Kova Money Whitepaper

A Self-Custodial Spending Infrastructure for the Stablecoin Economy

Version 1.0

June 2026

Author:

Pranav Jha

Founder, Kova Money

Email: pranav@kova.money

`https://kova.money`

Abstract

We propose Kova, a self-custodial crypto spending platform that enables users to make real-world payments directly from their own smart contract wallets without relinquishing custody of their assets. Stablecoin balances are locked in user-owned vaults with customizable spending parameters. All wallets are equipped with a programmable session interface that can be consumed by any card network or payment application, allowing stablecoins to be spent and settled across the real-world economy. A credit-against-collateral engine coordinates all card authorizations, with on-chain liens representing locked collateral and session keys representing the scoped permissions between them. The network authenticates each transaction via passkey-based signatures driven by hardware-bound cryptographic keys. Participation in the protocol contributes to the growth of an open and non-custodial infrastructure for spending stablecoins in the real world.

1 Introduction

Stablecoin spending has come to rely almost exclusively on custodial intermediaries to coordinate card issuance and settlement. The storage and exchange of digital dollars across all payment channels is taxed at the hands of a few centralized entities—ranging from crypto exchanges to neobank platforms—which increases the transaction costs for stablecoin holders. In addition, the traditional custodial system can no longer compete with the reality of a growing stablecoin economy, in which over $230 billion in circulating supply seeks a path to everyday commerce with vanishing friction. Individuals find their assets locked in custodial platforms subject to insolvency risk without recourse, and self-custody users themselves cannot feasibly navigate the multi-step process of converting on-chain assets to real-world spend. There exists neither sovereignty nor usability in the current infrastructure for stablecoin payments.

What is needed is an open and programmable infrastructure for self-custodial stablecoin spending. We propose Kova, a self-custodial crypto spending platform. Any user can lock, spend, and manage their stablecoins directly on Kova without transferring custody to rent-seeking intermediaries. Programmable session keys set by owners are embedded within each card authorization as it is validated and settled across any merchant terminal on the card network, preserving self-custody throughout the transaction lifecycle. Kova provides both the smart wallet infrastructure and the credit-against-collateral engine needed to make non-custodial finance feel as effortless as tapping a

card.

2 Architecture Overview

Crypto card products have evolved in distinct waves. Centralized exchanges [1] introduced the first crypto-funded debit cards. Neobank challengers [2] transformed this concept into dedicated spending platforms, unlocking new user experiences like instant crypto-to-fiat conversion. When faced with significant custodial risk events, platforms such as Gnosis Pay [3] emerged to address these limitations with on-chain settlement. However, while focusing on the Custody Trilemma—the trade-off between self-custody, usability, and card network compliance—the programmability and sovereignty of user wallets was overlooked, creating a barrier to true non-custodial adoption. This demands the rise of a new type of infrastructure: purpose-built smart wallet systems designed to address real-world spending, creating programmable markets for stablecoin commerce.

Kova is a purpose-built self-custodial spending platform powered by ERC-4337 account abstraction. It comprises a smart wallet layer alongside multiple highly specialized modules. The wallet layer provides full ERC-4337 compliance, enabling rapid adoption of existing tooling from the ecosystem. The Vault module, one of the specialized components, efficiently handles credit-against-collateral authorization as a native operation while optimizing settlement across complex card network flows. This module transforms stablecoins into spendable collateral. Although Kova focuses primarily on card spending, its flexible architecture enables the adoption of future modules that can expand far beyond payment-related applications.

Layer	**Component**	**Function**
Wallet Layer	KovaAccount (ERC-4337)	Smart contract wallet with passkey authentication, guardian recovery, and modular plugin architecture
Vault Layer	KovaVault	Credit-against-collateral engine that authorizes card spend against locked user assets
Session Layer	KovaCardSession	Scoped session keys with time, amount, and merchant category constraints
Streaming Layer	KovaDripper	On-chain salary streaming with direct composability into card session funding

Figure 1: Kova's protocol layers

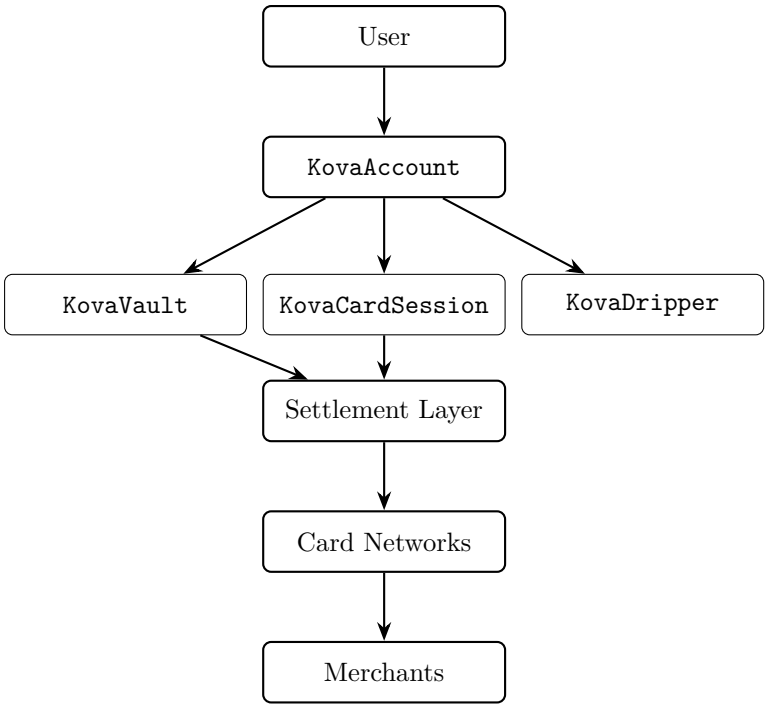

Figure 2: Kova system architecture

Kova runs on EVM-compatible Layer 2 chains with low transaction fees and fast finality. The smart contract modules work together to maintain self-custody invariants, process card authorizations, and protect the integrity of user funds, creating a secure non-custodial infrastructure open to all participants who follow the rules of the protocol. Following the principle of separation of concerns, the protocol is divided into four layers: a wallet layer, a vault layer, a session layer, and a streaming layer.

2.1 The Wallet Layer

The Wallet Layer serves as the cryptographic foundation of the platform, handling the deployment and authentication of all user accounts. It is powered by ERC-4337 account abstraction and passkey-based signing working in harmony. This layer is designed to deliver maximum usability to each user while preserving self-custody guarantees and key security.

ERC-4337 [4] replaces the traditional Ethereum transaction model—where an externally owned account (EOA) signs and pays for transactions—with a programmable account model where the smart contract itself defines its validation logic, fee payment strategy, and execution rules. The key components of the ERC-4337 stack as utilized by Kova:

- **UserOperation:** A pseudo-transaction object that encodes the user's intent (call data, gas limits, signature). UserOperations are submitted to a mempool and bundled by a Bundler into on-chain transactions.
- **EntryPoint:** The singleton contract (v0.7) that validates and executes UserOperations. All Kova wallets interact with the canonical EntryPoint.

- **Bundler:** An off-chain service that aggregates UserOperations and submits them on-chain. Kova utilizes ZeroDev's [5] bundler infrastructure for reliable operation submission and gas estimation.
- **Paymaster:** A contract that sponsors gas fees on behalf of users. KovaPaymaster enables gasless transactions for end users by paying gas in the background, funded by protocol revenue.

Each Kova wallet is deployed via `KovaAccountFactory`, a deterministic factory contract that uses `CREATE2` to derive wallet addresses from the user's passkey public key. This means a user's wallet address is known before deployment—they can receive funds at their Kova address before the wallet contract is ever created on-chain. The wallet is lazily deployed on the first transaction, bundled with the user's first UserOperation.

Kova is designed to provide deterministic wallet addresses across supported chains through a standardized factory deployment and `CREATE2`-based address derivation. A user with the same passkey will derive the same address on Polygon, Base, Arbitrum, or any EVM chain where the factory is deployed with consistent parameters.

2.2 Passkey Authentication

Kova wallets use WebAuthn/FIDO2 [6] passkeys as the primary authentication mechanism, replacing seed phrases entirely. When a user creates a Kova wallet, their device generates a P-256 (secp256r1) key pair stored in the device's secure enclave (Titan M2 on Pixel, Secure Enclave on iPhone, TPM on desktop). The private key never leaves the hardware module and cannot be exported.

Transaction signing works as follows: the Kova app constructs a UserOperation, hashes it, and presents the hash as a WebAuthn challenge. The user authenticates via biometric (fingerprint or face), the secure enclave signs the challenge with the P-256 key, and the signature is attached to the UserOperation. On-chain, the `KovaAccount` contract verifies the P-256 signature using the RIP-7212 precompile (where available) or a Solidity P-256 verifier fallback.

This design eliminates the single largest source of crypto asset loss: compromised or forgotten seed phrases. The user's authentication is bound to their biometric identity and hardware, not to a string of words they must secure independently.

2.3 Guardian Recovery

Device loss is handled through a guardian-based social recovery mechanism. During wallet setup, the user designates one or more guardians—trusted addresses that can collectively authorize a key rotation. Guardians can be friends, family members, or institutional recovery services.

The recovery process is time-locked: when a guardian initiates recovery, a configurable delay period (default: 48 hours) begins. During this window, the original owner can cancel the recovery if it was initiated maliciously. After the delay expires and the required guardian threshold is met (e.g., 2-of-3), the wallet's signing key is rotated to a new passkey controlled by the user on their new device.

Guardians cannot move funds, change spending permissions, or perform any action other than approving a key rotation. This ensures that the recovery mechanism cannot be weaponized against the user.

3 KovaVault: Credit-Against-Collateral

The central innovation of the Kova protocol is KovaVault: a smart contract mechanism that enables real-time card spending against on-chain collateral without transferring custody of that collateral to any third party. Just as Story [7] enabled the tracking of intellectual property through registration and versioning, KovaVault creates a programmable lien system for stablecoin collateral, making Kova the provenance and settlement layer for non-custodial card spend.

3.1 Collateral Model

When a user wants to enable card spending, they lock stablecoins into the KovaVault module attached to their wallet. These tokens remain in the user's smart contract wallet—they are not transferred to a Kova-controlled address. Instead, the vault places an on-chain lien: the locked tokens cannot be transferred or withdrawn while they serve as collateral for outstanding card spend.

The collateral ratio is straightforward for stablecoin-to-stablecoin backing: \$1 of USDC locked provides \$1 of spending capacity. For volatile assets used as collateral (e.g., ETH or WBTC), the vault enforces an overcollateralization ratio, initially set at 150%, with real-time price feeds from Chainlink [8] oracles determining the available spend limit.

3.2 Spend Authorization Flow

The card spend flow operates as a two-phase commit:

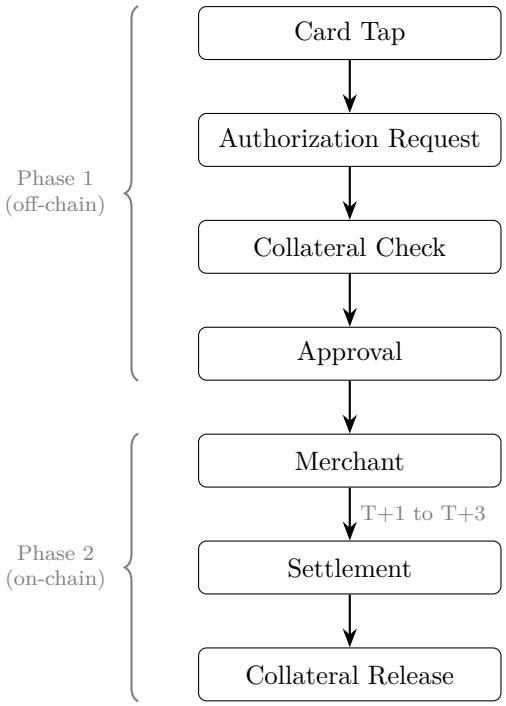

Figure 3: Card spend authorization and settlement flow

Phase 1 — Authorization. When the user taps their card at a terminal, the card network sends an authorization request. Kova's backend verifies that the user's vault has sufficient unlocked collateral, earmarks the amount (moving it from "available" to "pending" within the vault's internal accounting), and approves the authorization. No on-chain transaction occurs at this stage—authorization is handled off-chain for latency requirements (sub-200ms).

Phase 2 — Settlement. At settlement time (typically T+1 to T+3 per card network rules), Kova settles the transaction. The vault releases the earmarked collateral and transfers the exact spend amount to the settlement address. The user's remaining collateral is unlocked and available for future spending or withdrawal.

Between authorization and settlement, the user's collateral is in a "pending" state: visible in their wallet but not available for withdrawal. This mirrors how traditional debit card holds work, but with the critical difference that the funds remain in the user's contract throughout.

3.3 Liquidation and Margin Mechanics

For volatile collateral, the vault monitors the collateral ratio in real-time. If the value of the collateral drops below the maintenance margin (initially 120%), the vault enters a margin call state. The user receives a notification and has a grace period (configurable, default 4 hours) to add collateral or reduce their position. If the collateral ratio drops below the liquidation threshold (initially 110%), the vault automatically converts sufficient collateral to stablecoins via an on-chain DEX aggregator to restore the ratio above the target.

For stablecoin-only collateral (USDC, USDT, DAI), there is no liquidation risk—the collateral and

spend currency are 1:1 by definition. This is the expected default for most Kova users.

4 Session Keys and Programmable Spend

4.1 Card Sessions

`KovaCardSession` is a smart contract module that implements scoped, time-bound spending permissions. Rather than granting blanket access to the wallet's funds, each card session defines a precise envelope of what can be spent, when, and under what conditions.

Parameter	Type	Description
`maxAmount`	`uint256`	Maximum total spend for the session
`maxPerTx`	`uint256`	Maximum spend per individual transaction
`validAfter`	`uint48`	Session start timestamp
`validUntil`	`uint48`	Session expiry timestamp
`allowedMCCs`	`uint16[]`	Permitted merchant category codes
`blockedMCCs`	`uint16[]`	Explicitly blocked merchant categories
`cooldown`	`uint48`	Minimum time between transactions

Figure 4: Card session parameters

4.2 Permission Scoping

Session keys operate on the principle of least privilege. When a user enables their Kova card, the app creates a session key—an ephemeral key pair that is authorized by the wallet to execute spend operations within the defined constraints. The session key is stored on the device and used to sign card authorization requests without requiring biometric authentication for each tap.

This creates a two-tier permission model: the passkey (in secure enclave) is the master key that can do anything—transfer funds, change guardians, revoke sessions. The session key (in app memory) can only authorize card spend within its defined envelope. If the session key is compromised, the attacker's damage is bounded by the session parameters. The master passkey remains secure in hardware.

Level	Key Type	Storage	Capabilities
L0 (Master)	P-256 Passkey	Secure Enclave	Full wallet control: transfers, guardian management, session management
L1 (Recovery)	Guardian Addresses	On-chain (wallet)	Key rotation only. No fund access.
L2 (Session)	Ephemeral ECDSA	App Memory	Card spend within session envelope only

Figure 5: Key hierarchy and capabilities

4.3 Revocation

Sessions can be revoked instantly by the wallet owner via the master passkey. Revocation is an on-chain transaction that invalidates the session key, immediately preventing any further card authorizations. Users can also set automatic revocation rules: revoke if the session has been inactive for N hours, revoke if a transaction is declined, or revoke if spend velocity exceeds a threshold.

5 Dripper: On-Chain Salary Streaming

5.1 Stream Mechanics

`KovaDripper` is a smart contract module that enables continuous, per-second streaming of tokens from one address to another. An employer (or any payer) creates a stream by depositing tokens into the Dripper contract with the following parameters: recipient wallet address, token address, total amount, start time, and end time. Once the stream is active, tokens accrue to the recipient in real-time. The recipient can withdraw accrued tokens at any time without waiting for a pay period to complete.

The stream rate is calculated as:

$$\text{rate} = \frac{\text{totalAmount}}{\text{endTime} - \text{startTime}} \tag{1}$$

At any timestamp t, the claimable amount is:

$$\text{claimable} = \text{rate} \times (t - \text{startTime}) - \text{alreadyClaimed} \tag{2}$$

This is computed on-chain at claim time—no keeper or oracle is required.

5.2 Composability with Card Sessions

The Dripper's key innovation in the Kova context is its direct composability with the card session layer. A user can configure their wallet so that incoming Dripper streams automatically fund their KovaVault collateral, which in turn backs their active card session. The practical result: a freelancer receiving a USDC salary stream can spend their earnings in real-time as they accrue, via their Kova card, without any manual action.

This creates a closed loop: employer streams USDC $\rightarrow$ Dripper accrues to user's wallet $\rightarrow$ auto-collateralizes in KovaVault $\rightarrow$ user spends via card session. Every step is on-chain, every step is non-custodial, and the user never has to think about moving money between accounts.

6 Security Model

6.1 Threat Model

Kova's security model assumes the following adversarial conditions:

- **Compromised device:** An attacker gains access to the user's unlocked phone. Session keys may be exposed, but the passkey in the secure enclave remains protected by biometric gating. Damage is bounded by active session parameters.
- **Compromised backend:** Kova's off-chain infrastructure is breached. The attacker cannot move funds—all fund transfers require on-chain signatures from the user's wallet. The attacker could at most disrupt the authorization relay, resulting in declined card transactions (a denial-of-service, not a loss of funds).
- **Malicious guardian:** A guardian attempts unauthorized recovery. The time-lock delay gives the legitimate owner a window to cancel. A single malicious guardian below the threshold cannot initiate recovery alone.
- **Kova ceases operations:** Because wallets are self-custodial smart contracts deployed on public chains, users retain full access to their funds regardless of Kova's operational status. The user can interact with their wallet contract directly using any ERC-4337 compatible interface. Card spending would cease, but asset custody is unaffected.

6.2 Self-Custody Guarantees

Kova enforces the following invariants at the smart contract level:

- **Non-custodial:** At no point does any Kova-controlled address hold user funds. All assets remain in the user's smart contract wallet. The KovaVault lien is enforced by the user's own contract logic, not by an external custodian.
- **Unilateral exit:** The user can withdraw all unlocked funds at any time without Kova's cooperation. If Kova's off-chain services go offline, the user retains full on-chain access to their wallet and all funds not currently in a pending settlement state.
- **Bounded delegation:** Session keys can only operate within their defined parameter envelope. No session key can exceed its `maxAmount`, transact outside its time window, or bypass merchant category restrictions. These constraints are enforced on-chain and cannot be overridden by Kova's backend.
- **Recoverable:** Loss of the primary device does not result in loss of funds. The guardian recovery mechanism allows secure key rotation without exposing assets during the recovery process.

7 Network and Chain Strategy

Kova deploys on EVM-compatible L2 chains that offer low transaction fees and fast finality. The primary deployment targets are Polygon PoS and Base (Coinbase L2), selected for their stablecoin liquidity, low gas costs (sub-$0.01 per transaction), and established bridging infrastructure.

The smart wallet architecture is chain-agnostic by design. Because wallet addresses are derived deterministically from the user's passkey via `CREATE2`, the same user has the same address on every supported chain. Cross-chain expansion requires only deploying the factory and module contracts to the new chain—no user migration is necessary.

Future chain additions will be evaluated on three criteria: stablecoin TVL (must exceed $100M USDC/USDT), average transaction cost (must remain below $0.05), and availability of reliable price oracle infrastructure (Chainlink or equivalent).

8 Regulatory and Compliance Framework

Kova is designed to operate within existing payment network and financial compliance requirements. While Kova itself provides self-custodial wallet infrastructure and collateral management, certain regulated activities may be performed by licensed partners where required by applicable law.

The protocol is designed to support:

- **Identity verification (KYC)** through regulated onboarding providers.
- **Anti-Money Laundering (AML)** screening and transaction monitoring.
- **Sanctions screening** against applicable regulatory lists.
- **Card network compliance** through licensed issuing and settlement partners.
- **Regional compliance requirements** based on jurisdiction.

Kova does not take custody of user funds. Users retain ownership and control of assets held within their smart contract wallets at all times. Compliance requirements are expected to be fulfilled through integrations with regulated service providers as the ecosystem expands.

Specific implementation details may evolve based on regulatory developments, partner requirements, and jurisdictional considerations.

9 Roadmap

Phase	Milestones
Phase 1 — Foundation Q3–Q4 2026	Core smart wallet deployment (KovaAccount, KovaAccountFactory) on Polygon and Base. Passkey authentication and guardian recovery. Testnet launch with internal card authorization simulation.
Phase 2 — Vault & Card Q1 2027	KovaVault credit-against-collateral engine live on mainnet. Card issuance integration with licensed settlement infrastructure. KovaPaymaster gasless transaction sponsorship. First real-world card transactions.
Phase 3 — Programmability Q2–Q3 2027	KovaCardSession module with full parameter scoping (MCC filters, velocity limits, time bounds). KovaDripper salary streaming with auto-collateralization. Public API for third-party integrations.
Phase 4 — Scale Q4 2027+	Multi-chain expansion to additional L2 networks. Volatile collateral support with oracle-driven margin mechanics. Advanced session composability and programmable spend rules. Ecosystem growth through developer tooling and SDK release.

Table 1: Kova development roadmap

10 Conclusion

We have proposed a self-custodial system for stablecoin spending that does not rely on custodial intermediaries. Kova offers a purpose-built smart wallet infrastructure that creates both a non-custodial collateral engine and a programmable market for real-world card payments. Kova's modular architecture offers the extensibility and specialization needed to accommodate an evolving ecosystem of card network integrations and payment use cases. A wallet-native vault system embedded directly into user-owned smart contracts represents the economic commitments between collateral and card spend, creating a robust record of self-custodial guarantees. As the volume and velocity of stablecoin transactions scales from the acceleration of the digital economy, Kova's infrastructure will serve as the backbone of a new non-custodial spending economy. Whereas existing platforms operate as custodians of user assets in the form of centralized floats, Kova operates as a sovereignty layer for all stablecoin holders in the form of programmable self-custody.

References

[1] Coinbase. Coinbase Card: Spend crypto anywhere. `https://www.coinbase.com/card`, 2019.

[2] Revolut. Revolut Crypto: Buy, sell, and spend crypto. `https://www.revolut.com/crypto`, 2018.

[3] Gnosis. Gnosis Pay: On-chain payments. `https://gnosispay.com`, 2023.

[4] Vitalik Buterin, Yoav Weiss, Dror Tirosh, Shahaf Nacson, Alex Forshtat, Kristof Gazso, and Tjaden Hess. ERC-4337: Account Abstraction Using Alt Mempool. https://eips.ethereum.org/EIPS/eip-4337, 2021.

[5] ZeroDev. ZeroDev: Smart wallet infrastructure for ERC-4337. https://zerodev.app, 2023.

[6] World Wide Web Consortium. Web Authentication: An API for accessing Public Key Credentials. https://www.w3.org/TR/webauthn-2/, 2021.

[7] Story Foundation. Story: A peer-to-peer intellectual property network. https://www.story.foundation/whitepaper.pdf, 2025.

[8] Sergey Nazarov and Steve Ellis. Chainlink: A decentralized oracle network. https://chain.link/whitepaper, 2017.

[9] Jayden Windle, Benny Giang, Steve Jang, Druzy Venegas, and Raymond Huynh. ERC-6551: Non-fungible Token Bound Accounts. https://eips.ethereum.org/EIPS/eip-6551, 2023.

[10] MakerDAO. The Maker Protocol: MakerDAO's Multi-Collateral Dai (MCD) System. https://makerdao.com/en/whitepaper, 2019.

[11] Ulaş Erdoğan, Doğan Alpaslan, and Ariel Elperin. RIP-7212: Precompile for secp256r1 Curve Support. https://github.com/ethereum/RIPs/blob/master/RIPS/rip-7212.md, 2023.