Rapport – Application de Prédiction du Diabète pour Médecins

# 1. Introduction

Le diabète est une maladie chronique qui nécessite une détection précoce pour éviter des complications graves. L’objectif de ce projet est de fournir aux médecins un outil simple et efficace pour :  
- Enregistrer les informations des patients.  
- Prédire le risque de diabète à partir de données médicales.  
- Conserver les résultats dans une base de données.

# 2. Objectifs du projet

- Automatiser la prédiction du diabète grâce à un modèle Machine Learning.  
- Faciliter la gestion des patients (ajout, consultation, historique).  
- Améliorer la prise de décision médicale avec un outil d’aide simple.  
- Fournir une interface intuitive pour les médecins.

# 3. Architecture du système

L’application est composée de trois grandes parties :  
1. Frontend : Interface utilisateur réalisée avec HTML, CSS et Jinja2.  
2. Backend : API et logique métier développées en Python avec Flask.  
3. Machine Learning : Modèle de classification entraîné avec Scikit-learn.  
  
Schéma simplifié :  
Médecin → Interface Web → Flask Backend → Modèle ML → Résultat → Base de données

# 4. Fonctionnalités principales

- Authentification des médecins (inscription, connexion sécurisée).  
- Ajout de patients avec formulaire détaillé (âge, glucose, IMC…).  
- Prédiction du diabète via le bouton d’analyse.  
- Affichage du diagnostic (diabétique / non diabétique).  
- Enregistrement automatique des résultats dans PostgreSQL.  
- Consultation de la liste des patients avec leurs prédictions.

# 5. Technologies utilisées

|  |  |
| --- | --- |
| Domaine | Technologie |
| * Backend | Flask (Python) |
| * Base de données | PostgreSQL, SQLAlchemy |
| * Machine Learning | Scikit-learn, Joblib |
| * Frontend | HTML, CSS, Jinja2 |

# 6. Modèle de Machine Learning

• Données utilisées : Jeu de données Pima Indians Diabetes Dataset.  
• Prétraitement : StandardScaler pour normaliser les données.  
• Algorithme choisi : Random Forest Classifier (bonne précision et robustesse).  
• Export du modèle : Sauvegarde avec Joblib (model.pkl et scaler.pkl).  
  
Variables d’entrée pour la prédiction :  
1. Pregnancies  
2. Glucose  
3. Blood Pressure  
4. Skin Thickness  
5. Insulin  
6. BMI  
7. Diabetes Pedigree Function  
8. Age

# 7. Exemples de prédiction

Patient diabétique :  
Pregnancies: 4  
Glucose: 165  
Blood Pressure: 80  
Skin Thickness: 35  
Insulin: 200  
BMI: 33.5  
Diabetes Pedigree Function: 0.75  
Age: 45  
→ Résultat : Diabétique  
  
Patient non diabétique :  
Pregnancies: 1  
Glucose: 90  
Blood Pressure: 70  
Skin Thickness: 20  
Insulin: 80  
BMI: 22.0  
Diabetes Pedigree Function: 0.3  
Age: 28  
→ Résultat : Non diabétique

# 8. Difficultés rencontrées

- Mauvaise interprétation des résultats du modèle lors des premiers tests.  
- Gestion des dépendances Python et compatibilité entre versions.  
- Organisation des fichiers et des routes Flask.

# 9. Validation et tests

- Tests unitaires sur la fonction predict\_diabetes.  
- Vérification manuelle avec des données connues.  
- Comparaison avec les résultats attendus sur un échantillon.  
- Validation auprès d’un utilisateur cible (médecin).

# 10. Conclusion

Ce projet offre un outil fiable pour aider les médecins à prédire le diabète et à gérer leurs patients efficacement. Des améliorations possibles incluent :  
- Ajout d’un graphique d’historique des patients.  
- Intégration d’un système multi-langues.  
- Déploiement sur un serveur cloud pour un accès multi-utilisateurs.