Tarmac Technologies

L’application AGOA permet de suivre le déroulement d’un turnaround (touchée), étape entre l'atterrissage et le décollage d’un vol sur un même aéroport.
Un vol est effectué par une compagnie aérienne, d’un aéroport à un autre selon un planning défini à l’avance. L’heure exacte de décollage et d’atterrissage peuvent cependant différer.
Le numéro de vol (généralement un nombre) d’un flight est unique par jour et par compagnie.

Exercices

1. Data

A partir des informations précisées ci-dessus, mais également de vos recherches sur le sujet, proposez une base de données permettant de stocker et de requêter des informations au sujet des flights et des turnarounds.

Vous devrez nous fournir :

1. Les scripts SQL de création des tables ;
2. Des scripts SQL afin de pré remplir ces même tables (pas plus d’une dizaine de vols).

2. Requêtes

A partir des tables créées ci-dessus, proposez les requêtes SQL permettant d’obtenir les informations suivantes :

1. Afficher les 5 vols ayant la durée de vol la plus élevée dans l’ordre décroissant ;
2. Afficher pour tous les turnarounds, leur aéroport de référence ainsi que les aéroports de départ et d’arrivée associés ;
3. Afficher les turnarounds dont la compagnie du vol d’arrivée est différente de celle du vol départ ;
4. Afficher la ponctualité de chaque turnaround :
    - la différence entre l’heure d’arrivée prévue et actuelle,
    - la différence entre l’heure de départ prévue et actuelle,
    - La différence entre la durée du turnaround prévue et actuelle ;
5. Afficher les 2 couples compagnie aérienne et aéroport ayant la meilleur ponctualité (la différence la plus faible entre durée prévue et actuelle).

3. Développement

A partir du modèle défini ci-dessus, créez un service API utilisant le protocol REST permettant de gérer des flights et turnarounds.

Context technique :

- Créez un projet *Django* vide (Django 5.0 ou supérieur) ;
- Ne passez pas de temps sur le setup de l'ORM, utilisez *SQLite* fourni directement avec le template de création startproject ;
- Votre projet devra être versionné via *Git* et partagé sur un repo privé (GitHub par exemple) ;
- Il devra comporter un Readme détaillant les étapes nécessaires à son exécution dans un environnement local ;
- Pour la partie api vous pouvez utiliser le framework *django-rest-framework*. Nous vous recommandons également d'utiliser une librairie pour l'authentification par JWT.

Réalisations:

1. Créer une route (username + login) permettant de générer un token JWT afin d'accéder aux routes de l'api (toutes les autres routes doivent être accessibles uniquement via une authentification token JWT) ;
2. Créer les routes permettant de :
    - Créer, mettre à jour, supprimer un flight,
    - Créer un turnaround ;
3. Créer une route renvoyant la liste des turnarounds et de leurs flights pour une date et un airport sélectionnés ;
4. Créer une route renvoyant pour une date donnée la durée moyenne d'un turnaround (toutes compagnies confondues).

Critères d'évaluation:

Nous apporterons une attention particulière aux éléments suivants:

- Une utilisation correcte de l'anglais dans le code et dans la documentation ;
- Le bon fonctionnement des différents routes (auth, crud et requête) ;
- Le respect des normes REST ;
- La qualité du code (lisibilité, découpage, norme PEP8, efficacité) ;
- La présence de tests unitaires automatisés.

Nous apprécierons également:

- La présence de tests fonctionnels automatisés (tests des routes API) ;
- La présence d'un pipeline de CI (GitHub actions par exemple) ;
- La présence d'un outil de typage statique (MyPy par exemple) ;
- Un workflow Git suivi et détaillé ;
- L'utilisation de dépendances maintenues et sans failles de sécurité.

Nous restons disponibles pour toute question concernant ce test (modèle, règles métier, …).

Have fun !