Documentation du Serveur Flask

Ce serveur est une application Flask permettant de recevoir des données envoyées par un client via une requête **POST** à l'URL `/upload`. Il affiche les données reçues dans la console et renvoie une réponse JSON au client.

Importation des modules

- **Flask** : Framework web minimaliste pour Python.
- **request** : Permet d'accéder aux données envoyées par le client dans une requête.
- **jsonify** : Permet de renvoyer une réponse JSON compréhensible par le client.

```
from flask import Flask, request, jsonify
```

Création de l'application Flask

```
app = Flask(__name__)
```

L'application permet de configurer les routes, de gérer les requêtes et de lancer le serveur web, par exemple pour localiser les fichiers statiques et les templates.

Définition d'une route

```
@app.route('/upload', methods=['POST'])
```

Cette ligne associe la fonction de traitement des données à l'URL `/upload` et accepte uniquement les requêtes **POST**.

Explication de la fonction `upload_data()`

La fonction `upload_data()` est exécutée lorsqu'une requête **POST** est envoyée à l'URL `/upload`. Elle récupère les données JSON envoyées par le client, les affiche dans la console, et renvoie une réponse appropriée.

```
def upload_data():
    data = request.json
    if data:
        print(f"Reçu: {data}")
        return jsonify({"message": "Données reçues avec succès!"}), 200
    return jsonify({"error": "Aucune donnée reçue"}), 400
```

Explication ligne par ligne :

- `data = request.json` : Extrait le contenu JSON de la requête envoyée par le client.
- `if data:` : Vérifie si des données ont été reçues.
- `print(f"Reçu: {data}")` : Affiche les données reçues dans la console du serveur.
- `return jsonify({"message": "Données reçues avec succès!"}), 200` :
  - Renvoie une réponse JSON contenant un message de confirmation.
  - **200** : Code HTTP indiquant que la requête a été traitée avec succès.
- `return jsonify({"error": "Aucune donnée reçue"}), 400` :
  - Si aucune donnée n'est envoyée, retourne une réponse JSON avec un message d'erreur.
  - **400** : Code HTTP indiquant une mauvaise requête.

Démarrage du serveur

Le serveur Flask est démarré grâce au bloc suivant :

```
if __name__ == "__main__":
    print("Démarrage du serveur...")
    app.run(host='127.0.0.1', port=5000)
```

Explication ligne par ligne :

- `if __name__ == "__main__":`
  - Vérifie si le script est exécuté directement.
  - Si le script est exécuté en tant que module importé dans un autre fichier, ce bloc ne sera pas exécuté.
- `print("Démarrage du serveur...")`
  - Affiche un message dans la console pour indiquer que le serveur démarre.

- `app.run(host='127.0.0.1', port=5000)`
  - `app.run()` : Démarre le serveur Flask.
  - `host='127.0.0.1'` : Indique que le serveur ne sera accessible que localement (localhost).
  - `port=5000` : Définit le port d'écoute du serveur (5000 par défaut).
  - **Remarque :** L'adresse IP `127.0.0.1` peut être modifiée selon les besoins.
    - Pour rendre le serveur accessible depuis un autre appareil sur le réseau local, remplacez `'127.0.0.1'` par `'0.0.0.0'`.
    - Pour utiliser une adresse spécifique sur un réseau, remplacez-la par l'IP de la machine (ex: `'192.168.1.100'`).

Ainsi, lorsqu'on exécute ce script Python, un serveur web est lancé sur l'adresse `http://127.0.0.1:5000/`. Il reste en attente des requêtes envoyées par un client.