Utiliser Python pour récupérer des données via une API

🎯 Objectif : Utiliser Python pour **envoyer une requête HTTP** à une **API web** (météo, cartographie, bourse, réseaux sociaux, etc.) et **récupérer la réponse** (souvent en JSON).

Table des matières

1. Outil de base : le module requests....1
   a. Installation (si besoin) : pip install requests....1
2. Exemple de base : récupérer des données météo (API publique)....1
   a. Exemple avec une API gratuite : https://agify.io/documentation....1
   b. Code Python :....1
   c. Résultat attendu :....2
3. Requête POST (envoyer des données)....2
4. Résumé des méthodes HTTP....2
5. API avec authentification (token)....2
6. Traitement des réponses....2
7. En résumé....2

1. Outil de base : le module `requests`

La bibliothèque **`requests`** est la façon la plus simple de faire des requêtes HTTP en Python.

a. Installation (si besoin) : `pip install requests`

2. Exemple de base : récupérer des données météo (API publique)

a. Exemple avec une API gratuite : https://agify.io/documentation

Elle prédit l'âge probable d'une personne à partir de son prénom.

b. Code Python :

```
import requests

# 1. URL de l'API
url = "https://api.agify.io"

# 2. Paramètres à envoyer dans l'URL (ex : prénom)
params = {"name": "Alice"}

# 3. Envoi de la requête GET
response = requests.get(url, params=params)

# 4. Vérification du statut de la réponse
if response.status_code == 200:
    data = response.json()  # Conversion JSON → dictionnaire Python
    print(data)
else:
    print("Erreur lors de la requête :", response.status_code)
```

c. Résultat attendu :

```
{'name': 'Alice', 'age': 30, 'count': 123456}
```

3. Requête POST (envoyer des données)

Certaines API demandent d'envoyer des données (ex: formulaire, login, etc.)

```
url = "https://httpbin.org/post"
data = {"username": "bob", "password": "secret"}

response = requests.post(url, data=data)

print(response.json())
```

4. Résumé des méthodes HTTP

Méthode	Usage
`GET`	Lire / récupérer des données
`POST`	Envoyer des données
`PUT`	Modifier des données
`DELETE`	Supprimer des données

5. API avec authentification (token)

Certaines API nécessitent un **token** d'authentification 🔐 dans les **en-têtes** :

```
headers = {
    "Authorization": "Bearer MON_TOKEN"
}

response = requests.get("https://api.example.com/data", headers=headers)
```

6. Traitement des réponses

- `response.status_code` → le code HTTP (200 = OK, 404 = Not found, etc.)
- `response.text` → texte brut de la réponse
- `response.json()` → si la réponse est au format JSON (très courant)
- `response.headers` → les en-têtes HTTP

7. En résumé

Étape	
1. Importer `requests`	`pour utiliser les fonctions de gestion d'API`
2. Définir l'URL de l'API	il faut bien préciser avec quoi on souhaite communiquer
3. Préparer et envoyer une requête (`get`, `post`, etc.)	revoir au besoin les dictionnaires
4. Lire la réponse (`.json()`, `.text`, etc.)	exploiter encore les dictionnaires...

Tu est prêt.e pour améliorer cette application

lire la doc en ligne de l'application peut aider : https://agify.io/documentation