YieldLoop Whitepaper

Todd Koletsky 12-17-25 Version: v0.1.1

Table of Contents

1. Abstract

YieldLoop is a cycle-based trading and yield platform designed to turn **real, completed profit** into a persistent system.

The platform runs user-authorized strategy bots inside non-custodial vaults. Each execution cycle is discrete. At the end of a cycle, the system measures actual profit after all costs. If profit exists, a platform fee is taken, profit is converted into LOOP, and the system reinvests its share to keep the engine running. If no profit exists, nothing happens.

There are no projections, no promised APYs, no smoothing, and no automatic compounding without user consent.

LOOP is minted only when profit is real. System-owned LOOP is redeposited to generate future profit. Over time, retained surplus strengthens an internal accounting floor that never erodes due to system logic.
YieldLoop is designed to:

- Survive flat and down markets
- Avoid dependence on emissions or inflation
- Make every cycle auditable and final
- Keep user funds isolated
- Keep the system economically self-sustaining

This document explains how the system actually works, end to end, so it can be built correctly.

2. Problem

Most trading and yield platforms fail for the same reasons:

1. They measure success using **estimates instead of reality**
2. They mix **execution, incentives, and marketing**
3. They depend on **emissions, inflation, or constant growth**
4. They hide losses through smoothing, compounding assumptions, or dashboards
5. They collapse when markets go flat or down

2.1 Fake Yield and Assumed Performance

Many systems advertise yield that is:

- Projected, not realized
- Based on temporary incentives
- Masking losses with emissions
- Continuously re-marked without final settlement

Users see numbers go up even when no real profit exists. When incentives end or markets shift, the system breaks.

2.2 No Clear End to a Trade

In most platforms:

- Trades never truly “end”
- Positions roll forward indefinitely
- Performance is constantly re-evaluated
- Losses are deferred or hidden

This makes it impossible to say what actually happened in a given period.

2.3 Incentives That Fight the System

Platforms often:

- Earn fees on volume, not profit
- Profit when users lose
- Encourage constant reinvestment
- Require growth to survive

These incentives push systems toward risk-taking and leverage, even when it's not sustainable.

2.4 No Separation of Concerns

Execution, accounting, marketing, and token value are usually blended together:

- Tokens are used to pay yield
- Yield props up token price
- Token price props up perceived yield

This creates circular dependency and collapse risk.

2.5 Emissions-Based Death Spirals

Inflationary reward systems:

- Require constant new users
- Dilute existing holders
- Create artificial returns
- Fail when growth slows

Once emissions stop, yield disappears.

2.6 Lack of Finality

Without hard cycle boundaries:

- Users don't know when results are final
- Developers can't reason about state
- Auditing becomes subjective
- Trust degrades

2.7 The Core Problem

The core problem is simple:

Most systems never prove profit. They only assume it.

YieldLoop exists to fix that by forcing every cycle to end, settle, and tell the truth.

3. Solution

YieldLoop solves these problems by forcing **finality, separation, and truth** into the system.
The core idea is simple:
Nothing counts until the cycle ends.

3.1 Discrete Execution Cycles

YieldLoop operates in hard, discrete cycles.
Each cycle:

- Starts with explicit user authorization
- Runs strategy bots within fixed limits
- Ends at a known time
- Settles completely
- Produces a final, unchangeable result

There is no rolling state and no continuous re-evaluation. Every cycle either makes profit or it doesn't.

3.2 Profit Is Real or It Is Zero

At the end of a cycle, the system calculates profit using actual balances and actual costs.
profit = end_balance

- start_balance
- gas
- protocol costs

If profit is:

- `> 0` → it is real profit
- `<= 0` → it is treated as zero

There is no partial credit, smoothing, or carryover.

3.3 Fees Only on Profit

The platform only earns when users earn.

- No profit → no fee
- No volume-based fees
- No incentive to churn trades

- No incentive to increase risk

This aligns the system with long-term survival instead of short-term activity.

3.4 Profit Is Minted, Not Assumed

When profit exists:

- Fees are applied first
- Remaining profit is converted into LOOP
- LOOP represents completed, realized surplus

There is no yield token emission, no pre-mint, and no inflation schedule.
If the system does not produce profit, no LOOP is created.

3.5 Separation of Concerns

YieldLoop explicitly separates:

- Execution (bots)
- Accounting (profit calculation)
- Incentives (fees and LOOP)
- System sustainability (system deposit)

No part of the system depends on another part pretending profit exists.

3.6 No Emissions, No Death Spiral

YieldLoop does not use emissions to fake yield.

- No reward dilution
- No dependence on new users
- No APY promises
- No growth requirement to survive

The system can:

- Grow slowly
- Go sideways
- Shrink and still function.

3.7 System Deposit Creates Longevity

A portion of platform fees is retained by the system and redeposited into future cycles.
This creates:

- A self-funding execution engine

- A growing internal surplus when markets allow
- An internal floor that never erodes due to system logic

The system does not support price. It supports **itself**.

3.8 Finality as a Feature

Every cycle ends. Every result is final. Every number can be audited.
This makes the system:

- Easier to build
- Easier to reason about
- Easier to debug
- Easier to trust

YieldLoop's solution is not more complexity.
It is **less lying**.

4. Executive Summary

YieldLoop is a cycle-based profit system built to survive reality.
Users authorize strategy bots to run inside isolated vaults for a fixed period. At the end of each cycle, the system settles everything, measures real profit, and moves on. If profit exists, the platform takes a fee, profit is minted as LOOP, and the system reinvests its share. If no profit exists, the cycle closes flat.
There are no projections, no rolling balances, no assumed yield, and no emissions.
The system is built around a few non-negotiable rules:

- Nothing happens without user authorization
- No profit means no fees and no LOOP
- Every cycle ends and settles completely
- Users control compounding and redemption
- The system must be able to fund itself

LOOP is not a reward token. It is a representation of completed profit. LOOP only exists because a cycle finished with real surplus after costs. System-owned LOOP is redeposited to generate future profit and keep the platform alive over long periods without relying on growth or dilution.
Platform fees are split four ways:

- Development and operations
- Marketing
- LoopLabs
- System deposit

There is no explicit price support. Any strengthening of internal floor metrics happens automatically as retained surplus accumulates relative to LOOP outstanding. The floor cannot erode due to system logic, but it is not guaranteed to rise.
YieldLoop is intentionally boring:

- No APYs
- No leverage
- No hype mechanics
- No infinite loops

What it provides instead is finality, clarity, and an engine that only grows when it actually works.
This document exists to explain that engine so it can be built correctly.

5. System Architecture Overview

YieldLoop is built as a set of **separable, auditable components**. No single component controls the system end-to-end. Each part has a narrow responsibility.
At a high level, the system consists of:

- User Vaults
- Strategy Bots
- Execution Cycle Controller
- Settlement & Accounting Engine
- LOOP Token
- System Deposit Engine
- DApp / Admin Interfaces

5.1 User Vaults

- Each user has an isolated, non-custodial vault
- Vaults hold user assets
- Vaults enforce execution limits and permissions
- Vaults can be locked and unlocked by cycle state

Vault properties:

- No pooled user funds
- No cross-user risk
- Bots cannot act outside authorized bounds
- Vaults reject withdrawals during active cycles

Each vault tracks:

- Starting balance (cycle start)
- Ending balance (cycle end)

- Authorized strategies
- Post-cycle handling preferences

5.2 Strategy Bots

Strategy bots are execution modules.
They:

- Perform trades or yield actions
- Operate only during active cycles
- Read constraints from the vault
- Stop if constraints are violated

Bots do **not**:

- Decide allocations
- Change parameters mid-cycle
- Coordinate with other bots
- Access system-owned funds unless explicitly authorized

Bots are intentionally dumb and predictable.

5.3 Execution Cycle Controller

The Execution Cycle Controller is the state machine that drives everything.
Responsibilities:

- Transition vaults between cycle states
- Enforce timing boundaries
- Lock and unlock vault access
- Trigger settlement

Cycle states:

1. Inactive
2. Configured
3. Active
4. Settlement
5. Post-Cycle

No component can bypass this controller.

5.4 Settlement & Accounting Engine

This engine runs **once per cycle**, after execution ends.
Responsibilities:

- Snapshot vault balances

- Calculate profit or loss
- Deduct gas and protocol costs
- Determine if profit exists
- Trigger fee logic
- Trigger LOOP minting (if applicable)

Settlement is:

- Atomic
- Deterministic
- Final

Once settlement completes, results cannot be changed.

5.5 LOOP Token Contract

The LOOP contract:

- Mints LOOP only when instructed by settlement
- Never mints during execution
- Has no emissions schedule
- Does not respond to market conditions

LOOP supply grows **only** when profit is real.

5.6 System Deposit Engine

The System Deposit Engine handles system-owned funds.
It:

- Receives the system's portion of platform fees
- Holds non-user-owned LOOP and assets
- Reauthorizes system funds into future cycles
- Enforces non-withdrawable rules

System deposit funds:

- Are never user balances
- Cannot be redeemed by users
- Exist only to keep the platform running

This is the perpetual engine.

5.7 DApp & Admin Interfaces

The UI layer:

- Shows vault state
- Shows cycle status

- Shows finalized results
- Allows configuration and authorization

Admin interfaces:

- Enable / disable strategies
- Adjust fee parameters
- Control system deposits
- Pause execution in emergencies

Admins **cannot**:

- Touch user vault balances
- Modify settled results
- Mint LOOP manually

5.8 Data Flow Summary

User Vault ↓ Strategy Bots ↓ Execution Cycle Controller ↓ Settlement & Accounting ↓ LOOP Minting ↓ User Handling + System Deposit
Each arrow represents a **one-way boundary**.

5.9 Design Principle

If a component fails:

- The cycle ends
- Settlement occurs
- The system moves on

No retries that hide reality. No silent fixes. No magic.

6. Execution Cycle & State Machine

The execution cycle is the backbone of YieldLoop.
Every action in the system is gated by cycle state. No component is allowed to operate outside the current state. This is intentional and non-negotiable.

6.1 Cycle States

Each vault progresses through the following states, in order:

1. **Inactive**
2. **Configured**

3. **Active**
4. **Settlement**
5. **Post-Cycle**

States cannot be skipped, reordered, or partially entered.

6.2 Inactive State

This is the default state.
Characteristics:

- No execution allowed
- Vault fully accessible to the user
- Assets may be deposited or withdrawn
- No parameters are locked

Transitions:

- Inactive → Configured (user sets parameters)

6.3 Configured State

The user has defined cycle parameters but has not started execution.
Locked in this state:

- Strategy selection
- Allocation limits
- Slippage limits
- Post-cycle handling preferences

Still allowed:

- Parameter edits
- Cancel configuration
- Return to Inactive

Not allowed:

- Execution
- Parameter changes once cycle starts

Transitions:

- Configured → Active (user authorizes cycle)
- Configured → Inactive (user cancels)

6.4 Active State

Execution is live.
Characteristics:

- Strategy bots may operate
- Vault is locked against withdrawals
- Parameters are immutable
- System enforces all limits

Rules:

- Bots may only act within authorization
- If a bot fails, it halts
- Other bots may continue if authorized
- No human or admin intervention

Not allowed:

- Withdrawals
- Parameter changes
- Strategy substitution

Transitions:

- Active → Settlement (time expires or execution halts)

6.5 Settlement State

Execution has ended. Accounting begins.

Characteristics:

- No execution allowed
- Vault remains locked
- Settlement engine runs exactly once

Actions:

- Snapshot starting and ending balances
- Calculate gas and protocol costs
- Determine profit or loss
- Apply platform fee (if profit exists)
- Mint LOOP (if profit exists)

Settlement is atomic. If settlement fails, the cycle does not advance.

Transitions:

- Settlement → Post-Cycle (successful settlement)

6.6 Post-Cycle State

The cycle is complete.

Characteristics:

- Vault unlocks
- Results are final

- LOOP balances are visible
- User regains control

Allowed actions:

- Withdraw assets
- Redeem LOOP
- Reconfigure next cycle
- Reauthorize execution
- Remain idle

Not allowed:

- Modifying past results
- Reopening a settled cycle

Transitions:

- Post-Cycle → Configured (new cycle)
- Post-Cycle → Inactive (do nothing)

6.7 Zero or Negative Cycles

If profit ≤ 0:

- No platform fee
- No LOOP minted
- No system allocation
- Cycle closes flat

This is a valid and expected outcome.

6.8 Failure Handling

If any of the following occur during Active state:

- Bot error
- External protocol failure
- Chain instability
- Gas limit issues

Then:

- Execution halts
- System moves to Settlement
- Reality is recorded as-is

No retries. No smoothing. No overrides.

6.9 Emergency Stops

Admins may:

- Prevent new cycles from starting
- Pause execution globally

Admins may NOT:

- Intervene mid-cycle
- Change parameters
- Modify balances
- Override settlement results

Emergency stops preserve safety, not performance.

6.10 Why This Matters

This state machine ensures:

- Finality
- Auditability
- Predictability
- No hidden behavior

Every engineer should assume:
If it didn’t happen inside the cycle, it doesn’t exist.

7. Ecosystem Walkthrough

This section explains how all parts of the YieldLoop ecosystem interact over time, from a user’s first touch through long-term system operation.
This is not a user guide. It is a system flow explanation.

7.1 Entry Points

There are three primary entry points into the ecosystem:

- User deposits and cycle participation
- NFT participation (optional)
- Referral programs (pre- and post-launch)

None of these paths are required to use the core system except direct user authorization of execution cycles.

7.2 User Lifecycle

A typical user lifecycle looks like this:

1. User connects a wallet
2. User deploys or accesses a personal vault
3. User deposits assets
4. User configures a cycle
5. User authorizes execution
6. System runs a cycle
7. Cycle settles
8. User reviews results
9. User withdraws, compounds, or reauthorizes

At no point does the system act without user authorization.

7.3 Vault-Centric Design

Everything in YieldLoop revolves around the user vault.

- Strategies execute from the vault
- Profit is measured at the vault level
- LOOP is credited at the vault level
- Withdrawals and compounding occur at the vault level

There are no shared balances between users.

7.4 System-Owned Participation

In parallel with users, the system itself participates:

- The system accumulates LOOP from platform fees
- System-owned LOOP is deposited into cycles
- System profit feeds future cycles
- This creates a self-sustaining execution engine

System participation follows the same cycle rules as users but uses system-owned vaults.

7.5 Optional Programs

Optional ecosystem programs include:

- NFTs
- Referrals
- Governor participation
- Bounty programs

These programs:

- Do not affect execution logic
- Do not change strategy behavior
- Do not change profit calculation
- Do not grant execution privileges

They exist to support growth, coordination, and long-term development.

7.6 Long-Term Loop

Over time, the ecosystem forms a loop:
User Capital + System Capital ↓ Execution Cycles ↓ Real Profit ↓ LOOP Minting ↓ System Deposit Grows ↓ Stronger Internal Floor ↓ More Durable System
This loop can slow down, pause, or reverse in bad markets without breaking the system.

7.7 Key Takeaway

YieldLoop is not designed to maximize short-term yield.
It is designed to:

- Make profit only when possible
- Tell the truth when it is not
- Keep running without dilution
- Let users decide when to participate

Everything else is optional.

8. Presale Referral Program

The presale referral program exists to bootstrap early awareness and community participation before public launch.
This program is optional, time-limited, and separate from core system execution.

8.1 Purpose

The goals of the presale referral program are:

- Seed an initial community
- Reward early outreach and effort
- Identify high-signal contributors before launch
- Create distribution without emissions

This program does not affect trading, yield, execution priority, or profit mechanics.

8.2 Structure

The presale referral program is milestone-based, not volume-based.
Participants earn eligibility by:

- Inviting real people
- Verifying they are human
- Participating meaningfully (not bots or drive-bys)

There is no passive earning.

8.3 Verification Requirements

A referral is considered valid only if the referred participant:

- Is a unique, real human
- Joins official community channels
- Verifies identity as a human (method defined by ops)
- Introduces themselves
- Acknowledges who referred them
- Engages meaningfully (questions, discussion, feedback)

Spam, automation, or fake accounts invalidate eligibility.

8.4 Presale Referral Tiers

Tier 1 — Early Outreach

- Requirement: 25 verified human referrals
- Limit: First 10 participants to reach the threshold
- Reward: Eligibility to receive 1 Supporter NFT at launch

Tier 2 — Early Leadership

- Requirement: 100 verified human referrals
- Limit: First 3 participants to reach the threshold
- Reward: Eligibility to receive 1 Governor NFT at launch

Governor NFTs granted through this program follow the same rules as all Governor NFTs (non-transferable, revocable, advisory only).

8.5 No Financial Claims

Presale referral rewards:

- Are not income
- Are not yield
- Are not compensation
- Do not represent profit-sharing
- Do not imply future benefits

They are recognition for early effort only.

8.6 Enforcement and Revocation

The team may:

- Disqualify referrals for abuse
- Revoke eligibility if fraud is discovered
- Modify or terminate the program

All decisions are final.

8.7 Expiration

The presale referral program:

- Exists only before public launch
- Ends automatically at launch
- Does not carry forward

Unmet thresholds expire without rollover.

8.8 Key Constraint

Presale referrals do not:

- Affect platform fees
- Affect execution logic
- Affect LOOP issuance
- Affect future referral eligibility

They exist entirely outside the core engine.

9. NFT Program

The NFT program is an **optional utility and coordination layer** within the YieldLoop ecosystem.

NFTs do not affect:

- Strategy execution
- Yield generation
- Risk exposure
- Profit calculation
- LOOP minting
- Cycle timing or priority

NFTs are not required to use the platform.

9.1 Purpose

The NFT program exists to:

- Provide protocol-fee discounts
- Signal participation and trusted roles
- Gate access to specific UI and community areas
- Support marketing, partnerships, and early funding

NFTs do **not**:

- Represent ownership or equity
- Grant profit rights
- Guarantee benefits
- Grant execution, admin, or settlement authority

9.2 NFT Tiers

YieldLoop supports two NFT tiers:

1. **Supporter NFT**
2. **Governor NFT**

Each tier has clearly defined and limited utility.

9.3 Supporter NFT

What it is

- A general-purpose utility NFT for supporters of the YieldLoop platform

Pricing

- Initial proposed mint price: **$300 USD equivalent**
- Implemented as a **BNB-denominated mint price**
- The mint price is **adjustable over time via governance**
- Price changes apply **prospectively only**
- No on-chain oracle is required in v1

Supply

- Unlimited

Discount

- Grants a **5% platform-fee discount**
- Applied:
    - Only at settlement
    - Only on profitable cycles
    - Only to the platform fee
- Does not apply to gas, slippage, execution losses, or third-party fees

Access

- Unlocks a **Supporter-only community channel**
- May unlock Supporter-only UI visibility or features

Minting

- Public mint when enabled
- The team may **admin-mint and award Supporter NFTs** for:
    - Marketing
    - Partnerships
    - Promotions
    - Onboarding
    - Community initiatives

Transferability

- Freely transferable
- Benefits follow the wallet holding the NFT

Supporter NFTs do not grant governance, advisory, or decision-making authority.

9.4 Governor NFT

What it is

- A role-based NFT representing a trusted advisory and support position

Acquisition

- Not publicly sold by default
- Typically bestowed by the team via admin mint
- Any future public availability must be explicitly disclosed

Supply

- Limited by policy, not hard-capped

Discount

- Grants a **10% platform-fee discount**
- Overrides Supporter discount if both are present

Role

- Advisory and support only

- May participate in:
    - Strategy discussion (non-binding)
    - Risk review
    - Education and onboarding
    - Partnerships and ecosystem support

Transferability

- **Non-transferable by default**
- Transfer allowed only if:
    - Explicitly authorized by team multisig
    - Approved for a specific token and recipient
- Unauthorized transfers must revert

Revocation

- Governor NFTs may be revoked and burned by admin action
- Reasons include inactivity, abuse, misrepresentation, or loss of trust

One-per-wallet / person

- Intended limit: one Governor NFT per wallet
- Contract-level enforcement recommended
- One-per-person enforced by policy (no KYC required)

Governor NFTs grant **no binding authority**over:

- Contracts
- Treasury
- Execution
- Settlement
- Admin controls

9.5 Mint Proceeds and Treasury

- All NFT mint proceeds route to the **team treasury**
- Used for:
    - Development
    - Audits
    - Infrastructure
    - Operations
    - Long-term platform support

No refunds under any circumstances.

9.6 Discount Enforcement

- Discounts are applied **on-chain at settlement**

- Discounts apply only if profit exists
- Discounts cannot reduce protocol fees below system minimums
- Governor discount overrides Supporter discount

NFT ownership does not affect execution behavior or outcomes.

9.7 Separation From Core Engine

The NFT system is intentionally isolated from:

- Strategy execution
- Yield generation
- Profit measurement
- LOOP issuance
- System deposit mechanics

Removal of the NFT system does not affect core platform operation.

9.8 Relationship to Governance and Bounties

- Governor NFTs may participate in advisory processes
- Governor NFTs may be eligible for discretionary bounties
- Bounties are:
    - Not automatic
    - Not guaranteed
    - Sourced from the marketing budget
    - Defined separately (see Sections 17 and 18)

NFT ownership alone does not entitle holders to bounties.

9.9 Summary

NFTs are optional tools for:

- Fee discounts
- Access
- Recognition
- Coordination

They are **not** part of the yield engine and do not affect how the system generates or records profit.

10. Post-Launch Referral Program

The post-launch referral program exists to encourage organic growth **after** the YieldLoop platform is live.
This program is optional and operates entirely outside the core execution, settlement, and accounting engine.

10.1 Purpose

The goals of the post-launch referral program are:

- Reward users for successful outreach
- Encourage long-term platform adoption
- Avoid emissions, ponzi-style incentives, or perpetual payouts
- Keep referral rewards bounded and auditable

The referral program does not affect:

- Strategy execution
- Risk exposure
- Profit calculation
- LOOP minting rules
- Cycle behavior

10.2 Eligibility

A user becomes eligible to participate in the post-launch referral program only after:

- Completing their first deposit
- Completing at least one full execution cycle

Referral codes are not issued pre-deposit.

10.3 Referral Code Issuance

- Each eligible user receives a unique referral code
- Referral codes may be shared voluntarily
- Use of a referral code is optional for referred users

Referral codes do not expire unless revoked for abuse.

10.4 Qualified Referral

A referral is considered qualified only if:

- A referred user completes a deposit
- The referred user completes execution cycles

- The referred user produces net positive outcomes

If no positive cycle occurs, no referral credit is generated.

10.5 Referral Credit Mechanics

Referral rewards are calculated as:

- **5% of the platform fees** generated by the referred user
- Applies only to the referred user's **first six (6) positive execution cycles**

Key constraints:

- Credits are sourced **only from platform fees**
- Credits do not reduce the referred user's profit
- Credits do not affect execution behavior
- Credits are capped and non-recurring

After six positive cycles, no further referral credits accrue for that referred user.

10.6 Settlement and Timing

- Referral credits are calculated only after settlement
- Credits are recorded as finalized accounting entries
- No estimates or projections are displayed

Referral credits are issued in LOOP or as platform credit (implementation-defined).

10.7 Non-Financial Nature

Referral credits:

- Are not income
- Are not yield
- Are not profit-sharing
- Do not represent compensation for services
- Do not create agency or employment relationships

They are promotional offsets sourced from platform fees.

10.8 Anti-Abuse Controls

The system and administrators may:

- Reject self-referrals
- Detect circular referral patterns
- Revoke referral eligibility for abuse

- Freeze or cancel referral credits

All determinations are final.

10.9 Change Management

The referral program may be:

- Modified
- Paused
- Replaced
- Discontinued

All changes:

- Apply prospectively only
- Do not affect settled cycles
- Are disclosed prior to activation

10.10 Summary

The post-launch referral program is:

- Bounded
- Optional
- Fee-sourced
- Non-inflationary

It rewards real growth without distorting the core execution engine.

11. Ecosystem Walkthrough

This section describes how all major components of YieldLoop interact over time. It is written for builders and reviewers who need to understand **system flow**, not marketing.

11.1 High-Level Flow

At a high level, YieldLoop consists of four interacting layers:

1. User-owned vaults
2. Execution strategies
3. Settlement and accounting
4. LOOP issuance and system recycling

Optional programs (NFTs, referrals, governors) sit **outside** this flow and do not interfere with it.

11.2 User Entry

A user enters the system by:

1. Connecting a wallet
2. Deploying or accessing a personal vault
3. Depositing assets into that vault
4. Configuring execution parameters
5. Explicitly authorizing a cycle

No pooled capital exists at any point.

11.3 Execution Layer

Once a cycle is authorized:

- Strategy modules operate only within the vault
- All actions are bounded by user parameters
- No human intervention occurs mid-cycle
- No configuration changes are allowed

Execution halts automatically at cycle end or on failure conditions.

11.4 Settlement and Accounting Layer

After execution ends:

- All positions are finalized
- Costs are accounted for
- Start and end vault balances are compared
- A definitive result is produced

There are only two possible outcomes:

- Positive cycle
- Non-positive cycle (zeroed)

No smoothing or carryover occurs.

11.5 LOOP Issuance Layer

If a cycle is positive:

- Net profit is converted into LOOP

- LOOP is fully backed by realized profit
- LOOP is issued to the vault according to user-selected settings:
    - Withdraw
    - Compound
    - Partial split

If a cycle is not positive:

- No LOOP is issued

11.6 Platform Participation

In parallel with users:

- Platform fees are collected from positive cycles
- Fees are converted into LOOP
- Platform-owned LOOP is deposited into system vaults
- System vaults participate in future cycles

This creates a **self-reinforcing execution loop** without emissions.

11.7 Capital Recycling Loop

Over time, the system forms this loop:
User Capital + System Capital ↓ Execution Cycles ↓ Realized Profit ↓ LOOP Issuance ↓ System Deposit Grows ↓ More Capital per Cycle ↓ Greater System Resilience
This loop can shrink, pause, or reverse without breaking the protocol.

11.8 Optional Programs (Isolated)

NFTs, referrals, governors, and bounties:

- Do not alter execution
- Do not affect settlement
- Do not mint LOOP
- Do not change strategy behavior

They operate purely at the **coordination and incentive layer**.

11.9 Key Design Result

YieldLoop does not depend on:

- Continuous inflows
- Emissions

- Narrative APYs
- User churn

It survives by:

- Executing only when authorized
- Issuing LOOP only when profit exists
- Recycling capital deterministically

This is the core ecosystem logic.

12. Post-Cycle Walkthrough & Platform Fees

This section describes **exactly what happens at the end of an execution cycle**, how results are processed, and how platform fees are handled.
Everything in this section occurs **after execution has fully stopped**.

12.1 Cycle Completion

An execution cycle ends when:

- The predefined cycle duration expires, or
- Execution halts due to failure conditions

At this point:

- No further trades or actions may occur
- Vault state is frozen for settlement
- No user intervention is possible

12.2 Outcome Determination

The system compares:

- Vault balance at cycle start
- Vault balance at cycle end

After subtracting:

- Gas costs
- Protocol interaction fees
- Any execution-related expenses

Two outcomes are possible:

- **Non-positive cycle**
    - No profit

    - No LOOP minted
    - No platform fee charged
- **Positive cycle**
    - Net profit exists
    - LOOP minting proceeds
    - Platform fee applies

There is no partial credit.

12.3 Positive Cycle Flow (User Side)

If a cycle is positive:

1. Net profit is finalized
2. Profit is **converted into LOOP**
3. LOOP is fully backed by realized profit
4. LOOP is issued to the user vault

The user's **pre-selected post-cycle setting**determines how issued LOOP is handled:

- Withdraw LOOP
- Compound LOOP
- Partial split (withdraw / compound)

No other behavior is permitted.

12.4 Platform Fee Application

After profit is finalized but **before user LOOP distribution**:

- A platform fee is calculated as a percentage of net profit
- The fee applies **only to profit**
- No fees are taken from principal

If no profit exists, this section is skipped entirely.

12.5 Platform Fee Split (4-Way)

Collected platform fees are split **equally across four destinations**:

1. **Dev / Ops**
    - Engineering
    - Infrastructure
    - Audits
    - Maintenance
2. **Marketing**
    - Growth

    - Partnerships
    - Community programs
    - Referral funding
    - Governor bounties
3. **LoopLabs**
    - Research
    - Experimental strategy development
    - Long-term system design
    - Non-production testing
4. **System Deposit**
    - Converted to LOOP
    - Deposited into system-owned vaults
    - Re-enters future execution cycles

Each destination receives **25% of the platform fee**.

12.6 System Deposit Mechanics

The **System Deposit** is critical.

- Platform-owned LOOP is treated exactly like user LOOP
- It enters execution cycles
- It earns or loses based on real performance
- It compounds over time when profitable

This creates a **perpetual internal participant**without emissions.

If the system deposit performs poorly:

- It shrinks
- It does not get subsidized
- It does not get replenished by minting

12.7 No Floor Manipulation

There is:

- No artificial price support
- No buyback-and-burn
- No guaranteed floor movement

The LOOP floor strengthens **only** by:

- Real profit
- System deposits
- Backed issuance

The floor may:

- Rise
- Flatten
- Temporarily decline

No guarantees exist.

12.8 Accounting Finalization

Once all splits and distributions are complete:

- Accounting records are finalized
- Results become immutable
- Users regain full control
- Withdrawals and reauthorization become available

Nothing from this cycle carries assumptions into the next.

12.9 Key Takeaway

YieldLoop does not promise yield.
It:

- Executes
- Measures
- Settles
- Issues LOOP only when real profit exists
- Recycles capital deterministically

Everything else is noise.

13. LOOP Token and Mechanics

LOOP is the **accounting and settlement token** of the YieldLoop system.
LOOP is not emitted on a schedule, not inflated, and not used as an incentive. LOOP exists **only** as a representation of realized, settled profit.

13.1 What LOOP Is

LOOP is a **receipt for verified profit**.
Every unit of LOOP corresponds to profit that:

- Was actually earned
- Was fully settled

- Was not estimated, projected, or assumed

LOOP is used to:

- Represent realized profit
- Enable compounding without custody
- Allow both users and the system to re-enter cycles using the same rules

LOOP is **not**:

- A reward token
- A yield emission
- A speculative asset
- A price-managed token

13.2 When LOOP Is Minted

LOOP is minted **only** at the end of an execution cycle and **only** if that cycle is positive.
All of the following must be true:

- Execution has fully stopped
- Settlement is complete
- Net profit exists after all costs

If any condition fails:

- No LOOP is minted
- No platform fee is charged
- The cycle records zero

There is no partial minting.

13.3 Source of LOOP and Backing

LOOP is minted directly from **realized profit**.
Minting flow:

1. Execution completes
2. Net profit is finalized
3. Platform fee is calculated
4. Remaining profit is converted into LOOP
5. LOOP is issued to the vault

There is:

- No pre-mint
- No discretionary minting
- No emission schedule
- No inflation

Every LOOP minted is backed by profit that already exists.

13.4 Post-Cycle Distribution

Once LOOP is minted, it is handled according to **user-selected post-cycle parameters**, chosen before execution:

- Withdraw LOOP
- Compound LOOP
- Partial withdraw / compound split

These parameters:

- Are fixed for the cycle
- Cannot be changed mid-cycle
- Are executed mechanically

13.5 Platform Fee Interaction

Platform fees are applied **before** LOOP is issued to the user.
Flow:

1. Net profit finalized
2. Platform fee calculated
3. Platform fee portion converted into LOOP
4. Remaining LOOP issued to the user vault

Platform-owned LOOP follows the **exact same rules** as user LOOP.

13.6 System-Owned LOOP

The system accumulates LOOP from the **System Deposit** portion of platform fees.
System-owned LOOP:

- Is deposited into system vaults
- Participates in execution cycles
- Gains or fails exactly like user capital
- Receives no protection or subsidy

This creates a **self-funding execution participant** without emissions.

13.7 LOOP Supply Characteristics

LOOP supply is:

- Variable
- Performance-dependent

- Non-inflationary

Supply behavior:

- Increases only when real profit exists
- Pauses when profit does not exist
- Has no target or cap

There is no supply smoothing.

13.8 Accounting Floor (Non-Decreasing)

LOOP has a **non-decreasing accounting floor**.
The floor is defined as:
The cumulative total of all realized, settled profit ever minted into LOOP,
minus any LOOP that has been redeemed or withdrawn.
Once LOOP is minted:

- Its backing is locked
- Its accounting value does not decrease
- It is not clawed back due to future losses

Losses affect **future minting only**, never past LOOP.

13.9 What Can and Cannot Change

What can change

- Rate of future LOOP growth
- Execution frequency
- Time required to redeem
- Liquidity availability during cycles

What cannot change

- Previously minted LOOP backing
- The accounting floor
- Validity of issued LOOP

The floor is monotonic.

13.10 No Retroactive Loss Allocation

YieldLoop does not apply losses retroactively.

- LOOP is not burned due to losses
- LOOP backing is not reduced
- LOOP is not repriced downward

If execution fails, the system simply does not mint new LOOP.

13.11 Redeemability

LOOP is redeemable **inside the YieldLoop system**.
Redemption occurs when:

- A user withdraws LOOP
- LOOP is used as authorized capital in a cycle
- LOOP is converted through execution logic into underlying assets

Redemption is subject to:

- Liquidity
- Execution constraints
- External protocol conditions

There is no promise of instant redemption.

13.12 No Burn, No Buyback

The system does not:

- Burn LOOP
- Buy back LOOP
- Destroy supply intentionally

LOOP supply changes only through:

- Minting from realized profit
- Redemption by users

13.13 Failure Mode Behavior

In extended adverse conditions:

- LOOP issuance pauses
- Platform fees drop to zero
- System deposit growth halts
- Execution cycles may pause

The system enters a **dormant but solvent state**.
Previously issued LOOP remains valid and redeemable under system rules.

13.14 Core Design Outcome

LOOP is **truth encoded as a token**.

- Good cycles grow it

- Bad cycles do not erase it
- Nothing is promised
- Nothing is hidden

LOOP does not speculate. LOOP records reality.

14. Strategy Modules

Strategy modules are **predefined execution programs** that operate inside user-owned vaults during an authorized execution cycle.
They define *what actions are allowed*, not *what outcomes are expected*.

14.1 Purpose

Strategy modules exist to:

- Execute deterministic actions on user-authorized assets
- Interact with approved external protocols
- Operate within strict, inspectable rules
- Produce measurable outcomes at cycle end

Strategy modules do not:

- Make decisions on behalf of users
- Adapt or optimize dynamically
- Predict market behavior
- Guarantee outcomes

14.2 Strategy as Capability, Not Advice

A strategy module represents an **execution capability**.
It is inert unless:

- Explicitly selected by the user
- Explicitly authorized for a specific cycle
- Explicitly bounded by user parameters

The system does not:

- Recommend strategies
- Rank strategies
- Auto-select strategies
- Substitute strategies mid-cycle

This prevents the system from functioning as managed trading or discretionary execution.

14.3 Deterministic Behavior

Each strategy module is deterministic:

- Given the same inputs and on-chain conditions, behavior is consistent
- No learning
- No AI-driven modification
- No mid-cycle mutation

If a rule fails, the module halts.
The system never improvises.

14.4 Execution Boundaries (Global)

All strategy modules share the following hard limits:

- No leverage
- No borrowing
- No derivatives
- No margin
- No shorting
- No mid-cycle parameter changes
- No human intervention during execution

These boundaries are non-overridable.

14.5 Initial Strategy Set

The initial strategy set is intentionally limited.

14.5.1 Spot Trading — Major Assets

Purpose

- Execute simple buy/sell spot trades on approved venues

Assets

- BTC
- ETH
- SOL
- Additional assets may be added prospectively

Behavior

- Spot only
- User-defined limits on size, frequency, slippage

Exclusions

- No leverage
- No derivatives
- No shorting

14.5.2 Spot Trading — PAXG

Purpose

- Spot trading of tokenized gold (PAXG)

Behavior

- Spot only
- Approved venues only
- User-defined limits apply

Notes

- No claims about physical redemption
- Issuer and custody risks are external

14.5.3 LP / Yield Vault Deployment

Purpose

- Deploy assets into approved LPs or vault-style protocols

Behavior

- Deposit and withdraw only
- Reward claiming only if explicitly enabled
- No forced compounding unless authorized

Risks

- Impermanent loss
- Protocol risk
- Liquidity risk

14.5.4 Stablecoin Yield Deployment

Purpose

- Deploy stablecoins into non-staking yield sources

Behavior

- Lending markets or vaults only
- Approved venues only
- Withdraw per authorization

Notes

- No peg guarantees
- No insolvency protection

14.6 User-Controlled Parameters

Users define, per cycle:

- Strategy selection
- Asset inclusion
- Capital allocation per strategy
- Slippage limits
- Execution frequency
- Reward handling (claim / idle)

Defaults may exist but imply no recommendation.

14.7 Halt Conditions

Execution halts if:

- Slippage exceeds limits
- Liquidity checks fail
- External protocol fails
- Execution errors occur
- Emergency controls activate

Halt means:

- No retries
- No substitution
- Proceed to settlement

14.8 Strategy Isolation

Strategies:

- Do not share state
- Do not coordinate implicitly
- Do not rebalance against each other
- Do not migrate capital mid-cycle

Each strategy's effect is visible only at settlement.

14.9 Strategy Lifecycle

1. Defined and audited
2. Approved for availability
3. User-selected
4. User-authorized

5. Executed during cycle
6. Halted at cycle end
7. Measured in settlement

No shortcuts.

14.10 Removal and Change

Strategies may be:

- Disabled
- Modified
- Replaced

All changes:

- Apply prospectively only
- Do not affect active cycles
- Are documented

14.11 Key Takeaway

Strategies are **tools**, not promises.
They execute rules. They do not manage outcomes.

15. DApp / UI / UX

The YieldLoop DApp exists to **accurately reflect system behavior**, enforce authorization boundaries, and ensure users act with **clear awareness of risk and limitations**.
The interface must not persuade, optimize, recommend, or imply outcomes.
It must surface **explicit warnings, disclosures, and constraints** at all relevant decision points.

15.1 Core UI Principles

The UI must:

- Never imply expected returns or yield
- Never display projected performance
- Never rank strategies or users
- Never gamify execution

- Never obscure risk or constraints
- Never auto-authorize actions

Clarity and correctness take priority over engagement.

15.2 Mandatory Global Disclosures

The following disclosures must be **visible and accessible at all times**, including via a persistent footer or modal link:

- YieldLoop executes user-authorized actions only
- YieldLoop does not provide investment advice
- YieldLoop does not guarantee profit or capital preservation
- YieldLoop is non-custodial
- Loss of funds is possible, including total loss
- Smart contract and protocol risks exist
- Past results do not predict future outcomes

Users must be able to view these disclosures without authorizing execution.

15.3 Primary Screens

The DApp includes:

1. Wallet Connection
2. Vault Overview
3. Cycle Configuration
4. Authorization & Confirmation
5. Active Cycle Status
6. Post-Cycle Results
7. History & Records
8. Optional Programs (NFTs, referrals, governors)
9. Admin (restricted)

Warnings and notices appear contextually within these screens.

15.4 Wallet Connection

Wallet connection screen must display:

- Supported networks
- Non-custodial notice
- “Connecting a wallet does not initiate trading” notice

No execution may occur on wallet connection alone.

15.5 Vault Overview

Displays:

- Vault address
- Asset balances
- Idle vs authorized capital
- Current cycle state
- LOOP balance (if any)

Must include a notice:
“Assets in this vault remain under your control. Execution occurs only when you explicitly authorize a cycle.”
No performance projections are shown.

15.6 Cycle Configuration

During configuration, the UI must:

- Clearly label all parameters as user-selected
- Display default values as defaults, not recommendations
- Prevent configuration if inputs are invalid

Mandatory notice:
“These settings define what actions may occur. YieldLoop will not modify or optimize them.”

15.7 Authorization & Confirmation (Critical)

Before authorization, the UI must present a **blocking confirmation screen** that includes:

- Full parameter summary
- Strategy modules enabled
- Assets involved
- Capital allocation
- Cycle duration
- Platform fee rate
- NFT discount (if applicable)
- Statement that execution cannot be modified mid-cycle
- Statement that withdrawals are disabled during execution
- Statement that losses are possible

The user must explicitly acknowledge:

- “I understand that YieldLoop does not guarantee profit.”

- “I understand that I may lose some or all of my funds.”
- “I understand that execution cannot be stopped once authorized.”

Authorization requires an affirmative action (checkbox + confirm).

15.8 Active Cycle Status

During an active cycle:

- UI must clearly show “EXECUTION IN PROGRESS”
- Configuration controls must be disabled
- Withdrawal controls must be disabled

Mandatory notice:
“Execution is active. Parameters cannot be changed and funds cannot be withdrawn until settlement completes.”
No metrics, charts, or live P&L are shown.

15.9 Post-Cycle Results

After settlement, the UI displays:

- Starting vault balance
- Ending vault balance
- Net result
- LOOP issued (if any)
- Platform fee charged (if any)

Mandatory notice:
“Results shown are historical and final. They do not predict future performance.”

15.10 History and Records

Users may view:

- Prior cycles
- Authorization parameters
- Settlement records
- Accounting entries

Records are read-only and immutable.

15.11 Optional Programs

NFTs, referrals, and governor features:

- Must be visually separated from execution controls
- Must include notices stating they do not affect trading outcomes
- Must not be framed as profit opportunities

15.12 Error Handling and Failure Notices

Errors must:

- Clearly describe what failed
- Explain whether execution occurred or halted
- Never retry silently
- Never hide partial failures

If execution halts, the UI must state:
“Execution stopped due to a failure condition. Settlement will proceed using current vault state.”

15.13 UX Safeguards

The UI must prevent:

- Accidental authorization
- Confusing terminology
- Implicit guarantees
- Hidden dependencies

All irreversible actions must be clearly labeled.

15.14 Accessibility and Simplicity

The UI favors:

- Plain language
- Minimal steps
- Explicit state indicators
- Clear warnings over aesthetics

15.15 Key Takeaway

The DApp is a **truthful interface**, not a promise.
If something is risky, irreversible, or constrained, the UI must say so clearly and repeatedly.

16. Admin

Administrative functions exist solely to maintain **system operability, security, and continuity**.
Admins do not control user assets, execution outcomes, or profit distribution.

16.1 Scope of Administrative Authority

Admins may perform the following actions:

- Deploy new versions of contracts
- Enable or disable strategy modules for future cycles
- Update supported asset and protocol lists
- Adjust protocol fee parameters prospectively
- Update NFT pricing and discount parameters
- Manage treasury wallets
- Trigger emergency controls

Admins may not:

- Modify active execution cycles
- Alter settled outcomes
- Move user funds
- Override user authorization
- Mint LOOP outside settlement rules

16.2 Multisig Requirement

All privileged actions must be controlled by a **multisig wallet**.

- No single admin key
- No unilateral execution
- Threshold defined by policy

This applies to:

- Contract upgrades
- Treasury actions
- Emergency pauses
- Governor minting and revocation

16.3 Prospective-Only Changes

All admin changes:

- Apply only to future cycles
- Do not affect active cycles
- Do not alter settled accounting

If a change cannot be applied prospectively, it is not permitted.

16.4 Emergency Controls

Admins may activate emergency controls if:

- A critical vulnerability is discovered
- A dependent protocol fails
- Settlement integrity is at risk

Emergency actions may:

- Pause new cycle authorization
- Halt active execution

Emergency actions may not:

- Re-route assets
- Complete execution after halt
- Modify accounting

16.5 Transparency and Logging

All admin actions must:

- Emit on-chain events
- Be logged and inspectable
- Be attributable to multisig approval

Hidden admin behavior is not allowed.

16.6 Admin UI

Admin interfaces are:

- Access-controlled
- Isolated from user UI
- Logged
- Non-public by default

No admin function is exposed to end users.

16.7 Removal and Rotation

Admin keys may be:

- Rotated
- Revoked
- Reassigned

Rotation does not affect system behavior.

16.8 What Admins Are Not

Admins are not:

- Traders
- Asset managers
- Advisors
- Fiduciaries

They operate the system — they do not participate in outcomes.

16.9 Key Takeaway

Admins keep the system **running and honest**.
They do not make money for users.

17. Customizable Governance System for Governors

YieldLoop includes an **advisory governance framework** designed to collect structured input without surrendering operational control or creating implied authority.
At launch, YieldLoop is **not a DAO**.
Governance exists to inform decisions, not to execute them.

17.1 Purpose

The governance system exists to:

- Gather informed feedback from trusted participants
- Stress-test system changes before deployment
- Improve long-term design and risk awareness

- Create structured discussion without binding power

Governance does not:

- Control execution
- Control funds
- Approve trades
- Override admin authority
- Create ownership or fiduciary duty

17.2 Governor Role Recap

Governors are holders of the **Governor NFT**and serve as:

- Advisors
- Reviewers
- Contributors
- Thought partners

They are not:

- Operators
- Trustees
- Controllers
- Decision-makers

17.3 Scope of Governance Topics

Governance discussions may include:

- Strategy module enablement or removal
- Risk boundaries and constraints
- Fee structure changes
- NFT program adjustments
- UI/UX clarity improvements
- Roadmap prioritization
- Emergency response post-mortems

Governance may not vote on:

- Live execution
- User funds
- Individual outcomes
- Treasury withdrawals
- Contract keys

17.4 Governance Mechanics

Governance input may be collected via:

- Non-binding votes
- Structured proposals
- Comment periods
- Think-tank sessions
- Written reviews

All governance outputs are **advisory only**.

17.5 Customizability

Governance mechanics are configurable:

- Voting weights (if any)
- Participation thresholds
- Topic scope
- Session frequency
- Visibility (public vs private)

Changes to governance mechanics:

- Apply prospectively
- Do not imply retroactive authority
- Do not grant rights

17.6 No Reliance or Expectation

Participation in governance:

- Does not guarantee influence
- Does not guarantee compensation
- Does not imply responsibility
- Does not create reliance by users

The team retains final authority.

17.7 Transparency

Governance discussions may be:

- Logged
- Published
- Archived

Transparency is encouraged but not required for all sessions.

17.8 Removal and Participation Limits

Governor participation may be:

- Limited
- Rotated
- Revoked

Inactivity, abuse, or misrepresentation may result in removal.

17.9 Key Takeaway

Governance is a **signal channel**, not a control plane.
The system listens — it does not obey.

18. Governors Bounty Program

The Governors Bounty Program exists to **reward meaningful advisory and ecosystem contributions** without creating employment, compensation, revenue-sharing, or yield expectations.
Bounties are discretionary, bounded, and non-recurring.

18.1 Purpose

The purpose of the Governors Bounty Program is to:

- Incentivize high-quality advisory input
- Reward time-bound, outcome-specific contributions
- Encourage honest critique and risk discovery
- Support ecosystem growth and coordination

The program is not designed to:

- Provide income
- Create recurring payments
- Substitute for employment or contracting
- Distribute protocol revenue

18.2 Eligibility

Eligibility requires:

- Holding a valid Governor NFT
- Being invited or approved for a specific bounty activity
- Completing defined participation requirements

Holding a Governor NFT alone does **not**guarantee eligibility.

18.3 Bounty Types

Bounties may be issued for activities such as:

- Think-tank participation
- Strategy review and critique
- Risk analysis and adversarial testing
- Documentation review
- Educational content creation
- Partnerships and outreach support
- Post-incident or post-mortem analysis

Each bounty is tied to a **specific task or session**.

18.4 Funding Source

Governor bounties are funded **exclusively from the marketing budget**.

- Marketing budget is funded from the platform fee split
- A configurable **0.5%–1.0% of the marketing allocation** may be reserved for bounties
- No additional token minting occurs for bounties

Bounties are a **use of funds**, not a revenue stream.

18.5 Bounty Definition (Per Instance)

Each bounty instance must define in advance:

- Purpose or task
- Eligibility requirements
- Participation window
- Verification criteria
- Total bounty pool amount
- Distribution method

No open-ended or perpetual bounties are allowed.

18.6 Distribution Mechanics

Bounties are distributed:

- Manually in v1
- Via multisig approval
- In LOOP or another designated token

Distribution may be:

- Equal split among verified participants
- Fixed amounts per participant
- Other predefined structures

No retroactive changes.

18.7 Cool-Down and Rotation

To prevent concentration and implied compensation patterns:

- A Governor who receives a bounty becomes **ineligible for the immediately following bounty**
- Cool-down applies regardless of bounty type
- Governors who do not receive a bounty are not affected

This enforces rotation and diversity of input.

18.8 No Entitlement, No Reliance

Bounties:

- Are not guaranteed
- Do not accrue
- Do not imply future eligibility
- Do not create expectations of continuity

Failure to receive a bounty carries no penalty.

18.9 No Employment or Agency

Participation in the bounty program:

- Does not create employment
- Does not create contractor status
- Does not create agency
- Does not create fiduciary duty
- Does not authorize representation of YieldLoop

All participation is voluntary.

18.10 Transparency and Logging

Bounty distributions may be:

- Logged
- Summarized
- Audited internally

Public disclosure is optional and policy-defined.

18.11 Termination and Modification

The bounty program may be:

- Modified
- Paused
- Discontinued

All changes:

- Apply prospectively only
- Do not affect completed distributions

18.12 Key Takeaway

Governor bounties are **discretionary recognition**, not compensation.
They reward contribution — not position, ownership, or expectation.

19. Failure Modes & Safeguards

This section enumerates known failure modes and the safeguards designed to contain them.
YieldLoop does not assume continuous success.
It is designed to **fail closed**, not fail optimistically.

19.1 Strategy-Level Failures

Failure Modes

- Slippage exceeds bounds
- Liquidity disappears mid-execution
- External protocol reverts or pauses

- Unexpected price movement

Safeguards

- Hard slippage caps
- Pre-trade liquidity checks
- Immediate execution halt on violation
- Proceed directly to settlement

No retries. No substitutions.

19.2 Smart Contract Failures

Failure Modes

- Contract bugs
- Unexpected state transitions
- Reentrancy attempts

Safeguards

- Audited contracts
- Minimal state mutation
- No external callbacks during settlement
- Emergency pause controls

19.3 Oracle and Data Failures

Failure Modes

- Price feed manipulation
- Oracle outages
- Stale data

Safeguards

- Minimal oracle dependence
- Conservative validation
- Execution halt on inconsistent data

Execution does not proceed on uncertain inputs.

19.4 Execution Engine Failures

Failure Modes

- Gas spikes
- Network congestion
- Partial execution

Safeguards

- Gas ceiling enforcement
- Pre-flight gas estimation
- Abort on partial completion
- Settlement using last valid state

19.5 Accounting and Settlement Failures

Failure Modes

- Inconsistent balance snapshots
- Fee miscalculation
- LOOP mint mismatch

Safeguards

- Deterministic accounting paths
- Single settlement authority per cycle
- Settlement reversion on mismatch
- No manual overrides

19.6 Admin and Governance Failures

Failure Modes

- Compromised admin key
- Malicious or negligent admin action
- Governance capture attempts

Safeguards

- Multisig enforcement
- Role separation
- Prospective-only changes
- Event logging and review

19.7 Economic and Market Failures

Failure Modes

- Prolonged unprofitable conditions
- Market regime shifts
- Liquidity fragmentation

Safeguards

- No emissions
- No obligations to execute
- Cycle pause capability

- Dormant-but-solvent design

The system can wait.

19.8 User Error

Failure Modes

- Misconfiguration
- Over-allocation
- Misunderstanding risk

Safeguards

- Explicit warnings
- Hard parameter bounds
- Confirmation steps
- Immutable authorization

Users cannot accidentally authorize unsafe behavior.

19.9 External Dependency Failure

Failure Modes

- DEX outages
- Bridge failures
- Stablecoin depegs

Safeguards

- Whitelisted dependencies
- Rapid disablement of affected strategies
- Execution halt on detection

No forced exposure.

19.10 Catastrophic Failure

Failure Modes

- Chain halt
- Severe exploit
- Unrecoverable state corruption

Safeguards

- Global emergency pause
- Preservation of vault ownership
- No post-halt execution

Recovery prioritizes fund safety over continuity.

19.11 Design Philosophy

YieldLoop assumes:

- Things will break
- Markets will change
- Humans will make mistakes

Safeguards exist to **stop damage**, not to save face.

19.12 Key Takeaway

When something goes wrong, the system:

- Stops
- Records what happened
- Preserves ownership
- Refuses to guess

Failure is contained, not hidden.

20. Assumptions & Constraints

YieldLoop is built on explicit assumptions and hard constraints.
Anything outside these bounds is intentionally unsupported.

20.1 Market Assumptions

The system assumes:

- Markets are volatile
- Liquidity can disappear without warning
- Profit is intermittent, not continuous
- Lossless execution is impossible

YieldLoop does not assume:

- Efficient markets
- Stable correlations
- Continuous profitability
- Predictable outcomes

20.2 User Assumptions

The system assumes users:

- Understand they may lose some or all of their funds
- Are capable of authorizing actions knowingly
- Accept irreversible execution once a cycle starts
- Are responsible for their own configuration choices

The system does not assume:

- Financial sophistication
- Risk tolerance
- Homogeneous user behavior

Hence the UI enforces guardrails.

20.3 Technical Assumptions

The system assumes:

- Underlying blockchains continue to operate
- Smart contract execution is deterministic
- External protocols behave within documented bounds
- Network congestion may occur

YieldLoop does not assume:

- Zero bugs
- Perfect uptime
- Instant finality
- Infinite throughput

20.4 Legal and Regulatory Constraints (Internal)

Internally, the system is constrained to:

- Avoid representations of guaranteed returns
- Avoid projections or promises
- Avoid discretionary control over user assets
- Avoid retroactive modification of outcomes

These constraints shape architecture, not marketing.

20.5 Execution Constraints

Hard constraints include:

- No leverage

- No borrowing
- No derivatives
- No margin
- No shorting
- No mid-cycle modification
- No discretionary execution

These are architectural limits, not policy choices.

20.6 Accounting Constraints

Accounting is constrained to:

- End-of-cycle settlement only
- Binary outcome recognition (profit or zero)
- No smoothing
- No carryover assumptions
- No retroactive changes

This prevents narrative accounting.

20.7 Token Constraints

LOOP is constrained to:

- Mint only from realized profit
- Never decrease its accounting floor
- Never inflate supply arbitrarily
- Never burn or reprice retroactively

LOOP cannot be used as an incentive lever.

20.8 Admin Constraints

Admins are constrained to:

- Prospective-only changes
- Multisig-controlled actions
- No access to user funds
- No override of settlement

Admin power is deliberately limited.

20.9 Economic Constraints

The system accepts:

- Periods of zero growth
- Extended dormancy
- Reduced participation
- Slow recovery

YieldLoop does not require constant inflows to survive.

20.10 What YieldLoop Refuses to Do

YieldLoop will not:

- Promise yield
- Mask losses
- Smooth returns
- Emit inflationary rewards
- Depend on hype-driven growth

These are not oversights. They are design choices.

20.11 Key Takeaway

YieldLoop is not optimized for upside narratives.
It is optimized for:

- Truth
- Durability
- Auditability
- Survival

21. Roadmap

This roadmap reflects **build order, dependency order, and risk order** — not marketing milestones.
Dates are intentionally omitted. Progress is gated by correctness, not speed.

21.1 Phase 0 — Foundations (Complete or In Progress)

Focus: correctness and survivability.

- Core execution cycle logic
- Deterministic settlement engine

- LOOP minting from realized profit only
- Non-decreasing accounting floor
- Vault isolation per user
- Platform fee routing and system deposit
- Emergency pause and halt controls
- Internal documentation and threat modeling

Exit condition:

- System can execute, halt, settle, and remain solvent under failure.

21.2 Phase 1 — MVP Launch

Focus: minimal surface area, maximum clarity.

- Limited strategy modules (spot, LP, stable yield)
- Full DApp UI with warnings and disclosures
- Manual cycle authorization
- Immutable post-cycle records
- NFT minting (Supporter + Governor)
- Post-launch referral program
- Multisig admin controls
- Logging and observability

Exit condition:

- Real users can complete full cycles without ambiguity or intervention.

21.3 Phase 2 — Hardening and Audit

Focus: adversarial validation.

- Independent smart contract audits
- Stress testing under adverse conditions
- Failure injection and halt testing
- UI/UX clarity review
- Governance dry runs (non-binding)
- Bounty-based review sessions

Exit condition:

- Known failure modes are bounded and documented.

21.4 Phase 3 — Strategy Expansion

Focus: controlled capability growth.

- Additional approved assets

- Additional LP / vault integrations
- Strategy parameter refinement
- Conservative execution optimizations
- Improved capital efficiency (within constraints)

All strategy additions:

- Are opt-in
- Are audited
- Apply prospectively only

Exit condition:

- Expanded capability without expanded risk surface.

21.5 Phase 4 — Automation and Tooling

Focus: reducing friction without adding discretion.

- Improved authorization UX
- Cycle scheduling tools
- Reporting and export tools
- Better system observability
- Admin tooling improvements

No automation may:

- Select strategies
- Modify user parameters
- Execute without authorization

Exit condition:

- Reduced friction, unchanged trust model.

21.6 Phase 5 — Ecosystem Maturity

Focus: longevity.

- Governance refinement
- Governor program evolution
- LoopLabs research outputs
- Documentation and education
- External integrations (selective)

Growth is measured by:

- System uptime
- Accounting integrity
- User understanding
- Survival through bad markets

21.7 Explicit Non-Goals

YieldLoop does not aim to:

- Chase APYs
- Maximize TVL at all costs
- Compete on hype or speed
- Become a generalized trading platform
- Promise returns

These remain non-goals.

21.8 Final Note

YieldLoop advances when:

- The system is correct
- The risks are understood
- The truth is preserved

If progress slows, that is acceptable. If correctness fails, it is not.