A SIMPLE LEDGER FOR USEFUL MONEY

LDGR

A product and trust model for stable value, predictable settlement, and self-custody that people can actually use.

Thesis

The next useful money network does not need a general-purpose virtual machine, speculative complexity, or a custodial account model. It needs clear assets, predictable settlement, protocol-configured fees, familiar onboarding, and a wallet that remains the user's wallet.

Version 0.4 - product vision and system overview - November 2, 2026

This paper describes implemented capabilities as of November 2, 2026, with testnet prototypes, configuration-dependent services and future direction identified separately. Implementation does not imply that every service or integration is enabled on every network.

01 / The idea

Make the ledger disappear into the product.

Most people do not need another chain to reason about. They need to send stable value, pay a merchant, manage a subscription, receive a gift, or move money between familiar ecosystems without surrendering their keys. LDGR is designed around those verbs.

LDGR is an independent, asset-aware ledger with one settlement authority per network. It intentionally is not a public blockchain, a general-purpose smart-contract VM, or a multi-validator network. That constraint is a product choice: reduce moving parts, make settlement behavior legible, and spend engineering effort on reliable money movement.

> **THE CORE PROMISE**
>
> A small set of signed, integer-denominated ledger operations can do most of the useful work: transfer value, hold it under explicit terms, exchange it, and make settled activity publicly inspectable through transaction hashes and read APIs - while custody stays with the user.

Principles

- Self-custody first. User keys live in the local wallet vault. The ledger verifies signed requests; it does not hold ordinary user keys.
- Assets are explicit. Every balance, transaction, signature, and query includes an asset. Amounts are decimal strings at the edge and integer atomic units in the ledger.
- Useful simplicity. Named templates and a restricted, testnet contract language express bounded financial workflows without arbitrary executable settlement code.
- Open verification. A public explorer and transaction hashes make settled ledger activity inspectable without turning the wallet into a bank.
- Interoperability over lock-in. Users can bring supported external assets into the ledger, while merchants integrate through wallets, web components and payment-verification APIs.

02 / What works today

The ledger is already a product surface.

LDGR now spans installable and extension wallets, merchant payments, asset exchange, funded contracts and public verification. Public access uses edge compute and is distributed by design, while one explicit ledger authority remains responsible for settlement. The table describes implemented surfaces; external funding and operator services require matching network configuration.

Capability	What it enables	Availability / surface
Self-custody wallets	Send, receive, inspect activity and approve requests locally.	Web wallet, extension, CLI; mainnet + testnet
Merchant payments	Pay and subscribe widgets, LDGR402 challenges and server-side receipt verification.	SDK + Utilities; mainnet + testnet
Ledra Checkout	Fund a browser wallet with Base USDC or TON USDT and pay in USDL.	Pay only; configured inventory + attestation
Asset exchange	Atomic routes of up to three AMM pools, liquidity and a swap widget.	Wallets, Utilities, Explorer
Funded contracts	Subscriptions, gifts, splits, atomic offers, milestone escrows and claimable distributions.	Named templates; wallet-reviewed actions
Privacy features	Encrypted transfer messages and reusable confidential-payment codes.	Web wallet + extension; explicit limits
Ledra Contracts	Author bounded escrow programs and simulate them in Studio.	New deployments: testnet only
Ledra Memory	Encrypted project memory with scoped agent access and prepaid service credit.	Separate testnet prototype
Playground	Claim, tip, payment, gift-drop and meme-mint experiments.	Testnet only

A feature in the codebase is not a blanket availability or performance guarantee. Testnet assets have no mainnet reserve claim. Checkout rails, external bridges, price feeds and checkpoint publication depend on operator setup.

03 / Assets and stable value

One ledger. Four distinct asset models.

Useful money starts with precise asset identity. Every balance, transaction, signature, bridge operation, and query names its asset. Display names and icons make those assets legible to people, while exact codes and integer atomic units keep accounting unambiguous.

Fees are protocol-configured and expressed in integer units. Supported transfers can use a fresh fee-sponsor signature or, for an enabled permissionless token, pay the fee in that transferred token. Sponsorship is bound to the exact request; it is not a standing authorization to spend. Clients quote current terms before confirmation.

> **STABLECOIN IS A PAYMENT CATEGORY, NOT A BLANKET GUARANTEE**
>
> A stablecoin payment selects an asset intended to represent stable value. The wallet and merchant must still identify the exact asset, issuer or bridge origin, amount, recipient, and settled transaction hash.

Asset model	What it is	Boundary
Ledra	The native asset: LEDRA on mainnet and tLEDRA on testnet.	Ledger-native; not USDL and not an EVM wrap.
Ledra Dollar	USDL: a separate dollar-denominated token whose mainnet policy specifies off-ledger reserves and redemption through LDGR Operations.	Testnet USDL is for testing; AMM liquidity does not establish backing.
User tokens	Permissionless fixed or mintable fungible tokens registered on-ledger.	No permissionless bridge minting; issuer policy remains explicit.
EVM wraps	Supported operations-registered assets credited against confirmed vault deposits.	Asset-specific trust and withdrawal policy; distinct from native Ledra.

04 / Merchant rails without merchant custody

Request in the browser. Verify on the server.

LDGR's browser SDK provides connect, native pay, token pay, subscribe and swap components. Utilities builders produce copyable embeds, and a React connection adapter serves application integrations. A shared chooser supports the extension, Web Wallet, encrypted Ledra Connect QR sessions and an origin-local site wallet.

PAYMENT VERIFICATION FLOW

Request	Review + sign	Settle	Verify + fulfill
Site asks for recipient, asset, amount, and reference.	Wallet shows exact terms and signs locally.	Ledger accepts the signed action and returns a transaction hash.	Merchant checks hash, status, recipient, asset, and amount server-side.

> **BROWSER EVENTS ARE NOT FULFILLMENT PROOF**
>
> A widget event improves UX, but only independent server-side verification of the settled transaction authorizes delivery.

Widget	Purpose	Settlement discipline
Connect	Lets a site request an LDGR wallet connection.	Connection is identity context, not permission to spend.
Pay	Requests a direct native Ledra payment.	Every payment receives a wallet confirmation.
Stable pay	Requests a direct payment in a selected ledger token or stablecoin.	Merchant checks recipient, amount, asset and settled state server-side.
Subscribe	Opens a plan-bound, prepaid subscription lock.	Merchant verifies current entitlement before granting access.
Swap	Quotes an asset exchange across a bounded AMM route.	The wallet reviews the route, exact input and minimum output before signing.

Subscriptions are prepaid, not silent auto-debits. A payer locks a known amount for known periods; the payee claims unlocked periods, and the payer can cancel the unclaimed remainder. Access still depends on a current authoritative entitlement check.

LDGR402 adds HTTP payment challenges with exact network, asset, amount, recipient and expiry terms. The merchant verifies the settled proof against its saved challenge and prevents receipt reuse before fulfilling an order.

For USDL Pay requests, configured Ledra Checkout rails can fund a local browser wallet from Base USDC or TON USDT. One sponsored batch pays the merchant net of the disclosed service fee and pays that fee to the treasury. Verification binds the transaction hash, Checkout session and saved gross order terms; a browser success event is insufficient.

05 / Bridges and stable-value funding

External funding keeps its own trust boundaries.

The public EVM wrap bridge accepts supported assets into an operator-controlled vault and credits distinct trusted-issuer ledger assets after the configured confirmation path. Withdrawals require ledger spend authority and the matching external release process. The MetaMask get-started flow can create a ledger identity, guide a supported deposit and offer a configuration-dependent native grant.

TON USDT funding for Ledra Dollar uses a separate inventory route. A memo identifies the recipient; a finality-aware attester authorizes a transfer from pre-funded USDL inventory. It does not mint USDL. The inbound route requires configured inventory and attestation; outbound TON USDT withdrawals remain disabled.

Ledra Checkout also uses pre-funded USDL inventory for its Base USDC and TON USDT rails. Base instructions require an exact amount; TON instructions require the exact comment. Late or unmatched funding needs reconciliation. External funding and final ledger payment are separate steps, not one cross-network atomic transaction.

PUBLIC WRAP LIFECYCLE

Deposit	Confirm	Credit	Authorize	Release
User locks a supported asset in the matching EVM vault.	The deposit waits for its configured confirmation threshold.	A verified attestation credits the distinct wrapped ledger asset.	A withdrawal starts with ledger spend authority.	The matching vault process releases the external asset.

> **INTEROPERABILITY, NOT CONFUSION**
>
> Wrapped assets, native assets, and issuer tokens remain separate ledger assets. The UI should lead with human names and clear origin, while the ledger preserves the exact asset policy behind each balance.

What is intentionally separate

- The public EVM bridge is for supported wrapped assets. It is not a claim that every token is freely bridgeable.
- The authenticated partner-native bridge is separate from the public wrap bridge and fails closed until its partner registry is configured.
- The EVM path is a bridge into the ledger, not an attempt to recreate an EVM inside it.

06 / A wallet that remains a wallet

Local keys, explicit review, multiple paths to recovery.

Ledra Wallet is an installable mobile and desktop web wallet at wlt.ldgr.ltd, alongside the Chrome MV3 Ledra Wallet (LDGR) extension. Both keep ordinary user signing local. The web wallet supports passkey-first or recovery-key onboarding, network switching, token balances, send, receive, activity and atomic AMM route swaps.

The web wallet can reopen a passkey account in read-only mode and request authentication when signing is needed. Its private signer stays in page memory until lock or timeout. Recovery-key backups remain user-controlled; optional passphrase protection encrypts local recovery vaults.

Ledra Connect carries encrypted cross-device requests to the Web Wallet for review. Sites can also create a separate origin-local browser wallet with an explicit recovery backup. That wallet belongs to the current website storage, which can be cleared; users should keep only small amounts there. A terminal CLI provides another self-custody path.

Design choice	Why it matters
Local signing	The ledger can verify intent and settle state without keeping normal user private keys.
Bech32 identities	Addresses are network-specific and readable enough to make mainnet/testnet separation explicit.
Per-action confirmations	A connection is not spending authority. Wallet review binds the account, network, exact assets and terms before local signing.
Display-name assets	Balances and confirmations prioritize a token's name and icon, with technical codes as secondary detail.
Public transaction hashes	Users and merchants can link settled activity to a durable public identifier instead of an internal sequence.
Uncertain submissions	Supported payment and contract flows retain the exact signed request for retry when settlement is unknown; a fresh signature must not silently create a duplicate payment.

Self-custody does not mean user-hostile. The product challenge is to preserve possession of keys while providing recognizable names, recovery, clear balances, bridge onboarding, and a dependable activity trail.

07 / Privacy with visible limits

Private content is not a private ledger.

LDGR now supports two distinct privacy features: encrypted transfer messages and confidential payments. Neither hides the ledger from its settlement authority, and neither turns public transaction history into anonymous cash.

Encrypted transfer messages are composed and encrypted locally for the sender and recipient using their verified wallet keys. First-party wallets support messages up to 256 characters, including a zero-value native transfer used only for a message. The encrypted envelope is signed, committed and publicly retrievable; plaintext and decryption keys stay with the endpoints. This is payment-linked messaging, not a general chat service.

Feature	What it protects	What remains visible
Encrypted message	Message content is encrypted for its endpoints.	Public ciphertext, transaction parties, asset, amount and timing.
Confidential payment	A reusable receive code derives a fresh one-time address for each payment.	Sender, one-time destination, asset, amount and timing; the authority can observe transfers.
Local discovery	The wallet scans a shared public feed with its private scan key and can rebuild from recovery material.	Public feed records; later address reuse and consolidation can create visible links.

Confidential payments are implemented on mainnet and testnet in the extension and web wallet. Their reusable code stays out of the signed ledger record. Wallets discover holdings locally and keep them separate from the balance available to the ordinary Send flow.

Current first-party spending is limited to moving one native confidential-address balance at a time into the same recovery wallet's primary address. That move publicly links the addresses. Token-output spending, arbitrary destinations and automatic multi-address consolidation remain unavailable; encrypted messages are not supported for confidential outputs.

The word confidential refers to reducing public linkage to a reusable receiver code. It does not promise hidden amounts, sender anonymity, protection from endpoint compromise or compatibility with another network's payment-code scheme.

08 / Funded contracts and Studio

Express the agreement. Fund its limits.

Named contracts cover escrow, hashlocks, multisignature approvals, vesting, subscriptions, splits and gifts. Newer templates add fixed-counterparty exchanges, staged payments and recipient claims. Their rules live in the ledger; wallets review authoritative terms before signing.

Contract	Implemented behavior	Boundary
Atomic offer	A maker escrows one fungible asset for a fixed amount of another from one designated taker.	No public orderbook, partial fills or cross-ledger exchange. Web-wallet Offers is beta.
Milestone escrow	A payer pre-funds ordered payments to a fixed beneficiary and approves releases.	The ledger does not arbitrate work; the beneficiary cannot force approval.
Claimable distribution	A funder fixes recipient allocations; each recipient claims before expiry.	Only the funder recovers the unclaimed remainder after expiry; no early redirection.
Ledra Contracts	Immutable programs combine ordered steps, alternative payout branches, signer quorums, time gates and hashlocks.	New deployments are testnet only; all possible branch paths are fully funded.

Ledra Studio offers a JSON editor, examples, validation, local simulation and export at /utilities/studio/. Wallets independently review the complete program, participants, payouts and prepaid fee. Deployments use one exact asset and funded escrow; approvals advance a bounded program, while the funder can recover residual escrow after expiry.

The restricted language has no loops, recursion, external reads, callbacks, cross-contract calls, dynamic recipients, minting or wallet allowances. A time gate still needs a signed invocation; there is no automatic program runner. Programs and terms are public, so authors must not put secrets into their source.

This extends the original named-template model without adding a general-purpose settlement VM. A separate gated helper preview can suggest input, but cannot authorize a payment or supply trusted settlement data.

09 / Markets and bounded automation

Automate funded instructions, not wallet access.

LDGR's constant-product AMM supports permissionless pools, liquidity provision and atomic swap routes through up to three pools. The signed request fixes its input and minimum output; the whole route settles or fails together. A bounded quote search is not a guarantee of the best price across every possible market.

Liquidity shares remain internal accounting entries, not transferable tokens. New pools separate accrued LP fees from trading reserves, so LPs can claim fees without withdrawing their liquidity. Older pools require an explicit owner migration to that accounting. Pool ownership, fees and repricing remain visible control and economic risks.

> **AUTOMATION HAS ALREADY STARTED**
>
> Bounded escrow execution is implemented. Broad autonomous wallet agents remain future work. An alarm, price feed or external assistant cannot invent new spending authority.

Mechanism	What can happen	Funding boundary
Scheduled split	A finite schedule releases fixed amounts among fixed recipient weights.	Pre-funded asset escrow and a separate native fee reserve; no later wallet debit.
AMM repricing	A pool owner sets a bounded target, trade caps and execution window; eligible swaps can trigger an atomic reprice.	Only the schedule's escrow and fee reserve can be spent.
Optional renewal	An opted-in AMM schedule can renew its time/run window while funded.	Renewal does not replenish escrow or grant unlimited wallet authority.
USD target mode	A mainnet operator-fed direct price snapshot can guide repricing.	Missing or stale data pauses execution; this is not a decentralized oracle or a guaranteed peg.

Visible schedules can be arbitraged or sandwiched. Limits bound authorized spending; they do not guarantee profitable trades or protect an economic target.

10 / System design and honest scale

Distributed by design; one authority for settlement.

LDGR uses edge compute for public access and distributes read paths, live delivery, and product surfaces so they can scale independently. Settlement remains anchored by one named ledger authority per network. That clear boundary keeps balance and nonce disputes resolvable without pretending the system uses decentralized consensus.

Settlement and explorer freshness are separate measures: the ledger can settle durably before the public projection catches up. Scaling work means keeping that settlement path compact while expanding public reads, event delivery, and integration capacity around it.

Optional operator-signed checkpoints bind a ledger sequence to a hash-chain endpoint. Offline journal verification and local witness retention can check consistency against explicitly trusted operator keys. Full journal verification requires an operator-authorized raw export; public Explorer responses omit some hash-covered metadata. Checkpoints do not create blocks, validators or decentralized consensus, and publication requires operator configuration.

ONE SETTLEMENT CORE, DISTRIBUTED PUBLIC PATHS

Local wallet	Edge access	Ledger authority	Public read model	Live delivery
Ordinary user keys stay under user control.	APIs and product surfaces receive signed requests.	One network authority owns signed settlement and the hash chain.	Rebuildable history and analytics can lag without changing balances.	Filtered updates fan out only after durable work.

> **THE INVARIANT**
>
> Public read delay may exist. It must never create an alternate balance. The read model is rebuildable; the ledger authority remains the durable source of truth.

Explorer confirmation telemetry measures client submission through successful durable commit, separately from indexing. It excludes the response return trip and depends on client-clock quality and retained observations; it is not an end-to-end latency promise. Recovery and projection-rebuild tooling preserve the same authority boundary.

11 / From primitives to a consumer app

Turn ledger primitives into useful habits.

The consumer opportunity is a money app that feels useful before it feels financial. A person should be able to receive a gift, pay a friend, subscribe to a creator, buy a small digital item, and bring existing stable value into the same experience.

The installable wallet now brings everyday balances, transfers and swaps to mobile and desktop. Renewable usernames, public objects, collectibles, widgets and testnet Playground experiments support broader applications. These are usable components; a unified commerce, social and messaging application remains a product direction.

A strong consumer app should be opinionated about the moments that matter

- Onboarding: offer passkeys, a recoverable local wallet or a configured external funding path, with clear storage and recovery expectations.
- Identity: use human names and purposeful profiles without making public address data feel like a social graph by default.
- Money moments: make payment, gifting, subscriptions, and shared purchasing feel native to the application.
- Trust: show clear receipts, transaction hashes, token provenance, and explicit consent when money moves.
- Developer distribution: let merchants and creators bring the payment rail to their own sites with embeddable components.

> The product wedge is concrete: payments, gifts, subscriptions, identity, and receipts that remain understandable when money moves.

12 / Ledra Memory

A separate service for encrypted project memory.

Ledra Memory is a testnet prototype built on LDGR, not an addition to monetary consensus. Its client and local MCP server store encrypted project facts and recall them by similarity. The owner can separately publish a ciphertext commitment through existing public ledger objects.

Content and private metadata are encrypted locally with a separate encryption key. The service stores encrypted envelopes and a bounded, rebuildable search index. An explicitly selected embedding provider processes text and queries: a hosted provider receives that plaintext, while a local provider can keep embedding input on the device.

SCOPED SERVICE ACCESS

Fund

Owner buys prepaid service credit with a verified testnet Ledra transfer.

Grant

Owner selects a space, permissions, expiry and spending cap for an agent.

Use

Agent signs bounded service requests; the client encrypts and decrypts content.

Revoke

Owner can stop future service access without handing over the ledger wallet.

> **A SERVICE GRANT IS NOT WALLET DELEGATION**
>
> Agent grants spend prepaid Memory service credit, not funds from the owner's ledger wallet. Credit is a service entitlement, not a transferable token or redeemable balance; automatic wallet refill and refunds are not implemented.

- The service can see vectors, identifiers, timestamps and sizes. Encrypted storage does not make semantic search opaque to the operator.
- Revocation stops future access but cannot retract content or encryption keys already copied.
- Saving facts requires an explicit request or scoped opt-in; recalled content is untrusted data, not permission to act.
- Mainnet is unsupported. Optional public anchors commit to ciphertext, not the private plaintext.

Memory illustrates a paid application built on ordinary ledger transfers. Its application grants, credit and storage remain separate from ledger balances and settlement authority.

13 / Future direction

Build outward without compromising the custody boundary.

The following directions remain future work. The web wallet, encrypted transfer messages, funded schedules and testnet Memory already provide narrower capabilities; they do not establish the broader products below.

Direction	Opportunity	Non-negotiable boundary
General messaging	Conversations and richer private payment context beyond short transfer messages.	Current transfer ciphertext is public. A future general messenger needs a separate transport, retention policy and end-to-end encryption model.
Agentic wallets	Users and businesses may want agents that prepare actions, watch conditions, coordinate recurring work, or propose transactions.	Existing escrow schedules and Memory service grants do not authorize arbitrary wallet spending. Broader delegation needs inspectable limits, revocation and a reviewed security model.
Scaling	More users, merchants, realtime readers and public integrations need larger read and event capacity.	Do not dilute the single authoritative settlement invariant merely to chase throughput headlines.
Consumer application	A cohesive experience can build on the shipped web wallet to combine commerce, identity, gifts and richer communication.	Do not trade self-custody for convenience or disguise speculative complexity as product progress.

A privacy feature must state what is public, who can decrypt, what is retained and what revocation can actually stop. Payment-linked ciphertext and live ledger events are not a general chat protocol.

14 / A responsible path for wallet agents

Agents should extend agency, not absorb it.

The useful version of an agentic wallet is not an autonomous black box with a permanent key. It is a bounded assistant that helps a person or organization understand, prepare and carry out a narrowly authorized intention.

Today, prepaid schedules can execute narrow ledger instructions, and Memory grants can authorize capped service use. Neither gives an assistant general access to the owner's wallet. Payment reminders, reconciliation, transaction drafting and policy checks are useful directions; broader autonomous wallet execution still needs a separately reviewed delegation model.

> **A POSSIBLE DELEGATION MODEL**
>
> Scope authority by asset, amount, recipient class, time window and action type. Make limits visible in the wallet. Require an independent revocation path. Bind every delegated action to auditable policy and preserve a transaction hash for the result.

What must not happen

- A convenience feature must not quietly turn a self-custody wallet into a hosted account.
- A standing authorization should not become a broad, opaque approval that a user cannot inspect or cancel.
- An agent should not be allowed to create new spending power by interpreting vague natural-language intent.
- Automation must remain more constrained than a user's own signing authority, not less.

15 / Trust model

Clarity about trust is part of the product.

LDGR makes trade-offs that should be legible to users, partners and developers. The named ledger authority is trusted to settle ledger state correctly. Operator-controlled EVM vault and release infrastructure is trusted for registered wrap assets. Issuer inventory bridges additionally depend on the issuer's external system, attester, and pre-funded inventory; they do not make the issuer asset trustless.

The operating model must keep authority ownership, bridge responsibilities, recovery material, upgrades, incident handling, and public communication explicit. The explorer can be rebuilt for visibility, but it is not a second settlement authority or a substitute for operational accountability.

Claim	What it means in practice
Self-custody	Ordinary user private keys stay in the local wallet vault; the ledger service is not a general key custodian.
Public verification	Explorer and read APIs expose transactions and ledger state, while the public projection remains rebuildable rather than becoming settlement authority.
Bridge accountability	Bridge operations use durable identities and idempotent handling, but external vaults, partners, issuers, and attesters remain explicit trust boundaries.
Change + incidents	Protocol or operational changes require controlled rollout, testing, review, and transparent incident handling.
No general-purpose VM	Named templates and testnet bounded programs settle explicit funded terms. They cannot call arbitrary code or spend an unfunded user wallet.
Explicit privacy limits	Ledger records and encrypted transfer envelopes are public. Confidential codes reduce receiver linkage; Memory content is encrypted off-ledger, with vectors visible to its service.
Checkable operator statements	Configured signed checkpoints and journal verification can expose inconsistency against trusted keys; they do not remove trust in the settlement authority.

This is a product vision, not a security guarantee. Production use demands continued testing, operational controls, independent review, transparent incident handling, and conservative expansion of bridge and automation capabilities.

16 / The path

Make useful money boring in the best way.

LDGR begins with a simple proposition: stable value and low-cost ledger operations should be easy to understand, easy to integrate, and hard to misrepresent. The product can grow from direct transfers into a consumer experience with commerce, subscriptions, identity, gifts and carefully designed communication - without abandoning the fundamental promise that the user owns the wallet.

> The work ahead is not to accumulate every feature associated with crypto. It is to make a small number of financial actions exceptionally clear, reliable, and available wherever people already transact.

IMPLEMENTED CAPABILITIES

Web and extension wallets; local signing; Ledra Connect; token payments and LDGR402; configured USDL Checkout and bridge funding; atomic AMM routes and swap widgets; funded contracts and schedules; encrypted transfer messages; confidential-payment discovery and limited native consolidation; public receipts and operational verification tooling.

TESTNET AND EXPERIMENTAL

New Ledra Contracts deployments and Studio funding are testnet only. Ledra Memory is a separate testnet prototype, and Playground remains testnet-only. Web-wallet atomic offers are beta. External funding and operator services require their documented configuration and readiness checks.

NEXT HORIZON

Broader consumer commerce; a general encrypted messenger; carefully reviewed wallet delegation; and more capacity for public reads, events and integrations while preserving one authoritative settlement core.

Appendix / Terms and public references

Read the same words the same way.

This paper is a product vision and system overview, not the canonical protocol specification. Current behavior belongs in the public documentation and protocol reference; transaction-level facts belong in the Explorer.

> **PUBLIC REFERENCE**
>
> Docs: ldgr.ltd/docs | Wallet: wlt.ldgr.ltd | Explorer: ldgr.ltd/explorer | EVM wraps: ldgr.ltd/bridge | Studio: ldgr.ltd/utilities/studio/ | USDL: ldgr.ltd/usdl | Whitepaper: ldgr.ltd/whitepaper.pdf

Terminology

Term	Meaning
Ledger	The authoritative record of balances, signed operations, transactions and asset rules.
Self-custody	The user controls the private key material needed to authorize ordinary spending.
Stablecoin payment	A direct payment in a selected ledger token or supported wrapped asset intended to represent stable value.
Funded contract	A named template or restricted immutable program with explicit conditions, fixed authority and escrow-backed payouts.
Trusted-issuer wrap	A ledger asset credited against a supported EVM vault deposit and released through its matching withdrawal process.
Confidential payment	A payment to a locally derived one-time destination; amounts and transaction metadata remain public.
Checkpoint	An optional signed operator statement about a ledger endpoint, not a consensus certificate.
Service credit	Prepaid application entitlement, separate from the owner's authoritative ledger balance.

For current limits, use the public guides for wallets, payments, confidential payments, Ledra Contracts and Ledra Memory. Reserve and redemption policy belongs to the USDL documentation. This edition reflects repository capabilities as of November 2, 2026; it is not a live deployment audit.

Use transaction hashes as public receipts. Treat historical implementation plans as decision context rather than current product guarantees.