

دليل ربط منصات SaaS العامة مع BeanzPay

هذا الدليل مخصص لأي منصة SaaS تريد إضافة BeanzPay كطبقة دفع وإدارة تاجر. صُمم ليُستخدم في بناء Connector مستقل من الخادم إلى الخادم (S2S)، مع رحلة موافقة مالك التاجر المطلوبة عند إصدار اعتماد مالي أو توسيع نطاقاته. لا يطلب الدليل من المنصة جمع بيانات البطاقة، ولا يضع اسم مزود الدفع في تجربة التاجر أو العميل النهائي.

تحديث الدليل: 21 أكتوبر 2026. مرجع خط أساس التشغيل المراجع: Partner API 28959793176ed4894b39ce1210b88afef60188d3. متطلبات هوية المالك في هذه المراجعة هي عقد GET /api/partner/live/integration. هذا 1.9.0؛ تحقق من نسخة الاكتشاف الفعلية قبل تطبيقها. سجل النشر النهائي يحدد الإصدار الذي تُنفذ عليه التحقق. هذا تحديث توثيقي للعقد، ولا يمثل اختبار قبول مالي مشتركاً أو تفعيل كل قدرة لكل حساب. القواعد الواردة هنا تصف خط الأساس المراجع في النسخة الحالية. قبل تفعيل أي ميزة إنتاجية، نفّذ أولاً GET /api/partner/live/integration استخدم بالمفتاح المخصص للشريك. استخدم GET /api/partner/integration لعقد Test فقط، ثم نفّذ GET /api/merchant/v1/features بعد إصدار اعتماد التاجر. اعتبر نتيجة الاكتشاف مصدر القرار النهائي للنسخ والبيئة والصلاحيات والقدرات والحدود.

هذه النسخة تشمل عقد المرشح لإصلاح رحلة العقد واسترجاع عمليات الأسرار. اكتشاف الشريك في المرشح 1.4.0 و Merchant API 1.5.0؛ يبقى إثبات النشر والقبول المالي سجلين منفصلين. لا يكفي ملف الدليل لإثبات توفر المسارات على الإنتاج.

1. حدود المسؤوليات

تحتفظ منصة SaaS بما يلي:

- معرفات الربط المحلية في قاعدة بياناتها، مع ربط كل سجل بـ platform_id, merchant_id, environment و beanzpay_id.
- المفتاح السري للشريك أو Bearer الخاص بالتاجر داخل Secret Manager فقط.
- نسخة الطلب الأصلي، و Idempotency-Key، و reference قبل كل عملية كتابة.
- حالة الطلب المحلية، وسجل محاولات الـ webhooks، وآخر نتيجة مؤكدة.

تتولى BeanzPay ما يلي:

- تثبيت حدود التاجر والبيئة والصلاحيات والـ MID داخليًا.
- إنشاء checkout مستضاف أو embedded وفق أصل الموقع المسجل.
- التحقق من التوقيع، وحفظ الحالة المالية، وإنشاء سجل refund فرعي مستقل.
- إدارة Webhooks، وإعادة المحاولة المقيدة، والـ reconciliation الموثق.
- استقبال طلبات التشغيل اليدوية مثل Apple Pay وتغيير وسيلة الدفع.

لا تنقل المنصة إلى متصفح التاجر أي Partner Secret أو Merchant Bearer أو سر Webhook. ولا تعرض للعميل اسم أو استجابة المزود upstream؛ استخدم رسائل مثل payment_pending, payment_confirmed, refund_requires_reconciliation.

2. البيئات والعناوين

استخدم متغيرات بيئة قابلة للتغيير بدل تضمين العناوين في الكود:

PARTNER_TEST_BASE_URL = https://<beanzpay-host>/api/partner
 PARTNER_LIVE_BASE_URL = https://<beanzpay-host>/api/partner/live
 MERCHANT_BASE_URL = https://<beanzpay-host>/api/merchant/v1

في النشر الحالي يكون مضيف BeanzPay هو المضيف المنشور، لكن ال Connector العام هو https://beanzpay.com؛ يجب أن يقرأ العنوان من إعداداته وأن يرفض redirects والعناوين خارج allowlist. تغيير العنوان لا يحول Test إلى Live ولا يغير الاعتماد.

السطح	الغرض	المصادقة	ملاحظة التفعيل
Partner Test	اكتشاف، onboarding إدارة بيانات الاختبار، وعمليات مالية Test إن ظهرت كقدرة	X-Partner-Key + X-Partner-Secret	الدفع Test مشروط بملف Test حقيقي ومفعّل؛ وجود المسار وحده لا يكفي
Partner production administration	onboarding الإنتاج، claim، التقارير والإدارة	نفس ترويسة الشريك	السطح الإداري الإنتاجي ليس سطح الدفع المالي الإنتاجي
Merchant Native Test	الدفع والتقارير وال webhooks بيئة التاجر Test	Bearer مربوط بتاجر وبيئة ونطاقات	يحتاج موافقة وإصدار اعتماد وملف Test حقيقي
Merchant Native Live	الدفع وال refund والاستعلامات المقيدة	Bearer Live مربوط بالتاجر وال MID	يحتاج capability/owner/provider gates مستقلة

لا تخط production الخاصة برحلة onboarding مع live الخاصة بالعمليات المالية. قد يملك التاجر ملفي Test و Live منفصلين.

3. المصادقة والملكية والصلاحيات

Partner S2S

أرسل الترويسات من خادم المنصة فقط:

X-Partner-Key: <partner-key>
 X-Partner-Secret: <partner-secret>
 Accept: application/json

الترويسات مطلوبة معًا. لا يكفي المفتاح العام وحده، ولا تُحفظ القيم في Postman collection أو frontend bundle أو سجلات الطلبات. اجعل كل بيئة لها Secret Store منفصلًا.

Merchant Native

Authorization: Bearer <merchant-bound-token>
 Accept: application/json

الاعتماد يثبت التاجر والبيئة والنطاقات؛ لذلك لا تضاف environment في body أو query لعمليات Native. لا تقبل المنصة معرف تاجر أو MID من العميل لاختيار حساب آخر. افحص GET /account ثم GET /features وسجل credential.scopes وبيانات الجاهزية دون تسجيل الاعتماد نفسه. يعيد /features أيضًا data.effective_scopes؛ لا تخلطها باسم scopes داخل كائن اعتماد /account.

النطاقات الحالية القابلة للإصدار هي:

account:read	payments:create	payments:read
payments:reconcile	payments:manage	refunds:create
refunds:read	refunds:reconcile	transactions:read
transfers:read	webhooks:read	webhooks:manage
invoices:read	invoices:manage	subscriptions:read
subscriptions:manage	card_tokens:read	payments:authcap

وجود اسم نطاق مثل payments:authcap في المفردات لا يثبت وجود مسار capture/void عام. اعتمد قائمة العمليات الفعلية في OpenAPI ونتيجة الجاهزية معًا.

اطلب أقل مجموعة لازمة. أي إضافة نطاق إلى تفويض موجود تحتاج موافقة صريحة جديدة؛ لا توسع grant قديمًا تلقائيًا.

لا يقدم اعتماد Partner الحالي مفاتيح فرعية أو نطاقات موظفين مقيدة بكل تاجر؛ scope_enforcement=not_supported_for_legacy_partner_credentials لا يساوي تفويضًا ماليًا. تطبق المنصة أدوار موظفيها وعزل المتاجر في خادمها، ولا توزع اعتماد الشركاء.

اقرأ GET /merchants/{merchant}/native-credentials لمعرفة scopes, authorization_id, active، ووقت الانتهاء دون السر، واقرأ GET /merchants/{merchant}/native-authorizations للتفويض، نسخته، موافقة المالك، انتهائه وسحبه. expires_at=null في التفويض يعني لا حد زمني له؛ يبقى قابلاً للسحب ومشروطًا بالملكية والجاهزية. اعتماد Native نفسه مدته الحالية سنة تقويمية (P1Y)، ويحد خمسة اعتمادات فعالة لكل بيئة وفق الاكتشاف. سحب أو انتهاء أو تدوير اعتماد Native المفوض، أو سحب تفويضه، يمنع استخدامه ويوقف الإشعارات المرتبطة به؛ لا ينتقل endpoint أو محاولاته تلقائيًا لاعتماد جديد. endpoints التي أنشأها المالك مباشرة لها إدارة مستقلة؛ افحصها وألغها صراحة عند إنهاء الربط. لا يفيد ذلك حذف السجل المالي أو إلغاء كل إشعارات المالك.

رحلة المالك

عندما تكون المنصة تريد ربط تاجر موجود أو إصدار اعتماد Native، تكون موافقة المالك والتوقيع وتحديد البيئة والنطاقات لازمة للإصدار المالي، وليست خطوة اختيارية يمكن تخطيها:

1. أنشئ السجل أو استخدم POST /merchants/claim مع claim token أصدره مالك التاجر من رحلة تحقق BeanzPay. claim token لمرة واحدة ومربوط بالشريك والبيئة.

2. اطلب من المالك الموافقة على التاجر، البيئة، والنطاقات المطلوبة في رحلة owner authorization. لا تضع secret في رابط الاستمرار.

3. اقرأ GET /merchants/{merchant}/native-authorizations قبل الإصدار.

4. أصدر الاعتماد عبر POST /merchants/{merchant}/native-credentials بعد تحقق owner/provider readiness. يظهر Bearer مرة واحدة فقط في قناة سرية.

5. عند الدوران استخدم POST .../{credential}/rotate؛ احفظ الاعتماد الجديد قبل إبطال القديم محليًا، ثم اختبره على route قراءة غير مالية. الاعتماد القديم يفشل بعد الدوران ولا توجد فترة سماح عامة.

6. عند الإيقاف استخدم DELETE، أو وقف ال Connector، وامسح القيمة من الذاكرة وال Secret Manager وفق سياسة الاحتفاظ لديك.

استرجاع عمليات إصدار الأسرار

يجب إرسال Idempotency-Key بطول 8-120 من ASCII المرئي دون مسافات عند POST POST /webhooks/{merchant}/native-credentials وعند الدوران، وكذلك POST /webhooks/{endpoint}/rotate. احفظ المفتاح والطلب المعياري والبيئة ومعرف التفويض قبل أول محاولة. هذه عمليات محلية في قاعدة BeanzPay؛ ليست طلب دفع أو استرداد لدى المزود.

أول استجابة ناجحة فقط تعيد token أو secret ومعها operation. احفظ السر فورًا في Secret Manager دون logs. مع بقاء صلاحية actor والتفويض وربط المورد، إعادة نفس العملية المكتملة تعيد secret_already_delivered 409 و errors.operation فقط؛ تغيير body الفعّال بعد validation أو نوع العملية أو المورد أو التفويض بالمفتاح نفسه يعيد idempotency_conflict 409. لا تعيد المنصة السر، ولا تنفذ mutation ثانية بهذه الإعادة.

بعد timeout أو انقطاع الرد، استعلم بالمفتاح الأصلي في الرأس:

```
GET /api/partner/live/merchants/{merchant}/native-credential-operations?environment=live
Idempotency-Key: credential-issue-synthetic-0001
```

```
GET {MERCHANT_BASE_URL}/webhook-operations
Idempotency-Key: webhook-create-synthetic-0001
```

الأول يحتاج اعتماد الشريك والتاجر والبيئة test|live الإلزامية. الثاني يحتاج Bearer نفسه أو اعتمادًا صالحًا ضمن التفويض نفسه ونطاق webhooks:read؛ البيئة من Bearer. لا ترسل مفتاح العملية في query. الرد مسطح {...}: {operation: {...}} وفيه kind, result_type, result_id, status=completed و secret_retrievable=false. لا يوجد TTL معلن للإيصال ولا pending/failed قابل للقراءة؛ الحجز وتغيير المورد والإكمال commit ذري واحد. الخطأ قبل commit يعكس المعاملة المحلية كاملة. لا تجعل record_not_found 404 إذرًا لإصدار بديل تلقائي.

إذا ثبت الإكمال وضاع السر، لا يمكن استعادته من الإيصال. ابدأ إجراء recovery مقصودًا: اعرف result_id، ثم دُور المورد المعروف بمفتاح جديد بعد اعتماد المشغل والتأكد من التفويض والعزل، واحفظ السر البديل. إذا تعذر إثبات النتيجة، صعد إلى عمليات BeanzPay. لا تنشئ مفتاحًا جديدًا في كل مزامنة. لا تطبق قاعدة إعادة الدفع أو الاسترداد على استجابة سر أحادي العرض. العينات في secret-operations-client.mjs لا تعيد POST تلقائيًا.

حدود صلاحية استرجاع الإيصال

ضمان العملية هو عدم إنشاء mutation ثانية وعدم كشف السر ثانية؛ ليس حق قراءة دائمًا بعد سحب التفويض. قد يرجع الطلب 401 أو 404 إذا زالت المصادقة أو التفويض أو المورد الذي يُفحص قبل ledger. استعلام العملية يحتاج صلاحية actor والبيئة الحاليين؛ سحب Native grant قد يمنح Bearer من الاستعلام، وعندها تتبع عمليات BeanzPay أو مسار الشريك المصرح نفسه دون توسيع الصلاحيات.

يُحسب request hash من الحقول الفعّالة التي اعتمدها Validator، مع ترتيب مفاتيح objects. ترتيب array مثل events يبقى مهمًا؛ الحقول المجهولة المتجاهلة لا تغيره. نقص أو فساد رأس Idempotency-Key يعيد 422 Laravel بصيغة message و errors؛ أما تعارض العملية وإكمالها و 404 فتستخدم type و message و errors المسطحة.

قراءة العقد واستمرار رحلة المالك

تجهيز العقد يخضع لاكتمال المراجعة ورسوم التأسيس وموافقة المزود وفق حالة الطلب. السداد وحده لا يعد عقدًا جاهزًا فورًا؛ تابع الحالة وحدث جاهزية العقد. بعد توقيع المالك يبقى تفويض Native والبيئة والنطاقات وجاهزية الدفع شروطًا مستقلة لإصدار الاعتماد.

اقرأ GET /applications/{application}/contract على Partner Test أو production. الرد مسطح ويحفظ application_id, merchant_id, environment و application_status، ثم contract=null قبل التجهيز أو metadata للعقد الحالي: id, version, status, signed_at, sha256, signed_verified, signature_method=email_otp, owner_auth_required=true و رابطي عرض وتنزيل المالك. القراءة لا تجهز عقدًا ولا تصدر إشعارًا. SHA يخص PDF غير الموقع قبل التوقيع ثم PDF الموقع بعده؛ signed_verified نتيجة تحقق سلامة فعلية، وليس alias لحالة التاجر أو علامة قديمة.

رابطا owner_view_url و owner_download_url يحتاجان جلسة المالك الصحيحة؛ لا يمثلان تنزيل PDF مجهولاً أو S2S، ولا تعيد هذه الواجهة snapshot أو بيانات المالك أو الرسوم. للحصول على رابط استمرار للطلب نفسه أرسل body:

POST {PARTNER_LIVE_BASE_URL}/applications/{application}/owner-continuation

تحفظ العملية application و merchant و UUID الاستمرار و return_url السابق. الرابط الصالح يحفظ expiry نفسه؛ المنتهي يجدد 48 ساعة. لا تعيد إرسال تسجيل التاجر لتجديد الرابط، ولا ترسل return_url أو external_reference أو merchant_id أو package_slug هنا؛ تُرفض هذه الحقول بـ 422. الرد يعيد owner_continuation_url و metadata و next_action كإرشاد: contract_signature, partner_authorization أو merchant_onboarding. الرابط يفرض OTP وموافقة المالك والبوابات الفعلية، ولا يعد redirect تلقائياً إلى المنصة. عدم تطابق الشريك أو البيئة أو الملكية يعيد 404.

استقبل contract.ready عند commit تجهيز PDF و contract.signed عند commit توقيع موثق. كل milestone محجوز بصورة دائمة مستقلة عن merchant.activated؛ لا تستخدم ACTIVE أو legacy flags لتوليد توقيع سابق أو backfill. تدعم القناة التكرار المحتمل بمعرف حدث ثابت؛ احفظ dedupe دائماً. عند فقد الإشعار اقرأ metadata العقد وراقب قناة Partner أو اطلب متابعة العمليات؛ القراءة لا تعيد إرسال الحدث. توقيع العقد لا يفعل المال ولا يغني عن التفويض والجاهزية. لا يعد التحديث إرسال أحداث بأثر رجعي لكل عقد موجود قبل النشر؛ استخدم قراءة العقد الحالي للاستعلام عن العقود السابقة والتعافي من إشعار مفقود.

4. الاكتشاف قبل كل إصدار

ابدأ دائماً باكتشاف الإنتاج الإداري، ثم استخدم عقد Test عندما يكون المطلوب اختباراً منفصلاً:

```
GET {PARTNER_LIVE_BASE_URL}/integration
GET {PARTNER_LIVE_BASE_URL}/packages
GET {PARTNER_LIVE_BASE_URL}/documents/required?
merchant_type=company&owners_count=1&signatory=owner&signatory_count=1
# Test only:
```

```
GET {PARTNER_TEST_BASE_URL}/integration
```

سجل نسخة العقد، environment aliases، النطاقات، أنواع الطلبات التشغيلية، حدود body، سياسة retry، ونطاقات ال SLA. لا تعتبر package catalog سعراً نهائياً؛ السعر التجاري quote يلتقطه BeanzPay عند قبول الطلب ولا يعاد تسعييره عند activation.

بعد ربط التاجر:

```
GET {MERCHANT_BASE_URL}/account
GET {MERCHANT_BASE_URL}/features
```

الحالة registered تعني أن route موجود. الحالة conditional تعني أن عليك انتظار profile أو owner consent أو capability. لا تعرض زرّاً أو وعدّاً تجارياً اعتماداً على وجود route فقط.

5. رحلة onboarding الأساسية S2S

المسار الإنتاجي الأساسي هو POST /api/partner/live/merchants من خادم المنصة. لا تجعل Hosted Connect هو المسار الافتراضي الآلي؛ يمكن عرضه كمسار بديل عندما يظهر في الاكتشاف، لكنه لا يستبدل مرجع الطلب أو موافقة المالك.

الحقول الإلزامية الحالية في طلب إنشاء merchant هي:

```
{
  "external_reference": "platform-merchant-00042",
  "package_slug": "<allowed-package-from-discovery>",
  "merchant_type": "company",
  "commercial_registration": "<registration-number-from-secure-form>",
  "business_name": "Example Merchant",
  "email": "owner@example.test",
  "mobile": "05XXXXXXX",
  "website": "https://merchant.example.invalid",
  "iban": "SA<22-digits>",
  "owner_name": "Verified Owner",
  "owner_national_id": "<owner-id-from-secure-form>"
}
```

المثال هيكلي؛ القيم بين <...> ليست بيانات صالحة للإرسال. للشركة يلزم commercial_registration، وللعامل الحر freelancer_doc_number وفق عقد النوع.

external_reference ثابت ومفهرس في قاعدة المنصة؛ external_id alias قديم ولا يُرسل مع قيمة مختلفة. لا تختار package_slug من قائمة محلية؛ استخدم حزمة مسموحة من GET /packages. أرسل settlement_schedule وحقوق النشاط المشروطة إذا طلبها الاكتشاف. أرسل retail_price_minor فقط إذا كان عقد الشريك يطلب السعر؛ لا ترسل commercial_pricing_quote أو أي quote محسوب من العميل. BeanzPay يحفظ snapshot تجاريًا موثوقًا من الخادم عند القبول، ولا يعيد تسعير طلب مقبول عند تفعيل التاجر.

تعود الاستجابة بسجل application وربما owner_continuation_url وبيانات ربط ثابتة مثل application_id، merchant_id، وexternal_references. احفظها قبل إظهار حالة النجاح. مسار الاستمرار opaque ومؤقت، ويستخدمه المالك الموثق لـ:

1. التحقق في بوابة BeanzPay عبر OTP/البريد أو المسار الذي يعيده الخادم؛ لا تقدمه على أنه تحقق حكومي لمجرد وجود اسم داخلي مختلف في metadata.
2. توقيع الشروط المعروضة في رحلة المالك.
3. الموافقة الصريحة على البيئة وقائمة النطاقات المطلوبة.

يمكن إرسال document_fields اختياريًا ضمن طلب الإنشاء: خريطة مفاتيحها أنواع المستندات وخصائصها كما تعيدها المتطلبات الديناميكية، مثل company_address وowner_id وowner_address_2. تُحفظ الحقوق المعروفة الجزئية، وتُفحص القيم المرسله حسب تعريفها. تُرفض أسماء المستندات والحقوق غير المعروفة والخانات التي تتجاوز عدد الملاك أو المفوضين، وإرسال المفتاح الأساسي و1_للخانة نفسها. مثال على جزء البيانات الأولية داخل جسم الإنشاء:

```
{"document_fields":{"company_address":{"city":"Riyadh","country":"SA"}}
```

هذا يحفظ جزءًا من العنوان ولا يعلن اكتماله؛ address العام لا يحل محل بيانات العنوان الوطني للمنشأة.

تبقى هذه البيانات جزءًا من طلب الإنشاء الأصلي؛ إعادة الطلب بالمرجع نفسه والبيانات نفسها تعيد الطلب القائم، ولا تستخدم تغييرها في إعادة الإنشاء لتعديل طلب سابق. عند رفع ملف للخانة نفسها تُحفظ البيانات الأولية عند غياب نسخة أحدث، ويمكن استكمال بيانات المستند غير المتعلق بالهوية في أول رفع. تتقدم بيانات المستند الأحدث على البيانات الأولية، ولا تُدمج هوية جديدة مع رقم هوية أو تاريخ ميلاد من نسخة قديمة. بيانات مستند يتطلب ملقًا لا تُعد استلامًا للملف، ولا تمثل موافقة المالك أو اعتماد المراجعة أو التفعيل المالي.

بعد ذلك اقرأ حالات وعمليات الرحلة من المسارات الديناميكية التي يعيدها العقد، ومنها GET /merchants/{merchant}/application، GET /merchants/{merchant}/environment، GET /merchants/{merchant}/capabilities، وواجهات /applications/{application} للوثائق والإرسال والتعديلات والتimeline والإلغاء. لا تخرع حالة من اسم الصفحة؛ خزن status وnext_actions كما يعيدها الخادم.

قراءة GET /applications/{application}/documents تشخيص محلي لا يغيّر الطلب. تبقى أعلام uploaded وall_required_uploads_complete وall_required_complete مؤشرات استلام ووجود فقط. يعيد كل مستند أيضًا

metadata_complete وmissing_fields (أسماء الحقول فقط) وmetadata_source وmetadata_receipt_mismatch. وبعيد الملخص all_required_metadata_complete وmissing_required_metadata حسب نوع/موضع المستند، وreview_ready وreadiness_scope وstale_required_document_receipts. لا تصبح review_ready=true إلا باكمال الملفات المطلوبة وصحة بيانات المستندات المطلوبة؛ المستندات الاختيارية لا تمنعها. هوية المالك owner_id تتطلب الآن صورة أمامية فعلية واحدة لكل مالك مع الحقول النصية. تبقى هوية المفوض signatory_id بلا ملف إذا أعادت المتطلبات requires_upload=false. بعد الإرسال يمنع owner_identity_files_available=false الجاهزية عند فقد ملف هوية مالك فعليًا ولو بقي uploaded=true تكون القيمة null قبل الإرسال أو دون تاجر.

قبل الإرسال يكون النطاق application_receipt_snapshot. بعد وجود submitted_at يكون النطاق:

application_receipts_with_matched_current_metadata

تظهر تصحيحات بيانات المراجعة الحالية فقط عند تطابق موضع المستند والملف أو إيصال الاستلام ومراجعته. تغيير مسار الملف يبطل الإيصال القديم ولو نُسخَت بيانات إيصال قديم إلى الملف الجديد. السجل الحالي المفقود أو المستبدل أو الملتبس يمنع الجاهزية ويظهر في قائمة الإيصالات القديمة. قد تكون missing_fields فارغة عند عدم التطابق لأن الحقول مكتملة لكن الإيصال لم يعد يمثل السجل الحالي. مصدر البيانات يميز لقطة الطلب عن السجل الحالي المطابق أو غير المطابق أو المفقود. لا تعيد القراءة رفع المستندات أو إرسال الطلب ولا تحوّل review_status=received إلى اعتماد. الجاهزية هنا اكتمال محلي للبيانات، ولا تعني موافقة المزود أو موافقة المالك أو تفعيل التاجر. قد يجعل تصحيح بيانات مفوض لا يتطلب ملفًا في السجل الحالي المطابق الطلب جاهزًا للمراجعة، بينما يبقى عداد الاستلام القديم جزئيًا. أما مالك owner_id فلا تكفي بيانات هويته وحدها بعد هذا التحديث: يجب وجود الصورة الأمامية المطلوبة أيضًا.

متطلبات المستندات ورفعها

للاكتشاف الأولي أرسل merchant_type=company أو freelancer؛ اسم حقل النوع في الاستجابة هو account_type، أما account_type=company في query فلا يحدد نوع الاكتشاف، وحذفه يترك الافتراضي freelancer. للطلب القائم اقرأ دائمًا GET /applications/{application}/documents؛ فهو المرجع لخانات هذا الطلب وعدد الملاك والمفوضين وrequires_upload وحقوله الفعلية.

ارفع الملف والبيانات معًا إلى POST /applications/{application}/documents. الحقول type وfile مطلوبة؛ fields مصفوفة Multipart بمفاتيح محاطة بأقواس، وليست نص JSON. يقبل الملف PDF أو JPEG أو PNG حتى 2048 KiB، مع فحص محتواه الفعلي؛ تغيير الامتداد لا يجعل الملف صالحًا. قيم حقول رفع Multipart نصوص بحد 255 حرفًا، مع قواعد الحقل الأدق. تختلف هذه الحدود عن حدود document_fields في طلب الإنشاء. دع CURL يحدد boundary تلقائيًا. Content-Type وboundary تلقائيًا.

أمثلة تركيب فقط ببيانات وهمية: صُدِّر المتغيرات من مخزن آمن، واستبدل المسارات بملفات مصرح بها وقيم البيانات الفعلية. لم تُنفذ هذه الأوامر على تاجر حقيقي. APPLICATION_ID وPARTNER_LIVE_BASE_URL=https://beanzpay.com/api/partner/live هما إعدادا خادم المنصة؛ الآيان أدناه مثال تنسيقي ولا يثبت ملكية حساب.

```
curl --fail-with-body --request POST \
  "$PARTNER_LIVE_BASE_URL/applications/$APPLICATION_ID/documents" \
  --header "X-Partner-Key: $PARTNER_KEY" \
  --header "X-Partner-Secret: $PARTNER_SECRET" \
  --header 'Accept: application/json' \
  --form 'type=bank_iban_certificate' \
  --form 'file=@/secure/example-bank.pdf;type=application/pdf' \
  --form 'fields[holder]=Example Merchant' \
  --form 'fields[iban]=SA1000000000000000000000'
```

```
curl --fail-with-body --request POST \
  "$PARTNER_LIVE_BASE_URL/applications/$APPLICATION_ID/documents" \
  --header "X-Partner-Key: $PARTNER_KEY" \
  --header "X-Partner-Secret: $PARTNER_SECRET" \
  --header 'Accept: application/json' \
  --form 'type=owner_address' \
  --form 'file=@/secure/example-address.pdf;type=application/pdf' \
  --form 'fields[name]=Example Owner' \
  --form 'fields[street]=Example Street' \
  --form 'fields[district]=Example District' \
  --form 'fields[building_number]=1234' \
  --form 'fields[postal_code]=12345' \
  --form 'fields[city]=Riyadh' \
  --form 'fields[country]=SA'
```

secondary_number اختياري في العنوان؛ استخدم اسم slot المكتشف مثل owner_address_2 عند تعدد الملاك. حقول شهادة الضريبة تشمل number، وحقول السجل التجاري تشمل number كما يعيدها المخطط. لا يستخرج رفع الصورة بياناتها تلقائيًا، ولا تترك النصوص فارغة اعتمادًا على وجود الملف.

لحفظ الحقول النصية لهوية المالك أو المفوض استخدم JSON دون ملف في طلب PATCH نفسه:

```
PATCH {PARTNER_LIVE_BASE_URL}/applications/{application}/documents/owner_id/fields
Content-Type: application/json
```

```
{"fields":{"id":"1234567890","mobile":"0501234567","date_of_birth":"1990-01-01"}}
```

هذه بيانات وهمية؛ الهوية والجوال والتاريخ مطلوبة مع عمر لا يقل عن 18 عامًا. PATCH يحفظ النصوص ولا يرفع صورة. من Partner API 1.9.0 تحتاج كل خانة owner_id إلى صورة أمامية فعلية واحدة ويظهر requires_upload=true. لا يوجد حقل صورة خلفية إضافي في عقد الشريك الحالي. عند تعدد الملاك استخدم الخانة المكتشفة مثل owner_id_2 ولا تَعد استخدام صورة مالك للمالك آخر.

```
curl --fail-with-body --silent --show-error \
  --url "$PARTNER_LIVE_BASE_URL/applications/$APPLICATION_ID/documents" \
  --header "X-Partner-Key: $PARTNER_KEY" \
  --header "X-Partner-Secret: $PARTNER_SECRET" \
  --header 'Accept: application/json' \
  --form-string 'type=owner_id' \
  --form 'file=@/secure/synthetic-owner-front.jpg;type=image/jpeg' \
  --form-string 'fields[id]=1234567890' \
  --form-string 'fields[mobile]=0501234567' \
  --form-string 'fields[date_of_birth]=1990-01-01'
```

هذا قالب بيانات وهمية، وليس إثبات رفع Live. حد ملف Partner هو 2048 KiB بصيغ PDF أو JPEG أو PNG مع فحص المحتوى. يضبط cURL حد Multipart؛ لا تضبطه يدويًا ولا ترسل fields كنص JSON. لا يستخرج رفع الصورة النصوص تلقائيًا.

هوية المفوض تبقى حقولًا فقط متى أعاد العقد requires_upload=false. لا يغيّر PATCH حقول العنوان أو البنك. الرفع و PATCH متاحان حسب حالة الطلب، ولا يتطلب الرفع موافقة Native مسبقًا. يوجد استثناء محدد لاستكمال صورة مالك مطلوبة ومفقودة في طلب مدفوع قائم: يلزم بقاء الطلب في المعالجة بعد السداد، وربه بالتاجر والشريك الصحيحين، ووجود حساب المزود ومرجع السداد الأصلي، وأن تكون حالة التاجر ضمن نافذة الإصلاح المسموحة. يسمح الاستثناء برفع الصورة المفرد أو الدفعي وب PATCH لنصوص الخانة المفقودة فقط؛ لا يفتح تعديل بقية المستندات أو حذفها. يجب اكتمال النصوص قبل تخزين الصورة، ويمكن إبقاء الحقول الحالية الكاملة عند عدم إرسال حقول أحدث. بوابة مستندات المالك تتيح الاستكمال أيضًا.

بعد اكتمال كل الملفات والبيانات المطلوبة تعود المستندات إلى مراجعة BeanzPay المحلية تلقائيًا ويعيد الرفع application_status=under_review عند الانتقال؛ لا تَعد submit. يحدث ذلك دون إعادة التسجيل أو السداد أو تغيير معرفات الربط. لا يرسل الرفع وحده الطلب إلى المزود، ولا يلغي اعتماد المستندات الأخرى؛ تراجع الإدارة الصورة

الجديدة قبل استكمال الربط. تبقى بقية تعديلات الطلب المدفوع مقيدة بحالتها. إذا بقيت صورة مالك آخر أو بيانات مطلوبة ناقصة، فلا يعني استلام ملف واحد اكتمال الانتقال للمراجعة.

أعد قراءة GET /applications/{application}/documents وحالة الطلب بعد الرفع. نجاح الرفع هو إيصال استلام، ولا يعني اعتماد الصورة أو الموافقة المالية. إذا أعادت الحالة 400 أو الحقول 422، تابع الطلب نفسه ولا تنشئ طلبًا بديلًا.

تجديد استكمال المالك والشروط

رابط المالك صالح حاليًا 48 ساعة؛ انتهاءه يعيد 410 Gone. لتجديده أرسل POST /applications/{application}/owner-continuation بلا body باستخدام معرف الطلب الحالي. يتجدد المنتهي فقط مع نفس UUID ومعرفي الطلب والتاجر وreturn_url المحفوظ؛ الصالح لا يمدد. لا تعد إنشاء التاجر ولا ترسل بياناته أو return_url في طلب التجديد. تابع العقد الحالي وحالة الطلب عبر القراءة وPartner lifecycle؛ إعادة التوجيه التلقائي إلى المنصة ليست وعدًا في هذا العقد.

شروط التاجر الرسمية: <https://beanzpay.com/terms>، والنسخة الثابتة الحالية <https://beanzpay.com/terms/1.0> المنشورة والنافذة في 1 سبتمبر 2026. احتفظ بالنسخة الثابتة المعروضة عند الموافقة. تحفظ BeanzPay نسخة العقد وبصمته ودليل توقيع email OTP؛ لا يوجد في طلب إنشاء التاجر حقل عام terms_accepted يحل محله. مسار portal/nafeetha الحالي اسم داخلي لتوقيع email OTP، وليس نفاذًا حكوميًا. شروط التاجر، وتفويض Native، والاتفاق التجاري للشريك ثلاث موافقات مختلفة؛ لا تستخدم مسودة اتفاق شريك بدل شروط التاجر.

Claim لتاجر موجود

عند ربط حساب قائم، يصدر مالك الحساب claim token أحادي الاستخدام من رحلة المالك، ثم يرسل الخادم:

```
POST {PARTNER_LIVE_BASE_URL}/merchants/claim
X-Partner-Key: <partner-key>
X-Partner-Secret: <partner-secret>
Idempotency-Key: claim-merchant-00042
Content-Type: application/json
```

{"claim_token": "<owner-issued-token>", "external_reference": "platform-merchant-00042"}
الclaim مربوط بالشريك والبيئة ولا ينشئ merchant جديدًا أو اعتمادًا أو رسوًا أو عمولة بأثر رجعي. إذا ضاعت الاستجابة، أعد القراءة أو أعد نفس المفتاح والجسم؛ لا تنشئ claim ثانيًا.

6. مسار بناء ال Connector

المسار الأساسي S2S

هذا هو المسار الموصى به لمنصة SaaS:

1. خزن platform_merchant_id واطلب onboarding أو claim.
2. انتظر readiness وeffective capability.
3. أنشئ payment من backend المنصة باستخدام Bearer الصحيح.
4. أرسل العميل إلى checkout_url أو استخدم embed_url عند قبول embedded origin. لا تجمع raw card data.
5. استقبل Native webhook، تحقق من التوقيع والهوية، ثم اقرأ payment محليًا.
6. نفذ refund ك child operation واحفظ حالة الأب والابن.

مسار المالك/التاجر الموجود

يصلح عندما يكون التاجر قد بدأ حسابه خارج المنصة. استخدم claim بال external reference الثابت، ولا تنشئ سجلًا جديدًا لمجرد أن الاستجابة ضاعت. ال claim يحفظ التاريخ ولا ينشئ setup fee أو عمولة بأثر رجعي.

7. الدفع وال checkout

إنشاء Payment

POST /payments يحتاج Idempotency-Key من ASCII المرنئي (0x21-0x7e، بلا مسافات) بطول 8-120. الجسم الأساسي:

```
{
  "amount": 12500,
  "currency": "SAR",
  "reference": "order-2026-00042",
  "customer": {
    "first_name": "Customer",
    "last_name": "Example",
    "email": "customer@example.test",
    "phone_number": "+966500000000"
  },
  "expires_in": 1800
}
```

amount وحدة صغرى صحيحة وليست decimal. لا تحول المبلغ مرتين ولا تعتمد على قيمة قادمة من browser. reference ثابت لكل order؛ لا تنشئ reference جديدًا بعد timeout. إذا أضفت checkout ف origin يطابق أصل موقع التاجر المسجل حرفيًا، و HTTPS return_url على نفس الأصل ومسار بلا query أو fragment أو userinfo أو locale هو ar أو en. منفذ مخصص.

ال response يعطي BeanzPay UUID محليًا، checkout_url وربما embed_url، الحالة، العملة، المبلغ، المرجع، وقت الانتهاء، و metadata آمنة. لا يوجد provider ID أو card secret في response.

فقدان الاستجابة

- 201/200 مع سجل معروف: احفظ payment.id و reference.
- payment_outcome_unknown 409 أو payment_outcome_unknown 503: لا ترسل POST جديدًا. أعد نفس body مع نفس Idempotency-Key أو استخدم query بالمرجع إن كانت الحالة تسمح.
- payment_rejected 422: لم يُثبت إنشاء الدفع. صحح السبب ثم استخدم key جديدًا لمحاولة جديدة، ولا تعيد استخدام reference لطلب مختلف.
- GET /payments و GET /payments/{payment} يقرآن آخر حالة محلية مؤكدة؛ لا يفترضان inquiry فورًا.

8. refund: كامل، جزئي، مرجع الابن، والحالة المجهولة

استخدم:

POST /payments/{payment}/refund
Idempotency-Key: refund-key-2026-0001

ل full refund احذف amount ليحل الخادم remaining refundable amount. للجزئي أرسل integer أصغر من أو يساوي المتبقي. يمكن إرسال reason و refund_reference reference هو alias قديم لنفس child reference. إذا أرسلنا معًا يجب أن يتطابقا وإلا يكون الرد 422 قبل أي provider submission. نفس ال alias ينتجان نفس fingerprint، ولا يلغيان child operation قديمًا.

ال child response يحتوي amount, status, cumulative_refunded_amount, remaining_refundable_amount, fees, reference, payment_reference. لا تعدّ العملية ناجحة إلا عند status=refunded. الحالات الأخرى تعني:

الحالة	معناها	تصرف ال Connector
submitting	حجز محلي قبل/أثناء التقديم	اقرأ نفس السجل ولا تنشئ child آخر
pending_callback	تُفذ الطلب وينتظر دليلاً موثقاً	انتظر webhook أو inquiry المسموح
requires_reconciliation	لا يوجد دليل نهائي بعد	استخدم reconcile المخصص أو إجراء المشغل
refunded	تأكيد نهائي للعملية	حدّث الرصيد بال child amount
failed	رفض نهائي محفوظ	لا تعيد نفس العملية بـ key جديد إلا بعد إصلاح السبب

إذا أصبح outcome unknown:

1. استخدم GET /refunds?idempotency_key=<same-key> لاسترجاع child.

2. أو استخدم GET /refunds?refund_reference=<child-ref> إن كان محفوظاً.

3. استخدم payment_reference للبحث عن الأب، مع الانتباه أن query alias reference القديم في GET refunds يعني مرجع الأب لا child.

4. POST /payments/{payment}/reconcile لا يثبت نتيجة child refund وحده. استخدم POST

refunds/{refund}/reconcile بصلاحيّة refunds:reconcile وجسم JSON فارغ {} للتحقق من الابن نفسه؛ هذا استعلام لا يعيد إرسال الاسترداد.

5. لا تعيد POST بمفتاح جديد، ولا تعتبر timeout نجاحاً أو فشلاً.

تجري المطابقة الآلية كل خمس دقائق بحد 50 عملية في الدورة، للعمليات المربوطة بمعرف مزود معروف وعند جاهزية Live/inquiry؛ لا تشمل كل نتيجة مجهولة بلا ربط. اقرأ provider_operation_bound, reconcile checked_at, last_provider_verified_at, last_provider_verification_source وفي رد requires_operator_review, support_reference, next_action غياب الربط أو الدليل النهائي يستلزم عمليات BeanzPay بنفس معرف الابن، دون استرداد بديل. المثال الكامل للإنشاء والقراءة والبحث والنتيجة غير المحسومة موجود في examples/refund-recovery-fixture.json داخل الحزمة.

الخادم يحجز المبلغ تزامنياً ويمنع تجاوز إجمالي المبلغ القابل للاسترداد؛ لا تحسب المنصة المتبقي من قراءة قديمة ثم تفترض قبول استردادين متزامنين. رسوم platform_recorded_minor قيود محلية، وprovider_actual_confirmed=false لا يثبت الرسوم الفعلية. مدد الاسترداد ورسومه تعتمد على حساب التاجر وطريقته وعقده؛ لا تعرض مدة أو صفر رسوم لمجرد قبول المخطط للمبلغ.

9. Webhooks والتوقيع وإعادة المحاولة

Native financial webhook contract 3.0.0

قبل قبول HTTP 2xx:

1. اقرأ raw request bytes دون إعادة serialization.

2. تحقق من X-BeanzPay-Timestamp وأن محاولة التوقيع ضمن نافذة 300 ثانية.
3. احسب (HMAC-SHA256(header_timestamp + "." + raw_body, endpoint_secret)).
4. تحقق أن X-BeanzPay-Webhook-ID = X-BeanzPay-Event-ID = body.event_id
5. تحقق أن body.timestamp = body.occurred_at حرفيًا؛ timestamp الخاص بالمحاولة قد يختلف عن وقت الحدث.
6. طابق refund child و merchant, environment, reference, amount, currency مع سجلك المحلي. في partial refund يكون data.amount مبلغ الابن، لا مبلغ الأب.
7. خزن event_id قبل الرد 2xx. التكرار يعاد إلى نتيجة المعالجة نفسها.

الرؤوس اللازمة هي X-BeanzPay-Signature, X-BeanzPay-Timestamp, X-BeanzPay-Event, X-BeanzPay-Environment, Webhook-ID, X-BeanzPay-Event-ID, X-BeanzPay-Contract-Version

Native endpoint management

- GET /webhooks: metadata أو body.secret فقط، بلا secret أو body.
 - POST /webhooks: endpoint_url HTTPS عام و events من payment.paid, payment.failed, payment.refunded, payment.voided يعاد secret مرة واحدة.
 - POST /webhooks/{endpoint}/rotate: secret جديد، والقديم يفشل فورًا.
 - DELETE /webhooks/{endpoint}: إلغاء الربط.
 - GET /webhooks/deliveries: metadata آمنة، و GET .../{delivery} لمحاولات الإرسال دون raw body أو secret.
 - POST .../{delivery}/replay: سبب نصي، key جديد خاص بال replay، حد أقصى ثلاث محاولات يدوية مع cooldown 30 ثانية. الإرسال الآلي خمس محاولات بمهلة 30 ثانية و backoff baseline: 60, 300, 900, 3600 ثانية.
- يتاح endpoint نشط واحد لكل تاجر وبيئة. POST /webhooks بمفتاح جديد حتى للعنوان نفسه يستبدل السجل والسجل السابقين، فلا تشغله في كل مزامنة. لا توجد مدة انتهاء مستقلة معلنة للسجل؛ يبقى صالحًا حتى الاستبدال أو التدوير أو الإلغاء، بلا مهلة سماح للقديم. المحاولات المعلقة لا تنتقل تلقائيًا للربط الجديد. احفظ سجل الدفعات غير المحسومة وأعد قراءتها، ثم استخدم replay المفيد عند صلاحية الربط. سجلات Native المالية مصنفة financial_evidence ومحمية من الحذف العام؛ لا توجد مدة أيام ثابتة مفروضة عند إنشائها في هذا الخط الأساسي. لا تطبق سياسة تنظيف سجلات Partner عليها. الأمثلة الموقعة الاصطناعية لجميع الأحداث الأربعة تولد وتتحقق محليًا في native-webhook-verifier.test.mjs، وليست إيصالات مالية.

Partner lifecycle webhook

استخدم GET /webhook/events و GET /webhook/deliveries و GET /webhook على Partner. الواجهة HTTPS عامة ولا تُقبل redirects أو عناوين private/reserved. لا تعرض هذه raw payload أو signature أو response body؛ replay إجراء مشغل وليس route عامًا. ويمكن استعمال POST /payments/verify-signature للتحقق من payload/timestamp/signature وفق العقد.

قناة lifecycle_v1 تتعلق بدورة التاجر والكتالوج؛ لا تستخدمها لتأكيد مال العميل. وجود payment/refund في enum تاريخي لا يجعله حدث الحساب الحالي؛ اقرأ GET /webhook/events؛ تتضمن قناة المرشح contract.ready و contract.signed المستقلين عن التفعيل، وعيناهما الموقعة الخام في partner-contract-events-fixture.json. أحداث application.* لرسم التأسيس مستقلة عن Native.

10. Native management والاشتراكات

إدارة الحساب والـ features

GET /account و GET /features للقراءة. لا يعيدان provider credentials أو integration IDs أو card tokens. اجعل واجهة منصتك تعرض readiness, scope والبيئة فقط.

Plans و enrollment

الـ handles المحلية UUID: plan, enrollment, subscription, card, operation إنشاء الخطة وتحديثها وإيقافها واستئنافها يحتاج subscriptions:manage القراءة تحتاج subscriptions:read.

إنشاء enrollment هو first-3DS flow:

POST /subscription-plans/{plan}/enrollments

Idempotency-Key: subscription-enroll-2026-0001

يُحفظ الخادم الطلب، ويعيد checkout محليًا. بعد اكتمال 3DS لا تعتمد على callback وحده؛ استخدم GET /subscription-enrollments/{enrollment} أو reconcile الفارغ بعد أن يثبت inquiry الموثق المبلغ والعملة والبيئة، ويربط exactly one subscription whose initial_transaction matches caller provider ID. لا يقبل ID.

GET /subscription-operations?idempotency_key=<same-key> recovery محلي بعد فقدان الاستجابة. الحالة unknown ليست نجاحًا، ولا ترسل POST بديلًا.

تحتاج الاشتراكات ملف التاجر والبيئة ومفتاح الربط المطلوب وتفعيل MOTO و 3DS ودليل الجاهزية. لا يعني وجود routes إتاحتها لكل حساب. دعم callback lifecycle الكامل ليس مكتملاً في هذا الخط الأساسي؛ يعتمد تأكيد الالتحاق على inquiry الموثق المملوك. هذه اشتراكات تحصيل من عميل التاجر، وليست اشتراك المزود للتاجر أو API دفع عام بأي card token.

cards و subscription إدارة

• GET/PUT /subscriptions/{subscription} للقراءة والتحديث.

• POST /subscriptions/{subscription}/{action} حيث action = suspend, resume أو cancel.

• GET .../transactions يعيد تاريخًا آمنًا.

• GET .../cards يحتاج subscriptions:read و card_tokens:read ويعيد UUID محليًا و metadata masked فقط.

• POST .../cards/{card}/{action} حيث action = delete أو primary، ولا يتعامل Connector مع raw token أو provider card ID.

11. الفواتير الخاصة بالعميل

POST /customer-invoices ينشئ مستند دفع branded مربوطًا بـ Native Hosted intention. الحقول الأساسية customer; items, amount, currency و اختياري لكن مجموع line items يجب أن يساوي amount تمامًا. المستند ليس tax invoice ولا provider fee invoice.

استخدم:

• GET /customer-invoices مع status أو idempotency_key.

• GET /customer-invoices/{invoice} و .../document.

• POST .../send مرة واحدة مع business contact الموجود في حساب التاجر.

• POST .../cancel فقط للفواتير غير المدفوعة محليًا.

حالة payment_unknown لا تعني paid. حفظ key قبل create/send/cancel آتمًا بلا إرسال بريد أو payment POST ثانٍ.

إرسال الفاتورة يتم من مرسل BeanzPay المعتمد؛ contact التاجر يظهر في النص ولا يضمن Reply-To. أرسل {} إلى .../send مع المفتاح المحفوظ. الإلغاء يوقف checkout محليًا فقط، ولا ينفذ refund أو void، والدفع المؤكد المتأخر يتقدم على الإلغاء المحلي. الفاتورة الضريبية يصدرها نظام التاجر. يمكن مشاركة رابط السداد أو QR دون إرسال مفاتيح مالية إلى المتصفح.

12. العمليات اليدوية وApple Pay

ينشئ Partner request سجلًا محليًا وmetadata outbox فقط:

```
{
  "request_type": "apple_pay_activation",
  "service_key": "apple_pay",
  "action": "activate",
  "reason": "Activate the registered merchant domain",
  "details": {"domain": "shop.example.com"}
}
```

الأنواع الحالية: apple_pay_activation, payment_method_change, provider_account_lifecycle, capability_activation, fee_change, support_ticket
استخدم Idempotency-Key بطول 8-120؛ نفس body/key يعيد نفس الطلب، وتغيير body مع نفس key يعيد 409.

المسار الكامل: POST /merchants/{merchant}/operational-requests في Partner Live؛ صلاحية الشريك الحالية هي الحساب المصادق عليه وعلاقة التاجر، وليست نطاقات action. Apple Pay لـ Native هو activate أو verify_installation وdetails.domain مطلوب. مثال التغيير: payment_method_change مع service_key من قاموس الطرق action=enable|disable أو مثال حساب المزود: provider_account_lifecycle, action=disable|reactivate service_key=account.

Apple Pay لـ

1. ترفع المنصة الطلب للتاجر والنطاق والبيئة، وتحفظ معرفه.

2. تراجع عمليات BeanzPay الطلب وتتواصل يدويًا مع المزود، وتسجل نتيجة مؤكدة؛ لا يرسل إنشاء الطلب بريدًا إلى المزود تلقائيًا.

3. تقرأ المنصة .../{request}/apple-pay/artifact/download وتنزل .../{request}/apple-pay/artifact بالصادقة، وتطابق SHA-256.

4. تثبت الملف تلقائيًا على نطاق التاجر المعني في .well-known/apple-developer-merchantid-domain association عبر HTTPS عام دون تسجيل دخول أو redirect، ثم ترسل POST .../{request}/apple-pay/installation بالhash المطلوب ومفتاح مستقل.

5. تقرأ provider_state وeffective_state والجاهزية لكل حساب ونطاق. تثبيت الملف أو تطابقه لا يثبت موافقة المزود ولا يفعّل جميع التجار دفعة واحدة.

الهدف التشغيلي هو إنهاء التنفيذ بالكامل خلال ثلاثة أيام عمل، وفق تقويم الرياض (الأحد-الخميس، مع استبعاد العطلات المضبوطة). في التنفيذ الحالي يبدأ الحساب من requested_at عند إنشاء السجل، ويحفظ تقويم الطلب due_at؛ لا تنتظر المنصة قبولًا لاحقًا لبدء ساعة جديدة. اعرض الموعد العائد من الخادم وoverdue، وصعد التأخر دون اعتماد تلقائي. هذا هدف عمليات BeanzPay وليس ضمانًا زمنيًا من المزود. احفظ sla.configured, business_days, calendar, due_at, overdue, late_finished, escalation_required provider_sla_guarantee=false ما لم يقرر عقد موثق خلافه.

تأكيد المزود وحده لا يجعل قدرة بلا adapter متاحة ماليًا؛ قد تظل `api_effective=false` لتوقيف إنشاء مدفوعات جديدة محليًا فورًا توجد قراءة `controls/payment-creation` و `POST` إلى `.../suspend` أو `.../resume` مع السبب. هذا منفصل عن طلب تعطيل حساب المزود ذي المهلة التشغيلية، ولا يلغي تلقائيًا روابط قديمة أو اشتراكات أو دفعات قائمة. التذاكر عبر `{request}/ticket` و `...` وردودها مسار متابعة محفوظ، وليست تنفيذًا ماليًا بديلًا.

13. التسعير التجاري والكتالوج

كل منصة لها عقد تجاري منفصل. لا تنسخ نسبة أو `waiver` من منصة أخرى، ولا تضع أي نسبة ثابتة أو سياسة خصم خاصة بشريك بعينه في ال `Connector`.

القواعد العامة:

تفاصيل سياسة الحصة من الأساس وإعفاء تحمل الخصم ودفتر الحصص أدناه مشروطة بأن يعلنها عقد الشريك واكتشافه. بعض الحسابات تعمل بعقد تجاري سابق؛ لا تحولها لهذه السياسة أو تعرض لها خيار الإعفاء دون تفعيل. صريح من BeanzPay

- الكتالوج المركزي يعرض الباقات والسعر الحالي وال `quote metadata`.
 - `quote` الذي يوافق عليه التاجر `immutable`؛ تحديث `catalog` لا يعيد تسعير طلب قديم.
 - أي `discount` أو `subsidy` أو `waiver` يجب أن يأتي من `quote` صادر من الخادم مع `basis, revision, effective` و `audit metadata`، `time`.
 - لا يتجاوز سعر البيع سعر الباقية الأساسي المسموح؛ الحصة وأساسها قبل الضريبة والإعفاءات تقرأ من سياسة الشريك وال `quote`، لا من نسبة ثابتة في الكود.
 - عند سياسة حصة BeanzPay الثابتة من الأساس يكون صافي الشريك سعر البيع ناقص الحصة، وقد يصبح سالبًا. عند إعفاء تحمل الخصم المعتمد تكون نسبة الشريك من سعر البيع الفعلي؛ مثال تعليمي بنسبة 25% وسعر بيع 50 يعطي 12.50. المثال لا يمنح الإعفاء أو النسبة لأي حساب؛ تحقق من السياسة الفعلية في ال `quote`.
 - استرجاع `setup fee` الإداري ليس `refund` لدفعة عميل ولا `route` يمكن للشريك تشغيله. عند رد رسوم التأسيس كاملة وتأكيده، تعكس الحصص فلا تبقى حصة لأي طرف؛ لا تعكسها لنتيجة مجهولة أو لمجرد طلب استرداد أو تعطيل حساب.
 - عندما تكون سياسة دفتر الحصص مفعلة، يصدر كشفها في اليوم السابع الميلادي؛ هذا موعد النظام الحالي وليس خيار يوم قابلاً للتغيير من الموصل. تسجيل `paid` توثيق للمطابقة ولا ينفذ تحويلًا بنكيًا. الاعتراض والتصحيح وإثبات السداد وفق الاتفاق التشغيلي والتجاري، بلا سحب آلي من أموال التجار.
 - حساب المنصة الذاتي كتاجر مستقل عن حساب إدارة الشريك. الإعفاء من التأسيس أو الاشتراك واستبعاده من دفتر الحصص يتطلب إعدادًا معتمدًا، ولا يعفي رسوم المعاملات والاسترداد تلقائيًا.
 - `statement` التجاري يوضح `credit/debit` و `base/tax/retail/discount/benefit` حسب العقد، لكنه ليس رصيد بنك أو `settlement` أو `payout` أو `tax invoice`.
- إذا احتاجت منصتك سعرًا خاصًا، اطلب عقدًا تجاريًا جديدًا ولا تستنتج السعر من `GET /packages` وحده.

حدود التكافؤ مع أي بوابة أخرى

المتطلب	ما يتيح العقد الحالي
قائمة/تفاصيل/QR/إلغاء رابط الدفع	متاحة في Native مع النطاق والجاهزية؛ الإلغاء محلي والدفع المتأخر يعالج ماليًا
عمليات مالية وفلاتر وتصدير metadata و metadata	اقرأ payments / و transactions / ومسارات التصدير و metadata في OpenAPI؛ ليست فلاتر Partner Test دليلًا على Native Live
رصيد متاح أو معلق وتسوية بنكية رسمية	غير متاحة بالعقد الحالي؛ تقارير المعاملات والكشف التجاري لا تعوضها
إنشاء payout أو إدارة البنوك أو تحويل جماعي	غير متاحة؛ transfers / قراءة فقط ولا يثبت الرصيد المتاح
capture/void عام أو دفع عام بالتوكن	غير متاحة كعمليات عامة؛ لا تستبدلها بـ refund
تحويل بين التجار أو اشتراك المزود للتاجر	خارج العقد الحالي ويحتاج منتجًا وعقدًا مستقلين
كوبونات وخصومات دفعات العملاء	منفصلة عن خصومات رسوم الباقات وغير متاحة ضمنها

أي بديل يدوي للرصيد أو التسوية يحتاج نموذج تقرير ومصدرًا ودورية ومعرفات ربط واعتماد الطرفين؛ لا تعد به بلا إجراء قائم. تحفظ CAPABILITY_MATRIX.json حالة كل قدرة، والبيئة والصلاحيات والقيود. وتحفظ IMPROVEMENT_ROADMAP.md ما يحتاج تطويرًا ومسؤول تنفيذه واختبار إغلاقه دون اختلاق موعد التزام.

14. الأخطاء والحدود والـ retries

مثال خطأ عمليات Native المالية:

```
{
  "error": {"code": "payment_outcome_unknown", "message": "Safe generic message"},
  "errors": ["field_name"]
}
```

errors أسماء حقول فقط؛ لا تعكس القيم المدخلة. احفظ X-Request-ID للربط في الدعم. المعاني العملية:

HTTP	الاستخدام
401	credential مفقود أو غير صالح
403	scope/owner/capability غير كافٍ
404	resource خارج tenant/environment أو غير موجود
409	conflict أو unknown outcome أو replay مختلف
413	body أكبر من الحد
422	validation أو definitive rejection؛ أصلح الطلب قبل retry
429	rate limit؛ احترم سياسة الرد ولا تغير idempotency key
503	unavailable أو inquiry/outcome unknown؛ اقرأ error.code

حدود baseline المهمة: Idempotency-Key بين 8 و120 printable ASCII؛ payment body حتى 64 KiB؛ refund body حتى 16 KiB؛ pagination Native؛ operational requests حتى 50؛ replay اليدوي ثلاث مرات، والأكلي خمس مرات. راجع discovery لأن الحدود قابلة للتكوين.

لا تستخدم retry عام لكل 5xx. القرار يكون:

1. تحقق من أن العملية لم تكن definitive rejection.
2. إن كانت write unknown، أعد نفس body/key أو اقرأ resource.
3. إن كانت rate limit، انتظر server policy.
4. إن كانت manual/unsupported، أوقف العملية وافتح operational request.

وبالنسبة إلى أي POST مالي، فإن انقطاع النقل أو timeout أو 408 أو 429 أو أي 5xx، وكذلك رد 2xx ناقص أو غير مطابق للمخطط، يبقى unknown ما لم تثبت قراءة موثوقة النتيجة النهائية. احفظ نفس المفتاح وbody الأصليين، واسترجع بنفس المفتاح أو بقراءة مقيدة بالنطاق؛ لا تنشئ key أو reference أو عملية بديلة تلقائياً. يسمح بمحاولة جديدة فقط بعد إثبات أن رد 4xx الموثق رفض الطلب نهائياً.

15. سجل الحالة والقبول

لكل عملية يجب أن تسجل المنصة:

```
connector_request_id
beanzpay_request_id / resource_id
merchant_id + environment
idempotency_key_hash (log لا القيمة في)
body_fingerprint
reference / refund_reference
created_at, last_checked_at
status, error.code, X-Request-ID
webhook event_id and verification result
```

قائمة قبول قبل الإنتاج:

- [] discovery محفوظ ويطابق الإصدار والـ scopes والـ limits.
- [] Test و Live secrets منفصلان وموصولان بالتاجر الصحيح.
- [] كل POST يحفظ key/body قبل الإرسال.
- [] في العمليات المالية duplicate same key/body يعيد نفس المورد؛ conflict body يعطي 409.
- [] timeout لا ينشئ POST ثانياً.
- [] full/partial refund يحفظ parent/child و cumulative/remaining.
- [] unknown refund يملك recovery بالـ key وبـ refund_reference.
- [] webhook يثبت raw bytes و HMAC والهوية والوقت و dedupe قبل 2xx.
- [] rotation/revocation يوقف old secret والدفع غير المصرح.
- [] لا provider ID أو card token أو secret يعبر frontend أو logs.
- [] Apple Pay request لا يدعي التسجيل التلقائي.
- [] الأسعار والـ waiver تأتي من quote/contract، بلا hardcode.
- [] real merchant Test UAT موثق منفصلاً؛ لا يكفي نجاح الاختبارات المحلية.

16. الدعم والتشغيل

عند فتح تذكرة دعم أرسل connector_request_id, X-Request-ID, resource UUID, environment, version و error.code فقط. لا ترسل Authorization, Partner Secret, Webhook secret, raw payload, card data أو ملف البيئة.

للحالات المالية المجهولة اذكر هل لديك payment.id أو refund.id أو المرجع أو الـ idempotency key hash، ثم استخدم query/read-only recovery قبل أي محاولة جديدة. للحالات التشغيلية اذكر operational_request.id، request_type، الحالة، ووقت آخر تغيير. لا تحاول تغيير capability أو إنشاء اعتماد بديل من واجهة العميل.

17. فجوات يجب حسمها مع BeanzPay قبل البناء النهائي

- حساب Test حقيقي مستقل لكل منصة و merchant، مع callback/HMAC قابل للاختبار.
- أسماء وقنوات تسليم secrets ودورة rotation خارج API.
- SLA المتفق عليه تجارياً؛ ثلاثة أيام عمل هو target baseline لإكمال الطلبات التشغيلية من requested_at ووفق due_at وتقوم الطلب، وليس ضمناً للمزود.
- مصفوفة العملات والضرائب إن تجاوزت المنصة العملة الحالية.
- عقد رسمي إن احتاجت المنصة balance أو settlement أو payout.
- سياسة الاحتفاظ بسجلات webhook و invoice، ومكان الإقامة القانوني للبيانات.
- قائمة events الإضافية إن كانت المنصة تحتاج domain events غير الأربع الحالية.
- لا تُغلق هذه النقاط بافتراضات من واجهة أو من package catalog؛ اطلب تحديثاً موثقاً للـ discovery/OpenAPI والعقد التجاري.

حدود بيانات المستندات الأولية: 128 خانة إجمالاً، و 20 خانة اختيارية متكررة لكل نوع، و 2000 حرف لكل قيمة مع تطبيق أي حد أدق خاص بالحقل. لا تتجاوز خانات المالك أو المفوض العدد المعلن.

18. إغلاق الربط وتسليم الدعم

تفصل خطة القبول بين مراجعة الوثائق، وبناء الموصل، ونشره، وتفعيل الحساب، والقبول المالي المشترك. لكل اختبار احفظ الإصدار والبيئة والحساب ومرجع العملية والنتيجة والجهة التي نفذته؛ لا تنسب اختبار BeanzPay المحلي للطرفين.

المسؤولون	الدليل المطلوب	اختبار القبول
المنصة وBeanzPay	نفس IDs ومرجع ثابت، نوع صحيح ومتطلبات متطابقة	تسجيل شركة/عامل حر وإعادة الطلب وتجديد رابط المالك
المنصة وBeanzPay	غياب صورة المالك يمنع الجاهزية؛ نصوص المفوض تكفي إذا لا يتطلب ملقًا	صورة مالك وحقوقها ومفوض بلا ملف
المنصة وBeanzPay	رفض وصول دون تسريب أو تغيير سجل آخر	موظف/متجر/تاجر/بيئة خاطئة
المالك والمنصة وBeanzPay	أقل scopes، سر مرة واحدة، القديم مرفوض	إصدار/تدوير/سحب اعتماد وموافقة المالك
المنصة وBeanzPay والحساب المفعّل	إثبات مالي ومراجع ثابتة ومنع التكرار	دفع ناجح وفاشل ومتأخر وفقد استجابة
المنصة وBeanzPay	تحقق raw-body وربط event ودفتر dedupe	إشعار صحيح/محرف/مكرر/قديم و replay
المنصة وBeanzPay	child دائم، إجمالي ومتبقي ومصدر تأكيد، لا POST بديل	استرداد كامل وجزئي ومتزامن ومجهول
المنصة وعمليات BeanzPay	طلب artifact، due_at، دليل effective واختبار الحساب	Apple Pay وطرق الدفع وتعطيل المزود
المنصة	قراءة واسترداد القديم على بوابته وحسابه وبيئته الأصلية	تغيير بوابة التاجر
المنصة وBeanzPay	inquiry مملوك، بريد/إلغاء/دفع متأخر بلا ضريبة مزعومة	اشتراك وفاتورة سداد عند إدراجهما بالنطاق

قبل الاختبار يتفق الطرفان على نافذة التنفيذ وقناة التصعيد ومسؤولي التقنية والعمليات ومدد المتابعة. عند نتيجة مالية غير محسومة يوقف الموصل إنشاء بديل للعملية ويحفظها للتتبع؛ لا يمسحها ولا يعلن نجاحًا. يستأنف بعد قراءة أو مطابقة موثوقة، أو قرار عمليات موثق عند تعذر الحسم. إعادة الخدمة لا توسع الصلاحيات.

لمشكلة مستند أرسل: وقت المحاولة ومنطقته، application UUID، X-Request-ID، نوع/slot المستند، HTTP، وأسماء الحقول المرفوضة وحجم الملف و MIME فقط. فَرِّق بين القيم المخزنة وسجل تاريخي لم يُحفظ؛ لا تنسب سبب 422 قديمًا للتخمين. لا ترسل محتوى المستند أو بيانات الهوية أو البنك أو المفاتيح. قراءة الطلب وتشخيصه لا تتطلب إنشاء تسجيل جديد أو تغيير موافقات المالك أو تفعيلًا ماليًا.