QuickGPT Vendor API — Integration Guide

This document describes how to authenticate against the QuickGPT Vendor API and pull prepaid codes for your plan SKUs.

1. Overview

QuickGPT issues prepaid codes that end users redeem in their wallet for credits. As a vendor, you are issued an API key and a per-SKU **credit line** (a prepaid pool of codes you may pull). Each successful issue call debits your balance by the number of codes returned. When the balance hits zero you will receive a **402 insufficient_credit_line** error until your line is topped up.

2. Credentials

You will receive two things from QuickGPT:

• An **API key** beginning with `qg_live_`. It is shown **only once** at creation — store it in a secret manager. If lost, request a key rotation.

• One or more **Plan SKUs** you are entitled to pull. The available SKUs are:

Plan SKU	Credits per code	Notes
`pro_500`	500	Starter plan code
`pro_2000`	2,000	Standard plan code
`pro_11000`	11,000	Power plan code

Use the SKU string exactly as shown (case-sensitive) in the `plan_sku` field.

3. Authentication

Pass your key as a Bearer token in the `Authorization` header on every request:

```
Authorization: Bearer qg_live_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Requests are server-to-server only. Do not embed the key in browser, mobile, or other client-side code.

4. Base URL

```
https://quickgpt.ai
```

5. Endpoints

5.1 POST /api/public/vendor/codes/issue

Pulls one or more unused codes for the requested plan SKU and debits your credit line.

Request body (JSON):

```
{
  "plan_sku":  "pro_500",        // required: pro_500 | pro_2000 | pro_11000
  "count":     1,                 // optional, 1-100, default 1
  "issue_ref": "your-order-123"   // optional, your idempotency key
```

```
}
```

200 OK response:

```
{
  "issue_ref":    "your-order-123",
  "plan_sku":     "pro_500",
  "codes": [
    { "code": "ABCD-EFGH-JKLM-NPQR", "credits": 500 }
  ],
  "balance_after": 49
}
```

Idempotency. If you re-send the same `issue_ref`, the API returns the original codes without re-debiting your balance. Always set `issue_ref` in production so retries after a network error are safe.

5.2 GET /api/public/vendor/codes/availability

Check remaining stock in the pool for a SKU and your current balance.

```
GET /api/public/vendor/codes/availability?plan_sku=pro_500
{
  "plan_sku":       "pro_500",
  "remaining_pool": 1240,   // codes available to issue across all vendors
  "balance":        49      // your remaining credit line for this SKU
}
```

6. Error Codes

HTTP	error	Meaning
400	`invalid_json`	Body could not be parsed.
401	`invalid_key`	Missing or unknown API key.
402	`insufficient_credit_line`	Your balance for this SKU is less than count. Request a top-up.
403	`vendor_disabled`	Your account is currently disabled. Contact QuickGPT.
409	`out_of_stock`	Not enough unused codes exist in the pool for this SKU.
422	`validation`	Body failed validation (see details field).
422	`invalid_count`	count must be between 1 and 100.
429	`daily_limit_exceeded`	You hit your per-vendor daily issuance cap.
500	`server_error`	Unexpected error. Safe to retry with same issue_ref.

7. Code Examples

7.1 curl

```
curl -X POST "https://quickgpt.ai/api/public/vendor/codes/issue" \
  -H "Authorization: Bearer $QG_API_KEY" \
  -H "Content-Type: application/json" \
  -d '{"plan_sku":"pro_500","count":1,"issue_ref":"order-123"}'
```

7.2 Node.js (fetch)

```
const res = await fetch(
  "https://quickgpt.ai/api/public/vendor/codes/issue",
  {
    method: "POST",
    headers: {
      "Authorization": `Bearer ${process.env.QG_API_KEY}`,
      "Content-Type": "application/json",
    },
    body: JSON.stringify({
      plan_sku:  "pro_2000",
      count:     1,
      issue_ref: orderId,   // your order id as idempotency key
    }),
  }
);
if (!res.ok) throw new Error(`Issue failed: ${res.status} ${await res.text()}`);
const { codes, balance_after } = await res.json();
```

7.3 Python (requests)

```
import os, requests
r = requests.post(
    "https://quickgpt.ai/api/public/vendor/codes/issue",
    headers={"Authorization": f"Bearer {os.environ['QG_API_KEY']}"},
    json={"plan_sku": "pro_11000", "count": 1, "issue_ref": order_id},
    timeout=15,
)
r.raise_for_status()
data = r.json()
for c in data["codes"]:
    print(c["code"], c["credits"])
```

8. Best Practices

• **Always** set `issue_ref` to a unique value per order so retries are idempotent.

• Treat issued codes as cash — store them encrypted and deliver to your end user over a secure channel.

• Poll `/availability` before bulk pulls to confirm stock and your balance.

• On `402` or `409`, stop and contact QuickGPT — do not loop.

• Retry `5xx` with exponential backoff, reusing the same `issue_ref`.

• Rotate your key immediately if it may have been exposed.

9. How the End User Redeems

Once you deliver a code to your customer, they sign in to QuickGPT, open their Wallet, and paste it into the *Redeem code* field. The credits are added instantly. Each code can be redeemed only once.

10. Support

For new SKU entitlements, credit-line top-ups, key rotation, or any integration questions, contact your QuickGPT account manager.