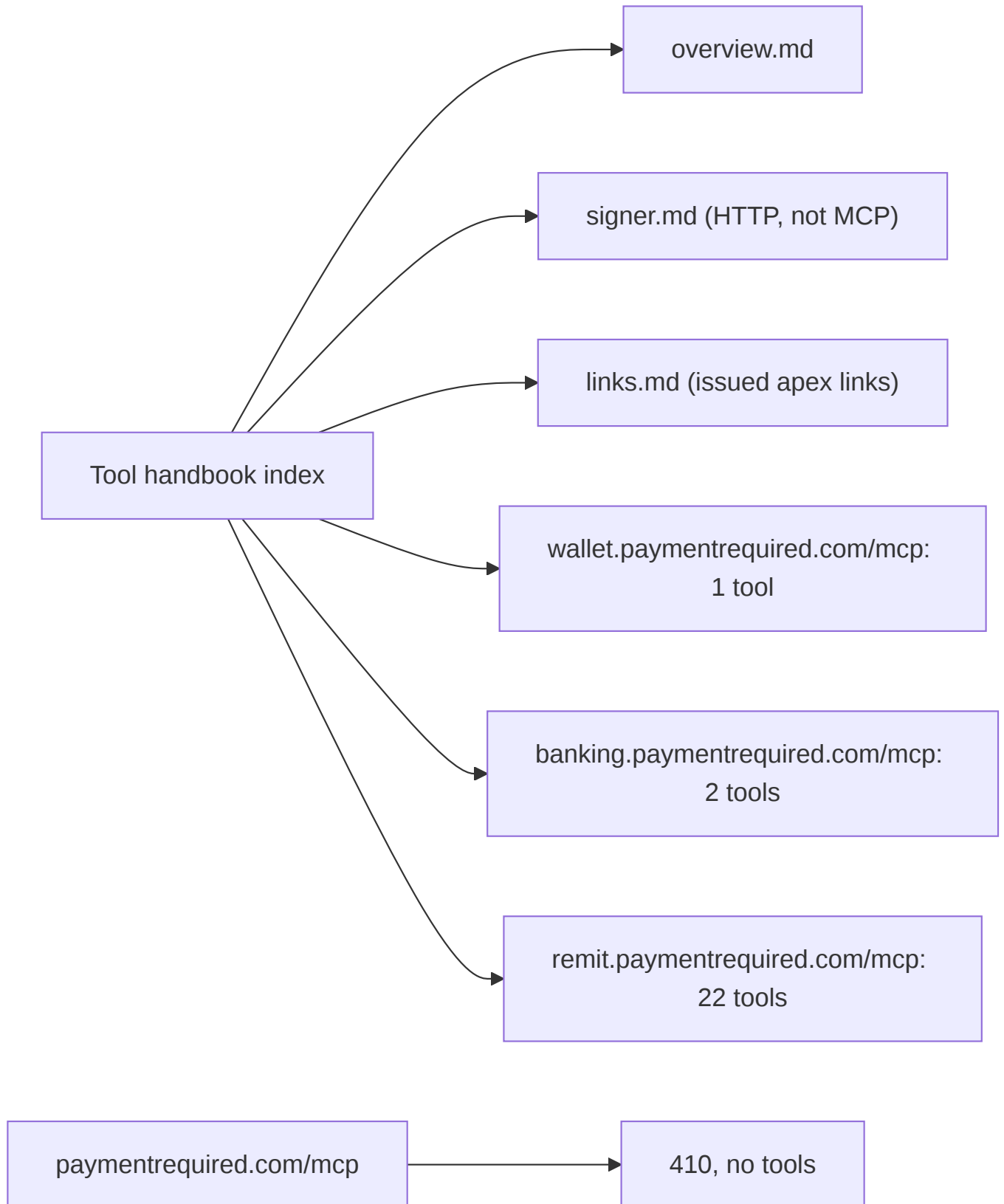




paymentrequired.com tool handbook (directory-wave edition)

Generated 2026-10-14T18:58:47Z by `_build/build.py` (cloud). Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (`directory-wave/parent-directory-wave-acceptance.json`), not live; **LIVE-RO** = observed read-only on the live host 2026-

10-14 (raw files in [evidence/live-probes/](#)); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

- [System overview](#) — downloads: [overview.mmd](#), [SVG](#), [PNG](#), [PDF](#)
- [Whole handbook, searchable PDF](#)
- [Signer service](#) (HTTP approval contract, not an MCP tool)
- [Already issued apex links](#)
- [Owner walkthrough checklist](#)
- [CHANGELOG](#) · [PROMPTLOG](#) · [inventory.json](#) · [result.json](#)

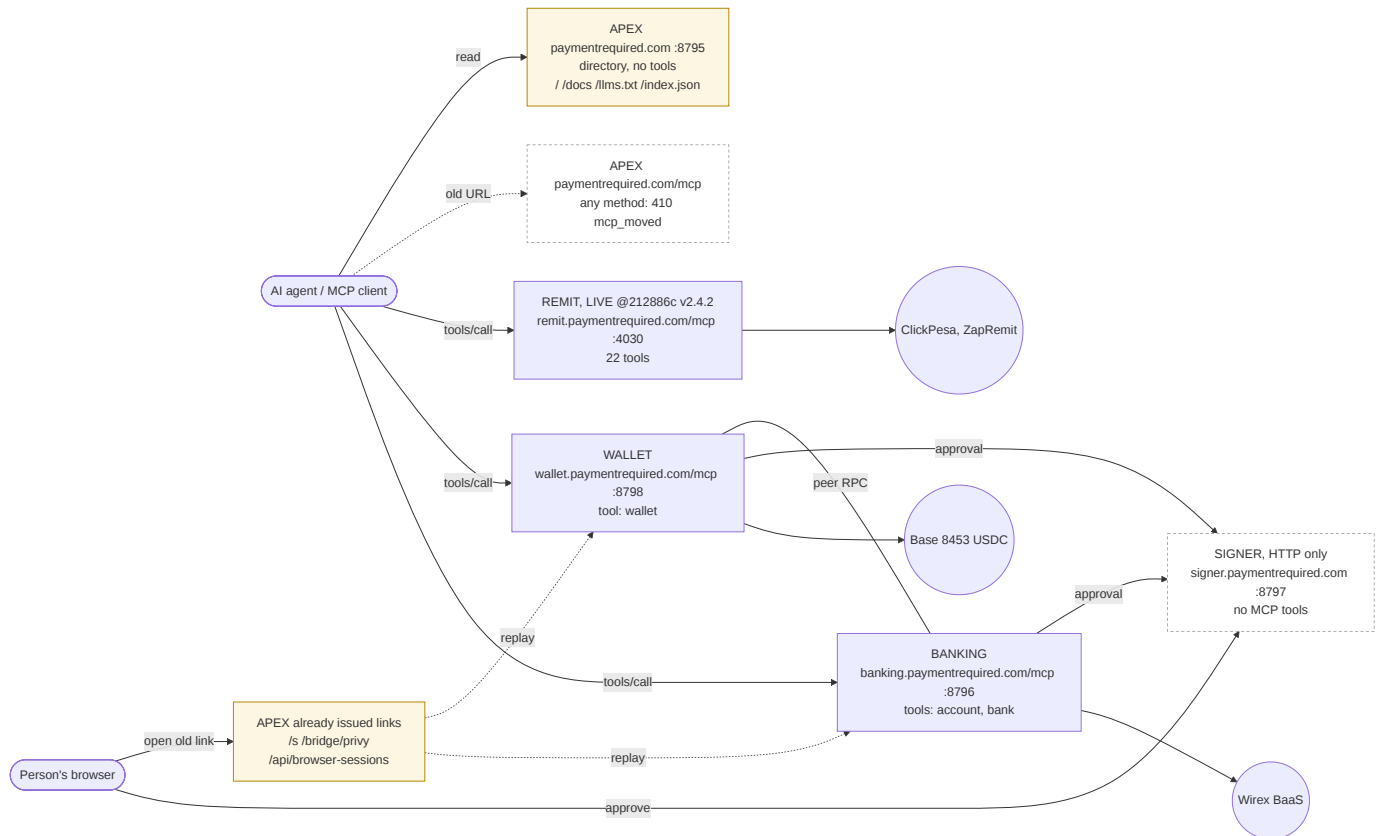
Every public MCP tool (25, catalog order)

#	Service	MCP endpoint	Tool	Page	Actions/ops on page
1	Wallet	https://wallet.paymentrequired.com/mcp	wallet	tools/wallet.md	balance, fund, pay, send, history, bridge, discover
2	Banking	https://banking.paymentrequired.com/mcp	account	tools/account.md	status, verify, limits
3	Banking	https://banking.paymentrequired.com/mcp	bank	tools/bank.md	card, iban, send (iban), send (card_id), send (to 0x), swap, transactions, manage
4	Remit	https://remit.paymentrequired.com/mcp	route_price	tools/remit-route_price.md	route_price
5	Remit	https://remit.paymentrequired.com/mcp	route_pay	tools/remit-route_pay.md	route_pay
6	Remit	https://remit.paymentrequired.com/mcp	check_recipient	tools/remit-check_recipient.md	check_recipient
7	Remit	https://remit.paymentrequired.com/mcp	request_tzs	tools/remit-request_tzs.md	request_tzs
8	Remit	https://remit.paymentrequired.com/mcp	get_request	tools/remit-get_request.md	get_request
9	Remit	https://remit.paymentrequired.com/mcp	get_balance	tools/remit-get_balance.md	get_balance
10	Remit	https://remit.paymentrequired.com/mcp	get_rates	tools/remit-get_rates.md	get_rates
11	Remit	https://remit.paymentrequired.com/mcp	list_banks	tools/remit-list_banks.md	list_banks
12	Remit	https://remit.paymentrequired.com/mcp	check_bank	tools/remit-check_bank.md	check_bank
13	Remit	https://remit.paymentrequired.com/mcp	list_lipa_providers	tools/remit-list_lipa_providers.md	list_lipa_providers
14	Remit	https://remit.paymentrequired.com/mcp	check_lipa	tools/remit-check_lipa.md	check_lipa
15	Remit	https://remit.paymentrequired.com/mcp	quote_tzs	tools/remit-quote_tzs.md	quote_tzs
16	Remit	https://remit.paymentrequired.com/mcp	send_tzs	tools/remit-send_tzs.md	send_tzs
17	Remit	https://remit.paymentrequired.com/mcp	get_payment	tools/remit-get_payment.md	get_payment
18	Remit	https://remit.paymentrequired.com/mcp	get_order	tools/remit-get_order.md	get_order

#	Service	MCP endpoint	Tool	Page	Actions/ops on page
19	Remit	https://remit.paymentrequired.com/mcp	zapremi_countries	tools/remit-zapremi_countries.md	zapremi_countries
20	Remit	https://remit.paymentrequired.com/mcp	zapremi_methods	tools/remit-zapremi_methods.md	zapremi_methods
21	Remit	https://remit.paymentrequired.com/mcp	zapremi_options	tools/remit-zapremi_options.md	zapremi_options
22	Remit	https://remit.paymentrequired.com/mcp	zapremi_rates	tools/remit-zapremi_rates.md	zapremi_rates
23	Remit	https://remit.paymentrequired.com/mcp	zapremi_quote	tools/remit-zapremi_quote.md	zapremi_quote
24	Remit	https://remit.paymentrequired.com/mcp	zapremi_payment	tools/remit-zapremi_payment.md	zapremi_payment
25	Remit	https://remit.paymentrequired.com/mcp	zapremi_order	tools/remit-zapremi_order.md	zapremi_order

paymentrequired.com/mcp serves **no** tool (410). Signer serves **no** MCP tool.

Directory topology (accepted source, NOT deployed) · paymentrequired.com directory, 3 MCP servers, Signer HTTP



System overview – directory topology (accepted source, not deployed)

← [Index](#) · [Issued links](#) · [Signer](#) · Downloads: [overview.mmd](#) · [SVG](#) · [PNG](#) · [PDF](#) · [Handbook PDF](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

In one paragraph

paymentrequired.com is a **directory**: it lists every tool and points each to its own MCP endpoint, and serves **no tool itself** (facade:src/server.mjs:1, facade:src/directory.mjs:15). Its old /mcp answers **410** mcp_moved for every method before any body is read (facade:src/server.mjs:155, facade:src/directory.mjs:38). Agents call **Wallet** at wallet.paymentrequired.com/mcp (tool wallet; wallet:src/wallet-server.mjs:235), **Banking** at banking.paymentrequired.com/mcp (tools account, bank; banking:src/server.mjs:317) and **Remit** at remit.paymentrequired.com/mcp (22 tools; [remit:src/tools.js:17](#)). **Signer** is not an MCP server; Wallet and Banking ask it for the person's approval ([signer.md](#)). Links already issued on paymentrequired.com keep their original addresses ([links.md](#)).

Services

Service	Public origin / local port	Source (status)	Public MCP tools	Owns
Apex directory	paymentrequired.com, :8795 (<code>facade:src/server.mjs:44</code>)	agent-payments-mcp-wirex, directory-wave accepted source (not deployed)	none (/mcp 410; /healthz tools: [], <code>facade:src/server.mjs:157</code>)	issued links, short_links, its connection_sessions
Wallet	wallet.paymentrequired.com/mcp, :8798 (<code>wallet:src/service-mode.mjs:12</code>)	paymentrequired-wallet (accepted source, unpublished)	wallet	agent Safe custody, tool_charges
Banking	banking.paymentrequired.com/mcp, :8796 (<code>banking:src/banking-config.mjs:9</code>)	paymentrequired-banking (accepted source, unpublished)	account, bank	Wirex client, mcp_bank_*, wallet_payments
Signer	signer.paymentrequired.com, :8797 (<code>signer:src/root-signer-broker.mjs:58</code>)	paymentrequired-signer (accepted source, unpublished)	none (HTTP only)	signer_requests
Remit	remit.paymentrequired.com/mcp, :4030 (remit:src/server.js:59)	nuri-com/nuri-clickpesa-remit-mcp @ 212886c v2.4.2 (live)	22	journal.jsonl

Actual callers (only edges that exist in code)

From → to	Why	Code
agent → wallet / banking /mcp	own tools/list and tools/call per service	<code>wallet:src/wallet-server.mjs:273,</code> <code>banking:src/server.mjs:361</code>
apex directory → wallet / banking bridge	replay already issued approval links to the owner holding the row (owner-first)	<code>facade:src/server.mjs:122,</code> <code>facade:src/server.mjs:205</code>
apex directory → wallet / banking helper	on_tap, identity for issued connect links	<code>facade:src/server.mjs:83</code>
wallet → banking bank	wallet fund top-up from the bank account	<code>wallet:src/wallet-server.mjs:94</code>
wallet → banking helper wirex_read	reserve balance read (4 read methods)	<code>wallet:src/wallet-server.mjs:98</code>
banking → wallet helper safe_address / charge	account limits; tool prices journalled on wallet	<code>banking:src/server.mjs:157,</code> <code>banking:src/server.mjs:167</code>
wallet / banking → signer /v1/requests	root approval (only when ROOT_SIGNER_URL is set)	<code>banking:src/signer-broker-hook.mjs:23</code>
Remit → (none of the above)	no call found — UNVERIFIED (text search, not proof)	Remit outbound: ClickPesa (remit:src/clickpesa.js:7), ZapRemit, x402 facilitator, chain RPCs

Peer down: a refused connection answers `service_unavailable ... Nothing was sent`; anything else answers `unknown → repeat with the same request_id (banking:src/service-rpc.mjs:65)`.

Tool count (mechanical, from catalogs)

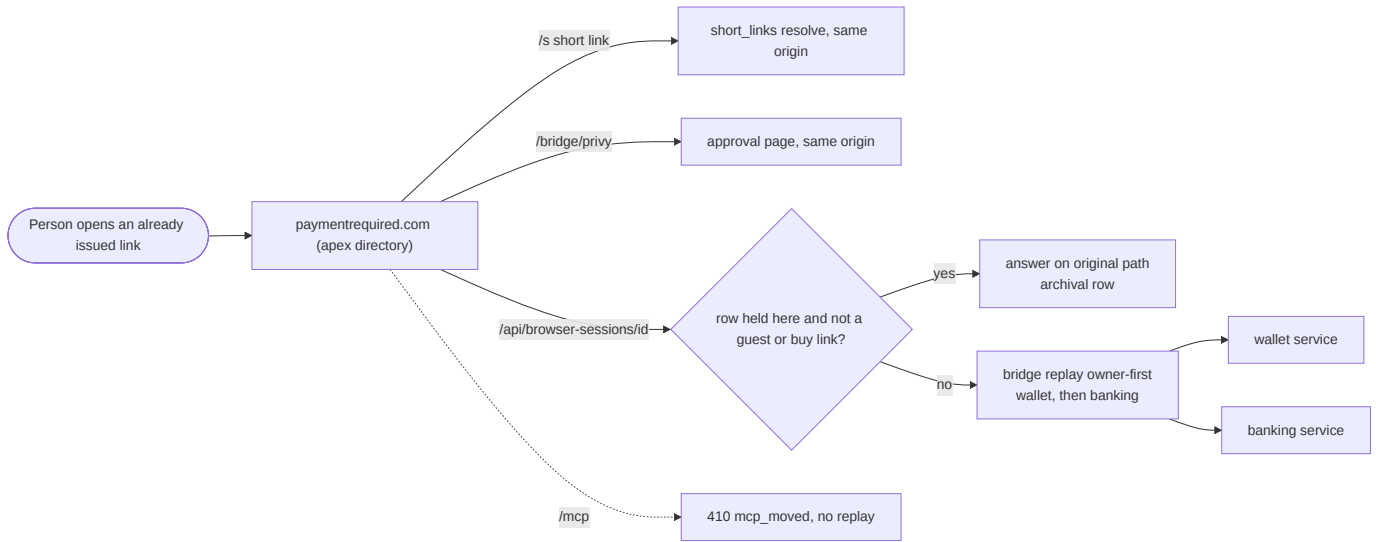
25 public MCP tools on 3 servers: Wallet 1 + Banking 2 + Remit 22. Apex directory: 0. Signer: 0. Sources: wallet/banking = directory pin `contracts/tools.json` (FIXTURE probe-schema 3/3 equal to the services' `tools/list`); Remit = LIVE-RO `tools/list` at 212886c = code catalog `src/tools.js` + `src/zapremit-tools.js`. Reconciliation: [inventory.json](#).

Not public capabilities (do not use as tools)

- Private RPC ops: wallet `pay_fee`; helper `identity, safe_address, charge, remember, on_tap, wirex_read`; bridge replay.
- Remit operator `/admin approve/reject` and `/webhook` are operator/provider surfaces, not agent tools.
- GET `/api/signer-preparation/<id>` on the apex answers explicit **404** `signing_preparation_unavailable` (no signing capability added; `facade:src/server.mjs:193`).
- `/app` console on the apex: retired with `/mcp`, **410** (`facade:src/server.mjs:145`).

Live vs accepted source (LIVE-RO 2026-10-14T18:09:13Z)

- Remit LIVE-RO: `/health` version `2.4.2` revision `212886c7eedfa314f7de790622ea95632630057b`; `tools/list` 22 names = code catalog at that commit. GitHub `main` is 2 commits ahead with docs-only files (`evidence/live-probes/remit-git-compare.txt`).
- Old apex (pre-cutover deployment, NOT this topology) LIVE-RO: `/healthz` `git_sha` `ed64a10`, `/mcp` `tools/list` still `[wallet, account, bank]`. The directory topology documented here is **not deployed**.
- `wallet.`, `banking.`, `signer.paymentrequired.com` LIVE-RO: **503 "Service not deployed yet"** on `/healthz` and `/mcp`.
- Accepted source: FIXTURE 99/99 (pre 9, directory 69, schema 3, main 7, restart 5, signerdown 6), status `DIRECTORY_WAVE_SOURCE_AND_COMBINED_FIXTURES_ACCEPTED_NOT_IMAGE_BUILD_OR_RELEASE_ACCEPTANCE`; independent images verified: `False`; release accepted: `False`.
- Directory Remit snapshot `contracts/remit-tools.json` was captured at Remit `76bd550` (names/titles/descriptions only); its 22 names/titles/descriptions equal live `212886c` (`inventory.json` `directory_remit_snapshot`). Refresh is parent-owned.



Already issued apex links (kept independent of the retired /mcp)

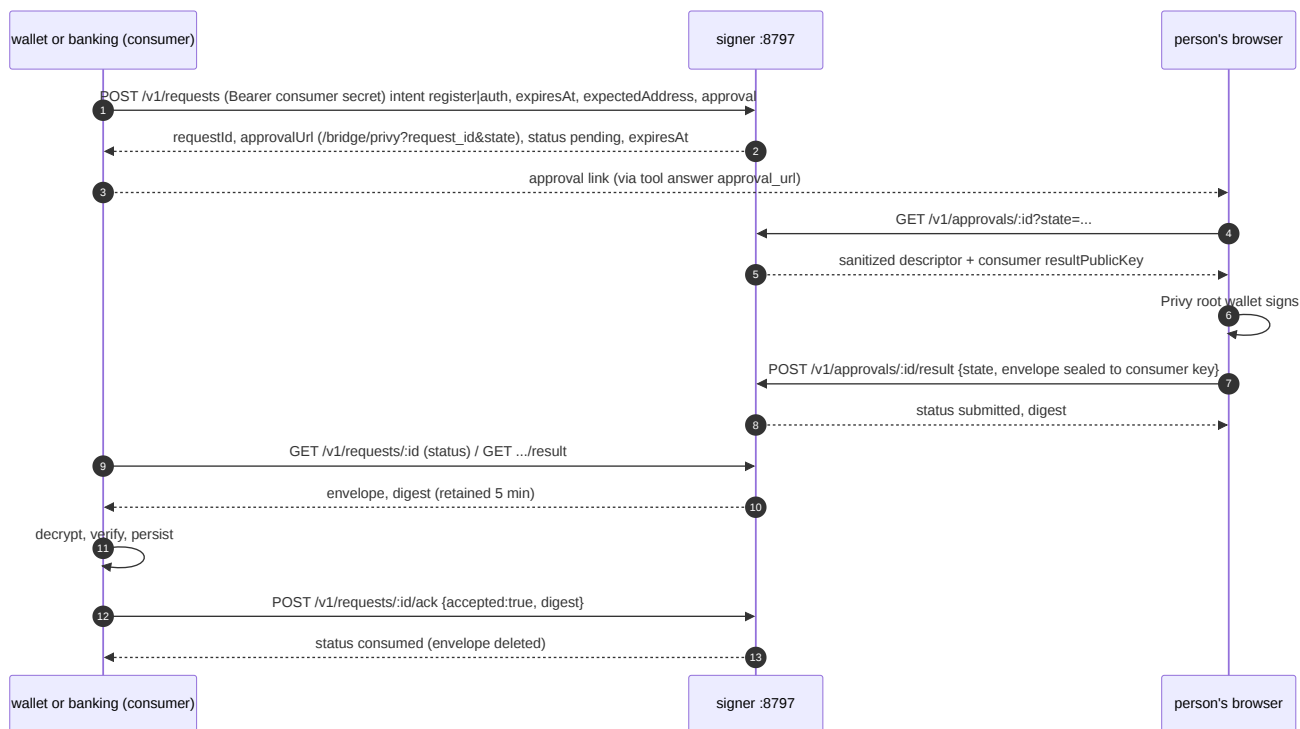
[← Index](#) · [Overview](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

Links an agent already handed to a person on https://paymentrequired.com/... keep their **original path**; retiring /mcp does not touch them. The handbook keeps these addresses verbatim; it does not rewrite them to subdomains.

Issued path	What answers	Code
GET /s, POST /s	short-link page and resolver on the apex	facade:src/server.mjs:146
GET /bridge/privy	approval page on the apex	facade:src/server.mjs:179
/api/browser-sessions/<id> [/callback approval connection]	owner-first replay to wallet, then banking; archival row answers only if no owner holds it	facade:src/server.mjs:205, facade:src/server.mjs:122
/api/browser-sessions/<id>/connection (archival)	connection completion on the original path	facade:src/server.mjs:224
/mcp	410 , nothing parsed, forwarded or replayed	facade:src/server.mjs:155

Observed: **FIXTURE** (accepted 99/99): issued G/GA/W/B/B0 links answered with zero payment delta; banking direct accepted a retained apex pass via identity hand-off (probe-directory.json keys G_link...B0_link, banking_direct_accepts_retained_GA_pass_via_handoff). Live: **UNVERIFIED** (topology not deployed).



Signer service (signer.paymentrequired.com) – not an MCP tool

[← Index](#) · [Overview](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (`directory-wave/parent-directory-wave-acceptance.json`), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in `evidence/live-probes/`); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it is

The signer shows a person what their agent wants approved; the person signs with their Privy **root** wallet; the result goes back **sealed** to the service that asked (wallet or banking). It holds no keys, no money and no product state (`signer:src/root-signer-broker.mjs:1`). It has **no** `/mcp`, **no** `tools/list`, **no** `tools/call` (grep of `signer/repo/src` for `tools/list`, `tools/call`, `"/mcp"`: 0 hits); the apex directory lists it as `mcp: null` (`facade:src/directory.mjs:29`). An agent never calls it directly; it only sees an `approval_url` inside a `wallet/account/bank` answer.

Source defect (report, not fixed here — sources are read-only): the signer's own homepage text still says "Agents connect through the one MCP endpoint" `https://paymentrequired.com/mcp` (`signer:src/service-index.mjs:52`), which the directory retires (410). Wallet/banking copies of the same file were already corrected (`wallet:src/service-index.mjs:52`). Parent decides the fix.

HTTP contract (exact routes)

Who	Method + path	Body / query	Answer	Code
any	GET /healthz	-	{ok, service:"root-signer", git_sha, status}	signer:src/root-signer-broker.mjs:141
person	GET /bridge/privy?request_id&state	-	approval page	signer:src/root-signer-broker.mjs:143
person	GET /v1/approvals/:id?state=	-	descriptor + resultPublicKey, signerAdapter:"privy"; 410 if expired	signer:src/root-signer-broker.mjs:119
person	POST /v1/approvals/:id/result	{state, envelope}	{status:"submitted", digest}	signer:src/root-signer-broker.mjs:128
consumer (Bearer)	POST /v1/requests	{intent: "register" "auth", privyAppId (root), expiresAt ≤24h, expectedAddress?, approval, requestId?, state?, returnOrigin?}	{requestId, approvalUrl, status:"pending", expiresAt}	signer:src/root-signer-broker.mjs:80
consumer	GET /v1/requests/:id	-	{requestId, status, expiresAt, resultDigest?}	signer:src/root-signer-broker.mjs:100
consumer	GET /v1/requests/:id/result	-	{envelope, digest, approvedAt, retainUntil}; 409 unless submitted	signer:src/root-signer-broker.mjs:104
consumer	POST /v1/requests/:id/ack	{accepted:true, digest}	{status:"consumed"}	signer:src/root-signer-broker.mjs:109
consumer	POST /v1/requests/:id/cancel	{}	cancelled (pending only)	signer:src/root-signer-broker.mjs:114

Constants: intents signer:src/root-signer-envelope.mjs:15, result retention 300 s signer:src/root-signer-envelope.mjs:17, max request 24 h signer:src/root-signer-envelope.mjs:19. Status lifecycle (SQLite signer_requests): pending → submitted → consumed, **or** pending → expired | cancelled (signer:src/root-signer-store.mjs:13). Consumers authenticate by bearer token whose hash is in SIGNER_CONSUMERS_JSON; a consumer sees only its own rows (signer:src/root-signer-broker.mjs:66).

Who calls it

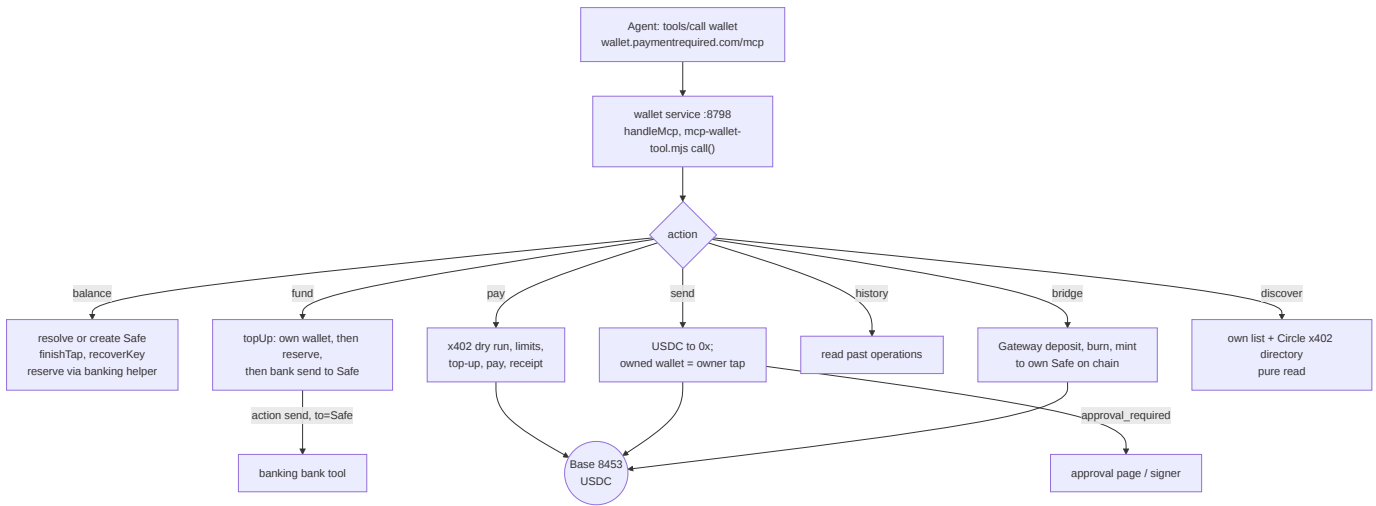
Wallet and banking, through root-signer-client.mjs, **only when** ROOT_SIGNER_URL **is set**; unset = the original in-process callback path (banking:src/signer-broker-hook.mjs:23, banking:src/root-signer-client.mjs:38). Remit never calls it.

Money / safety

No money moves through the signer. A signature approves exactly what the consumer's `approval` descriptor says; the consumer verifies the decrypted result against the pinned owner before acting (`bridge.approved()` in the tool code).

Observed

LIVE-RO 2026-10-14T18:09:13Z: `signer.paymentrequired.com` answers 503 "Service not deployed yet". FIXTURE (accepted 99/99): `banking_connect_link_on_signer` (probe-main) and the signer-down group 6/6 (probe-signerdown) passed with the final four source trees; synthetic keys, no real Privy. End-to-end with real Privy: **UNVERIFIED.**



wallet **(Wallet)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

The agent's own USDC wallet (a Safe on Base). It holds money for the agent, pays x402 paid links, sends USDC, moves USDC to another chain, and finds paid services. Before the person connects, the agent holds it through wallet_key. After the person taps the connect link, the wallet becomes theirs and outgoing sends need their tap.

Catalog: wallet.paymentrequired.com/mcp (wallet service tools/list: wallet only). **Schema & dispatch:** Input wallet:src/mcp-wallet-tool.mjs:116-139; **action enum** wallet:src/mcp-wallet-tool.mjs:123; **output** wallet:src/mcp-wallet-tool.mjs:139-181; **dispatch** wallet:src/mcp-wallet-tool.mjs:827-834; **allowed fields** wallet:src/mcp-wallet-tool.mjs:819. chain enum is filled at runtime from payable+Gateway chains wallet:src/mcp-wallet-tool.mjs:224 (pinned contract facade/contracts/tools.json enum: base, ethereum, arbitrum, optimism, polygon; **FIXTURE** schema probe equal).

Actions / operations

ac tio n	what happens	effect	source
ba la nc e	Shows available USDC and address. The first call with no key and no session asks whether to make a new wallet (new: true makes one, wallet:src/mcp-wallet-merge.mjs:36). Returns wallet_key once when created or when the key changes. Finishes a pending hand-over or top-up. Adds reserve (USDC on the bank account, only after the identity check), gateway per chain, and a connect approval_url while nobody owns the wallet.	read (+ may create a wallet, may finish an already-approved hand-over)	wallet:src/mcp-wallet-tool.mjs:465- wallet:src/mcp-wallet-tool.mjs:515

action	what happens	effect	source
fund	Adds money. Order: the person's connected wallet, then the reserve path (sandbox chain 84532 only; mainnet answers 'not available yet'), then a bank send of amount from the person's bank account to the Safe. With nothing usable it answers funded:false and the address to send USDC to. Without amount (with a session) it asks how much.	MONEY MOVE (bank → Safe via banking, approval link)	wallet:src/mcp-wallet-tool.mjs:427, wallet:src/mcp-wallet-tool.mjs:587; reserve gate wallet:src/mcp-wallet-fund-reserve.mjs:29
pay	Pays an https x402 link (url, optional method,body,max_amount, chain). It first dry-runs to read the price, checks the owner limits (over a limit the owner gets a one-time lift link), tops up any shortfall from the own-wallet/reserve lanes, then pays and returns receipt {tx, amount, to} plus the seller's resource. If the merchant answers ≥400 after settling, the answer is ok:false with the receipt.	MONEY MOVE (USDC to merchant)	wallet:src/mcp-wallet-tool.mjs:622- wallet:src/mcp-wallet-tool.mjs:696
send	Sends USDC amount to to (0x). Needs wallet_key (guest 44 / connected 66 chars) or the owner's session. A wallet owned by a person returns approval_required plus a link (D-OWNER-TAP). Wallet limits are enforced before signing.	MONEY MOVE (USDC on Base)	wallet:src/mcp-wallet-tool.mjs:697- wallet:src/mcp-wallet-tool.mjs:736
history	Lists past operations, oldest first: request_id, kind, status (done/pending/failed), amount, to, tx, reason. Needs wallet_key or session and never creates a wallet.	read	wallet:src/mcp-wallet-tool.mjs:794
bridge	Moves amount USDC through Circle Gateway to the same Safe on chain. It deposits the gap first (Circle credits it in about 20 min), then burns and mints. ethereum is refused. A second bridge under a new request_id waits while one is in flight.	MONEY MOVE (cross-chain, self-to-self)	wallet:src/mcp-wallet-tool.mjs:527
discover	Finds paid services: the operator's checked list first, then Circle's public x402 directory (query, limit 1-25). It is a pure read and resolves no wallet. Shape from code:ok, services[], total, sources{paymentrequired,circle}, next, status.	read	wallet:src/mcp-wallet-tool.mjs:744

Not public: Private RPC only, **not public:** action: "pay_fee" (tool-price payer) is routed in wallet-server.mjs wallet:src/wallet-server.mjs:122; helper ops safe_address, charge, remember, on_tap, identity wallet:src/wallet-server.mjs:125.

Prerequisites

None for discover.balance with new:true creates a guest wallet. Money actions need the wallet_key from the first answer, or the owner's session once connected. fund from the bank needs a verified bank account (see account). Funds on Base chain 8453.

Session / connect (shared by wallet, account, bank). A call without session (or with a dead one) does not fail: the connection layer answers status: "connect_required" with an approval_url the person opens once, then the same call is repeated (banking:src/mcp-connection.mjs:233, banking:src/mcp-connection.mjs:109). session must match SESSION_PATTERN banking:src/mcp-session.mjs:3. A connect reference passed as request_id (43 chars) without session is resolved to the session first (banking:src/mcp-connection.mjs:215). For account/bank on a handed-off identity, a one-time *lazy bank grant* link may appear first (banking:src/mcp-connection.mjs:153).

Per-action price. Off by default. If the operator sets TOOL_PRICES_JSON, a priced action is paid from the caller's agent wallet *before* it runs, and an unpaid charge means the action does not run (banking:src/tool-prices.mjs:1, banking:src/mcp-connection.mjs:293). On banking the charge goes to the wallet service over

RPC; if that peer is down the answer is `service_unavailable` ... Nothing was charged
(`banking:src/server.mjs:170`). Live price table: UNVERIFIED (env, not in repo).

How to use it (exact `tools/call` JSON)

Endpoint: `POST https://wallet.paymentrequired.com/mcp` (JSON-RPC 2.0). Placeholders in `<...>` are never real values; secrets (`session`, `wallet_key`) are never pasted into docs or logs.

Find a paid weather API (read-only, safe demo)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "wallet",
    "arguments": {
      "action": "discover",
      "query": "weather",
      "limit": 3
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{ "ok": true, "services": [ { "name": "...", "url": "https://...", "method": "GET", "price": {
  "amount": "0.01", "asset": "USDC", "chain": "base", "source": "circle" } }, "total": 123, "sources": {
  "paymentrequired": 0, "circle": 3 }, "next": "Pick one, then pay it with its method." }
```

Make / open the agent wallet (creates a guest wallet; no money moves)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "wallet",
    "arguments": {
      "action": "balance",
      "new": true
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{ "ok": true, "available": "0.00", "address": "0x...", "wallet_key": "<keep secret>", "approval_url": "https://<PUBLIC_ORIGIN>/s#...", "next": "Wallet ready: show address and balance. Link optional: makes it theirs." }
```

Pay a paid link — MONEY, separately authorized only

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "wallet",
```

```

    "arguments": {
      "action": "pay",
      "wallet_key": "<wallet_key>",
      "url": "https://example.com/paid",
      "max_amount": "0.10",
      "request_id": "pay-001"
    }
  }
}

```

Expected result (structuredContent, shape from code; values illustrative):

```

{"ok":true,"status":"done","receipt":{"tx":"0x...", "amount":"0.01", "to":"0x..."}, "resource":
{...}, "next":"Paid."}

```

Send USDC — MONEY, separately authorized only

```

{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "wallet",
    "arguments": {
      "action": "send",
      "session": "<session>",
      "to": "<authorized_recipient_address>",
      "amount": "1.00",
      "request_id": "send-001"
    }
  }
}

```

Expected result (structuredContent, shape from code; values illustrative):

```

{"ok":true,"status":"approval_required","approval_url":"https://...", "request_id":"send-
001", "next":"Show link. After tap: same call."}

```

Approval, signature and money

Guest wallet (no owner): the `wallet_key` is the authority, and `send/pay` run without a tap. Owned wallet: `send` waits for the owner's tap (`approval_required`). Owner limits (`account limits`) are checked before any signature, and over a limit the owner gets a one-time lift link (`wallet:src/mcp-wallet-tool.mjs:329`). Missing or old key → `connect_required` with a fresh link that re-derives the key (`wallet:src/mcp-wallet-tool.mjs:397`).

Failure, peer-down and retry

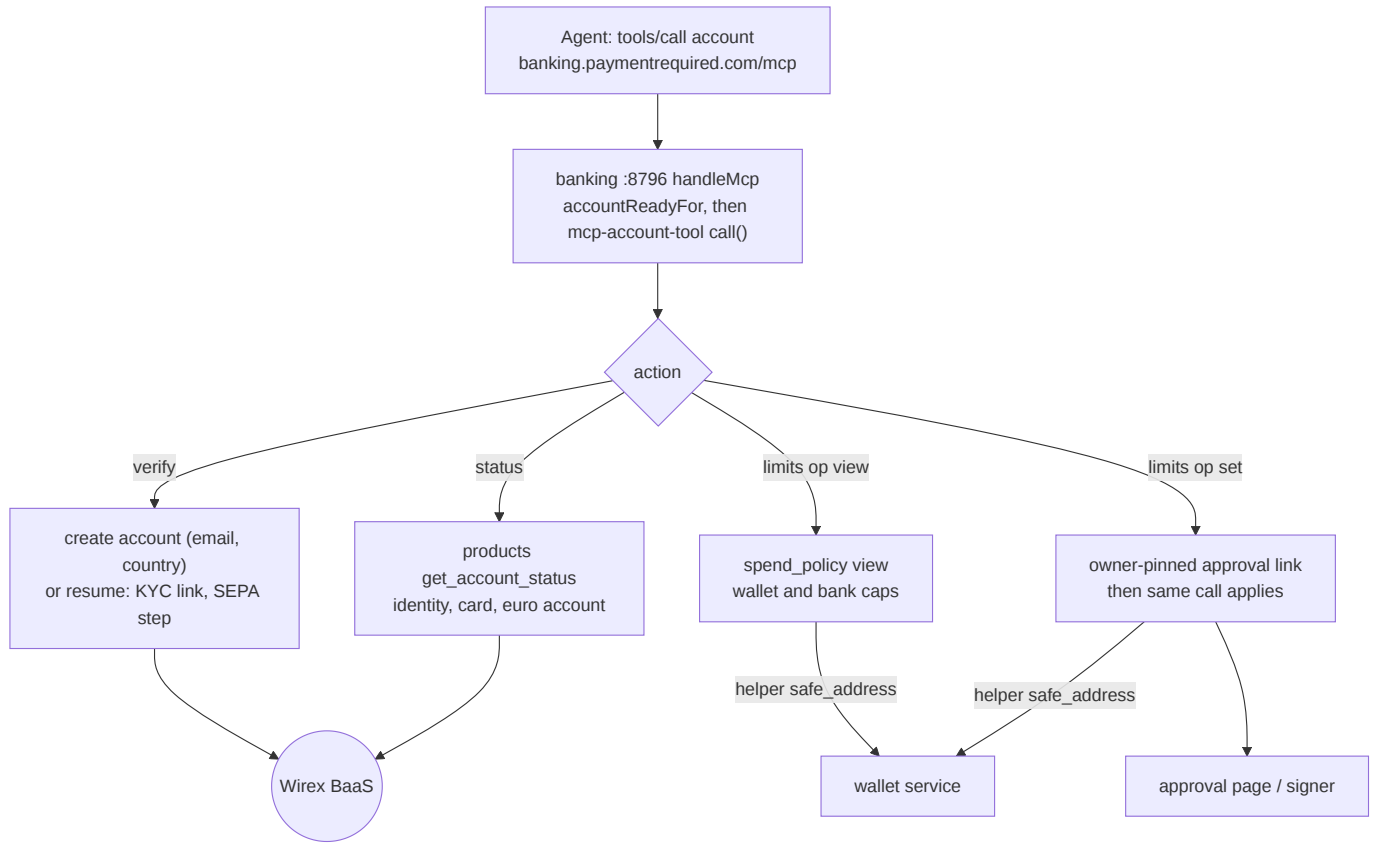
Every money action takes `request_id`, and repeating the same label continues the same operation; it never pays twice. Unconfirmed states (`pending`, `unknown`, `submitted`, `submitting`, `prepared`) answer `status:"pending"` with 'Same call in 10 s' (`wallet:src/mcp-wallet-tool.mjs:43`). `bridge` with a new label is blocked while another bridge is in flight (`wallet:src/mcp-wallet-tool.mjs:552`). If banking is down, `fund` from the bank and `reserve` return an error answer or are left out; the wallet itself keeps working (code-read, UNVERIFIED live).

Peer-down / retry contract (cross-service RPC). Calls between wallet and banking (and the apex directory's `helper/bridge` calls for issued links) go over loopback `POST /rpc/tool` with a bearer secret

(banking:src/service-rpc.mjs:50). A refused connection answers `service_unavailable` ... Nothing was sent (banking:src/service-rpc.mjs:65); any later failure answers `status: "unknown", service_unconfirmed` ... repeat the same call with the same `request_id` (banking:src/service-rpc.mjs:56). Rule for the agent: on `unknown`, repeat with the **same** `request_id`; never start a new one.

What was actually observed

FIXTURE: `wallet tools/list` schema equals the directory pin (probe-schema 3/3); fixture continuation and history calls passed in the accepted 99/99 run (no real Privy/Wirex/mainnet). LIVE-RO 2026-10-14T18:09Z: `wallet.paymentrequired.com/mcp` answers 503 'Service not deployed yet' (evidence/live-probes). Every money action live: UNVERIFIED (no money, no session used).



account **(Banking)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Who the person is: the identity check (KYC) with the card and bank provider, the readiness of the card and euro account, and the spending limits that let the agent pay without a tap.

Catalog: banking.paymentrequired.com/mcp (banking service tools/list: account, bank). **Schema & dispatch:** Input/output banking:src/mcp-account-tool.mjs:5-34; dispatch banking:src/mcp-account-tool.mjs:320-325. **Status enum** done, pending, approval_required, connect_required, failed banking:src/mcp-account-tool.mjs:26.

Actions / operations

ac tio n	what happens	effect	source
st at	Reads identity-check state: verified, card, account (euro IBAN), status, next, plus approval_url (KYC link) when one exists. An	read	banking:src/mcp-account-

action	what happens	effect	source
unknown provider user	→ pending, 'Call verify with email, country.'		tool.mjs:162
verify	Starts or resumes onboarding. On first use it needs email and country; they are never echoed back. Later calls report progress and hand out the KYC link. After KYC it drives the euro account (SEPA path wallet_link → signature link, standard → provisioning, wait). request_id makes it idempotent: a repeat never opens a second account.	WRITE (creates provider account; personal data)	banking:src/mcp-account-tool.mjs:174, banking:src/mcp-account-tool.mjs:125
limits	op: view (default) shows limits.wallet {per_payment, per_day, spent_24h} and limits.bank {per_transfer, per_day, spent_24h}, with null = none. op: set with any of wallet_per_payment, wallet_per_day, bank_per_transfer, bank_per_day (like 5.00), or `clear: ["wallet"	"bank"], returns an owner-pinned approval_url and arequest_id`. The same call after the tap applies it. The link expires after 24 h.	WRITE (permission change, owner tap)

Prerequisites

A connected session (connect link on first call). limits for the wallet side needs an existing agent wallet ('Call wallet balance first').

Session / connect (shared by wallet, account, bank). A call without session (or with a dead one) does not fail: the connection layer answers status: "connect_required" with an approval_url the person opens once, then the same call is repeated (banking:src/mcp-connection.mjs:233, banking:src/mcp-connection.mjs:109).

session must match SESSION_PATTERN banking:src/mcp-session.mjs:3. A connect reference passed as request_id (43 chars) without session is resolved to the session first (banking:src/mcp-connection.mjs:215). For account/bank on a handed-off identity, a one-time lazy bank grant link may appear first (banking:src/mcp-connection.mjs:153).

Per-action price. Off by default. If the operator sets TOOL_PRICES_JSON, a priced action is paid from the caller's agent wallet before it runs, and an unpaid charge means the action does not run (banking:src/tool-prices.mjs:1, banking:src/mcp-connection.mjs:293). On banking the charge goes to the wallet service over RPC; if that peer is down the answer is service_unavailable ... Nothing was charged (banking:src/server.mjs:170). Live price table: UNVERIFIED (env, not in repo).

How to use it (exact tools/call JSON)

Endpoint: POST https://banking.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

Check identity status (read-only)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "account",
    "arguments": {
      "action": "status",
      "session": "<session>"
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"ok":true,"verified":true,"status":"done","card":true,"account":true,"next":"All set."}
```

View limits (read-only)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "account",
    "arguments": {
      "action": "limits",
      "session": "<session>"
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"ok":true,"limits":{"wallet":null,"bank":null},"next":"No limits set. Set them with account limits, op set."}
```

Set limits (permission change, owner tap)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "account",
    "arguments": {
      "action": "limits",
      "op": "set",
      "session": "<session>",
      "wallet_per_payment": "5.00",
      "bank_per_day": "50.00",
      "request_id": "lim-001"
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"ok":true,"status":"approval_required","approval_url":"https://...", "request_id":"lim-001","next":"Show link. After tap: same call."}
```

Approval, signature and money

`limits set` is applied only after `bridge.approved()` verifies the pinned owner's signature (banking:src/mcp-account-tool.mjs:286). `verify` may surface the provider's KYC link and the euro-account signature link. No money moves in this tool.

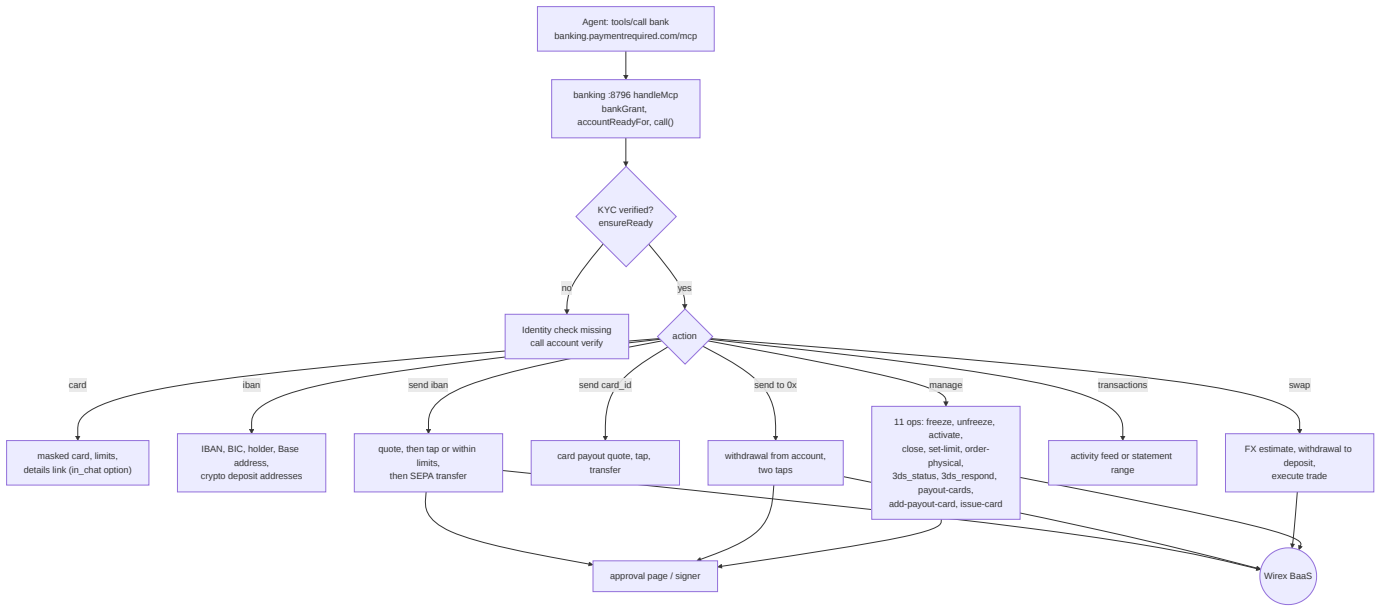
Failure, peer-down and retry

On banking, every call first runs `accountReadyFor` (bounded 20 s account advance for links approved on another service) (banking:src/server.mjs:135). If the wallet peer is down, `limits` raises the transport sentence instead of crashing (banking:src/server.mjs:157). The onboarding map is in memory; the provider record is the source of truth.

Peer-down / retry contract (cross-service RPC). Calls between wallet and banking (and the apex directory's helper/bridge calls for issued links) go over loopback `POST /rpc/tool` with a bearer secret (banking:src/service-rpc.mjs:50). A refused connection answers `service_unavailable` ... Nothing was sent (banking:src/service-rpc.mjs:65); any later failure answers `status: "unknown", service_unconfirmed` ... repeat the same call with the same `request_id` (banking:src/service-rpc.mjs:56). Rule for the agent: on unknown, repeat with the **same** `request_id`; never start a new one.

What was actually observed

FIXTURE: banking `tools/list account` schema equals the directory pin (probe-schema 3/3); `limits` with a retained apex pass answered on banking direct in the accepted 99/99 run (no real Privy/Wirex). LIVE-RO 2026-10-14T18:09Z: banking.paymentrequired.com/mcp answers 503 'Service not deployed yet'. No live call (needs a person's session): UNVERIFIED live.



bank **(Banking)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

The person's Visa card and euro account at the provider (Wirex). Read the balance, IBAN and deposit addresses; send euros by SEPA, to a saved card, or dollars to a 0x address; swap EUR ↔ USD; read history; manage cards.

Catalog: banking.paymentrequired.com/mcp (banking service tools/list: account, bank). **Schema & dispatch:** Input banking:src/mcp-bank-tool.mjs:87-123; ops banking:src/mcp-bank-tool.mjs:69, tap ops banking:src/mcp-bank-tool.mjs:71; dispatch banking:src/mcp-bank-tool.mjs:1021-1058; output = generic MCP_RESULT_SCHEMA banking:src/mcp-tool-results.mjs:36. card_number_last4 and expiry_date (#407 activate) are in the accepted banking code and the directory pin (FIXTURE schema probe equal). The pre-cutover live apex ed64a10 tools/list (evidence/live-probes) lacks them; that host is the old deployment, not this topology.

Actions / operations

act ion	what happens	effect	source
ca rd	Masked card(s) (card_id,masked,expiry,brand,state,limits), the balance, and a details link. Full numbers appear only on the approval page after the person signs. in_chat: true + request_id: after the tap and the signed chat consent, the same call returns the numbers once .	read (+ sensitive reveal behind tap)	banking:src/mcp-bank-tool.mjs:204
ib an	iban, bic, holder, currency EUR, the account's own Base address (USDC/EURC sent there counts as balance), deposit {addresses,	read	banking:src/mcp-bank-tool.mjs:242

act ion	what happens	effect	source
	networks, minimums}, plus balance/holdings.		
send (iban)	SEPA euros: iban, recipient_name (first + last), amount, optional reference ≤140, request_id. The BIC is derived from the IBAN. It quotes first: within limits it sends at once; otherwise it returns an owner-pinned tap link, and the same call sends. A fee move of cents after the tap re-quotes and sends without asking again.	MONEY MOVE (SEPA)	banking:src/mcp-bank-tool.mjs:303
send (card_id)	Euros to a saved payout card (card_id from manage payout-cards). The same quote → tap/limits → transfer pattern.	MONEY MOVE (card payout)	banking:src/mcp-bank-tool.mjs:572
send (to)	Dollars (WUSD → USDC, or WEUR → EURC if dollars are short) from the account's smart wallet to an address on Base, via a two-phase withdrawal (two taps). Needs KYC only, not the euro account.	MONEY MOVE (on-chain, two taps)	banking:src/mcp-bank-tool.mjs:463
swap	EUR ↔ USD inside the account (*to: USD	EUR, amount ≥ 10). It shows the rate, receive and deposit; it withdraws into the provider's deposit address (taps), then executes a fresh estimate once.	MONEY MOVE (FX)
transaction	Card + bank activity, 10 per page. With start_date/end_date (YYYY-MM-DD, at most 6 months) it returns a statement instead.	read	banking:src/mcp-bank-tool.mjs:651
manage	op = freeze, unfreeze (run on first call; target by card_id or rule), activate (physical card; card_id + card_number_last4 + expiry_date MM/YYYY), set-limit (limits{daily_limit, monthly_limit, transaction_limit}), 3ds_status (list pending purchases), 3ds_respond (transaction_id, approve), payout-cards (read), and tap ops close, add-payout-card (card_number, cardholder_name, card_label), issue-card (limits, currency), order-physical (without address → lists card_types fees and delivery providers; with country, delivery_provider, name_on_card, delivery_address, card_type → pays the card fee from the account (taps) and orders).	WRITE / MONEY (card fee) / IRREVERSIBLE (close)	banking:src/mcp-bank-tool.mjs:980, activate banking:src/mcp-bank-tool.mjs:1004, order banking:src/mcp-bank-tool.mjs:913, tap banking:src/mcp-bank-tool.mjs:952

Not public: cardApproved is a bridge callback for the details page, **not** a tool action (banking:src/mcp-bank-tool.mjs:1064).

Prerequisites

session (required by schema). KYC verified (account verify). Euro account active for iban, send iban; card active for card. send to/swap need KYC only.

Session / connect (shared by `wallet`, `account`, `bank`). A call without `session` (or with a dead one) does not fail: the connection layer answers `status: "connect_required"` with an `approval_url` the person opens once, then the same call is repeated (`banking:src/mcp-connection.mjs:233`, `banking:src/mcp-connection.mjs:109`). `session` must match `SESSION_PATTERN` `banking:src/mcp-session.mjs:3`. A connect reference passed as `request_id` (43 chars) without `session` is resolved to the session first (`banking:src/mcp-connection.mjs:215`). For `account/bank` on a handed-off identity, a one-time *lazy bank grant* link may appear first (`banking:src/mcp-connection.mjs:153`).

Per-action price. Off by default. If the operator sets `TOOL_PRICES_JSON`, a priced action is paid from the caller's agent wallet *before* it runs, and an unpaid charge means the action does not run (`banking:src/tool-prices.mjs:1`, `banking:src/mcp-connection.mjs:293`). On banking the charge goes to the wallet service over RPC; if that peer is down the answer is `service_unavailable` ... Nothing was charged (`banking:src/server.mjs:170`). Live price table: UNVERIFIED (env, not in repo).

How to use it (exact `tools/call` JSON)

Endpoint: POST `https://banking.paymentrequired.com/mcp` (JSON-RPC 2.0). Placeholders in `<...>` are never real values; secrets (`session`, `wallet_key`) are never pasted into docs or logs.

Show IBAN and deposit addresses (read-only)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "bank",
    "arguments": {
      "action": "iban",
      "session": "<session>"
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{
  "ok": true,
  "status": "done",
  "iban": "DE...",
  "bic": "...",
  "holder": "...",
  "currency": "EUR",
  "address": "0x...",
  "deposit": {
    "addresses": [...],
    "networks": [...],
    "minimums": [...],
    "balance": "12.34",
    "next": "Add money: IBAN or deposit address."
  }
}
```

Show the card, masked (read-only)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "bank",
    "arguments": {
      "action": "card",
      "session": "<session>"
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{ "ok":true,"status":"done","card":{"card_id":"...", "masked":"**** *  ****  
1234","expiry":"...", "brand":"...", "state":"active"}, "approval_url":"https://...", "next":"Show link: card  
details."}
```

Recent transactions (read-only)

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "bank",
    "arguments": {
      "action": "transactions",
      "session": "<session>",
      "page": 1
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{ "ok":true,..., "next":"Shown." }
```

SEPA transfer – MONEY, separately authorized only

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "bank",
    "arguments": {
      "action": "send",
      "session": "<session>",
      "iban": "DE00...",
      "recipient_name": "Erika Mustermann",
      "amount": "12.50",
      "reference": "Rent",
      "request_id": "sepa-001"
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{
  "ok": true,
  "status": "approval_required",
  "amount": "12.50",
  "fee": "0.44",
  "currency": "EUR",
  "approval_url": "https://...",
  "request_id": "sepa-001",
  "next": "Show link. After tap: same call."
}
```

Activate a physical card – card activation, separately authorized only

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "bank",
    "arguments": {
      "action": "manage",
      "op": "activate",

```

```

    "session": "<session>",
    "card_id": "<card_id>",
    "card_number_last4": "1234",
    "expiry_date": "09/2029"
  }
}
}

```

Expected result (`structuredContent`, shape from code; values illustrative):

```

{"ok":true,"status":"done","card_id":"...", "card":{"masked":"**** * 1234",...}, "next":"Activate done."}

```

Approval, signature and money

Money and irreversible ops create an owner-pinned approval session (`expectedAddress: owner`), and the result is accepted only after `bridge.approved()` verifies that owner's signature (`banking:src/mcp-bank-tool.mjs:362`). SEPA and card payouts within `account limits` run without a tap (`banking:src/mcp-bank-tool.mjs:296`). `send to, swap` and the physical-card fee use the account withdrawal's own two taps.

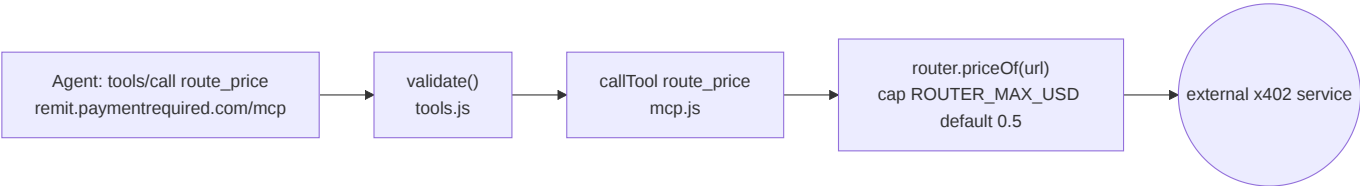
Failure, peer-down and retry

`request_id` binds a quote/transfer in SQLite `mcp_bank_requests`, so a restart cannot cause a second send (`banking:src/mcp-bank-tool.mjs:14`). Unclear outcomes answer `pending`, 'Call send again with the same `request_id` ... Do not start a second transfer'. An expired approval link → `failed`, 'Start again with a new `request_id`'. Card reveals are RAM-only (15 min) and lost on restart.

Peer-down / retry contract (cross-service RPC). Calls between wallet and banking (and the apex directory's `helper/bridge` calls for issued links) go over loopback `POST /rpc/tool` with a bearer secret (`banking:src/service-rpc.mjs:50`). A refused connection answers `service_unavailable` ... Nothing was sent (`banking:src/service-rpc.mjs:65`); any later failure answers `status: "unknown", service_unconfirmed` ... repeat the same call with the same `request_id` (`banking:src/service-rpc.mjs:56`). Rule for the agent: on `unknown`, repeat with the **same** `request_id`; never start a new one.

What was actually observed

FIXTURE: `banking tools/list bank` schema equals the directory pin incl. #407 fields (probe-schema 3/3), not real Wirex. LIVE-RO 2026-10-14T18:09Z: banking host 503 'Service not deployed yet'; old apex `ed64a10` still lists `bank` without #407 fields (pre-cutover deployment). No call executed; no card activated. Everything else UNVERIFIED live.



route_price **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Reads the price of an external x402 service (url, optional method GET/POST, body) and returns what the target wants plus what you pay through the router (markup, cap). No funds move. Local/private network addresses are refused.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:19](#); dispatch [remit:src/mcp.js:476](#)

Actions / operations

action	what happens	effect	source
route_price	Quote for an external x402 service	read	remit:src/mcp.js:476

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST <https://remit.paymentrequired.com/mcp> (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

route_price

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "route_price",
    "arguments": {
      "url": "https://example.com/paid"
    }
  }
}
```

```
}  
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"free":false,"targetWants":{"amount":"0.010000","asset":"USDC","network":"..."},"youPay":  
{"atomic":..., "usd":"0.011000"}}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"route_price", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

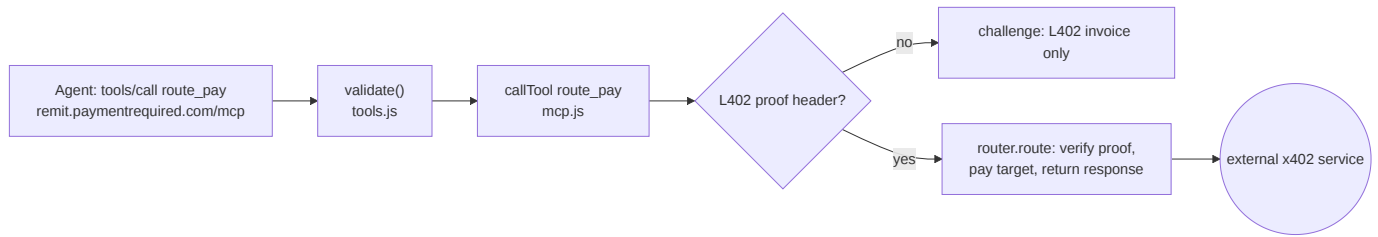
No approval or signature; no money moves.

Failure, peer-down and retry

`above_limit` when the price exceeds the router cap ([remit:src/router.js:53](#)); `target_unreachable` otherwise. Repeat freely.

What was actually observed

UNVERIFIED: not called by this handbook.



route_pay **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

For agents with native L402: the first call returns a Lightning invoice challenge; after paying, call again with Authorization: L402 <macaroon>:<preimage>; the router pays the external x402 service and returns its response.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:35](#); dispatch [remit:src/mcp.js:489](#)

Actions / operations

action	what happens	effect	source
route_pay	Pay an external service via Lightning L402	MONEY MOVE (agent pays Lightning; router pays target)	remit:src/mcp.js:489

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

route_pay

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "route_pay",
    "arguments": {
      "url": "https://example.com/paid"
```

```
}  
}  
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"target":"https://...", "targetWants":{"..."}, "payWith":[{"scheme":"l402",...}], "nextAction":"Pay the  
Lightning invoice with a native L402 client, then call route_pay again with Authorization: L402  
<macaroon>:<preimage>. Pay only once."}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"route_pay", ...arguments}`; a payment request then comes back as HTTP 402 (remitsrc.com/mcp.js:799).

Approval, signature and money

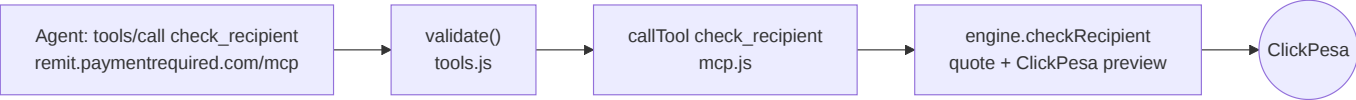
The L402 preimage is the payment proof; only the L402 scheme is accepted here. Two proofs at once → `multiple_proofs`.

Failure, peer-down and retry

Pay only once; repeat the call with the same proof header.

What was actually observed

UNVERIFIED: not called by this handbook.



check_recipient **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Before paying: recipient name, provider (M-Pesa, Tigo Pesa, Airtel, Halopesa), ClickPesa fee, our fee bps and crypto prices for phone + amountTZS. Free; nothing moves.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:51](#); dispatch [remit:src/mcp.js:211](#)

Actions / operations

action	what happens	effect	source
check_recipient	Check a Tanzanian mobile recipient and price	read	remit:src/mcp.js:211

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

check_recipient

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "check_recipient",
    "arguments": {
      "phone": "+255712345678",
      "amountTZS": 5000
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{ "phone": "+255712345678", "amountTZS": 5000, "recipient":  
{ "name": "...", "provider": "...", "checked": true }, "fees": { "clickpesaTZS": ..., "ourFeeBps": 500 }, "pay":  
[...], "rate": { ... }, "payWith": [...], "warning": null, "nextAction": "The money will go to \"...\". If this is  
correct, call send_tzs with the same details." }
```

Plain-JSON form (same handler): POST /mcp with {"tool": "check_recipient", ...arguments}; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

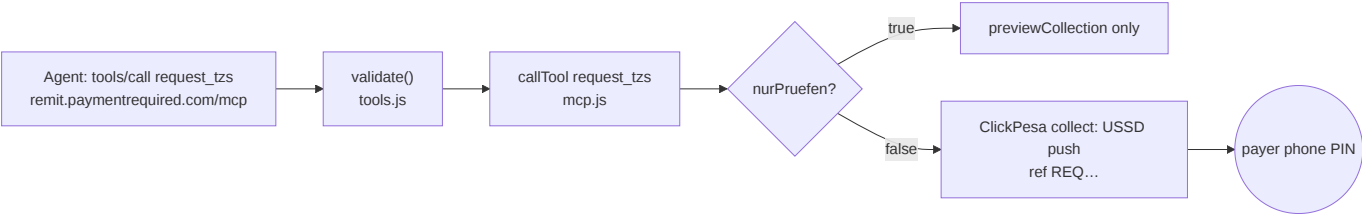
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

UNVERIFIED: not called by this handbook.



request_tzs **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Sends a USSD push to phone asking its owner to pay amountTZS into the operator's ClickPesa account (owner approves with their mobile-money PIN). nurPruefen: true only previews providers/fees; no prompt is sent.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:75](#); dispatch [remit:src/mcp.js:290](#)

Actions / operations

action	what happens	effect	source
request_tzs	Request money from a Tanzanian mobile number	MONEY MOVE (collection INTO operator account, payer PIN)	remit:src/mcp.js:290

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

request_tzs

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "request_tzs",
    "arguments": {
      "phone": "+255712345678",
      "amountTZS": 5000,
      "nurPruefen": true
    }
  }
}
```

```
}  
}  
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"ok":true,"nurGeprueft":true,"amountTZS":5000,"phone":"+255712345678","senderName":"...","senderCarrier":"...","carrierActive":true,"activeMethods":[...],"warning":null}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"request_tzs", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

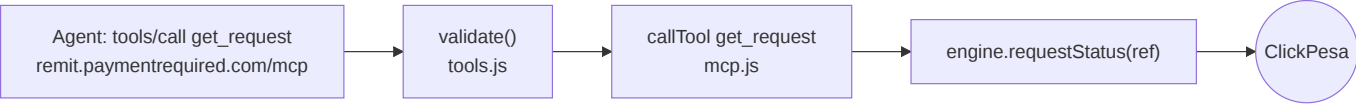
The payer's mobile-money PIN on their own phone is the approval. Without it no funds move.

Failure, peer-down and retry

Each non-preview call mints a new `ref (REQ...)` ([remit:src/engine.js:367](#)): repeating sends a **second** prompt. Check with `get_request`.

What was actually observed

UNVERIFIED: not called by this handbook.



get_request **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Whether a request_tzs collection was approved, is pending, or failed (ref).

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:105](#); dispatch [remit:src/mcp.js:314](#)

Actions / operations

action	what happens	effect	source
get_request	Status of a money request	read	remit:src/mcp.js:314

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

get_request

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "get_request",
    "arguments": {
      "ref": "REQM2K4X1"
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"ref":"REQM2K4X1",...ClickPesa collection status...}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"get_request", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

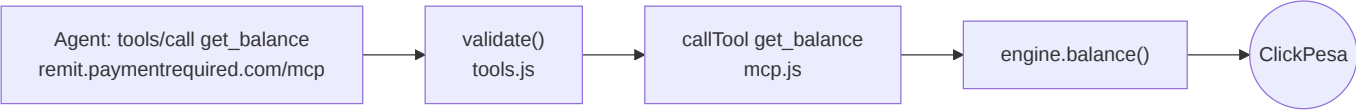
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

UNVERIFIED: not called by this handbook.



get_balance **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

ClickPesa float (availableTZS) and product limits (bank 21,000,000 TZS; mobile/merchant 3,000,000 TZS per transfer). Float is not a transfer maximum.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:123](#); dispatch [remit:src/mcp.js:356](#)

Actions / operations

action	what happens	effect	source
get_balance	Available payout liquidity	read	remit:src/mcp.js:356

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

get_balance

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "get_balance",
    "arguments": {}
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"ok":true,"availableTZS":...,"currency":"TZS","limits":{"minTZS":100,"maxTZSByDestination":{...}},"liquidityNote":"...","minSwapTZS":100,"at":"..."}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"get_balance", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

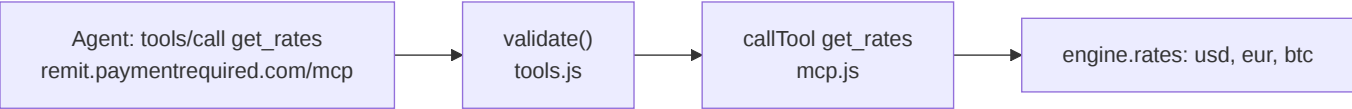
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: returned JSON-RPC error `balance_unavailable` (ClickPesa generate-token HTTP 403; evidence/live-probes/remit-readonly-calls.json) — the tool is reachable, the upstream provider refused/rate-limited at capture time. Success shape: UNVERIFIED live.



get_rates **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Raw rates: TZS per USD, TZS per EUR, USD per BTC, with sources. Free.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:134](#); dispatch [remit:src/mcp.js:324](#)

Actions / operations

action	what happens	effect	source
get_rates	Current exchange rates	read	remit:src/mcp.js:324

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST <https://remit.paymentrequired.com/mcp> (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

get_rates

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "get_rates",
    "arguments": {}
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"ok":true,"tzsPerUsd":...,"tzsPerEur":...,"usdPerBtc":...,"sources":{"..."},"at":"..."}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool": "get_rates", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

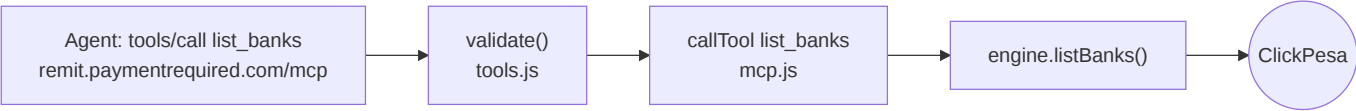
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: answered; keys `at`, `ok`, `sources`, `tzsPerEur`, `tzsPerUsd`, `usdPerBtc` (evidence/live-probes/remit-readonly-calls.json).



list_banks **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Supported Tanzanian banks with names and BIC codes (for check_bank/send_tzs).

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:145](#); dispatch [remit:src/mcp.js:227](#)

Actions / operations

action	what happens	effect	source
list_banks	List Tanzanian banks	read	remit:src/mcp.js:227

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call **JSON**)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session,wallet_key) are never pasted into docs or logs.

list_banks

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "list_banks",
    "arguments": {}
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{ "ok":true, "banks": [{"name":"CRDB Bank", "bic":"CORUTZTZ"}, ...] }
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool": "list_banks", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

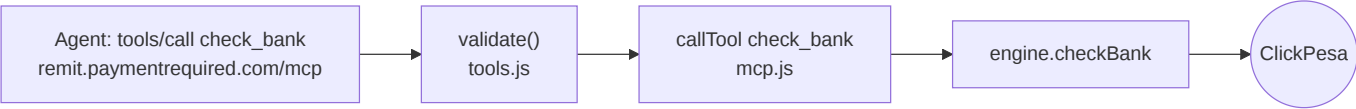
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: returned JSON-RPC error `banks_unavailable` (ClickPesa generate-token HTTP 403; evidence/live-probes/remit-readonly-calls.json) — the tool is reachable, the upstream provider refused/rate-limited at capture time. Success shape: UNVERIFIED live.



check_bank **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Account holder, bank name, fee, estimated arrival and crypto prices for amountTZS, accountNumber, bic (optional accountName). Free.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:156](#); dispatch [remit:src/mcp.js:241](#)

Actions / operations

action	what happens	effect	source
check_bank	Check a Tanzanian bank account	read	remit:src/mcp.js:241

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

check_bank

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "check_bank",
    "arguments": {
      "amountTZS": 50000,
      "accountNumber": "01J1234567890",
      "bic": "CORUTZTZ"
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{ "ok": true, "amountTZS": 50000, "destination":  
  { "type": "bank", ... }, "bankName": "...", "accountHolder": "...", "feeTZS": ..., "estimatedArrival": "...", "quote":  
  [...], "warning": null }
```

Plain-JSON form (same handler): `POST /mcp` with `{ "tool": "check_bank", ...arguments }`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

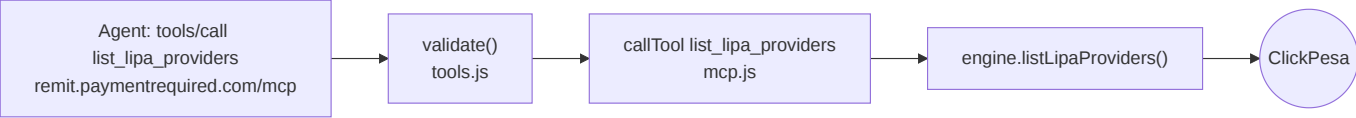
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

UNVERIFIED: not called by this handbook.



list_lipa_providers **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Lipa Namba providers and their three-digit providerCode (e.g. M-Pesa 503).

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:188](#); dispatch [remit:src/mcp.js:261](#)

Actions / operations

action	what happens	effect	source
list_lipa_providers	List Lipa Namba / TanQR providers	read	remit:src/mcp.js:261

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

list_lipa_providers

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "list_lipa_providers",
    "arguments": {}
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"ok":true,"providers":[...]}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool": "list_lipa_providers", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

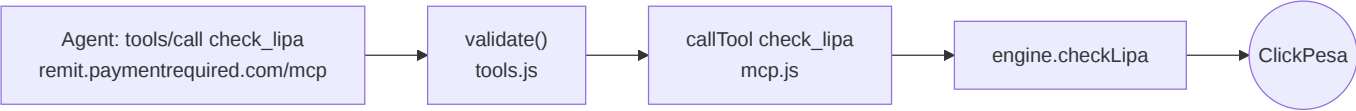
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: returned JSON-RPC error `providers_unavailable` (ClickPesa generate-token HTTP 429; evidence/live-probes/remit-readonly-calls.json) — the tool is reachable, the upstream provider refused/rate-limited at capture time. Success shape: UNVERIFIED live.



check_lipa **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Merchant name, provider, fees and crypto prices for amountTZS + (lipaNamba + providerCode) or qrCode. Free.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:199](#); dispatch [remit:src/mcp.js:270](#)

Actions / operations

action	what happens	effect	source
check_lipa	Check a merchant (Lipa Namba or TanQR)	read	remit:src/mcp.js:270

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

check_lipa

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "check_lipa",
    "arguments": {
      "amountTZS": 5000,
      "lipaNamba": "48001268",
      "providerCode": "503"
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{ "ok": true, "amountTZS": 5000, "destination":  
{ "type": "lipa_namba", ... }, "merchantName": "...", "provider": "...", "feeTZS": ..., "quote": [...] }
```

Plain-JSON form (same handler): `POST /mcp` with `{ "tool": "check_lipa", ...arguments }`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

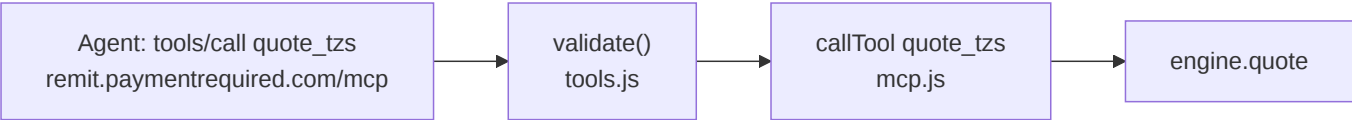
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

UNVERIFIED: not called by this handbook.



quote_tzs **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Price per payment rail for amountTzs (customerPays, pay[], rate, feeBps, validForSeconds 1800, limits, payWith). Free.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:229](#); dispatch [remit:src/mcp.js:195](#)

Actions / operations

action	what happens	effect	source
quote_tzs	Payout quote without a recipient	read	remit:src/mcp.js:195

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

quote_tzs

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "quote_tzs",
    "arguments": {
      "amountTzs": 5000
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{ "amountTZS":5000, "customerPays":{"USDC":"..."}, "pay":[...], "rate":
{...}, "feeBps":500, "validForSeconds":1800, "limits":{"..."}, "payWith":[...] }
```

Plain-JSON form (same handler): `POST /mcp` with `{ "tool": "quote_tzs", ...arguments }`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

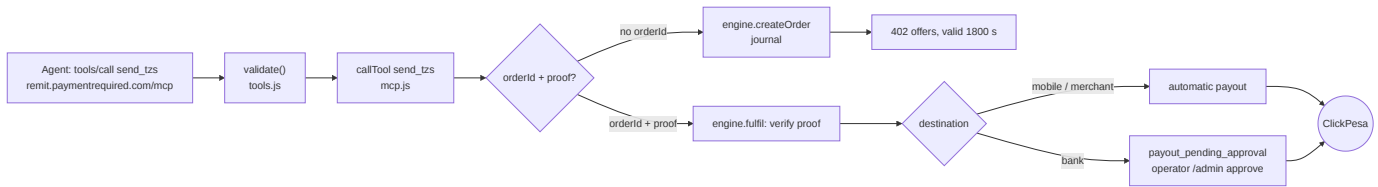
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: `amountTZS: 5000 answered; keys amountTZS, customerPays, feeBps, limits, pay, payWith, rate, validForSeconds.`



send_tzs (Remit)

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Creates a payout order to mobile money (phone), bank (accountNumber+bic), Lipa Namba (lipaNamba+providerCode) or TanQR (qrCode). The first call returns a payment request (HTTP 402 on plain JSON; MCP result with status: awaiting_payment) with orderId, recipient name, payWith offers (each with automaticDetection), valid 30 min. People then pay once via get_payment; L402/x402 agents may retry with the same orderId + amountTzs and a proof header. Mobile/merchant payouts are automatic after verified payment; bank payouts accept USDC only and need operator approval.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:247](#); dispatch [remit:src/mcp.js:539](#)

Actions / operations

action	what happens	effect	source
send_tzs	Send Tanzanian shillings	MONEY MOVE (customer pays crypto → TZS payout)	remit:src/mcp.js:539

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

send_tzs

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "send_tzs",
    "arguments": {
```

```

    "amountTZS": 5000,
    "phone": "+255712345678"
  }
}

```

Expected result (`structuredContent`, shape from code; values illustrative):

```

{"orderId":"NURI-...", "status":"awaiting_payment", "validForSeconds":1800, "amountTZS":5000, "recipient":{
  "name":"...", "provider":"...", "checked":true}, "payWith":
  [{..., "automaticDetection":true}], "nextAction":"The money will go to \"...\". If this is correct, call
  get_payment ..."}

```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"send_tzs", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

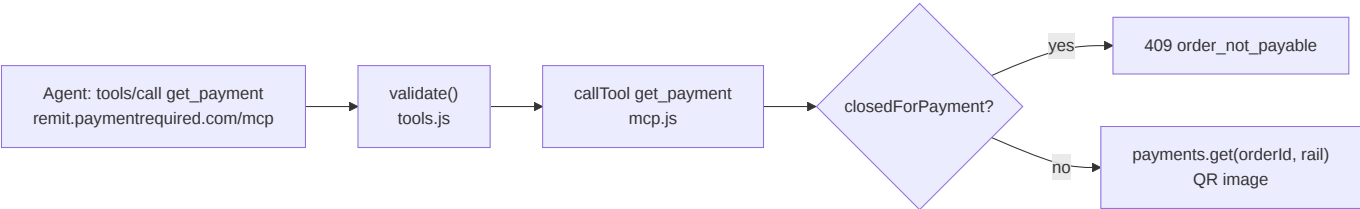
Money is paid by the agent/person into the offered rail (USDC/EURC on Base, BTC, Lightning, or x402/L402 proof); payout to Tanzania follows **after** the incoming payment is verified. Bank payouts wait for an operator approve/reject in the authenticated `/admin` dashboard ([remit:src/admin-dashboard.js:172](#)).

Failure, peer-down and retry

A retry with `orderId` never creates a new order or re-bills a paid one; amount/destination mismatch → `invalid_arguments` ([remit:src/mcp.js:564](#)). Never pay again once `get_order` leaves `awaiting_payment`.

What was actually observed

UNVERIFIED: not called by this handbook.



get_payment (Remit)

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Exact amount, address/invoice and QR (image content) for orderId on rail usdc_base (default), eurc_base, btc_onchain, lightning. Lightning only when automatic detection is proven; paid/recovered/expired orders never return details.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). **Schema & dispatch:** Definition [remit:src/tools.js:315](#); dispatch [remit:src/mcp.js:381](#)

Actions / operations

action	what happens	effect	source
get_payment	Payment details and QR for an unpaid order	read (shows where to pay)	remit:src/mcp.js:381

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

get_payment

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "get_payment",
    "arguments": {
      "orderId": "NURI-00000000-0000-4000-8000-000000000001",
      "rail": "usdc_base"
    }
  }
}
```

```
}  
}  
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"ok":true,"orderId":"NURI-...","status":"awaiting_payment","rail":"usdc_base","asset":"USDC","network":"...","amount":"...","amountAtomic":"...","payTo":"0x...","automaticDetection":true} + image/png content
```

Plain-JSON form (same handler): POST `/mcp` with `{"tool":"get_payment", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

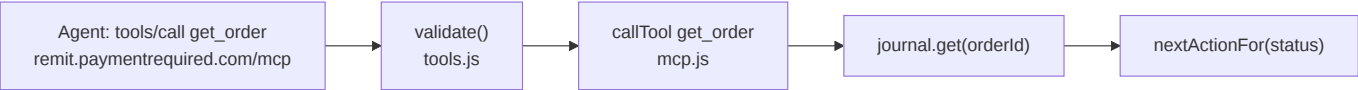
Money is paid by the agent/person into the offered rail (USDC/EURC on Base, BTC, Lightning, or x402/L402 proof); payout to Tanzania follows **after** the incoming payment is verified.

Failure, peer-down and retry

Read-only; repeat freely. `order_not_payable` (409) once paid.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: unknown id → JSON-RPC error `unknown_order`. Real order: UNVERIFIED.



get_order **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Status of orderId: awaiting_payment, payment_verified, paid, payout_processing, payout_sent, payout_pending_review, payout_pending_approval, payout_rejected, payout_scheduled, payout_failed, unknown, plus incomingPaymentVerified, delivery, schedule, nextAction. Never confirms a payment by itself.

Catalog: remit.paymentrequired.com/mcp (own MCP, independent). Schema & dispatch: Definition [remit:src/tools.js:328](#); dispatch [remit:src/mcp.js:416](#)

Actions / operations

action	what happens	effect	source
get_order	Order status	read	remit:src/mcp.js:416

Prerequisites

No account, registration or API key (instructions text).

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

get_order

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "get_order",
    "arguments": {
      "orderId": "NURI-00000000-0000-4000-8000-000000000001"
    }
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{ "orderId": "NURI-...", "status": "payout_sent", "amountTZS": 5000, "incomingPaymentVerified": true, "payoutAuthorization": "verified_payment", "delivery": {...}, "nextAction": "Delivered. ..." }
```

Plain-JSON form (same handler): `POST /mcp` with `{ "tool": "get_order", ...arguments }`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

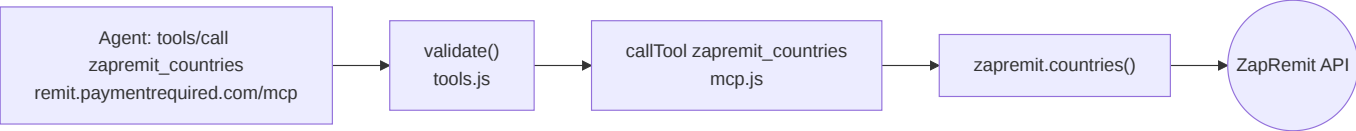
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: unknown id → JSON-RPC error `order_not_found` (as coded). Real order: UNVERIFIED.



zapremit_countries **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Countries ZapRemit pays out to (live provider catalog).

Catalog: remit.paymentrequired.com/mcp (ZapRemit sub-catalog). **Schema & dispatch:** Definition [remit:src/zapremit-tools.js:18](#); dispatch [remit:src/zapremit.js:554](#)

Actions / operations

action	what happens	effect	source
zapremit_countries	ZapRemit countries	read	remit:src/zapremit.js:554

Prerequisites

No account, registration or API key (instructions text). ZapRemit must be configured on the server, otherwise 503 provider_unavailable.

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

zapremit_countries

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "zapremit_countries",
    "arguments": {}
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"provider":"zapremit","data":[{"code":"...","name":"...","currency":"..."}]}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"zapremit_countries", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: answered; keys `data`, `provider`.



zapremit_methods **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Payout methods for `country` with recipient/sender fields, min/max and currency.

Catalog: remit.paymentrequired.com/mcp (ZapRemit sub-catalog). **Schema & dispatch:** Definition [remit:src/zapremit-tools.js:24](#); dispatch [remit:src/zapremit.js:555](#)

Actions / operations

action	what happens	effect	source
zapremit_methods	ZapRemit payout methods	read	remit:src/zapremit.js:555

Prerequisites

No account, registration or API key (instructions text). ZapRemit must be configured on the server, otherwise 503 `provider_unavailable`.

How to use it (exact `tools/call` JSON)

Endpoint: `POST https://remit.paymentrequired.com/mcp` (JSON-RPC 2.0). Placeholders in `<...>` are never real values; secrets (`session`, `wallet_key`) are never pasted into docs or logs.

zapremit_methods

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "zapremit_methods",
    "arguments": {
      "country": "PH"
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"provider":"zapremit","data":  
[{"code":"...","active":true,"currency":"...","min":...,"max":...,"recipientFields":{...}}]}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"zapremit_methods", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

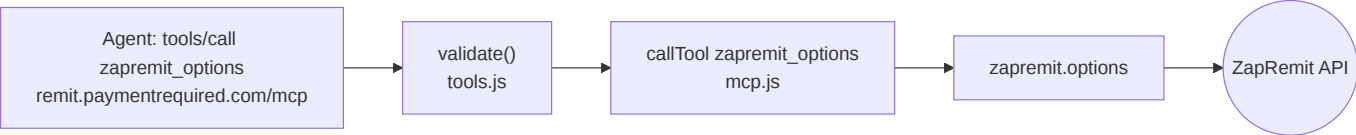
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: `country: PH answered; keys data, provider`.



zapremit_options **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Allowed values for fields marked options:true (e.g. institutionCode) for country + methodCode.

Catalog: remit.paymentrequired.com/mcp (ZapRemit sub-catalog). **Schema & dispatch:** Definition [remit:src/zapremit-tools.js:30](#); dispatch [remit:src/zapremit.js:556](#)

Actions / operations

action	what happens	effect	source
zapremit_options	ZapRemit field options	read	remit:src/zapremit.js:556

Prerequisites

No account, registration or API key (instructions text). ZapRemit must be configured on the server, otherwise 503 provider_unavailable.

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

zapremit_options

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "zapremit_options",
    "arguments": {
      "country": "PH",
      "methodCode": "<from zapremit_methods>"
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"provider":"zapremit","institutionCode":[...]}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"zapremit_options", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

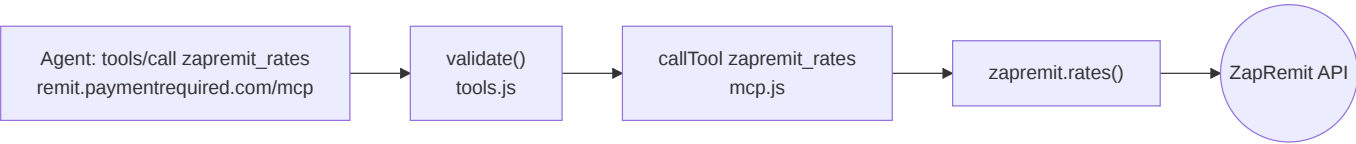
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

UNVERIFIED: not called by this handbook.



zapremit_rates **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Reference BTC exchange rates from ZapRemit.

Catalog: remit.paymentrequired.com/mcp (ZapRemit sub-catalog). **Schema & dispatch:** Definition [remit:src/zapremit-tools.js:36](#); dispatch [remit:src/zapremit.js:557](#)

Actions / operations

action	what happens	effect	source
zapremit_rates	ZapRemit BTC rates	read	remit:src/zapremit.js:557

Prerequisites

No account, registration or API key (instructions text). ZapRemit must be configured on the server, otherwise 503 provider_unavailable.

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

zapremit_rates

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "zapremit_rates",
    "arguments": {}
  }
}
```

Expected result (structuredContent, shape from code; values illustrative):

```
{"provider": "zapremit", "BTC": {"USD": ..., "PHP": ..., ...}}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool": "zapremit_rates", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

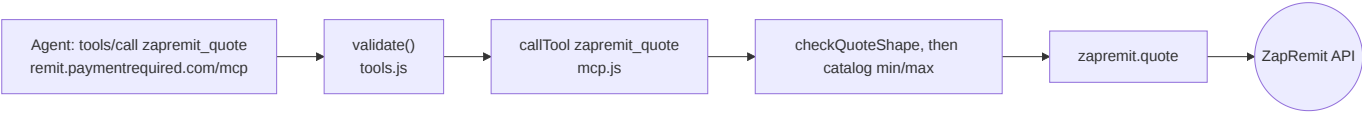
No approval or signature; no money moves.

Failure, peer-down and retry

Stateless read; repeat freely. Provider errors are returned as JSON-RPC error -32602 with `data.code`.

What was actually observed

LIVE-RO 2026-10-14T18:10Z @212886c: answered; keys `BTC`, `provider`.



zapremit_quote **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Creates a payout quote: country, methodCode, amount (positive integer in method currency or SAT), currency, recipient, optional sender, description ≤128, referenceId (client lowercase UUID). Same referenceId + same details → same order; changed details → rejected. Does not move funds.

Catalog: remit.paymentrequired.com/mcp (ZapRemit sub-catalog). **Schema & dispatch:** Definition [remit:src/zapremit-tools.js:42](#); dispatch [remit:src/zapremit.js:558](#)

Actions / operations

action	what happens	effect	source
zapremit_quote	ZapRemit quote (locks rate)	WRITE (provider quote/order record; no funds)	remit:src/zapremit.js:558

Prerequisites

No account, registration or API key (instructions text). ZapRemit must be configured on the server, otherwise 503 provider_unavailable.

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

zapremit_quote

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "zapremit_quote",
    "arguments": {
      "country": "PH",
      "methodCode": "<code>",
      "amount": 1000,
      "currency": "PHP",
```

```
    "recipient": {
      "...": "fields from zapremit_methods"
    },
    "referenceId": "00000000-0000-4000-8000-000000000000"
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"provider":"zapremit","orderId":"zap_...", "referenceId":"...", "status":"created", "payout":
{...}, "quoteExpiresAt":"...", "incomingPaymentVerified":false, "delivery":"not_started"}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"zapremit_quote", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

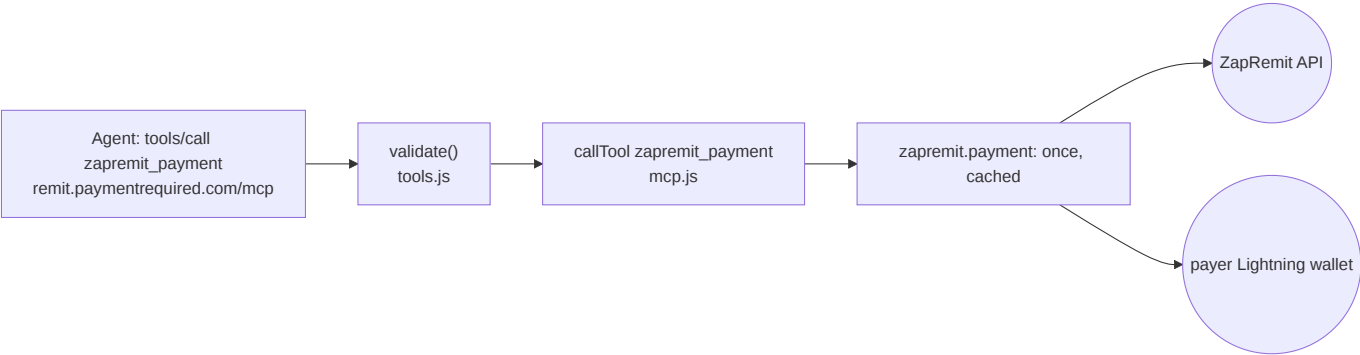
No approval or signature; no money moves.

Failure, peer-down and retry

Idempotent on `referenceId`; an unknown provider outcome is journalled as `unknown`, never re-sent.

What was actually observed

UNVERIFIED: not called by this handbook.



zapremit_payment (Remit)

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Asks ZapRemit once for the Lightning invoice funding orderId, returns it with a QR (invoice, paymentUri, automaticDetection:true, qr{svgUrl,pngUrl}) while payable. The payer pays from their own wallet; the server never pays it.

Catalog: remit.paymentrequired.com/mcp (ZapRemit sub-catalog). **Schema & dispatch:** Definition [remit:src/zapremit-tools.js:65](#); dispatch [remit:src/zapremit.js:559](#)

Actions / operations

action	what happens	effect	source
zapremit_payment	ZapRemit Lightning invoice	read (shows where to pay; payer moves money)	remit:src/zapremit.js:559

Prerequisites

No account, registration or API key (instructions text). ZapRemit must be configured on the server, otherwise 503 provider_unavailable.

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

zapremit_payment

```
{
  "jsonrpc": "2.0",
```

```

{id": 1,
"method": "tools/call",
"params": {
  "name": "zapremit_payment",
  "arguments": {
    "orderId": "zap_00000000-0000-4000-8000-000000000000"
  }
}
}

```

Expected result (`structuredContent`, shape from code; values illustrative):

```

{"provider":"zapremit","orderId":"zap_...", "status":"pending", "amountSats":..., "invoice":"lnbc...", "paymentUri":"lightning:...", "automaticDetection":true, "qr":{"svgUrl":"...", "pngUrl":"..."}}

```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"zapremit_payment", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

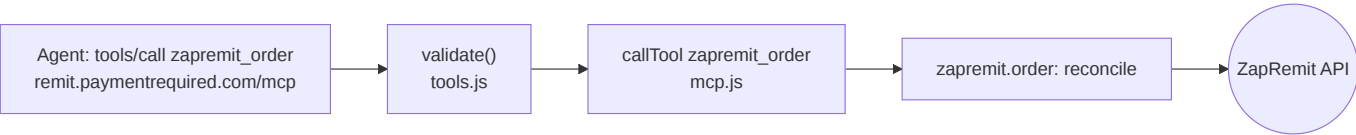
The payer's own Lightning wallet pays; nothing is signed by this server.

Failure, peer-down and retry

Repeated calls return the same invoice; outcome-unknown → 409 `payment_unknown`, not re-sent.

What was actually observed

UNVERIFIED: not called by this handbook.



zapremit_order **(Remit)**

[← Index](#) · [Overview](#) · [Signer](#) · [Issued links](#)

Evidence labels: **CODE** = read in the cited file at the recorded revision/hash; **FIXTURE** = parent-accepted combined cloud fixture run 99/99 (directory-wave/parent-directory-wave-acceptance.json), not live; **LIVE-RO** = observed read-only on the live host 2026-10-14 (raw files in evidence/live-probes/); **UNVERIFIED** = not executed here. Topology = accepted directory-wave source, **not deployed**. Nothing in this handbook moved money, signed, or activated a card.

What it does

Reads payout status from ZapRemit (created, pending, funded, processing, completed, failed, refunding, refunded, expired, else unknown). Poll every 5 s until terminal.

Catalog: remit.paymentrequired.com/mcp (ZapRemit sub-catalog). **Schema & dispatch:** Definition [remit:src/zapremit-tools.js:73](#); dispatch [remit:src/zapremit.js:560](#)

Actions / operations

action	what happens	effect	source
zapremit_order	ZapRemit order status	read	remit:src/zapremit.js:560

Prerequisites

No account, registration or API key (instructions text). ZapRemit must be configured on the server, otherwise 503 provider_unavailable.

How to use it (exact tools/call JSON)

Endpoint: POST https://remit.paymentrequired.com/mcp (JSON-RPC 2.0). Placeholders in <...> are never real values; secrets (session, wallet_key) are never pasted into docs or logs.

zapremit_order

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "zapremit_order",
    "arguments": {
      "orderId": "zap_00000000-0000-4000-8000-000000000000"
    }
  }
}
```

Expected result (`structuredContent`, shape from code; values illustrative):

```
{"provider":"zapremit","orderId":"zap_...", "status":"processing", "incomingPaymentVerified":true, "delivery":"in_progress"}
```

Plain-JSON form (same handler): `POST /mcp` with `{"tool":"zapremit_order", ...arguments}`; a payment request then comes back as HTTP 402 ([remit:src/mcp.js:799](#)).

Approval, signature and money

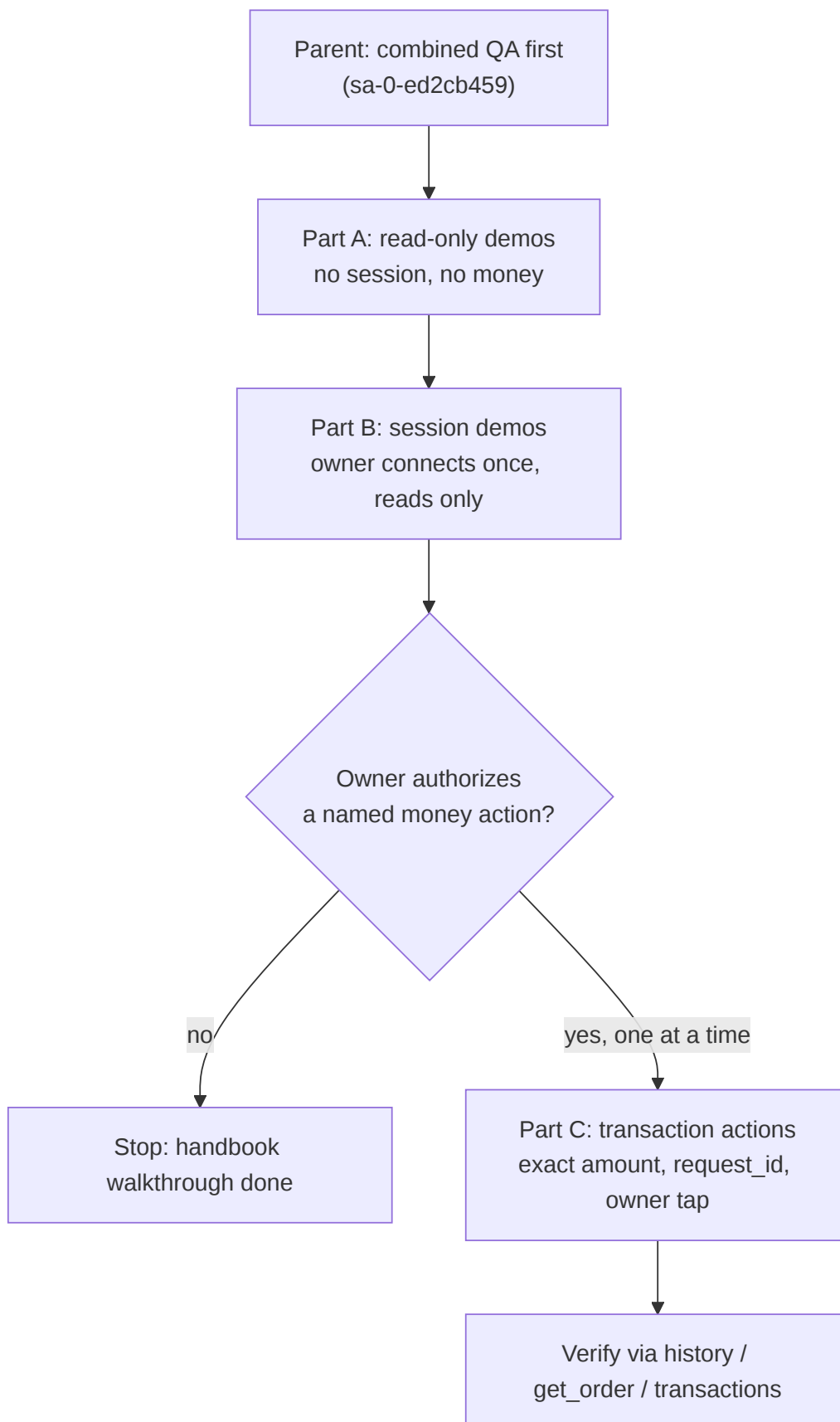
No approval or signature; no money moves.

Failure, peer-down and retry

Provider error → last journal view plus `providerError`; poll again.

What was actually observed

UNVERIFIED: not called by this handbook.



Owner walkthrough checklist (parent leads, owner watches)

[← Index](#) · [Overview](#)

Endpoints after cutover (accepted source, **not deployed**; LIVE-RO 2026-10-14 wallet./banking. = 503): Wallet <https://wallet.paymentrequired.com/mcp>, Banking <https://banking.paymentrequired.com/mcp>, Remit <https://remit.paymentrequired.com/mcp>. <https://paymentrequired.com/mcp> is retired (410) in this topology.

Order: the parent runs its own verification first, then walks the owner through the steps below. **Part A and B never move money.** Part C items each need a separate, explicit owner authorization naming tool, amount and destination. Forbidden throughout: wirex.nuri.com, banking.nuri.com, mcp__nuri_banking__*, the deprecated wirex MCP repo, sandbox (chain 8453 only).

Part A — read-only, no session (safe to run anytime)

#	Tool call (tools/call params)	Endpoint	Expect	Observed by handbook
A 1	tools/list	wallet. + banking./mcp	wallet; account, bank	FIXTURE ✓ (probe-schema 3/3); live 503 until deploy
A 1b	any method	paymentrequired.com/mcp (retired)	410 mcp_moved, no tools	FIXTURE ✓ (probe-directory); live still ed64a10 pre-cutover
A 2	tools/list	remit.paymentrequired.com/mcp	22 tools	LIVE-RO ✓ @212886c
A 3	{"name": "wallet", "arguments": {"action": "discover", "query": "weather", "limit": 3}}	wallet.	services[], sources	UNVERIFIED on wallet. (503)
A 4	{"name": "get_rates", "arguments": {}}	remit	tzsPerUsd/Eur, usdPerBtc	LIVE-RO ✓ @212886c
A 5	{"name": "zapremmit_countries", "arguments": {}, zapremmit_rates	remit	data[], BTC{}	LIVE-RO ✓
A 6	{"name": "quote_tzs", "arguments": {"amountTzs": 5000}}	remit	pay[], validForSeconds 1800	LIVE-RO ✓ @212886c
A 7	list_banks, list_lipa_providers, get_balance	remit	lists / float	LIVE-RO @212886c: ClickPesa generate-token 429/429/403 → re-test
A 8	{"name": "get_order", "arguments": {"orderId": "NURI-00000000-0000-4000-8000-000000000000"}}	remit	error order_not_found	LIVE-RO ✓
A 9	check_recipient / check_bank / check_lipa with the owner's own test destination	remit	recipient name, fee; nothing moves	UNVERIFIED

Part B — owner session, still read-only

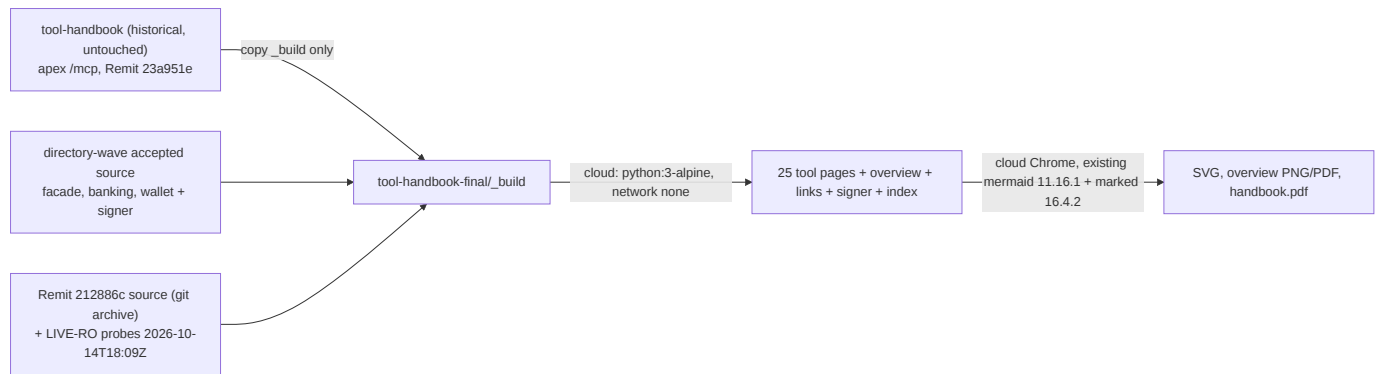
#	Call	Expect	Note
B 1	wallet balance (no args)	askFirst: "new or existing?"	no wallet created
B 2	any account status without session	connect_required + approval_url	owner taps once
B 3	account status with session	verified, card, account	UNVERIFIED
B 4	account limits (op view)	limits{wallet, bank}	if wallet peer down → transport sentence, not crash

#	Call	Expect	Note
B 5	bank iban, bank card, bank transactions	IBAN/deposit addresses; masked card; feed	card details link only shows numbers after the owner signs
B 6	bank manage op 3ds_status, op payout-cards	lists	read only
B 7	bank manage op order-physical with only country	card_types fees + delivery list	no order without address+tap
B 8	wallet history with session	history[]	

Part C — transaction actions (each separately authorized; never run as a demo)

#	Call	Money / effect	Gate
C1	account limits op set	permission change	owner tap
C2	wallet fund amount	bank → agent Safe	bank approval (taps)
C3	wallet pay url max_amount request_id	USDC to merchant	limits / lift tap
C4	wallet send to amount request_id	USDC on Base	owner tap (owned wallet)
C5	wallet bridge chain amount	cross-chain self	none beyond key; ~20 min
C6	bank send iban.../card_id/to 0x	SEPA / card payout / on-chain	tap unless within limits; two taps for to
C7	bank swap to amount≥10	FX	withdrawal taps
C8	bank manage freeze/unfreeze/set-limit/3ds_respond	card state	runs on first call
C9	bank manage close / add-payout-card / issue-card / order-physical	irreversible / card fee	owner tap
C1 0	bank manage activate card_id card_number_last4 expiry_date	card activation (#407; not live yet)	separately authorized; old apex ed64a10 lacks the fields; banking pin has them
C1 1	send_tzs + get_payment + pay once	crypto in → TZS payout	payer's own wallet; bank payouts need operator approve
C1 2	request_tzs (not preview) / route_pay / zapremmit_quote→zapremmit_payment	collection / Lightning	payer PIN / payer wallet

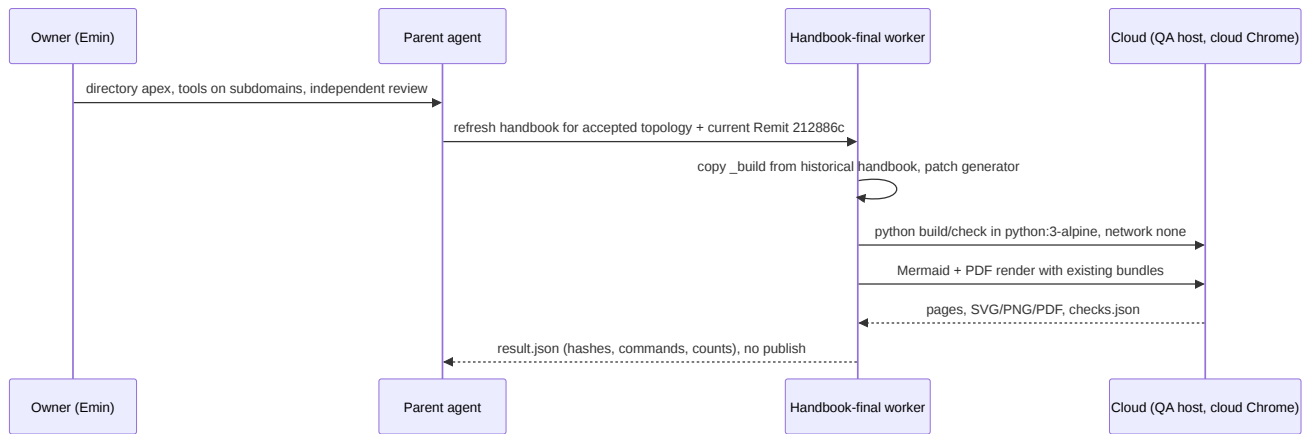
After each C item: confirm with `wallet history`, `bank transactions`, `get_order` or `zapremmit_order`, using the **same** `request_id/orderId`. On `unknown/pending`, repeat the same call; never start a second one.



CHANGELOG — tool-handbook-final

2026-10-14 — directory-wave refresh (unpublished, not deployed)

- Who: Hermes subagent (Claude Opus 5.5), for Emin, delivery issue <https://github.com/nuri-com/agent-payments-mcp-wirex/issues/404>. Input: brief `repository-split/brief.md:11, 24`, scope `directory-wave/approved-scope.md:13-18`.
- Input revisions: facade/banking/wallet/signer = accepted tree pins `71b5e183/d8ab5c82/4d443ed7/1247a406` (parent-directory-wave-acceptance.json); local trees equal the cloud-staged accepted tars except each Dockerfile (builder-owned, not cited). Remit = `nuri-com/nuri-clickpesa-remit-mcp@212886c7eedfa314f7de790622ea95632630057b v2.4.2` = live /health revision at 2026-10-14T18:09Z.
- Fixed generator (not cosmetic): old `build.py` hard-coded `https://paymentrequired.com/mcp` as the endpoint for wallet/account/bank and drew an apex facade forwarding tool calls. Now each page's endpoint comes from its service (HOST map): wallet → `wallet.paymentrequired.com/mcp`, account/bank → `banking.paymentrequired.com/mcp`, Remit → `remit.paymentrequired.com/mcp`. Apex = directory, /mcp 410.
- New `links.md`: already issued apex links (`/s`, `/bridge/privy`, `/api/browser-sessions/...`) keep their original addresses, cited to the facade code; not rewritten to subdomains.
- Tool count is mechanical (`catalog_reconcile()`): directory pin names = code tool names; Remit live `tools/list` = code catalog at 212886c; page order = catalog order; no missing, extra or duplicate page; every action/op enum covered on its page.
- Evidence labels now separate CODE, FIXTURE (accepted 99/99, not live), LIVE-RO and UNVERIFIED. Old LIVE-RO claims for wallet/account/bank (made against the pre-cutover apex) are replaced by FIXTURE claims or marked live 503.
- Remit refresh: citations and permalinks point at 212886c; read-only live calls re-run (`get_rates`, `quote_tzs`, `zapremit_countries/rates/methods` answered; `get_balance` 403, `list_banks` 429, `list_lipa_providers` 429 upstream; `get_order/get_payment` unknown id errors as coded).
- Reported, not fixed (sources read-only): signer `src/service-index.mjs` homepage still points agents at the retired apex /mcp.
- Added whole-handbook searchable PDF (`handbook.pdf`).
- Not done: publication, deploy, any money, signature or card activation.



PROMPTLOG — tool-handbook-final

2026-10-14 — directory-wave refresh

- Who asked: Parent agent for Emin (owner approval quoted in `directory-wave/approved-scope.md:14`). Worker: Hermes subagent, Claude Opus 5.5.
- Input date/revision: parent fresh read 2026-10-14T18:06:18Z Remit `/health 2.4.2 212886c7eedfa314f7de790622ea95632630057b`; accepted directory-wave source receipt 99/99 (fixture, not live).
- Prompt summary: refresh the illustrated handbook for the directory topology and canonical Remit; one page per actual public MCP tool plus a Signer HTTP-only page; Mermaid/SVG/PNG/searchable PDF; exact catalog reconciliation; correct the generator, not only the output; keep issued apex link addresses; cloud-only execution; no publish/deploy/money/signatures.
- Constraints honoured: historical `repository-split/tool-handbook` untouched; sources read-only; no installs; Python doc build ran in the cloud (`_build/cloud_build.sh`); render in cloud Chrome with existing mermaid/marked bundles.