PARTNER API · v1.1

دليل ربط الشركاء

الدليل الرسمي والكامل لربط متجرك أو تطبيقك أو بوتك بـ**AXIS STORE**: استعرض الكتالوج بأسعار شريحتك، وأنشئ الطلبات برمجياً، وتابع تسليمها — بلا أي تدخّل يدوي.

الإصدار	تاريخ الإصدار	عنوان الواجهة	الناشر
v1.1	**2026-09-27**	**https://YOUR-DOMAIN/api/partner/v1**	**AXIS GROUP**

المحتويات

كل قسم مستقل — ابدأ بـ«البدء السريع» إن أردت أول طلب خلال دقائق.

1 نظرة عامة

واجهة REST بسيطة بصيغة JSON، مصمَّمة لتُبنى عليها متاجر التجزئة والبوتات وأنظمة الموزّعين.

تتيح لك واجهة الشركاء أن تبيع منتجات **AXIS STORE** من داخل نظامك أنت: تجلب الكتالوج بأسعار شريحتك، وتنشئ الطلب نيابةً عن عميلك، فيُخصم ثمنه من محفظتك لدينا، ويُنفَّذ آلياً، ثم تستلم بيانات التسليم (كوداً أو تأكيد شحن) لتعرضها لعميلك.

الكتالوج بأسعارك

كل الأقسام والمنتجات، بحقول الطلب وقيود الكمية، بسعر شريحتك مباشرةً.

طلبات آمنة للتكرار

مرجعك الفريد يمنع ازدواج الطلب والخصم حتى لو أعدت الإرسال بعد انقطاع.

تنفيذ آلي

الطلب يُرسَل للتنفيذ فور إنشائه، وتتابع حالته حتى الاكتمال أو الاسترداد.

كيف يجري الطلب

`GET /products` → `GET /quote` → `POST /orders` → `GET /orders/{ref}` → `completed ✓`

1. تجلب المنتج لتعرف **الحقول المطلوبة** (مثل معرّف اللاعب) و**قيود الكمية**.
2. تسعّر الطلب اختيارياً لتعرض لعميلك المبلغ الدقيق.
3. تنشئ الطلب بمرجعٍ فريد من طرفك — يُخصم ثمنه من محفظتك ويبدأ تنفيذه.
4. تتابع حالته حتى يصير `completed` فتعرض بيانات التسليم، أو `refunded` فيعود المبلغ لمحفظتك تلقائياً.

2 من يحق له الربط

الربط البرمجي ميزة لشركائنا في الشرائح العليا، بعد موافقة الإدارة.

الشريحة	هامشنا على سعر المصدر	الربط البرمجي
شريك VIP	4%	متاح بعد الموافقة
شريك VIP PRO	3%	متاح بعد الموافقة

- اطلب الترقية من صفحة **«مستواي والأسعار»** في حسابك، وتُراجَع من الإدارة.
- الأسعار في الواجهة هي أسعار شريحتك **لحظة الطلب**: ترقيتك تخفّضها فوراً على كل نداء.
- إن سُحبت الصلاحية من شريحتك تتوقف مفاتيحك فوراً عن العمل (تُعاد `401`).

3 البدء السريع

من صفر إلى أول طلب في خمس خطوات.

1. **أنشئ مفتاحاً:** من حسابك ← «الربط البرمجي API» ← «مفتاح جديد». يبدأ المفتاح بـ `axk_` و**يُعرَض مرة واحدة فقط** — احفظه فوراً في مكان آمن على خادمك.
2. **اشحن محفظتك:** كل طلب يُخصم من رصيد محفظتك لدينا. تأكّد من الرصيد عبر `/me`.
3. **تحقّق من الربط:** نداء `GET /me` يعيد اسمك وشريحتك ورصيدك.
4. **اجلب منتجاً:** `GET /products?q=...` لتعرف `slug` المنتج وحقوله المطلوبة.
5. **أنشئ الطلب وتابعه:** `POST /orders` بمرجعك الفريد، ثم `GET /orders/{reference}` حتى الاكتمال.

```
# 1) Verify the connection
curl -H "X-API-Key: axk_YOUR_KEY" https://YOUR-DOMAIN/api/partner/v1/me

# 2) Find a product
curl -H "X-API-Key: axk_YOUR_KEY" "https://YOUR-DOMAIN/api/partner/v1/products?q=pubg"

# 3) Create an order — with your own unique reference
curl -X POST https://YOUR-DOMAIN/api/partner/v1/orders \
  -H "X-API-Key: axk_YOUR_KEY" -H "Content-Type: application/json" \
  -d '{"product":"pubg-uc-60","quantity":1,"reference":"SHOP-10001","fields":
{"player_id":"5123456789"}}'

# 4) Track its status
curl -H "X-API-Key: axk_YOUR_KEY" https://YOUR-DOMAIN/api/partner/v1/orders/SHOP-10001
```

4 أساسيات الواجهة

قواعد ثابتة تنطبق على كل نقطة نهاية.

العنوان الأساسي	`https://YOUR-DOMAIN/api/partner/v1`
البروتوكول	HTTPS دائماً — المفتاح يُرسَل في كل نداء، ولا يجوز أن يمرّ عبر اتصال غير مشفّر.
الصيغة	JSON بترميز UTF-8 في الطلب والرد. أرسل `Content-Type: application/json` مع كل `POST`.

المصادقة	ترويسة `X-API-Key` في كل نداء (القسم 5).
المبالغ	بالدولار الأمريكي **كنصوص عشرية** مثل `"12.50"` أو `"0.000607"` — لا تحوّلها إلى `float` ؛ استخدم نوعاً عشرياً دقيقاً (`Decimal` / `BigDecimal`).
التواريخ	ISO 8601، مثل `2026-09-27T14:05:31` .
لغة البيانات	المعامل `lang=ar` (افتراضي) أو `lang=en` على نقاط الكتالوج — يغيّر أسماء المنتجات والأقسام والحقول.
لغة رسائل الخطأ	ترويسة `Accept-Language: en` للإنجليزية؛ العربية افتراضاً.
الصفحات	القوائم تقبل `page` (من 1) و `per_page` (حتى 100)، وتعيد `total` و `pages` .
الإصدار	المسار يحمل الإصدار الرئيسي `/v1` . نضيف حقولاً جديدة دون كسر ما سبق — تجاهل أي حقل لا تعرفه.

5 المصادقة وأمان المفتاح

مفتاحك يعادل محفظتك: من يملكه يستطيع الشراء برصيدك.

```
X-API-Key: axk_3Jq9vT2kR8...
```

لا تضع المفتاح في الرابط أبداً.

المفتاح في سلسلة الاستعلام (?key=) لا يُقبل ويُعاد 401، لأن الروابط تُحفظ في سجلات كل خادم ووسيط على الطريق.

- **يُعرَض مرة واحدة** عند إنشائه، ونخزّن بصمته فقط — لا نستطيع استرجاعه لك. من فقده أنشأ غيره.
- **احفظه على خادمك فقط** (متغيّر بيئة أو خزنة أسرار) — لا في تطبيق جوال ولا في كود متصفح ولا في مستودع كود.
- **حتى 5 مفاتيح فعّالة** لكل حساب — مفتاحٌ لكل نظام (متجر، بوت...) يسهّل الإلغاء المنفرد.
- **الإلغاء فوري:** من صفحة مفاتيحك؛ أي نظام يستخدم المفتاح يتلقى `401` من النداء التالي.
- ترى لكل مفتاح **عدد النداءات وآخر استخدام** — راقبها لاكتشاف أي استخدام غير متوقّع.

عند الاشتباه بتسريب:

أنشئ مفتاحاً جديداً، حدّث نظامك به، ثم ألغِ القديم — بلا انقطاع للخدمة.

6 حدود الاستخدام

حدودٌ سخية لكل مفتاح تحمي استقرار الخدمة للجميع.

الحدّ	القيمة
كل النداءات لكل مفتاح	**120** نداءً في الدقيقة
إنشاء الطلبات لكل حساب	**60** طلباً في الدقيقة
مراجع الاستعلام الدفعي	**50** مرجعاً في النداء
عناصر الصفحة الواحدة	**100**

عند تجاوز الحد تُعاد `429` مع ترويسة `Retry-After` (بالثواني) والحقل `details.retry_after` :

```
HTTP/1.1 429 Too Many Requests
Retry-After: 17

{"code": "rate_limited", "message": "محاولات كثيرة. أعد المحاولة بعد 17 ثانية.", "details":
{"retry_after": 17}}
```

انتظر المدة المذكورة ثم أعد المحاولة. القيم الحالية تعيدها `/me` في `limits` — اقرأها من هناك بدل تثبيتها في كودك.

7 الحساب والرصيد

أول نداء تجريه — ونقطة فحص صحة الربط.

`GET` `/me`

يعيد هوية الحساب وشريحته ورصيد المحفظة وحدود الاستخدام.

مثال الرد — 200

```
{
  "account_no": "90111620",
  "account_seq": 1024,
  "name": "متجر النجم",
  "tier": { "key": "vip", "name": "شريك VIP", "margin_pct": "4.0000" },
  "balance_usd": "250.00",
  "balance_display": "$250.00",
  "currency": "USD",
  "api_version": "1.1",
  "limits": { "requests_per_minute": 120, "orders_per_minute": 60, "batch_max": 50
}
}
```


نصيحة:

راقب `balance_usd` دورياً ونبّه فريقك حين ينخفض — الطلب برصيدٍ غير كافٍ يُرفض بـ402.

8 الأقسام

لبناء قوائم التصفّح في متجرك أو لمزامنة الكتالوج قسماً قسماً.

GET `/categories`

المعامل	النوع	الوصف
`lang`	ar | en	لغة الأسماء. الافتراضي `ar`.

مثال الرد — 200


```
{
  "items": [
    { "slug": "pubg-global", "name": "ببجي العالمية", "folder": "الألعاب",
      "image": "https://YOUR-DOMAIN/media/...", "products_count": 12 }
  ],
  "total": 1
}
```


تُعاد الأقسام التي فيها منتجات فعّالة فقط. مرّر `slug` القسم إلى `/products?category=`.

9 المنتجات

كل ما تحتاجه لعرض المنتج وطلبه: السعر بشريحتك، الحقول المطلوبة، وقيود الكمية.

GET `/products`

المعامل	النوع	الوصف
`q`	نص	بحث في اسم المنتج.
`category`	نص	`slug` القسم.
`page` / `per_page`	عدد	الصفحة (من 1) وعدد العناصر (1–100، الافتراضي 50).
`lang`	ar | en	لغة الأسماء والحقول.

مثال الرد — 200


```
{
  "items": [{
    "slug": "pubg-uc-60",
    "name": "60 شدة",
    "category": "ببجي العالمية", "category_slug": "pubg-global",
    "image": "https://YOUR-DOMAIN/media/...",
    "price_usd": "0.9152",
    "min_order_total_usd": "0.9152",
    "in_stock": true,
    "fields": [
      { "key": "player_id", "label": "معرّف اللاعب", "hint": "", "type": "number",
"required": true }
    ],
    "min_quantity": 1, "max_quantity": 0, "quantity_step": 1,
    "allowed_quantities": [], "price_unit": 0
  }],
  "page": 1, "per_page": 50, "total": 1, "pages": 1
}
```


حقول المنتج

الحقل	المعنى
`slug`	المعرّف الثابت للمنتج — استخدمه في `/quote` و `/orders` .

الحقل	المعنى
`price_usd`	سعر **الوحدة** بشريحتك (أو سعر كل `price_unit` وحدة إن كانت أكبر من صفر).
`min_order_total_usd`	ثمن **أقلّ كمية** يمكن طلبها — الأنسب للعرض حين يكون سعر الوحدة كسراً من السنت.
`in_stock`	هل المنتج متاح للطلب الآن.
`fields`	الحقول التي **يجب** إرسالها مع الطلب (القسم 11).
`min_quantity` / `max_quantity`	أقل وأكثر كمية (`0` = بلا حدّ أعلى).
`quantity_step`	الكمية يجب أن تكون من مضاعفاته.
`allowed_quantities`	إن لم تكن فارغة: **هذه الكميات وحدها** مقبولة.
`price_unit`	إن كان أكبر من صفر: `price_usd` هو ثمن كل `price_unit` وحدة (القسم 14).

أنواع الحقول (`fields[].type`)

النوع	التحقق على خادمنا
`text`	نص حتى 2000 حرف.
`number`	أرقام فقط (يُسمح بفاصلة عشرية واحدة).
`email`	بريد إلكتروني صالح.
`url`	رابط يبدأ بـ http أو https.

GET `/products/{slug}`

منتج واحد بكل ما سبق، إضافةً إلى `description` (وصف المنتج وشروطه) و `variants` (الخيارات إن وُجدت):

```
{
  "slug": "netflix-premium", "name": "نتفليكس بريميوم",...  ,
  "description": "يُسلَّم خلال 5 دقائق…",
  "variants": [
    { "id": 41, "name": "شهر", "price_usd": "6.20" },
    { "id": 42, "name": "3 أشهر", "price_usd": "17.40" }
  ]
}
```

إن كانت `variants` غير فارغة فأرسل `variant_id` المختار مع الطلب.

مزامنة الكتالوج:
خزّن المنتجات لديك وحدّثها كل 15–60 دقيقة، لا عند كل زيارة لمتجرك. الأسعار تتغيّر بتغيّر تكلفة المصدر، والسعر الملزِم هو ما يُحسب لحظة الطلب.

10 التسعير المسبق

احسب ثمن الطلب الدقيق قبل إنشائه — مع التحقق من قيود الكمية.

`GET` `/quote`

المعامل	النوع	الوصف
`product` *	نص	`slug` المنتج.
`quantity`	عدد	الكمية (الافتراضي 1).
`variant_id`	عدد	الخيار، إن كان للمنتج خيارات.

```
curl -H "X-API-Key: axk_YOUR_KEY" "https://YOUR-DOMAIN/api/partner/v1/quote?
product=tiktok-followers&quantity=5000"

{
  "product": "tiktok-followers", "quantity": 5000, "variant_id": null,
  "unit_price_usd": "1.50", "price_unit": 1000,
  "total_usd": "7.50"
}
```

التسعير **إعلامي**: يُعاد حسابه داخل معاملة الطلب نفسه، ولا يُحجز السعر. كمية تخالف القيود تُرفض هنا بنفس خطأ الطلب (`409` `product_unavailable`) — استخدمه للتحقق قبل الخصم.

11 إنشاء طلب

يُخصم الثمن من محفظتك ويبدأ التنفيذ فوراً — في نداء واحد.

`POST` `/orders`

الحقل	النوع	الوصف
`product` *	نص	`slug` المنتج.
`reference` *	نص ≤ 100	**مرجعك الفريد** لهذا الطلب (رقم طلب متجرك مثلاً). يمنع الازدواج — القسم 15.
`quantity`	عدد	الكمية (الافتراضي 1) وفق قيود المنتج.
`fields`	كائن	قيم الحقول: `{"player_id": "5123456789"}` — المفاتيح هي `fields[].key` من المنتج.
`variant_id`	عدد	الخيار، إن كان للمنتج خيارات.
`note`	نص ≤ 500	ملاحظة داخلية تظهر مع الطلب.

```
curl -X POST https://YOUR-DOMAIN/api/partner/v1/orders \
  -H "X-API-Key: axk_YOUR_KEY" -H "Content-Type: application/json" \
  -d '{
    "product": "pubg-uc-60",
    "quantity": 1,
    "reference": "SHOP-10001",
    "fields": { "player_id": "5123456789" }
  }'
```

الرد — 201

```
{
  "reference": "AX-ORD-3F9A2B01",
  "partner_reference": "SHOP-10001",
  "status": "processing",
  "product": "pubg-uc-60", "product_name": "60 شدة",
  "quantity": 1, "unit_price_usd": "0.9152", "total_usd": "0.9152",
  "fields": { "player_id": "5123456789" },
  "delivery": "",
  "created_at": "2026-09-27T14:05:31", "completed_at": null
}
```


- الرد يحمل مرجعنا `reference` ومرجعك `partner_reference` — تستطيع المتابعة بأيٍّ منهما.
- منتجات كثيرة تكتمل خلال ثوانٍ فيعود الرد مباشرةً بـ `completed` ومعه `delivery` .
- السعر يُحسب على خادمنا دائماً؛ لا يُقبل أي حقل سعر من طرفك.

الحقول المطلوبة:

حقلٌ `required` ناقص أو بصيغة خاطئة يُرفض بـ422 **قبل** أي خصم. تحقّق منها في متجرك قبل الإرسال لتعرض لعميلك رسالة واضحة.

12 متابعة الطلبات

طلبٌ واحد، أو دفعة مراجع، أو كل طلباتك صفحةً صفحة.

GET /orders/{reference}

حالة طلب واحد — `{reference}` مرجعنا (`AX-ORD-…`) أو مرجعك أنت. الرد بنفس شكل رد الإنشاء.


```
curl -H "X-API-Key: axk_YOUR_KEY" https://YOUR-DOMAIN/api/partner/v1/orders/SHOP-10001

{ "reference": "AX-ORD-3F9A2B01", "partner_reference": "SHOP-10001", "status": "completed",
  ..., "delivery": "5123456789 تم الشحن إلى المعرّف", "completed_at": "2026-09-27T14:05:44" }
```

GET /orders?references=A,B,C

استعلام دفعي لحتى 50 مرجعاً (مرجعنا أو مرجعك، مفصولة بفواصل) في نداء واحد — الطريقة الموصى بها لمتابعة عدّة طلبات معلّقة:

```
{
  "items": [ { "partner_reference": "SHOP-10001", "status": "completed", ... },
             { "partner_reference": "SHOP-10002", "status": "processing", ... } ],
  "missing": [ "SHOP-99999" ]
}
```

تُعاد العناصر بترتيب المراجع المرسلة، و `missing` يسرد ما لم يُعثر عليه.

GET /orders

كل طلباتك، الأحدث أولاً.

المعامل	الوصف
`status`	تصفية بحالة واحدة، مثل `processing` .
`per_page` / `page`	الصفحة وعدد العناصر (حتى 100).

```
{ "items": [ ... ], "page": 1, "per_page": 50, "total": 318, "pages": 7 }
```

13 دورة حياة الطلب

كل طلب ينتهي إلى إحدى حالتين: مكتمل بتسليمه، أو مُسترَد إلى محفظتك.

`paid` → `processing` → `completed`

`processing` → `refunded` (يعود المبلغ لمحفظتك تلقائياً)

الحالة	المعنى	نهائية؟	ماذا تفعل
paid	خُصم الثمن والطلب في طريقه للتنفيذ.	لا	تابع.

الحالة	المعنى	نهائية؟	ماذا تفعل
processing	قيد التنفيذ لدى المصدر.	لا	تابِع كل 10–30 ثانية.
completed	اكتمل؛ بيانات التسليم في `delivery`.	نعم	اعرض التسليم لعميلك وتوقّف عن المتابعة.
refunded	تعذّر التنفيذ فأُعيد المبلغ كاملاً إلى محفظتك.	نعم	أبلغ عميلك وأعد له ماله في متجرك.
rejected / failed	رُفض الطلب.	نعم	راجع الطلب في حسابك أو تواصل مع الدعم.
cancelled	أُلغي قبل الدفع.	نعم	لا خصم على هذا الطلب.

عن `delivery`:

نصّ يصف التسليم (كود، بيانات حساب، أو تأكيد شحن)، ويظهر **فقط** عند completed. قد يحتوي عدّة أسطر — اعرضه كما هو مع الحفاظ على فواصل الأسطر.

الاسترداد تلقائي:

لا تحتاج أن تطلبه. حين يصير الطلب refunded يكون المبلغ قد عاد إلى رصيدك لدينا فعلاً.

14 الكميات والتسعير

ثلاثة أنماط للكمية — تعامل معها في متجرك كما يلي.

أ) منتج بالقطعة — `price_unit = 0`

الإجمالي = `price_usd × quantity`. بطاقة هدايا بـ `"5.20"` × 3 = `"15.60"`.

ب) منتج بالوحدة المجمّعة — `price_unit > 0`

`price_usd` هو ثمن كل `price_unit` وحدة، والكمية يجب أن تكون من مضاعفاتها. متابعون بـ `"1.50"` لكل 1000؛ طلب 5000 = 1.50 × 5 = `"7.50"`.

ج) كميات محدَّدة — `allowed_quantities` غير فارغة

تُقبل هذه القيم فقط، مثل `[110, 150, 210]`. اعرضها لعميلك كخيارات لا كحقل حرّ.

القيد	عند مخالفته
أقل من `min_quantity`	`409 product_unavailable` مع رسالة توضّح القيد — قبل أي خصم.
أكثر من `max_quantity` (إن لم يكن 0)	
ليس من مضاعفات `price_unit`	
خارج `allowed_quantities`	

للعرض:
حين يكون سعر الوحدة كسراً صغيراً (مثل `"0.000607"`) اعرض `min_order_total_usd` مع أقل كمية — «6.07$ لكل 10,000» أوضح لعميلك من «0.00$».

15 منع ازدواج الطلبات

أهم قاعدة في الربط: الشبكة تنقطع، والخصم يجب ألّا يتكرر.

كل طلب يحمل `reference` فريداً من طرفك. إن أعدت إرسال الطلب **بنفس المرجع** — لانقطاع اتصال أو مهلة أو خطأ في نظامك — يُعاد إليك **الطلب الأول نفسه** ولا يُنشأ طلب ثانٍ ولا يُخصم مرّتين.

- استخدم مرجعاً ثابتاً لكل عملية شراء في متجرك (رقم طلب عميلك مثلاً)، لا رقماً عشوائياً عند كل محاولة.
- **عند انتهاء المهلة دون رد:** أعد الإرسال بنفس المرجع، أو استعلم `GET /orders/{reference}` — ستجد الطلب إن كان أُنشئ.
- المرجع خاصّ بحسابك: شريكٌ آخر يستخدم المرجع نفسه لا يتعارض معك.
- المرجع نفسه بمحتوى مختلف (منتج أو كمية أخرى) يعيد الطلب الأول أيضاً — لكل عملية شراء جديدة مرجع جديد.

16 الأخطاء

كل خطأ يعود بصيغة واحدة ورمز ثابت — ابنِ منطقك على `code` لا على نص الرسالة.

```
HTTP/1.1 402 Payment Required

{
  "code": "insufficient_balance",
  "message": "رصيد المحفظة غير كافٍ لإتمام العملية.",
  "details": {}
}
```

HTTP	code	السبب	الإجراء
401	`unauthenticated`	المفتاح مفقود أو غير صالح أو ملغى، أو سُحبت صلاحية الربط.	تحقّق من الترويسة والمفتاح.
402	`insufficient_balance`	الرصيد لا يكفي للطلب.	اشحن محفظتك. لا خصم حدث.
403	`forbidden`	العملية غير مسموحة لحسابك.	راجع شريحتك.
403	`account_suspended`	الحساب موقوف.	تواصل مع الدعم.
404	`not_found`	منتج أو قسم أو طلب غير موجود (أو ليس لك).	تحقّق من `slug` أو المرجع.
409	`product_unavailable`	المنتج غير متاح، أو الكمية تخالف القيود، أو السعر غير مضبوط.	اقرأ الرسالة وعدّل الطلب.
409	`out_of_stock`	نفد المخزون.	أعد المحاولة لاحقاً.
409	`conflict`	تعارض في حالة الطلب.	استعلم عن الطلب.
422	`validation_error`	حقل ناقص أو بصيغة خاطئة، أو `reference` مفقود.	صحّح المدخلات. لا خصم حدث.
429	`rate_limited`	تجاوز حدّ النداءات.	انتظر `Retry-After` ثانية.
503	`store_closed`	المتجر في صيانة مؤقتة.	أعد المحاولة بعد دقائق.
503	`orders_disabled`	استقبال الطلبات متوقف مؤقتاً.	أعد المحاولة لاحقاً.
5xx	`http_5xx`	خطأ غير متوقع أو انقطاع.	أعد المحاولة **بنفس** `reference`.

متى يحدث الخصم؟

فقط حين يعود 201 من `POST /orders`. كل أخطاء 4xx أعلاه تُرفض **قبل** الخصم. وعند 5xx أو انقطاع الاتصال لا تفترض شيئاً — استعلم بمرجعك.

17 أمثلة كاملة

دورة كاملة: تحقّق ← طلب ← متابعة حتى النتيجة، بمعالجة صحيحة للأخطاء وإعادة المحاولة.

Python

```python
import os, time, requests

API = "https://YOUR-DOMAIN/api/partner/v1"
HEADERS = {"X-API-Key": os.environ["AXIS_API_KEY"]}   # key from an environment variable

def call(method, path, **kw):
    for attempt in range(5):
        try:
            r = requests.request(method, API + path, headers=HEADERS, timeout=30, **kw)
        except requests.RequestException:
            time.sleep(2 ** attempt); continue          # network error: retry with the same input
        if r.status_code == 429:
            time.sleep(int(r.headers.get("Retry-After", "5"))); continue
        if r.status_code >= 500:
            time.sleep(2 ** attempt); continue
        data = r.json()
        if r.status_code >= 400:
            raise RuntimeError(f"{data['code']}: {data['message']}")
        return data
    raise RuntimeError("service unavailable")

def buy(product, player_id, shop_order_id, quantity=1):
    order = call("POST", "/orders", json={
        "product": product, "quantity": quantity,
        "reference": f"SHOP-{shop_order_id}",        # fixed per purchase
        "fields": {"player_id": player_id}})
    while order["status"] in ("paid", "processing"):
        time.sleep(15)
        order = call("GET", f"/orders/{order['reference']}")
    return order                                     # completed or refunded

result = buy("pubg-uc-60", "5123456789", 10001)
print(result["status"], result["delivery"])
```

PHP

```php
<?php
$api = 'https://YOUR-DOMAIN/api/partner/v1';
$key = getenv('AXIS_API_KEY');

function axis($method, $path, $body = null) {
    global $api, $key;
    $ch = curl_init($api . $path);
    curl_setopt_array($ch, [
        CURLOPT_CUSTOMREQUEST  => $method,
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_TIMEOUT        => 30,
        CURLOPT_HTTPHEADER     => ["X-API-Key: $key", 'Content-Type: application/json'],
        CURLOPT_POSTFIELDS     => $body ? json_encode($body, JSON_UNESCAPED_UNICODE) :
null,
    ]);
    $raw = curl_exec($ch);
    $status = curl_getinfo($ch, CURLINFO_HTTP_CODE);
    curl_close($ch);
    $data = json_decode($raw, true);
    if ($status >= 400) throw new Exception($data['code'] . ': ' . $data['message']);
    return $data;
}

$order = axis('POST', '/orders', [
    'product'   => 'pubg-uc-60',
    'reference' => 'SHOP-10001',
    'fields'    => ['player_id' => '5123456789'],
]);
while (in_array($order['status'], ['paid', 'processing'])) {
    sleep(15);
    $order = axis('GET', '/orders/' . $order['reference']);
}
echo $order['status'] . PHP_EOL . $order['delivery'];
```

Node.js

```
const API = 'https://YOUR-DOMAIN/api/partner/v1';
const headers = { 'X-API-Key': process.env.AXIS_API_KEY, 'Content-Type':
'application/json' };

async function axis(method, path, body) {
  const res = await fetch(API + path, { method, headers, body: body &&
JSON.stringify(body),
                                       signal: AbortSignal.timeout(30000) });
  const data = await res.json();
  if (!res.ok) throw new Error(`${data.code}: ${data.message}`);
  return data;
}

let order = await axis('POST', '/orders', {
  product: 'pubg-uc-60', reference: 'SHOP-10001', fields: { player_id: '5123456789' },
});
while (['paid', 'processing'].includes(order.status)) {
  await new Promise(r ⇒ setTimeout(r, 15000));
  order = await axis('GET', `/orders/${order.reference}`);
}
console.log(order.status, order.delivery);
```

18 أفضل الممارسات

ما يميّز ربطاً يعمل لسنوات عن ربطٍ يتعطّل عند أول انقطاع.

الموضوع	التوصية
المتابعة	تابِع الطلبات المعلّقة كل 10–30 ثانية، وتوقّف عند الحالة النهائية. لعدّة طلبات استخدم الاستعلام الدفعي (`?references=`) بدل نداء لكل طلب.
المهلة	مهلة 30 ثانية لكل نداء. عند انتهائها لا تفترض فشل الطلب — استعلم بمرجعك.
إعادة المحاولة	أعد فقط عند انقطاع الشبكة و5xx و429، بتأخير متزايد (2، 4، 8 ثوانٍ…)، وبنفس `reference` دائماً.
لا تُعِد أخطاء 4xx	401/402/404/409/422 لن تنجح بالتكرار — صحّح السبب أولاً.
الكتالوج	خزّنه لديك وحدّثه دورياً؛ لا تجلبه عند كل زيارة. أعد التحقق بـ `/quote` قبل الطلبات الكبيرة.

الموضوع	التوصية
الرصيد	راقب `/me` وضع حدّاً أدنى للتنبيه، كي لا تتوقف مبيعاتك برفض `402` .
الحقول	تحقّق في متجرك من الحقول المطلوبة وأنواعها قبل الإرسال — رسالة واضحة لعميلك أفضل من رفضٍ متأخر.
المبالغ	أنواع عشرية دقيقة دائماً. لا `float` في الحسابات المالية.
السجلات	سجّل `reference` و `partner_reference` لكل طلب — يختصران أي مراجعة مع الدعم. **لا تسجّل المفتاح.**
الحقول الجديدة	قد نضيف حقولاً للردود مستقبلاً. تجاهل ما لا تعرفه ولا تعتمد على ترتيب الحقول.

19 قائمة التحقق قبل الإطلاق

راجعها بنداً بنداً قبل فتح الربط لعملائك.

	البند
☐	المفتاح محفوظ على الخادم فقط (متغيّر بيئة)، ولا يظهر في أي كود متصفح أو تطبيق أو سجل.
☐	نداء /me يعيد حسابك وشريحتك الصحيحة.
☐	كل طلب يحمل reference فريداً وثابتاً لعملية الشراء، ويُحفظ في قاعدة بياناتك قبل الإرسال.
☐	الحقول المطلوبة وقيود الكمية تُتحقّق في متجرك قبل الطلب.
☐	المهلة وإعادة المحاولة تستخدمان نفس reference، ولا تُعاد أخطاء 4xx.
☐	المتابعة تتوقف عند completed refundedg rejectedg failedg.
☐	حالة refunded تُعيد المبلغ لعميلك في متجرك.
☐	التعامل مع 429 باحترام Retry-After.
☐	تنبيه عند انخفاض الرصيد.
☐	جرّبت طلباً حقيقياً صغيراً من البداية حتى التسليم.

20 الدعم وسجل التغييرات

نحن هنا حين تحتاجنا.

- **الدعم الفني:** من صفحة **«الدعم»** في حسابك على https://YOUR-DOMAIN — اذكر مرجع الطلب (`AX-ORD-…` أو مرجعك) وتوقيت النداء.
- **مفاتيحك:** https://YOUR-DOMAIN/api-access
- **هذه الوثيقة على الإنترنت:** https://YOUR-DOMAIN/developers

سجل التغييرات

الإصدار	التغييرات
1.1	الأقسام، تفاصيل المنتج بالحقول والخيارات وقيود الكمية، قائمة الطلبات والاستعلام الدفعي، لغة البيانات `lang` ، حدود الاستخدام في `/me` ، التنفيذ الآلي لطلبات الواجهة.

الإصدار	التغييرات
1.0	الحساب، المنتجات، التسعير، إنشاء الطلب ومتابعته.