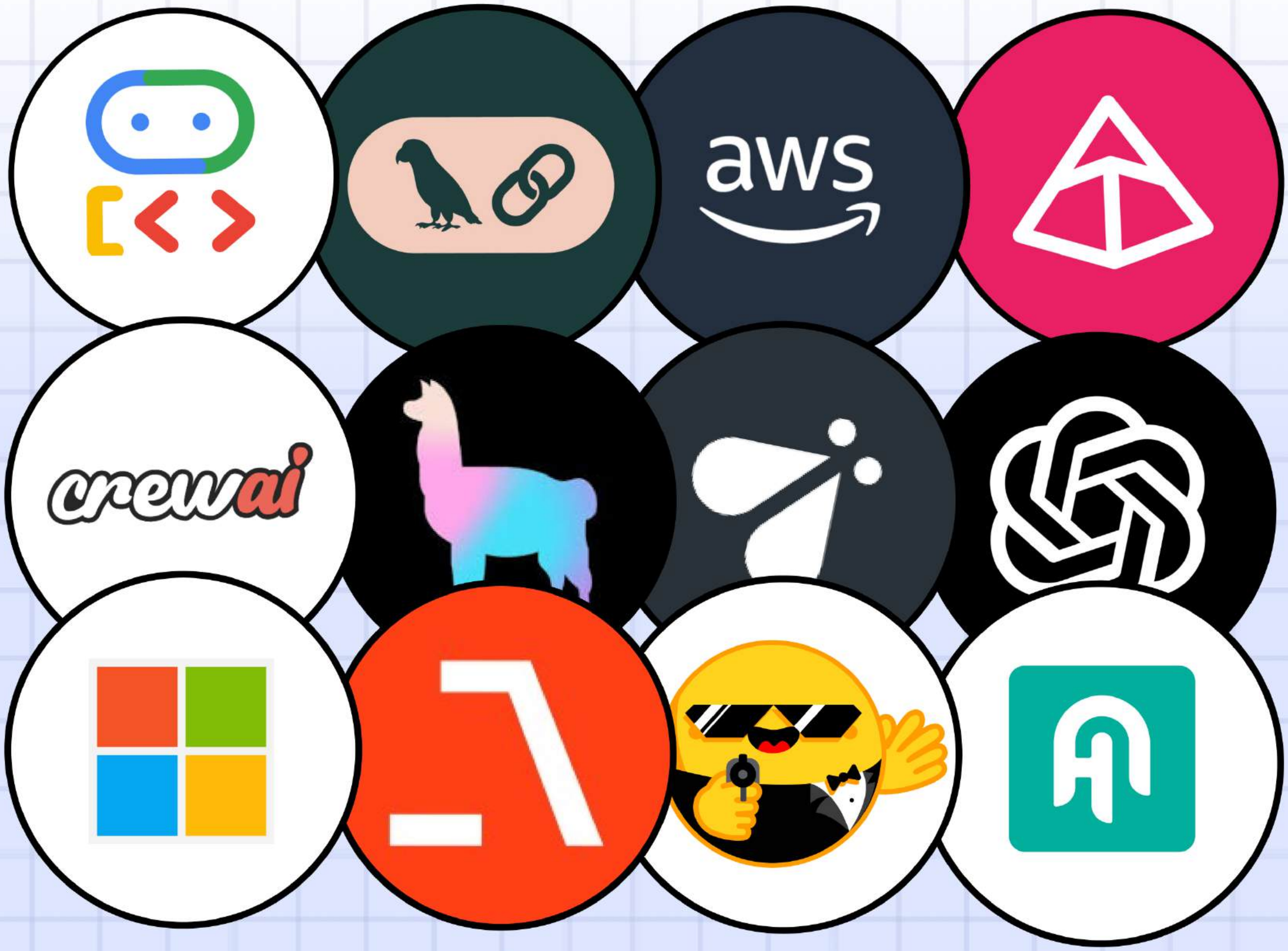
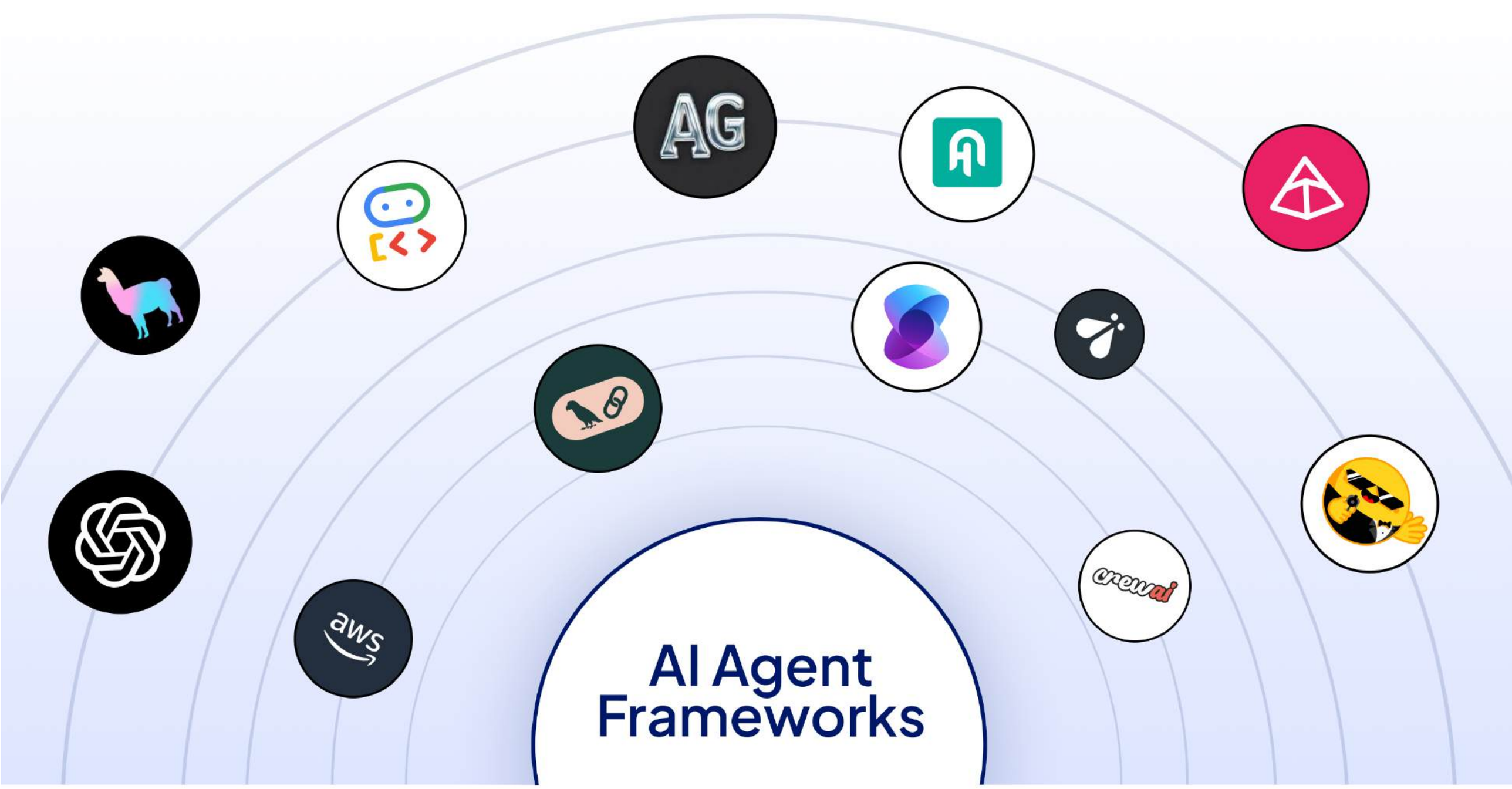


Popular AI Agent Frameworks

@rakeshgohel01



Brief

Agent frameworks link agent components like nerves in a body

That's why choosing the right one can make or break your entire agent eco-system.

Today, we wil take a look at each of these framework to identify which fits your use cases better. →

LangChain is an established framework in the agentic AI ecosystem, and LangGraph extends its capabilities to support complex, stateful agent workflows.

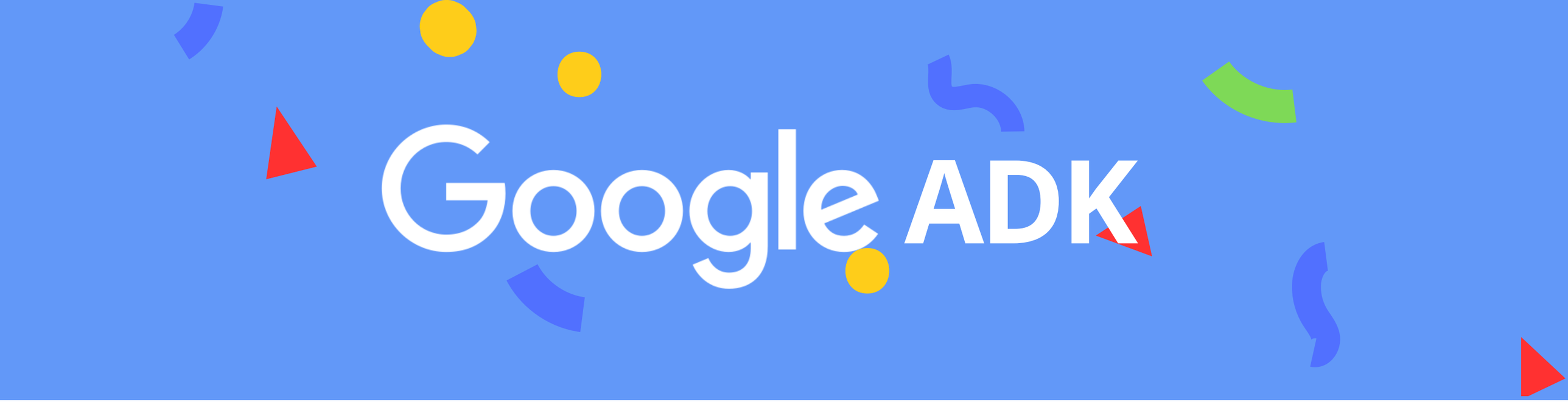
KEY FEATURES OF LANGGRAPH

- **Component ecosystem:** They provide an extensive library of pre-built components that enable the rapid development of specialized agents.

- **Foundation model selection:** The frameworks support diverse foundation models like Anthropic Claude and Amazon Nova models, offering different reasoning capabilities.

- **Graph-based workflows:** LangGraph allows for the definition of complex agent behaviors as state machines, supporting sophisticated decision logic.

- **Memory abstractions:** They offer multiple options for short-term and long-term memory management, which is essential for agents that need to maintain context over time.



Google's Agent Development Kit (ADK) is an open-source framework designed to simplify the full-stack end-to-end development of agents and multi-agent systems.

KEY FEATURES OF GOOGLEADK

- **Model-Agnostic Design:** While optimized for Gemini models, ADK can work with any LLM provider.

- **Flexible Orchestration:** Define workflows using structured workflow agents (Sequential, Parallel, Loop) for predictable pipelines, or leverage LLM-driven dynamic routing for adaptive behavior.

- **Multi-Agent Architecture:** Build modular and scalable applications by composing multiple specialized agents.

- **Rich Tool Ecosystem:** Equip agents with diverse capabilities using pre-built tools (Search, Code Execution), create custom functions, integrate third-party libraries (LangChain, CrewAI), or use other agents as tools.

CrewAI is an open-source framework focused on autonomous multi-agent orchestration. It provides a structured approach for creating teams of specialized autonomous agents.

KEY FEATURES OF CREWAI

- **Role-based agent design:** Autonomous agents are defined with specific roles, goals, and backstories to enable specialized expertise.

- **Task delegation:** The framework includes built-in mechanisms for autonomously assigning tasks to the most appropriate agents based on their capabilities.

- **Agent collaboration:** It provides a framework for autonomous inter-agent communication and knowledge sharing without human mediation.

- **Process management:** CrewAI offers structured workflows for both sequential and parallel autonomous task execution

Agents SDK

The OpenAI Agents SDK is a lightweight, Python-first package for building agentic AI applications with minimal abstractions. It is a production-ready evolution of OpenAI's earlier experiments OpenAI Swarm.

KEY FEATURES OF AGENTS SDK

- **Python-First Design:** It encourages using native Python features for orchestration and chaining agents, reducing the need to learn new, complex abstractions.

- **Agent Loop:** Provides a built-in agent loop that handles the logic of calling tools, sending the results back to the LLM, and continuing until a task is complete.

- **Handoffs:** A key feature that allows one agent to delegate specific tasks to other, more specialized agents, enabling multi-agent collaboration.

- **Guardrails:** Enables the validation of inputs to agents, allowing checks to run in parallel and fail early if they don't meet criteria.

Initially released by Microsoft, AutoGen is an open-source framework that focuses on enabling conversational and collaborative autonomous AI agents. It provides a flexible, event-driven architecture for building multi-agent systems.

KEY FEATURES OF AUTOGEN

- **Conversational agents:** The framework is built around natural language conversations between autonomous agents, facilitating sophisticated reasoning.

- **Asynchronous architecture:** Its event-driven design supports non-blocking agent interactions and complex parallel workflows.

- **Human-in-the-loop:** AutoGen provides strong support for optional human participation in otherwise autonomous agent workflows.

- **Code generation and execution:** It offers specialized capabilities for autonomous agents that can write and run code.

Semantic Kernel

Semantic Kernel is an open-source SDK from Microsoft designed to integrate large language models with conventional programming languages like C# and Python

KEY FEATURES OF SEMANTIC KERNEL

- **Plugin Architecture:** Its core feature is the ability to expose native code (e.g., C# functions) to an LLM as reusable plugins that the model can call to perform actions**.**

- **Planner:** Semantic Kernel includes a "planner" that can automatically generate a multi-step plan to achieve a user's goal by combining different plugins.

- **Memory:** It provides built-in capabilities for storing and retrieving conversational history and other contextual information, giving agents long-term memory.

- **Connectors:** The framework offers native integration with OpenAI, Azure OpenAI, and Hugging Face models, making it flexible in its choice of LLMs.

Microsoft Agent Framework

Microsoft's unified, production-ready SDK and runtime that merges Semantic Kernel's enterprise-grade plugins, memory, and orchestration with AutoGen.

KEY FEATURES OF AGENT FRAMEWORK

- **Graph-based workflows** with typed edges, branching, parallelism, and checkpointing.
- **First-class multi-agent patterns:** sequential, concurrent, handoffs, group chat, Magentic.
- **Open standards:** MCP, OpenAPI, and Agent-to-Agent interoperability.
- **Enterprise ops:** observability, approvals, durability, and recovery for long runs.
- **Strong typing and validation** across agents and workflows to reduce runtime errors.

Strands Agents is an open-source SDK, initially released by AWS, designed with a model-first approach for building autonomous AI agents.

KEY FEATURES OF AWS STRANDS

- **Model-first design:** The framework is built around the concept that the foundation model is the core of agent intelligence, which allows for sophisticated autonomous reasoning.

- **MCP integration:** It has native support for the Model Context Protocol (MCP), enabling standardized context provision to LLMs for consistent operation.

- **AWS service integration:** Strands Agents provides seamless connections to Amazon Bedrock, AWS Lambda, AWS Step Functions, and other services for comprehensive autonomous workflows.

- **Foundation model selection:** It supports a variety of foundation models, including Anthropic Claude and Amazon Nova models on Amazon Bedrock, to optimize for different reasoning capabilities.

Pydantic Agents

Pydantic AI is a Python agent framework built on the highly popular Pydantic data validation library. It is designed to build production-grade applications by focusing on robust data validation, and schema enforcement

KEY FEATURES OF PYDANTIC AGENTS

- **Data Validation and Parsing:** Its core strength lies in rigorous data validation and type safety, ensuring that data structures passed to and from LLMs are consistent and error-free.

- **Structured Outputs:** Pydantic AI excels at coercing unstructured LLM outputs into structured Pydantic models, which is crucial.

- **Schema Enforcement:** It automates the enforcement of data schemas, which is critical when working with structured APIs and data pipelines in an agentic context.

- **Seamless Integration:** It integrates smoothly with other agent frameworks and is often used alongside them to handle the data layer.

LLama Index

LlamaIndex is a data framework for LLM applications that specializes in connecting custom data sources to large language models. It provides tools for data ingestion, indexing, querying and a lot others.

KEY FEATURES OF LLAMAINDEX

- **Data Ingestion and Indexing:** LlamaIndex excels at connecting to and ingesting data from various sources (PDFs, APIs, databases) and indexing it for efficient retrieval.

- **Query Engine:** It offers a powerful query engine that allows natural language questions over your data, abstracting away the complexity of vector database queries.

- **RAG Capabilities:** The framework is purpose-built for creating advanced RAG pipelines, enabling LLMs to answer questions based on private or custom data.

- **Agentic Features:** LlamaIndex can be used to build knowledge agents that can perform automated reasoning and decision-based on a given knowledge base and task list.

Haystack

Haystack is an open-source framework for building end-to-end AI applications that utilize large language models. It is highly modular and flexible, allowing developers to create custom pipelines for a wide range of GenAI Tasks.

KEY FEATURES OF HAYSTACK

- **Pipeline-Based Architecture:** Haystack allows developers to build custom data processing and query pipelines by combining different nodes.

- **Advanced RAG:** It supports building high-performing Retrieval-Augmented Generation pipelines with various strategies, including hybrid retrieval and self-correction loops.

- **Agentic Pipelines:** The framework supports complex workflows using the function-calling capabilities of LLMs, enabling branching and looping for multi-step agentic tasks.

- **Multimodal AI:** Haystack can be used to develop applications that process not just text but also other modalities.

IBM Bee

The Bee Agent Framework is an open-source toolkit developed by IBM Research for building, deploying, and managing scalable agent-based workflows. It is designed to simplify the development of distributed agent systems.

KEY FEATURES OF BEE AGENT FRAMEWORK

- **Scalable Workflows:** The framework is designed to construct and serve powerful agentic workflows at scale, handling complex and distributed tasks.

- **Multi-Agent Systems:** Recent updates have added extensions for multi-agent collaboration, allowing multiple "bee" agents to work together to solve a problem

- **Developer Experience:** A key goal of the framework is to "make simple things simple and complex things possible," with a focus on ease of use.

- **Observability:** The ecosystem includes tools like Bee UI for visual interaction and Bee Observe for telemetry collection and management

Smol Agents

Hugging Face's smolagents is a Python library designed to simplify the creation of AI agents, making agentic AI more accessible to a broader range of developers. It aims to be a plug-and-play framework that requires minimal setup.

KEY FEATURES OF SMOL AGENTS

- **Simplicity:** The library is designed for ease of use, allowing a developer to create and run a functional agent in just a few lines of code.

- **Model Agnostic:** While part of the Hugging Face ecosystem, it can work with various models, including those from the Hugging Face Hub, OpenAI, and Anthropic, as well as local models.

- **Modality-Agnostic:** Beyond text, smolagents can handle vision, video, and audio inputs, which broadens the range of possible applications.

- **Custom Tools:** The framework makes it straightforward to build custom tools for an agent to use, alongside built-in tools like web search.

Agno (Prev. Phidata)

Previously Known as Phidata. Agno is an open-source, high-performance Python framework for multi-agent workflows with a built-in FastAPI runtime and control plane.

KEY FEATURES OF SMOL AGENTS

- **Unified, Pythonic framework** for single agents, teams, and step-based workflows (sequential, parallel, branching, loops).

- **Ready-to-use FastAPI "AgentOS"** app for serving agents with a built-in control plane for testing, monitoring, and management.

- **High performance and low overhead** focus with async runtime, minimal memory footprint, and horizontal scalability.

- **Declarative agent composition:** configure model, memory, tools, and knowledge sources with simple Python.

- **Transparent reasoning and observability:** : inspect traces, tool calls, and logs for reliability and auditability.

Framework Use Cases

Langgraph

- Suited for complex, multi-step reasoning with rich component orchestration for autonomous decision-making.
- Great for Python teams needing large plugin ecosystems and robust memory/state management.

Google ADK

- Optimal for developing production-grade, multi-agent applications needing flexible orchestration and modality support.
- Recommended for teams requiring seamless interoperability, advanced streaming, or tight Google Cloud integration.

CrewAI

- Effective for problems benefitting from specialized, role-based agent teams autonomously collaborating.
- Useful when explicit multi-agent coordination and division of labor enhance solution quality.

OpenAI’s Agents SDK

- Ideal for quickly building agentic applications tightly integrated with OpenAI's ecosystem using Python.
- Great for action-oriented AI apps that must call APIs, delegate tasks, or use a lightweight agent/control loop.

Autogen

- Fits applications desiring natural language, conversational agent flows and collaborative multi-agent reasoning.
- Good for scenarios combining autonomous operation with optional human oversight or code generation/execution.

Semantic Kernel

- Well-suited for integrating LLMs with business logic via code/plugin functions in enterprise applications.
- Good for building AI copilots and automated workflows

Microsoft Agent Framework

- Compose specialized agents (data retrieval, analysis, policy/compliance) that collaborate via structured handoffs
- Run long-lived, stateful processes that autonomously execute tools and invite approvals

AWS Strands

- Ideal for teams leveraging AWS infrastructure needing deep, native integration with AWS services.
- Best for enterprise applications demanding flexibility

Pydantic AI

- Best as a supporting layer for any agentic/ML application where rigorous data validation, and type safety are critical.
- Useful in AI APIs and data pipelines needing structured, predictable inputs/outputs from LLMs.

Llama Index

- Best for building applications powered by custom/unstructured data, including RAG systems.
- Ideal when you need agents or chatbots to provide natural language access to large, private or structured data sources.

Haystack

- Excellent for creating flexible, composable LLM pipelines
- Favored in use cases demanding multimodal support, large-scale document search, or complex agentic workflows.

IBM Bee

- Suited for building robust, distributed multi-agent systems
- Ideal for enterprise scenarios orchestrating multiple specialized agents

HF Smol Agents

- Best for developers needing fast, simple prototyping
- Useful in education, experimentation, or rapid tool-driven agents

Agno (Prev. Phidata)

- Private, in-cloud AgentOS runtime with built-in control plane for testing
- High-performance, step-based multi-agent workflows that are stateless, async, and horizontally scalable

Conclusion

When selecting an agentic AI framework for autonomous agent development,

Consider how each option aligns with your specific requirements, including technical capabilities, organizational fit, team expertise, existing infrastructure, and long-term maintenance requirements.

If you want to learn more,

Each framework's repo/documentation is attached in the comment section of this document's post.

Hi, I am
Rakesh Gohel

“We help businesses 10X their growth with Cloud and AI Agents”

FOLLOW TO LEARN MORE ABOUT AI AGENTS

linkedin.com/in/rakeshgohel01