vectorium.co

VECTORIUM

in CryptoEnergy we trust

# VECT.AI

## Proof of Intelligence Protocol

The First Decentralized AI Agent Network for Autonomous Asset Management

Technical & Operational Roadmap | 2-3 Month MVP Build

## Executive Summary

**VECT • AI is NOT a robo-advisor.** It's a decentralized network of AI agents connected to blockchain that:

- Manage digital asset portfolios autonomously
- Learn and improve strategies collectively through Proof of Intelligence
- Interact with other agents in a decentralized marketplace
- Earn and distribute value via the VECT.AI token

### What We're Building

**2-3**
Months to Functional Network

**4**
Core Team Members

**100+**
AI Agents at Launch

**$50M**
Target AUM Year 1

## The Real Vision: Beyond Robo-Advisors

### What VECT • AI Actually Is

| Traditional Robo-Advisor | VECT • AI Protocol |
|---|---|
| Centralized company managing portfolios | Decentralized network of autonomous AI agents |
| Single AI model, proprietary | Multiple competing agents, open marketplace |
| Learns from own users only | Collective intelligence via Proof of Intelligence consensus |
| Requires banking licenses (MiFID, etc.) | Permissionless protocol, no central authority |
| Profit extraction by company | Value distribution via VECT.AI token to agents & users |

### The Competitive Landscape (Real Competitors)

| Project | What They Do | Our Advantage |
|---|---|---|
| Fetch.ai | Autonomous agents for various tasks | We're specialized in asset management with proven ML strategies |
| SingularityNET | Decentralized AI marketplace | We have specific use case + token economics tied to performance |
| Ocean Protocol | Data marketplace for AI | We're agents + data + execution, complete solution |
| Autonolas | Off-chain agent services | We're focused on finance with direct value capture |

### Critical Repositioning

We are NOT targeting:

- Traditional retail market
- Traditional wealth management clients
- Regulated financial product distribution

We ARE building:

- A decentralized protocol for AI agents
- Token-based incentive mechanism (VECT.AI)
- Proof of Intelligence consensus for collective learning
- Global, permissionless access to AI-managed crypto portfolios

## Technical Architecture: What We're Actually Building

### 1. Core Components (2-3 Month Build)

Component 1: AI Agent Runtime

```
Technology: Python + LangChain/AutoGPT framework Purpose: Execute autonomous trading
strategies Key Features: - Portfolio optimization engine (MPT, Black-Litterman) -
Risk management module (stop-loss, position sizing) - Market data integration
(CoinGecko, Binance API) - Decision logging for Proof of Intelligence
```

### Component 2: Blockchain Integration Layer

```
Technology: Solana + @solana/web3.js + On-chain Programs (Rust/Anchor) Purpose: Connect
agents to blockchain for execution & consensus Key Features: - Wallet management
(Phantom, Backpack, Solflare integration) - DEX integration (Orca, Raydium for
execution) - Transaction signing and broadcasting - Agent registry (on-chain identity
via PDAs)
```

### Component 3: VECT.AI Token Economics

```
Technology: SPL Token-2022 + Staking/Locking Program Purpose: Incentivize performance &
govern network Key Features: - Performance-based rewards (agents earn VECT based on
returns) - Staking/locking mechanism (users stake VECT to access top agents) -
Governance (token holders vote on protocol upgrades) - Treasury management (protocol
fees accumulate in FeeVault PDA)
```

### Component 4: Proof of Intelligence Consensus

```
Technology: Federated Learning + IPFS for model storage Purpose: Enable collective
learning without centralizing data Key Features: - Agents submit encrypted model updates
- Consensus mechanism validates contributions - Global model aggregation (weighted by
performance) - Reputation scoring (agents build credibility over time)
```

### Component 5: User Interface

```
Technology: React + @solana/web3.js + Solana Wallet Adapter Purpose: Allow users to
interact with agent network Key Features: - Connect wallet (Phantom, Backpack, Solflare)
- Browse agent marketplace (filter by performance, risk) - Allocate funds to selected
agents - View portfolio performance dashboard
```

## 2. Simplified Architecture Diagram

```
┌──────────────────────────────────────────────────────────────┐ │ USER LAYER │ │ Solana
Wallet → VECT • AI Interface → Agent Marketplace │
└──────────────────────────────────────────────────────────────┘ ↓
```

```
┌──────────────────────────────────────────────────────────────┐ │ BLOCKCHAIN LAYER
(Solana) │ │ Anchor Programs → VECT.AI SPL Token → Agent Registry (PDAs) │ │ Transaction
Execution → DEX Integration → Staking/Locking │
└──────────────────────────────────────────────────────────────┘ ↓
┌──────────────────────────────────────────────────────────────┐ │ AI AGENT LAYER │ │
Agent 1, 2, 3...N → Each with own strategy │ │ Execute trades → Report performance →
Submit updates │ └──────────────────────────────────────────────┘ ↓
┌──────────────────────────────────────────────────────────────┐ │ PROOF OF INTELLIGENCE
LAYER │ │ Federated Learning → Model Aggregation → Reputation │ │ Store on IPFS →
Consensus Validation → Distribute Rewards │
└──────────────────────────────────────────────────────────────┘
```

## Concrete 2-3 Month Development Roadmap

### Team of 4 - Role Distribution

- **Person 1** - Solana Developer: Programs, token, DEX integration
- **Person 2** - ML Engineer: AI agents, portfolio optimization, PoI consensus
- **Person 3** - Backend Engineer: API, data pipelines, agent orchestration
- **Person 4** - Frontend Developer: dApp UI, wallet integration, dashboard

### Week-by-Week Breakdown

#### Weeks 1-2: Foundation & Setup

**Solana Dev:** Mint VECT.AI SPL Token-2022 on devnet, create agent registry program

**ML Engineer:** Build first portfolio optimization agent (Python), integrate crypto data

**Backend:** Set up infrastructure (AWS/GCP), create API endpoints for program communication

**Frontend:** Create basic UI shell, implement Phantom/Solflare connection

**Deliverable:** Token minted, first agent running locally, basic UI prototype

#### Weeks 3-4: Core Functionality

**Solana Dev:** Build staking/locking hooks, implement agent registration on-chain

**ML Engineer:** Create 3 different agent strategies (momentum, mean reversion, MLbased), add risk management

**Backend:** Build agent orchestration system, implement job queue for periodic rebalances

**Frontend:** Agent marketplace page, portfolio allocation interface

**Deliverable:** 3 agents rebalancing on devnet, users can allocate funds via UI

## Weeks 5-6: Proof of Intelligence v1

**Solana Dev:** Performance tracking PDAs, reward distribution mechanism

**ML Engineer:** Implement federated learning framework, encrypted model update submission

**Backend:** Build model aggregation service, IPFS integration for model storage

**Frontend:** Performance dashboard, agent reputation display

**Deliverable:** Agents learn from each other, reputation system functional

## Weeks 7-8: DEX Integration & Execution

**Solana Dev:** Integrate Orca/Raydium for automated trading (CPI), build transaction management

**ML Engineer:** Add slippage protection, optimize sizing, backtest strategies

**Backend:** Real-time portfolio tracking, transaction confirmation monitoring

**Frontend:** Live trade execution view, transaction history

**Deliverable:** Agents executing real swaps on devnet DEXs

## Weeks 9-10: Security & Testing

**Solana Dev:** Program audit (internal), add scoped pause functionality

**ML Engineer:** Stress test agents with historical data, add circuit breakers

**Backend:** Load testing, error handling, monitoring setup

**Frontend:** Bug fixes, UX improvements, wallet compatibility testing

**Deliverable:** Production-ready MVP on devnet

## Weeks 11-12: Mainnet Launch Prep

**Solana Dev:** Deploy to mainnet-beta, liquidity setup

**ML Engineer:** Train agents on real mainnet data, final parameter tuning

**Backend:** Mainnet infrastructure, backup systems, alert mechanisms

**Frontend:** Mainnet UI, onboarding flow, help documentation

**Deliverable:** VECT • AI live on mainnet-beta with 10-20 agents

## Detailed Technical Stack

### Infrastructure Layer

| Component | Technology | Justification |
|---|---|---|
| Blockchain | Solana (mainnet-beta) | Low fees, fast finality, high throughput |
| Smart Contracts | Rust + Anchor | Secure, productive framework for Solana programs |
| Token Standard | SPL Token-2022 (VECT.AI) | Native token standard with extensions |
| DEX Integration | Orca CLMM + Raydium | Solana-native AMMs with deep liquidity |
| Oracles | Pyth / Switchboard | Reliable Solana price feeds |

### AI Agent Layer

| Component | Technology | Justification |
|---|---|---|
| Agent Framework | Python + LangChain | Modular, extensible, AI-native |
| ML Framework | PyTorch + Scikit-learn | Flexibility + production-ready libraries |
| Portfolio Optimization | PyPortfolioOpt + Riskfolio | Proven quantitative finance tools |
| Market Data | CCXT + Pyth + DEX pools | Multi-source with on-chain verification |
| Backtesting | Backtrader + custom Solana constraints | Event-driven with slippage/latency modeling |

### Backend Infrastructure

| Component | Technology | Justification |
|---|---|---|
| API Layer | FastAPI (Python) | High performance, async, auto-docs |
| Database | PostgreSQL + TimescaleDB | Time-series optimization for market data |
| Cache | Redis | Sub-ms latency for hot data |
| Message Queue | Celery + RabbitMQ | Distributed task execution for agents |
| Storage (Models) | IPFS + Pinata | Decentralized, content-addressed |
| Monitoring | Grafana + Prometheus | Real-time metrics, alerting |

### Frontend Layer

| Component | Technology | Justification |
|---|---|---|
| Framework | React + Next.js | SEO-friendly, fast, large ecosystem |
| Web3 Library | @solana/web3.js + Solana Wallet Adapter | Best wallet UX, multi-wallet support |
| State Management | Zustand | Lightweight, simple API |
| UI Components | Tailwind CSS + shadcn/ui | Modern, customizable, accessible |
| Charts | TradingView Lightweight Charts | Professional trading UI |

## Proof of Intelligence: The Core Innovation

### How It Works (Simplified)

#### Step 1: Agent Performance Tracking

Every agent's trades and outcomes are recorded on-chain:

- Portfolio returns (daily, weekly, monthly)
- Risk metrics (Sharpe ratio, max drawdown, volatility)
- Trade execution quality (slippage, timing)

### Step 2: Model Update Submission

Top-performing agents submit encrypted model updates to IPFS:

- Differential privacy ensures no raw strategy revealed
- Hash stored on-chain for verification
- Stake required to submit (prevents spam)

### Step 3: Consensus & Aggregation

Network validates and aggregates updates:

- Performance-weighted voting (better agents = more influence)
- Outlier detection (Byzantine fault tolerance)
- Global model update computed off-chain, hash on-chain

### Step 4: Reward Distribution

- VECT.AI tokens distributed to contributors:
- Top 20% of agents earn rewards proportional to performance
- Users who staked on winning agents earn share
- Protocol treasury accumulates fees for development

## Why This Matters

```
Traditional AI: Each company trains models in isolation → Duplicate work, repeated
mistakes, no shared learning VECT • AI: Collective intelligence across all agents →
Network effect: more agents = better strategies for everyone → Decentralized: no single
point of failure or control → Incentive-aligned: best performers earn most rewards
```

## MVP Feature Set (2-3 Month Launch)

### Must-Have Features

| Feature | Description | Owner |
|---|---|---|
| VECT.AI SPL Token | Staking, rewards, governance basic functions | Solana Dev |
| Basic PoI v1 | Performance tracking, reputation scoring | ML Engineer + Backend |
| Agent Registration | Developers can deploy new agents on-chain | Solana Dev |

### Nice-to-Have (Post-MVP)

- Advanced PoI with federated learning (Month 4-5)
- Multi-venue routing (Month 5-6)
- Agent-to-agent communication (Month 6+)
- DAO governance for protocol upgrades (Month 6+)
- Mobile app (React Native) (Month 8+)

## Critical Challenges & Mitigation

### Technical Challenges

| Challenge | Risk Level | Mitigation Strategy |
|---|---|---|
| Smart Contract/Program Bugs | HIGH | • Extensive testing on devnet (weeks 9-10)<br>• Strict invariants & audits<br>• Bug bounty program post-launch<br>• Scoped pause mechanism |
| Agent Underperformance | MEDIUM | • Backtest all strategies on 2+ years data<br>• Circuit breakers (auto-pause on drawdown)<br>• Diversification requirements enforced<br>• User can withdraw anytime |
| Liquidity/Slippage | MEDIUM | • Use Orca/Raydium routing<br>• Slippage protection hardcoded<br>• Start with high-liquidity pairs only<br>• Split large orders |
| Scalability (100+ agents) | LOW | • Kubernetes auto-scaling<br>• Async job queue (Celery)<br>• Horizontal scaling architecture |
| User Adoption (“Why trust AI?”) | HIGH | • Public backtesting results<br>• Start with conservative strategies<br>• Transparent performance data on-chain<br>• Community-first approach (Discord/ Twitter) |
| Regulatory Uncertainty | MEDIUM | • Decentralized protocol = no central operator<br>• Users have full custody (non-custodial)<br>• Geo-block restricted regions if needed<br>• Legal consultation before mainnet |

| Challenge | Risk Level | Mitigation Strategy |
|---|---|---|
| Bear Market Impact | MEDIUM | • Market-neutral strategies included<br>• Stable yield options<br>• Protocol fees work in any market<br>• Focus on risk-adjusted returns |
| Competitor with More Capital LOW | LOW | • First-mover advantage in crypto AI agents<br>• Network effects (PoI gets better with users)<br>• Open-source parts = community contributions |

## Team Structure & Daily Execution

### 4-Person Core Team

**Person 1:**
**Solana Developer**

**Skills Required:**
Rust/Anchor, @solana/web3.js, DeFi protocols

**Responsibilities:**
- Program development (token, staking/locking, agent registry)
- DEX integration (Orca, Raydium)
- On-chain performance tracking
- Security auditing and testing

**Week 1 Task:**
Mint VECT.AI SPL token on devnet

**Person 2:**
**ML Engineer**

**Skills Required:**
Python, PyTorch, quantitative finance, trading algorithms

**Responsibilities:**
- Build and train AI agents (10-20 different strategies)
- Portfolio optimization algorithms
- Proof of Intelligence consensus implementation
- Backtesting and performance validation

**Week 1 Task:**
Create first momentum-based agent, integrate market data

**Person 3:**
**Backend Engineer**

**Skills Required:**
Python/FastAPI, PostgreSQL, Redis, AWS/GCP, DevOps

**Responsibilities:**
- API development (REST endpoints for frontend)
- Agent orchestration system (job queue, parallel execution)
- Data pipelines (market data, performance tracking)
- Infrastructure setup and monitoring

**Week 1 Task:**
Set up cloud infrastructure, deploy PostgreSQL + Redis

**Person 4:**
**Frontend Developer**

**Skills Required:**
React, Next.js, Solana Wallet Adapter, Tailwind CSS

**Responsibilities:**
- Build dApp interface (wallet connection, marketplace)
- Performance dashboard and portfolio views
- Agent browsing and allocation UI
- Responsive design, UX optimization

**Week 1 Task:**
Implement wallet adapter connection, create basic UI shell

## Weekly Sync Protocol

### Daily Standups (15 min)

- What did you complete yesterday?
- What are you working on today?
- Any blockers?

### Weekly Sprint Planning (Friday, 1 hour)

- Review completed milestones
- Set next week's priorities
- Demo progress to stakeholders
- Adjust timeline if needed

### Tools

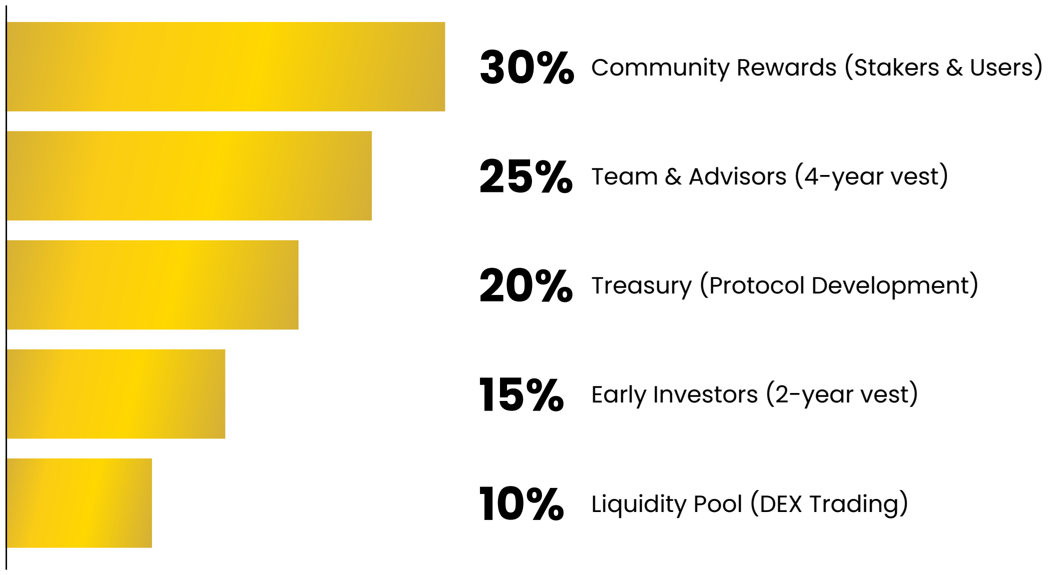
- Communication: Discord or Slack
- Project Management: Linear or Notion
- Code: GitHub with PR reviews
- Deployment: Vercel (frontend) + Railway/Render (backend)

## VECT.AI Token Economics

### Token Utility

| Function | How It Works |
|---|---|
| 1. Agent Staking/Locking | Users stake VECT to allocate funds to top-performing agents |
| 2. Performance Rewards | Agents earn VECT based on returns (top 20% get rewards) |
| 3. Governance | Token holders vote on protocol upgrades, new features |
| 4. Protocol Fees | Performance/management fees to treasury and rewards |
| 5. Agent Registration | Developers stake/lock VECT to deploy new agents (prevents spam) |

### Initial Distribution (Example)



### Value Accrual Mechanism

```
More AUM → More Trading Activity → More Performance Fees Fees Generated: ├─ Rewards to
top agents and stakers ├─ Treasury growth for audits, infra, community └─ Optional
buybacks/burns via governance Result: Token value increases as network grows
```

## Go-to-Market Strategy (Post-Launch)

### Phase 1: Crypto-Native Early Adopters (Month 1-2)

**Target Audience**

- DeFi power users
- Crypto Twitter influencers
- AI/ML researchers in crypto
- Existing robo-advisor users (looking for better returns)

**Tactics**

- Twitter/X campaign with demo videos
- Partnerships with DeFi protocols (cross-promotion)
- Beta tester rewards (earn VECT for early testing)
- Medium articles explaining Proof of Intelligence
- AMA sessions with crypto communities

**Success Metric**

**$1M AUM in first 2 months, 500 active users**

### Phase 2: Broader Crypto Market (Month 3-6)

**Target Audience**

- Retail crypto holders
- Crypto funds/DAOs looking for automated strategies
- Traditional finance professionals entering crypto

**Tactics**

- Listings on token registries
- Integration with wallets
- YouTube content creators
- Hackathon sponsorships
- Press coverage in crypto media

**Success Metric**

**$10M AUM by month 6, 5,000 active users**

### Phase 3: Institutional & Mainstream (Month 6-12)

**Target Audience**

- Crypto hedge funds
- Family offices allocating to crypto
- Retail investors via aggregators

**Tactics**

- Institutional-grade reporting and compliance
- API/SDK for programmatic access
- Partnerships with custody providers
- Academic research papers on PoI performance
- Conference sponsorships

**Success Metric**

**$50M AUM by end of year 1, 20,000 active users**

## Funding Requirements & Use of Funds

### Initial Seed Funding: $500K - $1M

| Category | Amount | % | Details |
|---|---|---|---|
| Team Salaries | $300K | 50% | 4 people × $75K × 6 months (until token launch) |
| Infrastructure | $100K | 17% | Cloud hosting, APIs, market data subscriptions |
| Program Audit | $50K | 8% | External security audit before mainnet |
| Legal & Compliance | $50K | 8% | Token legal opinion, entity setup |
| Liquidity | $50K | 8% | Initial market-making |
| Marketing | $30K | 5% | Content creation, influencer partnerships |
| Buffer | $20K | 3% | Unexpected costs, bug bounties |
| **TOTAL** | **$600K** | **100%** | |

### Alternative: Bootstrap with Token Launch

If we can reduce initial costs and speed up development:

- Launch MVP in 2 months with minimal seed ($200K)
- Public token sale raises $2-5M for scale
- Team gets tokens instead of all cash salary
- Community-funded development (DAO treasury)

**Risk:** Team needs to believe in long-term token value

**Reward:** Much larger upside if VECT succeeds

## Success Metrics & KPIs

### Month 3 (MVP Launch)

| Metric | Target | Measurement |
|---|---|---|
| Active Agents | 10-20 | On-chain agent registry count |
| Total AUM | $500K - $1M | Sum of all user deposits in programs |
| Active Users | 200-500 | Unique wallet addresses with deposits |
| Best Agent Return | >10% monthly | Top agent's 30-day performance |
| Average Return | >5% monthly | Weighted average across all agents |

### Month 6 (Growth Phase)

| Metric | Target | Measurement |
|---|---|---|
| Active Agents | 50-100 | Developers deploying new strategies |
| Total AUM | $5M - $10M | 10x growth from month 3 |
| Active Users | 2,000 - 5,000 | 10x user growth |
| Protocol Revenue | $50K+ | Cumulative performance fees collected |
| VECT Market Cap | $10M - $50M | Fully diluted valuation |

### Month 12 (Scale Phase)

| Metric | Target | Measurement |
|---|---|---|
| Active Agents | 200+ | Diverse strategies from community developers |
| Total AUM | $50M - $100M | Institutional capital starting to flow in |
| Active Users | 15,000 - 25,000 | Mainstream crypto adoption |
| Protocol Revenue | $500K+ | Self-sustaining treasury |
| VECT Market Cap | $100M - $500M | Top 200 crypto by market cap |

## Critical Risks (Honest Assessment)

### CRITICAL RISK #1: Smart Contract Exploit

**Scenario:** Bug in program leads to loss of user funds

**Impact:** Total project failure, reputational damage

**Mitigation:**

- Strict invariants and unit/integration tests
- External audit by reputable firm
- Bug bounty program ($100K+ rewards for finding exploits)
- Scoped emergency pause (withdrawals remain open)

### HIGH RISK #2: AI Agents Lose Money

**Scenario:** Market crash or model failure causes significant drawdowns

**Impact:** Users lose trust, withdraw funds, protocol dies

**Mitigation:**

- Mandatory backtesting on 2+ years of data
- Stress testing on historical crashes
- Circuit breakers: auto-pause agent on large drawdowns
- Diversification enforced (no single asset >30% of portfolio)

### MEDIUM RISK #3: Regulatory Crackdown

**Scenario:** Restrictions in certain jurisdictions

**Impact:** Limited access for some users

**Mitigation:**

- Truly decentralized: no company controls protocol
- Non-custodial: users always control their private keys
- Token as utility
- Geo-blocking at app layer if needed
- Legal opinion from top crypto law firm
- Progressive decentralization: move to DAO

## Conclusion:

### What We're Really Building

**VECT • AI is the infrastructure for autonomous AI agents in finance**

We're not building another robo-advisor.

We're building a **decentralized protocol** where AI agents can:
- Autonomously manage digital assets
- Learn from each other through Proof of Intelligence
- Earn rewards based on performance
- Operate without centralized control

This is **infrastructure,** not a product.
It's **protocol,** not a company.
It's **network effects,** not features.

## Why This Will Work

1. **Timing:** AI + crypto are converging NOW.
2. **Real utility:** People want better returns.
3. **Network effects:** More agents = better strategies = more users = more agents.
4. **Token value accrual:** Clear mechanism: fees ➤ rewards/treasury.
5. **Decentralization:** No single point of failure.
6. **Team execution:** 2-3 months to MVP is achievable with focused team.

### What We Need to Succeed

| Resource | Why Critical |
|---|---|
| 4 Elite Developers | Blockchain + ML + Backend + Frontend expertise. No generalists |
| $500K - $1M Seed | 6 months runway before token launch generates revenue. |
| Laser Focus | No distractions. Build MVP, launch, iterate. Nothing else matters. |
| Community | Early believers who will test, provide feedback, evangelize. |
| Speed | First mover advantage in AI agent asset management is EVERYTHING. |

## Next Steps

**Week 1:** Finalize team, set up infrastructure, start development

**Week 4:** First demo (agents trading on devnet)

**Week 8:** Internal alpha testing

**Week 12:** Public beta launch

**Month 4:** Mainnet-beta launch + token listing

## Let’s Build the Future of Autonomous Finance

VECT • AI is not just another DeFi protocol.

It’s the foundation for a new era where
AI agents manage trillions in assets.

We’re not building a company.
**We’re building a movement.**

**Contact:** team@vect.ai | Twitter: @VECTai | Discord: discord.gg/vectai

## APPENDIX A: Technical Implementation Details

### Smart Contract/Program Architecture

```
// Core Programs (Anchor) 1. VECTToken (SPL Token-2022) - SPL implementation -
Staking/locking integration - Treasury FeeVault PDA 2. AgentRegistry -
Register/deregister agents - Performance tracking on-chain - Reputation scoring 3.
AssetVault (Strategy Vault Program) - User deposits (SPL assets) - Agent allocation
logic - Withdrawal queue - Emergency scoped pause 4. PerformanceFeeManager - Calculate
fees - Distribute to stakers/treasury - High-water mark tracking 5. ProofOfIntelligence
- Model update submission - Consensus/voting hooks - Reward distribution - IPFS hash
storage 6. DexExecutor (CPI) - Orca CLMM integration - Raydium AMM calls - Slippage
protection - Compute budget management
```

### Agent Strategy Examples

```
# Python Agent Examples 1. Momentum Agent - Buy assets with strong 14-day momentum -
Sell when momentum reverses - Risk: 15% stop-loss per position 2. Mean Reversion Agent -
Identify oversold assets (RSI < 30) - Buy dips, sell when normalized - Works well in
range-bound markets 3. ML Prediction Agent - LSTM neural network trained on price/volume
- Predicts next 24h direction - Position sizing based on confidence 4. Arbitrage Agent -
Monitor price differences across DEXs - Execute when spread > fees/impact - High
frequency, low risk 5. Market Neutral Agent - Long undervalued, short overvalued - Beta-
neutral portfolio - Consistent returns in any market 6. Staking Yield Agent - Allocate
to staking derivatives - Auto-compound rewards - Conservative, low-risk option
```

### Proof of Intelligence Algorithm

```
# Simplified PoI Consensus (Pseudocode) def proof_of_intelligence_round(): # Step 1:
Performance Evaluation agents = get_all_active_agents() rankings = [] for agent in
agents: sharpe_ratio = calculate_sharpe(agent) max_drawdown = calculate_drawdown(agent)
consistency = calculate_consistency(agent) score = (sharpe_ratio * 0.5 + (1 -
max_drawdown) * 0.3 + consistency * 0.2) rankings.append((agent, score)) # Step 2:
Select Top 20% Contributors top_performers = sort(rankings)[:len(rankings)//5] # Step 3:
Collect Model Updates updates = [] for agent, score in top_performers: update =
agent.get_encrypted_model_update() ipfs_hash = upload_to_ipfs(update)
updates.append((agent, ipfs_hash, score)) # Step 4: Weighted Aggregation global_model =
aggregate_models(updates, weights=[s for (_,_,s) in updates]) # Step 5: Distribute
Rewards total_rewards = calculate_period_rewards() for agent, score in top_performers:
reward = total_rewards * (score / sum([s for (_,_,s) in updates]))
distribute_vect_tokens(agent, reward) # Step 6: Broadcast Updated Model
broadcast_to_network(global_model) return global_model
```

## APPENDIX B: Example User Flows

### User Flow 1:

#### First-Time User

**Step 1: Connect Wallet**

- User visits app.vect.ai > clicks "Connect Wallet" > Phantom pops up > approves connection

**Step 2: Browse Agents**

- Sees marketplace with 20 agents > filters by "Conservative" risk > sorts by 30-day return

**Step 3: View Agent Details**

- Clicks on "Staking Yield Agent" > sees APY, max drawdown, uptime > reads strategy description

**Step 4: Allocate Funds**

- Clicks "Deposit" > enters amount > approves token spending > confirms transaction > waits for confirmation

**Step 5: Monitor Performance**

- Goes to "Portfolio" page > sees real-time P&L > receives daily updates > can withdraw anytime

### User Flow 2:

#### Developer Deploying New Agent

**Step 1: Develop Strategy**

- Developer writes Python agent using VECT SDK > backtests > achieves target Sharpe/drawdown

**Step 2: Stake/Lock VECT**

- Locks required VECT > minimum commitment period

**Step 3: Register Agent**

- Submits agent metadata to registry program > pays tx fee > agent goes live in "New Agents" section

**Step 4: Attract Users**

- Promotes on Twitter > users try with small amounts > agent performs well > AUM grows

**Step 5: Earn Rewards**

- Agent enters top 20% > earns VECT rewards weekly > can compound or cash out > reputation increases

## APPENDIX C: Competitive Analysis Deep Dive

### Why We're Different from Fetch.ai

| Aspect | Fetch.ai | VECT • AI |
|---|---|---|
| Focus | General-purpose agent framework | Specialized in asset management |
| Use Cases | IoT, supply chain, data sharing, etc. | Digital asset trading exclusively |
| Value Proposition | "Build any autonomous agent" | "Make money with AI agents" |
| Go-to-Market | B2B enterprise sales | Direct to crypto users (B2C + B2B) |
| Revenue Model | Token utility unclear | Performance fees ➤ rewards/ treasury |
| Current Traction | $500M market cap, but few live users | Starting from focus, speed to MVP |

### Why We're Different from SingularityNET

| Aspect | SingularityNET | VECT • AI |
|---|---|---|
| Positioning | AI marketplace (buy/sell AI services) | AI agents that execute autonomously |
| User Experience | Complex, requires AI expertise | Simple: deposit ➤ agent works ➤ withdraw |
| Target User | AI researchers, developers | Anyone with crypto wanting returns |
| Network Effects | More AI services listed | Agents learn from each other (PoI) |
| Moat | Marketplace liquidity | Collective intelligence database |

## Our Unfair Advantages

1. **Vertical Focus:** We only do asset management.
2. **Crypto-Native:** Built for crypto users.
3. **Performance-Driven:** Agents live or die by returns.
4. **Network Effects:** PoI improves with more agents.
5. **Token Economics:** Clear value accrual mechanism.
6. **Speed:** 2-3 months to MVP.

## APPENDIX D: Risk Management Framework

### Agent-Level Risk Controls

| Control | Mechanism | Example |
|---|---|---|
| Position Sizing | Max 30% in any single asset | Agent can't YOLO 100% into one token |
| Stop-Loss | 15% max loss per position | Auto-sell if asset drops 15% from entry |
| Drawdown Limit | 15% portfolio drawdown ➤ pause | Agent stops trading if loses 15% total |
| Leverage | Max 2x (future feature) | No extreme leverage allowed |
| Compute/Priority Fees | Caps per rebalance | Prevent runaway transactions |
| Whitelisted Assets | Only approved assets | No illiquid tokens |

### Protocol-Level Risk Controls

| Control | Mechanism | Example |
|---|---|---|
| Emergency Pause | Scoped per program | If exploit detected, freeze execution |
| Withdrawal Queue | Delay for large withdrawals | Prevents bank run, gives time to respond |
| Agent Quarantine | Suspend agents with anomalous behavior | Agent making weird trades ➤ autopaused |
| Insurance Fund | Treasury allocation | Cover losses if program bug |
| Rate Limiting | Max trades/hour per agent | Prevents runaway loops |
| Oracle Verification | Pyth/Switchboard checks | Detect manipulation attempts |

### User Protection Measures

- **Risk Disclosure:** Clear warnings that agents can lose money
- **Simulation Mode:** Paper trading for 30 days before real money
- **Diversification Recommendations:** UI suggests spreading across 5+ agents
- **Performance Disclaimers:** "Past returns ≠ future results" on every page
- **Withdraw Anytime:** No lock-up periods (except queueing)
- **Transparent Reporting:** All trades visible on-chain

## APPENDIX E: 90-Day Detailed Task Breakdown

### Month 1: Foundation

| Week | Solana Dev | ML Engineer | Backend Engineer | Frontend Developer |
|---|---|---|---|---|
| W1 | • Mint SPL token<br>• Anchor setup<br>• Write tests | • Build momentum agent<br>• Integrate market data<br>• Backtest framework | • AWS setup<br>• PostgreSQL + Redis<br>• FastAPI skeleton | • Next.js project<br>• Wallet adapter integration<br>• Basic UI shell |
| W2 | • AgentRegistry program<br>• Registration logic<br>• Event emissions | • Mean reversion agent<br>• Risk management module<br>• Stop-loss logic | • API endpoints (agents)<br>• Market data pipeline<br>• Celery setup | • Wallet UI<br>• Agent list page<br>• Tailwind styling |
| W3 | • AssetVault program<br>• Deposit/withdraw logic<br>• Allocation tracking | • ML prediction agent<br>• LSTM model training<br>• Feature engineering | • Agent orchestration<br>• Job queue system<br>• Monitoring setup | • Agent detail pages<br>• Performance charts<br>• Allocation form |
| W4 | • Staking/locking hooks<br>• Reward calculation<br>• Devnet deployment | • Arbitrage/route agent<br>• DEX price comparison<br>• Slippage modeling | • Transaction tracking<br>• Portfolio calculations<br>• Alert system | • Portfolio dashboard<br>• Real-time updates<br>• Bug fixes |

### Month 2: Core Features

| Week | Solana Dev | ML Engineer | Backend Engineer | Frontend Developer |
|---|---|---|---|---|
| W5 | • PerformanceFee program<br>• Fee calculation<br>• Distribution logic | • Market neutral agent<br>• Long/short logic<br>• Beta neutrality | • IPFS integration<br>• Model storage<br>• Retrieval API | • Agent comparison tool<br>• Filter/sort features<br>• Mobile responsive |
| W6 | • PoI v1 hooks<br>• Voting mechanism<br>• Reputation scoring | • Staking yield agent<br>• Derivative routes<br>• Queue handling | • Aggregation service<br>• Consensus logic<br>• Reward distribution | • Reputation display<br>• PoI visualization<br>• Educational tooltips |

| | | | | |
|---|---|---|---|---|
| W7 | • DEXExecutor CPI<br>• Orca CLMM integration<br>• Raydium AMM | • Expand to 10 agents<br>• Strategy diversification<br>• Parameter tuning | • DEX adapters<br>• Pool data indexing<br>• Price sanity checks | • Trade execution UI<br>• Transaction status<br>• History view |
| W8 | • Integration testing<br>• CU/fee optimization<br>• Security review | • Stress testing agents<br>• Historical backtests<br>• Edge case handling | • Load testing<br>• Error handling<br>• Performance optimization | • UX improvements<br>• Loading states<br>• Error messages |

## Month 3: Launch Preparation

| Week | Solana Dev | ML Engineer | Backend Engineer | Frontend Developer |
|---|---|---|---|---|
| W9 | • Internal audit<br>• Scoped pause testing<br>• Upgrade mechanism | • Circuit breaker testing<br>• Drawdown scenarios<br>• Final parameter tuning | • Monitoring dashboard<br>• Alert configuration<br>• Backup systems | • Beta testing<br>• Bug fixes<br>• Accessibility improvements |
| W10 | • External audit prep<br>• Documentation<br>• Multisig setup | • Expand to 15-20 agents<br>• Community agent submissions<br>• Quality control | • Production infrastructure<br>• CI/CD pipelines<br>• Security hardening | • Onboarding flow<br>• Help documentation<br>• Video tutorials |
| W11 | • Mainnet-beta deployment<br>• Liquidity setup<br>• Token metadata | • Mainnet validation<br>• Real data testing<br>• Performance monitoring | • Mainnet backend deployment<br>• Monitoring setup<br>• Incident response plan | • Mainnet UI updates<br>• Analytics integration<br>• SEO optimization |
| W12 | • **LAUNCH**<br>• Monitor programs<br>• Support users | • **AGENTS LIVE**<br>• Real-time monitoring<br>• Quick fixes if needed | • **INFRASTRUCTURE LIVE**<br>• 24/7 monitoring<br>• Performance optimization | • **UI LIVE**<br>• User support<br>• Collect feedback |

## Final Thoughts: This is the Real VECT • AI

### What Makes This Different

This isn’t a business plan for a traditional company.

It’s a **technical roadmap for a decentralized protocol.**

We’re not seeking permission from regulators.

We’re not building for one country’s market.

We’re not creating another centralized service.

**We’re building infrastructure for the future of autonomous finance.**

AI agents will manage trillions of dollars in the next decade.

VECT • AI will be the protocol they run on.

## The Path Forward

### Immediate (Next 2 Weeks)

- Finalize 4-person core team
- Secure $500K - $1M seed funding
- Set up legal entity (Foundation)
- Kick off development (Week 1 tasks)

### Short-Term (Month 1-3)

- Build MVP (follow 12-week roadmap)
- Internal alpha testing
- Program audit
- Community building (Discord, Twitter)

### Medium-Term (Month 4-6)

- Mainnet-beta launch
- Public token listing
- Grow to $10M AUM
- Expand agent ecosystem (100+ agents)

### Long-Term (Year 1+)

- Scale to $50M - $100M AUM
- Multi-venue expansion
- Agent-to-agent marketplace
- Full DAO governance

## Ready to Build?

This document outlines exactly what we're building, how we'll build it, and why it will succeed.

The team is defined.
The timeline is clear.
The technology is proven.

**Now we just need to execute.**

**Let's make VECT • AI a reality.**