1. **.github/workflows** :
   * Ce répertoire contient les workflows GitHub Actions, utilisés pour l'intégration et le déploiement continus (CI/CD). Ces workflows permettent d'automatiser des tâches comme les tests, la génération d'artefacts, ou le déploiement sur des environnements comme Heroku.
   * Tu mentionnes une mise à jour récente des artefacts, ce qui indique que ton pipeline CI/CD génère probablement des artefacts à chaque commit pour les tests ou le déploiement.
2. **app** :
   * Ce répertoire contient probablement ton application principale, c’est-à-dire le code qui exécute les fonctionnalités du projet. Il peut s’agir de scripts Python, de fichiers d’API, ou d’autres composants du projet.
   * Si ton projet est lié à un modèle ML ou une application web, c’est ici que tu trouverais les principaux fichiers exécutables.
3. **artefact** :
   * Ce répertoire est destiné au stockage des artefacts générés pendant l'exécution du pipeline ou du projet. Les artefacts peuvent inclure des modèles ML entraînés, des fichiers de données ou d’autres fichiers résultant des étapes intermédiaires du projet.
   * Les mises à jour récentes de ce répertoire montrent qu’il évolue en parallèle avec l’évolution du projet.
4. **tests** :
   * Ce répertoire contient les tests unitaires ou fonctionnels utilisés pour valider ton code. Les tests sont essentiels pour s'assurer que les nouvelles modifications n’introduisent pas de régressions dans ton projet.
   * La mention de la correction d’un fichier deploy.yaml il y a deux semaines suggère que tu as mis en place des tests pour le processus de déploiement.
5. **.gitignore** :
   * Ce fichier liste les fichiers ou répertoires à ignorer par Git. Cela inclut souvent des fichiers générés (comme les artefacts), des données sensibles, ou des configurations locales qui ne devraient pas être versionnées dans le dépôt.
6. **Procfile** :
   * Ce fichier est spécifique au déploiement sur Heroku. Il indique à Heroku quel processus doit être exécuté pour démarrer l'application (par exemple, lancer une application web ou un service).
   * Cela suggère que ton projet est déployé sur une plateforme cloud, avec un environnement d’exécution géré par Heroku.
7. **code\_final.py** :
   * Il s'agit probablement du fichier principal du projet. Ce script pourrait contenir la logique finale de l'application ou orchestrer le fonctionnement des différentes parties du projet.
8. **requirements.txt** :
   * Ce fichier contient la liste des dépendances Python nécessaires pour exécuter ton projet. Chaque librairie y est spécifiée avec sa version, permettant ainsi d'assurer que l'environnement de production utilise les mêmes dépendances que l'environnement de développement.
   * Sa mise à jour récente indique que tu as probablement ajouté ou modifié des dépendances pour ton projet.

Package :

 **Streamlit** :

* Utilisé pour créer des interfaces web interactives. Tu peux l'utiliser pour visualiser des données, afficher les résultats de ton modèle ou créer des dashboards d’interaction pour les utilisateurs.
* Il est souvent utilisé pour déployer rapidement des démonstrations de modèles ML avec une interface conviviale.

 **MLflow** :

* Un outil pour gérer le cycle de vie des modèles de machine learning. Il permet de suivre les expériences, de versionner les modèles, et de les déployer.
* Dans ton projet, il pourrait être utilisé pour gérer les expériences et faciliter le suivi des performances des modèles en production.

 **Joblib** :

* Utilisé pour la sérialisation (sauvegarde) et la désérialisation (chargement) de modèles ML. Joblib est souvent utilisé pour enregistrer des objets volumineux comme des modèles ou des pipelines, car il est plus efficace que pickle pour certains types d'objets.
* Il joue un rôle dans la persistance des modèles que tu entraînes.

 **Spacy** :

* Une bibliothèque de traitement du langage naturel (NLP) puissante. Si ton projet inclut des tâches de traitement de texte, comme l'analyse de texte, la classification ou l'extraction d'entités nommées, Spacy est probablement utilisé pour cela.

 **Pytest** :

* Utilisé pour écrire et exécuter des tests unitaires dans ton projet. Pytest permet de vérifier que le code fonctionne comme prévu, y compris les fonctions liées à ton API ou à ton modèle ML.
* Il assure la stabilité et la fiabilité du projet grâce à des tests automatisés.