

REPUBLIQUE DU CAMEROUN

Paix – Travail – Patrie

\*\*\*\*\*  
MINISTÈRE DE  
L'ENSEIGNEMENT SUPERIEUR

\*\*\*\*\*  
ECOLE NATIONALE  
SUPERIEURE POLYTECHNIQUE DE  
DOUALA



REPUBLIC OF CAMEROON

Peace – Work – Fatherland

\*\*\*\*\*

MINISTRY OF HIGHER  
EDUCATION

\*\*\*\*\*  
NATIONAL ADVANCED  
SCHOOL OF ENGINEERING OF DOUALA



# INGENIERIE ET CONCEPTION LOGICIELLE

REALISATION D'UNE APPLICATION POUR LA  
MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE

Département : Génie informatique et  
télécommunications

Axe : Génie Logiciel – Groupe 14

Rédigé et présenté par :

- JEUDIEU TSAGUE AUDREY DANIELLE 22G00167
- FEUPA TCHAKOUNTE PIERRE NATHAN 22G00131
- KITIO MATCHOUSSA GERBEAU 22G00196
- MPACKO DARREN NGOLE 24G01115
- AKOUDJOU MBA NOËLLY CHRISTELA 22G00007

Sous la supervision de  
Dr. IHONOCK

Année académique  
2025-2026

## **Table des matières**

<b>Table des matières .....</b>	ii
Liste des figures.....	iv
Liste des tableaux .....	v
<b>INTRODUCTION GÉNÉRALE.....</b>	1
<b>PARTIE 1 : PRESENTATION DU PROJET .....</b>	3
I-    Contexte bancaire et enjeux de la gestion des risques.....	3
1-    Un environnement réglementaire et économique en mutation .....	3
2-    La typologie des risques bancaires .....	3
3-    Les enjeux stratégiques .....	4
II-    Problématique .....	4
1-    Les limites des systèmes actuels .....	4
2-    Question centrale .....	4
3-    L'ambition du projet .....	4
<b>PARTIE 2 : OBJECTIFS ET APPROCHE MVP .....</b>	5
I-    Objectif du projet.....	5
II-    Justification de l'approche MVP .....	5
1-    Définition et objectifs du MVP.....	5
2-    Les bénéfices clés du choix du MVP .....	6
<b>PARTIE 3 : ANALYSE FONCTIONNELLE ET NON FONCTIONNELLE.....</b>	7
I-    Fonctionnement du Système.....	7
1-    Architecture Fonctionnelle du Système .....	7
2-    Flux Opérationnel Central : Le Cycle de Vie de la Demande de Crédit.....	7
II-    Fonctionnement du Reporting et de l'Audit.....	8
III-    Modèle Statistique de Calcul de la Probabilité de Défaut (PD) .....	8
1-    Fondements Théoriques et Justification du Choix .....	8
2-    Mécanisme de Calcul : Le Scorecard Linéaire Pondéré .....	9
3-    Détermination et Justification des Pondérations .....	9
IV-    Besoins Fonctionnels .....	11
1-    Gestion du Référentiel Client et des Données d'Entrée .....	11

## MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE

---

2-	Audit, Reporting et Archivage .....	12
V-	Besoins Non Fonctionnels .....	12
1-	Performance et Efficacité.....	13
2-	Sécurité et Fiabilité (Contraintes MVP).....	13
3-	Maintenabilité et Évolutivité.....	13
4-	Ergonomie et Interface Utilisateur (UX) .....	14
PARTIE 4: CONCEPTION.....		15
I-	Normes et standards utilisés .....	15
4-	ISO/ IEC 25010.....	15
5-	Le cadre RGPD (Règlement Général pour la Protection des Données) .....	15
6-	Normes bancaires internationales .....	15
II-	Le langage de modélisation UML .....	16
1-	Présentation d'UML.....	16
2-	Outils de modélisation .....	16
III-	Étude fonctionnelle.....	17
1-	Le diagramme de cas d'utilisation .....	17
2-	Le diagramme de classe .....	20
3-	Diagramme d'activité.....	21
4-	Diagramme de séquence .....	23
IV-	Spécification technique.....	25
1-	Architecture Logicielle : Approche Monolithique Côté Client .....	25
2-	Modélisation et Implémentation du Risque .....	25
3-	Structure des Données Clés (LocalStorage).....	25
PARTIE 5 : IMPLEMENTATION ET TESTS.....		27
I-	Les langages de programmation et technologies .....	27
1-	Le langage de programmation TypeScript .....	27
2-	Le framework React.js.....	28
3-	Le framework Tailwind CSS .....	28
4-	La bibliothèque de visualisation : Recharts .....	29
5-	LocalStorage (API Web) .....	29
II-	L'environnement de travail .....	30
1-	Outils matériels.....	30
2-	Outils logiciels .....	31
III-	INTERFACES DU SYSTÈME .....	32
CONCLUSION ET PERSPECTIVES .....		36

## Liste des figures

<b>Figure 1:Logo Star UML .....</b>	16
<b>Figure 2: Logo de draw.io.....</b>	17
<b>Figure 3: Diagramme de cas d'utilisation du système.....</b>	18
<b>Figure 4: Diagramme de classe du système.....</b>	20
<b>Figure 5: Diagramme d'activité du système.....</b>	22
<b>Figure 6: Diagramme de séquence du système .....</b>	24
<b>Figure 7: Logo de TypeScript.....</b>	27
<b>Figure 8: Logo React.js .....</b>	28
<b>Figure 9: Logo Recharts .....</b>	29
<b>Figure 10: Outils et technologies utilisés .....</b>	30
<b>Figure 11: Tableau de bord de l'application.....</b>	32
<b>Figure 12: Gestion des clients .....</b>	32
<b>Figure 13: Cr éation d'un client .....</b>	33
<b>Figure 14: Demande de cr édit .....</b>	33
<b>Figure 15: Historique des d cisions .....</b>	34
<b>Figure 16: D cision finale .....</b>	34
<b>Figure 17: Tableau de bord apr s d cision finale.....</b>	35
<b>Figure 18: Historique apr s d cision finale.....</b>	35

**Liste des tableaux**

**Tableau 1: Détermination et justification des pondérations ..... 10**

## **INTRODUCTION GÉNÉRALE**

Dans le paysage financier mondial, la modélisation des risques de crédit est devenue le pilier central de la stabilité des institutions. Historiquement, cette discipline repose sur l'application de modèles statistiques et mathématiques visant à anticiper la probabilité qu'un emprunteur fasse défaut sur ses obligations. Avec l'évolution des marchés et le durcissement des régulations internationales, la capacité à quantifier le risque de manière objective est passée d'un simple avantage compétitif à une exigence réglementaire vitale pour la sécurité financière globale.

Cette nécessité de modélisation prend une dimension critique lorsqu'on l'applique au cas particulier de la gestion bancaire quotidienne. Pour une banque, le crédit représente à la fois sa principale source de revenus et son exposition la plus vulnérable. Au sein de cet environnement, les gestionnaires de risques sont confrontés à un défi majeur : traiter une masse croissante de données clients tout en garantissant des décisions rapides et une solvabilité optimale. La transition vers des outils numériques est donc devenue inévitable pour remplacer des méthodes d'analyse souvent manuelles, chronophages et sujettes à l'erreur humaine.

C'est précisément à ce stade que surgit le problème fondamental auquel font face de nombreuses structures : le manque d'outils agiles permettant une évaluation standardisée et transparente du risque. Cette situation engendre une problématique centrale : **Comment concevoir un système d'information capable d'automatiser le cycle d'évaluation du crédit et de fournir un score de risque fiable, tout en restant flexible face aux contraintes opérationnelles d'une banque ?**

Pour répondre à ce défi, nous avons développé une application innovante de modélisation de risques de crédit. Conçue pour offrir une précision accrue dans l'aide à la décision, notre solution s'aligne rigoureusement sur les standards bancaires internationaux. Elle intègre les principes des **accords de Bâle (Bâle II et III)** pour la gestion des fonds propres, respecte les exigences de la norme **IFRS 9** concernant l'estimation des pertes de crédit attendues, et intègre les protocoles **KYC (Know Your Customer)** pour une vérification stricte de l'identité et de la moralité des clients.

## MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE

---

Cette application se distingue par une architecture robuste à trois niveaux séparant la présentation, la logique métier et la persistance des données. Elle tire ses atouts de sa capacité à transformer des données complexes en une "Scorecard" claire, permettant de calculer instantanément la probabilité de défaut (PD). Sa réalisation repose sur une stack technologique moderne incluant **React.js** pour une interface utilisateur réactive, **TypeScript** pour la sécurité du typage des calculs financiers, et **Tailwind CSS** pour une ergonomie optimale.

Enfin, la réalisation de ce projet a suivi un processus rigoureux et structuré, débutant par une analyse approfondie des enjeux métiers et des besoins des analystes. Cette phase de diagnostic a été suivie d'une étape de conception utilisant le langage de modélisation UML pour structurer le système, avant d'aboutir à une phase d'implémentation itérative selon l'approche MVP (Minimum Viable Product). Ce document expose ainsi les différentes étapes de cette démarche, de la théorie statistique à la mise en service d'un outil fonctionnel et performant.

## **PARTIE 1 : PRESENTATION DU PROJET**

La stabilité du système financier repose sur la capacité des institutions bancaires à anticiper et à maîtriser les incertitudes inhérentes à leurs activités. Dans cette optique, cette première partie expose les fondements de notre étude en explorant le paysage bancaire actuel, les défis liés à la gestion des risques, ainsi que la problématique centrale qui a motivé le développement de notre solution.

### **I- Contexte bancaire et enjeux de la gestion des risques**

Le secteur bancaire évolue aujourd'hui dans un environnement complexe, marqué par une réglementation de plus en plus stricte et une volatilité économique croissante. La gestion des risques n'est plus seulement une obligation de conformité, mais un levier stratégique de performance.

#### **1- Un environnement réglementaire et économique en mutation**

Les banques doivent faire face à des exigences prudentielles rigoureuses (telles que les accords de Bâle III et Bâle IV). Ces normes imposent une surveillance accrue des fonds propres et une évaluation précise des risques d'exposition. Parallèlement, la digitalisation des services financiers multiplie les points de contact et, par extension, les sources potentielles de vulnérabilité.

#### **2- La typologie des risques bancaires**

Pour garantir la pérennité de l'institution, la gestion des risques doit couvrir plusieurs dimensions critiques :

- **Le risque de crédit** : lié à l'incapacité des contreparties à honorer leurs engagements.
- **Le risque de marché** : relatif aux pertes potentielles dues aux variations des cours, des taux ou des changes.
- **Le risque opérationnel** : découlant de défaillances internes, qu'elles soient humaines, techniques ou procédurales.
- **Le risque de liquidité** : concernant la difficulté pour la banque de faire face à ses échéances de paiement.

### **3- Les enjeux stratégiques**

L'enjeu majeur réside dans l'équilibre entre la rentabilité et la sécurité. Une gestion efficace des risques permet non seulement de protéger les actifs de la banque et de ses clients, mais aussi d'optimiser l'allocation du capital et d'améliorer la notation de crédit de l'institution sur les marchés internationaux.

### **II- Problématique**

Malgré l'existence de systèmes de contrôle, de nombreuses institutions bancaires se heurtent encore à des limites structurelles et technologiques qui freinent une gestion proactive des risques.

#### **1- Les limites des systèmes actuels**

Le constat métier révèle souvent une fragmentation des données de risque entre différents départements (le phénomène de "silos"). Cette dispersion empêche d'avoir une vision consolidée et en temps réel de l'exposition globale de la banque. De plus, de nombreux processus de reporting reposent encore sur des méthodes manuelles ou semi-automatisées, augmentant ainsi le risque d'erreur humaine et les délais de réaction.

#### **2- Question centrale**

Face à ces constats, une question fondamentale se pose : Comment concevoir et déployer une solution numérique capable de centraliser, d'analyser et de modéliser les risques bancaires afin de transformer des données brutes en indicateurs de décision fiables et immédiats ?

#### **3- L'ambition du projet**

C'est pour répondre à ce défi que notre projet a été initié. Il s'agit de proposer une plateforme capable de pallier les insuffisances du suivi manuel par l'automatisation des calculs d'indicateurs clés et la production de rapports de conformité dynamiques. L'objectif est de passer d'une gestion des risques "subie" (constat après sinistre) à une gestion "pilotée" (anticipation par la donnée).

En définitive, la présentation du contexte et de la problématique met en lumière l'impératif pour les banques de se doter d'outils technologiques avancés. Cette analyse préalable constitue le socle sur lequel nous allons bâtir notre cahier des charges technique et fonctionnel.

## **PARTIE 2 : OBJECTIFS ET APPROCHE MVP**

La définition des finalités d'un projet est une étape déterminante pour assurer l'alignement entre les attentes des parties prenantes et les développements techniques. Après avoir identifié les enjeux de la gestion des risques bancaires, cette seconde partie détaille les objectifs que nous nous sommes fixés, ainsi que la méthodologie agile adoptée pour le déploiement de la solution : l'approche MVP (Minimum Viable Product).

### **I- Objectif du projet**

L'ambition centrale de ce projet est de concevoir un outil d'aide à la décision capable d'optimiser le pilotage des risques au sein d'une institution financière. Cet objectif général se décline en plusieurs axes opérationnels :

- L'automatisation du suivi : Réduire les tâches manuelles de collecte et de traitement des données pour limiter le risque opérationnel lié aux erreurs humaines.
- La centralisation des indicateurs : Regrouper sur un tableau de bord unique les indicateurs clés de performance (KPI) et les indicateurs clés de risque (KRI).
- L'amélioration de la réactivité : Permettre une détection précoce des anomalies ou des dépassements de seuils critiques afin d'alerter les gestionnaires en temps réel.

En atteignant ces objectifs, la solution vise à renforcer la résilience de l'institution face aux crises potentielles. Toutefois, pour garantir une mise en œuvre efficace dans un environnement bancaire complexe, il a été nécessaire de choisir une approche de développement pragmatique, ce qui nous conduit à la justification du choix du MVP.

### **II- Justification de l'approche MVP**

Dans un contexte où les besoins métiers peuvent être vastes et complexes, l'adoption d'une approche par "Produit Minimum Viable" (MVP) s'est imposée comme la stratégie la plus pertinente pour ce projet.

#### **1- Définition et objectifs du MVP**

Le concept de MVP consiste à développer une version du produit qui ne comporte que les fonctionnalités essentielles permettant de satisfaire les besoins primaires des utilisateurs et de recueillir leurs retours. Loin d'être un produit inachevé, le MVP est une version optimisée qui vise à valider les hypothèses métiers les plus critiques dès les premières phases du cycle de développement.

## **2- Les bénéfices clés du choix du MVP**

Ce choix méthodologique repose sur trois piliers stratégiques qui garantissent la viabilité du projet :

### **a) Focalisation sur la valeur métier (Risque)**

L'approche MVP nous oblige à hiérarchiser les fonctionnalités. Dans le cadre de la gestion des risques bancaires, cela nous permet de nous concentrer prioritairement sur les modules à forte valeur ajoutée, tels que le calcul des expositions au risque de crédit ou la génération des alertes réglementaires. En éliminant le superflu, nous garantissons que chaque ligne de code répond à une exigence métier concrète.

### **b) Gestion des contraintes temporelles et de ressources**

Le secteur bancaire est soumis à des délais de mise en conformité souvent serrés. Le MVP permet de livrer une solution fonctionnelle dans un laps de temps réduit, évitant ainsi l'effet "tunnel" des longs projets de développement. Cette gestion optimisée des ressources (humaines et techniques) assure un meilleur contrôle des coûts du projet.

### **c) Itérations et potentiels d'évolution**

Le MVP constitue le socle d'un processus itératif. En livrant rapidement une première base, nous ouvrons la porte à des améliorations continues basées sur l'expérience réelle des gestionnaires de risques. Cette flexibilité permet d'ajuster l'outil aux évolutions futures des normes bancaires sans avoir à refondre l'intégralité du système.

L'adoption de l'approche MVP nous permet donc de concilier les exigences de rigueur du domaine bancaire avec l'agilité nécessaire au développement logiciel moderne. Cette vision stratégique étant désormais établie, il convient de traduire ces objectifs en spécifications concrètes à travers l'analyse des besoins et l'élaboration du cahier des charges fonctionnel.

## **PARTIE 3 : ANALYSE FONCTIONNELLE ET NON FONCTIONNELLE**

### **I- Fonctionnement du Système**

Cette partie du rapport décrit le processus opérationnel et les interactions entre les différents composants techniques et logiques qui constituent le cœur du Système de Gestion des Risques de Crédit (MVP).

#### **1- Architecture Fonctionnelle du Système**

L'architecture du MVP est conçue pour être simple et efficace. Elle suit le modèle logique des applications d'entreprise : une **Couche de Présentation** (l'interface utilisateur React et Vite), une **Couche Logique Métier** (le service TypeScript de modélisation) et une **Couche de Données** (le LocalStorage). Bien que l'implémentation soit consolidée dans une seule application côté client, cette séparation des préoccupations assure la maintenabilité et l'évolutivité future vers une architecture complète (avec un Back-end dédié).

#### **2- Flux Opérationnel Central : Le Cycle de Vie de la Demande de Crédit**

Le fonctionnement du système est centré sur la séquence d'événements qui transforme une nouvelle demande de prêt en une décision finale, garantissant rapidité et traçabilité.

- **Préparation du Dossier et Saisie des Données :** L'Agent de Crédit commence par sélectionner ou créer le profil du client dans l'application. Toutes les données nécessaires au calcul de la PD (revenu, charges, statut, etc.) ainsi que les paramètres du prêt (montant, durée) sont saisis via l'interface utilisateur.
- **Déclenchement du Moteur de Risque :** La soumission du formulaire déclenche l'exécution du service de modélisation (creditModel.ts). Ce service récupère les données et procède immédiatement au calcul de la mensualité estimée, indispensable pour évaluer l'impact du nouveau crédit.
- **Évaluation Quantitative (Le Scorecard) :** Le cœur du fonctionnement réside dans l'application du Scorecard Linéaire Pondéré. Le modèle calcule les cinq scores normalisés (Endettement, Crédit/Revenu, Stabilité Professionnelle, Ancienneté, Âge)

et les agrège en les multipliant par leur poids respectif. La somme de ces contributions donne la **Probabilité de Défaut (PD)** finale.

- **Prise de Décision Automatisée et Enregistrement :** La PD est transmise à la fonction de décision qui la compare aux seuils de 35% et 60%. La décision finale (Accepté, Analyse, ou Refus) est prise et affichée à l'Agent de Crédit. Simultanément, la demande complète, la PD et la décision sont enregistrées dans le LocalStorage pour des raisons de traçabilité et d'audit.

### **II- Fonctionnement du Reporting et de l'Audit**

Le système inclut des fonctionnalités essentielles pour le pilotage et la conformité, utilisées principalement par le Gestionnaire de Risque.

- **Tableau de Bord Dynamique :** Lors de l'accès à la section Reporting, l'application interroge et agrège en temps réel toutes les données de l'historique de demandes. Les résultats sont visualisés à l'aide de graphiques Recharts, permettant de suivre les indicateurs clés de performance (KPIs) comme la proportion de dossiers Acceptés, Refusés et en Analyse.
- **Fonction d'Export pour Audit :** Pour faciliter les processus d'audit ou l'analyse externe, le système met à disposition une fonction d'export. Celle-ci extrait toutes les informations pertinentes des demandes (y compris la PD et la décision) dans un format CSV, permettant la portabilité des données vers des outils d'analyse avancés.

### **III- Modèle Statistique de Calcul de la Probabilité de Défaut (PD)**

Cette section décrit la logique métier qui constitue le cœur du Système de Gestion des Risques de Crédit. Le modèle a été conçu pour attribuer un score de risque quantitatif à chaque demande, permettant une prise de décision rapide et justifiable.

#### **1- Fondements Théoriques et Justification du Choix**

Le modèle de risque est inspiré des pratiques établies en finance et en gestion des risques, tout en étant adapté aux contraintes d'un Minimum Viable Product (MVP).

- **Régression Logistique (Approche Simplifiée) :** Le principe fondamental est emprunté aux modèles de régression : la PD est une probabilité prédictive en attribuant

des poids aux variables explicatives. Cela garantit une méthode éprouvée pour évaluer le risque de défaut.

- **Conformité aux Accords de Bâle II/III :** La sélection des variables respecte la distinction exigée par la réglementation bancaire entre les facteurs **quantitatifs** (chiffres financiers, capacité de paiement) et les facteurs **qualitatifs** (stabilité de l'emprunteur).
- **Analyse Multi-Dimensionnelle du Risque :** Similaire aux systèmes de *scoring* reconnus (comme le Score FICO), le modèle combine plusieurs dimensions du risque (capacité financière, stabilité d'emploi, historique de vie) pour former un score unique et global.

### **2- Mécanisme de Calcul : Le Scorecard Linéaire Pondéré**

Le modèle utilise l'approche du **Scorecard Linéaire Pondéré**, caractérisée par sa transparence et sa facilité d'implémentation. La PD est calculée par la somme des scores de risque normalisés de cinq facteurs explicatifs, chacun étant multiplié par un poids défini.

La formule mathématique générale utilisée est :

$$PD = \sum_{i=1}^5 (\text{Score}_i \times \text{Poids}_i)$$

Chaque Score\_i est un nombre normalisé compris entre \$0\$ (risque minimal) et 1 (risque maximal).

### **3- Détermination et Justification des Pondérations**

Étant donné les contraintes d'un projet académique (absence de données historiques réelles), les pondérations ne sont pas obtenues par un calcul statistique complexe, mais par une approche **heuristique** (basée sur une simulation d'expertise métier).

Les pondérations sont **fixées** par le concepteur du système mais sont choisies de manière à refléter l'importance relative de chaque facteur dans une décision de crédit bancaire.

*Tableau 1: Détermination et justification des pondérations*

Facteur	Poids	Justification
<b>Ratio d'Endettement</b>	30%	<b>Le plus important.</b> Représente la capacité de remboursement immédiate.
<b>Ratio Crédit/Revenu</b>	25%	<b>Très important.</b> Représente l'impact de la <i>nouvelle</i> charge financière.
<b>Statut Professionnel</b>	20%	Stabilité et régularité des revenus futurs.
<b>Ancienneté</b>	15%	Proxy pour la stabilité professionnelle générale.
<b>Âge</b>	10%	Facteur secondaire influençant la phase de vie.

### **IV- Besoins Fonctionnels**

Ils définissent ce que le Système de Gestion des Risques de Crédit doit faire pour atteindre l'objectif d'automatisation de l'évaluation du risque. Ces exigences sont classées par domaine d'activité.

#### **1- Gestion du Référentiel Client et des Données d'Entrée**

Ce bloc concerne la préparation et la gestion des données de base nécessaires au modèle.

##### ➤ **Gestion Complète du Client (CRUD)**

- Le système doit permettre la **Création, la Consultation, la Modification et la Suppression (CRUD)** des fiches clients.
- Le formulaire de création client doit capturer toutes les variables explicatives du modèle : revenu mensuel net, charges mensuelles, statut professionnel, ancienneté professionnelle, et âge.

##### ➤ **Saisie des Paramètres de Prêt**

- Le système doit fournir un formulaire pour associer une demande de crédit à un client existant.
- Ce formulaire doit permettre la saisie des deux paramètres financiers variables de la demande : le **Montant du Crédit** et la **Durée du Crédit** (en mois).

##### - **Cœur du Système : Évaluation du Risque**

Ce bloc représente la fonctionnalité principale du moteur de risque.

##### ➤ **Calcul de la Probabilité de Défaut (PD)**

- Lors de la soumission d'une demande, le système doit appeler le module de modélisation (creditModel.ts).
- Ce module doit exécuter le **Scorecard Linéaire Pondéré** en calculant et en sommant les contributions pondérées des cinq facteurs (Endettement, Crédit/Revenu, Statut, Ancienneté, Âge) pour obtenir une PD comprise entre 0 et 1.

##### ➤ **Détermination de la Décision Automatique**

- Le système doit comparer la PD calculée aux seuils définis (35% et 60%) pour attribuer immédiatement l'un des trois statuts de décision :
  - **Accepté** ( $PD < 35\%$ )
  - **Analyse** ( $35\% < PD < 60\%$ )
  - **Refus** ( $PD \geq 60\%$  à 100%)

➤ **Justification et Interprétabilité du Résultat**

- Le système doit afficher la **PD et la Décision finale** de manière claire.
- Il doit également afficher les scores individuels et leurs contributions (par exemple : le score du Ratio d'Endettement) pour justifier pourquoi la décision a été prise (répondant au besoin d'interprétabilité).

## **2- Audit, Reporting et Archivage**

Ces fonctions sont essentielles pour le suivi post-décision et la conformité.

➤ **Archivage des Décisions (Traçabilité)**

- Toutes les demandes soumises, y compris la PD, la décision et la date, doivent être stockées dans une base de données simulée (LocalStorage) pour garantir la traçabilité et l'auditabilité.

➤ **Tableau de Bord et Reporting**

- Le système doit offrir une interface de reporting visuelle (Tableau de Bord) au Gestionnaire de Risque.
- Celle-ci doit afficher des indicateurs clés (KPIs) tels que le nombre total de demandes et la répartition en pourcentage des décisions (Accepté vs. Refusé vs. Analyse).

➤ **Export des Données d'Historique**

- Le système doit permettre l'export de l'historique complet des demandes et des décisions au format **CSV** pour faciliter les audits externes et les analyses statistiques ultérieures.

## **V- Besoins Non Fonctionnels**

Ils définissent les critères de performance, de qualité et les contraintes techniques qui encadrent le développement et le fonctionnement du Système de Gestion des Risques de Crédit.

### **1- Performance et Efficacité**

- **Temps de Réponse (Latence)** : Le calcul de la Probabilité de Défaut (PD) et l'affichage de la décision associée doivent être **instantanés** (temps de réponse inférieur à 1 seconde). Ceci est crucial pour fluidifier le processus de décision de l'Agent de Crédit.
- **Scalabilité du Modèle** : Bien que l'MVP utilise un jeu de données limité, la logique du modèle doit être conçue pour gérer un **volume élevé** de calculs de PD sans dégradation de performance.

### **2- Sécurité et Fiabilité (Contraintes MVP)**

- **Intégrité des Données** : L'intégrité des données clients (revenu, statut) et des données de décision (PD, statut) doit être garantie. L'utilisation de **TypeScript** contribue à la fiabilité en appliquant un typage fort à la logique métier.
- **Fiabilité du Modèle** : Le modèle de PD doit garantir une cohérence parfaite : pour les mêmes données d'entrée, la PD et la décision doivent être **identiques** à chaque exécution.

### **3- Maintenabilité et Évolutivité**

- **Maintenabilité du Code** : Le code doit être structuré de manière modulaire, en respectant la séparation des préoccupations (logique métier dans creditModel.ts, interface utilisateur, persistance). Cela facilite la correction des bugs et l'ajout futur de fonctionnalités.
- **Évolutivité Technique** : L'architecture doit permettre une migration aisée :
  - Le remplacement futur du stockage LocalStorage par une base de données d'entreprise sans affecter la logique Front-end.
  - L'intégration ultérieure d'un modèle statistique plus sophistiqué (comme le modèle de *Machine Learning* réel) sans nécessiter la refonte de l'interface utilisateur.
- **Documentabilité** : Toutes les formules critiques, les seuils de risque et les pondérations doivent être clairement documentés dans le code source et le rapport (comme dans la section Modèle Statistique).

#### **4- Ergonomie et Interface Utilisateur (UX)**

- **Ergonomie de Saisie :** Les formulaires de saisie client et de demande de crédit doivent être intuitifs et nécessiter un minimum de clics pour maximiser la productivité de l'Agent de Crédit.
- **Clarté de la Décision :** L'affichage de la décision (Accepté, Analyse, Refus) doit être immédiat et visuellement distinct (par exemple, par un code couleur) pour éviter toute ambiguïté.
- **Accessibilité du Reporting :** Le Tableau de Bord doit offrir une visualisation claire des indicateurs clés (KPIs) pour le Gestionnaire de Risque.

## **PARTIE 4 : CONCEPTION**

La partie de conception constitue le pont entre l'analyse fonctionnelle et la réalisation technique. Elle définit un cadre réglementaire et les structures logiques permettant d'assurer la fiabilité du système de scoring.

### **I- Normes et standards utilisés**

L'application de modélisation de risques de crédit ne peut être isolée des cadres normatifs stricts qui régissent l'ingénierie logicielle et le secteur bancaire.

#### **4- ISO/ IEC 25010**

Nous adoptons cette norme internationale pour garantir la qualité du produit logiciel. Elle encadre nos développements sur plusieurs axes clés, notamment :

- La **fiabilité** : Capacité du système à fournir des scores exacts et reproductibles.
- L'**utilisabilité** : Facilité pour l'agent de crédit d'opérer les saisies sans erreur.
- La **sécurité** : Protection contre les accès non autorisés aux données financières.

#### **5- Le cadre RGPD (Règlement Général pour la Protection des Données)**

Bien que le RGPD soit européen, nous nous alignons sur ses principes et sur la loi n°2010/012 relative à la cybersécurité et la cybercriminalité au Cameroun pour la protection des données personnelles :

- **Minimisation des données** : Seules les données strictement nécessaires au calcul du risque sont collectées.
- **Droit à l'information** : L'algorithme doit pouvoir justifier un refus (principe de transparence).

#### **6- Normes bancaires internationales**

Le système intègre nativement les exigences de régulation prudentielle :

- **Accords de Bâle (Bâle II & Bâle III)** : Notre modèle aide à déterminer le capital réglementaire en évaluant le risque de défaut.

- **Normes IFRS 9 :** L'application permet d'estimer les « Pertes de Crédit Attendues » (ECL) dès l'octroi, passant d'un modèle de pertes subies à un modèle de pertes prévisibles.
- **Le dispositif KYC (Know Your Customer) :** Le workflow impose la documentation complète de l'identité et de l'origine des revenus du client avant toute simulation.

## **II- Le langage de modélisation UML**

### **1- Présentation d'UML**

Le langage UML (Unified Modeling Language) ou en français Langage de Modélisation Unifié, a été pensé pour être un langage de modélisation visuelle commun, et riche sémantiquement et syntaxiquement. Il est destiné à l'architecture, la conception et la mise en œuvre de systèmes logiciels complexes aussi bien par leur structure que par leur comportement.

Il facilite la compréhension du fonctionnement globale de l'application, aussi bien pour les concepteurs que pour les utilisateurs non techniques.

Plusieurs types de diagrammes ont été réalisés à l'aide d'UML pour notre plateforme, notamment les diagrammes de cas d'utilisation pour illustrer les interactions entre les utilisateurs et le système, le diagramme de classes pour représenter les entités ainsi que les relations entre elles, ainsi que les diagrammes de séquence et d'activités pour modéliser les flux d'échange et le déroulement logique des actions.



***Figure 1:Logo Star UML***

StarUML est un outil de modélisation logiciel utilisé pour concevoir et documenter les systèmes logiciels en utilisant le langage de modélisation unifié (UML).

### **2- Outils de modélisation**

Afin de réaliser nos différents diagrammes, nous avons utilisé le logiciel Draw.io. C'est un logiciel en ligne gratuit qui permet de réaliser facilement des diagrammes visuels tels que des diagrammes UML, organigrammes, schémas de processus ou diagrammes de flux. Il

est accessible via un navigateur, sans installation ; il propose une interface intuitive par glisser-déposer et est donc idéal pour réaliser des dessins techniques ou visuels.

Nous avons ci-dessous l'icône du logiciel Draw.io



*Figure 2: Logo de draw.io*

### III- Étude fonctionnelle

L'étude fonctionnelle dans cette partie permettra de faire ressortir les diagrammes obtenus de notre analyse. Pour notre étude fonctionnelle, nous avons réalisé le diagramme de cas d'utilisation pour l'analyse des besoins, le diagramme de classes pour l'analyse du domaine et enfin les diagrammes de séquence et d'activités pour l'analyse applicative.

#### 1- Le diagramme de cas d'utilisation

Le diagramme de cas d'utilisation est une représentation graphique des interactions entre les utilisateurs et un système informatique. Ce diagramme est constitué de plusieurs éléments qui entrent dans sa réalisation : les acteurs, les cas d'utilisation et les relations.

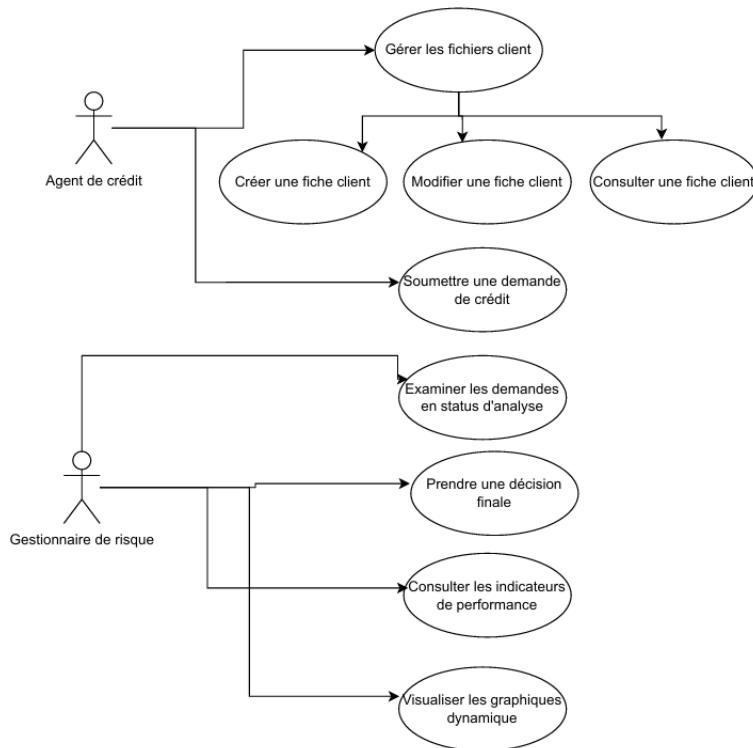
##### a) Identification des acteurs

Les acteurs du système représentent des rôles joués par des entités externes du système et qui interagissent directement avec ce dernier.

Les acteurs de notre système sont :

- Le **Client** : Fournisseur des données sources, il interagit avec le système via l'agent pour obtenir sa simulation.
- L'**Agent de risque** : Utilisateur opérationnel en agence, chargé de la collecte, de la saisie et de l'initialisation du scoring.
- Le **Gestionnaire de risque** : Expert décisionnaire qui analyse les scores, traite les dossiers complexes et valide l'octroi final.
- L'**Administrateur** : Responsable technique garant de la sécurité, des accès et de la maintenance du moteur de calcul.

Le diagramme de cas d'utilisation de notre système est le suivant :



**Figure 3: Diagramme de cas d'utilisation du système**

#### b) Description détaillée des cas d'utilisation

- **L'Agent de Crédit :** C'est l'utilisateur opérationnel principal de l'application ; il est l'intermédiaire entre le demandeur et le moteur de décision. Ce dernier a la possibilité de :
  - **Gérer les fiches clients (CRUD)** : L'agent crée, modifie ou consulte les informations financières et personnelles de l'emprunteur (revenus, charges, statut) pour constituer le dossier.
  - **Soumettre une demande de crédit** : L'agent saisit les paramètres du prêt (montant et durée) et lance l'évaluation pour un client sélectionné.
  - **Obtenir une décision instantanée** : Dès la soumission, l'agent reçoit le verdict automatique du système (Accepté, Refusé ou Analyse) basé sur le calcul de la Probabilité de Défaut (PD).
  - **Consulter les justifications du score** : L'agent peut lire le détail des points attribués à chaque facteur (endettement, ancienneté, etc.) pour expliquer le résultat au client.

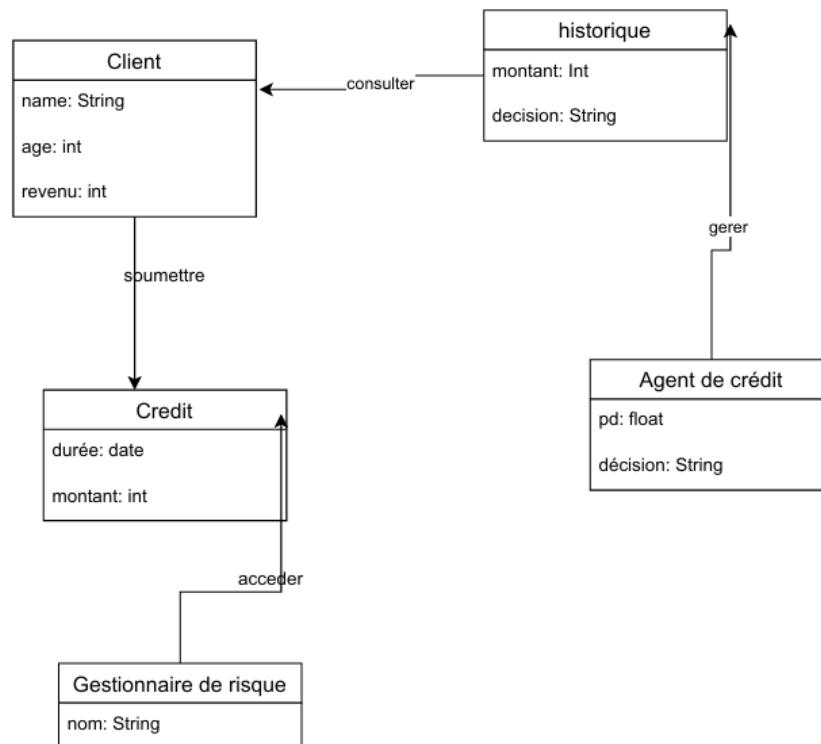
- **Travailler en mode local :** Grâce au stockage interne (LocalStorage), l'agent peut gérer ses dossiers et consulter l'historique sans dépendre d'un serveur externe complexe.
- Le **gestionnaire de risque** : Il supervise la politique de crédit et accompagne les décisions complexes. Pour qu'un utilisateur puisse accéder au tableau de bord, soumettre une demande ou extraire des données sensibles, celui-ci doit impérativement s'identifier via son profil sécurisé.
- Ses prérogatives sont :
- **Analyser les dossiers litigieux** : Le gestionnaire examine les demandes ayant reçu le statut "Analyse" pour prendre une décision finale basée sur son expertise.
  - **Suivre les statistiques globales (Dashboard)** : Il consulte les indicateurs de performance (taux d'acceptation, volume de crédit) via des graphiques dynamiques pour piloter la stratégie de la banque.
- L'**Administrateur (Organisateur)** : Il gère l'ensemble de la plateforme pour garantir la fiabilité du modèle et la sécurité des données. Il se charge de :
- **Paramétriser les seuils de risque** : L'administrateur ajuste les paliers de décision (ex: modifier le seuil de 35% à 40%) pour adapter la rigueur du système à la stratégie actuelle.
  - **Gérer les utilisateurs** : Il crée, modifie ou réinitialise les accès des agents de crédit et des gestionnaires de risque.
  - **Exporter l'historique pour audit** : Il génère des fichiers CSV regroupant l'intégralité des décisions prises par le système pour les transmettre aux autorités de régulation (conformité Bâle II/III).
  - **Maintenir le référentiel** : Il s'assure que les catégories de statuts professionnels et les barèmes de calcul sont à jour et pertinents.

## 2- Le diagramme de classe

Dans l'analyse des besoins, nous avons réalisé les diagrammes de cas d'utilisation qui permettent de faire une étude dynamique en décrivant les différentes interactions. Nous allons à présent faire une étude statique du système. Le diagramme de classe consistera en la description du système en montrant les classes, les attributs, les opérations et les relations entre les entités du système.

Le diagramme de classe structure les données et définit les relations entre les différentes entités de l'application de scoring.

Le diagramme de classe de notre système est le suivant :



**Figure 4: Diagramme de classe du système**

### a) Description du diagramme de classe

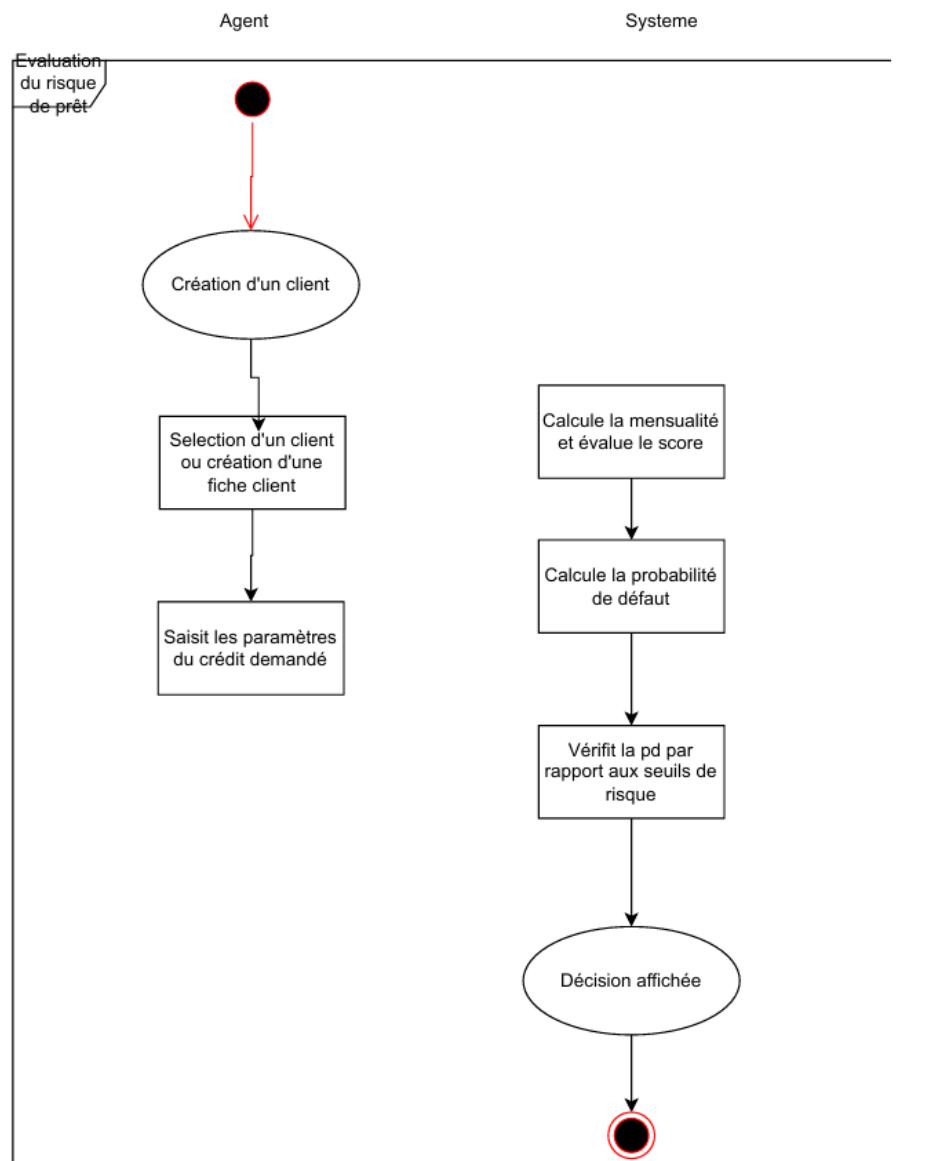
- Un **Client** peut soumettre plusieurs **Demandes de Crédit**, chacune correspondant à un dossier de prêt spécifique avec son propre montant et sa propre durée.
- Chaque **Demande de Crédit** est rattachée à un client unique et contient les résultats calculés par le système (Probabilité de Défaut et Décision finale).
- Un **Agent de Crédit** a la possibilité de gérer la liste des **Clients** ainsi que l'historique des **Demandes de Crédit** qu'il a initiées.
- Le **Gestionnaire de Risque** peut accéder à l'ensemble des **Demandes de Crédit** du portefeuille pour consulter les statistiques et valider les dossiers complexes.
- Une **Demande de Crédit** est étroitement liée aux attributs financiers du client (revenus, charges, statut) pour permettre au moteur de calcul de générer un score de risque précis.

### 3- Diagramme d'activité

Le diagramme d'activités est utilisé pour représenter le déroulement logique d'un processus ou d'une activité, étape par étape. Il décrit le flux de travail, les décisions, les boucles et les actions successives représentées dans le système, du début à la fin. Il est particulièrement utile pour représenter les scénarios de traitement, comme dans notre cas, le processus de gestion d'une demande de crédit en mettant en évidence les différentes étapes.

### a) Diagramme de cas d'utilisation du système

Le diagramme d'activité de notre système est le suivant :



**Figure 5: Diagramme d'activité du système**

#### b) Description du diagramme d'activité

Ce diagramme nous présente les différentes étapes du processus lorsqu'un agent de crédit souhaite évaluer le risque d'un dossier de prêt dans l'application.

En effet, pour réaliser une évaluation, l'agent doit tout d'abord sélectionner un client au sein du référentiel ou créer une nouvelle fiche en saisissant ses données financières (revenus, charges) et professionnelles. Une fois le profil client validé, l'agent saisit les paramètres du crédit souhaité, à savoir le **montant** et la **durée**.

Dès la soumission de la demande, le système lance automatiquement le moteur de risque. Le processus commence par le calcul de la mensualité théorique, puis l'application

évalue successivement les cinq scores de risque normalisés : l'endettement, le ratio crédit/revenu, la stabilité du statut professionnel, l'ancienneté et la tranche d'âge. Le système agrège ensuite ces scores en appliquant les pondérations définies pour obtenir la **Probabilité de Défaut (PD) globale**.

Une fois la **PD** calculée, le système effectue un test de validité par rapport aux seuils de risque :

- Si la **PD** est inférieure à **35%**, le système marque automatiquement le crédit comme "**Accepté**".
- Si la **PD** est supérieure ou égale à **60%**, l'application rejette le dossier avec le statut "**Refus**".
- Si le score se situe entre ces deux paliers, le dossier est orienté vers un statut "**Analyse**" pour une expertise humaine.

À la fin de cette évaluation, le système affiche instantanément la décision ainsi qu'une justification détaillée des scores obtenus. Simultanément, le dossier est archivé de manière persistante dans le stockage local. Enfin, l'utilisateur peut accéder au tableau de bord pour visualiser les statistiques de performance globale ou exporter l'historique complet des décisions effectuées.

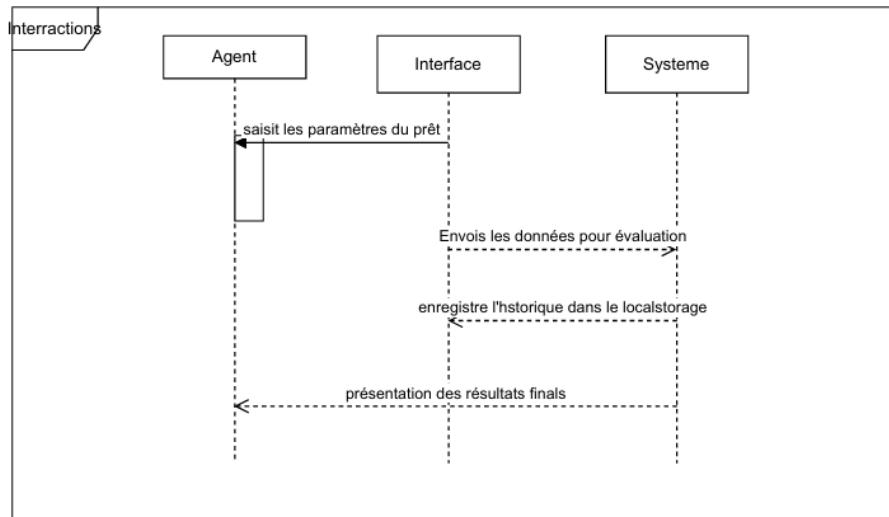
### **4- Diagramme de séquence**

Le diagramme de séquence est un outil de modélisation dynamique issu de la famille UML, qui se concentre sur la chronologie des échanges de messages entre les différents objets ou acteurs du système pour une fonctionnalité précise. Contrairement aux diagrammes statiques, il permet de visualiser « l'histoire » d'un cas d'utilisation, en montrant l'ordre exact dans lequel les informations circulent, du client vers l'interface, puis de l'interface vers le moteur de calcul et le stockage.

Dans le cadre de l'étude fonctionnelle de notre application de risque de crédit, son importance est capitale. Il permet de détailler avec une précision chirurgicale le flux opérationnel d'une demande de prêt : de la saisie des revenus par l'agent de crédit jusqu'à la réception de la décision finale générée par l'algorithme. En isolant ces interactions, il aide à identifier d'éventuels goulots d'étranglement ou des failles dans la logique de communication, garantissant ainsi que

chaque étape du processus bancaire est correctement orchestrée avant même le début de l'implémentation.

Le diagramme de séquence de notre système est :



**Figure 6: Diagramme de séquence du système**

Ce diagramme illustre la séquence des interactions entre l'Agent de Crédit, l'interface utilisateur et le moteur de risque interne lors de l'évaluation d'un dossier.

Concrètement, l'Agent commence par sélectionner un client et saisit les paramètres du prêt, tels que le montant et la durée, via le formulaire de l'application. Une fois ces informations soumises, l'interface transmet les données au moteur de risque (creditModel.ts) qui lance immédiatement la procédure d'évaluation.

Le moteur de risque procède alors à une série de calculs internes : il détermine d'abord la mensualité théorique, puis sollicite les fonctions de scoring pour évaluer chaque variable (endettement, stabilité, âge). Il agrège ces résultats pour produire la **Probabilité de Défaut (PD)** finale. Le système compare ensuite ce score aux seuils de risque pour générer la décision automatique (Accepté, Analyse ou Refus).

Une fois la décision arrêtée, le système envoie une requête de stockage au **LocalStorage** pour enregistrer l'historique complet de la demande, incluant les scores détaillés et la décision. Enfin, l'application présente à l'Agent de Crédit le résultat final accompagné d'une justification transparente pour chaque facteur de risque, lui permettant d'expliquer la décision au client ou de transmettre le dossier pour une analyse approfondie.

## IV- Spécification technique

### 1- Architecture Logicielle : Approche Monolithique Côté Client

Dans un souci de simplicité et de rapidité d'exécution pour l'MVP, une architecture **mono-applicative (Front-end Only)** a été retenue. La logique métier, la persistance des données et la présentation sont regroupées dans une seule application, simulant la communication avec un service Back-end par une abstraction en TypeScript.

- **Front-end (Client Web)** : Couche de présentation et d'interaction utilisateur.
- **Logique Métier/Modèle** : Fonctionnalité de calcul de la PD et de décision.

### 2- Modélisation et Implémentation du Risque

Le cœur du système est implémenté dans un module dédié, assurant la séparation des préoccupations :

- **Module** : /utils/creditModel.ts (ou un service similaire).
- **Logique de Calcul** : Implémentation du **Scorecard Linéaire Pondéré** qui prend les données client en entrée et produit la Probabilité de Défaut (PD) en sortie.
- **Décision** : La fonction determineDecision() applique les seuils de risque (35% et 60%) pour classer la demande en Accepté, Analyse ou Refus.

### 3- Structure des Données Clés (LocalStorage)

Les données sont stockées sous forme d'objets JSON dans le LocalStorage, simulant les tables d'une base de données relationnelle.

Entité	Champs Clés	Type de Données
Client	id, nom, prénom, revenu, charges, statutProfessionnel, anciennete_emploi, âge	number, string, number

## MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE

---

Entité	Champs Clés	Type de Données
Demande	id, id_client (Relation), montant, duree, PD, decision, date	number, number, string

## **PARTIE 5 : IMPLEMENTATION ET TESTS**

La phase d'implémentation constitue l'étape de concrétisation où les modèles théoriques et les diagrammes de conception sont traduits en instructions informatiques. Pour garantir la robustesse et la pertinence du système de modélisation des risques de crédit, nous avons sélectionné un ensemble de technologies modernes et complémentaires, dont nous présenterons ici les caractéristiques avant d'en justifier l'usage pour notre application.

### **I- Les langages de programmation et technologies**

#### **1- Le langage de programmation TypeScript**

TypeScript est un langage de programmation libre et open source développé par Microsoft, qui se définit comme un sur-ensemble typé de JavaScript. Son rôle principal est d'ajouter une sécurité de typage au code source, permettant de détecter les erreurs avant même l'exécution du programme.

TypeScript est un sur-ensemble de JavaScript qui ajoute un typage statique strict. C'est l'outil critique pour le module /utils/creditModel.ts. Il garantit que les variables financières (revenus, charges, montants) sont toujours traitées comme des nombres, évitant ainsi des erreurs de calcul. Il permet de définir des interfaces claires pour les objets Client et Demande, facilitant l'évolution du code et il identifie les bugs potentiels dès la phase de développement (ex: propriété manquante dans un objet client).

Dans le cadre de notre application bancaire, le choix de TypeScript est fondamental pour garantir l'intégrité des données financières. En effet, la manipulation de variables critiques telles que les revenus, le ratio d'endettement ou les montants de prêts ne tolère aucune approximation. En imposant un typage strict (par exemple, forcer une variable à être de type number et non string), TypeScript prévient les erreurs de calcul au sein de notre moteur de scoring. Cette sécurité est un gage de fiabilité indispensable pour un système d'aide à la décision où chaque point de score influence l'octroi d'un crédit.



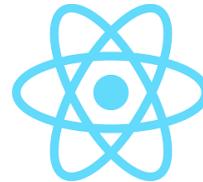
*Figure 7: Logo de TypeScript*

### 2- Le framework React.js

React est une bibliothèque JavaScript développée par Meta, spécialisée dans la création d'interfaces utilisateur modernes et réactives basées sur des composants. C'est une bibliothèque JavaScript front-end, destinée à faciliter la création d'interfaces utilisateur interactives à travers une architecture basée sur des composants réutilisables.

Il a permis de découper l'application en composants réutilisables (formulaire de saisie, cartes de résultats, tableaux), de gérer l'affichage instantané de la Probabilité de Défaut (PD) dès que l'utilisateur soumet une demande, sans recharger la page et Il assure la synchronisation entre les données saisies par l'Agent de Crédit et les calculs effectués par le moteur de risque.

L'utilisation de React.js pour ce projet se justifie par la nécessité d'une interface réactive et performante. Pour un agent de crédit, la rapidité de traitement est un enjeu majeur ; React permet ici de recalculer instantanément le score de risque dès qu'un paramètre est modifié dans le formulaire, sans avoir à recharger la page. Cette fluidité améliore non seulement l'expérience utilisateur, mais permet aussi une simulation dynamique des risques en temps réel, essentielle lors d'un entretien avec un client.



*Figure 8: Logo React.js*

### 3- Le framework Tailwind CSS

Tailwind CSS est un framework de conception de feuilles de style (CSS) de type "utility-first". Contrairement aux frameworks traditionnels, il permet de construire des designs personnalisés en combinant des classes prédéfinies directement dans le code HTML.

Ce choix technologique a été guidé par des impératifs d'ergonomie et de clarté visuelle. Dans notre application, la lisibilité de la décision (Acceptation, Analyse ou Refus) est primordiale. Tailwind CSS nous a permis de mettre en place rapidement un système de signalétique visuelle efficace, utilisant des codes couleurs standardisés (Vert pour le succès, Orange pour l'alerte, Rouge pour le danger). Cette structuration visuelle réduit la charge cognitive de l'utilisateur et sécurise le processus décisionnel.

### 4- La bibliothèque de visualisation : Recharts

Recharts est une bibliothèque de graphiques composites construite pour React. Elle utilise le format SVG pour générer des visualisations de données précises, légères et modulables.

Elle transforme les données brutes stockées dans le LocalStorage en graphiques parlants (camemberts pour la répartition des décisions, graphiques à barres pour le volume de demandes). Recharts permet au Gestionnaire de Risque de repérer d'un coup d'œil une augmentation anormale des refus ou des dossiers en analyse, facilitant le pilotage stratégique.

L'intégration de Recharts répond au besoin d'audit et de pilotage du risque exprimé dans le cahier des charges. Une liste brute de dossiers de crédit est difficile à interpréter pour un gestionnaire de risques ; en revanche, des graphiques dynamiques (diagrammes circulaires pour la répartition des décisions, histogrammes pour les volumes de prêts) permettent une analyse macroscopique immédiate. Cette visualisation est l'outil clé qui transforme les données stockées en informations stratégiques exploitables par la direction de la banque.



*Figure 9: Logo Recharts*

### 5- LocalStorage (API Web)

Le LocalStorage est une API web intégrée aux navigateurs modernes qui permet de stocker des données de manière persistante sous forme de paires clé-valeur. Contrairement aux cookies, les données ne sont pas envoyées au serveur et offrent une capacité de stockage plus importante.

LocalStorage est une API du navigateur web permettant de stocker des données sous forme de paires clé-valeur de manière persistante. Pour ce MVP, il remplace une base de données complexe (comme PostgreSQL). Il permet de sauvegarder les dossiers clients et l'historique des crédits même après la fermeture du navigateur. Il élimine le besoin de configurer un serveur Back-end et une API externe, permettant de se concentrer exclusivement sur la logique.

Pour ce MVP, l'adoption du LocalStorage est un choix stratégique d'agilité. Il remplace avantageusement une base de données complexe tout en offrant la persistance nécessaire : les dossiers clients et l'historique des scores ne sont pas perdus à la fermeture du navigateur. Cela

permet de démontrer la viabilité du système et la gestion de l'historique d'audit sans les lourdeurs d'une infrastructure back-end, tout en restant conforme à l'objectif de rapidité de déploiement d'un prototype.



**Figure 10: Outils et technologies utilisés**

### **II- L'environnement de travail**

La mise en œuvre de ces technologies s'est appuyée sur un environnement technique cohérent, structuré autour d'outils matériels et logiciels adaptés au développement professionnel.

#### **1- Outils matériels**

Le développement a été réalisé sur des équipements informatiques capables de supporter les flux de travail modernes. Nous avons eu à utiliser 05 ordinateurs ayant les caractéristiques suivantes :

- Une machine portable de marque DELL et de modèle Latitude E5470 avec :
  - Un processeur Intel (R) Core (TM) i5-6300CPU @ 2.40GHz (4CPUs), ~2.5Ghz
  - Une RAM de 8192 MB
  - Un système d'exploitation KALI Linux 2024
- Une machine portable de marque LENOVO et de modèle Thinkpad Yoga 11 e avec :
  - Un processeur Quad- Core 7th génération

- Une RAM de 8192 MB
  - Un système d'exploitation Ubuntu 2022.04
- Deux (02) machines portables de marque hp et de modèle ProBook avec :
- Un processeur Intel (R) Core (TM) i3-5005U CPU @ 2.00GHz, ~2.00GHz
  - Une RAM 4096 MB
  - Un système d'exploitation Windows 10 Professionnel 64-bit
- Une machine portable de marque ASUS et modèle : ASUS ROG Zephyrus G14 (GA401) avec :
- Processeur : AMD Ryzen 7 5800HS (8 cœurs, 16 threads) @ 3.2 GHz (max boost 4.4 GHz)
  - Mémoire RAM : 16 GB DDR4 (extensible)
  - Système d'exploitation : Windows 11 Famille 64-bit
  - Carte graphique : NVIDIA GeForce RTX 3050 Ti (4 GB dédiés)
  - Stockage : SSD NVMe de 512 GB
  - Écran : 14 pouces, IPS, Full HD (1920x1080), 144 Hz

### 2- Outils logiciels

L'écosystème logiciel a été soigneusement sélectionné pour optimiser la productivité et la qualité du code :

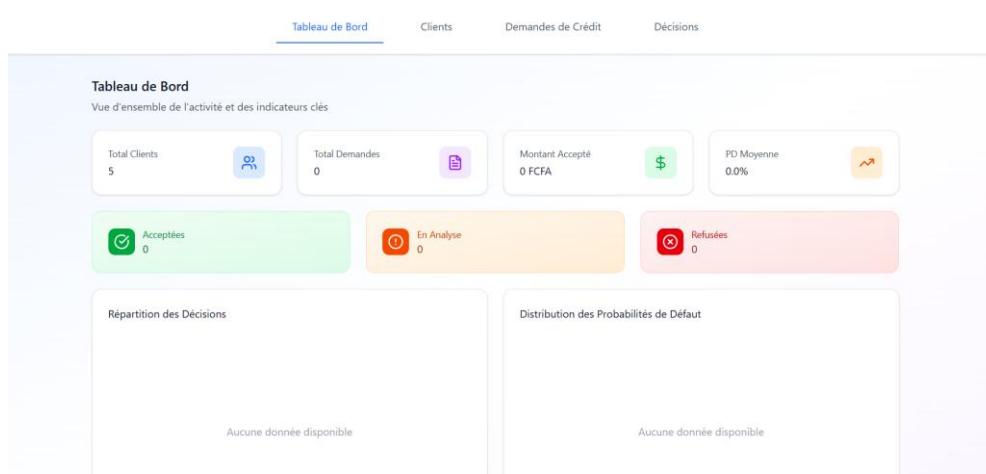
- **Visual Studio Code (VS Code)** : Choisi pour sa légèreté et ses extensions performantes dédiées à TypeScript et React.
- **Vite.js** : Utilisé comme outil de construction (build tool) pour sa rapidité exceptionnelle, garantissant un cycle de développement fluide.
- **Draw.io** : Cet outil a été indispensable pour la réalisation des diagrammes UML (cas d'utilisation, classes, séquences), assurant que le code reste fidèle à la conception architecturale.
- **Navigateurs modernes (Chrome/Edge)** : Leurs outils de développement intégrés ont permis de surveiller en temps réel l'état du LocalStorage et de déboguer les algorithmes de calcul de la Probabilité de Défaut (PD).

En somme, cette synergie technologique et matérielle a permis de livrer une application stable, capable de répondre aux exigences métier rigoureuses de la modélisation des risques bancaires.

### III- INTERFACES DU SYSTÈME

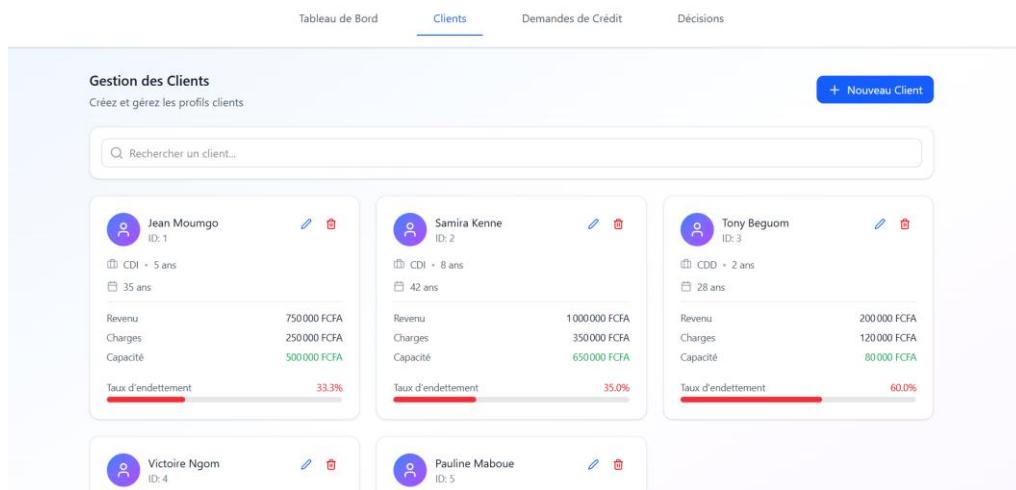
Les différentes interfaces de notre plateforme sont :

- Après la saisie de la commande **npm run dev** dans le terminal de notre éditeur de code, on accède au tableau de bord de l'application qui se présente comme suit :



**Figure 11: Tableau de bord de l'application**

- Ensuite, on accède à la partie client où il y a 5 clients par défaut et où on a la possibilité d'ajouter des clients, de les modifier ou de supprimer:



**Figure 12: Gestion des clients**

- La page de création de client se présente comme suit:

## MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE

Gestion des Clients  
Créez et gérez les profils clients

Nouveau Client

Nom \_\_\_\_\_ Prénom \_\_\_\_\_

Revenu mensuel (FCFA) \_\_\_\_\_ Charges mensuelles (FCFA) \_\_\_\_\_

Statut professionnel  
CDI

Ancienneté emploi (années) \_\_\_\_\_

Âge \_\_\_\_\_

**Créer** **Annuler**

Rechercher un client...

**Figure 13: Création d'un client**

- La page de demande de crédit se présente comme suit:

Tableau de Bord Clients **Demandes de Crédit** Décisions

Demande de Crédit  
Soumettez une nouvelle demande de crédit et obtenez une décision automatique

Informations de la demande

Client  
Sélectionnez un client

Montant du crédit (FCFA)  
Ex: 500000

Durée (mois)  
Sélectionnez une durée

Objet du crédit  
Sélectionnez un objet

**Soumettre la demande**

**i** Sélectionnez un client pour voir son profil

Modèle d'évaluation  
Le calcul de la probabilité de défaut (PD) prend en compte :

- Ratio d'endettement (30%)
- Ratio crédit/revenu (25%)
- Statut professionnel (20%)
- Ancienneté d'emploi (15%)
- Âge du client (10%)

Seuils de décision :

- PD < 35% : Accepté
- 35% ≤ PD < 60% : Analyse
- PD ≥ 60% : Refus

**Figure 14: Demande de crédit**

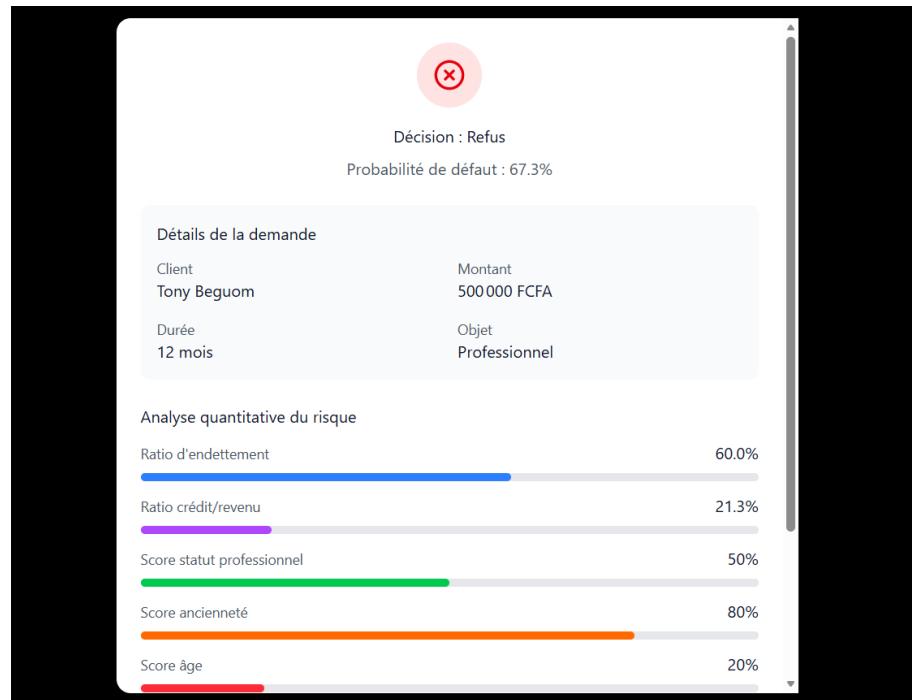
- On peut voir l'historique des décisions prises à partir de la page décisions qui se présente comme suit et où on a également la possibilité d'exporter les décisions :

## MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE

The screenshot shows a web-based application interface for managing credit decisions. At the top, there are tabs: 'Tableau de Bord', 'Clients', 'Demandes de Crédit', and 'Décisions'. The 'Décisions' tab is currently selected. Below the tabs, a header reads 'Décisions de Crédit' and 'Consultez l'historique des demandes et décisions'. There is a green 'Exporter CSV' button. A navigation bar below the header includes filters: 'Toutes (0)' (selected), 'Acceptées (0)', 'En analyse (0)', and 'Refusées (0)'. A table header follows with columns: DATE, CLIENT, OBJET, MONTANT, DURÉE, PD, DÉCISION, and ACTIONS. A message 'Aucune demande trouvée' (No request found) is displayed in the center of the table area.

**Figure 15: Historique des décisions**

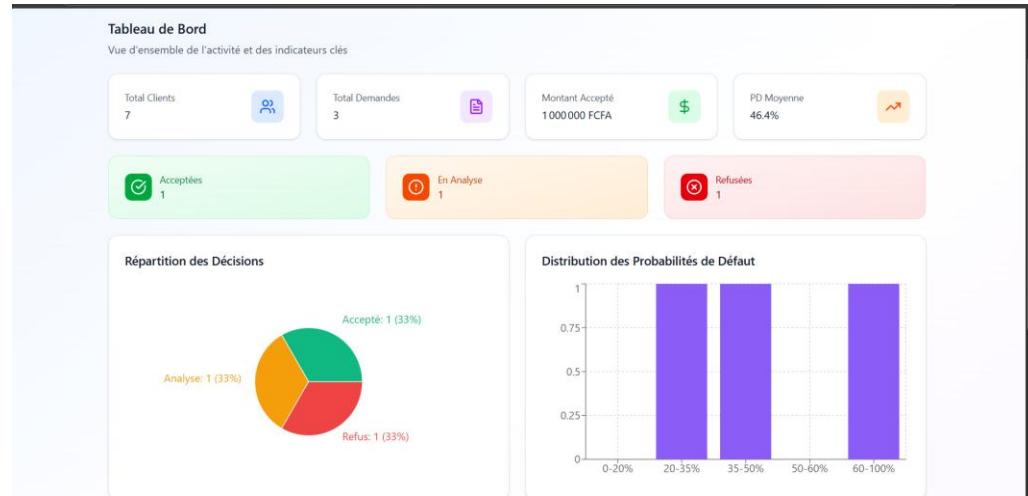
- Après avoir soumis des demandes de crédit, on est dirigé sur cette page avec les examinassions nécessaires et la décision finale:



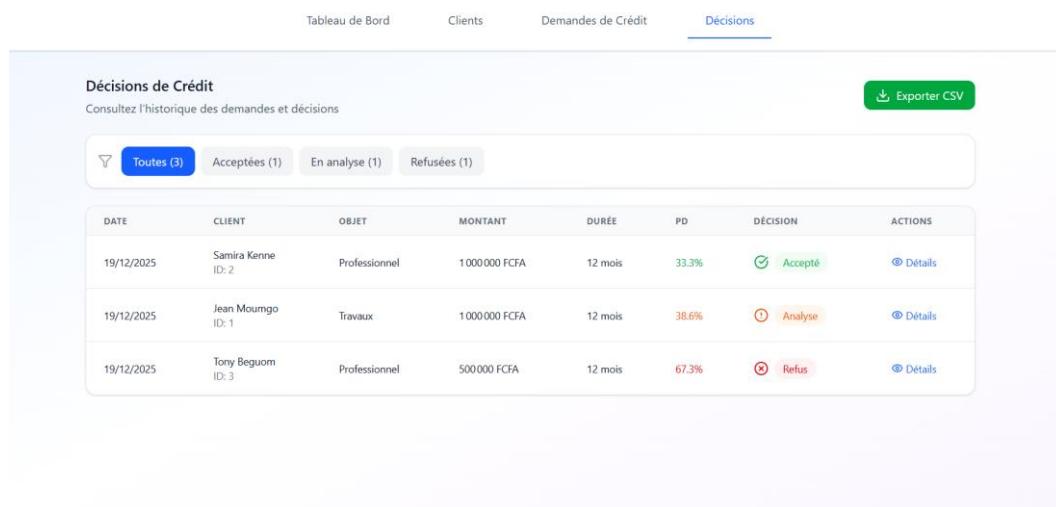
**Figure 16: Décision finale**

- Suite à cela, on retourne sur le tableau de bord pour observer les changements :

## MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE



➤ Et on a dans décisions l'historique :



## **CONCLUSION ET PERSPECTIVES**

Dans le cadre de ce projet d'ingénierie et conception logicielle, nous avons développé une plateforme dédiée à la modélisation des risques de crédit. Cet outil répond à un besoin critique de modernisation et de sécurisation des processus décisionnels bancaires, en remplaçant les méthodes d'évaluation traditionnelles, souvent manuelles ou sujettes aux biais humains, par une solution numérique centralisée, objective et structurée.

En s'appuyant sur des normes et référentiels de régulation bancaire internationaux tels que Bâle (Bâle II & III) pour la solvabilité et IFRS 9 (pour la prédition des pertes), ainsi que sur les principes du RGPD pour la protection des données sensibles, notre projet propose une solution à la fois conforme aux exigences réglementaires et adaptée aux enjeux de l'industrie financière.

Le succès de cette application repose sur une double conformité, à la fois technologique et réglementaire : **Sur le plan bancaire**, l'outil intègre les standards internationaux de Bâle III pour le calcul de la solvabilité et s'aligne sur la norme IFRS 9, permettant une gestion prédictive des pertes de crédit attendues. Le respect du protocole KYC (Know Your Customer) garantit une documentation conforme de l'identité et de la situation financière des clients. **Sur le plan logiciel et éthique**, le projet respecte les principes du RGPD pour la protection des données sensibles et assure une transparence algorithmique via l'utilisation de Reason Codes, permettant de justifier chaque décision de manière éthique et auditable.

Grâce à l'utilisation de technologies modernes et robustes telles que React JS pour une interface utilisateur réactive et TypeScript pour la fiabilité des calculs métier, cette plateforme offre une expérience intuitive tout en garantissant une précision absolue dans le calcul de la Probabilité de Défaut (PD).

Les fonctionnalités implémentées, notamment la collecte des données clients via des formulaires intelligents, le moteur de scoring basé sur un scorecard linéaire pondéré, et le tableau de bord de reporting permettent une évaluation instantanée et transparente de la solvabilité.

Ce projet a non seulement permis de mettre en œuvre une solution technique complète, mais a aussi démontré l'importance de l'automatisation pour réduire le taux de créances douteuses et renforcer la stabilité financière de l'institution.

Bien que fonctionnelle et conforme au cahier des charges, cette première version sous forme de MVP (Minimum Viable Product) reste perfectible et ouvre la voie à plusieurs axes d'amélioration. À l'avenir, il serait pertinent de migrer la persistance des données du

## **MODELISATION DES RISQUES DE CREDIT : CAS D'UNE BANQUE**

---

LocalStorage vers une base de données centralisée (type PostgreSQL) et d'envisager un déploiement sur une infrastructure Cloud sécurisée pour garantir l'accessibilité multi-agences. L'intégration de modules de Machine Learning (algorithmes de type Forêts Aléatoires ou Régression Logistique) permettrait d'affiner la précision du scoring en apprenant des comportements historiques de remboursement, dépassant ainsi les limites des modèles linéaires statiques. De plus, l'ajout d'un système de notification automatisé en temps réel (email ou SMS) permettrait d'informer immédiatement les clients et les gestionnaires de l'évolution du statut des dossiers de crédit. Enfin, il serait également intéressant d'interconnecter la plateforme avec les API d'Open Banking pour automatiser la récupération des flux financiers des clients, réduisant ainsi les erreurs de saisie et les risques de fraude documentaire.

Ces perspectives permettront d'offrir une solution encore plus robuste, prédictive et adaptée aux mutations numériques du secteur bancaire, tout en favorisant une culture de gestion des risques proactive et axée sur la donnée.