IUT PARIS CITE
BUT VCOD
2025-2026

Projet d'évaluation : Ingénierie logicielle appliquée à un Projet Data

L'objectif de ce projet est d'appliquer les bonnes pratiques d'ingénierie logicielle vues pendant ce cours à un projet Data existant fourni sous forme de notebook python.

⇒Vous devez travailler en équipe (min 2, max 4) et utiliser l'outil collaboratif GIT ainsi qu'un répertoire projet sur github
⇒Pour la partie CI/CD, il est recommandé d'utiliser Github Actions comme votre projet sera déjà sur Github
⇒Vous êtes libres d'utiliser l'IDE python de votre choix
⇒Deadline du projet : Dimanche 8 Février 2026
⇒Restitution : URL de votre projet github à partager avec dinamedy@hotmail.com + un rapport détaillé (pdf ou doc) à déposer dans un drive dédié à votre groupe qui vous sera partagé.

Description et objectif du projet

Vous allez travailler sur le projet existant **Titanic Survival Prediction** disponible en accès libre sur Kaggle. Ce projet consiste à traiter les données, effectuer une analyse exploratoire (EDA), et construire un modèle d'apprentissage machine (ML) pour prédire les chances de survie.

Votre mission :

Le **code sous format notebook est déjà disponible sur kaggle** pour ce projet, votre mission sera de réécrire ce projet en respectant les bonnes pratiques d'ingénierie logicielle vues en cours et TPs. Plus précisément, vous devez :

1. Refactoriser le code du notebook en scripts Python modulaires et réutilisables.
2. Appliquer les meilleures pratiques d'ingénierie logicielle, comme la conformité à **PEP 8**, les tests unitaires et la documentation.
3. Utiliser **Git et GitHub** pour la collaboration en équipe.
4. Mettre en place et simuler un **pipeline CI/CD** pour automatiser les tests, le linting et le déploiement (via **Github Action**).

Ces étapes sont détaillées ci-dessous :

Projet étape par étape

1. Refactoriser le Code

⇒ Télécharger les données

Téléchargez les Données Titanic depuis Kaggle

⇒ Notebook de départ

Plusieurs versions de notebook existent sur Kaggle pour ce projet, pour ce projet vous allez partir d'un notebook simple sous forme de tutoriel : Titanic Tutorial

⇒Passer du Notebook python aux scripts python ".py":

- Diviser le notebook monolithique en plusieurs scripts/modules Python au format ".py" tels que :

    - `data_preprocessing.py` pour le nettoyage des données et l'ingénierie des caractéristiques.
    - `model_training.py` pour l'entraînement et la sauvegarde des modèles.
    - `model_evaluation.py` pour l'évaluation des performances du modèle.
- Assurez-vous que les scripts fonctionnent de bout en bout avec des arguments ou des fichiers de configuration appropriés.
- Vous pouvez utiliser le template de cookie cutter vu en cours ou un autre de votre choix

⇒Qualité de code :

- Appliquer les règles de style **PEP 8** à vos scripts python.
- Utiliser des noms de fonctions et de variables explicites.
- Utiliser une solution de gestion des librairies pour la reproductibilité de votre code comme poetry, conda ou pyenv

2. Ajouter des tests unitaires

- Écrire des tests unitaires pour les composants clés, comme :
    - Les fonctions de prétraitement des données.
    - La logique d'entraînement et d'évaluation des modèles.
- Il est préférable d'utiliser `pytest` pour les tests (plus simple)

3. Ajouter de la documentation

- Ajouter des **docstrings** dans les scripts pour décrire l'objectif des fonctions et modules.
- Créer un fichier `README.md` expliquant :
    - Les objectifs du projet.
    - Les instructions d'installation et d'utilisation.
    - Les contributions des membres de l'équipe.

4. Utiliser Git et GitHub pour la collaboration en équipe

- Chaque équipe utilisera **GIT** en local et un répertoire GitHub partagé.
- Suivre une stratégie de branches (par exemple, `main`, `develop`, et branches de fonctionnalités).
- Utiliser des **pull-requests** pour demander la revue et le merge de code.
- Utiliser des messages de commit explicites pour faciliter la compréhension et la collaboration

5. Mettre en Place une Pipeline CI/CD

- Utiliser Github Actions pour automatiser :
    - Le linting avec `flake8` ou `pylint`.
    - Le formatage avec `black`.
    - L'exécution des tests unitaires avec `pytest`.
- **Optionnel (points supplémentaires)** : Simuler le déploiement en créant un conteneur Docker pour le projet et en automatisant sa construction et son push sur Docker Hub.

6. Livrables

- Le **répertoire GitHub** (partagé avec dinamedy@hotmail.com) de votre équipe avec au moins :
    - Un dossier `src` contenant des scripts Python modulaires.
    - Un dossier `tests` contenant des tests unitaires.
    - Un dossier `docs` contenant la documentation.
    - Les librairies nécessaires au projet (requirement.txt)
    - Des commits bien organisés.
    - Un pipeline CI/CD.
    - **Optionnel (points supplémentaires)** : Un `Dockerfile` pour la containerisation.
- Un **rapport détaillé** expliquant votre démarche (pdf ou doc) à déposer dans un drive dédié à votre groupe qui vous sera partagé.