vltUSDC — Passive LP Yield with Built-in VLT Upside

Deposit USDC and receive vltUSDC — a single token that lives in your wallet, earns auto-compounding trading-fee yield, and stays redeemable any time. Dampened exposure to VLT, with nothing to manage and nothing to trust.

How It Works

When you deposit USDC, the vault mints you **vltUSDC — a standard ERC-20 token sent straight to your wallet**. Your entire position is simply that token balance: you hold it yourself, and it's transferable and composable anywhere in DeFi. Underneath, your USDC sits in a VLT/USDC liquidity position earning a 1% fee on every trade, and those fees auto-reinvest so each token grows in value. Redeem any time to receive both underlying tokens back in kind — half of your value is available as USDC the moment you need it, with zero slippage.

Why It's Different

- **Real yield, not emissions.** Returns come from actual trading volume, compounded — no inflationary rewards, no reflexive peg mechanics.
- **Lower-beta exposure.** Roughly half your capital sits in USDC, so you ride VLT's growth with materially less volatility than holding it outright.
- **Truly passive.** Auto-compounding, no rebalancing, no keeper at all — deposits themselves trigger the compound, and 100% of every harvest reinvests for holders (no fee of any kind).
- **Self-custodied and verifiable.** vltUSDC sits in your own wallet — never ours. Non-upgradeable, no oracle, hardcoded fee, and no admin keys at all — the vault is fully ownerless.
- **Self-reinforcing liquidity.** Protocol buy-flow and ecosystem fee routing deepen the pool, which captures more fees — compounding the compounding.
- **Variable yield and asset growth.** Both the fee yield and the token's value scale with market demand and trading activity — the more the pool is used, the more it compounds.
- **Asset-backed price.** vltUSDC's value is set by the real VLT/USDC pool it holds — not a promised peg. No algo tricks, no hidden leverage.

The line: vltUSDC turns VLT's trading activity into a passive, compounding return stream with half-anchored exposure and zero active management — a yield-and-growth vault you can fully verify and never have to babysit.

Learn more and participate at bankroll.network.

Use Case Overview — A Pillar of Bankroll Network

Bankroll Network is an on-chain financial cooperative built on one reserve asset, **VLT**. Its instruments stack into three reinforcing pillars — and vltUSDC is the income-and-growth layer that puts idle stablecoins to work.

- **Pillar 01 — VLT as store of value.** ETH concentrate and protocol-owned liquidity. The cooperative's reserve asset, backed by permanently locked liquidity.
- **Pillar 02 — VLT as collateral.** Borrow against VLT for on-chain credit — access liquidity without selling your position.
- **Pillar 03 — vltUSDC (this instrument).** Passive income and growth. Idle USDC becomes the cooperative's productive, deepest liquidity layer.

Why It Outperforms a Standard LP

- **3.3× fees.** The 1% fee tier is the right rate for a volatile asset, and it earns 0.7 points more per trade than the 0.3% standard — 3.3× the fee rate. This is the single biggest reason returns here beat a generic V2-style LP.
- **One venue.** Every deposit aggregates into a single canonical VLT/USDC pool — the deepest market for VLT. That low-slippage depth is what lets the 1% fee capture trade flow rather than repel it.
- **POL flywheel.** Protocol-owned liquidity: ecosystem fee routing and buy-flow continuously deepen the pool, compounding volume and fees — depth a scatter of individual LPs can't match.

The role: vltUSDC pays holders while making VLT itself more tradeable for everyone — the cooperative's liquidity layer, not a private LP position.

Technical Deep Dive — vltUSDC Under the Hood

Architecture

vltUSDC is an ERC-20 share over a single full-range Uniswap V4 VLT/USDC position. Shares are denominated in the pool's **liquidity units (L)**, never in dollars — so no price oracle exists anywhere in the system. The token reports `decimals() == 0` for exactly that reason: balances read as the real L value rather than an invented 18-decimal fraction. Your balance is a pro-rata claim on the position's liquidity, and redemption returns the underlying tokens directly.

Every claim below is checkable from the contract's own views, without trusting this document: `positionLiquidity()` and `totalSupply()` (liquidity per share), `previewRedeem(shares)` (what a balance is worth in kind, right now), `compoundClaimable()` (what the next compound would reinvest), `feeApr()` (lifetime/7d/30d growth in bps), and `totalFeesVlt()` / `totalFeesUsdc()` (lifetime realized fees).

Core Flows

- `deposit(vlt, usdc, minShares, deadline, recipient)` — Pulls both tokens, adds liquidity (no swap inside the vault), refunds any imbalanced excess to the payer, and mints shares to `recipient` pro-rata to the liquidity actually added (ΔL) — measured from the pool, not contract balances, which neutralizes donation and first-deposit inflation attacks. USDC-only entry goes through the periphery `ZapHelper.zapDeposit(...)`, which buys VLT on the open market (buy pressure) and deposits the pair; shares still mint straight to the end wallet.
- `redeem(shares, receiver)` — Burns the caller's shares, removes the pro-rata slice of liquidity, and returns both tokens in kind to `receiver` — no swap, no oracle, no forced sale, no slippage inputs needed (in-kind exit can't be sandwiched for value). Cannot be rendered insolvent: you only ever withdraw what the position already holds. A USDC-only exit is available through the same periphery (`ZapHelper.zapRedeem` / `zapRedeemWithPermit`, the latter consuming an EIP-2612 share permit so it costs one signature and one transaction).
- `donate(vlt, usdc, donor, deadline)` — A swap-free, no-mint liquidity add at the pool's own price: the caller gifts both tokens, the vault folds them into the position, **mints no shares**, and refunds the unusable leg to the payer. L rises against unchanged supply, so every holder's redemption value grows at that block. This is the sanctioned path for ecosystem fee routing (see the constraint below); the `Donate` event credits `donor`, so a periphery zapper can pay on someone's behalf.
- **Auto-compound (inside `deposit`)** — No keeper and no compound fee. Once ~$100 of *claimable value* has accumulated — the position's pending pool fees plus any balances already retained on the vault — the next deposit collects it first: it auto-rebalances within tightly capped bounds, reinvests 100% as liquidity (no fee of any kind is taken), and mints no new shares — so L rises against a fixed supply and every

holder's redemption value grows automatically. There is no compound entrypoint at all: the vault's entire write surface is `deposit`, `donate`, and `redeem`, and anyone may force a compound in a quiet market with a small deposit.

Trust Model

- **Non-upgradeable.** No proxy, no migration path — the code that ships is the code that runs.
- **Fully ownerless.** There is no owner, no admin role, no pause, no sweep — every parameter is a constant or fixed at deploy; deposit/redeem are permissionless and compounding rides on deposits. Nobody, including the deployer, holds any key over the vault.
- **Solvent by construction.** Shares are liquidity claims; in-kind redemption can't exceed what's in the pool.
- **Oracle-free.** Liquidity-denominated shares plus in-kind exit remove every price-manipulation surface.
- **Hardened entrypoints.** Reentrancy guards, SafeERC20, checks-effects-interactions, first-deposit lock, deadlines and slippage bounds on entry, and a compound rebalance bounded to a ≤5% price move (unfilled remainder folds forward).
- **Exit always open.** Nothing in the system is pausable — redemption least of all.

Risk Surface

- **Impermanent loss.** On large VLT moves the position underperforms holding the two tokens separately.
- **Underlying asset risk.** VLT is volatile with concentrated liquidity; price and depth risk apply.
- **Variable yield.** Fees track trading volume; quiet markets compound slower.
- **Deposit-swap exposure.** The USDC to VLT entry swap (in the replaceable ZapHelper periphery, never the vault) executes an off-chain-computed route, bounded by on-chain `minVltOut` and a deadline.
- **Smart-contract risk.** Audited independently by Shieldify before launch (final report: 11 Informational, no Critical/High/Medium/Low), plus two internal reviews and a Slither-clean build. An audit reduces risk; it never removes it.

Fee-routing constraint (binding, from the Shieldify review): ecosystem fee routing pushes value to holders via `vault.donate(vlt, usdc, donor, deadline)` — a swap-free, no-mint liquidity add at the pool's own price — in small recurring tranches. It does NOT use `PoolManager.donate()`, and no routing mechanism may make large one-sided donations to the pool: donation-inflated one-sided fee states are the only condition under which the deposit-triggered compound's rebalance swap was shown sandwichable (Shieldify L-03; ordinary trading-fee states were demonstrated unprofitable to attack). Large one-off donations go through `vault.donate()` via private submission (atomic L/share jumps are front-runnable).

Parameters

Parameter	Value
Chain · AMM	Ethereum mainnet · Uniswap V4
Pool · fee tier · range	VLT/USDC · 1% · full range (tickSpacing 200)
Share token	`Bankroll VLT–USDC LP` · **vltUSDC** · `decimals() == 0` (raw L) · EIP-2612 permit
Share unit	Pro-rata pool liquidity (L) — no fixed cap
Write surface	`deposit`, `donate`, `redeem` — no compound entrypoint, no admin functions
Compound fee	None — 100% of harvested fees reinvest for holders
Auto-compound trigger	$100 of claimable value (pending fees + retained balances), hardcoded
VLT token	0x6b785a0322126826d8226d77e173d75DAfb84d11
Vault contract	0xee8d4c5c768AadCd3517Aa8C908De300305D0A7f (deployed 2026-07-22)
ZapHelper (periphery)	0x348A57b1dc6E3dCAa645DE6e4E864924B410525D
Audit	Shieldify — final report: 11 Informational, 0 Critical/High/Medium/Low
Source	github.com/bankrollnetwork/bankroll-contracts — verified on Etherscan

Technical summary for evaluation only; not a specification or an audit. The deployed contracts are immutable — this document describes them, and the contracts themselves are the source of truth.