PISKY: Autonomous Financial Compute Protocol

Version 2.0
January 2026

Abstract

PISKY is a Solana-native, privacy-preserving, stable-denominated coordination protocol for collective capital compounding. It operates as an autonomous treasury management system that continuously monitors, collects, and converts creator fees without human intervention. By blending deterministic treasury automation, AI-driven revenue generation, and emergent governance through simple, rule-based systems, PISKY creates a self-sustaining capital coordination mechanism designed to outlast market cycles, narratives, and individual actors.

At its core, PISKY replaces traditional capital providers with an AI-assisted Treasury OS that automates fee collection from decentralized exchanges like PumpFun, converts volatile assets to stablecoins, and maintains a 1:1 USDC-backed peg for its stable rewards token, sPISKY. The protocol fosters a deflationary economy for its native token, PISKY—a fixed-supply token that serves as the governance and value-accrual asset. Value accrues to PISKY through market buybacks funded by real profits, staking rewards distributed in sPISKY, and permanent supply reductions via transaction burns.

PISKY introduces Cellular Agents—autonomous AI systems that generate real revenue for the collective—and a Lattice-based reputation layer for privacy-preserving coordination. Built on principles of autonomy over authority, transparency over trust, compounding over consumption, and endurance over excitement, PISKY emphasizes mechanical execution, bounded parameters, and emergent order from local rules. All value accrues from realized profits, not emissions or speculation, transforming passive token holding into active participation in a self-improving capital coordination network.

Simple rules forge emergent order and enduring capital.

Table of Contents

Links & Resources
- Website: pisky.xyz
- Twitter/X: @piskyparty
- Telegram: t.me/piskyparty
- PISKY Token Address: BiHnJu8P8hcDEKzVKLzC1D22StvTZjC7AFFUfF2kpump
- sPISKY Token Address: EMSG6AMzo6E8qzajVL9YHDPibdA79zp2FevfyCeTUN5w
- Treasury Address: GaYLfXdt3GHbvtSuGmFX8NuM7yqtqcrGNt2ebdMWyyex

This whitepaper is for informational purposes only and does not constitute financial advice. The PISKY protocol is experimental software, and users should conduct their own research before participating.

1. Introduction

1.1 The Problem

Traditional community tokens and DAOs on Solana face systemic challenges:

- **Treasury Opacity**: Funds accumulate without transparent management or deployment strategies.
- **Human Dependency**: Manual intervention creates bottlenecks, errors, and trust vulnerabilities.
- **Narrative Fragility**: Projects rise and fall based on attention cycles rather than fundamental value creation.

- **Coordination Failures**: Communities struggle to align incentives and compound collective resources.
- **Fragmented Treasury Management**: Volatile revenue streams and inefficient coordination expose projects to volatility.
- **Governance Issues**: Systems often devolve into discretionary control or speculative emissions.
- **Lack of Tools**: Communities lack privacy-preserving reputation, autonomous value generation, and stable, compounding capital.

1.2 The PISKY Solution

PISKY addresses these challenges through autonomous financial compute—a system where treasury management, fee collection, and capital deployment operate algorithmically with full transparency. The protocol transforms passive token holding into active participation in a self-improving capital coordination network.

As a comprehensive protocol for Collective Capital Coordination, PISKY provides:

- **Automated Treasury**: An off-chain, AI-assisted OS that claims fees from PumpFun, swaps to USDC, and maintains sPISKY's stable peg.
- **Deflationary Economy**: PISKY accrues value through buybacks, burns, and sPISKY rewards funded by real profits.
- **Cellular Agents**: AI-driven "workers" that generate revenue autonomously, feeding profits back to the collective.
- **Lattice Governance**: A rule-based system for emergent coordination, paired with on-chain delegated voting.
- **Reputation Service**: An on-chain aggregator scoring user trustworthiness based on activity, enabling trust in decentralized interactions.

PISKY evolves from concepts like Cellular Automata Governance into a robust framework, integrating live components (e.g., Clean Sweep burns, PiskyFi swap) with upcoming features (staking, agents, reputation).

1.3 Key Innovations

- **Autonomous Treasury Operations**: 24/7 automated fee collection, conversion, and deployment.
- **sPISKY Stablecoin**: A 1:1 USDC-backed stable token for protocol stability.

- **Transparent Receipts**: Cryptographically signed records of every treasury action.
- **Collective Capital Compounding**: Systematic reinvestment of protocol revenues.
- **Cellular Agents**: Autonomous AI systems for revenue generation.
- **Lattice Reputation Layer**: Privacy-preserving coordination without financial entitlement.

2. Vision & Philosophy

2.1 Communities as Cellular Automata

PISKY uses cellular automata as a lens to understand community dynamics. Just like cells in Conway's Game of Life, community members are constantly interacting, influencing each other, and collectively determining whether the community thrives or fades.

The chaos is the point. Communities are unpredictable, messy, and beautiful. By studying how simple rules create complex emergent behavior, we can better understand the invisible forces that make communities flourish or fade away.

2.2 Core Principles

- **Autonomy Over Authority**: The protocol operates without gatekeepers. No single entity controls treasury operations—the system executes predefined rules transparently and consistently.
- **Transparency Over Trust**: Every treasury action generates a cryptographically signed, timestamped receipt that anyone can verify. Trust is replaced by verification.
- **Compounding Over Consumption**: Rather than extracting value, the protocol continuously reinvests revenues into collective infrastructure and holder benefits.
- **Endurance Over Excitement**: Built to outlast narratives, cycles, and individual actors. The protocol prioritizes long-term sustainability over short-term gains.

2.3 Design Philosophy

- **Determinism and Rules**: Simple local rules create emergent global order, ensuring predictability and longevity.
- **Real Revenue Focus**: No emissions, inflation, or promised yields—value stems from realized SOL/USDC profits.

- **Privacy and Autonomy**: Off-chain computation with on-chain enforcement; AI assists but policies enforce.
- **Collective Compounding**: Tools for communities to coordinate capital without central points of failure.

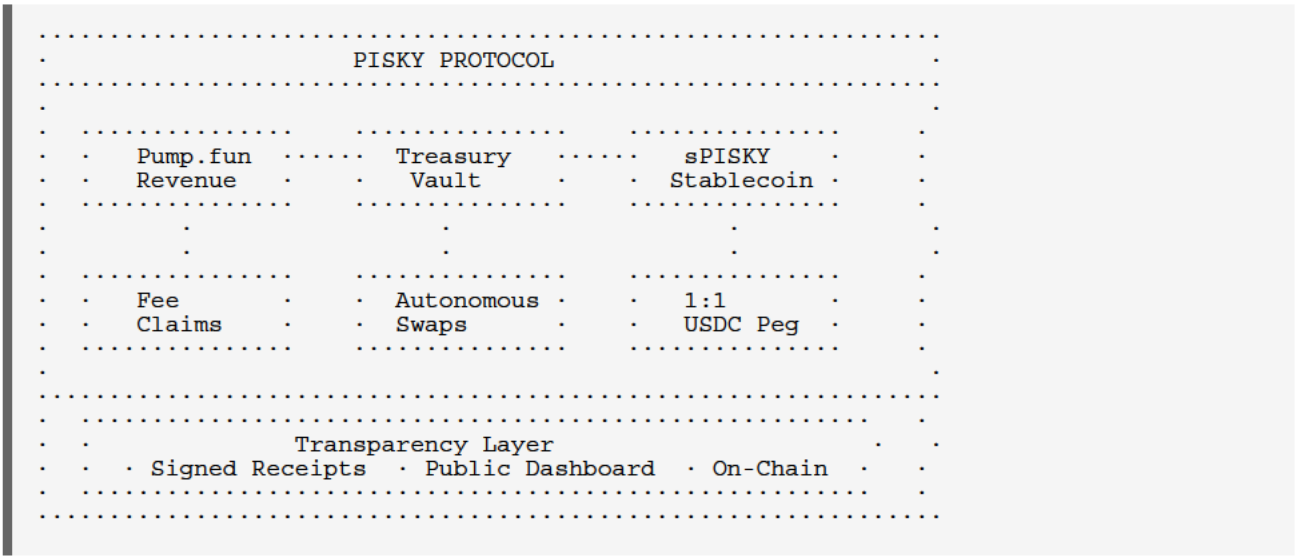
3. Core Design Invariants

These fixed properties cannot be altered by governance:

1. **Fixed-Supply PISKY**: No minting, emissions, or inflation; supply is constant or decreasing via burns.
2. **Revenue-Only Distribution**: Rewards (in sPISKY) come from realized profits or market purchases; no dilution.
3. **Separation of Concerns**: Treasury ≠ speculation; reputation ≠ financial entitlement; agents operate autonomously.
4. **Mechanical Execution**: Bounded parameters, transparent on-chain enforcement, no discretionary control.
5. **Stable Denomination**: All rewards and reserves use sPISKY/USDC for volatility protection.

4. Protocol Architecture

4.1 System Overview



The PISKY Treasury operates as an autonomous system continuously monitoring and managing protocol revenues.

Revenue Sources: - Pump.fun creator fee distributions - PumpSwap trading fees - Protocol activity fees

Operation Cycle: 1. Monitor: Continuously track fee accumulation in the Pump.fun vault 2. Threshold Check: Compare accumulated fees against action threshold (currently 0.02 SOL) 3. Claim: When threshold is met, automatically claim accumulated fees 4. Convert: Swap claimed SOL to USDC via Jupiter aggregator 5. Distribute: Allocate funds according to protocol rules 6. Record: Generate signed receipt for full transparency

Timing: - Operations execute on a regular schedule (approximately every 30-60 minutes) - All actions are logged with timestamps and transaction signatures - Dashboard displays real-time countdown to next action

4.2 Component Stack

Frontend Layer: - React 19 with TypeScript for type-safe development - Tailwind CSS v4 for responsive, modern UI - Framer Motion for fluid animations - Solana Wallet Standard for universal wallet support

Backend Layer: - Express.js server with RESTful API design - PostgreSQL database with Drizzle ORM - Real-time WebSocket connections for live updates - Scheduled jobs for continuous treasury operations

Blockchain Layer: - Solana for high-speed, low-cost transactions - Helius RPC and webhooks for reliable data feeds - Jupiter aggregator for optimal swap execution - SPL Token standard for PISKY and sPISKY

5. Economic Layer

5.1 PISKY Token

PISKY is the fixed-supply, deflationary token at the heart of the protocol, designed for governance, staking, and long-term value accrual.

- **Contract Address**: BiHnJu8P8hcDEKzVKLzC1D22StvTZjC7AFFUfF2kpump
- **Properties**: SPL Token on Solana, 6 decimal places, fair launch via Pump.fun.
- **Supply Mechanics**: Total supply is capped with no new issuance; reductions occur via burns from transaction taxes and buybacks.

- **Accrual Mechanisms**: Value increases through treasury-funded market buybacks (using realized SOL/USDC profits), staking rewards distributed in sPISKY, and direct supply burns.
- **Utility**: Holders can stake PISKY to earn sPISKY rewards, participate in governance voting, and integrate with Cellular Agents for collective benefits.
- **No Direct Usage Requirement**: PISKY benefits indirectly from protocol activity—revenue drives buy pressure, while burns reduce circulating supply, creating scarcity without forcing token use in every action.
- **Economic Role**: As a non-emissive asset, PISKY captures the upside of the ecosystem's real revenue, making it a store of value for participants in collective capital compounding.

5.2 sPISKY Stable Rewards Token

sPISKY is the protocol's stable-denominated token, designed to provide price stability within the ecosystem.

- **Contract Address**: EMSG6AMzo6E8qzajVL9YHDPibdA79zp2FevfyCeTUN5w
- **Peg Mechanism**: 1 sPISKY = 1 USDC (1:1 backing); reserves held in USDC maintain full collateralization; minting and burning automatically maintains the peg.
- **Peg Maintenance**: Treasury OS automates mint/burn to maintain peg via USDC reserves.
- **Rewards Distribution**: Staking rewards and agent profits are paid in sPISKY, funded by market-purchased or converted assets.
- **Benefits**: Provides stable, non-volatile rewards, enabling compounding without SOL exposure.
- **Use Cases**: Stable unit of account for protocol operations; reserve asset for treasury stability; coordination primitive for governance decisions.

5.3 Clean Sweep (Deflationary Burns)

Clean Sweep enables users to consolidate multiple tokens into PISKY through a streamlined interface and applies a 1% transaction tax on PISKY trades.

- **Process**: Connect wallet and view all SPL tokens; select tokens to convert to PISKY; receive quotes via Jupiter aggregator; execute swaps sequentially with progress tracking; mandatory 1% Burn—user approves burning 1% of converted PISKY.
- **Mechanism**: Tax collected in PISKY and burned immediately during transactions.
- **Effects**: Usage-based deflation; supply reduces transparently with activity.
- **Independence**: Operates as a direct burn engine, not a treasury source.

- **Benefits**: Simplifies portfolio consolidation; contributes to PISKY supply reduction through burns; strengthens protocol by increasing holder base.
- **Sources**: Clean Sweep mandatory 1% burns; voluntary burns; protocol-level burns.
- **Transparency**: Total burned displayed on dashboard; individual burn events recorded; full history available via API.

5.4 Treasury OS (AI-Assisted Automation)

The Treasury OS is a deterministic, off-chain framework that automates revenue collection and allocation.

- **Integration**: Claims creator fees from PumpFun vaults, converts SOL to USDC via Jupiter DEX.
- **Tick-Based Loop**: Automates claims, swaps, distributions, and peg rebalancing with configurable thresholds.
- **AI Assistance**: Used for decision rationalization and reporting, bounded by policies (e.g., no overrides).
- **Wallet Structure**: AI Custodian (fee receiver), COLLECTOR (swap executor), TREASURY (USDC/sPISKY holder). The AI Custodian controls the private keys of the wallet that have access to the creator fees.
- **Reconciliation**: Handles interruptions via balance deltas and signed receipts for auditability.
- **Cadence System**: Determines tick intervals using value-based scaling and spike detection for responsive operations.
- **Policy Enforcement**: Deterministic bounds on sweeps; all-or-nothing strategy for efficiency.
- **Self-Sustainability**: Funds its own operations from claimed fees, never touching manually deposited SOL.

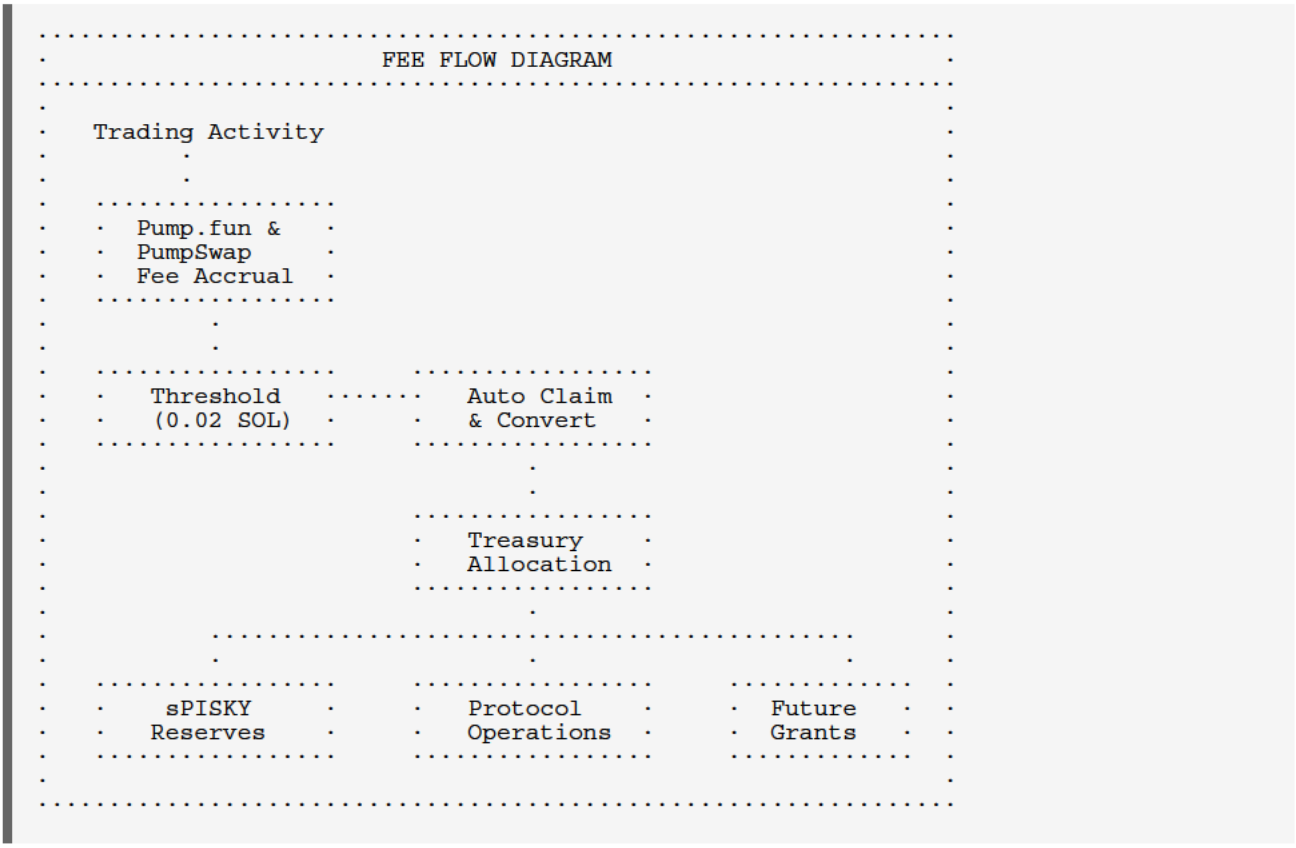
5.5 PiskyFi: Swap and Staking Protocol

PiskyFi provides user-facing tools for interacting with PISKY, focusing on liquidity and rewards.

- **Swap**: Live; integrated with Jupiter aggregator to enable seamless buying and selling of PISKY. Users can swap SOL or other assets for PISKY, with fees contributing to treasury revenue.
- **Staking**: Upcoming; stake PISKY to earn sPISKY rewards directly from treasury allocations. No lockups, no risk to stakers—rewards are funded by realized profits converted to sPISKY.

- **Allocation Example**: 60% profits to PISKY buybacks, 25% to sPISKY staking rewards, 15% to treasury growth.
- **Close Accounts (Rent Recovery)**: Users can recover SOL locked as rent in empty token accounts. Identifies token accounts with zero balance; closes accounts and returns rent to wallet; typical recovery: 0.002-0.003 SOL per account.

5.6 Fee Flow and Deflationary Mechanics



- **Clean Sweep Burns**: 1% of all PISKY acquired through Clean Sweep is permanently burned; burns executed on-chain with verifiable transactions; running total displayed on dashboard.
- **Supply Dynamics**: No new PISKY minting; continuous reduction through burns; long-term deflationary trajectory.

6. Cellular Agents: Autonomous Revenue Generation

6.1 Overview

Cellular Agents are autonomous AI systems acting as "digital workers" for the collective.

- **Characteristics**: 24/7 operation, specialized tasks (e.g., yield farming, arbitrage), revenue-focused.
- **Expansion**: New agents added via governance; profits flow to treasury for buybacks/rewards.
- **Types**: Farm Agents (DeFi yields), Arbitrage Agents (price discrepancies), Trading Agents (market exploitation), Utility Agents (services).
- **Vision**: Transform passive holding into ownership in a fleet of value-generating AIs.

6.2 PISKYBOT: The First Cellular Agent

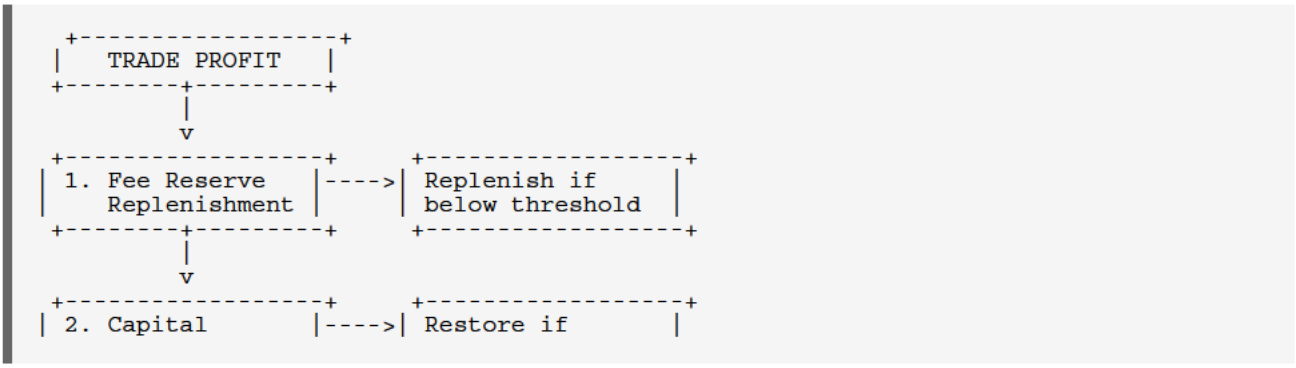
PISKYBOT is a fully autonomous, conservative Solana memecoin trading bot designed with a singular mission: generate profits from memecoin trading and convert those profits into PISKY tokens while preserving original capital.

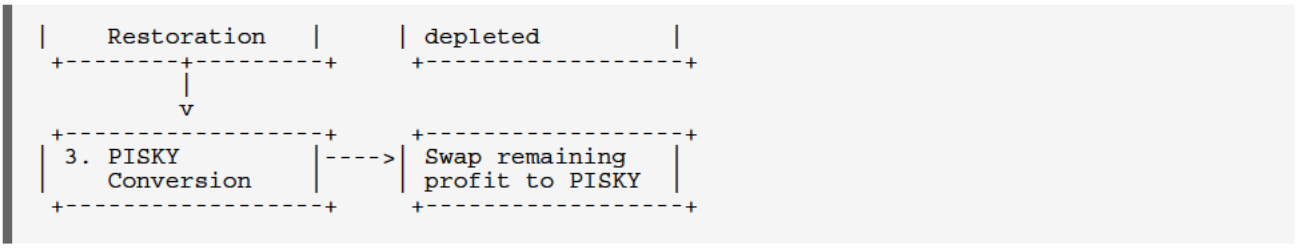
- **Key Design Principles**:

Principle	Implementation
Capital Preservation	Original SOL balance is protected and never risked
Self-Sustaining	Bot pays its own transaction fees from profits
Profit Conversion	All net profits automatically convert to PISKY
Conservative Bias	Multiple safety layers prevent high-risk trades
MEV Protection	Jupiter Ultra API prevents front-running attacks

- **Core Objective: PISKY Accumulation**: The bot exists for one purpose: accumulate PISKY tokens (BiHnJu8P8hcDEKzVKLzC1D22StvTZjC7AFFUfF2kpump).

Profit Flow Diagram:





Profit Routing Logic: ```python def _route_profit(profit_sol, trade): remaining = profit_sol

```
 # Priority 1: Ensure we can keep trading (fees)
fee_deficit = fee_reserve_needed - current_fee_reserve
if fee_deficit > 0:
    replenish = min(remaining, fee_deficit)
    fee_reserve += replenish
    remaining -= replenish

# Priority 2: Protect original investment
capital_deficit = original_capital - current_capital
if capital_deficit > 0:
    restore = min(remaining, capital_deficit)
    remaining -= restore

# Priority 3: THE GOAL - Convert to PISKY
if remaining >= 0.0001:  # min_profit_for_conversion
    jupiter.buy_token(PISKY_MINT, remaining)
```

```

- **Capital Preservation System**: The bot operates on a three-tier reserve system ensuring it can run indefinitely without SOL top-ups.

**Reserve Tiers**:

| Tier | Purpose | Calculation |
|---|---|---|
| Capital Reserve | Original SOL balance (never touched) | starting_balance |
| Fee Reserve | Transaction fee buffer | 3 × 0.00002 SOL × 5 = 0.0003 SOL |
| Trading Balance | Available for positions | total - capital - fees |

**Fee Reserve Mathematics**: Each complete trade cycle requires 3 transactions: 1. Buy token (entry) 2. Sell token (exit) 3. Profit conversion to PISKY

Fee Reserve = 3 × estimated_tx_fee × multiplier = 3 × 0.00002 SOL × 5 = 0.0003 SOL

The 5x multiplier provides safety margin for network congestion.

- **Risk Management Mathematics**: **Position Sizing**: Position size is calculated dynamically based on signal confidence: `python base_size = portfolio_balance × (position_size_percent / 100) # Confidence multiplier if probability >= 85%: multiplier = 1.5 # High confidence = larger position elif probability >= 75%:`
```

```
multiplier = 1.0 # Medium confidence = standard else: multiplier = 0.75 # Lower confidence = reduced size position = base_size × multiplier # Apply limits position = min(position, max_position_limit) position = min(position, available_balance × 0.95) position = max(position, 0.001) # Minimum trade size
```

Example (1 SOL portfolio, 2% base, 80% probability): base_size = 1.0 × 0.02 = 0.02 SOL multiplier = 1.0 (probability between 75-85%) position = 0.02 × 1.0 = 0.02 SOL (~$3.00 at $150/SOL)

Stop-Loss & Take-Profit Calculations: Prices are calculated as percentages from entry: `python stop_loss_price = entry_price × (1 + stop_loss_percent / 100) take_profit_price = entry_price × (1 + take_profit_percent / 100)`

Default settings (optimized for memecoin volatility):

Parameter	Value	Rationale
Stop-Loss	-2.5%	Tight to cut losses fast (accounts for ~2% slippage)
Take-Profit	+5%	Realistic target based on observed wins
Max Hold Time	8 min	Exit before reversals common in memecoins

Profit/Loss Calculation: `python pnl_percent = ((current_price - entry_price) / entry_price) × 100 pnl_usd = position_size_usd × (pnl_percent / 100) pnl_sol = position_size_sol × (pnl_percent / 100)`

Daily Drawdown Protection: Optional circuit breaker to halt trading after significant losses: `python if drawdown_limit_enabled: if daily_pnl_percent <= -max_daily_drawdown_percent: return False, "Daily drawdown limit reached"`

Default: 15% max drawdown (disabled by default to avoid blocking after restart)

- **Dual-Layer AI Intelligence**: PISKYBOT uses two complementary AI systems working together.

Layer 1: XGBoost ML Model: A gradient-boosted classifier trained to predict profitable trades.

Features Used (13 total):

Feature	Description	Weight Impact

liquidity_usd	Pool liquidity in USD	High - filters illiquid tokens
volume_24h	24-hour trading volume	High - indicates interest
volume_5m	5-minute volume	Medium - recent activity
price_change_5m	5-min price change %	High - momentum indicator
price_change_1h	1-hour price change %	Medium - trend direction
price_change_24h	24-hour price change %	Low - longer trend
rsi_14	14-period RSI	Medium - overbought/oversold
ema_ratio	EMA(5)/EMA(20) ratio	Medium - trend strength
volume_spike	Volume vs average ratio	High - unusual activity
momentum_score	Composite momentum 0-100	High - overall strength
market_cap	Token market cap	Low - size indicator
rug_score	Safety risk 0-100	Very High - rug prevention
holder_count	Number of holders	Medium - distribution

Training Approach: The model is trained on synthetic conservative data with rules ensuring a BUY label only for high-probability scenarios.

Probability Adjustments: Raw ML probability is adjusted by factors like rug checks, liquidity, and momentum.

Layer 2: LLM Agent (GPT-4o-mini): Optional AI agent providing intelligent contextual analysis.

LLM Response Impact: Adjusts probabilities based on recommendations (e.g., "AVOID" penalizes by 50%).

LLM System Prompt (Conservative Bias): Instructed to default to AVOID for red flags, only recommend BUY if confidence >= 70%, and prioritize liquidity.

- **Rug Detection & Safety Checks: Risk Score Calculation (0-100)**:

Check	Points Added	Condition
LP not burned/locked	+20	Neither LP burned nor locked

Mint not renounced	+15	Can create new tokens
Freeze authority active	+10	Can freeze holder balances
Whale concentration	+0.5/% over 20%	Top holder > 20%
Dev wallet high	+10	Dev holds > 22.5%
Low holder count	+15	< 50 holders
Unknown holders	+5	Count unavailable
Low liquidity	+15	Below minimum threshold
Suspicious patterns	+10	Volume > 10x liquidity OR extreme moves
Honeypot detected	+40	Volume but < $1k liquidity
Very new token	+10	< 30 minutes old
New token	+5	< 1 hour old

Bonuses (reduce score): - High liquidity (2x minimum): -5 points - Strong momentum + volume: -10 points

Risk Levels:

Score	Level	Trading Allowed
0-24	LOW	Yes
25-49	MEDIUM	Yes
50-74	HIGH	No
75-100	CRITICAL	No

Pass/Fail Criteria: A token passes if risk_score < 60 AND not is_honeypot AND (lp_burned OR lp_locked OR not required) AND (mint_renounced OR not required) AND liquidity >= min_liquidity × 0.5.

- **Trade Execution Flow**: Complete Trading Cycle:
- DISCOVERY: Fetch trending tokens from Dexscreener; apply basic filters.

- ANALYSIS: Run rug detection; extract ML features; generate probabilities; optional LLM analysis.
- FILTERING: Check cooldowns, momentum, probability thresholds, max positions.
- ENTRY: Calculate position size; execute Jupiter swap; set stops.
- MONITORING: Update price; calculate P&L; check exit conditions.
- EXIT: Execute swap; record outcome; route profit.
- COOLDOWN: Block token for 10 minutes if stop-loss.

Trade States:

State	Description
PENDING	Order placed, awaiting execution
OPEN	Position active, monitoring
CLOSED_PROFIT	Exited at take-profit
CLOSED_LOSS	Exited at stop-loss
CLOSED_TIMEOUT	Exited at max hold time
CANCELLED	Manually cancelled
FAILED	Execution failed

- **Jupiter Ultra API Integration**: Provides MEV protection, gasless swaps, best routes, fast execution, and built-in safety checks.

Swap Execution Flow: ```python def execute_swap(input_mint, output_mint, amount): # 1. Get optimal route order = client.order(UltraOrderRequest(input_mint=input_mint, output_mint=output_mint, amount=amount, taker=wallet_pubkey))

```
 # 2. Sign transaction
tx = VersionedTransaction.from_bytes(order["transaction"])
signed_tx = sign_with_keypair(tx, keypair)

# 3. Execute with retries
for attempt in range(3):
    result = client.execute(signed_tx, order["requestId"])
    if result.success:
        return result
    sleep(1)

return failure
```

```

- **Configuration Reference**: Includes detailed parameters for risk, token filters, capital preservation, and environment variables.

# 7. Governance and Coordination Layer

## 7.1 Lattice: Emergent Reputation System

Evolving from Cellular Automata, Lattice is a grid-based system for visualization and coordination.

- **Grid Structure**: 75x75 interactive grid (5,625 cells); each cell represents one holder; color-coded by balance tier; position based on balance ranking.
- **Cell Assignment**: Automatic, random assignment on first PISKY possession; states evolve via rules (e.g., Seed → Builder based on activity).
- **Signals**: On-chain (holding, staking, voting) + off-chain (linked X activity, capped).
- **Emergent Behavior**: Local rules trigger events like badges, state changes, or proposals—non-financial, bounded.
- **Purpose**: Provides identity, memory, and signal for long-term alignment without rewards.
- **Features**: Pan and zoom navigation; hover tooltips with holder details; connected wallet highlighting; X/Twitter account linking; activity history per holder.
- **Data Pipeline**: Helius webhooks provide real-time transfer events; periodic reconciliation ensures accuracy; PostgreSQL stores holder state and events.

## 7.2 On-Chain Voting with Delegation

- **Mechanism**: Moving fully on-chain; delegated voting for efficiency.
- **Scope**: Adjust parameters (e.g., allocation weights, agent deployments) within bounds.
- **Constraints**: Cannot mint, enable leverage, or bypass timelocks; forward-only changes.

## 7.3 Reputation Service

- **Overview**: On-chain aggregator scoring user trustworthiness via activity profiles.
- **Data Sources**: Apps interacted with, trade history/habits, bad actor associations.
- **Scoring**: Holistic "trustworthiness" metric, similar to RugCheck for tokens but for individuals.
```

- **Utility**: Verifiable profiles for community trust; integrable by any project.
- **Privacy**: Aggregated metrics only; no raw data exposure.

7.4 Voting Power and Initiatives

Voting power is calculated based on: - PISKY Balance: Higher balance = more voting power - Holding Duration: Longer holding = multiplied voting power - Activity: Participation in previous votes

Formula: Voting Power = PISKY Balance × Holding Multiplier

Community members can propose and vote on protocol initiatives through the Initiatives system.

Proposal Types: - Treasury allocation decisions - Protocol parameter changes - Community grants - Feature prioritization

7.5 Privacy-Preserving Coordination

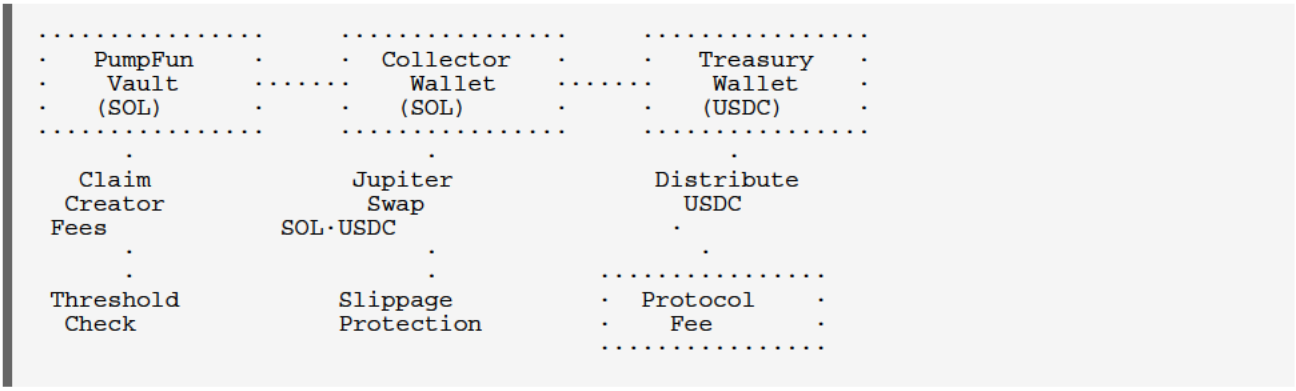
While all treasury operations are transparent, individual holder privacy is respected: - Optional X/Twitter linking - No KYC requirements - Pseudonymous participation - Wallet-based identity

The dashboard provides real-time community statistics: - Total holder count - Active participants - Voting participation rates - Treasury health metrics

8. System Architecture and Flows

8.1 Treasury Flow

Complete Fund Flow:



After each successful swap, USDC is distributed: feeAmount = swapOutputUsdc * (protocolFee.bps / 10000) donationAmount = swapOutputUsdc * (donation.bps / 10000)

toTreasury = swapOutputUsdc - feeAmount - donationAmount

Buffer Management: The collector wallet maintains a SOL buffer for transaction fees.

8.2 Fund Flow Diagram

Fund Flow: PumpFun Fees (SOL) → Treasury OS Claim/Swap → USDC Reserves → sPISKY Mint for Rewards + PISKY Market Buybacks.

Parallel: Cellular Agents Profits → Treasury → Buybacks/Burns + sPISKY Rewards.

8.3 The Cadence System

Three-Tier Priority for tick intervals: - Priority 1: Launch Safety Window (fixed interval during first 24 hours). - Priority 2: Value-Based Baseline (logarithmic scaling with smoothstep for interval calculation). - Priority 3: Spike Detection (accelerates cadence on significant value changes).

9. Security and Enforcement

9.1 Key Management

- Private keys are stored in encrypted environment variables.
- Keys never appear in logs or receipts.
- Each wallet has the minimum necessary permissions.

9.2 Transaction Safety

- Minimum sweep thresholds prevent dust transactions.
- Slippage protection (0.5% default) prevents sandwich attacks.
- Post-swap validation ensures expected USDC outputs.

9.3 RPC Reliability

- Extended confirmation polling (30 seconds).
- Signature status fallback for timeout handling.
- Balance-based verification independent of RPC responses.

9.4 Operational Limits

- Minimum chunk thresholds (0.02 SOL) prevent uneconomical swaps.
- Wallet reserves (0.012 SOL) ensure gas availability for future operations.
- Per-tick reconciliation catches missed state updates from RPC failures.
- Off-Chain/On-Chain Balance: Computation off-chain for flexibility; enforcement on-chain via multisig attestations.
- Protections: Slippage bounds, retries, reconciliation; no single-point control.
- Auditability: Signed receipts for every action; public verification.
- Risks Mitigated: No leverage, principal preservation, bounded AI influence.

10. Economic Model and Loops

PISKY creates self-reinforcing loops driven by real revenue.

10.1 Self-Sustaining Operations

The system is designed to fund itself entirely from claimed fees, never touching manual SOL deposits.

10.2 Transaction Cost Accounting

The wallet reserve covers all operational costs (e.g., ~0.001 SOL per cycle).

10.3 Protocol Fee Structure

Configurable fee extraction from swap outputs (e.g., 100 bps = 1%).

10.4 Expected Yield

Given typical PumpFun creator fee rates (0.1-1% of volume), net treasury inflows scale with activity.

Loops: - **Loop A: Treasury Revenue Loop**: Fees and profits flow to Treasury OS → Converted to USDC → Allocated to sPISKY rewards + PISKY buybacks. - **Loop B: Deflation Loop**: Clean Sweep collects 1% PISKY tax → Immediate burns. - **Loop C: Agent Expansion Loop**: Cellular Agents generate profits → Feed into Treasury → Scale buybacks, rewards, and new agents.

These loops interconnect: Increased activity boosts burns and fees, while profits fund stable rewards and buy pressure, compounding collective capital without emissions.

11. Technical Infrastructure

11.1 Database Schema

Core Tables: - holders: Wallet addresses, balances, Twitter linking - holder_events: Transfer and burn event history - lattice_state: Bootstrap/reconcile timestamps, burn totals - initiatives: Community proposals - votes: Participation records

11.2 API Endpoints

Treasury: - GET /api/treasury/state - Current treasury status - GET /api/treasury/receipts - Recent activity receipts

Lattice: - GET /api/lattice/grid - Holder grid data - POST /api/lattice/bootstrap - Initial holder sync - POST /api/lattice/reconcile - Balance updates

Market: - GET /api/market/price - Current PISKY price - GET /api/market/stats - Market statistics

Clean Sweep: - GET /api/clean-sweep/tokens/:address - Wallet tokens - POST /api/clean-sweep/quote - Conversion quotes

11.3 External Integrations

- **Helius**: RPC endpoints for Solana data; webhooks for real-time transfer notifications; token holder queries.
- **Jupiter**: Swap aggregation for optimal pricing; quote API for Clean Sweep; route optimization.
- **Pisky Vault API**: Real-time treasury state; receipt history; peg status monitoring.

Security Considerations: - Rate limiting on API endpoints - Method whitelisting for RPC proxy - Signed receipts prevent tampering - No private keys stored in application

12. Roadmap

Phase 1: Foundation (Completed):

- ✓ PISKY token launch on Pump.fun
- ✓ Treasury management infrastructure
- ✓ sPISKY stablecoin deployment
- ✓ Holder Lattice visualization
- ✓ Clean Sweep functionality
- ✓ Burn tracking system

Phase 2: Enhancement (In Progress):

- ✓ Real-time treasury dashboard
- ✓ Mobile-responsive interface
- ✓ X/Twitter holder linking
- ■ Enhanced governance tools
- ■ Advanced analytics

Phase 3: Cellular Agents (Coming Soon):

- ■ AI-powered treasury optimization
- ■ Autonomous market operations
- ■ Predictive analytics
- ■ Cross-chain expansion (EVM compatibility)

Phase 4: Full Autonomy:

- ■ Complete autonomous operation
- ■ Zero-intervention treasury management
- ■ Community-driven evolution
- ■ Protocol ossification options

13. Conclusion

PISKY represents a new paradigm in community token infrastructure. By combining autonomous treasury management with transparent operations and collective coordination, the protocol creates a self-sustaining system designed for long-term value creation.

The thesis is simple: when communities operate with clear rules, transparent execution, and aligned incentives, they can achieve outcomes that surpass what any individual actor could accomplish alone. PISKY evolves token projects into resilient ecosystems, emphasizing

mechanical execution over speculation, and collective compounding over extraction.

No emissions. No leverage. Built to endure.