ePlaza

GUÍA ESENCIAL PARA COMERCIOS E INTEGRADORES

Cobra desde cualquier banco de Venezuela

Cómo funciona ePlaza, cómo cobrar desde el portal y cómo integrarlo en tu sistema: enlaces de pago, webhooks y el paso a producción.

eplaza-qa-docs.pagaconquickpay.com

Versión del 19 de noviembre de 2026

Contenido

PARTE 1 Entender ePlaza

Qué es ePlaza 3

¿Cuál solución necesito? 4

Cómo funciona un cobro 4

Estados de un cobro 5

Comisión y liquidación 7

PARTE 2 Cobrar sin programar

Cobrar: enlace, QR y WhatsApp 9

PARTE 3 Integrar desde tu sistema

Cómo se integra 11

Tu primer cobro en 5 minutos 12

Llaves y entornos 13

Enlaces de pago 15

Webhooks 25

Errores y rechazos 31

Pasar a producción 35

Cada título del índice es un enlace. Lo que no está en esta guía (suscripciones, cobro por lote, bots, propinas, glosario de campos y más) está en eplaza-qa-docs.pagaconquickpay.com; la lista va al final.

PARTE 1

Entender ePlaza

Qué es, cómo llega el dinero y qué significa cada estado de un cobro.

Qué es ePlaza y cómo llega el dinero

Con ePlaza cobras en bolívares a cualquier persona con cuenta en un banco venezolano. Tu cliente abre un enlace, elige su banco, escribe su teléfono y su cédula, y confirma con la **clave de 8 dígitos que le envía su banco** (por SMS, notificación o correo). No se registra ni instala nada.

Lo que pasa	Quién lo hace
Creas el cobro (un enlace, un QR o un botón)	Tú, desde el portal o desde tu sistema
Abre el enlace y paga con la clave de su banco	Tu cliente
Confirma el pago con el banco	ePlaza
Te avisa (en el portal y, si integraste, por webhook)	ePlaza
Te transfiere el dinero a tu cuenta	ePlaza

Un pago ocurre en dos momentos

1. **El pago.** El banco confirma el débito y el cobro queda **pagado**. Desde tu sistema te llega el webhook `link.pagado`.
2. **La acreditación.** ePlaza te transfiere el monto, menos la comisión, a tu cuenta. Con la liquidación inmediata (activándose por etapas) te llega además el webhook `link.acreditado` con el desglose.

> **Entrega cuando el cobro esté pagado, no antes.** Que el cliente vuelva a tu web o te mande una captura no prueba nada: la prueba es el estado `pagado` (webhook o consulta).

Dos formas de usarlo

- **Desde el portal:** sin programar. Creas cobros, los mandas por WhatsApp o correo, haces cobranza masiva y suscripciones. → El portal de comercios
- **Desde tu sistema:** lo integras con la API en tu web, tu app o tu bot. → Cómo se integra

¿No sabes cuál? → ¿Cuál solución necesito?

¿Cuál solución necesito?

Elegir mal no rompe nada, pero trabajas de más. Ubícate en la primera tabla que te describa.

Si no programas

Si...	Usa...	Dónde
Cobras un monto puntual a una persona	Un cobro con monto (enlace + QR, o por WhatsApp)	Cobrar
Quieres que te paguen el monto que ellos escriban	Tu página de pago `/p/tu-usuario`	Página de pago
Cobras a muchos clientes a la vez (condominio, colegio, cartera)	Cobranza masiva desde un Excel	Clientes y cobranza
Cobras lo mismo cada mes, semana o año	Suscripciones	Suscripciones
Tienes una web y quieres un botón "Pagar"	El botón de tu portal (copiar y pegar)	Botón para tu web

Si programas

Si...	Usa...	Guía
Tu web, app o bot crea un cobro por pedido	Un enlace de pago por pedido + webhook	Enlaces de pago
Generas cobros en bloque desde tu sistema	Cobro por lote (hasta 500 por llamada)	Cobro por lote
Tu bot de WhatsApp cobra	Enlace en un mensaje con botón + webhook	Cobrar desde tu bot
Quieres pedir la clave del banco desde tu propia pantalla	API directa (sólo aliados)	API directa

> **Lo más simple casi siempre gana.** Si dudas entre el portal y la API, empieza por el portal: puedes pasar a la API después sin cambiar nada de tu cuenta.

Cómo funciona un cobro

Todo en ePlaza es un **cobro**: un monto, un concepto y una fecha de vencimiento. Cada cobro tiene un **enlace** (`https://eplaza-qa-pay.pagaconquickpay.com/l/…`) donde tu cliente paga.

De principio a fin

1. **Creas el cobro.** Desde el portal o con `POST /switch/v1/links`. Te devolvemos su `id` y su `url`.
2. **Se lo haces llegar.** Por WhatsApp, correo, QR, un botón en tu web o tu bot.
3. **Tu cliente paga.** Elige su banco, escribe su teléfono y su cédula, recibe la clave de 8 dígitos de su banco y la escribe.
4. **Se confirma con el banco.** El cobro pasa a `pagado`. Un cobro se paga **una sola vez**: nadie puede pagarlo dos veces.
5. **Te enteras.** En el portal (Transacciones y el aviso "Recibiste...") y, si integraste, con el webhook `link.pagado`.
6. **Te acreditamos.** El monto, menos la comisión, a tu cuenta. Con la liquidación inmediata (activándose por etapas) llega también `link.acreditado`.

Lo que conviene saber

- **El monto lo fijas tú.** El cliente no lo puede cambiar (salvo en tu página de pago, donde él lo escribe).
- **Vence.** Por omisión a las 24 horas; tú eliges entre 1 hora y 30 días. Un cobro vencido ya no se puede pagar.
- **Prueba y producción no se mezclan.** Un cobro creado con tu llave de prueba nunca mueve dinero, aunque lo abra alguien de verdad. → Modo prueba
- **Tus datos viajan con el cobro.** Lo que pongas en `metadata` (tu número de pedido, por ejemplo) vuelve en las consultas.

Siguiente: Estados de un cobro

Estados de un cobro

Un cobro siempre está en uno de estos estados. Los consultas con `GET /switch/v1/links/:id` (campo `estado`).

Estado	Qué significa	¿Entregas?
`pendiente`	Creado, nadie lo ha pagado	No
`cobrando`	Un cliente está pagando en este momento	Todavía no
`procesando`	El banco recibió el débito y todavía no confirma que el dinero se movió	Todavía no: espera
`revision`	Lo estamos verificando con el banco	Todavía no: espera
`pagado`	El banco confirmó el pago: el dinero es tuyo	**Sí**
`expirado`	Venció sin pagarse	No
`cancelado`	Lo cancelaste antes de que lo pagaran	No

Cómo se mueve

```
pendiente → cobrando → pagado
                    ↘ procesando / revision → pagado
pendiente → expirado | cancelado
```

- Según el banco de tu cliente, la confirmación puede llegar en el acto (el cobro pasa directo a `pagado`) o el cobro pasa por `procesando` hasta que el banco confirma (suele tardar menos de un minuto, pero a veces bastante más) y entonces queda `pagado` y te llega `link.pagado`.
- Si el banco **rechaza** el pago (clave equivocada, sin saldo), el cobro vuelve a `pendiente` y el cliente puede intentarlo otra vez con una clave nueva. También puede volver a `pendiente` desde `procesando`, si el banco termina rechazándolo.
- A veces el banco cancela el débito después de recibirlo, por ejemplo porque se agotó su tiempo de espera. Cuando lo confirmamos (el aviso del banco, su consulta y nuestro estado de cuenta coinciden en que el dinero no entró), el cobro sale de `procesando` y vuelve a `pendiente` (o queda `expirado`, si ya venció), y tu cliente puede pagar otra vez. Esa confirmación puede llegar desde los 30 minutos, cuando nuestro estado de cuenta ya muestra movimientos posteriores al pago; si alguna señal falta, puede tardar varias horas.
- `procesando` y `revision` **no son rechazos**: el pago puede terminar aprobado. Nunca le cobres de nuevo a un cliente con un cobro en esos estados.
- Un cobro `pendiente` pasa a `expirado` al llegar su `expires_at` (por omisión, 24 horas después de creado). Desde ese momento la consulta, el listado (también el filtro `estado`), los lotes y tu portal lo muestran `expirado`, la página de pago ya no deja pagarlo y cancelarlo responde `409`. Vencer no envía webhook.
- Un cobro que ya estaba en curso (`cobrando`, `procesando`, `revision`) cuando llegó su `expires_at` no vence por eso: sigue hasta `pagado`. Si el banco termina rechazándolo, queda `expirado`.

- `pagado` y `cancelado` son finales: no cambian más. `expirado` también, salvo un caso: un pago con tarjeta que Credicard confirma después del vencimiento pasa el cobro a `pagado` y te llega `link.pagado`.

> **Una regla:** entrega sólo con `pagado`. Todo lo demás es "todavía no".

Pagos (dinero que sale)

Un pago (`operacion: "pago"` en los avisos `transaccion.*`, por ejemplo la transferencia del neto a tu cuenta o un reintegro) también pasa a `LIQUIDADA`. **En un pago a otro banco, `LIQUIDADA` puede volverse `RECHAZADA`** si el banco receptor rechaza la transferencia, por ejemplo porque la cuenta no está a nombre de la cédula o el RIF indicados. En ese caso llega `transaccion.rechazada` con el mismo `data.id`: el dinero no llegó al beneficiario y no se vuelve a pagar solo. Si es la transferencia de tu dinero, revisa en Ajustes › Dónde recibes tu dinero que tu cuenta esté a tu nombre.

Comisión y liquidación

Cuánto cuesta

Cada pago recibido paga una comisión, que se descuenta del pago: tu cliente paga el monto exacto del cobro y a ti te llega el monto menos la comisión. Tu tarifa te la confirmamos al abrir tu cuenta.

Cuándo llega el dinero

La **liquidación inmediata** te transfiere el neto de cada pago apenas el banco lo confirma. La estamos **activando por etapas**: hoy se aplica a pagos pequeños mientras terminamos las pruebas con dinero real; te avisaremos cuando cubra todos tus cobros. Por cada pago liquidado te llega el webhook `link.acreditado` con el desglose:

Campo	Qué es
`monto_bruto`	Lo que pagó tu cliente
`comision`	La comisión descontada
`monto_acreditado`	Lo que llegó a tu cuenta
`liquidacion_txn_id`	El id de la transferencia

El neto irá a tu cuenta principal. Si registraste varias cuentas, al cobrar **desde el portal** puedes elegir en cuál cae cada cobro; por la API va siempre a la principal.

Las cuentas deben estar a tu nombre. A nombre de tu cédula si te registraste como persona, o del RIF de tu empresa si te registraste como negocio. Así lo verificamos con un pago de Bs 1 al registrarte.

PARTE 2

Cobrar sin programar

Enlaces, QR y WhatsApp desde el portal de comercios.

Cobrar: enlace, QR y WhatsApp

En **Cobrar › Cobro con monto**:

1. Escribe el monto en **bolívares (Bs), dólares (US$) o euros (€)**; el selector arranca en **bolívares** y puedes cambiarlo en cada cobro. En dólares o euros se convierte a bolívares a la tasa BCV del momento y debajo del monto ves la equivalencia (por ejemplo, «≈ Bs 9.806,15 a tasa BCV del 09/10»): tu cliente siempre paga en bolívares. El dólar usa la tasa BCV de ePlaza; el euro, la tasa oficial del euro, con máximo 36 horas de antigüedad. Si en ese momento no tenemos una tasa del euro confiable, la opción **€** aparece deshabilitada («Tasa del euro no disponible ahora») y no se puede cobrar en euros hasta que haya.
2. Si quieres, agrega un concepto (lo ve tu cliente) o elige productos de tu catálogo.
3. Elige cuándo vence: 1 hora, 24 horas, 3 días, 7 días o 30 días.
4. Si tienes sucursales, elige la sucursal. Si tienes más de una cuenta, en **¿Dónde recibes este pago?** ves la cuenta (banco y teléfono, con parte del número oculto): viene la de la sucursal, o la principal, y la puedes cambiar. Si la cambias, la recordamos para esa sucursal en ese navegador.
5. Pulsa **Crear cobro**.

Te damos el enlace y su **QR** (puedes descargarlo para imprimirlo). Compártelo como quieras.

Por WhatsApp desde ePlaza

Con **Cobrar por WhatsApp**, le llega a tu cliente un mensaje de ePlaza con tu nombre, el monto y un botón **Pagar** que abre el pago dentro de WhatsApp, sin salir de la conversación.

- Pides el nombre y el WhatsApp sólo después de crear el cobro.
- Las personas a las que ya les cobraste aparecen en **Cobrar de nuevo**: tocas a la persona, pones el monto y el envío sale ya lleno.
- El botón **Cobrar por WhatsApp** se enciende cuando el monto llega al **mínimo del día**: US$ 8 en bolívares, a la tasa BCV y redondeado hacia arriba a Bs 500. Lo ves en **Ajustes › Cobros por WhatsApp**.
- Debajo del mínimo aparece **Enviar igual por WhatsApp** si te quedan envíos de prueba (5 en los últimos 30 días) o tienes el plan WhatsApp sin mínimo con cupo. Si no, envía el enlace por correo o compártelo tú. → Cobros por WhatsApp y tu plan

Dividir una cuenta

En **Cobrar › Dividir cuenta** repartes un total entre varias personas: cada una recibe su propio cobro y en Transacciones los ves juntos. El total lo puedes poner en **Bs, US$ o €**, igual que en un cobro con monto: cada parte se convierte a bolívares a la tasa BCV del momento y ves lo que paga cada persona. Debajo del total aparece su equivalente en bolívares («Total ≈ Bs …») y, junto a cada persona, el de su parte.

Después

En **Transacciones** ves cada cobro con su estado y lo filtras por fecha, estado, monto y más (Filtrar tus transacciones). Mientras esté pendiente puedes reenviarlo o cancelarlo.

PARTE 3

Integrar desde tu sistema

Llaves, enlaces de pago, webhooks, errores y el paso a producción.

Cómo se integra

Integrar ePlaza son dos decisiones.

1. ¿Dónde paga tu cliente?

Opción	Cómo	Para
En la página de pago de ePlaza (recomendado)	Creas un enlace y llevas al cliente a su `url`	Webs, apps, bots, correos: casi todo
En tu propia pantalla	Pides tú la clave del banco con la API directa	Sólo aliados con contrato → API directa

Con la página de ePlaza no manejas datos bancarios ni la clave del banco: sólo un `id` y una `url`.

2. ¿Cómo te enteras del pago?

Opción	Cómo	Cuándo
Webhook (recomendado)	Te enviamos `link.pagado` a tu servidor, firmado	Siempre que puedas recibir un POST
Consulta	`GET /switch/v1/links/:id` hasta ver `pagado`	Como respaldo, o si no puedes recibir webhooks

El camino completo

```
Tu servidor ──POST /links──▶ ePlaza      (te devuelve id y url)
Tu cliente  ──abre la url──▶ página de pago (paga con la clave de su banco)
ePlaza    ──link.pagado──▶ tu servidor    (entregas el pedido)
```

> **La llave va sólo en tu servidor.** Nunca en el navegador, en una app móvil ni en un bot del lado del cliente: con ella cualquiera crearía cobros a tu nombre.

Empieza aquí: Tu primer cobro en 5 minutos

Tu primer cobro en 5 minutos

Cuatro pasos con tu **llave de prueba**: no se mueve dinero real.

1. Saca tu llave de prueba

En tu portal ve a **Integraciones › Llaves de API › Crear llave**, elige **Prueba** y copia el valor. Tiene esta forma:

```
tu-comercio:sk_test_…
```

Guárdala en una variable:

```bash
export QP_KEY="tu-comercio:sk_test_…"
```

2. Crea un enlace de pago

```bash
curl -s https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/links \
  -H "x-switch-key: $QP_KEY" \
  -H "content-type: application/json" \
  -d '{ "monto": 150, "concepto": "Pedido #1024", "metadata": { "pedido": "1024" } }'
```

Te devuelve, entre otros campos:

```json
{
  "id": "3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40",
  "url": "https://eplaza-qa-pay.pagaconquickpay.com/l/3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40",
  "estado": "pendiente",
  "mode": "test"
}
```

3. Págalo

Abre la `url`. Como es un cobro de prueba, la página te lo indica y acepta **cualquier clave de 8 dígitos** (para forzar un rechazo o un pago en proceso: Modo prueba): elige un banco, escribe un teléfono y una cédula de ejemplo (`04141234567`, `V12345678`) y paga.

4. Confirma que está pagado

```bash
curl -s https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/links/3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40 \
  -H "x-switch-key: $QP_KEY"
```

`"estado": "pagado"`. Listo: ese es el ciclo completo.

Y ahora, el porqué

- Qué significa cada estado → Estados de un cobro
- Enterarte sin consultar → Webhooks
- Todos los campos del enlace → Enlaces de pago
- Cuando quieras cobrar de verdad → Pasar a producción

Llaves y entornos

Toda llamada lleva la cabecera:

```
x-switch-key: <tu-comercio>:<llave>
```

Dos tipos de llave

Llave	Empieza con	Qué hace
Prueba	`sk_test_`	Simula todo de punta a punta. **Nunca mueve dinero**
Producción	`sk_live_`	Cobra de verdad

El modo lo decide la llave, no tu código. No hay ningún campo `test=true`: la misma petición con `sk_test_` es de prueba y con `sk_live_` es real. Un cobro creado en prueba queda de prueba para

siempre, aunque lo abra alguien de verdad.

Cada llave sólo ve y toca lo de su modo. Con `sk_test_` no se consulta, lista, edita ni cancela nada creado con `sk_live_` (enlaces, lotes, órdenes, suscripciones, productos, propinas y equipo): responde `404` como si no existiera. Y una `idempotency_key` usada en prueba no se cruza con producción.

Hay dos cosas del comercio que no tienen modo y por eso exigen `sk_live_`: los webhooks (`POST`, `PATCH` y `DELETE /switch/v1/webhooks`, porque reciben también los avisos de producción) y la configuración de tus Flows de WhatsApp (`POST /switch/v1/flow/config`, `PUT /switch/v1/flows/definicion`). Con `sk_test_` responden `403` con `{ "error": "requiere_llave_live" }`; listar y ver webhooks sí funciona con cualquier llave. Si aún no tienes llave de producción, registra tu URL desde el portal (Integraciones › Webhooks).

Dónde se sacan

En tu portal, **Integraciones › Llaves de API**:

- La de prueba la creas cuando quieras.
- La de producción se activa cuando tu cuenta está aprobada.
- La llave se muestra **una sola vez**. Si la pierdes, revócala y crea otra.
- Puedes tener varias (una por sistema) y revocar cada una por separado: lo que use una llave revocada recibe `401` de inmediato.
- Por API (`POST /switch/v1/api-keys` y `POST /switch/v1/api-keys/:id/revoke`), **una llave de prueba sólo crea y revoca llaves de prueba**. Para crear o revocar una de producción, llama con una `sk_live_`; con una `sk_test_` responde `403` con `{ "error": "requiere_llave_live" }`.
- El `label` de una llave nueva (1 a 120 caracteres) es libre, salvo que no puede empezar por `service-key:` ni `partner:` (son de uso interno): responde `400` con `{ "error": "label_reservado" }`.

Dónde se guardan

> **Sólo en tu servidor**, en una variable de entorno o un gestor de secretos. Nunca en el código del navegador, en una app móvil ni en un repositorio.

URL base

	URL
API	`https://eplaza-qa-pay.pagaconquickpay.com/switch/v1`
Página de pago	`https://eplaza-qa-pay.pagaconquickpay.com/l/<id>`

Es la misma para prueba y producción: cambia sólo la llave.

Enlaces de pago

Un enlace de pago es un cobro con su propia dirección (`https://eplaza-qa-pay.pagaconquickpay.com/l/…`). Tu cliente lo abre, elige su banco y autoriza con la clave de 8 dígitos que le manda su banco (por SMS, notificación o correo). La página la aloja ePlaza: tú no manejas datos bancarios.

Sirve para cobrar por WhatsApp, correo, SMS o QR: creas el enlace, lo envías y te enteras cuando lo pagan.

1. Crear un enlace

```http
POST https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/links
x-switch-key: tu-comercio:sk_test_…
Content-Type: application/json

{
  "monto": 450.00,
  "concepto": "Servicio de consultoría",
  "expira_horas": 72,
  "success_url": "https://tu-dominio.com/gracias?pedido=100",
  "metadata": { "cliente_id": "CLI-001", "factura": "FAC-2026-100" },
  "idempotency_key": "factura-FAC-2026-100"
}
```

Respuesta `201`:

```json
{
  "id": "3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40",
  "url": "https://eplaza-qa-pay.pagaconquickpay.com/l/3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40",
  "wa_deeplink": null,
  "qr_png": null,
  "monto": 450,
  "moneda": "VES",
  "concepto": "Servicio de consultoría",
  "estado": "pendiente",
  "expires_at": "2026-10-11T14:00:00.000Z",
  "success_url": "https://tu-dominio.com/gracias?pedido=100",
  "failure_url": null,
  "metadata": { "cliente_id": "CLI-001", "factura": "FAC-2026-100" },
  "mode": "test"
}
```

Guarda el `id` junto a tu pedido: con él consultas el cobro y reconoces su webhook. Para mandarle el cobro a tu cliente usa `url`.

Campos que puedes enviar

Campo	Tipo	Obligatorio	Notas
`monto`	número	Sí	Mayor que cero, en bolívares.
`concepto`	texto	Sí	Lo que ve tu cliente al pagar. Hasta 140 caracteres.
`moneda`	texto	No	`VES` (la única hoy).
`expira_horas`	número entero	No	De 1 a 720 (30 días). Por omisión, 24.
`expira_en`	fecha ISO 8601	No	Vencimiento exacto; si lo envías, manda sobre `expira_horas`. Tiene que ser futuro.
`success_url`	texto	No	A dónde vuelve tu cliente al pagar. Empieza con `https://`, hasta 400 caracteres. → Volver a tu web
`failure_url`	texto	No	A dónde vuelve tu cliente si el pago no puede continuar. Empieza con `https://`, hasta 400 caracteres. → Volver a tu web
`metadata`	objeto	No	Tus datos: hasta 20 claves (nombre ≤ 40, valor ≤ 200 caracteres). Te los devuelve la respuesta de crear; **no viajan en el webhook**, así que guarda la relación `id` ↔ pedido en tu sistema.
`idempotency_key`	texto	No	Hasta 64 caracteres. Ver abajo.

> Si el banco rechaza, tu cliente **no** sale de la página de pago: puede intentarlo otra vez con una clave nueva. Sólo vuelve a tu `failure_url` si el cobro ya no se puede pagar (venció o lo cancelaste) o si él elige **Cancelar y volver**.

2. No duplicar cobros: `idempotency_key`

Va **en el cuerpo** de la petición. Si repites la creación con la misma `idempotency_key` (por ejemplo, porque se cortó la red y no sabes si llegó), recibes el **mismo** enlace en vez de uno nuevo:

- Primera vez: `201` y el enlace nuevo.
- Repetición: `200`, el mismo enlace y `"idempotent_replay": true`.

La llave vale **dentro del modo**: la misma `idempotency_key` con tu llave de prueba y con la de producción son dos enlaces distintos (uno `test` y otro `live`); nunca recibes con `sk_live_` un enlace creado con `sk_test_`. Si recibes `409` con `"error": "idempotency_key_en_uso"`, esa llave ya la usaste en el otro modo: usa otra.

Usa algo que identifique tu pedido (`factura-FAC-2026-100`), no un valor al azar por intento.

3. Compartir el enlace

Mándale `url` a tu cliente. En WhatsApp, lo mejor es un mensaje con botón (`cta_url` en la API de WhatsApp Cloud): abre la página de pago con un toque.

```json
{
  "messaging_product": "whatsapp",
  "to": "584141234567",
  "type": "interactive",
  "interactive": {
    "type": "cta_url",
    "body": { "text": "Tienes un pago de Bs 450 pendiente. Puedes pagarlo desde cualquier banco." },
    "action": {
      "name": "cta_url",
      "parameters": { "display_text": "Pagar ahora", "url": "https://eplaza-qa-pay.pagaconquickpay.com/l/3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40" }
    }
  }
}
```

El mensaje sale de **tu** número de WhatsApp, y su costo en Meta es tuyo.

4. Lo que hace tu cliente

1. Abre el enlace. Ve tu nombre, el concepto y el monto.
2. Elige su banco y escribe el teléfono afiliado a ese banco y la **cédula del dueño de la cuenta**: si paga con la cuenta de otra persona, va la cédula de esa persona. Si el banco rechaza el pago porque los datos no corresponden a la cuenta (`BENEFICIARIO_NO_COINCIDE`), la página lo dice, le confirma que no se le cobró y lo devuelve a sus datos con la cédula marcada para revisarla. Si quiere, escribe su nombre. Antes de pedir la clave, la página le dice cómo se la va a mandar su banco: Bancaribe por correo electrónico, Mercantil por SMS, Banesco en su app y Banco Plaza con una notificación; con los demás bancos, que su banco se la envía por SMS, notificación o correo.
3. Escribe la clave de 8 dígitos que le manda su banco. Si se equivocó de banco, toca **← Cambiar banco** (o el botón atrás del teléfono, que vuelve al paso anterior sin salir de la página) y pide la clave del banco correcto sin esperar.
4. Mientras el banco confirma, la página muestra **Procesando tu pago** y no deja reenviar el pago. Si tu cliente intenta salir o recargar, el navegador le pregunta antes; si recarga igual, vuelve a esa misma pantalla, no al formulario. Si el banco tarda más de lo normal (90 segundos), se lo dice sin invitarlo a pagar de nuevo, y la página se actualiza sola cuando el banco responde. A los 3 minutos sin confirmación, o apenas el banco avisa que canceló o rechazó la operación, la página deja de pedirle que no la cierre: le explica qué pasa, le dice que no vuelva a pagar y que puede cerrarla y abrir el mismo enlace más tarde. Si a los 10 minutos el banco todavía no confirma, la página le dice que no pudimos confirmar su pago y que, si necesita resolver su compra ya, se comunique contigo o con nosotros con la referencia de la página: los 8 primeros caracteres del `id` del enlace, en mayúsculas. **Mientras el enlace siga en `procesando`, no le cobres por otro medio:** el pago todavía puede confirmarse. Sólo con pagos reales; en modo prueba no aparece. El cobro sigue en `procesando` hasta que el banco confirme (`pagado`) o lo confirmemos cancelado: entonces vuelve a `pendiente` (o `expirado`, si ya venció), desde los 30 minutos o, si falta alguna señal, en varias horas (Estados de un cobro).
5. Ve la confirmación. Si enviaste `success_url`, vuelve a tu web.

Un enlace se paga **una sola vez**. Sólo tras un rechazo del banco puede intentarlo otra vez, con una clave nueva.

Si el banco rechaza el pago

La página le dice a tu cliente, en palabras y sin códigos, por qué no pasó y qué puede hacer:

- **Un dato no corresponde** (la cédula no es la del dueño de la cuenta, el banco no encuentra la cuenta o el teléfono no está afiliado): vuelve a sus datos para corregirlos, en vez de pedir otra clave con los mismos datos.
- **La cuenta o el banco no pueden ahora** (saldo insuficiente, cuenta bloqueada o que no permite este pago, límite de pagos, banco no disponible, falla técnica al pedir la clave): se le sugiere pagar con otra cuenta o banco.
- **La clave no es válida**: pide una clave nueva.

- **El banco no está procesando pagos**: se le dice que intente más tarde.
- **Algo de tu cuenta o nuestro**: le pedimos que nos escriba a hola@pagaconquickpay.com.

Si el rechazo es del banco y ese banco tiene fallas ahora, también ve el aviso de un banco con fallas.

Sólo le decimos «No se te cobró» cuando es seguro: si el banco rechazó el pago en el momento por algo del pago o de la cuenta (por ejemplo, la clave, el saldo o un límite), si el pago no llegó a enviarse al banco, o si el banco no le pudo mandar la clave. En los demás rechazos ve el motivo y qué hacer, sin esa frase. Si el banco no confirma en el momento, la página no lo da por rechazado: le pide que no vuelva a pagar mientras se confirma (ver el punto 4). Si el banco lo rechaza después, ve "El banco no aprobó el pago", sin "No se te cobró". El estado del cobro y la API no cambian.

Cierre diario de los bancos

Todas las noches, de **11:45 p. m. a 1:30 a. m.** (hora de Venezuela), los bancos hacen su cierre diario y no atienden pagos. En esa franja la página de pago:

- Muestra arriba el aviso **Bancos en mantenimiento** con el texto "Los bancos están en su mantenimiento diario y no se pueden hacer pagos. Podrás pagar a partir de la 1:30 a. m. (hora de Venezuela)." y no deja pedir la clave de pago. No se le cobra nada a tu cliente.
- Desde **15 minutos antes** (11:30 p. m.) muestra un aviso que no bloquea: "En 12 minutos los bancos entran en mantenimiento (de 11:45 p. m. a 1:30 a. m.). Si vas a pagar, hazlo ahora." Los minutos se actualizan solos.
- Se actualiza sola cada 30 segundos: una página abierta antes de la franja se bloquea al empezar y se desbloquea a la 1:30 a. m. sin recargar.

Si alguien pide la clave dentro de la franja, la página recibe `503` con `{ "error": "cierre_bancario", "motivo": "Los bancos están en su mantenimiento diario y no se pueden hacer pagos. Podrás pagar a partir de la 1:30 a. m. (hora de Venezuela).", "reintentar_en": 5400 }` (`reintentar_en`, en segundos, y la cabecera `Retry-After` dicen cuánto falta para que termine) y muestra el mismo aviso.

Lo que ya estaba en curso no se corta: un pago con la clave ya pedida sigue su camino y los avisos `link.pagado` y `link.acreditado` llegan como siempre. El enlace sigue `pendiente` y tu cliente lo paga después de la 1:30 a. m. con el mismo enlace, si no vence antes. Los enlaces de **prueba** no tienen esta franja: no pasan por el banco.

Bancos no disponibles por ahora

Con algunos bancos los pagos casi nunca se completan: lo medimos sobre los últimos 90 días de pagos con débito inmediato (por ejemplo, en la medición del 18 de octubre de 2026, con el 0175 se aprobó 1 de 422 claves pedidas). Un banco así sigue en la lista de la página de pago, pero **deshabilitado**, con la etiqueta **(no disponible por ahora)** y la nota "Los bancos marcados «no disponible por ahora» no se pueden usar: con ellos los pagos casi nunca se completan." Tu cliente elige otro banco.

- ePlaza recalcula la medición una vez al día. Un banco puede quedar no disponible cuando aprueba el 10 % o menos de las claves pedidas (con al menos 50 en 90 días), siempre que ePlaza lo haya revisado; vuelve cuando pasa del 15 %.

- Si alguien igual pide la clave con ese banco (por ejemplo, una página que quedó abierta desde antes), la página recibe `409` con `{ "error": "banco_no_disponible", "banco": "0175", "motivo": "No disponible por ahora: con este banco los pagos casi nunca se completan. Elige otro banco." }`. No se le pide la clave al banco ni se cobra nada.
- Los enlaces de **prueba** no tienen esta regla: no pasan por el banco.

Una falla general para procesar pagos

Si el banco que procesa los cobros está fallando para todos (no el banco de tu cliente), la página de pago le avisa antes de pedir la clave: "Estamos teniendo fallas para procesar pagos en este momento. Si tu pago falla, no se te cobra nada. Intenta de nuevo en unos minutos." No le impide intentarlo, y no le ofrece elegir otro banco, porque cambiar de banco no ayuda. Se quita solo cuando el servicio se normaliza. Lo ves también en la página de estado y en el portal → Si hay una falla general. No sale en los enlaces de **prueba**.

Un banco con fallas ahora

Antes de pedir la clave, la página de pago puede avisarle a tu cliente que el banco que eligió tiene problemas en este momento: "Tu banco está presentando fallas ahora; puedes usar otra cuenta o intentar más tarde." No le impide pagar. Uno de los casos en que sale:

- Un banco acepta el intento de pago y después, al liquidarlo, lo cancela por tiempo de espera. Si eso le pasa con un mismo banco a 3 o más intentos de pago en la última hora (contando los de todos los comercios) y son el 40 % o más de los intentos con ese banco en esa hora, la página avisa a quien elige ese banco.
- Por esta causa, el aviso se quita solo cuando en la última hora ese banco tiene menos de 3 intentos cancelados así.
- Si le pasa a más de 2 bancos a la vez, esta regla no marca a ninguno: parece una falla general, no de cada banco.
- No sale durante el cierre diario de los bancos, que tiene su propio aviso, ni en los enlaces de **prueba**.

Pagar con tarjeta (piloto)

Está en **piloto** y apagado: ePlaza lo enciende por comercio. Si no está activo para tu cuenta, tu página de pago se ve exactamente como arriba.

Con el piloto activo, debajo de los bancos tu cliente ve **Paga con tarjeta de débito o crédito**: se aceptan tarjetas de crédito y las tarjetas de débito de Banco de Venezuela, Bancaribe, Bancrecer, Banco del Tesoro, Bancamiga, Banco Digital de los Trabajadores, BANFANB, Banco Fondo Común, Banco Plaza y R4. Antes del botón **Pagar con tarjeta** ve cuánto pagará: «Con tarjeta pagas Bs 103,00 (incluye recargo de 3 % por pago con tarjeta)» para un cobro de Bs 100. Al tocarlo pasa a la página de pago de Credicard, que es donde escribe su tarjeta: ePlaza no ve ni guarda los datos de la tarjeta. Ese formulario dura 6 minutos.

- **El recargo lo paga tu cliente**, no tú: se cobra a la tarjeta el monto (con la propina, si agregó una) más el recargo. A ti se te liquida lo mismo que si te hubiera pagado con débito inmediato: el monto (con propina) menos tu comisión de siempre.

- Sólo en cobros en bolívares de **producción**, desde Bs 5 (el mínimo de Credicard) y hasta el tope del piloto, contando el recargo. No aparece en enlaces de prueba ni en enlaces con pago dividido.
- El enlace se sigue pagando **una sola vez**: mientras el formulario de la tarjeta está abierto, el enlace queda `cobrando` y no se puede pagar con débito inmediato (ni abrir otro formulario); mientras un pago con débito inmediato está en curso, no se puede pagar con tarjeta.
- El enlace pasa a `pagado` sólo cuando Credicard confirma el cobro aprobado. Si lo rechaza o el formulario vence sin pago, el enlace vuelve a `pendiente` y tu cliente puede pagarlo otra vez (con tarjeta o con débito inmediato).
- Te llega el mismo aviso `link.pagado`: `monto`, `propina` y `total` son los del cobro (sin el recargo), con `"metodo": "tarjeta"`, `recargo_tarjeta` (lo que pagó de más tu cliente), en `referencia` la referencia del recibo de la tarjeta y en `txn_id` el id del pago con tarjeta. → Webhooks
- **La liquidación de los pagos con tarjeta no es inmediata**: Credicard abona los pagos con tarjeta en su cierre nocturno. Cuando el abono llega, ePlaza te liquida ese pago y te llega `link.acreditado`, como con cualquier otro pago.

Quiero ser cliente

Debajo de la cédula, tu cliente ve el campo **Tu nombre** (opcional) y la casilla **"Quiero ser cliente de *tu comercio* y recibir sus novedades." **La casilla viene** desmarcada**: sólo la marca quien quiere. Si la marca, el nombre es obligatorio.

Cuando el cobro queda `pagado` (no antes), esa persona aparece en **Clientes** del portal con la etiqueta **Se registró al pagar** y **Acepta novedades**. Si ya la tenías (misma cédula, o mismo WhatsApp sin cédula), no se duplica ni se le cambia el nombre que le pusiste: sólo se marca que acepta novedades. En **Clientes** puedes filtrar por **Aceptan novedades**.

Si quien paga ya está registrado en ePlaza por WhatsApp con esa cédula, se usa el nombre de su registro: no tiene que registrarse dos veces.

Si quien cobra es una **cuenta personal**, la casilla dice **"Guardar mis datos para pagarle más rápido a *su nombre*."** y marcarla no vuelve obligatorio el nombre; sin nombre, el contacto se guarda con el teléfono de quien pagó. Esa casilla no es permiso para enviarle novedades.

Con un enlace de **prueba** se ve igual, pero el cliente queda marcado **Se registró al pagar (prueba)** y no cuenta como permiso para enviarle novedades.

Por la API sólo te llega si lo pidió: `pagador.alta_cliente` al consultar el enlace y en el aviso `link.pagado`. Nunca su cédula, teléfono ni nombre.

5. Enterarte del pago

- **Webhook** `link.pagado` (recomendado): registra tu URL en el portal (**Integraciones › Webhooks**) o con `POST /switch/v1/webhooks`. → Webhooks
- **Consultar el enlace**:

```bash
curl -s https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/links/3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40 \
  -H "x-switch-key: $QP_KEY"
```

```json
{
  "id": "3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40",
  "url": "https://eplaza-qa-pay.pagaconquickpay.com/l/3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40",
  "wa_deeplink": null,
  "qr_png": null,
  "comercio": "tu-comercio",
  "monto": 450,
  "moneda": "VES",
  "concepto": "Servicio de consultoría",
  "estado": "pagado",
  "expires_at": "2026-10-11T14:00:00.000Z",
  "txn_id": "b639fac6-1d2e-4f00-9a1b-3c4d5e6f7a8b",
  "created_at": "2026-10-08T14:00:00.000Z",
  "paid_at": "2026-10-08T14:03:22.000Z",
  "propina": 0,
  "total": 450,
  "pagador": { "alta_cliente": false },
  "codigo_verificacion": "QP-7K3M9-XQ4TA"
}
```

`pagador.alta_cliente` es `true` si quien pagó marcó "Quiero ser cliente de *tu comercio* y recibir sus novedades" en la página de pago. → Página de pago

`propina` y `total` vienen siempre: si activaste propinas y tu cliente agregó una, `total` = `monto` + `propina` es lo que se le debitó. `monto` no cambia.

Cuando el enlace está `pagado`, la respuesta trae `codigo_verificacion`: el mismo código que ve tu cliente en su comprobante. Cualquiera puede verificarlo.

Sólo consultas los enlaces del modo de tu llave: un enlace real consultado con `sk_test_` (o uno de prueba con `sk_live_`) responde `404`.

Qué significa cada `estado`: → Estados de un cobro

6. Listar tus enlaces

```bash
curl -s "https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/links?estado=pagado&limit=20" \
  -H "x-switch-key: $QP_KEY"
```

Devuelve una **lista** de enlaces (cada uno con la forma de la consulta, incluido `codigo_verificacion` si está pagado), del más nuevo al más viejo.

Los filtros se combinan entre sí (todos tienen que cumplirse). Por ejemplo, los pagados o en proceso del 8 de octubre en Venezuela que digan "mensualidad":

```bash
curl -s "https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/links?
estado=pagado,procesando&desde=2026-10-08T04:00:00Z&hasta=2026-10-09T04:00:00Z&q=mensualidad" \
  -H "x-switch-key: $QP_KEY"
```

Parámetro	Notas
`estado`	Opcional. Uno o varios separados por coma: `pendiente`, `cobrando`, `procesando`, `revision`, `pagado`, `expirado`, `cancelado`. Un valor que no es un estado se ignora. Un enlace `pendiente` cuyo `expires_at` ya pasó cuenta como `expirado`.
`desde` · `hasta`	Opcionales. Fecha ISO 8601 (`2026-10-08T04:00:00Z`) o milisegundos desde 1970. Se compara con la fecha del pago o, si aún no se pagó, con la de creación. `desde` entra y `hasta` no. Un día de Venezuela va de las 04:00 Z de ese día a las 04:00 Z del siguiente.
`q`	Opcional. Texto a buscar en el concepto, el nombre de quien pagó, la referencia bancaria del pago, el `id` del enlace o su `codigo_verificacion`. Con varias palabras, cada una tiene que aparecer en alguno de esos campos. No distingue mayúsculas ni tildes.
`min` · `max`	Opcionales. Monto del enlace en Bs (sin propina), ambos incluidos.
`con_propina`	Opcional. `true`: sólo los que tienen propina. `false`: sólo los que no.
`mode`	Opcional. `test` o `live`. Sólo ves los enlaces del modo de tu llave: con `sk_test_`, los de prueba; con `sk_live_`, los reales. Si pides el otro modo, la lista viene vacía.
`origen`	Opcional. Cómo se creó: `api`, `panel`, `mi_pagina`, `boton` o `whatsapp`.
`limit`	Cuántos traer. Por omisión 50, máximo 200.
`offset`	Cuántos saltar, para paginar.

Si `desde`, `hasta`, `min`, `max`, `con_propina`, `mode` u `origen` traen un valor que no se entiende, responde `400` con `{ "error": "validation", "parametro": "…", "detail": "…" }` en lugar de ignorarlo.

7. Cancelar un enlace

Si el pedido ya no va (el cliente se arrepintió, te equivocaste en el monto), cancela el enlace:

```bash
curl -s -X POST https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/links/3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40/cancelar \
  -H "x-switch-key: $QP_KEY"
```

```json
{ "id": "3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40", "estado": "cancelado" }
```

Sólo se puede cancelar un enlace **`pendiente`**. Si tu cliente ya está pagando (`cobrando`, `procesando`, `revision`), ya pagó o el enlace ya venció (`expirado`), responde `409` con `{ "error": "no_cancelable", "estado": "…" }` y no se toca nada: cancelar a mitad de un pago dejaría el dinero sin dueño. Si el `id` no es tuyo, no existe o es del otro modo (un enlace real con tu llave de prueba, o al revés), `404`. Cuando alguien abre un enlace cancelado ve que fue cancelado (y, si pusiste `failure_url`, vuelve a tu web).

8. Muchos cobros de una vez

Si le cobras a una lista de personas (mensualidades, cuotas), crea hasta 500 enlaces en una sola petición: → Cobro por lote

Verificar un comprobante

El comprobante de un pago exitoso lleva un **código de verificación** (por ejemplo `QP-7K3M9-XQ4TA`) y la línea "Verifícalo en eplaza-qa-pay.pagaconquickpay.com/verificar". Con ese código, tú o cualquier persona pueden comprobar que el comprobante es auténtico. Una captura con el monto editado no pasa la verificación, porque el código depende del monto, la moneda y la fecha del pago.

- **En la web:** `https://eplaza-qa-pay.pagaconquickpay.com/verificar` (también acepta `?codigo=QP-…`, para ponerlo en un QR o un enlace).
- **Por API, sin llave:**

```bash
curl -s "https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/verificar/QP-7K3M9-XQ4TA"
```

```json
{
  "valido": true,
  "estado": "válido",
  "monto": 150,
  "moneda": "VES",
  "comercio": "Tu Comercio",
  "fecha_hora": "2026-10-09T18:03:22.000Z",
  "fecha_hora_caracas": "09/10/2026, 02:03 p. m.",
  "referencia_bancaria": "****2808",
  "prueba": false,
  "mensaje": "Comprobante auténtico emitido por ePlaza.",
  "concepto": "Mensualidad de octubre",
  "referencia_externa": null,
  "verificaciones": 1,
  "primera_verificacion": "2026-10-09T18:05:10.000Z",
  "primera_verificacion_caracas": "09/10/2026, 02:05 p. m."
}
```

- `concepto` es el del enlace que se pagó y `referencia_externa` la referencia de tu pedido si el enlace vino de un lote (`null` si no). Sirven para comprobar que el comprobante es del pedido que te están presentando, y no de otro del mismo monto.
- `verificaciones` dice cuántas veces se ha verificado ese código, **contando esta consulta**, y `primera_verificacion` cuándo fue la primera (en UTC y en hora de Caracas). Si te presentan un comprobante que ya se verificó antes, lo verás aquí. Pueden venir `null` si por ahora no se pudo registrar la consulta.
- Si el código no existe responde `404` con `"estado": "no_encontrado"` y el mensaje "No encontramos un comprobante con ese código.".
- Los pagos de **prueba** también tienen código, y la verificación los marca con `"prueba": true` y el mensaje "Pago de prueba (no movió dinero).".
- Nunca se muestran datos de quien pagó (teléfono, cédula ni nombre). La referencia bancaria sale enmascarada, con sólo los últimos 4 dígitos.
- Límite: 20 consultas por minuto por IP (`429` si te pasas).
- Tu servidor recibe el mismo código en `codigo_verificacion`, al consultar el enlace `pagado` y en el aviso `link.pagado`.

Webhooks

Un webhook es un aviso que ePlaza le manda a tu servidor (un `POST` con JSON) cuando pasa algo con tus cobros. Es la forma recomendada de enterarte de un pago: no tienes que consultar.

Eventos

Evento	Cuándo llega
`link.pagado`	El banco confirmó el pago de un cobro: ya puedes entregar
`link.acreditado`	Te transferimos el neto de ese pago a tu cuenta. Llega con la liquidación inmediata, que se está activando por etapas
`link.devuelto`	Una devolución que hiciste desde el portal llegó a tu cliente. Las devoluciones están en piloto: si no están activas para tu cuenta, este aviso no llega
`transaccion.aprobada`	API directa: el banco confirmó la operación (`LIQUIDADA`). Llega una sola vez por operación, junto con `transaccion.liquidada`
`transaccion.liquidada`	API directa: la operación quedó `LIQUIDADA` . En un cobro, confirmada por el banco. En un pago a otro banco, puede volverse `RECHAZADA` después (ver abajo)
`transaccion.rechazada`	API directa: el banco rechazó la operación (`RECHAZADA`). Puede llegar sobre un pago que ya avisó `transaccion.liquidada`

`link.pagado` sale cuando el banco **confirma** que el dinero se movió, no cuando recibe la orden de débito. Según el banco de tu cliente, la confirmación puede llegar en el mismo momento del pago o el cobro pasa primero por `procesando` y el aviso llega cuando el banco lo confirma: suele tardar menos de un minuto, pero a veces tarda bastante más. Si el banco no lo aprueba, el cobro vuelve a `pendiente` y no llega ningún `link.pagado` . Llega una sola vez por cobro. → Estados de un cobro

`transaccion.aprobada` sigue la misma regla: sale cuando el banco **confirma** la operación (`LIQUIDADA`), no cuando recibe la orden de débito (`ACEPTADA` , "enviada a BCV"). Por eso llega junto con `transaccion.liquidada` y su `data.estado` es `"LIQUIDADA"` . Con una llave de prueba, el cobro simulado avisa `transaccion.aprobada` en el acto con `data.estado: "ACEPTADA"` y `referencia: "SIMULADO"` . El cuerpo de los avisos `transaccion.*` es `{ "type", "sentAt", "data": { "id", "estado", "monto", "moneda", "concepto", "referencia", "operacion" } }` , donde `data.id` es el `id` que te devolvió la operación.

En un pago a otro banco, `LIQUIDADA` puede volverse `RECHAZADA` . Cuando el dinero sale hacia una cuenta de otro banco, el banco receptor puede rechazar la transferencia después (por ejemplo, porque la cuenta no está a nombre de la cédula o el RIF indicados). Si eso pasa, la misma operación (`data.id`) recibe `transaccion.rechazada` aunque antes haya recibido `transaccion.liquidada` : el dinero no llegó al beneficiario. ePlaza no lo vuelve a pagar solo. Si tu sistema marcó ese pago como hecho, revértelo al recibir `transaccion.rechazada` . En los cobros esto no pasa.

Un reintegro también puede volverse `RECHAZADA` , aunque sea a una cuenta del mismo banco. El `0000` del envío dice que el dinero salió hacia el banco, no que llegó. Si, cerrado el día en Venezuela, el banco no confirmó la liquidación y el débito no aparece en el estado de cuenta de ePlaza, el reintegro pasa a `RECHAZADA` con `motivo_rechazo: SIN_DEBITO_EN_EXTRACTO` y llega `transaccion.rechazada` con el mismo `data.id` . Aplica a los reintegros de cualquier comercio. Se puede reintentar con `retry_nonce` cuando ePlaza lo da por firme (no antes del tercer día y sólo si el

débito sigue sin aparecer); antes, el reintento responde `409 reintegro_intento_ambiguo`. Si ePlaza no puede darlo por firme (queda agotado), no se vuelve reintentable y sigue respondiendo `409 reintegro_intento_ambiguo` hasta que una persona lo revise. Si el banco lo rechazó por un dato de la cuenta (por ejemplo, la cuenta no está a nombre de la cédula indicada), no se reintenta: corrige el destino. ePlaza no lo re-paga solo. → Estados de un cobro

`data.operacion` dice de qué es la transacción, para que un envío de dinero no se confunda con un pago que recibiste: `cobro` (te pagaron), `pago` (salió dinero: por ejemplo, la transferencia del neto a tu cuenta o un reintegro), `reverso`, y las tres de una devolución: `devolucion_debito` (el débito a tu cuenta, con la clave de tu banco, para devolverle a tu cliente), `devolucion` (el dinero que le llega a tu cliente) y `devolucion_retorno` (si el banco de tu cliente rechazó la devolución, te regresamos lo que te debitamos). Entrega mercancía sólo con `operacion: "cobro"`.

1. Registra tu URL

Lo más fácil: en tu portal, **Integraciones › Webhooks › Agregar URL** (sirve aunque todavía no tengas llave de producción). O por API, **con una llave** `sk_live_`:

```bash
curl -s https://eplaza-qa-pay.pagaconquickpay.com/switch/v1/webhooks \
  -H "x-switch-key: $QP_KEY" -H "content-type: application/json" \
  -d '{ "url": "https://tu-dominio.com/webhooks/quickpay", "eventos": ["link.pagado",
"link.acreditado"] }'
```

El webhook recibe también los avisos de producción, así que crear, cambiar o borrar webhooks por API exige una `sk_live_`: con una `sk_test_` responde `403` con `{ "error": "requiere_llave_live" }`. Con la llave de prueba sí puedes listarlos y ver sus entregas. La URL debe empezar con `https://`. La respuesta trae el `secret` con el que verificas la firma: **se muestra una sola vez**, guárdalo.

Reglas de la URL:

- Tiene que llegar a una **dirección pública de Internet**. Si apunta (o su nombre resuelve) a una red interna — `localhost`, `127.x`, `10.x`, `172.16–31.x`, `192.168.x`, `100.64–127.x`, `169.254.x` o sus equivalentes IPv6— el registro responde `422` con `{ "error": "destino_bloqueado" }`. Lo mismo al cambiar la URL con `PATCH`.
- El servidor tiene que tener **dominio** (`tu-dominio.com`). Un nombre sin punto responde `422` con `{ "error": "url_invalida" }`; el caso más común es el `https://` escrito dos veces (`https://https//tu-dominio.com/…`).
- **No seguimos redirecciones.** Si tu servidor responde `301`, `302`, `307` o `308`, el aviso cuenta como no entregado y se reintenta: registra la URL final.
- Hasta **10 webhooks por comercio**. Con 10, registrar otro responde `409` con `{ "error": "limite_webhooks" }`: borra uno que no uses.

2. Lo que te llega

Cabeceras:

Cabecera	Qué es
`x-quicklink-event`	El evento, p. ej. `link.pagado`
`x-quicklink-timestamp`	Segundos desde 1970 (UTC) en que se firmó
`x-quicklink-signature-v2`	`v2=<hex>`: HMAC-SHA256 de `timestamp + "." + cuerpo`
`x-quicklink-signature`	`sha256=<hex>`: HMAC-SHA256 del cuerpo (firma anterior, se mantiene por compatibilidad)

Cuerpo de `link.pagado`:

```json
{
  "type": "link.pagado",
  "sentAt": "2026-10-08T14:03:22.000Z",
  "data": {
    "link_id": "3f6c2a9e-8b1d-4c7a-9e2f-5a1b7c3d9e40",
    "txn_id": "b639fac6-1d2e-4f00-9a1b-3c4d5e6f7a8b",
    "estado": "pagado",
    "monto": 150,
    "moneda": "VES",
    "concepto": "Pedido #1024",
    "referencia": "10242808",
    "mode": "live",
    "codigo_verificacion": "QP-7K3M9-XQ4TA",
    "propina": 0,
    "total": 150,
    "pagador": { "alta_cliente": false }
  }
}
```

`monto` es lo que pidió el cobro. Si activaste propinas y tu cliente agregó una, viene en `propina` y `total` es lo que se le debitó (`monto` + `propina`, en un solo débito). Sin propina, `propina` es `0` y `total` es igual a `monto`.

`codigo_verificacion` es el código que lleva el comprobante de tu cliente: sirve para verificar que es auténtico. Puede venir `null`; ignóralo en ese caso.

`pagador.alta_cliente` es `true` si quien pagó marcó en la página de pago "Quiero ser cliente de *tu comercio* y recibir sus novedades". Esa persona ya aparece en **Clientes** del portal; el aviso no trae su

cédula, teléfono ni nombre. → Página de pago

`mode` dice si el aviso es de un cobro de **producción** (`live`) o de **prueba** (`test`). En `link.acreditado` viene igual.

`metodo` sólo viene cuando tu cliente pagó con tarjeta (piloto): `"metodo": "tarjeta"`, `recargo_tarjeta` es el recargo por pago con tarjeta que pagó tu cliente (no se te liquida) y `referencia` es la del recibo de la tarjeta. `monto`, `propina` y `total` son los del cobro, sin el recargo. Con débito inmediato el aviso no trae `metodo` ni `recargo_tarjeta`.

Cuerpo de `link.devuelto` (piloto):

JSON

```
{
  "type": "link.devuelto",
  "sentAt": "2026-10-14T21:33:59.304Z",
  "data": {
    "link_id": "4e38874a-eb60-457b-ba53-371a74dbaff4",
    "devolucion_id": "786d2085-00d3-4f61-95c4-b7509cfb5276",
    "estado": "completada",
    "monto": 1,
    "monto_devuelto": 1,
    "moneda": "VES",
    "referencia_bancaria": "SIMULADO",
    "mode": "test"
  }
}
```

Con una llave de prueba `referencia_bancaria` llega como `"SIMULADO"`; en producción es la referencia del crédito a tu cliente.

`monto` es lo que se devolvió en esta devolución y `monto_devuelto` lo devuelto en total de ese cobro (una devolución puede ser parcial y un cobro puede tener varias). Llega una vez por devolución, cuando el dinero ya salió hacia tu cliente.

`link.acreditado` trae además `liquidacion_txn_id`, `monto_bruto`, `comision` y `monto_acreditado`. Con pago dividido (piloto) llega cuando todo el reparto terminó (cada parte pagada o cerrada sin pagar) y trae además `reparto`; `monto_acreditado` es sólo tu parte. Con propina, `monto_bruto` es el `total` (la comisión se calcula sobre lo debitado y la propina te llega con el resto del pago). Si devolviste parte de ese pago desde el portal antes de que te lo acreditáramos (piloto de devoluciones), `monto_acreditado` llega ya descontado: lo devuelto y el costo del banco de la devolución. En ese caso `monto_acreditado` es menor que `monto_bruto − comision`. Si devolviste todo lo que te tocaba, no llega `link.acreditado`. Con donación (piloto), `link.pagado` y `link.acreditado` traen `donacion` y `monto_acreditado` no la incluye: la donación se le transfiere a la fundación y no paga comisión. → Glosario

3. Verifica la firma (siempre)

Usa la firma **v2** y rechaza avisos con más de 5 minutos: así nadie te reenvía uno viejo. Calcula sobre el cuerpo **exacto** que llegó, antes de parsearlo.

Node.js

```JS
import { createHmac, timingSafeEqual } from 'node:crypto';

function verificar(cuerpo, firma, timestamp, secreto) {
  if (!firma || !timestamp || Math.abs(Date.now() / 1000 - Number(timestamp)) > 300) return
false;
  const esperado = Buffer.from('v2=' + createHmac('sha256',
secreto).update(`${timestamp}.${cuerpo}`).digest('hex'));
  const recibido = Buffer.from(String(firma));
  return esperado.length === recibido.length && timingSafeEqual(esperado, recibido);
}

// Express: app.post('/webhooks/quickpay', express.text({ type: '*/*' }), (req, res) => { … })
//   verificar(req.body, req.get('x-quicklink-signature-v2'), req.get('x-quicklink-timestamp'),
process.env.QUICKPAY_WEBHOOK_SECRET)
```

El SDK trae esto listo: `verificarWebhook({ cuerpo, firma, timestamp, secreto })`.

PHP

```PHP
$cuerpo = file_get_contents('php://input');
$ts = $_SERVER['HTTP_X_QUICKLINK_TIMESTAMP'] ?? '';
$firma = $_SERVER['HTTP_X_QUICKLINK_SIGNATURE_V2'] ?? '';
$esperado = 'v2=' . hash_hmac('sha256', $ts . '.' . $cuerpo,
getenv('QUICKPAY_WEBHOOK_SECRET'));
if (!$ts || abs(time() - (int)$ts) > 300 || !hash_equals($esperado, $firma)) {
http_response_code(401); exit; }
```

4. Responde rápido

Responde `200` en menos de 8 segundos y procesa después. Si no respondemos con `2xx`, **reintentamos hasta 6 veces**: de inmediato, a los 30 segundos, 2 minutos, 5 minutos, 15 minutos y 1 hora.

5. No entregues dos veces

Un aviso puede llegar más de una vez (por los reintentos). Guarda qué `link_id` (en los avisos `transaccion.*`, `data.id`; en `link.devuelto`, `devolucion_id`) ya procesaste para cada evento y, si llega repetido, responde `200` sin volver a entregar.

6. Prueba y revisa las entregas

- **Probar:** en el portal, **Integraciones › Webhooks › Probar** envía a tu URL un aviso `webhook.test` firmado igual que los reales (mismas cabeceras, firma v2) y te dice si respondió `2xx`. Ese aviso no trae `data`: respóndele `200` sin procesarlo.
- **Eliminar:** en el portal, **Integraciones › Webhooks › Eliminar** borra la URL aunque todavía no tengas llave de producción (por API, `DELETE /switch/v1/webhooks/:id` exige una `sk_live_`).
- **Historial:** `GET /switch/v1/webhooks/:id/deliveries` te da las últimas entregas con su estado, el HTTP que respondió tu servidor y los intentos. Si en el momento de entregar tu URL resolvía a una red interna, la entrega queda `fallido` sin reintentos, con `status: null` y `error: "destino_bloqueado"`; en las demás, `error` viene `null`. Con una `sk_test_` sólo salen las entregas de avisos de prueba (`data.mode: "test"`); las de producción, y los avisos `transaccion.*`, se ven con una `sk_live_`.

7. Administrar por API

Método	Ruta	Para
GET	`/webhooks`	Listar (sin el secreto). Sirve con cualquier llave
GET	`/webhooks/:id`	Ver uno
PATCH	`/webhooks/:id`	Cambiar `url`, `eventos` o `estado` (`activo` / `inactivo`). Sólo `sk_live_`
DELETE	`/webhooks/:id`	Borrar. Sólo `sk_live_`

Revisa `data.mode` antes de entregar. Llegan los avisos de prueba y los de producción al mismo webhook: si `mode` es `test`, el pago fue simulado y no debes entregar nada de verdad. → Modo prueba

Errores y rechazos

Cuando la API responde con error

El cuerpo siempre trae `error` y, a veces, `detail`:

JSON

```
{ "error": "validation", "detail": { "fieldErrors": { "monto": ["Number must be greater than 0"] } } }
```

HTTP	`error`	Qué pasó	Qué haces
400	`validation`	Un campo falta o no es válido. Al listar enlaces con un filtro que no se entiende, viene además `parametro` con su nombre y `detail` es un texto	Revisa `detail` (y `parametro`)
401	`unauthorized`	Falta la llave, está mal escrita o fue revocada	Revisa `x-switch-key`
403	`comercio_suspendido`	Tu comercio está suspendido: no se pueden crear enlaces ni lotes (ni de prueba ni de producción)	Escríbenos para reactivarlo
403	`requiere_llave_live`	Con una llave de prueba intentaste crear o revocar una llave de producción	Usa una `sk_live_` → Llaves
404	`not_found`	Ese cobro no existe o no es de tu comercio	Revisa el `id`
409	p. ej. `item_duplicado`	Conflicto: en un lote, una `idempotency_key` de ítem ya usada	Revisa `detail`
409	`limite_webhooks`	Ya tienes 10 webhooks registrados	Borra uno que no uses → Webhooks
422	`destino_bloqueado`	La URL del webhook apunta a una red interna	Usa una URL pública → Webhooks
422	`url_invalida`	La URL del webhook no tiene un dominio válido (p. ej. `https://` escrito dos veces)	Revisa la URL → Webhooks
400 / 422 / 503	`split_…`	El pago dividido no está activo, el reparto no es válido, una cuenta no está aprobada en el portal o un control de seguridad la frena	→ Pago dividido
400 / 422 / 503	`donacion_…`	Las donaciones no están activas, la fundación no está habilitada para tu cuenta, la donación no es válida o no te queda nada después de la donación y la comisión	→ Donaciones
409	`idempotency_key_en_uso`	Esa `idempotency_key` ya la usaste con una llave del otro modo (prueba o producción)	Usa otra llave de idempotencia
409	`tope_diario_excedido` / `tope_acumulado_excedido`	Al pedir la clave de pago (página de pago, WhatsApp Flow o `POST /cobrar/token`): ese cobro pasaría el tope de lo que tu cuenta puede recibir hoy o en total. Trae `motivo` (el texto	Tu cliente no paga ahora; escríbenos si necesitas más tope → Límites

HTTP	`error`	Qué pasó	Qué haces
		para tu cliente); con tu llave, además `tope` y `usado` (en Bs) y, si tu tope está en dólares, `topes_moneda`, `tope_usd` y `tasa_bcv`	
409	`banco_no_disponible`	En la página de pago, al pedir la clave con un banco que hoy está no disponible (con él los pagos casi nunca se completan). Trae `banco` y `motivo` (el texto para tu cliente)	Que tu cliente elija otro banco → Enlaces de pago
429	—	Pasaste el límite de peticiones	Espera y reintenta → Límites
503	`cierre_bancario`	En la página de pago, al pedir la clave entre las 11:45 p. m. y la 1:30 a. m. (hora de Venezuela): los bancos están en su cierre diario. Trae `motivo` (el texto para tu cliente) y `reintentar_en` (segundos hasta que termine)	Tu cliente paga después de la 1:30 a. m. con el mismo enlace → Cierre diario
5xx	—	Falla nuestra	Reintenta con la misma `idempotency_key`

Cuando el banco rechaza el pago

Eso no te llega como error de la API: tu cliente lo ve en la página de pago, con el motivo en palabras y qué puede hacer (corregir sus datos, pagar con otra cuenta o pedir una clave nueva), y puede intentarlo otra vez. El cobro sigue `pendiente` hasta que lo pague o venza. → Si el banco rechaza el pago

Si tu comercio está suspendido

Mientras dure la suspensión:

- Crear enlaces (`POST /links`) o lotes (`POST /lotes`) responde `403` con `{ "error": "comercio_suspendido" }`.
- Un enlace que estaba `pendiente` no se puede empezar a pagar: tu cliente ve "Este comercio no está recibiendo pagos" y no se le cobra nada.
- Lo que ya se estaba pagando (`cobrando` , `procesando` , `revision`) termina normalmente, y el dinero de lo ya pagado se te sigue liquidando.
- Consultar y listar tus enlaces sigue funcionando.

Reintentar sin duplicar

Si una creación falla por red o por un `5xx`, reintenta con la **misma** `idempotency_key`: si el primer intento sí llegó, recibes el mismo enlace en vez de uno nuevo.

```JSON
{ "monto": 150, "concepto": "Pedido #1024", "idempotency_key": "pedido-1024" }
```

Pasar a producción

Antes de cobrar de verdad, comprueba en prueba

- [ ] Creas un enlace y guardas su `id` junto a tu pedido.
- [ ] Pagas el enlace de prueba y tu sistema lo ve `pagado`.
- [ ] Tu webhook recibe `link.pagado`, **verifica la firma** y responde `200`.
- [ ] Si llega el mismo aviso dos veces, no entregas dos veces.
- [ ] Entregas sólo con `pagado`, nunca por volver a tu `success_url`.
- [ ] Un cobro en `procesando` o `revision` no le pide al cliente pagar otra vez.
- [ ] Probaste los casos difíciles con las claves de prueba: `11111111` (rechazo) y `22222222` (procesando), o `POST /links/:id/simular`. → Modo prueba
- [ ] Tu webhook ignora (no entrega) los avisos con `"mode": "test"`.
- [ ] Tu llave está sólo en tu servidor.

El día que pasas

Lo único que cambia es **la llave**:

1. Tu cuenta debe estar aprobada (el portal te muestra "Puede cobrar" en Ajustes).
2. En **Integraciones › Llaves de API**, crea una llave de **Producción** (`sk_live_…`).
3. Cámbiala en tu servidor.
4. Haz un cobro real pequeño y confírmalo de punta a punta.

Los avisos de prueba y de producción llegan al mismo webhook (por eso crearlo o cambiarlo por API exige `sk_live_` ; desde el portal puedes registrarlo antes de tener la llave de producción), pero cada uno trae `data.mode` (`test` o `live`). Si es `test` , el pago fue simulado: no entregues de verdad.

Lo demás está en línea

Esta guía trae lo necesario para empezar. Estas guías están en el sitio, siempre actualizadas:

El portal de comercios
eplaza-qa-docs.pagaconquickpay.com/developers/portal

Clientes y cobranza masiva
eplaza-qa-docs.pagaconquickpay.com/developers/clientes-y-cobranza

Productos
eplaza-qa-docs.pagaconquickpay.com/developers/productos

Botón para tu web
eplaza-qa-docs.pagaconquickpay.com/developers/boton-de-pago

Cobro por lote
eplaza-qa-docs.pagaconquickpay.com/developers/cobro-por-lote

Donaciones (piloto)
eplaza-qa-docs.pagaconquickpay.com/developers/donaciones

Volver a tu web y verificar
eplaza-qa-docs.pagaconquickpay.com/developers/volver-a-tu-web

API directa
eplaza-qa-docs.pagaconquickpay.com/developers/api-directa

Límites y reglas
eplaza-qa-docs.pagaconquickpay.com/developers/limites-y-reglas

Con tu IA o en PDF
eplaza-qa-docs.pagaconquickpay.com/developers/con-tu-ia

Tu página de pago
eplaza-qa-docs.pagaconquickpay.com/developers/pagina-de-pago

Suscripciones
eplaza-qa-docs.pagaconquickpay.com/developers/suscripciones

Propinas
eplaza-qa-docs.pagaconquickpay.com/developers/propinas

Equipo y cuentas
eplaza-qa-docs.pagaconquickpay.com/developers/equipo-y-cuentas

Pago dividido (piloto)
eplaza-qa-docs.pagaconquickpay.com/developers/pago-dividido

Cobrar desde tu bot
eplaza-qa-docs.pagaconquickpay.com/developers/integracion-para-bots

Modo prueba
eplaza-qa-docs.pagaconquickpay.com/developers/datos-de-prueba

Entornos y URLs
eplaza-qa-docs.pagaconquickpay.com/developers/referencia

Glosario de campos
eplaza-qa-docs.pagaconquickpay.com/developers/glosario

eplaza-qa-docs.pagaconquickpay.com

Referencia completa de la API, novedades, SDK, colección Postman y la documentación en un solo archivo para tu IA.