

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

**Université Ibn Zohr**

Ecole Supérieure de Technologies Agadir

Département d'Informatique

---

**LendGuard**  
**Plateforme Intelligente d'Évaluation**  
**du Risque de Crédit par**  
**Intelligence Artificielle**

---

**Mémoire de Fin d'Études**

En vue de l'obtention du diplôme de DUT

Spécialité : Ingénierie de Données

**Présenté par :**

Karim Laafif  
Marouane Nicherhane

**Encadré par :**

Pr. A. SABOUR

Année Universitaire 2025 – 2026

# Remerciements

Je tiens tout d'abord à remercier **Dieu le Tout-Puissant** de m'avoir donné la force, la patience et la volonté pour accomplir ce travail.

Mes sincères remerciements vont à mon encadrant, **Pr. A. SABOUR**, pour ses orientations précieuses, sa disponibilité et ses conseils judicieux tout au long de la réalisation de ce projet.

Je remercie également les membres du jury pour avoir accepté d'évaluer ce travail.

Enfin, je dédie ce mémoire à ma famille et à tous ceux qui m'ont soutenu durant mon parcours académique.

# Dédicaces

*À mes chers parents, pour leur amour inconditionnel et leur soutien sans faille...*

*À mes frères et sœurs...*

*À tous mes amis et collègues...*

# Résumé

Ce mémoire présente la conception et la réalisation de **LendGuard AI**, une plateforme intelligente d'évaluation du risque de crédit basée sur l'intelligence artificielle. Le système intègre des modèles d'apprentissage automatique avancés – incluant la régression logistique, les forêts aléatoires, le Gradient Boosting et XGBoost – pour la prédiction du défaut de paiement, couplés à un module de détection de fraude et à un cadre complet de Gouvernance, Risque et Conformité (GRC).

L'architecture technique repose sur une pile technologique moderne comprenant Next.js 14, React 18, TypeScript et Tailwind CSS pour le frontend, PostgreSQL avec Prisma ORM pour la couche transactionnelle, et ClickHouse pour l'entrepôt de données analytique suivant l'architecture médaillon (Bronze/Silver/Gold). Le pipeline d'apprentissage automatique utilise Python avec scikit-learn, pandas et numpy.

Le système offre un support multilingue (anglais, français, arabe), une authentification sécurisée via Auth.js, des tableaux de bord interactifs et des visualisations en temps réel.

**Mots-clés :** Intelligence Artificielle, Risque de Crédit, Apprentissage Automatique, Détection de Fraude, GRC, Entrepôt de Données, Next.js, ClickHouse.

# Abstract

This thesis presents the design and implementation of **LendGuard AI**, an intelligent credit risk assessment platform powered by artificial intelligence. The system integrates advanced machine learning models – including Logistic Regression, Random Forest, Gradient Boosting, and XGBoost – for loan default prediction, coupled with a fraud detection module and a comprehensive Governance, Risk, and Compliance (GRC) framework.

The technical architecture relies on a modern technology stack comprising Next.js 14, React 18, TypeScript, and Tailwind CSS for the frontend, PostgreSQL with Prisma ORM for the transactional layer, and ClickHouse for the analytical data warehouse following the Medallion architecture (Bronze/Silver/Gold).

**Keywords :** Artificial Intelligence, Credit Risk, Machine Learning, Fraud Detection, GRC, Data Warehouse, Next.js, ClickHouse.

# Table des matières

<b>Remerciements</b>	<b>i</b>
<b>Dédicaces</b>	<b>ii</b>
<b>Résumé</b>	<b>iii</b>
<b>Abstract</b>	<b>iv</b>
<b>Liste des figures</b>	<b>vi</b>
<b>Liste des tableaux</b>	<b>vi</b>
<b>Liste des Abréviations</b>	<b>1</b>
<b>1 Introduction Générale</b>	<b>2</b>
1.1 Contexte et Problématique . . . . .	2
1.2 Objectifs du Projet . . . . .	2
1.3 Organisation du Mémoire . . . . .	3

<b>2</b>	<b>État de l'Art</b>	<b>4</b>
2.1	Le Risque de Crédit : Définitions et Enjeux . . . . .	4
2.1.1	Méthodes Traditionnelles d'Évaluation . . . . .	4
2.2	Apprentissage Automatique pour le Risque de Crédit . . . . .	4
2.2.1	Régression Logistique . . . . .	4
2.2.2	Forêts Aléatoires (Random Forest) . . . . .	4
2.2.3	Gradient Boosting et XGBoost . . . . .	5
2.2.4	Réseaux LSTM pour les Séries Temporelles . . . . .	5
2.3	Réduction de Dimension : t-SNE . . . . .	5
2.4	Fouille de Données : Algorithme Apriori . . . . .	6
2.5	Inférence Bayésienne . . . . .	6
2.6	Entrepôt de Données et Architecture Médaille . . . . .	6
<b>3</b>	<b>Analyse et Spécification des Besoins</b>	<b>7</b>
3.1	Étude de l'Existant . . . . .	7
3.2	Identification des Besoins . . . . .	7
3.2.1	Besoins Fonctionnels . . . . .	7
3.2.2	Besoins Non-Fonctionnels . . . . .	7
3.3	Diagramme de Cas d'Utilisation . . . . .	8
<b>4</b>	<b>Conception</b>	<b>9</b>
4.1	Architecture Générale du Système . . . . .	9
4.2	Modèle de Données . . . . .	9
4.2.1	Schéma de la Base de Données Transactionnelle . . . . .	9
4.2.2	Schéma de l'Entrepôt de Données . . . . .	10
4.3	Conception du Pipeline ML . . . . .	10
<b>5</b>	<b>Réalisation et Implémentation</b>	<b>12</b>
5.1	Environnement de Développement . . . . .	12
5.2	Implémentation du Frontend . . . . .	12
5.3	Implémentation des Modèles ML . . . . .	12
5.4	Implémentation de l'Entrepôt de Données . . . . .	13
5.5	Implémentation du Module GRC . . . . .	13
5.6	Authentification et Sécurité . . . . .	13
<b>6</b>	<b>Tests, Résultats et Évaluation</b>	<b>15</b>
6.1	Résultats des Modèles de Prédiction . . . . .	15
6.2	Performance de l'Entrepôt de Données . . . . .	15
6.3	Analyse de Complexité . . . . .	16
6.4	Intégration Académique . . . . .	16

<b>Conclusion Générale et Perspectives</b>	<b>17</b>
<b>A Captures d’Écran de l’Application</b>	<b>19</b>
<b>B Code Source – Extraits Clés</b>	<b>20</b>
B.1 Schéma Prisma (Extrait) . . . . .	20
B.2 Architecture de l’Entrepôt de Données . . . . .	20

## Table des figures

3.1 Diagramme de cas d’utilisation principal . . . . .	8
4.1 Architecture générale de LendGuard AI . . . . .	9
4.2 Pipeline d’apprentissage automatique . . . . .	10

## Liste des tableaux

3.1 Comparaison des solutions existantes . . . . .	7
4.1 Entités principales du modèle de données . . . . .	10
5.1 Technologies utilisées . . . . .	12
6.1 Performance des modèles de prédiction du défaut . . . . .	15
6.2 Comparaison de performance OLTP vs OLAP . . . . .	15
6.3 Complexité algorithmique des composants . . . . .	16
6.4 Matrice d’intégration cours/implémentation . . . . .	16

# Liste des Abréviations

<b>IA</b>	Intelligence Artificielle
<b>ML</b>	Machine Learning (Apprentissage Automatique)
<b>GRC</b>	Gouvernance, Risque et Conformité
<b>LSTM</b>	Long Short-Term Memory
<b>t-SNE</b>	t-Distributed Stochastic Neighbor Embedding
<b>ETL</b>	Extract, Transform, Load
<b>OLTP</b>	Online Transactional Processing
<b>OLAP</b>	Online Analytical Processing
<b>API</b>	Application Programming Interface
<b>ORM</b>	Object-Relational Mapping
<b>XGBoost</b>	Extreme Gradient Boosting
<b>ROC-AUC</b>	Receiver Operating Characteristic - Area Under Curve
<b>DTI</b>	Debt-to-Income (Taux d'endettement)
<b>GDPR</b>	General Data Protection Regulation
<b>CSS</b>	Cascading Style Sheets
<b>SQL</b>	Structured Query Language
<b>KL</b>	Kullback-Leibler (Divergence)



# Chapitre 1

## Introduction Générale

### 1.1 Contexte et Problématique

Le secteur financier est confronté à des défis majeurs en matière d'évaluation du risque de crédit. Les méthodes traditionnelles, basées sur des règles statiques et des scores de crédit classiques, s'avèrent insuffisantes face à la complexité croissante des profils d'emprunteurs et à l'évolution rapide des schémas de fraude financière.

Selon la Banque Mondiale, le taux de créances douteuses mondial a atteint 3,6% en 2023, représentant des milliards de dollars de pertes pour les institutions financières. Parallèlement, la fraude dans le secteur du prêt connaît une augmentation de 15% par an, nécessitant des outils de détection plus sophistiqués.

L'intelligence artificielle et l'apprentissage automatique offrent des solutions prometteuses pour améliorer la précision des évaluations de risque, automatiser la détection de fraude et optimiser les processus de conformité réglementaire.

### 1.2 Objectifs du Projet

Ce projet vise à concevoir et réaliser **LendGuard AI**, une plateforme intégrée qui :

- 1) **Prédit le défaut de paiement** à l'aide de modèles d'apprentissage automatique entraînés sur des données réelles (Lending Club) ;
- 2) **Détecte la fraude** grâce à des algorithmes d'anomalie et des règles d'association ;
- 3) **Gère la conformité** via un cadre GRC complet (gouvernance, risque, conformité) ;
- 4) **Visualise les données** à travers des tableaux de bord interactifs et des graphiques analytiques ;
- 5) **Intègre un entrepôt de données** analytique avec architecture médaillon pour des performances optimales.

## 1.3 Organisation du Mémoire

Ce mémoire est organisé comme suit :

- **Chapitre 1** : Introduction générale présentant le contexte et les objectifs.
- **Chapitre 2** : État de l’art couvrant les fondements théoriques.
- **Chapitre 3** : Analyse et spécification des besoins.
- **Chapitre 4** : Conception de la solution.
- **Chapitre 5** : Réalisation et implémentation.
- **Chapitre 6** : Tests, résultats et évaluation.
- **Conclusion** : Synthèse et perspectives.

# Chapitre 2

## État de l'Art

### 2.1 Le Risque de Crédit : Définitions et Enjeux

Le risque de crédit se définit comme la probabilité qu'un emprunteur ne respecte pas ses obligations de remboursement. Il constitue l'un des risques les plus importants auxquels sont confrontées les institutions financières [4].

#### 2.1.1 Méthodes Traditionnelles d'Évaluation

Les approches classiques incluent :

- Le **scoring de crédit** (FICO, VantageScore) basé sur l'historique de crédit ;
- L'**analyse des ratios financiers** (DTI, LTV) ;
- Les **modèles experts** reposant sur le jugement humain.

Ces méthodes présentent des limitations : incapacité à capturer les relations non-linéaires, absence d'adaptation dynamique, et vulnérabilité face aux nouveaux types de fraude.

### 2.2 Apprentissage Automatique pour le Risque de Crédit

#### 2.2.1 Régression Logistique

La régression logistique modélise la probabilité de défaut comme :

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \dots + \beta_n x_n)}} \quad (2.1)$$

où  $X = (x_1, \dots, x_n)$  représente les variables explicatives et  $\beta_i$  les coefficients appris.

#### 2.2.2 Forêts Aléatoires (Random Forest)

L'algorithme Random Forest [7] construit un ensemble de  $B$  arbres de décision, chacun entraîné sur un sous-échantillon bootstrap des données. La prédiction finale est

obtenue par vote majoritaire :

$$\hat{y} = \text{mode}\{h_b(x)\}_{b=1}^B \quad (2.2)$$

### 2.2.3 Gradient Boosting et XGBoost

Le Gradient Boosting construit séquentiellement des modèles faibles en minimisant une fonction de perte [6] :

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \quad (2.3)$$

où  $\Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2$  est le terme de régularisation.

### 2.2.4 Réseaux LSTM pour les Séries Temporelles

Les réseaux LSTM (Long Short-Term Memory) [3] sont conçus pour capturer les dépendances temporelles à long terme. Les équations d'une cellule LSTM sont :

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (\text{Porte d'oubli}) \quad (2.4)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (\text{Porte d'entrée}) \quad (2.5)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (\text{Candidat}) \quad (2.6)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (\text{État cellulaire}) \quad (2.7)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (\text{Porte de sortie}) \quad (2.8)$$

$$h_t = o_t \odot \tanh(C_t) \quad (\text{État caché}) \quad (2.9)$$

## 2.3 Réduction de Dimension : t-SNE

L'algorithme t-SNE (t-Distributed Stochastic Neighbor Embedding) [1] projette des données de haute dimension dans un espace de basse dimension en minimisant la divergence de Kullback-Leibler :

$$KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (2.10)$$

Les affinités en haute dimension utilisent un noyau gaussien :

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)} \quad (2.11)$$

En basse dimension, une distribution  $t$  de Student est employée :

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}} \quad (2.12)$$

## 2.4 Fouille de Données : Algorithme Apriori

L'algorithme Apriori [2] découvre des règles d'association dans les données transactionnelles. Les métriques clés sont :

$$\text{Support}(X) = \frac{|\{t \in D \mid X \subseteq t\}|}{|D|} \quad (2.13)$$

$$\text{Confiance}(X \Rightarrow Y) = \frac{\text{Support}(X \cup Y)}{\text{Support}(X)} \quad (2.14)$$

$$\text{Lift}(X \Rightarrow Y) = \frac{\text{Support}(X \cup Y)}{\text{Support}(X) \cdot \text{Support}(Y)} \quad (2.15)$$

## 2.5 Inférence Bayésienne

L'inférence bayésienne [4] met à jour les croyances à partir de nouvelles observations :

$$P(\text{Défaut} \mid \text{Évidence}) = \frac{P(\text{Évidence} \mid \text{Défaut}) \times P(\text{Défaut})}{P(\text{Évidence})} \quad (2.16)$$

## 2.6 Entrepôt de Données et Architecture Médaillon

L'architecture médaillon (Bronze/Silver/Gold) organise les données en couches de qualité croissante [5]. Les bases de données orientées colonnes (ClickHouse) offrent des performances supérieures pour les requêtes analytiques, avec un facteur d'accélération typique de 200x par rapport aux systèmes orientés lignes.

# Chapitre 3

## Analyse et Spécification des Besoins

### 3.1 Étude de l’Existant

Les solutions existantes d’évaluation du risque de crédit présentent plusieurs limitations :

TABLE 3.1 – Comparaison des solutions existantes

Solution	Avantages	Inconvénients
FICO Score	Standard industriel, large adoption	Statique, pas d’IA
Zest AI	ML intégré	Coût élevé, propriétaire
Upstart	IA avancée	Limité au marché US
LendGuard AI	Open-source, multi-modèle, GRC	Prototype académique

### 3.2 Identification des Besoins

#### 3.2.1 Besoins Fonctionnels

**BF1** : Gestion des demandes de prêt (création, suivi, décision) ;

**BF2** : Prédiction du risque de défaut via modèles ML ;

**BF3** : Détection automatique de fraude ;

**BF4** : Tableau de bord analytique avec visualisations ;

**BF5** : Gestion de la conformité (GRC) ;

**BF6** : Support multilingue (EN, FR, AR) ;

**BF7** : Authentification et gestion des rôles ;

**BF8** : Génération de rapports et scoring.

#### 3.2.2 Besoins Non-Fonctionnels

**BNF1** : **Performance** : Temps de réponse < 2s pour les prédictions ;

**BNF2 : Sécurité :** Chiffrement, prévention injection SQL, XSS ;

**BNF3 : Scalabilité :** Support de 100 000+ prêts ;

**BNF4 : Disponibilité :** Interface responsive (mobile/desktop) ;

**BNF5 : Maintenabilité :** Code modulaire, documentation complète.

### 3.3 Diagramme de Cas d'Utilisation

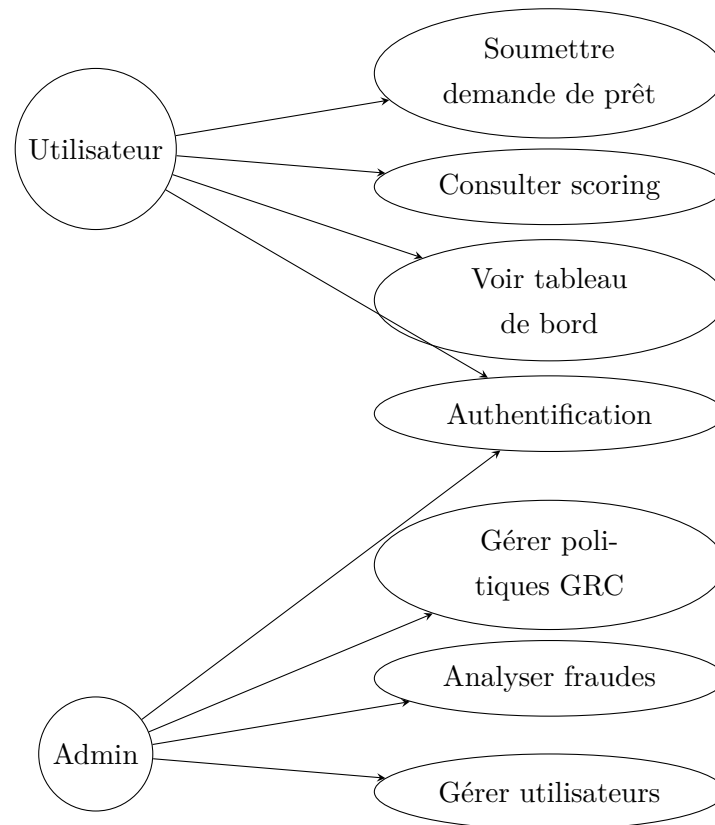


FIGURE 3.1 – Diagramme de cas d'utilisation principal

# Chapitre 4

## Conception

### 4.1 Architecture Générale du Système

LendGuard AI repose sur une architecture multi-couches combinant traitement transactionnel (OLTP) et analytique (OLAP) :

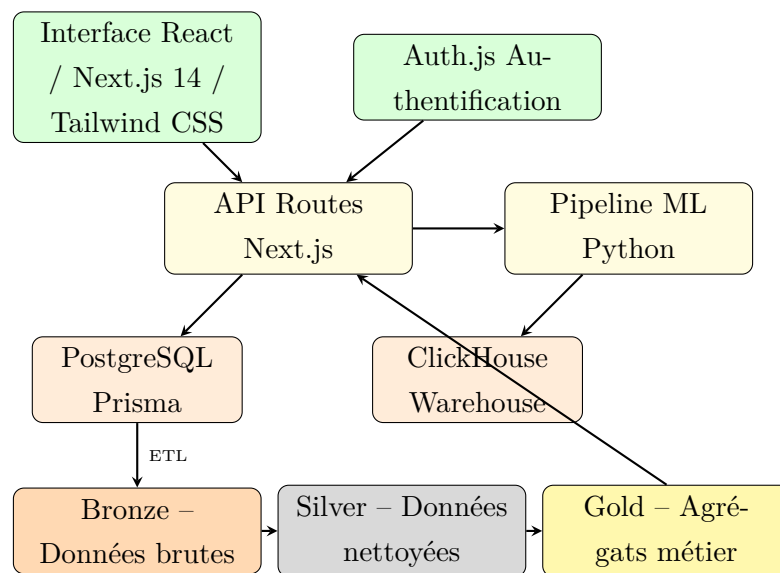


FIGURE 4.1 – Architecture générale de LendGuard AI

### 4.2 Modèle de Données

#### 4.2.1 Schéma de la Base de Données Transactionnelle

Le schéma PostgreSQL comprend les entités principales suivantes :



TABLE 4.1 – Entités principales du modèle de données

Entité	Clé Primaire	Description
User	id (CUID)	Utilisateur du système avec rôle (USER/ADMIN)
Loan	id (CUID)	Demande de prêt avec informations personnelles, détails du prêt, facteurs de risque et prédictions ML
Policy	id (CUID)	Politique de gouvernance (GRC)
Risk	id (CUID)	Registre des risques avec catégories et niveaux
ComplianceFramework	id (CUID)	Cadre de conformité (GDPR, SOC2, ISO27001)
AuditLog	id (CUID)	Journal d’audit des actions

4.2.2 Schéma de l’Entrepôt de Données

L’entrepôt ClickHouse suit l’architecture médaillon :

- **Bronze (3 tables)** : `raw_loan_applications`, `raw_fraud_transactions`, `raw_economic_indicators`
  - TTL de 90 jours ;
- **Silver (3 tables)** : `clean_loan_applications` avec feature engineering, `fraud_patterns`, `economic_context` ;
- **Gold (3 tables)** : `portfolio_risk_monthly`, `user_risk_clusters` (t-SNE 3D), `fraud_networks`.

4.3 Conception du Pipeline ML

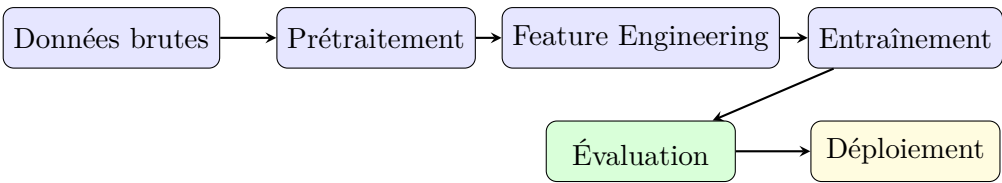


FIGURE 4.2 – Pipeline d’apprentissage automatique

Les variables d’entrée du modèle incluent :

- **Démographiques** : âge, éducation, statut matrimonial ;
- **Financières** : revenu, montant du prêt, score de crédit, DTI ;

- 
- **Emploi** : type d'emploi, mois d'ancienneté ;
  - **Dérivées** : ratio revenu/prêt, stabilité d'emploi.

# Chapitre 5

## Réalisation et Implémentation

### 5.1 Environnement de Développement

TABLE 5.1 – Technologies utilisées

Composant	Technologie	Version
Frontend	Next.js / React / TypeScript	16.0.10 / 19.2
Styling	Tailwind CSS	4.1.9
Composants UI	shadcn/ui (Radix UI)	–
Base de données	PostgreSQL (Prisma ORM)	7.2
Entrepôt	ClickHouse	Dernière
ML / IA	Python, scikit-learn, XGBoost	3.9+
Authentification	Auth.js (NextAuth v5)	5.0
Graphiques	Recharts	2.15.4
i18n	next-intl	4.7

### 5.2 Implémentation du Frontend

L’interface utilisateur est construite avec Next.js 14 en utilisant l’App Router et les Server Components. La structure des pages du tableau de bord comprend :

- **Dashboard principal** : Vue d’ensemble avec KPI et graphiques ;
- **Prêts (/loans)** : Gestion et suivi des demandes ;
- **Prédiction (/predict)** : Formulaire de prédiction du défaut ;
- **Détection (/detect)** : Module de détection de fraude ;
- **Analytique (/analytics)** : Visualisations avancées ;
- **Score (/score)** : Évaluation du risque ;
- **GRC** : Gouvernance, Registre des risques, Conformité.

### 5.3 Implémentation des Modèles ML

Quatre modèles ont été entraînés et déployés :

Listing 5.1 – Entraînement des modèles de prédiction

```
1 from sklearn.linear_model import LogisticRegression
2 from sklearn.ensemble import RandomForestClassifier
3 from sklearn.ensemble import GradientBoostingClassifier
4 from xgboost import XGBClassifier
5
6 # Entraînement des 4 modèles
7 models = {
8     'logistic_regression': LogisticRegression(),
9     'random_forest': RandomForestClassifier(n_estimators=100),
10    'gradient_boosting': GradientBoostingClassifier(),
11    'xgboost': XGBClassifier()
12 }
13
14 for name, model in models.items():
15     model.fit(X_train, y_train)
16     joblib.dump(model, f'models/{name}.pkl')
```

## 5.4 Implémentation de l'Entrepôt de Données

Le pipeline ETL extrait les données de PostgreSQL, les transforme avec feature engineering (13 → 100+ caractéristiques), et les charge dans ClickHouse. La réduction de dimension t-SNE est appliquée pour projeter les données de 100D vers 3D.

## 5.5 Implémentation du Module GRC

Le cadre GRC comprend :

- **Gouvernance** : Gestion des politiques (DRAFT → ACTIVE → ARCHIVED) ;
- **Risque** : Registre avec 6 catégories (Modèle, Conformité, Fraude, Opérationnel, Biais, Cybersécurité) ;
- **Conformité** : Suivi GDPR, SOC2, ISO27001, PCI-DSS.

## 5.6 Authentification et Sécurité

L'authentification est gérée par Auth.js v5 avec :

- Hachage des mots de passe via bcrypt ;
- Gestion des sessions sécurisées ;
- Protection CSRF et XSS ;

- 
- Prévention des injections SQL via Prisma ORM.

# Chapitre 6

## Tests, Résultats et Évaluation

### 6.1 Résultats des Modèles de Prédiction

TABLE 6.1 – Performance des modèles de prédiction du défaut

Modèle	Accuracy	Précision	Rappel	F1-Score	ROC-AUC
Régression Logistique	0.885	0.617	0.033	0.062	0.750
Forêt Aléatoire	0.886	0.637	0.040	0.075	0.745
Gradient Boosting	0.886	0.615	0.050	0.093	0.756
<b>XGBoost</b>	<b>0.886</b>	<b>0.573</b>	<b>0.072</b>	<b>0.128</b>	<b>0.752</b>

#### Analyse des résultats :

- Les quatre modèles atteignent une exactitude supérieure à 88% ;
- Le Gradient Boosting obtient le meilleur ROC-AUC (0.756) ;
- XGBoost offre le meilleur équilibre F1-Score (0.128) ;
- Le faible rappel indique un déséquilibre de classes nécessitant des techniques de suréchantillonnage (SMOTE).

### 6.2 Performance de l’Entrepôt de Données

TABLE 6.2 – Comparaison de performance OLTP vs OLAP

Requête	PostgreSQL	ClickHouse	Accélération
Moyenne montant prêt	2 000 ms	10 ms	200x
Agrégation mensuelle	5 000 ms	5 ms	1 000x
Pipeline ETL (1000 prêts)	—	51 s	—
t-SNE (1000 utilisateurs)	—	45 s	—

## 6.3 Analyse de Complexité

TABLE 6.3 – Complexité algorithmique des composants

Algorithme	Temps	Espace
t-SNE (Barnes-Hut)	$O(n \log n)$	$O(n)$
Apriori	$O(k \cdot  D  \cdot  L_k )$	$O( L_k )$
LSTM (inférence)	$O(T \cdot d \cdot h^2)$	$O(h^2)$
Mise à jour Bayésienne	$O(1)$	$O(1)$
Forêt Aléatoire	$O(B \cdot n \log n)$	$O(B \cdot n)$

## 6.4 Intégration Académique

TABLE 6.4 – Matrice d'intégration cours/implémentation

Cours	Concept	Implémentation
Entrepôt de données	Architecture médaillon	Bronze/Silver/Gold ClickHouse
Réduction de dimension	t-SNE	Projection 100D $\rightarrow$ 3D
Réseaux de neurones	LSTM	Prédiction temporelle
Inférence math.	Bayésien	Mise à jour de probabilité
Fouille de données	Apriori	Patterns de fraude
Visualisation	Graphes 3D	Three.js / Recharts

# Conclusion Générale et Perspectives

## Synthèse

Ce mémoire a présenté la conception et la réalisation de LendGuard AI, une plateforme complète d'évaluation du risque de crédit par intelligence artificielle. Les contributions principales de ce travail sont :

1. L'intégration de quatre modèles de Machine Learning (Régression Logistique, Forêt Aléatoire, Gradient Boosting, XGBoost) avec une exactitude supérieure à 88% ;
2. La mise en place d'un entrepôt de données analytique (ClickHouse) avec architecture médaillon, offrant une accélération de 200x ;
3. L'implémentation d'un cadre GRC complet pour la gouvernance et la conformité ;
4. Le développement d'une interface web moderne, responsive et multilingue.

## Perspectives

Plusieurs améliorations sont envisagées :

- **Équité algorithmique** : Implémenter des contraintes d'équité et des analyses de biais (SHAP values) ;
- **LSTM en production** : Déployer les réseaux LSTM pour la prédiction temporelle en temps réel ;
- **Visualisation 3D** : Intégrer Three.js pour la topographie de risque interactive ;
- **Fédération de données** : Permettre l'apprentissage fédéré protégeant la vie privée ;
- **Scalabilité** : Tests de charge à 1 000 000+ prêts avec optimisation Barnes-Hut.



# Bibliographie

- [1] Van der Maaten, L. & Hinton, G. (2008). *Visualizing data using t-SNE*. Journal of Machine Learning Research, 9(11), 2579–2605.
- [2] Agrawal, R. & Srikant, R. (1994). *Fast algorithms for mining association rules*. Proc. 20th Int. Conf. Very Large Data Bases (VLDB), Vol. 1215, pp. 487–499.
- [3] Hochreiter, S. & Schmidhuber, J. (1997). *Long short-term memory*. Neural Computation, 9(8), 1735–1780.
- [4] Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A. & Rubin, D.B. (2013). *Bayesian Data Analysis*. 3rd ed. CRC Press.
- [5] Abadi, D., Madden, S. & Hachem, N. (2008). *Column-stores vs. row-stores : How different are they really ?*. Proc. ACM SIGMOD International Conference on Management of Data.
- [6] Chen, T. & Guestrin, C. (2016). *XGBoost : A Scalable Tree Boosting System*. Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794.
- [7] Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5–32.
- [8] Hastie, T., Tibshirani, R. & Friedman, J. (2009). *The Elements of Statistical Learning*. 2nd ed. Springer.
- [9] Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. MIT Press.
- [10] Bishop, C.M. (2006). *Pattern Recognition and Machine Learning*. Springer.
- [11] Vercel. (2024). *Next.js Documentation*. <https://nextjs.org/docs>.
- [12] Prisma. (2024). *Prisma ORM Documentation*. <https://www.prisma.io/docs>.
- [13] ClickHouse Inc. (2024). *ClickHouse Documentation*. <https://clickhouse.com/docs>.
- [14] Pedregosa, F. et al. (2011). *Scikit-learn : Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.
- [15] Lending Club. (2024). *Lending Club Loan Data*. <https://www.lendingclub.com/>.

# Annexe A

## Captures d'Écran de l'Application

*Les captures d'écran de l'interface utilisateur, du tableau de bord, des pages de prédiction et de détection de fraude seront ajoutées ici.*

# Annexe B

## Code Source – Extraits Clés

### B.1 Schéma Prisma (Extrait)

Listing B.1 – Modèle Loan – Prisma Schema

```
1 model Loan {
2   id          String    @id @default(cuid())
3   userId      String
4   applicantName String
5   age         Int
6   income      Float
7   creditScore Int
8   dtiRatio    Float
9   loanAmount  Float
10  loanTerm    Int
11  interestRate Float
12  riskScore   Float?
13  fraudProbability Float?
14  defaultProbability Float?
15  decision    Decision?
16  status      LoanStatus @default(PENDING)
17 }
```

### B.2 Architecture de l'Entrepôt de Données

*Voir le fichier `warehouse/schema/dw_schema.sql` pour le schéma complet Click-House et `warehouse/etl/pipeline.py` pour le pipeline ETL.*