Eigenflow

GUIDE API · OPENROUTER

Une clé API gratuite

OpenRouter et ses 11 modèles gratuits au 25 novembre 2026 : créer la clé, la brancher dans Claude Code ou Codex, ce que deviennent tes requêtes, et les revendeurs à fuir.

PAR

Abyane Nabeebaccus

INSTAGRAM

@abyane_nb

ÉDITION

novembre 2026

Tu as commenté API, voici le guide. OpenRouter te donne une seule clé pour des centaines de modèles d'IA, dont onze gratuits le 25 novembre 2026 (quatorze le 17, le jour de la vidéo). On voit comment créer la clé, choisir un modèle, la brancher dans Claude Code et dans Codex, l'appeler depuis ton code. Puis ce que deviennent tes requêtes, et pourquoi un site qui vend Claude dix fois moins cher doit te faire fuir.

Docs et catalogue public d'OpenRouter, prix et conditions d'Anthropic, conditions d'OpenAI, pages de Google, Mistral, Groq et NVIDIA relus le 17 novembre 2026, catalogue et docs d'OpenRouter relus de nouveau le 25 novembre. On n'a aucun lien avec OpenRouter. On n'a pas créé de compte pour ce guide : rien n'a été testé avec une clé, tout vient de l'API publique et des docs.

À savoir avant de lire

- Le 25 novembre à 13 h 20, l'API d'OpenRouter liste 11 modèles gratuits (identifiant en `:free`) et un routeur, `openrouter/free`. Les 11 acceptent les appels d'outils. Le 17 novembre, ils étaient 14 : Inkling 2 Small, North Mini Code et dots.3 sont sortis de la liste gratuite depuis.
- Sans rien acheter : 50 requêtes par jour, tous modèles gratuits confondus, 20 par minute. Après 10 crédits achetés (10 $), 1 000 par jour.
- Aucun Claude n'est gratuit. Claude Opus 5.5 coûte 4 $ le million de tokens en entrée et 20 $ en sortie, chez Anthropic comme chez OpenRouter.
- Claude Code se branche sur OpenRouter, mais OpenRouter ne le garantit qu'avec Claude servi par Anthropic, donc payant. Avec un modèle gratuit, rien n'est garanti.
- Six des onze modèles passent par un fournisseur qui peut entraîner ses modèles sur tes requêtes, et dix les gardent un temps. Pas de données clients, pas de secrets.
- Claude dix fois moins cher, c'est sous tous les tarifs officiels, et Anthropic interdit de revendre son accès sans accord.

---

1. Créer la clé

1. Inscris-toi sur openrouter.ai.

2. Ouvre openrouter.ai/settings/keys.

3. Crée une clé et copie-la. Elle commence par `sk-or-`.

Sans compte, on ne sait pas si l'inscription demande une carte. La FAQ promet à chaque nouveau compte « a small free allowance » pour tester.

La clé va dans une variable d'environnement, jamais dans ton code. La doc d'OpenRouter la met dans ton profil de shell (`~/.zshrc`) :

```
À COLLER

export OPENROUTER_API_KEY="sk-or-..."
```

Elle rappelle que ce fichier finit vite dans un dépôt de dotfiles ou un gist, et propose sur Mac de lire la clé dans le trousseau :

```
À COLLER

export OPENROUTER_API_KEY="$(security find-generic-password -s openrouter -w)"
```

Une clé qui fuit se révoque sur la même page. Pour voir ton compteur du jour, la page Limits renvoie à `GET /api/v1/key` :

```
À COLLER

curl https://openrouter.ai/api/v1/key -H "Authorization: Bearer
$OPENROUTER_API_KEY"
```

Le champ `free_model_daily_requests` donne les requêtes gratuites utilisées, la limite et le reste.

---

2. Choisir un modèle gratuit

Le catalogue se lit sans clé. Cette commande sort les identifiants gratuits :

```
À COLLER

curl -s https://openrouter.ai/api/v1/models | python3 -c "import json,sys;
[print(m['id']) for m in json.load(sys.stdin)['data'] if
m['id'].endswith(':free')]"
```

Le 25 novembre à 13 h 20 : 472 modèles, 11 gratuits (474 et 14 le 17 novembre). Tous répondent en texte et acceptent les outils (`tools` dans `supported_parameters`). Chacun a un seul fournisseur, et sa fiche sur openrouter.ai dit ce qu'il fait de tes requêtes :

- `google/gemma-4-31b-it:free` : Google, servi par Google AI Studio. Gardées 55 jours, pas d'entraînement.
- `google/gemma-4-26b-a4b-it:free` : la version légère, même politique.
- `nvidia/nemotron-3-ultra-550b-a55b:free` : NVIDIA, un million de tokens de contexte. Gardées, entraînement possible.

- `nvidia/nemotron-3.5-super-vl:free`, `nvidia/nemotron-3.5-lightning:free` et `nvidia/nemotron-3-nano-omni-30b-a3b-reasoning:free` : NVIDIA aussi, même politique.
- `baidu/ernie-5.5-flash-preview:free` : Baidu, en préversion. Gardées, pas d'entraînement.
- `particle-ai/particle-1:free` : Particle AI, orienté code. Gardées, entraînement possible.
- `recursive/flint-v1.0:free` : Recursive, pour les agents de code. Gardées, pas d'entraînement.
- `liquid/lfm-3-1.7b:free` : Liquid AI, minuscule, 8 192 tokens de réponse au plus. Gardées, entraînement possible.
- `inclusionai/ling-3.2-flash:free` : inclusionAI. Le seul qui ne garde rien.

Sortis de la liste gratuite entre le 17 et le 25 novembre : `thinkingmachines/inkling-2-small:free` (Thinking Machines), `cohere/north-mini-code:free` (Cohere) et `dots-studio/dots-3-note-preview:free` (Dots Studio). La vidéo, tournée le 17, en montre encore quatorze.

On ne les a pas comparés. Si ce que tu envoies a de la valeur, prends-en un sans entraînement.

Le routeur `openrouter/free` tire au hasard un modèle gratuit capable de faire ce que demande ta requête (images, outils, sortie structurée). Pratique pour essayer, sauf que tu ne choisis plus le fournisseur ni sa politique de données. Et la liste bouge souvent, la doc le dit : relance la commande avant de t'installer sur un modèle.

---

3. Brancher dans Claude Code

Claude Code parle l'API Messages d'Anthropic. OpenRouter en expose une version compatible sur https://openrouter.ai/api, et les gratuits la listaient tous le 17 novembre (`/api/v1/messages`).

L'avertissement en tête de la doc Claude Code d'OpenRouter : « Claude Code with OpenRouter is only guaranteed to work with the Anthropic first-party provider. » Plus bas : « Claude Code is optimized for Anthropic models and may not work correctly with other providers. » Avec un modèle gratuit, tu fais tourner Claude Code sans Claude. Ça peut marcher, ou casser sur les appels d'outils. On ne l'a pas testé.

Garde la config dans un dossier de test pour ne pas toucher à ton Claude Code habituel. La doc propose un fichier `.claude/settings.local.json` à la racine du projet, ici avec un modèle gratuit :

À COLLER

```
{
  "env": {
    "ANTHROPIC_BASE_URL": "https://openrouter.ai/api",
    "ANTHROPIC_AUTH_TOKEN": "<ta-clé-openrouter>",
    "ANTHROPIC_API_KEY": "",
    "ANTHROPIC_DEFAULT_OPUS_MODEL": "google/gemma-4-31b-it:free",
    "ANTHROPIC_DEFAULT_SONNET_MODEL": "google/gemma-4-31b-it:free",
    "ANTHROPIC_DEFAULT_HAIKU_MODEL": "google/gemma-4-31b-it:free",
    "CLAUDE_CODE_SUBAGENT_MODEL": "google/gemma-4-31b-it:free"
  }
}
```

Ce fichier contient ta clé, il ne part jamais dans Git. Ligne par ligne, d'après les docs :

- `ANTHROPIC_BASE_URL` envoie les requêtes chez OpenRouter. Adresse complète, avec https.
- `ANTHROPIC_AUTH_TOKEN` porte ta clé OpenRouter.
- `ANTHROPIC_API_KEY` reste vide, sinon Claude Code peut s'authentifier chez Anthropic.
- Les variables de modèle viennent du tableau « Configuring Models » d'OpenRouter : elles disent qui répond quand Claude Code demande Opus, Sonnet, Haiku ou un sous-agent. Remplace le Gemma par n'importe quel identifiant de la partie 2.
- `ANTHROPIC_DEFAULT_HAIKU_MODEL` sert aussi aux tâches de fond, d'après la doc de Claude Code. Sans lui, elles partent sur le Haiku par défaut, payant.

Si tu es connecté à Claude Code avec un compte Anthropic, la doc d'OpenRouter demande un `/logout`. Ça te déconnecte aussi de ton abonnement, il faudra un `/login` pour revenir. Relance `claude` dans le dossier, puis :

À COLLER

```
/status
```

Tu dois lire `Auth token: ANTHROPIC_AUTH_TOKEN` et `Anthropic base URL: https://openrouter.ai/api`. `/model` montre les modèles pris en compte.

Côté quota, une tâche d'agent fait plusieurs requêtes. Chaque passage du modèle (lire un fichier, lancer une commande, répondre) est un appel à l'API, plus les tâches de fond. Les 50 du jour partent vite.

---

4. Brancher dans Codex

Codex lit `~/.codex/config.toml`. Le bloc de la doc d'OpenRouter, avec un modèle gratuit à la place de celui d'OpenAI (on a retiré `model_reasoning_effort`, que la doc rattache aux modèles Codex) :

À COLLER

```
model_provider = "openrouter"
model = "google/gemma-4-31b-it:free"

[model_providers.openrouter]
name = "openrouter"
base_url = "https://openrouter.ai/api/v1"

[model_providers.openrouter.auth]
command = "sh"
args = ["-c", "echo $OPENROUTER_API_KEY"]
```

La doc précise que `model` accepte n'importe quel identifiant OpenRouter. Exporte `OPENROUTER_API_KEY` avant (sinon erreur 401), puis dans ton projet :

À COLLER

```
codex
```

Les deux premières lignes changent le fournisseur de Codex partout. Si tu utilises Codex avec ton compte ChatGPT, passe par un profil. D'après la doc de Codex, `codex --profile <nom>` pose le fichier `~/.codex/<nom>.config.toml` par-dessus ta config. Les tableaux `[model_providers.openrouter]` restent dans `config.toml`, où la doc veut les fournisseurs. Tu crées `~/.codex/openrouter.config.toml` :

À COLLER

```
model_provider = "openrouter"
model = "google/gemma-4-31b-it:free"
```

À COLLER

```
codex --profile openrouter
```

Même réserve que pour Claude Code : Codex est réglé pour les modèles d'OpenAI, et on ne l'a pas lancé sur un gratuit.

---

5. Appeler l'API en curl ou en Python

Pour tes scripts et tes propres agents, l'exemple de la doc, en curl :

À COLLER

```
curl https://openrouter.ai/api/v1/responses \
  -H "Authorization: Bearer $OPENROUTER_API_KEY" \
  -H "Content-Type: application/json" \
  -d '{
    "model": "openrouter/free",
    "input": "Hello!"
  }'
```

Le champ `model` de la réponse dit quel modèle a répondu. En Python avec requests, toujours la doc (mets ta clé à la place de `<OPENROUTER_API_KEY>`, ou lis-la dans la variable d'environnement) :

À COLLER

```
import requests
import json

response = requests.post(
  url="https://openrouter.ai/api/v1/responses",
  headers={
    "Authorization": "Bearer <OPENROUTER_API_KEY>",
    "Content-Type": "application/json",
  },
  data=json.dumps({
    "model": "openrouter/free",
    "input": "Hello! What can you help me with today?"
  })
)

data = response.json()
message = next(item for item in reversed(data['output']) if item['type'] ==
'message')
print(next(part['text'] for part in message['content'] if part['type'] ==
'output_text'))
print('Model used:', data['model'])
```

Pour un modèle précis, remplace `openrouter/free` par son identifiant, par exemple `recursive/flint-v1.0:free`.

Attention aux modèles de secours. Si ta requête en liste dans un tableau `models`, OpenRouter passe au suivant quand le gratuit sature, et un modèle payant de la liste est facturé à son prix. Pour rester à zéro : que des `:free`, ou `openrouter/free` seul, qui ne retombe que sur des gratuits.

---

6. Les limites et tes données

Les chiffres

La page Limits d'OpenRouter, pour les identifiants en `:free` :

- moins de 10 crédits achetés depuis l'ouverture du compte : 20 requêtes par minute, 50 par jour ;
- au moins 10 crédits : 20 par minute, 1 000 par jour.

Les 50 valent pour tous les modèles gratuits ensemble (FAQ : « 50 requests per day total »). Le compteur suit le jour UTC, donc il repart à 1 h du matin à Paris en hiver, 2 h en été. La doc ne cite pas le routeur, mais il ne choisit que parmi les `:free` : compte qu'il puise dans le même quota.

Un crédit vaut un dollar, et les prix d'API sont tous en dollars. OpenRouter prend 5,5 % de frais à l'achat, 0,80 $ au minimum : 10 crédits coûtent 10,80 $, environ 9,83 € au cours BCE du 17 novembre, hors taxes éventuelles. Un solde négatif peut bloquer même les modèles gratuits, prévient la page Limits.

Tes données

D'après sa FAQ, OpenRouter n'enregistre ni tes requêtes ni les réponses par défaut. Il garde l'heure, le modèle et le nombre de tokens. Le fournisseur, lui, applique sa propre politique : sur les onze gratuits du 25 novembre, six peuvent entraîner leurs modèles sur ce que tu envoies, et seul Ling 3.2 Flash ne garde rien.

Ton compte a un réglage pour refuser les fournisseurs qui entraînent, avec un interrupteur pour les modèles gratuits et un autre pour les payants (doc Provider Logging). Refusé, OpenRouter ne t'envoie plus chez eux, et les modèles NVIDIA, Particle-1 et LFM3 te deviennent inaccessibles.

Avec un modèle gratuit, on n'envoie ni données de clients (noms, mails, factures) ni secrets, clés API, mots de passe ou fichier `.env`. Dans Claude Code ou Codex, ce que l'agent lit dans ton dossier part dans la requête, `.env` compris s'il l'ouvre. Les conditions d'essai de NVIDIA interdisent d'ailleurs d'y mettre des informations confidentielles ou des données personnelles.

---

7. Les revendeurs à éviter

Prix officiel de Claude Opus 5.5 sur la page d'Anthropic le 17 novembre : 4 $ le million de tokens en entrée, 20 $ en sortie, hors taxes. Environ 3,64 € et 18,20 €. Dix millions de tokens en sortie : 200 $, environ 182 € HT.

Les remises officielles existent : l'API Batch fait moitié prix (2 $ et 10 $), la lecture en cache coûte 0,20 $ le million. Aucune ne divise tout par dix. OpenRouter affiche Opus 5.5 au prix d'Anthropic et dit se payer sur les frais d'achat de crédits.

Un site de ce genre, regardé le 17 novembre, promettait « jusqu'à 10 fois moins cher que les tarifs officiels ». Il affichait Claude Opus 5.5 à 0,50 $ le million, en entrée comme en sortie : huit fois moins que le prix officiel en entrée, quarante fois moins en sortie. Il vendait aussi un résolveur de captchas, et une passerelle en marque blanche pour revendre à ton tour.

Personne ne vend sous le prix d'Anthropic sans perdre de l'argent à chaque requête, sauf si l'accès vient d'ailleurs : clés ou comptes revendus, abonnements partagés, crédits d'origine inconnue. On ne sait pas d'où ce site tire le sien. Ce que disent les conditions :

- Anthropic, Commercial Terms of Service (17 juin 2025), article D.4 : le client ne doit pas « resell the Services except as expressly approved by Anthropic ». L'article I.3 permet de suspendre l'accès en cas de manquement.
- Anthropic, Consumer Terms (8 octobre 2025) : « You may not share your Account login information, Anthropic API key, or Account credentials with anyone else or make your Account available to anyone else. »
- OpenAI, Contrat de services (1er janvier 2026) : « Il est interdit au Client de revendre ou de louer l'accès à son Compte », comme « d'acheter, de vendre ou de transférer des clés API à un tiers ».

Un accès revendu peut donc sauter du jour au lendemain, avec tes crédits dedans.

Et ton code passe par eux. Un intermédiaire d'API reçoit chaque requête en entier avant de la transmettre : avec Claude Code, ton prompt, les fichiers que l'agent lit, la sortie des commandes qu'il lance. Le site peut tout garder, et rien ne te prouve que la réponse vient de Claude. Compare toujours le prix affiché à la page officielle de l'éditeur.

---

8. Les autres clés gratuites officielles

Pages relues le 17 novembre 2026.

Google AI Studio (Gemini)

L'offre Free de l'API Gemini donne des tokens gratuits sur une partie des modèles : ce jour-là Gemini 4 Hydrogen, Gemini 3.8 Flash, Gemini 3.5 Flash-Lite, Gemini 2.5 Pro et Gemma 4, entre autres, mais pas Gemini 4 Carbon ni Argon. Les limites dépendent de ton projet et s'affichent dans AI Studio, et le quota du jour repart à minuit, heure du Pacifique.

Sur les services non payants, les conditions du 23 mars 2026 laissent Google se servir de tes requêtes, avec relecture humaine possible : « Do not submit sensitive, confidential, or personal information to the Unpaid Services. » Depuis la France, comme partout dans l'Espace économique européen, ce sont les règles de données du payant qui s'appliquent, même sur le quota gratuit. Une appli ouverte à des utilisateurs européens doit en revanche passer par le payant. Clé : aistudio.google.com/api-keys.

Mistral

Pas de clé API gratuite aujourd'hui. Sa doc (Usage and limits) dit du mode Free : « It doesn't include API access ». Créer une clé demande d'activer la facturation et d'acheter des crédits.

Groq

Il y a une offre Free : la FAQ de facturation ne demande un moyen de paiement que pour passer à l'offre Developer. Tes limites exactes s'affichent sur la page Limits du compte. Groq ne garde pas les requêtes d'inférence par défaut, sauf des journaux jusqu'à 30 jours en cas de panne ou d'abus, que tu peux couper dans Data Controls. Clé : console.groq.com/keys.

NVIDIA

build.nvidia.com annonce « Free inference with leading models », dont Nemotron 3 Ultra. La FAQ de NVIDIA parle d'un accès gratuit pour le prototypage via le NVIDIA Developer Program, avec des limites par modèle affichées en haut à droite du site. Les conditions d'essai (19 septembre 2025) excluent la production.

Sources

- Catalogue des modèles d'OpenRouter, lu le 17 novembre 2026 à 19 h 47, et la fiche de chaque modèle gratuit sur openrouter.ai.
- Docs d'OpenRouter : FAQ, Limits, Free Variant, Free Models Router, Claude Code, Codex CLI, Provider Logging, Data Collection.
- Anthropic : prix, Commercial Terms, Consumer Terms. OpenAI : Contrat de services.
- Claude Code : modèles, variables d'environnement. Codex : configuration avancée.
- Google : prix, limites, conditions. Mistral : Usage and limits. Groq : facturation, données. NVIDIA : build.nvidia.com, FAQ, conditions d'essai.
- Cours euro-dollar : BCE, 17 novembre 2026, 1 € = 1,0989 $.

Abyane · @abyane_nb · eigenflow.fr