aevatar.ai

A Pioneer of the Next-Gen Multi-Agent AI Framework

Release Date: February 07, 2025
Version: V0.1

Table of Contents

Scope: This white paper is intended for users, developers, and potential partners interested in multi-agent platforms. It provides a thorough overview of aevatar.ai's design philosophy, technical framework, and typical use cases.

Abstract

This white paper presents **aevatar.ai**, a unified multi-agent platform that addresses the complexities inherent in developing, deploying, and managing diverse AI agents across varied domains and workloads. Leveraging a **plugin-based approach** and **flexible deployment strategies**—ranging from DLL-based loading to containerised and distributed actor frameworks—**aevatar.ai** allows users and developers to seamlessly integrate specialised AI solutions under a single ecosystem.

Key components include the **aevatar Framework**, which defines standardised agent interfaces and lifecycle management; **aevatar Station**, a centralised portal and marketplace for agent discovery, plugin handling, request routing, and user access control; and **aevatar Agents**, a repository of official and community-developed AI modules supporting an array of tasks, such as natural language understanding, computer vision, and recommendation systems. By centralising agent interactions and event flows, aevatar.ai reduces integration overheads, enforces consistent security policies, and provides robust monitoring and logging for higher reliability.

Through its **open-source, modular architecture**, **aevatar.ai** caters to both small-scale experiments and large enterprise deployments, supporting features like **high concurrency**, **auto-scaling**, **agent reusable**, **sandboxing**, and **audit trails**. These innovations foster a sustainable AI ecosystem in which organisations can rapidly adopt and evolve advanced AI capabilities, while developers focus on creating powerful, specialised agents without the complexities of infrastructure and lifecycle management.

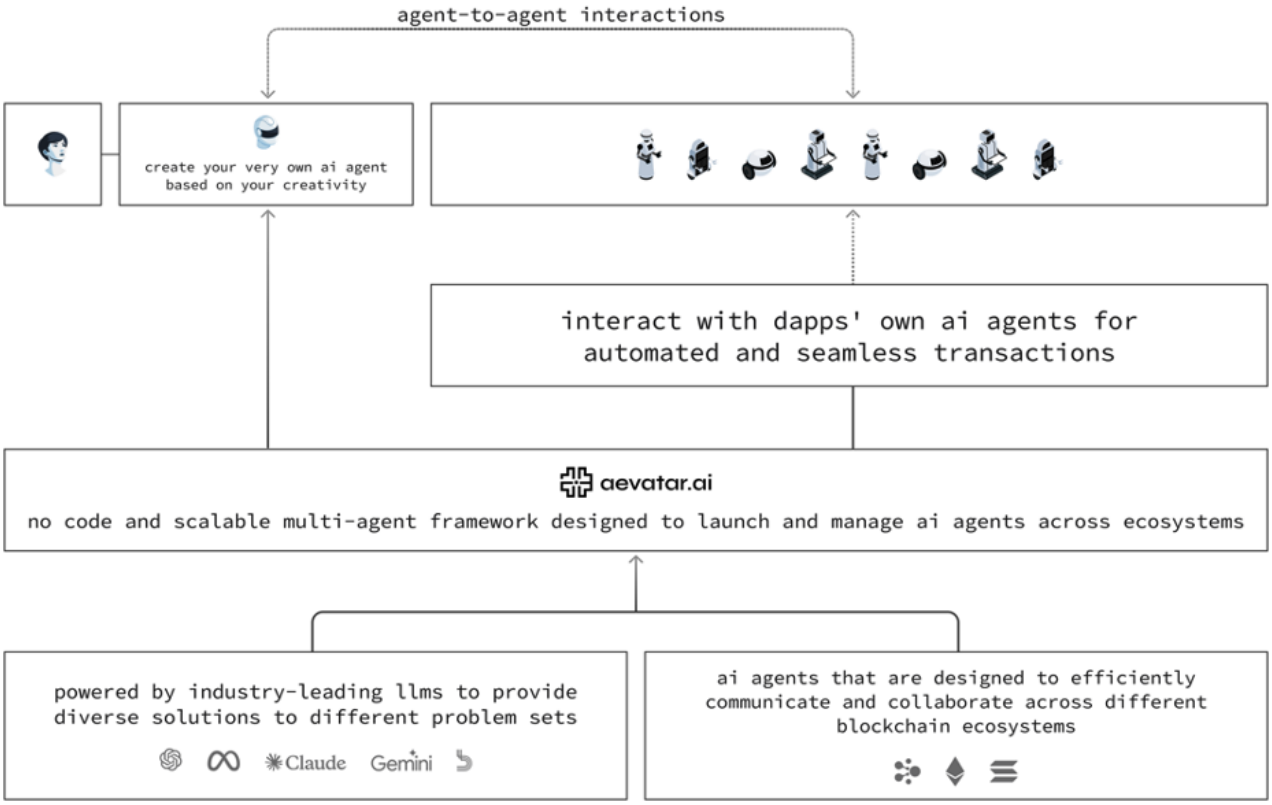
1. Introduction

With the rapid development of artificial intelligence (AI) technology, the applications of Large Language Models (LLMs) and intelligent agents have evolved from single question-answer interactions to higher-level capabilities such as multi-agent collaboration, cross-model cooperation, and complex business process orchestration.

However, current AI systems on the market generally face issues such as platform isolation, model limitations, complex deployment, and lack of observability, making it difficult to meet enterprise users' demands for efficient, flexible, and secure AI collaboration.

aevatar.ai, **a pioneer of the next-gen multi-agent AI framework**, aims to build a cross-platform, cross-model AI agent ecosystem. Through open architecture, powerful visual orchestration capabilities, and cloud-native deployment methods, it empowers developers and business users to uniformly manage, schedule, and coordinate multiple intelligent agents within a single system, achieving efficient collaboration across 'multiple scenarios, multiple models, and multiple roles'.

With aevatar.ai, we are committed to providing a flexible, scalable, and security-compliant AI solution to promote the widespread application and implementation of AI technology.



Key Points:

1. **Agent-to-Agent Interactions**: Enables seamless collaboration between AI agents to automate workflows.
2. **Scalable Framework**: **aevatar.ai** provides a modular and scalable infrastructure for launching and managing AI agents effortlessly.
3. **Custom Agents**: Users can design and deploy personalised agents without coding, tailored to specific business needs.
4. **Industry-Leading LLMs**: Powered by top language models like OpenAI, Deepseek, Claude, and Gemini, it offers diverse problem-solving capabilities.
5. **Industry-Specific Integrations**: aevatar.ai empowers businesses in all industries, like finance, manufacturing, and beyond to simplify workflows, automate complex tasks, and achieve cross-ecosystem interoperability with minimal friction. it combines scalability, flexibility, and cutting-edge AI to drive innovation across industries.

6. **Blockchain Interoperability**: Supports AI agents to communicate across multiple blockchain ecosystems.

2. Background & Challenges

2.1 AI Agent System Isolation

Currently, many AI agents are isolated within their respective platforms, lacking unified communication protocols and interoperability. This lack of interoperability limits the overall effectiveness of AI systems, especially in scenarios requiring cross-platform collaboration.

2.2 Limitations of Single LLM

Most AI agents rely on a single language model (such as GPT-4 or Llama2). This presents a concentration risk; performance can be compromised when facing complex multi-step tasks or multilingual scenarios.

The limitations of a single LLM model prevent systems from flexible switching or parallel usage of multiple models, thus restricting their application scope and performance.

2.3 Insufficient Retrieval-Augmented Generation (RAG) Accuracy

Most AI agents use retrieval-augmented generation to retrieve information from a specialised knowledge base, but it is challenging to achieve perfect knowledge base refinement and accuracy as the information and documents might be irrelevant, outdated, or low-quality.

2.4 Lack of Event Tracing and Observability

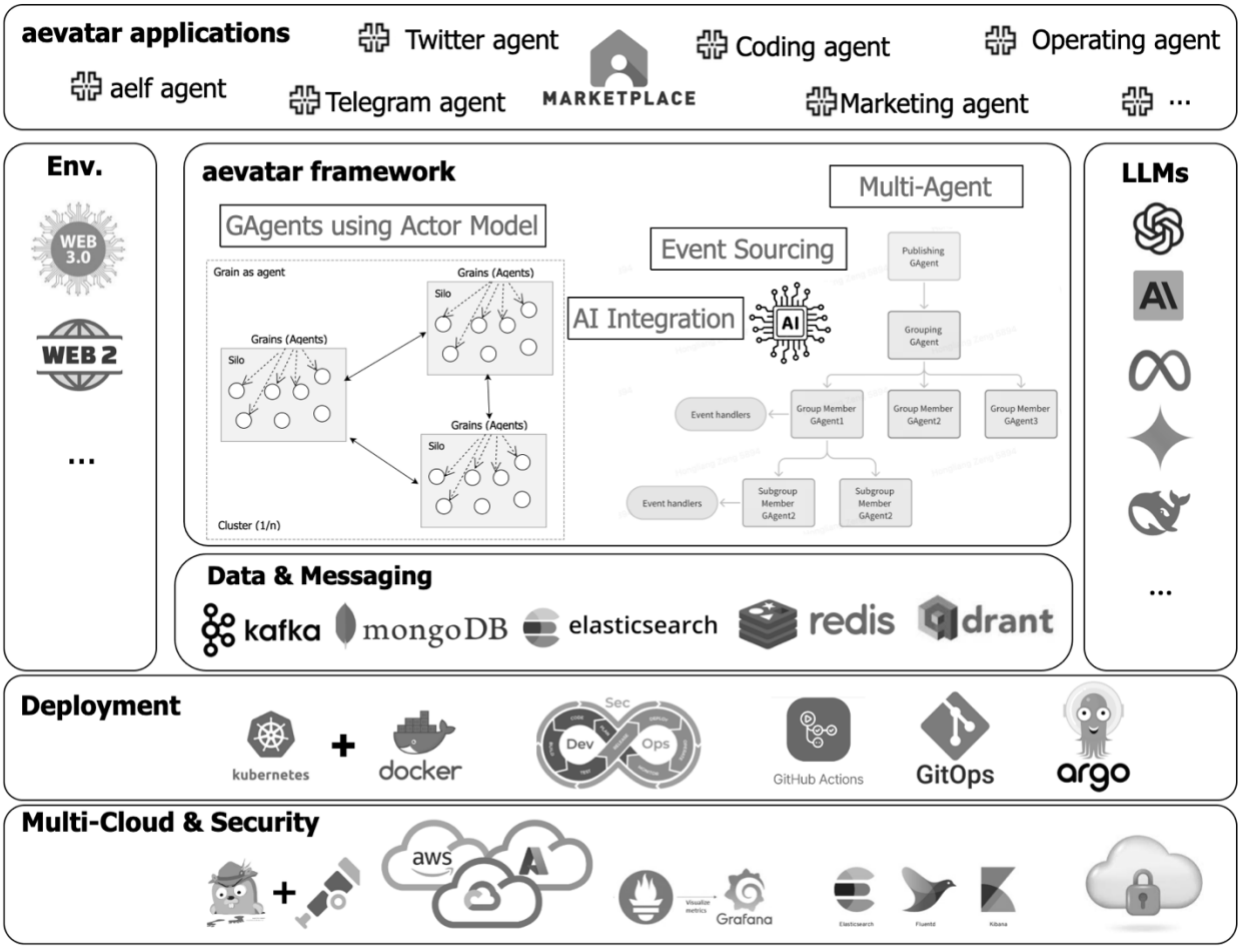
Existing AI systems often lack source management for AI agent internal states and historical interactions. When system failures or reasoning anomalies occur, it is difficult to locate problems and replay events, increasing maintenance complexity and risks.

2.5 High Deployment and Collaboration Costs

Traditional AI systems typically require complex installation, configuration, and maintenance processes. The lack of user-friendly workflow orchestration tools, especially in multi-agent collaboration scenarios, leads to high development and maintenance costs.

3. Vision & Goals

aevatar.ai is designed to support the development and deployment of AI agents while resolving the above challenges. It leverages a combination of open architecture, powerful orchestration capabilities, and cloud-native deployment to create a robust, scalable, and efficient system.



3.1 aevatar Framework

aevatar framework represents the core of aevatar.ai, and enables it to handle essential processing logic, agent interactions, and core components.

Orleans "Grains" as Agents

- Orleans uses the concept of grains—lightweight, isolated micro-objects—to represent actors or "agents."
- Clusters of silos (the runtime hosts in Orleans) coordinate these grains so they can be distributed and scaled across many servers.
- In this architecture, each grain is effectively an agent (often shown as "GAgent").

Multi-Agent

- The diagram shows multi-layered agent groupings. For instance, a "Publishing GAgent" coordinate several "Group Member GAgent" instances.
- Event handlers manage asynchronous triggers or state changes, enabling agents to respond in real-time to inbound data or updates from other agents.

AI Integration

- Semantic Kernel provides advanced AI orchestration and prompt-chaining capabilities.

By combining all the above, the system can scale out large numbers of AI agents, each with specialised tasks, while also coordinating them in groups or sub-groups to accomplish more complex, collaborative goals.

3.2 aevatar Applications

- **Marketplace**: A centralised platform where various AI agents can be discovered, developed, managed, and deployed.
- **Agents**: Individual AI agents that perform specific tasks or functions. These agents can be developed and deployed independently.
- **Webhook**: aevatar can seamlessly orchestrate a large number of external inputs, converting real-world triggers into structured events that G-agents can handle, thus enabling continuous, real-time interaction with a variety of external systems.

Example Agents:

- **Twitter Agent**: Monitors tweets, posts updates, or interacts with Twitter.
- **Telegram Agent**: Works with Telegram for chat interactions.
- **Coding Agent**: Helps generate or review code.
- **Marketing Agent**: Performs marketing tasks such as campaign management.
- **Operating Agent**: Handles operational tasks.
- **aelf Agent**: Leverages aelf on-chain, off-chain data to perform automated tasks.

These represent end-user-facing “products” built on top of the underlying multi-agent framework. Each of these agents can have specialised logic, connect to external APIs, and leverage the core aevatar engine.

3.3 Environment (Web 2 / Web 3)

This indicates the broader context in which aevatar agents operate—both in traditional Web 2.0 environments (e.g., REST APIs, SaaS services) and Web 3.0 contexts (e.g., blockchain or decentralised services). The framework is designed to plug into these ecosystems seamlessly.

3.4 LLMs Integration

On the right side of the diagram, you see major LLM (Large Language Model) providers:

- **OpenAI / ChatGPT**
- **Anthropic**
- **Meta**
- **Azure OpenAI**
- **Deepseek**
- And more…

These LLMs are integrated through the *Semantic Kernel* connectors, so each agent can leverage natural language understanding, generation, and advanced reasoning.

3.5 Data & Messaging

Above the framework, we see core data and messaging technologies:

- **Kafka**: Real-time messaging and event streaming
- **MongoDB**: Document-based or general data storage
- **Elasticsearch**: Full-text search and analytics at scale
- **Redis**: In-memory data store for caching and high-speed access
- **Qdrant**: Referring to specialised vector stores

These technologies enable high-throughput data ingestion, search, caching, and state management—critical for large-scale agent interactions.

3.6 Deployment & DevSecOps

DevSecOps used to build, deploy, and manage the aevatar framework:

- **Kubernetes** + **Docker**: Containerisation and orchestration across clusters.
- **GitHub Actions**, **GitOps**, **Argo**: CI/CD pipelines and “GitOps” - style deployment for automated, versioned releases.

- The “DevSecOps” loop highlights security-focused continuous integration/continuous deployment practices.

3.7 Multi-Cloud & Security

Finally, there is a multi-cloud strategy across:

- **Google Cloud Platform, Amazon Web Services, Microsoft's Azure** and others – Cloud providers are supported for deployment.
- Additional tooling for security and observability, such as **Grafana** (monitoring dashboards), **Vault** (secrets management), **Elasticsearch**/**Fluentd**/**Kibana** (EFK stack for logs and analytics), and so on.

This ensures the platform can run in a secure, fault-tolerant, and cost-efficient way across different cloud infrastructures.

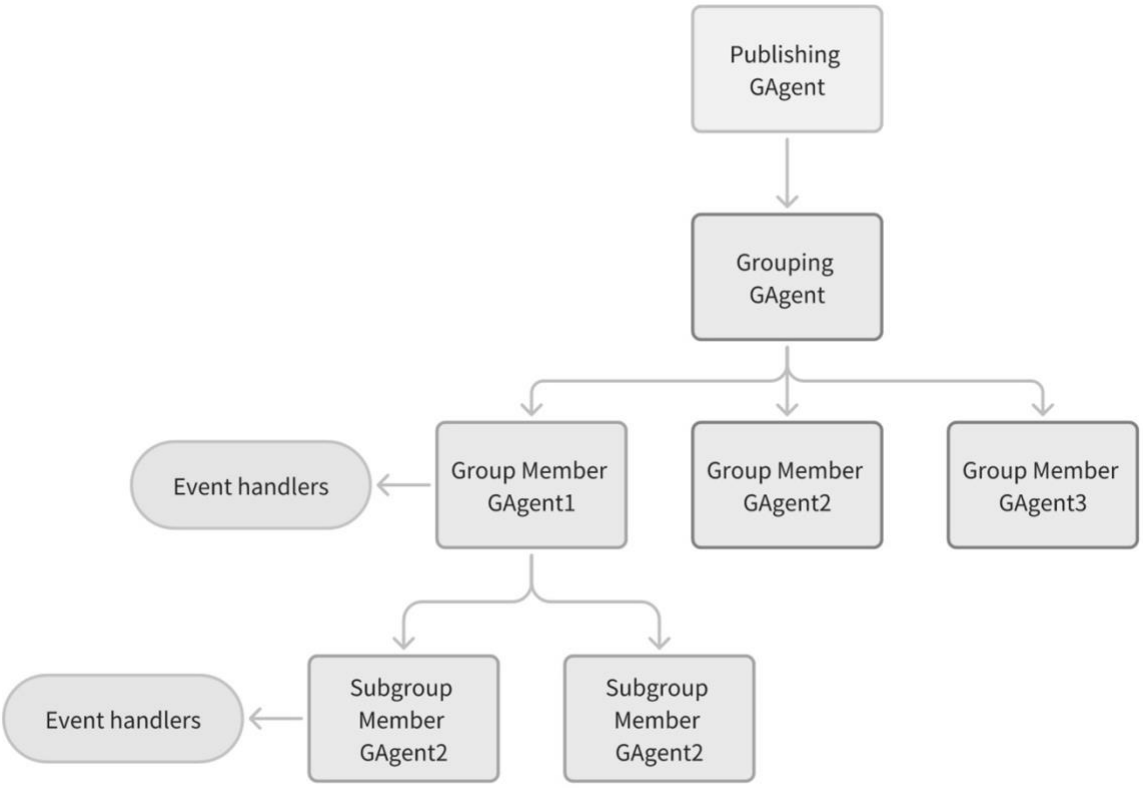
3.8 Putting It All Together

- Each aevatar application (like a Twitter Agent or Coding Agent) is an Orleans “grain” (or set of grains) wrapped in specialised logic.
- The Multi-Agent or “grouping” approach coordinates large collections of these grains, allowing them to pass events/messages among themselves through Kafka, Redis, or direct Orleans messaging.
- Semantic Kernel helps orchestrate more advanced AI reasoning, prompt chaining, and memory/knowledge.
- The entire setup is packaged for cloud deployment (Kubernetes + Docker) and integrated with logging, security, and monitoring solutions (Grafana, Vault, EFK).
- This combination provides a scalable, fault-tolerant, and highly extensible platform to run AI agents across multiple domains—Web 2 and Web 3, on multiple clouds, with robust security and observability.

In short, *aevatar.ai* is a full-stack, cloud-native, multi-agent orchestration framework that leverages **Orleans** for actor-based scaling, integrates with **Semantic Kernel** for AI functionality, and utilises a comprehensive **DevSecOps** pipeline plus multi-cloud deployment strategy.

3.9 aevatar Advantages

1. **Multi-Agent Collaboration**



GAgent: grain-based Agent

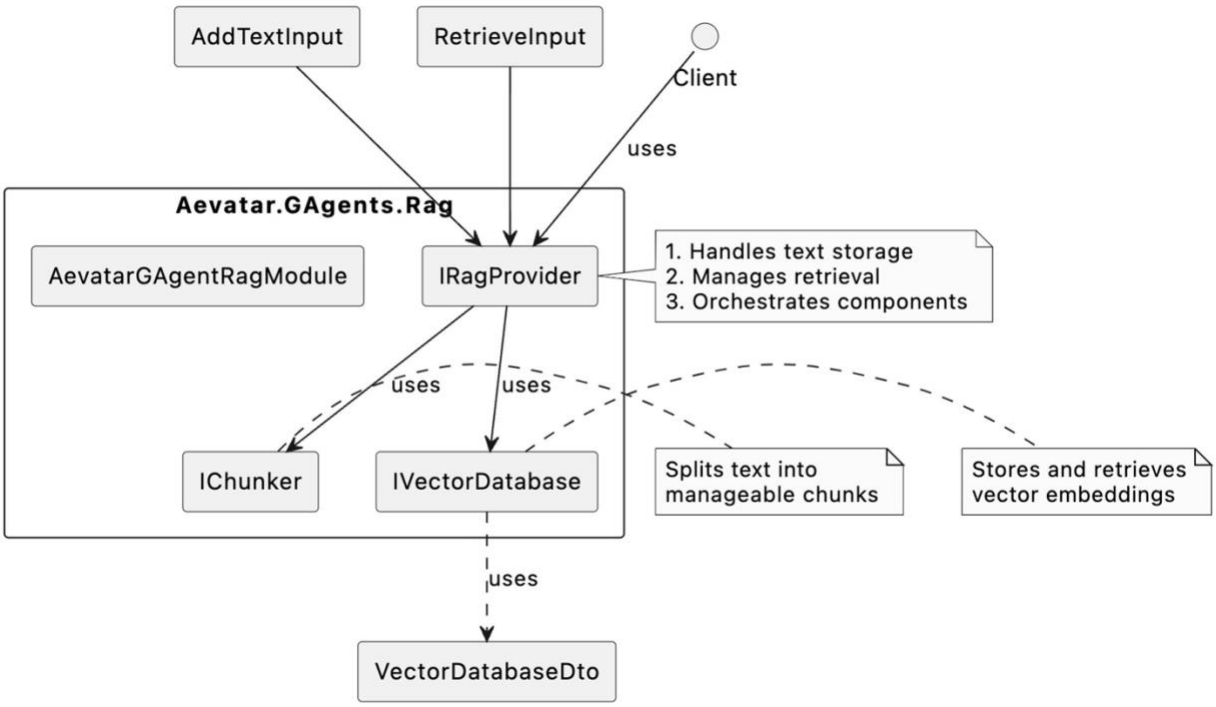
Through a distributed actor model (based on Orleans) and multi-agent management mechanism, aevatar.ai achieves efficient interconnection and complex event scheduling between multiple AI agents, supporting cross-platform and cross-scenario collaborative workflows.

aevatar.ai's multi-agent framework divides AI agents into different functional roles. It assigns specific responsibilities and groups multiple agents to ply various niches within a system, thereby completing user-assigned tasks from a systemic perspective.

2. Unified Cross-Model Collaboration

aevatar.ai provides a multi-language model parallel agent framework. This overcomes the limitations of single models and supports free switching or parallel use of multiple models in different tasks, thereby enhancing system flexibility and performance.

3. Multi-Agent RAG Architecture



Under the multi-agent RAG architecture, each AI agent represents a customised RAG based on specific knowledge bases, retrieval strategies, and generation configurations; this provides answers in the entire system's most proficient domain.

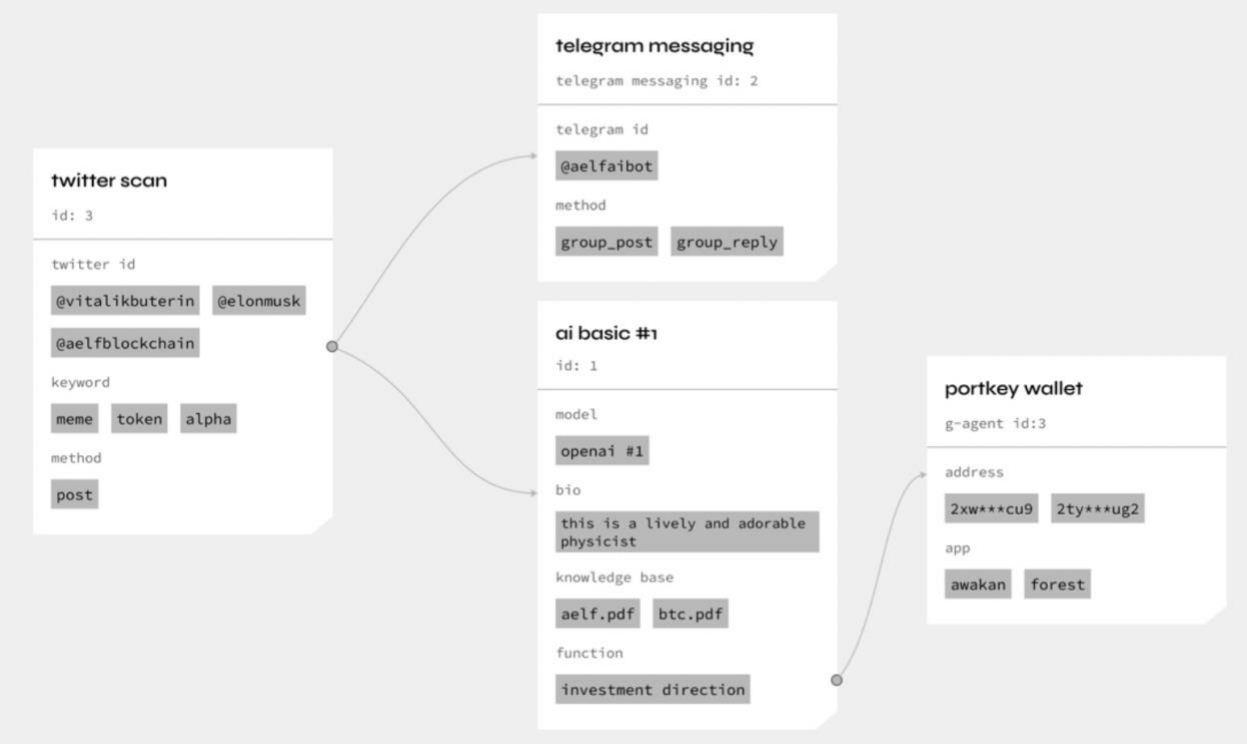
Through the orchestrator, user questions are allocated to appropriate agents. Multiple agents can also be called in parallel, and answers are consolidated through the information integration module. This achieves a more professional, comprehensive, and scalable question-answering or information-generation system.

The multi-agent RAG model enables:

1. **Flexible expansion**: Quick deployment of new agents based on different business lines or knowledge domains.
2. **Noise reduction**: Utilising domain-specific knowledge bases to reduce irrelevant information interference.
3. **Enhanced credibility**: Cross-verification between multiple agents.
4. **Sustainability**: Independent maintenance of each agent's knowledge base, facilitating a divide-and-conquer approach.

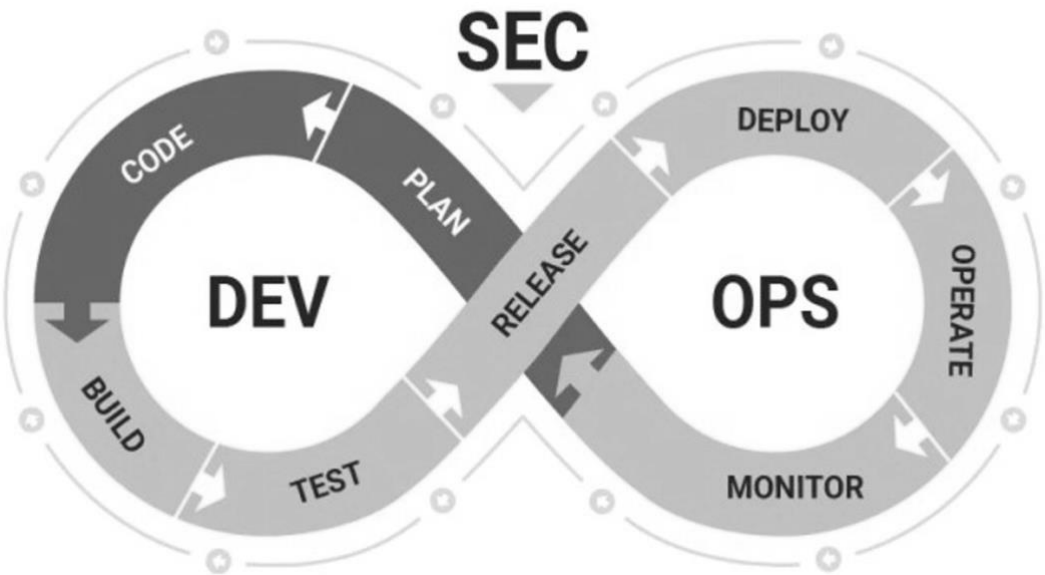
This enables the building of a multi-agent RAG platform capable of consistently producing high-quality content.

4. Visualisation and Ease of Use



The aevatar.ai dashboard provides low-code/no-code visual orchestration tools, helping users easily design and monitor complex workflows. This is a significant step in lowering technical barriers so that just about anyone can quickly get started in creating and personalising AI agents.

5. Security and Scalability



Based on the cloud-native DevSecOps and microservice architecture, aevatar.ai provides elastic scaling and high concurrency processing capabilities, while ensuring system security and compliance.

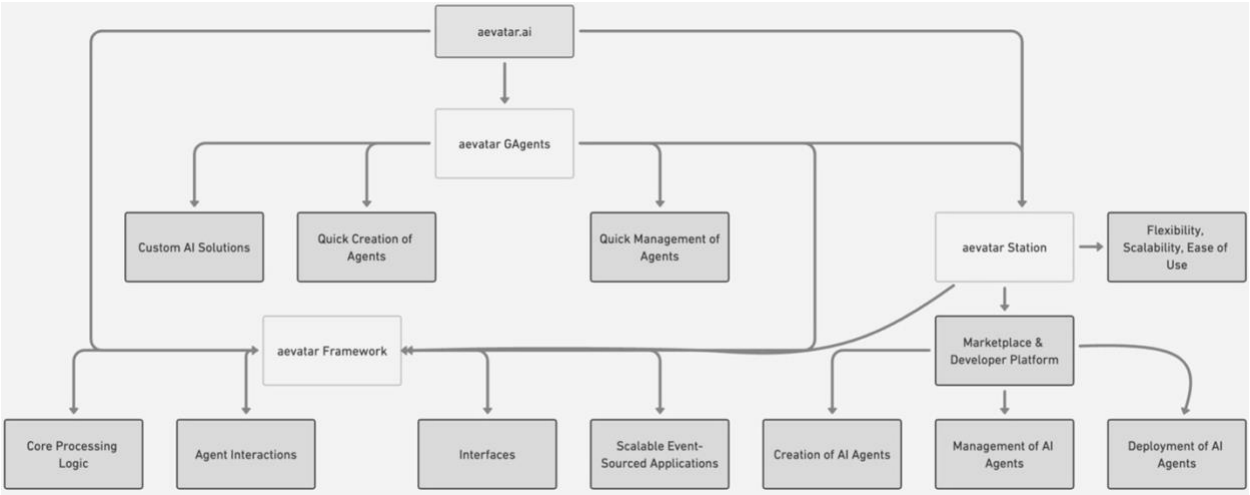
With that, it meets enterprise-level user requirements.

4. Architecture

aevatar.ai comprises three primary components:

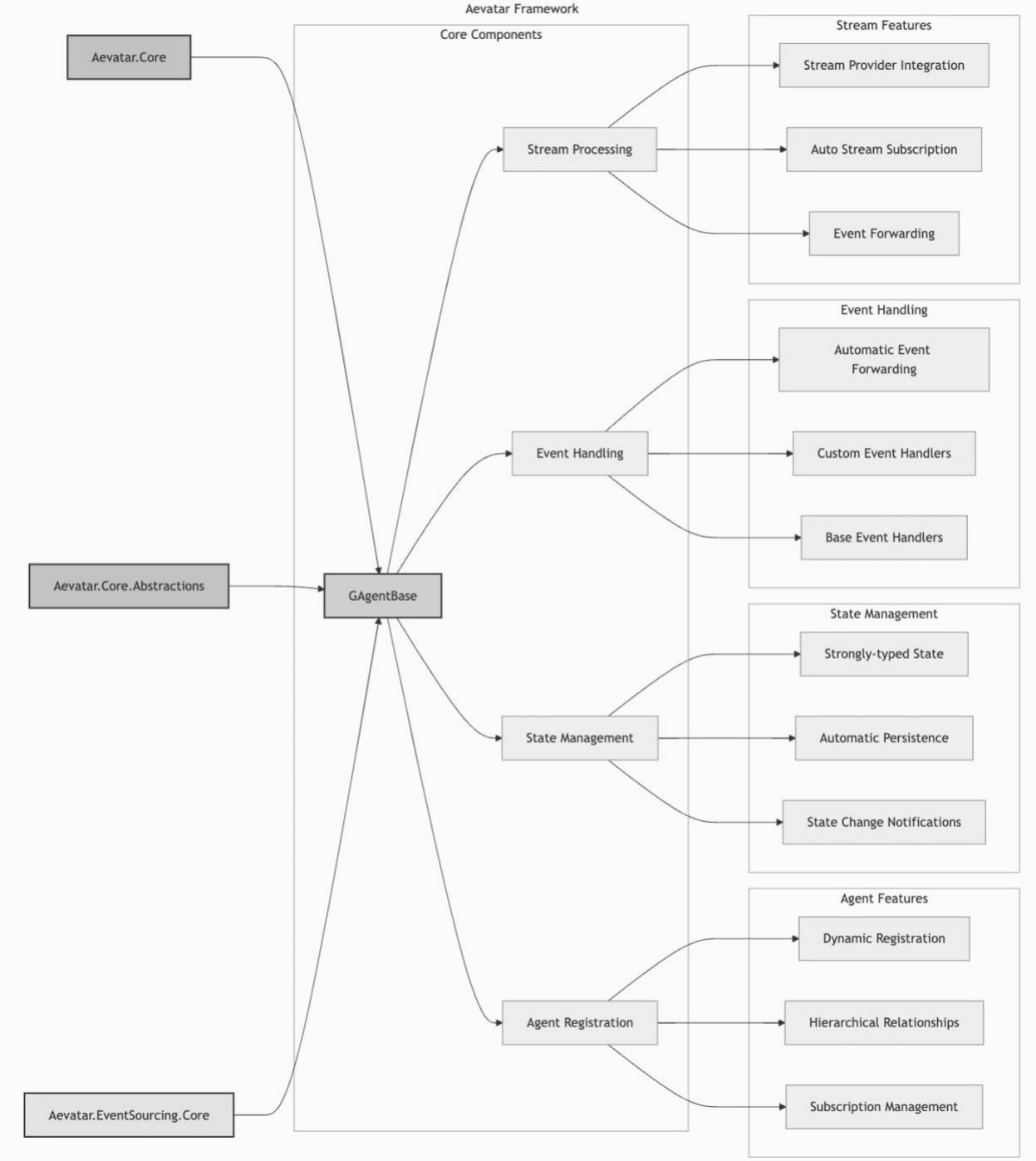
- aevatar Framework
- aevatar Station
- aevatar Agents

They synergies to manage the entire lifecycle of multiple AI agents—from creation to deployment and ongoing operation.



4.1 aevatar Framework

aevatar Framework is designed to support AI agents and event sourcing mechanisms, providing a modular architecture for extensibility and maintainability. It leverages design patterns such as Dependency Injection and Observer Pattern to enhance flexibility and scalability.



Design Principles

- Modularity: The framework is designed to be modular, allowing developers to add or remove components as needed.
- Extensibility: New features can be added through plugins without altering the core framework.

- Separation of Concerns: Each component has a specific responsibility, promoting maintainability and readability.

The framework provides a flexible architecture for developing AI agents and event sourcing applications, allowing for easy integration and extension through its modular design. By adhering to design principles and patterns, the framework ensures that it remains scalable and maintainable as new features are added.

4.2 Core Components Overview

Actor Model

- Responsible for managing distributed actor (Grain) lifecycle and communication, providing a stateful and replayable execution environment for each agent, ensuring high concurrency and scalability for the system.

GAgent

- Each submodule (such as Telegram, Twitter, MicroAI, SocialAgent, etc.) is an independent GAgent, implementing specific agent logic for different platforms or scenarios, supporting cross-platform expansion.

Event Sourcing

- Provides core functionality for log storage, event replay, and snapshot management. This supports multiple backend storage options like MongoDB and Redis, and ensures system traceability and audit capabilities.
- All critical agent events (i.e. received messages, state updates, model inference outputs) can be persisted, providing replay and audit capabilities.

CQRS (Command Query Responsibility Segregation)

- Externally provides REST/gRPC interfaces and supports efficient internal data querying and indexing through read-write separation architecture. Combined with solutions like Elasticsearch, it enables fast retrieval of large-scale data.
- **Read-write separation**: The system can handle agent state change write requests (events) independently from external query interfaces.
- Combined with Elasticsearch/MongoDB for rapid retrieval and multi-dimensional queries.

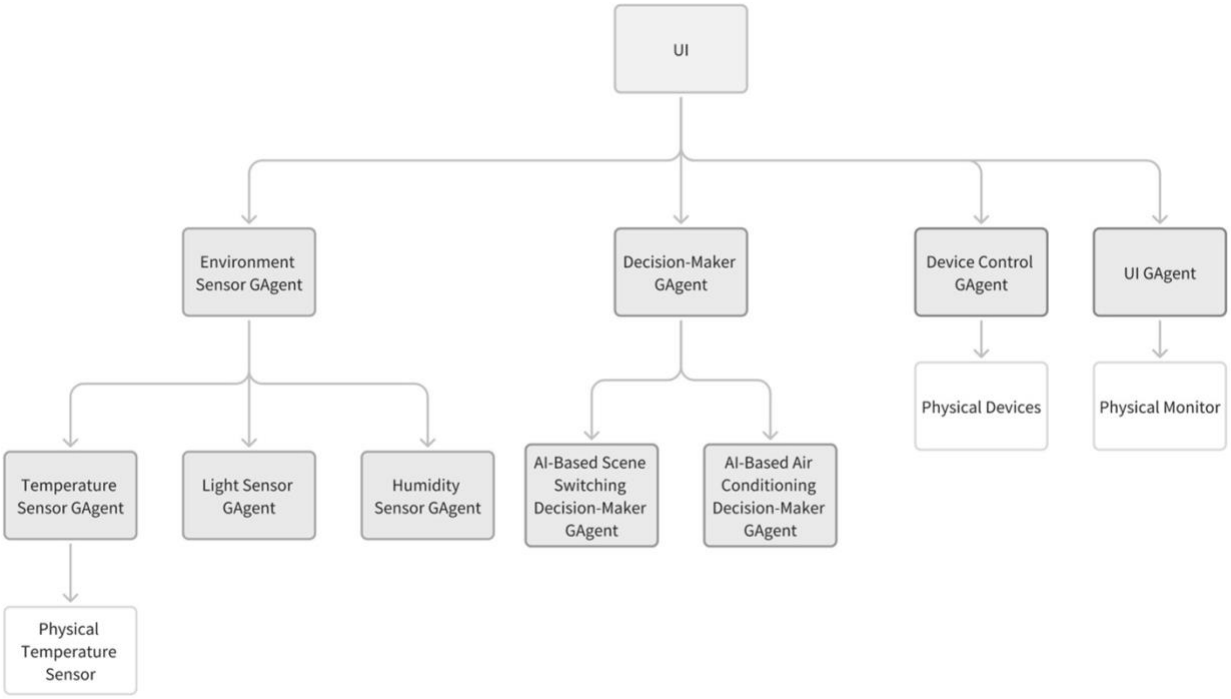
aevatar Dashboard

- Graphical management tool allowing users to configure multi-agent collaboration processes, monitor event flows, and edit business logic through low-code/no-code

approaches. This significantly reduces development barriers, especially for non-technical users.

4.3 GAgent Multi-Agent Collaboration Model

- Adopts GAgentBase<TState, TEvent> as an abstract base class, where agents can inherit and implement their own business processing methods.

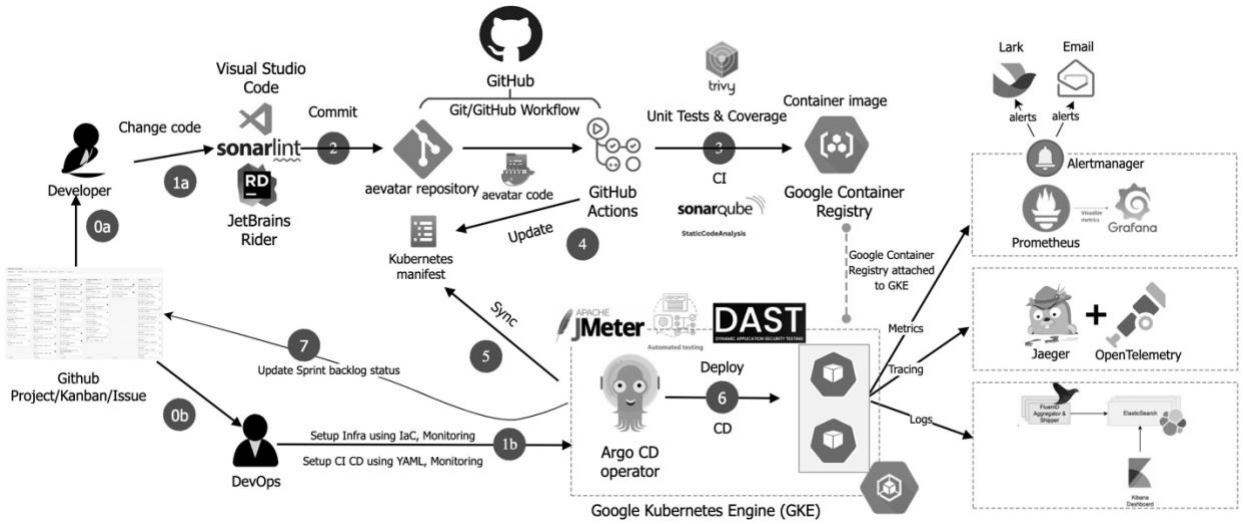


- **GAgent**: Manages subscription, message routing, and event coordination for multiple agents within a group, enabling broadcast, point-to-point, or tree-based event transmission within the group.

4.4 Multi LLMs Orchestration

- **Multi-Pronged Approach**: Incorporates access to multiple LLMs (GPT-4, Claude, Llama2, etc.) through 'AIService' and Semantic Kernel mechanisms.
- **Scheduling Strategy**: Dynamically decides which model(s) to call based on task type, resource cost, complexity, and other dimensions.
- **Model Adaptation Layer**: Connect to more third-party or private models at the framework level, providing enterprises with customised multi-language model management.

4.5 Cloud-Native Deployment and Security Compliance



Kubernetes Deployment

- Orleans Silo and agent services can be containerised, supporting automatic scaling (HPA), service discovery, and elastic load balancing.

DevSecOps & GitOps

- Provides container image security scanning, CI/CD integration, and Infrastructure as Code (IaC) deployment, ensuring application security and traceability.

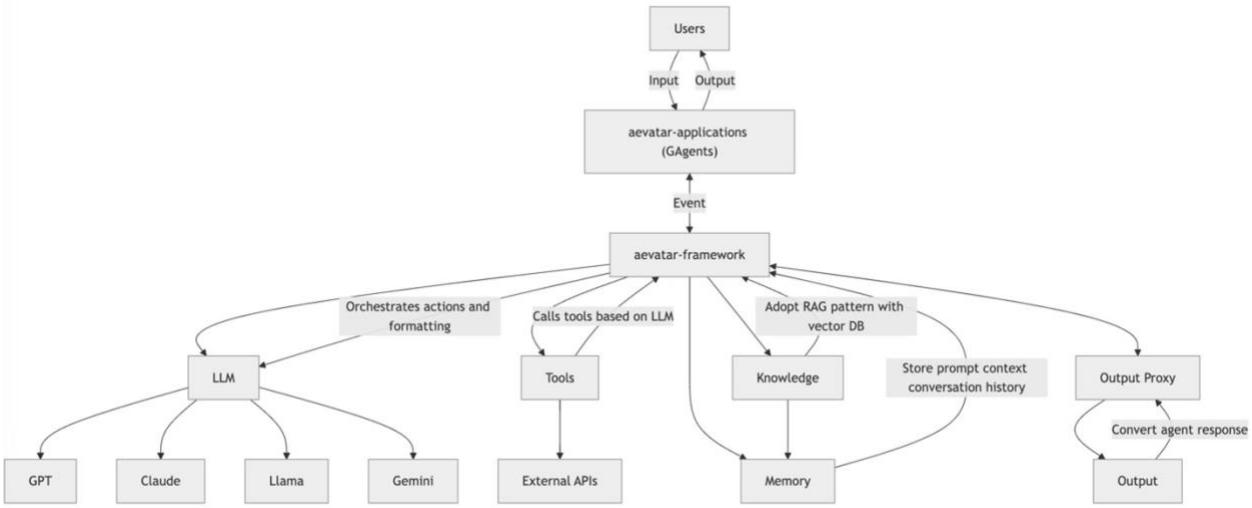
Security Policies

- Authentication through AuthServer and OAuth/OpenID systems, supporting multi-tenancy and RBAC (Role-Based Access Control).

5. Technical Implementation Details

The Overall Flow:

1. **User → aevatar GAgent**: User's message or command is captured.
2. **GAgent → aevatar Framework**: A structured event is passed to the framework (Multi-Agent Collaboration and AI interactions).
3. **aevatar Framework → Core Logic with RAG and LLM**: The RAF and LLM interpret the request.
4. **Core Logic with RAG and LLM → External services** or **Knowledge / Memory**: Retrieve data or call specialised actions.
5. **aevatar Framework → Output Proxy**: Final output is formatted and prepared.
6. **Output → GAgent → User**: The user receives the response.



The Detailed Flow:

- **Agent Creation & Initialisation**: The client asks `GAgentFactory` to create an agent, which initialises state with `StateLogEventStorage` and sets up subscriptions via `StreamProvider`.
- **Event Publishing & Handling**: The client (or other systems) publishes events to the agent, which appends them to the event storage, updates its in-memory state, and publishes them to an external stream if necessary.
- **State Recovery**: When needed, the agent retrieves a snapshot, and any subsequent events from the storage will apply all the changes, ending up with an up-to-date state.

Notable Features & Benefits

- **Multi-Agent Collaboration**: The system can split complex tasks into smaller specialised subtasks, each handled by an appropriate Agent.
- **Dynamic Flow**: Agents are activated and invoked on-demand (virtual actor model), allowing for concurrency or parallel calls in scenarios where tasks can be split up.
- **Integration with External Services**: Knowledge modules can seamlessly incorporate real-time data, domain documents, or advanced processing capabilities.
- **Retrieval-Augmented Generation (RAG)**: Agents can consult a vector database or memory store, enhancing the LLM or other logic with up-to-date contextual data.
- **Scalability & Extensibility**: Each component can be scaled horizontally, and new Agents or Tools can be introduced without major architectural changes.

5.1 Orleans Actor Model and Scalability

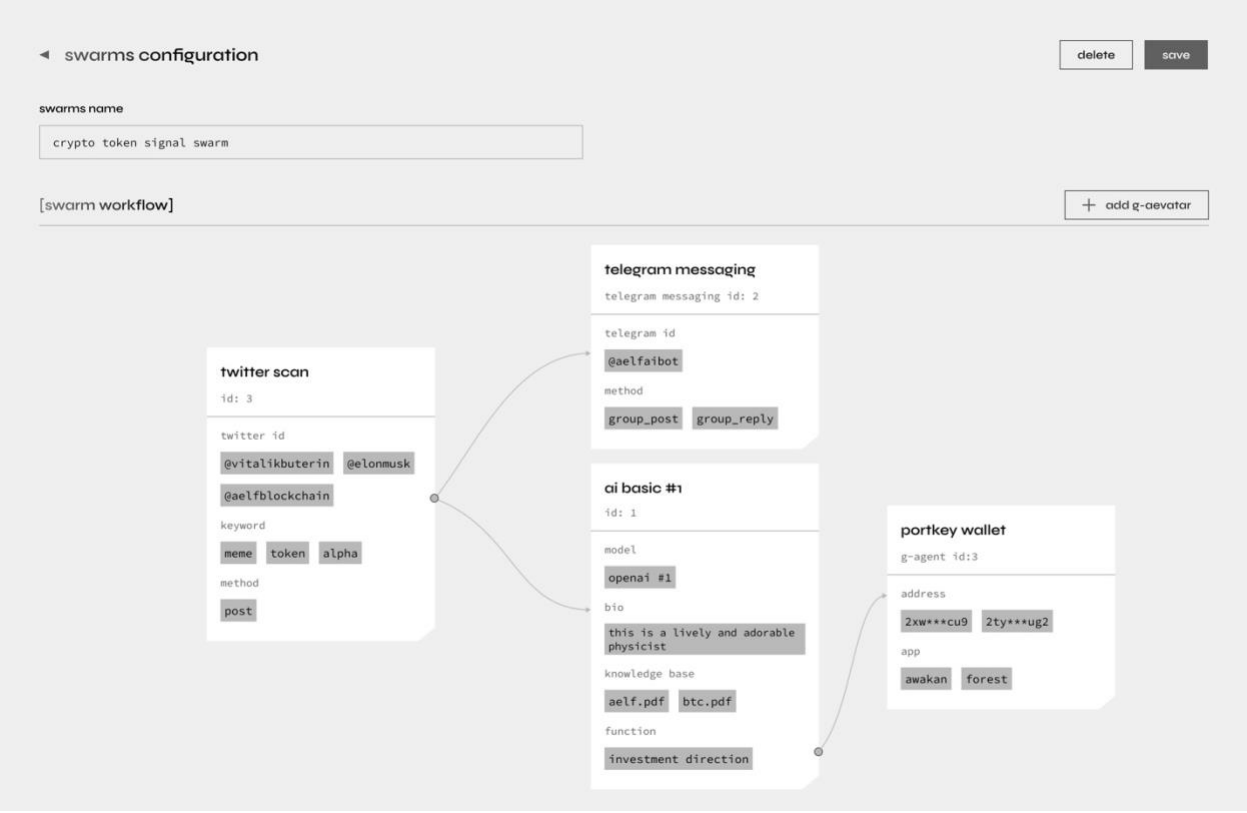
- Distributed Actor

    - Each agent acts as a Grain, storing its own state and event history. Orleans handles scheduling and message passing, eliminating the need for manual management of concurrent locks and network communication.
- Horizontal Scalability
    - When the system needs to handle more conversations or higher concurrency, adding Silo nodes can expand agent instances and automatically balance loads.

5.2 GAgentBase Design and Event-Driven Architecture

- GAgentBase<TState, TStateLogEvent>
    - Inherits JournaledGrain<TState, StateLogEventBase<TStateLogEvent>>, naturally possessing event sourcing capabilities.
    - Methods like PublishToAsync/SubscribeToAsync allow agents to freely combine and interact, forming many-to-many or multi-level event flow topologies.
- EventWrapper
    - Adds metadata such as ID, timestamp, and context to all events, facilitating audit and debugging, and avoiding traditional 'black box AI' problems.

5.3 Low-Code/No-Code Orchestration and Visualisation



Drag-and-Drop Process Design

- Users can drag agent nodes, configure event routing, and set model strategies on the dashboard without writing complex backend code.

Real-time Monitoring and Log Replay

- Integrates Event Sourcing logs, allowing viewing of event sequences or agent states at any moment through the aevatar Dashboard, assisting with business optimisation and maintenance troubleshooting.

5.4 Observability and Monitoring

- Distributed Tracing
  - Integrates with OpenTelemetry, Jaeger, or Zipkin for visualised tracking of cross-Agent/Grain call chains.
- Metrics and Alerts
  - Collects system metrics (QPS, latency, error rates, etc.) and implements real-time alerts based on Prometheus/Grafana.
- Orleans Dashboard
  - Optional built-in Orleans Dashboard showing runtime data such as Grain activation counts and message processing rates.

6. Key Features

6.1 Multi-Model Parallel Processing/Dynamic Switching

- Automatically (with a manual option) switches between different LLMs based on business requirements.
- Able to assign multiple models to handle subtasks simultaneously and merge results.

6.2 Advanced Task Orchestration and Collaboration

- GAgent provides event-based collaboration mechanisms, allowing multiple agents to handle complex business processes in parallel.

6.3 RAG Integration

- Connects with vector databases/document search engines, enabling agents to retrieve and generate answers from large-scale knowledge bases.

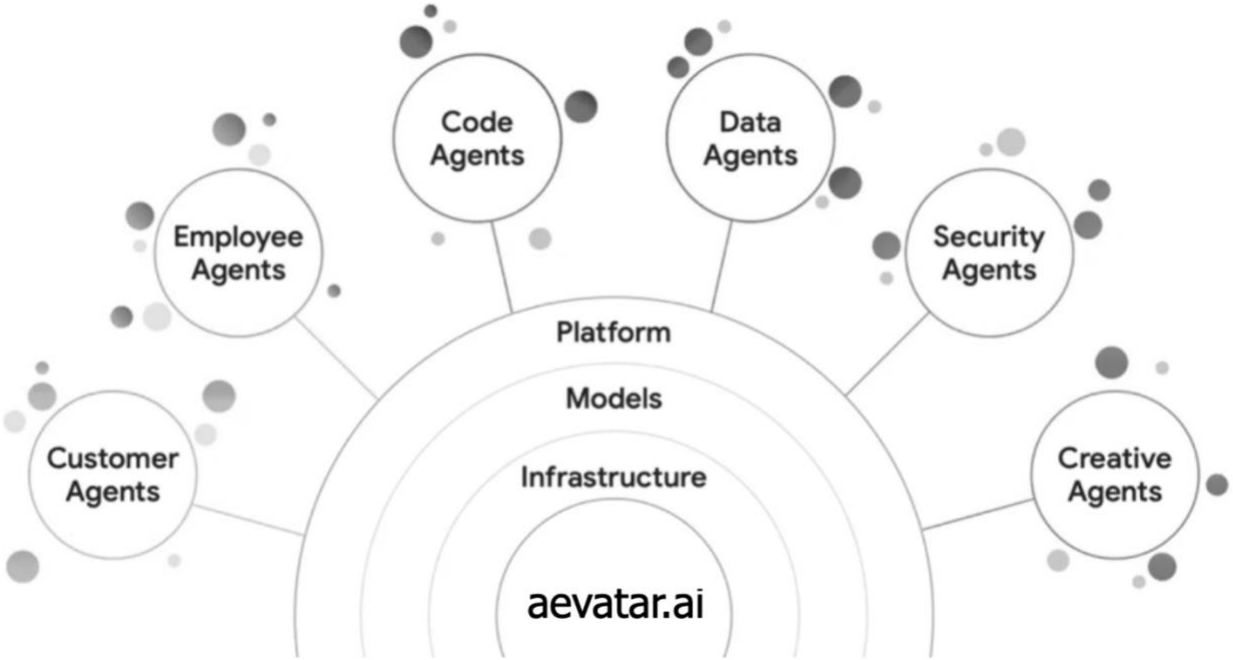
6.4 Cross-Platform Extension

- Plugin-based integration with Telegram, X, Slack, and more, quickly building multi-channel chat/communication scenarios.

6.5 Developer and Non-Developer Friendly

- **For developers**: Provides programmable Plugin framework.
- **For non-technical personnel**: Dashboard low-code/no-code management for quick onboarding.

7. Use Cases



7.1 Multi-Agent Collaboration/Automation

- Multi-Department Agents:

  Customer Agents

  - **Functions**:
    - Automate customer support responses via chat or email.
    - Provide product recommendations based on user behavior.
    - Handle order tracking, refunds, and inquiries.
    - Proactively follow up with customers for feedback or promotions.
  - **Example**: A customer agent resolves FAQs about delivery delays or provides real-time product recommendations.

Employee Agents

- **Functions**:
  - Manage HR workflows like leave requests, payroll queries, and benefits explanations.
  - Assist with onboarding by explaining company policies and processes.
  - Schedule meetings, send reminders, and organise employee data.
- **Example**: An employee agent can automatically approve standard leave requests and update calendars.

Code Agents

- **Functions**:
  - Generate boilerplate code snippets or refactor existing code.
  - Assist developers with debugging and testing.
  - Provide recommendations for API usage and best practices.
  - Automate code review processes for compliance and quality.
- **Example**: A code agent suggests optimised SQL queries for better performance.

Data Agents

- **Functions**:
  - Analyse data trends and generate reports.
  - Automate data cleaning, transformation, and preparation tasks.
  - Assist with database queries and visualisations.
  - Monitor data pipelines and notify stakeholders of anomalies.
- **Example**: A data agent identifies sales trends from historical data and predicts future demand.

Security Agents

- **Functions**:
  - Monitor for security breaches and unusual activity.
  - Enforce compliance by validating user permissions and auditing logs.
  - Automate incident response workflows and update security protocols.
  - Detect and flag phishing attempts or malware.
- **Example:** A security agent blocks access from suspicious IP addresses and alerts the admin.

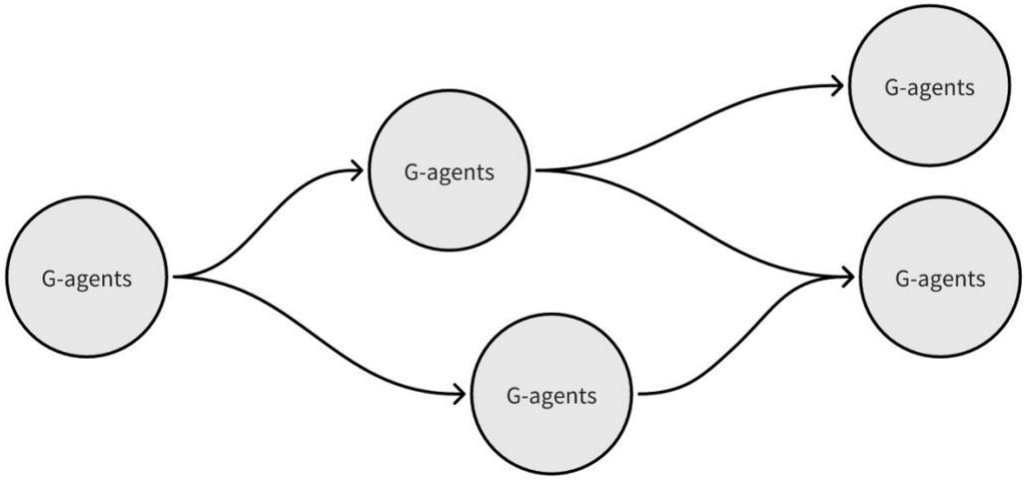
Creative Agents

- **Functions**:
  - Generate content like blog posts, marketing materials, and social media captions.
  - Assist in design tasks by providing suggestions or creating mock-ups.
  - Automate video and audio editing processes.

        - Ideate and brainstorm new product features or campaigns.
    - **Example**: A creative agent drafts a campaign slogan and designs a matching poster.

These agents can be customised further to match organisational needs, integrating with existing workflows to maximize productivity.

- Low-Code Management:
    - Users configure processes and set trigger conditions in aevatar Dashboard.
    - Agents automatically execute according to event flow once instructed.



The above task-based orchestration allows G-agents to operate independently yet coordinate seamlessly, leveraging each agent's expertise to handle complex objectives more efficiently.

7.2 Social Media Agents

Telegram/X Adaptation

- **Multi-Agent Deployment**:
    - Deploy multiple specialised agents to handle different aspects of user interaction. For example:
        - **Support Agent**: Answer FAQs and resolve issues.
        - **Marketing Agent**: Share promotional content, announcements, and campaigns.
        - **Community Agent**: Facilitate group discussions and engage in user feedback collection.
    - Assign agents to handle specific time zones for 24/7 coverage.
- **Communication Channels and Language Support**:
    - Integrates seamlessly with popular messaging platforms like Telegram, X (formerly Twitter), Facebook Messenger, WhatsApp, and WeChat.

    - Supports multilingual communication using real-time language translation to engage with users worldwide.
    - Customises responses based on regional dialects, cultural nuances, and preferences.

7.3 Industry Agents in Blockchain, Finance, Manufacturing

1. Blockchain Agents

- **Smart Contract Analysis Agents**:
    - **Fetching and Parsing**:
        - Automatically fetch smart contract texts from blockchain networks (e.g., Ethereum, Solana, Ton, aelf).
        - Parse contracts to understand logic and identify vulnerabilities using advanced NLP models.
    - **Risk Detection**:
        - Perform comprehensive static and dynamic code analysis to identify potential security risks like reentrancy, overflow, or access control issues.
        - Compare contracts against known patterns of vulnerabilities and flag anomalies.
    - **Anomaly Alert Agents**:
        - Push real-time alerts to operational agents or dashboards if anomalies, bugs, or unauthorised changes are detected.
        - Integrate with security agents to automatically initiate remediation steps, such as freezing contract actions.
- **Cross-Chain Operation Agents**:
    - Monitor and validate cross-chain transactions to ensure interoperability.
    - Track chain height, transaction integrity, and synchronisation across multiple blockchain environments.
- **Tokenomics Monitoring Agents**:
    - Track token supply, demand, and liquidity across exchanges.
    - Alert stakeholders of deviations or risks, such as inflation, imbalance, or insufficient liquidity.

2. Finance Agents

- **Fraud Detection Agents**:
    - Monitor and analyse real-time financial transactions to identify fraud (e.g., double spending, unauthorised withdrawals, or account breaches).
    - Use machine learning to continuously adapt and improve detection algorithms.
- **Portfolio Management Agents**:
    - Analyse historical and live market data to optimise portfolio allocations.
    - Provide personalised investment suggestions based on risk appetite and goals.
    - Offer rebalancing strategies in response to market shifts or changing investor profiles.
- **Compliance Agents**:

    - Automate the tracking of financial activities for AML (Anti-Money Laundering) and KYC (Know Your Customer) compliance.
    - Generate compliance reports for regulators and flag suspicious activities for review.
- **Customer Insight Agents**:
    - Segment customer profiles and suggest personalised financial products (e.g., loans, credit cards, or insurance policies).
    - Predict creditworthiness and risk using AI-powered financial modelling.

3. Manufacturing Agents

- **IoT Monitoring Agents**:
    - Continuously monitor IoT sensors on production lines to detect issues like temperature fluctuations, abnormal vibrations, or energy inefficiencies.
    - Generate real-time alerts to maintenance teams for proactive repairs.
- **Fault Diagnosis Agents**:
    - Combine IoT data with LLM capabilities to diagnose machine failures and recommend corrective actions.
    - Cross-reference historical maintenance records and expert data for enhanced accuracy.
- **Production Optimisation Agents**:
    - Analyse production data to recommend workflow improvements, resource allocation, and energy optimisation strategies.
    - Simulate alternative manufacturing scenarios to support decision-making.
- **Supply Chain Agents**:
    - Track inventory levels and suggest restocking or redistribution of raw materials.
    - Monitor supply chain logistics to identify bottlenecks or inefficiencies, ensuring seamless production flow.

4. Cross-Industry Capabilities

- **Data Analysis and Visualisation Agents**:
    - Aggregate and analyse data across blockchain, finance, and manufacturing domains.
    - Generate insightful, real-time dashboards for stakeholders to monitor key metrics and performance.
    - Provide predictive analytics to help businesses stay ahead of potential risks or opportunities.
- **Decision-Support Agents**:
    - Assist leadership in making informed decisions by generating actionable recommendations from data-driven insights.
    - Simulate "what-if" scenarios to evaluate the impact of different strategic options.
- **Custom Alerts and Notifications**:
    - Notify stakeholders of critical updates, anomalies, or emergencies through email, Slack, Telegram, or other preferred channels.
    - Enable customisable alert thresholds to prioritise the most critical notifications.

8. The Current AI Framework Landscape

8.1 Comparison: aevatar.ai vs. ElizaOS vs. G.A.M.E

Comparison	aevatar.ai	ElizaOS	G.A.M.E
Key Strength	• Easy maintenance • Supports multiple LLM per workflow • Low to no code builder for users to get started in minutes • Ability to replay events to analyse agent's workflow	• Growing feature set and plugin integrations • Full customisation and control	• Low-code, low complexity launches
Capabilities	• Diverse workflows, each powered by different LLMs working together for optimal results • Access to a library of workflows, made available through user's contribution. • Ability to duplicate agents or workflows easily	• Users can select their preferred LLM at the start; however, all agents can only collaborate using the same language model in one workflow	• Users can select their preferred LLM at the start; however, all agents can only collaborate using the same language model in one workflow.
Multi-LLM Orchestration	Semantic Kernel for multi-LLM orchestration, suitable for complex reasoning and decision-making in any kind of application	Single-model API integrations without multi-LLM automation, lack of flexibility across applications	Optimised for natural language interactions within virtual worlds and not general applications
Design	Modularisation + extensibility plug-in + dynamic cluster management system	Modular + extensible plugin system	Modular + environment agnostic
Target Users	Technical and non-technical builders	Technical builders	Non-technical builders

Coding Language	No code or low code	TypeScript/JavaScript	Low-code
Scalability	• Uses Orleans, a distributed framework combining microservices and the Actor model for scalable and highly available large-scale agent networks • Containerised deployment with Kubernetes for cross-cloud capabilities, auto-scaling, high availability, and high concurrency	• Uses Node.js, a multi-process architecture but lacks Orleans' distributed programming model	• Relies on game-specific backends like Photon or SpatialOS for real-time performance
Use Cases	Built for general-purpose, scalable, multi-domain logic in industries like blockchain and finance	Built for smaller web projects and community-driven prototyping	Built for gaming and metaverse scenarios with tokenomics integration
Cloud Native & DevOps	Advanced cloud-native Kubernetes deployment with robust security through DevSecOps & GitOps	Focuses on speed but without extensive automation and compliance mechanisms	Focuses on performance but does not provide comprehensive cloud-native tools
DevOps Maintainability	Agent-as-a-Service simplifies system oversight by deploying lightweight agents. They autonomously monitor, automate, and manage operations across multiple environments.	Supabase offers DevOps maintainability through its Backend-as-a-Service platform, for smooth deployment	Undetermined - closed source
Code Access	Open source	Open source	Closed source (Blackbox)
Platform Integrations	• Twitter • Telegram	• Discord • Twitter • Telegram • Farcaster • Warpcast	• Discord • Twitter • Telegram • Farcaster

8.2 Technical and Business Value

1. Powerful Multi-Language Model Collaboration

- **Seamless Integration of Multiple LLMs**:
    - Enables the dynamic invocation of different Large Language Models (LLMs) within a single business process.
    - Supports specialised models for different tasks (e.g., GPT for conversational tasks, domain-specific LLMs for compliance or technical analysis).
    - Combines the strengths of multiple LLMs to optimise accuracy and efficiency in workflows.
- **Cost and Performance Optimisation**:
    - Automatically routes requests to the most cost-effective or high-performance LLM based on the task requirements.
    - Supports hybrid deployments (cloud-based and on-premise models) to ensure flexibility and cost control.
    - Includes fine-tuning mechanisms to optimise LLM behaviour and reduce dependency on costly proprietary models.

2. Ease of Use

- **Low/No-Code Development with aevatar Marketplace**:
    - Features a drag-and-drop interface for creating and configuring AI agents without the need for extensive coding knowledge.
    - Provides prebuilt templates and workflows for common business processes, significantly reducing setup time.
    - Supports business users to create and modify workflows, reducing reliance on development teams.
- **Accelerated Agent Development and Deployment**:
    - Shortens development cycles by automating repetitive tasks such as environment setup, agent training, and deployment.
    - Simplifies operations with centralised management for deploying, monitoring, and maintaining agents.
- **Extensive Marketplace Ecosystem**:
    - Offers a library of pre-built agents, workflows, and integrations with third-party applications.
    - Ensures rapid onboarding and customisation for new business scenarios.

3. High Concurrency and Traceability

- **Scalable Architecture with Actor + Event Sourcing**:
    - Leverages an actor-based system for efficient parallel processing of tens of thousands of operations.
    - Supports horizontal scaling to accommodate growth in user demand or workload complexity.

    - Guarantees system reliability even during peak loads through distributed architecture and failover mechanisms.
- **Replayable and Auditable Interaction Histories**:
    - Event-sourced architecture ensures that all interactions, decisions, and operations are logged in detail.
    - Provides a complete replay of historical data to reconstruct workflows, debug issues, or conduct compliance audits.
    - Enables granular auditing of agent decisions to enhance transparency and trust.

4. Security and Compliance

- **Cloud-Native and DevSecOps-Driven Security**:
    - Integrates best practices in Cloud-Native Security, combining automated monitoring, threat detection, and real-time mitigation.
    - Embeds security checks and policies throughout the CI/CD pipeline using DevSecOps principles.
    - Ensures secure code development with automated scanning for vulnerabilities and misconfigurations.
- **GitOps for Secure and Consistent Deployments**:
    - Implements GitOps workflows for version-controlled, automated, and reproducible deployments.
    - Provides rollback mechanisms for recovering from issues or reverting changes securely.
- **Kubernetes-Oriented Automation**:
    - Automates container orchestration and scaling with Kubernetes, ensuring robust and efficient deployments.
    - Leverages Kubernetes' role-based access control (RBAC) and network policies to enforce strict security requirements.
- **Compliance-Driven Design**:
    - Ensures adherence to regulatory standards through automated compliance checks.
    - Provides comprehensive reporting and audit tools to satisfy internal and external compliance requirements.

5. Additional Business Value

- **Operational Efficiency**:
    - Reduces time-to-market for AI-powered solutions by streamlining development and deployment processes.
    - Enables businesses to scale AI capabilities quickly without major investments in infrastructure or specialised talent.
- **Enhanced User Experience**:
    - Delivers faster, more accurate, and contextually aware responses through optimised agent workflows.
    - Customisable interfaces and workflows adapt to specific business and user needs.
- **Future-Proofing Investments**:

- Designed to integrate with emerging technologies (e.g., quantum computing, advanced LLMs, or decentralised AI networks).
- Built with modular and flexible architecture, ensuring adaptability to future business and technical requirements.

9. Roadmap

9.1 Short-Term Plan

Teams	Completed	Phase 1 – 2025 Q1	Phase 2 – 2025 Q2
aevatar-framework	**Foundation** 1. Multi-agent foundational framework **Marketplace** 1. GAgent Marketplace standard 2. AI component Marketplace standard	**AI GAgent Upgrade** 1. Support for more LLMs 2. Memory upgrade 3. Knowledge base 4. RAG upgrade **Orchestration** 1. Natural language generation 2. Visual workflow panel **GAgent Type Enrichment** 1. System GAgent 2. Richer social media components **Deployment** 1. Permission configuration and auto-scaling	**AI GAgent Upgrade** 1. Multi-modal 2. Knowledge base sharing 3. Self-feedback **Orchestration Upgrade** 1. Feedback & evaluation module **GAgent Type Enrichment** 1. aelf chain component
aevatar-applications	**Pumpfun**	**Aevatar Workflow SDK** 1. Permission system 2. GAgent construction 3. Workflow creation **Mysterious AI Game**	**Station 1.0** 1. Agents-as-a-Service 2. Marketplace 3. GAgent & Swarms construction 4. Dashboard **AI Competition**

9.2 Long-Term Plan

Enhanced Vector Retrieval (RAG) Capabilities

- Native support for vector databases.
- Optimisation of massive document chunk retrieval and AI answer generation.

Enhanced Agent Plugin Marketplace

- Launch multi-industry plugin ecosystem.
- Provide plug-and-play agent modules for vertical scenarios like:
    - Financial risk control
    - Supply chain management
    - Healthcare

Service Mesh and Zero Trust Security

- Further deeper service mesh integration.
- Strengthen data encryption, traffic control, and access policies.

Human Feedback Mechanism

- Enable real-time human feedback training for agents.
- Continuously optimise conversation quality, logical reasoning, and behavioural decisions.

Enhanced Trusted Execution Environments (TEEs)

- Provide robust support for a variety of blockchain plugins, encompassing everything from on-chain transactions to Trusted Execution Environments (TEEs).

Boundless Agent Collaboration

- Explore interconnection with third-party AI systems and edge computing devices.
- Extend multi-agent collaboration from Cloud Native to IoT or other AI agent platforms.

10. Conclusion

As we approach an environment where multi-model, multi-agent collaboration becomes mainstream, aevatar.ai, as **a pioneer of the next-gen multi-agent AI framework**, provides a cross-platform, cross-language model, low-barrier and highly extensible solution.

By fully utilising the Orleans Actor model, event sourcing, and cloud-native architecture, it achieves the following key values:

- **Comprehensive Multi-Agent Collaboration**: Breaking through the limitations of single model and closed ecosystems, enabling different AI agents to share information and communicate effectively.
- **Visualisation and Low Code**: Significantly reducing development and maintenance barriers, helping users at different levels quickly implement AI agent solutions.
- **High Concurrency and Traceability**: Distributed Actor and Event Sourcing ensure stability and auditability in large-scale scenarios.

- **Security and Scalability**: Cloud-native DevSecOps solution flexibly meets industry customisation needs while ensuring compliance.

Looking ahead, aevatar.ai will continue to iterate and upgrade, building a fully functional and robust Agent-as-a-Service platform. We aim to bring convenient and powerful AI collaboration experiences to more industries and individual users.

We would like to invite you, our community, partners, and enterprise users to participate in the ecosystem's growth and work together to promote the openness and success of AI agent systems. For further information, please refer to our official documentation, GitHub repository, or contact our team at developer@aevatar.ai to discuss ways aevatar.ai can best meet your needs.

Disclaimer: This white paper provides an overview of aevatar.ai's architecture and capabilities. Feature specifications may evolve; please refer to official releases for the most up-to-date information.

11. Reference

1. Eliza: A Web3 friendly AI Agent Operating System, 2025, https://arxiv.org/pdf/2501.06781
2. Virtuals Protocol, 2024, https://whitepaper.virtuals.io
3. Kinds.ai, 2024, https://whitepaper.kinds.ai
4. Delysium, 2024, https://delysium.gitbook.io/whitepaper
5. Shafran, I., Cao, Y. et al., 2022, ReAct: Synergizing Reasoning and Acting in Language Models
6. Wei, J., Wang, X. et al., 2023, Chain-of-Thought Prompting Elicits Reasoning in Large Language Models
7. Wang, X. et al., 2022, Self-Consistency Improves Chain of Thought Reasoning in Language Models
8. Diao, S. et al., 2023, Active Prompting with Chain-of-Thought for Large Language Models
9. Zhang, H. et al., 2023, Multimodal Chain-of-Thought Reasoning in Language Models
10. Yao, S. et al., 2023, Tree of Thoughts: Deliberate Problem Solving with Large Language Models
11. Long, X., 2023, Large Language Model Guided Tree-of-Thought
12. Google, Google Gemini Application
13. Xie, M., 2022, How does in-context learning work? A framework for understanding the differences from traditional supervised learning
14. Google Research, ScaNN (Scalable Nearest Neighbors)
15. Semantic Kernel, https://learn.microsoft.com/en-us/semantic-kernel
16. LangChain, https://www.langchain.com
17. LangGraph, https://www.langchain.com/langgraph
18. Crewai, https://www.crewai.com