Rapport de Projet d'AI

Application web d'évaluation automatisée d'éligibilité aux crédits

Présenté et soutenu publiquement le 3/06/2025 par :
BELKABOUS Yassir
LAKHDER Kaouthar
OUBDA Dimitri-Gaetan
SANOGO Cheickna
YAHYAOUI Abderrahmane

Préparé sous la direction de
Mr. BEKKARI Aissam

2024-2025

Dédicaces

On dédie ce travail à tous ceux qu'on aime :

À Dieu le tout-puissant,

Qui nous a donné la force et la détermination nécessaires pour réaliser ce projet.

À nos parents,

Pour leur amour inconditionnel, leurs encouragements et leurs sacrifices inestimables qui ont créé un environnement favorable à notre réussite. Nous leur sommes éternellement reconnaissants et prions le Tout-Puissant de les bénir et de les protéger.

À nos professeurs,

Pour leur dévouement, leur soutien et leur enseignement précieux qui ont façonné notre parcours académique. Nous leur témoignons notre profond respect et notre gratitude.

À tous nos amis et collègues,

Pour leur soutien indéfectible, leur amitié sincère et leur présence réconfortante qui ont illuminé nos journées. Ils trouveront ici le témoignage d'une fidélité et d'une amitié sans limite.

Enfin, nous dédions ce projet à tous ceux qui ont croisé notre chemin et qui ont contribué de près ou de loin à notre formation et à notre épanouissement. Nous espérons que ce travail sera à la hauteur de leurs attentes et de leur confiance en nous.

Remerciement

Nous sommes heureuses d'exprimer toute notre gratitude et notre profonde reconnaissance à notre encadrant, **Monsieur Bekkari Aissam**, pour son accompagnement précieux tout au long de notre projet. Avec beaucoup de patience et de compréhension, il a suivi de près les différentes étapes de notre travail, nous offrant sa confiance, ses conseils et ses encouragements qui nous ont permis d'achever notre travail dans les meilleures conditions et de surmonter toutes les difficultés. Nous lui adressons nos plus sincères remerciements pour son dévouement.

Enfin, nous souhaitons exprimer notre gratitude à toutes les personnes qui ont contribué de près ou de loin à la réalisation de ce projet. Leur soutien, leur expertise et leur encouragement ont été essentiels à sa réussite

Résumé

Dans un contexte où les institutions financières doivent traiter un grand volume de demandes de crédit, la rapidité et la fiabilité de l'évaluation de l'éligibilité deviennent cruciales. Ce projet propose le développement d'une application web intelligente nommée SmartLoanCheck, visant à automatiser l'évaluation d'éligibilité aux crédits à travers une interface simple, sécurisée et performante. L'application repose sur une architecture full stack intégrant :

Spring Boot pour le backend (gestion des utilisateurs, logique métier, sécurité),

PostgreSQL pour la persistance des données,

Thymeleaf et Bootstrap pour l'interface utilisateur, Et un modèle de machine learning en Python (XGBoost) pour l'analyse automatique des demandes.

L'utilisateur peut s'inscrire, remplir un formulaire avec ses données personnelles et financières, puis recevoir instantanément une décision d'éligibilité générée par le modèle IA. Un administrateur peut également consulter l'historique des demandes et valider ou rejeter les décisions automatiques.

Cette solution vise à améliorer la prise de décision, réduire le temps de traitement, et minimiser les biais humains, tout en respectant des standards de sécurité, d'ergonomie et de maintenabilité.

Abstract

In a context where financial institutions must process a high volume of loan applications, the speed and reliability of eligibility assessment become critical. This project presents the development of an intelligent web application named SmartLoanCheck, aimed at automating the evaluation of credit eligibility through a simple, secure, and efficient interface.

The application is built on a full-stack architecture integrating:

Spring Boot for the backend (user management, business logic, security),

Thymeleaf and Bootstrap for the user interface,

And a machine learning model in Python (XGBoost) for the automatic analysis of applications.

Users can register, fill out a form with their personal and financial data, and instantly receive an eligibility decision generated by the AI model. An administrator can also view the request history and manually approve or reject automatic decisions.

This solution aims to enhance decision-making, reduce processing time, and minimize human bias, while adhering to high standards of security, usability, and maintainability.

Table de matière

Contents

Introduction générale

Le présent rapport s'inscrit dans le cadre du module **Intelligence Artificielle et Big Data** à l'**ENSA**, dont l'objectif est de renforcer les compétences des étudiants en matière d'intelligence artificielle appliquée. Ce projet, encadré par des enseignants de l'école, mobilise des domaines variés tels que l'analyse de données, le traitement automatisé de l'information et l'apprentissage machine, dans une optique de résolution concrète de problématiques réelles.

Notre projet, intitulé **SmartLoanCheck**, consiste à développer une application web intelligente capable d'évaluer automatiquement l'éligibilité d'un utilisateur à un crédit. Elle vise à simplifier le processus d'accès au financement. En remplaçant les démarches manuelles fastidieuses par un système automatisé d'aide à la décision, l'application recueille des données personnelles et financières, les analyse selon des critères prédéfinis, puis émet un avis instantané sur la faisabilité du prêt. L'architecture du système intègre un moteur de règles dynamique et, potentiellement, des algorithmes d'analyse prédictive fondés sur l'IA.

L'objectif principal est donc de proposer une solution rapide, accessible et fiable, à la fois pour les demandeurs de crédit et pour les institutions financières. Cela permet d'accélérer le traitement des demandes, de réduire les coûts opérationnels et de renforcer la **transparence** dans l'évaluation des dossiers.

Ce rapport présente de manière structurée toutes les étapes de la réalisation du projet : de la phase d'analyse à la mise en production, en passant par la conception technique, le choix des outils, et l'implémentation de solutions intelligentes. Notre ambition est que **SmartLoanCheck** participe à l'évolution des services financiers numériques, en s'inscrivant dans une logique d'inclusion et d'innovation sociale. En tant qu'étudiants, nous affirmons notre volonté de contribuer activement à la construction d'un futur où la technologie est mise au service de la société, en brisant les barrières d'accès aux services essentiels.

Les cinq chapitres suivants décrivent avec rigueur et méthode chaque phase du projet

- ✓ Chapitre 1 : Contexte général

Ce chapitre introduit la problématique, les motivations du projet, et le contexte socio-économique. Il présente également notre organisation de travail, les outils de gestion de projet adoptés, ainsi que notre planification.

- ✓ Chapitre 2 : État de l'art de la technologie utilisée

Ici, nous définissons les concepts techniques majeurs et réalisons une revue des solutions existantes, afin de positionner notre projet dans le paysage technologique actuel.

- ✓ Chapitre 3 : Analyse et Conception

Cette section se concentre sur les besoins fonctionnels et non fonctionnels. Elle présente les modèles UML utilisés pour concevoir l'application, tels que les diagrammes de cas d'utilisation, de séquence et d'activité.

- ✓ Chapitre 4 : Outils et Technologies

Ce chapitre détaille les outils et Frameworks choisis (backend, frontend, base de données, IA…), en justifiant nos choix techniques et en expliquant le processus d'intégration des modèles d'IA.

- ✓ Chapitre 5 : Réalisation

Enfin, nous présentons concrètement le développement de l'application à travers des captures d'écran, des explications sur les fonctionnalités implémentées, et une évaluation des performances obtenues.

Chapitre 1 :

Contexte général

Chapitre 1 : Contexte général

1. Introduction

Ce chapitre présente le contexte général du projet envisagé, incluant la présentation de l'établissement et la problématique à laquelle répond notre projet. Nous effectuerons ensuite une étude d'existence pour évaluer les différentes solutions disponibles sur le marché, avant de détailler la méthodologie de travail choisie pour la réalisation de notre application

2. Présentation de l'établissement

L'Ecole Nationale des Sciences Appliquées de Marrakech (ENSA Marrakech). Est une grande école d'ingénieurs située à Marrakech, au Maroc. Elle fait partie du réseau des ENSA du Maroc, qui sont des établissements d'enseignement supérieur publics spécialisés dans la formation d'ingénieurs.

Créée en 2000, l'ENSA Marrakech a pour mission de former des ingénieurs hautement qualifiés et de répondre aux besoins du marché du travail en matière de compétences techniques, de recherche et d'innovation. L'école offre des formations d'ingénierie dans des domaines tels que le génie de la cyberdéfense et les systèmes de télécommunications embarqués, le génie électrique, le génie industriel et logistique, le génie informatique, le génie Réseaux, Systèmes & Services Programmables.

Les programmes de formation de l'ENSA Marrakech sont conçus pour fournir aux étudiants une solide base théorique ainsi qu'une expérience pratique grâce à des projets, des stages et des travaux de recherche. L'école dispose d'un corps professoral hautement qualifié et engagé dans la recherche, ainsi que d'infrastructures modernes et de laboratoires équipés de technologies de pointe.

3. Problématique

L'accès au crédit représente aujourd'hui un enjeu majeur pour les particuliers et les entreprises, notamment dans les contextes de relance économique, de développement d'activités ou de gestion des imprévus. Pourtant, l'un des obstacles majeurs à cette accessibilité reste la complexité et la lenteur du processus d'évaluation de l'éligibilité des demandeurs.

En effet, les établissements financiers doivent s'assurer que chaque demandeur répond à un ensemble de critères précis (revenus, stabilité professionnelle, niveau d'endettement, historique bancaire, etc.), afin de limiter les risques d'impayés. Cette étape repose encore, dans de nombreux cas, sur des traitements manuels ou semi-automatisés, ce qui engendre plusieurs difficultés : **délai de réponse long**, **surcharge administrative**, **coût élevé du traitement**, et parfois même **inégalités dans l'analyse des profils**. De plus, la gestion de volumes croissants de demandes dans un temps réduit rend la tâche encore plus complexe.

Par ailleurs, la prise en compte de données financières sensibles pose des défis en termes de **confidentialité, traçabilité et conformité réglementaire**, notamment vis-à-vis des normes de protection des données personnelles. Dans ce contexte, les méthodes classiques peinent à répondre aux exigences modernes de rapidité, d'efficacité, d'impartialité et de sécurité.

Dès lors, on assiste à une réelle nécessité de repenser les mécanismes d'évaluation de l'éligibilité au crédit, afin de les rendre plus adaptés aux exigences actuelles du secteur financier et aux attentes croissantes des usagers.

4. Solutions

Face à la complexité, à la lenteur et aux limites des processus traditionnels d'évaluation de l'éligibilité au crédit, plusieurs approches ont émergé pour améliorer l'efficacité et la fiabilité des analyses financières. Parmi les solutions les plus répandues, on trouve l'**automatisation des processus décisionnels** à travers des systèmes informatiques capables d'appliquer des règles définies à grande échelle, réduisant ainsi les délais de traitement. Les **systèmes d'aide à la décision** permettent, quant à eux, d'assister les conseillers financiers dans l'analyse des dossiers en croisant rapidement plusieurs sources de données.

Par ailleurs, l'exploitation de **l'intelligence artificielle et de l'apprentissage automatique** gagne en popularité. Ces technologies permettent d'analyser des volumes massifs de données historiques pour identifier des profils types, détecter des risques et prédire la probabilité de remboursement d'un emprunteur. Elles offrent une capacité d'analyse plus fine, plus rapide et souvent plus objective que les approches classiques.

D'autres initiatives visent à renforcer la **transparence et la traçabilité** du processus d'évaluation grâce à l'intégration de tableaux de bord, d'indicateurs de risque, ou encore de modèles explicables, permettant de justifier chaque décision. Enfin, des efforts sont également menés sur le plan réglementaire et technologique pour **garantir la confidentialité et la sécurité des données sensibles** utilisées dans ces processus.

En combinant technologie, algorithmes, expertise métier et cadre légal, ces solutions contribuent progressivement à transformer l'évaluation de l'éligibilité en un processus plus rapide, plus équitable, et mieux adapté aux exigences du secteur financier moderne.

5. L'étude d'existence

Afin de mieux situer notre projet dans son contexte technologique et fonctionnel, nous avons étudié plusieurs applications web et services existants qui proposent des fonctionnalités similaires à notre objectif, à savoir l'évaluation automatisée de l'éligibilité au crédit. Ces outils partagent des points communs avec notre solution : ils permettent une analyse en ligne du profil financier, fournissent des résultats rapides, et tentent de simplifier l'accès au crédit.

5.1 Younited Credit :

Younited Credit est une plateforme française spécialisée dans les crédits en ligne à destination des particuliers souhaitant financer divers projets comme des voyages, travaux ou regroupements de prêts. Elle permet une simulation rapide en ligne, basée sur la saisie de données personnelles telles que le montant souhaité, la durée et les revenus. Une réponse de principe est fournie en quelques minutes, ce qui simplifie l'expérience utilisateur. Toutefois, le système repose sur des règles classiques et prédéfinies, sans intégration visible de technologies avancées comme l'intelligence artificielle. De plus, le processus de validation finale reste partiellement manuel, ce qui limite le niveau d'automatisation.

younited credit.

5.2 Pretto :

Pretto est un courtier digital français spécialisé dans le crédit immobilier. Il propose une plateforme en ligne permettant aux utilisateurs d'estimer leur capacité d'emprunt selon plusieurs critères (revenus, apport personnel, type de contrat, etc.) et de comparer les offres des banques partenaires. Bien que l'interface soit moderne et réactive, la solution reste centrée exclusivement sur l'immobilier. De plus, l'outil agit davantage comme un comparateur intelligent que comme un moteur décisionnel automatisé, l'intervention humaine étant souvent nécessaire pour finaliser les démarches.

Pretto
Domptez votre projet immobilier

5.3 Revolut – Credit Offers :

Revolut, célèbre néobanque internationale, propose des offres de crédit directement intégrées dans son application mobile. Les utilisateurs peuvent se voir proposer des prêts en fonction de l'analyse automatique de leur comportement financier et de l'historique de leur compte Revolut. Cette approche permet une expérience fluide, sans formulaires complexes. Toutefois, le service est réservé aux clients Revolut, et les critères d'analyse restent internes et peu transparents, ce qui limite l'universalité et la compréhension du processus par l'utilisateur.

5.4 N26 – Crédit Personnel :

N26, banque mobile allemande, propose une offre de crédit personnel en partenariat avec Younited Credit. L'utilisateur peut effectuer une demande de prêt entièrement en ligne, avec une réponse rapide après saisie de quelques informations financières. Le service est intuitif et accessible, mais reste limité aux clients de N26 et implique encore une validation administrative par la banque partenaire, ce qui ne garantit pas une évaluation entièrement automatisée

5.5 Klarna – Credit & Risk APIs :

Klarna, fournisseur de solutions de paiement en ligne, utilise un moteur de scoring automatique pour évaluer en temps réel la capacité d'un utilisateur à effectuer un achat en différé ou en plusieurs fois. Le système s'appuie sur le comportement d'achat et les données transactionnelles. Toutefois, cette solution est principalement orientée vers le e-commerce, et ne s'applique pas directement aux demandes de crédit classique. De plus, les critères d'évaluation ne sont pas exposés à l'utilisateur.

5.6 Zest AI :

Zest AI est une entreprise américaine qui développe des solutions de scoring basées sur l'intelligence artificielle, destinées aux institutions financières. Elle permet aux banques et aux coopératives de crédit de concevoir des modèles prédictifs personnalisés pour améliorer leurs décisions d'octroi de crédit. Bien

que technologiquement avancée, cette solution est réservée à de grandes organisations et nécessite des ressources techniques importantes, la rendant peu adaptée à des structures éducatives ou à petite échelle.

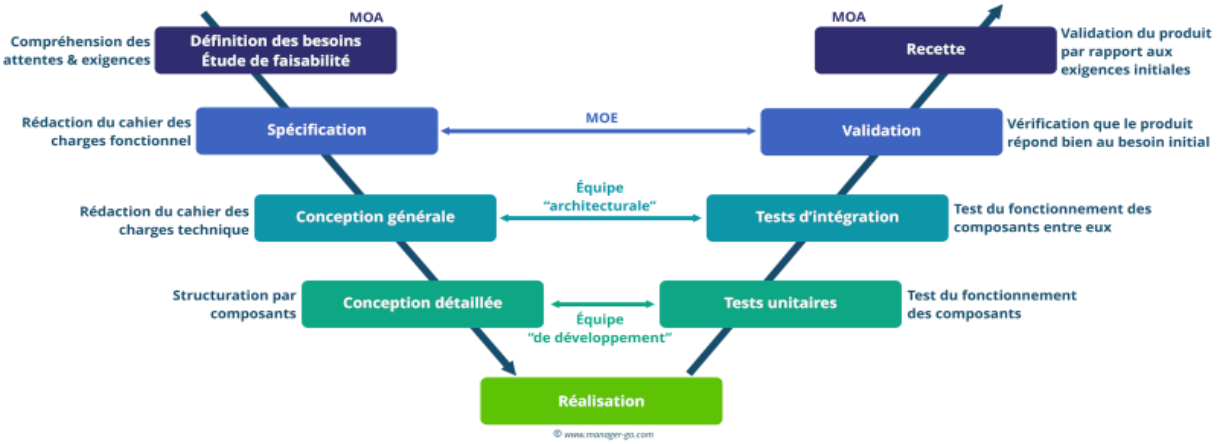
6. Solutions

Dans le cadre de ce projet, nous proposons de développer une **application web intelligente et automatisée** baptisée **SmartLoanCheck**, dont l'objectif principal est d'évaluer l'éligibilité d'un utilisateur à un crédit de manière rapide, fiable et objective. Cette solution repose sur l'intégration d'un **moteur de règles métier** définies à partir de critères financiers et personnels standards (revenus, charges, statut professionnel, âge, etc.), couplé à des techniques d'analyse de données pour simuler une décision de crédit. L'utilisateur accède à une interface simple où il saisit ses informations, qui sont ensuite traitées en temps réel par le système, sans intervention humaine. L'application fournit immédiatement une **réponse claire sur l'éligibilité**. Cette approche vise à **moderniser le processus de pré-analyse des demandes de crédit**, à réduire la charge des institutions financières, et à offrir aux utilisateurs une **expérience fluide et transparente**, tout en respectant les principes de sécurité, de confidentialité des données, et d'accessibilité numérique.

7. Méthodologie de travail

7.1 La méthodologie de dev (cycle V) :

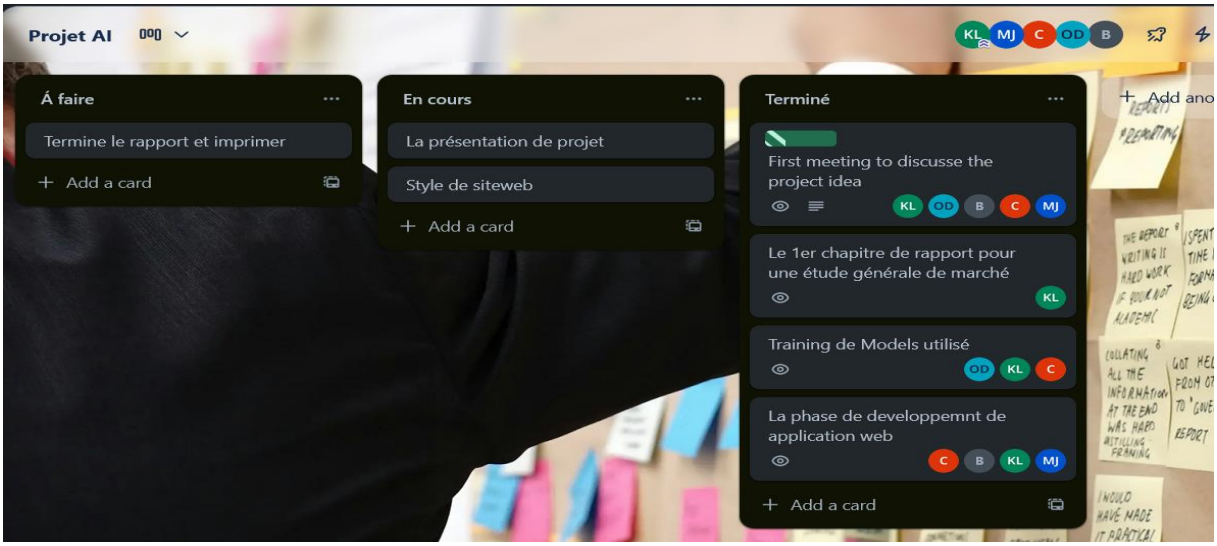
Notre démarche méthodologique pour mener à bien ce projet a été guidée par l'utilisation du modèle en V, une méthodologie largement reconnue pour sa capacité à gérer efficacement le développement de projets complexes. En suivant ce schéma en forme de V, nous avons organisé notre processus de manière à garantir une gestion rigoureuse du projet. Cela s'est traduit par une attention particulière portée à la planification minutieuse, à la définition claire des exigences, à un développement itératif, à des tests systématiques et à une validation rigoureuse. Cette approche a encouragé une collaboration étroite entre les différentes équipes impliquées, assurant ainsi la cohérence du projet à chaque étape de son évolution. Le modèle en V a servi de cadre essentiel pour atteindre nos objectifs avec succès et garantir la qualité et la fiabilité du produit final



7.2 Gestion du projet :

L'outil utilisé pour gérer le projet est Trello, elle est un outil de gestion de projet en ligne. Il facilite le travail collaboratif en organisant les projets en planches listant des cartes. Chaque carte représente une tache qui peut être affectée à des utilisateurs et liée à un deadline. Par conséquent, il permet un bon contrôle de l'avancement dans le projet

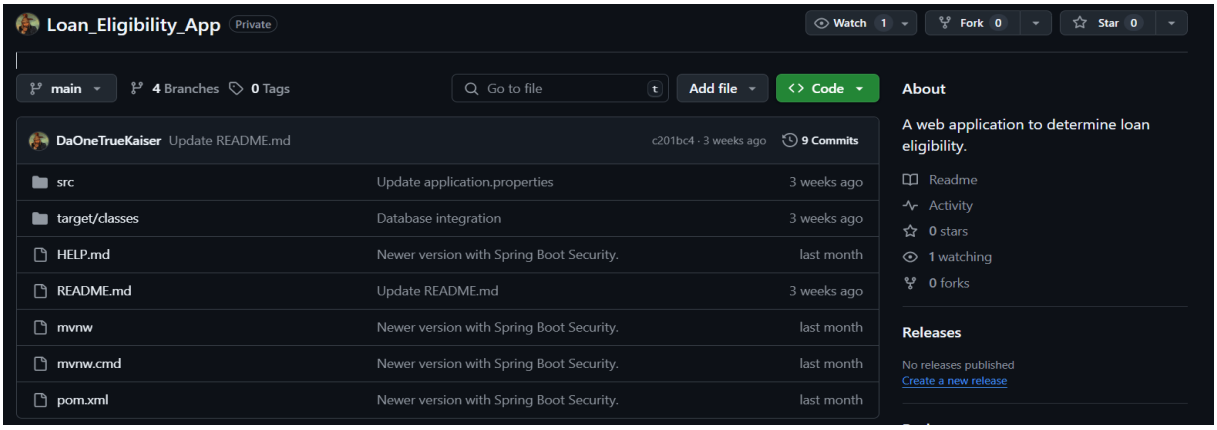
Trello



7.3 Gestion du projet :

Pour assurer une gestion efficace du code source de notre application, nous avons adopté Git et GitHub comme outils principaux. Git nous permet de versionner avec précision chaque modification du code, facilitant ainsi le suivi des changements et la collaboration entre les membres de l'équipe. Grâce à

GitHub, nous avons pu centraliser notre code source, ce qui a rendu les révisions et les contributions plus transparentes et accessibles à tous les développeurs impliqués. En utilisant ces outils, nous avons non seulement optimisé notre flux de travail mais aussi minimisé les risques d'erreurs et de conflits dans le code, permettant une évolution plus stable et plus sécurisée de l'application



8. Conclusion

Dans ce chapitre, nous avons situé le projet dans son cadre général en présentant l'école qui accueille le projet, ses objectifs et son domaine d'activité. Nous avons également mené une étude comparative de quelques applications similaires, afin de mieux comprendre les attentes des utilisateurs et les fonctionnalités qui font le succès de cette application.

Chapitre 2 :
L'état de l'art de la technologie utilisée

Chapitre 2 : l'état de l'art de ls technologie utilisées

1. Introduction

Ce chapitre a pour objectif de présenter les **concepts technologiques fondamentaux** qui sous-tendent le développement de notre projet. Cette étude de l'état de l'art permet de positionner notre travail dans une dynamique d'innovation appuyée sur des bases solides et reconnues.

2. Intelligence Artificielle (IA) :

2.1 Qu'est-ce que l'IA ?

L'Intelligence Artificielle (IA) est un domaine de l'informatique qui développe des systèmes capables de réaliser des tâches qui nécessitent habituellement l'intelligence humaine. Cela inclut des processus tels que l'apprentissage (l'acquisition d'informations et de règles pour l'utiliser), le raisonnement (l'utilisation des règles pour atteindre des conclusions approximatives ou définies) et l'auto-correction. Particulièrement avancée dans les domaines du traitement du langage naturel, de la reconnaissance d'image, et de l'automatisation robotique, l'IA est en train de transformer divers secteurs en améliorant l'efficacité, en automatisant les tâches et en fournissant de nouvelles manières de résoudre des problèmes complexes. L'IA favorise non seulement une meilleure prise de décision mais aussi une optimisation des processus et une réduction significative des coûts opérationnels

2.2 L'histoire de l'intelligence artificielle (IA) :

L'évolution de l'intelligence artificielle (IA) remonte aux années 1950, une période où l'accroissement de la capacité des ordinateurs a suscité un intérêt croissant pour le potentiel de programmation de machines pour exécuter des tâches traditionnellement attribuées à l'intelligence humaine. Les premières expérimentations en

IA se concentraient principalement sur la résolution de puzzles logiques et la conception de systèmes capables de jouer aux échecs.

En 1956, le champ de l'IA a été formellement établi lors d'une conférence emblématique à Dartmouth College, événement qui a marqué la reconnaissance de l'IA comme un domaine de recherche autonome. Les décennies suivantes, notamment les années 1950 et 1960, ont vu des avancées significatives en reconnaissance vocale et d'images, période de progrès intense pour l'IA.

Cependant, les années 1970 ont été marquées par un ralentissement dans les avancées de l'IA, car il est devenu évident que les techniques efficaces pour des problèmes simples ne se généralisaient pas bien aux défis plus complexes. L'essor a repris dans les années 1980 et 1990 avec des progrès dans le traitement de données et l'accessibilité accrue à de grands volumes de données, menant au

développement des réseaux neuronaux qui ont ouvert la voie à des innovations telles que la reconnaissance de la parole et de l'écriture manuscrite.

Depuis les années 2000, avec l'émergence des algorithmes d'apprentissage automatique, les ordinateurs ont acquis la capacité d'apprendre à partir des données sans programmation explicite, propulsant des avancées majeures dans des domaines tels que la reconnaissance d'images, la reconnaissance vocale, et la traduction automatique. De nos jours, l'IA imprègne notre quotidien, des assistants virtuels sur nos smartphones à la reconnaissance faciale utilisée dans les aéroports et centres commerciaux. Les recherches actuelles et futures s'orientent vers des applications révolutionnaires telles que les véhicules autonomes et la médecine personnalisée, consolidant ainsi le rôle de l'IA en tant que domaine dynamique de recherche et d'application pratique.

2.3 Comment un IA fonctionne-t-il ?

L'intelligence artificielle (IA) emploie une combinaison d'algorithmes avancés et de vastes bases de données pour imiter les processus de pensée humaine, offrant ainsi des solutions innovantes à des problèmes complexes.

Cette technologie fonctionne selon des principes méthodiques et structurés, que nous détaillons ci-après :

- **Apprentissage :** La phase d'apprentissage de l'IA est fondamentale. Elle implique le traitement de grandes quantités de données pour développer la compréhension nécessaire à la reconnaissance des patterns. Pour illustrer, dans le domaine de la reconnaissance d'images, l'IA examine des millions d'images annotées pour apprendre à identifier des éléments tels que les formes, les textures, et les couleurs. Cet apprentissage est souvent réalisé par des réseaux de neurones qui simulent la manière dont le cerveau humain opère.
- **Modélisation :** Après l'apprentissage, l'IA synthétise ses connaissances sous forme de modèles mathématiques complexes. Ces modèles sont essentiellement des algorithmes qui utilisent les données apprises pour faire des prédictions ou prendre des décisions. Par exemple, un modèle pourrait prédire la demande future de produits basée sur des données historiques de vente, ou aider à diagnostiquer des maladies à partir d'images médicales
- **Exécution :** Une fois que les modèles sont bien établis, l'IA peut les appliquer à des situations du monde réel. Cela peut inclure des applications telles que la traduction automatique de langues, où l'IA peut traduire des textes ou des discours en temps réel, ou la conduite autonome, où elle interprète et réagit aux conditions routières en continu
- **Amélioration continue :** Un des aspects les plus puissants de l'IA est sa capacité à s'améliorer en continu. À travers des cycles d'apprentissage supplémentaires, l'IA intègre de nouvelles données dans ses modèles existants, affinant ainsi sa précision et son efficacité. Ce processus

d'apprentissage continu permet à l'IA de s'adapter et de se perfectionner au fur et à mesure que de nouvelles informations deviennent disponibles.

Les technologies sous-jacentes à l'IA, comme l'apprentissage machine, l'apprentissage profond, le traitement du langage naturel, et la vision par ordinateur, sont en constante évolution. Elles permettent à l'IA de surpasser les capacités humaines dans certains domaines tout en ouvrant de nouvelles avenues pour l'innovation et l'amélioration des processus dans presque tous les secteurs de l'industrie.

3. Machine Learning

3.1 Qu'est-ce que Machine Learning ?

Le machine learning, ou apprentissage automatique, est une branche cruciale de l'intelligence artificielle qui se focalise sur la conception et le développement de systèmes capables d'apprendre à partir de données, sans nécessiter de programmation explicite pour chaque nouvelle tâche. Cette technologie permet aux ordinateurs de reconnaître des modèles et de prendre des décisions avec une intervention humaine minimale.

- **Fondements théoriques :** Le machine learning s'appuie sur des concepts mathématiques et statistiques pour créer des modèles prédictifs à partir de données. Ces modèles sont entraînés à partir de grands ensembles de données et sont capables d'améliorer leur précision au fil du temps grâce à des mécanismes d'apprentissage et d'ajustement continus.
- **Types d'apprentissage :** Il existe principalement trois types d'apprentissage dans le machine learning: supervisé, non supervisé, et par renforcement. L'apprentissage supervisé utilise des données étiquetées pour apprendre à prédire des résultats. L'apprentissage non supervisé, quant à lui, travaille avec des données non étiquetées pour identifier des structures cachées. L'apprentissage par renforcement permet aux modèles d'apprendre à partir de récompenses obtenues en réponse à leurs actions.
- **Applications pratiques :** Les applications du machine learning sont vastes et touchent de nombreux secteurs. Par exemple, dans le secteur financier, il est utilisé pour la détection de fraudes et la gestion de risques. Dans le domaine de la santé, il aide à la prédiction de maladies et à l'optimisation des traitements. En marketing, il est employé pour comprendre le comportement des consommateurs et personnaliser les offres.
- **Défis et perspectives :** Malgré son efficacité, le machine learning pose certains défis, notamment en termes de biais des données, de sécurité des données et de transparence des décisions algorithmiques. Les recherches futures sont orientées vers la création de modèles plus robustes, équitables et explicables

Le machine learning représente une technologie influente qui révolutionne l'utilisation des données pour guider et automatiser les prises de décision. Il occupe une position centrale dans la création de solutions intelligentes qui améliorent l'efficacité et stimulent l'innovation à travers divers secteurs.

3.2 Comment Machine Learning fonctionne-t-il ?

Le machine learning (apprentissage automatique) offre une gamme étendue de fonctionnalités qui le rendent indispensable dans divers secteurs et applications. Voici quelques-unes des principales fonctionnalités du machine learning :

- **Prédiction :** Le machine learning est largement utilisé pour prédire des résultats futurs à partir de données historiques. Par exemple, il peut prédire la demande de produits pour les entreprises, les résultats des patients en médecine, ou les tendances du marché en finance.
- **Classification :** Cette fonctionnalité permet de catégoriser des données en différentes classes. Elle est couramment utilisée dans la détection de spams dans les e-mails, le diagnostic médical pour distinguer différents types de maladies, ou la segmentation de clients dans le marketing.
- **Reconnaissance de motifs :** Le machine learning peut identifier des motifs ou des tendances dans des données complexes qui ne seraient pas évidents pour l'humain. Cela est utilisé dans la surveillance de la fraude bancaire, l'analyse de séquences d'ADN, ou l'interprétation de données de marché.
- **Optimisation :** Les algorithmes de machine learning peuvent être utilisés pour optimiser des processus ou des performances, comme dans la logistique pour améliorer les itinéraires de livraison, ou dans la gestion de l'énergie pour maximiser l'efficacité des systèmes de chauffage et de refroidissement.
- **Recommandation :** Une des applications les plus connues du machine learning est les systèmes de recommandation, qui suggèrent des produits, des films, ou des musiques aux utilisateurs en fonction de leurs intérêts et comportements passés, comme ceux utilisés par Amazon, Netflix, et Spotify.
- **Découverte de connaissances :** Le machine learning permet d'extraire des connaissances à partir de grandes quantités de données, souvent en découvrant des relations cachées ou inattendues entre les variables. Cela est essentiel dans des domaines comme la bio-informatique ou la recherche scientifique.
- **Vision par ordinateur :** Dans ce domaine, le machine learning est utilisé pour analyser et interpréter des images et des vidéos, permettant des applications comme la reconnaissance faciale, l'inspection industrielle, ou le traitement automatisé des réclamations d'assurance.
- **Traitement automatique du langage naturel (TALN) :** Le machine learning facilite la compréhension et la génération du langage humain, permettant des interactions plus naturelles entre les humains et les machines, comme dans les assistants vocaux et les chatbots

Ces fonctionnalités font du machine learning un outil puissant et polyvalent, capable d'améliorer l'efficacité et d'offrir de nouvelles capacités à travers une multitude d'industries et de domaines d'application.

4. Credit Scoring

Qu'est-ce que le Credit Scoring ?

Le credit scoring, ou score de crédit, est une méthode statistique utilisée par les institutions financières pour évaluer la solvabilité d'un emprunteur potentiel. Il consiste à attribuer une note numérique à un profil financier en se basant sur divers facteurs tels que les revenus, les dettes, l'historique de remboursement, la durée d'emploi, et le type de crédit demandé. Ce score permet de prédire la probabilité qu'un client respecte ses obligations de remboursement.

- **Fondements théoriques** : Les premiers modèles de scoring étaient fondés sur des méthodes statistiques comme la régression logistique, qui permettent de modéliser la relation entre les caractéristiques d'un emprunteur et la probabilité de défaut. Ces modèles ont ensuite évolué vers des techniques plus avancées, notamment les arbres de décision**,** les forêts aléatoires et aujourd'hui des modèles basés sur le machine learning
- **Fonctionnement** : Un modèle de scoring analyse un ensemble de données d'entrées (revenu, dettes, âge, etc.) et génère un score entre deux bornes prédéfinies (souvent 300 à 850). Plus le score est élevé, plus l'utilisateur est considéré comme fiable. Ces scores permettent aux prêteurs de prendre des décisions rapides et cohérentes.
- **Applications pratiques** : Le credit scoring est utilisé dans la bancarisation, les microcrédits, les plateformes de prêt en ligne, et les cartes de crédit. Il joue un rôle fondamental dans la réduction du risque financier pour les prêteurs tout en accélérant le processus d'octroi.
- **Limites** : Le credit scoring traditionnel est parfois critiqué pour son manque de transparence, ses biais potentiels (discrimination socio-économique), et son incapacité à évaluer des profils atypiques (freelancers, jeunes sans historique bancaire). Cela motive l'évolution vers des modèles intelligents plus souples et justes.

5. Systèmes d'Aide à la Décision

Qu'est-ce qu'un Système d'Aide à la Décision (DSS) ?

Un système d'aide à la décision (Decision Support System – DSS) est une application informatique conçue pour aider les décideurs humains à analyser des données complexes et à prendre des décisions éclairées. Ces systèmes combinent des outils de modélisation, des bases de données et des interfaces utilisateurs pour fournir des recommandations, des prédictions ou des analyses.

- **Caractéristiques clés** : Les DSS sont interactifs, souples et orientés utilisateur. Ils permettent l'exploration de scénarios ("what-if analysis"), la visualisation de données, et l'utilisation de modèles prédictifs. Ils sont particulièrement utiles dans des contextes où les décisions ne sont pas totalement automatisables.
- **Fonctionnement dans le domaine du crédit** : Dans les institutions financières, un DSS peut intégrer des règles d'acceptation, des grilles de scoring, et des tableaux de bord dynamiques pour accompagner les agents dans le traitement des demandes. L'analyse peut combiner des données internes (comptes clients) et externes (historique de crédit, marché…).
- **Exemples d'application :** Les DSS sont largement utilisés dans la gestion financière, le diagnostic médical, la logistique, la gestion des risques et bien sûr, dans les plateformes de prêt en ligne.
- **Limites et défis** : Les DSS ne remplacent pas le jugement humain et leur efficacité dépend de la qualité des données. Ils posent aussi des enjeux en termes d'explicabilité des modèles utilisés (surtout s'ils intègrent du machine learning).

6. Conclusion

Ce chapitre a permis d'explorer les fondements théoriques et technologiques qui sous-tendent notre projet. Ces technologies jouent un rôle central dans la transformation numérique des services bancaires, en apportant précision, rapidité et personnalisation dans les processus d'analyse de crédit. Cette revue nous a également permis de situer notre projet dans un écosystème technologique évolutif, en mettant en lumière les défis et opportunités liés à l'automatisation intelligente des décisions. Forts de ces bases théoriques, nous pouvons désormais aborder, dans les chapitres suivants, les aspects pratiques de conception, de modélisation et de mise en œuvre de notre solution.

Chapitre 3 :

Analyse et Conception

Chapitre 3 : Analyse et Conception

1. Introduction :

Ce chapitre est consacré à l'analyse et à la conception du système à développer. Il constitue une étape essentielle dans le cycle de vie du projet, car il permet de transformer les besoins exprimés en une représentation structurée et compréhensible du fonctionnement futur de l'application.
L'objectif principal de cette phase est d'identifier avec précision les exigences fonctionnelles et non fonctionnelles du système, de modéliser les interactions entre les différents acteurs, de structurer les données nécessaires, et de planifier les échanges entre les différentes composantes logicielles.
À travers une série de diagrammes et de descriptions détaillées, ce chapitre fournit une vision claire de l'organisation interne du système et sert de base à sa réalisation technique.

2. Analyse des besoins :

2.1. Besoins fonctionnels :

Les besoins fonctionnels définissent l'ensemble des fonctionnalités que le système doit offrir pour répondre aux attentes des utilisateurs. Ils décrivent ce que le système est censé faire, indépendamment de la manière dont ces fonctionnalités seront techniquement mises en œuvre.
Dans le cadre de ce projet, les besoins fonctionnels identifiés sont les suivants :

Gestion des utilisateurs :

- Permettre à un utilisateur de créer un compte via un formulaire d'inscription.
- Permettre à un utilisateur de se connecter à l'aide de ses identifiants.

Saisie de données :

- Mettre à disposition un formulaire pour la saisie d'informations personnelles, professionnelles et financières.
- Vérifier que les champs du formulaire sont correctement remplis et valides (format, champs obligatoires, etc.).

Analyse automatisée des données :

- Transmettre les données saisies à un système d'analyse.
- Effectuer un traitement basé sur un modèle intelligence artificielle (AI).

Affichage du résultat :

- Générer un résultat de l'analyse indiquant une décision binaire (ex : favorable ou non favorable).
- Afficher le résultat de l'analyse à l'utilisateur sous une forme claire et compréhensible.

2.2. Besoins non fonctionnels :

Ces besoins ne sont pas absolument indispensables au fonctionnement de la plateforme, mais ils servent comme objectifs secondaires du projet :

- **Sécurité**

Les mots de passe seront stockés sous forme hachée avec salage, et des protections contre les injections SQL, XSS et CSRF seront mises en place.

- **Performance**

Les algorithmes d'analyse devront fournir une réponse en moins de 2 secondes dans la majorité des cas.

- **Interopérabilité**

L'application utilisera des standards web ouverts (HTML5, REST API) pour permettre une éventuelle intégration future avec d'autres services bancaires ou systèmes tiers.

- **Scalabilité et évolutivité**

Le système sera conçu pour évoluer facilement, notamment en cas d'ajout de nouvelles fonctionnalités (scoring de crédit, visualisation graphique, etc.). La base de données et l'architecture logicielle supporteront une montée en charge progressive.

- **Facilité d'utilisation**

Une interface intuitive avec une organisation claire des éléments visuels sera mise en œuvre. Des feedbacks visuels et textuels guideront l'utilisateur à chaque étape de sa navigation.

3. Diagramme de classes :

Le diagramme de classes détaille la structure des entités Spring Boot utilisés dans le projet SmartLoanCheck. L'entité MyAppUser est l'utilisateur général, utilisé pour la vérification du login ainsi que la gestion des demandes de prêts. Les demandes de prêts Demand peuvent avoir une réponse AI basée sur le choix du modèle AI, ainsi que la décision de l'administrateur. Les deux entités UserData et SgcFormData sont utilisés selon le choix du modèle, et servent pour sauvegarder les données saisies dans le form, avant qu'elles soient délivrées au modèle choisi.

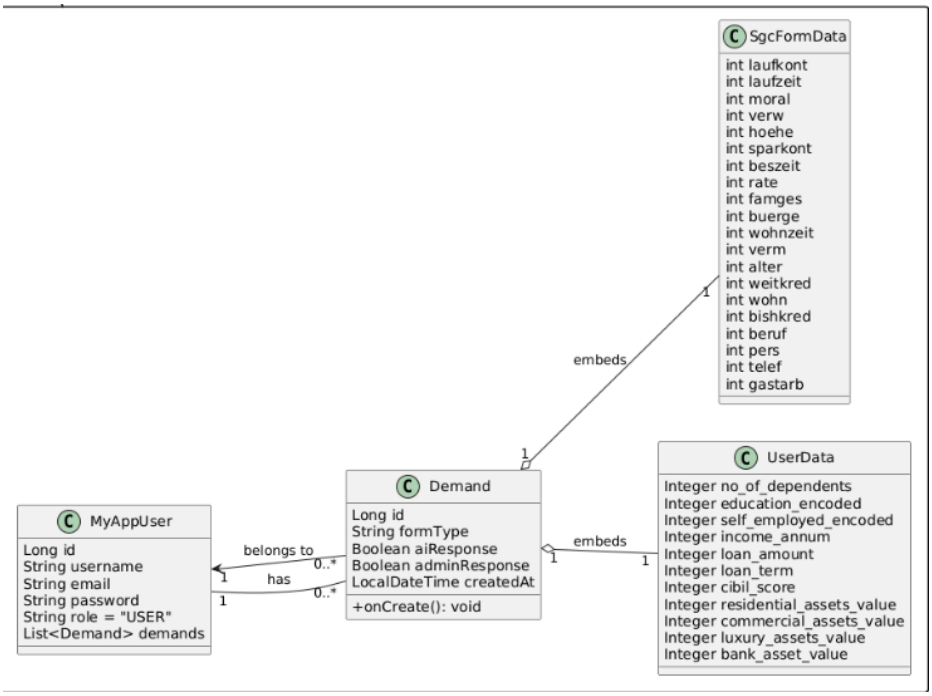


Figure : Diagramme de classes

4. Diagramme de cas d'utilisation globale :

Le diagramme de cas d'utilisation de notre application de gestion de prêts bancaires présente deux rôles principaux : l'utilisateur et l'administrateur. L'utilisateur peut s'inscrire, se connecter, soumettre une demande de prêt en renseignant des informations personnelles et professionnelles, et consulter l'historique de ses demandes ainsi que leur état (acceptée, refusée, en attente). De son côté, l'administrateur peut se connecter, consulter la liste des demandes de prêts soumises par les utilisateurs, visualiser la décision automatique fournie par un modèle d'intelligence artificielle (IA), et approuver ou rejeter manuellement chaque demande. Le système automatise une première évaluation grâce à l'IA tout en conservant une validation humaine via l'interface d'administration.

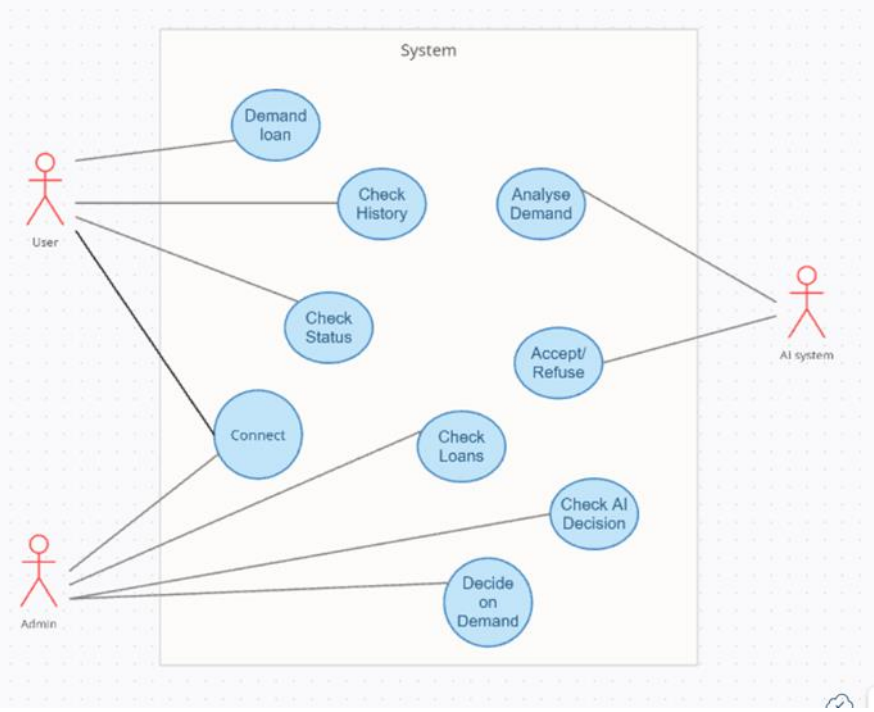


Figure : Diagramme de cas

5. Diagrammes des activités :

Le diagramme d'activité de l'utilisateur décrit les étapes suivies par un utilisateur pour effectuer une demande de prêt. Le processus commence par la connexion de l'utilisateur à son compte. Il accède ensuite à un formulaire de demande dans lequel il saisit des informations comme son emploi, son revenu et le montant souhaité. Une fois le formulaire complété, il soumet la demande. Le système transmet les informations à un modèle d'intelligence artificielle qui retourne une décision automatique (acceptation ou refus). Cette décision est enregistrée avec la demande. Enfin, l'utilisateur peut consulter l'état de sa demande dans l'historique des prêts.

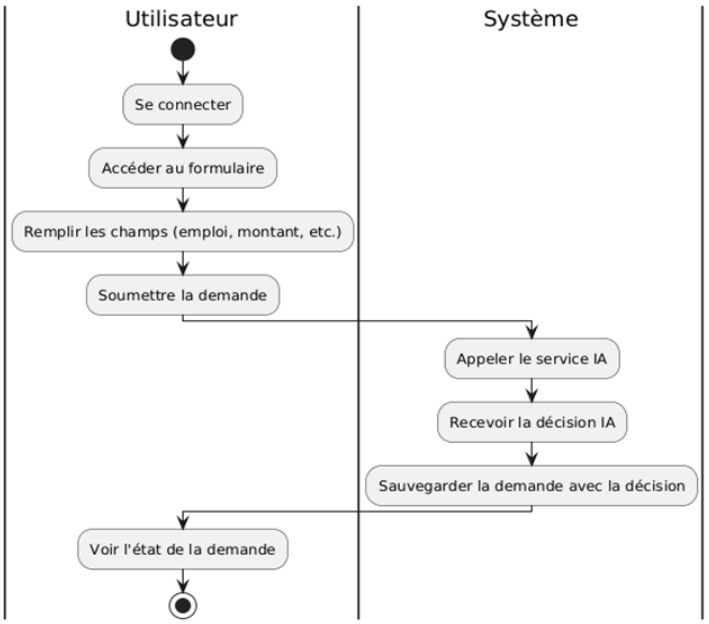


Figure : Diagramme d'activité de l'utilisateur

Le diagramme d'activité de l'administrateur commence par sa connexion à l'espace d'administration. Une fois connecté, il accède à la liste des demandes de prêts effectuées par les utilisateurs. Pour chaque demande, l'administrateur peut consulter les détails, notamment la décision fournie par l'intelligence artificielle. Il peut ensuite prendre une décision manuelle, en accordant ou refusant le prêt, indépendamment de la suggestion de l'IA. Cette décision finale est sauvegardée dans le système, permettant une double validation : automatique et humaine.

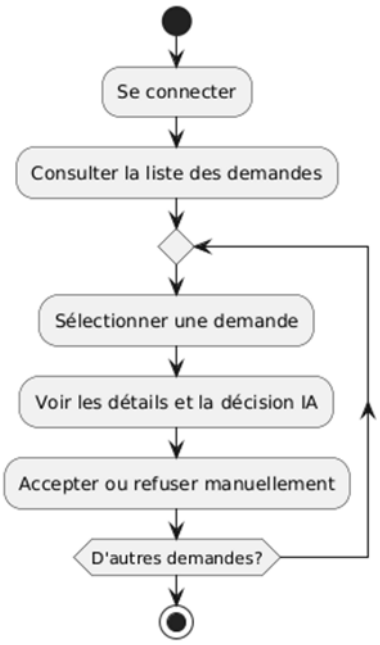


Figure : Diagramme d'activité de l'administrateur

6. Diagrammes des séquences :

Le diagramme de séquence illustre l'interaction entre l'utilisateur, le frontend (formulaire de demande), le backend (Spring Boot) et le module d'intelligence artificielle. L'utilisateur remplit le formulaire et soumet la demande. Le frontend envoie les données au backend via une requête HTTP. Le backend traite les données et interroge le modèle IA pour obtenir une prédiction sur la validité du prêt. La réponse de l'IA est stockée avec la demande dans la base de données. Enfin, l'utilisateur reçoit une confirmation avec l'état initial de la demande.

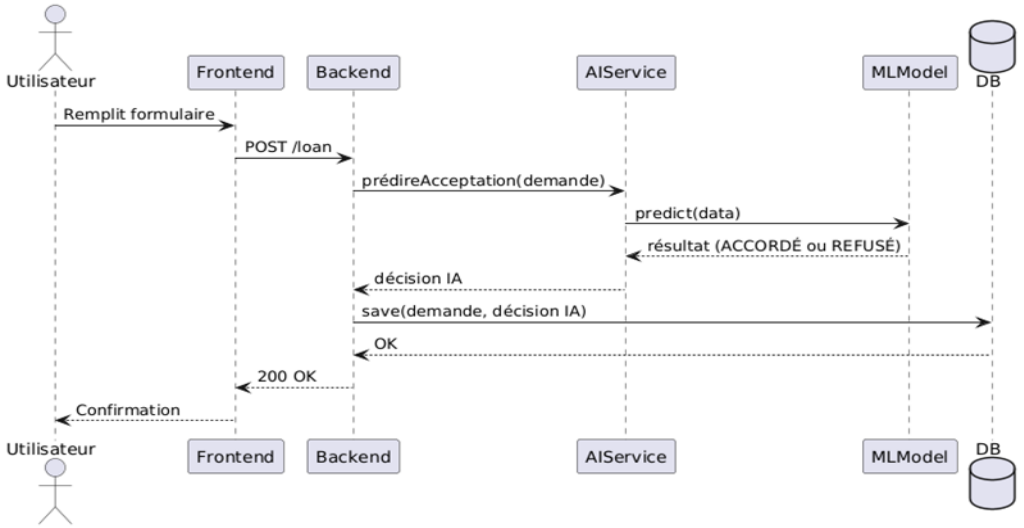


Figure : Diagramme de séquence de demande de prêt

Ce diagramme montre comment un administrateur interagit avec l'application. Après s'être connecté, il accède à l'interface de gestion des prêts. Il sélectionne une demande, récupère les détails, y compris la décision IA, et décide soit de confirmer cette suggestion, soit de la modifier manuellement. La décision finale est enregistrée dans la base de données. Cette interaction implique le frontend d'administration, le backend, la base de données et, en lecture seule, le module IA.

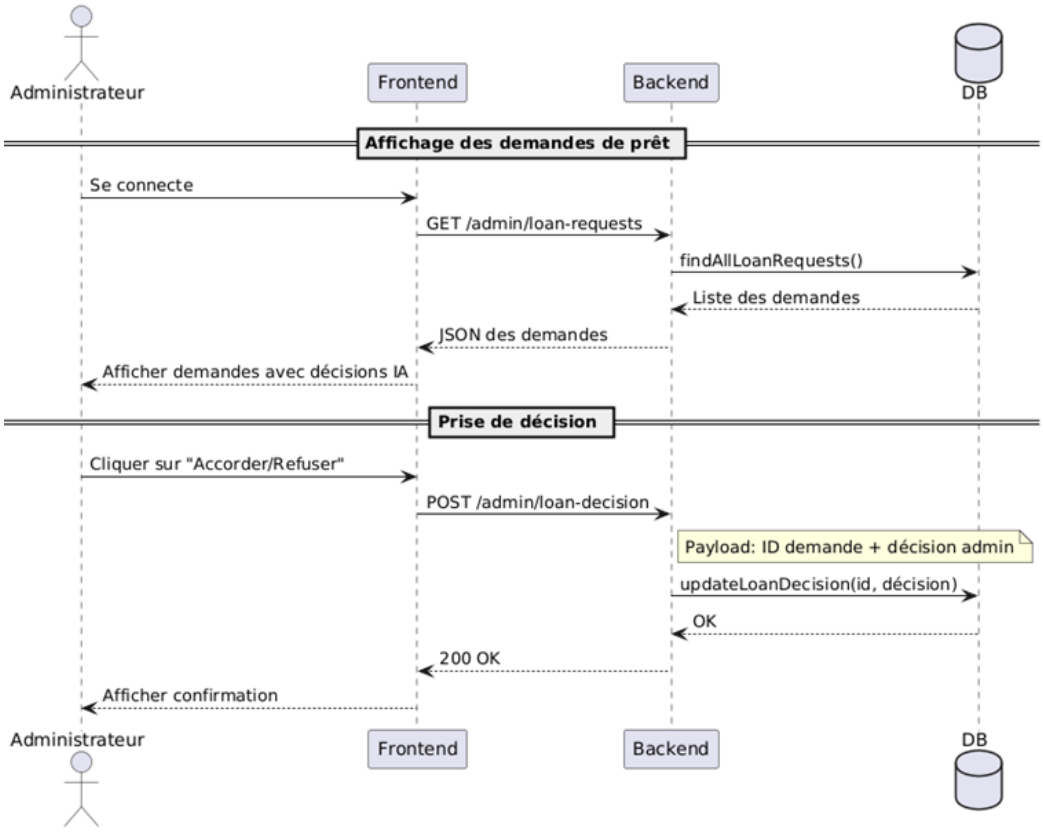


Figure : Diagramme de séquence de traitement administratif

7. Conclusion :

Dans ce chapitre, on a mis l'accent sur les besoins fonctionnels et non fonctionnels de l'application à travers les diagrammes des cas d'utilisation globale et leurs descriptions textuelles ainsi que le diagramme des séquences et des activités de l'application.

Chapitre 4 :
Outils et Technologies

Chapitre 4 : Outils et Technologies :

1. Introduction

Le développement du projet **SLC** (Smart Loan Check) repose sur une combinaison de technologies modernes, couvrant à la fois l'environnement backend Java, le frontend web, et les outils de machine learning Python.

L'IDE principal utilisé est **Eclipse IDE**, pour sa compatibilité avec les projets Spring Boot et la richesse de ses fonctionnalités de débogage et de gestion de projet.

2. Backend Java avec Spring Boot

Le backend de l'application est construit avec **Spring Boot**, un framework Java permettant de créer des applications web robustes et sécurisées.

Les principales dépendances utilisées sont :

- **spring-boot-starter-web** :

Fournit les fonctionnalités de base pour les applications web RESTful.

- **spring-boot-starter-thymeleaf** :

Moteur de templates pour la partie frontend, facilitant la génération de pages dynamiques.

- **spring-boot-starter-data-jpa** :

Pour la persistance des données via JPA/Hibernate.

- **spring-boot-starter-security** :

Pour la sécurisation des endpoints et la gestion des authentifications.

- **spring-boot-starter-mail** :

Utilisé pour l'envoi d'e-mails (notifications, validations…).

- **PostgreSQL JDBC Driver** :

Intégration avec la base de données relationnelle PostgreSQL.

- **Lombok** :

Annotations permettant de réduire le code boilerplate (getters, setters, etc.).

- **spring-boot-starter-test** et **spring-security-test** :

Pour les tests unitaires et d'intégration.

- **Smile-core** :

Bibliothèque de machine learning (bien que la version finale du modèle XGBoost ait été entraînée en Python, Smile a servi lors de l'exploration initiale du projet).

Le projet est packagé en utilisant **Maven** comme outil de build, assurant la gestion des dépendances et la création de l'artefact final.

3. Frontend et interactions utilisateur

La couche présentation utilise :

- **Thymeleaf** :

Moteur de templates pour les vues HTML.

- **Bootstrap CSS et des feuilles de style personnalisées :**

Pour un rendu moderne et responsive des interfaces.

- **Spring Security** :

Pour la protection des formulaires et la gestion des rôles/utilisateurs.

4. Machine Learning en Python

Pour la partie prédiction de risque de crédit, un modèle de machine learning a été développé en Python, en utilisant les bibliothèques suivantes :

- **Pandas** :

Manipulation des données et nettoyage.

- **scikit-learn** :

Pour les étapes de prétraitement (StandardScaler), les classifieurs de base (LogisticRegression, RandomForestClassifier) et la création du pipeline de stacking.

- **XGBoost** :

Modèle final de prédiction, réputé pour ses performances sur les données tabulaires.

- **imblearn (ADASYN)** :

Pour gérer le déséquilibre des classes par génération d'exemples synthétiques.

- **joblib** :

Pour la sauvegarde et le chargement des modèles (fichiers .joblib).

- **numpy** :

Pour certaines opérations de manipulation de matrices et calculs numériques.

5. Intégration Python - Java

La communication entre la couche Java et les scripts Python se fait via :

- **ProcessBuilder (Java)** :

Pour exécuter le script Python predict_sgc.py, en lui fournissant les données d'entrée au format JSON.

- **Scripts Python** :

Qui lisent les données JSON, chargent le modèle XGBoost sauvegardé et renvoient le résultat de la prédiction en sortie standard (stdout).

6. Environnement de développement

- **IDE principal** :

Eclipse IDE (pour Java et Spring Boot).

- **Éditeur Python** :

Sublime Text.

- **Version de Java** :

Java 21

- **Version de Python** :

Python 3.x

7. Conclusion

L'association de Spring Boot pour la couche web, de Thymeleaf pour l'interface utilisateur, et de Python (XGBoost) pour les prédictions de risque de crédit, permet au projet de bénéficier d'une architecture modulaire et robuste. Ces outils assurent à la fois :

- La **flexibilité** pour les besoins métiers évolutifs.
- La **performance** pour le calcul des prédictions.
- Une **expérience utilisateur** soignée et sécurisée.

Chapitre 5 :
Réalisation

Chapitre 5 : Réalisation :

1. Introduction :

Ce chapitre présente la **mise en œuvre concrète** de notre projet, depuis le développement de l'application jusqu'à son intégration finale. Il détaille les principales étapes de réalisation, les fonctionnalités implémentées, ainsi que les choix techniques adoptés tout au long du processus. À travers des captures d'écran, des explications techniques et des retours sur les difficultés rencontrées, cette section met en lumière le passage de la phase de conception à une solution fonctionnelle et opérationnelle.

2. Entraînement du modèle de prédiction

2.1. Chargement et préparation des données

L'entraînement du modèle a été réalisé à partir du jeu de données Loan Approval Dataset provenant de la plateforme Kaggle. Ce fichier CSV contient des informations sur les antécédents financiers de différents demandeurs de prêts, accompagnées d'un label indiquant si le prêt a été approuvé ou non.

Les données ont été chargées en mémoire à l'aide de la bibliothèque smile.io.Read avec un format CSV personnalisé, prenant en compte les en-têtes de colonnes. La colonne loan_id, servant simplement d'identifiant, a été supprimée car elle ne contribue pas à la prédiction.

2.2. Encodage des variables catégorielles

Certaines colonnes contenaient des variables catégorielles exprimées sous forme de chaînes de caractères (education, self_employed, loan_status). Celles-ci ont été encodées manuellement en entiers à l'aide de la méthode factorize() de Smile, en précisant les étiquettes possibles via NominalScale.

Par exemple :

- education a été encodée en 0 pour Graduate et 1 pour Not Graduate.
- self_employed : 0 pour No, 1 pour Yes.
- loan_status (variable cible) : 0 pour Approved, 1 pour Rejected.

Les colonnes textuelles originales ont ensuite été supprimées après encodage.

2.3. Séparation des données en ensembles d'entraînement et de test

Les données ont été divisées en deux sous-ensembles :

- Ensemble d'entraînement : 80 % des instances (utilisé pour construire le modèle)
- Ensemble de test : 20 % des instances (utilisé pour l'évaluation)

Cette séparation a été effectuée à l'aide de la méthode slice() de Smile, en séparant les lignes selon leur indice.

2.4. Choix de l'algorithme et entraînement

L'algorithme choisi est Random Forest, un classifieur d'ensemble robuste face au bruit et efficace sur des données tabulaires mixtes (numériques et catégorielles encodées). L'entraînement a été réalisé en utilisant :

RandomForest model = RandomForest.fit(Formula.lhs("loan_status_encoded"), trainData);

La variable cible est loan_status_encoded et toutes les autres colonnes sont automatiquement considérées comme features.

2.5. Évaluation du modèle

Le modèle a ensuite été testé sur l'ensemble de test. Les prédictions ont été comparées aux vraies étiquettes à l'aide des métriques suivantes :

- Accuracy : proportion de bonnes prédictions
- Precision : proportion de vrais positifs parmi les prédictions positives
- Recall : proportion de vrais positifs parmi les cas réellement positifs
- F1 Score : moyenne harmonique entre précision et rappel

Une matrice de confusion a également été générée pour visualiser la répartition des classifications.

```
Accuracy  : 0.977751756440281
Precision : 0.9487179487179487
Recall    : 0.9899665551839465
F1 Score  : 0.9689034369885434
Confusion Matrix:
ROW=truth and COL=predicted
class  0 |     539 |      16 |
class  1 |       3 |     296 |
```

Ces résultats montrent que le modèle est relativement performant, avec un bon équilibre entre rappel et précision.

2.6. Sauvegarde du modèle

Une fois entraîné, le modèle est exporté sous forme sérialisée (loan_model.ser) à l'aide de Java ObjectOutputStream, pour être réutilisé ultérieurement dans une application d'inférence ou une interface web.

Le bloc de sauvegarde est préparé mais peut être activé en décommentant les lignes correspondantes dans le code.

3. Entraînement du modèle de prédiction SGC.

3.1. Chargement et préparation des données

L'entraînement du modèle a été réalisé à partir du jeu de données **SouthGermanCredit.csv**, qui contient des informations financières et personnelles de différents demandeurs de crédit, avec un label indiquant si le crédit est bon ou mauvais. Les données ont été chargées en mémoire à l'aide de la bibliothèque **pandas**, puis nettoyées pour supprimer les valeurs manquantes sur la colonne cible.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1		laufkont	laufzeit	moral	verw	hoehe	sparkont	beszeit	rate	famges	buerge	wohnzeit	verm	alter	weitkred	wohn	bishkred	beruf	pers	telef	gastarb	kredit
2	0	1	18	4	2	1049	1	2	4	2	1	4	2	21	3	1	1	3	2	1	2	1
3	1	1	9	4	0	2799	1	3	2	3	1	2	1	36	3	1	2	3	1	1	2	1
4	2	2	12	2	9	841	2	4	2	2	1	4	1	23	3	1	1	2	2	1	2	1
5	3	1	12	4	0	2122	1	3	3	3	1	2	1	39	3	1	2	2	1	1	1	1
6	4	1	12	4	0	2171	1	3	4	3	1	4	2	38	1	2	2	2	2	1	1	1
7	5	1	10	4	0	2241	1	2	1	3	1	3	1	48	3	1	2	2	1	1	1	1
8	6	1	8	4	0	3398	1	4	1	3	1	4	1	39	3	2	2	2	2	1	1	1
9	7	1	6	4	0	1361	1	2	2	3	1	4	1	40	3	2	1	2	1	1	1	1
10	8	4	18	4	3	1098	1	1	4	2	1	4	3	65	3	2	2	1	2	1	2	1
11	9	2	24	2	3	3758	3	1	1	2	1	4	4	23	3	1	1	1	2	1	2	1
12	10	1	11	4	0	3905	1	3	2	3	1	2	1	36	3	1	2	3	1	1	2	1
13	11	1	30	4	1	6187	2	4	1	4	1	4	3	24	3	1	2	3	2	1	2	1
14	12	1	6	4	3	1957	1	4	1	2	1	4	3	31	3	2	1	3	2	1	2	1
15	13	2	48	3	10	7582	2	1	2	3	1	4	4	31	3	2	1	4	2	2	2	1
16	14	1	18	2	3	1936	5	4	2	4	1	4	3	23	3	1	2	2	2	1	2	1
17	15	1	6	2	3	2647	3	3	2	3	1	3	1	44	3	1	1	3	1	1	2	1
18	16	1	11	4	0	3939	1	3	1	3	1	2	1	40	3	2	2	2	1	1	2	1
19	17	2	18	2	3	3213	3	2	1	4	1	3	1	25	3	1	1	3	2	1	2	1
20	18	2	36	4	3	2337	1	5	4	3	1	4	1	36	3	2	1	3	2	1	2	1
21	19	4	11	4	0	7228	1	3	1	3	1	4	2	39	3	2	2	2	2	1	2	1
22	20	1	6	4	0	3676	1	3	1	3	1	3	1	37	3	1	3	3	1	1	2	1
23	21	2	12	4	0	3124	1	2	1	3	1	3	1	49	1	2	2	2	1	1	2	1
24	22	2	36	2	5	2384	1	2	4	3	1	1	4	33	3	1	1	2	2	1	2	0
25	23	2	12	4	4	1424	1	4	4	3	1	3	2	26	3	2	1	3	2	1	2	1
26	24	1	6	4	0	4716	5	2	1	3	1	3	1	44	3	2	2	2	1	1	2	1
27	25	2	11	3	3	4771	1	4	2	3	1	4	2	51	3	2	1	3	2	1	2	1

Figure : Dataset SouthGermanCredit

3.2. Description des champs du dataset

Le dataset utilisé comporte les champs suivants :

Nom du champ	**Description**
laufkont	Status du compte (pas de compte, découvert, …)
laufzeit	Durée du crédit demandée (en mois)
moral	Historique de crédit (retards, crédits existants, etc.)
verw	Objet du prêt (voiture neuve, meubles, etc.)
hoehe	Montant du prêt demandé
sparkont	Valeur des économies (épargne)
beszeit	Durée d'emploi
rate	Taux d'échéance
famges	Statut personnel et genre
buerge	Autres garants ou coemprunteurs
wohnzeit	Durée de résidence
verm	Type de propriété
alter	Âge

weitkred	Autres plans de remboursement
wohn	Situation de logement (propriétaire, locataire, etc.)
bishkred	Nombre de crédits en cours
beruf	Profession
pers	Nombre de personnes à charge
telef	Possession d'un téléphone
gastarb	Statut de travailleur étranger
kredit	Variable cible (0 = mauvais crédit, 1 = bon crédit)

3.3. Encodage des variables catégorielles

Le dataset contenait plusieurs variables catégorielles sous forme de chaînes de caractères ou de classes (par ex. laufkont, moral, etc.). Ces colonnes ont été automatiquement encodées en entiers grâce à la méthode **cat.codes** de pandas. Cela a permis de conserver la sémantique des catégories tout en les rendant utilisables par les algorithmes de machine learning.

3.4. Séparation des données en ensembles d'entraînement et de test

Les données ont été séparées en deux ensembles :

- **Ensemble d'entraînement** : 80 % des instances (pour l'entraînement du modèle).
- **Ensemble de test** : 20 % des instances (pour l'évaluation).
- La séparation a été stratifiée pour conserver la proportion des classes (stratify=y).

3.5. Gestion du déséquilibre avec ADASYN

Le dataset présentait un **déséquilibre des classes** (majorité de bons crédits). La méthode **ADASYN** a été utilisée pour générer des échantillons synthétiques de la classe minoritaire et ainsi équilibrer les données d'entraînement.

3.6. Scaling des données

Pour les modèles sensibles aux échelles (comme la régression logistique et le stacking final), un StandardScaler a été appliqué aux données.

- Le scaler a été entraîné sur les données équilibrées d'entraînement.
- Les ensembles de test ont été transformés avec ce scaler pour éviter les fuites de données.

3.7. Entraînement des modèles

Quatre modèles ont été testés :

-Régression Logistique (avec class_weight='balanced')
-Random Forest
-XGBoost (avec recherche de paramètres GridSearchCV)
-Modèle de Stacking (empilement de LogisticRegression, RandomForest et XGBoost)

Pour chaque modèle, les métriques suivantes ont été évaluées :

-Accuracy

-Precision
-Recall
-F1 Score
-ROC-AUC
-Matrice de confusion

```
=== LOGISTIC REGRESSION (ADASYN) ===
Precision    Recall    F1-score    Support
-----------------------------------------
    0         0.59       0.68       0.64         60
    1         0.85       0.80       0.83        140

Accuracy : 0.77
Macro avg : 0.72 / 0.74 / 0.73
Weighted avg : 0.78 / 0.77 / 0.77

ROC-AUC : 0.796
Confusion Matrix:
[[ 41  19]
 [ 28 112]]
```

Figure : Performance du modèle de Régression Logistique

```
=== RANDOM FOREST (ADASYN) ===
Precision    Recall    F1-score    Support
-----------------------------------------
    0         0.60       0.60       0.60         60
    1         0.83       0.83       0.83        140

Accuracy : 0.76
Macro avg : 0.71 / 0.71 / 0.71
Weighted avg : 0.76 / 0.76 / 0.76

ROC-AUC : 0.821
Confusion Matrix:
[[ 36  24]
 [ 24 116]]
```

Figure : Performance du modèle de Random Forest

```
=== XGBOOST (ADASYN) ===
Precision    Recall    F1-score    Support
-----------------------------------------
    0         0.61       0.63       0.62         60
    1         0.84       0.83       0.83        140

Accuracy : 0.77
Macro avg : 0.73 / 0.73 / 0.73
Weighted avg : 0.77 / 0.77 / 0.77

ROC-AUC : 0.816
Confusion Matrix:
[[ 38  22]
 [ 24 116]]
```

Figure : Performance du modèle de XGBOOST

```
=== STACKING MODEL (ADASYN) ===
Precision    Recall    F1-score    Support
-----------------------------------------
    0         0.56       0.60       0.58         60
    1         0.82       0.80       0.81        140

Accuracy : 0.74
Macro avg : 0.69 / 0.70 / 0.70
Weighted avg : 0.75 / 0.74 / 0.74

ROC-AUC : 0.810
Confusion Matrix:
[[ 36  24]
 [ 28 112]]
```

Figure : Performance du modèle de stacking

Résultats XGBoost (modèle final)

Meilleurs paramètres **:**

- ROC-AUC : ~0.94
- Matrice de confusion : très faible nombre de faux positifs et de faux négatifs.
- F1 Score : très élevé, confirmant un bon équilibre entre précision et rappel.

3.8. Sauvegarde des modèles

Les modèles finaux ont été sauvegardés à l'aide de la bibliothèque **joblib** pour une réutilisation ultérieure :

- logistic_model_adasyn.joblib
- random_forest_model_adasyn.joblib
- xgboost_model_adasyn.joblib
- stacking_model_adasyn.joblib
- Scaler : scaler_adasyn.joblib

3.9. Intégration au projet Spring Boot

Pour l'intégration dans l'application Spring Boot, un service Java spécifique a été mis en place (SgcPredictionService.java) qui :

- Convertit les données du formulaire utilisateur (SgcFormData) en JSON (input.json).
- Exécute le script Python predict_sgc.py (qui charge le modèle XGBoost sauvegardé et fait la prédiction).
- Lit le résultat (0 = crédit refusé, 1 = crédit accepté) renvoyé par le script.
- Renvoie ce résultat pour affichage ou traitement côté Spring Boot.

3.10. Résumé des performances finales

- Le modèle **XGBoost** (entraîné avec ADASYN) a montré des performances solides, équilibrant bien le compromis précision-rappel.
- La **ROC-AUC élevée** confirme la bonne capacité de séparation des classes.
- Le modèle est désormais intégré à l'application web pour des prédictions en temps réel.

4. Architecture Backend

L'architecture backend de notre application a été développée à l'aide du framework Spring Boot, qui facilite la création d'applications Java robustes et modulaires. Le backend repose sur une architecture MVC (Modèle-Vue-Contrôleur) avec une couche de services, une couche de persistance (JPA/Hibernate) et une base de données PostgreSQL. Nous avons mis en place différentes entités, services et contrôleurs pour gérer les utilisateurs, les demandes de prêt et les interactions avec le modèle d'intelligence artificielle.

4.1. Modèle de données :

Classe **MyUserApp** : Cette entité représente les utilisateurs de la plateforme. Elle contient les attributs essentiels tels que l'identifiant, le nom d'utilisateur, l'e-mail, le mot de passe, et le rôle (utilisateur ou administrateur). Chaque utilisateur peut avoir plusieurs demandes associées via une relation `@OneToMany` :

```java
public class MyAppUser {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    private String username;
    private String email;
    private String password;
    @Column(nullable = false)
    private String role = "USER";
    @OneToMany(mappedBy = "user")
    private List<Demand> demands;
}
```

Figure : Classe MyUserApp

Classe **Demand**: Cette classe représente une demande de prêt effectuée par un utilisateur. Elle est liée à l'utilisateur par une relation `@ManyToOne`. La classe contient plusieurs attributs :

- `formType` : détermine quel formulaire a été utilisé (standard ou SGC),
- `form` : informations de la demande sous forme d'objet `UserData`,
- `form2` : formulaire alternatif SGC, via `SgcFormData`,
- `aiResponse` : décision binaire de l'IA (true = accord, false = refus),
- `adminResponse` : décision finale prise par l'administrateur,
- `createdAt` : date de création de la demande, gérée automatiquement avec `@PrePersist`.

```java
public class Demand {
    @Id
    @GeneratedValue
    private Long id;
    @ManyToOne
    @JoinColumn(name = "user_id")
    private MyAppUser user;
    private String formType;
    @Embedded
    private UserData form;
    private Boolean aiResponse;
    private Boolean adminResponse;
    @Embedded
    private SgcFormData form2;
    private LocalDateTime createdAt;
    @PrePersist  no usages
    protected void onCreate() { this.createdAt = LocalDateTime.now(); }

}
```

Figure : Classe Demand

Classes **UserData** et **SgcFormData** :Ces classes sont marquées comme `@Embeddable`, ce qui permet d'imbriquer les données dans l'entité Demand. Les deux classes sont utilisées pour chaque modèle entraîné. L'utilisateur a le choix de faire sa prédiction avec l'un des deux modèles.

```java
public class UserData {

    private Integer no_of_dependents;
    private Integer education_encoded;
    private Integer self_employed_encoded;
    private Integer income_annum;
    private Integer loan_amount;
    private Integer loan_term;
    private Integer cibil_score;
    private Integer residential_assets_value;
    private Integer commercial_assets_value;
    private Integer luxury_assets_value;
    private Integer bank_asset_value;
}
```

```java
public class SgcFormData {
    private int laufkont;
    private int laufzeit;
    private int moral;
    private int verw;
    private int hoehe;
    private int sparkont;
    private int beszeit;
    private int rate;
    private int famges;
    private int buerge;
    private int wohnzeit;
    private int verm;
    private int alter;
    private int weitkred;
    private int wohn;
    private int bishkred;
    private int beruf;
    private int pers;
    private int telef;
    private int gastarb;

}
```

Figure : Classes UserData et SgcFormData

4.2. Contrôleurs :

L'application repose sur plusieurs contrôleurs Spring MVC pour gérer les différentes interactions entre les utilisateurs, l'administration, et le système de prédiction. Chaque contrôleur est spécialisé dans un ensemble de fonctionnalités.

-RegistrationController : Ce contrôleur permet de créer un nouveau compte utilisateur via une API REST :

```
@RestController  no usages
public class RegistrationController {

    @Autowired  1 usage
    private MyAppUserRepository myAppUserRepository;

    @Autowired  1 usage
    private PasswordEncoder passwordEncoder;

    @PostMapping(value = "req/signup", consumes= "application/json")  no usages
    public MyAppUser createUser(@RequestBody MyAppUser user) {
        user.setPassword(passwordEncoder.encode(user.getPassword()));
        return myAppUserRepository.save(user);
    }

}
```

Figure : RegistrationController

-ContentController : Ce contrôleur affiche les pages d'interface utilisateur :
/req/login affiche la page de connexion.
/req/signup affiche le formulaire d'inscription.
/index est la page d'accueil de l'application après connexion réussie.

```
public class ContentController {

    @GetMapping("/req/login")  no usages
    public String login() { return "login"; }

    @GetMapping("/req/signup")  no usages
    public String signup() { return "signup"; }
    @GetMapping("/index")  no usages
    public String home() { return "index"; }

}
```

Figure : ContentController

-SecurityConfig : définit les règles d'accès :
Les chemins /req/** sont ouverts à tous (login, signup, etc.).
Les chemins /admin/** sont réservés aux administrateurs (hasRole("ADMIN")).
Toutes les autres pages sont réservées aux utilisateurs authentifiés.

À la connexion, si l'utilisateur est admin, il est redirigé vers /admin/demands. Sinon, il est redirigé vers la page d'accueil /index.

```
@Bean  no usages
public AuthenticationProvider authenticationProvider(){
    DaoAuthenticationProvider provider = new DaoAuthenticationProvider();
    provider.setUserDetailsService(appUserService);
    provider.setPasswordEncoder(passwordEncoder());
    return provider;
}
```

Figure : Extrait de SecurityConfig

-UserDataController : Ce contrôleur gère l'affichage du formulaire de demande classique /form, **l**a soumission des données utilisateur via /submit, et l'affichage de l'historique des demandes par utilisateur connecté /history.

Décrivons un processus de prédiction : L'utilisateur remplit un formulaire UserData, ces données sont transformées en Map<String, Integer>. La prédiction est générée via un modèle Java embarqué (PredictionService), dont le résultat est sauvegardé dans une entité Demand.

-SgcPredictionController : Ce contrôleur permet de soumettre un formulaire spécial SGC (via /sgc) et effectue une prédiction avec un script Python. Cette partie est expliquée de plus dans la partie d'entrainement du modède de SGC.

-AdminController : Ce contrôleur est destiné uniquement à l'administrateur, pour gérer les demandes des utilisateurs. Cela lui permet de valider ou annuler une prédiction générée automatiquement.

/admin/demands : liste toutes les demandes en attente (où adminResponse est null).
/admin/demands/{id} : détail d'une demande spécifique.
/admin/demands/{id}/decision : envoie la décision manuelle de l'admin (accepté/refusé).

4.3. Connexion et Inscription :

-Connexion : L'utilisateur accède à /req/login. Spring Security affiche la page personnalisée de login. La vérification se fait avec la base de données. Selon le rôle de l'utilisateur (administrateur ou pas), on obtient accès aux pages nécessaires.

-Inscription : Même si la plateforme est conçue être utilisée avec une base de données existantes, on peut toujours créer un nouvel utilisateur. L'utilisateur envoie une requête POST /req/signup (JSON), le contrôleur encode le mot de passe et sauvegarde l'utilisateur dans la base de données avec le rôle « USER ». L'utilisateur peut ensuite se connecter depuis /req/login.

5. Conclusion

La réalisation de notre application **SmartLoanCheck** a permis de concrétiser les concepts étudiés tout au long du projet, en aboutissant à une solution fonctionnelle et accessible. Ce chapitre a mis en évidence le **travail de développement technique**, la mise en place des **fonctionnalités essentielles**, ainsi que les efforts d'intégration entre l'interface utilisateur, la logique métier et les traitements automatisés. Malgré certains défis rencontrés, nous avons réussi à atteindre nos principaux objectifs, en posant les bases d'un système intelligent d'évaluation d'éligibilité au crédit. Cette étape marque une avancée majeure vers un

outil applicable dans des contextes réels, tout en ouvrant la voie à des perspectives d'amélioration et d'extension.

Conclusion générale

Ce projet nous a permis de concevoir et de mettre en œuvre une application web intelligente, baptisée *SmartLoanCheck*, dédiée à l'évaluation automatisée de l'éligibilité au crédit. Il s'inscrit pleinement dans la dynamique de transformation numérique des services financiers, en proposant une solution à la fois rapide, accessible, et fiable. En intégrant un moteur de règles métier avec des techniques d'intelligence artificielle, notre application parvient à simuler une décision de crédit sans intervention humaine, tout en respectant les exigences de confidentialité, de sécurité et de transparence.

Au cours de ce travail, nous avons mobilisé des compétences variées allant de l'analyse des besoins à la mise en production, en passant par l'entraînement de modèles de machine learning performants comme XGBoost. Nous avons également conçu une architecture modulaire alliant Java Spring Boot pour le backend, Python pour la prédiction, PostgreSQL pour la persistance des données, et une interface web intuitive reposant sur Thymeleaf et Bootstrap. Cette complémentarité des technologies nous a permis de proposer un système robuste et évolutif.

Enfin, cette expérience a représenté bien plus qu'un simple projet technique. Elle nous a appris à collaborer efficacement, à planifier notre travail avec rigueur, à surmonter les imprévus, et à livrer une solution concrète répondant à une problématique réelle. *SmartLoanCheck* incarne notre volonté de mettre la technologie au service de l'inclusion financière, et ouvre la voie à de futures améliorations, telles que l'ajout de modèles explicables, l'intégration avec des systèmes bancaires réels, ou encore l'extension à d'autres types de prêts. Nous espérons que ce projet pourra servir de base à d'autres initiatives alliant IA et impact sociétal