

LANDWISE

Loan Recommendation & Tracking Platform

Rapport de Projet de Fin d'Études – Licence en Génie Logiciel

Réalisé intégralement par :

Ingénieur en Génie Logiciel

Entreprise d'accueil : VERMEG – Paris

Année académique : 2023 – 2024

1. Présentation Générale du Projet

Nom du projet	Landwise – Plateforme de Recommandation et de Suivi de Crédits
Réalisé par	Mohamed Rayen Farhat
Diplôme	Licence en Génie Logiciel
Entreprise	VERMEG – Solutions Logicielles Financières, Paris
Client	VERMEG / Satoripop
Méthodologie	Scrum (sprints de 2 semaines, outil : Jira Software)
Durée	Février 2023 – Juin 2023
Type d'application	Web Full-Stack (MERN Stack + Flask/Python IA)

Contexte et Problématique

Dans un environnement économique où la gestion financière personnelle est essentielle, accéder et gérer des produits de crédit reste complexe et chronophage pour les particuliers. Les consommateurs se heurtent à trois obstacles majeurs : la difficulté de sélectionner l'offre de crédit la mieux adaptée parmi une multitude d'options, le manque d'informations fiables sur les conditions de remboursement, et l'absence de personnalisation des recommandations selon le profil financier de chaque utilisateur. Ce projet, réalisé intégralement par Mohamed Rayen Farhat dans le cadre de son stage de fin d'études chez VERMEG, répond directement à ces enjeux.

Solution Proposée – Landwise

Landwise est une application web complète qui révolutionne l'interaction des utilisateurs avec les produits financiers. Elle permet de parcourir et comparer une large gamme de contrats de crédit, de consulter les détails de chaque offre (taux, durée, conditions), de soumettre une demande directement sur la plateforme, et de recevoir des recommandations personnalisées via un module d'intelligence artificielle entraîné sur des données réelles. Un panneau d'administration complet permet à l'admin de gérer intégralement le catalogue de crédits.

2. Stack Technique & Environnement de Développement

Technologies Utilisées

MongoDB	Express.js	React.js
Node.js	Flask (Python)	Redux Toolkit
Firebase Auth	React Router DOM	Tailwind CSS
Google OAuth	JWT	StarUML
Jira Software	GitHub	Visual Studio Code

Architecture MERN + IA

Frontend (Client)	React.js – composants fonctionnels, Redux Toolkit pour la gestion d'état, React Router DOM pour la navigation, Tailwind CSS pour le design responsive.
Backend (Serveur)	Node.js + Express.js – API RESTful, gestion des routes et contrôleurs (auth, utilisateur, contrat). JWT pour la sécurisation des endpoints.
IA / ML	Flask (Python) – micro-framework exposant un modèle de classification entraîné sur un dataset CSV pour évaluer l'éligibilité au crédit et retourner les meilleurs contrats correspondants.
Base de données	MongoDB (NoSQL) – stockage des utilisateurs, contrats de crédit et données applicatives. Modèles : User, LoanContract.
Authentification	Firebase Auth (email/password + Google OAuth) + JWT pour la session.
Gestion de projet	Jira Software (Scrum) – backlog, sprints de 2 semaines, stories, retrospectives. Gantt couvrant Fév.–Juin 2023.

Architecture Projet (Arborescence)

api/controllers/	auth.controller.js · listing.controller.js · user.controller.js
api/models/	listing.model.js · user.model.js
api/routes/	auth.route.js · listing.route.js · user.route.js
api/utills/	error.js · verifyUser.js
client/src/components/	Header.jsx · OAuth.jsx · PrivateRoute.jsx · PrivateRouteAdmin.jsx
client/src/pages/	About · createListing · Home · listing · modal · Profile · SignIn · SignUp

client/src/redux/	user/userSlice.js · store.js
Flask (IA)	Modèle Python entraîné + API /predict exposée via Flask

3. Analyse des Besoins et Spécifications

Analyse de l'Existant

Avant de concevoir Landwise, une analyse comparative des plateformes existantes a été menée. Deux solutions majeures du marché ont été étudiées :

- **NerdWallet** – comparateur de produits financiers (cartes, prêts). Manque de personnalisation par profil et absence de gestion de contrats.
- **Credit Karma** – scores de crédit gratuits et recommandations basiques. Filtres partiels, pas de gestion de portefeuille de crédits.

Tableau Comparatif

Feature	NerdWallet	Credit Karma	Landwise
User-Friendly Interface	✓	✓	✓
Comprehensive Product Comparisons	✓	✓	✓
Personalized Credit Recommendations	✗	✗	✓
Customized Credit Contract Filters	✗	✓	✓
Credit Contract Management	✗	✗	✓

Besoins Fonctionnels

Gestion des accès	Authentification sécurisée (email/password + Google OAuth) pour utilisateurs et administrateurs.
Gestion de profil	Mise à jour des informations personnelles, photo de profil, mot de passe, suppression du compte.
Gestion des contrats	Parcourir, comparer, consulter les détails et gérer les contrats de crédit disponibles.
Recherche & Filtres	Barre de recherche avancée et filtres multicritères pour cibler les offres selon les besoins.
Favoris	Liste de contrats favoris consultable à tout moment depuis le profil.
Recommandation IA	Module Flask/Python évaluant l'éligibilité et proposant les meilleurs contrats correspondants.
Administration	Ajout, modification, suppression de contrats de crédit ; gestion du catalogue en temps réel.
Sécurité	Chiffrement des données, JWT, routes protégées (PrivateRoute / PrivateRouteAdmin).

Besoins Non-Fonctionnels

Performance	Temps de réponse optimisé ; gestion de volumes élevés d'interactions et de données.
Ergonomie	Interface intuitive et accessible, adaptée à tous les niveaux d'expertise technique.
Scalabilité	Architecture conçue pour monter en charge sans dégradation des performances.
Fiabilité	Haute disponibilité, reprise rapide après incident, cohérence des données.
Maintenabilité	Code modulaire (composants React, controllers Express), facile à faire évoluer.

4. Modélisation UML

Acteurs du Système

Utilisateur (User)	Compte personnel – peut parcourir et filtrer les contrats, tester son éligibilité IA, gérer ses favoris, mettre à jour son profil.
Administrateur (Admin)	Hérite de tous les droits utilisateur + peut créer, modifier et supprimer des contrats de crédit ; accès au panneau d'administration depuis son profil.

Cas d'Utilisation Principaux

4.3 – Authentification

Titre	Authenticate
Acteurs	User, Admin
Résumé	Connexion sécurisée via email/mot de passe ou Google OAuth (Firebase). Validation des credentials, génération d'un JWT, redirection vers le tableau de bord.
Scénario principal	1. Saisie email + mot de passe → 2. isValid() → 3. HTTP POST /login → 4. LoginCredentialsCheck → 5. Retour JWT + objet User
Scénario alternatif	Identifiants invalides → message d'erreur affiché.

4.3 – Inscription

Titre	Register
Acteurs	User
Résumé	Création de compte via formulaire (username, email, mot de passe) avec validation OTP par email, ou inscription directe via Google.
Scénario principal	1. Accès écran inscription → 2. Remplissage formulaire → 3. Envoi code OTP → 4. Validation → 5. Redirection accueil
Scénario alternatif	Email invalide / problème serveur / connexion internet.

4.3 – Gestion du Profil

Titre	Manage Profile
Acteurs	User, Admin
Résumé	Mise à jour de la photo, du nom d'utilisateur, de l'email ou du mot de passe. Possibilité de se déconnecter ou de supprimer le compte.
Scénario principal	1. Accès page profil → 2. Modification des informations → 3. isValid() → 4. HTTP PATCH /updateProfile → 5. Confirmation

Scénario alternatif	Format de données invalide / erreur serveur.
----------------------------	--

4.3 – Consultation des Contrats

Titre	View Loan Contracts
Acteurs	User, Admin
Résumé	Navigation dans la liste des contrats de crédit disponibles depuis la page d'accueil.
Scénario principal	1. Accès page d'accueil → 2. HTTP GET /listings → 3. Affichage de la grille de contrats
Scénario alternatif	Erreur serveur.

4.3 – Recherche Avancée

Titre	Manage Searches
Acteurs	User, Admin
Résumé	Utilisation de la barre de recherche pour filtrer les contrats selon des critères spécifiques.
Scénario principal	1. Accès barre de recherche → 2. Saisie critères → 3. Soumission → 4. Affichage des résultats
Scénario alternatif	Aucun résultat / erreur serveur.

4.3 – Éligibilité IA

Titre	Check AI-based Eligibility
Acteurs	User
Résumé	Soumission d'un formulaire de données financières (revenus, score CIBIL, actifs, montant souhaité, durée) ; évaluation par le modèle IA Flask et affichage du verdict (Approved / Rejected) avec les meilleurs contrats correspondants.
Scénario principal	1. Accès page éligibilité → 2. Remplissage formulaire → 3. isValid() → 4. HTTP POST /apply → 5. Évaluation IA → 6. Affichage résultat + contrats recommandés
Scénario alternatif	Informations insuffisantes / format invalide / erreur serveur.

4.3 – Admin – Gestion des Contrats

Titre	Manage Loan Contracts (Admin)
Acteurs	Admin
Résumé	Création, modification et suppression de contrats de crédit depuis le panneau d'administration.
Scénario principal	1. Accès page administration (profil admin) → 2. Ajout / Édition / Suppression d'un contrat → 3. Confirmation → 4. Mise à jour MongoDB
Scénario alternatif	Informations de contrat invalides / erreur serveur.

5. Conception – Diagrammes UML

Diagramme de Classes Principal

Le système s'articule autour de trois classes principales :

User (classe parente)	Attributs : userID, email, password, googleID, profileID, username, profilePhoto. Méthodes : register(), login(), logout(), updateProfile(), deleteProfile().
NormalUser (hérite de User)	Méthode supplémentaire : viewLoanContracts(). Relation 0..* avec LoanContract (consultation).
AdminUser (hérite de User)	Méthode supplémentaire : manageContracts(). Relation 0..* avec LoanContract (CRUD complet).
LoanContract	Attributs : contractID, amount (Number), interestRate (Number), duration (Number). Méthodes : createContract(), updateContract(), deleteContract().

Diagrammes de Séquence

• Authentification

L'utilisateur remplit le formulaire de connexion. Le frontend appelle isValid(). Si valide : HTTP POST /login → AuthenticationController.LoginCredentialsCheck() → vérification en base. Si trouvé : retour JWT + objet User. Sinon : Error Response Message.

• Test d'Éligibilité IA

L'utilisateur accède à la page 'Test Loan Eligibility'. Remplissage du formulaire (dépendants, éducation, statut emploi, revenus, montant, durée, score CIBIL, actifs). isValid() → HTTP POST /apply → LoanEligibilityController.SubmitLoanApplication() → stockage en base → évaluation par le modèle IA Flask → retour des contrats recommandés → affichage.

• Gestion du Profil Utilisateur

Accès à la page profil → mise à jour des informations → isValid() → HTTP PATCH /updateProfile → UserProfileController.UpdateProfile() → confirmation HTTP 200 → affichage des nouvelles informations.

• Gestion du Profil Admin

L'admin consulte la liste de ses contrats créés (HTTP GET /adminContracts). Suppression : HTTP DELETE /deleteContract → DeleteContract(Request) → confirmation. Modification : HTTP PATCH /editContract → UpdateContract(Request) → confirmation.

• Ajout d'un Contrat (Admin)

L'admin accède à la page de création de contrat → remplit le formulaire (nom, description, montant, taux, durée, exigences, disponibilité, image) → isValid() → HTTP POST /addContract → AddNewLoanContract(Request) → stockage MongoDB → confirmation et affichage sur la page d'accueil pour tous les utilisateurs.

6. Implémentation – Interfaces Principales

Interface d'Authentification (Sign In)

Formulaire centré avec champs email et mot de passe, bouton 'SIGN IN' et option 'CONTINUE WITH GOOGLE'. Lien vers la page d'inscription. Design épuré avec grille de fond (dots pattern) propre à l'identité visuelle Landwise.

Interface d'Inscription (Sign Up)

Formulaire de création de compte (username, email, mot de passe) avec OTP email pour validation. Alternative Google OAuth. Redirection automatique vers le tableau de bord après validation du code.

Page d'Accueil – Catalogue de Crédits

Grille de cartes de contrats de crédit (avec logo banque, titre, description résumée). Barre de recherche en haut, navigation vers les détails de chaque contrat. Accessible sans authentification pour parcourir les offres.

Détail d'un Contrat (Contract Details)

Vue complète : nom du contrat, montant, taux d'intérêt (5.5%), durée (60 mois), mensualité, revenu annuel requis, score CIBIL requis, actifs résidentiels, commerciaux et de luxe requis, actifs bancaires, statut de disponibilité.

Création de Contrat (Admin)

Formulaire admin : Nom du contrat, Description, Montant, Taux d'intérêt (fixe/variable), Mensualité, Revenu annuel requis, Score CIBIL, Actifs (résidentiel, commercial, luxe, bancaire), Image, Disponibilité. Accès restreint à l'admin (PrivateRouteAdmin).

Profil Utilisateur (Normal)

Mise à jour du username, email, mot de passe. Bouton UPDATE, lien 'Delete account', lien 'Sign out'. Bouton 'VIEW YOUR FAVOURITE CONTRACTS' pour accéder aux favoris.

Profil Administrateur

Mêmes options que l'utilisateur + bouton 'CREATE A NEW CONTRACT' (accès formulaire de création) + lien 'VIEW YOUR CREATED CONTRACTS' (liste des contrats créés avec options Modifier / Supprimer pour chacun).

Test d'Éligibilité IA

Formulaire : No of Dependents, Education (Graduate/Not Graduate), Self Employed, Income Per Annum, Loan Amount, Loan Term (Months), CIBIL Score, Residential / Commercial / Luxury / Bank Asset Value. Résultat : 'AI Prediction: Approved' (fond vert) ou 'AI Prediction: Rejected' (fond rouge), avec liste des contrats éligibles.

7. Module d'Intelligence Artificielle

Architecture du Modèle

Le module IA de Landwise, développé intégralement en Python et exposé via Flask, constitue le cœur différenciateur de la plateforme. Il évalue l'éligibilité d'un utilisateur à un crédit à partir de ses données financières et retourne les meilleurs contrats disponibles en base.

Données d'entrée	Nombre de dépendants, niveau d'éducation, statut d'emploi, revenu annuel, montant souhaité, durée (mois), score CIBIL, valeur des actifs résidentiels, commerciaux, de luxe et bancaires.
Entraînement	Dataset CSV de données de prêts réels. Prétraitement, feature engineering, entraînement d'un classificateur (Approved / Rejected).
Exposition API	Flask micro-framework – endpoint HTTP POST /apply. Reçoit les données JSON du backend Express, évalue, retourne le verdict et les IDs de contrats correspondants.
Intégration	Le backend Node.js/Express appelle Flask en interne. Le résultat remonte au frontend React qui affiche le verdict coloré (vert = approuvé, rouge = rejeté) et la liste des contrats recommandés filtrés depuis MongoDB.
Cas de test (approuvé)	2 dépendants, Graduate, Not Self-Employed, 9 600 000 revenu, 29 900 000 montant, 12 mois, CIBIL 779, actifs résidentiels 2 400 000, commerciaux 8 200 000, luxe 29 400 000, bancaires 5 000 000 → Prediction: Approved
Cas de test (rejeté)	5 dépendants, Not Graduate, Self-Employed, 9 600 000 revenu, 9 800 000 montant, 20 mois, CIBIL 382, actifs résidentiels 12 400 000, commerciaux 8 200 000, luxe 29 400 000, bancaires 5 000 000 → Prediction: Rejected

8. Planning & Méthodologie Scrum

Méthodologie Scrum

Le projet a suivi la méthodologie agile Scrum, organisé en sprints de 2 semaines gérés via Jira Software. Chaque sprint comprenait une planification, un développement itératif, une revue et une rétrospective.

Product Owner	Définit les besoins, modélise le produit, priorise les fonctionnalités selon les exigences utilisateurs et les objectifs métier.
Development Team	Implémente les user stories par cycles itératifs, construit et affine l'application selon les spécifications définies.
Scrum Master	Veille au respect de la méthodologie, facilite le processus, améliore la collaboration et lève les obstacles au bon déroulement des sprints.

Gantt – Phases du Projet

Phase	Période	Durée
Requirement Gathering (Recueil des besoins)	Fév. 2023	2 semaines
Design Phase (Conception UML & Architecture)	Fév. – Mars 2023	3 semaines
Development Phase 1 (Backend + Auth)	Mars 2023	3 semaines
Testing Phase 1 (Tests fonctionnels)	Avr. 2023	2 semaines
Development Phase 2 (Frontend + IA)	Avr. – Mai 2023	4 semaines
Testing Phase 2 (Tests intégration)	Mai – Juin 2023	2 semaines
Deployment (Déploiement & documentation)	Juin 2023	1 semaine

9. Limites & Perspectives d'Évolution

Limites Actuelles

- Le modèle IA est entraîné sur un dataset unique (CSV) ; des données plus diversifiées amélioreraient la précision des prédictions.
- Pas de système de notifications en temps réel (email, SMS) pour les mises à jour de statut de crédit.
- L'interface mobile n'a pas fait l'objet d'une optimisation native (PWA non implémentée).
- Pas encore de module de suivi des remboursements avec calendrier et rappels automatiques.
- L'administration des utilisateurs (ban, rôles multiples) n'est pas encore implémentée côté admin.

Pistes d'Évolution

- Intégration d'API bancaires réelles (Open Banking) pour des données en temps réel.
- Amélioration du modèle IA : deep learning, enrichissement des features, meilleure explainability.
- Module de suivi de remboursement avec tableau de bord analytique et alertes automatiques.
- Application mobile React Native pour une expérience cross-platform complète.
- Système de scoring interne permettant à l'utilisateur de suivre l'évolution de son profil financier.
- Internationalisation (i18n) pour cibler les marchés francophones et anglophones.
- Déploiement cloud (AWS / GCP) avec CI/CD via GitHub Actions.

10. Conclusion Générale

Ce projet de fin d'études représente l'aboutissement d'un parcours complet en Génie Logiciel, depuis la conception jusqu'au déploiement d'une solution web Full-Stack intégrant l'intelligence artificielle. Réalisé intégralement par Mohamed Rayen Farhat au sein de l'entreprise VERMEG à Paris, Landwise répond concrètement aux lacunes identifiées dans les plateformes financières existantes.

L'application combine un frontend React moderne (Redux, Tailwind CSS, Google OAuth), un backend robuste (Node.js/Express/MongoDB), et un module IA Python/Flask pour la recommandation personnalisée de crédits. La méthodologie Scrum a permis une gestion agile et itérative du développement, avec des livraisons régulières validées par les parties prenantes.

Au-delà de l'aspect technique, ce projet démontre la capacité à orchestrer l'ensemble du cycle de développement logiciel : analyse des besoins, modélisation UML, architecture système, implémentation, tests et documentation. Il constitue une base solide pour de futures évolutions vers un produit financier à part entière, et marque une étape décisive dans la formation d'ingénieur en Génie Logiciel de Mohamed Rayen Farhat.

Projet	Landwise – Loan Recommendation & Tracking Platform
Réalisé par	Mohamed Rayen Farhat – Licence en Génie Logiciel
Entreprise	VERMEG – Paris (2023)
Stack	MERN (MongoDB, Express, React, Node.js) + Flask/Python IA + Firebase + JWT