

|  |
| --- |
| Rapport de Projet : Outil d'Estimation des Prix d’Appartements au Maroc |
|  |
| Créé par : el kadmiri younes |

Contexte du projet :

Dans un marché immobilier en constante évolution, estimer le juste prix d’un bien est un enjeu crucial. Ce projet vise à concevoir un simulateur intelligent d’évaluation des prix d’appartements basé sur le machine learning.

Développé dans le cadre d’une mission fictive pour l’entreprise SalesHouses, l’objectif est de fournir un outil fiable et rapide pour prédire les prix à partir de caractéristiques comme la superficie, la localisation, le nombre de chambres, etc.

Objectifs :

* Construire un modèle de régression supervisée capable de prédire le prix d’un appartement.
* Nettoyer, transformer et analyser les données immobilières.
* Évaluer et comparer plusieurs modèles de machine learning.
* Sauvegarder le meilleur modèle et le rendre utilisable via une interface ou API.
* Assurer la reproductibilité, la documentation et le déploiement.

Architecture du projet :

bash

CopierModifier

appartement-pricing-ml/

├── app.py # App principale (exécution globale)

├── main.py # Entrée du pipeline ML (POO)

├── predict\_single.py # Script de prédiction unitaire

├── data/ # Contient le fichier CSV source

│ └── appartements\_data\_db.csv

├── models/ # Contient tous les modèles entraînés

│ └── best\_model\_X.pkl

├── notebooks/ # Notebooks d'analyse exploratoire (EDA)

├── src/ # Code source du pipeline

│ ├── preprocessing.py # Classe Preprocessor (nettoyage, scaling, etc.)

│ ├── modeling.py # Classe ModelFactory, ModelTrainer

│ ├── evaluation.py # Classe Evaluator (MSE, MAE, RMSE, R²)

│ ├── database.py # Fonction de sauvegarde PostgreSQL

│ └── utils.py # Outils annexes

├── conception/ # Diagrammes UML

│ ├── Diagramme de cas d’utilisation

│ ├── Diagramme de séquence

│ ├── Diagramme de classes

│ ├── Diagramme de déploiement

│ └── Diagramme d’activité

├── Dockerfile # Dockerisation de l’app

├── requirements.txt # Dépendances Python

└── rapport.docx # Rapport actuel

Méthodologie :

1. Analyse Exploratoire (EDA)

Via plusieurs notebooks :

* Aperçu du dataset (df.info(), df.describe())
* Visualisation de la distribution des prix et surfaces
* Corrélation avec les autres variables
* Détection des outliers (IQR, boxplots)

2. Prétraitement des données

* Transformation de la colonne equipment en colonnes booléennes
* Nettoyage et uniformisation de city\_name
* Imputation des valeurs manquantes
* Suppression des outliers
* Encodage et mise à l’échelle

3. Sélection des variables

* Choix des colonnes corrélées au prix
* Élimination des variables trop corrélées entre elles (redondance)

4. Entraînement des modèles

Modèles testés :

* Régression linéaire
* Random Forest
* Gradient Boosting
* ExtraTrees
* SVR

5. Évaluation des modèles

Métriques utilisées :

* MAE : Erreur absolue moyenne
* MSE : Erreur quadratique moyenne
* RMSE : Racine de MSE
* R² : Score de détermination

6. Optimisation

* Recherche d’hyperparamètres via GridSearchCV
* Sélection du meilleur modèle (score R² élevé, faible RMSE)

7. Persistences

* Sauvegarde du meilleur modèle (joblib)
* Insertion des données nettoyées et prédictions dans PostgreSQL

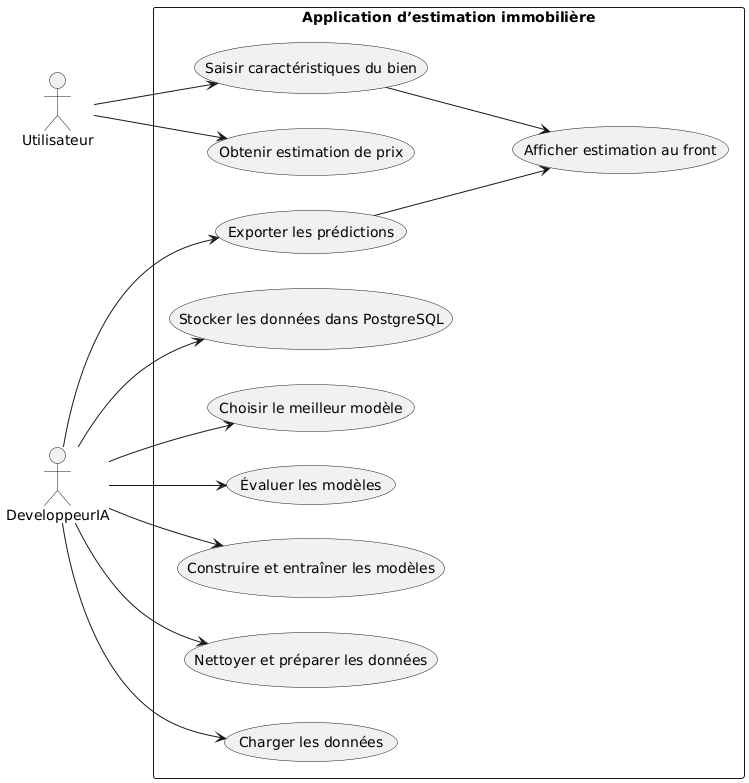
Déploiement & CI/CD :

* Un Dockerfile permet de déployer l’application dans un conteneur.
* Une pipeline GitHub Actions (main.yml) automatise les tests et la génération de modèles.
* L’application peut être étendue avec une API (ex. FastAPI) pour la mise en production.

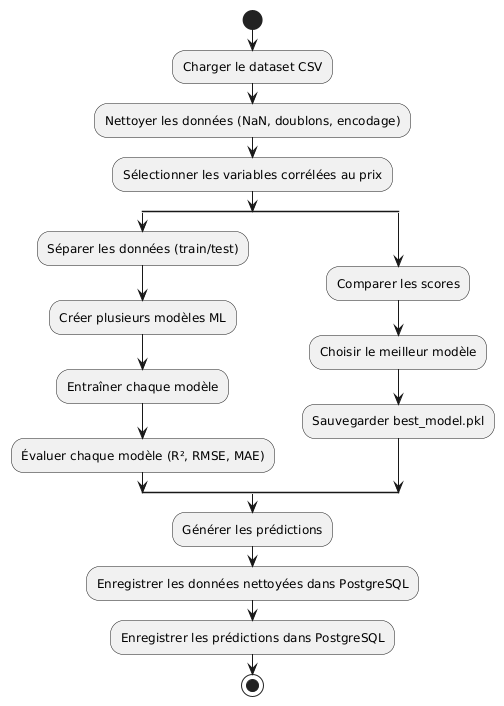
Diagrammes :

Les diagrammes UML sont dans le dossier conception :

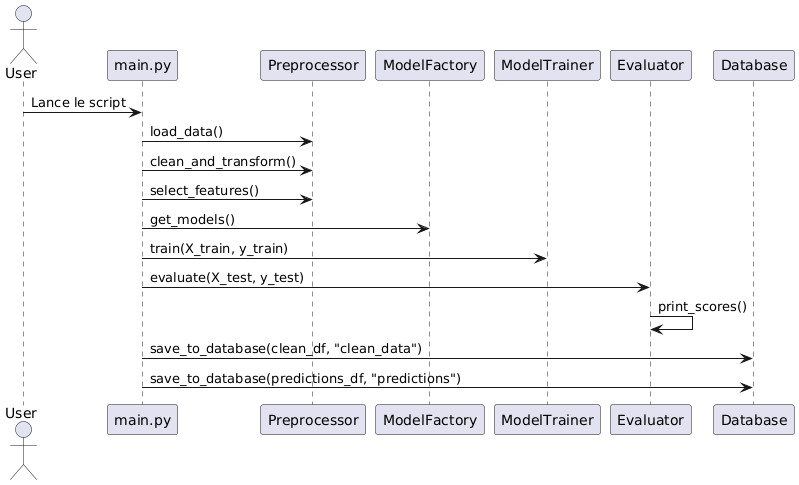
* Diagramme de cas d’utilisation : utilisateur → simulation prix



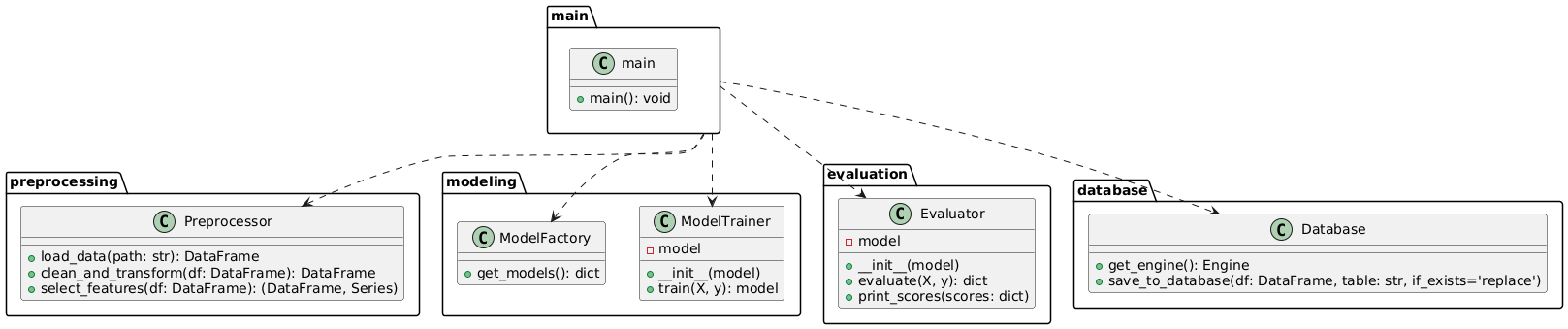
* Diagramme d’activités : pipeline de nettoyage → modélisation → prédiction



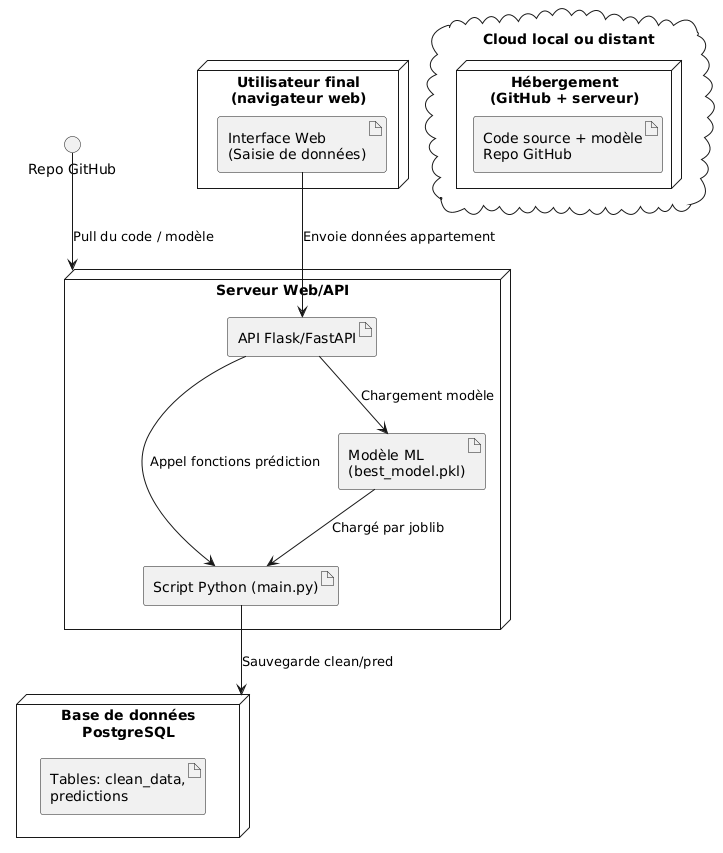
* Diagramme de séquence : interaction entre modules



* Diagramme de classes : structure POO



* Diagramme de déploiement : Docker, PostgreSQL, code Python



Planning (Gantt & Kanban) :

* Diagramme de Gantt : réalisation en 5 jours (14 au 18 juillet)
* Organisation Kanban (Jira) : tâches comme EDA, Feature Engineering, Entraînement, etc.

Conclusion :

Ce projet met en œuvre un cycle complet de data science en entreprise : de l’EDA à la mise en production. Le modèle est entraîné, évalué, optimisé, documenté et prêt à être intégré à une interface utilisateur.