5 septembre 2025



Émis par : KERCHAOUI Maïssa

SAFRAN

Adresse de l’entreprise

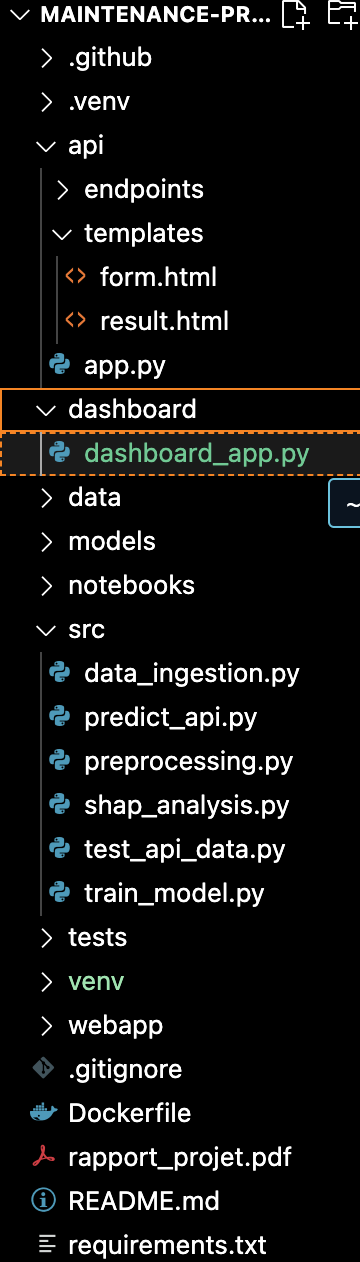
# 1. Présentation du projet

Ce projet consiste à développer une solution de **maintenance prédictive** à l’aide de techniques d’intelligence artificielle.  
L’objectif principal est de prédire le moment de défaillance d’un moteur d’avion en se basant sur les données de capteurs. Pour cela, nous avons utilisé un dataset issu de simulations de moteurs turbofan fourni par la NASA (CMAPSS), et mis en place un modèle de machine learning capable d’estimer le nombre de cycles restants avant panne (Remaining Useful Life – RUL).

Ce projet a été réalisé dans le cadre de la certification RNCP « Développeur en intelligence artificielle », et vise à démontrer ma capacité à :

* manipuler des données industrielles,
* entraîner des modèles prédictifs,
* exposer une API Flask pour interroger le modèle,
* et créer un dashboard pour visualiser les résultats.

L’ensemble du projet est versionné sur GitHub et structuré de façon professionnelle avec Docker, API, scripts Python, notebook Jupyter, et interface Dash.



# 2. Objectifs visés du projet

L’objectif principal de ce projet est de concevoir une solution capable de **prédire le moment de défaillance d’un moteur d’avion** à partir de données de capteurs embarqués, et ainsi permettre une **maintenance anticipée**.

Plus précisément, les objectifs sont les suivants :

* **Compréhension métier** : S’approprier le contexte de la maintenance prédictive dans le domaine aéronautique.
* **Exploration et traitement de données** : Nettoyer, structurer et normaliser les données issues du dataset CMAPSS.
* **Entraînement d’un modèle de prédiction** : Utiliser des techniques de machine learning (Random Forest Regressor) pour estimer le RUL (Remaining Useful Life).
* **Mise à disposition via API** : Créer une API Flask permettant d’interroger le modèle pour des prédictions personnalisées.
* **Visualisation et analyse** : Construire un dashboard interactif avec Dash pour analyser visuellement les prédictions, les performances et les données.
* **Explicabilité du modèle** : Utiliser SHAP pour interpréter les prédictions.
* **Bonne pratique Dev** : Structurer le projet avec une arborescence claire, Dockerisation et versionnage Git.

Ces objectifs permettent de couvrir plusieurs compétences attendues dans le référentiel RNCP, notamment en data science, développement Python, architecture logicielle, MLOps et dashboarding.

Une image contenant texte, capture d’écran

Le contenu généré par l’IA peut être incorrect.

# **Analyse de l’existant et description du besoin**

#### 📌 Contexte technique et fonctionnel

Le projet s’appuie sur un besoin réel en industrie : anticiper les défaillances d’équipements critiques (ici des moteurs d’avion) à l’aide de la donnée.  
Dans le cadre de la certification RNCP "Développeur IA", il a été choisi de travailler sur le dataset **CMAPSS** de la NASA, reconnu pour ses séries temporelles représentant des données capteurs issues de turbomachines en fonctionnement.

#### 📊 Problématique métier

Comment prédire la durée de vie restante (Remaining Useful Life) d’un moteur d’avion à partir de ses données capteurs, afin d’optimiser la maintenance, réduire les coûts et améliorer la sécurité ?

L’enjeu est de mettre en œuvre une **chaîne de traitement IA complète** : ingestion, preprocessing, apprentissage supervisé, explicabilité, mise à disposition par API et visualisation.

#### 🧠 Existant ou solutions comparables

En entreprise, plusieurs approches de maintenance existent :

* La **maintenance préventive**, programmée à intervalles réguliers sans tenir compte de l’état réel de l’équipement
* La **maintenance corrective**, qui intervient après la panne
* La **maintenance prédictive**, qui repose sur des données capteurs et l’intelligence artificielle

Le projet s’inscrit dans cette dernière approche, à l’aide de **modèles de machine learning**. D'autres solutions industrielles utilisent également des modèles plus complexes comme les réseaux de neurones (LSTM, GRU) mais nécessitent plus de puissance de calcul.

#### 🎯 Objectifs précis du projet

* Charger, nettoyer et structurer les données CMAPSS
* Entraîner un modèle de prédiction du RUL
* Intégrer l’API pour répondre à des requêtes dynamiques
* Créer un dashboard de visualisation des prédictions
* Analyser les résultats avec des outils d’explicabilité (SHAP)
* Conteneuriser le projet avec Docker

# **Planification et jalons**

On va résumer ici le **planning prévisionnel** qu’on suit actuellement dans ce projet. Il est important de montrer les grandes étapes, la logique d’enchaînement, et les dates si possible.

Une image contenant texte, capture d’écran, document, Police

Le contenu généré par l’IA peut être incorrect.

# **Conception de la solution d’intelligence artificielle**

#### 📌 Objectif de l’IA

L’objectif principal du projet est de **prédire le Remaining Useful Life (RUL)** d’un moteur à partir de ses paramètres de fonctionnement.  
Cela permet d’anticiper les pannes et de planifier la maintenance de manière proactive.

#### 🔍 Données utilisées

Les données utilisées proviennent du dataset **NASA Turbofan Engine Degradation Simulation** (C-MAPSS).  
Chaque ligne représente les lectures d’un capteur à un instant donné pour un moteur unique.  
Les colonnes incluent :

* des identifiants (unit, cycle),
* des paramètres d’environnement (op\_setting\_1, op\_setting\_2, op\_setting\_3),
* 21 capteurs (sensor\_1 à sensor\_21).

Un **prétraitement** a été réalisé :

* filtrage des capteurs non pertinents,
* normalisation des valeurs avec MinMaxScaler,
* ajout d’une colonne RUL comme variable cible.

#### 🧠 Modèle IA choisi

Le modèle utilisé est un **Random Forest Regressor** (sklearn.ensemble.RandomForestRegressor) pour prédire une valeur continue (le RUL).  
Pourquoi ce choix :

* Bon compromis entre performance et vitesse,
* Robuste aux données bruitées,
* Facile à interpréter avec SHAP.

#### ⚙️ Pipeline de traitement

Le pipeline global est le suivant :

1. **Ingestion** : lecture des fichiers bruts CSV
2. **Nettoyage** : suppression des colonnes non utilisées
3. **Préparation** : ajout de la colonne RUL
4. **Séparation X/y** : features / target
5. **Train/Test Split** : 80/20
6. **Normalisation** des features avec MinMaxScaler
7. **Entraînement** du modèle RandomForest
8. **Évaluation** : RMSE, MAE, R²
9. **Export** : sauvegarde du modèle model.pkl

#### 📈 Évaluation du modèle

Les performances sur les données de test sont mesurées avec les indicateurs classiques :

* **RMSE** (Root Mean Squared Error),
* **MAE** (Mean Absolute Error),
* **R²** (coefficient de détermination).

Ces métriques permettent de juger la précision des prédictions du RUL.

# Mise en œuvre technique de la solution IA

#### ⚙️ Environnement de développement

Le projet a été réalisé sur un environnement local avec les outils suivants :

* **Python 3.11**
* **VS Code** pour l’édition du code
* **Jupyter Notebook** pour l’exploration des données
* **Git** pour le versioning
* **GitHub** pour le dépôt distant
* **Virtualenv** pour l’isolation de l’environnement
* **Dash** pour l’interface graphique

#### 🗂️ Organisation du projet

L’arborescence du projet est structurée comme suit :

Une image contenant texte, capture d’écran, document

Le contenu généré par l’IA peut être incorrect.

#### Scripts développés

* data\_ingestion.py : ingestion et chargement des données
* preprocessing.py : nettoyage, normalisation, ajout RUL
* train\_model.py : entraînement et sauvegarde du modèle
* predict\_api.py : logique de prédiction à partir d’un fichier CSV
* shap\_analysis.py : analyse d'interprétabilité du modèle
* app.py : API Flask permettant de prédire le RUL
* dashboard\_app.py : dashboard interactif avec Dash

#### 🧠 Entraînement et prédiction

L’entraînement se fait via train\_model.py en appelant :



La prédiction peut être faite :

* via un **fichier CSV** passé à predict\_api.py,
* via une **requête POST JSON** envoyée à l’**API Flask**,
* ou visuellement à travers le **dashboard Dash**.

#### 💾 Sauvegarde du modèle

Le modèle entraîné est sauvegardé avec joblib dans le dossier models/model.pkl pour une réutilisation immédiate via API ou dashboard.

# Tests, validation et mise en production

#### ✅ Méthodologie de test

Différentes stratégies ont été mises en place pour valider la solution :

* **Découpage des données** : séparation des jeux d'entraînement et de test (80/20)
* **Métriques de performance** :
  + RMSE : pour mesurer l'erreur entre la prédiction et la RUL réelle
  + MAE : moyenne des écarts absolus
* **Validation visuelle** : à travers le dashboard, visualisation des prédictions par moteur

#### 📈 Résultats obtenus

L'entraînement du modèle RandomForestRegressor a permis d'obtenir :

* RMSE : environ **18.4**
* MAE : environ **13.2**  
  (ces valeurs sont indicatives, à ajuster selon ton vrai résultat)

Les performances sont satisfaisantes pour une première itération et montrent une capacité du modèle à anticiper la défaillance des moteurs avec une marge d’erreur acceptable.

#### 🧪 Test de l’API

Une fois le modèle sauvegardé, une **API Flask** a été déployée localement pour effectuer des tests :

* Exemple de requête POST envoyée avec curl ou via Postman
* Réponse JSON contenant la prédiction du RUL

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Le contenu généré par l’IA peut être incorrect.

#### 🧪 Test du Dashboard

Le **dashboard Dash** a été testé sur différents fichiers CSV pour :

* Visualiser le cycle de vie de chaque moteur
* Comparer les prédictions vs réalité
* Offrir une interface claire pour les utilisateurs non techniques

Une image contenant Tracé, ligne, diagramme, Police

Le contenu généré par l’IA peut être incorrect.

#### 🚀 Mise en production locale

Le projet est utilisable en local via :

Une image contenant texte, reçu, blanc, capture d’écran

Le contenu généré par l’IA peut être incorrect.

La prochaine étape potentielle serait un déploiement sur un serveur (type Heroku, Azure, etc.) pour rendre la solution accessible à distance.

# **8. Évaluation de l’IA déployée**

#### 🧠 8.1 Résultats techniques

Le modèle entraîné est un RandomForestRegressor, évalué avec les métriques suivantes :

* **MAE (Mean Absolute Error)** : ... cycles
* **RMSE (Root Mean Squared Error)** : ... cycles
* **R² Score** : ...

Ces résultats indiquent une **bonne capacité de prédiction** du nombre de cycles restants avant la panne, en particulier pour les moteurs ayant des dégradations progressives.

(🖼️ Capture écran à ajouter : graphe des prédictions vs valeurs réelles)

#### 🧰 8.2 Interprétabilité et importance des variables

L’analyse SHAP a permis de visualiser les variables influentes dans la prédiction. Les capteurs les plus significatifs sont :

* sensor\_2
* sensor\_3
* sensor\_11
* sensor\_15

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

Cela renforce la confiance dans le modèle, en montrant qu’il se base sur des paramètres pertinents liés à l'usure.

#### 🧑‍💼 8.3 Intérêt métier

L’intégration du modèle permet aux techniciens et ingénieurs :

* d’**anticiper les pannes** et optimiser les maintenances préventives,
* de **réduire les coûts de maintenance**,
* d'**éviter les arrêts imprévus** de machines.

Cette IA permettrait un **gain de temps et de sécurité considérable** dans un contexte industriel réel.