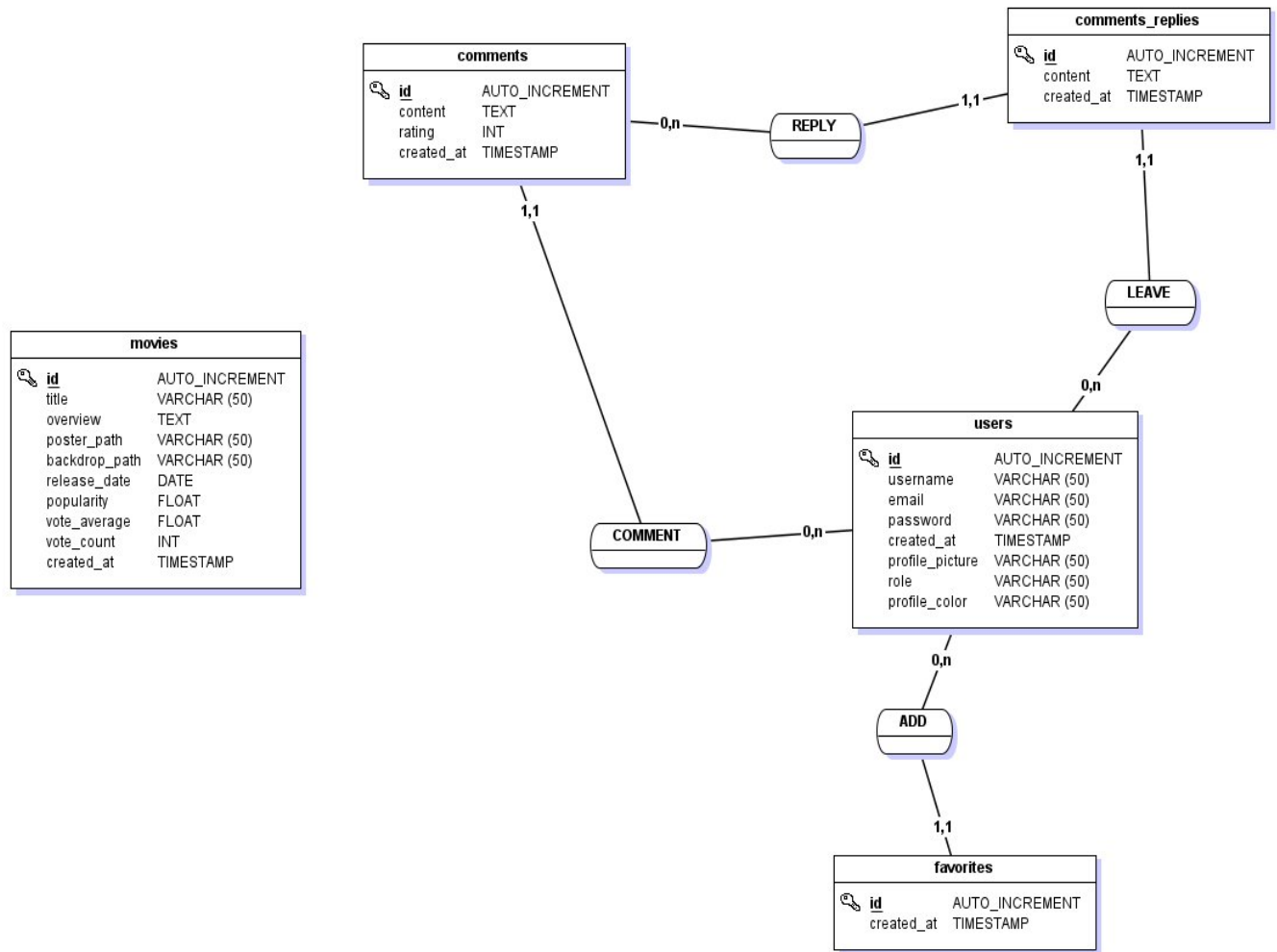
- **users** : représente les utilisateurs de la plateforme, avec leurs informations personnelles (email, mot de passe, profil, rôle, etc.)
- **movies** : correspond aux films d'animation référencés dans la base, avec leurs métadonnées (titre, résumé, popularité, etc.)
- **comments** : les commentaires laissés par les utilisateurs sur les films.
- **comments\_replies** : permet de gérer les réponses à des commentaires, créant une forme de discussion autour des films.
- **favorites** : entité permettant à un utilisateur de marquer un film comme favori.

Les **relations entre ces entités** sont décrites à travers des associations nommées :

- **COMMENT** : lie un utilisateur à un ou plusieurs commentaires.
- **REPLY** : permet à un commentaire d'avoir une ou plusieurs réponses.
- **LEAVE** : lie une réponse à un utilisateur.
- **ADD** : permet à un utilisateur d'ajouter des films à sa liste de favoris.

Chaque relation est annotée avec une **cardinalité** (ex : 0,n ; 1,1) qui permet de définir combien d'occurrences d'une entité peuvent être associées à une autre.

## Schéma MCD



## Modèle logique des données

À partir du MCD, j'ai défini un Modèle Logique des Données structurant les informations de façon plus technique et en introduisant les notions de clés primaires, clés étrangères et types de données.

Chaque entité a été convertie en table :

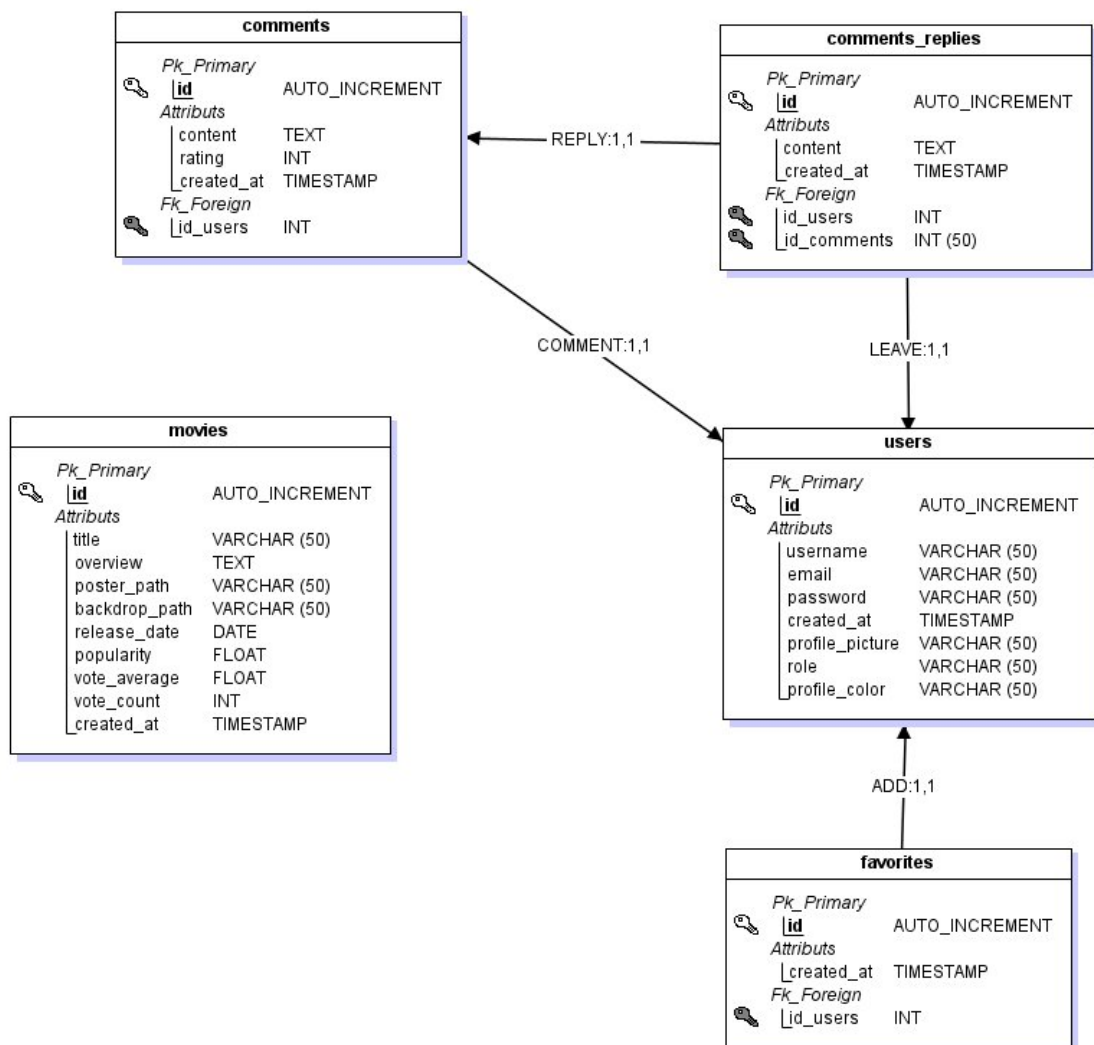
- **users** contient les comptes utilisateurs avec leurs informations et préférences
- **movies** répertorie les films d'animation disponibles
- **comments** stocke les avis laissés sur les films
- **comments\_replies** enregistre les réponses à ces avis
- **favorites** fait le lien entre les utilisateurs et leurs films favoris

Les **relations entre entités** sont désormais représentées par des **clés étrangères** (id\_users, id\_comments...), ce qui garantit l'intégrité des données.

Ce modèle logique m'a permis de préparer efficacement l'implémentation technique dans MySQL (MPD), tout en validant la cohérence des structures de données et leur interconnexion.



## Schéma MLD



## Modèle physique des données

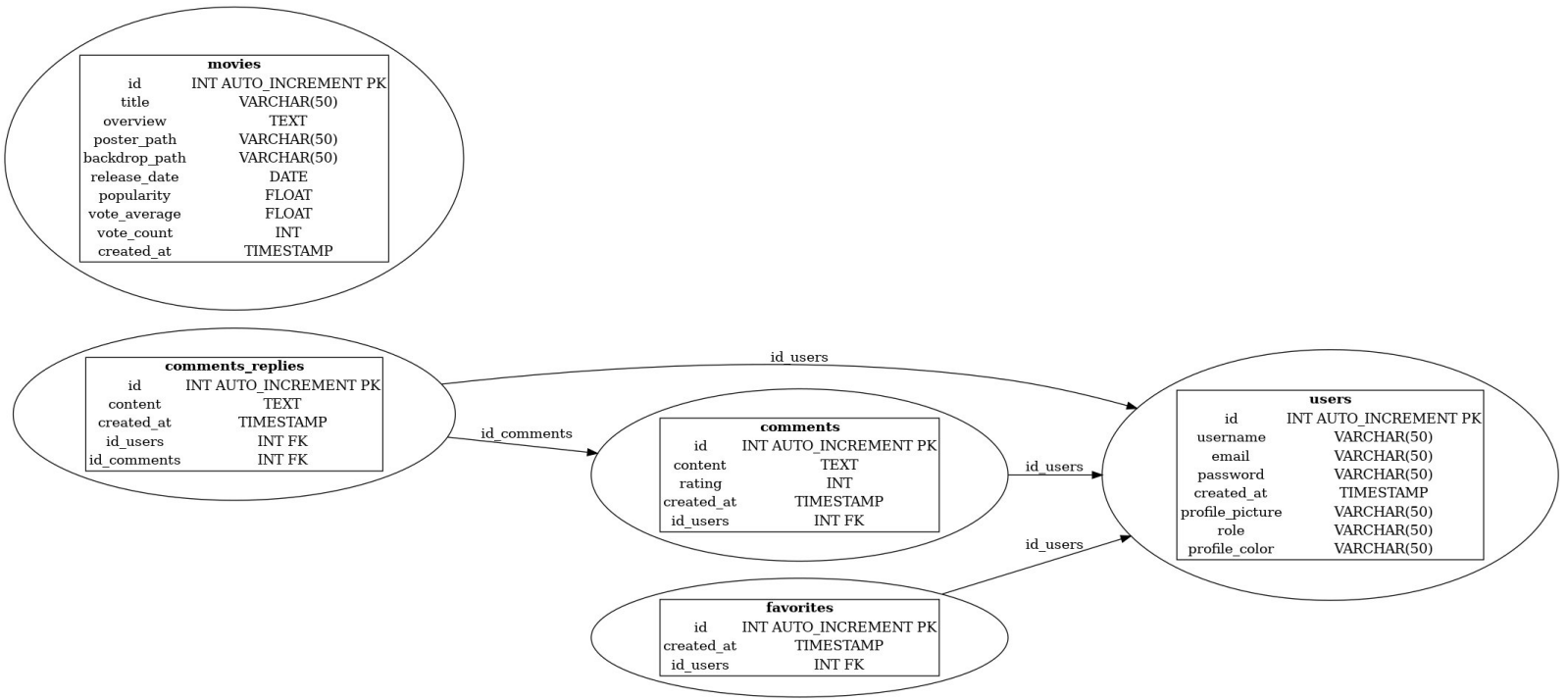
Après avoir validé le modèle logique, j'ai procédé à sa mise en œuvre concrète dans MySQL via l'interface phpMyAdmin. Cette étape correspond au **Modèle Physique des Données (MPD)**, qui décrit précisément la structure de la base telle qu'elle a été créée dans le Système de Gestion de Base de Données (SGBD).

Voici les principales caractéristiques de cette implémentation :

- **Clés primaires auto-incrémentées** sur toutes les entités principales, pour assurer une identification unique des enregistrements.
- **Clés étrangères** clairement définies pour représenter les relations entre les tables (ex : id\_users, id\_comments, etc.), garantissant ainsi l'intégrité référentielle.
- **Types de données adaptés :**
  - VARCHAR pour les textes courts (nom, email, etc.)
  - TEXT pour les contenus longs (commentaires)
  - TIMESTAMP pour les dates
  - FLOAT pour les notes ou les indicateurs de popularité
- **Indexation** sur certains champs fréquemment utilisés dans les requêtes, pour améliorer les performances (par exemple : id\_film, id\_users)
- **Relations many-to-many** implémentées via une table de liaison (favorites) entre les utilisateurs et leurs films favoris.

Cette phase m'a permis de traduire de manière fidèle et efficace la modélisation conceptuelle du projet dans un environnement relationnel fonctionnel et cohérent, tout en respectant les bonnes pratiques d'optimisation et de normalisation.

# Schéma MPD



## CP6 – Développer des composants d'accès aux données SQL et NoSQL

### Contexte

Cette compétence a été mobilisée dans deux contextes :

- D'une part, dans le projet **Animatech**, où l'ensemble des opérations dynamiques repose sur l'échange avec une base de données relationnelle (MySQL).
- D'autre part, dans un projet fictif appelé **FlashNotes**, simulant l'utilisation d'une base NoSQL (MongoDB), afin de comprendre les différences fondamentales entre modèles relationnels et non relationnels.

L'objectif était de savoir construire, manipuler et interroger les données depuis le serveur avec des composants sécurisés et structurés, tout en respectant les principes d'organisation de chaque modèle.

### Connexion SQL Accès aux données relationnelles (Projet Animatech)

Pour établir une connexion sécurisée à la base MySQL, j'ai utilisé **PDO** en PHP, qui offre un bon niveau d'abstraction et prend en charge les requêtes préparées.

Exemple :

```

try {
    $this->db = new PDO(
        "mysql:host=" . DB_HOST . ";dbname=" . DB_NAME . ";charset=utf8",
        DB_USER,
        DB_PASS,
        [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]
    );
} catch(PDOException $e) {
    die("Erreur de connexion : " . $e->getMessage());
}

```

## Requêtes SQL

Dans Animatech, les données utilisateurs, films enregistrés et commentaires sont stockés en base **MySQL**.

L'accès aux données est réalisé via **PDO** (PHP Data Object) pour sécuriser les échanges.

### Sécurisation via requêtes préparées :

```

$stmt = $this->db->prepare("SELECT * FROM users WHERE id = ?");
$stmt->execute([$userId]);
$user = $stmt->fetch(PDO::FETCH_ASSOC);

```

### CRUD utilisé sur plusieurs entités :

- SELECT pour récupérer la liste des films favoris d'un utilisateur
- INSERT pour enregistrer un nouveau commentaire
- DELETE pour retirer un film des favoris
- UPDATE pour éditer des infos de profil (optionnel)

L'utilisation de **fonctions PHP dédiées** permettait de centraliser les appels et d'éviter la duplication de code.

## Connexion MongoDB

La base **MongoDB** a été simulée dans le projet fictif **FlashNotes**, une plateforme de prise de notes par thème. La structure des données y est plus souple, les documents JSON sont stockés dans des collections.

Connexion via **MongoDB Compass** :

- Création d'une base flashnotes
- Ajout d'une collection notes
- Insertion de documents simulés pour tester des filtres dynamiques

```
const { MongoClient } = require('mongodb');
const client = new MongoClient("mongodb://localhost:27017");
await client.connect();
const db = client.db("flashnotes");
const notes = db.collection("notes");
```

## Requêtes MongoDB

Ajout d'une nouvelle note :

```
await notes.insertOne({
  titre: "Comprendre les Promises",
  contenu: "Les Promises permettent de gérer l'asynchrone en JavaScript...",
  tags: ["JavaScript", "Asynchrone"],
  date: new Date("2025-06-30")
});
```

Recherche des notes par tag :

```
const results = await notes.find({ tags: "JavaScript" }).toArray();
```

Filtrage par période :

```
const results = await notes.find({  
  date: { $gte: new Date("2025-06-01"), $lte: new Date("2025-06-30") }  
}).toArray();
```

Mise à jour d'une note :

```
await notes.updateOne(  
  { titre: "Comprendre les Promises" },  
  { $set: { contenu: "Mise à jour du contenu avec exemple de .then().catch()" } }  
);
```

Suppression d'une note :

```
await notes.deleteOne({ titre: "Hooks React" });
```

## CP7 – Développer des composants métier côté serveur

### Contexte

Au cours de mon apprentissage, j'ai été amené à développer la logique métier côté serveur, notamment dans le cadre du projet **Animatech**. Ce projet répertorie des films d'animation à venir, avec un système de gestion des utilisateurs, des avis et des favoris.

Pour ce faire, j'ai conçu des **routes API**, géré les différentes actions serveur (création, lecture, modification, suppression de données), et mis en place une **structure sécurisée** côté back-end en PHP procédural.

### Développement

La logique métier a été répartie dans des fichiers dédiés. Par exemple, j'ai mis en place un système permettant :

- aux utilisateurs de créer un compte,
- de se connecter (avec vérification du mot de passe via `password_verify()`),
- de consulter leurs favoris,
- d'ajouter ou supprimer des films dans leur liste.

#### *Exemple : traitement d'ajout d'un favori*

```
$stmt = $this->db->prepare("INSERT INTO favorites (user_id, movie_id) VALUES (?, ?)");  
$success = $stmt->execute([$userId, $movieId]);
```



## Routes API

J'ai mis en place des **routes REST** simples, bien que le projet ne soit pas basé sur un framework MVC.

Méthode	Route	Action
GET	/movies	Récupérer tous les films
GET	/movies/:id	Récupérer un film spécifique
POST	/favorites/add	Ajouter un film aux favoris
DELETE	/favorites/delete/:id	Supprimer un film des favoris
POST	/comments/add	Ajouter un commentaire à un film

Ces routes étaient déclenchées via des appels AJAX (fetch) en JavaScript côté client.

## Sécurisation

J'ai mis en place plusieurs mesures de sécurité essentielles :

- **Utilisation de requêtes préparées** avec PDO pour éviter les injections SQL.
- **Hash des mots de passe** à l'inscription avec password\_hash().

- **Vérification de session** à chaque action sensible :

```
if (!isset($_SESSION['user']) && !isset($_SESSION['user_id'])) {
    $_SESSION['error'] = "Vous devez être connecté pour ajouter un film aux favoris.";
    header('Location: index.php?action=login');
    exit;
}
```

- **Filtrage et nettoyage** des entrées utilisateur :

```
function clean_input($data) {
    $data = trim($data);
    $data = stripslashes($data);
    $data = htmlspecialchars($data, ENT_QUOTES | ENT_HTML5, 'UTF-8');
    return $data;
}
```

En complément, les routes étaient protégées contre un usage non authentifié. L’affichage dynamique dans Animatech était conditionné à la session active, ce qui permettait d’avoir une interface propre entre visiteurs et utilisateurs connectés.

## CP8 – Documenter le déploiement d’une application dynamique web

### Contexte

Une fois le développement d’un site web terminé, il est essentiel de procéder à son **déploiement** pour le rendre accessible au public. Cela implique de transférer l’application du poste de travail local vers un serveur web distant. Ce processus est indispensable pour tester le site en conditions réelles, recueillir des retours d’utilisateurs, ou encore le mettre à disposition d’un client ou d’un recruteur.

## Pourquoi déployer le projet, pour qui, environnement

Dans le cadre de ma formation DWWM à La Plateforme.io, j'ai déployé le projet **Animatech** afin de :

- permettre une consultation en ligne du site sans avoir besoin d'un environnement de développement local ;
- tester la stabilité de l'application et sa compatibilité en production.

Le déploiement a été effectué dans un environnement **Linux** distant, via **Plesk** comme panneau de gestion, et **FileZilla** pour le transfert FTP.

## Caractéristiques du serveur

**Système** : Ubuntu 20.04 LTS (hébergement mutualisé)

**Serveur web** : Apache 2

**Langage côté serveur** : PHP 8.2

**Base de données** : MySQL 5.7

**Accès FTP** : Oui (via FileZilla)

**Certificat SSL** : Activé via Let's Encrypt

**Nom de domaine** : fournis par l'hébergeur (accès via sous-domaine temporaire)

## Déploiement de l'application

Le processus de déploiement s'est déroulé en plusieurs étapes :

1. **Préparation du dossier local :**
  - Suppression des fichiers inutiles (tests, backup)
  - Configuration des chemins d'accès relatifs
  - Ajout d'un fichier .htaccess pour la réécriture d'URL et la sécurité
2. **Export de la base de données :**
  - Exportation du schéma + contenu avec phpMyAdmin
  - Adaptation du fichier .env ou config.php pour les identifiants de production
3. **Transfert des fichiers :**
  - Connexion FTP au serveur distant
  - Transfert du dossier complet du site vers le répertoire httpdocs sur Plesk

## Déploiement via FileZilla et Plesk

### Transfert avec FileZilla

- Connexion en mode SFTP avec les identifiants fournis
- Glisser-déposer du dossier Animatech côté local vers /httpdocs/distant
- Vérification de la hiérarchie des fichiers (index.php à la racine)

### Configuration via Plesk

- Création d'une base de données MySQL via le panneau Plesk
- Import du fichier .sql via l'interface ou phpMyAdmin intégré
- Edition du fichier config.php :

```
php
CopierModifier
define('DB_HOST', 'localhost');
define('DB_NAME', 'animatech');
define('DB_USER', 'admin');
define('DB_PASS', 'motdepasse');
```

- Attribution des droits 755 aux dossiers, 644 aux fichiers

### Résultat

Une fois le site mis en ligne, j'ai pu vérifier :

- le bon fonctionnement des routes (connexion, ajout de favoris...),
- le chargement dynamique des films via l'API TMDB,
- la compatibilité responsive mobile,
- la persistance des données utilisateur.

## Analyse critique et bilan

Au terme de cette formation, plusieurs points forts et axes d'amélioration se dégagent.

Forces :

- Bonne capacité d'adaptation aux outils et environnements de développement.
- Curiosité technique constante : j'aime chercher, expérimenter, corriger.
- Travail autonome renforcé par des projets concrets.
- Progression significative sur la structuration de code (front et back).

Faiblesses :

- Manque de rigueur sur l'optimisation (temps de chargement, code CSS superflu...).
- Difficultés initiales à bien structurer les bases de données.
- Besoin d'approfondir certains outils avancés (frameworks, tests, CI/CD).

Axes d'amélioration :

- Continuer à pratiquer sur des projets concrets pour gagner en assurance.
- Approfondir la logique algorithmique et les design patterns.
- Approcher des standards professionnels : documentation, sécurité, tests unitaires.

Ce que je retiens :

- La formation m'a permis de passer de la découverte du code à la réalisation de vrais projets concrets.
- J'ai compris l'importance du travail en équipe, des outils collaboratifs et de la rigueur dans le développement.

## Remerciements

Je tiens à remercier tout d'abord **l'équipe pédagogique de La Plateforme**, et en particulier **Nicolas Degabriel**, notre formateur principal, pour son accompagnement rigoureux, sa disponibilité, et ses explications toujours claires et adaptées. Grâce à lui, j'ai pu développer mes compétences techniques tout en gagnant en autonomie et en méthode.

Un grand merci également à **l'Atelier**, l'entreprise qui nous a formés aux réalités du monde professionnel et au fonctionnement concret d'une équipe de production. Cette immersion m'a permis d'adopter une posture de vrai salarié, en m'exerçant à la gestion de projet, à la communication en équipe et au respect des délais.

Je n'oublie pas **mes camarades de promotion**, avec qui j'ai partagé entraide, projets collaboratifs et défis techniques. Leur soutien a rendu cette formation à la fois plus enrichissante et plus humaine.

Je remercie également **les Petits Débrouillards**, qui ont été à l'origine de ma découverte du secteur numérique et m'ont permis de poser mes premiers repères dans ce domaine.

Enfin, je remercie **ma famille et mes proches** pour leur patience, leur encouragement constant et leur soutien moral, essentiels tout au long de ces 16 mois exigeants.

## Conclusion

Ce projet m'a permis de mettre en pratique l'ensemble des compétences acquises durant la formation **Développeur Web et Web Mobile** à La Plateforme.io. À travers plusieurs réalisations concrètes – du site vitrine CV au site dynamique Animatech, en passant par une maquette collaborative sur Figma et un projet NoSQL fictif – j'ai pu expérimenter toutes les étapes de la conception web : de l'installation de l'environnement jusqu'au déploiement en ligne.

Ces projets ont été l'occasion de :

- renforcer ma maîtrise du **développement front-end** (HTML, CSS, JavaScript) avec une première approche du responsive design sur le site CV ;
- approfondir mes compétences **back-end** en manipulant **PHP**, **MySQL**, et l'intégration d'**API externes** ;
- explorer l'univers du **NoSQL** à travers MongoDB, même dans un cadre fictif, pour en comprendre les logiques non relationnelles ;
- participer à une **collaboration en équipe** autour d'un projet client, avec les contraintes que cela implique (temps, communication, adaptation graphique) ;
- apprendre à **documenter et déployer** proprement une application dynamique, en me confrontant aux réalités d'un hébergement distant.

Au-delà des compétences techniques, cette formation et ces projets m'ont permis de mieux comprendre le **cycle de vie complet d'un projet web** : analyse des besoins, conception, développement, test, correction, et mise en production.

Aujourd'hui, je me sens prêt à continuer dans le domaine et surtout de continuer à me former ou à poursuivre mes projets personnels avec davantage d'assurance, d'autonomie, et de méthode. Ce dossier marque la



fin d'une étape, mais surtout le début d'un parcours professionnel dans lequel je compte bien continuer à apprendre, progresser, et construire.

## Lexique / vocabulaire technique

- CRUD : Create, Read, Update, Delete – opérations de base sur une base de données.
- API : Application Programming Interface – permet à deux applications de communiquer.
- Responsive : Adaptabilité d'un site à différents écrans (mobile, tablette, desktop).
- Media Queries : Règles CSS permettant d'adapter le style en fonction de la résolution.
- PDO : PHP Data Objects – méthode sécurisée de communication avec une base de données.
- MVC : Modèle-Vue-Contrôleur – architecture pour séparer les responsabilités dans une application.
- JWT : JSON Web Token – méthode d'authentification sécurisée par token.
- NoSQL : Système de base de données non relationnelle.
- ORM : Object Relational Mapping – outil de liaison entre objets et base de données.
- Figma : outil de design UX/UI en ligne collaboratif.
- Git : système de versionnement pour suivre l'évolution du code.
- Node.js : environnement d'exécution JavaScript côté serveur.
- MongoDB : base de données orientée documents (NoSQL).
- Route API : Point d'entrée dans une API permettant l'accès à des fonctionnalités spécifiques.

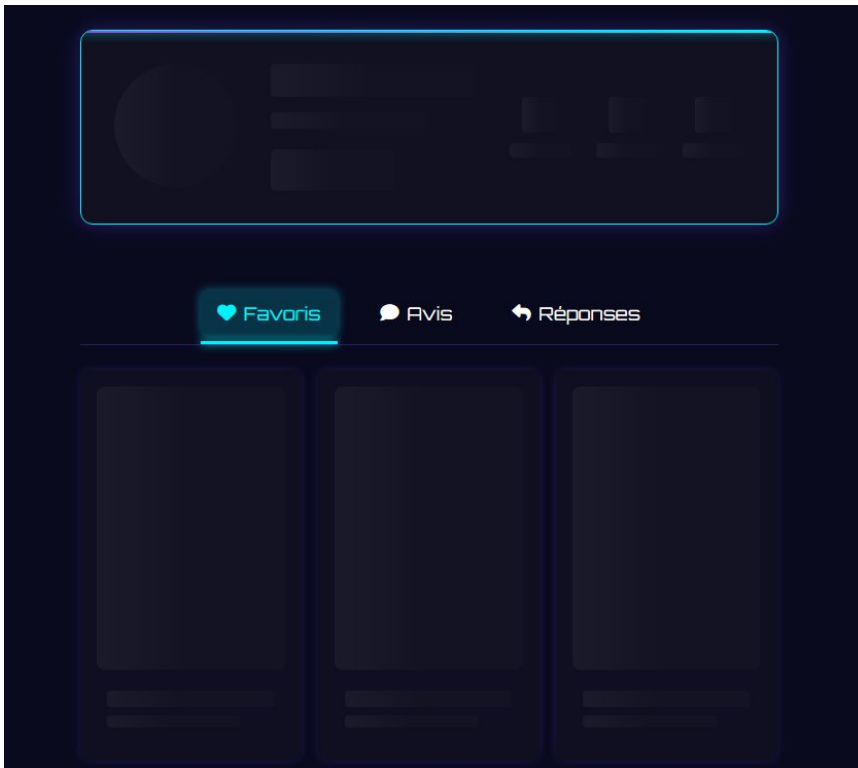
## Annexe

Dans une volonté d'améliorer l'expérience utilisateur, j'ai intégré dans le projet Animatech plusieurs détails subtils mais qui enrichissent l'interface.

Par exemple, l'ajout de skeletons à l'affichage permet d'indiquer visuellement un chargement en cours lorsque les données issues de l'API ou de la base de données ne sont pas encore disponibles. Ce type de feedback visuel est de plus en plus utilisé dans les interfaces modernes, car il améliore la perception de réactivité et rend l'attente plus agréable. C'est une attention aux détails qui témoigne d'une réflexion centrée sur l'utilisateur.

De plus, un travail a été fait sur l'ergonomie de certains éléments interactifs du site, comme les boutons de favoris qui réagissent dynamiquement, ou les fiches films qui se mettent à jour sans rechargement de page grâce à l'implémentation d'AJAX. Ces éléments participent à rendre l'interface plus fluide et intuitive.

Illustration de l'ajout du skeleton :



## Charte Graphique du site Animatech

Le site Animatech adopte un **style visuel néon / cyberpunk**, avec une forte identité graphique marquée par des couleurs lumineuses sur fond sombre. L'objectif était de créer une ambiance immersive et moderne, cohérente avec l'univers de l'animation numérique et la cible jeune du projet.

### Palette de couleurs

Élément	Couleur principale	Usage dans le site
Fond principal	#0C0B1E	Fond très sombre pour contraste élevé
Texte / titres principaux	#00F0FF	Cyan électrique, très visible
Boutons / champs /	#AE4BFF	Violet néon, utilisé dans les

Élément	Couleur principale	Usage dans le site
accents		formulaire
Étoiles de notation	#8329E6	Violet profond pour les badges de score
Textes secondaires	#FFFFFF et #B1B1B1	Blanc pur ou gris clair selon le contexte
Bordures néon / glow	#00FFFF + ombre	Apporte du relief lumineux sur les blocs

Les effets de **glow (ombres portées fluo)** sont utilisés sur les encadrés et les boutons pour renforcer la sensation de profondeur et de modernité.

## Typographie

- **Police utilisée** : Orbitron
- **Origine** : Google Fonts
- **Style** : Futuriste, techno, anguleuse
- **Raisons du choix** :
  - Parfaitement adaptée à l'univers "digital / animation".
  - Très lisible en titre.
  - Esthétique marquée sans être difficile à lire.

## Intentions de design

- **Ambiance générale** : immersive, digitale, inspirée des interfaces de jeux ou dashboards futuristes.
- **Objectif UX** : lisibilité claire, bonne hiérarchie visuelle, contraste fort pour l'accessibilité.
- **Organisation** :
  - Interface en **mode sombre** pour le confort visuel.
  - Utilisation de **cartes** (films, profils, etc.) pour structurer les contenus de manière modulaire.
  - Effets de **hover interactifs** et de **transitions douces** pour enrichir l'expérience utilisateur.