KRR Data

# MLOPS

## LOAN APPLICATION DEPLOYMENT

Fait Par
**Kallel - Rigoni - Rodner**

bit.ly/loan-default-prediction

github.com/Medkallel/Loan-Default-Prediction

# Sommaire

KRR Data


01 Notre équipe

02 Introduction

03 Données & Développement du modèle

04 Déploiement sur AWS

05 Défis rencontrés

06 Démonstrations

# Notre équipe

KRR Data

**Kallel Mohamed**

Etudiant DU Data Analytics

**Jean-Christophe Rigoni**

Etudiant DU Data Analytics

**Simon-Pierre Rodner**

MW & RF Ingénieur
Etudiant DU Data Analytics

# Contexte du projet

Ce projet utilise les techniques de **ML Ops**.

Le but est de créer et déployer une application d'aide à la décision d'obtention de crédit.

Nous utilisons un modèle de machine learning capable d'évaluer le risque de défaut de crédit, en se basant sur les données de clients précédent.

- **Source:**
  - Loan_Data.csv
- **Modèles:**
  - Regression logistique de Scikit-Learn
  - Random forest de Scikit-Learn
- **Technologies:**
  - Python
  - Git et Github
  - Mlflow
  - Flask
  - AWS ECS et ECR

KRR Data

# Analyse des Données

**Loan_Data.csv**

**Dimensions**

10K lignes

8 colonnes

**Type de données**

Int 64 / float 64

**Valeurs manquantes**

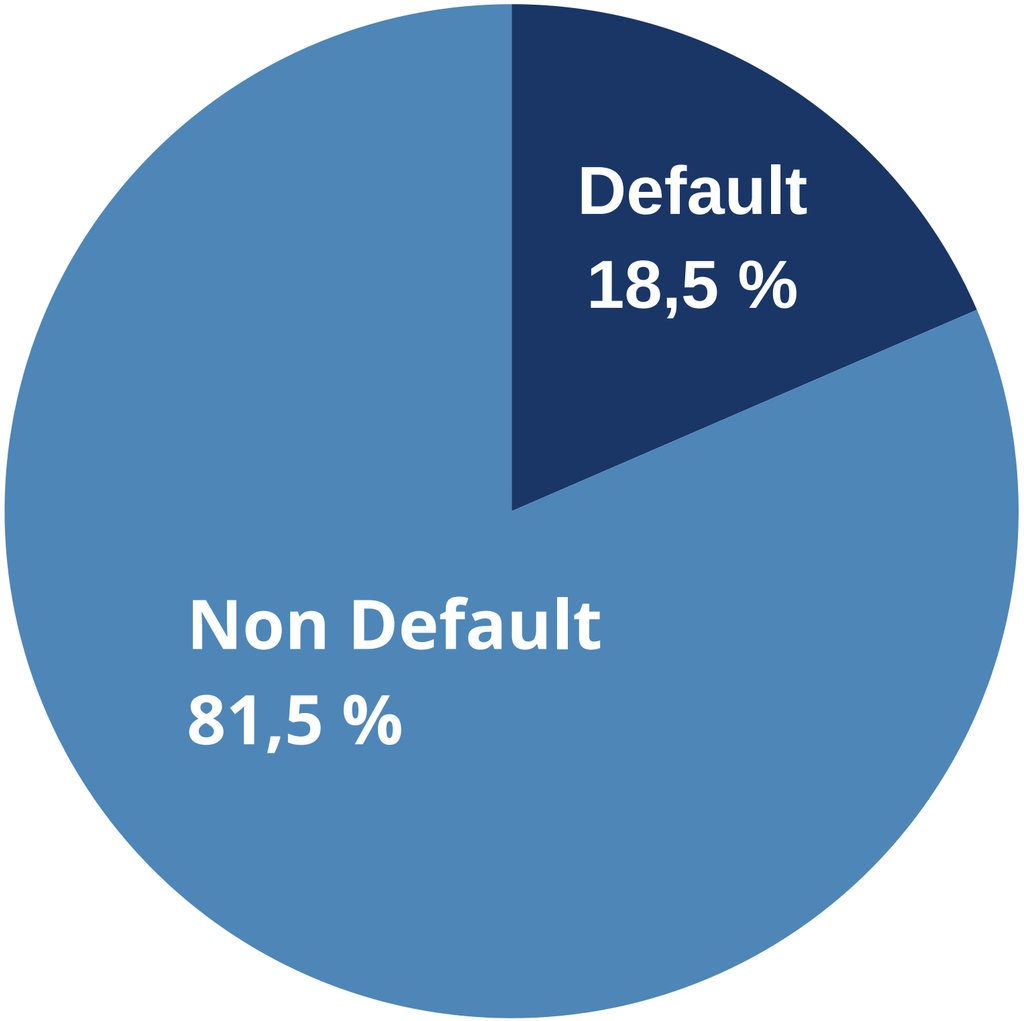
0

KRR Data

# Analyse & modèle

**Problème**
Dataset Déséquilibré

**Solution**
Utilisation de poids des classes équilibrés




KRR Data

# Analyse & Modèle

## Problème

**Forte correlation des variables explicatives**

## Solution

**Modèle RandomForest**

## Matrice De Corrélations

| | customer_id | credit_lines_outstanding | loan_amt_outstanding | total_debt_outstanding | income | years_employed | fico_score | default |
|---|---|---|---|---|---|---|---|---|
| customer_id | 1 | 0.01 | −0.01 | 0 | −0.01 | −0.01 | 0.01 | 0.01 |
| credit_lines_outstanding | 0.01 | 1 | 0.08 | 0.85 | 0.02 | −0.09 | −0.26 | 0.86 |
| loan_amt_outstanding | −0.01 | 0.08 | 1 | 0.4 | 0.84 | −0.16 | −0.03 | 0.1 |
| total_debt_outstanding | 0 | 0.85 | 0.4 | 1 | 0.39 | −0.17 | −0.23 | 0.76 |
| income | −0.01 | 0.02 | 0.84 | 0.39 | 1 | 0 | −0.01 | 0.02 |
| years_employed | −0.01 | −0.09 | −0.16 | −0.17 | 0 | 1 | 0.26 | −0.28 |
| fico_score | 0.01 | −0.26 | −0.03 | −0.23 | −0.01 | 0.26 | 1 | −0.32 |
| default | 0.01 | 0.86 | 0.1 | 0.76 | 0.02 | −0.28 | −0.32 | 1 |

KRR Data

# Analyse & Modèle

KRR Data

**Comparaison de Regression logistique & Ramdom Forest avec *MlFlow***

## Modèles testés

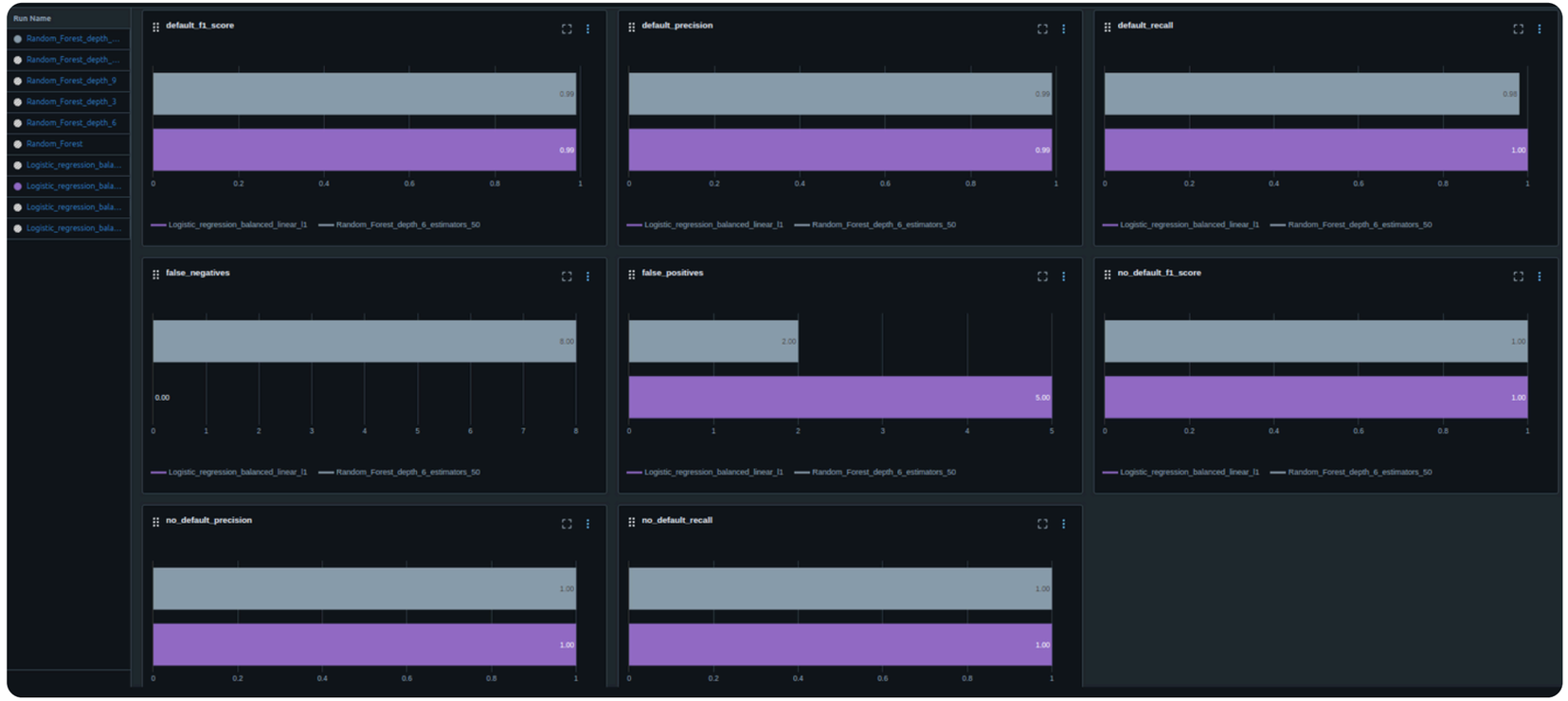
- Régression logistique
- Random Forest

## Métriques

- F1-score - Precision - Recall
- FN - FP

## Choix du modèle

**Régression logistique**



# Versioning - Git

## Exemple Historique Commits

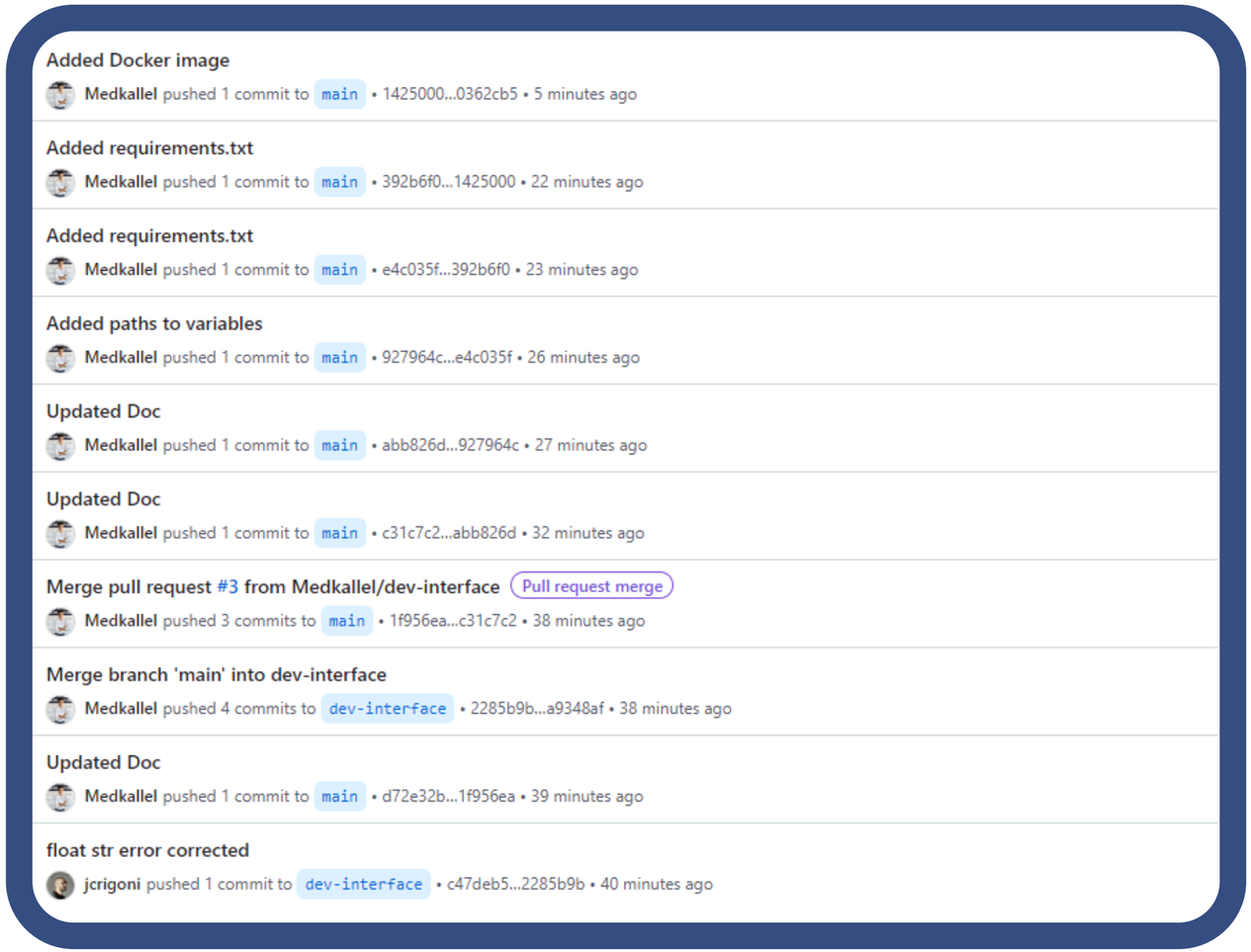
**Git** : Système de versionning.

**GitHub** : Plateforme collaborative en ligne.

**Repository:** Medkallel/Loan-Default-Prediction

**Contributors:**

Mohamed Kallel
Jean-Christophe Rigoni
Simon-Pierre Rodner




KRR Data

# Chronologie de deploiement d’application avec les solutions cloud d’AWS

**Phase 2**
**Stockage Images**
Upload des images Dockers dans un repository.

**Phase 4**
**Configuration de la tâche**
- Rôle
- Conteneur
- Port

IAM → ECR → Cluster → Task → Service

**Phase 1**
**Clé d’accès**
Permet l’authentification de utilisateur pour l’utilisation des **ressources AWS.**

**Phase 3**
**Groupe de ressource**
Structure regroupant les ressources. Exécution des tâches et gestion des services.

KRR Data

# Chronologie de deploiement d’application avec les solutions cloud d’AWS

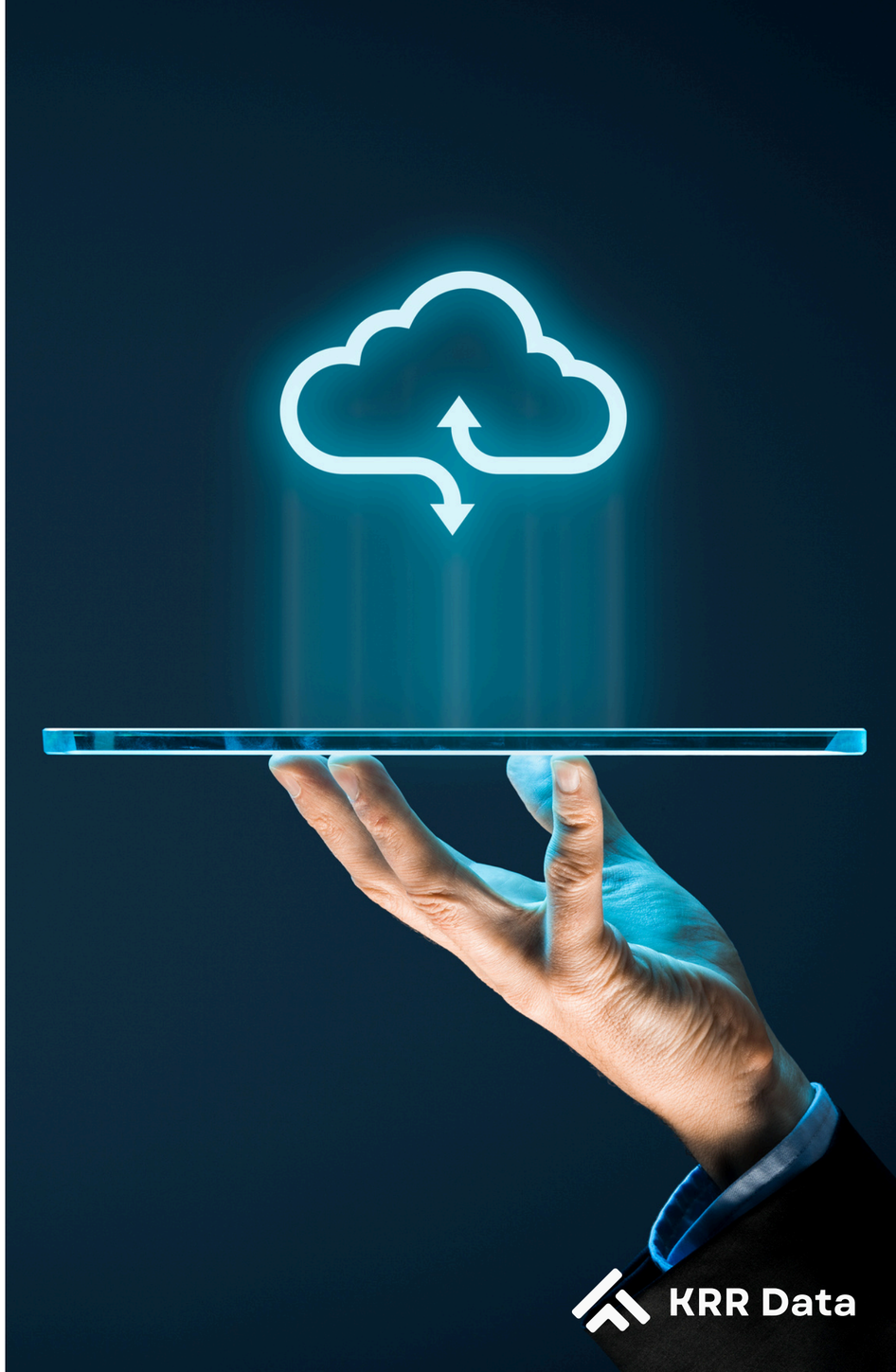
**Service**

**Phase 5**

**Déploiement et gestion des tâches**

**Rôle :** Gère le déploiement et le contrôle de la bonne exécution des taches en cours.

**Utilisation :** Assure que le nombre souhaité de tâche est en cours d’exécution et gère les mises à jour et le placement des taches dans le cluster



# CI/CD

KRR Data

**Déploiement CI/CD automatisé**

**ci_pipeline**
succeeded 11 hours ago in 53s

- Set up job
- Run actions/checkout@v1
- Set up Python 3.11
- Install dependencies
- Format
- Complete job

**Construction d'image docker**

**cd_pipeline**
succeeded 11 hours ago in 2m 29s

- Set up job
- Run actions/checkout@v2
- Get current date
- Build the Docker image
- Docker Hub login
- Push Docker image to Docker Hub
- GitHub Container Registry login
- Push Docker image to GitHub Container Registry
- Post Run actions/checkout@v2
- Complete job

**Push vers Docker Hub & GitHub**

**Déploiement sur Aws**

**Deploy**
succeeded 4 hours ago in 5m 53s

- Set up job
- Checkout
- Configure AWS credentials
- Login to Amazon ECR
- Build, tag, and push image to Amazon ECR
- Download task definition
- Fill in the new image ID in the Amazon ECS task definition
- Deploy Amazon ECS task definition
- Post Login to Amazon ECR
- Post Configure AWS credentials
- Post Checkout
- Complete job

# Défis Rencontrés

La mise en pratique d'un process Mlops comme tout projet comporte quelques difficultés.

## Organisation

Mettre en place les process de Mlops dans un cadre collaboratifs, nous a poussé vers un changement de méthodologie.

**Point positif: efficacité augmentée**

## Optimisation des modèles

Comme tout projet de Data, l'optimisation des modèles est une tâche délicate.

**Point positif : Dataset optimisé pour le modèle**

## Gestion des dépendances

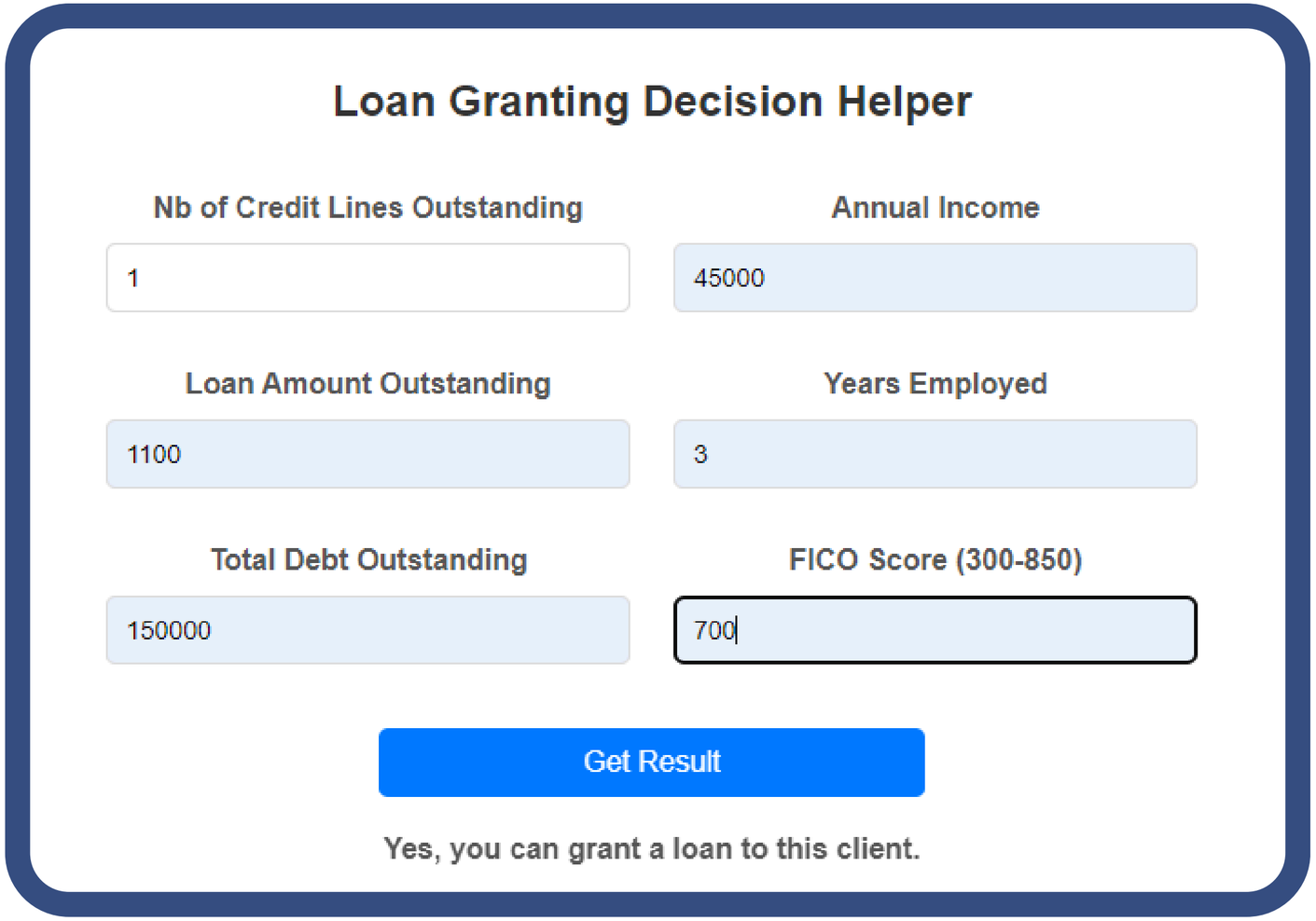
Avoir des requirements adaptés au besoin de l'application dans le conteneur.

**Point positif : Meilleur compréhension des environements virtuels et des conteneurs.**

KRR Data

# Demonstration

Aperçu de l'application **Flask** deployée



KRR Data

# THANK YOU

Fait Par
**Kallel - Rigoni - Rodner**

bit.ly/loan-default-prediction

github.com/Medkallel/Loan-Default-Prediction