**AI Agent Architecture and Development Plan for the 1000 Project**

**Overview**  
The AI agent for the 1000 Project acts as the central intelligence of the ecosystem, automating key processes, enhancing transparency, and fostering community engagement. It is designed with modularity, scalability, and seamless blockchain integration in mind, ensuring compatibility with smart contracts and user interfaces.

**Core Functions**

1. **Reward and Burn Automation:**
   * Automates daily alternating reward and burn cycles.
   * Interfaces directly with the Reward and Burn Contract for execution.
2. **Wallet Analytics:**
   * Tracks wallet balances, transaction histories, and cooldown statuses.
   * Verifies eligibility for rewards based on holding patterns and cooldown periods.
3. **Governance Management:**
   * Facilitates community proposal submissions and voting processes.
   * Delivers real-time updates on voting outcomes and governance statistics.
4. **Community Engagement:**
   * Hosts interactive trivia events, polls, and governance discussions.
   * Provides notifications for rewards, cooldown periods, and governance updates through Telegram/Discord bots.

**Proposed Architecture**

1. **Data Module:**
   * **Purpose:** Collect and process blockchain and external API data.
   * **Sources:**
     + Blockchain explorers (e.g., Shidoscan, Etherscan) for wallet data.
     + Chainlink VRF for randomness.
   * **Output:** Provides actionable insights for logic execution, such as eligible wallets and transaction timestamps.
2. **Logic Module:**
   * **Purpose:** Implements reward, burn, and cooldown functionalities.
   * **Features:**
     + Calculates rewards and burns based on percentages and wallet activity.
     + Enforces dynamic cooldowns using blockchain data.
3. **Engagement Module:**
   * **Purpose:** Manages user-facing interactions and community events.
   * **Features:**
     + Integrates with Telegram/Discord APIs for notifications and updates.
     + Organizes trivia, polls, and governance events to drive engagement.
4. **Governance Module:**
   * **Purpose:** Oversees community-driven decision-making processes.
   * **Features:**
     + Facilitates proposal creation and voting mechanisms.
     + Logs all votes and outcomes transparently on-chain.
5. **Middleware:**
   * **Purpose:** Acts as a bridge between blockchain data, AI logic, and user interfaces.
   * **Tools:** Python or Node.js for API connections, data processing, and AI integration.

**Technical Requirements**

1. **Blockchain Integration:**
   * Smart contract interactions for reward/burn execution and wallet data.
   * Integration with Chainlink VRF for random wallet selection.
2. **APIs:**
   * Telegram/Discord APIs for notifications and engagement tools.
   * Blockchain explorers or Shido-specific APIs for real-time data collection.
3. **AI/ML Frameworks:**
   * Lightweight AI models for wallet activity analysis (e.g., TensorFlow, PyTorch).
   * Rule-based logic for reward and cooldown validation.
4. **Database:**
   * Stores snapshot data for wallets, transaction logs, and governance proposals.
   * Preferred solutions: PostgreSQL or MongoDB for scalability.

**Development Milestones**

1. **Phase 1: Foundation**
   * Define smart contract interaction endpoints.
   * Build the Data and Logic Modules to automate rewards and burns.
2. **Phase 2: Community Engagement**
   * Integrate Telegram and Discord bots for notifications.
   * Develop the Engagement Module with initial trivia and polls.
3. **Phase 3: Governance Integration**
   * Deploy the Governance Module for proposal creation and voting.
   * Implement real-time governance tracking.
4. **Phase 4: AI Integration**
   * Implement wallet activity analytics for advanced eligibility verification.
   * Optimize reward and burn percentages with AI-driven algorithms.
5. **Phase 5: Full Deployment**
   * Conduct comprehensive testing on the Shido testnet.
   * Deploy the AI agent and associated smart contracts to the mainnet.

**Next Steps**

1. Review the architecture and development plan with the technical team.
2. Finalize the tools, frameworks, and libraries for development.
3. Begin development of the Data and Logic Modules as the foundation of the AI agent.