Documentation de lAPI REST pour la gestion et la classification des écritures bancaires

Documentation de lAPI REST pour la gestion et la classification des écritures bancaires

Ce document accompagne les fichiers `model.py` et `commande_bash.sh`, qui implémentent une API REST en Python avec Flask pour gérer et classer automatiquement des écritures bancaires, comme requis par le sujet.

Conformité au sujet

Le sujet demande une API REST avec Flask pour gérer des écritures bancaires (id, date, texte, montant) à laide de données mock et classer automatiquement les écritures à partir de fichiers CSV ou Excel. Le fichier `model.py` définit une liste `entries` avec trois écritures initiales : salaire (Revenus), loyer (Logement), courses (Alimentation). Il implémente les routes pour lister, afficher, ajouter, modifier, supprimer des écritures, et uploader des fichiers CSV/Excel. La fonction `classify_entry` attribue des catégories (Revenus, Logement, Alimentation, Services, Divers) selon des mots-clés dans le texte, comme “salaire” pour Revenus ou “facture” pour Services. La route `/api/upload` lit les fichiers, vérifie les colonnes `date`, `text`, `amount`, et ajoute les écritures avec classification automatique.

Configuration

Naviguez vers `/srv/alumni/iut2402201/tecweb/projet_web_avance/api/api_sae`. Créez un environnement virtuel avec `python3 -m venv venv` et activez-le avec `source venv/bin/activate`. Installez les dépendances avec `pip install flask pandas openpyxl`. Placez les fichiers CSV/Excel (`BANQUE.csv`, `test12.xlsx`) dans le répertoire.

Lancement

Exécutez `python3 model.py` pour démarrer le serveur sur `http://localhost:8083`. Testez avec `curl http://localhost:8083/api/entries` pour afficher les écritures mock en JSON.

Utilisation des commandes cURL

Le fichier `commande_bash.sh` contient les commandes pour interagir avec lAPI. La commande `curl http://localhost:8083/api/entries` liste toutes les écritures. La commande `curl http://localhost:8083/api/entries/1` affiche lécriture ID 1. La commande

```
curl -X POST -H "Content-Type: application/json" -d '{"date":"2025-03-04","text":"Factur
```

ajoute une écriture avec la catégorie Services. La commande

```
curl -X PUT -H "Content-Type: application/json" -d '{"amount":-75.00}' http://localhost:
```

modifie lécriture ID 1. La commande `curl -X DELETE http://localhost:8083/api/entries/1` supprime lécriture ID 1. Pour uploader un fichier, utilisez

```
curl -X POST -H "Content-Type: multipart/form-data" -F "file=@test.csv" http://localhost
```

où `test.csv` contient les colonnes `date`, `text`, `amount`.

Exemple

Lancez `python3 model.py`. Exécutez `curl http://localhost:8083/api/entries` pour lister les écritures. Ajoutez une écriture avec

```
curl -X POST -H "Content-Type: application/json" -d '{"date":"2025-03-04","text":"Factur
```

Uploadez un CSV avec

```
curl -X POST -H "Content-Type: multipart/form-data" -F "file=@test.csv" http://localhost
```

Vérifiez avec `curl http://localhost:8083/api/entries`.

Conseils

Assurez-vous que les fichiers CSV/Excel contiennent les colonnes `date`, `text`, `amount`. Testez avec Postman pour plus de simplicité. Les données sont en mémoire ; redémarrer le serveur réinitialise `entries`.