République Tunisienne
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Monastir
Institut Supérieur d'Informatique de Mahdia

MEMOIRE

N° d'ordre : LCS-GLSI-3-2026-045

MEMOIRE

Présenté à

Institut Supérieur d'Informatique de Mahdia

Effectué à

VERMEG Tunis

En vue de l'obtention du diplôme de
Licence en Sciences D'Inforamtique
Licence en Informatiques De Gestion

Elaboré par :
Moktada Zaidi, Hayouni Mohamed

SmartFinance Advisor

Soutenu le : 15 juin 2026
devant le jury composé de :

Président	: Hamdi Aloulou
Rapporteur	: Abdehafidh Marwa
Encadrant Académique	: Imen Toumia
Encadrant Professionnel	: Mariem Zhioua

Année universitaire 2025-2026

Dédicaces



À ***Allah****, le Tout-Puissant,*
qui m'a accordé la force, la volonté et la patience
d'accomplir ce travail. Toute louange Lui appartient.

À mon ***binôme*** *Mohamed HAYOUNI,*
pour les nuits blanches partagées, les doutes surmontés ensemble
et cette complicité qui a rendu ce projet possible.

À mon ***père*** *Faicel ZAIDI,*
mon boss et mon éternel modèle, celui qui m'a appris que rien ne
s'obtient sans effort et sans caractère. Tu as toujours été là, solide
comme un roc, pour me guider et me soutenir. Ce travail est le tien
autant que le mien, merci baba.

À ma ***mère*** *Sameh YAAKOUBI,*
l'âme de mon âme, celle sans qui je ne serais rien. Ton amour infini et
tes prières douces ont été mon plus grand soutien tout au long de ce
chemin. Ce travail, je te le dédie avec tout l'amour que j'ai dans le
cœur, merci omi.

À mon ***petit frère*** *Moudhaffer ZAIDI,*
ma joie au quotidien. Que ce travail t'inspire
à poursuivre tes rêves avec ambition et confiance.

À mes ***enseignants****,*
pour le savoir transmis avec générosité et exigence.
C'est grâce à vous que je deviens l'ingénieur que je veux être.

À mes ***Amis*** *d'enfance,*
Yacine, Anas, Rayen, Louay, Ayoub et Adem
pour les encouragements, les fous rires et les souvenirs partagés.
Merci d'avoir été là, simplement.

*À mes **Amis** de Mahdia,*

Achref, Yazid, Habib, Anas, Wassim, Iheb, Saïd et Ayoub
Ceux avec qui j'ai ri aux éclats,galéré et grandi. Vous avez rendu ce long chemin tellement plus leger et plus beau, juste par votre présence et votre bonne humeur contagieuse.

*À ma **famille scouts**,*

pour les valeurs de service, de solidarité et de dépassement de soi qui guident mes pas bien au-delà du terrain.

*À mes **Collègues** De VERMEG,*

Zeineb, Mariem, Iheb et Nour
Votre générosité, votre patience et votre partage ont fait de ce stage bien plus qu'une simple étape professionnelle, une vraie école de vie.

À vous tous, je dédie ce travail avec toute ma sincérité et ma gratitude.

Moktada ZAIDI

Dédicaces



*À **Allah**, le Tout-Puissant,*
qui m'a accordé la force, la volonté et la patience
d'accomplir ce travail. Toute louange Lui appartient.

*À **mon binôme de PFE, Moktada Zaidi**,*
pour cette belle aventure partagée, ta patience et ton sérieux,
pour les nuits blanches, les doutes surmontés et les fous rires.
Ce projet est autant le tien que le mien.

*À **ma mère, Najwa Mdaïni**,*
source d'amour, de tendresse et de sacrifices silencieux,
dont les prières et la foi indéfectible ont été ma plus grande force.
Tout ce que je suis, je te le dois.

*À **mon père, Mohamed Habib Hayouni**,*
mon modèle, mon pilier,
qui m'a transmis la rigueur, l'honnêteté et le goût de l'effort.
Merci pour ton soutien indéfectible et ta confiance en moi.

*À **ma sœur, Nada Hayouni**,*
pour ta présence, ta bonne humeur et tous ces moments partagés
qui illuminent mes journées. Tu es ma complicité au quotidien.

*À **mes amis de toujours**,*
Hamed, Aziz, Yazid, Amen, Mahdi et Mahmoud,
vous êtes ma famille de cœur.
Merci pour votre loyauté, votre soutien
et tous ces souvenirs qui m'accompagneront toute la vie.

*À **ceux que j'ai eu la chance de rencontrer à Mahdia**,*
Achref, Hédi, Amin, Adem, Habib, Anas,
Wassim, Iheb, Chiheb, Reyen, Ayoub et Saïd,
pour les bons moments, les éclats de rire
et cette belle période partagée ensemble.
Chacun de vous a marqué ce chapitre de ma vie.

*À **mes collègues de Vermeg**,*
Mariem, Zeineb et Iheb,
pour l'ambiance chaleureuse, le travail partagé
et tous ces moments qui ont rendu ce stage inoubliable.
Merci d'avoir fait de Vermeg une belle expérience.

« Le succès n'est pas final, l'échec n'est pas fatal : c'est le courage de continuer qui compte. »

Mohamed HAYOUNI

Remerciements

Avant tout, nous rendons grâce à Dieu le Tout-Puissant de nous avoir accordé la santé, la volonté et le courage nécessaires pour mener à bien ce travail.

Nous tenons à exprimer notre sincère reconnaissance à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce projet de fin d'études.

Nos premiers remerciements vont à notre encadrante académique, **Madame Imen Toumia**, pour sa disponibilité, sa bienveillance et la qualité de son accompagnement tout au long de ce projet. Ses conseils éclairés, ses orientations pertinentes et son soutien constant nous ont permis d'avancer avec méthode et confiance.

Nous adressons également nos remerciements les plus sincères à notre encadrante professionnelle, **Madame Meriem Zhioua**, pour nous avoir accueillis au sein de la société et accompagnés avec patience et professionnalisme. Son expérience, sa générosité et ses conseils pratiques ont considérablement enrichi notre apprentissage et fait de cette expérience une étape formatrice à plus d'un titre.

Nous exprimons par ailleurs notre gratitude aux membres du jury, qui nous ont fait l'honneur d'évaluer ce travail. Leur expertise et leur regard critique représentent pour nous une opportunité précieuse de recul et de progression.

Enfin, nous remercions chaleureusement toutes celles et ceux qui ont croisé notre chemin au fil de cette aventure et qui, par un mot ou un geste, ont contribué à faire de cette expérience un souvenir marquant.

Mohamed HAYOUNI
Moktada ZAIDI

INTRODUCTION GENERALE **16**

1 CADRE GÉNÉRAL DU PROJET **17**
1.1 Introduction . . . 17
1.2 Présentation de l'Organisme d'Accueil : VERMEG . . . 17
1.2.1 VERMEG : accompagner les institutions financières vers une réussite durable . . . 17
1.2.1.1 Historique et fondation . . . 18
1.2.1.2 Activités et domaines d'expertise . . . 18
1.2.1.3 Présence mondiale . . . 18
1.3 Contexte général . . . 18
1.3.1 Problématique . . . 18
1.3.2 Étude de l'existant . . . 18
1.3.3 Critique de l'existant . . . 19
1.3.4 Solution proposée . . . 19
1.4 Méthodologie de travail . . . 20
1.4.1 Étude comparative . . . 20
1.4.1.1 Fonctionnement de la méthode . . . 21
1.4.1.2 Les rôles Scrum . . . 21
1.5 Langage et architecture de modélisation . . . 22
1.5.1 Langage de modélisation unifié (UML) . . . 22
1.5.2 Architecture en couches . . . 23
1.6 Conclusion . . . 23

2 ÉTUDE PRÉALABLE **24**
2.1 Introduction . . . 24
2.2 Identification des acteurs . . . 24
2.3 Spécification des besoins . . . 24
2.3.1 Besoins fonctionnels . . . 24
2.3.2 Besoins non fonctionnels . . . 25
2.3.3 Diagramme de cas d'utilisation global . . . 25
2.3.4 Diagramme de classes global . . . 26
2.4 Organisation du projet avec Scrum . . . 27
2.4.1 Équipe et rôles . . . 27
2.4.2 Backlog du produit . . . 28
2.4.3 Organisation des sprints . . . 30
2.4.4 Planification . . . 31
2.5 Méthodologie CRISP-DM . . . 31
2.5.1 Présentation de la méthodologie CRISP-DM . . . 31
2.5.2 Justification du choix méthodologique . . . 32

2.6 Environnement de travail . . . 32
2.6.1 Environnement matériel . . . 32
2.6.2 Environnement logiciel . . . 33
2.7 Conclusion . . . 36

3 ETL ET EXTRACTION DES DONNÉES 37
3.1 Introduction . . . 37
3.2 Backlog du Sprint 1 . . . 37
3.3 Spécifications fonctionnelles . . . 38
3.3.1 Diagramme de cas d'utilisation du Sprint 1 . . . 38
3.3.2 Description détaillée des cas d'utilisation . . . 38
3.3.2.1 Description Détaillée du cas d'utilisation "Ingestion des données" 38
3.3.2.2 Description Détaillée du cas d'utilisation "Consultation des tableaux de bord" . . . 39
3.3.2.3 Description Détaillée du cas d'utilisation "Visualisation des graphiques" . . . 40
3.3.2.4 Description Détaillée du cas d'utilisation "Analyse des performances" . . . 40
3.4 Conception . . . 41
3.4.1 Diagrammes de séquence . . . 41
3.4.1.1 Diagramme de séquence "Ingestion De Données" . . . 41
3.5 Flux de données détaillé : l'architecture Medallion . . . 42
3.6 Compréhension métier . . . 42
3.6.1 Objectifs métier . . . 42
3.6.2 Évaluation de la situation . . . 43
3.6.3 Objectifs de data mining . . . 43
3.7 Compréhension des données . . . 43
3.7.1 Collecte des données . . . 43
3.7.2 Structure des données OHLCV . . . 45
3.7.3 Qualité des données et défis . . . 45
3.7.4 Synthèse de l'exploration . . . 45
3.8 Préparation des données . . . 45
3.8.1 Intégration et normalisation . . . 45
3.8.2 Nettoyage et transformation . . . 46
3.8.3 Validations métier (contraintes de données) . . . 46
3.8.4 Chargement (Couche Silver) . . . 46
3.9 Modélisation des données (*Feature Engineering*) . . . 46
3.9.1 Standardisation et nettoyage : la Couche Silver . . . 46
3.9.2 Modélisation métier et *Feature Engineering* : la Couche Gold . . . 47
3.9.3 Chargement et optimisation (base de données) . . . 47
3.10 Évaluation . . . 48
3.10.1 Profilage global et volumétrie . . . 48
3.10.2 Audit de la qualité des données . . . 48
3.10.3 Évaluation des KPIs et distributions statistiques . . . 48

3.10.3.1 Distribution des rendements et volatilité . . . 48
3.10.3.2 Analyse des corrélations . . . 49
3.10.4 Conclusion de l'évaluation . . . 50
3.11 Mise en œuvre et validation du tableau de bord . . . 50
3.11.1 Construction et validation de l'API analytique . . . 50
3.11.2 Développement de l'interface de visualisation (Frontend) . . . 51
3.11.2.1 Vue d'ensemble (Overview) . . . 51
3.11.2.2 Analyse technique (Price Chart) . . . 51
3.11.2.3 Rendements mensuels (Monthly Returns) . . . 52
3.11.2.4 Analyse du drawdown . . . 53
3.11.2.5 Volatilité (Volatility) . . . 54
3.12 Conclusion . . . 55

4 SPRINT 2 : INSCRIPTION, AUTHENTIFICATION ET GESTION DES COMPTES 56
4.1 Introduction . . . 56
4.2 Backlog du Sprint 2 . . . 56
4.3 Spécifications fonctionnelles . . . 56
4.3.1 Diagramme de cas d'utilisation du Sprint 2 . . . 57
4.3.2 Description détaillée des cas d'utilisation . . . 58
4.3.2.1 Description Détaillée du cas d'utilisation "Valider les demandes d'inscription" . . . 58
4.3.2.2 Description Détaillée du cas d'utilisation "Modifier le rôle d'un utilisateur" . . . 58
4.3.2.3 Description Détaillée du cas d'utilisation "Supprimer un utilisateur" . . . 59
4.3.2.4 Description Détaillée du cas d'utilisation "S'inscrire" . . . 59
4.3.2.5 Description Détaillée du cas d'utilisation "Réinitialiser le mot de passe" . . . 60
4.4 Conception . . . 60
4.4.1 Diagrammes de séquence du Sprint 2 . . . 60
4.4.1.1 Diagramme de séquence « S'authentifier » . . . 60
4.4.1.2 Diagramme de séquence « Gérer les comptes » . . . 61
4.5 Réalisation . . . 62
4.5.1 Interface d'inscription . . . 62
4.5.2 Interface d'authentification . . . 63
4.5.3 Interface de réinitialisation du mot de passe . . . 64
4.5.3.1 Étape 1 — Saisie de l'adresse e-mail . . . 64
4.5.3.2 Étape 2 — Validation du code de vérification . . . 64
4.5.3.3 Étape 3 — Définition du nouveau mot de passe . . . 65
4.5.4 Interface de gestion des demandes en attente . . . 66
4.5.5 Interface de gestion des utilisateurs actifs . . . 67
4.6 Aspects de sécurité . . . 68
4.7 Conclusion . . . 69

5 SPRINT 3 : DEVELOPPEMENT DU MODELE PREDICTIF 70
5.1 Introduction . . . 70
5.2 Backlog Du Sprint 3 . . . 70
5.3 Spécification Fonctionnelle . . . 70
5.3.1 Diagramme de Cas d'Utilisation du Sprint 3 . . . 71
5.3.2 Description Détaillée des Cas d'Utilisation . . . 71
5.3.2.1 Description Détaillé du cas d'utilisation "Générer des prédictions" 71
5.3.2.2 Description Détaillé du cas d'utilisation "Prévoir les clôtures" . 72
5.3.2.3 Description Détaillé du cas d'utilisation "Collecter les données financières" . . . 72
5.3.2.4 Description Détaillé du cas d'utilisation "Consulter les résultats" 73
5.4 Conception . . . 74
5.4.1 Diagrammes de Séquences . . . 74
5.4.1.1 Diagrammes de Séquence "Génerer Des Prédictions" . . . 74
5.5 Définition et concept du Machine Learning . . . 74
5.5.1 Intérêt du Machine Learning pour la Business Intelligence . . . 75
5.5.2 Étude comparative des modèles de prédiction . . . 75
5.5.3 Justification du choix de l'algorithme . . . 76
5.6 Mise en place du modèle . . . 77
5.6.1 Importation des bibliothèques nécessaires . . . 77
5.6.2 Préparation pour la prédiction . . . 78
5.6.3 Génération des Prédictions . . . 79
5.7 Évaluation . . . 79
5.7.1 L'Erreur Absolue Moyenne (MAE) . . . 80
5.7.2 La Racine de l'Erreur Quadratique Moyenne (RMSE) . . . 80
5.7.3 Score de précision global (Accuracy) . . . 80
5.7.4 Résultats de l'évaluation des prédictions . . . 80
5.8 Intégration du modèle dans l'interface de l'application . . . 81
5.8.1 Architecture de flux et de consommation des données . . . 81
5.8.2 Gestion des états et Expérience Utilisateur (UX) . . . 81
5.8.3 Restitution visuelle et cône d'incertitude . . . 82
5.9 Conclusion . . . 83

6 SPRINT 4 : DEVELOPPEMENT DE L'ASSISTANT INTELLIGENT LEXA 84
6.1 Introduction . . . 84
6.2 Backlog Du Sprint 4 . . . 84
6.3 Spécification Fonctionnelle . . . 85
6.3.1 Diagramme de cas d'utilisation . . . 85
6.3.2 Description détaillée des cas d'utilisation . . . 86
6.3.2.1 Ingester la loi financière . . . 86
6.3.2.2 Description Détaillée De cas d'utilisation "Utiliser le Chatbot en Mode Juridique" . . . 87

6.3.2.3 Description Détaillée De cas d'utilisation "Obtenir des explications juridiques" . . . 89
6.3.2.4 Description Détaillée De cas d'utilisation "Utiliser le Chatbot en Mode Analyste" . . . 90
6.3.2.5 Description Détaillée De cas d'utilisation "Explication de performances" . . . 90
6.3.2.6 Description Détaillée De cas d'utilisation "Obtention des rapports" 91
6.4 Conception . . . 92
6.4.1 Diagrammes de séquence du Sprint 4 . . . 92
6.4.1.1 LEXA en mode Analyste . . . 92
6.4.1.2 Module d'ingestion de lois . . . 93
6.4.1.3 LEXA en mode Juridique . . . 94
6.5 Module d'ingestion des lois financières . . . 94
6.5.1 Présentation du processus **RAG** (*Retrieval-Augmented Generation*) . . . 94
6.5.2 Architecture Médaillon pour l'ingestion des lois financières . . . 95
6.5.2.1 Couche Bronze . . . 95
6.5.2.2 Couche Silver . . . 95
6.5.2.3 Couche Gold . . . 95
6.6 L'assistant intelligent LEXA . . . 96
6.7 LEXA en mode Juridique . . . 96
6.8 LEXA en mode Analyse . . . 98
6.9 Intégration du rapport de Data Profiling dans LEXA . . . 99
6.10 Conclusion . . . 100

CONCLUSION GENERALE **101**

BIBLIOGRAPHIE **102**

Table des figures

1.1 Bâtiment VERMEG Biwa . . . 17
1.2 Logo Bloomberg Terminal . . . 19
1.3 Logo FactSet . . . 19
1.4 Workflow SmartFinance Advisor . . . 20
1.5 Les six étapes de la méthode Waterfall . . . 20
1.6 Cycle de vie de l'approche SCRUM . . . 21
1.7 L'équipe SCRUM . . . 22
1.8 Logo UML . . . 23

2.1 Diagramme de cas d'utilisation global . . . 26
2.2 Diagramme de classes global . . . 27
2.3 Découpage du projet en sprints . . . 31
2.4 Diagramme de GANTT . . . 31
2.5 Cycle de vie de la méthodologie CRISP-DM . . . 32
2.6 Logo Visual Studio Code . . . 33
2.7 Logo Angular . . . 34
2.8 Logo TypeScript . . . 34
2.9 Logo Tailwind CSS . . . 34
2.10 Logo Python . . . 34
2.11 Logo Flask . . . 35
2.12 Logo PostgreSQL . . . 35
2.13 Logo Supabase . . . 35
2.14 Logo Postman . . . 35
2.15 Logo Git . . . 35
2.16 Logo GitHub . . . 36
2.17 Logo Overleaf . . . 36
2.18 Logo Chart.js . . . 36
2.19 Logo Draw.io . . . 36

3.1 Diagramme de cas d'utilisation du Sprint 1 . . . 38
3.2 Diagramme de séquence "Ingestion Des Données" . . . 42
3.3 Extrait de code d'extraction des fichiers CSV depuis Stooq . . . 44
3.4 Extrait de code d'extraction via l'API REST Tiingo . . . 44
3.5 Distribution statistique des rendements mensuels extraite du rapport de *Data Profiling* . . . 49
3.6 Matrice de corrélation des métriques boursières et des moyennes mobiles (SMA) 49
3.7 Test Postman : récupération des KPIs financiers (SMA, Drawdown, Rendement) au format JSON . . . 50
3.8 Vue d'ensemble du tableau de bord présentant les indicateurs essentiels et la dynamique générale de l'actif . . . 51
3.9 Module d'analyse technique mettant en regard le prix de l'actif et ses moyennes mobiles pour identifier la tendance . . . 52

3.10 Visualisation des rendements mensuels permettant d'identifier les phases de gains, les replis marquants et les effets de saisonnalité . . . 53
3.11 Tableau de bord dédié au drawdown, illustrant l'ampleur des baisses et la capacité de récupération de l'actif . . . 54
3.12 Suivi de la volatilité glissante permettant de distinguer les périodes de calme relatif des épisodes de forte agitation du marché . . . 55
3.13 Visualisation de l'intensité des épisodes de forte agitation du marché . . . 55

4.1 Diagramme de cas d'utilisation du Sprint 2 . . . 57
4.2 Diagramme de séquence « S'authentifier » . . . 61
4.3 Diagramme de séquence « Gérer les comptes » . . . 62
4.4 Interface d'inscription . . . 63
4.5 Interface d'authentification . . . 63
4.6 Interface de réinitialisation — Saisie de l'adresse e-mail . . . 64
4.7 Interface de réinitialisation — Envoi du code de vérification . . . 65
4.8 Interface de réinitialisation — Saisie du code de vérification . . . 65
4.9 Interface de réinitialisation — Définition du nouveau mot de passe . . . 66
4.10 Interface des demandes en attente . . . 66
4.11 E-mail de confirmation envoyé à l'utilisateur après validation de sa demande . . . 67
4.12 Interface de gestion des utilisateurs actifs . . . 68
4.13 Schéma du hachage des mots de passe avec bcrypt . . . 69

5.1 Diagramme de cas d'utilisation du Sprint 3 . . . 71
5.2 Diagramme de séquence « Génerer des prédictions » . . . 74
5.3 Concept du Machine Learning . . . 75
5.4 Importation des bibliothèques nécessaires . . . 77
5.5 Préparation pour la prédiction – partie 1 . . . 78
5.6 Préparation pour la prédiction – partie 2 . . . 79
5.7 Génération des prédictions et des cônes d'incertitude . . . 79
5.8 Interface graphique de prédiction – vue 1 . . . 82
5.9 Interface graphique de prédiction – vue 2 . . . 83

6.1 Diagramme de cas d'utilisation du Sprint 4 . . . 86
6.2 Diagramme de séquence « Chatbot LEXA - Mode Analyste » . . . 93
6.3 Diagramme de séquence « Module d'ingestion de lois » . . . 93
6.4 Diagramme de séquence « Chatbot LEXA - Mode Juridique (RAG) » . . . 94
6.5 Interface LEXA — module d'ingestion de données . . . 96
6.6 Workflow de LEXA en mode Juridique . . . 97
6.7 Interface de LEXA en mode Juridique . . . 98
6.8 Workflow de LEXA en mode Analyste . . . 98
6.9 Interface de LEXA en mode Analyste . . . 99
6.10 Workflow du module Data Profiling dans LEXA . . . 100

Liste des tableaux

2.1 Rôles et responsabilités au sein de l'équipe Scrum . . . 28
2.2 Backlog général de SmartFinance Advisor . . . 28
2.3 Caractéristiques de l'ordinateur 1 . . . 33
2.4 Caractéristiques de l'ordinateur 2 . . . 33

3.1 Découpage fonctionnel et technique du pipeline Medallion . . . 37
3.2 Description du cas d'utilisation : Ingestion des données financières . . . 39
3.3 Description du cas d'utilisation : Consultation des tableaux de bord . . . 39
3.4 Description du cas d'utilisation : Visualisation des graphiques . . . 40
3.5 Description du cas d'utilisation : Analyse des performances financières . . . 41
3.6 Structure des données brutes en couche Bronze . . . 45

4.1 Backlog du Sprint 2 . . . 56
4.2 Description détaillée du cas d'utilisation « Valider les demandes d'inscription » . . . 58
4.3 Description détaillée du cas d'utilisation « Modifier le rôle d'un utilisateur » . . . 58
4.4 Description détaillée du cas d'utilisation « Supprimer un utilisateur » . . . 59
4.5 Description détaillée du cas d'utilisation « S'inscrire » . . . 59
4.6 Description détaillée du cas d'utilisation « Réinitialiser le mot de passe » . . . 60

5.1 Backlog du Sprint – Module de Prédiction . . . 70
5.2 Description détaillée du cas d'utilisation « Générer des prédictions » . . . 72
5.3 Description détaillée du cas d'utilisation « Prévoir les clôtures » . . . 72
5.4 Description détaillée du cas d'utilisation « Collecter les données financières » . . . 73
5.5 Description détaillée du cas d'utilisation « Consulter les résultats » . . . 73
5.6 Comparaison des principaux algorithmes d'apprentissage automatique pour la prédiction de séries temporelles . . . 76
5.7 Synthèse des performances prédictives *in-sample* du modèle Prophet par actif. . . . 81

6.1 Backlog du Sprint 4 . . . 85
6.2 Description détaillée du cas d'utilisation « Ingérer la loi financière » . . . 87
6.3 Description détaillée du cas d'utilisation « Utiliser le Chatbot en Mode Juridique » 88
6.4 Description détaillée du cas d'utilisation « Obtenir des explications juridiques » 89
6.5 Description détaillée du cas d'utilisation « Utiliser le Chatbot en Mode Analyste » 90
6.6 Description détaillée du cas d'utilisation « Explication de performances » . . . 91
6.7 Description détaillée du cas d'utilisation « Obtention des rapports » . . . 92

ETL	Extract, Transform, Load
CRISP-DM	Cross Industry Standard Process for Data Mining
RAG	Retrieval-Augmented Generation
UML	Unified Modeling Language
SCRUM	Cadre agile de gestion de projet
OHLCV	Open, High, Low, Close, Volume
OHLC	Open, High, Low, Close
SQL	Structured Query Language
JSON	JavaScript Object Notation
HTML	HyperText Markup Language
PDF	Portable Document Format
CSV	Comma-Separated Values
API	Application Programming Interface
REST	Representational State Transfer
HTTP	HyperText Transfer Protocol
JWT	JSON Web Token
URL	Uniform Resource Locator
KPI	Key Performance Indicator
SMA	Simple Moving Average
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
sMAPE	Symmetric Mean Absolute Percentage Error
ETF	Exchange-Traded Fund
ML	Machine Learning
BI	Business Intelligence
IA	Intelligence Artificielle
AI	Artificial Intelligence
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
GAM	Generalized Additive Model
ARIMA	AutoRegressive Integrated Moving Average
SARIMA	Seasonal AutoRegressive Integrated Moving Average
LLM	Large Language Model

CSS	Cascading Style Sheets
RxJS	Reactive Extensions for JavaScript
UX	User Experience
UI	User Interface
SGBD	Système de Gestion de Base de Données
RAM	Random Access Memory
SSD	Solid State Drive
AAPL	Apple Inc.
TSLA	Tesla Inc.
MSFT	Microsoft Corporation
GOOG	Alphabet Inc. (Google)
JPM	JPMorgan Chase & Co.
SPY	SPDR S&P 500 Exchange-Traded Fund

INTRODUCTION GENERALE

Les banques et les sociétés de gestion d'actifs accumulent aujourd'hui des volumes considérables de données historiques sur les marchés financiers. Pourtant, dans la pratique, ces données demeurent souvent sous-exploitées, dispersées dans des systèmes hétérogènes ou simplement mal structurées. Des outils d'analyse de référence comme Bloomberg Terminal ou FactSet existent certes, mais leur coût prohibitif et leur inadaptation au contexte tunisien en font des solutions largement inaccessibles pour la majorité des institutions financières locales.

C'est dans ce contexte que VERMEG, entreprise tunisienne spécialisée dans les logiciels dédiés au secteur financier, nous a confié ce projet de fin d'études. La mission qui nous a été assignée consistait à concevoir, de bout en bout, une plateforme d'analyse financière taillée pour les besoins réels du marché tunisien. La problématique centrale peut se formuler ainsi : comment permettre à un analyste financier de passer de la donnée brute à la décision concrète, depuis une interface unifiée, sans avoir à naviguer entre une multitude d'outils differents ?

Pour répondre à cette question, le développement a été organisé en quatre sprints successifs et complémentaires. Le premier a consisté à mettre en place un pipeline ETL fondé sur l'architecture Medallion, chargé de collecter les données depuis Stooq et Tiingo, de les nettoyer et de calculer des indicateurs financiers clés tels que les moyennes mobiles, le drawdown ou encore la volatilité. Le deuxième sprint a couvert la couche de gestion des accès : inscription des utilisateurs, authentification par matricule et réinitialisation de mot de passe. Le troisième sprint a introduit un module de prédiction s'appuyant sur l'algorithme Prophet, qui atteint une précision comprise entre 91 % et 98 % selon les actifs analysés. Enfin, le quatrième sprint a donné naissance à LEXA, un assistant conversationnel capable de répondre à des questions d'analyse financière comme de réglementation tunisienne.

Ce mémoire est structuré en six chapitres qui restituent fidèlement la progression du projet : le cadre général, l'étude préalable, puis un chapitre dédié à chacun des quatre sprints de développement.

1.1 Introduction

Ce premier chapitre pose les fondations contextuelles du projet. Avant d'aborder les aspects techniques, il convient de comprendre l'environnement dans lequel SmartFinance Advisor a vu le jour : une entreprise tunisienne devenue acteur reconnu de la FinTech mondiale, un secteur financier local insuffisamment outillé en matière d'analyse de données, et une opportunité réelle d'apporter une solution sur mesure. Nous présenterons successivement l'organisme d'accueil VERMEG, la problématique identifiée, puis les choix méthodologiques et architecturaux qui structurent l'ensemble du développement [1].

1.2 Présentation de l'Organisme d'Accueil : VERMEG

1.2.1 VERMEG : accompagner les institutions financières vers une réussite durable

Fondée à Tunis il y a plus de trente ans, VERMEG s'est progressivement imposée dans un secteur particulièrement compétitif : celui des logiciels dédiés aux banques et aux compagnies d'assurance. Ce qui surprend au premier abord, c'est l'envergure internationale qu'a su acquérir cette entreprise d'origine tunisienne : ses clients se trouvent aujourd'hui en Europe, en Afrique, au Moyen-Orient et en Amérique du Nord. Sa force réside notamment dans sa capacité à intégrer des technologies récentes au sein d'environnements financiers souvent rigides et fortement réglementés. [1]



FIGURE 1.1 – Bâtiment VERMEG Biwa

1.2.1.1 Historique et fondation

VERMEG a été fondée en 1993 à Tunis. À l'époque, peu auraient imaginé qu'une jeune entreprise tunisienne allait s'imposer comme un acteur incontournable de la FinTech mondiale. Pourtant, en concentrant ses efforts sur un créneau précis, la transformation numérique des institutions financières, elle a su bâtir une expertise solide et une réputation internationale, lui permettant de décrocher des contrats avec de grandes banques européennes et africaines.

1.2.1.2 Activités et domaines d'expertise

VERMEG intervient concrètement lorsqu'une banque ou une compagnie d'assurance souhaite moderniser ses systèmes d'information. Les domaines couverts sont variés : gestion des titres financiers, suivi des risques, conformité réglementaire ou encore aide à la décision. Ce qui distingue l'entreprise, et que nous avons pu observer lors de notre stage, c'est que ses équipes ne commercialisent pas des solutions génériques. Chaque outil est adapté aux contraintes spécifiques du client, ce qui confère à VERMEG une valeur ajoutée difficilement reproductible.

1.2.1.3 Présence mondiale

Présente en Europe, en Afrique, au Moyen-Orient et en Amérique du Nord, l'entreprise s'appuie sur une démarche d'innovation continue. Elle intègre dans ses solutions des technologies de pointe telles que le Big Data, le Cloud Computing, le Machine Learning et l'Intelligence Artificielle, afin de répondre aux évolutions rapides du secteur financier mondial.

1.3 Contexte général

1.3.1 Problématique

Dans le secteur financier tunisien, les données se sont imposées comme des actifs stratégiques incontournables. Pourtant, un écart persistant demeure entre la richesse des données disponibles et la capacité réelle des établissements à en tirer des décisions éclairées. C'est précisément dans cet espace que VERMEG positionne son expertise, en développant des solutions logicielles qui répondent aux réalités et aux exigences spécifiques du marché tunisien.

1.3.2 Étude de l'existant

Plusieurs outils d'analyse financière sont actuellement utilisés en Tunisie pour appuyer la prise de décision. Parmi les plus répandus, on distingue notamment :

— **Bloomberg Terminal** : Il s'agit d'une référence mondiale dans le domaine de l'analyse financière. Cette plateforme centralise les cours boursiers, les actualités économiques et des outils d'analyse avancés au sein d'une interface unifiée. Très prisée par les professionnels de la finance, elle offre une couverture exhaustive des marchés internationaux. Toutefois, son coût d'accès particulièrement élevé la rend inaccessible à la majorité des institutions financières tunisiennes, et son interface n'est pas conçue pour tenir compte des spécificités réglementaires du marché local [2].



FIGURE 1.2 – Logo Bloomberg Terminal

— **FactSet** : Davantage orientée vers les gestionnaires de portefeuilles et les analystes institutionnels, cette plateforme offre un accès consolidé aux données des marchés mondiaux ainsi qu'aux informations financières des entreprises publiques et privées. Bien qu'elle constitue un outil performant pour les grandes structures, elle souffre des mêmes limitations que Bloomberg dans le contexte tunisien : coût prohibitif et absence d'adaptation au cadre réglementaire local [3].



FIGURE 1.3 – Logo FactSet

1.3.3 Critique de l'existant

Si des plateformes telles que Bloomberg Terminal ou FactSet disposent indéniablement de fonctionnalités avancées en matière de collecte, d'analyse et de visualisation des données financières, leur adoption par les institutions tunisiennes reste marginale. Ce constat s'explique par plusieurs facteurs : un coût d'accès prohibitif, une absence totale de personnalisation et, surtout, une inadaptation à l'environnement réglementaire tunisien. Ces lacunes mettent en évidence la nécessité de disposer d'une solution dédiée, pensée et conçue en adéquation avec les spécificités du secteur financier local.

1.3.4 Solution proposée

Fort de ces constats, SmartFinance Advisor a été développée pour répondre précisément aux besoins des institutions financières tunisiennes. La plateforme propose une solution intégrée permettant de :

— Gérer et analyser les données financières locales, en les restituant sous forme de graphiques et de tableaux synthétiques ;
— Générer des prévisions fiables sur l'évolution de ces données, à partir de modèles calibrés pour le marché tunisien ;
— Fournir des recommandations automatiques et personnalisées, tout en garantissant la conformité avec la législation financière en vigueur ;
— Proposer une interface ergonomique, intuitive et intégrable avec les systèmes d'information existants.

SmartFinance Advisor se présente ainsi comme un outil innovant capable de pallier les insuffisances des solutions existantes, d'automatiser la conformité réglementaire et d'accompagner concrètement la transformation numérique des institutions financières tunisiennes.

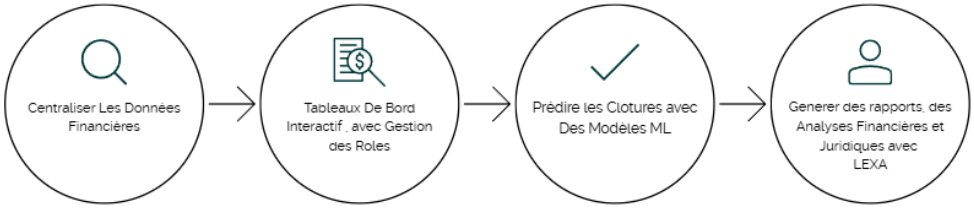


FIGURE 1.4 – Workflow SmartFinance Advisor

1.4 Méthodologie de travail

1.4.1 Étude comparative

Le choix d'une méthodologie de développement constitue une décision structurante pour tout projet logiciel. Deux grandes familles s'offrent généralement aux équipes : les méthodes traditionnelles, dites séquentielles, et les méthodes agiles, fondées sur l'itération et l'adaptabilité.

— **Waterfall** : Cette approche repose sur un enchaînement séquentiel et rigoureux de cinq phases distinctes — recueil des exigences, conception, implémentation, vérification et maintenance. Introduite par l'informaticien Winston Royce dans un article de recherche publié en 1970, elle présente l'avantage d'une planification claire et d'une documentation exhaustive. En revanche, sa rigidité la rend peu adaptée aux projets dont les besoins sont susceptibles d'évoluer en cours de développement [4].



FIGURE 1.5 – Les six étapes de la méthode Waterfall

Le choix de Scrum s'est imposé naturellement au regard des contraintes réelles de notre projet. SmartFinance Advisor est une plateforme dont les besoins ont évolué tout au long du développement : les exigences relatives au module de prédiction, par exemple, n'étaient pas entièrement définies en amont et ont été progressivement affinées à la suite des premiers résultats du pipeline ETL. Une méthode traditionnelle comme Waterfall, qui exige de figer l'intégralité des spécifications avant de commencer le développement, n'aurait pas toléré de tels ajustements sans remettre en question l'ensemble du planning.

Scrum nous a permis de découper le projet en quatre sprints indépendants, chacun produisant un incrément fonctionnel testable : le pipeline ETL, le système d'authentification, le module Prophet et l'assistant LEXA. Cette granularité a facilité les échanges avec les encadrants et permis d'identifier rapidement les blocages techniques, notamment les contraintes de performance liées aux insertions PostgreSQL, sans avoir à attendre la fin du projet pour y remédier.

1.4.1.1 Fonctionnement de la méthode

Scrum s'articule autour de sprints, des cycles de travail d'une durée de deux à quatre semaines [6], à l'issue desquels un incrément fonctionnel est livré. Plusieurs événements rythment ce processus :

— **Sprints :** Chaque sprint débute par une séance de planification au cours de laquelle l'équipe identifie et répartit les tâches à réaliser. Un livrable opérationnel est attendu à la clôture de chaque cycle.

— **Réunions quotidiennes :** Ces courtes réunions de quinze minutes, tenues chaque jour, favorisent la communication au sein de l'équipe Scrum en permettant à chacun de faire le point sur l'avancement de ses tâches et de signaler d'éventuels blocages.

— **Rétrospectives de sprint :** À l'issue de chaque sprint, une rétrospective animée par le Scrum Master offre à l'équipe l'occasion de dresser un bilan collectif, d'identifier ce qui a bien fonctionné et de définir des axes d'amélioration pour les cycles suivants.

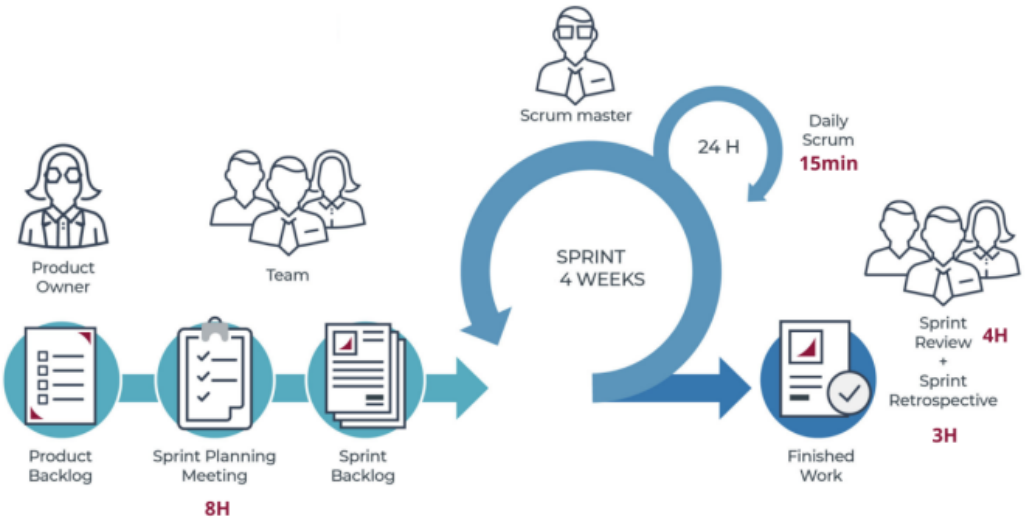


FIGURE 1.6 – Cycle de vie de l'approche SCRUM

1.4.1.2 Les rôles Scrum

La méthode Scrum repose sur une organisation claire des responsabilités, répartie entre trois rôles complémentaires :

— **Scrum Master :** Garant de la bonne application de la méthodologie, il facilite les cérémonies Scrum, aide l'équipe à lever les obstacles et veille au respect des principes agiles tout au long du projet.

— **Product Owner :** Responsable de la vision produit, il priorise le backlog et s'assure que les développements réalisés sont en adéquation avec les attentes des parties prenantes. Il constitue le principal point de liaison entre l'équipe technique et les commanditaires du projet.
— **Équipe de réalisation :** Composée de développeurs et de spécialistes aux compétences complémentaires, elle prend en charge l'implémentation des fonctionnalités et la livraison des incréments à l'issue de chaque sprint, en travaillant de manière autonome et auto-organisée.

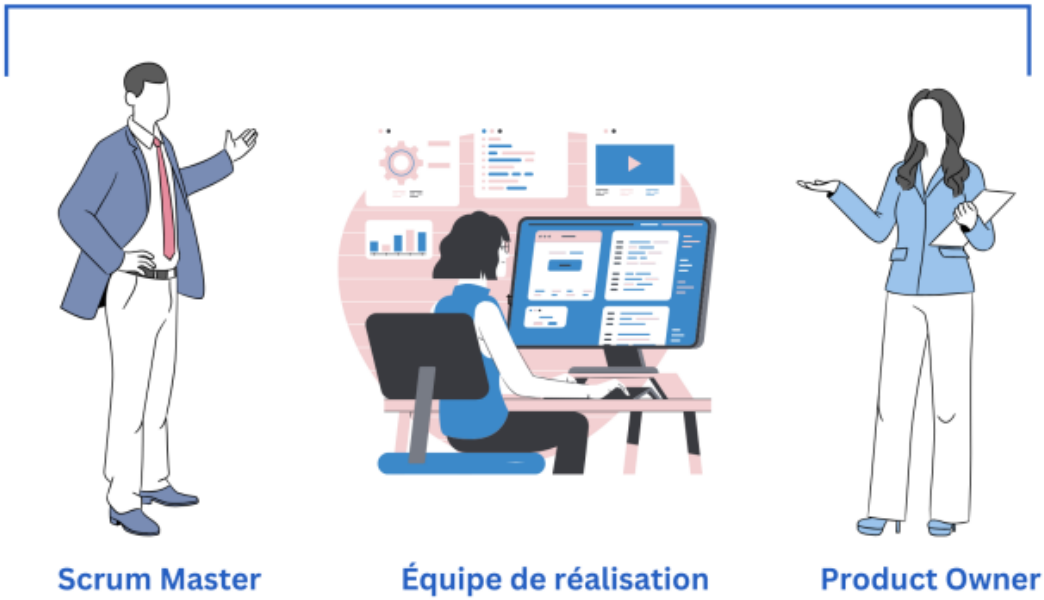


FIGURE 1.7 – L'équipe SCRUM

1.5 Langage et architecture de modélisation

1.5.1 Langage de modélisation unifié (UML)

Le langage de modélisation unifié (UML) constitue un standard reconnu pour la représentation visuelle des systèmes logiciels. Il permet de formaliser de manière rigoureuse les différents aspects d'une application qu'il s'agisse de ses acteurs, de ses processus, de ses interactions ou de sa structure de données, facilitant ainsi la communication entre les membres de l'équipe et la compréhension globale du système [7]. Dans le cadre de ce projet, nous avons eu recours aux diagrammes suivants :

— **Diagramme de cas d'utilisation :** pour décrire le fonctionnement général de l'application et les interactions entre les acteurs et le système ;
— **Diagramme de classes :** pour représenter la structure statique de l'application et les relations entre ses composants ;
— **Diagramme de séquence :** pour illustrer les interactions dynamiques entre les différents éléments du système lors de l'exécution des scénarios fonctionnels.



FIGURE 1.8 – Logo UML

1.5.2 Architecture en couches

Pour structurer la conception des diagrammes de séquence, nous avons adopté une architecture en couches. Cette approche consiste à diviser l'application en plusieurs niveaux logiques distincts, chacun ayant un rôle bien défini et pouvant être développé, testé et maintenu de manière indépendante. Particulièrement adaptée aux applications cloud et aux systèmes d'information complexes, l'architecture en couches simplifie l'infrastructure applicative tout en offrant une grande flexibilité et une meilleure portabilité du code [8].

1.6 Conclusion

Ce premier chapitre a permis de situer le projet dans son contexte réel : une entreprise d'envergure internationale confrontée à un besoin non couvert sur le marché financier tunisien. Les limites des solutions existantes Bloomberg Terminal et FactSet en tête, trop onéreuses et inadaptées au cadre réglementaire local, justifient pleinement le développement de SmartFinance Advisor. La méthodologie Scrum, associée à une architecture en couches, offre un cadre de travail à la fois structuré et évolutif, propice à la conduite des développements qui feront l'objet des chapitres suivants.

2.1 Introduction

Avant de produire la moindre ligne de code, il est indispensable de définir précisément ce que le système doit accomplir et comment l'équipe entend s'organiser pour le construire. Ce chapitre répond à ces deux questions fondamentales : il identifie les acteurs du système, formalise leurs besoins fonctionnels et non fonctionnels, et détaille l'environnement de travail matériel et logiciel qui supportera l'ensemble du développement. La planification en sprints Scrum et l'adoption de la méthodologie CRISP-DM pour la composante data y sont également présentées et justifiées.

2.2 Identification des acteurs

L'identification des acteurs constitue une étape préliminaire essentielle à la modélisation du système. En s'appuyant sur les caractéristiques fonctionnelles de l'application, trois acteurs principaux ont été identifiés :

- **Responsable Financier :** Il supervise les analystes financiers, consulte les tableaux de bord et accède à des recommandations fiscales adaptées à son périmètre de décision.
- **Analyste Financier :** Il importe les données financières dans la plateforme, consulte les dashboards analytiques et reçoit des recommandations personnalisées.
- **Système IA :** Il génère les prédictions financières à partir des données historiques.
- **LEXA :** Il donne des explications aux données financières, aux données juridiques et génère des rapports financiers.

2.3 Spécification des besoins

Afin de garantir l'efficacité et la cohérence de l'application, il est nécessaire de distinguer deux catégories de besoins : les besoins fonctionnels, qui décrivent les fonctionnalités attendues du système, et les besoins non fonctionnels, qui définissent les contraintes de qualité auxquelles il doit se conformer. Cette démarche facilite l'identification des différents modules de conception et contribue à l'élaboration d'un système répondant aux attentes de l'ensemble des parties prenantes.

2.3.1 Besoins fonctionnels

- **S'inscrire :** Permettre à un nouvel utilisateur de créer son compte d'Analyste Financier en renseignant les informations nécessaires à son identification sur la plateforme.
- **S'authentifier :** Permettre à un utilisateur de se connecter à la plateforme à l'aide de son matricule et de son mot de passe, afin d'accéder aux fonctionnalités correspondant à son rôle.

— **Gérer les comptes :** Offrir au Responsable Financier la possibilité de consulter, modifier ou supprimer les comptes des Analystes Financiers enregistrés sur la plateforme.
— **Lancer le pipeline d'ingestion et l'importation des données :** Déclencher l'ingestion des sources de données, assurer le contrôle de qualité et de conformité des fichiers, surveiller l'état des traitements en cours et valider l'import final des données dans la plateforme.
— **Importation manuelle de fichiers CSV ou Excel :** Permettre l'import manuel de fichiers structurés, en assurant la vérification de leur format et de leur structure, la prévisualisation des données, la correction des erreurs de cartographie, puis la validation de leur intégration.
— **Consulter les tableaux de bord :** Accéder aux dashboards analytiques, filtrer les données selon la période et les indicateurs souhaités, visualiser les indicateurs clés de performance (KPI) et exporter les vues nécessaires au reporting.
— **Module de prédiction :** Prédire le prix de clôture des actifs financiers à partir des données historiques, afin d'aider les utilisateurs à anticiper les tendances du marché et à appuyer leur prise de décision.
— **Recevoir des recommandations de LEXA :** Fournir des recommandations fiscales personnalisées, fondées sur les informations financières de l'utilisateur et conformes à la loi de finances tunisienne, tout en offrant des explications sur les règles appliquées.

2.3.2 Besoins non fonctionnels

— **Performance :** Le système doit garantir des temps de réponse rapides et une exécution efficiente des traitements, afin de satisfaire les exigences opérationnelles des utilisateurs.
— **Maintenabilité :** Le code source doit être clair, bien structuré et documenté, de manière à faciliter les interventions de maintenance et les évolutions futures du système.
— **Scalabilité :** L'architecture doit permettre au système de s'adapter à l'augmentation du volume de données et du nombre d'utilisateurs, sans dégradation notable des performances.
— **Sécurité :** La protection des données sensibles sera assurée par la mise en place d'un mécanisme d'authentification sécurisé, couplé à une gestion fine des rôles et des permissions, afin de contrôler précisément les actions autorisées pour chaque profil utilisateur.

2.3.3 Diagramme de cas d'utilisation global

Le diagramme de cas d'utilisation présenté ci-dessous modélise l'ensemble des interactions entre les acteurs et le système. On y distingue deux acteurs humains (Le **Responsable Financier** et l'**Analyste Financier**), ainsi que deux systèmes externes : le **Système IA** et **LEXA**. Le Responsable Financier est en charge de l'ingestion des données et de la gestion des accès, tandis que l'Analyste Financier peut s'inscrire et consulter les tableaux de bord. Les systèmes externes, quant à eux, assurent respectivement la génération des prédictions et la fourniture des recommandations fiscales.

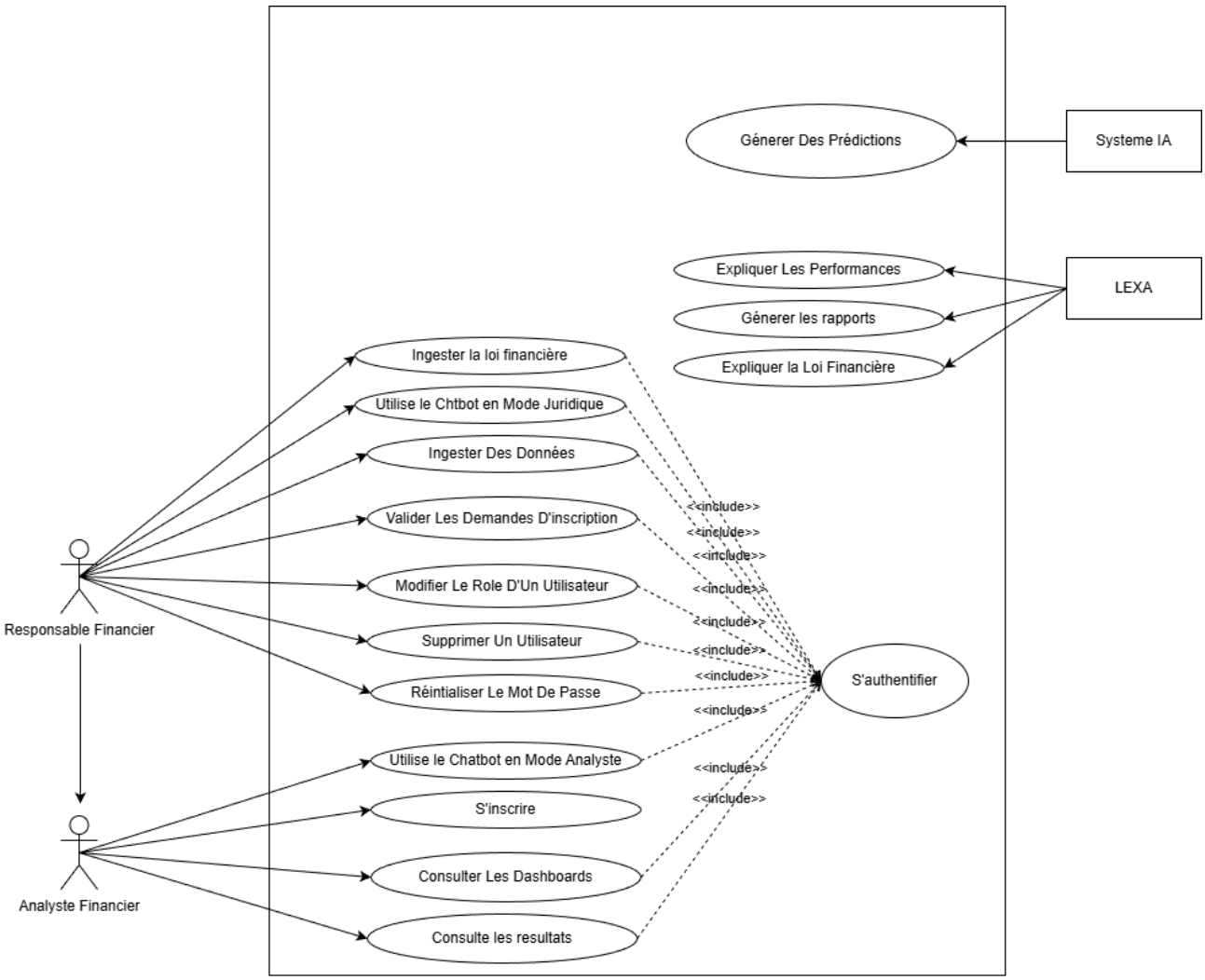


FIGURE 2.1 – Diagramme de cas d'utilisation global

2.3.4 Diagramme de classes global

La figure ci-dessous présente le diagramme de classes global du système, illustrant la structure statique de l'application ainsi que les relations entre ses principaux composants.

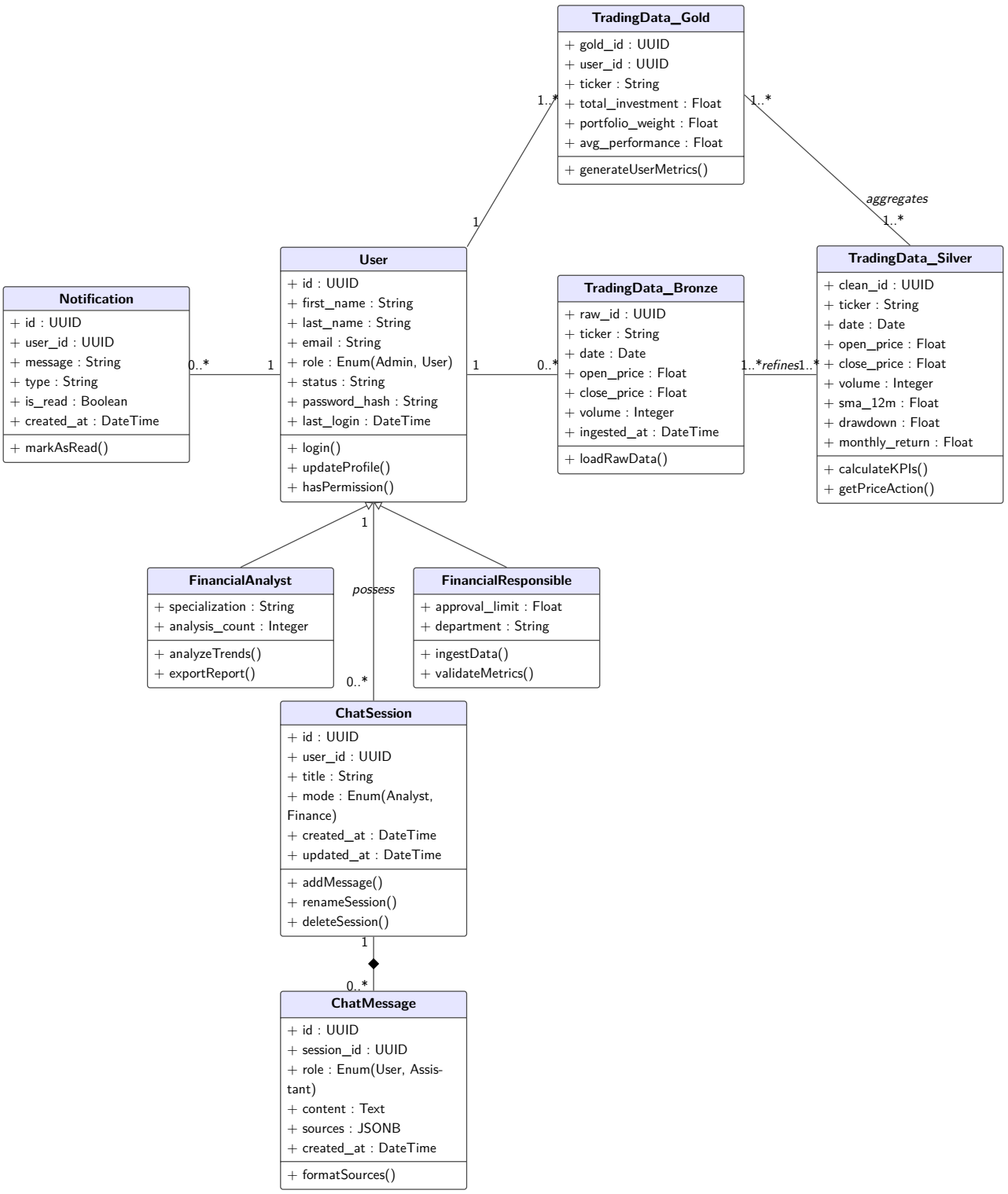


FIGURE 2.2 – Diagramme de classes global

2.4 Organisation du projet avec Scrum

2.4.1 Équipe et rôles

Le tableau 2.1 présente la composition de l'équipe Scrum et les responsabilités associées à chaque rôle.

TABLE 2.1 – Rôles et responsabilités au sein de l'équipe Scrum

Rôle	Responsabilités	Nom
Product Owner	Définir la vision produit, prioriser le backlog et optimiser la valeur livrée à chaque sprint.	Mme Zhioua Mariem
Scrum Master	Faciliter la mise en œuvre du processus Scrum et lever les obstacles rencontrés par l'équipe.	Mme Toumia Imen
Équipe de développement	Concevoir et réaliser les fonctionnalités et livrables du projet.	Hayouni Mohamed Zaidi Moktada

2.4.2 Backlog du produit

TABLE 2.2 – Backlog général de SmartFinance Advisor

ID	Fonctionnalité	User Story	Tâche
Sprint 1 — ETL et Extraction des Données			
1	Contrôle d'ingestion (API)	En tant qu'administrateur, je souhaite pouvoir déclencher manuellement le pipeline ETL depuis l'API afin de garder le contrôle sur les opérations d'ingestion.	Mettre en place la route HTTP `/start` avec protection par authentification ; introduire un verrou anti-concurrence pour éviter les exécutions parallèles ; lancer `main.py` de façon asynchrone.
2	Supervision (API)	En tant qu'administrateur, je souhaite suivre en temps réel l'avancement du traitement en consultant les journaux d'exécution directement depuis l'interface.	Mettre en mémoire tampon le `stdout` du sous-processus ; exposer les logs via la route HTTP `/logs` ; intégrer la lecture en continu côté frontend.
3	Extraction (Couche Bronze)	Le système doit être capable d'extraire les données brutes issues des sources externes (Stooq, Tiingo) de manière rapide et non bloquante.	Effectuer les appels API de façon asynchrone ; convertir les résultats en DataFrames pandas ; procéder à l'insertion brute en base SQL via la commande `COPY`.
4	Nettoyage (Couche Silver)	Le système doit harmoniser et nettoyer les historiques collectés depuis les différentes sources, en éliminant les doublons et en unifiant les formats.	Renommer les colonnes selon une convention unifiée ; appliquer les transtypages nécessaires ; filtrer les entrées non conformes ; effectuer un upsert vers la table silver.
5	Agrégation (Couche Gold)	Le système doit calculer les indicateurs financiers clés, notamment la moyenne mobile simple et le drawdown, afin d'alimenter les tableaux de bord analytiques.	Mettre en œuvre le fenêtrage pandas ; calculer la volatilité glissante ainsi que les pics cumulés et drawdowns ; effectuer l'upsert vers la table gold.

ID	Fonctionnalité	User Story	Tâche
Sprint 2 — Inscription, Authentification et Gestion des Comptes			
6	S'inscrire	En tant que nouvel utilisateur, je souhaite pouvoir créer un compte personnel afin d'obtenir un accès à la plateforme.	Concevoir une page d'inscription permettant de saisir et soumettre les informations personnelles nécessaires à la création du compte.
7	Se connecter	En tant qu'utilisateur enregistré, je souhaite m'authentifier sur la plateforme pour accéder à mes données et fonctionnalités.	Réaliser une page de connexion sécurisée permettant à chaque utilisateur de s'identifier à l'aide de son matricule et de son mot de passe.
8	Mot de passe oublié	En tant qu'utilisateur, je souhaite pouvoir réinitialiser mon mot de passe de manière autonome lorsque je ne m'en souviens plus.	Mettre en place un processus de réinitialisation en trois étapes : saisie de l'adresse e-mail, réception d'un code de vérification, puis définition d'un nouveau mot de passe après validation du code reçu.
9	Accepter de nouveaux utilisateurs	En tant qu'administrateur, je souhaite examiner et valider les demandes d'inscription afin de contrôler l'accès au système et de transmettre les identifiants aux utilisateurs approuvés.	Développer une interface d'administration permettant de consulter les demandes en attente, d'approuver les inscriptions et de déclencher automatiquement l'envoi des identifiants par e-mail.
10	Modifier les informations des utilisateurs	En tant qu'administrateur, je souhaite pouvoir mettre à jour les informations d'un compte utilisateur afin de maintenir l'exactitude des données dans le système.	Concevoir une interface d'édition permettant à l'administrateur de consulter et de modifier les attributs d'un profil utilisateur existant.
11	Supprimer des comptes utilisateurs	En tant qu'administrateur, je souhaite pouvoir supprimer les comptes qui ne sont plus actifs, afin de maintenir une base d'utilisateurs à jour.	Implémenter une fonctionnalité de suppression de compte assortie d'une étape de confirmation explicite, pour prévenir toute suppression accidentelle.

ID	Fonctionnalité	User Story	Tâche
Sprint 3 — Développement du Modèle Prédictif			
12	Modélisation prédictive	En tant qu'analyste, je souhaite disposer d'un modèle de prévision temporelle capable de projeter l'évolution future des données financières.	Concevoir et intégrer un modèle basé sur l'algorithme Prophet, entraîné sur les données historiques, et produisant des prévisions accompagnées de leurs intervalles de confiance.
13	Évaluation de fiabilité	En tant qu'utilisateur, je souhaite pouvoir consulter les métriques de performance du modèle afin d'estimer le niveau de confiance à accorder aux prévisions.	Implémenter le calcul des indicateurs d'erreur standards (MAE, RMSE, sMAPE) et les agréger en un score de précision synthétique affiché dans l'interface.
14	Visualisation graphique	En tant qu'utilisateur, je souhaite visualiser les projections futures sur un graphique interactif pour faciliter l'interprétation des tendances.	Intégrer un composant graphique affichant simultanément les données historiques, la courbe de prévision et les bornes de l'intervalle de confiance associé.
15	Configuration d'horizon	En tant qu'utilisateur, je souhaite définir librement la fenêtre temporelle de prédiction afin d'analyser les tendances à court, moyen ou long terme selon mes besoins.	Ajouter dans l'interface des contrôles de sélection permettant de paramétrer la fréquence et l'étendue de l'horizon de prévision.

ID	Fonctionnalité	User Story	Tâche
Sprint 4 — Développement de l'Assistant Intelligent LEXA			
16	Interface de Chat	En tant qu'utilisateur, je souhaite interagir avec l'assistant LEXA au travers d'une interface de messagerie fluide, afin de poser mes questions et obtenir des réponses en temps utile.	Développer l'interface de discussion (frontend) : gestion de l'envoi des messages, affichage des états de chargement liés aux appels IA, et rendu structuré des réponses reçues.
17	Mode de Discussion	En tant qu'utilisateur, je souhaite pouvoir choisir entre plusieurs modes de discussion (Analyste ou Juridique) pour obtenir des réponses adaptées à mon contexte et à mes besoins métier.	Mettre en place un sélecteur de mode côté client et ajuster dynamiquement le comportement du backend en fonction du domaine sélectionné.
18	Recherche RAG	En tant qu'utilisateur opérant en mode Juridique, je souhaite que les réponses de LEXA soient fondées sur des textes légaux réels issus de la base documentaire, afin de garantir leur pertinence et leur fiabilité.	Développer les services RAG côté backend : interrogation de la base vectorielle ChromaDB, extraction des passages sources pertinents, puis génération de la réponse finale via l'API Gemini.
19	Gestion des Sessions	En tant qu'utilisateur, je souhaite retrouver l'historique de mes échanges avec LEXA sous forme de sessions sauvegardées, pour pouvoir reprendre ou consulter d'anciennes analyses.	Mettre en œuvre la persistance des conversations en base PostgreSQL : création, lecture, renommage et suppression des sessions utilisateur.
20	Ingestion de Lois	En tant qu'administrateur, je souhaite pouvoir importer de nouveaux textes réglementaires (lois, décrets, circulaires) afin d'enrichir continuellement la base de connaissances de l'assistant.	Créer une interface d'administration dédiée, offrant un tableau de bord permettant l'import de documents textuels et le déclenchement du pipeline d'ingestion correspondant.
21	Pipeline ETL Loi	En tant qu'administrateur, je souhaite que les documents importés soient automatiquement traités, nettoyés et vectorisés, sans intervention manuelle supplémentaire.	Concevoir le pipeline de traitement interne : extraction du texte brut (couche Bronze), segmentation et nettoyage des contenus (couche Silver), génération des embeddings et insertion dans ChromaDB (couche Gold).

2.4.3 Organisation des sprints

La définition de la stratégie de découpage en sprints relève de la responsabilité du Product Owner, qui veille à garantir une planification cohérente et une mise en œuvre progressive et maîtrisée du projet. Cette étape de concertation favorise une vision commune au sein de l'équipe et assure une approche structurée dans la conduite du développement.

La figure 2.3 illustre les cinq sprints identifiés pour ce projet.

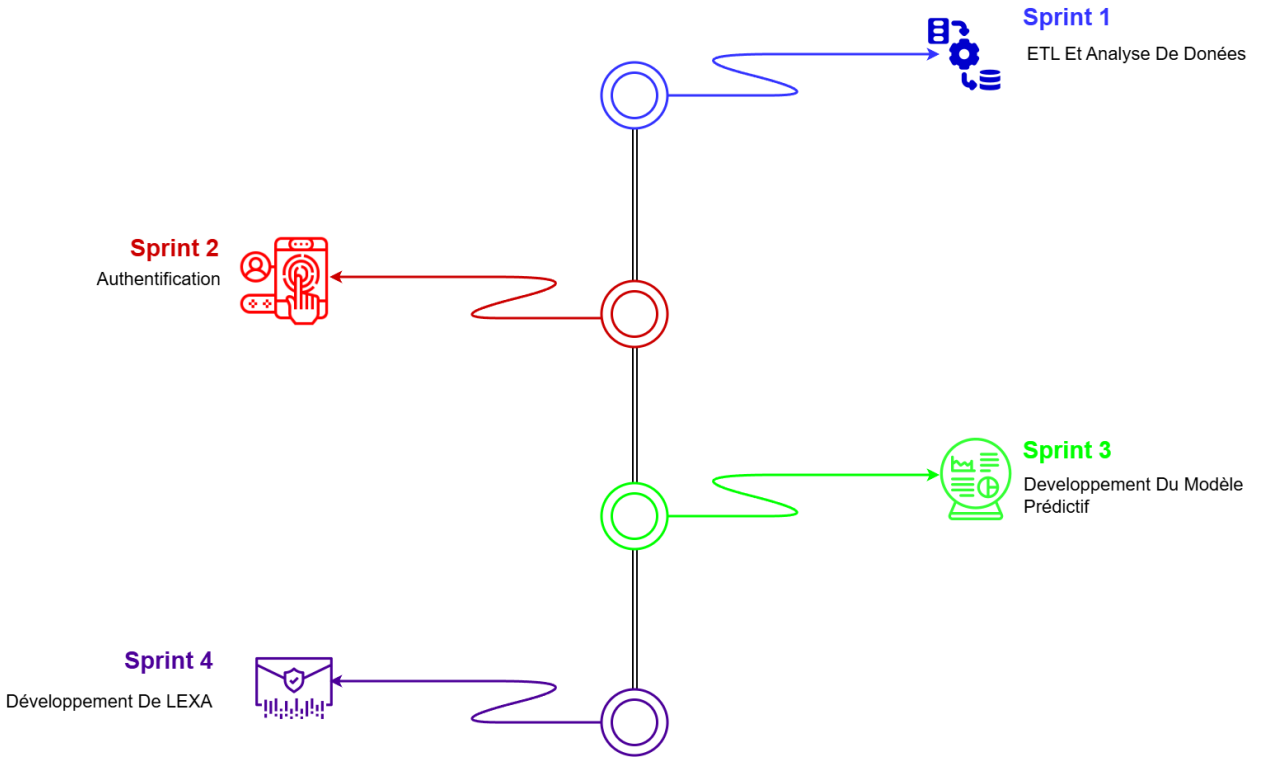


FIGURE 2.3 – Découpage du projet en sprints

2.4.4 Planification

La planification du projet a été formalisée à l'aide d'un diagramme de Gantt, qui offre une représentation visuelle claire de la séquence des tâches, des dépendances qui les relient et de leur distribution dans le temps. Cet outil nous a permis d'assurer un suivi rigoureux de l'avancement du projet et d'anticiper les éventuels chevauchements entre les différentes phases de développement.

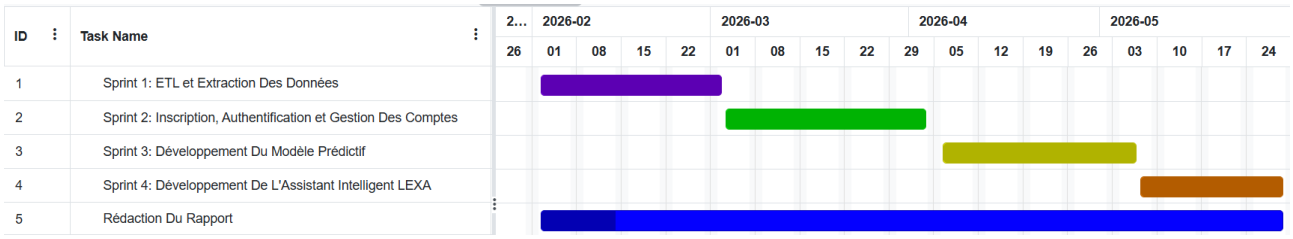


FIGURE 2.4 – Diagramme de GANTT

2.5 Méthodologie CRISP-DM

2.5.1 Présentation de la méthodologie CRISP-DM

Afin d'assurer une gestion rigoureuse de la composante data du projet, nous avons adopté la méthodologie CRISP-DM (*Cross Industry Standard Process for Data Mining*), publiée en 1999 pour standardiser les processus d'exploration de données dans l'ensemble des secteurs d'activité [9]. Cette méthodologie a permis de structurer les différentes étapes du traitement des données,

depuis la compréhension du besoin métier jusqu'à la préparation, la modélisation et l'évaluation des résultats.

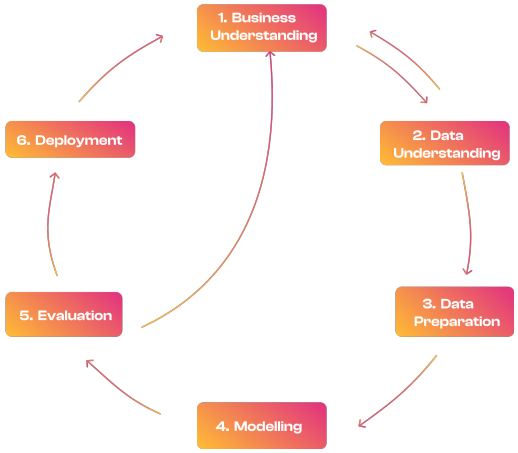


FIGURE 2.5 – Cycle de vie de la méthodologie CRISP-DM

2.5.2 Justification du choix méthodologique

CRISP-DM a été retenu en complément de Scrum pour encadrer spécifiquement la partie data du projet, qui représente une part significative du travail réalisé : collecte, nettoyage, transformation, modélisation et évaluation des données boursières. Ces étapes se distinguent des tâches de développement logiciel classiques par leur nature itérative et fortement interdépendante, qui impose des allers-retours fréquents entre la compréhension des données et leur préparation.

Dans notre contexte, CRISP-DM nous a guidés depuis l'identification des sources de données Stooq et Tiingo jusqu'à la validation du pipeline via le rapport de *data profiling* généré avec `ydata-profiling`. Le caractère cyclique de la méthodologie nous a notamment permis de revenir sur la phase de nettoyage après avoir détecté des incohérences OHLCV (prix *High* inférieur au prix *Low*) lors de l'évaluation de la couche Silver, sans pour autant remettre en cause l'architecture globale du système.

2.6 Environnement de travail

Cette section décrit l'environnement de travail mis en place pour la réalisation du projet. Il se compose de deux volets complémentaires : l'environnement matériel, qui précise les caractéristiques techniques des équipements utilisés, et l'environnement logiciel, qui recense les technologies retenues, langages de programmation, outils de développement et système de gestion de base de données.

2.6.1 Environnement matériel

Le développement du projet a été conduit sur deux postes de travail dont les caractéristiques sont détaillées dans les tableaux ci-dessous.

Table 2.3 – Caractéristiques de l'ordinateur 1

Ordinateur	**MSI**
Processeur	11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz
RAM	16,0 Go
Stockage	512 Go SSD
Système d'exploitation	Windows 11 PRO
Carte graphique	Nvidia GeForce GTX 1650

Table 2.4 – Caractéristiques de l'ordinateur 2

Ordinateur	**MSI**
Processeur	11th Gen Intel(R) Core(TM) i5-11400H @ 2.70GHz
RAM	16,0 Go
Stockage	512 Go SSD
Système d'exploitation	Windows 11 PRO
Carte graphique	Nvidia GeForce GTX 1650

2.6.2 Environnement logiciel

Le choix des outils et technologies a été guidé par les contraintes concrètes du projet plutôt que par des préférences arbitraires. Chaque composant de la stack a été retenu pour sa capacité à répondre à un besoin précis, qu'il s'agisse du traitement des données, de l'exposition des API ou de la construction de l'interface utilisateur.

Visual Studio Code a été adopté comme éditeur principal par les deux membres de l'équipe. Son intégration native avec Git et sa capacité à gérer simultanément des fichiers TypeScript et des scripts Python en ont fait un environnement de développement particulièrement adapté à notre stack hybride [10].



Figure 2.6 – Logo Visual Studio Code

Angular a été retenu pour la construction de l'interface web. Ce framework de Google repose sur une architecture en composants indépendants [11], ce qui a facilité le travail en parallèle sur différentes parties de l'interface. Son système de services et d'observables s'est également bien prêté à la gestion des appels API asynchrones vers le backend Flask.



FIGURE 2.7 – Logo Angular

TypeScript, imposé par Angular, s'est révélé être un atout indéniable sur un projet de cette envergure. Le typage statique a permis de détecter en amont plusieurs catégories d'erreurs qui n'auraient été visibles qu'à l'exécution en JavaScript pur [12], contribuant ainsi à la robustesse de l'interface.



FIGURE 2.8 – Logo TypeScript

Tailwind CSS a été utilisé pour la mise en forme de l'interface. Son approche utilitaire, qui consiste à appliquer des classes directement dans le balisage HTML [13], accélère sensiblement le développement et garantit une cohérence visuelle sans nécessiter la gestion de fichiers CSS séparés.



FIGURE 2.9 – Logo Tailwind CSS

Python constitue le langage de référence pour l'ensemble du backend et du pipeline de données [14]. Ce choix s'est imposé naturellement au regard de la richesse de son écosystème : `pandas` pour la manipulation des DataFrames, `Prophet` pour les prédictions de séries temporelles, et `psycopg2` pour les interactions avec PostgreSQL.



FIGURE 2.10 – Logo Python

Flask a été choisi pour exposer la logique métier sous forme d'API REST. Par rapport à Django, il offre une plus grande liberté dans la structuration des routes [15], ce qui nous a permis de définir des endpoints spécifiques pour les KPI, les prédictions et la gestion des utilisateurs, sans être contraints par une architecture imposée.



FIGURE 2.11 – Logo Flask

PostgreSQL constitue le système de gestion de base de données central du projet [16]. La base a été organisée en trois schémas distincts, *bronze*, *silver* et *gold*, afin de séparer proprement les couches de traitement. PostgreSQL a démontré une bonne tenue en charge sur nos 54 000 enregistrements, et le support des *upserts* idempotents a permis d'éviter les doublons lors des relances successives du pipeline.



FIGURE 2.12 – Logo PostgreSQL

Supabase a été utilisé pour héberger l'instance PostgreSQL, évitant ainsi la mise en place d'une infrastructure serveur dédiée. La plateforme fournit une interface d'administration pratique et permet une connexion directe via `psycopg2`, sans surcoût opérationnel [17].



FIGURE 2.13 – Logo Supabase

Postman a été systématiquement utilisé pour valider chaque endpoint avant sa connexion avec le frontend Angular. Cet outil a permis de déboguer les réponses JSON et de mesurer les temps de réponse des routes analytiques dans un environnement isolé du frontend [18].



FIGURE 2.14 – Logo Postman

Git et GitHub ont assuré la gestion des versions tout au long du projet. L'équipe a adopté une stratégie de branches par sprint, permettant d'isoler les développements en cours des versions stables et de faciliter les revues de code entre membres [19].



FIGURE 2.15 – Logo Git



FIGURE 2.16 – Logo GitHub

Overleaf a servi de plateforme collaborative pour la rédaction du présent rapport en LaTeX [20]. Son éditeur en ligne a permis aux deux membres de l'équipe de contribuer simultanément au document, tout en garantissant une mise en page rigoureuse et cohérente.



FIGURE 2.17 – Logo Overleaf

Chart.js a été intégré dans Angular pour la réalisation des graphiques du tableau de bord [21]. Plus léger que D3.js pour notre cas d'usage, il offre une intégration directe via des composants *canvas* et couvre l'ensemble des types de visualisations dont nous avions besoin.



FIGURE 2.18 – Logo Chart.js

Draw.io a été utilisé pour la conception de l'ensemble des diagrammes UML, cas d'utilisation, séquence et classes [22]. Son interface graphique intuitive et son export au format XML ont permis de produire rapidement les artefacts de modélisation à chaque sprint.



FIGURE 2.19 – Logo Draw.io

2.7 Conclusion

À l'issue de cette étude préalable, le périmètre fonctionnel du projet est clairement délimité et l'équipe dispose d'un plan de travail structuré en quatre sprints progressifs. Les choix technologiques ont été motivés par les contraintes effectives du projet, volume de données financières, nécessité d'une API performante, interface réactive, et non par des considérations arbitraires. Le chapitre suivant entre dans le vif du sujet en abordant la construction du pipeline ETL, premier livrable concret du projet.

3.1 Introduction

La disponibilité et la fiabilité des données constituent le socle sur lequel repose l'ensemble de la plateforme : sans pipeline robuste, ni le tableau de bord ni le module de prédiction ne peuvent fonctionner correctement. L'enjeu de ce premier sprint est précisément de construire une chaîne de traitement capable de récupérer les données boursières depuis deux sources distinctes, de les nettoyer et de calculer les indicateurs nécessaires au reste de l'application. Ce pipeline est structuré en trois couches successives désignées Bronze, Silver et Gold. Chacune assumant un rôle bien défini dans le processus de transformation des données.

3.2 Backlog du Sprint 1

Le tableau ci-dessous présente le backlog du premier sprint, découpé en fonctionnalités et tâches techniques associées.

TABLE 3.1 – Découpage fonctionnel et technique du pipeline Medallion

ID	Fonctionnalité	User Story	Tâche
1	Contrôle ingestion (API)	En tant qu'administrateur, je veux pouvoir déclencher le pipeline ETL via l'API.	Créer la route HTTP `/start` protégée ; implémenter un verrou anti-concurrence ; lancer `main.py` en asynchrone.
2	Supervision (API)	En tant qu'administrateur, je veux consulter les logs en temps réel pour surveiller l'avancement du traitement.	Bufferiser le `stdout` du sous-processus ; créer la route HTTP `/logs` ; intégrer la lecture côté frontend.
3	Extraction (Couche Bronze)	Le système doit extraire les données sources (Stooq, Tiingo) de manière rapide et asynchrone.	Appels API asynchrones ; conversion en DataFrames pandas ; insertion brute SQL via `COPY`.
4	Nettoyage (Couche Silver)	Le système doit harmoniser, nettoyer et dédoublonner les historiques provenant des différentes sources.	Renommage unifié des colonnes ; transtypage ; application des règles de filtrage ; upsert vers la table `silver`.
5	Agrégation (Couche Gold)	Le système doit calculer les KPIs financiers (SMA, Drawdown) pour alimenter les tableaux de bord.	Fenêtrage pandas ; calcul de la volatilité glissante ; calcul des pics cumulés et drawdowns ; upsert vers la table `gold`.

3.3 Spécifications fonctionnelles

3.3.1 Diagramme de cas d'utilisation du Sprint 1

Dans le cadre de ce sprint, le **Responsable Financier** est chargé de l'ingestion des données et de la supervision des accès des **Analystes Financiers**. Ces derniers peuvent, quant à eux, consulter les tableaux de bord, visualiser les graphiques et analyser les indicateurs de performance.

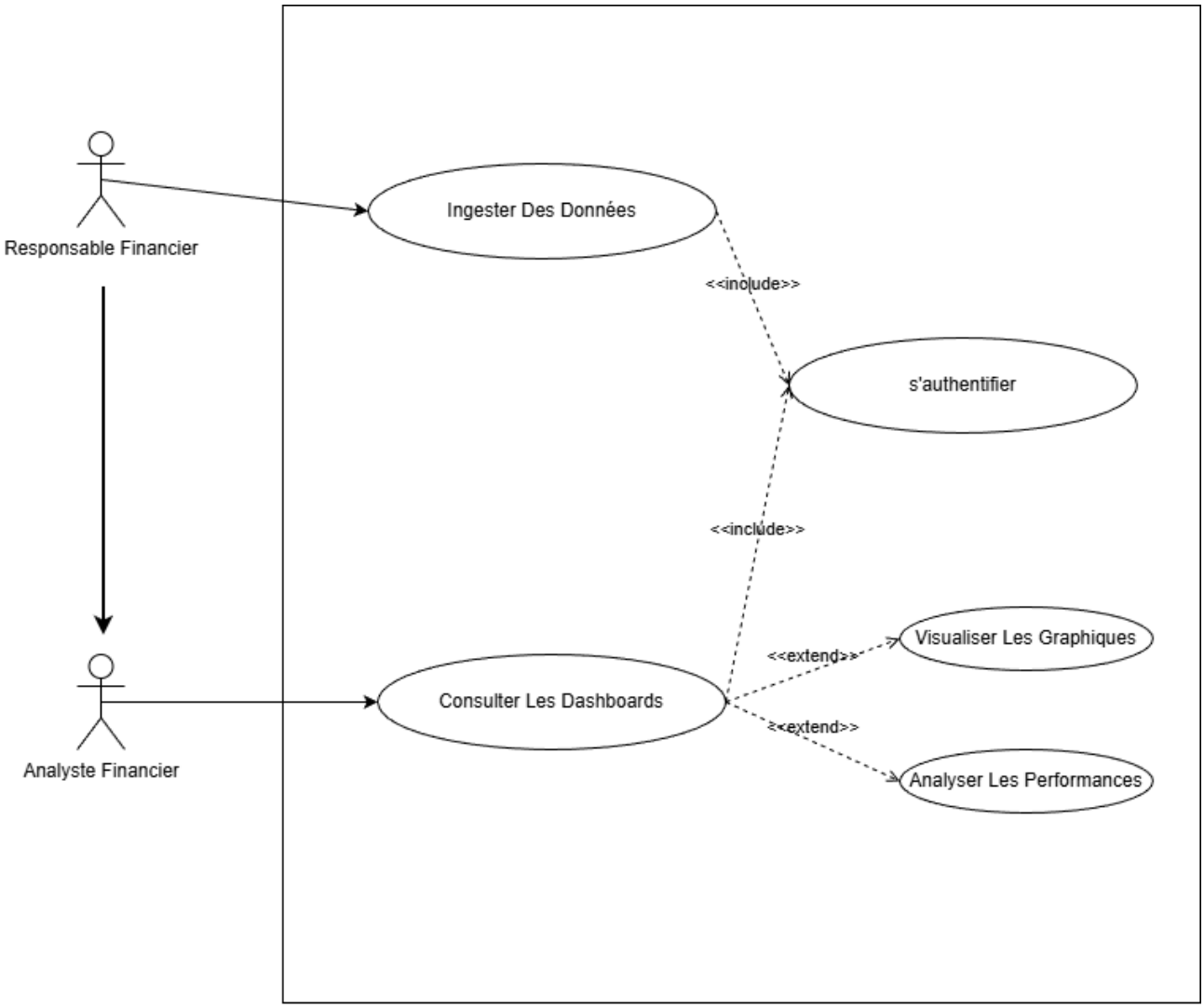


FIGURE 3.1 – Diagramme de cas d'utilisation du Sprint 1

3.3.2 Description détaillée des cas d'utilisation

3.3.2.1 Description Détaillée du cas d'utilisation "Ingestion des données"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation relatif à l'ingestion des données financières.

Table 3.2 – Description du cas d'utilisation : Ingestion des données financières

Titre	Cas d'utilisation : Ingérer des données
Acteur principal	Responsable Financier
Description	Le responsable financier importe des données financières depuis différentes sources afin de les intégrer dans le système.
Préconditions	1. L'utilisateur est authentifié. 2. Les sources de données sont accessibles.
Postconditions	1. Les données sont intégrées dans le système. 2. Elles sont stockées dans la couche Bronze.
Scénario nominal	1. Accéder à l'interface d'ingestion. 2. Sélectionner la source de données. 3. Vérifier le format des données. 4. Lancer l'importation. 5. Confirmer l'intégration.
Scénarios alternatifs	Données invalides ou erreur d'accès → échec de l'importation.
Règles de gestion	Les données doivent respecter un format valide avant d'être ingérées.

3.3.2.2 Description Détaillée du cas d'utilisation "Consultation des tableaux de bord"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation relatif à la consultation des tableaux de bord.

Table 3.3 – Description du cas d'utilisation : Consultation des tableaux de bord

Titre	Cas d'utilisation : Consulter les tableaux de bord
Acteur principal	Responsable Financier / Analyste Financier
Description	L'analyste consulte les tableaux de bord afin de visualiser les indicateurs financiers.
Préconditions	1. L'utilisateur est authentifié. 2. Les données sont disponibles dans le système.
Postconditions	1. Les tableaux de bord sont affichés. 2. Les données sont exploitables visuellement.
Scénario nominal	1. Accéder à la page des tableaux de bord. 2. Charger les données. 3. Afficher les visualisations.
Scénarios alternatifs	Aucune donnée disponible → affichage d'un message informatif.
Règles de gestion	Les tableaux de bord n'affichent que des données préalablement validées.

3.3.2.3 Description Détaillée du cas d'utilisation "Visualisation des graphiques"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation relatif à la visualisation des graphiques.

TABLE 3.4 – Description du cas d'utilisation : Visualisation des graphiques

Titre	Cas d'utilisation : Visualiser les graphiques
Acteur principal	Responsable Financier / Analyste Financier
Description	Permet d'afficher des représentations graphiques des données financières pour en faciliter l'interprétation visuelle.
Préconditions	1. L'utilisateur est authentifié. 2. Un tableau de bord est déjà chargé.
Postconditions	1. Les graphiques sont affichés. 2. Les données sont restituées sous une forme facilement interprétable.
Scénario nominal	1. Sélectionner un indicateur. 2. Générer le graphique correspondant. 3. Afficher le graphique.
Scénarios alternatifs	Erreur d'affichage → rechargement du graphique.
Règles de gestion	Les graphiques doivent refléter des données actualisées.

3.3.2.4 Description Détaillée du cas d'utilisation "Analyse des performances"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation relatif à l'analyse des performances financières.

TABLE 3.5 – Description du cas d'utilisation : Analyse des performances financières

Titre	Cas d'utilisation : Analyser les performances
Acteur principal	Responsable Financier / Analyste Financier
Description	L'analyste évalue les performances financières en s'appuyant sur les indicateurs et les visualisations disponibles.
Préconditions	1. L'utilisateur est authentifié. 2. Les données sont disponibles dans les tableaux de bord.
Postconditions	1. Des conclusions sont tirées à partir des données. 2. Des décisions peuvent être prises sur cette base.
Scénario nominal	1. Consulter les indicateurs clés. 2. Identifier les tendances observables. 3. Interpréter les résultats.
Scénarios alternatifs	Données insuffisantes → analyse partielle ou limitée.
Règles de gestion	L'analyse doit reposer sur des données fiables et à jour.

3.4 Conception

3.4.1 Diagrammes de séquence

Les diagrammes de séquence permettent de modéliser les interactions dynamiques entre les acteurs du système et ses composants internes. Ils illustrent le déroulement temporel des opérations ainsi que les échanges de messages entre les entités impliquées.

Dans notre système, ces diagrammes mettent en évidence les principales fonctionnalités couvertes par ce sprint, notamment l'ingestion des données, la consultation des tableaux de bord et l'analyse des performances financières.

3.4.1.1 Diagramme de séquence "Ingestion De Données"

Le script d'Ingestion déclenche séquentiellement les traitements à travers les trois couches successives, enregistre les résultats d'exécution en base de données, puis assure le nettoyage des fichiers temporaires locaux.

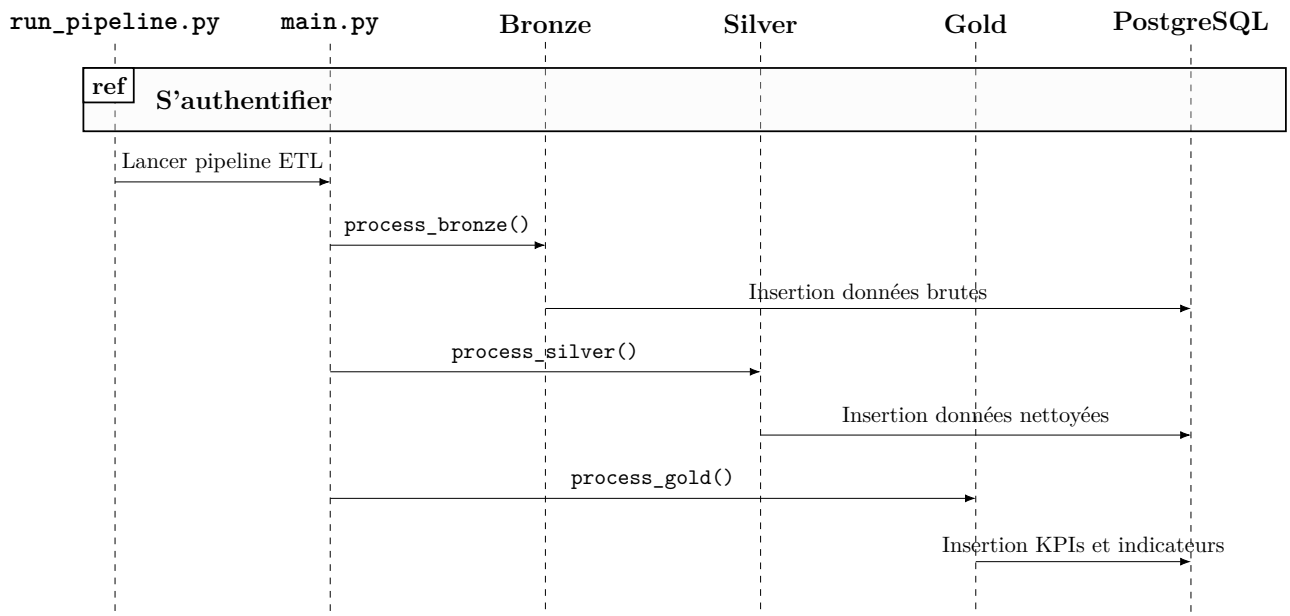


FIGURE 3.2 – Diagramme de séquence "Ingestion Des Données"

3.5 Flux de données détaillé : l'architecture Medallion

Le principe fondateur de l'architecture Medallion repose sur l'immutabilité des données brutes. Les données entrent dans la couche Bronze dans leur état original, exactement telles qu'elles sont reçues depuis Stooq ou Tiingo. C'est uniquement dans la couche Silver que débute le travail de nettoyage, de standardisation et de vérification des cohérences. La couche Gold, quant à elle, contient les données enrichies avec l'ensemble des indicateurs calculés, directement consommables par le frontend [23].

Cette séparation stricte des responsabilités s'est avérée déterminante lors du développement : lorsque des anomalies ont été détectées dans les calculs de drawdown, il a été possible de remonter jusqu'à la couche Bronze pour identifier l'origine du problème, sans avoir altéré les données d'origine .

3.6 Compréhension métier

La première étape de la méthodologie CRISP-DM porte sur la compréhension des objectifs du projet au niveau métier, avant toute considération technique.

3.6.1 Objectifs métier

L'objectif principal est de concevoir une plateforme d'aide à la décision financière capable de transformer des données brutes en indicateurs stratégiques exploitables. Les besoins identifiés se structurent autour de trois axes :

— **Centralisation :** Consolider les données issues de fichiers CSV historiques et d'API de marché (Stooq, Tiingo) au sein d'une base de données unifiée.

— **Fiabilité :** Garantir l'intégrité des données OHLCV (*Open, High, Low, Close, Volume*) par l'application de règles de validation strictes.

— **Automatisation :** Réduire les tâches manuelles en calculant automatiquement les indicateurs de performance (KPIs) à chaque exécution du pipeline.

3.6.2 Évaluation de la situation

Le projet repose sur une architecture **Medallion** implémentée en Python, avec PostgreSQL comme système de gestion de base de données.

— **Ressources :** Des modules dédiés ont été développés pour chacune des trois couches : ingestion (*Bronze*), nettoyage (*Silver*) et analyse (*Gold*).

— **Contraintes :** Le pipeline doit gérer les valeurs manquantes, les formats de dates hétérogènes ainsi que les valeurs aberrantes (*outliers*) présentes dans les flux de données sources.

3.6.3 Objectifs de data mining

Pour répondre aux besoins métier identifiés, le backend doit effectuer les transformations suivantes :

— **Standardisation (Couche Silver) :** Conversion des types de données et vérification des relations logiques entre les prix, notamment :

$$Low \leq Open,\ Close \leq High$$

— **Enrichissement (Couche Gold) :** Calcul de huit indicateurs clés, parmi lesquels :

— La *Moyenne Mobile Simple* (SMA 12M et 24M) pour l'identification des tendances ;

— Le *Drawdown*, indicateur de risque calculé comme suit :

$$Drawdown = \frac{Close - Peak}{Peak}$$

— La *Volatilité glissante* (*Rolling Volatility*), pour quantifier l'incertitude du marché.

3.7 Compréhension des données

Cette phase détaille l'origine, la structure et la qualité des données financières utilisées dans le pipeline.

3.7.1 Collecte des données

Le pipeline consomme des données provenant de deux sources principales et complémentaires :

— **Stooq (fichiers CSV) :** Cette source fournit un historique approfondi des actifs sélectionnés (AAPL, TSLA, etc.), permettant d'alimenter les algorithmes de prédiction avec plusieurs décennies de données.

```python
def _download_stooq_csv_to_file(label: str, interval: str = API_INTERVAL, out_dir: str = "data") -> str | None:
    """
    Download Stooq CSV for one ticker and save locally.
    Returns the local filepath, or None on failure.
    """
    _ensure_data_dir(out_dir)
    stooq_ticker = stooq_label_map.get(label, label.lower())
    url = _build_stooq_url(stooq_ticker=stooq_ticker, interval=interval, start_date=API_START_DATE)
    out_path = os.path.join(out_dir, f"{label}.csv")

    try:
        # Télecharger Le Fichier CSV
        r = requests.get(url, timeout=60)
        if r.status_code != 200:
            print(f"CSV {label}: HTTP {r.status_code}")
            return None
        with open(out_path, "wb") as f:
            f.write(r.content)
        return out_path
    except Exception as e:
        print(f"CSV {label}: download error ({e})")
        return None
```

FIGURE 3.3 – Extrait de code d'extraction des fichiers CSV depuis Stooq

— **Tiingo (API REST) :** Cette source permet de mettre à jour le pipeline de manière dynamique via des requêtes HTTP, assurant ainsi la fraîcheur des données affichées sur le tableau de bord.

```python
def extract_api_to_dataframe():
    """Extract raw API data from Tiingo (all tickers) to a dataframe."""
    print("Extracting API (tiingo)...")
    try:
        if not TIINGO_API_KEY:
            print("API error: missing TIINGO_API_KEY")
            return None
        frames = []
        ingestion_ts = str(datetime.datetime.now())
        end_date = datetime.date.today().isoformat()
        for label in labels:
            try:
                api_url = f"{TIINGO_BASE_URL}/{label.lower()}/prices?startDate={TIINGO_START_DATE}&endDate={end_date}"
                response = requests.get(
                    api_url,
                    headers={"Authorization": f"Token {TIINGO_API_KEY}"},
                    timeout=60,
                )
            except Exception as req_err:
                print(f"API {label}: request error ({req_err})")
                continue
            if response.status_code != 200:
                print(f"API {label}: HTTP {response.status_code}")
                continue
            data = response.json()
            if not data:
                print(f"API {label}: empty")
                continue
            df = pd.DataFrame(data)
            df["ticker"] = label
            df["ingestion_timestamp"] = ingestion_ts
            df["source_name"] = "TIINGO_API"
            df["source_file"] = api_url
            frames.append(df)
```

FIGURE 3.4 – Extrait de code d'extraction via l'API REST Tiingo

3.7.2 Structure des données OHLCV

Chaque enregistrement représente une journée de cotation pour un actif donné et contient cinq informations : le prix d'ouverture de la séance, le niveau le plus haut atteint dans la journée, le niveau le plus bas, le prix de clôture et le volume total de titres échangés [24]. Ce format, universellement désigné sous l'acronyme OHLCV, constitue le standard dans le domaine des données boursières. En pratique, les deux sources utilisées Stooq et Tiingo ne nomment pas ces colonnes de façon identique, ce qui a constitué le premier défi de normalisation du pipeline.

TABLE 3.6 – Structure des données brutes en couche Bronze

Champ	**Type (Bronze)**	**Description**
Date	Text / String	Date de la séance boursière
Open / Close	Text / String	Prix d'ouverture et de clôture
High / Low	Text / String	Extrêmes de prix durant la séance
Volume	Text / String	Quantité de titres échangés

3.7.3 Qualité des données et défis

L'analyse exploratoire a mis en évidence plusieurs défis techniques à anticiper :

— **Typage :** Les valeurs stockées sous forme textuelle doivent être converties en types numériques (`float`, `integer`) pour permettre les calculs.

— **Doublons :** La fusion des sources CSV et API présente un risque de duplication des enregistrements, nécessitant un mécanisme de dédoublonnage.

— **Règles métier :** Des contrôles de cohérence des prix doivent être appliqués — par exemple, le prix de clôture doit nécessairement être compris entre le *High* et le *Low*.

3.7.4 Synthèse de l'exploration

Les données brutes sont stockées dans leur intégralité dans le schéma `bronze` de PostgreSQL. Cette approche préserve un historique immuable des données sources avant l'application des filtres de nettoyage, garantissant ainsi la traçabilité complète du pipeline.

3.8 Préparation des données

La phase de préparation a pour objectif de transformer les données brutes en données nettoyées et standardisées, prêtes à alimenter les étapes d'analyse. C'est à ce stade que la majorité des règles de qualité des données sont mises en œuvre.

3.8.1 Intégration et normalisation

Le premier défi consiste à fusionner des sources hétérogènes de manière cohérente. Pour ce faire, un processus de mappage a été développé afin de renommer les colonnes de chaque source selon un schéma standard commun :

— Renommage des champs : `Open/open` → `open_price`, `Volume/volume` → `volume`.

— Concaténation des DataFrames issus des sources CSV et API afin de constituer un jeu de données unifié.

3.8.2 Nettoyage et transformation

Le pipeline de nettoyage applique les étapes suivantes via la bibliothèque *Pandas* :

1. **Conversion des types :** Les colonnes de type `TEXT` sont converties en `FLOAT` pour les prix et en `DATETIME` pour les dates.
2. **Suppression des valeurs nulles :** Les lignes incomplètes sont éliminées via la méthode `dropna()` appliquée aux colonnes critiques.
3. **Dédoublonnage :** Les enregistrements redondants sont supprimés sur la base de la clé métier composite (`ticker`, `date`).

3.8.3 Validations métier (contraintes de données)

Des filtres propres au domaine financier sont appliqués afin d'écarter les données aberrantes :

$$\text{Filtre} = \begin{cases} \text{conserver} & \text{si } High \geq Low \text{ et } Volume \geq 0 \\ \text{rejeter} & \text{sinon} \end{cases} \tag{3.1}$$

3.8.4 Chargement (Couche Silver)

Les données résultantes sont persistées dans le schéma `silver` de PostgreSQL. Ce schéma garantit que toutes les analyses ultérieures — calculs de moyennes mobiles, drawdowns, volatilité — reposent sur des données dont l'intégrité a été rigoureusement vérifiée.

3.9 Modélisation des données (*Feature Engineering*)

Dans le contexte de ce pipeline de données, la phase de modélisation ne porte pas sur le Machine Learning, mais sur l'**ingénierie des caractéristiques** (*Feature Engineering*) et la **modélisation analytique en base de données**. Pour structurer ce flux transactionnel complexe, nous avons retenu l'**architecture Medallion** (Bronze, Silver, Gold), désormais reconnue comme un standard en Data Engineering.

3.9.1 Standardisation et nettoyage : la Couche Silver

La modélisation débute par la transformation des données brutes de la couche Bronze, fichiers CSV Stooq et données Tiingo, vers un modèle relationnel strict. Le script Silver réalise une harmonisation critique à plusieurs niveaux :

— **Alignement des schémas :** Fusion des données multisources par mappage des colonnes hétérogènes vers des identifiants standards unifiés (`date`, `open_price`, `close_price`, etc.).

— **Typage strict :** Conversion des valeurs financières en nombres flottants (`FLOAT`) via `pd.to_numeric`, et des volumes de transaction en entiers longs (`Int64`), éliminant ainsi les incohérences de types.
— **Filtrage d'intégrité boursière :** Application de règles métier pour assainir l'historique financier :
  — Suppression des journées de cotation où le prix le plus haut est inférieur au prix le plus bas (`high_price < low_price`) ;
  — Exclusion des volumes de transaction négatifs (`volume < 0`).
— **Dédoublonnage :** Suppression des enregistrements redondants sur la base du couple de clés métier (`ticker`, `date`).

3.9.2 Modélisation métier et *Feature Engineering* : la Couche Gold

La couche Gold constitue le cœur de la modélisation analytique. Elle enrichit les données assainies de la couche Silver par le calcul de nouveaux indicateurs clés de performance (KPIs), destinés à être consommés par le frontend (tableau de bord). Les agrégations et calculs en fenêtres glissantes (*Rolling Windows*) s'appuient sur la bibliothèque *Pandas* :

1. **Moyennes Mobiles Simples (SMA) :** Calcul de la SMA sur 12 et 24 jours, permettant de lisser la série temporelle et d'identifier la direction générale de la tendance boursière.
2. **Rendement (*Return*) :** Calcul dynamique de la variation en pourcentage d'une journée à l'autre (`pct_change() * 100`), exprimant la rentabilité brute de l'actif.
3. **Volatilité glissante (*Rolling Volatility*) :** Modélisation du risque financier par l'écart-type des rendements sur une fenêtre glissante de 12 périodes, servant d'indicateur direct de l'incertitude du marché.
4. **Drawdown (perte maximale latente) :** Le sommet historique cumulé du prix de clôture est d'abord calculé (*cummax*). Le drawdown représente ensuite la chute relative par rapport à ce pic :
$$Drawdown = \left(\frac{Close_Price}{Running_Peak_Close} \right) - 1 \tag{3.2}$$
5. **Amplitude journalière (*Range Percent*) :** Ratio entre la fluctuation intra-journalière ($High - Low$) et le prix de clôture, permettant de détecter les journées de forte indécision du marché.

3.9.3 Chargement et optimisation (base de données)

Une fois le *Feature Engineering* achevé, les données des couches `silver.ohlcv_cleaned` et `gold.ohlcv_kpis` sont injectées dans PostgreSQL. Afin de répondre aux exigences de performance sur plusieurs dizaines de milliers d'enregistrements temporels, une stratégie d'**insertion en masse** (*Bulk Upsert*) a été mise en place :

— Utilisation de l'instruction `COPY FROM STDIN`, combinée au buffer `io.StringIO()`, pour transférer les DataFrames directement en mémoire vers PostgreSQL.
— Recours à des tables temporaires couplées à une stratégie d'*upsert* avec `ON CONFLICT DO UPDATE`, garantissant l'idempotence totale du pipeline : en cas de ré-exécution, les agrégations sont mises à jour sans créer de doublons.

3.10 Évaluation

Cette phase du cycle CRISP-DM a pour objectif d'évaluer rigoureusement la qualité des modèles de données de la couche *Gold* avant leur déploiement opérationnel. À cette fin, un rapport d'audit exhaustif (*Data Profiling*) a été généré de manière automatisée via la bibliothèque `ydata-profiling`, outil de référence pour la génération de rapports statistiques détaillés incluant visualisations et métriques de qualité [25].

3.10.1 Profilage global et volumétrie

Le rapport HTML généré atteste de la robustesse de l'extraction et des transformations appliquées en amont. L'analyse porte sur un jeu de données final comprenant **54 209 enregistrements transactionnels quotidiens**, répartis sur **7 actifs financiers**. Cette volumétrie, couvrant plusieurs décennies d'historique agrégé, garantit que les algorithmes de prédiction et le tableau de bord frontend disposent d'une base statistiquement significative.

3.10.2 Audit de la qualité des données

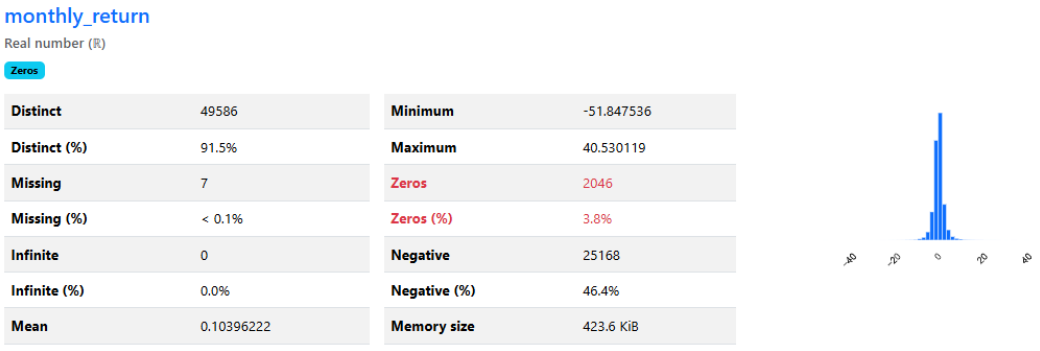
L'une des exigences majeures du projet réside dans la fiabilité des prix de marché. Les résultats de l'évaluation confirment que le pipeline a atteint ses objectifs de nettoyage :

- **Intégrité absolue des prix (OHLC) :** L'audit révèle **0 % d'anomalies boursières** (`invalid_ohlc_flag = 0`). Aucune journée ne présente un prix *High* inférieur au prix *Low*, garantissant la cohérence complète de la série temporelle.
- **Gestion maîtrisée des valeurs manquantes :** Le rapport identifie uniquement 7 valeurs manquantes pour la variable `monthly_return` et 14 pour `rolling_volatility_12m`. Ces lacunes sont mathématiquement attendues et justifiées : elles correspondent à l'absence d'historique suffisant lors des premiers jours de cotation de chacun des 7 actifs, soit la période de *warm-up* inhérente aux calculs de fenêtres glissantes.

3.10.3 Évaluation des KPIs et distributions statistiques

3.10.3.1 Distribution des rendements et volatilité

L'analyse de la distribution des rendements mensuels et de la volatilité met en évidence la nature hétéroscédastique — à variance non constante — caractéristique des marchés financiers.



monthly_return

Real number (ℝ)

Zeros

Distinct	49586	**Minimum**	-51.847536
Distinct (%)	91.5%	**Maximum**	40.530119
Missing	7	**Zeros**	2046
Missing (%)	< 0.1%	**Zeros (%)**	3.8%
Infinite	0	**Negative**	25168
Infinite (%)	0.0%	**Negative (%)**	46.4%
Mean	0.10396222	**Memory size**	423.6 KiB

FIGURE 3.5 – Distribution statistique des rendements mensuels extraite du rapport de *Data Profiling*

L'analyse visuelle du *Drawdown* permet également d'identifier clairement les périodes de crises économiques majeures, validant ainsi la formule mathématique implémentée lors de la phase de modélisation.

3.10.3.2 Analyse des corrélations

Afin de prévenir tout problème de multicolinéarité dans les modèles prédictifs, le rapport de profilage inclut une matrice de corrélation (Pearson/Spearman).

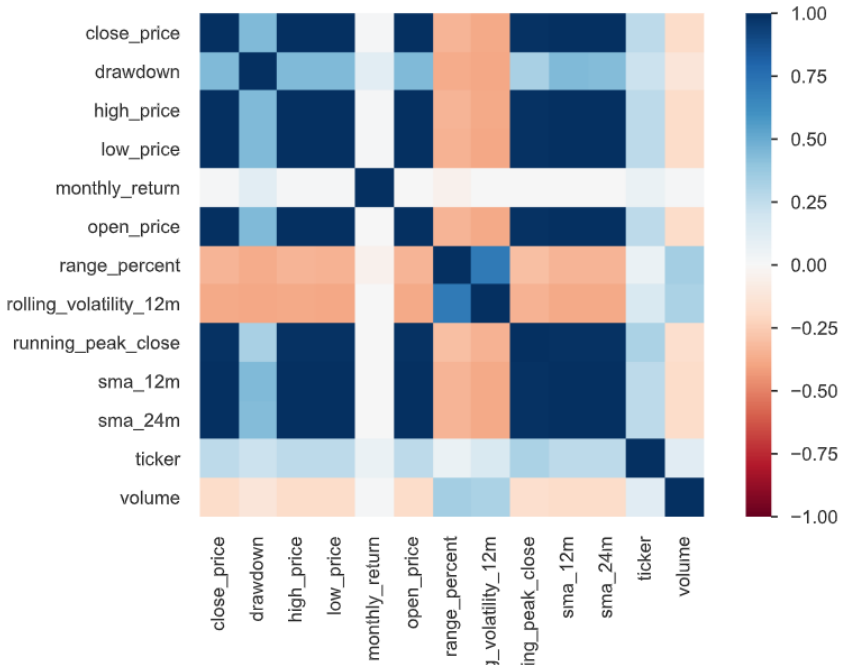


FIGURE 3.6 – Matrice de corrélation des métriques boursières et des moyennes mobiles (SMA)

L'analyse de cette matrice révèle, conformément aux attentes, une forte corrélation entre les prix historiques et les moyennes mobiles (SMA 12m et 24m). Elle confirme également l'indépendance de la volatilité par rapport à la direction absolue du prix, un comportement fondamental pour l'analyse des risques asymétriques.

3.10.4 Conclusion de l'évaluation

Cette phase d'audit a permis de valider de bout en bout la fiabilité du pipeline de données. La cohérence sémantique des types, l'absence de doublons temporels et l'exactitude calculatoire des variables métier autorisent le passage à la phase de déploiement et à la consommation des données par le frontend Angular.

3.11 Mise en œuvre et validation du tableau de bord

La dernière étape du cycle CRISP-DM consiste à exposer les données opérationnelles via une API REST sécurisée, puis à les restituer sous une forme intelligible au sein d'un tableau de bord de data-visualisation.

3.11.1 Construction et validation de l'API analytique

Afin d'assurer un découplage rigoureux entre les données stockées dans PostgreSQL (couche Gold) et l'interface utilisateur, une API REST a été développée avec **Flask** (Python). Cette API orchestre la distribution des indicateurs clés de performance via des requêtes optimisées.

Elle s'articule autour de plusieurs routes décisionnelles, notamment `/api/kpis` et `/api/kpis/summary`. Chaque route a été validée à l'aide de **Postman** afin de garantir la conformité de la structure JSON retournée et les temps de réponse attendus.

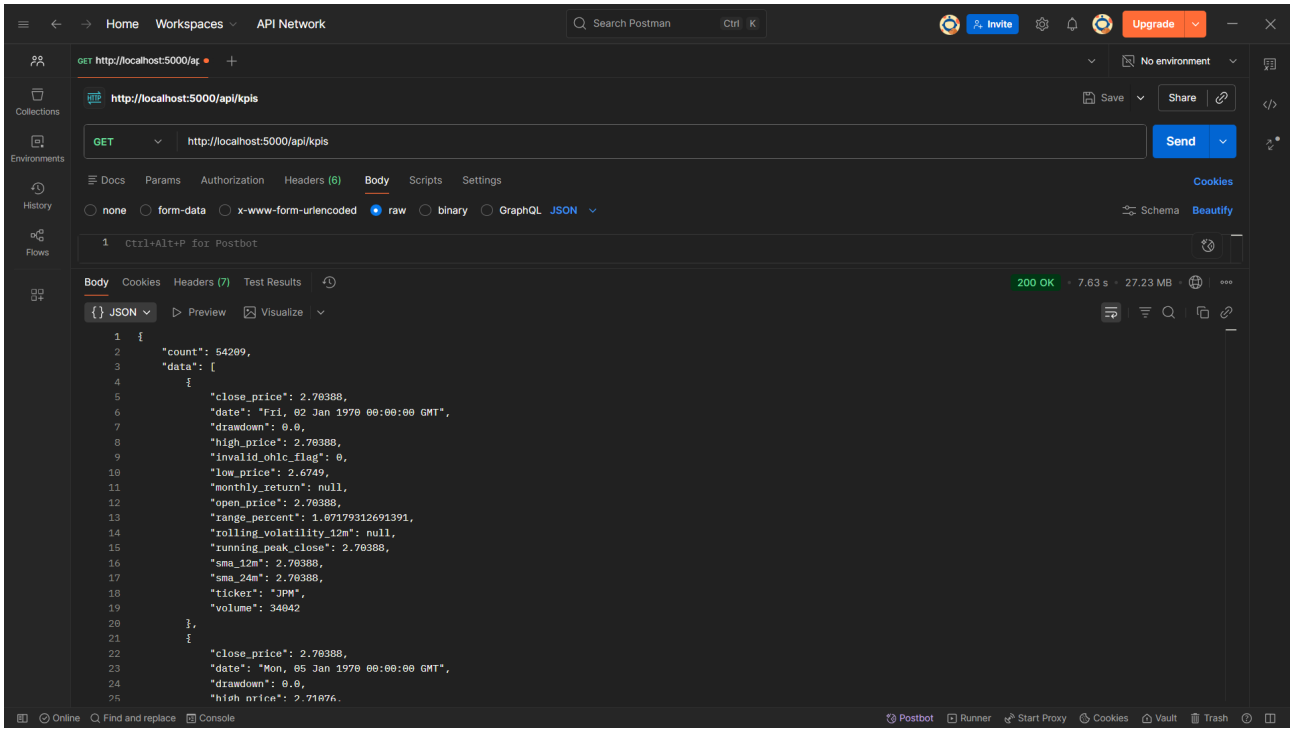


FIGURE 3.7 – Test Postman : récupération des KPIs financiers (SMA, Drawdown, Rendement) au format JSON

Les endpoints sont conçus pour être dynamiques : ils acceptent des paramètres de requête (`ticker`, `start_date`, `limit`), permettant au frontend de ne charger que la portion de données

strictement nécessaire à l'affichage. L'endpoint `summary` effectue, quant à lui, des agrégations à la volée (moyennes, minimums et maximums) sur la volatilité et les pertes maximales.

3.11.2 Développement de l'interface de visualisation (Frontend)

3.11.2.1 Vue d'ensemble (Overview)

Ce premier écran constitue le point d'entrée du tableau de bord. Il synthétise la santé globale de l'actif sélectionné en centralisant les grands indicateurs de performance — rendement moyen, volatilité historique et perte maximale enregistrée (*Max Drawdown*). Le graphique principal superpose l'évolution du prix de clôture et les volumes de transactions, permettant de vérifier rapidement si une hausse de prix est soutenue par un volume acheteur significatif.

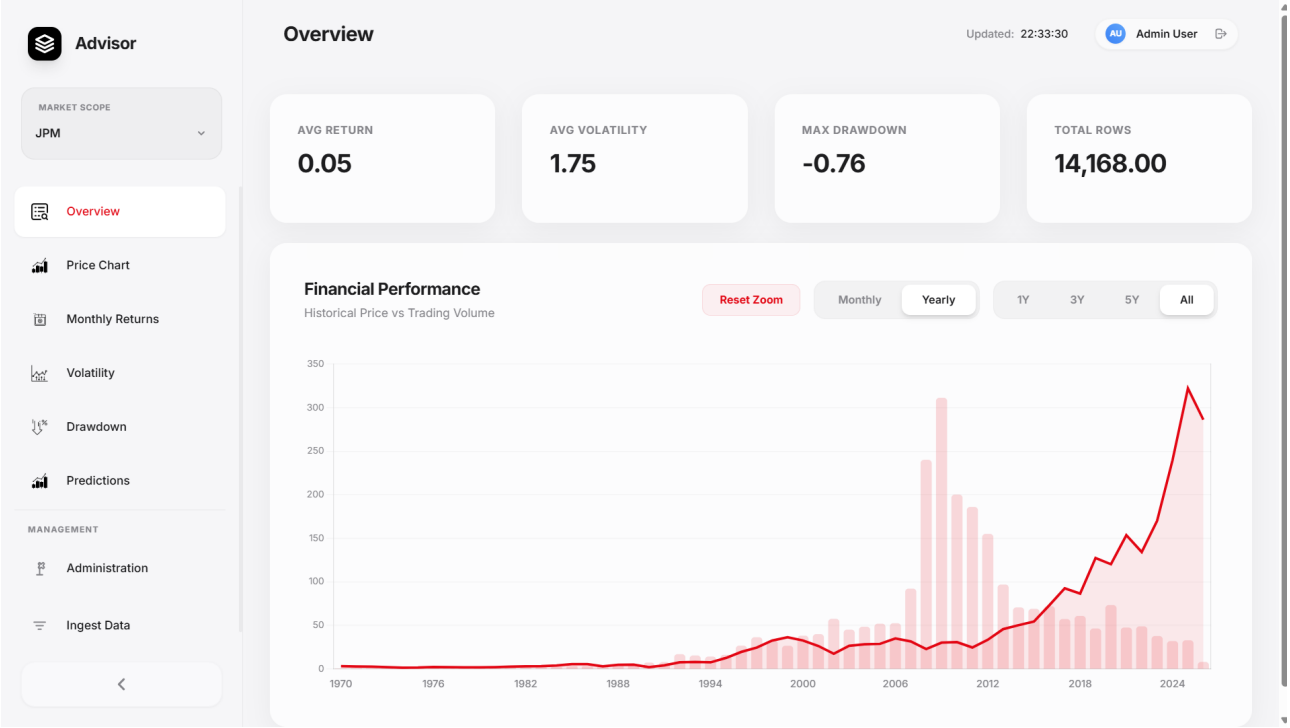


FIGURE 3.8 – Vue d'ensemble du tableau de bord présentant les indicateurs essentiels et la dynamique générale de l'actif

3.11.2.2 Analyse technique (Price Chart)

Ce module est conçu pour lisser les fluctuations quotidiennes du marché et faire ressortir la tendance de fond. Il superpose le prix de l'actif avec ses moyennes mobiles à 12 et 24 mois. Les croisements entre ces courbes constituent des signaux techniques classiques, permettant de déterminer si l'actif évolue dans une phase haussière (*Bull market*) ou baissière (*Bear market*), indépendamment de la volatilité journalière.

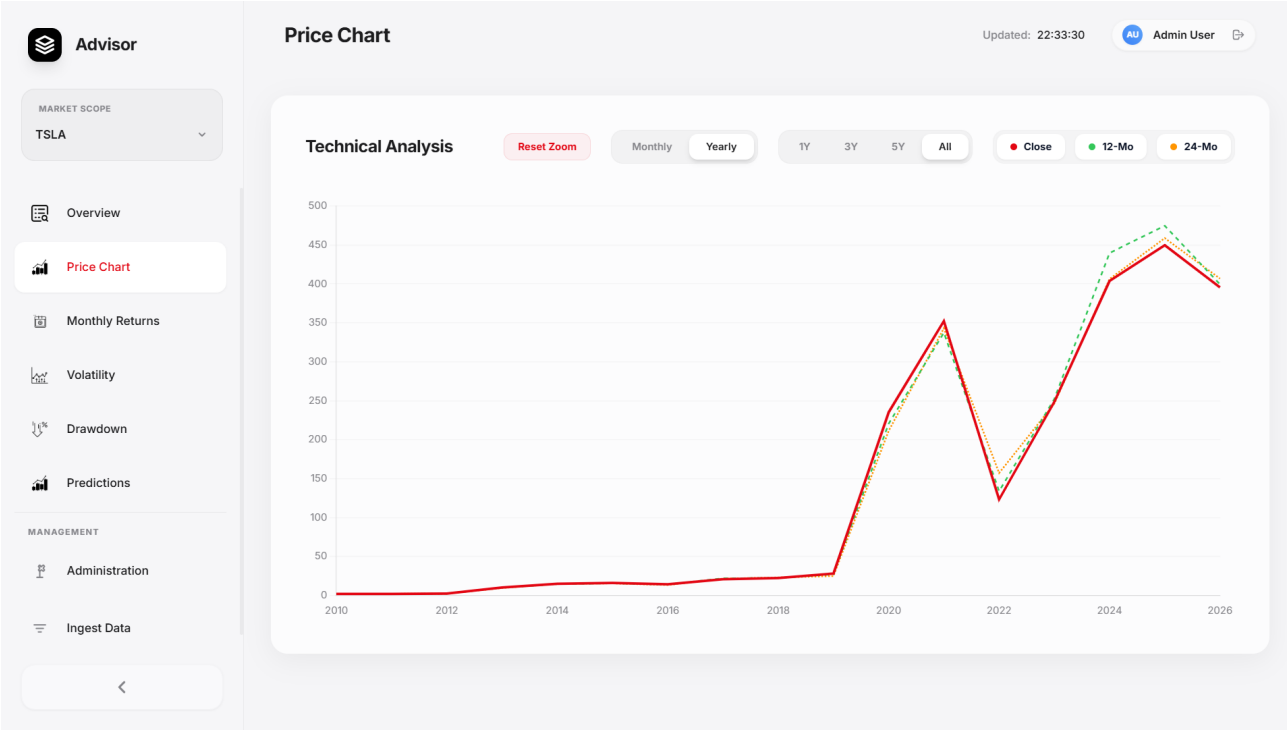


FIGURE 3.9 – Module d'analyse technique mettant en regard le prix de l'actif et ses moyennes mobiles pour identifier la tendance

3.11.2.3 Rendements mensuels (Monthly Returns)

Ce graphique en barres est dédié à l'analyse de la rentabilité période par période. Il isole le pourcentage de gain ou de perte pour chaque mois, permettant de repérer d'éventuels effets de saisonnalité ou d'identifier des séquences régulières de gains opposées à des épisodes de repli ponctuel.

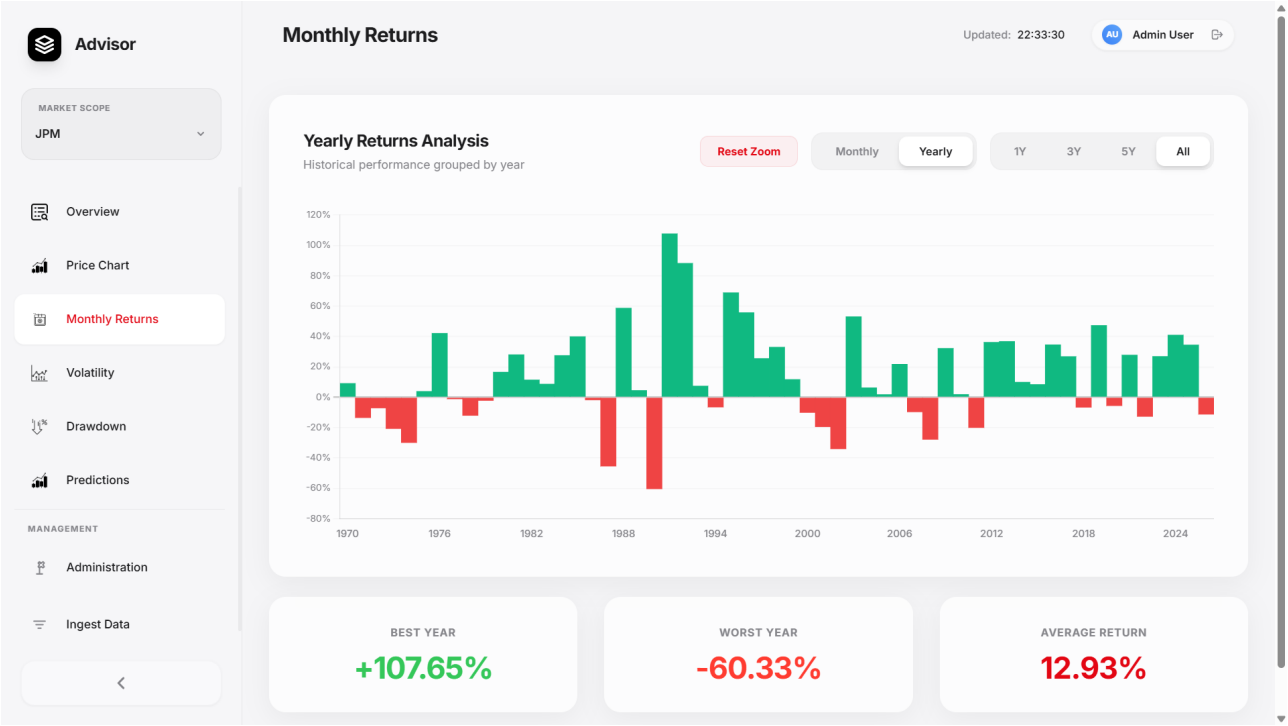


FIGURE 3.10 – Visualisation des rendements mensuels permettant d'identifier les phases de gains, les replis marquants et les effets de saisonnalité

3.11.2.4 Analyse du drawdown

Ce tableau de bord est entièrement dédié à l'évaluation du risque. Il compare le prix actuel de l'actif à son plus haut historique (*Peak*), tandis qu'un second graphique isole l'amplitude de la chute en pourcentage. Cet outil répond à une question fondamentale pour tout investisseur : quelle perte maximale aurait-il subi en achetant au moment le plus défavorable, et combien de temps aurait-il fallu pour récupérer sa mise initiale ?

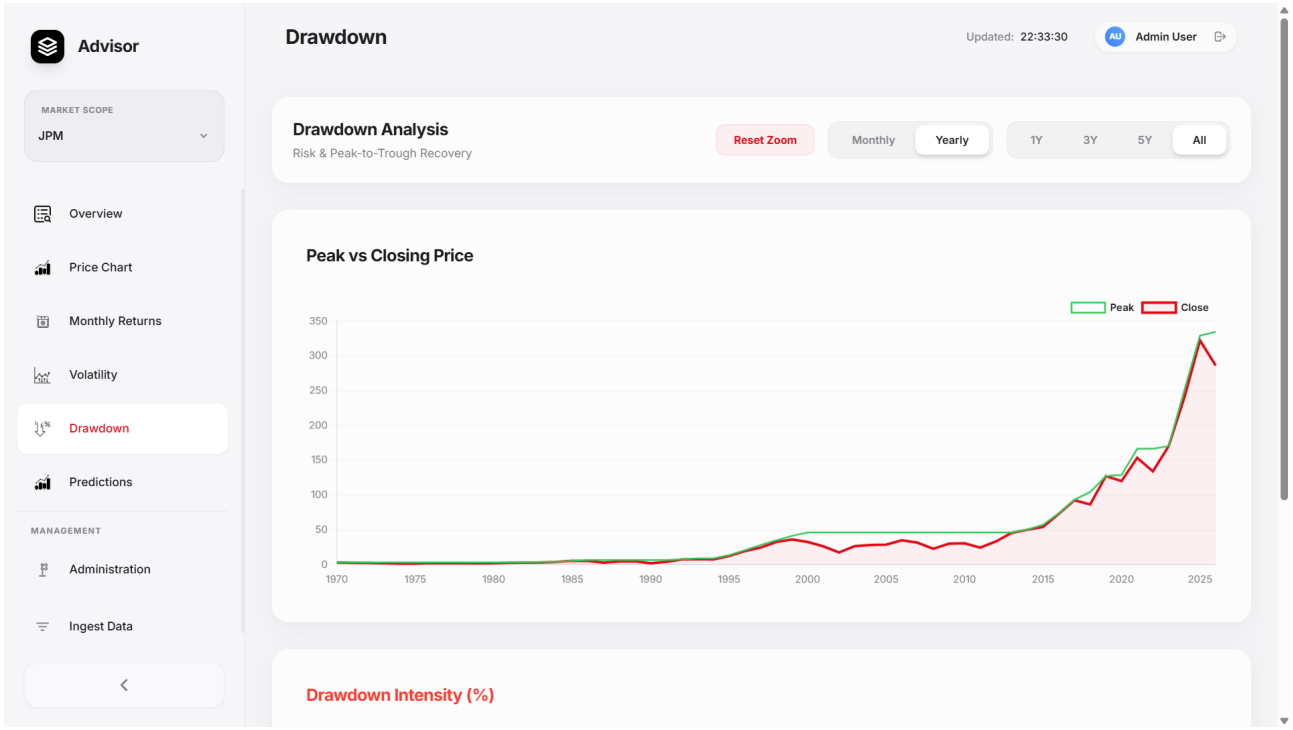


FIGURE 3.11 – Tableau de bord dédié au drawdown, illustrant l'ampleur des baisses et la capacité de récupération de l'actif

3.11.2.5 Volatilité (Volatility)

Ce module trace l'évolution de la volatilité sur 12 mois glissants, comparée à une ligne de référence correspondant à la volatilité moyenne historique. Une volatilité qui s'envole au-dessus de sa moyenne signale généralement un changement de régime de marché — qu'il s'agisse d'incertitude, de panique ou d'euphorie — et permet d'ajuster l'exposition au risque en conséquence.

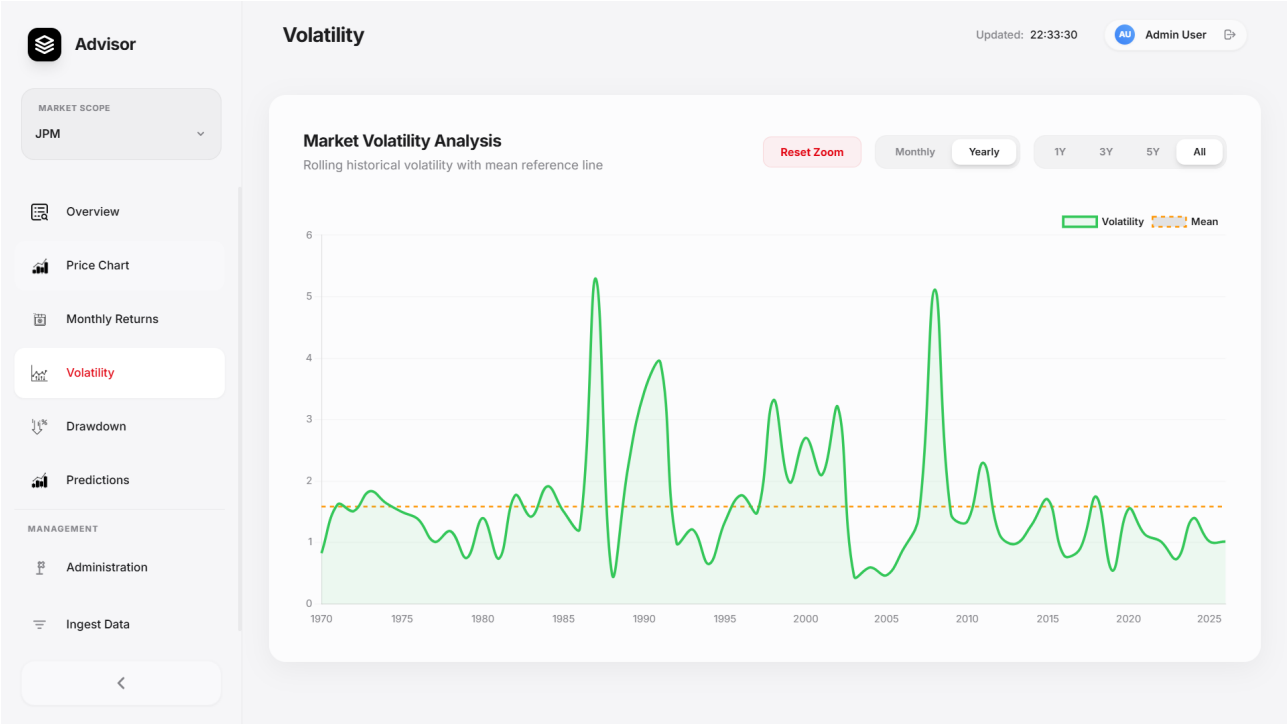


FIGURE 3.12 – Suivi de la volatilité glissante permettant de distinguer les périodes de calme relatif des épisodes de forte agitation du marché

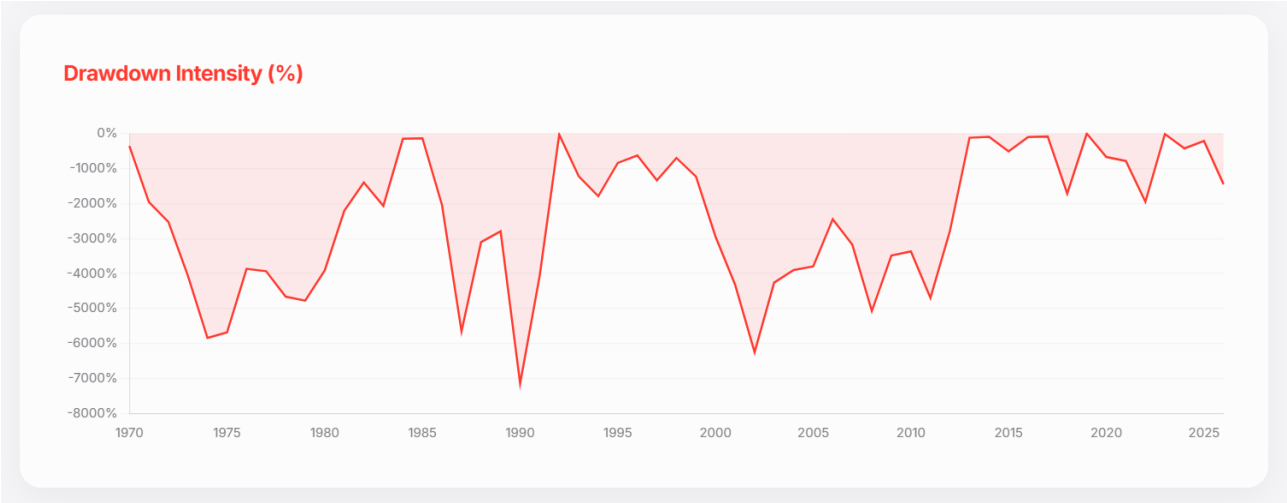


FIGURE 3.13 – Visualisation de l'intensité des épisodes de forte agitation du marché

3.12 Conclusion

Le pipeline ETL développé au cours de ce sprint traite désormais plus de 54 000 enregistrements quotidiens répartis sur sept actifs financiers, avec un taux d'anomalies nul et une latence maîtrisée grâce aux insertions en masse. L'architecture Medallion a pleinement démontré sa valeur : chaque couche remplit un rôle précis et clairement délimité, et les données exposées via l'API Flask sont prêtes à alimenter aussi bien le tableau de bord que le module de prédiction développé dans les sprints suivants. Le sprint 2 aborde un défi de nature différente : sécuriser et contrôler les accès à la plateforme.

4.1 Introduction

Un système d'analyse financière manipule des données sensibles et ne saurait être accessible sans un contrôle d'accès rigoureux. Ce deuxième sprint a donc été consacré à la mise en place de l'ensemble des mécanismes liés à la gestion des identités : inscription, authentification, réinitialisation du mot de passe et administration des comptes. Chaque fonctionnalité a été conçue en intégrant les contraintes de sécurité dès la phase de conception, et non comme une couche ajoutée après coup.

4.2 Backlog du Sprint 2

Le tableau ci-dessous présente le backlog du deuxième sprint, structuré en fonctionnalités, *user stories* et tâches associées.

Table 4.1 – Backlog du Sprint 2

ID	Fonctionnalité	User Story	Tâche
1	**S'inscrire**	En tant qu'utilisateur, je souhaite pouvoir créer un compte afin d'accéder au système.	Développer une page d'inscription permettant aux nouveaux utilisateurs de soumettre leurs informations personnelles pour enregistrement.
2	**Se connecter**	En tant qu'utilisateur, je souhaite pouvoir m'authentifier sur la plateforme.	Développer une page de connexion permettant à chaque utilisateur de s'authentifier à l'aide de son matricule et de son mot de passe.
3	**Mot de passe oublié**	En tant qu'utilisateur, je souhaite pouvoir réinitialiser mon mot de passe lorsque je l'ai oublié.	Développer un processus de réinitialisation en trois étapes : saisie de l'adresse électronique, réception d'un code de vérification, et définition d'un nouveau mot de passe après validation du code.
4	**Accepter de nouveaux utilisateurs**	En tant qu'administrateur, je souhaite pouvoir valider les demandes d'inscription afin d'accorder l'accès au système et de transmettre les identifiants par voie électronique.	Créer une interface d'administration permettant d'examiner les demandes en attente, d'approuver les nouveaux utilisateurs et de leur envoyer automatiquement leurs identifiants par e-mail.
5	**Modifier les informations des utilisateurs**	En tant qu'administrateur, je souhaite pouvoir mettre à jour les informations des utilisateurs afin de garantir l'exactitude des données dans le système.	Développer une interface permettant à l'administrateur de consulter et de modifier les attributs d'un compte utilisateur existant.
6	**Supprimer des comptes utilisateurs**	En tant qu'administrateur, je souhaite pouvoir supprimer des comptes utilisateurs devenus obsolètes.	Implémenter une fonctionnalité de suppression de comptes assortie d'une confirmation obligatoire avant toute action irréversible.

4.3 Spécifications fonctionnelles

Afin de définir précisément ce que le système doit être en mesure d'accomplir, cette section s'appuie sur un diagramme de cas d'utilisation représentant les interactions entre les différents

acteurs et la plateforme au cours de ce sprint. Chaque cas d'utilisation est ensuite décrit de manière structurée : préconditions, postconditions, scénario nominal et scénarios alternatifs. Ces éléments constituent une référence commune pour les phases de conception, de développement et de validation.

4.3.1 Diagramme de cas d'utilisation du Sprint 2

Le **Responsable Financier** est chargé de la gestion des comptes : il valide les demandes d'inscription, modifie les rôles et supprime les utilisateurs. Les actions de validation et de suppression déclenchent automatiquement l'envoi d'un e-mail de notification, modélisé par une relation d'inclusion. L'**Analyste Financier**, quant à lui, peut s'inscrire sur la plateforme et réinitialiser son mot de passe.

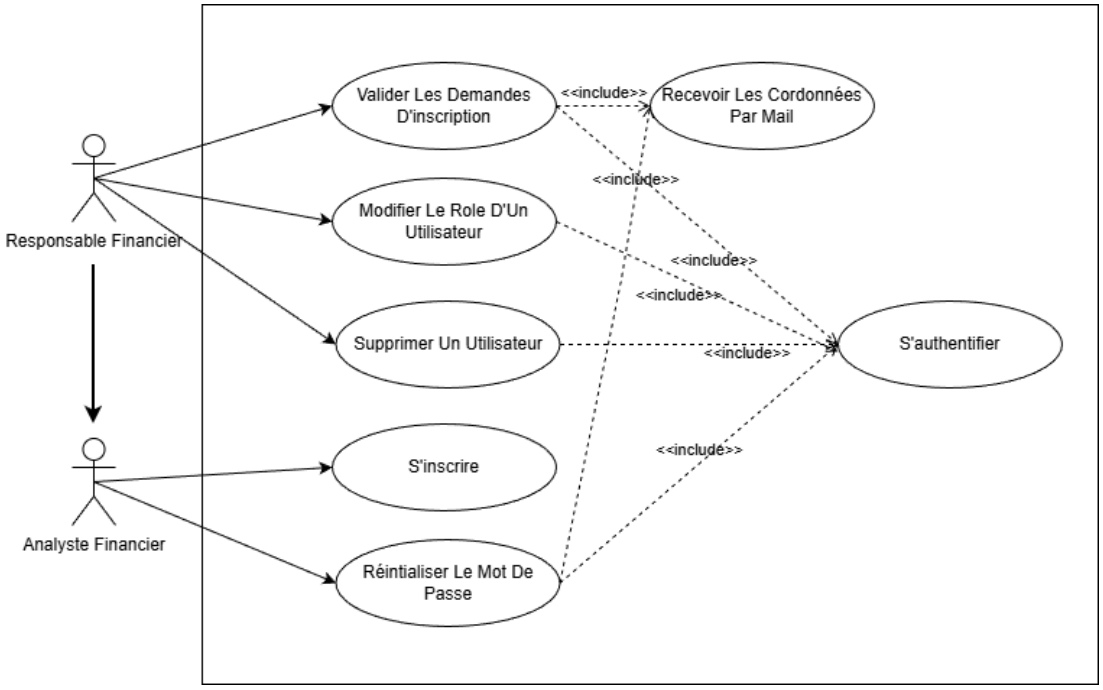


FIGURE 4.1 – Diagramme de cas d'utilisation du Sprint 2

4.3.2 Description détaillée des cas d'utilisation

4.3.2.1 Description Détaillée du cas d'utilisation "Valider les demandes d'inscription"

TABLE 4.2 – Description détaillée du cas d'utilisation « Valider les demandes d'inscription »

Titre	Cas d'utilisation : Valider les demandes d'inscription
Acteur principal	Responsable Financier
Description	Le Responsable Financier consulte les demandes en attente, analyse les informations reçues, puis valide ou rejette chaque demande.
Préconditions	1. L'utilisateur est authentifié. 2. Des demandes sont disponibles dans le système.
Postconditions	1. La demande est validée ou rejetée. 2. Le statut est mis à jour en base de données.
Scénario nominal	1. Accéder à la liste des demandes en attente. 2. Consulter les informations soumises. 3. Prendre une décision (validation ou rejet). 4. Enregistrer la décision dans le système.
Scénarios alternatifs	Données incomplètes ou invalides → rejet de la demande.
Règles de gestion	Chaque demande ne peut être traitée qu'une seule fois.

4.3.2.2 Description Détaillée du cas d'utilisation "Modifier le rôle d'un utilisateur"

TABLE 4.3 – Description détaillée du cas d'utilisation « Modifier le rôle d'un utilisateur »

Titre	Cas d'utilisation : Modifier le rôle d'un utilisateur
Acteur principal	Responsable Financier
Description	Le Responsable Financier sélectionne un utilisateur existant et lui attribue un nouveau rôle dans le système.
Préconditions	1. L'utilisateur est authentifié. 2. Le compte cible existe dans le système.
Postconditions	Le rôle de l'utilisateur est mis à jour.
Scénario nominal	1. Sélectionner un utilisateur dans la liste. 2. Choisir le nouveau rôle à attribuer. 3. Valider la modification.
Scénarios alternatifs	Rôle invalide ou non reconnu → refus de la modification.
Règles de gestion	La modification des rôles est réservée au Responsable Financier.

4.3.2.3 Description Détaillée du cas d'utilisation "Supprimer un utilisateur"

TABLE 4.4 – Description détaillée du cas d'utilisation « Supprimer un utilisateur »

Titre	Cas d'utilisation : Supprimer un utilisateur
Acteur principal	Responsable Financier
Description	Le Responsable Financier supprime définitivement un compte utilisateur après confirmation explicite de l'opération.
Préconditions	1. L'utilisateur est authentifié. 2. Le compte cible existe dans le système.
Postconditions	Le compte utilisateur est supprimé de la base de données.
Scénario nominal	1. Sélectionner l'utilisateur à supprimer. 2. Confirmer la suppression via la boîte de dialogue. 3. Exécuter la suppression.
Scénarios alternatifs	Annulation de l'opération par l'administrateur → aucune modification.
Règles de gestion	La suppression est définitive et irréversible.

4.3.2.4 Description Détaillée du cas d'utilisation "S'inscrire"

TABLE 4.5 – Description détaillée du cas d'utilisation « S'inscrire »

Titre	Cas d'utilisation : S'inscrire
Acteurs	Responsable Financier, Analyste Financier
Description	Le nouvel utilisateur soumet ses informations personnelles via le formulaire d'inscription afin de demander l'accès à la plateforme.
Préconditions	Aucun compte actif associé à l'adresse e-mail fournie.
Postconditions	La demande d'inscription est enregistrée avec le statut « en attente ».
Scénario nominal	1. Renseigner le formulaire d'inscription. 2. Soumettre la demande. 3. Recevoir un message de confirmation.
Scénarios alternatifs	Informations incomplètes ou invalides → affichage d'un message d'erreur.
Règles de gestion	La demande est soumise à validation par le Responsable Financier avant activation.

4.3.2.5 Description Détaillée du cas d'utilisation "Réinitialiser le mot de passe"

TABLE 4.6 – Description détaillée du cas d'utilisation « Réinitialiser le mot de passe »

Titre	Cas d'utilisation : Réinitialiser le mot de passe
Acteurs	Responsable Financier, Analyste Financier
Description	L'utilisateur initie une procédure de réinitialisation de son mot de passe via l'envoi d'un code de vérification par e-mail.
Préconditions	Un compte actif est associé à l'adresse e-mail saisie.
Postconditions	Un nouveau mot de passe est défini et actif.
Scénario nominal	1. Saisir l'adresse e-mail associée au compte. 2. Recevoir et saisir le code de vérification. 3. Définir et confirmer le nouveau mot de passe.
Scénarios alternatifs	Code expiré ou invalide → nouvelle demande de réinitialisation requise.
Règles de gestion	Le processus doit satisfaire aux exigences de sécurité définies (complexité du mot de passe, expiration du code).

4.4 Conception

Cette section regroupe les principaux artefacts de conception élaborés pour le Sprint 2. Les diagrammes de séquence permettent de suivre, étape par étape, les échanges entre les composants du système lors de l'exécution de chaque fonctionnalité — connexion, réinitialisation du mot de passe, validation des demandes et gestion des comptes. Ils constituent un support essentiel pour comprendre le déroulement dynamique des interactions et anticiper les cas limites à traiter.

4.4.1 Diagrammes de séquence du Sprint 2

4.4.1.1 Diagramme de séquence « S'authentifier »

La figure 4.2 illustre le déroulement du processus d'authentification, depuis la saisie des identifiants jusqu'à la redirection vers le tableau de bord.

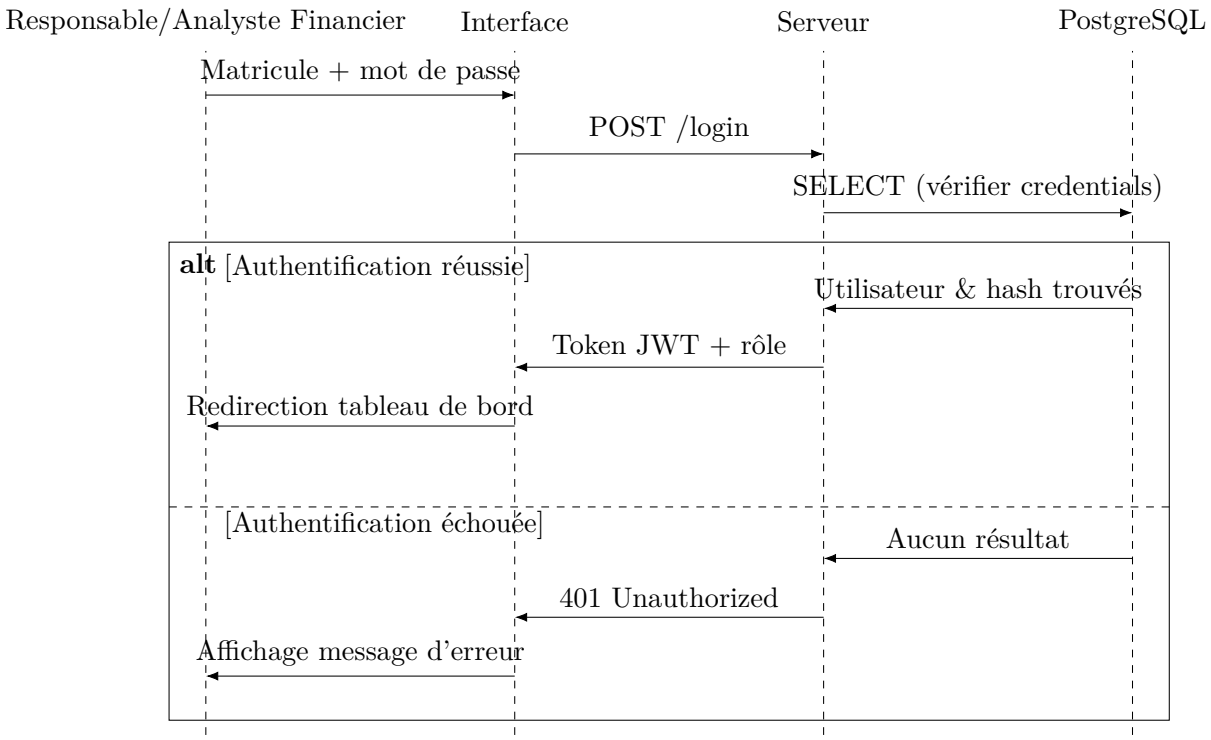


FIGURE 4.2 – Diagramme de séquence « S'authentifier »

4.4.1.2 Diagramme de séquence « Gérer les comptes »

La figure 4.3 offre une vue d'ensemble des interactions liées à la gestion des comptes utilisateurs, en couvrant la consultation de la liste ainsi que les actions de modification et de suppression.

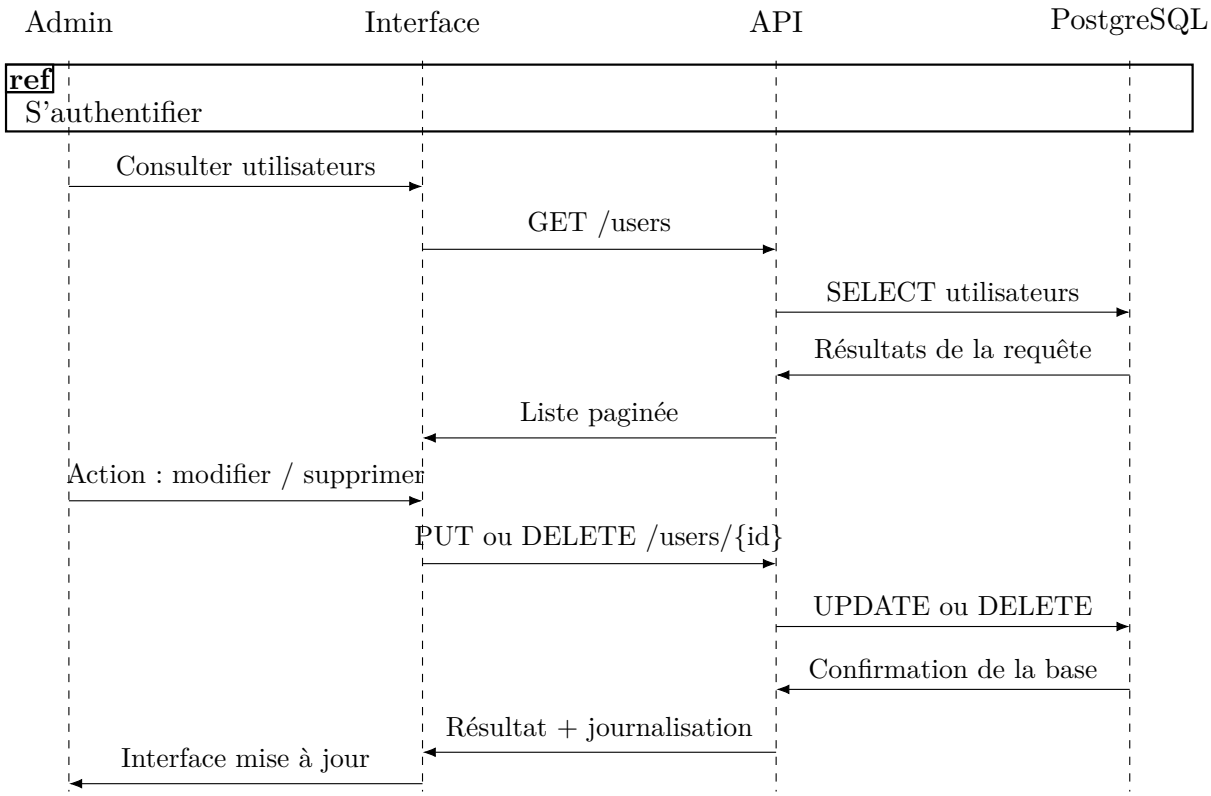


FIGURE 4.3 – Diagramme de séquence « Gérer les comptes »

4.5 Réalisation

Cette section présente les interfaces développées dans le cadre de ce sprint, ainsi que les choix techniques retenus pour assurer la protection des données des utilisateurs.

4.5.1 Interface d'inscription

L'interface d'inscription, présentée à la figure 4.4, a été conçue pour être aussi simple et intuitive que possible. Elle regroupe les informations essentielles — nom et prénom, adresse e-mail, numéro de téléphone et rôle dans l'organisation — au sein d'un formulaire épuré. Un bouton de soumission déclenche l'enregistrement de la demande, et des messages d'erreur explicites guident l'utilisateur en cas d'information manquante ou incorrectement saisie.

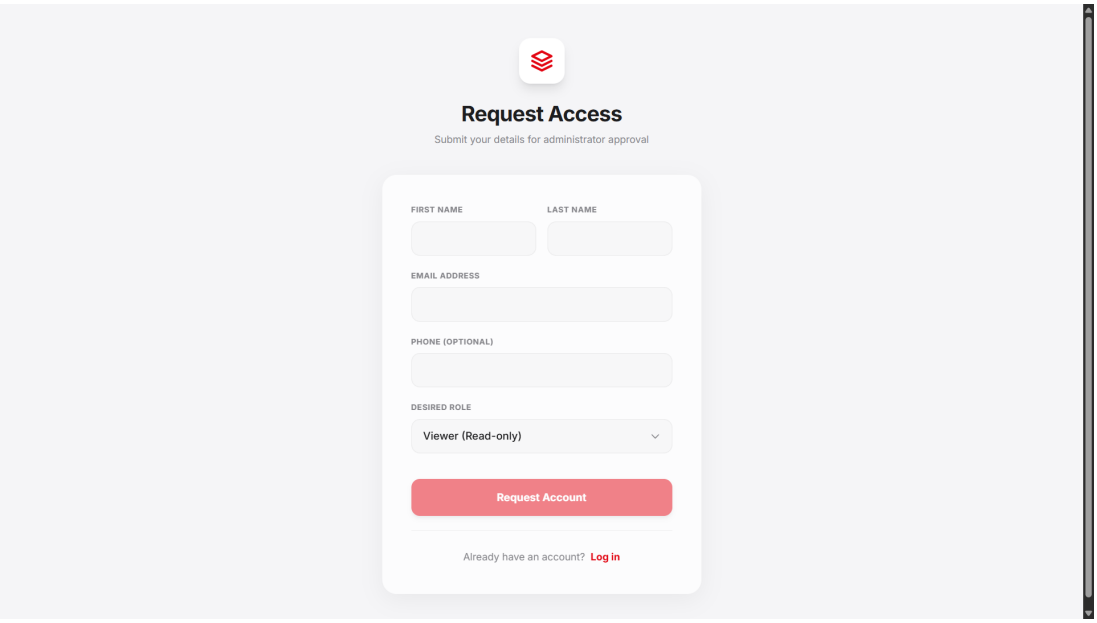


FIGURE 4.4 – Interface d'inscription

4.5.2 Interface d'authentification

La page de connexion, illustrée à la figure 4.5, donne accès à la plateforme à tout utilisateur disposant d'un compte actif. Deux champs suffisent : le matricule et le mot de passe. Un lien de réinitialisation est directement accessible sur cette page pour les utilisateurs ayant perdu leurs identifiants. L'ensemble des échanges est chiffré afin de garantir la confidentialité des données transmises.

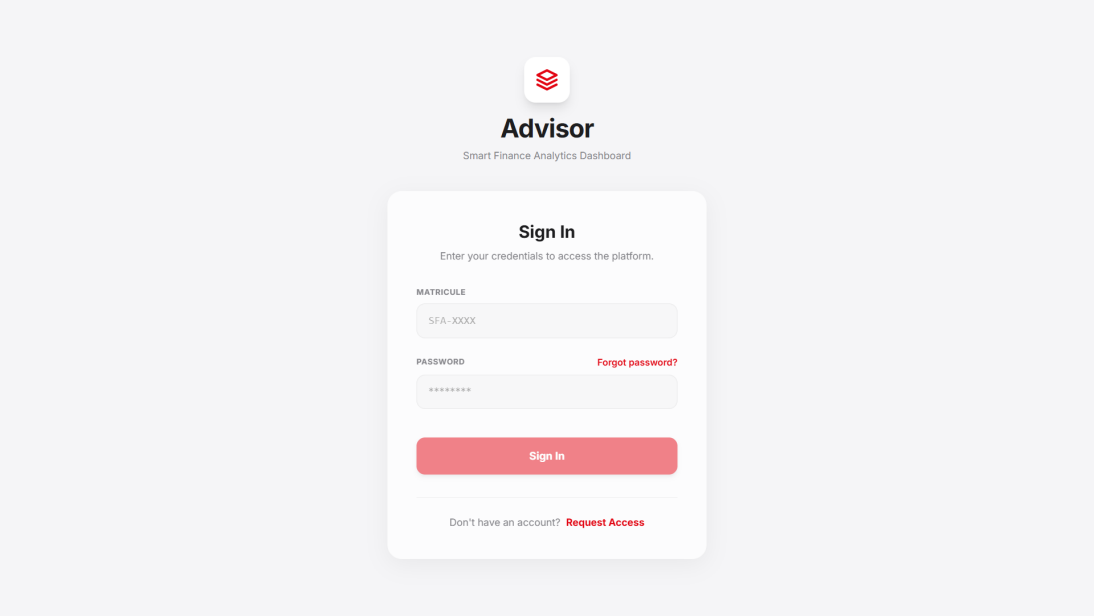


FIGURE 4.5 – Interface d'authentification

4.5.3 Interface de réinitialisation du mot de passe

La procédure de réinitialisation du mot de passe se déroule en trois étapes successives, conçues pour garantir que seule la personne légitime peut modifier ses identifiants.

4.5.3.1 Étape 1 — Saisie de l'adresse e-mail

L'utilisateur commence par indiquer l'adresse e-mail associée à son compte dans le formulaire dédié (figure 4.6).

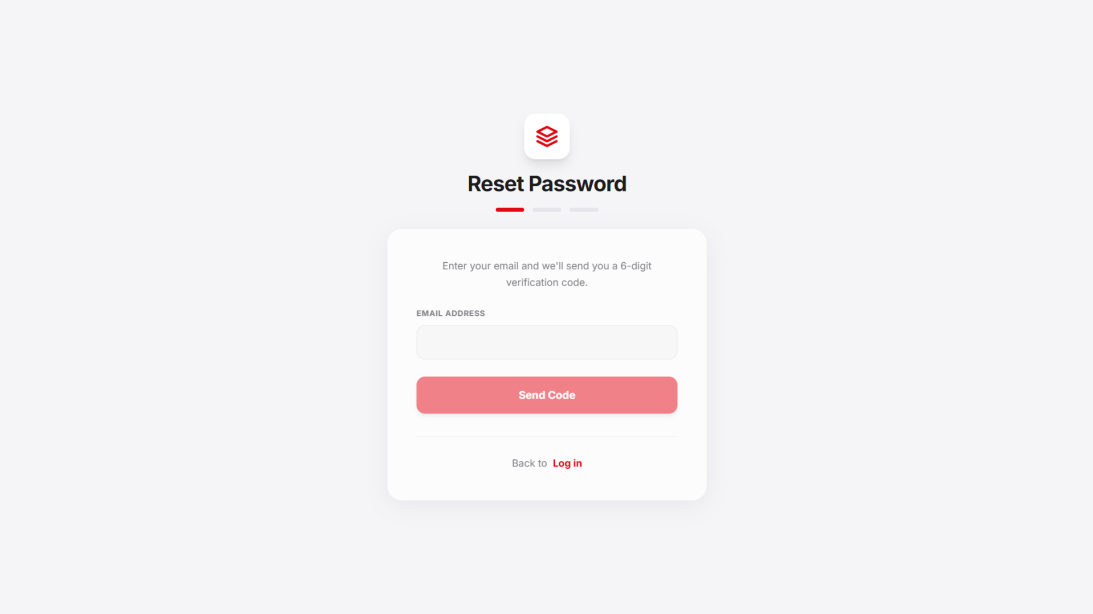


FIGURE 4.6 – Interface de réinitialisation — Saisie de l'adresse e-mail

4.5.3.2 Étape 2 — Validation du code de vérification

Si l'adresse est reconnue par le système, un code à six chiffres est envoyé à l'utilisateur par e-mail (figures 4.7 et 4.8). Ce code doit être saisi pour poursuivre la procédure.

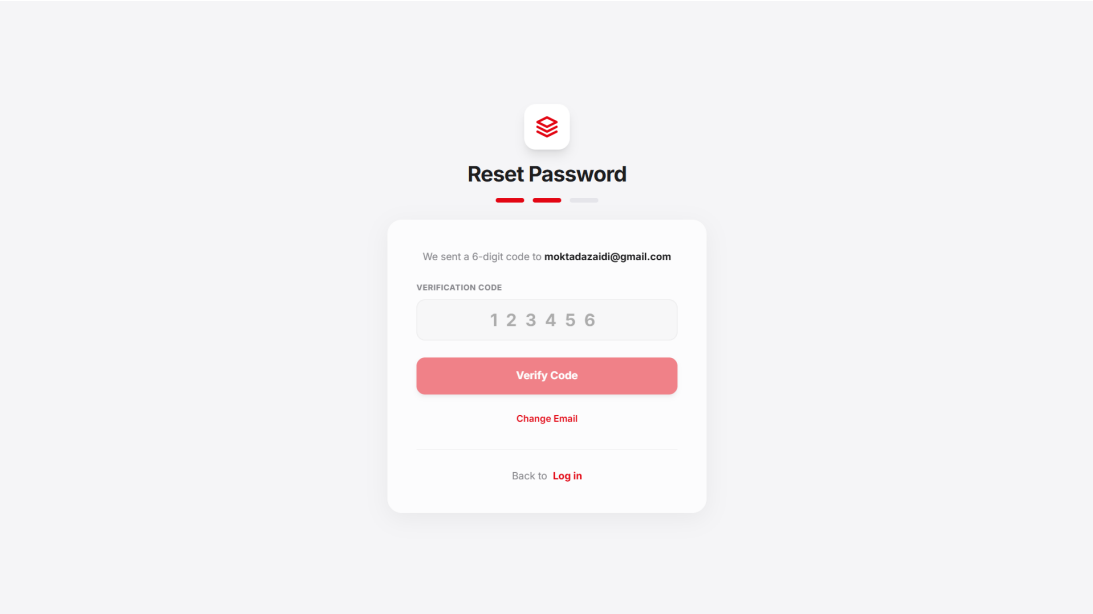


FIGURE 4.7 – Interface de réinitialisation — Envoi du code de vérification

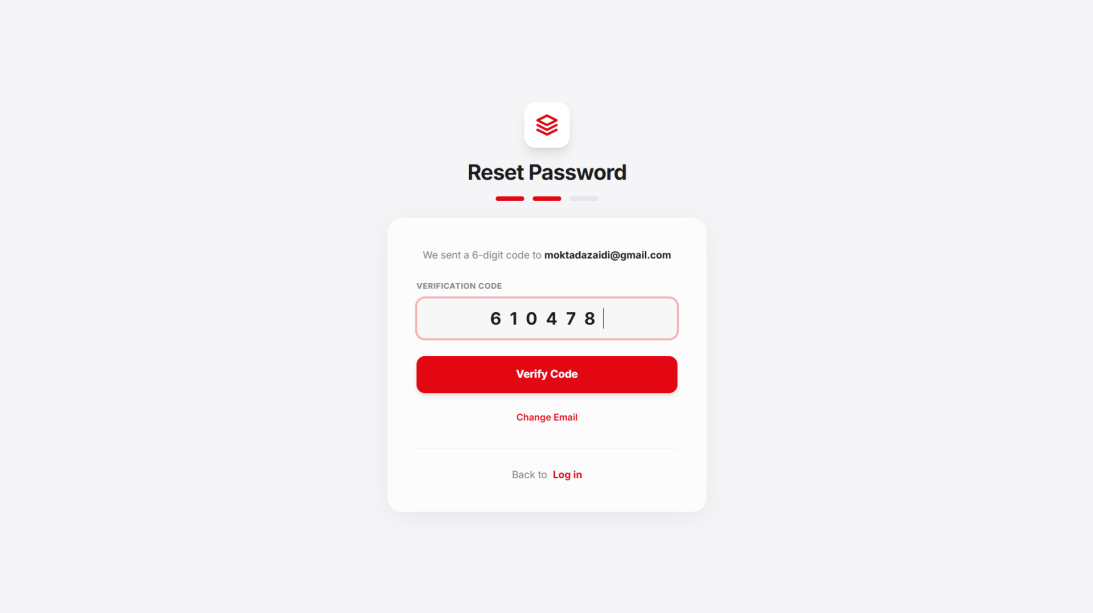


FIGURE 4.8 – Interface de réinitialisation — Saisie du code de vérification

4.5.3.3 Étape 3 — Définition du nouveau mot de passe

Une fois le code validé, l'utilisateur peut définir librement son nouveau mot de passe (figure 4.9). Ce mécanisme en trois étapes garantit que la procédure ne peut être initiée que par le titulaire légitime du compte, renforçant ainsi considérablement la sécurité globale du système.

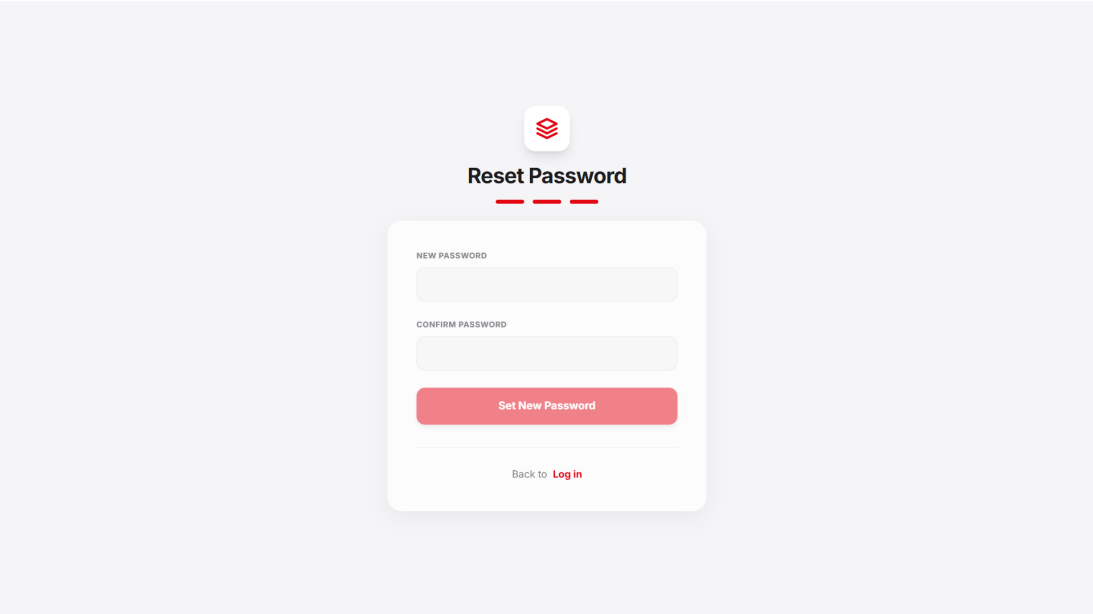


FIGURE 4.9 – Interface de réinitialisation — Définition du nouveau mot de passe

4.5.4 Interface de gestion des demandes en attente

L'administrateur dispose d'une interface dédiée (figure 4.10) dans laquelle s'affichent toutes les demandes d'inscription en cours de traitement. Pour chacune d'elles, il peut choisir de l'accepter ou de la rejeter. Lorsqu'une demande est acceptée, le système génère automatiquement les identifiants de l'utilisateur et les lui transmet par e-mail.

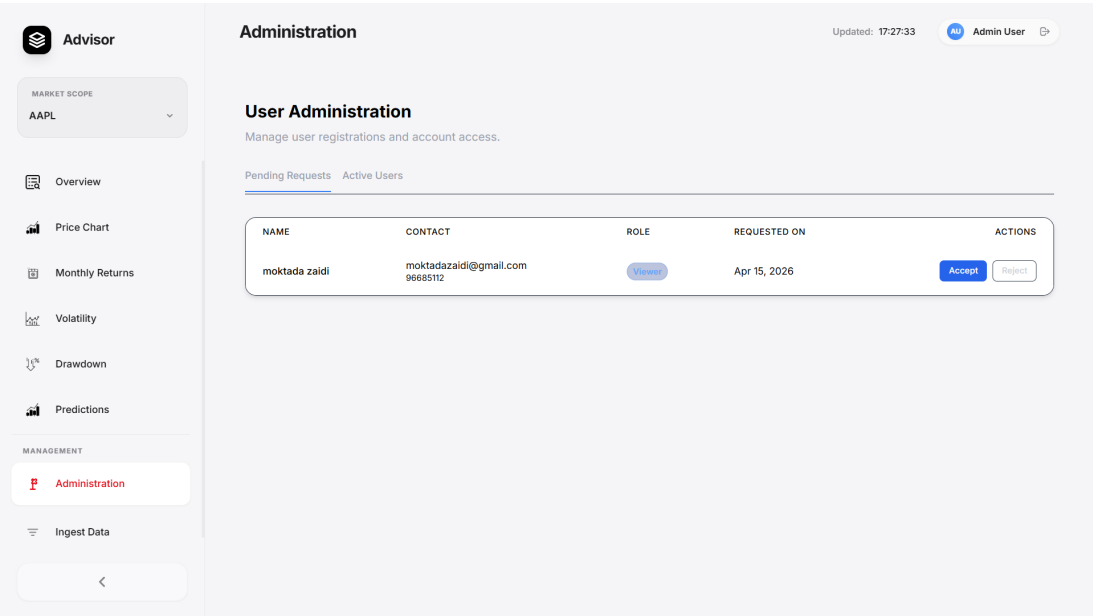


FIGURE 4.10 – Interface des demandes en attente

La figure 4.11 présente un exemple de l'e-mail reçu par l'utilisateur après validation de sa demande, contenant ses identifiants de connexion.

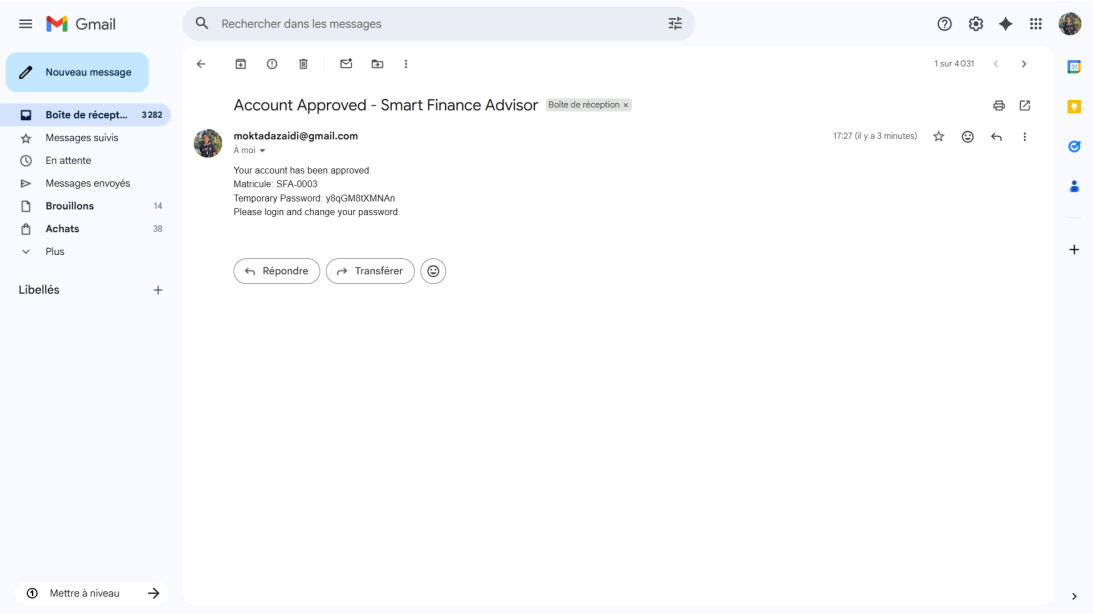


FIGURE 4.11 – E-mail de confirmation envoyé à l'utilisateur après validation de sa demande

4.5.5 Interface de gestion des utilisateurs actifs

Une fois les comptes activés, l'administrateur dispose d'une liste centralisée de l'ensemble des utilisateurs (figure 4.12). Depuis cet écran, deux actions sont disponibles :

— **Modifier les informations :** mettre à jour les données d'un compte existant ;

— **Supprimer un compte :** retirer du système tout compte devenu obsolète ou inutile.

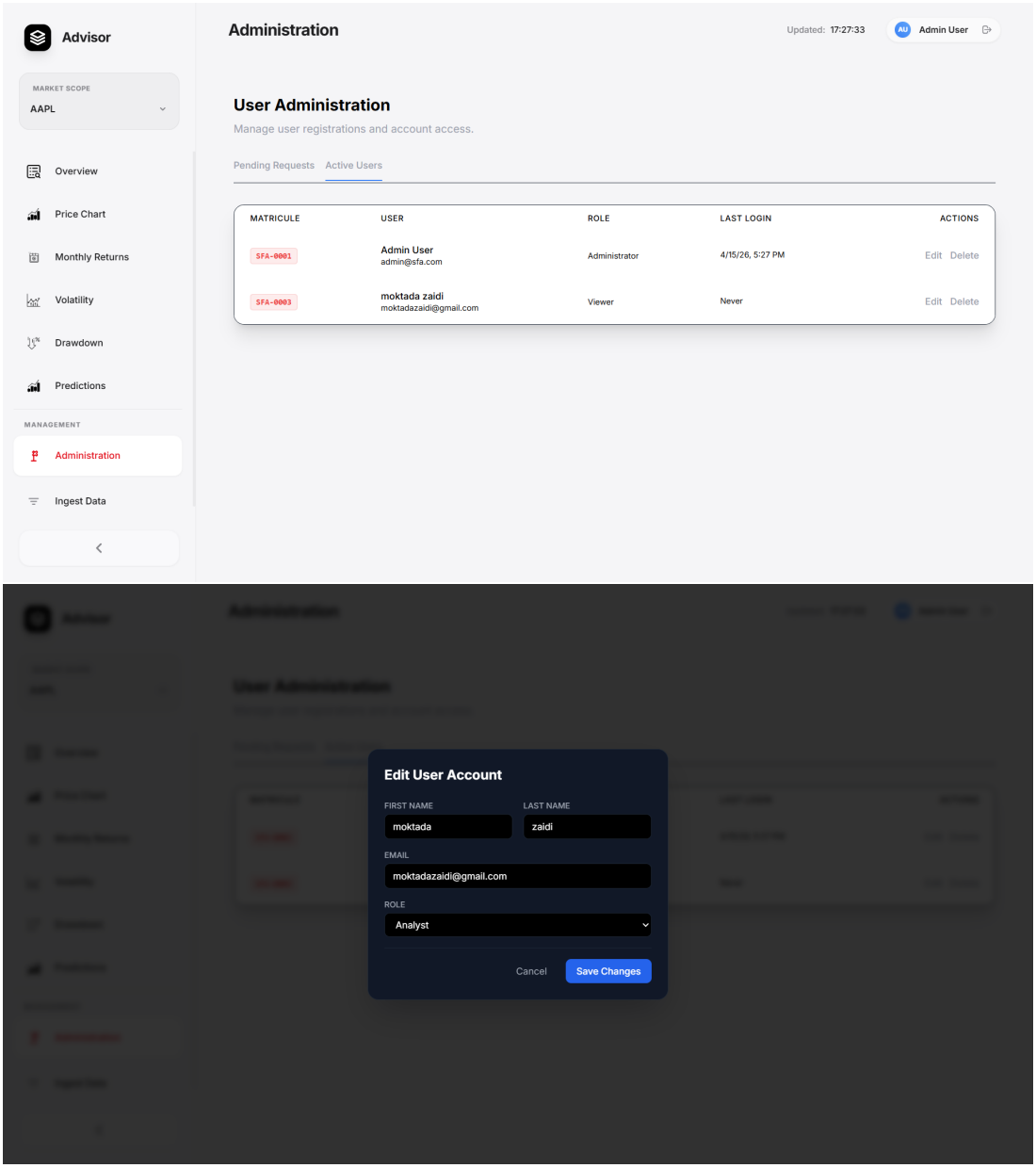


FIGURE 4.12 – Interface de gestion des utilisateurs actifs

4.6 Aspects de sécurité

La protection des mots de passe a été traitée comme une exigence fondamentale dès la phase de conception. Le choix s'est porté sur l'algorithme de hachage **bcrypt** [26], largement reconnu dans l'industrie pour sa robustesse face aux attaques par force brute. Son principal atout réside dans son facteur de coût adaptatif (*work factor*) : à mesure que les capacités de calcul augmentent, il suffit d'ajuster ce paramètre pour maintenir le même niveau de protection, sans modifier le reste du code.

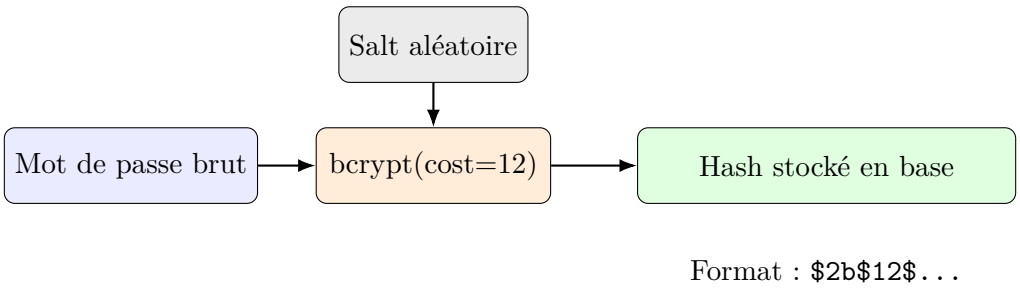


FIGURE 4.13 – Schéma du hachage des mots de passe avec bcrypt

Comme l'illustre la figure 4.13, chaque mot de passe est combiné à un sel aléatoire avant d'être haché. Cette approche rend inefficaces les attaques par dictionnaire et par tables précalculées (*rainbow tables*) : même dans l'hypothèse d'une compromission de la base de données, retrouver les mots de passe originaux à partir des hachages stockés demeurerait une tâche extrêmement coûteuse pour un attaquant.

4.7 Conclusion

Ce sprint livre un module de gestion des identités complet et sécurisé. Les mots de passe sont hachés avec bcrypt et un sel aléatoire, la procédure de réinitialisation repose sur un code à usage unique transmis par e-mail, et le Responsable Financier dispose d'un panneau d'administration centralisé pour gérer l'ensemble du cycle de vie des comptes. Ce socle de gestion des utilisateurs constituait un prérequis indispensable à l'exposition sécurisée des fonctionnalités analytiques développées dans les sprints suivants.

5.1 Introduction

Si l'analyse des données historiques constitue un fondement indispensable de toute plateforme de Business Intelligence, sa vraie valeur ajoutée réside dans sa capacité à anticiper. Ce troisième sprint marque précisément le passage de l'analyse rétrospective vers la prédiction : il s'agit de projeter l'évolution du prix de clôture d'actifs financiers à l'aide du modèle Prophet, puis de restituer ces prévisions de façon lisible et transparente à l'utilisateur, en y associant systématiquement leurs intervalles de confiance. Ce chapitre aborde successivement le choix algorithmique et sa justification, l'implémentation concrète du modèle, son évaluation quantitative, et enfin son intégration au sein de l'interface Angular.

5.2 Backlog Du Sprint 3

Le tableau 5.1 présente l'ensemble des fonctionnalités retenues pour ce sprint, sous la forme de *user stories* accompagnées des tâches techniques associées.

Table 5.1 – Backlog du Sprint – Module de Prédiction

ID	Fonctionnalité	User Story	Tâche
1	**Modélisation prédictive**	En tant qu'analyste, je souhaite intégrer un algorithme d'anticipation temporelle afin de projeter les données futures.	Développer un modèle basé sur l'algorithme Prophet pour analyser les données historiques et générer des prévisions accompagnées de leurs intervalles de confiance.
2	**Évaluation de fiabilité**	En tant qu'utilisateur, je souhaite connaître le taux de fiabilité du modèle de prédiction afin d'évaluer les risques associés.	Mettre en place un système de calcul de métriques d'erreur (MAE, RMSE, sMAPE) permettant d'établir un score de précision global.
3	**Visualisation graphique**	En tant qu'utilisateur, je souhaite visualiser les tendances futures sur un graphique afin de faciliter mon analyse globale.	Développer une interface intégrant un graphique interactif affichant simultanément l'historique, la prévision et ses bornes d'incertitude.
4	**Configuration d'horizon**	En tant qu'utilisateur, je souhaite pouvoir configurer la période prédictive afin d'étudier des tendances à court, moyen ou long terme.	Mettre en place des boutons de sélection dans l'interface permettant de modifier la fréquence et l'intervalle temporel de prédiction.

5.3 Spécification Fonctionnelle

Pour délimiter précisément le périmètre fonctionnel du système, cette section s'appuie sur un diagramme de cas d'utilisation qui modélise les interactions entre les différents acteurs et la plateforme au cours de ce sprint. Chaque cas d'utilisation est ensuite décrit de façon

structurée : on y précise les conditions nécessaires à son déclenchement, le résultat attendu en sortie, ainsi que le déroulement dans les situations nominales et les scénarios d'exception. Ces éléments servent de référence commune pour les phases de conception, de développement et de validation.

5.3.1 Diagramme de Cas d'Utilisation du Sprint 3

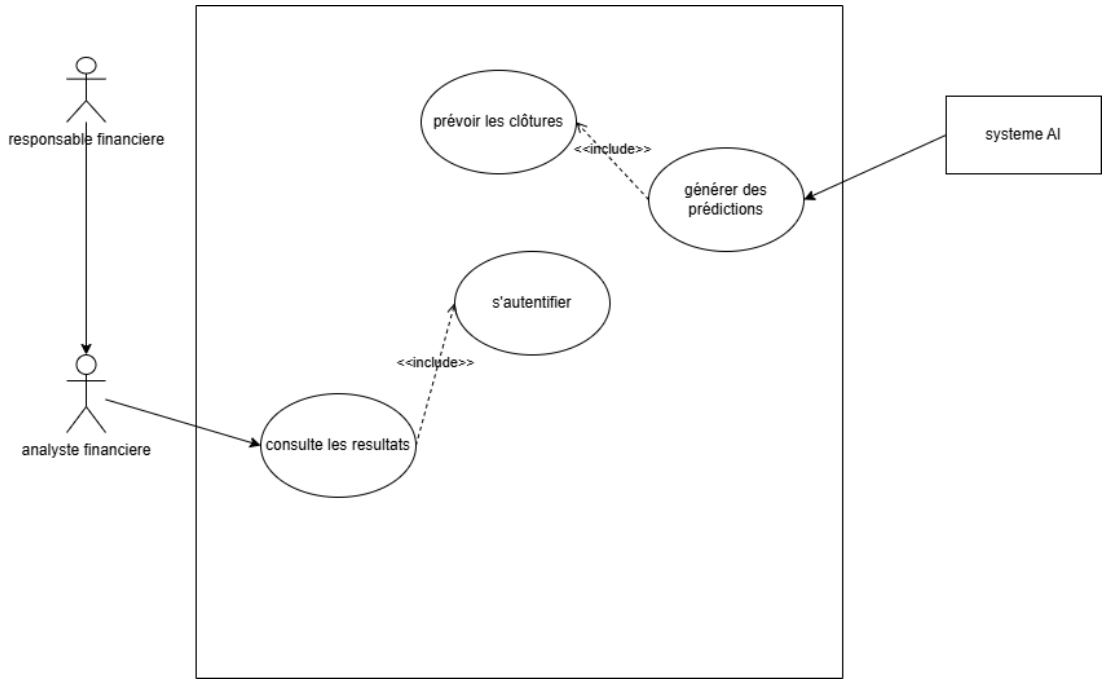


FIGURE 5.1 – Diagramme de cas d'utilisation du Sprint 3

5.3.2 Description Détaillée des Cas d'Utilisation

5.3.2.1 Description Détaillé du cas d'utilisation "Générer des prédictions"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation « Générer des prédictions ».

TABLE 5.2 – Description détaillée du cas d'utilisation « Générer des prédictions »

Titre	Cas d'utilisation : Générer des prédictions
Acteur principal	Système IA
Description	Le Système IA déclenche automatiquement la génération de prédictions financières à partir des données collectées, en appliquant les modèles d'apprentissage configurés.
Préconditions	1. Les données financières ont été collectées et normalisées. 2. Le modèle prédictif est chargé et opérationnel.
Postconditions	1. Les prédictions sont générées et enregistrées. 2. Les résultats sont disponibles pour consultation.
Scénario nominal	1. Le système détecte le déclencheur (planification ou nouvelles données). 2. Il charge le modèle prédictif en mémoire. 3. Il valide le modèle sur un jeu de données de test. 4. Il calcule les prévisions de clôtures financières. 5. Il enregistre les résultats et les marque comme disponibles.
Scénarios alternatifs	Score de validation insuffisant → suspension de la génération et émission d'une alerte.
Règles de gestion	Une seule génération de prédictions par cycle ; les résultats non validés ne sont jamais publiés.

5.3.2.2 Description Détaillé du cas d'utilisation "Prévoir les clôtures"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation « Prévoir les clôtures ».

TABLE 5.3 – Description détaillée du cas d'utilisation « Prévoir les clôtures »

Titre	Cas d'utilisation : Prévoir les clôtures
Acteur principal	Système IA
Description	Cas inclus dans UC-01. Le système calcule les valeurs prévisionnelles des clôtures financières pour les échéances à venir, sur la base des tendances détectées dans les données historiques.
Préconditions	1. Le modèle prédictif a été validé. 2. Les données historiques sont disponibles.
Postconditions	1. Les clôtures prévisionnelles sont calculées. 2. Les valeurs sont intégrées aux résultats de prédiction.
Scénario nominal	1. Le système récupère les séries temporelles financières. 2. Il applique le modèle aux données d'entrée. 3. Il produit les estimations pour chaque horizon temporel. 4. Les prévisions sont associées à un indice de confiance.
Scénarios alternatifs	Données insuffisantes pour un horizon → prévision partielle avec avertissement.
Règles de gestion	Chaque prévision est systématiquement accompagnée d'un intervalle de confiance calculé automatiquement.

5.3.2.3 Description Détaillé du cas d'utilisation "Collecter les données financières"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation « Collecter les données financières ».

TABLE 5.4 – Description détaillée du cas d'utilisation « Collecter les données financières »

Titre	Cas d'utilisation : Collecter les données financières
Acteur principal	Système IA
Description	Le Système IA interroge les sources de données configurées, nettoie et normalise les informations récupérées, puis les persiste dans le référentiel central en vue de leur exploitation prédictive.
Préconditions	1. Les sources de données sont accessibles et correctement configurées. 2. Le référentiel central est disponible en écriture.
Postconditions	1. Les données sont normalisées et stockées avec horodatage. 2. Un journal de collecte est produit et archivé.
Scénario nominal	1. Le système initie la collecte selon le calendrier défini. 2. Il interroge les APIs, flux et bases de données configurés. 3. Il applique les règles de nettoyage et de validation. 4. Il persiste les données validées avec traçabilité complète. 5. Il génère le journal récapitulatif de la collecte.
Scénarios alternatifs	Source indisponible → réessai automatique ; en cas d'échec persistant, une alerte est émise et la collecte s'effectue partiellement.
Règles de gestion	Toute donnée collectée est horodatée et traçable ; les données rejetées sont consignées dans le journal d'erreurs.

5.3.2.4 Description Détaillé du cas d'utilisation "Consulter les résultats"

Le tableau ci-dessous présente la description détaillée du cas d'utilisation « Consulter les résultats ».

TABLE 5.5 – Description détaillée du cas d'utilisation « Consulter les résultats »

Titre	Cas d'utilisation : Consulter les résultats
Acteur principal	Responsable Financier, Analyste Financier
Description	Après authentification, le Responsable Financier et l'Analyste Financier accèdent aux résultats de prédiction produits par le Système IA afin d'orienter leurs décisions stratégiques.
Préconditions	1. L'utilisateur est authentifié. 2. Des prédictions validées sont disponibles dans le système.
Postconditions	1. L'utilisateur a consulté les résultats souhaités. 2. La session est maintenue ou expirée selon la durée d'inactivité.
Scénario nominal	1. L'utilisateur accède à l'interface du système. 2. Il s'authentifie (inclusion : S'authentifier). 3. Il accède au tableau de bord des prédictions. 4. Il filtre les résultats par période, indicateur ou niveau de confiance. 5. Il consulte les tendances et les indicateurs clés.
Scénarios alternatifs	Authentification échouée → nouvelle tentative autorisée ; compte verrouillé après dépassement du seuil configuré.
Règles de gestion	L'accès est strictement conditionné à une authentification valide ; la session expire automatiquement après une période d'inactivité.

5.4 Conception

5.4.1 Diagrammes de Séquences

5.4.1.1 Diagrammes de Séquence "Génerer Des Prédictions"

La figure ci-dessus présente le diagramme de sequence "Generer des prédictions"

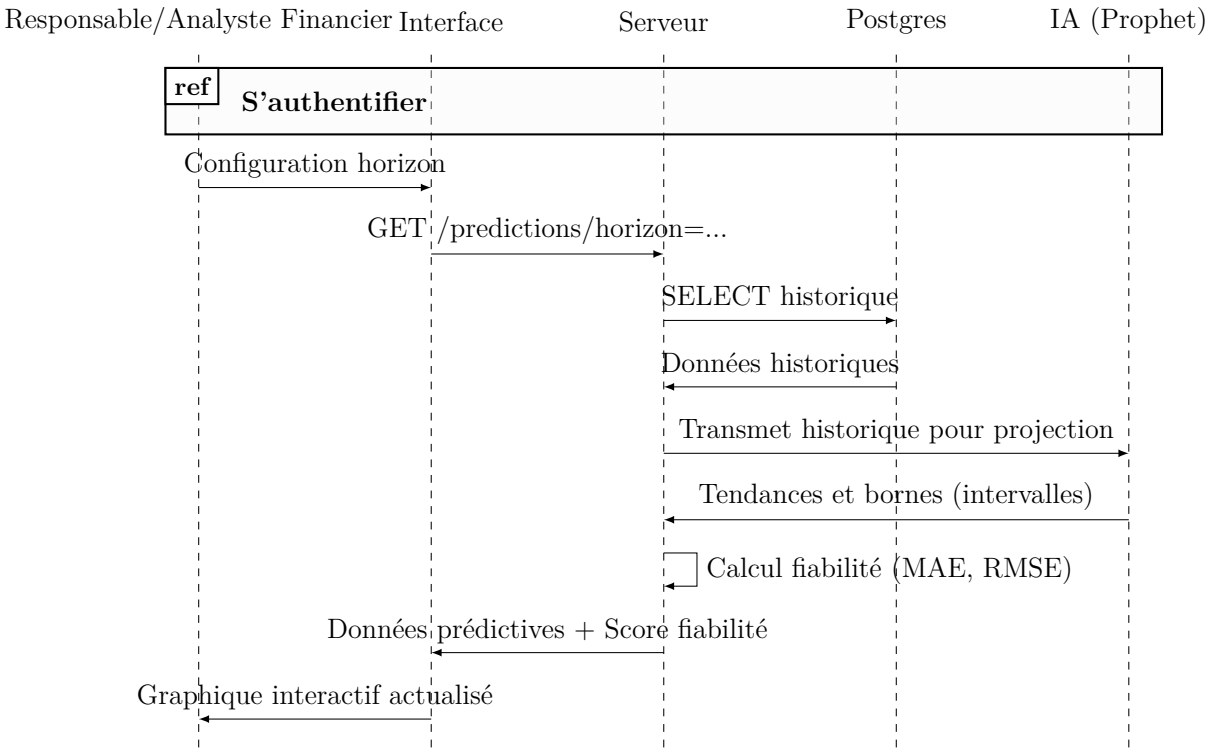


FIGURE 5.2 – Diagramme de séquence « Génerer des prédictions »

5.5 Définition et concept du Machine Learning

Dans une plateforme de Business Intelligence classique, l'objectif se limite généralement à représenter ce qui s'est passé : des graphiques, des tendances historiques, des comparaisons entre périodes. C'est une base nécessaire, mais insuffisante pour répondre à la question centrale qu'un analyste financier se pose au quotidien : qu'est-ce qui va se passer demain ? C'est précisément là que le machine learning prend tout son sens. Dans le cadre de SmartFinance Advisor, cela s'est traduit par l'intégration de Prophet directement dans le tableau de bord, avec des prévisions de prix associées à leurs intervalles de confiance[27].

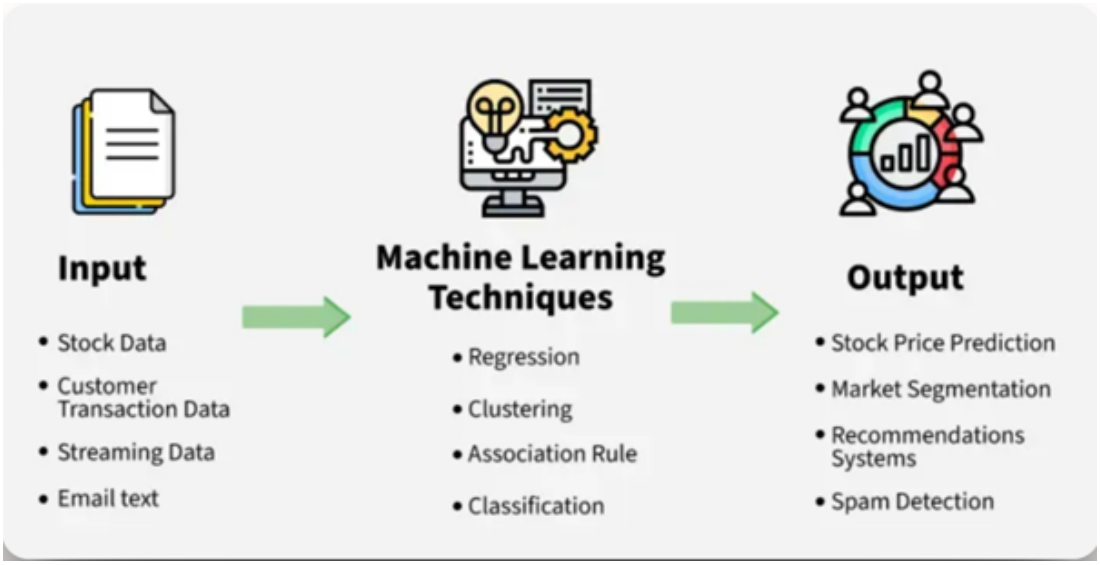


FIGURE 5.3 – Concept du Machine Learning

5.5.1 Intérêt du Machine Learning pour la Business Intelligence

L'apport du Machine Learning (ML) dans les plateformes de Business Intelligence (BI) tient à sa capacité à aller au-delà de la simple description des données. En s'appuyant sur des algorithmes capables d'apprendre à partir de l'historique, il permet d'automatiser la détection de tendances, d'anticiper des résultats futurs et de mettre en lumière des schémas autrement invisibles. L'analyse ne se cantonne plus au passé : elle devient proactive, orientant la prise de décision bien en amont des événements[28].

5.5.2 Étude comparative des modèles de prédiction

Avant de retenir Prophet, plusieurs algorithmes adaptés à la prédiction de séries temporelles ont été examinés. Le tableau 5.6 en présente une comparaison synthétique selon leurs avantages respectifs et leurs limites principales.

TABLE 5.6 – Comparaison des principaux algorithmes d'apprentissage automatique pour la prédiction de séries temporelles

Modèle	**Famille / approche**	**Avantages principaux**	**Limites et défis**
Prophet	Modèle généralisé additif (GAM)	Déploiement et interprétation facilités. Robuste face aux données manquantes et aux valeurs aberrantes. Capture efficacement la saisonnalité et les événements calendaires.	Moins adapté à la modélisation de dynamiques fortement non linéaires ou à très court terme.
ARIMA / SARIMA	Statistique classique	Fondements théoriques solides. Particulièrement adapté aux séries courtes et relativement linéaires.	Nécessite des données stationnaires. Le réglage des paramètres (p, d, q) est délicat et la modélisation de non-linéarités complexes reste limitée.
XGBoost / LightGBM	Machine Learning (arbres de décision)	Performances élevées et rapidité d'exécution. Capture efficacement les relations non linéaires.	Ne modélise pas nativement la temporalité : un important travail de *feature engineering* (fenêtres glissantes, retards, lags) est requis. Faible interprétabilité (effet boîte noire).
LSTM (RNN)	Deep Learning (réseaux récurrents)	Capable de mémoriser des dépendances séquentielles à long terme. Bien adapté aux signaux riches et denses.	Entraînement coûteux en ressources, sensibilité au surapprentissage et interprétabilité très limitée. Nécessite généralement de grands volumes de données.
Transformers*	Deep Learning (attention)	Très puissant pour l'analyse multivariée à grande échelle. Hautement parallélisable, avec une capacité à conserver un contexte global.	Mise en œuvre complexe, coûts de calcul élevés et besoins en données importants – souvent disproportionnés pour un projet de taille académique.

* Exemples de modèles temporels basés sur les Transformers : *Informer*, *TimeGPT* et *PatchTST*.

5.5.3 Justification du choix de l'algorithme

Le choix de Prophet n'a pas été immédiat : plusieurs alternatives ont été sérieusement envisagées avant de trancher. ARIMA a d'abord été écarté en raison de sa contrainte de stationnarité, difficile à satisfaire avec des prix d'actions soumis à des tendances longues et persistantes. Les

réseaux LSTM ont également été testés, mais le temps d'entraînement s'est révélé trop important pour l'infrastructure disponible, sans apporter d'amélioration notable des résultats sur nos actifs.

Prophet s'est finalement imposé pour trois raisons concrètes. D'abord, il prend en charge nativement les saisonnalités hebdomadaires et annuelles, que l'on observe clairement dans les historiques de JPMorgan ou Apple. Ensuite, il tolère les valeurs manquantes sans prétraitement spécifique, ce qui a représenté un avantage réel avec les données issues de Stooq. Enfin, son architecture additive permet de décomposer la prévision en tendance et en composantes saisonnières, rendant les résultats compréhensibles par un analyste non spécialisé en modélisation [29].

Le paramètre `changepoint_prior_scale` a été fixé à 0.15 après une série d'expérimentations : en dessous de cette valeur, le modèle ne détectait pas suffisamment les ruptures de tendance ; au-dessus, il sur-interprétait les fluctuations normales du marché.

5.6 Mise en place du modèle

5.6.1 Importation des bibliothèques nécessaires

La figure 5.4 illustre les bibliothèques importées dans le composant de prédiction.

```
1  import os
2  import sys
3  import pandas as pd
4  import numpy as np
5  import matplotlib
6  matplotlib.use('Agg')
7  import matplotlib.pyplot as plt
8  from prophet import Prophet
```

FIGURE 5.4 – Importation des bibliothèques nécessaires

Pour construire le pipeline d'intelligence artificielle et assurer la circulation des données depuis la base PostgreSQL jusqu'à la génération des prévisions, le composant `prediction.py` s'appuie sur un ensemble éprouvé de bibliothèques Python. Le choix de chaque dépendance a été guidé par des contraintes précises d'ingénierie des données et d'architecture logicielle.

Interaction système (`os` et `sys`) : L'architecture modulaire du projet impose une gestion dynamique des chemins d'accès et des contextes d'exécution. Les modules natifs `os` et `sys` garantissent l'indépendance du code vis-à-vis du système d'exploitation hébergeant le serveur. Ils permettent notamment de modifier l'arbre des importations (`sys.path.insert`) pour isoler la configuration de la base de données, ainsi que de

gérer automatiquement la création des dossiers de stockage (`os.makedirs`) dédiés aux rapports PDF.

Ingénierie des données (`pandas` et `numpy`) : L'ensemble de la phase de préparation et de traitement des données repose sur ces deux piliers de la science des données. `pandas` est utilisé pour convertir les résultats de requêtes SQL en *DataFrames* manipulables en mémoire, nettoyer les valeurs nulles, filtrer l'historique aux cinq dernières années de cotations, et formater rigoureusement les colonnes temporelle (`ds`) et cible (`y`) exigées par Prophet. `numpy`, quant à lui, intervient pour le calcul vectorisé des indicateurs de précision (sMAPE, RMSE) et pour garantir la cohérence mathématique des intervalles de confiance via `np.maximum`.

Rendu graphique en mode silencieux (`matplotlib`) : La génération des graphiques de prévision au format PDF a posé un défi spécifique : le script s'exécutant sur un serveur sans gestionnaire d'affichage, un appel standard à `matplotlib` aurait provoqué un arrêt de l'API. Pour contourner ce problème, le moteur a été explicitement configuré en mode non interactif via la directive `matplotlib.use('Agg')`, permettant la production d'images sans interface graphique.

Inférence prédictive (`Prophet`) : Véritable cœur du module, cette bibliothèque développée par Meta a été retenue pour sa capacité à décomposer les séries financières en composantes interprétables. L'instance Prophet a été paramétrée sur mesure – notamment via `changepoint_prior_scale = 0.15` – afin d'absorber la volatilité caractéristique des marchés boursiers tout en conservant une explicabilité suffisante pour les utilisateurs finaux.

5.6.2 Préparation pour la prédiction

Avant de pouvoir ajuster le modèle, les données doivent être extraites depuis PostgreSQL et mises en forme selon les conventions attendues par Prophet. Cette phase de préparation, illustrée aux figures 5.5 et 5.6, comprend le chargement des séries temporelles, le filtrage de l'historique et le nettoyage des valeurs aberrantes ou manquantes.

```
conn = get_database_connection()
try:
    cursor = conn.cursor()

    # 1. Fetch data from PostgreSQL
    query = "SELECT date AS ds, close_price, high_price, low_price FROM silver.ohlcv_cleaned WHERE ticker = %s ORDER BY date ASC"
    cursor.execute(query, (ticker,))

    rows = cursor.fetchall()
    columns = [desc[0] for desc in cursor.description]
    df = pd.DataFrame(rows, columns=columns)

    if df.empty:
        raise ValueError(f"No data found for ticker {ticker}")

    cursor.close()
finally:
    conn.close()
```

FIGURE 5.5 – Préparation pour la prédiction – partie 1

```python
# Clean data
df['ds'] = pd.to_datetime(df['ds'])
df['close_price'] = pd.to_numeric(df['close_price'])
df['high_price'] = pd.to_numeric(df['high_price'])
df['low_price'] = pd.to_numeric(df['low_price'])
df = df.dropna()

# Filter data to the last 5 years, or use the first available date if there is less than 5 years of data
max_date = df['ds'].max()
first_date = df['ds'].min()

target_min_date = max_date - pd.DateOffset(years=5)
min_date = max(first_date, target_min_date)

df = df[df['ds'] >= min_date]
```

FIGURE 5.6 – Préparation pour la prédiction – partie 2

5.6.3 Génération des Prédictions

Une fois les données préparées, le modèle est ajusté et les prévisions sont générées, accompagnées des cônes d'incertitude associés à chaque horizon temporel. La figure 5.7 illustre cette étape.

```python
# Prepare datasets for each model
df_close = df[['ds', 'close_price']].rename(columns={'close_price': 'y'})
df_high = df[['ds', 'high_price']].rename(columns={'high_price': 'y'})
df_low = df[['ds', 'low_price']].rename(columns={'low_price': 'y'})

model_close = get_prophet_model().fit(df_close)
model_high = get_prophet_model().fit(df_high)
model_low = get_prophet_model().fit(df_low)

# 3. Generate Predictions
future = model_close.make_future_dataframe(periods=periods, freq='D')

forecast_close = model_close.predict(future)
forecast_high = model_high.predict(future)
forecast_low = model_low.predict(future)

# Make upper and lower bounds perfectly logical using the predicted high and low prices
# We also ensure the math holds: high >= close >= low
forecast = forecast_close.copy()
forecast['yhat_upper'] = np.maximum(forecast_close['yhat'], forecast_high['yhat'])
forecast['yhat_lower'] = np.minimum(forecast_close['yhat'], forecast_low['yhat'])
```

FIGURE 5.7 – Génération des prédictions et des cônes d'incertitude

5.7 Évaluation

Afin de s'assurer de la fiabilité du modèle Prophet, un processus d'évaluation *in-sample* a été mis en place. Ce processus consiste à confronter rétroactivement les prix réels observés (y_i) aux valeurs estimées par le modèle ($\hat{y}_i$) sur la période d'entraînement. Pour obtenir une vision complète du comportement de l'algorithme face à la volatilité des marchés, quatre indicateurs de performance ont été calculés à l'aide d'opérations vectorielles (`numpy`).

5.7.1 L'Erreur Absolue Moyenne (MAE)

Le *Mean Absolute Error* mesure l'amplitude moyenne des erreurs de prévision, indépendamment de leur signe. Cet indicateur présente l'avantage d'être directement exprimé dans l'unité monétaire de l'actif étudié, ce qui le rend particulièrement intuitif à interpréter. Son implémentation en Python – `np.mean(np.abs(y_true - y_pred))` – correspond à la formule suivante :

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5.1}$$

5.7.2 La Racine de l'Erreur Quadratique Moyenne (RMSE)

Le *Root Mean Square Error* pénalise plus fortement les erreurs de grande amplitude en les élevant au carré avant de calculer la moyenne. Dans un contexte boursier, cette propriété est particulièrement pertinente : il est préférable d'accumuler de légères imprécisions plutôt que de commettre une erreur massive lors d'un événement de marché extrême. La formule correspondante, calculée via `np.sqrt(np.mean((y_true - y_pred)**2))`, est la suivante :

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{5.2}$$

5.7.3 Score de précision global (Accuracy)

Pour rendre ces résultats statistiques directement exploitables dans l'interface applicative, un indicateur agrégé de précision est calculé. Il traduit l'erreur sMAPE sous la forme d'un score de réussite exprimé en pourcentage, borné inférieurement à zéro :

$$\text{Précision } (\%) = \max\left(0,\, 100 - \text{sMAPE}\right) \tag{5.3}$$

Cette approche à plusieurs dimensions permet de s'assurer que l'intégration de Prophet ne relève pas d'une simple démonstration technique, mais répond de façon mesurable à des exigences concrètes de fiabilité financière.

5.7.4 Résultats de l'évaluation des prédictions

Le tableau 5.7 synthétise les performances obtenues par le modèle Prophet sur l'ensemble des actifs testés.

TABLE 5.7 – Synthèse des performances prédictives *in-sample* du modèle Prophet par actif.

Actif (Ticker)	**MAE ($)**	**RMSE ($)**	**Précision globale (%)**
Apple Inc. (AAPL)	4.82	6.21	**97.35**
Microsoft Corp. (MSFT)	8.90	11.33	**97.36**
Tesla Inc. (TSLA)	21.93	27.91	**91.61**
Google (GOOG)	4.26	5.60	**97.18**
JPMorgan Chase (JPM)	4.68	6.05	**97.42**
S&P 500 ETF (SPY)	7.12	9.51	**98.47**

Les résultats sont globalement satisfaisants, avec une précision comprise entre 91 % et 98 % selon l'actif. Tesla affiche logiquement les erreurs les plus élevées, en raison de la forte volatilité qui caractérise ce titre. À l'inverse, le S&P 500 ETF, dont les mouvements sont plus progressifs et réguliers, atteint le meilleur score de précision de l'ensemble.

5.8 Intégration du modèle dans l'interface de l'application

Un modèle algorithmique ne délivre sa pleine valeur que lorsqu'il est accessible et lisible par ses utilisateurs finaux. Une attention particulière a donc été portée à la restitution des prévisions de Prophet au sein de l'interface web développée sous **Angular**. Cette phase d'intégration soulève plusieurs défis techniques à l'intersection de l'ingénierie logicielle et de l'expérience utilisateur.

5.8.1 Architecture de flux et de consommation des données

La communication entre le backend analytique (Python) et l'interface client (TypeScript/Angular) s'appuie sur une architecture d'API REST asynchrone. À chaque requête, le backend génère les prédictions à la volée et renvoie un objet JSON structuré encapsulant à la fois les valeurs futures estimées (`yhat`), l'historique récent (`history`) et les bornes de l'intervalle de confiance (`yhat_lower` et `yhat_upper`).

Afin de préserver la fluidité de l'interface et d'éviter une surcharge mémoire lors du rendu des graphiques, une optimisation a été appliquée côté serveur : l'historique renvoyé est tronqué aux 1 500 derniers points. Cette limite garantit un temps de réponse acceptable tout en offrant à l'utilisateur un contexte visuel suffisant.

5.8.2 Gestion des états et Expérience Utilisateur (UX)

L'exécution de Prophet implique plusieurs étapes successives côté serveur (requête SQL, nettoyage, entraînement via *.fit()* et inférence via *.predict()*), ce qui induit un délai de traitement non négligeable. L'intégration Angular a donc été conçue pour gérer ce délai de façon transparente et robuste.

En s'appuyant sur les *Observables* de la bibliothèque RxJS, les états de l'interface sont pilotés dynamiquement : affichage d'un indicateur de chargement (*spinner*) pendant l'attente, gestion élégante des erreurs en cas d'échec du modèle, et réinitialisation propre de l'état via l'opérateur `finalize()`. Grâce à cette mécanique, l'utilisateur n'est jamais confronté à une interface figée sans retour visuel.

5.8.3 Restitution visuelle et cône d'incertitude

L'affichage final des résultats repose sur un composant graphique dédié. L'enjeu n'était pas seulement de tracer la tendance future, mais d'exprimer visuellement le niveau de risque associé à chaque prévision.

Pour y répondre, les bornes calculées par Prophet ont été exploitées pour construire ce que l'on appelle un **cône d'incertitude**. Sur le graphique, la courbe centrale représente la prédiction nominale de clôture, entourée d'une zone semi-transparente matérialisant l'intervalle de confiance – généralement paramétré à 95 %. Plus l'horizon de prévision s'éloigne dans le temps, plus cette zone s'élargit, traduisant visuellement et honnêtement le fait que l'incertitude croît mécaniquement avec la distance temporelle. Cette représentation est un standard reconnu dans les outils d'analyse financière professionnels.

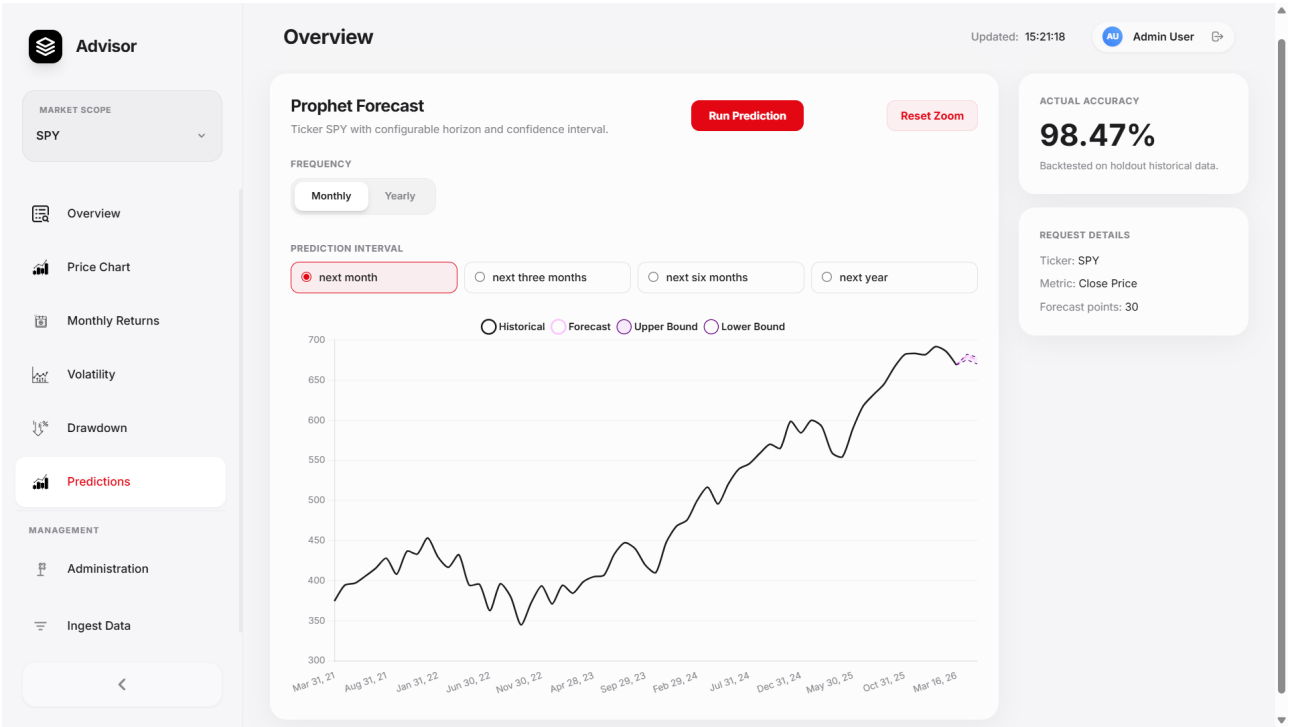


FIGURE 5.8 – Interface graphique de prédiction – vue 1

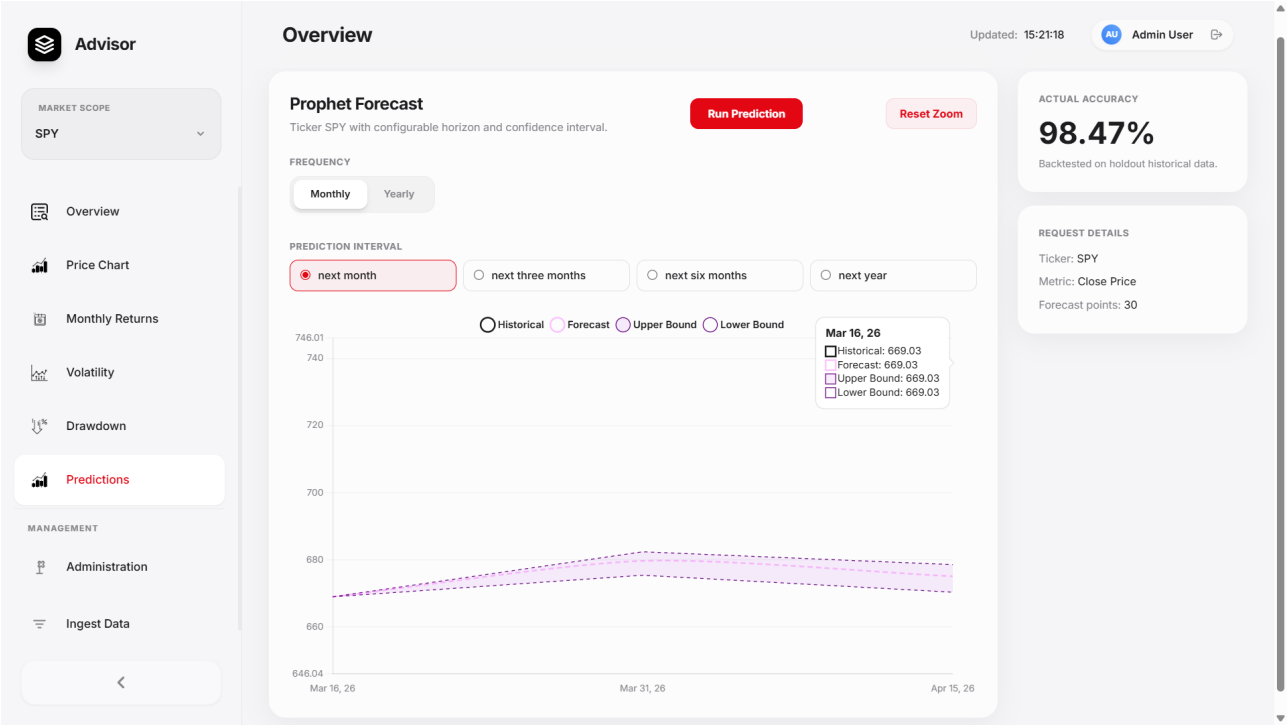


FIGURE 5.9 – Interface graphique de prédiction – vue 2

5.9 Conclusion

Ce chapitre a présenté l'ensemble du cycle de développement du module prédictif, depuis le choix algorithmique jusqu'à l'intégration dans l'interface utilisateur. Les résultats obtenus sur le banc de test sont encourageants : Prophet atteint une précision globale comprise entre 91 % et 98 % selon les actifs, avec des erreurs absolues moyennes inférieures à 22 dollars. Ces chiffres valident la pertinence du choix algorithmique pour le contexte applicatif visé.

Le module est désormais opérationnel et intégré au tableau de bord sous la forme d'un graphique interactif avec cône d'incertitude, offrant à l'utilisateur une lecture visuelle du risque associé à chaque horizon de prévision. Fort de ces fondations analytiques, le dernier sprint s'attaque à la couche intelligente de la plateforme : l'assistant conversationnel LEXA.

6.1 Introduction

Les quatre premiers sprints ont permis de poser les fondations techniques de la plateforme : les données sont collectées, nettoyées, les prédictions sont générées et les accès sont sécurisés. Cependant, un élément central faisait encore défaut. Un analyste confronté à un graphique de drawdown ou à une courbe de volatilité ne dispose pas nécessairement des clés pour en interpréter correctement les nuances. De même, lorsqu'il cherche à identifier la réglementation tunisienne applicable à un actif particulier, il se retrouve souvent à parcourir seul des textes juridiques denses et techniques. C'est précisément pour répondre à ces deux lacunes que l'assistant **LEXA** a été conçu et développé au cours de ce cinquième sprint.

6.2 Backlog Du Sprint 4

Le tableau ci-dessous présente le Backlog du Quatrième Sprint.

Table 6.1 – Backlog du Sprint 4

ID	Fonctionnalité	User Story	Tâche
1	**Interface de Chat**	En tant qu'utilisateur, je souhaite interagir avec l'assistant LEXA via une interface de messagerie réactive afin de poser mes questions et consulter les réponses obtenues.	Développer l'interface utilisateur de discussion (Frontend) en gérant l'envoi des messages, les états de chargement liés à l'IA ainsi que l'affichage des résultats.
2	**Mode de Discussion**	En tant qu'utilisateur, je souhaite orienter la discussion en sélectionnant un mode (Analyste ou Juridique) afin d'obtenir des réponses adaptées à mon contexte métier.	Implémenter un sélecteur de domaine côté client et adapter le comportement du backend en fonction du mode sélectionné.
3	**Recherche (RAG)**	En tant qu'utilisateur en mode Juridique, je souhaite que LEXA s'appuie sur des textes légaux réels extraits de la base de données pour étayer ses réponses.	Développer les services RAG (Backend) permettant d'interroger la base vectorielle (ChromaDB), de récupérer les textes sources et de générer la réponse via l'API Gemini.
4	**Gestion des Sessions**	En tant qu'utilisateur, je souhaite conserver l'historique de mes échanges sous forme de sessions afin de pouvoir reprendre d'anciennes analyses à tout moment.	Développer la persistance des conversations (Backend) sur PostgreSQL : création, lecture, renommage et suppression des sessions.
5	**Ingestion de Lois**	En tant qu'administrateur, je souhaite déclencher un module de chargement avec de nouvelles données (lois, réglementations) afin d'enrichir la base de connaissances de l'IA.	Créer une interface d'administration sécurisée offrant un tableau de bord dédié à l'import de données textuelles et au déclenchement du processus d'ingestion.
6	**Pipeline ETL Loi**	En tant qu'administrateur, je souhaite que les documents bruts importés soient automatiquement formatés, nettoyés puis vectorisés dans ChromaDB.	Concevoir le pipeline interne : extraction du texte brut (Bronze), segmentation et nettoyage (Silver), puis génération et insertion des embeddings vectoriels (Gold).

6.3 Spécification Fonctionnelle

Afin de bien cerner ce que le système doit être capable de réaliser, cette section s'appuie sur un diagramme de cas d'utilisation illustrant les interactions entre les différents acteurs et la plateforme tout au long de ce sprint. Chaque cas d'utilisation est ensuite décrit de manière structurée : on y précise les conditions nécessaires à son déclenchement, les résultats attendus, ainsi que le déroulement des scénarios nominaux et alternatifs. Ces éléments constituent une référence commune pour les phases de conception, de développement et de validation.

6.3.1 Diagramme de cas d'utilisation

Le diagramme ci-dessous illustre les interactions entre le **Responsable Financier** et l'**Analyste Financier** avec **LEXA**. Le premier gère les lois financières et sollicite des explications à caractère juridique, tandis que le second utilise le chatbot pour obtenir des

explications relatives à l'analyse financière et générer des rapports.

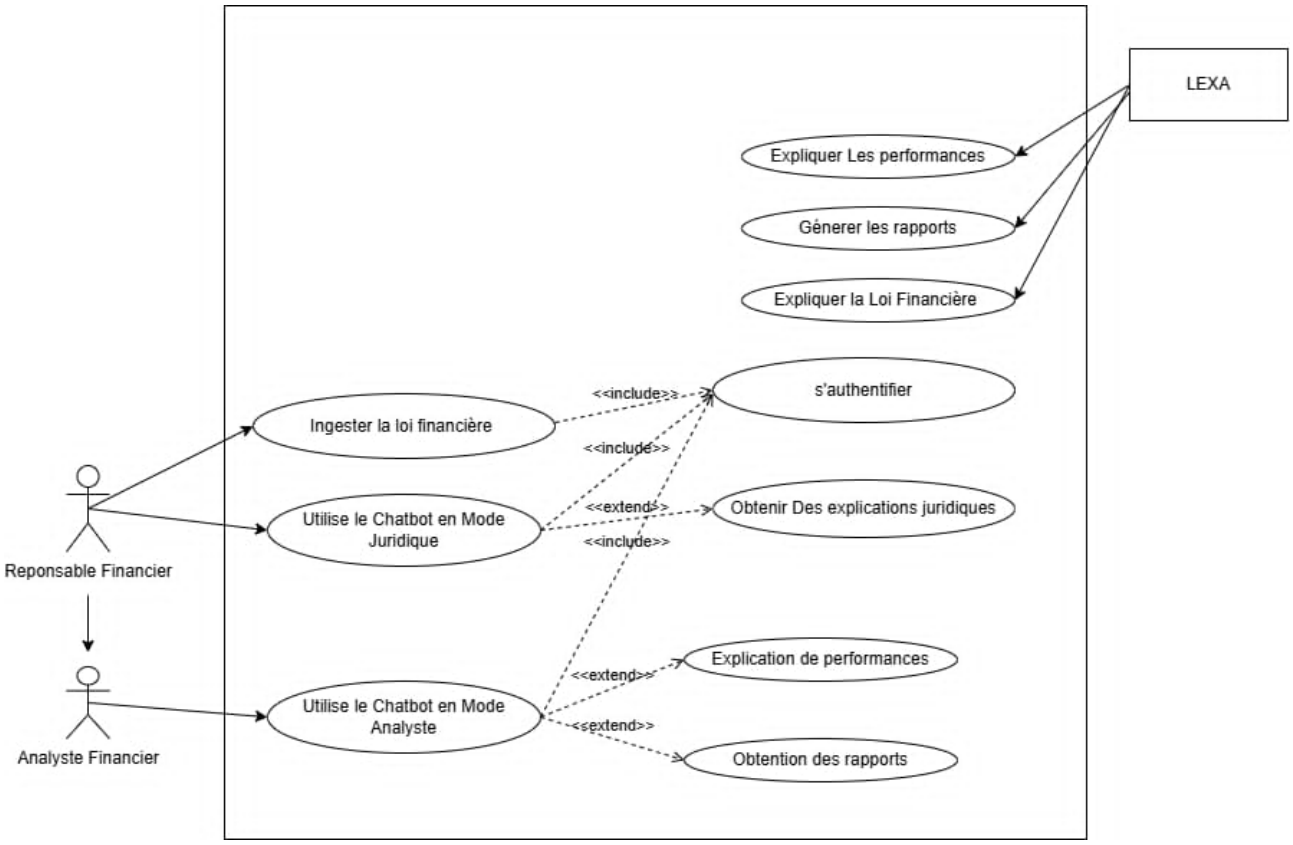


FIGURE 6.1 – Diagramme de cas d'utilisation du Sprint 4

6.3.2 Description détaillée des cas d'utilisation

6.3.2.1 Ingester la loi financière

Le tableau suivant détaille le cas d'utilisation relatif à l'ingestion d'une loi financière dans le système LEXA.

TABLE 6.2 – Description détaillée du cas d'utilisation « Ingérer la loi financière »

Titre	Cas d'utilisation : Ingérer la loi financière
Acteur principal	Responsable Financier
Description	Cas inclus dans UC-01. Le Responsable Financier soumet un document de loi financière en vue de son intégration dans la base de connaissances du système LEXA.
Préconditions	1. L'utilisateur a préalablement été authentifié. 2. Le document de loi financière est disponible dans un format compatible avec le système.
Postconditions	1. La loi financière a été ingérée et correctement indexée dans le système. 2. Son contenu est désormais accessible pour toute interrogation ultérieure.
Scénario nominal	1. Le Responsable Financier s'authentifie auprès du système. 2. Il sélectionne le document de loi financière qu'il souhaite ingérer. 3. Le système procède à la validation du format du document soumis. 4. Le contenu est extrait puis indexé au sein de la base de connaissances. 5. Une notification de succès est transmise à l'utilisateur à l'issue du traitement.
Scénarios alternatifs	Format de document non supporté → le système génère un message d'erreur explicite et invite l'utilisateur à soumettre un fichier conforme.
Règles de gestion	Seuls les formats documentaires homologués sont acceptés par le système. Par ailleurs, toute opération d'ingestion est conditionnée à une authentification préalable de l'utilisateur.

6.3.2.2 Description Détaillée De cas d'utilisation "Utiliser le Chatbot en Mode Juridique"

Le tableau suivant présente le cas d'utilisation correspondant à l'interaction avec le chatbot LEXA lorsque celui-ci opère en mode juridique.

TABLE 6.3 – Description détaillée du cas d'utilisation « Utiliser le Chatbot en Mode Juridique »

Titre	Cas d'utilisation : Utiliser le Chatbot en Mode Juridique
Acteur principal	Responsable Financier
Description	Le Responsable Financier interagit avec le chatbot LEXA en mode juridique afin d'obtenir des éclaircissements sur les lois financières préalablement ingérées dans le système.
Préconditions	1. L'utilisateur est authentifié et dispose d'une session active. 2. Au moins une loi financière a été ingérée dans la base de connaissances du système.
Postconditions	1. L'utilisateur a reçu une réponse juridique en lien avec sa requête. 2. L'ensemble de la conversation est sauvegardé dans l'historique du système.
Scénario nominal	1. L'utilisateur s'authentifie et accède à l'interface du chatbot. 2. Il active le mode juridique depuis les options disponibles. 3. Il formule et soumet une question relative à une loi financière. 4. Le système effectue une recherche dans la base de connaissances pour identifier les éléments pertinents. 5. Le chatbot restitue une explication juridique structurée, accompagnée des références légales correspondantes.
Scénarios alternatifs	Aucune loi correspondant à la requête n'est trouvée → le système en informe l'utilisateur et lui propose d'affiner ou de reformuler sa question.
Règles de gestion	Les réponses générées sont strictement fondées sur les lois financières présentes dans la base de connaissances. Chaque réponse doit obligatoirement mentionner la référence légale sur laquelle elle s'appuie.

6.3.2.3 Description Détaillée De cas d'utilisation "Obtenir des explications juridiques"

Ce cas d'utilisation, présenté dans le tableau ci-dessous, constitue une extension du cas « Utiliser le Chatbot en Mode Juridique » et décrit le traitement interne réalisé par le système pour produire une réponse juridique.

TABLE 6.4 – Description détaillée du cas d'utilisation « Obtenir des explications juridiques »

Titre	Cas d'utilisation : Obtenir des explications juridiques
Acteur principal	Système IA (LEXA)
Description	Cas étendu depuis « Utiliser le Chatbot en Mode Juridique ». À la réception d'une requête, le système analyse son contenu et génère une explication juridique fondée sur les lois financières disponibles dans sa base de connaissances.
Préconditions	1. Le mode juridique est activé au moment du traitement. 2. La base de connaissances juridiques est alimentée et opérationnelle.
Postconditions	1. Une explication juridique claire et compréhensible est retournée à l'utilisateur. 2. La source légale ayant servi de fondement à la réponse est explicitement citée.
Scénario nominal	1. Le système reçoit et interprète la requête soumise par l'utilisateur. 2. Il interroge la base de connaissances pour identifier les articles de loi pertinents. 3. Une réponse structurée et référencée est générée à partir des éléments trouvés. 4. Cette réponse est transmise et affichée à l'utilisateur.
Scénarios alternatifs	Contenu insuffisant dans la base de connaissances → une réponse partielle est fournie, assortie d'un avertissement signalant les limites de complétude.
Règles de gestion	Toute explication produite par le système doit impérativement faire référence à l'article de loi source ayant servi de base à la réponse.

6.3.2.4 Description Détaillée De cas d'utilisation "Utiliser le Chatbot en Mode Analyste"

Le tableau ci-dessous décrit le cas d'utilisation relatif à l'utilisation du chatbot LEXA dans son mode analyste, dédié à l'exploitation des données financières.

TABLE 6.5 – Description détaillée du cas d'utilisation « Utiliser le Chatbot en Mode Analyste »

Titre	Cas d'utilisation : Utiliser le Chatbot en Mode Analyste
Acteur principal	Analyste Financier
Description	L'Analyste Financier interagit avec le chatbot LEXA en mode analyste afin d'obtenir des interprétations des performances financières et de générer des rapports d'analyse adaptés à ses besoins.
Préconditions	1. L'utilisateur est authentifié et dispose d'une session active. 2. Les données financières nécessaires au traitement des requêtes sont disponibles dans le système.
Postconditions	1. L'utilisateur a obtenu les analyses ou rapports sollicités. 2. Les résultats produits sont accessibles pour consultation ou export selon ses besoins.
Scénario nominal	1. L'Analyste Financier sélectionne le mode analyste depuis l'interface du chatbot. 2. Il soumet une requête portant sur une analyse ou la génération d'un rapport. 3. Le système traite la demande en interrogeant les données financières disponibles. 4. Le chatbot restitue une analyse détaillée ou un rapport structuré en réponse à la requête.
Scénarios alternatifs	Données insuffisantes pour couvrir la requête → une analyse partielle est retournée, accompagnée d'une indication claire sur les données manquantes.
Règles de gestion	L'ensemble des analyses produites repose exclusivement sur les données financières validées et disponibles dans le système, à l'exclusion de toute source externe.

6.3.2.5 Description Détaillée De cas d'utilisation "Explication de performances"

Le tableau suivant détaille le cas d'utilisation « Explication de performances », qui constitue une extension du mode analyste du chatbot LEXA.

TABLE 6.6 – Description détaillée du cas d'utilisation « Explication de performances »

Titre	Cas d'utilisation : Explication de performances
Acteur principal	Système IA (LEXA)
Description	Cas étendu depuis « Utiliser le Chatbot en Mode Analyste ». Le système examine les indicateurs de performance financière disponibles et en fournit une interprétation détaillée et contextualisée à l'utilisateur.
Préconditions	1. Le mode analyste est activé et opérationnel. 2. Les données relatives aux performances financières sont disponibles et accessibles dans le système.
Postconditions	1. Une explication claire et structurée des performances financières est mise à disposition de l'utilisateur. 2. Les tendances significatives ainsi que les éventuelles anomalies détectées sont explicitement mises en évidence.
Scénario nominal	1. Le système reçoit une requête d'explication portant sur les performances financières. 2. Il procède à l'extraction des indicateurs financiers pertinents au regard de la demande. 3. Il analyse les tendances, variations et écarts observés dans les données. 4. Une synthèse explicative est générée et transmise à l'analyste.
Scénarios alternatifs	Données incomplètes ou insuffisantes → une explication partielle est fournie, assortie d'un signalement explicite des lacunes identifiées dans les données.
Règles de gestion	Toute explication de performances doit intégrer les indicateurs clés de performance (KPI) retenus, ainsi qu'une analyse des écarts par rapport aux valeurs de référence définies.

6.3.2.6 Description Détaillée De cas d'utilisation "Obtention des rapports"

Enfin, le tableau ci-dessous présente le cas d'utilisation « Obtention des rapports », également étendu depuis le mode analyste, et qui décrit le processus de génération automatique de rapports financiers.

TABLE 6.7 – Description détaillée du cas d'utilisation « Obtention des rapports »

Titre	Cas d'utilisation : Obtention des rapports
Acteur principal	Système IA (LEXA)
Description	Cas étendu depuis « Utiliser le Chatbot en Mode Analyste ». Sur la base des paramètres fournis par l'utilisateur, le système génère automatiquement des rapports financiers structurés à partir des données disponibles.
Préconditions	1. Le mode analyste est activé. 2. Les données financières nécessaires à la génération du rapport demandé sont disponibles dans le système.
Postconditions	1. Un rapport financier structuré a été produit conformément aux paramètres spécifiés. 2. Le rapport est mis à la disposition de l'utilisateur, qui peut le consulter ou le télécharger selon ses besoins.
Scénario nominal	1. L'utilisateur définit les paramètres du rapport souhaité (période couverte, type de rapport, indicateurs retenus). 2. Le système collecte et agrège les données correspondant aux critères spécifiés. 3. Il structure le contenu selon le modèle de rapport applicable. 4. Le rapport finalisé est présenté à l'utilisateur et rendu disponible pour export.
Scénarios alternatifs	Données manquantes pour la période demandée → un rapport partiel est généré, avec mention explicite des données absentes ou non disponibles.
Règles de gestion	Chaque rapport généré est automatiquement horodaté et archivé dans l'historique du système afin d'en assurer la traçabilité.

6.4 Conception

6.4.1 Diagrammes de séquence du Sprint 4

6.4.1.1 LEXA en mode Analyste

La figure ci-dessous décrit le flux d'interaction avec **LEXA** en mode Analyste :

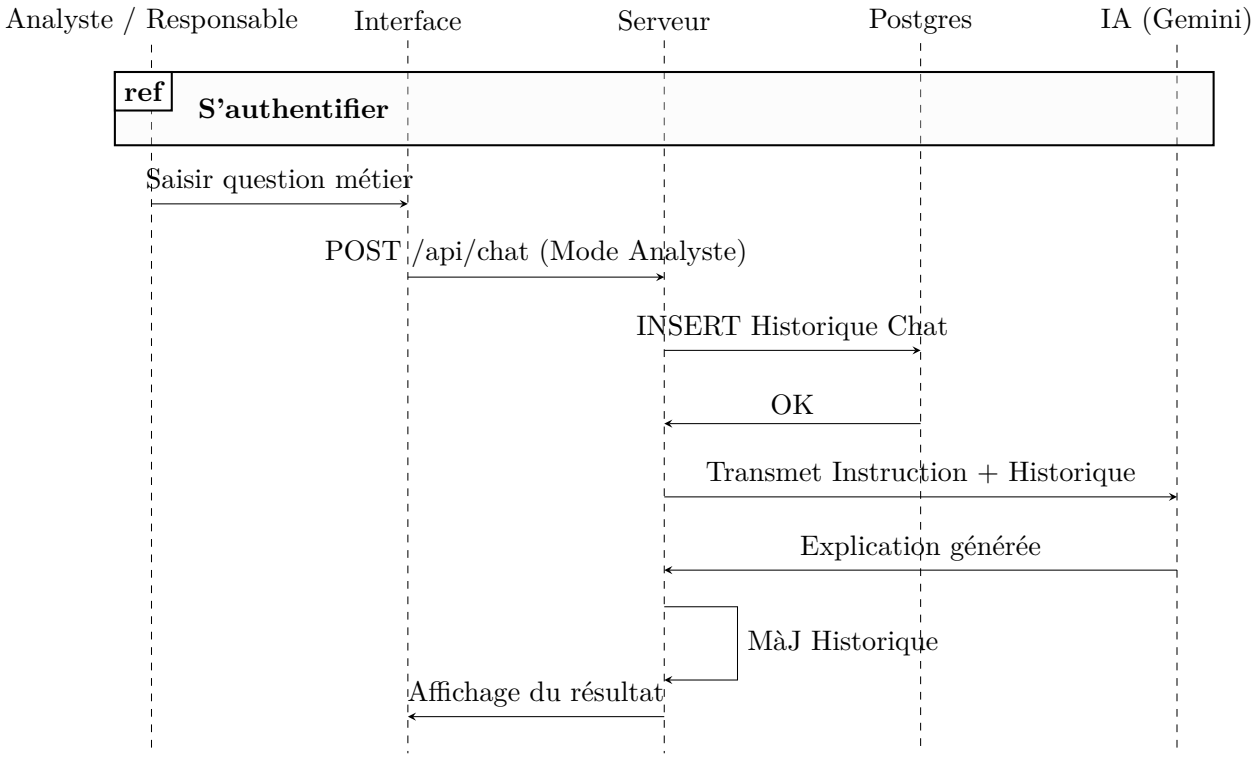


FIGURE 6.2 – Diagramme de séquence « Chatbot LEXA - Mode Analyste »

6.4.1.2 Module d'ingestion de lois

La figure ci-dessous décrit le flux d'ingestion des **lois financières** :

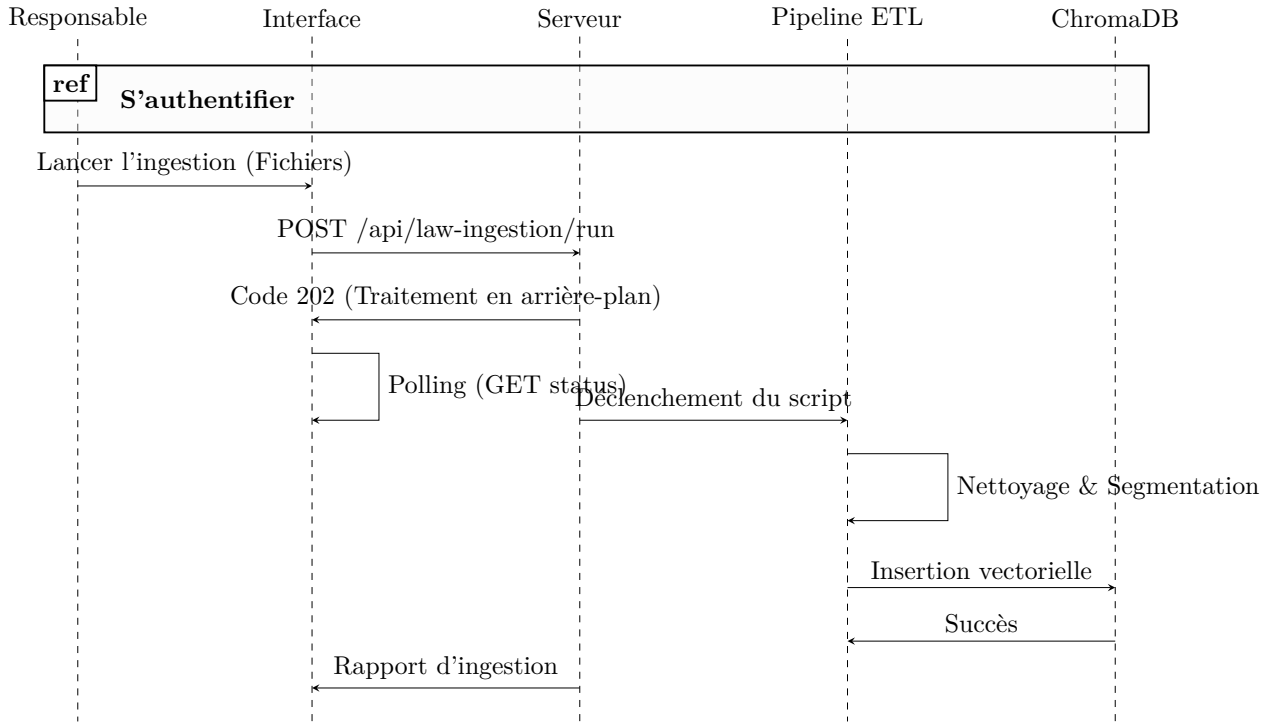


FIGURE 6.3 – Diagramme de séquence « Module d'ingestion de lois »

6.4.1.3 LEXA en mode Juridique

La figure ci-dessous décrit le flux d'interaction avec **LEXA** en mode Juridique :

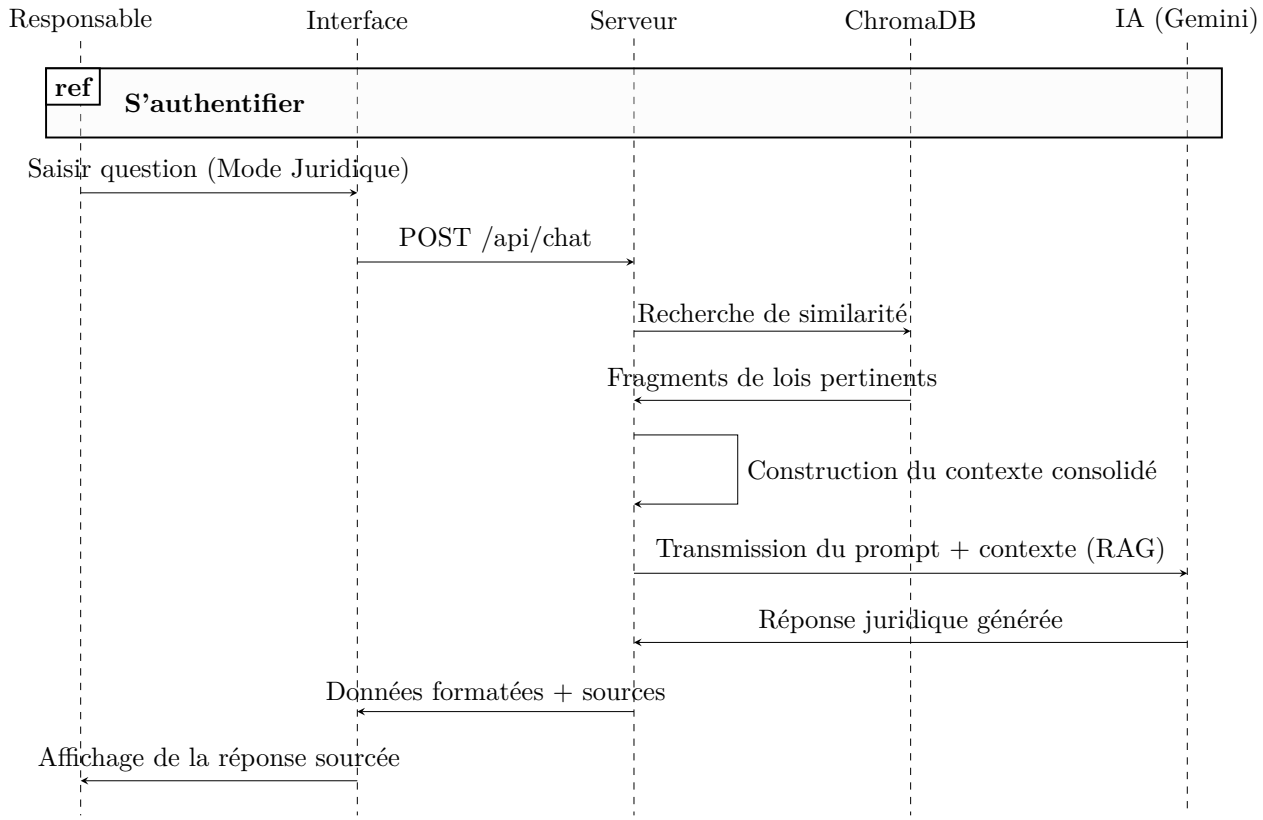


FIGURE 6.4 – Diagramme de séquence « Chatbot LEXA - Mode Juridique (RAG) »

6.5 Module d'ingestion des lois financières

6.5.1 Présentation du processus RAG (*Retrieval-Augmented Generation*)

La problématique à l'origine du mode Juridique de LEXA était simple à formuler, mais délicate à résoudre : comment permettre à un modèle de langage de répondre à des questions portant sur des textes légaux tunisiens qu'il n'a jamais rencontrés lors de son entraînement ? Réentraîner le modèle sur ce corpus spécifique était d'emblée écarté, en raison du coût prohibitif et de la complexité technique que cela implique. La solution retenue repose sur le paradigme **RAG** (*Retrieval-Augmented Generation*). Le principe est le suivant : lorsqu'un utilisateur formule une question, le système commence par interroger la base documentaire interne afin d'en extraire les passages les plus pertinents, qui sont ensuite injectés dans le prompt transmis au modèle. Ce dernier élabore alors sa réponse en s'appuyant sur ces extraits comme contexte, en citant explicitement ses sources. Cette approche est nettement plus accessible qu'un réentraînement complet et produit des réponses ancrées dans des textes réels et vérifiables [30].

6.5.2 Architecture Médaillon pour l'ingestion des lois financières

Afin de garantir une ingestion fiable, structurée et traçable des textes juridiques financiers tunisiens, nous avons adopté l'**Architecture Médaillon**, organisée en trois couches progressives.

6.5.2.1 Couche Bronze

La couche Bronze constitue le point d'entrée du pipeline de données. Son rôle est de collecter et de stocker les textes juridiques dans leur état brut (*as-is*), sans aucune transformation. La source officielle retenue est le **Ministère des Finances Tunisien**, dont la mission consiste à élaborer et mettre en œuvre la politique de l'État en matière financière, monétaire et fiscale, et qui participe à ce titre à la conception des objectifs de la politique économique ainsi qu'à la définition des moyens de réalisation appropriés [31].

6.5.2.2 Couche Silver

La couche **Silver** prend en charge le nettoyage, la normalisation et la structuration des données extraites de la couche Bronze. Les transformations appliquées dans le cadre de ce projet sont les suivantes :

Nettoyage du texte : Le contenu brut de chaque page est normalisé en contractant les espaces multiples en un seul espace et en supprimant les espaces superflus en début et en fin de chaîne. Les pages vides sont détectées et systématiquement exclues du traitement.

Détection de la langue : Le contenu textuel de chaque chunk est analysé afin de déterminer la langue dominante. Cette étape permet d'assigner automatiquement une étiquette linguistique (**ar** pour l'arabe, **fr** pour le français), information conservée en tant que métadonnée associée à chaque segment. Notons que le chunking désigne ici le découpage du texte en blocs de taille maîtrisée, de manière à faciliter la mémorisation et la récupération sémantique [32].

Segmentation du texte : Le contenu de chaque page est découpé en segments de taille contrôlée, selon les paramètres suivants :

- Taille maximale : 1 500 caractères par chunk.
- Chevauchement : 200 caractères entre deux chunks consécutifs, afin de préserver le contexte aux frontières des segments.
- Les chunks vides sont systématiquement ignorés.

Chargement et persistance dans PostgreSQL : Les chunks produits sont enregistrés de manière structurée dans la table PostgreSQL **law_silver.law_chunks**, en vue d'un traitement ultérieur par la couche Gold.

6.5.2.3 Couche Gold

Déduplication : Chaque chunk se voit attribuer un identifiant unique construit à partir du document, de la page et de son index de position. Les doublons éventuels sont détectés et supprimés avant toute insertion.

Sauvegarde PostgreSQL : Les chunks sont chargés en masse dans la table **law_gold.law_docume**
En cas de conflit, un mécanisme d'upsert garantit la mise à jour des entrées déjà existantes.

Indexation RAG : Les chunks sont finalement indexés dans la base vectorielle, accompagnés de leurs métadonnées (document, page, langue, source). Cette indexation permet d'effectuer des recherches sémantiques rapides et précises au moment de la génération des réponses.

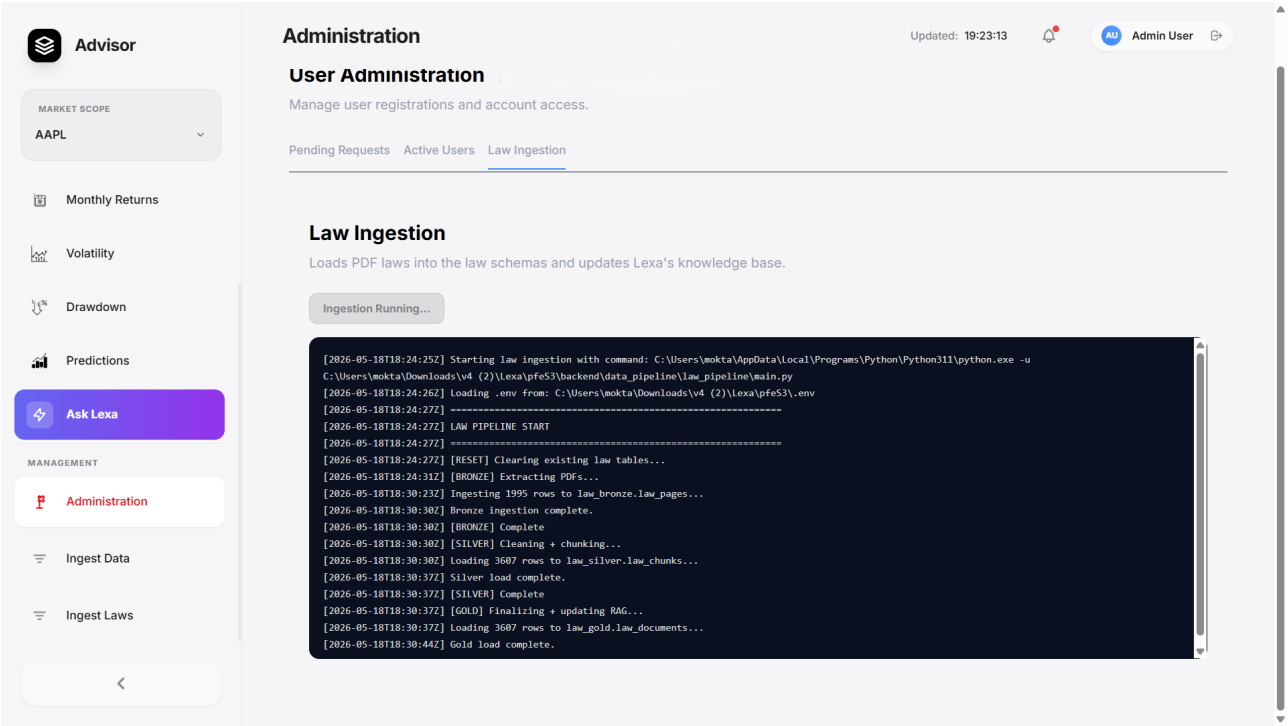


FIGURE 6.5 – Interface LEXA — module d'ingestion de données

6.6 L'assistant intelligent LEXA

LEXA est un assistant intelligent conçu pour opérer dans deux domaines distincts, chacun reposant sur une architecture et une logique de traitement adaptées à ses besoins spécifiques.

6.7 LEXA en mode Juridique

Dans ce mode, **LEXA** s'appuie sur une architecture fondée sur la **Génération Augmentée par Récupération (RAG)**. Le pipeline de traitement se décompose en plusieurs étapes séquentielles, illustrées par la figure ci-dessous.

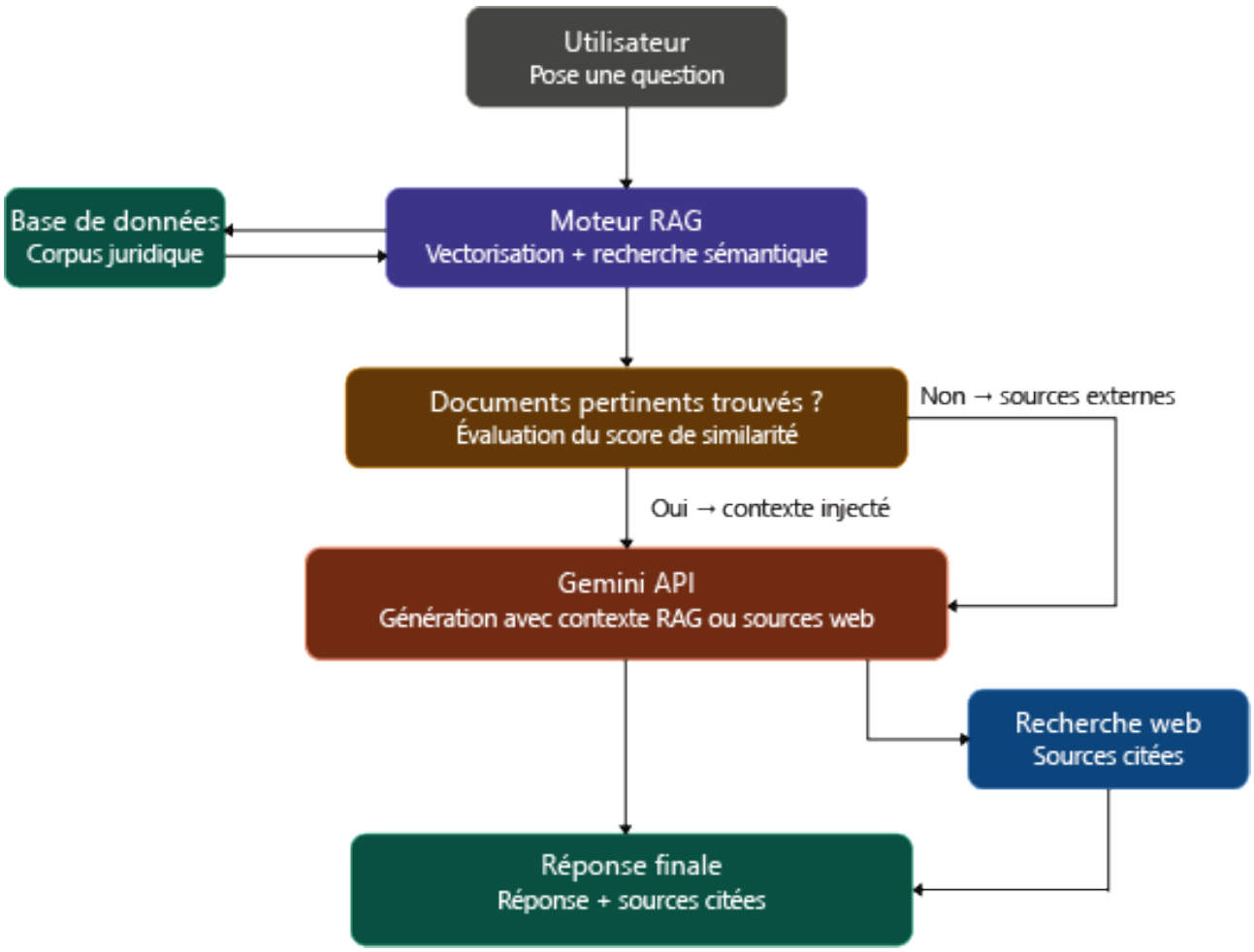


FIGURE 6.6 – Workflow de LEXA en mode Juridique

Lorsque le **Responsable Juridique** soumet une question, LEXA procède d'abord à une recherche sémantique dans la base de données interne, qui contient le corpus juridique indexé par le pipeline d'ingestion des lois financières.

— Si des documents pertinents sont identifiés, ils sont injectés en tant que contexte dans le prompt transmis à **Gemini**, qui se charge de formuler une réponse cohérente et sourcée.

— Dans le cas où le corpus interne s'avère insuffisant, **Gemini** étend automatiquement la recherche à des sources externes, qu'il cite explicitement dans sa réponse. De cette manière, l'utilisateur reçoit systématiquement une réponse **fiable** et **vérifiable**, indépendamment des éventuelles lacunes du corpus interne.

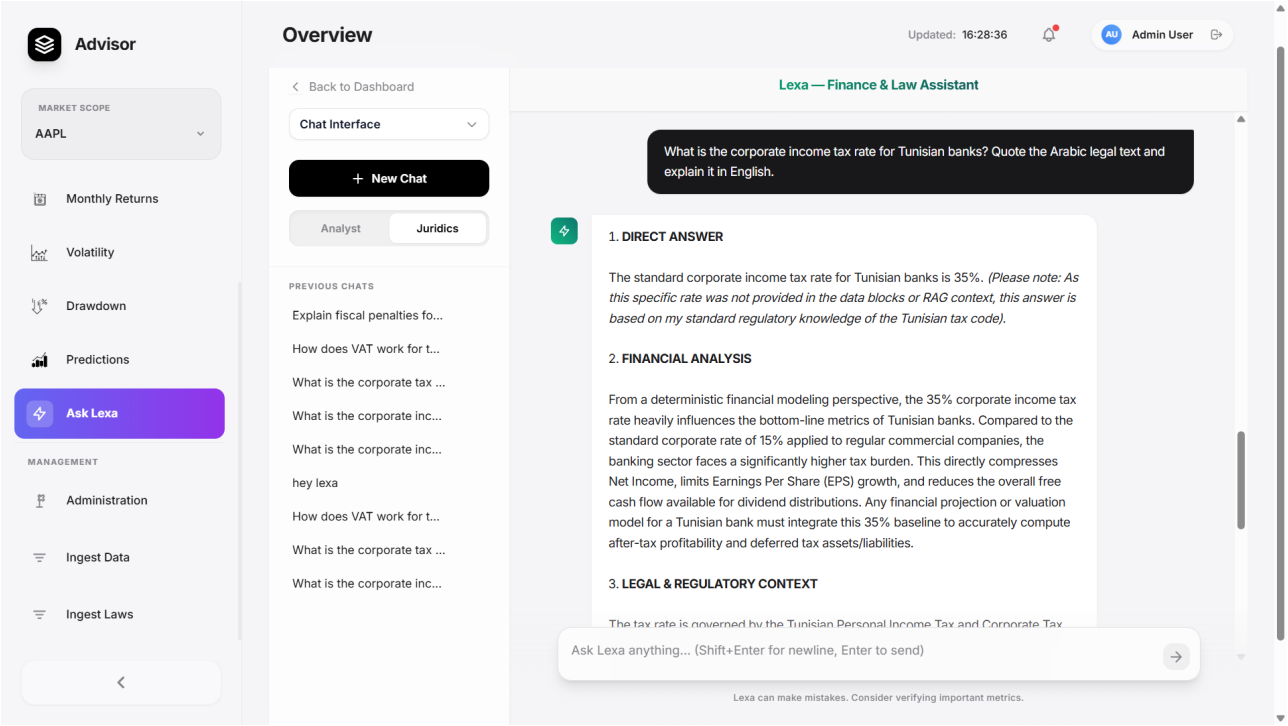


FIGURE 6.7 – Interface de LEXA en mode Juridique

6.8 LEXA en mode Analyse

Dans ce mode, **LEXA** adopte une démarche analytique orientée données de marché. Comme l'illustre la figure ci-dessous, le pipeline repose intégralement sur les données réelles stockées en base.

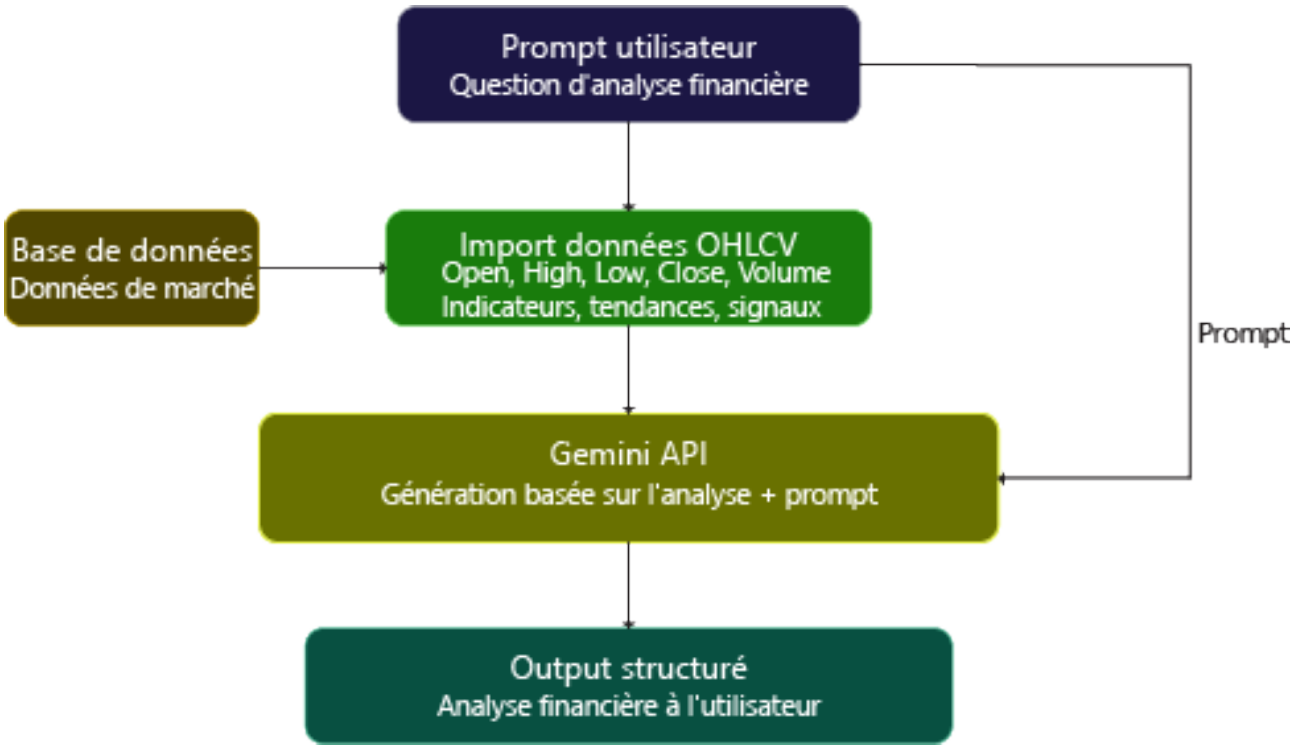


FIGURE 6.8 – Workflow de LEXA en mode Analyste

Le système commence par importer les données **OHLCV** (Open, High, Low, Close, Volume) ainsi que les indicateurs, tendances et signaux correspondant aux actifs financiers concernés, directement depuis la base de données. En tenant compte simultanément de la question posée par l'utilisateur (**Responsable Financier** ou **Analyste Financier**) et du contexte analytique disponible, **Gemini** produit une réponse structurée, ancrée dans les données réelles du marché et calibrée selon les besoins exprimés.

Lorsque les données en base s'avèrent insuffisantes pour répondre à la question posée, **Gemini** élargit automatiquement sa recherche à des sources externes, qu'il cite de manière explicite, garantissant ainsi la traçabilité de l'information fournie.

Ce mode assure que l'analyse financière reste dynamique, accessible et d'un niveau professionnel, quelle que soit la complexité de la question soumise.

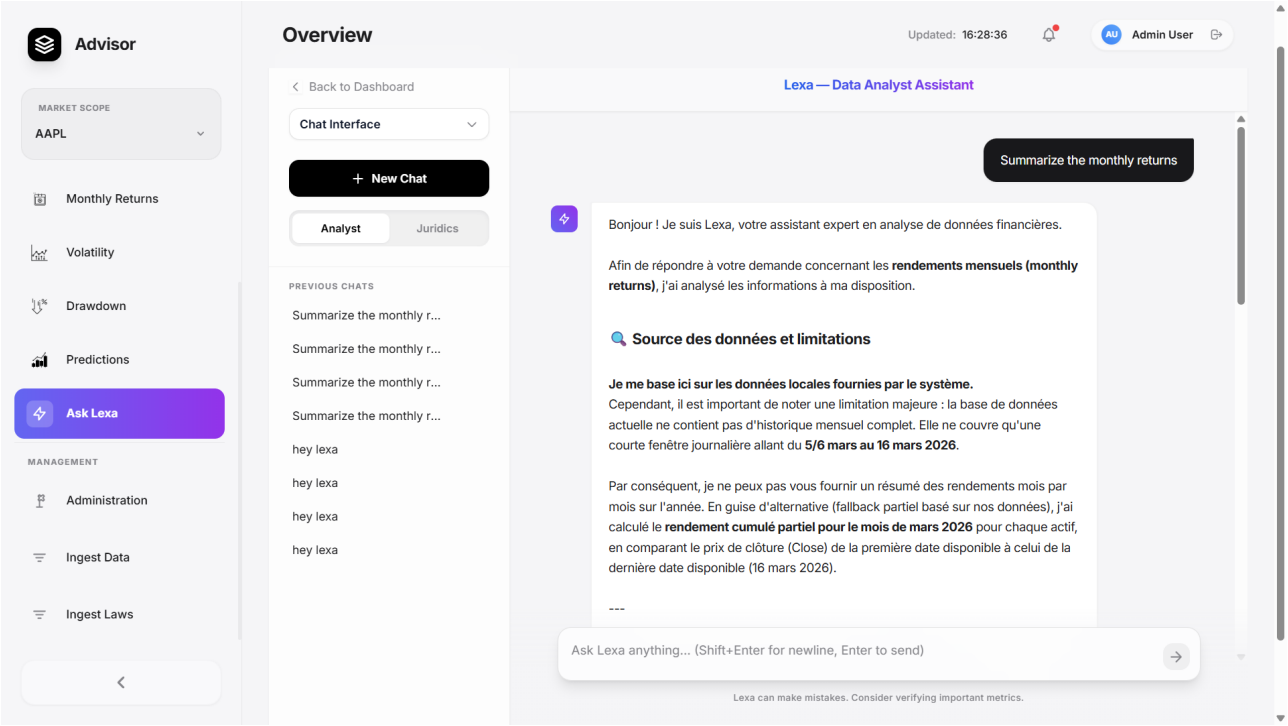


FIGURE 6.9 – Interface de LEXA en mode Analyste

6.9 Intégration du rapport de Data Profiling dans LEXA

Ce module met à disposition de l'utilisateur un rapport de profilage complet, généré directement depuis l'interface à partir des données réelles stockées en base. La figure ci-dessous présente le pipeline correspondant, dont l'accès est intégralement sécurisé par authentification JWT ; aucune route publique n'est exposée.

Le système s'appuie sur la table `gold.ohlcv_kpis`, qui centralise les données OHLCV enrichies des indicateurs calculés lors des sprints précédents. À partir de ces données, **YData Profiling** génère côté serveur un rapport HTML structuré, couvrant la distribution des variables, les corrélations, les valeurs manquantes et les anomalies détectées. Ce rapport est ensuite transmis au frontend Angular, qui le convertit en *blob URL* et l'injecte dans une `<iframe>` dédiée, garantissant qu'aucun contenu sensible n'est accessible sans token JWT valide. Un export au

format PDF est par ailleurs proposé via **WeasyPrint**, à l'attention des utilisateurs souhaitant archiver ou partager leurs analyses.

Si les données présentes en base s'avèrent insuffisantes ou absentes au moment de la génération, le backend renvoie un message d'erreur explicite accompagné du code HTTP approprié, évitant ainsi tout affichage silencieux ou comportement inattendu côté interface.

Ce module garantit que le profilage des données financières demeure accessible, sécurisé et exploitable directement depuis **LEXA**, sans recourir à un outil externe ni à une manipulation manuelle des données brutes.

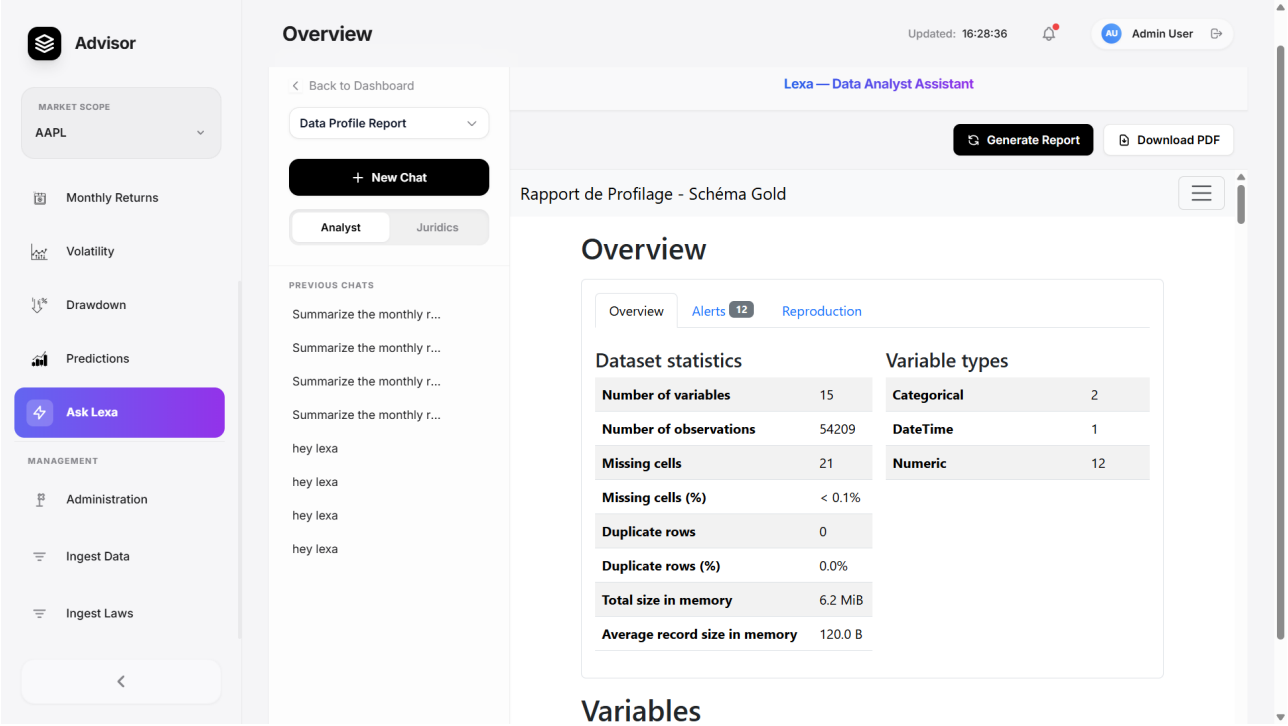


FIGURE 6.10 – Workflow du module Data Profiling dans LEXA

6.10 Conclusion

Ce chapitre a présenté le développement de **LEXA**, l'assistant intelligent de la plateforme **SmartFinance Advisor**. À travers la mise en place du pipeline d'ingestion des lois financières, le déploiement de ses deux modes de fonctionnement — **Juridique** et **Analyste** — et l'intégration du module de **Data Profiling**, LEXA s'affirme comme un outil d'aide à la décision à la fois fiable, sourcé et adapté au contexte financier tunisien. Les fondations ainsi posées ouvrent la voie à des évolutions futures, notamment l'enrichissement progressif du corpus juridique et l'intégration de données de marché en temps réel.

Au terme de ce projet de fin d'études, mené au sein de **VERMEG For Banking And Insurance Software**, nous pouvons affirmer que **SmartFinance Advisor** constitue une réponse concrète au manque d'outils d'analyse financière unifiés et adaptés au contexte tunisien.

Le travail a été structuré en quatre sprints complémentaires. Le **premier** a posé les fondations techniques avec un pipeline ETL basé sur l'architecture Medallion, traitant plus de **54 000 enregistrements** journaliers sur sept actifs sans aucune anomalie détectée. Le **deuxième** a sécurisé l'accès à la plateforme via un système complet de gestion des identités, avec hachage **bcrypt** et validation administrative. Le **troisième** a introduit la prévision des prix de clôture grâce à l'algorithme **Prophet**, atteignant une précision comprise entre **91 % et 98 %** selon les actifs. Le **quatrième**, enfin, a abouti au développement de **LEXA**, un assistant conversationnel opérant en modes *Analyste* et *Juridique*, s'appuyant sur une architecture **RAG** et l'API Gemini de Google.

Sur le plan technique, la stack retenue, **Angular**, **Flask**, **PostgreSQL** et **ChromaDB**, s'est révélée cohérente et bien adaptée aux exigences du projet. La méthodologie **Scrum**, combinée à **CRISP-DM** pour la partie data, a garanti un rythme de livraison régulier et maîtrisé.

Cette expérience nous a confrontés à des problématiques réelles : optimisation des insertions massives, intégration de modèles prédictifs dans une interface web, et conception d'un pipeline RAG fonctionnel. Elle a également renforcé notre capacité à travailler en équipe avec méthode et rigueur.

En perspective, plusieurs axes d'amélioration méritent d'être explorés : l'alimentation en données temps réel via WebSocket, l'enrichissement du corpus juridique de LEXA, et l'expérimentation de modèles prédictifs plus avancés tels que les **Transformers temporels** ou les architectures **LSTM**. Un déploiement cloud constituerait par ailleurs une étape naturelle vers la mise en production.

SmartFinance Advisor réunit ainsi, au sein d'une interface unique, l'ensemble des briques nécessaires pour passer de la donnée brute à la décision éclairée, posant les jalons d'un outil véritablement utile pour les professionnels du secteur financier tunisien.

BIBLIOGRAPHIE

Bibliographie

[1] Vermeg. Vermeg - About. Consulté le 27 Février 2026. url : https://www.vermeg.com/about-us/.

[2] Bloomberg Terminal. Consulté le 27 Février 2026. url : https://nyulaw.libguides.com/bloombergterminal.

[3] Investopedia. Qu'est-ce que FactSet ? Consulté le 27 Février 2026. url : https://www.investopedia.com/terms/f/factset.asp.

[4] Atlassian. Watefall Consulté le 27 Février 2026. url : https://www.atlassian.com/agile/project-management/waterfall-methodology.

[5] Atlassian. Agile Consulté le 27 Février 2026. url : https://www.atlassian.com/agile.

[6] Atlassian. Scrum. Consulté le 2 mars 2026. url : https://www.atlassian.com/fr/agile/scrum.

[7] IdeaScale. Qu'est-ce qu'un diagramme UML ? Guide complet avec exemples. Consulté le 2 mars 2026. url : https://ideascale.com/fr/blogues/uml-diagram-definition/#toc_What_Is_a_UML_Diagram.

[8] ScienceDirect. Consulté le 2 mars 2026. url : https://www.sciencedirect.com/topics/computer-science/layered-architecture.

[9] Data Science PM. Qu'est-ce que CRISP DM ?. Consulté le 2 mars 2026. url : https://www.datascience-pm.com/crisp-dm-2/.

[10] bility. Définition Visual Studio Code. Consulté le 9 mars 2026. url : https://bility.fr/definition-visual-studio-code/.

[11] Google. Qu'est-ce qu'Angular ? Consulté le 9 mars 2026. url : https://angular.dev/overview.

[12] TypeScript. Qu'est-ce que TypeScript ? Consulté le 9 mars 2026. url : https://www.w3schools.com/typescript/typescript_intro.php.

[13] Tailwind CSS. Qu'est-ce que Tailwind CSS ? Consulté le 9 mars 2026. url : https://tailwindcss.com/.

[14] Python. Qu'est-ce que Python ? Consulté le 9 mars 2026. url : https://docs.python.org/3/tutorial/index.html.

[15] Flask. Qu'est-ce que Flask ? Consulté le 9 mars 2026. url : https://flask.palletsprojects.com/en/stable/.

[16] PostgreSQL. Qu'est-ce que PostgreSQL ? Consulté le 9 mars 2026. url : https://www.postgresql.org/.

[17] Supabase. Qu'est-ce que Supabase ? Consulté le 9 mars 2026. url : https://supabase.com/.

[18] Postman. Postman. À propos de Postman. Consulté le 9 mars 2026. url : https://www.postman.com/company/about-postman/.

[19] GitHub. À propos de Git et Github. Consulté le 9 mars 2026. url : https://docs.github.com/fr/get-started/start-your-journey/about-github-and-git.

[20] Overleaf. À propos d'Overleaf. Consulté le 9 mars 2026. url : https://www.overleaf.com/about.

[21] W3Schools. Qu'est-ce que ChartJs. Consulté le 9 mars 2026. url : https://www.w3schools.com/ai/ai_chartjs.asp.

[22] Draw.io. à propos Draw.io. Consulté le 9 mars 2026. url : https://www.drawio.com/about.

[23] DataBricks. Qu'est-ce qu'une architecture en médaillon. Consulté le 10 avril 2026. url : https://www.databricks.com/blog/what-is-medallion-architecture.

[24] CoinAPI.io. Données OHLCV expliquées. Consulté le 10 avril 2026. url : https://www.coinapi.io/blog/understanding-ohlcv-in-market-data-analysis.

[25] YData-Profiling. Consulté le 10 avril 2026. url : https://docs.profiling.ydata.ai/latest/.

[26] JumpCloud. Définir Bcrypt . Consulté le 19 avril 2026. url : https://docs.profiling.ydata.ai/latest/.

[27] Talend. Définition du Machine Learning. Consulté le 2 mai 2026. url : https://www.talend.com/fr/resources/what-is-machine-learning/.

[28] ClicData. Qu'est-ce que l'apprentissage automatique dans le domaine de la BI ? Consulté le 2 mai 2026. url : https://www.clicdata.com/fr/guides/quest-ce-que-lapprentissage-automatique-dans-le-domaine-de-la-bi/.

[29] Prophet. Consulté le 2 mai 2026. url : https://facebook.github.io/prophet/.

[30] Aws.Amazon.com . Qu'est-ce que la génération à enrichissement contextuel ? Consulté le 22 mai 2026. url : https://aws.amazon.com/fr/what-is/retrieval-augmented-generation/.

[31] Ministère des Finances. Présentation. Consulté le 22 mai 2026. url : https://www.finances.gov.tn/fr/presentation.

[32] Teachup.com. Le Chunking. Consulté le 22 mai 2026. url : https://teachup.com/faciliter-la-memorisation-en-formation-digitale-grace-au-chunking.