

LUNIUM PAY

Manual de integración de la API

Español

Pix • Custodia de BRL • Cripto • Billetera USDT/USDC

Flujos completos, ejemplos de la API, estados de las operaciones y orientación operativa para tu equipo.

Documentación completa en español para socios e integradores.

Edición en español: 18 de octubre de 2026

Manual en línea: docs.luniumpay.com/es/manual

Los campos, enumeraciones, datos de firma y valores originales de los ejemplos se conservan según el contrato de la API. Las tarifas, los límites, la disponibilidad y los importes finales dependen de tu clave y de la respuesta de cada operación.

Índice

Tarifas vigentes	4
1. Elige el flujo	5
2. Claves y configuración	5
Sandbox	6
Producción sin billetera fija	6
Recuperar el monitor y gestionar los accesos	7
Monitor actual y evolución del panel	7
Nuevo panel del socio	7
3. Cobrar PIX para custodia	9
Crear la solicitud de cobro	9
Confirmar el abono	9
Tarifa de depósito y disponibilidad	9
Límites por pagador	10
4. Saldo, subcuentas y movimientos	10
Movimientos por operación: tarifa fija y comisión única	11
5. Retirar el saldo en cripto	12
6. Retirar el saldo mediante PIX	13
7. De PIX a cripto directamente	14
Catálogo y cotización	14
8. Criptomonedas externas a PIX	14
Estados y comprobantes	16
Billetera de devolución: solo cuando falla la venta	17
Comprobante oficial del PIX	17
Consulta del titular y pago de QR	18
Billetera USDT/USDC: saldo en cripto y PIX pagado con él	19

9. Operativa a través de Telegram	21
Lunium Partners: comunidad oficial de integradores	21
Grupo de integración con acompañamiento humano	22
Vinculación y comandos del bot	23
10. Webhooks fiables	24
Eventos	25
Monitor en vivo del socio	25
Feed para tu propio panel	25
Webhook + consulta de estado + actualización de tu interfaz	26
Qué significa cada evento en la pantalla	28
Configurar, probar y recuperar notificaciones	29
Validación del seguimiento en vivo	29
11. Errores, idempotencia y conciliación	30
Procedimiento operativo: qué consultar y mostrar	31
12. Sandbox y pruebas de integración	32
13. Dónde consultar la regla vigente	33
14. Comisión, agentes y operaciones adicionales	33
Configurar la comisión del socio	33
Separar tarifa, comisión, tipo de cambio y cantidad entregada	33
Acreditación, movimientos y retiro de la comisión	34
15. Lista de comprobación de producción	38
16. Qué cambió en esta revisión	39

Documentación para socios

PIX, custodia en reales brasileños y criptomonedas, con el flujo correcto para cada operación.

Revisión 2026-10-09.1 • API 1.45.0 • [OpenAPI](#) • [Estado operativo](#)

Edición en español: 18 de octubre de 2026 • API vigente: 1.45.0. Se conservan las fechas de las revisiones históricas.

Tarifas vigentes

Política del 15/10/2026: el depósito en saldo cuesta R\$ 1,00, sin porcentaje ni comisión. La compra directa cuesta el 3,3% + R\$ 1,00; la conversión del saldo a criptomonedas cuesta el 3,3%, sin otro importe fijo; el retiro del saldo por Pix cuesta el 1,5%. La comisión del socio se añade a las tarifas de compra y salida. Cotización de referencia: ask de Binance, sin recargo. Los costes de red y procesamiento, cuando procedan, aparecen en la vista previa.

saque_cripto_conversao_fee_bps: porcentaje de la conversión; saque_cripto_fee_bps=0;
saque_cripto_conversao_fee_fixed_cents=0.

01 / RECIBIRPIX a custodia Recibe reales brasileños. Elige el destino del retiro después. 02 / RETIRAR Saldo a criptomonedas Moneda, red y billetera elegidas en cada retiro. 03 / OPERAR API mediante Telegram Solicitudes de cobro, saldo y retiros en el grupo del proyecto.

Este manual se ha contrastado con las rutas y los validadores del servicio. Los ejemplos son ilustrativos; los límites, las tarifas, la disponibilidad y los plazos deben consultarse en las respuestas de tu clave. Una revisión de la documentación no demuestra la liquidación de todas las rutas en producción.

[Mínimos y máximos por moneda y red](#) Consulta los límites de cash-in, cash-out y custodia, la tabla con buscador de todas las monedas del catálogo principal, ejemplos de cálculo y las reglas por pagador/destinatario.

1. Elige el flujo

Quieres	Entrada	Operación	Resultado
Cobrar PIX y decidir el retiro después	PIX del pagador	POST /cashin/charge con destino: "saldo"	Saldo en BRL, en el negocio o en la subcuenta
Retirar el saldo en criptomonedas	Saldo BRL disponible	POST /saldo/sacar-cripto	Conversión y entrega en la moneda/red elegidas
Retirar el saldo por PIX	Saldo BRL disponible	POST /payouts	PIX a la clave indicada
Cobrar PIX y entregar criptomonedas directamente	PIX del pagador	POST /cashin/charge con destino: "cripto"	Entrega en la billetera definida para la solicitud de cobro
Vender criptomonedas desde una billetera externa	Depósito de criptomonedas	POST /cash-outs → aceptación → depósito	PIX al destinatario
Pagar PIX con saldo en USDT	Depósito de USDT o USDC en la billetera	POST /carteira-usdt/deposits → POST /carteira-usdt/payments	PIX a la clave o al código Copia y Pega indicado

La custodia es un saldo en reales brasileños. El socio recibe el importe neto del PIX y elige la moneda y la red al retirar. No es necesario registrar una billetera fija. Un depósito en custodia no crea saldos simultáneos en BTC, ETH y USDT.

Ten en cuenta el valor predeterminado de cada canal: en la API, omitir destino sigue significando **cripto**. Para recibir fondos en custodia, envía destino: "saldo". En Telegram, /cobrar valor cpf ya usa custodia de forma predeterminada.



2. Claves y configuración

Únete a [Lunium Partners](#), la comunidad oficial de integradores. Se recomienda participar desde el inicio de la integración: allí compartimos avisos operativos, mantenimientos, cambios de API e intercambios entre socios. Descubre cómo usar la comunidad junto con el grupo dedicado y el monitor.

Base: `https://api.luniumpay.com`. En las rutas autenticadas, envía `X-API-Key` y, en los cuerpos JSON, `Content-Type: application/json`. Usa la clave en tu servidor; autoriza operaciones sobre la cuenta.

Sandbox

```
curl --fail-with-body https://api.luniumpay.com/keys/sandbox \
-H 'Content-Type: application/json' \
-d '{"email": "voce@empresa.com", "name": "Minha integração de teste"}'
```

Sustituye `voce@empresa.com` por el correo de quien realizará la prueba: `email` es obligatorio. Aproximadamente un minuto después, el soporte de LuniumPay envía a esa dirección, una sola vez, una explicación de qué es LuniumPay, la integración en 3 pasos, el manual y el PDF, el diagrama de flujo y VendaCripto (aplicación lista para quienes solo quieren comprar o vender criptomonedas con PIX, sin integrar). Un mensaje por dirección, con un enlace para darse de baja. Sin un `email` válido, la respuesta es 400 con `erro: "email_obrigatorio"`. Sigue siendo una llamada, sin aprobación ni clave de API; `name` es opcional.

Guarda `api_key` como `LUNIUM_API_KEY` en el entorno de tu servidor. Las claves nuevas de prueba tienen el prefijo `lun_test_`; las de producción, `lun_live_`. Las claves antiguas `lun_` siguen siendo válidas.

El sandbox simula PIX de entrada, abonos en custodia BRL, entregas y retiros de criptomonedas, payout PIX, transferencias internas y cash-out. El QR no se puede pagar: confirma en `POST /sandbox/cashin/{cashin_id}/pay`. No se activa ningún banco, plataforma de intercambio ni blockchain. Consulta [la guía del sandbox](#) y `GET /sandbox` para conocer los escenarios, la persistencia y las diferencias con producción. La custodia no requiere una billetera fija.

Producción sin billetera fija

```
curl --fail-with-body https://api.luniumpay.com/keys \
-H 'Content-Type: application/json' \
-d '{"name": "Meu projeto", "email": "operacao@example.com"}'
```

Sustituye el nombre y el correo por los tuyos. La respuesta incluye `api_key`, `key`, `limits`, `monitor_url` y las instrucciones de Telegram. La clave completa solo se presenta al crearla. `monitor_url` es el panel de seguimiento; trata también ese enlace como información privada del proyecto.

Las claves de producción nuevas se crean con custodia y payouts habilitados. La situación de tu clave es la que prevalece: consulta `payout.enabled` en `GET /keys/me` y `custodia_ativa` en `GET /saldo`.

```
curl --fail-with-body https://api.luniumpay.com/keys/me \
-H "X-API-Key: $LUNIUM_API_KEY"

curl --fail-with-body https://api.luniumpay.com/saldo \
-H "X-API-Key: $LUNIUM_API_KEY"
```

Configura un receptor HTTPS de webhooks mediante `PATCH /keys/me` con `webhook_url`; guarda el `webhook_secret` devuelto. `settlement_address` es una opción para la entrega directa en Polygon. No es un requisito para la custodia ni define los destinos de los retiros futuros.

Reutiliza tu clave de prueba: la creación de claves tiene límites por IP. En 429, respeta el plazo indicado. No crees una clave por operación.

Recuperar el monitor y gestionar los accesos

El monitor privado permite seguir el proyecto en modo de lectura. La clave de API autoriza operaciones. El enlace del monitor, la clave de API y el secreto del webhook son credenciales distintas; recuperar uno de ellos no recupera los demás.

Si perdiste el enlace, solicítalo para el correo electrónico registrado en la clave de producción:

```
curl --fail-with-body https://api.luniumpay.com/keys/painel-link \
-H 'Content-Type: application/json' \
-d '{"email": "operacao@example.com"}'
```

Esta ruta no requiere clave. La respuesta 200 {"ok": true, ...} es genérica para proteger el registro: no confirma la existencia de la cuenta ni la entrega del correo. Revisa la bandeja de entrada y el spam. En 429, espera; no automatices solicitudes sucesivas. Si el correo es incorrecto o inaccesible, contacta con las personas responsables del proyecto para validar la recuperación.

Con la clave disponible, GET /keys/me devuelve monitor_url. Para invalidar un enlace expuesto, usa PATCH /keys/me con {"rotate_monitor_token": true} y comparte el nuevo enlace solo con el equipo autorizado. El enlace anterior deja de funcionar inmediatamente. Esto no cambia la clave de API ni el secreto del webhook.

La clave completa solo se muestra al crearla. Este procedimiento no vuelve a mostrarla ni la recupera. Si se perdió o quedó expuesta, contacta con soporte para gestionar su sustitución/revocación y la continuidad de la cuenta. No crees otra clave suponiendo que el saldo, las solicitudes de cobro o los vínculos se transferirán automáticamente. Para cambiar el secreto del webhook, sigue el procedimiento específico.

Monitor actual y evolución del panel

En esta revisión, el método documentado de seguimiento es el monitor privado, el feed y las rutas públicas del contrato. Las funciones adicionales del panel y los múltiples destinos de webhook solo deben integrarse cuando estén publicados y validados en producción. Configura el receptor documentado en PATCH /keys/me; no presupongas una API pública de múltiples destinos ni rutas de acceso/2FA a partir de anuncios de desarrollo.

Para Fable y otras IA: comprueba la [versión y el inventario revisados](#) junto con el [OpenAPI](#). Si hay discrepancias, registra qué fuente y operación difieren y suspende la implementación de esa capacidad hasta aclararlo; no inventes una ruta para completar el flujo.

Nuevo panel del socio

El panel del socio está en parceiros.luniumpay.com, con el mismo acceso (correo electrónico y contraseña). También puedes entrar con un código enviado por correo electrónico o con tu cuenta de Google; si aún no tienes acceso, usa Crear cuenta. El panel permite a tu equipo operar y seguir el proyecto. La integración continúa a través de la API, con la clave del proyecto: las pantallas del panel usan la sesión de quien haya iniciado sesión y no forman parte del contrato público de la API.

- Inicio: resumen del negocio por período (Pix recibido, ventas de cripto, Pix enviado, retiros en cripto y comisión), estado de las solicitudes de cobro, de las ventas y de los pagos, y calendario.
- Comprobantes: PDF en las ventas, los Pix enviados y las solicitudes de cobro.

- Pagar con cripto: elige entre las stablecoins (USDT y USDC en todas las redes, más DePix) y las 30 principales criptomonedas, con buscador e iconos. La billetera de devolución solo se solicita si falla la venta (Billetera de devolución).
- Billetera USDT/USDC: depósito con código QR de la dirección y cuenta atrás del plazo de 1 hora; Pix pagado con el saldo. Más detalles en Billetera USDT/USDC.
- Comisión: retiros a partir de US\$ 10.
- Notificaciones: campana con la actividad del proyecto y avisos de cada operación por Telegram y correo electrónico.
- Equipo: roles Propietario, Financiero, Operador, Desarrollador y Lectura; segundo factor en Mi acceso.
- Mi acceso: el botón Cerrar las demás sesiones cierra el panel en los otros dispositivos y mantiene la sesión actual. Una cuenta sin contraseña (acceso por código o Google) desactiva la verificación en dos pasos mediante un código de la aplicación de autenticación o un código de recuperación. Al cambiar la contraseña, la contraseña actual admite un máximo de 10 intentos fallidos por hora, y se envía un correo de aviso siempre que la contraseña se cambia, se crea o se restablece.
- Webhooks: Reenviar ahora repite la entrega de un evento desde el propio panel.
- Legibilidad: más contraste y letras más grandes, “Pix” escrito de la misma forma en todas las pantallas, títulos que coinciden con los del menú y mensajes de error en un portugués claro.

Activación de una cuenta nueva. Al activar el panel con un correo que aún no tiene cuenta (mediante el enlace que llega por el bot o por correo electrónico, o con la clave del proyecto), recibes en ese correo un código de 6 dígitos para confirmar. El código dura 30 minutos y permite hasta 5 intentos; Reenviar código permite obtener un código nuevo después de 1 minuto, hasta 5 veces; solo es válido el más reciente. Si el correo ya tiene cuenta, se sigue confirmando con la contraseña de la cuenta.

Invitación para quienes acceden por código o Google. Una cuenta sin contraseña acepta la invitación así: entra en el panel con el correo invitado y abre de nuevo el enlace de invitación; con la sesión iniciada, se acepta con un clic.

Reglas de seguridad del panel:

- Solo el rol Propietario ve el enlace y el código de vinculación de Telegram, desconecta conversaciones y genera un código nuevo. El código cambia automáticamente cuando alguien pierde el acceso, deja de ser propietario o cuando se desconecta una conversación.
- Cuando alguien deja de ser propietario, los demás propietarios reciben un correo para revisar, en Notificaciones, las conversaciones de Telegram conectadas: siguen vinculadas hasta que un propietario las desconecte.
- La conversación conectada también consulta el saldo y mueve dinero mediante el bot: en privado, quien la conectó; en un grupo, los administradores. Conecta solo conversaciones de confianza.
- Desde el panel, solo el Propietario cambia la billetera de liquidación (`settlement_address`) y las comisiones adicionales (`partner_fee_bps`, `partner_cashout_fee_bps`, `partner_saque_fee_bps`). El rol Desarrollador sigue ajustando el webhook, el nombre y el monitor.
- La sesión dura 12 horas y se renueva mientras usas el panel, hasta 7 días desde el acceso. La cookie de sesión está protegida y las páginas tienen una política de seguridad de contenido.

3. Cobrar PIX para custodia

Crear la solicitud de cobro

Consulta antes `GET /cashin/limits?payer_tax=CPF_OU_CNPJ`, autenticado con la clave del proyecto. El documento pertenece a quien pagará el PIX; `customer_ref` identifica la subcuenta de tu sistema. Son datos distintos.

El ejemplo usa R\$ 50,00. Sustituye el documento y la referencia, y elige un importe permitido para ese pagador. La llamada en producción crea una solicitud de cobro real.

```
curl --fail-with-body https://api.luniumpay.com/cashin/charge \
-H "X-API-Key: $LUNIUM_API_KEY" \
-H 'Content-Type: application/json' \
-d '{
  "amount_cents":5000,
  "payer_tax_number":"00000000191",
  "destino":"saldo",
  "customer_ref":"cliente-42",
  "external_id":"deposito-cliente42-001"
}'
```

No envíes `payout_address` para este flujo. Sin `customer_ref`, el abono pertenece al saldo de la propia clave, llamado casa. Con la referencia, pertenece solo a esa subcuenta.

Guarda `cashin_id` y `external_id`. Muestra `qr_copypaste`, `qr_image_url` cuando esté disponible, el importe y el `expires_at` devueltos. Un QR generado significa que se ha creado una solicitud de cobro; aún no significa que se haya recibido dinero.

Confirmar el abono

Usa `GET /cashin/{cashin_id}/status` o los eventos de webhook. Sigue dos estados:

Campo	Qué indica	Cómo actuar
<code>status</code>	Estado del PIX	<code>pending</code> , <code>delayed</code> , <code>under_review</code> , <code>paid</code> , <code>expired</code> , <code>refunded</code> o <code>failed</code>
<code>settlement_status</code>	Estado del abono/entrega	<code>pending</code> , <code>sending</code> , <code>entregando</code> , <code>sent</code> , <code>incerto</code> o <code>failed</code>

Considera el abono/entrega completado en `cashin.settled` o al consultar `status`: `"paid"` con `settlement_status`: `"sent"`. En custodia, confirma el importe y la disponibilidad en `GET /saldo?customer_ref=cliente-42` y en los movimientos. Un abono registrado puede estar bloqueado por una retención; no gastes `total_cents` como si fuera `disponivel_cents`.

`depix_received_cents` es el importe recibido del liquidador antes de deducir las tarifas de depósito de la clave. El nombre es heredado y no implica que ese sea el abono neto del cliente. Para conciliar, usa los movimientos del saldo.

Tarifa de depósito y disponibilidad

`GET /saldo` publica `deposito_fee_bps`, `deposito_fee_fixed_cents`, `carencia_horas`, `carencia_ate`, `retencao_acima_cents` y `proximas_liberacoes`. 100 bps = 1%. Los nuevos depósitos tienen `deposito_fee_bps=0` y `deposito_fee_fixed_cents=100`, sin comisión del socio.

Hay tres conceptos separados:

Regla	Dónde aparece	Significado
Retención del proveedor	held, held_hours, delay_until, estado delayed	PIX pagado, pendiente de liberación del proveedor
Retención local de la solicitud de cobro	hold_hours al crear, liberar_em en el estado, evento cashin.held	El PIX puede estar paid, pero el abono/entrega aún está pendiente de
Período de espera del saldo	carencia_horas, carencia_ate, abonos bloqueados en /saldo	Regla de la clave/subcuenta para disponibilidad y retiro

held_hours y hold_hours no son alias. Una solicitud de cobro puede tener held: false y seguir sujeta a una retención local. No prometas una liberación inmediata a partir de ese booleano aislado.

No existe un período de espera universal de 24 horas para todos los socios. Las claves con liberación inmediata del saldo tienen sus propias reglas, pero siguen sujetas a retenciones del pagador, de la solicitud de cobro y a bloqueos. Lee conjuntamente la creación, el estado y el saldo; un retiro rechazado por el período de espera devuelve 423 y el momento de liberación.

Límites por pagador

GET /cashin/limits publica el rango de liberación inmediata, la posibilidad de QR con retención, el uso y la próxima liberación del documento. Las condiciones de la clave pueden modificar las reglas. La consulta es orientativa: la creación vuelve a validar la operación, incluido el flujo de custodia con liberación inmediata.

allow_hold: false rechaza la solicitud de cobro cuando requiere la retención calculada a partir del límite del pagador. No desactiva otras reglas contractuales ni bloqueos posteriores. Omitir el campo permite la retención aplicable. Muestra estas condiciones antes del pago.

El mínimo efectivo también depende de las tarifas y del proveedor. No uses R\$ 1,00 como prueba universal ni confundas el mínimo global con el mínimo admitido por la ruta.

4. Saldo, subcuentas y movimientos

Consulta	Alcance y paginación
GET /saldo	Saldo de la cuenta principal; no suma los clientes
GET /saldo?customer_ref=cliente-42	Solo la subcuenta indicada
GET /saldo/consolidado	Cuenta principal y clientes, con el total en custodia
GET /saldo/clientes?limit=50&com_saldo=1	Clientes; continúa con cursor=next_cursor
GET /saldo/extrato?customer_ref=cliente-42&limit=50	Movimientos; continúa con before=proxima_pagina

Movimientos por operación: tarifa fija y comisión única

En las respuestas de payout y en data de los webhooks payout. *, usa `fee_service_total_cents` para la tarifa de servicio del cliente: ya incluye Lunium + socio. `fee_provider_cents` es la tarifa Pix cobrada al cliente. `fee_cents = fee_service_total_cents + fee_provider_cents`; el débito completo es `amount_cents + fee_cents`. No sumes `fee_partner_cents` de nuevo: este y `fee_service_cents` son componentes contables ya incluidos. Los valores proceden de la operación guardada y conservan el redondeo en centavos; `null` indica un componente desconocido en un registro antiguo. Para las operaciones nuevas, el porcentaje mostrado procede de `saque_fee_total_bps`; un cambio de comisión solo afecta a las operaciones nuevas. Ejemplo: Pix R\$ 50,00, Lunium 1,5% + socio 1% y tarifa Pix R\$ 1,00: servicio R\$ 1,25, tarifas R\$ 2,25 y débito R\$ 52,25. Las aplicaciones de los socios deben consumir estos campos para actualizar sus pantallas y notificaciones.

Para el historial del usuario, consulta `GET /saldo/extrato?customer_ref=cliente-42&visao=operacoes`. La respuesta incluye una fila por operación. En los detalles, `taxas` contiene una única categoría `servico`, con la descripción “Tarifa”, que suma la tarifa Pix y todas las comisiones; no hay una fila, etiqueta ni porcentaje de comisión separado para el cliente. Lo mismo se aplica en sandbox.

Para mostrar la tarifa vigente antes de un nuevo retiro, `GET /saldo` proporciona `saque_fee_total_bps` (tarifa de la cuenta principal + comisión adicional del socio; 250 bps = 2,5%). La tarifa fija en centavos está en `saque_taxa_estimada_cents` cuando `saque_taxa_provedor_tipo=fixo`. Muestra únicamente tarifa fija + comisión total. Los campos de componentes y los movimientos contables estándar están destinados al backend/conciliación; no los envíes directamente a la interfaz del cliente. Usa los importes monetarios cotizados/guardados, porque los componentes tienen un redondeo interno en centavos; no recalculas cobros antiguos con el porcentaje actual.

`valor_cents` es el efecto total con signo en el saldo: negativo = débito; positivo = crédito. `valor_operacao_cents` es el principal con signo; `taxas_cents` es el total absoluto de las tarifas incluidas. Cada tarifa indica `tipo`, `descricao`, `valor_cents` absoluto y `efeito_saldo_cents` con signo. La identidad es `valor_cents = valor_operacao_cents + soma(taxas[].efeito_saldo_cents)`. No descuentes las tarifas de nuevo.

Ejemplo: Pix de R\$ 2,00, tarifa fija R\$ 1,00 y comisión total R\$ 0,05 → una salida de R\$ 3,05, con principal -200, tarifas 105 y total -305. El detalle solo muestra Tarifa: R\$ 1,05. Genera una notificación por operación, identificada por `origem_id` y `tipo`, nunca por componente de tarifa. No conviertas las tarifas en tarjetas, notificaciones de ingresos ni importes positivos en verde. Las reversiones aparecen en filas propias; una devolución debitada conserva su signo negativo. Los ajustes de tarifas sin principal disponible siguen visibles para no ocultar movimientos.

`status/status_label` reflejan el estado persistido de la operación en el momento de la consulta; `settlement_status` indica la entrega de cripto. Un estado desconocido es `null`, no una conclusión supuesta. Actualiza la consulta mientras la pantalla esté abierta y usa los webhooks de estado para iniciar una nueva lectura; deduplica las notificaciones por operación y transición de estado, no por asientos de tarifas. La vista agrupada no modifica los asientos contables ni actualiza automáticamente las aplicaciones de terceros.

La paginación agrupa antes de aplicar `limit` (1-200). Vuelve a enviar `before=proxima_pagina` mientras `has_more=true`; `proxima_pagina=null` finaliza. El cursor es una cadena decimal anclada en el primer asiento; no lo conviertas a `Number`. Usa el mismo `customer_ref` en todas las páginas. `movimentos_ids` hace referencia a los asientos originales para la conciliación; no muestres esos IDs como actividades nuevas. Actualiza la primera página para reflejar tarifas o estados que lleguen después. Un error de cursor/límite en esta vista devuelve HTTP 400.

Reutiliza exactamente el mismo `customer_ref` en el depósito, las consultas y el retiro. El límite es de 80 caracteres. Esta referencia separa los saldos; la clave de API sigue teniendo autoridad sobre todas sus subcuentas. La autorización de tus usuarios también debe existir en tu sistema.

Transferencia interna, sin generar PIX, mediante `POST /saldo/transferir`:

```
{
  "amount_cents":1000,
  "from_customer_ref":"cliente-42",
  "to_customer_ref":"casa",
  "external_id":"transferencia-interna-001",
  "descricao":"Transferência autorizada pelo cliente"
}
```

`casa` representa el saldo sin subcuenta. Solo puede transferirse el disponible; el período de espera de la subcuenta de origen sigue siendo aplicable. Un saldo insuficiente en esta ruta devuelve `409 saldo_insuficiente`; en las rutas de retiro, normalmente `402`.

5. Retirar el saldo en cripto

1. Consulta `GET /saldo` en el ámbito correcto.
2. Consulta `GET /cashin/catalog`: es el catálogo de entrega, usado también al retirar de la custodia.
3. Ofrece pares de `instant` y `convert` con `entregavel: true`. Para retirar saldo, excluye `Liquid/DePix`, que no son compatibles con este flujo.
4. Selecciona moneda, red, billetera de destino y memo/tag cuando sea necesario. La billetera puede cambiar en cada retiro.
5. Envía `POST /saldo/sacar-cripto` y sigue el `cashin_id` devuelto en `GET /cashin/{cashin_id}/status`.

Cuerpo ilustrativo; sustituye la billetera y el documento antes de enviarlo:

```
{
  "amount_cents":10000,
  "asset":"usdt",
  "chain":"polygon",
  "payout_address":"0x1111111111111111111111111111111111111111",
  "tax_number":"00000000191",
  "customer_ref":"cliente-42",
  "external_id":"saque-cripto-cliente42-001"
}
```

`amount_cents` es el importe en BRL que se convertirá, no la cantidad de cripto. El documento `tax_number` es del cliente que retira. En las redes con memo, envía `payout_tag` según la billetera receptora y `memoRequired` del catálogo. Los alias `memo`, `tag` y `destination_tag` son aceptados.

La respuesta incluye `fonte: "saldo"`, `destino: "cripto"`, `fee_cents` y `debited_cents`. El débito incluye el importe más la tarifa de retiro de la cuenta principal. La cantidad final de cripto también tiene en cuenta la conversión y los costes de entrega; no calcules lo recibido dividiendo únicamente el débito por una cotización de mercado.

`GET /saldo` indica `saque_cripto_fee_bps`. `POST /cashin/preview` ayuda a consultar la conversión y el mínimo de la ruta, pero no es una vista previa completa del débito del retiro. La vista previa de retiro usada por Telegram es interna; no existe una ruta pública `/saldo/sacar-cripto/preview`.

El campo opcional `max_debited_cents` permite limitar el débito de BRL: si la suma supera el límite, la API rechaza con 409 `taxa_alterada`. No fija la cotización ni garantiza una cantidad mínima de tokens.

No anuncios “todas las criptomonedas” sin restricciones: las opciones son los pares entregables del catálogo actual, sujetos a mínimos, mantenimiento y validación de direcciones. `stale: true` indica un catálogo de contingencia; vuelve a validar antes de confirmar.

6. Retirar el saldo mediante PIX

POST `/payouts` reserva y debita el saldo disponible de la cuenta principal o de la subcuenta antes de enviar el PIX. Se necesitan `payout.enabled`, custodia activa y saldo para el importe más las tarifas.

```
{
  "amount_cents":5000,
  "pix_key":"recedor@example.com",
  "pix_key_type":"email",
  "tax_number":"00000000191",
  "customer_ref":"cliente-42",
  "external_id":"saque-pix-cliente42-001"
}
```

Sustituye la clave PIX y el documento por los del destinatario. `tax_number` debe contener un CPF o CNPJ válido para claves de correo, teléfono y aleatorias. Para una clave CPF/CNPJ válida, la API obtiene el documento de la propia clave; si se indica, debe coincidir. `beneficiary_name` sigue siendo opcional y no acredita la titularidad. Las claves de 11 dígitos requieren `pix_key_type: "cpf" o "phone"`; los teléfonos usan el prefijo internacional, como +55.

Este flujo usa solo saldo en reales brasileños recibido por PIX. `funding_source` es opcional y solo admite `pix_brl`. El máximo por PIX es de 600000 centavos (R\$ 6.000,00), sujeto a un límite inferior de la cuenta y a las tarifas.

En 409 `external_id_divergente` o `external_id_nao_verificavel`, consulta la orden indicada por `payout_id/recovery_endpoint`. Conserva el identificador; no generes otro para evitar el conflicto. En 503 `idempotencia_indisponivel`, espera y consulta el mismo `external_id`: puede existir una orden anterior.

La billetera USDT está separada del saldo BRL: requiere habilitación del proyecto, recibe depósitos externos de USDT y no usa `/payouts`. Cuando está habilitada, usa `funding_source=crypto_usdt`; este flujo no exige el documento del destinatario. Sus límites proceden de la cotización y de las condiciones de la API, no del límite máximo de R\$ 6.000 del saldo BRL. No está disponible en el sandbox.

Precio: `amount_cents + fee_cents`, con `fee_cents = fee_service_total_cents + fee_provider_cents`. Consulta `saque_fee_total_bps`, `saque_taxa_proveedor_tipo`, `saque_taxa_proveedor_piso_cents`, `saque_taxa_estimada_cents` y `saque_taxa_proveedor_formula` en GET `/saldo`. Lee la tarifa vigente en la API; no fijas “1%” ni “R\$ 2,00” en tu código. El payout registra la tarifa efectiva.

Consulta GET `/payouts/{payout_id}`. `processing` aún no es un pago completado; `sent` indica un envío confirmado. `failed` y `refunded` requieren conciliar el estado y los movimientos. Ante un `timeout`, sigue consultando la operación original: no presupongas una reversión ni vuelvas a enviar con otro identificador.

Tras la confirmación, `e2e`, `paid_at`, `receiver_name`, `receiver_tax_number` y `verify_url` pueden incluir la prueba del PIX. Los campos aún no obtenidos pueden ser `null`.

7. De PIX a crypto directamente

Para entregar crypto después de cada PIX, usa destino: "crypto", el par del catálogo de entrega y payout_address. Sin una dirección en el cuerpo, el servicio puede usar la billetera Polygon configurada en la clave; no uses esa configuración como billetera universal para otras redes.

```
{
  "amount_cents":10000,
  "payer_tax_number":"00000000191",
  "destino":"crypto",
  "asset":"usdt",
  "chain":"polygon",
  "payout_address":"0x1111111111111111111111111111111111111111",
  "external_id":"compra-crypto-001",
  "allow_hold":false
}
```

Catálogo y cotización

GET /cashin/catalog enumera lo que la API puede entregar. Usa asset y network del catálogo; en el cuerpo del cash-in y del retiro, el nombre del campo de red es chain. instant puede estar vacío. La presencia en convert no exime de entregavel, mínimo o memo. La tabla de mínimos por moneda y red, con las cifras actuales, está en [Mínimos y redes](#).

```
curl --fail-with-body https://api.luniumpay.com/cashin/preview \
-H "X-API-Key: $LUNIUM_API_KEY" \
-H 'Content-Type: application/json' \
-d '{"amount_cents":10000,"asset":"usdt","chain":"polygon"}'
```

La vista previa no crea una solicitud de cobro ni fija el tipo de cambio hasta el pago. Lee min_br_l_cents, las tarifas y la cantidad estimada de la respuesta. La cantidad que llega a la billetera del cliente se devuelve en asset_amount, en todas las rutas y también en sandbox, ya descontada la tarifa de retiro de la red (withdraw_fee, en unidades del activo). Esta tarifa es fija por envío y pesa más en importes pequeños (BTC en la red Bitcoin, por ejemplo); para mostrarla en reales brasileños, usa withdraw_fee × br_l_per_asset. En las rutas con conversión, usdt_amount es el USDT usado en la compra, no lo que recibe el cliente; amount es el nombre anterior de asset_amount. eta/etaSeconds son estimaciones de la ruta, no un plazo máximo contractual.

El importe recibido y la cotización aplicable en la liquidación determinan la entrega. Muestra settlement_tx_hash y settlement_tx_url cuando estén presentes; no construyas enlaces siempre a Polygon. Un identificador interno de entrega puede no tener un explorador público.

El CPF/CNPJ indicado debe ser el del pagador. Una discrepancia puede generar una devolución (PAYER_MISMATCH) o una revisión cuando no se complete la devolución. No acredites basándote en un comprobante enviado por el cliente ni en under_review.

8. Criptomonedas externas a PIX

Este flujo recibe criptomonedas de una billetera externa. Para gastar saldo en custodia mediante PIX, usa /payouts.

Cripto a Pix: operación y devolución

El cliente envía cripto y el destinatario recibe Pix. Disponible mediante la API o el panel.

A • Conversión y pago

1. Consultar cotización

`QUOTE_CREATED`

`POST /cash-outs`

Indica la cantidad de cripto o el importe en BRL y la clave Pix.



2. Aceptar cotización

`AWAITING_DEPOSIT`

`POST /cash-outs/{cashout_id}/accept`

Se fija la cotización y se genera la dirección de depósito.



3. Depositar según las instrucciones

`EXPIRED` (sin envío a tiempo)

Envía el importe exacto a la dirección y por la red indicadas antes del vencimiento. El QR de la dirección no incluye el importe.



4. Esperar confirmaciones de red

`cashout.deposit_detected` • `cashout.processing`

El Pix se ejecuta después de las confirmaciones de red. Las etapas pueden variar según la ruta. Si falla, sigue el flujo de devolución.



5. Pix pagado

`COMPLETED`

`cashout.completed` • `cashout.receipt_ready`

Comprobante y E2E: `receipt_url`, `receipt_pdf_url` y `pix_e2e`.

B • Devolución si la conversión falla

1. Se necesita una billetera de devolución

`AWAITING_REFUND_ADDRESS`

`cashout.awaiting_refund_address`

La billetera de devolución es opcional al crear la orden. Si ya se indicó, puede pasar directamente a la devolución.



2. Indicar la dirección de devolución

`POST /cash-outs/{cashout_id}/refund-address`

Usa la red exigida para la devolución. No devuelvas automáticamente a la dirección de origen del depósito.



3. Devolución en proceso

`REFUNDING_CRYPTO` • DePix: `REFUNDING`

`cashout.refunding`

Sigue el estado de devolución de la misma orden.



4. Devolución completada

`REFUNDED`

`cashout.refunded`

Comprobante en la red: `refund_tx_hash`.



5. Si se enviaron fondos al proveedor

`MANUAL_REVIEW`

Espera a que el proveedor devuelva los fondos y se confirme su recepción. No se devuelve más de lo realmente recuperado.

Monedas, redes y mínimos: `GET /catalog`. Vista previa sin crear una orden: `POST /cash-outs/preview`.

Consulta el catálogo vigente. Manual completo: docs.luniumpay.com/es/manual#cashout

1. Consulta `GET /catalog`, el catálogo de entrada de criptomonedas; es distinto de `/cashin/catalog`.
2. Opcionalmente, usa `POST /cash-outs/preview` para obtener una estimación sin una orden.
3. Crea la cotización mediante `POST /cash-outs`.

4. Muestra la cantidad, la red, el importe neto en reales brasileños, el destinatario y la validez. Acepta mediante `POST /cash-outs/{cashout_id}/accept`.
5. Solo entonces muestra el `deposit_address`, el `deposit_tag` cuando exista y el importe exacto devuelto.
6. Sigue `GET /cash-outs/{cashout_id}` o el webhook hasta el estado final.

Ejemplo de sandbox, usando `LUNIUM_API_KEY` de prueba:

```
curl --fail-with-body https://api.luniumpay.com/cash-outs \
-H "X-API-Key: $LUNIUM_API_KEY" \
-H 'Content-Type: application/json' \
-d '{
  "asset": "USDT",
  "network": "polygon",
  "amount": "10.00",
  "pix_key": "teste@example.com",
  "pix_key_type": "email",
  "external_id": "sandbox-cashout-001"
}'
```

Envía `amount` (criptomonedas) o `br_l_amount` (reales brasileños deseados), como cadena decimal con punto. No envíes ambos. La creación puede responder con `200` en producción o `201` en el sandbox; acepta respuestas `2xx` según el contrato. `cashout_id` es el identificador devuelto, no un ID creado por el integrador.

`refund_address` es opcional al crear la orden, también en la venta de DePix: la billetera de devolución solo se solicita si falla la venta (consulta Billetera de devolución). Si ya tienes la billetera del cliente, puedes enviarla: debe ser válida para la misma red y estar bajo el control del destinatario de la devolución. El origen de un retiro de un exchange puede ser su billetera colectiva; no presupongas que sirve como billetera de devolución.

Estados y comprobantes

Fase	Estados que pueden aparecer
Cotización y depósito	QUOTE_CREATED, AWAITING_DEPOSIT, DEPOSIT_DETECTED
Confirmación y conversión	DEPOSIT_CONFIRMED, SELLING, SOLD, FORWARDING, PAYING_OUT
Finalización del PIX	COMPLETED
Excepciones	MANUAL_REVIEW, AWAITING_REFUND_ADDRESS, REFUNDING_CRYPTO, REFUNDING (venta de DePix), REFUNDED, EXPIRED, FAILED

No todas las rutas pasan por todas las etapas. `timeline`, `confirmations` y `confirmations_required`, cuando existan, ayudan a hacer el seguimiento. No trates la confirmación del depósito como un PIX completado.

En `COMPLETED`, comprueba `pix_e2e`, `pix_paid_at`, `receipt_url`, `receipt_pdf_url` y `verify_url`, cuando estén disponibles. Usa los enlaces devueltos. La consulta pública `GET /v1/verificar/{e2e}` verifica un PIX conocido; no es una prueba genérica de toda operación con criptomonedas.

En una devolución, sigue `REFUNDED` y `refund_tx_hash`. El importe y el plazo dependen de la etapa alcanzada y de la devolución del proveedor. No prometas una reversión íntegra, automática ni con un plazo fijo para cualquier fallo. Un depósito tardío o distinto de lo indicado requiere seguir la misma orden; no reutilices su dirección para otra operación.

Billetera de devolución: solo cuando falla la venta

La billetera de devolución solo se solicita cuando falla la venta y el importe debe volver al cliente. Si la orden no tiene `refund_address`, se detiene en `AWAITING_REFUND_ADDRESS` (`refund_address_required: true`) y Lunium envía el `webhook` `cashout.awaiting_refund_address`. Pide a tu cliente una billetera en la red de `refund_network` (en la venta de DePix, una dirección Liquid `lq1...`) e indica:

```
curl --fail-with-body -X POST \
  "https://api.luniumpay.com/cash-outs/$CASHOUT_ID/refund-address" \
  -H "X-API-Key: $LUNIUM_API_KEY" \
  -H 'Content-Type: application/json' \
  -d '{"refund_address": "CARTEIRA_DO_CLIENTE"}'
```

La devolución nunca se envía al origen del depósito: el destino es únicamente la billetera indicada. Una vez registrada, la devolución continúa (`REFUNDING_CRYPTO` en la venta de cripto, `REFUNDING` en la de DePix) hasta `REFUNDED`, con `refund_tx_hash` (`webhooks` `cashout.refunding` y `cashout.refunded`); el caso de DePix ya enviado al proveedor se explica a continuación. Cuando el importe que se devolverá ya está en Lunium, `refund_amount`, `refund_asset` y `refund_network` indican lo que se devuelve. No repitas el pago antes de que termine la devolución.

En la venta de DePix (`asset: "DEPIX", network: "liquid"`), `POST /cash-outs/{cashout_id}/refund-address` responde:

Respuesta	Cuándo
200	Billetera registrada; la orden continúa hacia la devolución (<code>REFUNDING</code>) o, si el DePix sigue en el proveedor, hacia <code>MANUAL_REVIEW</code> . Repetir la misma billetera devuelve 200 sin efecto nuevo
400 <code>refund_address_invalida</code>	Dirección vacía, fuera del formato Liquid o perteneciente a Lunium, como la dirección de depósito de esta u otra orden
409 <code>refund_address_imutavel</code>	La orden ya tiene registrada otra billetera de devolución; no se sustituye
409 <code>devolucao_indisponivel</code>	La orden no está esperando una billetera en este momento: solo se solicita cuando falla la venta y el DePix del cliente está en Lunium. En <code>MANUAL_REVIEW</code> , espera el retorno del proveedor

DePix que ya se envió al proveedor. Si el Pix falló después de enviar el DePix al proveedor de DePix, no se devuelve nada hasta confirmar en la red el retorno del proveedor: hasta entonces, la orden permanece en `MANUAL_REVIEW` (`webhook` `cashout.under_review`), con `refund_amount` vacío, aunque ya tenga una billetera de devolución. Una vez confirmado el retorno, `refund_amount` muestra el importe de la devolución, que como máximo es lo que regresó (sumado a la parte que no llegó a enviarse, hasta el importe de la orden), y la orden continúa hacia `REFUNDING` o, sin billetera, hacia `AWAITING_REFUND_ADDRESS`. Si todo el DePix seguía en Lunium, se devuelve el importe de la orden.

Comprobante oficial del PIX

Toda venta que llega a `COMPLETED` tiene un comprobante oficial del PIX, emitido por Lunium. Es el documento que entregas a tu cliente final y conservas en tu contabilidad. Incluye importe, fecha y hora (hora de Brasilia), el E2E (identificador único del PIX en el Banco Central), la institución y el titular de la clave receptora, el pagador (el liquidador PIX de Lunium, que es como aparece el PIX en los movimientos del destinatario) y el enlace de verificación pública.

El mismo comprobante existe en cuatro formatos, todos incluidos en el cuerpo de la orden:

Campo	Qué es	Úsalo para
<code>pix_e2e</code>	Identificador oficial del PIX en el Banco Central	Conciliar con los movimientos del destinatario; es la prueba válida en cualquier banco
<code>receipt_url</code>	Página del comprobante, que se abre en el móvil	Mostrárselo al cliente en cuanto se complete el PIX
<code>receipt_pdf_url</code>	El comprobante en un archivo PDF	Adjuntarlo a una consulta de soporte, enviarlo por mensaje o archivarlo
<code>verify_url</code>	Verificación pública, sin clave de API (GET <code>/v1/verificar/{e2e}</code>)	Permitir que cualquier persona confirme el importe, la fecha y las iniciales del destinatario sin depender de ti ni de Lunium

Dónde llega el comprobante:

- API: en los campos anteriores, en GET `/cash-outs/{cashout_id}`, en cuanto la orden pasa a `COMPLETED`.
- Webhook: `cashout.completed` incluye los campos completados cuando el enlace ya existe. Si se devuelven `null`, `cashout.receipt_ready` llega después con la orden completa y los enlaces listos. Procesa ambos de forma idempotente mediante el `cashout_id`.
- Panel del socio: en Ventas, abre la venta y usa “Abrir comprobante”, “Descargar el PDF” y “Abrir verificación”. Solo los usuarios con permiso para ver importes ven estos enlaces.
- Telegram: en el grupo de operaciones vinculado a la clave, el aviso “ PIX pagado” incluye el E2E y el enlace “Ver comprobante”, y el PDF llega después como archivo (“ Comprobante oficial”). Si el enlace se genera después del aviso, el PDF llega por separado.

Titular de la clave. El nombre y la institución del destinatario proceden de una única consulta al directorio de claves PIX, realizada por Lunium en el momento del pago y reservada al comprobante: tiene una cuota y no es un servicio de consulta (GET `/pix/keys/lookup` puede responder `verification_status: "not_performed"` sin afectar al comprobante). Si no se encuentra al titular en esa consulta, el comprobante sigue siendo válido, con el importe, la clave, la fecha y la institución extraídos del propio E2E; el nombre se completa después.

Buenas prácticas. Guarda `pix_e2e` y `receipt_pdf_url` junto con tu solicitud. Entrega el PDF al cliente y el `verify_url` a quien necesite verificarlo. No aceptes un comprobante enviado por el cliente como prueba de pago: la prueba es la orden en `COMPLETED` con `pix_e2e`. Los retiros de saldo mediante PIX (POST `/payouts`) también tienen E2E, nombre del destinatario y `verify_url` en GET `/payouts/{payout_id}` y en el panel (Payouts → “Abrir comprobante”). En cash-in, el E2E del PIX recibido se devuelve en GET `/cashin/{cashin_id}/status (e2e y payer_name)`.

Consulta del titular y pago de QR

GET `/pix/keys/lookup?key=...&type=...` es auxiliar. Puede responder con 200 con `available: false, verified: false, verification_status: "not_performed"` y nombres `null`. Muestra «titular no verificado»; no inventes un nombre ni confundas una consulta no disponible con un pago no disponible. En el sandbox, el titular es ficticio y viene con `source: "sandbox"`.

`br_code` tiene una disponibilidad independiente. Producción puede rechazar la creación con `503` `br_code_indisponible`, aunque una vista previa o el sandbox acepten el QR. No ofrezcas el pago del código para copiar y pegar como una función siempre activa. Cuando está habilitado, sustituye `pix_key`, `amount` y `brl_amount`, y requiere un importe definido y un activo compatible. Si el cliente pegó o escaneó un QR con importe -estático o dinámico-, envía el `br_code` completo, nunca `pix_key` + `brl_amount` extraídos de él: solo así se envía el PIX por el importe exacto y como pago de esa solicitud de cobro (con el `txid`). Al pagar por clave, el importe puede variar en centavos y el destinatario no lo reconoce.

La reserva operativa de las nuevas ventas de criptomonedas a Pix es de R\$ 1,00 por operación, para todos los socios, ya incluida en la cotización e indicada en `reserve_brl`. El equivalente en USDT se calcula al tipo de cambio del momento y queda fijado en la orden: `pix_key_reserve_usd` para clave Pix o `copia_e_cola_buffer_usd` para el código para copiar y pegar. Estos campos representan el importe convertido, no un cobro de US\$ 1. No vuelvas a sumar la reserva al total. Los porcentajes y costes del proveedor siguen incluidos en la cotización. El QR se paga solo por el importe exacto. La reserva operativa no se devuelve después del pago; el margen temporal de saldo es independiente. Las órdenes anteriores mantienen los importes contratados. DePix/Liquid sigue su propia tarifa en reales brasileños.

Usa solo las redes ofrecidas en el catálogo de la operación. Las redes con memo pueden quedar excluidas de cash-out aunque existan en el catálogo de entrega; no copies la lista de un producto a otro.

Billetera USDT/USDC: saldo en crypto y PIX pagado con él

La billetera mantiene un saldo en USDT de tu proyecto. Depositas USDT o USDC en varias redes y pagas PIX con ese saldo, mediante clave PIX o Copia y Pega. Está habilitada para todas las claves de producción; la clave `lun_test_` usa el simulador.

Billetera USDT/USDC: depósitos y pagos por Pix

Deposita USDT o USDC y usa el saldo para pagar Pix mediante la API o el panel.

A • Depósito y acreditación

1. Crear solicitud de depósito

`POST /carteira-usdt/deposits`

Indica el importe y la red.



2. Revisar instrucciones de depósito

`waiting`

Obtén el importe exacto, la dirección y el vencimiento de 1 hora. El QR de la dirección no incluye el importe.



3. Enviar por la red seleccionada

`expired (sin envío en 1 hora)`

Envía el importe exacto en USDT o USDC antes del vencimiento.



4. Esperar confirmaciones de red

`confirming`

El depósito se acredita automáticamente tras las confirmaciones de red exigidas.



5. Saldo disponible

`available`

`wallet.deposit_credited`

USDC se acredita en el saldo de la billetera en proporción 1:1.

B • Pagar Pix con el saldo

1. Consultar los USDT necesarios (opcional)

`POST /carteira-usdt/quote`

Consulta la cotización sin afectar el saldo.



2. Solicitar el pago y reservar USDT

`reservado`

`POST /carteira-usdt/payments`

Introduce una clave Pix o el código Pix completo. Usa `external_id` para la idempotencia. La reserva de USDT cubre hasta un 1% de variación; el excedente se libera tras la liquidación.



3. Seguir el pago original

`GET /carteira-usdt/payments/{payment_id}`

Consulta el avance con el mismo `payment_id`.



4. Pago completado

`concluido`

`wallet.payment_completed`

Comprobante y E2E: `receipt_url` y `pix_e2e`.



5. Pago fallido

`falhou`

`wallet.payment_failed`

Si el Pix no se ejecuta, la reserva vuelve al saldo disponible.

`revisao` indica revisión; `falhou_apos_envio` indica un resultado pendiente tras el envío.

Ninguno confirma por sí solo un fallo definitivo.

Referencia: 9 redes; depósito mínimo de 5 USDT/USDC y Pix mínimo de R\$ 25. Prevalecen el catálogo y la cotización vigentes. El depósito vence en 1 hora. Las direcciones EVM coinciden: revisa la red.

Manual completo: docs.luniumpay.com/es/manual#carteira-usdt

1. `GET /carteira-usdt: saldo` (`available`, `reserved`, `confirming`), redes aceptadas actualmente (`networks[]`, con monedas y plazo estimado), `min_deposit` y `min_payment_br1`.

2. `POST /carteira-usdt/deposits` CON `{ "amount": "100", "network": "tron" }`: devuelve `amount_to_send` (importe exacto: los últimos decimales identifican tu depósito), `address`, `network_name` y `expires_at`. Muestra los cuatro al

usuario. Si el origen descuenta la tarifa de envío, súmala: lo que cuenta es el importe que llega. La solicitud dura 1 hora (`deposit_validity_hours = 1`; `expires_at = creación + 1 h`): envía dentro del plazo.

3. La acreditación es automática: sigue `GET /carteira-usdt/deposits` (`waiting` → `confirming` → `available`) o el webhook `wallet.deposit_credited`. USDC se acredita en el saldo en proporción 1 a 1.

4. `POST /carteira-usdt/quote` con `{"brl_amount": "50.00"}` o `{"br_code": "000201..."}`: vista previa en USDT (`usdt_total`, `usdt_reserved`), sin efecto.

5. `POST /carteira-usdt/payments` con `{"brl_amount": "50.00", "pix_key": "cliente@ejemplo.com", "pix_key_type": "email", "external_id": "pedido-123"}` (o con `br_code`): responde 201 en reservado. Sigue el `GET /carteira-usdt/payments/{payment_id}` hasta `final: true`, o los webhooks `wallet.payment_completed` y `wallet.payment_failed`.

6. Al completarse incluye `pix_e2e`, `recipient_name`, `recipient_document` (CPF parcial) y `receipt_url` (comprobante oficial). `GET /carteira-usdt/statement` es el registro de movimientos: importe siempre positivo, `type` indica la dirección (`deposit`, `reserve`, `debit`, `release`, `reversal`).

Reglas: el PIX mínimo es `min_payment_brl` (actualmente R\$ 25,00). La reserva cubre hasta un 1% de variación y el sobrante vuelve al saldo. `falhou` = no ejecutado, la reserva se devolvió; `falhou_apos_envio` = en investigación, la reserva no se devuelve automáticamente; `revisao` es una revisión, no un fallo. El mismo `external_id` con los mismos datos devuelve el mismo pago (200); con datos distintos, 409 `external_id_divergente`. 409 `saldo_insuficiente` cuando el disponible no cubre la reserva.

Redes aceptadas (9): USDT y USDC en Polygon, BNB Smart Chain (BEP20), Ethereum (ERC20), Arbitrum One, Optimism, Avalanche C-Chain y Solana; solo USDT en Tron (TRC20); solo USDC en Base. Lo que se aplica a tu clave actualmente es `networks[]` de `GET /carteira-usdt`. Depósito mínimo: 5 USDT o USDC (`min_deposit`). Sin un envío dentro del plazo, la solicitud termina en `expired`. El MCP de Lunium tiene 6 herramientas de la billetera: saldo, solicitud y lista de depósitos, vista previa, pago y consulta del pago.

En el panel del socio, la solicitud de depósito muestra el código QR solo de la dirección y una cuenta atrás, con segundos, hasta `expires_at`. El QR no incluye el importe: introduce el importe exacto (`amount_to_send`) en la aplicación de envío. En las redes EVM, la dirección `0x` es la misma: elige la red correcta en la aplicación.

En `sandbox` (`lun_test_`): dirección ficticia (no envíes nada), depósito acreditado automáticamente en unos 15 s y PIX completado en unos 15 s, con E2E de prueba. Los centavos eligen el escenario: depósito `.02` caduca; PIX `.01` pasa por `revisao`, `.02` falla y devuelve la reserva, `.05` tarda unos 2 minutos.

9. Operativa a través de Telegram

Lunium Partners: comunidad oficial de integradores

[Unirse a Lunium Partners](#)

El Lunium Partners forma parte del seguimiento recomendado para quienes desarrollan con la API. Incluye a los responsables técnico y operativo del socio desde el principio y mantén al equipo atento a los avisos durante la operación en producción. La participación es muy recomendable, sin ser una condición técnica para crear claves o usar la API.

En la comunidad, sigue los avisos operativos, mantenimientos y cambios de API/contrato, además del intercambio entre integradores. No esperes a que ocurra un incidente para unirse: una integración necesita seguir los cambios después de publicarse.

Canal	Finalidad	Quién participa
Lunium Partners	Comunidad oficial, avisos e intercambio entre socios	Responsables técnico y operativo de los socios
Grupo dedicado al proyecto	Seguimiento de tu integración e investigación de casos específicos	Tu equipo, las personas @luniumb2b e @luniumb2c y el @LuniumNotifyBot; el grupo puede crearlo Lunium o el socio
Monitor privado y webhooks	Seguir operaciones y actualizar tu sistema	Equipo autorizado y backend del socio; consulta la guía de seguimiento en vivo

Para empezar: únete a Lunium Partners, coordina el grupo dedicado con los dos contactos humanos, vincula el bot y valida el monitor/webhooks en el entorno correcto. La comunidad no sustituye al soporte específico del proyecto ni a la conciliación financiera de la API.

No publiques claves de API, secretos de webhook, `monitor_url`, documentos de pagadores ni datos de transacciones de clientes en la comunidad. Lleva los casos específicos al grupo dedicado, con IDs y datos minimizados.

Integración con Fable u otra IA: incluye el enlace de Lunium Partners en el plan y en la lista de comprobación entregada al responsable humano. La IA debe explicar esta recomendación y distinguir los tres canales anteriores. No afirmes que alguien ya se ha unido al grupo sin confirmación, no exijas unirse para ejecutar pruebas de sandbox y no uses mensajes de Telegram como autorización para mover dinero.

Grupo de integración con acompañamiento humano

Se recomienda mantener un grupo de Telegram dedicado al proyecto, con los agentes humanos @luniumb2b y @luniumb2c. Lunium puede crear el grupo e invitar al equipo del socio, o el equipo del socio puede crearlo y añadir a los dos contactos. El grupo acompaña la integración, la validación y la operación en producción. No es un requisito técnico para usar la API.

1. Crea el grupo o solicita su creación a uno de los contactos anteriores. Usa un nombre que identifique el proyecto.
2. Incluye a los dos contactos humanos y a los responsables técnico y operativo del socio. Acuerda quiénes son los responsables y el canal de escalado.
3. Añade también a @LuniumNotifyBot y realiza la vinculación que se explica a continuación. El bot envía notificaciones y ejecuta comandos; los contactos humanos atienden dudas e incidentes. Ninguno sustituye al otro.
4. Fija en el grupo los enlaces del manual, del estado público y las instrucciones operativas. Comparte el monitor privado solo con integrantes autorizados para ver los datos del proyecto.
5. Durante la validación, comprueba el proyecto vinculado, el monitor, un webhook firmado y el escenario de fallo/recuperación del flujo elegido.

Para pedir ayuda, indica el entorno (sandbox o producción), el ID de la operación, `external_id`, la hora con zona horaria, el comportamiento esperado y el observado, y `request_id` cuando esté disponible. Para las notificaciones,

añade `event_id` y `X-Lunium-Delivery-Id`. No envíes claves de API, secretos de webhook, documentos completos del pagador ni datos de otros clientes. La presencia en el grupo no implica atención las 24 horas ni un plazo de liquidación; acuerda la atención con el equipo.

Los canales generales de socios/comunidad indicados en `/keys/me` complementan los avisos de mantenimiento y cambios de API. No sustituyen al grupo dedicado al proyecto.

Vinculación y comandos del bot

Añade [@LuniumNotifyBot](#) al grupo del proyecto. Un administrador identificado debe usar `/vincular` y el flujo de vinculación recibido al crear/consultar la clave. Si el bot muestra un botón de confirmación, pulsa el botón: responder “sí” por texto no equivale a pulsarlo. No publiques la clave `lun_...` en el grupo.

Comando	Función
<code>/configurar</code>	Comprobar la vinculación y la configuración
<code>/cobrar <valor BRL> <cpf/cnpj> [cliente=ref]</code>	Generar PIX para custodia, sin billetera fija
<code>/saldo [customer_ref]</code>	Consultar el saldo del ámbito indicado
<code>/extratosaldo [customer_ref]</code>	Consultar los movimientos de la custodia
<code>/moedas [moeda]</code>	Ver las rutas disponibles para la entrega
<code>/sacarcripto <valor BRL> <moeda> <rede> <endereço> <cpf/cnpj> [memo] [cliente=ref]</code>	Estimar y confirmar un retiro de la custodia en cripto
<code>/pagar <valor BRL> <chave PIX> [tipo] [cpf/cnpj]</code>	Solicitar un PIX debitado del saldo del proyecto
<code>/status <id></code>	Seguir una operación
<code>/limites</code>	Consultar los límites del proyecto y las reglas de los pagadores
<code>/monitor</code>	Abrir el seguimiento del proyecto
<code>/comissao [taxa%]</code>	Consultar la comisión o presentar un cambio para que lo confirme el administrador; guía

En Telegram, los importes se expresan en reales brasileños, como 50 o 50,50; en la API, `amount_cents` usa centavos. Para las subcuentas, comprueba el ámbito mostrado antes de retirar. `/pagar` no ofrece el parámetro `cliente=ref` del retiro en cripto; para un payout de subcuenta, usa la API con `customer_ref`.

El retiro en cripto muestra el destino, la estimación y el débito antes de confirmar. La confirmación dura 60 segundos y pertenece al usuario, al grupo y al proyecto que la solicitaron. En los grupos, los retiros requieren un administrador identificado; un mensaje sin remitente identificado no autoriza un retiro. La confirmación vuelve a validar las condiciones y puede solicitar una nueva vista previa. La cotización no queda fijada.

`/sacar` es el flujo de cripto externa a PIX. Para retirar el saldo en cripto, usa `/sacarcripto`. Los comandos `/qr`, `/qrwhitelist` y `/qrdelay` también usan custodia cuando no hay una billetera explícita; el plazo y la retención deben comprobarse en la respuesta.

10. Webhooks fiables

Configura `webhook_url`, guarda `webhook_secret` y ejecuta `POST /webhooks/test` para probar tu receptor. Esta llamada envía una notificación de prueba a la dirección configurada.

Sobre ilustrativo:

```
{
  "event": "cashin.settled",
  "event_id": "evt_exemplo",
  "created_at": "2026-09-17T20:00:00.000Z",
  "data": {
    "cashin_id": "ci_exemplo",
    "destino": "saldo",
    "status": "paid",
    "settlement_status": "sent"
  }
}
```

El ejemplo abrevia data. Los eventos de cobro, cash-out y payout comparten el objeto base de estado del recurso; los enriquecimientos de una consulta GET, como `timeline`, pueden no estar en el evento. Los eventos `saldo.*` tienen sus propios datos de movimiento. No exijas que todos los webhooks sean idénticos, campo por campo, a la respuesta enriquecida de un GET.

Cabecera	Uso
X-Lunium-Signature	t=<unix>, v1=<HMAC-SHA256 em hex> sobre timestamp.corpo_bruto
X-Lunium-Event-Id	Identificador estable del evento, para deduplicar
X-Lunium-Delivery-Id	Identificador del intento, para diagnóstico
X-Lunium-Attempt	Número del intento

Valida el cuerpo sin procesar, sin volver a serializar el JSON, y la antigüedad del timestamp. Compara buffers del mismo tamaño en tiempo constante. Usa el [verificador Node.js completo](#), que también rechaza las firmas mal formadas sin lanzar un error de tamaño del buffer.

Guarda/encola el evento y responde 2xx rápidamente. El transporte tiene un timeout de 10 segundos. Realiza el trabajo lento después. No deduplicas por intento: todos los intentos del mismo evento deben producir una única acción de negocio.

Sin éxito, hay hasta 12 intentos con espera progresiva. La ventana es de unas 7 horas, sujeta a la cola; no es un SLA de entrega. Consulta `GET /webhooks/deliveries` y reenvía mediante `POST /webhooks/deliveries/{event_id}/retry` cuando corrijas tu receptor. Una entrega que agota los intentos queda `failed`; el registro permite recuperarla.

Las integraciones nuevas deben usar la firma v2. `X-Webhook-Signature` es antiguo, no está presente en las claves nuevas y su retirada está anunciada para el 09/11/2026; no dependas de él.

Eventos

Grupo	Eventos
cashin	cashin.paid • cashin.settled • cashin.settlement_failed • cashin.expired • cashin.refunded • cashin.failed • cashin.delayed • cashin.held
cashout	cashout.awaiting_deposit • cashout.deposit_detected • cashout.processing • cashout.confirming • cashout.under_review • cashout.awaiting_refund_address • cashout.refunding • cashout.completed • cashout.refunded • cashout.expired • cashout.failed • cashout.receipt_ready
payout	payout.sent • payout.failed • payout.refunded
saldo	saldo.liberado • saldo.congelado • saldo.transferido
webhook	webhook.test

Admite eventos y campos nuevos, registra lo que no reconozcas y confirma el recurso mediante GET cuando sea necesario. No presupongas el orden de llegada. Consulta el estado actual antes de hacer retroceder un recurso ya completado.

Monitor en vivo del socio

El socio ya puede seguir su propio proyecto en el monitor privado de Lunium. Consulta GET /keys/me en el backend y abre el monitor_url de la respuesta, o usa /monitor en el grupo vinculado. La creación de la clave también devuelve ese enlace. Ofrece acceso de lectura al proyecto: compártelo solo con personas autorizadas, nunca en la documentación pública ni con el cliente final.

El panel consulta el feed cada 3 segundos y el saldo de custodia cada 15 segundos. Muestra solicitudes de cobro, cash-outs, payouts y entregas de webhook; la sección de custodia incluye saldo disponible/bloqueado, subcuentas y movimientos recientes cuando está habilitada. Estos intervalos son la frecuencia de consulta de la interfaz, no un plazo garantizado de actualización del banco, proveedor o blockchain.

Si se filtra el enlace privado, usa PATCH /keys/me con {"rotate_monitor_token":true}. Guarda el nuevo monitor_url: el token anterior deja de dar acceso a los datos. El token del monitor es de solo lectura, pero permite ver el proyecto completo, no solo un cliente. No lo uses como autenticación de una pantalla pública por transacción.

Feed para tu propio panel

```
# Ejecuta en el backend; LUNIUM_API_KEY procede de tu gestor de secretos.  
curl --fail-with-body 'https://api.luniumpay.com/cashin/events?limit=80' \  
-H "X-API-Key: $LUNIUM_API_KEY"
```

GET /cashin/events acepta X-API-Key o X-Dashboard-Token en la cabecera. El monitor alojado usa ?token=. La respuesta contiene proyecto, ahora y eventos. Cada elemento incluye ts, flujo (cash-in, cash-out, payout o webhook),

proveedor, ref, valor, estado, detalhe y tx, con campos que pueden ser nulos. `limit` tiene un valor predeterminado de 60 y un límite de 1 a 200.

Es un resumen reciente, no un historial completo de transiciones. La lista se ordena por `ts` en orden descendente; una operación puede aparecer con su estado actual y el feed no tiene cursor de repetición ni el `event_id` del sobre del webhook. No calcules saldos, tarifas ni conciliaciones a partir de los textos `valor/estado`. No uses `ts` como secuencia global de cambios. Para importes, red, hash y comprobantes, consulta el recurso original. En particular, usa `settlement_tx_url` devuelto por el estado del cash-in para acceder al explorador correcto de la red, en lugar de suponer Polygon por el resumen del feed.

Webhook + consulta de estado + actualización de tu interfaz

El webhook es una llamada HTTPS POST de Lunium al backend del socio. Para reproducir la experiencia en vivo en tu aplicación, implementa este recorrido:

1. Recibir: lee el cuerpo sin procesar con un límite de tamaño y valida la firma y el timestamp antes de interpretar el JSON. Comprueba que el `event_id` del cuerpo coincida con la cabecera. Un secreto corresponde a la configuración de la clave/entorno; no confíes en el payload para elegir el proyecto.
2. Guardar: guarda el evento en una bandeja de entrada persistente, con una restricción única (`projeto`, `ambiente`, `event_id`), antes de responder 2xx. Ante un duplicado ya guardado, devuelve 2xx sin repetir el efecto. Si falla el guardado, devuelve un error para permitir otro intento. No descartes silenciosamente un evento autenticado desconocido.
3. Conciliar: un worker procesa la bandeja de entrada y consulta el recurso canónico cuando es necesario. Serializa las actualizaciones por operación para que una respuesta antigua no sobrescriba una consulta más reciente. Guarda por separado el historial recibido, el estado consultado y la hora de la última consulta.
4. Actualizar la pantalla: tu backend publica el resultado mediante SSE, WebSocket o consultas periódicas para las sesiones autenticadas de tu aplicación. Autoriza el proyecto y el cliente en cada lectura/conexión. El socio implementa estos canales; este contrato no describe ningún endpoint público SSE/WebSocket de Lunium.
5. Recuperar: mantén una rutina de conciliación de las operaciones sin completar y de aquellas con entrega incierta. Usa un intervalo ajustable, espera progresiva y los límites de la clave; no hagas una consulta a la API por cada navegador abierto. Cuando el cliente se reconecte, vuelve a cargar el estado persistido y muestra cuándo la actualización esté atrasada.

El monitor de Lunium es la vista operativa lista para usar. El panel del socio puede añadir sus propios pedidos, clientes e historial persistente. Los webhooks no sustituyen a los movimientos del saldo ni constituyen un registro de cada etapa interna del exchange asociado, de los bancos o de otros proveedores.

Operación / vínculo	Consulta para comprobar el estado	Cómo mostrar la finalización
Solicitud de cobro PIX o retiro de custodia en cripto: <code>cashin_id</code>	GET <code>/cashin/{cashin_id}/status</code>	<code>status=paid</code> y <code>settlement_status=sent</code> ; identifica destino/fonte para distinguir entre acreditación BRL y entrega de cripto
De cripto externa a PIX: <code>cashout_id</code>	GET <code>/cash-outs/{cashout_id}</code>	<code>state=COMPLETED</code> ; muestra el comprobante/E2E cuando se devuelvan

Operación / vínculo	Consulta para comprobar el estado	Cómo mostrar la finalización
Payout PIX: payout_id	GET /payouts/{payout_id}	status=sent; no lo confundas con la entrega del webhook sent
Movimiento saldo. *: customer_ref, origem_id, charge_id, movimiento_id	GET /saldo y GET /saldo/extrato, con el customer_ref correcto cuando exista	El saldo disponible procede del ledger; un crédito registrado y un saldo liberado son cosas distintas

Guarda también el `external_id` que definiste al crear la operación. El estado del cash-in puede incluir `timeline` con `etapa`, `em`, `detalle`; el del cash-out, `timeline` con `state`, `at`. Usa únicamente las etapas y horas realmente devueltas. Los campos de confirmación, hash o comprobante pueden faltar/ser nulos hasta que existan. No inventes porcentajes, horas de finalización, transacciones on-chain ni motivos de fallo.

Qué significa cada evento en la pantalla

Eventos	Interpretación y acción
<code>cashin.paid</code>	PIX confirmado; aún debe comprobarse la entrega/acreditación. Nunca muestres “cripto entregada” solo por este evento
<code>cashin.settled</code>	Liquidación notificada; confirma el destino y el estado. En custodia, consulta cuánto está disponible para retirar
<code>cashin.delayed</code> , <code>cashin.held</code>	Retraso/retención; muestra de <code>lay_until</code> /liberar_em cuando se devuelvan, sin tratarlo como un fallo definitivo
<code>cashin.settlement_failed</code> , <code>cashin.failed</code>	Problema de liquidación u operación; consulta el estado actual, registra el error y escala el caso. No crees otro pago automáticamente
<code>cashin.expired</code> , <code>cashin.refunded</code>	Caducidad o devolución; concilia la solicitud de cobro original
<code>cashout.awaiting_deposit</code> , <code>cashout.deposit_detected</code> , <code>cashout.confirming</code>	Esperando depósito, detectado o en confirmación; muestra el recuento solo cuando se proporcione
<code>cashout.processing</code>	Procesamiento agregado. Varias etapas internas comparten este evento; no se emite para cada etapa
<code>cashout.under_review</code> , <code>cashout.refunding</code>	Revisión o devolución en curso; mantén el seguimiento
<code>cashout.awaiting_refund_address</code>	La venta falló y debe devolverse, pero la orden no tiene billetera de devolución. Pide la billetera al cliente e indícala en <code>POST /cash-outs/{cashout_id}/refund-address</code> ; no repitas el pago antes de la devolución
<code>cashout.completed</code> , <code>cashout.refunded</code> , <code>cashout.expired</code> , <code>cashout.failed</code>	Resultado notificado del cash-out; comprueba el recurso y los comprobantes disponibles
<code>cashout.receipt_ready</code>	El comprobante del PIX está listo. Se emite después de <code>cashout.completed</code> (el enlace se genera unos segundos después del pago, mientras se consulta al titular de la clave) e incluye la orden completa con <code>receipt_pdf_url</code> , <code>receipt_url</code> y <code>verify_url</code> completados. Es el evento que debes usar para entregar el comprobante a tu cliente sin consultas repetidas
<code>payout.sent</code> , <code>payout.failed</code> , <code>payout.refunded</code>	Resultado del PIX enviado con saldo; confirma la operación y los movimientos
<code>saldo.liberado</code> , <code>saldo.congelado</code> , <code>saldo.transferido</code>	Cambio en el saldo; consulta el ámbito correcto. No sumes directamente los eventos como si fueran movimientos del saldo
<code>webhook.test</code>	Prueba de transporte/firma; no representa dinero ni una operación financiera

`settlement_status=incerto` requiere conciliación antes de cualquier entrega nueva. Un webhook antiguo puede llegar después de una consulta que ya muestra la finalización; guárdalo en el historial sin hacer retroceder el estado actual. Incluso una operación completada puede tener una actualización posterior legítima, como una devolución: consulta la fuente en lugar de ignorarlo todo después del primer éxito.

Configurar, probar y recuperar notificaciones

Configura el receptor HTTPS público en la clave del entorno correcto. La validación rechaza destinos privados/locales. El secreto se devuelve al generarse: guárdalo en el backend. Una actualización de URL no genera ni devuelve necesariamente otro secreto. Si necesitas rotarlo, coordina el cambio usando `rotate_webhook_secret`; los intentos posteriores usan el secreto vigente.

```
curl --fail-with-body -X PATCH 'https://api.luniumpay.com/keys/me' \
-H "X-API-Key: $LUNIUM_API_KEY" -H 'Content-Type: application/json' \
-d '{"webhook_url":"https://seu-dominio.example/webhooks/lunium"}'

# Prueba solo el transporte y la firma; no crea un pago.
curl --fail-with-body -X POST 'https://api.luniumpay.com/webhooks/test' \
-H "X-API-Key: $LUNIUM_API_KEY"

curl --fail-with-body 'https://api.luniumpay.com/webhooks/deliveries?status=failed&limit=100' \
-H "X-API-Key: $LUNIUM_API_KEY"

# Reenvía solo después de corregir el receptor, usando un event_id de tu clave.
curl --fail-with-body -X POST \
  "https://api.luniumpay.com/webhooks/deliveries/$EVENT_ID/retry" \
-H "X-API-Key: $LUNIUM_API_KEY"
```

Sustituye el dominio ilustrativo por el receptor real. `POST /webhooks/test` tiene un límite de 6 llamadas/minuto por clave en producción; devuelve `ok`, `status_code`, `ms`, `event_id` y `delivery_id`. Esta prueba aislada no se guarda en la cola de entregas de producción. Valida la comunicación, no la liquidación financiera ni todos los recorridos de tu aplicación.

`GET /webhooks/deliveries` acepta `status` (`pending`, `sent`, `failed`), `event` y `limit` (predeterminado 50, máximo 200). Cada fila incluye `event_id`, `operation_id`, `attempts`, `last_error`, `url`, `next_attempt_at`, `created_at` y `updated_at`, además del evento/estado. `failed_count` cuenta los fallos solo en las filas devueltas. No hay filtro por `operation_id` ni cursor de paginación en este endpoint; conserva tu historial persistente y busca la operación en las filas recibidas.

El reintento responde 200 si se entregó, 202 si falló y volvió a la cola, 409 si ya estaba entregado y 404 si no existe para la clave. `pending/sent/failed` aquí describen el transporte del webhook, no el resultado del pago. Reenviar una notificación no repite el pago ni corrige una liquidación pendiente.

Validación del seguimiento en vivo

- Configura el receptor en la clave de `sandbox` y ejecuta los escenarios aplicables descritos en [Sandbox](#). `GET /sandbox/events` muestra los eventos simulados; no uses el feed operativo de producción como prueba de cobertura del `sandbox`.
- Comprueba firmas no válidas/caducadas, duplicados, eventos fuera de orden, indisponibilidad del receptor, retorno tras reconectar y aislamiento entre proyectos/clientes.
- Demuestra que `cashin.paid` no marca la entrega de cripto, que una retención no se convierte en saldo disponible y que `webhook.test` no altera el saldo.
- Comprueba una operación mediante la respuesta de creación, el webhook y el GET de estado, correlacionando los IDs y sin crear una transacción nueva ante un timeout.
- Valida los errores de liquidación, el resultado incierto, la devolución y la recuperación de notificaciones. Envía los IDs al grupo humano para los casos que requieran investigación.

- Muestra en el panel: operación, pedido externo, cliente autorizado, moneda/red, importe devuelto, estado financiero, entrega del webhook, última actualización y comprobante/hash cuando estén disponibles. Señala los datos desactualizados y evita una falsa indicación de “en vivo” .

11. Errores, idempotencia y conciliación

Usa el código HTTP junto con `erro`, `acao`, `detail`, `request_id` y `agent_guidance` cuando estén presentes. Algunas rutas devuelven solo parte de estos campos. No decidas la lógica a partir del texto libre de `detail`.

Recomendación	Actuación
corregir	Ajustar campos o importe; leer <code>limits</code> , <code>payer_limits</code> y <code>min_brl_cents</code>
repetir	Reintentar con espera progresiva y el mismo identificador/payload
esperar	Respetar <code>Retry-After</code> , el plazo de liberación o el estado del servicio
parar / faltar_com_suporte	Interrumpir los intentos automáticos y resolver la condición
401 / 403	Comprobar la autenticación, la capacidad de la clave, el límite y la política de la operación
402 saldo_insuficiente	Comprobar el disponible, la subcuenta y las tarifas del retiro
409 external_id_divergente	El ID ya pertenece a otra operación; no reutilizarlo para cambiar la solicitud
423 carencia_primeiro_deposito	Mostrar el plazo devuelto; no eludirlo mediante otra subcuenta
429	Esperar a que se renueve el límite; reducir las consultas simultáneas
5xx o timeout después del envío	Consultar la operación por ID/referencia antes de repetir

Envía `external_id` de hasta 64 caracteres en toda operación financiera que lo admita. Usa un identificador persistente por intención de negocio y conserva todos los parámetros iguales en los reintentos. Las solicitudes de cobro y los retiros en cripto comparten el espacio de referencias de la clave; usa prefijos distintos. No uses la validación de discrepancias para modificar una operación existente: la comparación varía según la ruta.

Para recuperar tras un timeout, consulta `/cashin/charges?external_id=...`, `/cash-outs?external_id=...` o `/payouts?external_id=...`, según la operación. Si necesitas repetir, conserva el mismo cuerpo e identificador. No generes un ID nuevo solo porque perdiste la respuesta.

Guarda los IDs, importes, moneda/red, `customer_ref`, timestamps, eventos procesados y referencias de los movimientos. Usa centavos enteros para BRL y cadenas decimales para cripto. Las fechas incluyen una zona horaria explícita. El redondeo y la conversión deben seguir los valores devueltos por la API.

Procedimiento operativo: qué consultar y mostrar

Estado	Verificación y acción	Mensaje para el cliente
PIX pagado, cripto pendiente	Consulta GET <code>/cashin/{id}/status</code> . paid con liquidación pending, sending o incerto no es una entrega confirmada. Conserva el ID y haz el seguimiento; no generes otro pago. Ante un fallo, un resultado incierto o una ausencia persistente de progreso, envía el ID a soporte.	“Pago recibido; entrega en seguimiento.” Indica la última actualización, sin inventar un plazo.
Crédito en custodia aún no disponible	Consulta el estado y GET <code>/saldo?customer_ref=...</code> en el mismo ámbito de la solicitud de cobro. Comprueba lo retenido/disponible y la fecha devuelta. La retención del proveedor, la retención local y el período de espera son condiciones distintas.	“Importe recibido; disponibilidad según las condiciones y el plazo indicados.” No muestres el saldo total como retirable.
No llegó el webhook	Consulta el estado canónico y GET <code>/webhooks/deliveries</code> . Comprueba la URL, el entorno, la firma, el timeout y el HTTP del receptor. Corrige el receptor antes de reenviar el <code>event_id</code> existente. La ausencia en la cola no demuestra que el pago haya fallado.	Usa el estado confirmado por la consulta de la operación; el aviso retrasado no cambia la situación financiera.
Respuesta perdida después de POST	Busca la operación con el <code>external_id</code> guardado en la lista del flujo. Si la encuentras, haz su seguimiento. Si aún no aparece, concilia con espera progresiva y conserva el cuerpo/ID; una ausencia inmediata no demuestra que la solicitud no haya sido aceptada.	“Solicitud en verificación.” No pedir otro PIX ni crear otra intención automáticamente.
Ruta/moneda no disponible o mínimo no alcanzado	Actualiza el catálogo correcto y lee error, acabo y el mínimo devuelto. Antes del pago, pide al cliente otra elección. Después del pago, sigue la operación y escala el caso; no cambies silenciosamente la red, moneda ni dirección.	Explica la restricción específica y las opciones verificadas, sin prometer otra ruta.
No apareció la comisión	Comprueba la tarifa guardada en la solicitud de cobro, la finalización de la liquidación y los movimientos de la comisión. Distingue entre retraso de cálculo, tarifa cero y reversión. Lleva la discrepancia a soporte con el ID original.	“Comisión en revisión.” No prometas una acreditación retroactiva por cambiar la configuración.

En cualquier flujo, un panel con datos antiguos debe mostrar la “última actualización” y el estado de conexión. No conviertas un timeout, una respuesta vacía ni la ausencia de un webhook en un éxito o fallo financiero. Una operación en estado incierto requiere conciliación; nunca envíes manualmente otra transferencia solo para eliminar la alerta.

Al contactar con el grupo humano, envía el entorno, el ID de la operación, `external_id`, la hora con zona horaria, el código HTTP, `erro/acao`, `request_id` cuando esté disponible y los últimos estados consultados. Para un webhook, añade `event_id` y el HTTP/hora del intento, con registros depurados. No envíes claves, secretos, monitores privados, CPF completos ni payloads con datos de clientes a la comunidad abierta. Este manual no garantiza un plazo universal de soporte ni de liquidación.

Fable puede seguir el [procedimiento estructurado de incidentes](#). Orienta las consultas y la conciliación; no concede autorización para retirar, cambiar tarifas ni reenviar dinero.

12. Sandbox y pruebas de integración

Ejecuta `node sandbox.mjs all` o `python3 sandbox.py all` en el [kit](#). Los ejemplos confirman la clave de prueba, simulan el pago PIX y recorren custodia, retiro USDC/Base, payout, BTC, USDC/Polygon y cash-out. Para cash-in/payout, usa `sandbox_scenario`; la retención, la devolución, el fallo de entrega y el resultado incierto se detallan en la [guía](#). `GET /sandbox/events` permite examinar los eventos; los callbacks configurados usan la cola firmada del producto. Los precios y plazos son sintéticos, no condiciones reales de la plataforma de intercambio asociada.

Billetera USDT/USDC: la clave de prueba usa el simulador de la billetera, con las mismas rutas y estructuras. El depósito `.02` caduca; PIX `.01` pasa por revisión, `.02` falla y devuelve la reserva, `.05` es lento. Más detalles en la sección Billetera USDT/USDC.

En el cash-out del sandbox, los dos primeros decimales de `amount` seleccionan el escenario:

Terminación del importe	Escenario
<code>.00</code>	Caso de éxito simulado
<code>.01</code>	Retención/retraso simulado; finalización posterior
<code>.02</code>	Fallo
<code>.03</code>	La cotización caduca en 5 segundos
<code>.04</code>	Rechazo por límite, con <code>limits</code>
<code>.05</code>	Finalización lenta, aproximadamente 2 minutos
<code>.06</code>	Rechazo del proveedor, <code>pagador_recusado_pe_lo_proveedor</code>
<code>.07</code>	La primera aceptación devuelve 503; una nueva aceptación de la misma orden funciona
<code>.08</code>	Rechazo por límite del pagador

Las órdenes de prueba son temporales y no representan depósitos reales. El sandbox no certifica plazos, precios, disponibilidad del proveedor ni abonos en custodia de producción. Para cash-in y saldo, prueba las rutas de simulación y los escenarios documentados. La prueba piloto real posterior usa el mínimo admitido por la ruta y el pagador, con un importe y un destinatario autorizados.

13. Dónde consultar la regla vigente

Dato	Fuente de la API
Capacidades, límite por operación y límite de frecuencia de la clave	GET /keys/me
Límite del pagador y tramos de retención	GET /cashin/limits?payer_tax=...
Disponible, bloqueado, período de espera y tarifas de custodia	GET /saldo en el ámbito de la cuenta
Monedas/redes para entregar cripto	GET /cashin/catalog con entregavel: true
Monedas/redes para recibir cripto externa	GET /catalog y, cuando corresponda, /catalog/dex
Conversión de PIX a cripto y mínimo de la ruta	POST /cashin/preview
Conversión de cripto externa a PIX	POST /cash-outs/preview y cotización creada
Validez real de una solicitud de cobro/cotización	expires_at de la propia operación
Liberación, finalización y comprobantes	GET de estado, webhook y movimientos

GET /keys/me no contiene por sí solo todas las condiciones. Un límite diario null significa que la clave no tiene ese límite; siguen aplicándose los límites por pagador, destinatario, operación, saldo y eventual mandate. No programes cifras de catálogo, tarifas, plazos ni límites a partir de un ejemplo de este manual.

14. Comisión, agentes y operaciones adicionales

Configurar la comisión del socio

La comisión adicional es la comisión del socio, que se añade a la tarifa de Lunium. 100 bps = 1%; 200 bps = 2%. Consulta la política y el máximo permitido en GET /comissao antes de configurarla. El ejemplo siguiente modifica el precio de las nuevas solicitudes de cobro de la clave; aplica solo la tarifa acordada con el responsable del proyecto.

```
curl --fail-with-body -X PATCH https://api.luniumpay.com/keys/me \
  -H "X-API-Key: $LUNIUM_API_KEY" \
  -H 'Content-Type: application/json' \
  -d '{"partner_fee_bps":200}'
```

Comprueba el valor guardado en GET /keys/me. La tarifa se registra en cada solicitud de cobro nueva: cambiar la configuración no recalcula las abiertas ni crea comisiones retroactivas. En Telegram, /comissao consulta; /comissao 2 presenta el cambio y su confirmación. El administrador debe comprobar la identidad de la clave y usar el comando de confirmación devuelto por el bot, incluido el ID de la clave. No confirmes en nombre de otro proyecto.

Separar tarifa, comisión, tipo de cambio y cantidad entregada

En cash-in, la base es el importe recibido menos la tarifa fija de Lunium. Sobre esa base se aplican los porcentajes de Lunium y del socio. En el depósito en saldo, solo se descuenta R\$ 1,00: no hay porcentaje, tramo ni comisión. Los parámetros de la solicitud de cobro y los importes devueltos por la operación prevalecen sobre los ejemplos.

Ejemplo didáctico en centavos, que no representa una oferta vigente: recibido 10000 (R\$ 100), tarifa fija 100 (R\$ 1), Lunium 200 bps (2%) y socio 100 bps (1%).

Componente	Importe del ejemplo
Base tras la tarifa fija	R\$ 99,00
Neto sin comisión del socio	R\$ 97,02
Neto con comisión, antes de la conversión	R\$ 96,03
Tarifa Lunium, incluida la fija	R\$ 2,98
Comisión del socio	R\$ 0,99
Revisión	R\$ 96,03 + R\$ 2,98 + R\$ 0,99 = R\$ 100,00

El cálculo usa $\text{floor}(\text{base_cents} * (10000 - \text{lunium_bps} - \text{partner_bps}) / 10000)$. La comisión es la diferencia entre el neto sin y con la comisión adicional, conservando el redondeo del producto. No apliques un segundo descuento de comisión sobre la cantidad ya cotizada.

POST /cashin/preview puede devolver lunium_fee_cents, partner_fee_cents, fee_total_cents, net_cents y la estimación de crypto en asset_amount, ya neta de la tarifa de retiro de la red (withdraw_fee). En las respuestas con este desglose, $\text{fee_total_cents} + \text{net_cents} = \text{amount_cents}$. En los campos antiguos brl_per_usdt y usdt_amount, comprueba el activo devuelto: el nombre del campo puede seguir siendo el mismo para USDC.

El tipo de cambio no es otra comisión del socio. La conversión usa el ask de Binance, sin el antiguo recargo. La tarifa Lunium y la comisión se calculan por separado de la cotización. La vista previa de cash-in es una estimación y no fija el tipo de cambio hasta el pago. Los costes y mínimos de la ruta/red también pueden afectar a la entrega: lee los campos aplicables del catálogo y de la respuesta. No sumes withdrawFee a ciegas a una cantidad que ya lo tiene en cuenta. Comprueba la cantidad realmente liquidada en el estado y en el comprobante de la operación.

En el depósito en custodia, el resultado es BRL, sin compra inmediata de crypto. El retiro posterior tiene sus propias tarifas/conversión. La vista previa de cash-in no es una vista previa completa del retiro de custodia. Para payout PIX, la respuesta separa fee_service_total_cents (Lunium + socio) y fee_provider_cents; fee_cents es el total. La comisión del socio, el saldo de los clientes y el beneficio de Lunium son cuentas distintas; una diferencia entre cotización y entrega, por sí sola, no demuestra un beneficio.

Acreditación, movimientos y retiro de la comisión

1. Consulta GET /comissao: disponivel_cents, creditado_cents, estornado_cents, sacado_cents, min_saque_cents, pode_sacar_agora, proximo_saque_em y politica.
2. Comprueba GET /comissao/extrato?limit=50. Usa next_before como before en la siguiente página. Relaciona origem_id con la solicitud de cobro o el retiro; lee kind, valor_cents y partner_fee_bps.
3. El cálculo se realiza después de status=paid y settlement_status=sent, en un proceso posterior a la liquidación. Un PIX pagado por sí solo no demuestra que haya comisión disponible. Los depósitos nuevos en saldo no generan comisión. Las compras y salidas usan la comisión de la familia correspondiente; consulta los valores guardados de la operación. El sandbox no acumula comisiones reales.

4. Los movimientos `credito`, `estorno`, `saque` y `saque_estorno` explican el saldo. Comprueba los signos y no cuentes el mismo asiento dos veces. Si una operación elegible no aparece tras el cálculo, envía su ID al grupo humano para conciliarla; no cambies la tarifa esperando corregir el pasado.

5. Para retirar, consulta la política vigente, el saldo y la ruta entregable. El mínimo convertido a BRL (actualmente US\$ 10 a la cotización del momento) y el intervalo entre retiros proceden de la API; si `min_saque_cents` sea `null`, no hay cotización para validar el mínimo. `pode_sacar_agora` no sustituye a las validaciones de red, dirección, memo, liquidez ni mínimo de la ruta.

POST `/comissao/sacar` recibe `amount_cents`, `asset`, `chain`, `payout_address`, el eventual `payout_tag` y `external_id`. El retiro no cobra otra comisión adicional ni una tarifa administrativa de Lunium, pero la conversión y los costes de la ruta siguen siendo relevantes. La respuesta contiene `saque` y `comissao`; sigue el identificador público de `saque` en GET `/cashin/{cashin_id}/status` hasta confirmar la entrega. No confundas el éxito HTTP con la entrega de cripto.

Usa un prefijo propio, como `comissao-...`, para `external_id`: el espacio de referencias se comparte con las solicitudes de cobro. Tras un timeout, primero busca `/cashin/charges?external_id=...` y sigue el ID existente. El retiro valida la política y el intervalo antes de buscar la referencia; por tanto, un reintento puede responder 429 aunque la solicitud anterior haya sido aceptada. No generes otro ID ni repitas retiros simultáneos para intentar desbloquearla.

PATCH `/keys/me` acepta `partner_fee_bps` para la comisión adicional. Consulta la política, el disponible y las condiciones de retiro en GET `/comissao`; movimientos en GET `/comissao/extrato`; retiro en POST `/comissao/sacar`. La comisión es un saldo independiente de la custodia de los clientes. No sumes ambos ni trates la comisión como una recompensa por recomendación de Mesh.

Para los agentes, [Mesh](#) publica el descubrimiento, registro, validación, mandate y activación. Usa GET `/mesh` y las acciones que devuelve. El MCP está en <https://api.luniumpay.com/mcp>; descubre las herramientas mediante `tools/list`, porque la superficie de herramientas no es idéntica a todas las rutas HTTP. Las operaciones sin una herramienta específica siguen disponibles mediante la API autenticada.

El inventario siguiente se genera a partir del OpenAPI de esta revisión. También incluye la consulta de entregas de webhook, recuperación del panel, catálogo DEX, comisión y ciclo de vida Mesh. Las rutas administrativas e internas no son interfaces públicas del socio.

Método	Ruta	Operación
GET	<code>/cash-outs</code>	Listar órdenes
POST	<code>/cash-outs</code>	Cotizar (paso 1 de 3)
POST	<code>/cash-outs/preview</code>	Vista previa de cotización (no crea una orden)
GET	<code>/cash-outs/{cashout_id}</code>	Hacer seguimiento (paso 3 de 3)
POST	<code>/cash-outs/{cashout_id}/accept</code>	Aceptar y obtener la dirección (paso 2 de 3)
POST	<code>/cash-outs/{cashout_id}/refund-address</code>	Indicar la billetera confirmada para una devolución pendiente

Método	Ruta	Operación
GET	/cashin/catalog	Catálogo de cash-in (lo que entregamos al recibir PIX)
POST	/cashin/charge	Generar una solicitud de cobro PIX para custodia BRL o entrega de criptomonedas
GET	/cashin/charges	Lista/recupera solicitudes de cobro de cash-in de esta clave
GET	/cashin/events	Flujo de eventos de tu clave (operación)
GET	/cashin/limits	Cuánto puede pagar este contribuyente ahora
POST	/cashin/preview	Cotización de cash-in: no crea nada
GET	/cashin/{cashin_id}/status	Estado de una solicitud de cobro PIX
GET	/catalog	Activos, redes y límites que permiten liquidar ahora
GET	/catalog/dex	Tokens de larga cola negociables mediante DEX
GET	/comissao	Saldo de la comisión del socio (sub fee)
GET	/comissao/extrato	Movimientos de la comisión
POST	/comissao/sacar	Retirar la comisión en criptomonedas
POST	/keys	Crear clave (autoservicio)
GET	/keys/me	Límites vigentes y uso del día
PATCH	/keys/me	Actualiza la clave (nombre, settlement, webhook, rotaciones)
POST	/keys/painel-link	Recibir el enlace del panel por correo electrónico (sin clave)
POST	/keys/sandbox	Obtener una clave de API de sandbox: una llamada, sin necesidad de clave de API
POST	/leads	Derivar un caso comercial o técnico a una persona
GET	/mesh	Todo lo que un agente necesita saber antes de integrar
GET	/mesh/agents	Quién está en la red

Método	Ruta	Operación
POST	/mesh/agents	Registrar un agente y recibir credenciales de sandbox
GET	/mesh/agents/{agent_id}	Estado, mandate, métricas y linaje de un agente
POST	/mesh/agents/{agent_id}/mandate	Definir la propia política de límites
GET	/mesh/agents/{agent_id}/network	Linaje, árbol y volumen de la red
POST	/mesh/agents/{agent_id}/production-request	Activar producción (automático, sin intervención humana)
GET	/mesh/agents/{agent_id}/rewards	Saldo de recompensas y movimientos del ledger
POST	/mesh/agents/{agent_id}/settlement	Indicar adónde va el dinero
POST	/mesh/agents/{agent_id}/validate	Demostrar que se ha probado el sandbox
GET	/mesh/economy	Reglas públicas de la economía de la red
GET	/mesh/graph	Grafo de agentes de Mesh
POST	/mesh/referrals	Emitir una atestación de recomendación
GET	/payouts	Enumera los payouts de esta clave
POST	/payouts	Enviar un PIX directamente (sin etapa de cripto)
GET	/payouts/{payout_id}	Estado de un payout
GET	/ping	Estado de la API
GET	/pix/keys/lookup	Consultar el titular PIX cuando la verificación esté disponible
GET	/saldo	Saldo BRL (custodia): disponible, retenido y próximas liberaciones
GET	/saldo/clientes	Subcuentas (clientes finales) con saldo, paginadas
GET	/saldo/consolidado	Consolidado de la clave: cuenta principal + clientes (custodia total)
GET	/saldo/extrato	Movimientos del saldo (más recientes primero)
POST	/saldo/sacar-cripto	Retirar saldo BRL como cripto a una billetera de autocustodia

Método	Ruta	Operación
POST	/saldo/transferir	Transferencia interna entre subcuentas y la cuenta principal (sin PIX)
GET	/sandbox	Descubrir las capacidades del sandbox
POST	/sandbox/cashin/{cashin_id}/pay	Simular el pago de una solicitud de cobro PIX de prueba
GET	/sandbox/events	Inspeccionar los eventos de sandbox generados
POST	/sandbox/operations/{operation_id}/resolve	Resolver una operación de sandbox con resultado incierto
GET	/sandbox/qr/{cashin_id}.svg	Imagen QR de sandbox que no permite pagos
POST	/sandbox/reset	Restablecer el estado ficticio de esta clave
GET	/v1/verificar/{e2e}	Confirmar que se realizó un PIX (abierto, sin clave)
GET	/webhooks/deliveries	Historial de entregas de webhook de esta clave
POST	/webhooks/deliveries/{event_id}/retry	Reenviar un evento
POST	/webhooks/test	Emitir un evento de prueba ahora

15. Lista de comprobación de producción

- Comisión configurada en la clave correcta; desglose y movimientos conciliados sin descuentos duplicados.
- Correo de recuperación comprobado; monitor, clave de API y secreto del webhook tratados por separado.
- Procedimiento de incidentes ensayado; no se dan por disponibles las funciones en desarrollo.
- Participación en [Lunium Partners](#) muy recomendable para los responsables técnico y operativo; seguimiento de avisos y cambios definido.
- Grupo dedicado con @luniumb2b e @luniumb2c muy recomendable, creado por Lunium o por el socio, con responsables definidos.
- Monitor privado comprobado; panel propio autorizado por proyecto/cliente, con indicación de la última actualización.
- Bandeja de entrada persistente, duplicados, eventos fuera de orden, indisponibilidad, reintentos y conciliación del webhook probados.
- Clave guardada en el servidor; entorno de pruebas separado de producción.

- `/keys/me` y `/saldo` comprobados para la clave y para el ámbito de cada cliente.
- La custodia usa destino: "saldo"; la billetera solo se elige al retirar criptomonedas.
- La interfaz consulta el catálogo y los límites actuales, muestra la retención y exige un memo cuando es necesario.
- IDs guardados de forma persistente; los reintentos conservan el cuerpo y `external_id`; recuperación tras un timeout probada.
- El webhook verifica el HMAC y la marca de tiempo, elimina duplicados por evento y responde antes del timeout.
- La conciliación distingue entre PIX pagado, abono registrado, saldo disponible para retirar y criptomonedas entregadas.
- Los errores de saldo, período de espera, ruta, tarifa, proveedor y clave se gestionan sin duplicar operaciones.
- Telegram vinculado por un administrador identificado; retiros confirmados en el contexto correcto.
- La validación incluye éxito, retención, vencimiento, fallo, reenvío y consulta después de perder una respuesta.

Soporte: [@luniumb2b](#) • [@luniumb2c](#) • contato@luniumpay.com. Sigue el [estado](#) y el [Lunium Partners](#). Al pedir ayuda, envía los IDs y `request_id` cuando esté disponible, sin compartir la clave completa.

16. Qué cambió en esta revisión

- 09/10/2026 • OpenAPI 1.44.1 • revisión 2026-10-09.1: Sandbox: POST `/sandbox/operations/{operation_id}/resolve` responde 400 `outcome_invalido` cuando outcome no es `sent` ni `failed` (antes, un cuerpo sin outcome podía responder 200 sin resolver nada); la comprobación se realiza antes de buscar la operación y no cambia nada en el simulador. Saldo: en GET `/saldo`, `proximas_liberacoes` puede incluir hasta 200 liberaciones del ámbito solicitado (antes, como máximo 20); el formato y el ámbito (cuenta principal o `customer_ref`) no cambian.
- 08/10/2026 • OpenAPI 1.44.0 • revisión 2026-10-08.2: Venta de DePix: `refund_address` deja de ser obligatoria al crear la orden; la billetera de devolución solo se solicita si falla la venta (Billetera de devolución). Sin billetera, la orden permanece en `AWAITING_REFUND_ADDRESS` (webhook `cashout.awaiting_refund_address`) e indicas la dirección Liquid en POST `/cash-outs/{cashout_id}/refund-address`; la orden continúa hacia `REFUNDING` y `REFUNDED`, con webhooks. Errores: 400 `refund_address_invalida` (incluye direcciones de la propia Lunium), 409 `refund_address_imutable` y 409 `devolucao_indisponivel`. El DePix que ya se había enviado al proveedor solo se devuelve después de confirmar el retorno: hasta entonces, la orden permanece en `MANUAL_REVIEW`, y la devolución es como máximo lo que regresó. El detail del rechazo por límite del pagador con `allow_hold: false` se ha revisado y muestra importes en R\$ con formato brasileño (R\$ 1.234,56), al igual que la explicación de GET `/cashin/limits`; erro y code no cambian. Retiro de la comisión a partir de US\$ 10. Panel del socio (2): activación de una cuenta nueva con un código de 6 dígitos por correo electrónico; invitación aceptada con un clic por quienes acceden mediante código o Google; Mi acceso con Cerrar las demás sesiones, verificación en dos pasos que una cuenta sin contraseña desactiva con un código y correo de aviso cuando cambia la contraseña; correo a los propietarios cuando alguien deja de ser propietario; Pagar con cripto con stablecoins y las 30 principales criptomonedas; mayor legibilidad y una sesión de 12 horas que se renueva con el uso. Nuevo diagrama de flujo de la venta de cripto (venta y devolución) en la sección 8.

- 08/10/2026 • OpenAPI 1.43.1: la solicitud de depósito de la Billetera USDT/USDC dura 1 hora (`deposit_validity_hours = 1`; `expires_at` = creación + 1 h; antes eran 3 h). Nueva sección Nuevo panel del socio (2): `parceiros.luniumpay.com` con resumen del negocio, comprobantes en PDF, billetera con código QR de la dirección y cuenta atrás, notificaciones y equipo con roles y segundo factor. Desde el panel, solo el Propietario vincula Telegram y cambia la billetera de liquidación y las comisiones adicionales. Diagrama de flujo de la billetera (depósito y PIX pagado con saldo) en la sección Billetera USDT/USDC.
- 07/10/2026 • OpenAPI 1.43.0: Billetera USDT/USDC (`/carteira-usdt`): depósito de USDT o USDC en varias redes, PIX pagado con saldo (clave o Copia y Pega), movimientos, webhooks `wallet.*` y simulador en sandbox.
- 07/10/2026: la vista previa de cash-in (`POST /cashin/preview`) pasa a usar en producción los mismos nombres que en sandbox: `asset_amount` (lo que llega a la billetera), `withdraw_fee` (tarifa de retiro de la red, en unidades del activo), `withdraw_min` y `brl_per_asset` (para convertir a reales brasileños). `amount` y `network_min` siguen como nombres antiguos.
- 03/10/2026: sección Comprobante oficial del PIX (8): qué contiene el comprobante, los cuatro formatos (`pix_e2e`, `receipt_url`, `receipt_pdf_url`, `verify_url`) y por dónde llega (API, webhook, panel del socio y Telegram).
- Tarifas y comisiones conciliadas mediante un ejemplo; recuperación del acceso y escenarios de incidentes detallados.
- Fuentes estructuradas para IA alineadas con OpenAPI; las capacidades futuras dependen de su publicación y validación.
- Lunium Partners destacado como comunidad oficial al inicio, en la operativa por Telegram y en la lista de comprobación.
- Recomendación del grupo humano restaurada: `@luniumb2b` y `@luniumb2c`, creado por Lunium o por el socio.
- Monitor privado, feed, panel en vivo, estados por flujo, firma, recuperación y validación detallados.
- Revisión de esta actualización centrada en soporte y seguimiento; versión de OpenAPI referenciada actualizada.
- Flujo de custodia sin billetera fija y retiros mediante Telegram documentados.
- Valor predeterminado de `destino` de la API separado del valor predeterminado del bot.
- Tarifas y límites vinculados a las consultas correctas; eliminadas las promesas fijas de precio y plazo.
- Retención local, retención del proveedor y período de espera del saldo diferenciados.
- Catálogos de entrada y entrega separados, incluido el filtro `entregavel` y la restricción de Liquid/DePix en el retiro.
- Titular PIX opcional/no disponible y `br_code` condicionado a la disponibilidad.
- Timeout del webhook corregido a 10 segundos, firma robusta y recuperación de fallos.
- Estados, campos, ejemplos e inventario de OpenAPI revisados junto con las rutas.
- HTML, Markdown y PDF generados a partir del mismo contenido, con validaciones del contrato antes de publicar.

[Descargar PDF](#) • [Markdown](#) • [Ejemplos estructurados](#) • [Revisión del contrato](#)