

Appeler une API avec requests

curl et Bruno, c'est bien pour explorer. Mais pour automatiser, il faut appeler l'API depuis du code. Les exemples sont donnés en **Python**, avec la bibliothèque `requests`.

Prérequis

Python 3 et la bibliothèque `requests` : `pip install requests`.

Votre clé d'API n'a rien à faire dans le code (ni dans git). Passez-la par une variable d'environnement : `export API_KEY=votre-cle`.

Lire une ressource

```
import requests

BASE = "http://api-ritif.portlatch.com/v1"

res = requests.get(BASE + "/cocktails/3")
cocktail = res.json()
print(res.status_code, cocktail["name"])
```

Paramètres de query

```
res = requests.get(BASE + "/cocktails", params={"alcohol_free": "true", "per_page": 5})
```

Envoyer du JSON avec un header

```
import os
```

```
res = requests.post(
    BASE + "/orders",
    json={"cocktail_id": 3},
    headers={"X-API-Key": os.environ["API_KEY"]},
)
print(res.status_code, res.headers.get("Location"))
```

Avec `json=`, `requests` sérialise le body et ajoute le `Content-Type` tout seul.

Le piège : les erreurs HTTP

`requests` ne lève **pas** d'erreur sur un `404` ou un `500` : il ne le fait que si le réseau est coupé. Votre code continue donc tranquillement... avec une réponse d'erreur.

```
res = requests.get(BASE + "/cocktails/999")
res.raise_for_status() # lève une exception si le code est 4xx ou 5xx
```

En résumé

	Python (<code>requests</code>)
Code de statut	<code>res.status_code</code>
Succès (2xx) ?	<code>res.ok</code>
Header	<code>res.headers["Location"]</code>
Body JSON	<code>res.json()</code>
Envoyer du JSON	<code>json=...</code>
Lever une erreur	<code>res.raise_for_status()</code>

Revision #5

Created 2 October 2026 20:00:45 by Thibaud FRICHET

Updated 13 October 2026 19:13:37 by Thibaud FRICHET