



MINI PROJET / STAGE

2023/2024

Développement d'une Plateforme de Streaming avec Déploiement Cloud

Ingénierie Informatique et Réseaux

Réalisé par

Soulaimane Ouhmida

Encadré par :

Mme. BOUSBAA Zineb, EMSI

M. Azeddine Lakhal, Concentrix

M. Charif Abdelmajid, Concentrix

Dedicace

A mes parents, dont le soutien et les encouragements indéfectibles ont été mon pilier tout au long de mes études et de ce stage. À mes sœurs, pour leur soutien inconditionnel et leur présence réconfortante à chaque étape de ma vie. A mes amis, pour leur soutien indéfectible et leurs précieux encouragements. À **Monsieur Lakhal, Monsieur Charif et Monsieur Ait Ounajjar** pour leur patience, leurs conseils avisés et la confiance qu'ils m'ont accordée. Ce rapport est dédié à tous ceux qui ont contribué directement ou indirectement à mon expérience de formation.

Je tiens à remercier sincèrement mon cousin **Saad Riadi** pour son aide précieuse dans la recherche de mon stage. Grâce à ses conseils et à son soutien, j'ai pu vivre une expérience professionnelle enrichissante, qui a grandement contribué à mon développement personnel et professionnel.

Enfin, j'adresse mes sincères remerciements à toute l'équipe de **Concentrix** pour m'avoir accueilli chaleureusement et m'avoir offert un environnement enrichissant pour apprendre et grandir.

Remerciement

Je tiens tout d'abord à exprimer ma profonde gratitude à l'équipe de **Concentrix**, particulièrement à **M. Charif** pour son soutien constant, ses conseils avisés et la confiance qu'il m'a accordée tout au long de mon stage. Ses précieux enseignements et son accompagnement ont été essentiels pour mon développement professionnel.

Mes remerciements vont **M. Lakhal**, pour leur accueil chaleureux, leur collaboration et leur volonté de partager leurs connaissances. Leur expertise m'a permis d'acquérir une compréhension approfondie du domaine et de contribuer de manière significative aux projets de l'entreprise.

Je souhaite exprimer ma reconnaissance envers mes collègues, qui ont rendu mon expérience de travail enrichissante et agréable. Leur soutien et leur camaraderie ont contribué grandement à mon épanouissement professionnel au sein de l'équipe.

Enfin, je suis reconnaissant envers ma famille et mes amis pour leur soutien inconditionnel et leurs encouragements constants tout au long de cette période.

Résumé

À une époque où les services de streaming numérique sont omniprésents, des plateformes telles que **Netflix** et **Disney+** incarnent parfaitement la fusion entre technologie avancée et divertissement. Ces entreprises déploient des technologies de pointe pour offrir du contenu visuel de qualité à un large public.

Ce projet ambitieux a pour objectif de recréer l'infrastructure complète de Netflix, comprenant une interface utilisateur attrayante, des serveurs puissants, et une architecture de base de données complexe qui soutient une expérience utilisateur fluide et immersive.

L'interface utilisateur est conçue pour capter et maintenir l'attention des utilisateurs, tandis que les systèmes backend sophistiqués gèrent la diffusion du contenu et l'administration des comptes. Chaque composant de ce projet vise à reproduire les fonctionnalités distinctives qui ont fait la renommée de Netflix. En réalisant cette réplique détaillée, le projet met en lumière non seulement des compétences techniques impressionnantes, mais il explore également les défis de l'évolutivité, de la distribution de contenu, et du design centré sur l'utilisateur dans le paysage moderne des médias.

Ce travail nous conduit à décortiquer et à recréer la magie qui a permis à Netflix de devenir un acteur incontournable du divertissement mondial. En approfondissant ces aspects, le projet aspire à offrir une compréhension approfondie des technologies et des stratégies qui sous-tendent le succès des géants du streaming.

Mots clés :

Streaming vidéo / Optimisation de la recherche / Stockage Big Data / Faible latence / Sécurité des données / Cloud Computing / Services AWS / Architecture de microservices / Expérience utilisateur / UX / Recommandation de contenu / Bases de données MySQL / Cassandra / Bases de données relationnelles / MySQL / Développement d'API / Spring Boot / React / Frontend / Évolutivité / Équilibrage de charge / Optimisation des performances / Recommandations.

Abstract

In an era where digital streaming services are ubiquitous, platforms like **Netflix** and **Disney+** epitomize the fusion of advanced technology and entertainment. These companies deploy cutting-edge technologies to deliver quality visual content to a wide audience.

This ambitious project aims to recreate Netflix's entire infrastructure, including a visually appealing user interface, powerful servers, and a complex database architecture that supports a fluid and immersive user experience.

The user interface is designed to capture and maintain users' attention, while sophisticated backend systems handle content delivery and account administration. Each component of this project aims to replicate the distinctive features that Netflix is known for. By creating this detailed replication, the project not only highlights impressive technical skills, but also explores the challenges of scalability, content distribution, and user-centric design in the modern media landscape.

This work leads us to unpack and recreate the magic that has allowed Netflix to become a global entertainment powerhouse. By delving into these aspects, the project aims to provide an in-depth understanding of the technologies and strategies that underpin the success of streaming giants.

Keywords:

Video Streaming / Search Optimization / Big Data Storage / Low Latency / Data Security / Cloud Computing / AWS Services / Microservices Architecture / User Experience / UX / Content Recommendation / NoSQL Databases / Cassandra / Relational Databases / MySQL / API Development / Spring Boot / React / Frontend / Scalability / Load Balancing / Performance Optimization / Recommendations.

Glossaire

EMSI	Ecole Marocaines des sciences de l'ingénieur
API	Application Programming Interface
UML	Unified Modeling Language
AWS	Amazon Web Services
AWS S3	Scalable Storage Service
AWS EC2	Elastic Compute Cloud
AWS RDS	Relational Database Service
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
Gantt	Graphical representation of activity against time
MySQL	Michael Widenius's Structured Query Language

Liste des Figures

Figure 1.1: Logo de Concentrix	16
Figure 1.2: Emplacement du Concentrix dans le monde	17
Figure 1.3: Les valeurs de concentrix	18
Figure 1.4: Concentrix Maroc.....	20
Figure 1.5: Logo Webhelp	22
Figure 1.6: Emplacement du Webhelp dans le monde	23
Figure 1.7: Les valeurs de webhelp	24
Figure 1.8: Concentrix + Webhelp	25
Figure 9: Logo de Movify	27
Figure 10: Diagramme de Gantt.....	30
Figure 11: Diagramme du cloud.....	39
Figure 12: Logo de React JS.....	45
Figure 13: Logo de JavaScript	45
Figure 14: Logo de HTML.....	46
Figure 15: Logo de CSS	46
Figure 16: Logo de Tailwind	46
Figure 17: Logo de Java	46
Figure 18: Logo de Spring Boot.....	47
Figure 19: Logo de MySQL	47
Figure 20: Logo de Cassandra	47
Figure 21: Logo de Redis.....	47
Figure 22: Logo de AWS	48
Figure 23: Logo de Git	48
Figure 24: Logo de Jira.....	49
Figure 25: la page d'accueil de Movify	52
Figure 26: La page de lecteur vidéo.....	53
Figure 27: Architecture du backend.....	54
Figure 28: tableau de bord Stripe montrant les transactions et abonnements.....	54
Figure 29: Schéma du processus de scraping	55
Figure 30: Serveur Movify 1 dans AWS EC2	57
Figure 31: Images des films dans AWS S3.....	57
Figure 32: Mes certifications en cloud computing (AWS)	61
Figure 33: Structure du frontend	65
Figure 34: Page d'accueil	66
Figure 35: Page d'inscription.	67
Figure 36: Page de création du profil.	68
Figure 37: Page de Connexion (Login).	69
Figure 38: Page de connexion.....	70
Figure 39: Page de choisir le profil.....	71
Figure 40: Page d'accueil	71
Figure 41: Choisir (film / serie)	72

Figure 42: Page (film / serie)	73
Figure 43: Page d'un film	74
Figure 44: Page d'un film apres ajouter aux favoris.....	74
Figure 45: Watch list.....	75
Figure 46: Comment fonctionne stripe ?	75

Liste des tableaux

Tableau 1.1: Fiche technique du Concentrix	17
Tableau 1.2: Secteurs d'activité de concentrix	19
Tableau 1.3: Fiche technique du Webhelp	23
Tableau 1.4: Les secteurs d'activité de webhelp	24
Tableau 2.1: Les besoins clés de l'utilisateur final	Erreur ! Signet non défini.
Tableau 2.3: Les besoins non fonctionnelles.	Erreur ! Signet non défini.
Tableau 5: Stack technologique frontend (netflix)	Erreur ! Signet non défini.
Tableau 6: Stack technologique backend (netflix)	Erreur ! Signet non défini.
Tableau 7: Stack technologique bases de données (netflix)	Erreur ! Signet non défini.
Tableau 8: Stack technologique big data (netflix)	Erreur ! Signet non défini.
Tableau 9: Stack technologique devops (netflix)	Erreur ! Signet non défini.
Tableau 13: les types de bases de données.....	Erreur ! Signet non défini.
Tableau 14: Technologies utilisées dans backend.	Erreur ! Signet non défini.
Tableau 15: Technologies utilisées dans frontend.	65
Tableau 16: Les principaux composants cloud	Erreur ! Signet non défini.
Tableau 17: Les services AWS Utilisés.....	Erreur ! Signet non défini.
Tableau 18: Les étapes de déploiement du Frontend (React)	Erreur ! Signet non défini.
Tableau 19: Les étapes de déploiement du Backend (Spring Boot)	Erreur ! Signet non défini.
Tableau 20: Les étapes de la configuration des Bases de Données	Erreur ! Signet non défini.
Tableau 21: Sécurité et Conformité	Erreur ! Signet non défini.

Table des matières

Dedicace	2
Remerciement.....	3
Résumé	4
Abstract	5
Glossaire	6
Liste des Figures	7
Liste des tableaux	9
Table des matières	10
Introduction Générale	13
Contexte	13
Problématique	13
Objectifs	14
Structure du Rapport	14
Chapitre 1 : Présentation de l'organisme d'accueil.....	15
Introduction	16
I. Concentrix	16
Présentation.....	16
Les valeurs de Concentrix	18
Secteurs d'activité	18
Concentrix Maroc.....	20
II. Webhelp	22
Presentaion	22
Les valeurs de Webhelp	24
Secteurs d'activité	24
III. Concentrix + WebHelp	25
Conclusion.....	25
Chapitre 2 : Contexte du projet	26
Introduction	27
Présentation du projet.....	27
Cible de la solution	27
L'équipe de travail.....	28
Problématique et solution proposée	28

Besoins fonctionnels	28
Besoins non fonctionnels	28
Analyse de risque.....	29
Facteurs de succès.....	29
Planification du projet	29
Conclusion.....	30
Chapitre 3 : Analyse et Conception.....	31
Introduction	32
I. Modélisation UML.....	32
Diagrammes de cas d'utilisation	32
Diagrammes de classes.....	34
Diagrammes de séquence	35
II. Diagramme d'architecture Cloud	38
Qu'est-ce que le cloud computing ?	38
Conclusion.....	40
Chapitre 4 : Etude Technique	41
Introduction	42
I. Présentation de l'environnement de travail.....	42
II. Comparaison entre les solutions techniques existantes	43
1. Backend : Spring Boot vs Node.js	43
2. Frontend : React vs Angular.....	43
3. Base de données : MySQL vs PostgreSQL	43
4. Stockage des données volumineuses : Cassandra vs MongoDB	44
5. Caching : Redis vs Memcached.....	44
6. Hébergement : AWS vs Google Cloud	44
III. Technologies et outils utilisés	45
1. Design	45
2. Frontend	45
3. Backend.....	46
4. Databases.....	47
5. Scrapping.....	48
6. Cloud	48
7. Gestion du Code Source.....	48

8. Outils de Collaboration	49
Conclusion.....	49
Chapitre 5 : Mise en œuvre et Réalisation.....	50
Introduction	51
Implémentation et solution développée	51
I. Frontend : Développement de l'interface utilisateur avec React	51
II. Backend : API REST avec Spring Boot.....	53
III. Scraping de données avec Python	55
IV. Bases de données	56
V. Infrastructure Cloud	56
Conclusion.....	58
Chapitre 6 : Bilan Introduction.....	59
I. Bilan professionnel	60
II. Bilan technique.....	60
III. Bilan personnel.....	62
Conclusion.....	62
Conclusion Générale et Perspectives.....	63
Annexe	65
I. Développement Frontend	65
Technologies Utilisées	65
Développement des Fonctionnalités Principales	66
II. Interaction Frontend-Backend	70
Exemple d'Interaction Utilisateur.....	70
Récupération des Données.....	72
Mise à jour en Temps Réel	74
III. Processus de payment	75
How does Stripe payment processing work?	75
Références.....	79

Introduction Générale

Lors de mon stage d'été, j'ai entrepris de développer une application qui reproduit les principales fonctionnalités de Netflix, une plateforme de streaming vidéo de renommée mondiale. Ce projet ambitieux vise à intégrer la gestion des utilisateurs, le streaming de vidéos, et une interface utilisateur intuitive et engageante.

Contexte

L'industrie du streaming vidéo connaît une croissance spectaculaire, portée par des plateformes comme Netflix, qui redéfinissent la consommation de médias. Cette transformation repose sur des progrès technologiques majeurs dans le développement web, la gestion de bases de données, et les services cloud, qui permettent une diffusion fluide et accessible de contenu audiovisuel.

En 2023, 92,3 % des internautes regardaient des vidéos en ligne, avec une forte préférence pour les vidéos musicales, visionnées chaque semaine par un internaute sur deux. Le marché de la vidéo à la demande se diversifie rapidement, offrant une abondance de films et de séries qui plonge parfois les utilisateurs dans l'indécision. En effet, près d'un tiers des spectateurs ont passé plus de 10 minutes à chercher un film sur une plateforme de streaming, soit deux fois plus qu'en 2022. Cette tendance reflète une évolution significative des habitudes de visionnage, avec 44,2 % du temps total consacré à la télévision dédiée aux services de streaming au troisième trimestre 2023, contre 31,5 % cinq ans plus tôt.

Des plateformes comme Netflix, Disney+, Amazon Prime et YouTube dominent ce marché, Netflix s'étant rapidement imposé comme l'un des services de streaming les plus utilisés à l'échelle mondiale. En 2024, le chiffre d'affaires global du secteur du streaming vidéo devrait atteindre plus de 108 milliards de dollars, confirmant l'importance croissante de ces services dans l'écosystème médiatique [1].

Problématique

Dans un contexte où les services de streaming vidéo dominent les habitudes de consommation médiatique, la création d'une application imitant les principales fonctionnalités de Netflix présente de nombreux défis techniques et fonctionnels. Comment concevoir une plateforme capable de gérer efficacement de grandes quantités de données audiovisuelles, tout en assurant une expérience utilisateur fluide et réactive, similaire à celle offerte par des géants comme Netflix ? Il s'agit également de garantir la sécurité des informations personnelles des utilisateurs, ainsi que la scalabilité et la disponibilité du service dans un environnement en perpétuelle évolution.

Objectifs

L'objectif principal de ce projet est de recréer une plateforme de streaming vidéo, en s'inspirant des fonctionnalités essentielles de Netflix. Pour y parvenir, plusieurs sous-objectifs techniques sont définis :

- **Développer une interface utilisateur moderne et intuitive** : Créer une interface réactive et engageante, capable de reproduire l'expérience utilisateur fluide et immersive offerte par les grandes plateformes de streaming.
- **Construire un serveur web robuste** : Mettre en place un serveur web performant, accompagné d'une API qui gère les interactions utilisateurs, l'authentification, ainsi que la gestion des vidéos et des profils.
- **Concevoir une base de données optimisée** : Élaborer une base de données structurée pour stocker et récupérer rapidement des informations volumineuses, comme les vidéos, les préférences des utilisateurs, et les historiques de visionnage.
- **Intégrer les services cloud** : Utiliser des services cloud pour garantir la scalabilité, la haute disponibilité, et l'accès continu de l'application, en tirant parti des solutions de stockage et de distribution de contenu.
- **Assurer la sécurité des données** : Mettre en œuvre des mesures de sécurité pour protéger les informations sensibles des utilisateurs, telles que les données personnelles et les moyens de paiement.

Structure du Rapport

Ce rapport est organisé en plusieurs chapitres, chacun décrivant une étape spécifique du projet.

1. Présentation de l'organisme d'accueil ;
2. Contexte du projet ;
3. Analyse et Conception ;
4. Étude Technique ;
5. Mise en œuvre et Réalisation ;
6. Bilan .

CHAP1 : Présentation de l'organisme d'accueil

Introduction

Ce premier chapitre abordera la présentation de l'organisme d'accueil, l'entreprise Concentrix, ses domaines d'activités et ses métiers, et son rapprochement avec Webhelp.

I. Concentrix

Présentation



Figure 1.1: Logo de Concentrix

Concentrix Corporation est l'un des principaux fournisseurs mondiaux de solutions et de technologies d'expérience client (CX), améliorant les performances commerciales de certaines des meilleures marques au monde, dont plus de 100 clients du Fortune Global 500 et plus de 125 clients de la nouvelle économie.

Création	Concentrix a été fondée en 1983.
Forme Juridique	Société anonyme (corporation) cotée en bourse.
Slogan	We deliver extraordinary customer experiences.
Présence Mondiale	plus de 40 pays à travers le monde.
Siège Sociale	Fremont, Californie, États-Unis.
Directeur Général (CEO)	Chris Caldwell
Produits	<ul style="list-style-type: none"> • Gestion de l'expérience client • Externalisation des processus métiers (BPO) • Services informatiques • Solutions de technologie • Analytique et intelligence artificielle • Services de gestion des interactions clients (centres de contact, support technique, etc.)
Filiales	<ul style="list-style-type: none"> • Concentrix Catalyst • TigerSpike • Minacs

Effectif	Environ 290,000 employés à travers le monde.
Site Web	concentrix.com
Chiffre d’Affaire	Environ 5,6 milliards de dollars en 2021.

Tableau 1.1: Fiche technique du Concentrix

La figure suivante représente les régions de présence de Concentrix au monde et la répartition de ses consultants :

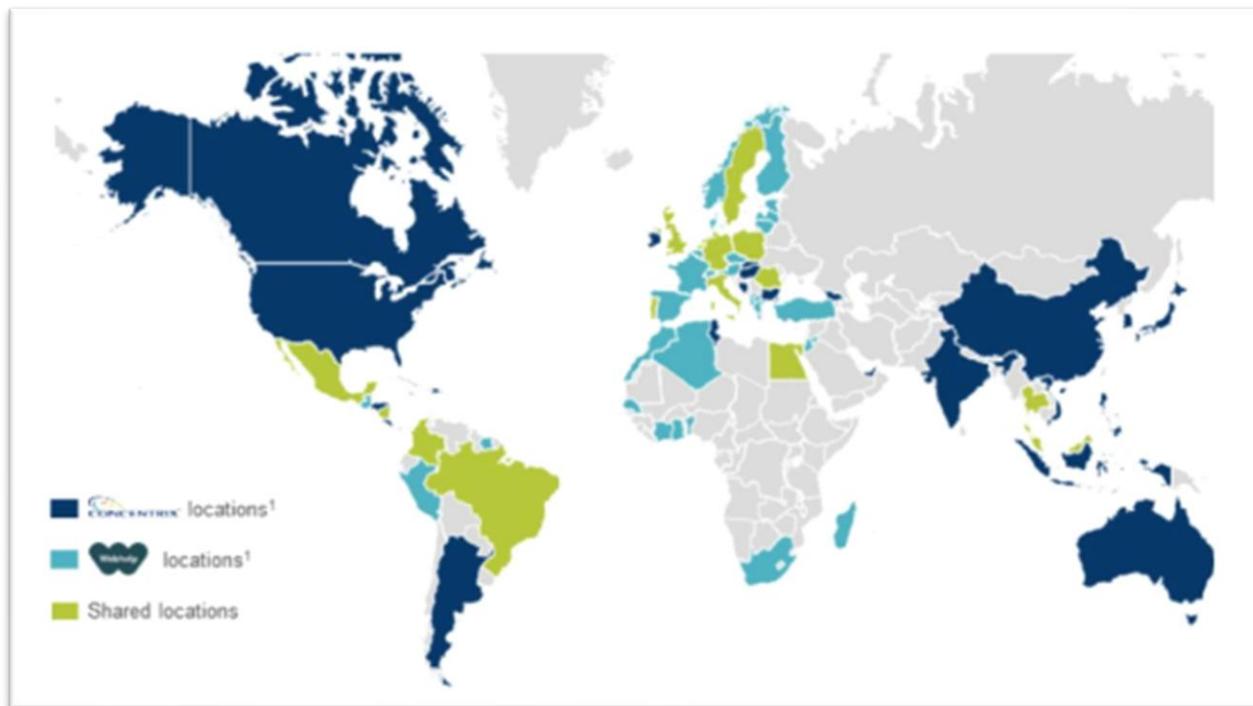


Figure 1.2: Emplacement du Concentrix dans le monde

Les valeurs de Concentrix

Les valeurs et la culture de concentrix reposent sur le respect d'un soutien individuel et mutuel entre collaborateurs, et d'une véritable collaboration. Les sept valeurs de Concentrix sont :



Figure 1.3: Les valeurs de concentrix

Secteurs d'activité

Concentrix opère dans plusieurs secteurs d'activité, fournissant une gamme diversifiée de services pour répondre aux besoins spécifiques de chaque industrie. Voici quelques-uns des principaux secteurs d'activité de Concentrix :

Technologie	<ul style="list-style-type: none"> • Services de support technique • Gestion des relations clients pour les entreprises technologiques • Solutions de service après-vente
Télécommunications	<ul style="list-style-type: none"> • Assistance clientèle pour les opérateurs de téléphonie mobile et les fournisseurs d'accès internet • Gestion des ventes et du support technique
Services financiers	<ul style="list-style-type: none"> • Services de gestion des comptes • Support client pour les banques et les compagnies d'assurance • Services de recouvrement et de gestion des fraudes
Santé	<ul style="list-style-type: none"> • Services de support aux patients • Gestion des réclamations d'assurance santé • Services de télésanté et de soins à distance

Commerce de détail et e-commerce	<ul style="list-style-type: none"> Gestion des commandes et des retours Support client pour les boutiques en ligne et les détaillants traditionnels Solutions de marketing numérique
Automobile	<ul style="list-style-type: none"> Services de support pour les constructeurs automobiles Gestion des réclamations de garantie Support technique pour les véhicules connectés
Voyages et loisirs	<ul style="list-style-type: none"> Services de réservation et de support client pour les compagnies aériennes, les hôtels et les agences de voyages Gestion des programmes de fidélité
Services publics et gouvernement	<ul style="list-style-type: none"> Assistance clientèle pour les services publics Solutions de gestion des citoyens pour les agences gouvernementales
Éducation	<ul style="list-style-type: none"> Services de support pour les établissements d'enseignement Gestion des inscriptions et des programmes éducatifs en ligne
Médias et divertissement	<ul style="list-style-type: none"> Support client pour les plateformes de streaming et de médias Gestion des abonnements et des services à la demande

Tableau 1.2: Secteurs d'activité de concentrinx

Concentrix Maroc

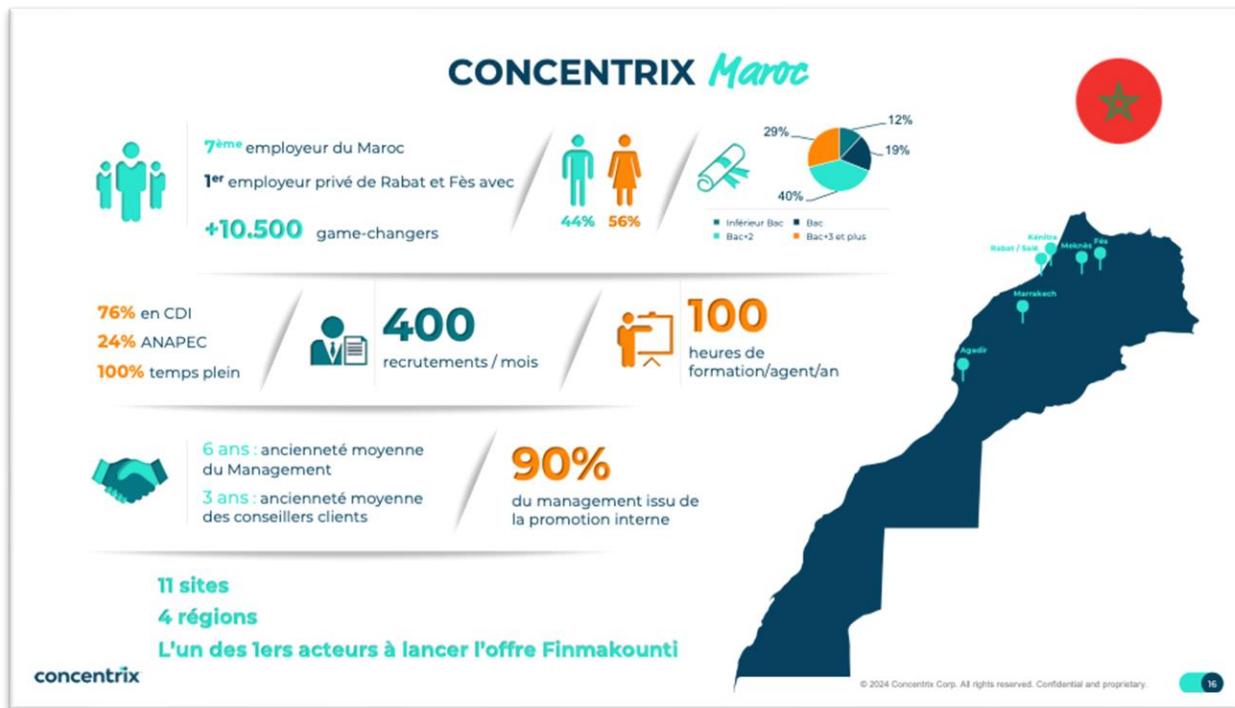


Figure 1.4: Concentrix Maroc

La figure suivante est la représentation schématique des liens hiérarchiques de Concentrix Maroc :

II. Webhelp

Présentation



Figure 1.5: Logo Webhelp

Webhelp est une entreprise française d'externalisation de la gestion de l'expérience client et des processus métier. Elle a été fondée en 2000 et son siège est situé à Paris en France.

Concrètement, ce groupe dirige des centres d'appels qui proposent à des entreprises des prestations de ligne directe, de télémarketing, de traitement de courriers et d'e-mails.

Création	Webhelp a été fondée en 2000.
Forme Juridique	Société par actions simplifiée (SAS).
Slogan	Making business more human.
Présence Mondiale	plus de 50 pays avec plus de 200 sites à travers le monde.
Siège Sociale	161 Rue de Courcelles, 75017 Paris, France.
Direction	<ul style="list-style-type: none"> • Co-fondateurs : Frédéric Jousset et Olivier Duha • CEO : Olivier Duha
Produits	<ul style="list-style-type: none"> • Gestion de l'expérience client • Externalisation des processus métiers (BPO) • Services de centres de contact • Services de support technique et d'assistance • Solutions de transformation numérique • Services de modération de contenu • Services de gestion des réseaux sociaux
Filiales	<ul style="list-style-type: none"> • Webhelp Payment Services • Webhelp Enterprise Sales Solutions

	<ul style="list-style-type: none"> • Webhelp Digital Consulting
Effectif	Environ 100,000 employés à travers le monde.
Site Web	webhelp.com
Chiffre d'Affaire	Environ 2,5 milliards euros en 2021.

Tableau 1.3: Fiche technique du Webhelp

La figure suivante représente les régions de présence de Concentrix au monde et la répartition de ses consultants :



Figure 1.6: Emplacement du Webhelp dans le monde

Les valeurs de Webhelp

Webhelp, dans l'ensemble du groupe est attaché à ses 5 valeurs : la reconnaissance, l'unité, l'engagement, l'exemplarité, le Wahou.



Figure 1.7: Les valeurs de webhelp

Secteurs d'activité

Webhelp opère principalement dans les secteurs suivants :

Expérience client	Fourniture de solutions pour améliorer l'interaction client à travers différents canaux comme le service client, le support technique, et la gestion des relations client.
Externalisation des processus métier (BPO)	Gestion déléguée de processus métier tels que la gestion des ressources humaines, la comptabilité, et d'autres fonctions administratives.
Conseil et technologie	Offre de conseils stratégiques et de solutions technologiques pour optimiser les opérations commerciales et améliorer l'efficacité opérationnelle.
Santé et bien-être	Services spécialisés dans le domaine de la santé, y compris la gestion des dossiers médicaux et l'assistance aux patients.

Tableau 1.4: Les secteurs d'activité de webhelp

III. Concentrix + WebHelp



Figure 1.8: Concentrix + Webhelp

Le 25 septembre 2023, Concentrix a annoncé avoir finalisé son rapprochement avec Webhelp et que l'intégration des deux sociétés est en cours. Pendant que la société issue de la fusion finalise son nom permanent, elle opérera sous le nom commercial Concentrix + Webhelp.

Conclusion

Dans ce chapitre, le cadre général du projet a été décrit. Après avoir présenté l'organisme d'accueil, présenté le projet, nous avons décrit par le périmètre de ce stage, la méthodologie adoptée pour atteindre la bonne conduite du projet. Le chapitre suivant sera consacré à l'étude fonctionnelle et technique du projet.

CHAP 2 : Contexte du projet

Introduction

Ce chapitre présente le cadre dans lequel le projet « **Movify** » a été développé. Il aborde les différents aspects du projet, y compris les modules et fonctionnalités principales, la structure de l'équipe, les besoins fonctionnels et non fonctionnels, ainsi que l'analyse des risques. Ce contexte est essentiel pour comprendre les défis techniques et organisationnels rencontrés tout au long du développement, ainsi que les facteurs de succès qui ont permis d'assurer l'avancement du projet.



Figure 9: Logo de Movify

Présentation du projet

Le projet **Movify** a pour objectif de reproduire les fonctionnalités principales de la plateforme de streaming. Le projet est structuré autour de plusieurs modules :

- **Frontend** : Développé en React, il permet une navigation fluide entre les différentes pages (films, séries, profil, etc.).
- **Backend** : Conçu en Spring Boot, il gère les requêtes utilisateur, l'authentification, la gestion des données et l'interaction avec les bases de données.
- **Bases de données** : MySQL, Redis, et Cassandra sont utilisées pour stocker les informations des utilisateurs, des contenus et de l'historique de visionnage.
- **Paiements** : Intégration avec Stripe pour les abonnements.
- **Cloud** : AWS (EC2, S3, CloudFront, etc.) est utilisé pour le déploiement et l'hébergement de l'application.

Cible de la solution

L'application cible les utilisateurs de services de streaming, amateurs de films et séries, et propose une interface intuitive et fluide permettant de profiter d'une expérience de visionnage optimale.

L'équipe de travail

Le projet a été réalisé dans le cadre d'un stage, ce qui signifie que toutes les tâches ont été menées à bien par une seule personne. Toutefois, dans un projet à plus grande échelle, des équipes spécialisées pourraient être mises en place :

- **Frontend** : Développement et design de l'interface utilisateur ;
- **Backend** : Création des services API et gestion des données ;
- **DevOps** : Déploiement et gestion des infrastructures cloud ;
- **Base de données** : Conception et optimisation des bases de données ;
- **Sécurité** : Mise en place de stratégies de protection des données.

Problématique et solution proposée

Le principal défi de ce projet était de créer une plateforme capable de gérer de grandes quantités de données et de répondre à de nombreux utilisateurs simultanément, tout en assurant une bonne performance. La solution proposée repose sur une architecture en microservices, une infrastructure cloud scalable (AWS), et des bases de données performantes. Mon rôle dans ce projet a été la réalisation complète de l'application, de la conception à l'implémentation.

Besoins fonctionnels

- Authentification sécurisée des utilisateurs;
- Navigation et recherche de films/séries ;
- Streaming vidéo fluide;
- Gestion des abonnements et paiements ;
- Recommandations basées sur les préférences et l'historique de visionnage.

Besoins non fonctionnels

- **Performance** : Temps de réponse rapide et faible latence ;
- **Sécurité** : Protection des données des utilisateurs ;
- **Scalabilité** : Capacité à supporter une augmentation du nombre d'utilisateurs sans dégradation de service ;

- **Fiabilité** : Disponibilité continue de l'application avec un minimum d'interruptions.

Analyse de risque

Les risques liés à la non-réalisation du projet incluent :

- **Problèmes matériels** : Défaillance des serveurs ou infrastructure cloud ;
- **Problèmes logiciels** : Bugs critiques ou incompatibilités ;
- **Retard dans la communication** : Mauvaise coordination ou partage d'informations entre les membres d'une éventuelle équipe ;
- **Dépendances externes** : Problèmes avec les fournisseurs de services (AWS, Stripe).

Facteurs de succès

Les facteurs qui ont contribué au succès du projet incluent :

- **Une architecture flexible et scalable** qui permet de répondre aux besoins en croissance de l'application ;
- **Un focus sur l'expérience utilisateur** avec une interface fluide et intuitive ;
- **La sécurité des données** et l'intégration des systèmes de paiement conformes ;
- **L'utilisation de services cloud** pour assurer une haute disponibilité et performance.

Planification du projet

Le projet a été planifié selon un schéma de développement agile avec des sprints de deux semaines. Le diagramme de Gantt, couramment utilisé en gestion de projet, est l'un des outils les plus efficaces pour représenter visuellement l'état d'avancement des différentes activités (tâches) qui constituent un projet. La colonne de gauche du diagramme énumère toutes les tâches à effectuer, tandis que la ligne d'en-tête représente les unités de temps les plus adaptées au projet (jours, semaines, mois etc.). Chaque tâche est matérialisée par une barre horizontale, dont la position et la longueur représentent la date de début, la durée et la date de fin [2].

Le diagramme de Gantt ci-dessous détaille les étapes de développement :

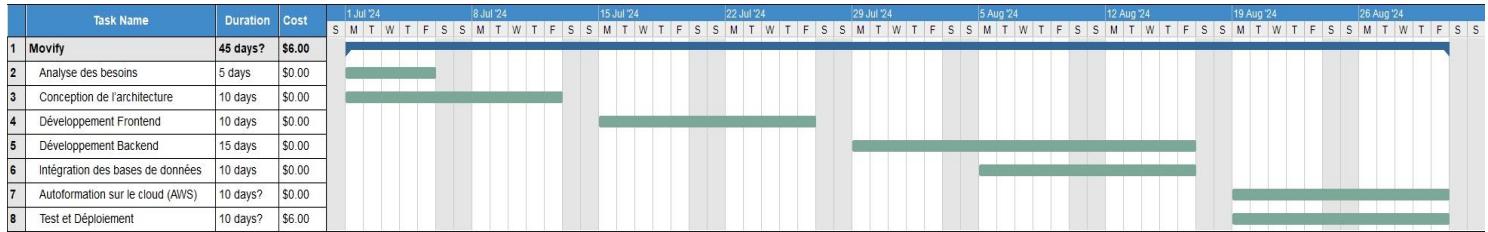


Figure 10: Diagramme de Gantt

Conclusion

Ce chapitre a fourni une vue d'ensemble du projet **Movify**, de la présentation des modules à l'analyse des besoins fonctionnels et non fonctionnels, en passant par la gestion des risques. Cette analyse permet de mieux comprendre les enjeux du projet ainsi que les éléments essentiels pour en assurer le succès.

CHAP 3 : Analyse et Conception

Introduction

Ce chapitre présente l'analyse et la conception de l'application Movify à travers plusieurs outils de modélisation. Utilisant le langage UML ainsi qu'un diagramme d'architecture cloud, ce chapitre détaille les différents composants du système, leurs interactions et la manière dont l'application est déployée dans un environnement cloud. Ces représentations permettent de mieux comprendre la structure technique et les processus clés du projet, assurant ainsi une architecture bien définie avant le développement.

I. Modélisation UML

Un diagramme UML est un moyen de visualiser des systèmes et des logiciels à l'aide du langage de modélisation unifié (UML). Les ingénieurs logiciels créent des diagrammes UML pour comprendre la conception, l'architecture du code et la mise en œuvre proposée de systèmes logiciels complexes [3].

Les diagrammes utilisés sont :

- **Diagramme des cas d'utilisation** : Il permet de visualiser les fonctionnalités principales que l'application offre aux utilisateurs ;
- **Diagramme de classes** : Il montre la structure du système en termes de classes et leurs relations ;
- **Diagramme de séquence** : Il met en évidence les interactions entre les différents objets du système au cours du temps ;

Diagrammes de cas d'utilisation

Les diagrammes de cas d'utilisation permettent de visualiser les interactions qu'un utilisateur ou un client peut avoir avec le système, illustrant les principales fonctionnalités telles que la recherche de contenu, la lecture de vidéos, et la gestion des profils [4].

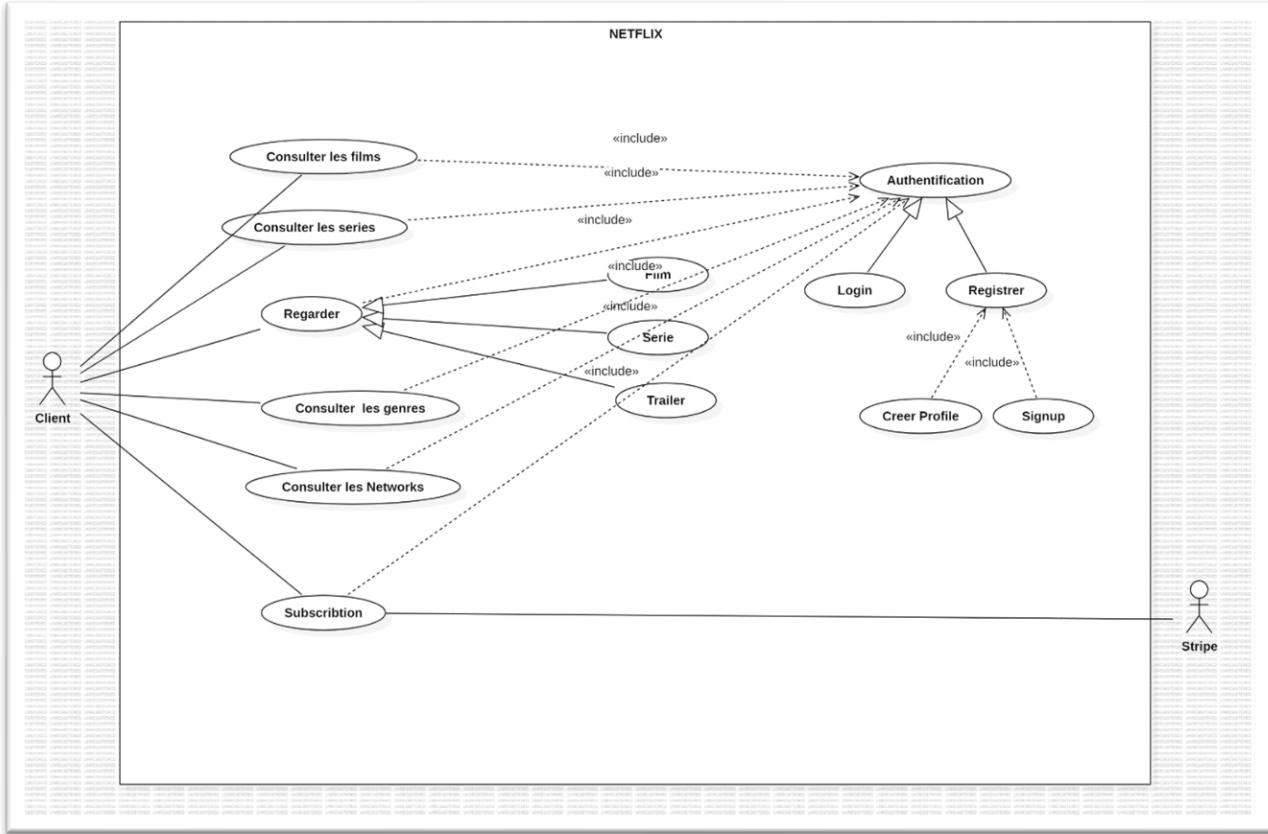


Figure 2.1 : Diagramme de cas d'utilisation

Description:

S'authentifier (Authentification)	<p>Ce cas d'utilisation représente le processus par lequel un utilisateur accède à son compte sur la plateforme.</p> <p>Register: Inscription d'un nouvel utilisateur sur la plateforme.</p> <p>Login: Connexion d'un utilisateur existant.</p> <p>Creer Profile: Création d'un profil utilisateur, souvent nécessaire après l'inscription.</p>
Consulter les films	<p>Ce cas d'utilisation permet à l'utilisateur de parcourir les films disponibles sur la plateforme.</p>
Consulter les séries	<p>Permet aux utilisateurs de parcourir les séries disponibles sur la plateforme.</p>

Consulter les genres	Permet aux utilisateurs de filtrer les contenus par genre (comédie, drame, action, etc.).
Consulter les chaînes	Permet aux utilisateurs de naviguer par chaînes ou collections de contenu spécifiques.
Regarder	<p>Ce cas d'utilisation inclut toutes les actions liées à la visualisation de contenu, comme regarder des bandes-annonces, des films, ou des épisodes de séries.</p> <p>Trailer: Regarder des bandes-annonces pour des films ou des séries.</p> <p>Film: Regarder des films complets.</p> <p>Episode: Regarder des épisodes de séries.</p>
Subscribe	Ce cas d'utilisation couvre le processus d'abonnement d'un utilisateur à la plateforme, y compris les interactions avec un service de paiement pour traiter les transactions financières.

Table 1: Description du diagramme de cas d'utilisation

Diagrammes de classes

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il est le seul obligatoire lors d'une telle modélisation. Le diagramme de classes montre la structure interne du système. Il permet de fournir une représentation abstraite des objets du système qui vont interagir ensemble pour réaliser les cas d'utilisation. Il s'agit d'une vue statique car nous ne tenons pas compte du facteur temporel dans le comportement du système. Les principaux éléments de cette vue statique sont les classes et leurs relations [5]. Ci-dessous le diagramme de classe qui correspond au projet :

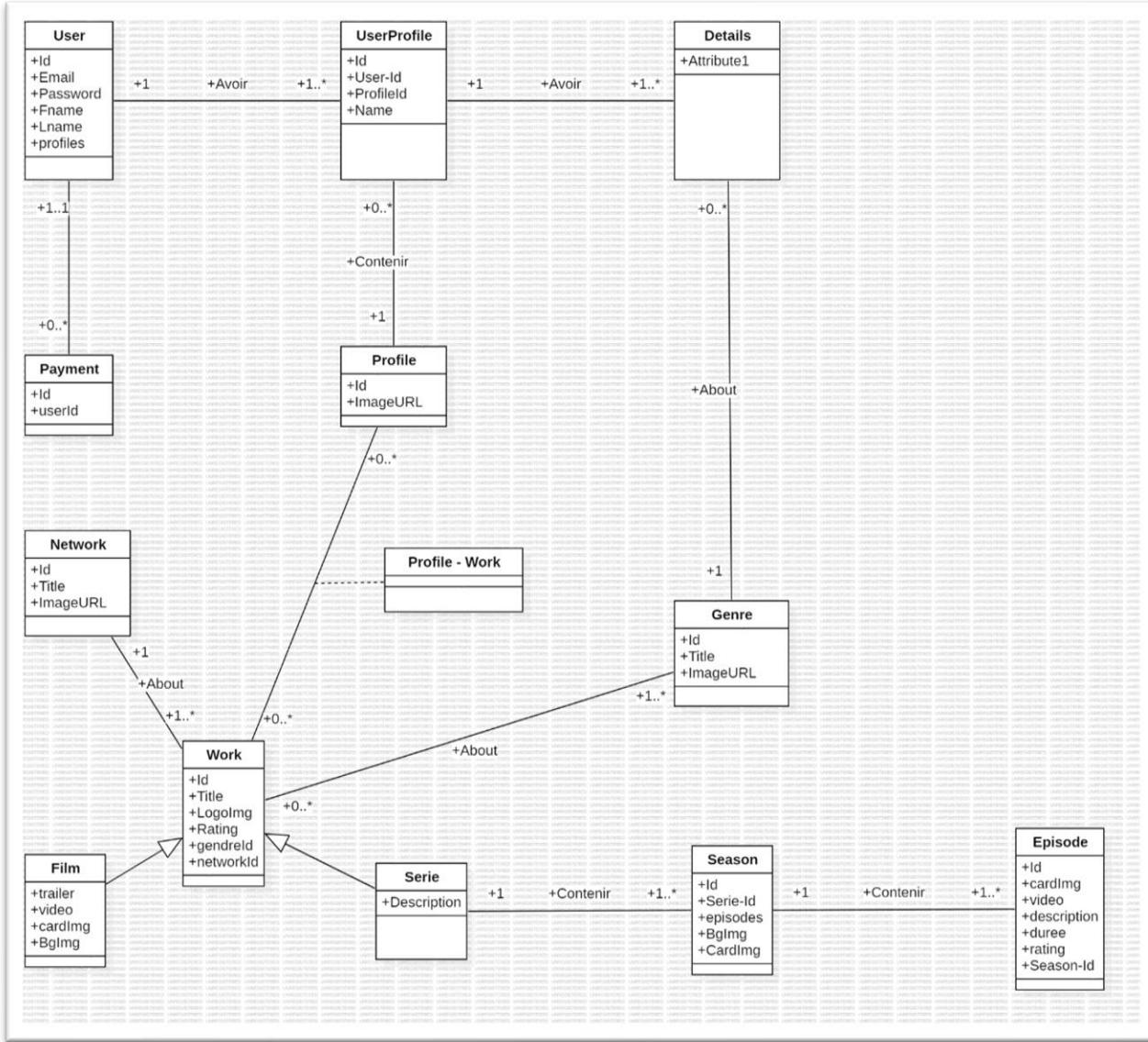


Figure 2.2 : Diagramme de classes

Diagrammes de séquence

Les diagrammes de séquence sont une solution populaire de modélisation dynamique en langage UML, car ils se concentrent plus précisément sur les lignes de vie, les processus et les objets qui vivent simultanément, et les messages qu'ils échangent entre eux pour exercer une fonction avant la fin de la ligne de vie [6].

1. Authentification des Utilisateurs

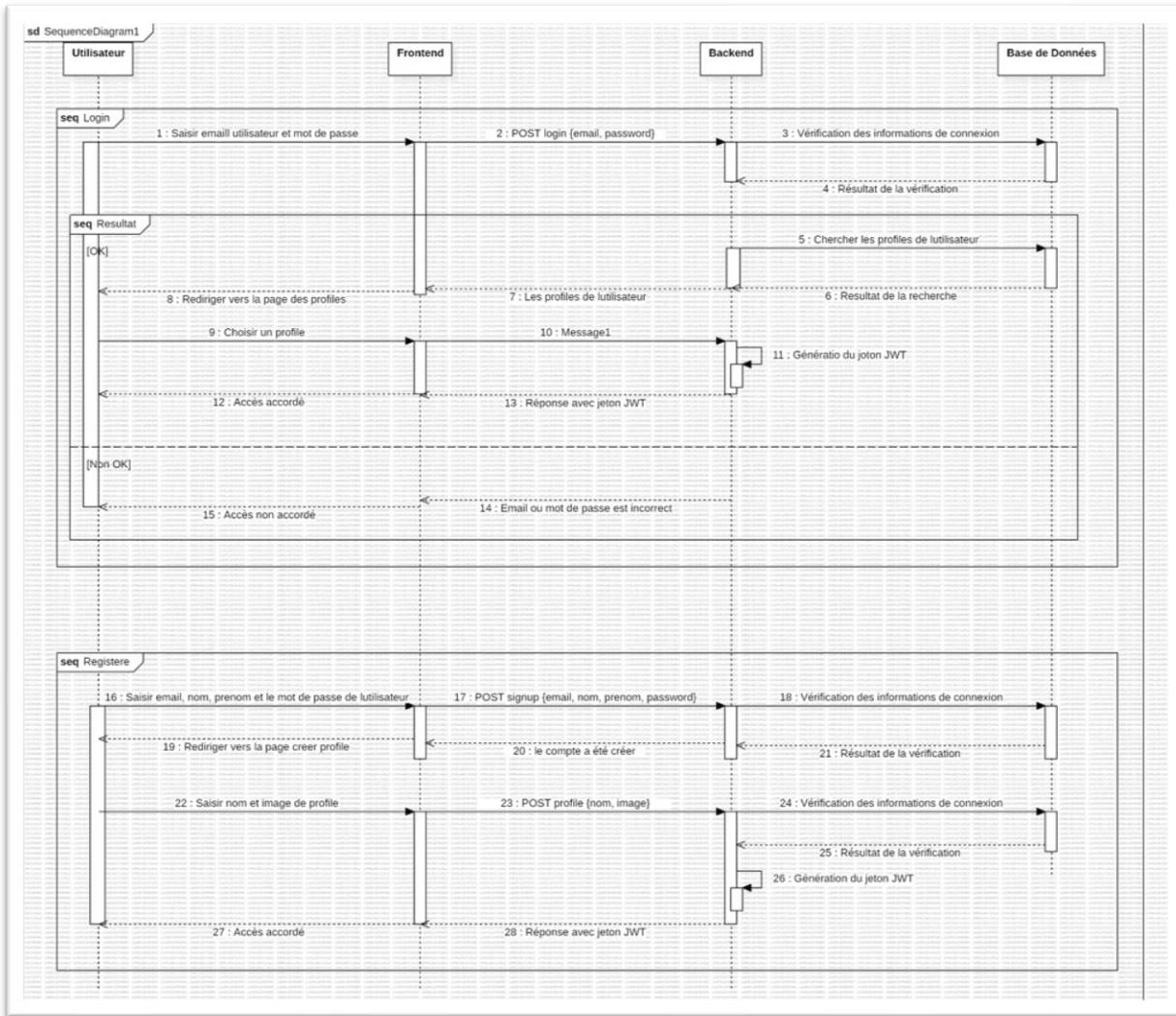


Figure 2.3: Diagramme de séquence "authentification"

Description:

- L'utilisateur entre ses informations de connexion (email d'utilisateur et mot de passe) sur le frontend.
- Le frontend envoie ces informations au backend via une requête HTTP (POST).
- Le backend vérifie les informations dans la base de données.
- Si les informations sont correctes, le backend génère un jeton JWT.
- Le backend renvoie le jeton au frontend.
- Le frontend stocke le jeton pour les futures requêtes authentifiées.

2. le Paiement de l'Abonnement

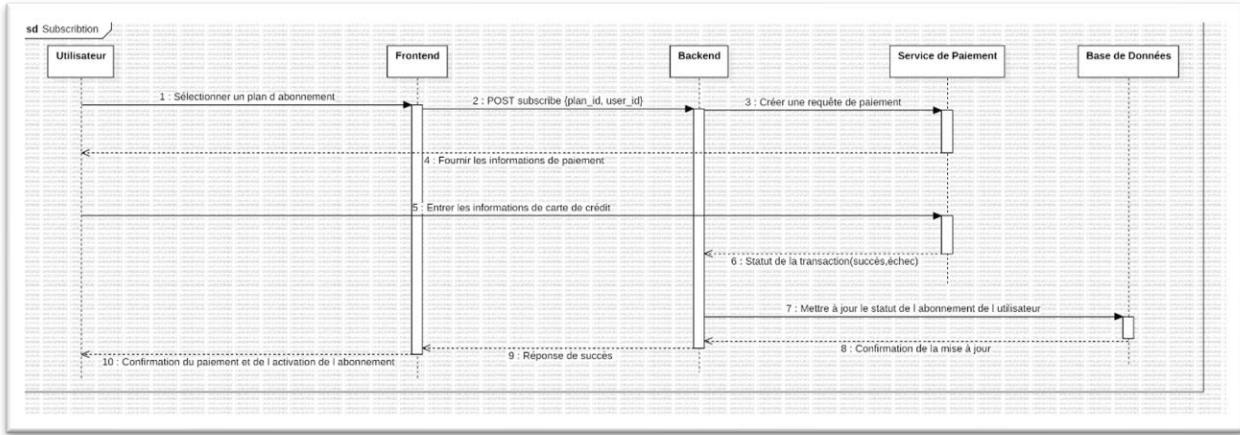


Figure 2.4: Diagramme de séquence " Paiement de l'abonnement "

Description :

- L'utilisateur sélectionne un plan d'abonnement sur le frontend.
- Le frontend envoie les informations du plan d'abonnement au backend.
- Le backend prépare une requête de paiement et la transmet au service de paiement tiers.
- Le service de paiement demande à l'utilisateur de fournir les informations de carte de crédit.
- L'utilisateur fournit les informations de paiement.
- Le service de paiement traite le paiement et renvoie le statut de la transaction au backend.
- Le backend met à jour le statut de l'abonnement de l'utilisateur dans la base de données.
- La base de données confirme la mise à jour.
- Le backend envoie une confirmation de succès au frontend.
- Le frontend affiche la confirmation du paiement et de l'activation de l'abonnement à l'utilisateur.

3. la Navigation et la Lecture de Vidéo

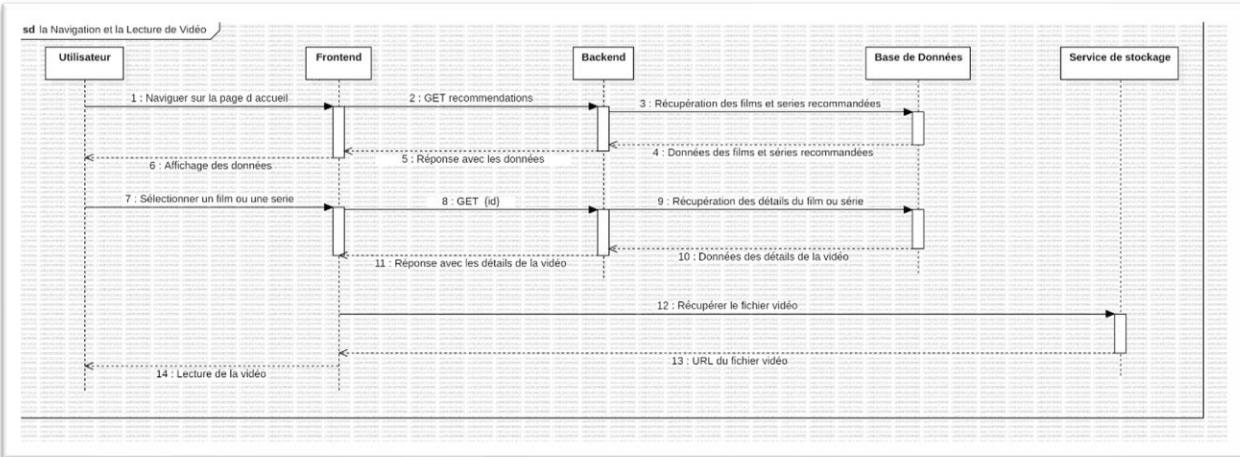


Figure 2.5: Diagramme de séquence " Navigation et lecture de vidéo "

Description :

- L'utilisateur navigue sur la page d'accueil de l'application.
- Le frontend demande la liste des vidéos recommandées au backend.
- Le backend récupère les données de la base de données.
- Le backend renvoie les données au frontend.
- L'utilisateur sélectionne une vidéo à lire.
- Le frontend demande les détails de la vidéo et l'URL du fichier média au backend.
- Le backend récupère les détails de la vidéo et l'URL de stockage.
- Le backend envoie les informations au frontend.
- Le frontend charge et lit la vidéo.

II. Diagramme d'architecture Cloud

En plus des diagrammes UML, l'architecture cloud du projet est un élément clé pour assurer la performance et la scalabilité du système. Le déploiement de l'application se fait sur AWS, en utilisant différents services pour gérer les interactions entre les composants et garantir une expérience utilisateur fluide.

Qu'est-ce que le cloud computing ?

Le terme « cloud » désigne les serveurs accessibles sur Internet, ainsi que les logiciels et bases de données qui fonctionnent sur ces serveurs. Les serveurs situés dans le cloud sont hébergés au sein de datacenters répartis dans le monde entier. L'utilisation du cloud computing (informatique cloud) permet aux utilisateurs et aux entreprises de s'affranchir de la nécessité de gérer des serveurs physiques eux-mêmes ou d'exécuter des applications logicielles sur leurs propres équipements [7].

Le diagramme d'architecture cloud inclut les éléments suivants :

Service	Description
AWS EC2	Hébergement du backend Spring Boot avec plusieurs instances pour garantir la haute disponibilité.
AWS Load Balancer	Distribution du trafic entre plusieurs instances pour équilibrer la charge
AWS RDS	Base de données relationnel pour stocker les informations utilisateur et le contenu.

AWS S3	Stockage des fichiers vidéo et autres ressources.
AWS ElastiCache	Utilisé pour optimiser les performances avec du caching.
Amazon Keyspaces	Pour stocker les grandes quantités de données non relationnelles, comme l'historique de visionnage, etc.
AWS CloudFront	Diffusion des vidéos et contenu statique via un CDN pour minimiser la latence.

Table 2: Les services AWS Utilisés

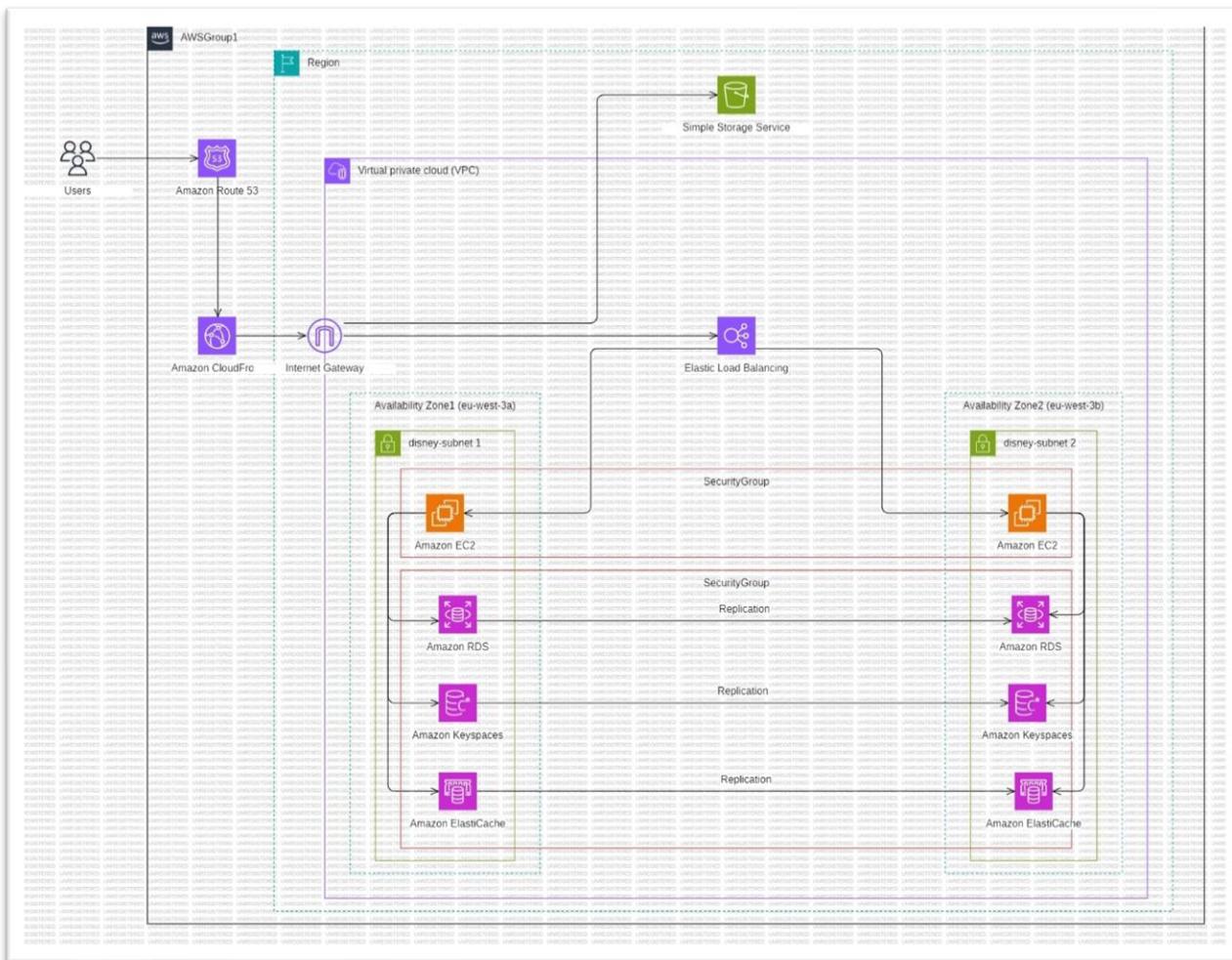


Figure 11: Diagramme du cloud

Conclusion

Ce chapitre a mis en avant l'utilisation des outils de modélisation pour concevoir l'application, notamment UML pour représenter les interactions internes et les flux de travail, ainsi que le diagramme d'architecture cloud pour illustrer le déploiement sur AWS. Ces représentations permettent d'avoir une vision globale et précise de l'architecture et des processus de l'application avant sa mise en œuvre, garantissant ainsi une meilleure organisation et scalabilité.

CHAP 4 : Etude Technique

Introduction

Ce chapitre se concentre sur l'analyse technique de l'application Movify, en détaillant l'environnement de travail, les technologies et outils utilisés, ainsi que les raisons derrière les choix effectués. De plus, une comparaison entre différentes solutions techniques existantes est présentée afin d'illustrer pourquoi certaines technologies ont été préférées pour ce projet.

I. Présentation de l'environnement de travail

L'environnement de développement du projet repose sur une combinaison d'outils et de technologies adaptés à un développement web moderne, en mettant l'accent sur la performance, la scalabilité et la facilité de déploiement.

Système d'exploitation	Windows 11 (local) et Amazon Linux 2 (serveur sur AWS EC2).
IDE	Visual Studio Code pour le frontend (React) et backend (Spring Boot).
Langages de programmation	<ul style="list-style-type: none"> • Java : Utilisé pour le backend, avec le framework Spring Boot. • JavaScript (ES6+) : Utilisé pour le développement frontend avec React. • Python : Utilisé pour le script permettant de récupérer des données de films et de séries provenant d'autres sites.
Bases de données	<ul style="list-style-type: none"> • MySQL pour les données relationnelles (utilisateurs, abonnements, films, series, etc.). • Cassandra pour les données volumineuses et non structurées (historique de visionnage, historique de recherche, etc.). • Redis pour le caching et l'amélioration des performances.
Cloud	Amazon Web Services (AWS) pour l'hébergement, le stockage et les services associés.

Table 3: Environnement de travail

II. Comparaison entre les solutions techniques existantes

Avant de choisir les technologies pour ce projet, différentes options techniques ont été évaluées selon plusieurs critères : la performance, la scalabilité, la facilité d'intégration et la gestion des coûts.

1. Backend : Spring Boot vs Node.js

- **Spring Boot (choix retenu)** : Ce framework Java est réputé pour sa robustesse, sa scalabilité et sa facilité d'intégration avec des systèmes complexes. Il offre des outils intégrés pour la gestion des API RESTful et la sécurité.
- **Node.js** : Bien que performant et adapté à des applications en temps réel, Node.js a été écarté en raison de la nécessité de gérer des transactions complexes et un système de sécurité plus robuste, ce que Spring Boot permet plus facilement.

2. Frontend : React vs Angular

- **React (choix retenu)** : Bibliothèque JavaScript populaire, React offre une grande flexibilité dans la gestion des composants et une courbe d'apprentissage rapide. De plus, son intégration avec Redux pour la gestion de l'état permet une expérience utilisateur fluide et rapide.
- **Angular** : Framework puissant pour les applications d'envergure, Angular a été écarté en raison de sa complexité et de son overhead, tandis que React répondait mieux aux besoins d'un projet individuel nécessitant une mise en place rapide.

3. Base de données : MySQL vs PostgreSQL

- **MySQL (choix retenu)** : Base de données relationnelle populaire et bien supportée par AWS RDS, MySQL offre de bonnes performances et est parfaitement adaptée aux besoins du projet, notamment pour la gestion des utilisateurs et des abonnements.
- **PostgreSQL** : Également une excellente option, mais MySQL a été choisi pour sa simplicité et sa plus grande popularité dans des environnements AWS.

4. Stockage des données volumineuses : Cassandra vs MongoDB

- **Cassandra (choix retenu)** : Cassandra est bien adapté pour des scénarios où les données sont massives et nécessitent un haut niveau de scalabilité.
- **MongoDB** : Bien que MongoDB soit une autre option NoSQL performante, Cassandra a été privilégié en raison de sa capacité à gérer de grandes quantités de données en lecture/écriture de manière distribuée.

5. Caching : Redis vs Memcached

- **Redis (choix retenu)** : Redis a été choisi pour sa flexibilité et ses performances en tant que solution de cache. Il est bien supporté par AWS ElastiCache et offre des fonctionnalités supplémentaires comme la persistance des données et les structures de données complexes.
- **Memcached** : Solution plus simple et légère, mais Redis offre davantage de fonctionnalités nécessaires pour ce projet.

6. Hébergement : AWS vs Google Cloud

- **AWS (choix retenu)** : AWS a été choisi en raison de son écosystème cloud mature et bien intégré, offrant une gamme de services tels qu'EC2, RDS, S3, ElastiCache et Keyspaces. Son modèle de facturation à la demande et sa large couverture géographique en font une option idéale pour ce projet.
- **Google Cloud** : Bien que Google Cloud soit une excellente alternative, AWS est plus largement adopté et offre plus d'options de services pour les projets nécessitant une infrastructure cloud complète.

III. Technologies et outils utilisés

1. Design



Figure 12: Logo de Photoshop

Adobe Photoshop est un logiciel de traitement d'images et de dessin assisté par ordinateur permettant de créer et d'améliorer une grande diversité de projets visuels, principalement des photographies, mais également des applications Web et mobiles, des illustrations 3D, de l'impression, du design graphique et des vidéos [8].

2. Frontend

React JS



Figure 13: Logo de React JS

React est une bibliothèque JavaScript open-source qui est utilisée pour construire des interfaces utilisateur spécifiquement pour des applications d'une seule page. Elle est utilisée pour gérer la couche d'affichage des applications web et mobiles. React a été créé par Jordan Walke, un ingénieur logiciel travaillant pour Facebook. React a été déployé pour la première fois sur Facebook en 2011 et sur Instagram en 2012 [9].

Javascript



Figure 14: Logo de JavaScript

JavaScript est un langage de programmation qui permet d'implémenter des mécanismes complexes sur une page web. À chaque fois qu'une page web fait plus que simplement afficher du contenu statique [10].

HTML



Figure 15: Logo de HTML

HTML est l'abréviation de « hypertext markup language » (langage de balisage hypertexte) et est un langage relativement simple utilisé pour créer des pages web. Comme il n'autorise pas les variables ou les fonctions, il n'est pas considéré comme un « langage de programmation », mais plutôt comme un « langage de balisage » [11].

CSS



Figure 16: Logo de CSS

CSS désigne Cascading Style Sheets (pour Feuilles de style en cascade). Il s'agit d'un langage de style dont la syntaxe est extrêmement simple mais son rendement est remarquable. En effet, le CSS s'intéresse à la mise en forme du contenu intégré avec du HTML [12].

Tailwind JS



Figure 17: Logo de Tailwind

Tailwind CSS est un framework permettant aux développeurs de personnaliser totalement et simplement le design de leur application ou de leur site web. Avec ce framework CSS, il est possible de créer un design d'interface au sein même du fichier HTML [13].

3. Backend

Java



Figure 18: Logo de Java

Java est un langage de programmation multiplateforme, orienté objet et largement utilisé pour coder des applications Web. Il s'agit d'un choix populaire parmi les développeurs depuis plus de deux décennies [14].

Spring Boot



Figure 19: Logo de Spring Boot

Spring Boot est un framework de développement JAVA. C'est une déclinaison du framework classique de Spring qui permet essentiellement de réaliser des microservices (ce sont la majeure partie du temps des services web qui sont regroupés en API) [15].

4. Databases

MySQL



Figure 20: Logo de MySQL

MySQL est un système de gestion de bases de données relationnelles SQL open source développé et supporté par Oracle. Peut être utilisé par les développeurs et des administrateurs de bases de données pour gérer efficacement les données de leurs projets [16].

Cassandra



Figure 21: Logo de Cassandra

Il s'agit d'un système de gestion de bases de données (SGBD) NoSQL open source. Cela signifie qu'il stocke les données sous forme de clé-valeur. Il stocke et manipule les données pour les restructurer [17].

Redis



Figure 22: Logo de Redis

Redis est un magasin de structures de données clé-valeur rapide, open source et en mémoire. Il vous permet de stocker des paires clé-valeur sur votre RAM. L'accès à la RAM est 150 000 fois plus rapide que l'accès à un disque et 500 fois plus rapide que l'accès au SSD [17].

5. Scrapping



Figure 23: Logo de Python

Python est un langage de programmation largement utilisé dans les applications Web, le développement de logiciels, la science des données et le machine learning (ML). Les développeurs utilisent Python parce que c'est un langage efficace et facile à apprendre, et qu'il peut s'exécuter sur de nombreuses plateformes différentes [18].

6. Cloud



Figure 24: Logo de AWS

AWS (Amazon Web Services) est une plateforme de cloud computing fournie par Amazon. Offre des outils tels que la puissance de calcul, le stockage de bases de données et les services de diffusion de contenu [19].

7. Gestion du Code Source



Figure 25: Logo de Git

Git est un outil DevOps utilisé pour la gestion du code source. Il s'agit d'un système de contrôle de version gratuit et open source utilisée pour gérer efficacement des projets de petite à très grande envergure. Git est habitué à suivre les modifications du code source [20].

8. Outils de Collaboration



Figure 26: Logo de Jira

Jira est l'outil de gestion de projet agile n°1 utilisé par les équipes pour planifier, suivre, publier et prendre en charge des logiciels de classe mondiale en toute confiance. Il s'agit de la source unique de vérité pour l'ensemble de votre cycle de développement, offrant aux équipes autonomes le contexte nécessaire pour agir rapidement tout en restant connectées à l'objectif commercial global [21].

Conclusion

Ce chapitre a présenté les différentes technologies et outils utilisés dans le projet, en justifiant les choix faits pour chaque composant. Après une analyse comparative des solutions techniques possibles, AWS et son écosystème de services ont été retenus pour leur robustesse et leur scalabilité, tandis que des technologies comme Java, React, et Redis ont été sélectionnées pour offrir une architecture performante et évolutive

CHAP 5 : Mise en œuvre et Réalisation

Introduction

Ce chapitre présente la réalisation technique de l'application, en couvrant les aspects du frontend, du backend, du scraping des données, de la gestion des bases de données et de l'infrastructure cloud utilisée pour héberger l'application.

Implémentation et solution développée

I. Frontend : Développement de l'interface utilisateur avec React

L'interface utilisateur a été développée en utilisant **React**, permettant de créer des composants réactifs et modulaires. Voici les principales fonctionnalités:

- **Page d'accueil** : Affichage dynamique des films et séries organisés en catégories comme « Nouveautés », « Populaires » et « Recommandations ».
- **Profil utilisateur** : Possibilité de créer plusieurs profils sur un compte avec des recommandations personnalisées pour chaque profil.
- **Lecteur vidéo intégré** : Un lecteur vidéo interactif avec des fonctionnalités comme la lecture, pause, et la gestion de la qualité vidéo.

Défis rencontrés :

- **Gestion de l'état global** : La mise en place de Redux pour synchroniser l'état de l'application entre les composants React s'est révélée cruciale pour éviter la duplication des données et garantir une expérience utilisateur fluide ;
- **Optimisation des performances** : La pagination et le chargement dynamique des contenus ont été implémentés pour éviter les temps de chargement prolongés, en particulier sur la page d'accueil.

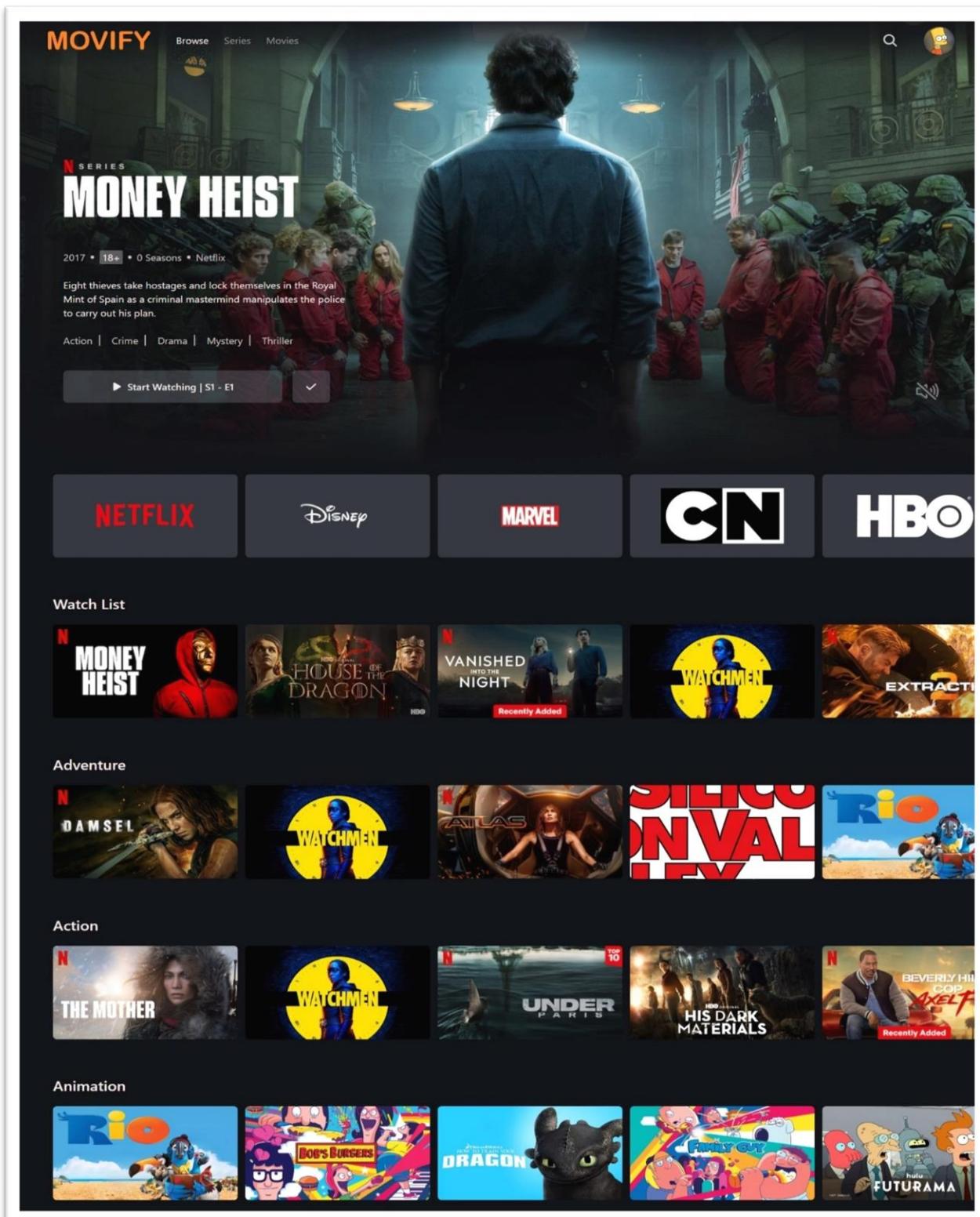


Figure 27: la page d'accueil de Movify



Figure 28: La page de lecteur vidéo

II. Backend : API REST avec Spring Boot

Le backend est géré par **Spring Boot**, un framework Java qui permet de créer et gérer des API RESTful robustes. Les principales fonctionnalités du backend incluent:

- **API de gestion des utilisateurs** : Authentification sécurisée par JWT et gestion des comptes et profils utilisateurs ;
- **API des films et séries** : Exposition des informations sur les films et séries, avec la possibilité de filtrer par genre, popularité, ou recommandations personnalisées ;
- **Intégration de Stripe** : Gestion des paiements et abonnements utilisateurs.

Défis rencontrés :

- **Scalabilité** : L'implémentation d'une base de données Cassandra distribuée (Amazon Keyspaces) pour gérer les volumes importants de données d'historique de visionnage a permis de répondre aux besoins de scalabilité de l'application.
- **Sécurité** : La mise en place d'un système d'authentification basé sur JWT a permis de sécuriser les interactions entre le frontend et l'API backend, garantissant ainsi la confidentialité des données utilisateur.

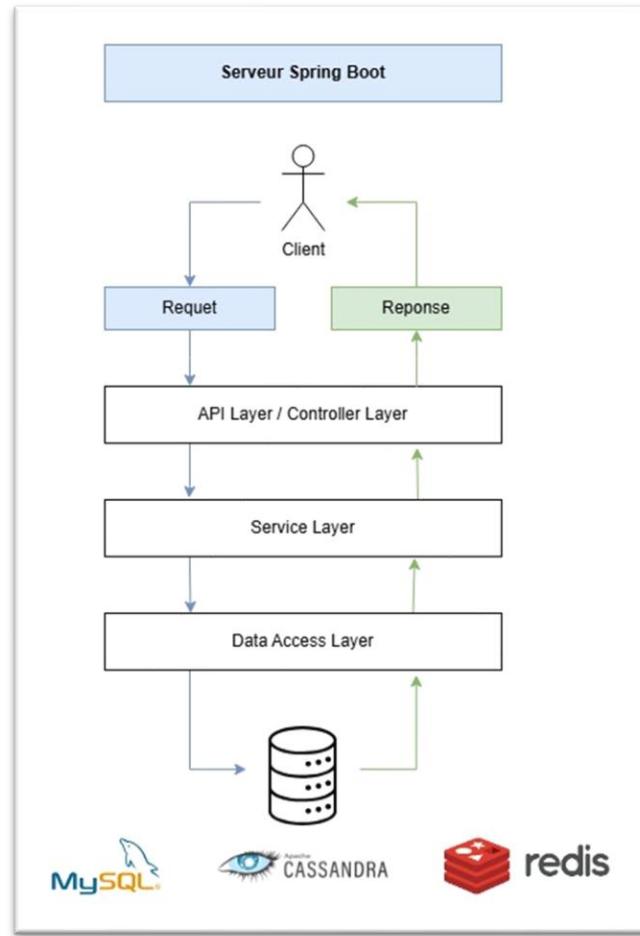


Figure 29: Architecture du backend

The screenshot shows the Stripe dashboard in "Test mode". The main header includes "Test mode", "Complete profile", and "Search" fields. The sidebar has links for Home, Balances, Transactions, Customers, and Product catalog. The "Transactions" section displays a summary of 41 transactions: 5 succeeded, 0 refunded, 0 disputed, 0 failed, and 0 uncaptured. Below this, a table lists individual transactions with columns for Amount, Payment method, Description, Customer, Date, Refunded date, and Dispute amount. Some transactions are marked as "Succeeded" with green checkmarks, while others are "Canceled" with red X's. The table also includes "Export" and "Edit columns" buttons.

Figure 30: tableau de bord Stripe montrant les transactions et abonnements

III. Scraping de données avec Python

Une partie clé du projet a consisté à automatiser la récupération des données des films et séries via un script Python de web scraping. Le scraping a été effectué à l'aide de **BeautifulSoup** et **requests** pour extraire des données depuis des sources en ligne.

Explication détaillée:

- **Scraping** : Le script envoie des requêtes HTTP aux pages des sites cibles et utilise BeautifulSoup pour extraire des informations comme le titre, le genre, la durée et l'image d'affiche ;
- **Insertion dans la base de données** : Une fois les données extraites, elles sont automatiquement insérées dans la base de données **MySQL**.
- **Insertion dans le service AWS de stockage** : Une fois les données extraites, les fichiers (images, vidéos) sont automatiquement insérées dans le service AWS S3 (Scalable Storage Service).

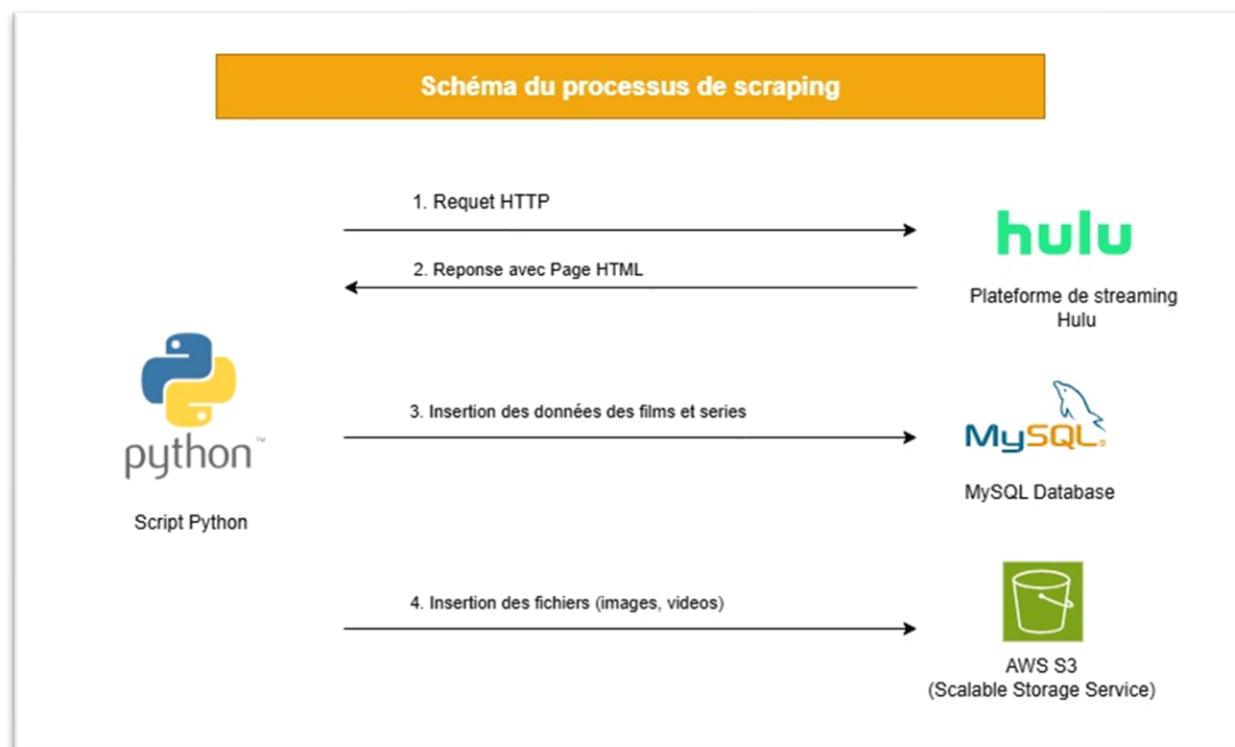


Figure 31: Schéma du processus de scraping

IV. Bases de données

L'application repose sur trois types de bases de données pour répondre à des besoins spécifiques :

- **MySQL (AWS RDS)** : Utilisée pour gérer les données relationnelles, telles que les informations des utilisateurs, les abonnements, et les métadonnées des films et séries ;
- **Cassandra (Amazon Keyspaces)** : Base de données NoSQL utilisée pour stocker l'historique de visionnage des utilisateurs et l'historique de recherche des utilisateurs, avec une capacité de traitement efficace pour les grands volumes de données ;
- **Redis (AWS ElastiCache)** : Utilisé pour le caching des données fréquemment consultées, comme les informations des films populaires ou les listes de tendances, afin d'améliorer les performances des requêtes.

Défis rencontrés :

Synchronisation des bases de données : Assurer la cohérence entre MySQL et Cassandra lors de la mise à jour des informations utilisateur a nécessité la mise en place d'une couche intermédiaire pour gérer les transactions.

V. Infrastructure Cloud

Le projet est déployé sur AWS en utilisant plusieurs services pour assurer la disponibilité, la scalabilité, et la performance de l'application :

- **Amazon EC2** : Utilisé pour héberger le backend Spring Boot. Deux instances EC2 sont déployées dans différentes zones de disponibilité pour assurer la redondance et la haute disponibilité ;
- **AWS Elastic Load Balancer** : Assure la répartition du trafic entre plusieurs instances EC2 pour équilibrer la charge et gérer les pics de trafic.
- **AWS S3** : Stockage des fichiers vidéo et des images (affiches de films);
- **AWS CloudFront** : Distribution des contenus (vidéo et images) via un CDN (Content Delivery Network) pour minimiser la latence et offrir une expérience de streaming fluide, même avec des utilisateurs répartis géographiquement ;

- **AWS RDS et Keyspaces** : Hébergement des bases de données MySQL et Cassandra respectivement ;

Défis rencontrés :

Optimisation des coûts : Ajustement des services utilisés et des configurations pour minimiser les coûts tout en assurant une performance optimale.

aws Services Search [Alt+S] Paris Soulalma ouhmida

S S EC VPC CloudFront RDS Certificate Manager Amazon Keypairs Route 53

```
2024-09-16T15:10:09:59.702Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.715Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.716Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.716Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.717Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.718Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.719Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.720Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:09:59.720Z WARN 21531 --- [demo] | s0-io-1] c.d.o.d.i.c.m.DefaultTopologyMonitor : [s0] Unable to determine broadcast RPC port. Trying to fall back to port used by the control connection.
2024-09-16T15:10:01:01.046Z INFO 21531 --- [demo] | main] o.s.b.a.e.web.EndpointLinksResolver : Exposing 1 endpoint beneath base path '/actuator'
2024-09-16T15:10:01:266Z INFO 21531 --- [demo] | main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/'
2024-09-16T15:10:01:312Z INFO 21531 --- [demo] | main] com.example.demo.demoApplication : Started DemoApplication in 22.715 seconds (process running for 24.494)
2024-09-16T15:10:06:478Z INFO 21531 --- [demo] [nio-8080-exec-1] o.a.c.c.C.(Tomcat).localhost.[]/ : Initializing Spring DispatcherServlet 'dispatcherServlet'
2024-09-16T15:10:06:479Z INFO 21531 --- [demo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initialization Servlet 'dispatcherServlet'
2024-09-16T15:10:06:480Z INFO 21531 --- [demo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 3 ms
Hibernate: select p1_0.id,p1_0.email,p1_0.fname,p1_0.lname,p1_0.pass,p1_0.profiles from users p1_0
Hibernate: select p1_0.id,p1_0.email,p1_0.fname,p1_0.lname,p1_0.pass,p1_0.profiles from users p1_0
```

Figure 32: Serveur Movify 1 dans AWS EC2

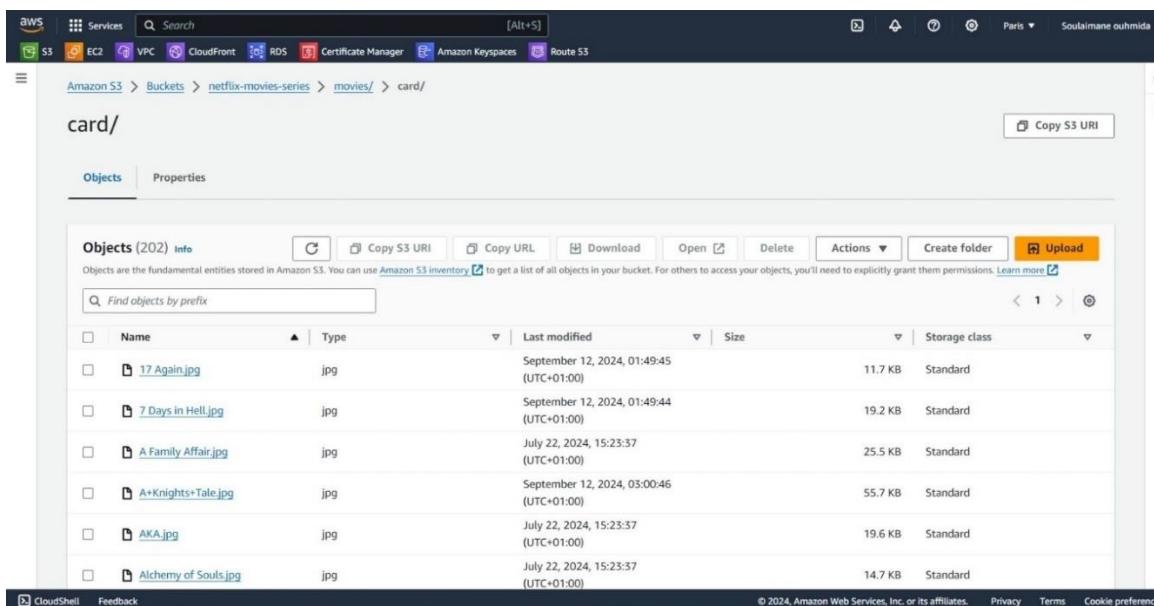


Figure 33: Images des films dans AWS S3

Conclusion

Le chapitre a couvert les aspects pratiques du développement du projet Movify, y compris l'interface utilisateur, le backend, le scraping de données, et l'infrastructure cloud. Ce projet a permis de développer une architecture complète et scalable, intégrant des technologies modernes comme React, Spring Boot, AWS, et Python pour le scraping de données.

CHAP 6 : Bilan

Introduction

Ce chapitre se concentre sur le bilan global du projet clone de Netflix, en abordant les aspects professionnels, techniques, et personnels. Il s'agit d'une évaluation des compétences acquises, des défis relevés, et des changements observés au cours du projet.

I. Bilan professionnel

Au cours de ce projet, j'ai eu l'opportunité de développer plusieurs compétences professionnelles essentielles :

- **Capacité d'écoute et d'analyse** : J'ai appris à identifier et comprendre les besoins du projet à travers une analyse minutieuse des solutions existantes. Bien que j'aie travaillé de manière autonome, j'ai cherché à intégrer les retours et conseils de mes tuteurs, ce qui a renforcé ma capacité d'écoute active.
- **Collaboration et travail en équipe** : Bien que le projet ait été réalisé individuellement durant mon stage, j'ai appris l'importance de la collaboration avec d'autres professionnels, comme mes superviseurs et collègues, pour obtenir des retours et ajuster certaines parties du projet. Ce projet m'a également sensibilisé à la gestion du travail en équipe, une compétence clé dans des projets de plus grande envergure.
- **Gestion du temps et des priorités** : J'ai dû planifier efficacement mes tâches pour respecter les délais tout en m'assurant que chaque fonctionnalité soit correctement implémentée. J'ai également appris à prioriser les parties critiques du projet pour livrer un produit final fonctionnel.

II. Bilan technique

Ce projet a été l'occasion d'enrichir mes compétences techniques dans plusieurs domaines. Voici un aperçu des technologies que j'ai maîtrisées :

- **Frontend avec React** : J'ai approfondi ma compréhension de **React**, en créant des composants dynamiques et interactifs. Cela m'a permis de développer une interface utilisateur performante et responsive, améliorant ainsi l'expérience utilisateur.
- **Backend avec Spring Boot** : Le backend a été développé avec **Spring Boot**, où j'ai mis en place des API RESTful sécurisées et efficaces. J'ai également intégré **Stripe** pour la gestion des paiements, ce qui m'a permis de travailler avec des services tiers de manière fluide.
- **Scraping de données avec Python** : Une partie importante du projet a été l'automatisation de la récupération des données avec un script Python utilisant **BeautifulSoup** et **requests**. Cela m'a permis de scraper des données de films et séries et de les stocker dans une base

de données MySQL. Ce processus m'a appris à manipuler et structurer de grandes quantités de données non structurées.

- **Cloud Computing avec AWS** : L'intégration de plusieurs services AWS tels que **EC2**, **S3**, **CloudFront**, **RDS**, et **Keyspaces** m'a permis de comprendre comment déployer et gérer une infrastructure cloud scalable. J'ai également validé mes compétences en obtenant une **certification AWS**, ce qui confirme ma maîtrise des services cloud. Cette certification renforce mes connaissances en conception d'architectures cloud sécurisées et performantes.
- **Gestion des bases de données** : J'ai utilisé **MySQL** pour les données relationnelles et **Cassandra** pour les données NoSQL, ce qui m'a permis de comprendre les avantages de chaque technologie selon les types de données. **Redis** via **ElastiCache** a également été utilisé pour optimiser les performances du projet en implémentant un système de cache.



Figure 34: Mes certifications en cloud computing (AWS)

III. Bilan personnel

Sur le plan personnel, ce projet a eu un impact significatif sur ma croissance et mon développement :

- **Confiance en soi** : Mener à bien un projet d'une telle complexité m'a permis de gagner en confiance. La capacité à résoudre des problèmes techniques et à livrer un produit final fonctionnel m'a donné la certitude que je pouvais aborder des projets encore plus ambitieux.
- **Résilience et persévérance** : Les défis rencontrés, notamment lors de la résolution de bugs et des difficultés de déploiement, m'ont appris à persévéérer et à rester concentré sur mes objectifs. J'ai développé une grande capacité à surmonter les obstacles en cherchant des solutions adaptées et en apprenant de mes erreurs.
- **Autonomie** : Travailler seul sur ce projet m'a appris à être autonome, à chercher les informations nécessaires, et à résoudre des problèmes par moi-même. Cette expérience a renforcé ma capacité à m'auto-former, un atout essentiel dans un secteur en constante évolution.
- **Adaptabilité** : L'intégration de plusieurs technologies nouvelles, comme **Cassandra** et **CloudFront**, m'a appris à m'adapter rapidement à de nouveaux environnements et à élargir mes compétences techniques en peu de temps.

Conclusion

Le bilan de ce projet est extrêmement positif. D'un point de vue professionnel, j'ai acquis des compétences clés en gestion de projet, collaboration, et gestion du temps. Technique, j'ai approfondi ma maîtrise de technologies modernes et validé mes compétences en cloud computing avec une certification AWS. Personnellement, ce projet a renforcé ma confiance, ma persévérance, et ma capacité à m'adapter à des environnements changeants. Ces apprentissages seront des atouts précieux pour mes futurs projets et ma carrière professionnelle.

Conclusion Générale et Perspectives

Ce projet de développement Movify a représenté une expérience complète en matière de conception et de mise en œuvre d'une application de streaming. Chaque phase du projet a contribué à l'atteinte des objectifs initiaux, en intégrant des aspects techniques et fonctionnels essentiels.

Le **Chapitre 1** a introduit le projet en définissant clairement ses objectifs, les défis à relever, et les motivations derrière la création d'une plateforme de streaming similaire à Netflix. Ce premier chapitre a posé les bases du projet en identifiant les besoins des utilisateurs et les fonctionnalités essentielles à intégrer.

Le **Chapitre 2** a exploré le contexte du projet en détail, en décrivant les différents modules de l'application, tels que le frontend, le backend, et les services cloud. Ce chapitre a également abordé les besoins fonctionnels (comme la gestion des utilisateurs et des contenus) et non fonctionnels (comme la performance et la sécurité), en identifiant les risques potentiels, tels que les problèmes matériels, logiciels, et les retards éventuels dans la gestion des informations.

Dans le **Chapitre 3**, nous avons détaillé la phase de conception à travers des diagrammes UML. Ces diagrammes ont permis de modéliser les interactions entre les utilisateurs et le système (diagramme des cas d'utilisation), de définir les structures de données (diagramme de classes), de comprendre les séquences d'opérations (diagramme de séquence), et de visualiser les flux de travail (diagramme d'activités). Ces outils ont facilité la planification et la structuration du projet en offrant une vue d'ensemble claire et organisée.

Le **Chapitre 4** a présenté l'environnement technique et les solutions choisies pour le projet. Nous avons comparé différentes technologies et outils, expliquant pourquoi des solutions comme React pour le développement frontend, Spring Boot pour le backend, et AWS pour le déploiement cloud ont été privilégiées. Ce chapitre a détaillé les critères de sélection des technologies, tels que la scalabilité, la performance et la compatibilité avec les besoins du projet.

Le **Chapitre 5** a fourni une vue d'ensemble détaillée de l'implémentation et des solutions développées. La réalisation du projet a impliqué la création de diverses fonctionnalités, comme le développement de l'interface utilisateur, la mise en place du backend, et l'intégration des services de cloud. Un aspect clé de ce chapitre a été le développement d'un script de scraping en Python, qui a permis de collecter des données de contenu depuis des sources externes et de les intégrer dans une base de données MySQL. Cette fonctionnalité a été cruciale pour enrichir l'application avec des contenus variés et actualisés.

Le **Chapitre 6** a offert un bilan complet du projet, en évaluant les compétences professionnelles et techniques acquises. Il a souligné la manière dont le projet a permis de renforcer mes capacités en gestion de projet, en développement logiciel, et en déploiement cloud. Ce chapitre a également exploré les aspects personnels du projet, tels que la confiance en soi et la résilience développées à travers les défis rencontrés.

En conclusion, le projet a abouti à une solution complète et fonctionnelle : une plateforme de streaming capable de gérer efficacement les contenus et de fournir une expérience utilisateur optimale. L'intégration de technologies modernes et le déploiement sur une infrastructure cloud scalable ont permis de créer une application robuste et performante. Ce projet est pertinent car il répond aux besoins identifiés des utilisateurs tout en surmontant les défis techniques.

Perspectives :

- **Nouvelles fonctionnalités** : Intégrer des systèmes de recommandation basés sur l'intelligence artificielle pour personnaliser les suggestions de contenu, permettre le téléchargement de vidéos pour une consultation hors ligne, et offrir des options de personnalisation de l'interface pour une meilleure expérience utilisateur.
- **Optimisation de la performance** : Affiner les algorithmes de streaming pour minimiser la latence et ajuster la qualité vidéo en fonction de la bande passante disponible, assurant ainsi une diffusion fluide même dans des conditions variables.
- **Sécurité renforcée** : Mettre en place des mesures de sécurité avancées pour protéger les données sensibles des utilisateurs, y compris le chiffrement des données et la détection des intrusions pour prévenir les attaques potentielles.
- **Expansion des capacités cloud** : Explorer l'utilisation de services cloud supplémentaires pour améliorer la scalabilité et la résilience de l'application, comme l'intégration de solutions de sauvegarde et de récupération après sinistre, ainsi que l'amélioration de la gestion des performances.

Ces améliorations proposées visent à rendre l'application encore plus performante et adaptable aux besoins futurs des utilisateurs, tout en consolidant les acquis du projet et en ouvrant la voie à des opportunités professionnelles dans le développement logiciel et l'architecture cloud.

Annexe

I. Développement Frontend

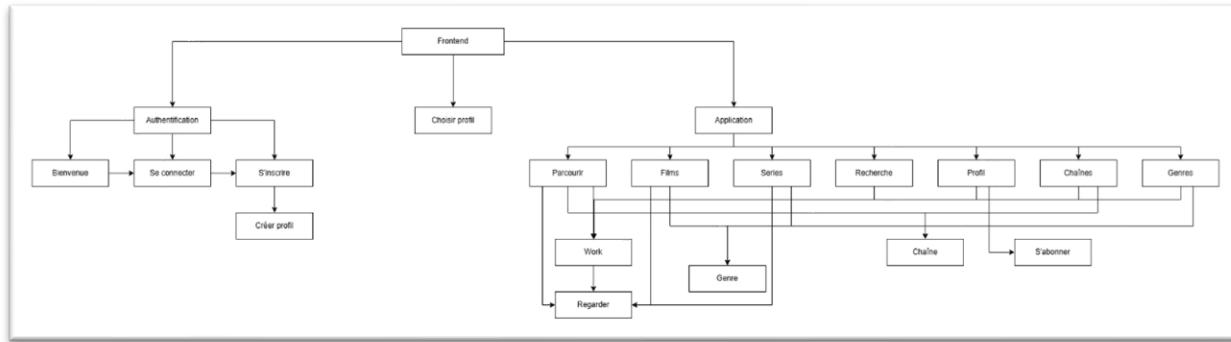


Figure 35: Structure du frontend

Technologies Utilisées

React	Utilisé pour construire l'interface utilisateur de votre application.
Redux	Une bibliothèque de gestion d'état qui fournit un moyen prévisible de stocker et de mettre à jour l'état global de l'application. En utilisant Redux, on va stocker toutes les données dont on a besoin dans un seul « store », qui peut être partagé entre tous les composants de l'application.
React Router	Une bibliothèque permet de gérer les routes de l'application afin de créer des liens et de naviguer entre différentes pages.
Axios	Une bibliothèque JavaScript populaire, offre un ensemble complet de fonctionnalités pour effectuer des requêtes HTTP.
React icons	Une bibliothèque open-source qui offre une vaste collection d'icônes prêtées à l'emploi pour les développeurs React.

Tableau 5: Technologies utilisées dans frontend.

Développement des Fonctionnalités Principales

Authentification

Lorsque l'utilisateur visite le site, la page d'accueil apparaît si l'utilisateur n'est pas identifié. Il peut alors choisir de se connecter à son compte, ou d'en créer un nouveau.

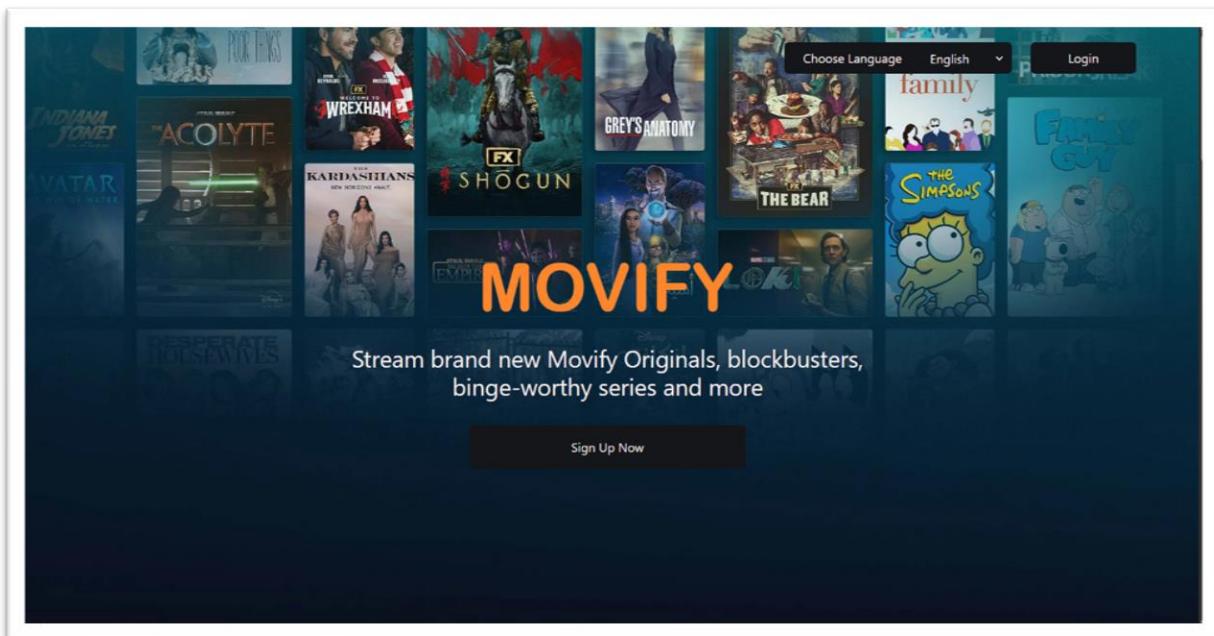


Figure 36: Page d'accueil.

Création du compte

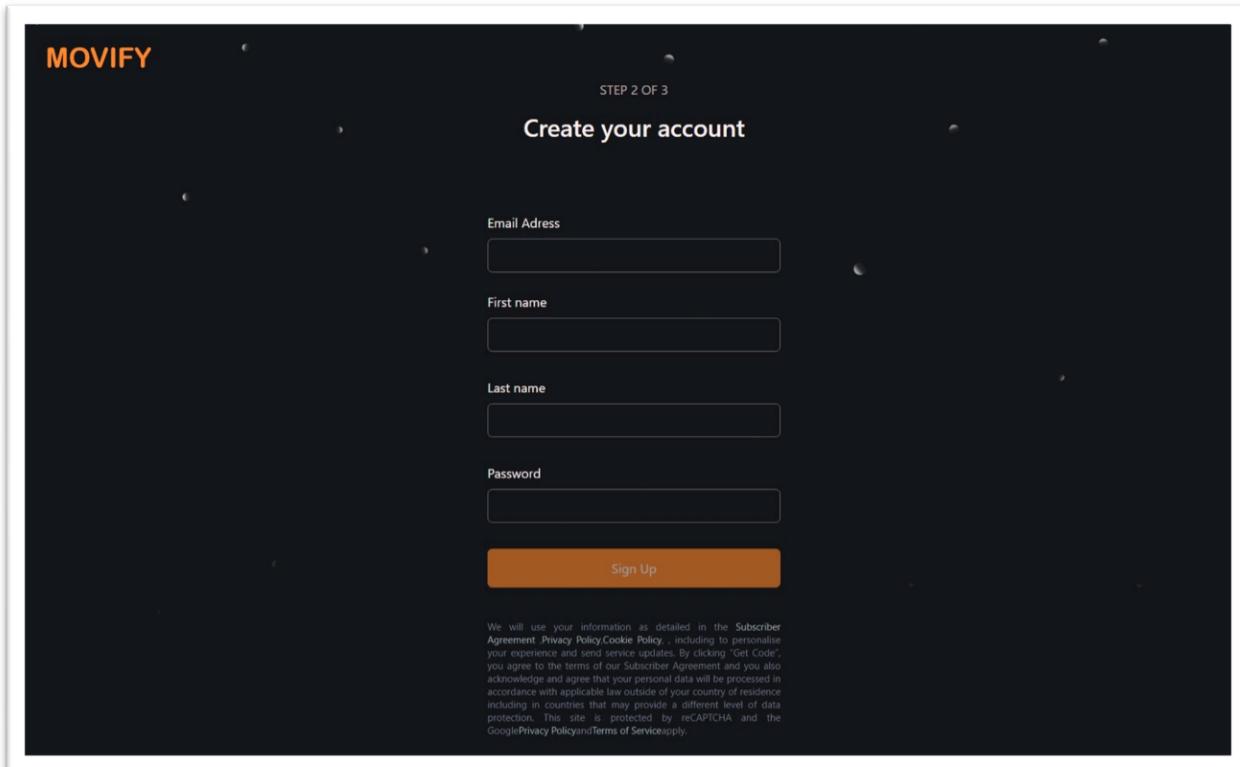
Le processus d'inscription et de création de profil se déroule en plusieurs étapes, chacune étant cruciale pour garantir une expérience utilisateur fluide et sécurisée.

1. Formulaire d'Inscription :

L'utilisateur commence par remplir un formulaire d'inscription comportant les champs suivants :

- **Email** : Adresse email de l'utilisateur, utilisée pour l'identification et la communication.
- **Prénom** : Prénom de l'utilisateur.
- **Nom de Famille** : Nom de famille de l'utilisateur.
- **Mot de Passe** : Mot de passe choisi par l'utilisateur, qui doit respecter des critères de sécurité définis (longueur minimale, complexité, etc.).

Une fois le formulaire rempli, les données sont envoyées au serveur via une requête POST. La validation des champs est effectuée côté client avant l'envoi pour garantir que toutes les informations requises sont présentes et conformes.



The screenshot shows a dark-themed sign-up form for 'MOVIFY'. At the top, it says 'STEP 2 OF 3' and 'Create your account'. There are four input fields: 'Email Address', 'First name', 'Last name', and 'Password'. Below these is a large orange 'Sign Up' button. At the bottom right, there is a small text block about data protection and terms of service, mentioning 'Subscriber Agreement', 'Privacy Policy', 'Cookie Policy', and 'reCAPTCHA'.

Figure 37: Page d'inscription.

2. Formulaire d'Inscription :

Après la création du compte, l'utilisateur est redirigé vers une page de création de profil. Cette étape permet à l'utilisateur de :

- **Choisir une Image de Profil** : L'utilisateur peut choisir parmi des images prédefinies.
- **Compléter les Informations de Profil** : L'utilisateur remplit un formulaire pour fournir des informations supplémentaires, telles que le nom du profil.

Les données du profil, y compris l'image et le nom, sont envoyées au serveur via une requête POST.

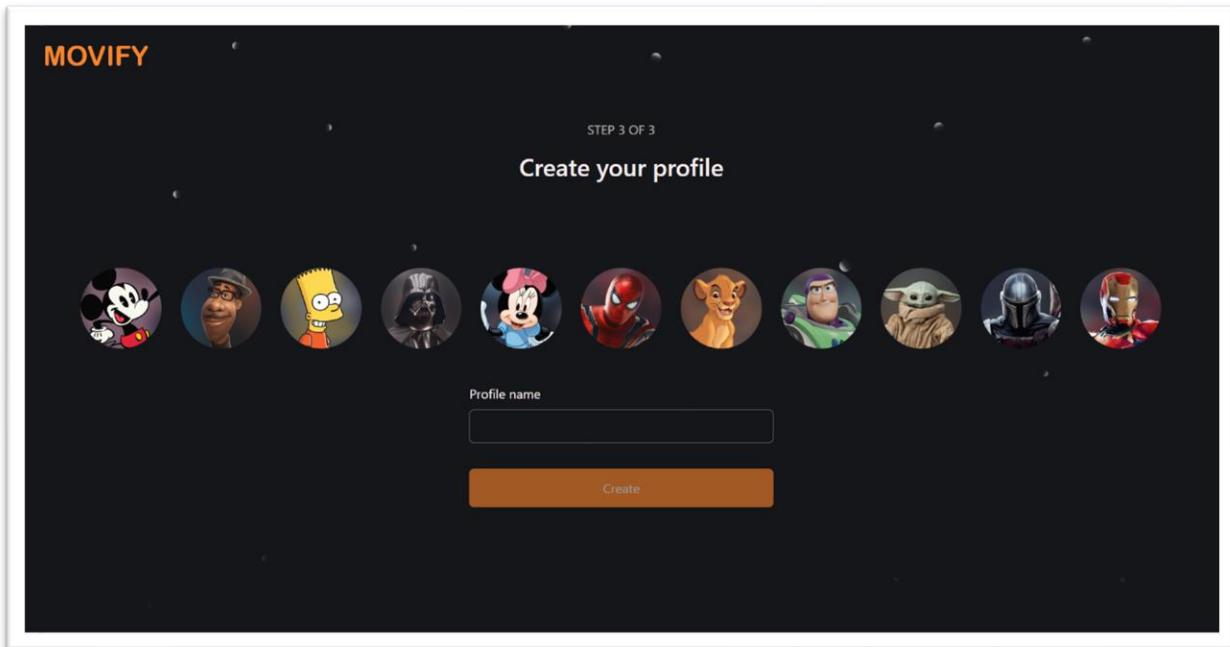


Figure 38: Page de création du profil.

Connexion

Le processus de connexion permet à un utilisateur de se connecter à son compte en utilisant ses identifiants. Voici les étapes détaillées pour la connexion:

1. Formulaire de Connexion :

L'utilisateur accède à un formulaire de connexion comportant les champs suivants :

- **Email** : Adresse email associée au compte utilisateur. Ce champ est utilisé pour identifier l'utilisateur.
- **Mot de Passe** : Mot de passe correspondant à l'adresse email.

Lors de la soumission du formulaire, les données sont envoyées au serveur via une requête POST. Une validation est effectuée côté client pour vérifier que les champs ne sont pas vides avant l'envoi.

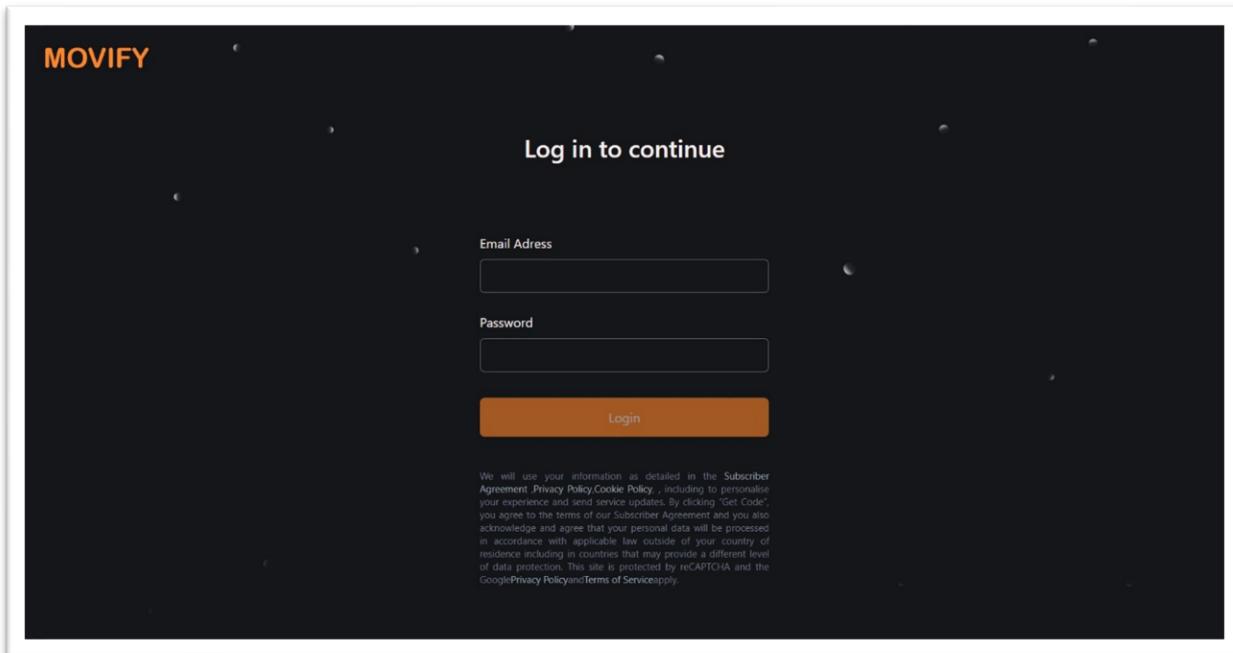


Figure 39: Page de Connexion (Login).

Affichage des Films et Séries

Décrivez comment les films sont récupérés et affichés. Par exemple, l'utilisation d'un composant MovieList pour afficher une liste de films en récupérant les données depuis l'API.

Paiement et abonnement

Si votre application permet aux utilisateurs de créer une liste de favoris ou un panier, détaillez comment cette fonctionnalité est implémentée et gérée.

II. Interaction Frontend-Backend

L'interaction entre le frontend et le backend est un processus clé dans notre application. Le frontend, envoie des requêtes HTTP via Axios à l'API REST exposée par le backend.

Exemple d'Interaction Utilisateur

Lorsqu'un utilisateur se connecte, une requête **POST** est envoyée au backend avec les informations d'identification de l'utilisateur (l'adresse-mail et le mot de passe).

Le backend authentifie ces informations à l'aide de **Spring Security**, génère un token JWT, et le renvoie au frontend.

Ce token est ensuite stocké dans le navigateur (dans localStorage) pour des requêtes ultérieures.

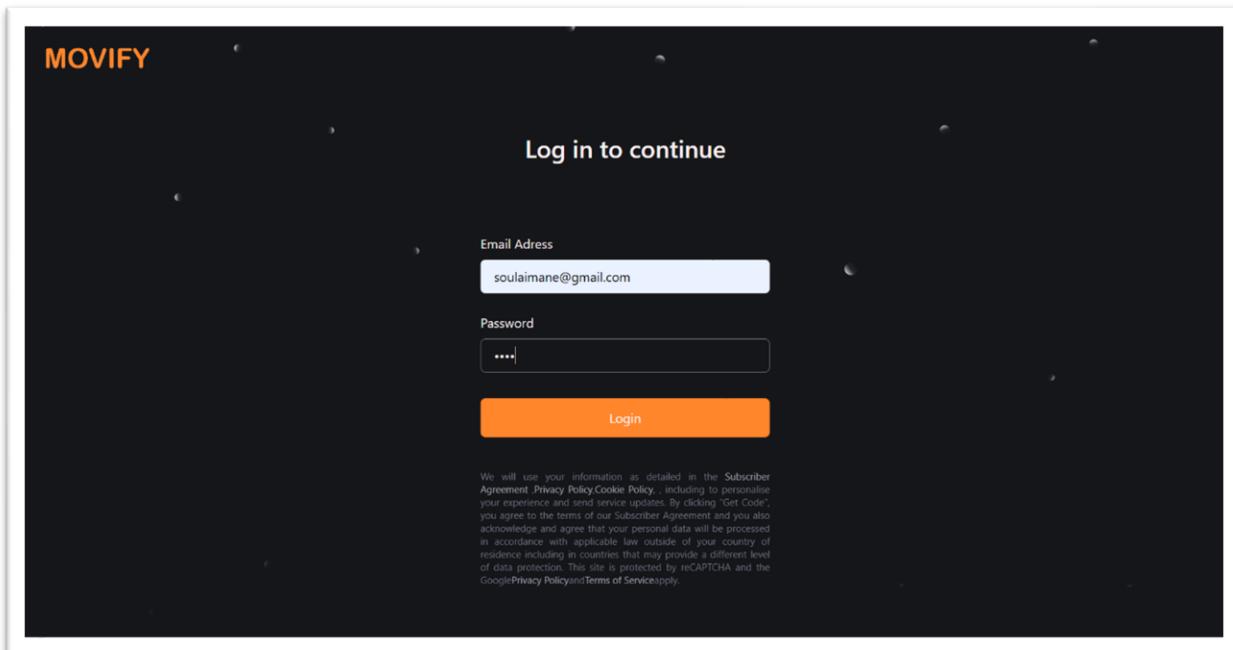


Figure 40: Page de connexion

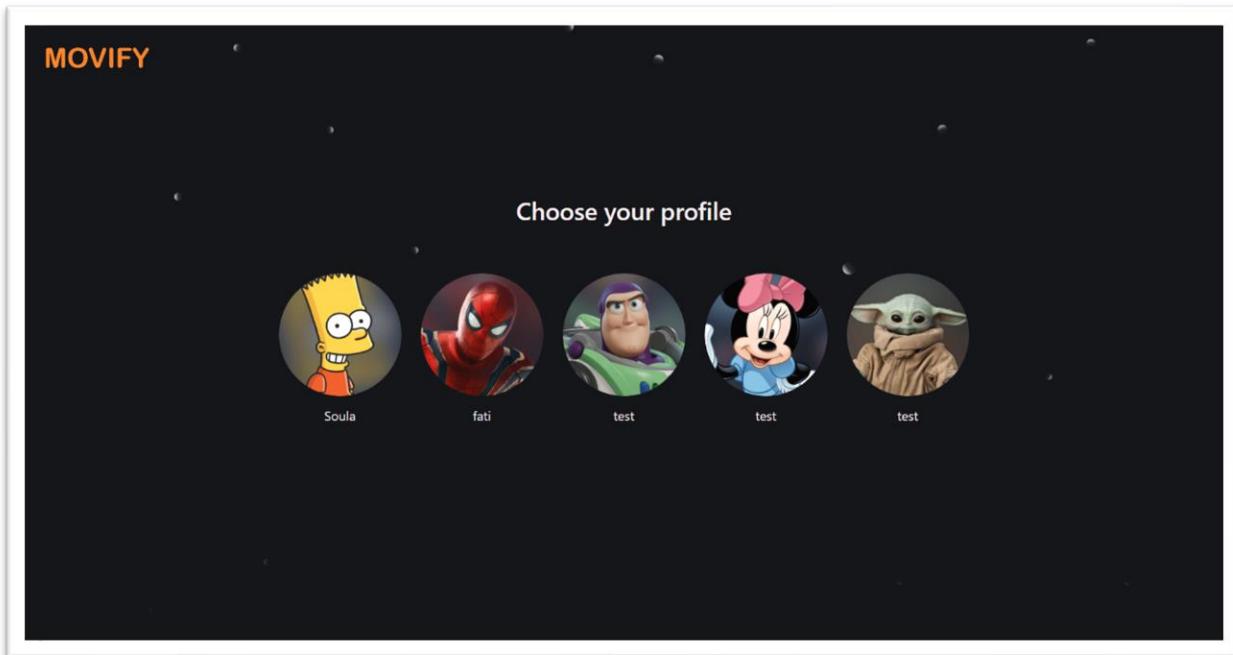


Figure 41: Page de choisir le profil

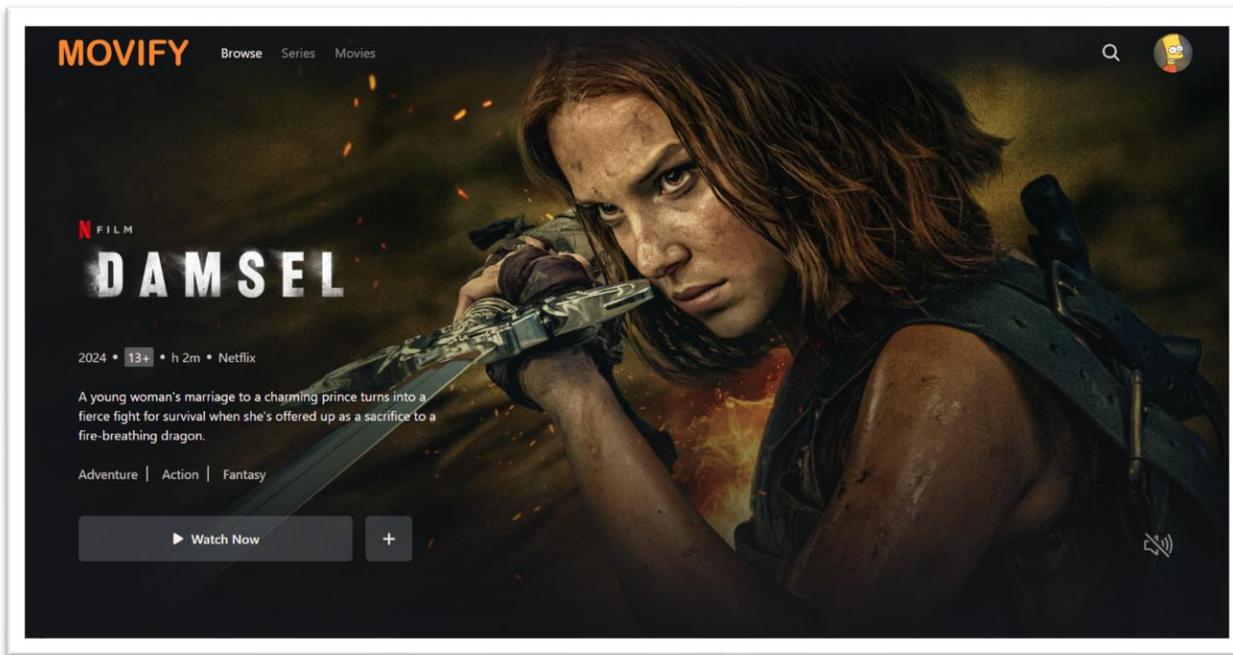


Figure 42: Page d'accueil

Récupération des Données

Lorsqu'un utilisateur demande des informations sur un film, une requête GET est envoyée par le frontend.

Le backend interroge la base de données Cassandra, récupère les informations du film et les renvoie au frontend sous forme de JSON, où elles sont affichées dynamiquement.

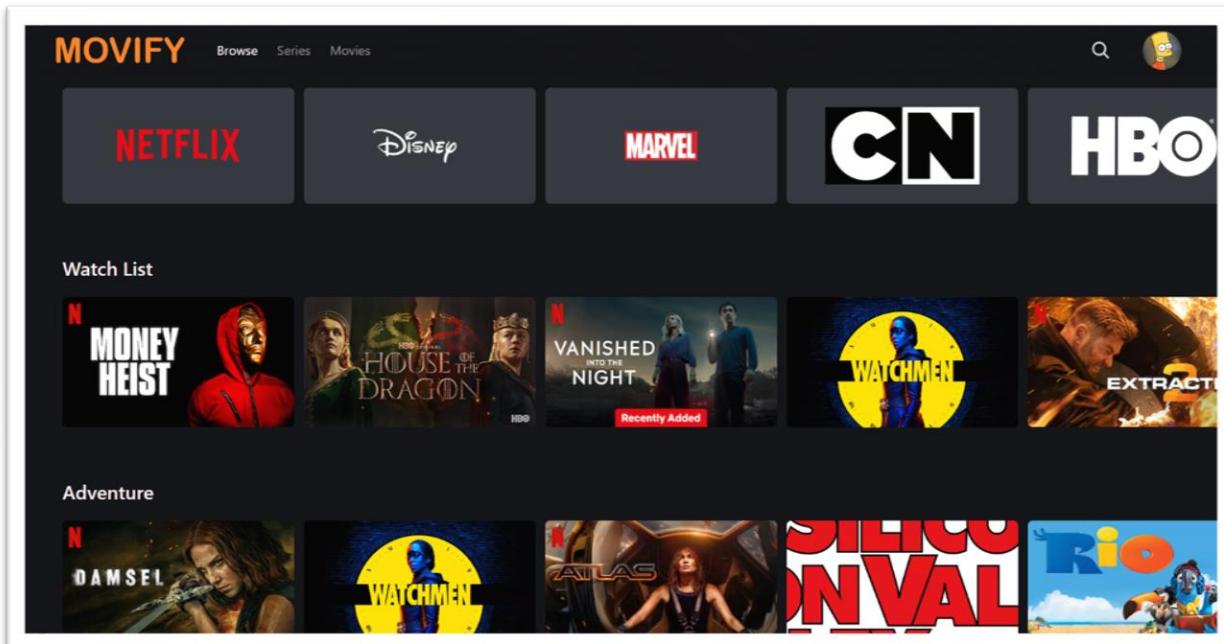


Figure 43: Choisir (film / serie)

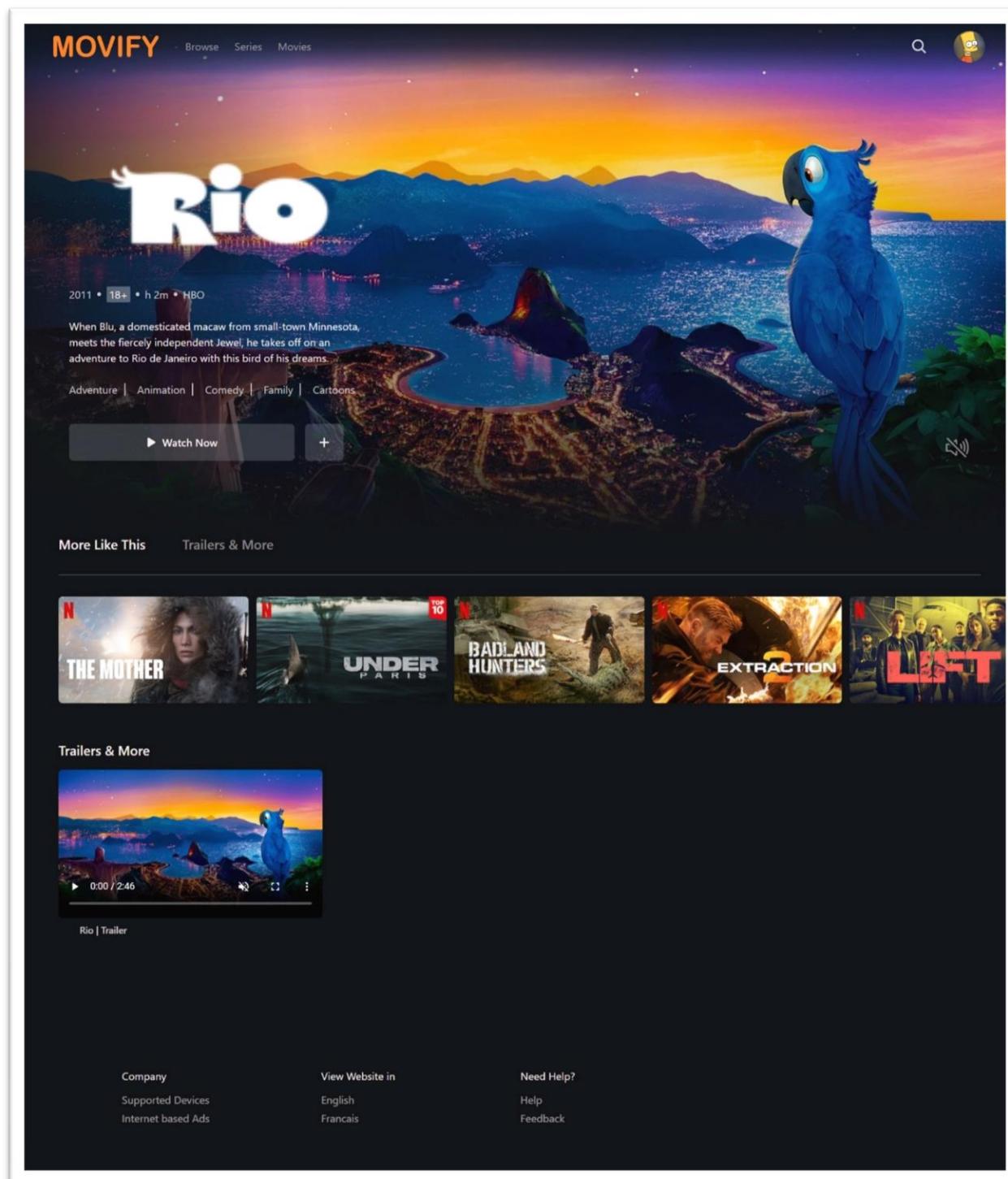


Figure 44: Page (film / serie)

Mise à jour en Temps Réel

Lorsqu'un utilisateur ajoute un film à sa liste de favoris, une requête POST est envoyée au backend, qui met à jour la base de données.

Le frontend reflète instantanément ces changements à l'utilisateur grâce à une mise à jour de l'état de l'application.

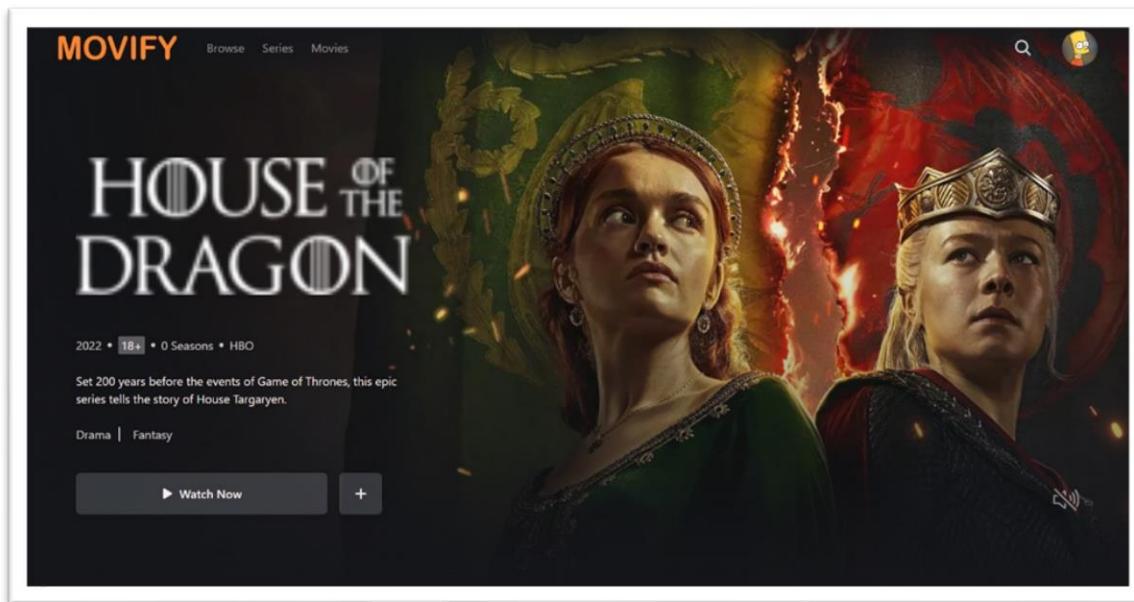


Figure 45: Page d'un film

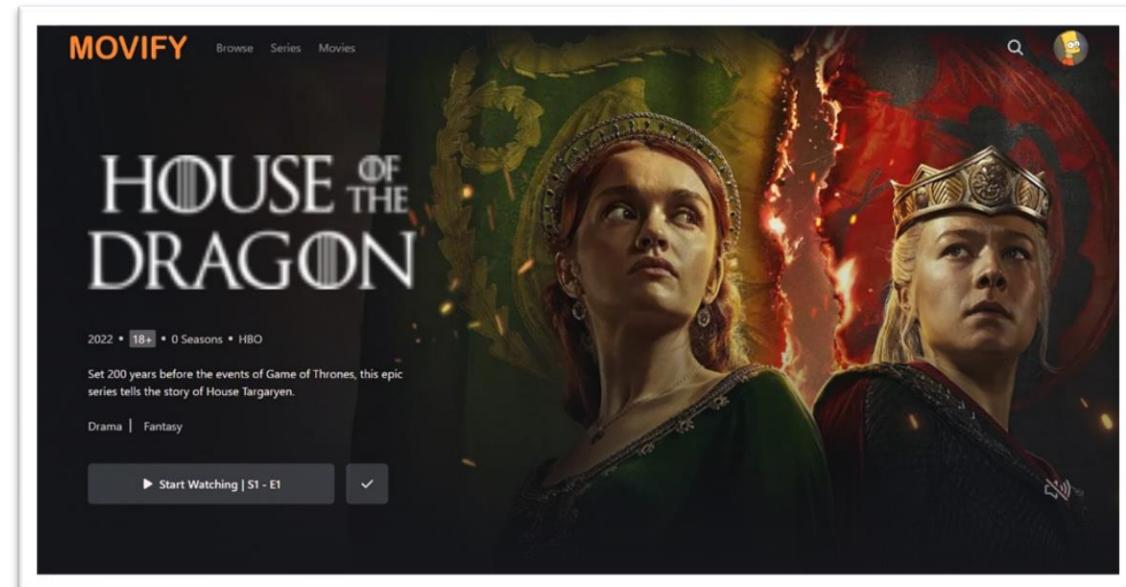


Figure 46: Page d'un film apres ajouter aux favoris

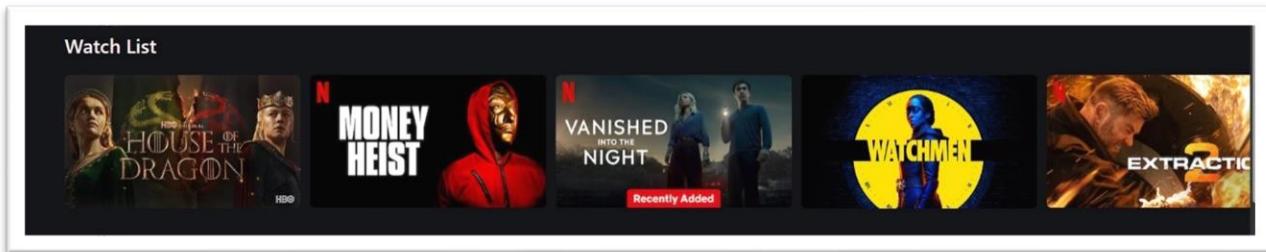


Figure 47: Watch list

III. Processus de payment

Stripe est un fournisseur de services de paiement qui permet aux commerçants d'accepter les cartes de crédit et de débit ou d'autres paiements. Sa solution de traitement des paiements, Stripe Payments, est particulièrement adaptée aux entreprises qui réalisent la plupart de leurs ventes en ligne, car la plupart de ses fonctionnalités uniques sont principalement orientées vers les ventes en ligne [22].

How does Stripe payment processing work?

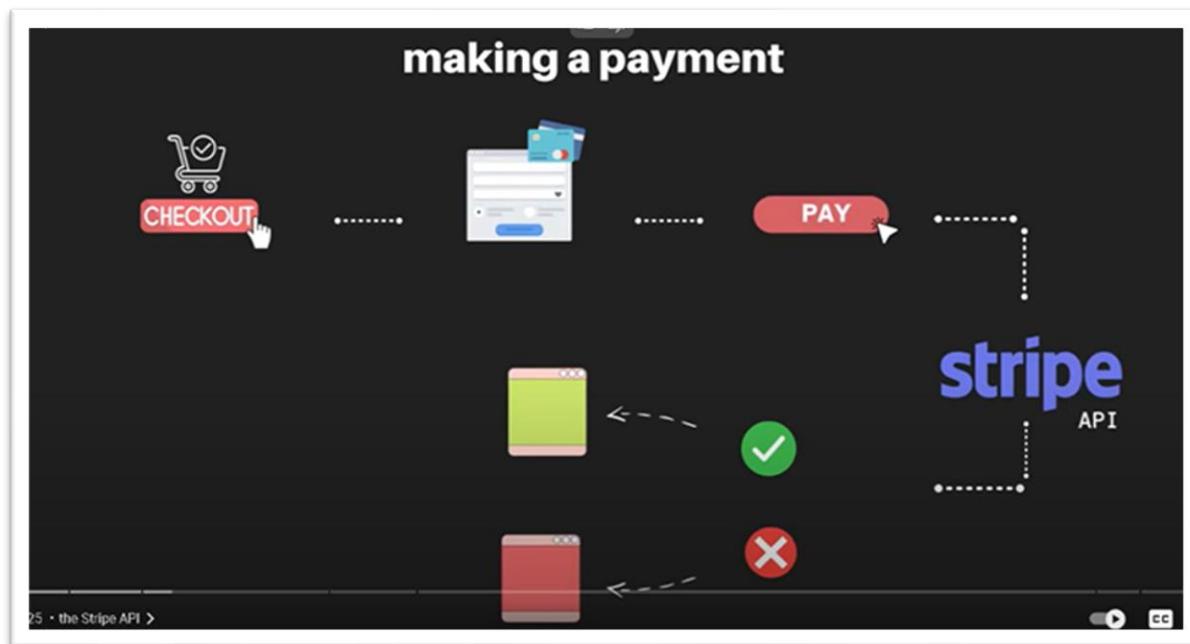


Figure 48: Comment fonctionne stripe ?

Stripe traite les paiements en six étapes.

1. Le client fournit les informations de sa carte, en ligne ou en personne ;
2. Ces informations de carte entrent dans la passerelle de paiement de Stripe, qui crypte les données ;
3. Stripe envoie ces données à l'acquéreur, qui est une banque qui traitera la transaction au nom du commerçant. Dans cette étape, Stripe fait office de commerçant (le propriétaire de l'entreprise étant un sous-commerçant). Cela signifie que les utilisateurs de Stripe n'ont pas besoin de créer un compte marchand, ce qui peut être fastidieux ;
4. Le paiement passe par un réseau de cartes de crédit, comme Visa ou Mastercard, jusqu'à la banque émettrice du titulaire de la carte ;
5. La banque émettrice approuve ou refuse la transaction ;
6. Ce signal circule de la banque émettrice via le réseau de cartes jusqu'à l'acquéreur, puis via la passerelle jusqu'au client, qui voit un message lui indiquant que le paiement a été accepté ou refusé.

Une fois que la banque émettrice du titulaire de la carte a finalisé son approbation, vous pouvez transférer des fonds de Stripe vers votre compte bancaire professionnel.

- Les clients Stripe peuvent recevoir des paiements une fois le traitement des transactions terminé (généralement dans les deux jours ouvrables) ;
- Les paiements peuvent également être effectués selon un calendrier de votre choix (quotidien, hebdomadaire ou mensuel).

Les frais Stripe que vous devrez payer varient en fonction du type de transaction facilitée :

- 2,7 % plus 5 cents pour les transactions en personne.
- 2,9 % plus 30 cents pour les transactions en ligne.
- 3,4 % plus 30 cents pour les transactions saisies manuellement.
- 4,4 % plus 30 cents pour les transactions par carte internationale.

X MOVIFY

Subscribe now and start streaming



Cancel anytime. (Effective at the end of your billing period)

Experience Disney+ content in Arabic

Exclusive originals

Easy-to-use parental controls

No ads or additional charges

Create up to 7 profiles

Stream on up to 4 devices at once

All 5.00 USD

Anime 3.00 USD

Continue >

When purchasing an annual subscription, the annual price is equivalent to 10 months of the monthly subscription price.

If you subscribe to Disney+, we will charge a recurring monthly or annual fee (or equivalent) to your stored payment method.

Cancellations must be received at least 24 hours before the end of the current billing period to be effective at the end of that period.

No refunds or credits will be given for partial months or years (or equivalents), unless required by law.

Company

Supported Devices

Internet based Ads

View Website in

English

Francais

Need Help?

Help

Feedback

←  Movify TEST MODE

All
\$5.00

Pay with card

Email
ahmed@gmail.com

Card information
4242 4242 4242 4242 
02 / 25 444 

Cardholder name
Ahmed Test

Country or region
Morocco

Pay 

Powered by  Terms Privacy

Movify  Search Developers Test mode   

Transactions  

Recommendation Use the Stripe Dashboard mobile app to quickly accept, review, or refund payments on the go.  

All	Succeeded	Refunded	Disputed	Failed	Uncaptured
42	6	0	0	0	0

Amount Payment method Description Customer Date Refunded date Dispute amount

| \$5.00 USD 4242 pi_3Pch0BvSAzweEr019P2yxQ ahmed@gmail.com Sep 16, 4:04 PM — — ... |
| \$5.00 USD — pi_3PyjQBvSAzweEr00BYiVytZ Sep 14, 12:01 AM — — ... |
| \$5.00 USD 4242 pi_3Px70JBvSAzweEr001c80mj t soulaimane@gmail.com Sep 9, 12:48 PM — — ... |
| \$5.00 USD 4242 pi_3Pwpm0BvSAzweEr01vcyHDc1 soulaimane@gmail.com Sep 8, 6:26 PM — — ... |
| \$5.00 USD — pi_3Pt6GnBvSAzweEr0123sMvBp Aug 29, 11:12 AM — — ... |
| \$5.00 USD — pi_3P6F2FBvSAzweEr00fd8BH2x Aug 29, 10:57 AM — — ... |
| \$5.00 USD — pi_3PxLEBvSAzweEr00tWicea6 Aug 28, 8:19 PM — — ... |
| \$5.00 USD — pi_3PqwfPBvSAzweEr00YauJf6 Aug 23, 12:06 PM — — ... |
| \$5.00 USD — pi_3PFKzPBvSAzweEr015rFwbe7 Jul 22, 12:05 PM — — ... |
| \$5.00 USD 4242 pi_3Pd0KgBvSAzweEr00Dobf1r soulaimane@gmail.com Jul 16, 3:30 PM — — ... |

Viewing 1-20 of 42 results  

Références

- [1] C. texte fournit des informations générales S. ne peut garantir que les informations soient complètes ou exactes E. raison de cycles de mise à jour variables et L. S. P. A. D. D. P. R. Q. C. R. D. L. Texte, « Thème: Streaming vidéo dans le monde », Statista. Consulté le: 14 septembre 2024. [En ligne]. Disponible sur: <https://fr.statista.com/themes/9169/streaming-video-dans-le-monde/>
- [2] « Qu'est-ce qu'un diagramme de Gantt ? », Gantt.com. Consulté le: 15 septembre 2024. [En ligne]. Disponible sur: <https://www.gantt.com>
- [3] « Qu'est-ce qu'un diagramme UML et à quoi sert-il ? | Miro ». Consulté le: 14 septembre 2024. [En ligne]. Disponible sur: <https://miro.com/fr/diagramme/qu-est-ce-qu-un-diagramme-uml/>
- [4] « Tout ce que vous devez savoir sur le diagramme de cas d'utilisation ». Consulté le: 14 septembre 2024. [En ligne]. Disponible sur: <https://fr.venngage.com/blog/diagrammes-de-utilisation/>
- [5] « Diagramme de classes UML : définition | Lucidchart ». Consulté le: 14 septembre 2024. [En ligne]. Disponible sur: <https://www.lucidchart.com/pages/fr/diagramme-de-classes-uml>
- [6] « Qu'est-ce qu'un diagramme de séquence UML ? », Lucidchart. Consulté le: 14 septembre 2024. [En ligne]. Disponible sur: <https://www.lucidchart.com/pages/fr/diagramme-de-sequence-uml>
- [7] « Qu'est-ce que le cloud ? | Définition cloud | Cloudflare ». Consulté le: 15 septembre 2024. [En ligne]. Disponible sur: <https://www.cloudflare.com/fr-fr/learning/cloud/what-is-the-cloud/>
- [8] « Photoshop - Notions de base et intermédiaires - Cégep Limoilou | Brio ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://www.brioeducation.ca/en/courses-and-activities/photoshop-notions-de-base-et-intermediaires-boccpo658/detail>
- [9] I. Links, « Qu'est-ce que ReactJS et pourquoi devrions-nous utiliser ReactJS ? | Ibraci Links ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://ibracilinks.com/blog/quest-ce-que-reactjs-et-pourquoi-devrions-nous-utiliser-reactjs>
- [10] « Qu'est-ce que le JavaScript ? - Apprendre le développement web | MDN ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: https://developer.mozilla.org/fr/docs/Learn/JavaScript/First_steps/What_is_JavaScript
- [11] « Qu'est-ce que le HTML ? Un guide pour débutants ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://kinsta.com/fr/base-de-connaissances/qu-est-ce-que-le-html/>
- [12] « Qu'est-ce que le CSS ? Feuilles de style en cascade expliquées simplement - IONOS ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://www.ionos.fr/digitalguide/sites-internet/web-design/quest-ce-que-le-css/>
- [13] « Tailwind CSS, le framework totalement personnalisable – Numendo ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://www.numendo.com/blog/framework/tailwind-css-framework-totalement-personnalisable/>
- [14] « What is Java? - Java Programming Language Explained - AWS ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://aws.amazon.com/what-is/java/>
- [15] « Qu'est-ce que Spring Boot ? » [En ligne]. Disponible sur: <https://www.axopen.com/spring-boot-lyon/>
- [16] « Qu'est-ce que MySQL ? Une explication simple pour les débutants ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://kinsta.com/fr/base-de-connaissances/qu-est-ce-que-mysql/>
- [17] « Qu'est-ce que Cassandra ? » [En ligne]. Disponible sur: <https://www.ovhcloud.com/fr/public-cloud/apache-cassandra/#:~:text=Qu'est%2Dce%20que%20Cassandra,les%20donn%C3%A9es%20pour%20les%20restructurer.>
- [18] « Qu'est-ce que Python ? – Le langage Python expliqué – AWS ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://aws.amazon.com/fr/what-is/python/>
- [19] « What is AWS? Ultimate guide to Amazon Web Services ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://www.techtarget.com/searchaws/definition/Amazon-Web-Services>

- [20] « Git - What is Git? ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F>
- [21] « Jira | Issue & Project Tracking Software | Atlassian ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://www.atlassian.com/software/jira>
- [22] « What Is Stripe, and How Does It Work to Accept Payments? - NerdWallet ». Consulté le: 16 septembre 2024. [En ligne]. Disponible sur: <https://www.nerdwallet.com/article/small-business/what-is-stripe>