



Université de la Manouba

École Nationale des Sciences de l'Informatique



RAPPORT DU PROJET DE CONCEPTION ET DE DÉVELOPPEMENT

Sujet : Conception et Développement d'une
application d'analyse et de prédiction des
audiences TV

Auteurs :

M. ABDELKEFI MOHAMED NAIM

M. GARGOURI AMIN

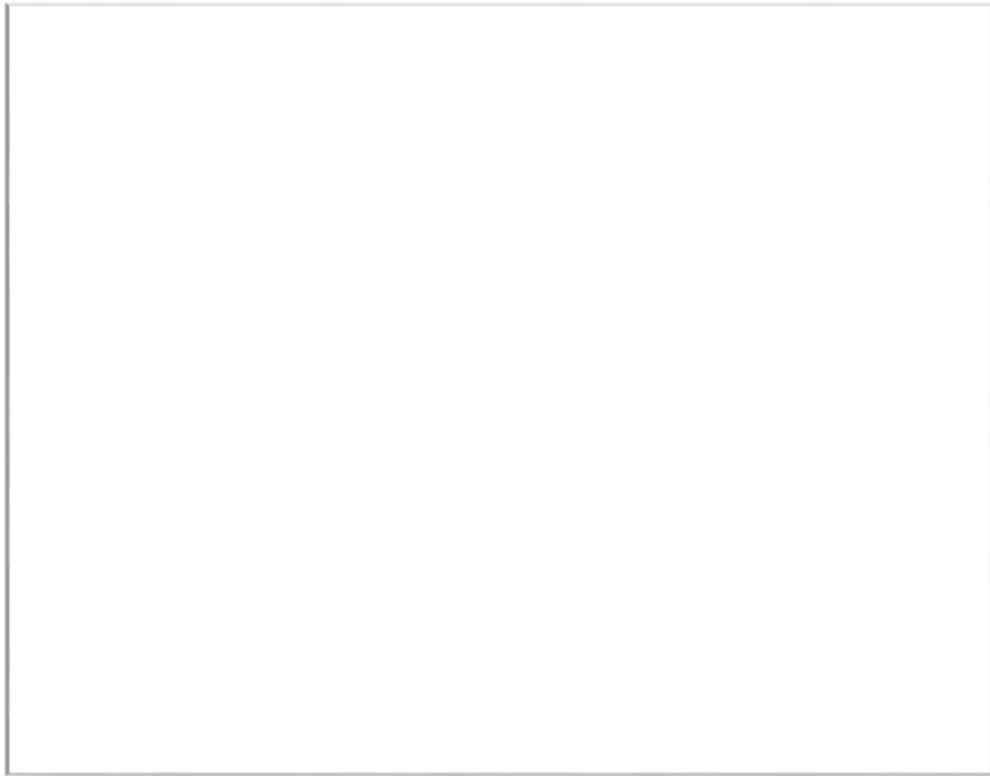
Encadrant :

Pr. BEN ABDELKADER CHIRAZ

Année Universitaire :2017 /2018

APPRÉCIATION ET SIGNATURE DE L'ENCADRANTE

M^{ME} BEN ABDELKADER CHIRAZ



REMERCIEMENTS

Nous tenons, avant de présenter notre travail, à exprimer notre gratitude envers les personnes qui nous ont, de près ou de loin, apporter leur soutien.

Nous adressons nos plus sincères remerciements et nos sentiments de reconnaissance à notre encadrante M^{me} BENABDELKADER CHIRAZ pour ses bonnes directives, sa disponibilité et surtout ses conseils précieux qui ont largement contribué à rehausser la valeur de ce travail. Nous remercions également tous les enseignants de l'école nationale des sciences de l'informatique pour la qualité de l'enseignement qu'ils nous ont prodigué durant nos études.

Que les membres de jury trouvent, ici, l'expression de nos gratitude pour l'honneur qu'ils nous font en acceptons de juger ce travail.

Table des matières

| | |
|---|-----------|
| Introduction générale | 8 |
| Chapitre 1 : Étude préalable | 9 |
| Introduction | 9 |
| 1.1 Contexte général du projet | 9 |
| 1.2 Étude de l'existant | 9 |
| 1.2.1 Présentation des solutions existantes | 10 |
| 1.2.1.1 Médiamétrie | 10 |
| 1.2.1.2 Acacia Web | 11 |
| 1.2.1.3 Audimat.tn | 12 |
| 1.2.2 Critique de l'existant | 13 |
| 1.2.2.1 Critères de comparaison | 13 |
| 1.2.2.2 Comparaison des applications cités | 13 |
| 1.2.3 Problèmes dégagés | 14 |
| 1.3 Solution adopté et travail demandé | 14 |
| Conclusion | 15 |
| Chapitre 2 : Analyse et spécification des besoins | 16 |
| Introduction | 16 |
| 2.1 Spécification informelle des besoins | 16 |
| 2.1.1 Définition des acteurs | 16 |
| 2.1.2 Analyse des besoins | 17 |
| 2.1.2.1 Besoins fonctionnels | 17 |
| 2.1.2.2 Besoins non fonctionnels | 18 |
| 2.2 Spécification semi-formelle des besoins | 18 |
| 2.2.1 Diagrammes de cas d'utilisation | 19 |
| 2.2.1.1 Diagramme de cas d'utilisation relatif à l'administrateur | 19 |
| 2.2.1.2 Diagramme de cas d'utilisation relatif à l'internaute | 20 |

| | | |
|---------------------------------|---|-----------|
| 2.2.2 | Description de quelques scénario | 21 |
| 2.2.2.1 | Scénario du cas d'utilisation "S'authentifier" | 21 |
| 2.2.2.2 | Scénario du cas d'utilisation "Consulter les audiences" | 22 |
| 2.2.2.3 | Scénario du cas d'utilisation "Créer les CronJob" | 23 |
| | Conclusion | 23 |
| Chapitre 3 : Conception | | 24 |
| | Introduction | 24 |
| 3.1 | Conception architecturale de l'application | 24 |
| 3.1.1 | Architecture physique | 24 |
| 3.1.1.1 | Description des architectures Types | 24 |
| 3.1.1.2 | Choix de l'architecture de l'application | 25 |
| 3.1.2 | Architecture logique | 27 |
| 3.1.2.1 | Choix du patron de conception | 27 |
| 3.1.2.2 | Fonctionnement de l'architecture choisie | 27 |
| 3.2 | Conception détaillé | 28 |
| 3.2.1 | Diagrammes de classes | 29 |
| 3.2.1.1 | Diagramme de classes du couche d'accès au données DAO | 29 |
| 3.2.1.2 | Diagramme de classes du couche Service | 30 |
| 3.2.2 | Conception de la base de données : | 31 |
| 3.2.3 | Diagramme de séquence objets | 32 |
| 3.2.3.1 | Scénario d'authentification | 32 |
| 3.2.3.2 | Scénario de consultation des audiences | 33 |
| 3.2.3.3 | Scénario de gestion les données d'audiences | 34 |
| | Conclusion | 35 |
| Chapitre 4 : Réalisation | | 36 |
| | Introduction | 36 |
| 4.1 | Environnement de travail | 36 |
| 4.1.1 | Environnement de développement matériel | 36 |
| 4.1.2 | Environnement de développement logiciel | 37 |
| 4.2 | Deep learning | 38 |
| 4.2.1 | Définition | 39 |
| 4.2.2 | LSTM : Long short-term memory | 39 |

| | | |
|---------|---|-----------|
| 4.2.3 | Implémentation du modèle | 39 |
| 4.2.3.1 | Représentation des données | 40 |
| 4.2.3.2 | Entraînement du modèle | 40 |
| 4.2.3.3 | Résultat d'apprentissage | 41 |
| 4.3 | Interfaces Homme-Machine | 42 |
| 4.3.1 | Interface d'accueil | 42 |
| 4.3.2 | Interface d'inscription | 42 |
| 4.3.3 | Interface d'authentification | 43 |
| 4.3.4 | Interface administrateur | 44 |
| 4.3.5 | Interfaces de consultation d'audience | 45 |
| 4.4 | Chronogramme de travail | 46 |
| | Conclusion | 47 |
| | Conclusion Générale | 48 |
| | Nétographie | 49 |

Table des figures

| | | |
|-----|--|----|
| 1.1 | Page espace membre de Médiamétrie | 10 |
| 1.2 | Interface d'Acacia Web | 11 |
| 1.3 | Interface d'Audimat.tn | 12 |
| 1.4 | Tableau comparatif des solutions existantes | 13 |
| 2.1 | Diagramme cas d'utilisation pour l'administrateur | 19 |
| 2.2 | Diagramme cas d'utilisation pour l'internaute | 20 |
| 2.3 | Diagramme de séquence système d'authentification | 21 |
| 2.4 | Diagramme de séquence système de consultation des audiences | 22 |
| 2.5 | Diagramme de séquence système de creation des CronJob | 23 |
| 3.1 | Architecture N-tiers | 25 |
| 3.2 | Diagramme de déploiement | 26 |
| 3.3 | Architecture MVC [N11] | 27 |
| 3.4 | Diagramme de classes du couche DAO | 29 |
| 3.5 | Diagramme de classes du couche Service | 30 |
| 3.6 | Modèle relationnel de la base de données | 31 |
| 3.7 | Diagramme de séquences : Authentification | 32 |
| 3.8 | Diagramme de séquences : Consultation des audiences par Chaîne | 33 |
| 3.9 | Diagramme de séquences : Gérer les données d'audience | 34 |
| 4.1 | Exemple de données d'apprentissage | 40 |
| 4.2 | Exemple de données d'apprentissage encodées | 40 |
| 4.3 | Courbe de perte (Loss) | 41 |
| 4.4 | Page d'accueil | 42 |
| 4.5 | Interface d'inscription | 43 |
| 4.6 | Interface d'authentification | 43 |
| 4.7 | Interface admin | 44 |
| 4.8 | Interface Cron Job | 44 |

| | | |
|------|--|----|
| 4.9 | Interface de consultation de historique d'audience filtré par date | 45 |
| 4.10 | Interface de consultation d'audience prédit filtré par emission. | 45 |
| 4.11 | Interface de consultation d'audience prédit filtré par chaine. | 46 |
| 4.12 | Chronogramme de travail | 47 |

INTRODUCTION GÉNÉRALE

La télévision occupe indéniablement, une place importante dans notre société. Suivie en moyenne deux heures par jour, elle tient pour toute la première place dans l'information et les loisirs.

De ce fait, le nombre de téléspectateurs et les parts de marché des différents programmes sont communiqués régulièrement aux diffuseurs, c'est pour cela que les mesures d'audience acquièrent actuellement une importance de plus en plus croissante.

C'est dans ce cadre que s'inscrit ce projet de conception et de développement effectué au sein de l'école Nationale des Sciences de l'informatique. Ce projet consiste à concevoir et développer une application web permettant d'analyser les taux d'audience des émissions et des chaînes télévisées situées en France et de fournir principalement des prévisions fiables de ces taux.

Par le biais du présent rapport, nous détaillons les étapes du travail réalisé tout au long de ce projet. Ce rapport s'articulant sur quatre chapitres, est organisé comme suit : dans le premier chapitre, nous présentons une étude de l'existant avant d'entamer les premières phases du développement du projet. Le deuxième chapitre est consacré à la spécification des besoins fonctionnels et non fonctionnels aussi bien qu'aux cas d'utilisation du système et d'identification des acteurs. Dans le troisième chapitre, nous sommes chargés de présenter les différents aspects conceptuels de l'application. Le quatrième chapitre porte, en dernière phase du projet, sur une illustration des détails de la réalisation de notre travail. Et pour couronner le travail, nous finissons par une conclusion générale avec quelques perspectives que nous avons vues principales.

Chapitre 1

Étude préalable

INTRODUCTION

Ce chapitre a pour objectif de situer le projet dans son contexte général, à savoir la problématique qui a inspiré la création de notre application, la description du projet et les objectifs à atteindre. Dans ce qui suit, nous élaborons une étude approfondie sur des solutions existantes en mettant l'accent sur leurs insuffisances. Pour finir nous présentons la solution adoptée pour résoudre ce problème ainsi que le travail demandé.

1.1 Contexte général du projet

Il est essentiel pour une chaîne de télévision, qu'elle soit publique ou privée, de connaître son public. L'outil primordial et le plus performant pour quantifier l'écoute des téléspectateurs des chaînes télévisées reste sans doute la mesure d'audience vu que les responsables d'émissions, attachés de presse et les directeurs sont tous préoccupés par leur taux d'audience de la veille. De ce fait, l'audimétrie prouve jour après jour sa présence dans l'arène médiatique comme l'outil le plus efficace visant à aider les acteurs des industries médiatiques à prendre plus rapidement, plus précisément et de façon subjective des décisions figurant des impacts financiers et publicitaires importants.[N1]

1.2 Étude de l'existant

Dans cette partie, nous présentons une étude des solutions existantes, en se focalisant sur les apports et les limites de chacune d'elles. Cette étude va nous permettre de dégager les besoins auxquels doit répondre notre application.

1.2.1 Présentation des solutions existantes

1.2.1.1 Médiamétrie

MÉDIAMÉTRIE [N2] : C'est une société spécialisée dans la mesure d'audience et les études marketing des médias audiovisuels et interactifs en France. Elle observe, mesure et analyse les comportements du public et les tendances du marché des médias et de la communication. Cette société dispose d'une application qui a pour objectif de fournir aux visiteurs toutes ses activités à propos des mesures et des chiffres d'audience collectés.

Cette application est conçue afin de permettre à l'utilisateur de :

- Créer son espace membre dans lequel il peut ajouter toutes les actualités et les mesures d'audience préférées.
- Filtrer en un seul clic les événements de ses univers médias préférés et retrouver-les à tout moment sur sa page d'accueil personnalisée.
- Filtrer son fil d'info de sa page d'accueil en sélectionnant dans la liste des univers ceux qu'il souhaite voir apparaître.

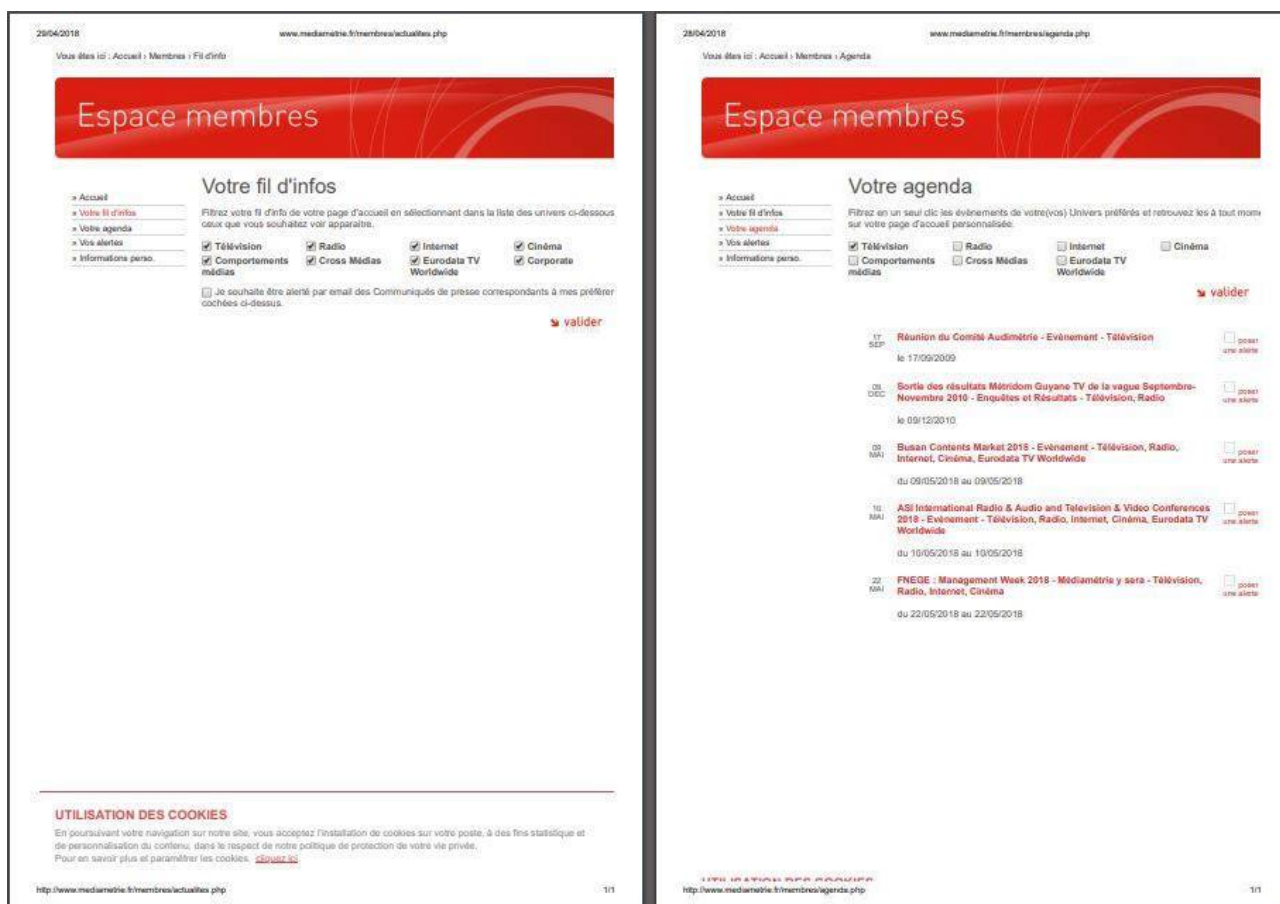


FIGURE 1.1 – Page espace membre de Médiamétrie

1.2.1.2 Acacia Web

ACACIA WEB [N3] : C'est une application d'analyse et de visualisation des audiences TV et radio. Elle est accessible depuis l'explorateur web. Elle offre également à ses utilisateurs un support pour accéder à la mesure précise et détaillée des comportements du public de la télévision en général et des principales catégories qui le composent.

Acacia Web sert l'utilisateur à :

- Choisir son environnement. L'environnement c'est tout simplement les différents régions sur lesquels les mesures d'audience sont opérées.
- Créer des requêtes auxquelles il peut choisir les différents éléments sur lesquels il souhaite produire ses résultats d'audience.
- Visualiser les résultats d'audience ainsi obtenues via des courbes qui illustrent les performances des programmes sous forme de graphes compréhensibles.
- Avoir une documentation bien élaborée via ACACIA Explorateur [N4], c'est une fonctionnalité d'aide à l'exploitation permettant de générer des rapports préformatés sous forme de fichier .PDF ou .XLS.



FIGURE 1.2 – Interface d'Acacia Web

1.2.1.3 Audimat.tn

AUDIMAT.TN [N5] : C'est le premier instrument indépendant de mesure instantanée de l'audience tunisienne des chaînes télévisées. Développé en 2011 par IRiS.tn, start-up tunisienne, suite à une collaboration entre chercheurs universitaires et ingénieurs tunisiens en informatique, en électronique, en télécommunication et en marketing, le système vient de voir le jour (1er mars 2012).

Cette plateforme permet à l'utilisateur de :

- Éclairer sa vision sur le processus de production des émissions télévisées tout en déterminant la "qualité" des programmes.
- Choisir une date instantanée illustrant en temps réel l'audience des chaînes télévisées tout en assurant une bonne approche de contrôle d'observation.
- Disposer d'un ciblage efficace des achats d'espaces publicitaires sur les chaînes télévisées tunisiennes par les boîtes de communication et offre un outil de business intelligence pour mieux gérer leur budget de communication.

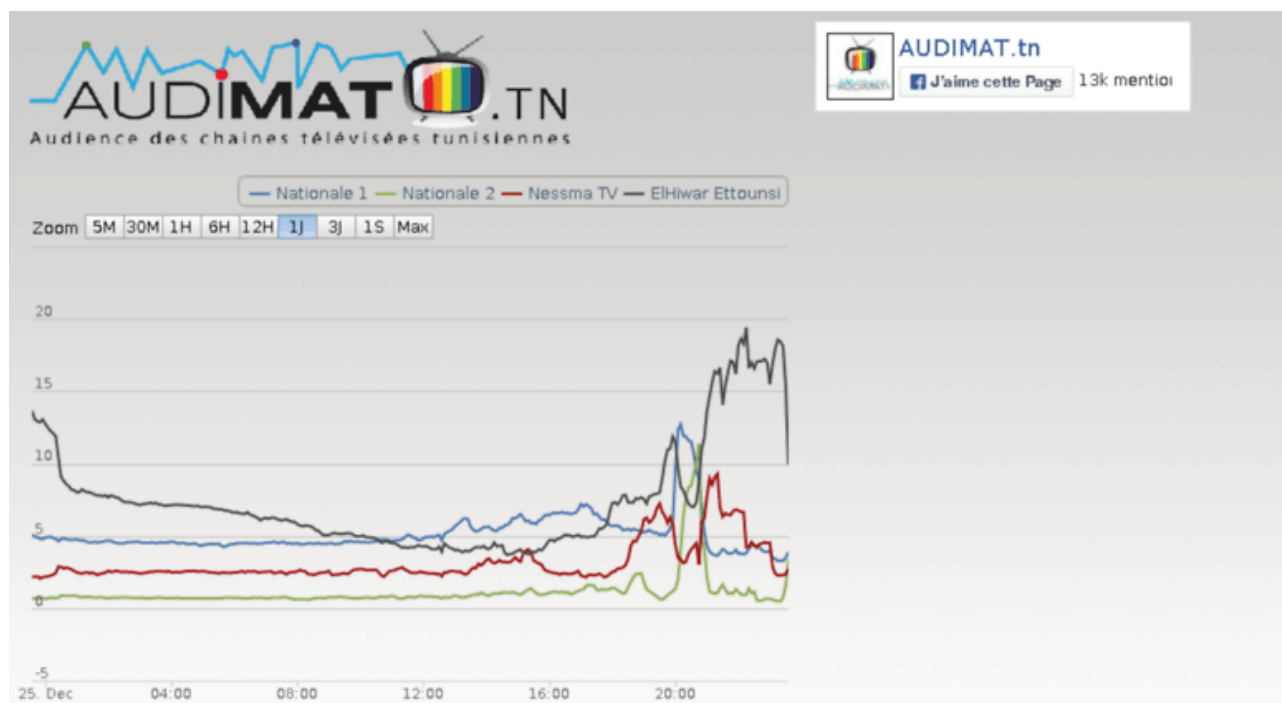


FIGURE 1.3 – Interface d'Audimat.tn

1.2.2 Critique de l'existant

Afin d'étudier de plus près les solutions présentées, nous évaluons dans cette partie les critères à prendre en considération ainsi qu'une étude comparative des outils introduits préalablement.

1.2.2.1 Critères de comparaison

Pour comparer les applications citées précédemment, nous allons nous baser sur les critères suivants :

- **Gratuité** : Est-ce que l'application permet de créer un compte gratuit ? contient elle des fonctionnalités payantes ?
- **Prédiction** : En dehors des Statistiques, est-ce que l'application fournit des données précises de prédiction et d'évaluation sur des médias spécifiques ?
- **Les médias visés** : Est-ce que l'application analyse les comportements du public des différents univers média ou elle se focalise en permanence à un type d'audience bien déterminé ?

1.2.2.2 Comparaison des applications cités

Le tableau ci-dessous présente la comparaison entre les trois solutions existantes selon les critères définis dans la partie précédente.

| Critères d'évaluation | AUDIMAT.TN | ACACIA WEB | MÉDIAMÉTRIE |
|-------------------------|-------------------------|---|--|
| Gratuité | Non | Oui pour l'installation et payante pour la consultation | Oui |
| Prédiction | Non | Non | Oui mais seulement pour les médias sociaux(internet) |
| Les médias visés | Seulement l'audience TV | visé le média TV et radio seulement | tous les types de média (Internet, cinéma, radio, TV, etc..) |

FIGURE 1.4 – Tableau comparatif des solutions existantes

1.2.3 Problèmes dégagés

Suite à l'étude qui a été faite, nous remarquons que les applications étudiées précédemment présentent quelques insuffisances auxquelles nous allons chercher à remédier. En effet, certaines applications apportent des exploitations utiles sur les résultats d'audience aidant de ce fait les directions des chaînes d'avoir une idée claire et cela à travers des rapports préformatés (MÉDIAMÉTRIE) et de fournir la possibilité aux utilisateurs de créer des requêtes auxquelles ils peuvent choisir les différents éléments sur lesquels ils souhaitent produire leurs résultats d'audience (ACACIA WEB). En plus de ça elles offrent une description instantanée à travers des mesures effectuées en tout temps et publiable en permanence (AUDIMAT.TN).

Malgré ces avantages, ces applications nécessitent des améliorations dans ses fonctionnalités puisqu'elles ne déterminent pas si un programme va réussir, s'il présentera une concurrence avec un programme d'une autre chaîne et s'il comportera des parties déficientes.

1.3 Solution adopté et travail demandé

Après une étude comparative sur les différentes solutions existantes, il est primordial au regard des limites déjà étudiées des applications existantes, de proposer une solution qui pourra répondre aux besoins de nos utilisateurs. Notre projet de conception et de développement consiste à concevoir et à développer une plateforme gratuite, présentant des améliorations par rapport aux solutions déjà citées. En effet, notre application met à disposition la possibilité aux utilisateurs de consulter des prédictions faites sur les audiences TV principalement composé des émissions et des chaînes télévisées situées en France. Pour cela, nous avons eu recours à implémenter un modèle LSTM pour la prévision des séries temporelles afin de bien couronner les résultats des tâches à prédire.

Cependant, elle ne s'agit pas d'une simple application de prédiction, elle permet aussi aux utilisateurs de choisir les différents types de filtrages sur lesquels ils souhaitent produire leurs résultats d'audience.

CONCLUSION

Dans ce chapitre, nous avons présenté le contexte général du projet suivi d'une étude approfondie de l'existant et de critique des solutions présentes. Ceci nous a permis de comprendre les besoins et d'envisager la solution la plus adéquate avec notre application. Le prochain chapitre est consacré à la présentation des besoins fonctionnels et non fonctionnels. Nous terminons par une spécification de ces besoins en nous basant sur les diagrammes d'UML.

Chapitre 2

Analyse et spécification des besoins

INTRODUCTION

Afin de respecter les critères fixés par le cahier des charges fonctionnelles, de garantir le succès et l'efficacité du projet et de définir avec précision le périmètre de la solution développée, une bonne analyse du système s'impose. Dans ce qui suit nous parvenons à une vue claire des différents besoins escomptés de notre projet. Au cours de ce chapitre, nous allons dégager d'une façon détaillée les besoins fonctionnels et les besoins non fonctionnels de notre application ainsi que les diagrammes des cas d'utilisation des différents acteurs et les divers scénarios qui expliquent ces cas.

2.1 Spécification informelle des besoins

La spécification informelle est l'ensemble des exigences fonctionnelles et non fonctionnelles exprimées en langage naturel. Cette étape permettra de faciliter la modélisation de nos besoins plus tard.

2.1.1 Définition des acteurs

Notre application comprend certains acteurs qui interagissent entre eux afin d'exploiter le site à travers ses interfaces. Dans ce qui suit, nous allons présenter deux types d'acteurs :

- L'internaute : C'est le visiteur de notre site qui a la possibilité de découvrir ses fonctionnalités.
- L'administrateur : il est à la fois considéré comme un internaute et le responsable de la mise à jour des informations et du contrôle des activités du site.

2.1.2 Analyse des besoins

Les besoins sont divisés en deux catégories, à savoir les besoins fonctionnels et les besoins non fonctionnels.

2.1.2.1 Besoins fonctionnels

Ce sont les actions et les réactions que le système doit faire suite à une demande d'un acteur principal. Tenant compte de la nature de l'application, on distingue les besoins par acteurs :

Internaute :

- Créer un compte : Le système doit permettre à l'internaute de s'inscrire en donnant son email et son mot de passe qui doit être confirmé par la suite pour la réussite de l'inscription.
- Gérer un compte : Le système doit permettre à l'internaute de gérer son compte. Ce besoin comporte les sous besoins suivants :
 1. Afficher le profil : Le système doit lui permettre de consulter son profil.
 2. Mettre à jour son profil : Le système doit lui permettre de mettre à jour les données de son profil.
- S'authentifier : Le système doit permettre l'authentification de l'utilisateur.
- Le système doit permettre à l'internaute de consulter les historiques d'audience des émissions et des chaînes télévisés pour une date antérieure donnée.
- Saisir le type de filtrage : Le système doit permettre à l'internaute de gérer le comportement des données d'audience en saisissant un filtrage par date, par émission ou par chaîne.
- Le système doit permettre à l'internaute de consulter les audiences prédites pour une date ultérieure donnée et bornée sur deux ou trois mois au maximum.
- Le système doit permettre à l'internaute d'enregistrer les résultats d'audience dans son profil.
- Le système doit permettre à l'internaute d'exporter au format PDF ou JPEG les résultats d'audience obtenues.

Administrateur :

- Le système doit permettre à l'administrateur de gérer les données d'audience : Cette fonctionnalité permet à l'administrateur d'ajouter, modifier et supprimer des données particulières.

- Créer des CronJob : Le système doit permettre à l'administrateur de créer des tâches automatiques(Cron Job) qui s'excutent selon une planification.
Les Cron Job : Ce sont des tâches qui s'exécutent automatiquement selon une planification, ils ont pour rôle de mettre à jour les données d'audience d'une façon automatique et de lancer l'algorithme d'apprentissage pour prédire de nouveaux ratios.
- Gérer les CronJob : Le système doit permettre à l'administrateur d'ajouter, modifier et supprimer des CronJob
- Le système doit permettre à l'administrateur de gérer les comptes des utilisateurs.

2.1.2.2 Besoins non fonctionnels

La solution que propose notre application doit garantir les critères suivants :

- Simplicité : l'application doit être simple à gérer par l'utilisateur.
- Rapidité : l'application doit être rapide pendant le chargement et le traitement des données, pour gagner du temps.
- Efficacité : l'application doit assurer l'efficacité pour aider vraiment l'utilisateur à prévoir les taux d'audiences des chaînes télévisées.
- Maintenabilité : Le code de l'application doit être bien lisible et compréhensible pour pouvoir le maintenir facilement et rapidement. Ceci facilite la mise à jour de l'application et permet l'amélioration de certains détails ainsi que l'ajout d'autres fonctionnalités.
- Sécurité : L'application doit assurer un niveau minimum de sécurité pour les informations traitées.

2.2 Spécification semi-formelle des besoins

La spécification semi-formelle se base sur une notation graphique, elle introduit un aspect formel mais des diagrammes annotés généralement par du texte informel. Ces notations sont particulièrement pratiques pour fournir une vue d'ensemble, statique ou dynamique, d'un système ou d'un sous-système. Cependant, leur sémantique doit être précisée. Pour ce qui suit, nous nous référerons aux diagrammes d'UML : les diagrammes de cas d'utilisation et les diagrammes de séquence.

2.2.1 Diagrammes de cas d'utilisation

Pour illustrer les fonctionnalités offertes par notre système, nous avons opté pour les diagrammes de cas d'utilisation. Ces diagrammes donnent une vue sur les fonctionnalités de notre système ainsi que les acteurs qui l'utilisent.

2.2.1.1 Diagramme de cas d'utilisation relatif à l'administrateur

La figure ci-dessous représente le diagramme de cas d'utilisation relatif à l'administrateur de notre application.

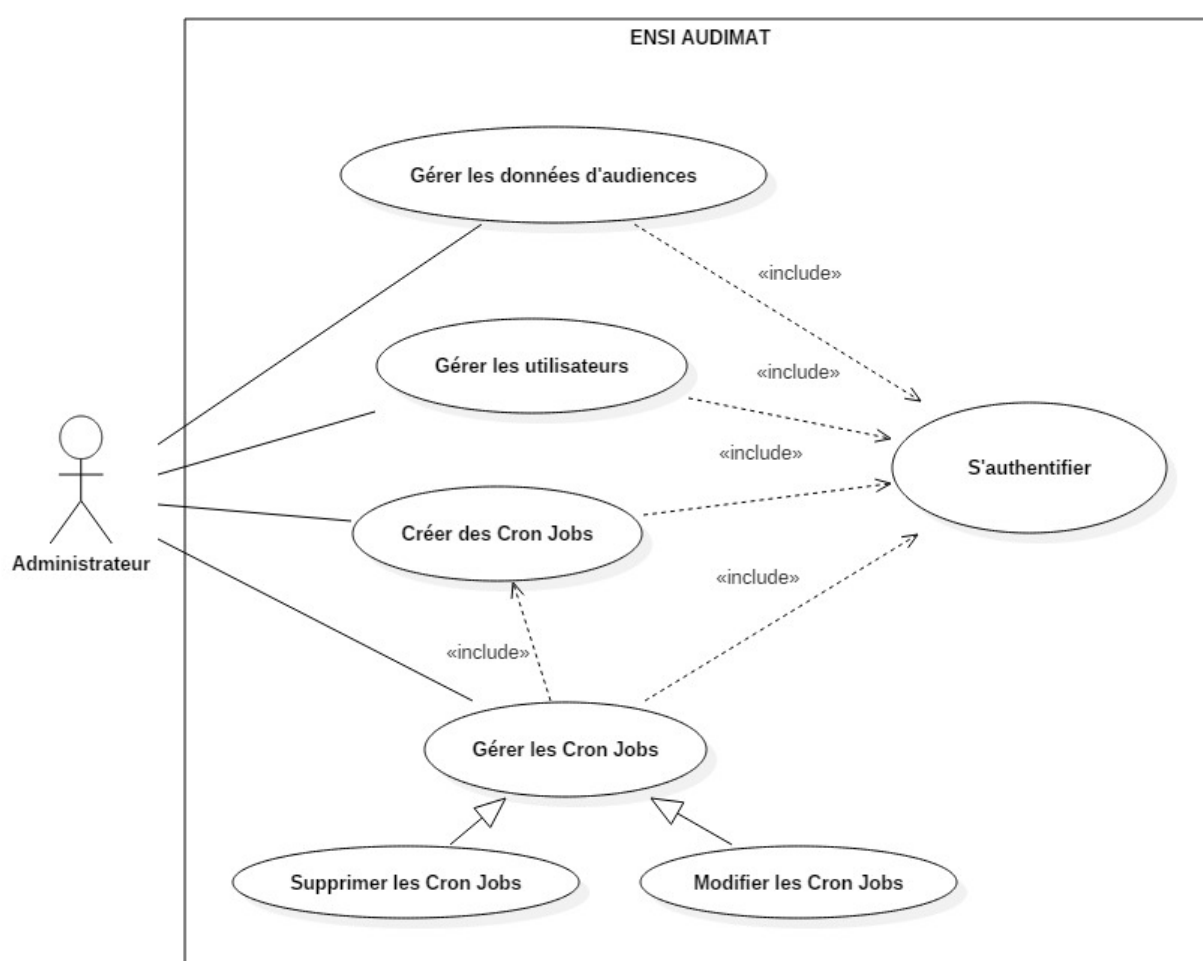


FIGURE 2.1 – Diagramme cas d'utilisation pour l'administrateur

Ce diagramme cas d'utilisation présente les différentes fonctionnalités que l'administrateur de notre application peut faire. L'administrateur doit s'authentifier pour accéder à son espace sur la plateforme. L'administrateur peut accéder à son propre espace où il peut créer et gérer des CronJobs qui s'exécutent selon une planification afin d'effectuer certaines tâches dont notre plateforme aura besoin pour rester propre, stable et bien rangé. L'administrateur peut également gérer les données d'audiences et c'est celui le responsable des comptes utilisateurs.

2.2.1.2 Diagramme de cas d'utilisation relatif à l'internaute

La figure ci-dessous représente le diagramme de cas d'utilisation relatif à l'internaute.

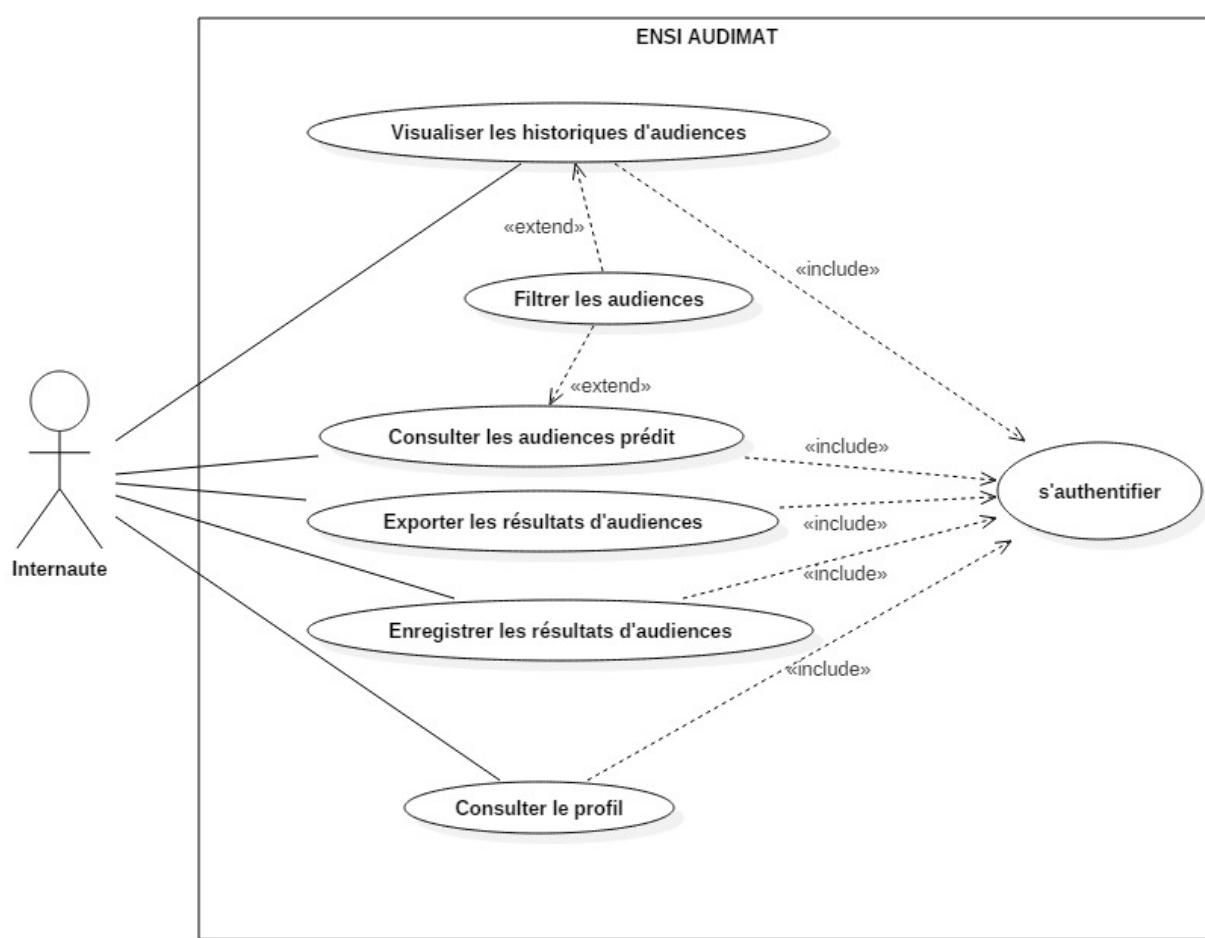


FIGURE 2.2 – Diagramme cas d'utilisation pour l'internaute

Ce diagramme de cas d'utilisation présente les différentes tâches qu'un l'internaute peut effectuer en accédant à son espace sur la plateforme. L'internaute doit s'authentifier pour accéder à son espace. Il peut consulter les historiques d'audiences qui existent sur la plateforme, il

peut suivre également les audiences prédit et effectuer des filtrages sur des chaînes et des émissions télévisées afin de suivre leur comportement au cour du temps. L'internaute peut aussi enregistrer les résultats d'audiences ou les exporter sous les formats JPEG ou PDF.

2.2.2 Description de quelques scénario

2.2.2.1 Scénario du cas d'utilisation "S'authentifier"

La figure ci-dessous représente le diagramme de séquence système du scénario d'authentification pour un utilisateur :

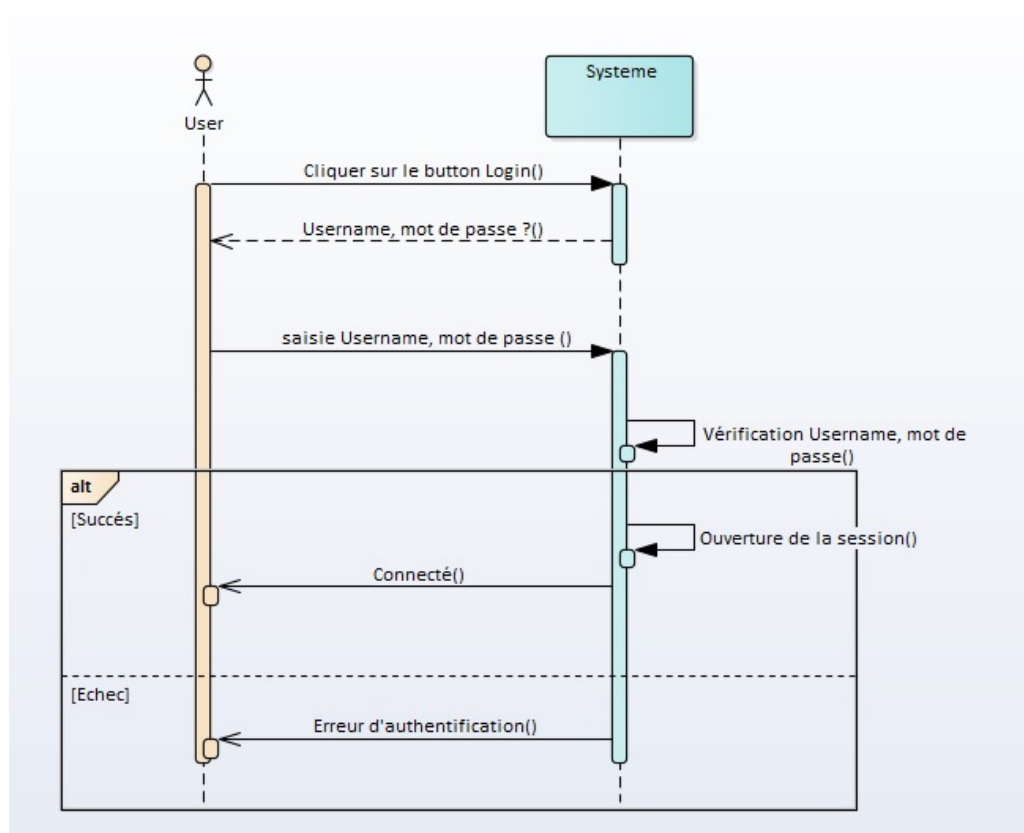


FIGURE 2.3 – Diagramme de séquence système d'authentification

Le cas d'utilisation commence lorsque l'utilisateur saisit l'URL de la page d'authentification.

Enchaînement nominal : L'utilisateur doit être enregistré dans la base de données.

Postconditions : aucune

Enchaînement nominal :

1. L'utilisateur accède à l'application.
2. L'utilisateur cliquer sur le bouton Login

3. le système fournit un formulaire d'authentification.
4. L'utilisateur doit saisir son identifiant et son mot de passe.
5. Le système vérifie les coordonnées saisies.
6. L'utilisateur est connecté.

Enchaînement alternatif :

Les coordonnées saisies sont incorrectes : L'enchaînement démarre au point 5 du scénario nominal. le système indique que l'identifiant ou le mot de passe est erroné. La séquence nominale reprend au point 4.

2.2.2.2 Scénario du cas d'utilisation "Consulter les audiences"

La figure ci-dessous représente le diagramme de séquence système du scénario de consultation des audiences pour un utilisateur :

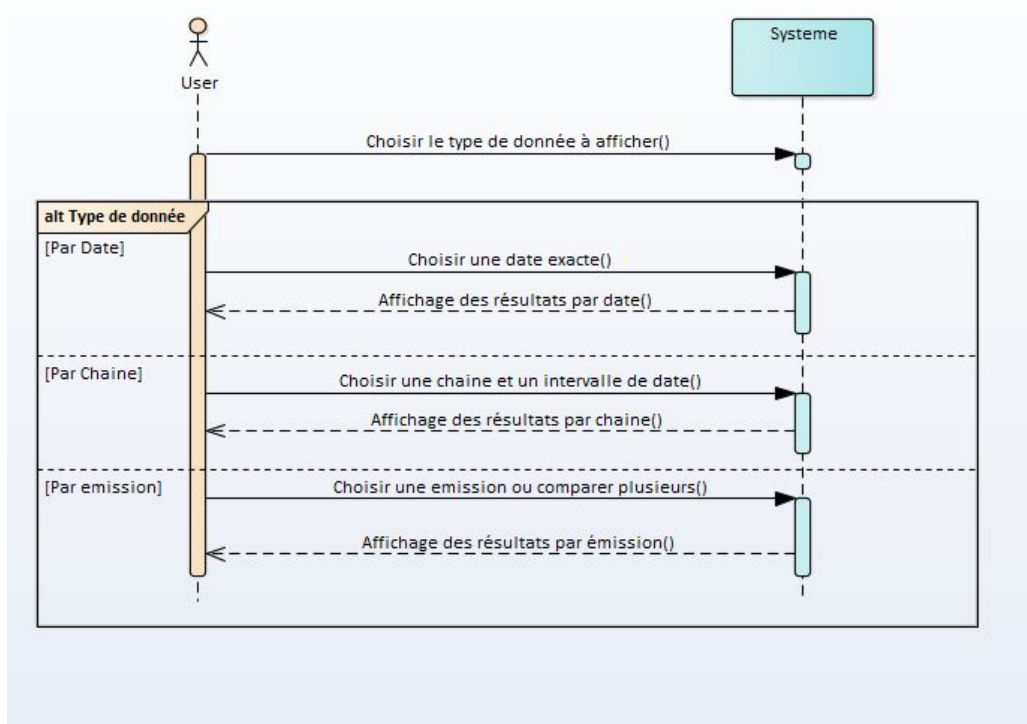


FIGURE 2.4 – Diagramme de séquence système de consultation des audiences

Pour consulter les audiences, l'utilisateur doit, d'abord, s'authentifier ensuite il peut choisir le type de donnéesw à consulter, selon son choix il saisit les paramètres nécessaires. Le système affiche les résultats correspondants.

2.2.2.3 Scénario du cas d'utilisation "Créer les CronJob"

La figure 2.5 ci-dessous représente le diagramme de séquence système du scénario de creation des CronJob pour un administrateur :

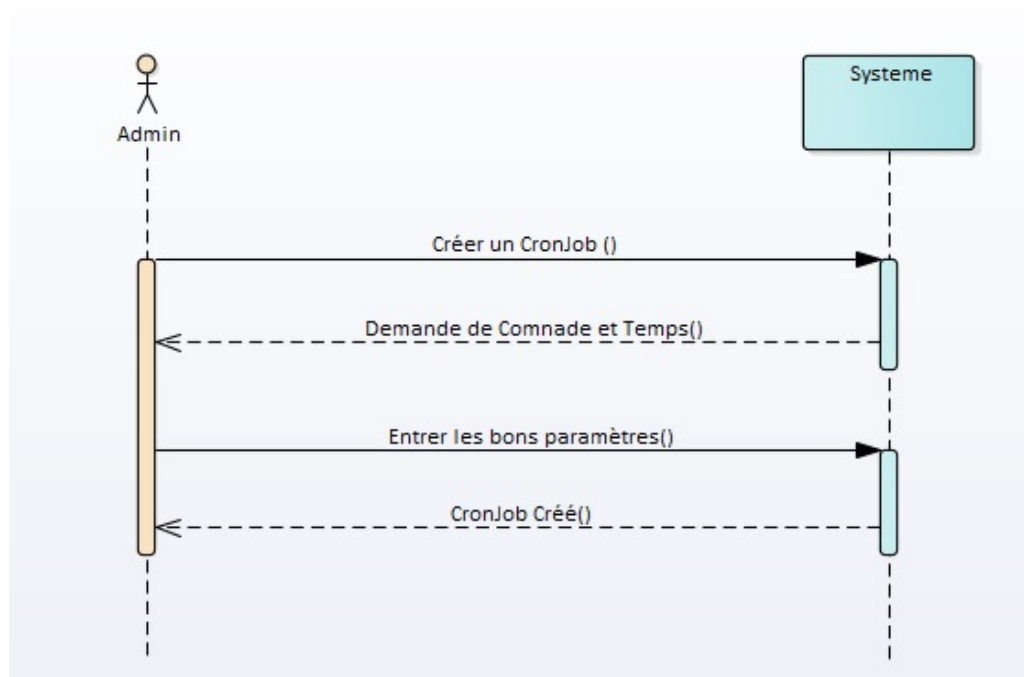


FIGURE 2.5 – Diagramme de séquence système de creation des CronJob

Pour créer les CronJob, l'administrateur doit, d'abord, s'authentifier ensuite il saisit les paramètres nécessaires. Le système crée le CronJob correspondant.

CONCLUSION

Ce chapitre nous a permis de couvrir quelques cas d'utilisation concernant les différents acteurs de notre application et de définir les besoins non fonctionnels. En effet, cette phase donne une vue claire du travail et permet d'initier à la phase qui s'occupe de la modélisation de notre application et qui sera décrite dans le chapitre suivant.

Chapitre 3

Conception

INTRODUCTION

Dans ce chapitre, nous allons entamer la tâche la plus importante dans l'élaboration de ce travail, à savoir la tâche de conception. En effet, nous présentons, en premier temps, l'architecture générale de notre application afin d'en extraire les différents modules qui la composent. Puis, nous détaillons chacun de ces modules conformément à la notation UML par la description des différents diagrammes de séquences relatifs aux cas d'utilisation qui ont été exprimés dans le chapitre précédent.

3.1 Conception architecturale de l'application

3.1.1 Architecture physique

Avant de détailler l'architecture de notre application, il est recommandé d'avoir une vue globale sur les différentes architectures types existantes.

3.1.1.1 Description des architectures Types

Les principales architectures à décrire dans ce paragraphe sont au nombre de trois à savoir : l'architecture 2-tiers, l'architecture 3-tiers et l'architecture N-tiers.

Architecture à deux niveaux (2-tiers) [N6] : Aussi appelé l'architecture « client-serveur » est assez simple. Vous avez d'un côté le client et de l'autre le serveur. Ce genre d'architecture peut se faire sur tout type d'architecture matérielles interconnectées. Concrètement, comment ça fonctionne ?

Le client demande un service au serveur comme par exemple la page Home.html, le serveur reçoit cette requête http, il effectue un traitement, et renvoie la réponse contenant la ressource demandée par le client.

Architecture à trois niveaux (3-tiers) [N7] : Dans l'architecture 3-tiers, un nouveau niveau fait son apparition. En effet, nous avons toujours le niveau 1 qui est le client. Le client est très légers étant donné qu'il n'a aucun rôle de traitement. Au niveau 2 nous avons le serveur d'application et enfin, au dernier niveau le serveur de base de donnée.

Les points importants d'une application 3-tiers :

- Le client ne sert qu'à requêter et à afficher les réponses du serveur.
- Le serveur lui s'occupe des calculs et même de requêter des serveurs additionnels.

Architecture à plusieurs niveaux (N-tiers) [N8] : Une architecture n-tiers comprend généralement une couche de présentation, une couche applicative, une couche objet métier et une couche d'accès aux données.

- La couche présentation : elle contient les différents types de clients légers, lourds ou riches.
- La couche applicative : elle contient les traitements représentant les règles métier.
- La couche d'objets métier : elle est représentée par les objets du domaine, c'est à dire l'ensemble des entités persistantes de l'application.
- La couche d'accès aux données : elle contient les usines d'objets métier, c'est à dire les classes chargées de créer et manipuler des objets métier de manière totalement transparente, indépendamment de leur mode de stockage (SGBDR, Fichier XML, ...).



FIGURE 3.1 – Architecture N-tiers

3.1.1.2 Choix de l'architecture de l'application

Pour la réalisation de notre application, on avait opté pour l'architecture 3-tiers. Ce choix est justifié par le fait que ce type d'architecture :

- Offre une flexibilité beaucoup plus importante que l'architecture 2-tiers. En effet, la portabilité du tiers serveur permet d'envisager une allocation et ou modification dynamique au grés des besoins évolutifs.
- Réduit les coûts de déploiement et d'administration. En effet, l'avantage principal d'une architecture trois tiers est la facilité de déploiement. L'application en elle-même n'est déployée que sur la partie serveur . Le client ne nécessite qu'une installation d'un navigateur web compatible avec l'application pour qu'il puisse y accéder. Cette facilité de déploiement aura pour conséquence de permettre une évolution régulière du système.
- Une sécurité garantie : avec ce type d'architecture l'accès à la base n'est effectué que par le serveur web, C'est le seul à connaître la façon de se connecter à cette base. Il ne partage aucune des informations permettant l'accès aux données, en particulier le login et le password de la base.[N9]

Pour illustrer le déploiement de notre application, nous avons utilisé le diagramme de déploiement, comme le montre la figure 3.2, qui illustre la disposition physique suivante qui prévoit dans sa vue la fragmentation de l'application en trois niveaux :

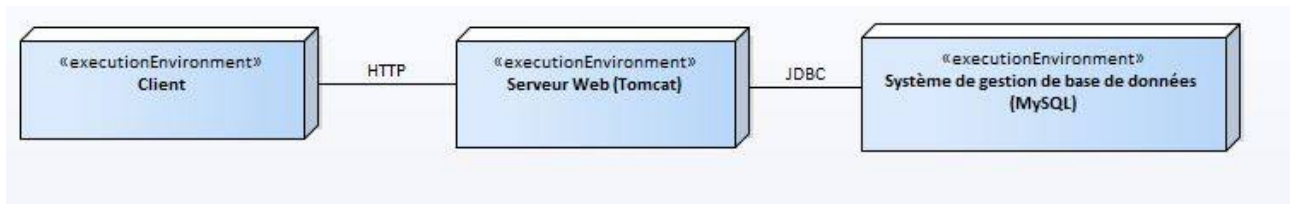


FIGURE 3.2 – Diagramme de déploiement

- **Le client** : ce tiers est passif. Il devra avoir accès à un navigateur web seulement où les utilisateurs lancent leurs demandes sous forme de http request et en reçoivent des réponses http.
- **Le serveur web** : Ce tiers est responsable de l'ensemble des traitements. Il regroupe notre couche présentation et l'interface reliant notre application avec le serveur de base de données à travers le connecteur JDBC.
- **Le serveur de base de données** : ce tier contiendra la base de données de notre application.

3.1.2 Architecture logique

3.1.2.1 Choix du patron de conception

Pour notre application, nous avons choisi de travailler avec le patron de Conception MVC. Le Modèle-Vue-Contrôleur (en abrégé MVC, de l'anglais Model-View-Controller) est une architecture et une méthode de conception qui organise l'interface homme-machine (IHM) d'une application logicielle. Ce paradigme divise l'IHM en un modèle (modèle de données), une vue (présentation, interface utilisateur) et un contrôleur (logique de contrôle, gestion des événements, synchronisation), chacun ayant un rôle précis dans l'interface [N10].

Nous l'avons choisi vu ses avantages nombreux :

- Il permet de formaliser l'application et de faciliter la compréhension de son mode de fonctionnement.
- Il permet de délimiter les différents modules de l'application et de simplifier leur création.

Les composants du modèle MVC sont :

- **Model** : la base de l'application. Il décrit les données et contrôle leur intégrité.
- **Vue** : L'interface avec laquelle interagit l'utilisateur. Elle est en interaction avec le modèle ainsi que le contrôleur.
- **Contrôleur** : Responsable de la synchronisation entre le modèle et la vue. Il gère les événements reçus par la vue et demande les changements nécessaires au modèle.

3.1.2.2 Fonctionnement de l'architecture choisie

Au sein de l'architecture 3-tiers adoptée, nous représentons dans la figure ci-dessous l'architecture logique de notre l'application :

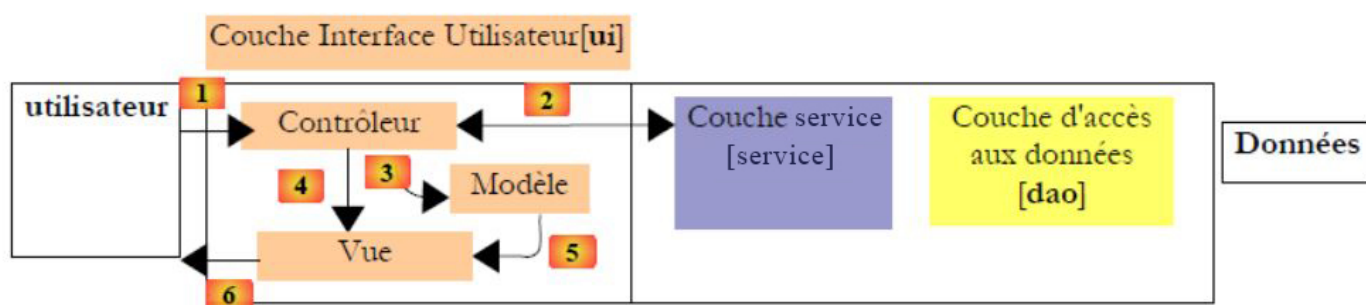


FIGURE 3.3 – Architecture MVC [N11]

Pour mieux éclairer le déroulement et le fonctionnement de notre architecture, nous allons expliquer les différentes étapes de la figure 3.3 ci-dessus :

- **Première étape** : l'utilisateur fait une demande au contrôleur. Celui-ci exécute toutes les actions demandées par l'utilisateur par une classe contenue dans le paquet suivant : **com.audience.tvprediction.controller**
- **Deuxième étape** : Le contrôleur traite une demande particulière de l'utilisateur. Pour ce faire, il peut avoir besoin de l'aide de la couche service. Une fois la demande de l'utilisateur traitée, celle-ci peut appeler diverses réponses.
- **Troisième étape** : Notre contrôleur choisit ensuite la réponse à envoyer pour l'utilisateur, cette phase nécessite plusieurs étapes :
 1. Choisir l'objet qui va générer la réponse. C'est ce qu'on appelle la vue V, le V de MVC. Ce choix dépend en général du résultat de l'exécution de l'action demandée par l'utilisateur.
 2. Lui fournir les données dont il a besoin pour générer cette réponse. En effet, celle-ci contient le plus souvent des informations calculées par la couche service. Ces informations forment ce qu'on appelle le modèle M de la vue, le M de MVC.
 3. La dernière étape consiste donc en le choix d'une vue V et la construction du modèle M nécessaire à celle-ci
- **Quatrième étape** : Le contrôleur demande à la vue choisie de s'afficher. Il s'agit d'une classe contenue dans le paquet **com.audience.tvprediction.view**
- **Cinquième étape** : Dans cette étape le générateur de vue View utilise le modèle préparé par le contrôleur Controller pour initialiser les parties dynamiques de la réponse qu'il doit envoyer à l'utilisateur
- **Sixième étape** : Dans la dernière étape la réponse est envoyée à l'utilisateur. La forme exacte de celle-ci dépend du générateur de vue. Dans notre cas, nous avons utilisé des fichiers ZUL étendus de la bibliothèque Zkoss afin d'avoir une bonne lisibilité des Charts.

3.2 Conception détaillée

Dans cette section, nous allons détailler la partie conception de notre application et présenter les différents diagrammes nécessaires ainsi qu'un plan du site à réaliser.

3.2.1 Diagrammes de classes

Dans le présent paragraphe, nous essayons de présenter une vue statique de notre application à travers le recours aux diagrammes de classes.

3.2.1.1 Diagramme de classes du couche d'accès au données DAO

Les classes de cette couche, représentées dans la figure ci-dessous, sépare nos objets métier de la base de données et assure la communication avec cette dernière en utilisant l'ORM Hibernate.

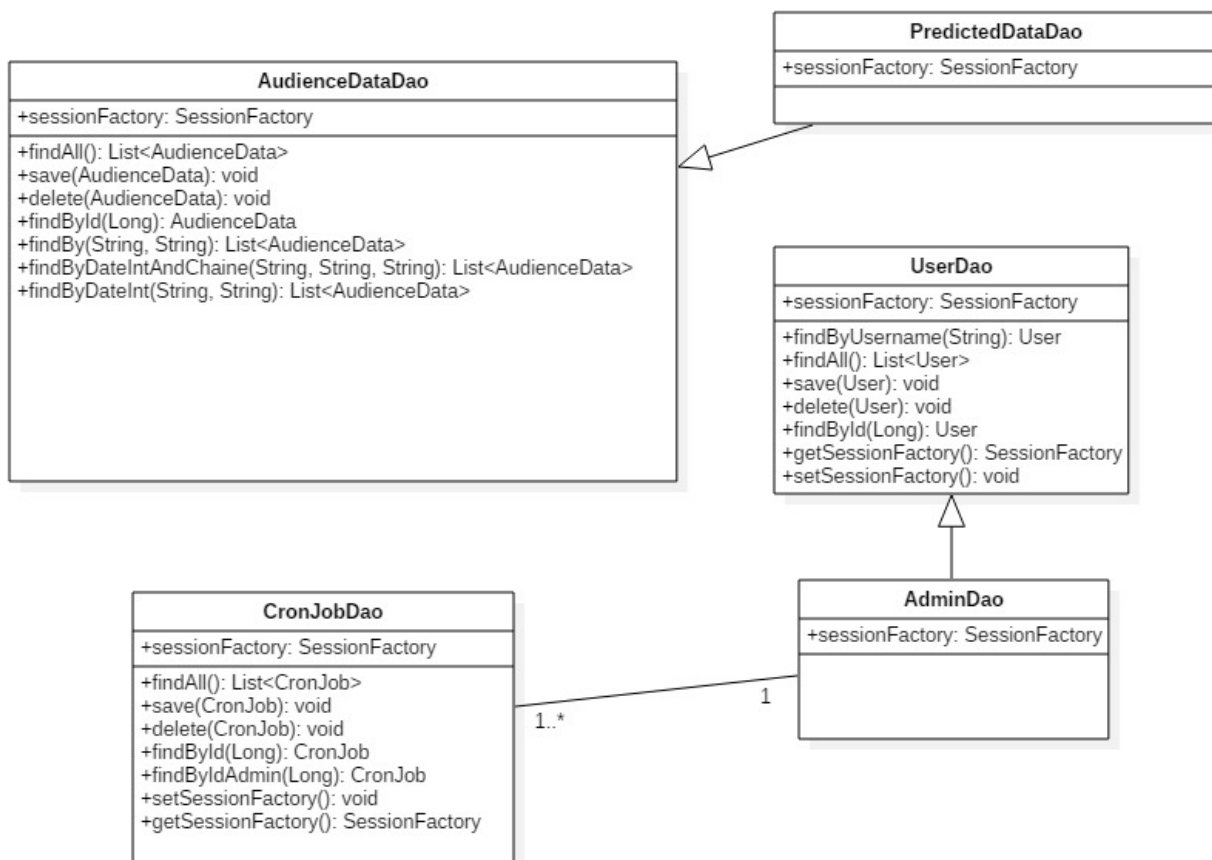


FIGURE 3.4 – Diagramme de classes du couche DAO

3.2.1.2 Diagramme de classes du couche Service

Les classes de cette couche, représentées dans la figure ci-dessous, assurent l'implémentation des fonctionnalités qu'offre notre application.

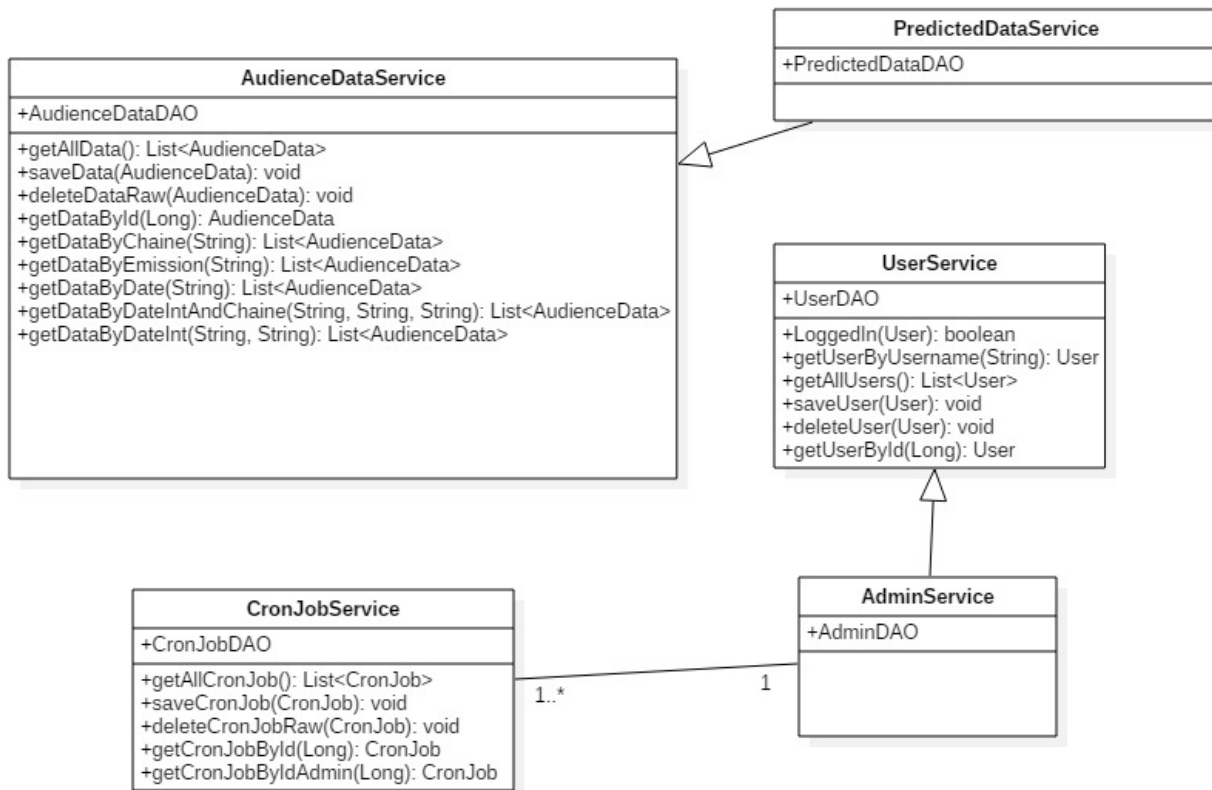


FIGURE 3.5 – Diagramme de classes du couche Service

Elles sont décrites comme suit :

AudienceDataService : Cette classe est relative aux opérations effectuées sur les données d'audience stockée dans la base de données : gestion, consultation.

PredictedDataService : Cette classe hérite de la classe "AudienceDataServices" et assure les mêmes fonctionnalités ais pour les données prédites.

UserService : Cette classe réfère à la gestion des utilisateurs.

AdminService : Cette classe hérite de la classe "UserServices" et assure les mêmes fonctionnalités. Elle se diffère de sa classe mère par les fonctionnalités spécifiques dédiées à l'administrateur comme la gestion des données et la gestion des Cron Job.

CronJobService : Cette classe est relative aux opérations effectuées sur les CronJob (Création, gestion, planification exécution etc..)

3.2.2 Conception de la base de données :

Dans cette section, nous nous intéressons à la conception de la base de données de notre système. Nous présentons dans la figure ci-dessous un modèle relationnel qui détaille notre base de données.

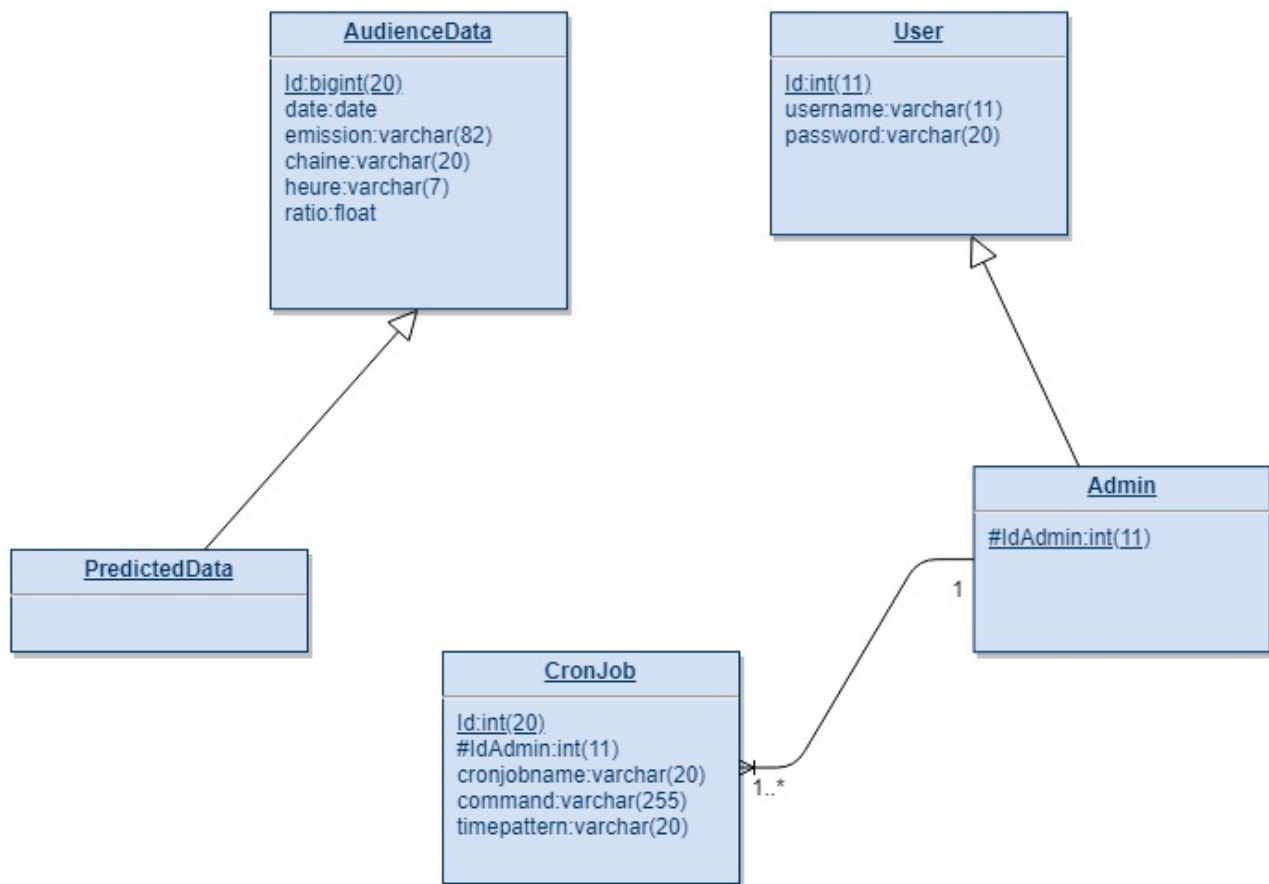


FIGURE 3.6 – Modèle relationnel de la base de données

Notre base de donnée est formée de cinq tables :

AudienceData : Cette table est réservée pour les données d'audience qui sont représentées par des dates antérieures.

PredictedData : Cette table de la table "AudienceData". Elle contient les données des audiences prédites.

User : Cette table regroupe l'ensemble d'informations des utilisateurs de notre application.

Admin : Cette table regroupe l'ensemble d'informations des administrateurs de l'application. Elle hérite les attributs de la table "User".

CronJob : Cette table contient les informations concernant un CronJob à savoir son nom, sa

commande et son timepattern. Il est à noter que la colonne "IdAdmin" est une clé étrangère qui référence la table Admin. Chaque CronJob appartient nécessairement à un Admin.

3.2.3 Diagramme de séquence objets

Pour modéliser l'aspect dynamique du système, nous présentons dans ce paragraphe quelques diagrammes de séquences. Un diagramme de séquences objet a pour but de montrer les relations entre objets d'un point de vue temporel en mettant l'accent sur la chronologie des envois de messages. Nous présentons dans ce qui suit des diagrammes de séquences objet de quelques scénarios.

3.2.3.1 Scénario d'authentification

La figure ci-dessous représente le diagramme de séquence objet d'authentification

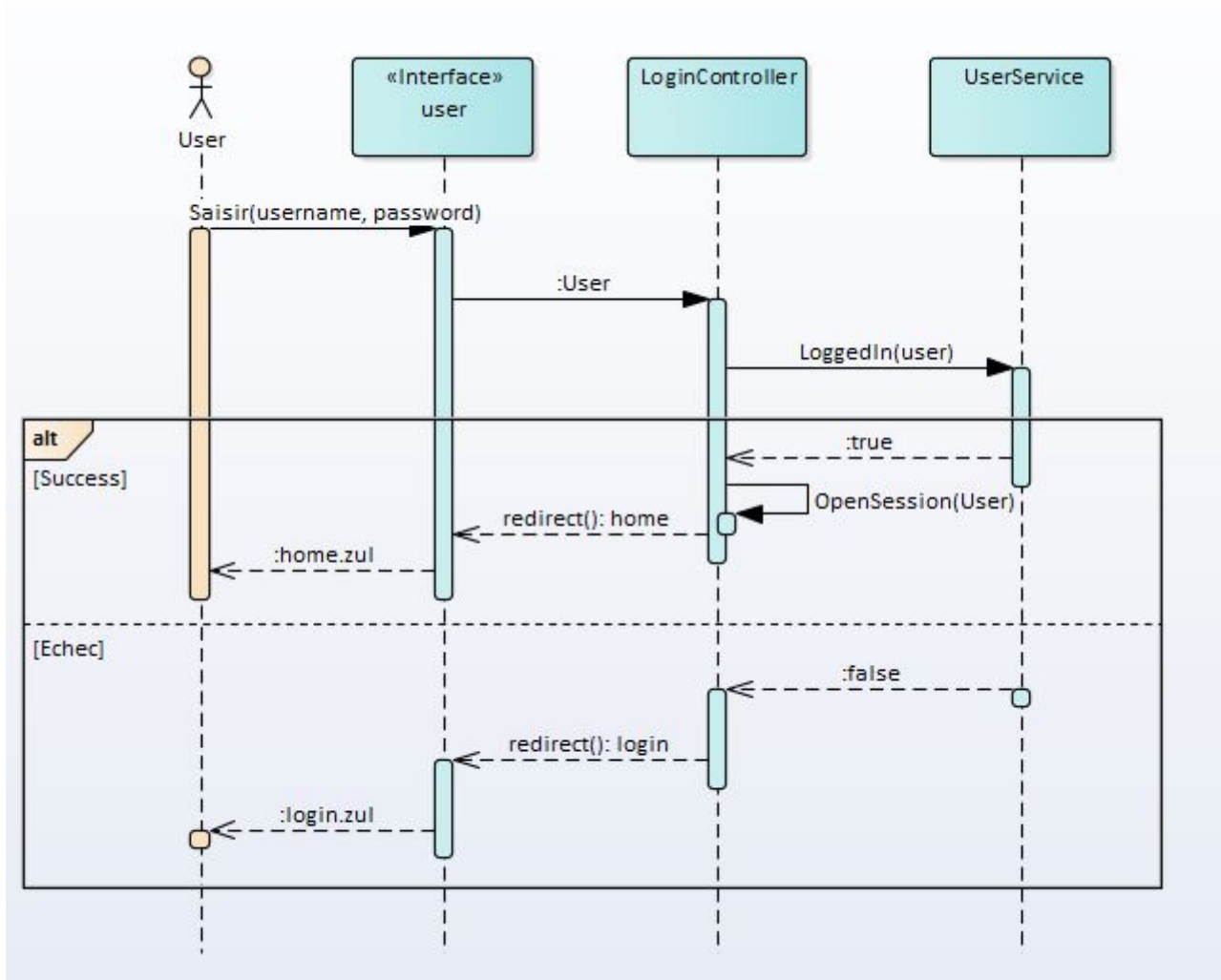


FIGURE 3.7 – Diagramme de séquences : Authentification

Un utilisateur saisit son username et son password via l'interface d'authentification, les classes "LoginController" et "UserService" se chargent d'exécuter cette action en vérifiant si les données saisies sont correctes à travers la méthode "LoggedIn(User)". Si les données saisies par l'utilisateur sont erronées alors il doit saisir à nouveau le login et/ou le mot de passe.

3.2.3.2 Scénario de consultation des audiences

La figure 3.8 présente le diagramme de séquence objet de consultations des audiences filtrées par chaîne.

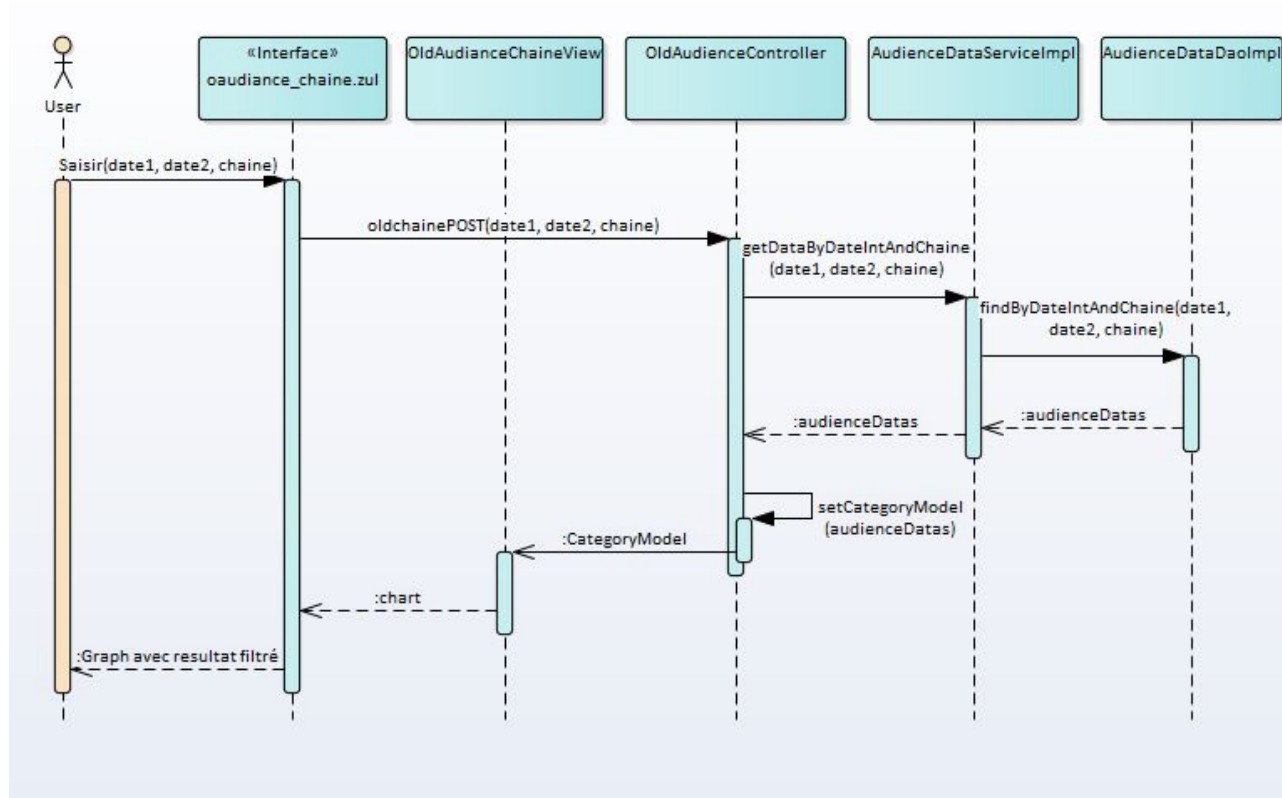


FIGURE 3.8 – Diagramme de séquences :Consultation des audiences par Chaine

L'utilisateur, après avoir choisi la nature de consultation (historique ou prédiction) et le type de filtrage des données à afficher (chaîne,date ou émission), il saisit les paramètres nécessaires pour sa demande. Ces derniers seront traités par le contrôleur qui fait appel à la couche service. Une fois la demande de l'utilisateur est traitée, celle-ci peut appeler diverses réponses. Le contrôleur choisit ensuite l'objet qui va générer la réponse et fournit les données dont il a besoin au modèle "setCatogoryModel(audienceDatas)". Enfin, le contrôleur demande à la vue

de s'afficher et c'est dans cette étape que le générateur de vue utilise le modèle préparé par le contrôleur pour initialiser les parties dynamiques de la réponse qu'il doit envoyer à l'utilisateur.

3.2.3.3 Scénario de gestion les données d'audiences

La figure ci-dessous représente le diagramme de séquence objet de gestion de données d'audience

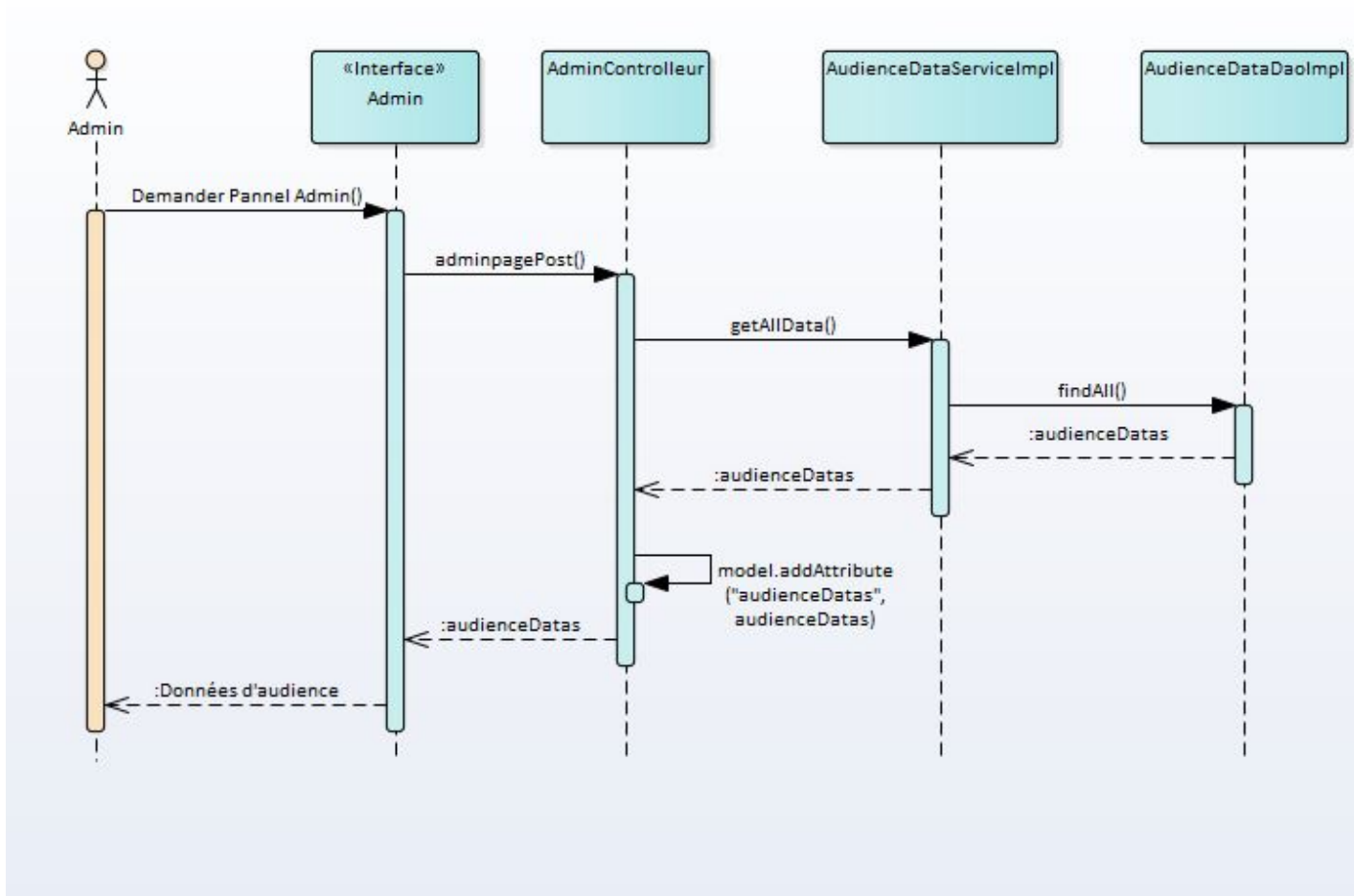


FIGURE 3.9 – Diagramme de séquences : Gérer les données d'audience

L'administrateur accède au panel Admin. Suite à cette demande, le contrôleur fait appel à la couche service qui récupère les données venant de la couche DAO à travers "AudienceDataServiceImpl" et "AudienceDataDaoImpl". Ces données seront affichés ensuite dans l'interface Admin.

CONCLUSION

A travers ce chapitre, nous avons présenté notre conception de l'application. Nous avons fourni, dans un premier lieu, une conception globale de l'organisation de notre système. Par la suite, nous avons présenté la conception détaillée de l'application à travers les diagrammes UML. Cette phase est nécessaire pour faciliter l'étape de la réalisation que nous allons la présenter dans le chapitre suivant.

Chapitre 4

Réalisation

INTRODUCTION

Après avoir achevé l'étape de la conception, en tenant compte des besoins fixés et des choix conceptuels effectués, nous consacrons ce chapitre à la description du travail réalisé. Nous commençons par décrire l'environnement matériel et logiciel sur lequel nous avons développé notre application. Ensuite, nous présentons l'état d'avancement du projet et nous mettrons en évidence le travail réalisé à travers quelques interfaces homme/machine. Enfin, nous finissons par un chronogramme qui décrit toutes les étapes de mise en oeuvre du travail.

4.1 Environnement de travail

Dans cette partie, nous commençons par l'étude des environnements techniques disponible pour la réalisation de notre projet, ensuite nous justifions les choix pris en matière d'environnement logiciel pour mener à terme la partie applicative.

4.1.1 Environnement de développement matériel

Le développement de l'application est réalisé sur deux ordinateurs ayant les caractéristiques suivantes :

Première station :

- Marque : Asus X550-VX
- Processeur : Intel Core i7-6700 HQ 2.6Ghz up to 3.5Ghz
- Mémoire RAM : 8 Go
- Disque dur : 1 TO

- Système d'exploitation : Windows 10
- Carte graphique : Nvidia GTX 950M

Deuxième station :

- Marque : Dell INSPIRON-7302
- Processeur : Intel Core i7 @2.50GHz
- Mémoire RAM : 6 Go
- Disque dur : 500 GO
- Système d'exploitation : Windows 10
- Carte graphique : Nvidia Geforce 920M

4.1.2 Environnement de développement logiciel

Dans cette partie, nous nous intéressons aux langages, aux bibliothèques et aux techniques de programmation utilisées tout au long de la réalisation de notre application en justifiant notre choix.

• Plateforme JAVA EE

Java Enterprise Edition [N12] est une spécification pour la technique Java d'Oracle plus particulièrement destinée aux applications d'entreprise. Ces applications sont considérées dans une approche multi-niveaux. Dans ce but, toute implémentation de cette spécification contient un ensemble d'extensions au framework Java standard an de faciliter notamment la création d'applications réparties.

• Framework Spring

Spring [N13] est un framework libre pour construire et définir l'infrastructure d'une application java. Il permet de simplifier le développement d'applications Java et est devenu un standard dans l'industrie du développement logiciel basé sur la plateforme Java, surtout dans le développement Java EE.

Spring est un conteneur léger qui facilite le développement avec des POJO (Plain Old Java Object), c'est-à-dire avec des classes Java qui n'ont pas besoin de s'exécuter dans un conteneur spécifique ou dans un serveur d'applications. Il se compose d'un noyau (core) et de plusieurs modules qui ajoutent des fonctionnalités.

• Hibernate pour l'implémentation de la couche d'accès au données :

Hibernate [N14] est un projet professionnel open-source. Ce type de technologie est appelé Framework de mapping objet-relationnel ou persistance objet des données. En effet, la couche

applicative voit les données comme des classes persistantes dont le contenu reste en mémoire même après la fin d'exécution du programme, d'où la persistance objet des données. L'intérêt de ce Framework est de pouvoir changer de base de données en n'ayant besoin de ne modifier que la couche d'accès. Nous avons utilisé Hibernate notamment grâce à ses bonnes performances et son ouverture à de nombreuses bases de données.

- **Zkoss pour la construction des interfaces et des graphes riches :**

ZK [N15] est un framework d'application Web Ajax open-source, écrit en Java, qui permet la création d'interfaces utilisateur graphiques et des graph pour des applications Web avec peu de connaissances en programmation. Pour l'utilisateur du framework, tout se passe comme si l'interface graphique était gérée entièrement côté serveur. Il est possible d'ajouter, enlever, modifier des composants graphiques. Le moteur de ZK se chargeant de répercuter automatiquement les modifications dans le navigateur du client. Le framework ZK permet : de gagner du temps par rapport aux framework WEB classiques, de coder de manière simple, d'avoir une application web full Java, de tracer plus rapidement et facilement les graphes ce qui la partie la plus importante dans la partie IHM de notre application.

- **TensorFlow**

TensorFlow [N16] est un outil open source d'apprentissage automatique développé par Google. Le code source a été ouvert le 9 novembre 2015 par Google et publié sous licence Apache. Il est basé sur l'infrastructure DistBelief, initiée par Google en 2011, et est doté d'une interface Python. TensorFlow est l'un des outils les plus utilisés en IA dans le domaine de l'apprentissage machine.

- **Keras**

Keras [N17] est une API de réseaux neuronaux de haut niveau, écrite en Python et capable de fonctionner sur TensorFlow, CNTK ou Theano. Il a été développé dans le but de permettre une expérimentation rapide. Pouvoir passer de l'idée au résultat avec le moins de retard possible est la clé pour faire de bonnes recherches.

4.2 Deep learning

Pour prédire les taux d'audiences on a utilisé le Deep Learning

4.2.1 Définition

Deep Learning [N18] est un sous-domaine du Machine Learning qui s'intéresse aux algorithmes inspirés par la structure et la fonction du cerveau, appelés réseaux de neurones artificiels. Les techniques d'apprentissage profond constituent une classe d'algorithmes d'apprentissage automatique qui utilisent différentes couches d'unité de traitement non linéaire pour l'extraction et la transformation des caractéristiques, chaque couche prend en entrée la sortie de la précédente, les algorithmes peuvent être supervisés ou non supervisés, et leurs applications comprennent la reconnaissance de modèles et la classification statistique ; fonctionnent avec un apprentissage à plusieurs niveaux de détail ou de représentation des données ; à travers les différentes couches, on passe de paramètres de bas niveau à des paramètres de plus haut niveau, où les différents niveaux correspondent à différents niveaux d'abstraction des données. Ces architectures permettent aujourd'hui de conférer du « sens » à des données en leur donnant la forme d'images, de sons ou de textes.

4.2.2 LSTM : Long short-term memory

Comment résoudre notre problème de prédiction en utilisant le Deep learning ?

Les problèmes de prédiction de séries temporelles sont un type difficile de problème de modélisation prédictive.

Contrairement à la modélisation prédictive par régression, ces séries ajoutent également la complexité d'une dépendance de séquence parmi les variables d'entrée.

Un type puissant de réseau neuronal conçu pour gérer la dépendance de séquence est appelé réseaux de neurones récurrents. Les réseaux neuronaux tels que (LSTM) [N19] sont capables de modéliser de façon quasi-transparente des problèmes avec plusieurs variables d'entrée. C'est un grand avantage dans la prévision des séries temporelles, où les méthodes linéaires classiques peuvent être difficiles à adapter aux problèmes de prévision multivariés ou à entrées multiples.

4.2.3 Implémentation du modèle

On a développé un modèle LSTM pour la prévision de séries temporelles multivariées dans la bibliothèque Deep Learning de Keras.

4.2.3.1 Représentation des données

Étant donné date, nom d'émission, heure et chaîne, la tâche consiste à prédire le ratio de taux d'audience de TV. Les données vont de septembre 2017 à présent. Ces données sont enregistrées dans un fichier CSV.

```

1 "01-09-2017","Koh-Lanta","TF1","Unknown","27.1%";
2 "01-09-2017","Les petits meurtres d'Agatha Christie","France 2","21h00","20.6%";
3 "01-09-2017","NCIS : enquêtes spéciales","M6","Unknown","12.4%";
4 "01-09-2017","SLC Salut les copains","France 3","20h55","7.5%";
5 "01-09-2017","Les routes de l'impossible","France 5","20h50","3.3%";
6 "01-09-2017","Enquête d'action","W9","21h00","2.7%";
7 "01-09-2017","Mentalist","TMC","21h00","2.6%";
8 "01-09-2017","Perpétuité pour deux","ARTE","20h55","2.3%";
9 "01-09-2017","Les éternels du rire","C8","21h00","2.0%";
10 "01-09-2017","1944 : Le Havre sous les bombes alliées","RMC","20h50","1.9%";
11 "01-09-2017","24 heures aux urgences","TFX","21h00","1.8%";
12 "01-09-2017","Soupçon de magie","6ter","21h00","1.5%";
13 "01-09-2017","Diane, femme flic","NRJ 12","20h55","1.3%";
14 "01-09-2017","Dinotasia","France 4","20h55","1.1%";

```

FIGURE 4.1 – Exemple de données d'apprentissage

Plusieurs algorithmes d'apprentissage ne peuvent pas fonctionner directement sur les données telles qu'elle est. Ils ont besoin que toutes les variables d'entrée ainsi que les variables de sortie soient numériques, d'où la nécessité d'encoder les données.

| | var1(t-1) | var2(t-1) | var3(t-1) | var4(t-1) | var5(t-1) | var1(t) |
|----|-----------|-----------|-----------|-----------|-----------|----------|
| 1 | 0.018382 | 0.338209 | 0.789474 | 1.000000 | 0.729730 | 0.018382 |
| 2 | 0.018382 | 0.618490 | 0.263158 | 0.545455 | 0.554054 | 0.018382 |
| 3 | 0.018382 | 0.710357 | 0.578947 | 1.000000 | 0.332432 | 0.018382 |
| 4 | 0.018382 | 0.820363 | 0.315789 | 0.454545 | 0.200000 | 0.018382 |
| 5 | 0.018382 | 0.624927 | 0.421053 | 0.363636 | 0.086486 | 0.018382 |
| 6 | 0.018382 | 0.214745 | 1.000000 | 0.545455 | 0.070270 | 0.018382 |
| 7 | 0.018382 | 0.676419 | 0.947368 | 0.545455 | 0.067568 | 0.018382 |
| 8 | 0.018382 | 0.762434 | 0.052632 | 0.454545 | 0.059459 | 0.018382 |
| 9 | 0.018382 | 0.639555 | 0.105263 | 0.545455 | 0.051351 | 0.018382 |
| 10 | 0.018382 | 0.004681 | 0.736842 | 0.363636 | 0.048649 | 0.018382 |
| 11 | 0.018382 | 0.006437 | 0.894737 | 0.545455 | 0.045946 | 0.018382 |
| 12 | 0.018382 | 0.857812 | 0.000000 | 0.545455 | 0.037838 | 0.018382 |
| 13 | 0.018382 | 0.184318 | 0.631579 | 0.454545 | 0.032432 | 0.018382 |
| 14 | 0.018382 | 0.184903 | 0.368421 | 0.454545 | 0.027027 | 0.018382 |

FIGURE 4.2 – Exemple de données d'apprentissage encodées

4.2.3.2 Entraînement du modèle

On a défini un modèle LSTM avec 50 neurones dans la couche cachée et un neurone pour les variables d'entrée ainsi qu'un pour les variables de sortie.

- **Fonction de perte :** En Deep learning, la notion principale est celle de perte d'information (loss en anglais) due à l'approximation de la réalité. Elle détermine à quel point notre modélisation du phénomène, qui est une approximation de la réalité, perd de l'information par rapport à la réalité observée à travers les données d'exemple. Dans notre modèle on a utilisé la fonction de perte Mean Absolute Error (MAE) [N19]

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |x_i - x|$$

• **Algorithme d'optimisation** : Les fonctions de perte sont des exemples illustratifs dans un algorithme de machine learning. Une grande partie des algorithmes s'appuient donc sur des méthodes d'optimisation numérique, c'est-à-dire des méthodes qui vont rechercher un maximum ou un minimum d'une fonction déterminée de manière exacte ou approximée. Notre fonction d'optimisation est la version efficace d'Adam d'une descente de gradient stochastique.

• **Époque et taille de lot** : Dans les réseaux de neurones en général, une époque est un passage de lot des données d'entraînement (70% des données) . Il peut prendre des centaines ou même parfois des milliers d'époques pour que l'algorithme d'apprentissage converge. Notre modèle est adapté pour 100 époques de formation avec une taille de lot de 72.

4.2.3.3 Résultat d'apprentissage

Après l'apprentissage on a obtenu de bons résultats avec un loss inférieur à 0.02 pour l'entraînement ainsi que pour la validation.

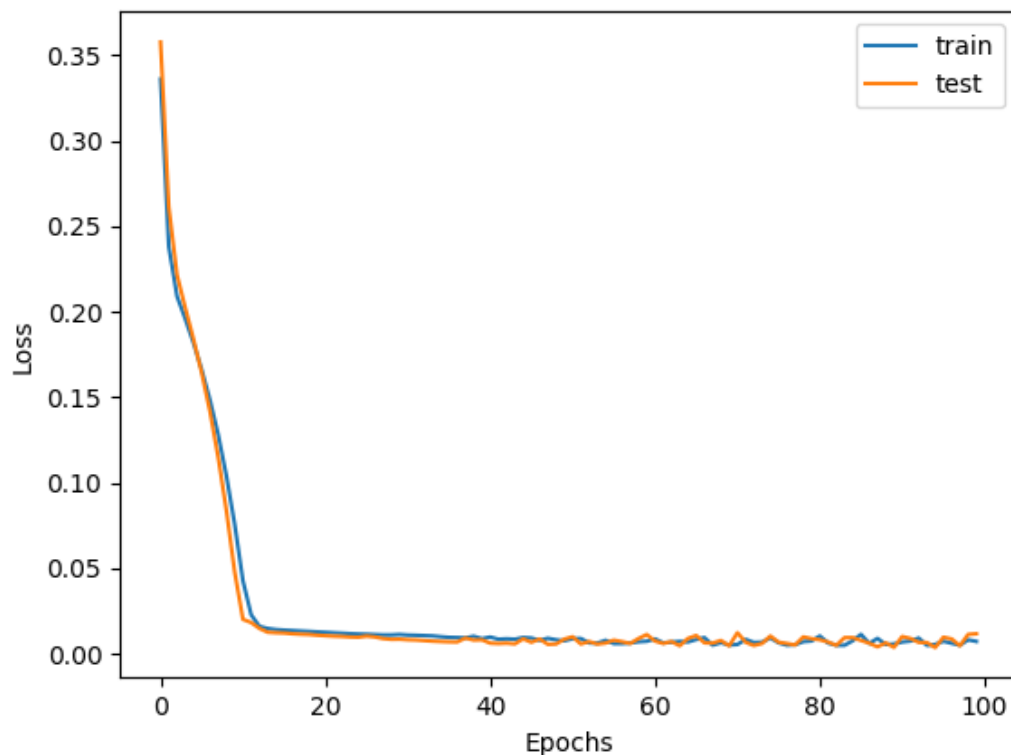


FIGURE 4.3 – Courbe de perte (Loss)

4.3 Interfaces Homme-Machine

Dans cette section, nous présentons quelques interfaces homme/machine que nous jugeons les plus significatives.

4.3.1 Interface d'accueil

La figure 4.4 représente l'interface d'accueil. En haut de cette interface est représenté le menu qui permet à l'utilisateur de s'authentifier s'il a déjà un compte ou bien de s'inscrire si non. En outre, cette interface met en valeur les fonctionnalités de l'application dans un langage clair et simple.

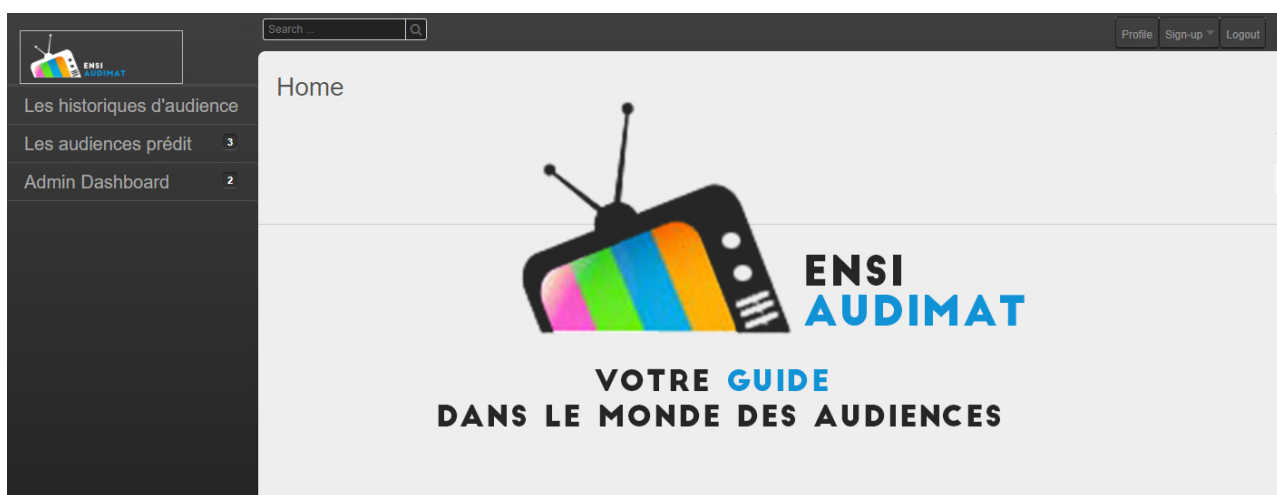
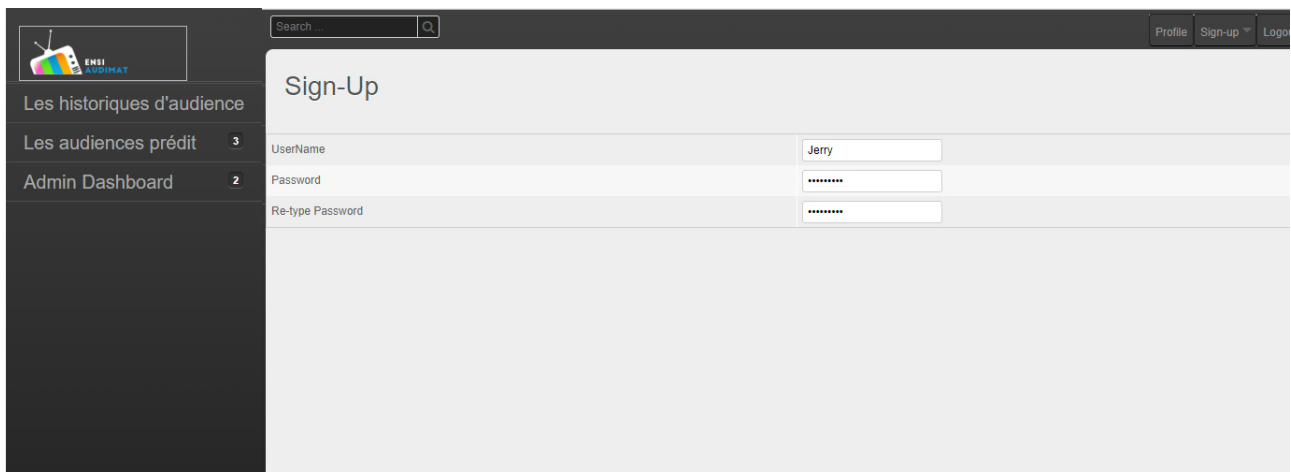


FIGURE 4.4 – Page d'accueil

4.3.2 Interface d'inscription

La figure 4.5 représente l'interface d'inscription. Pour s'inscrire, l'utilisateur doit saisir son nom et son mot de passe (celui-ci est tapé à deux reprises dans le but de le confirmer). Il est à noter que l'utilisateur est informé sur la convivialité du mot de passe

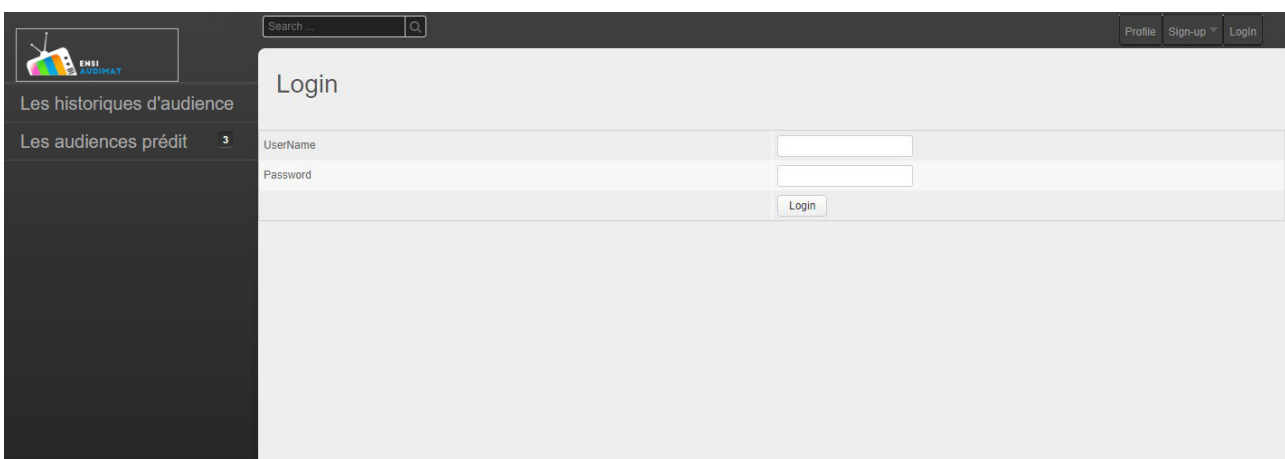


The screenshot shows a web application interface for signing up. On the left is a dark sidebar with the ENSI logo and menu items: 'Les historiques d'audience', 'Les audiences prédit 3', and 'Admin Dashboard 2'. The top header is dark with a search bar, a magnifying glass icon, and links for 'Profile', 'Sign-up', and 'Login'. The main content area is titled 'Sign-Up' and contains three input fields: 'UserName' with the value 'Jerry', 'Password' with masked characters '*****', and 'Re-type Password' also with masked characters '*****'.

FIGURE 4.5 – Interface d’inscription

4.3.3 Interface d’authentification

La figure 4.6 représente l’interface d’authentification. Pour s’authentifier, l’utilisateur inscrit ou l’admin doit saisir son mot de passe et son nom. Si les informations saisies sont erronées un message en rouge, indiquant qu’il faut vérifier ses informations, est affiché.



The screenshot shows the 'Login' interface of the same web application. The layout is consistent with Figure 4.5, including the sidebar and top header. The main content area is titled 'Login' and features two input fields: 'UserName' and 'Password'. Below these fields is a 'Login' button. The interface is clean and uses a light gray color scheme for the main content area.

FIGURE 4.6 – Interface d’authentification

4.3.4 Interface administrateur

Les figure 4.7 et 4.8 montrent l'interface dédiée à l'administrateur. Elles permettent d'accéder aux différentes fonctionnalités de contrôle des comptes gérés par l'administrateur. C'est lui aussi qui gère les données d'audience et crée les CronJob afin d'effectuer certaines tâches que notre plateforme aura besoin.

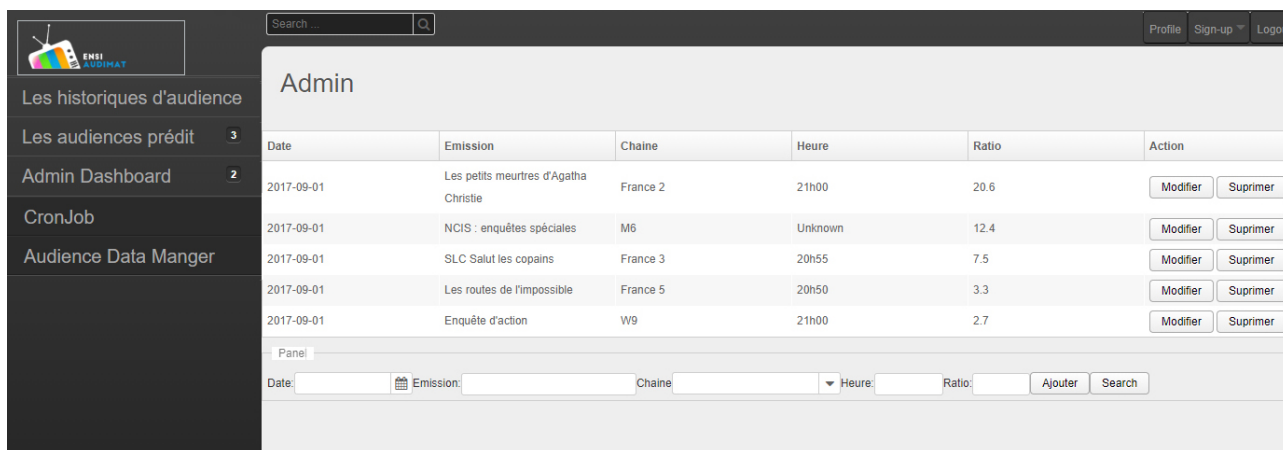


FIGURE 4.7 – Interface admin

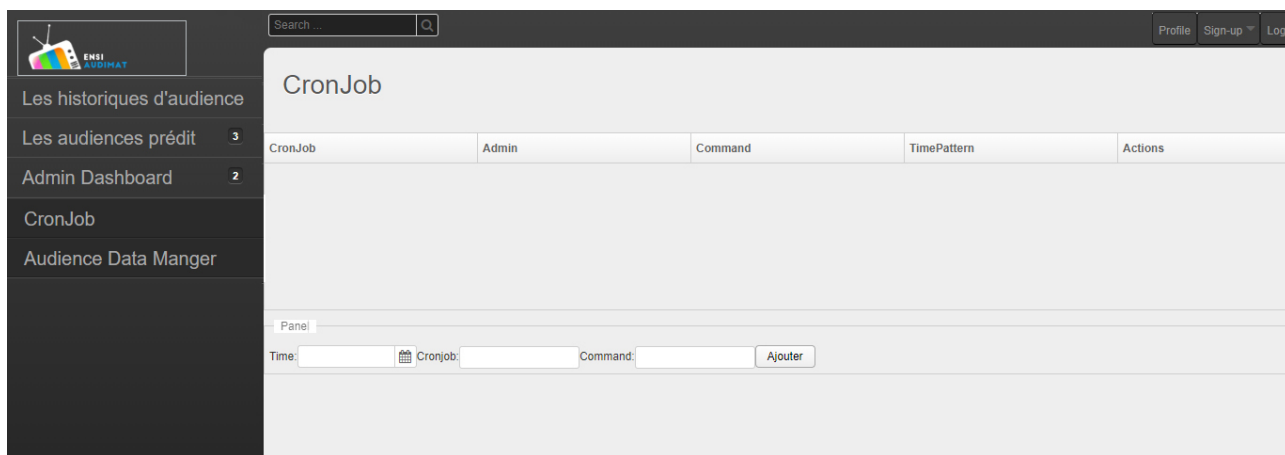


FIGURE 4.8 – Interface Cron Job

4.3.5 Interfaces de consultation d'audience

Les interfaces de consultation d'audience (Historique et prédit) présente la partie importante dans notre application. Ces interfaces sont illustrées dans les figures suivantes.

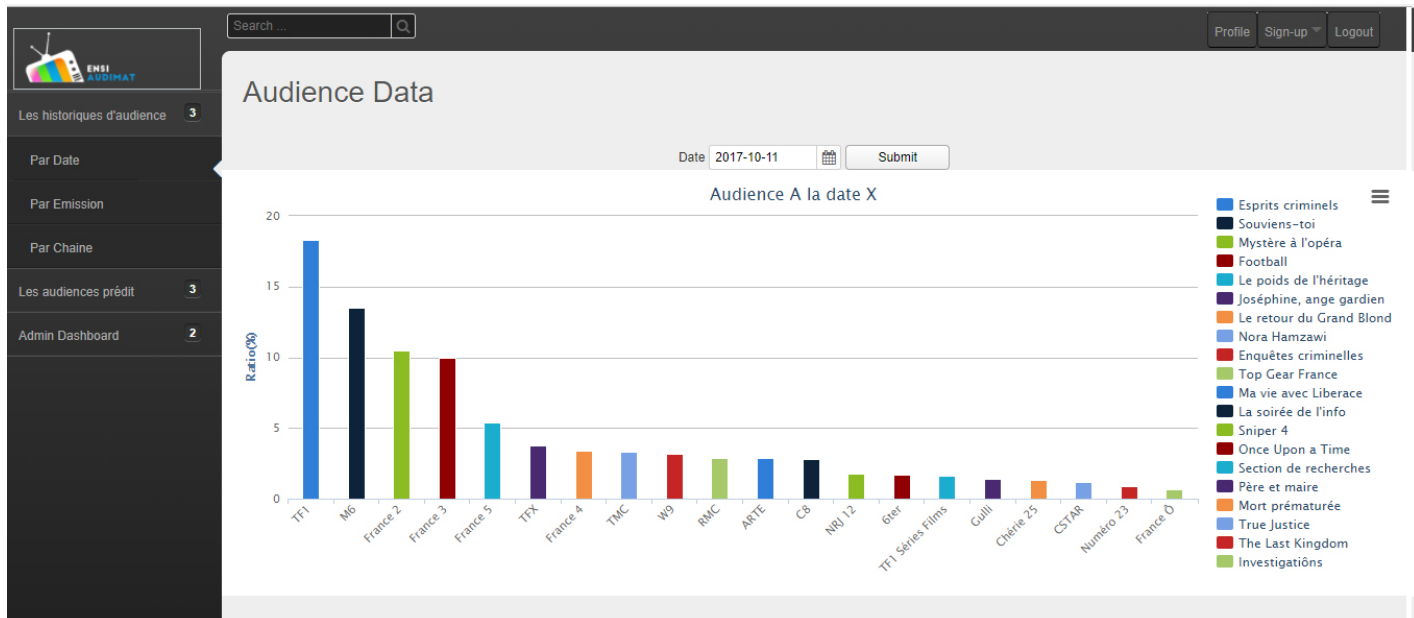


FIGURE 4.9 – Interface de consultation de historique d'audience filtré par date

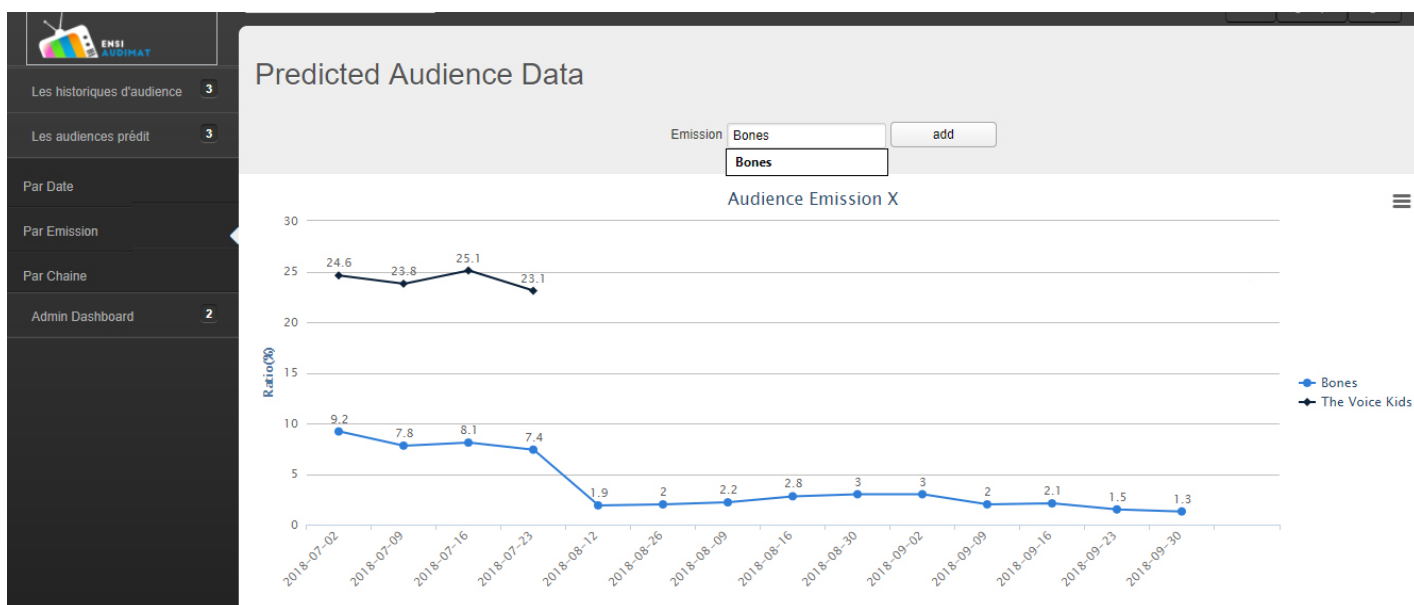


FIGURE 4.10 – Interface de consultation d'audience prédit filtré par emission.

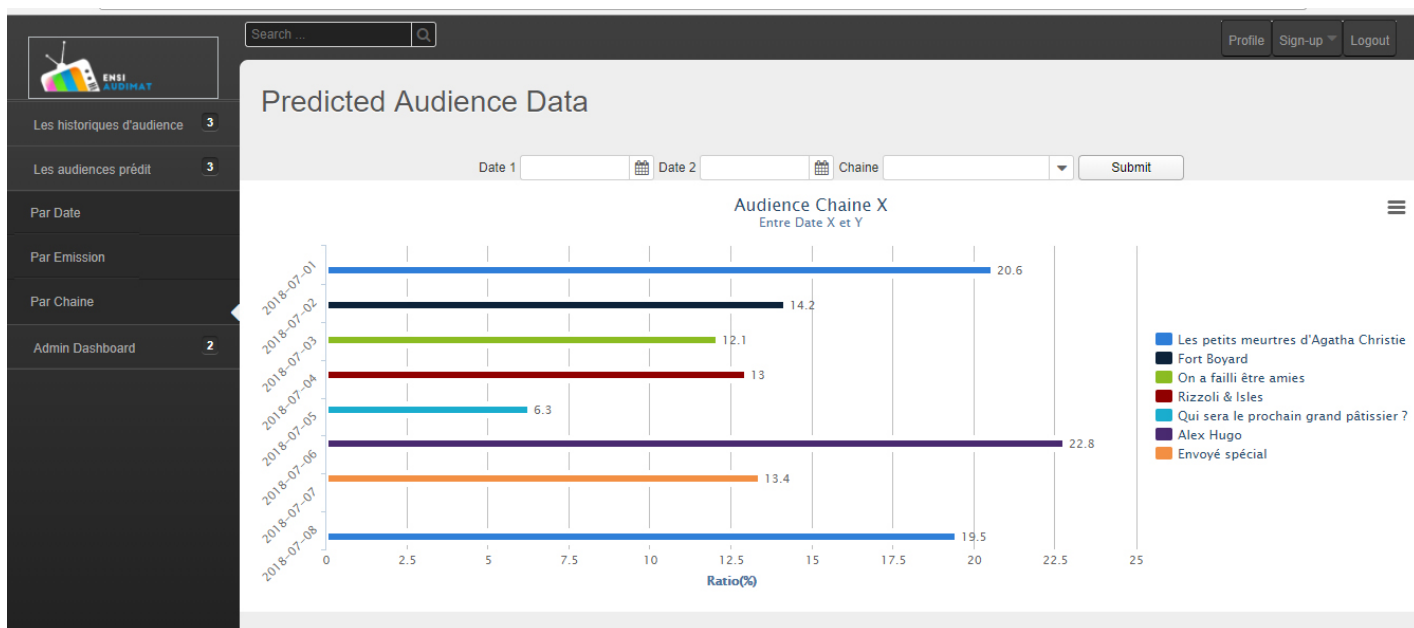


FIGURE 4.11 – Interface de consultation d'audience prédit filtré par chaine.

4.4 Chronogramme de travail

Ce travail a été réalisé sur une période de trois mois. Nous avons tout d'abord commencé par une étude approfondie de l'existant pour pouvoir comprendre les besoins et commencer le travail. Ensuite, après avoir dégagé et compris ce qui est demandé, nous avons pu dégager les spécifications fonctionnelles et non fonctionnelles. Puis nous avons entamé la phase de conception et implémenté les modules demandés après avoir passé par une phase de documentation sur les techniques à utiliser. Dans un souci d'organisation et afin de mener à bien le projet, nous avons établi un chronogramme détaillé des différentes tâches qui constituaient le projet. Il a été structuré comme le décrit la figure suivante.

| Mois | Février | | | | Mars | | | | Avril | | | | Mai | |
|---|---------|---|---|---|------|---|---|---|-------|---|---|---|-----|--|
| Semaine | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | |
| Spécification et réalisation du cahier de charge | | | | | | | | | | | | | | |
| Documentation et familiarisation avec les outils de travail | | | | | | | | | | | | | | |
| Conception | | | | | | | | | | | | | | |
| Implémentation | | | | | | | | | | | | | | |
| Rédaction du rapport | | | | | | | | | | | | | | |

FIGURE 4.12 – Chronogramme de travail

CONCLUSION

Tout au long de ce dernier chapitre, nous avons exposé la réalisation de notre travail. Dans un premier lieu, nous avons explicité l’environnement du travail, ensuite, les différentes entités de notre application en présentant des captures d’écran témoignant des différentes tâches.

CONCLUSION GÉNÉRALE

Durant ce projet, nous avons été amenés à implémenter une application dans le cadre du projet de conception et de développement. Nous sommes arrivé à mettre en oeuvre une application qui permet de prévoir les taux d'audiences des émissions et des chaînes télévisées situées en France auquel nous avons réussi à l'accomplir en adaptant un modèle implémenté et basé sur le concept « Deep Learning ».

Dans le présent rapport, nous avons détaillé les étapes par lesquelles nous sommes passés pour concevoir notre solution. Nous avons tout d'abord commencé par présenter le cadre général de notre travail et faire une étude sur les solutions existantes. Puis, nous avons spécifié les différentes fonctionnalités de l'application à développer. Ensuite, nous avons abordé la phase de conception qui nous a expliqué l'architecture de l'application. Finalement, l'étape de réalisation, au cours de laquelle nous avons précisé les logiciels utilisés ainsi que les interfaces du produit final.

Ce travail fut aussi d'un apport considérable. En effet, il a été une occasion d'acquérir de nouvelles notions concernant la technologie JEE et le Framework Spring. En outre, ce projet nous a permis également d'enrichir nos connaissances déjà acquises à l'ENSI, évidemment au niveau de la programmation JAVA aussi bien qu'au niveau des langages de définition et de manipulation des données.

L'application réalisée au cours de ce projet présente tout de même certaines insuffisances en raison des problèmes et obstacles confrontés au niveau développement. En effet, nous avons eu des problèmes concernant l'utilisation de la bibliothèque Zkoss, ce qui nous a causé une perte de temps au début de la phase d'implémentation.

Nétographie

[N1] <https://www.cairn.info/revue-hermes-la-revue-2003-3-page-35.html>

[N2] <https://www.futura-sciences.com/tech/definitions/technologie-mediometrie-16884/>

[N3] <http://radiopubafrika.unblog.fr/2016/04/21/avec-acacia-web>

[N4] [http://www.mediometrie.fr/television/solutions/audience-tv-au-senegal.php?](http://www.mediometrie.fr/television/solutions/audience-tv-au-senegal.php?id=145)

[id=145](http://www.mediometrie.fr/television/solutions/audience-tv-au-senegal.php?id=145)

[N5] <https://fr.wikipedia.org/wiki/Audimat.tn>

[N6] <https://www.supinfo.com/articles/single/574-architecture-2-tiers-vs->

[N7] <http://www.synergieintegrale-ci.com/3297-2/>

[N8] <https://dzone.com/articles/what-is-n-tier-architecture>

[N9] <http://cedric.babault.free.fr/rapport/node4.html>

[N10] <https://fr.wikipedia.org/wiki/Modèle-vue-contrôleur>

[N11] http://dchaffiol.free.fr/info/architecture/art_archi_ataf.html

[N12] https://fr.wikipedia.org/wiki/Java_EE

[N13] <https://docs.spring.io/spring/docs/current/spring-framework-reference/>

[web.html](https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html)

[N14] <https://www.jmdoudoux.fr/java/dej/chap-hibernate.htm>

[N15] [https://en.wikipedia.org/wiki/ZK_\(framework\)](https://en.wikipedia.org/wiki/ZK_(framework))

[N16] <https://fr.wikipedia.org/wiki/TensorFlow>

[N17] <https://keras.io/>

[N18] <https://machinelearningmastery.com/what-is-deep-learning/>

[N19] <https://machinelearningmastery.com/time-series-prediction-lstm>

الملخص:

تم تنفيذ هذا العمل في إطار مشروع التصميم والبرمجة لطلاب السنة الثانية هندسة بالمدرسة الوطنية لعلوم الإعلامية، ويتمثل في تصميم وتنفيذ منصة تحليل وتوقع نسب المشاهدة لبعض البرامج والقنوات التلفزيونية. ففي الوقت الحاضر يحتل التلفزيون بلا شك مكانا هاما في مجتمعنا وتعتبر نسب المشاهدة اهم عامل للمستثمرين وشركات سبر الآراء. وفي هذا الإطار يندرج هذا العمل.

هذا التقرير يفصل مختلف الخطوات التي قمنا بها.

RÉSUMÉ

Notre travail est proposé dans le cadre du projet de conception et de développement pour les élèves ingénieurs de l'Ecole Nationale des Sciences de l'Informatique (ENSI).

Notre tâche avait pour objectif d'implémenter une plateforme illustrant des prédictions faites sur des mesures d'audience TV. En s'adaptant sur un modèle d'intelligence pour la prévision des séries temporelles, nous avons pu représenter les taux d'audiences des différentes émissions et chaînes télévisées situées en France.

ABSTRACT

This project is carried out as a second year design and development project at the National School of Computer Science.

Our task was to implement a platform which consists in creating a web application that is set to analyze audience ratings of broadcasts and television channels in France and provide mainly reliable forecasting of these rates. This report details the various theoretical studies and applied steps we have taken to design and implement this application.