**Projet Airlines**

**Objectif**

Prédire le retard d’un vol donné (OUI/NON ou nombre de minutes)

Sources de données identifiées :

* [**Lufthansa API**](https://developer.lufthansa.com/docs/read/Home) pour les détails du vol
* [**OpenWeather API**](https://openweathermap.org/api) pour la météo à départ & arrivée
* **FlightStats API** pour le statut réel du vol (historique / vérité terrain)

**Étape 1 – Collecte des données**

* **Lufthansa API**

Avec cette API publique, nous avons la possibilité de récupérer beaucoup d’informations sur les vols, les aéroports, les compagnies aériennes, les villes, les trajets etc. Ci-dessous le tableau récapitulatif des end-points dont nous avons besoin pour ce projet.

| **Category** | **Endpoint** | **Description** | **Access Level** |
| --- | --- | --- | --- |
| Reference Data | /v1/mds-references/countries/{countryCode} | Permet de récupérer des informations sur les pays, y compris les noms et les codes. | Open |
|  | /v1/mds-references/cities/{cityCode} | Permet d’accéder aux détails des villes, tels que les noms et les codes IATA correspondants. | Open |
|  | /v1/mds-references/airports/{airportCode} | Permet d’obtenir des données sur les aéroports, y compris les noms, les codes IATA et les emplacements. | Open |
|  | /v1/mds-references/airports/nearest/{latitude},{longitude} | Permet de trouver les aéroports les plus proches d’une latitude et d’une longitude données. | Open |
|  | /v1/mds-references/airlines/{airlineCode} | Permet d’obtenir des informations sur les compagnies aériennes, y compris les codes IATA et OACI, ainsi que les noms. | Open |
|  | /v1/mds-references/aircraft/{aircraftCode} | Permet d’accéder aux détails des types d’appareils et à leurs codes respectifs. | Open |
| Operations | /v1/operations/schedules/{origin}/{destination}/{fromDateTime} | Permet de récupérer les vols programmés entre des origines et des destinations spécifiées à des dates particulières. | Open |
|  | /v1/operations/customerflightinformation/{flightNumber}/{date} | Permet de vérifier le statut en temps réel de vols spécifiques, y compris les mises à jour sur les retards ou les annulations. | Open |
|  | /v1/operations/customerflightinformation/route/{origin}/{destination}/{date} | Permet d’obtenir des informations sur le statut des vols opérant entre deux aéroports à une date donnée. | Open |
|  | /v1/operations/customerflightinformation/arrivals/{airportCode}/{fromDateTime} | Permet d’accéder aux mises à jour de statut pour tous les vols arrivant à un aéroport spécifique dans une plage horaire définie. | Open |
|  | /v1/operations/customerflightinformation/departures/{airportCode}/{fromDateTime} | Permet d’accéder aux mises à jour de statut pour tous les vols au départ d’un aéroport spécifique dans une plage horaire définie. | Open |
|  | /v1/flightops/crew/weather/{airportCode} | Permet d’accéder aux informations météorologiques pour des aéroports spécifiques. | Restricted Approval |

* **OpenWeather API (OneCall 3.0)**

Cette API nous permettra de récupérer la météo à l’aéroport de départ et d’arrivée (via coordonnées), très utile pour former notre dataset et pour impacter sur le résultat final : la prédiction de retard du vol.

GET {{base\_url\_ow}}?lat=48.8566&lon=2.3522&appid={{api\_key}}&units=metric&lang=fr

Cette ressource nous permettra de collecter des informations essentielles comme :

* + Température, vent, pluie, orages, visibilité, etc.
  + Météo à l’heure du vol
  + Paramètres utiles : hourly[], alerts[]

#### **FlightStats API**

Cette API nous permettra de récupérer des données historiques sur les vols. Avec ces historiques, on a déjà un aperçu du sérieux de la compagnie et mixé tout cela avec les données météos, toujours pour prédire le retard du vol.

Données historiques et vérifiées du vol :

* + Heure réelle départ/arrivée
  + Retard constaté
  + Donnée cible : is\_delayed, delay\_minutes

**Output**

| Variable | Définition | Source | Utilité |
| --- | --- | --- | --- |
| flight\_num | Numéro du vol, ex : LH400 | Lufthansa / FlightStats | Identifie le vol. Utile pour croiser des stats. |
| date | Date du vol prévue, ex : 2025-04-25 | Utilisateur / API | Contexte temporel (jour, saison, vacances, etc.) |
| origin | Code aéroport de départ, ex : FRA | Lufthansa | Utile pour météo et typologie de l’aéroport |
| dest | ode aéroport d’arrivée, ex : JFK | Lufthansa | Pareil que origin |
| sched\_dep | Heure prévue de départ (ISO), ex : 2025-04-25T10:30:00+00:00 | Lufthansa | Permet d’analyser les retards en fonction de l’heure |
| actual\_dep | Heure réelle de départ (si disponible) | FlightStats | Permet de calculer le retard réel. Mais le pilote peut rattraper ce temps de retard par une bonne météo ce jour la lui permettant d’accélérer |
| temp | Température (°C) au moment du départ (ou arrivée) | OpenWeather | Météo = facteur de retard potentiel |
| wind | Vitesse du vent (m/s ou km/h) | OpenWeather | Trop de vent = risque de retard ou annulation |
| rain | Probabilité de pluie (ou quantité, selon API), ex : 0.2 (20%) | OpenWeather | Pluie = perturbation potentielle |
| delay\_min | Nombre de minutes de retard effectif (réel\_dep - prévu\_dep) | Calculé via données réelles | Variable cible si régression |
| is\_delayed | Booléen : 1 si vol en retard (> 15 min), sinon 0 | Dérivé de delay\_min | Variable cible si classification |

### Questions

* Le stockage NoSQL vs base de données relationnelles ?

Réponse : tout est possible , le mieux est de s’amuser => table nosql et sql

exemple; villes et aéroport en fixe et temps en variable

* La détection des changements dans les données
* Doit-on stocker l’ensemble des données relatives au vol dans une base de données

?

Pas obligé mais on peut , mais pour la performance il vaut mieux faire plus plusieurs pour une performance et un projet plus robuste

* Comment déterminer les informations qui nous seront utiles ?

C’est à nous de le voir