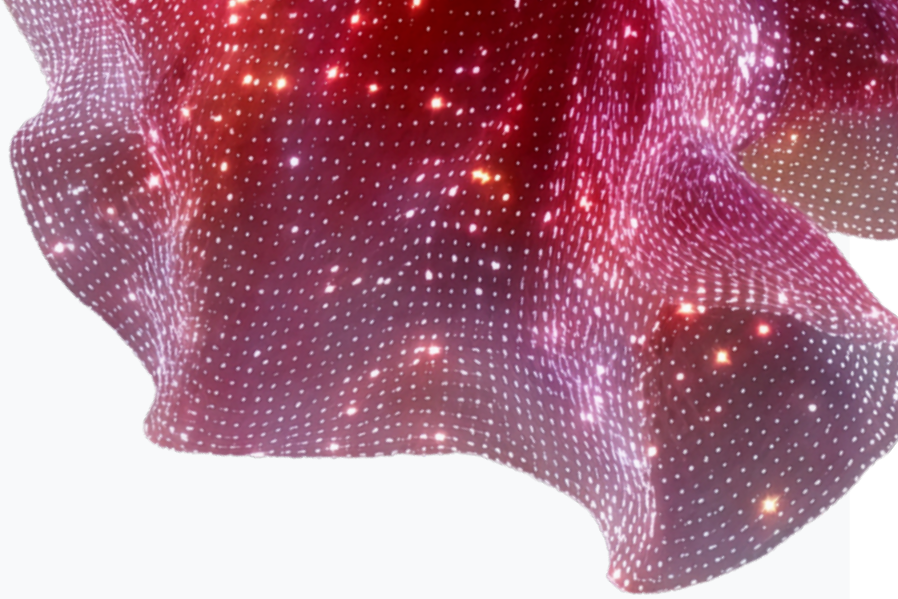
Google Cloud

An enterprise guide to multi-agent systems



Executive summary

AI agents are dominating boardroom conversations right now. Having already seen the value of generative AI, executives now want to put these more autonomous entities to work.

The reason is simple—AI agents move beyond incremental efficiency. Unlocking scale and productivity, they help teams manage complex workflows, optimize supply chains in real-time, resolve customer issues 24/7, and so much more.

Already, 39% of executives say their organization has launched more than 10 AI agents.[1] Yet scaling from a promising prototype to a production-ready agent reveals a new set of challenges. These obstacles can include:

- **Managing complex agentic systems.** How do you ensure your AI agents can operate securely, reliably, and efficiently at a global scale—especially when integrated into complex, existing business systems?
- **Optimizing performance at scale.** How can you ensure a great user experience while optimizing for quality, speed, and operational cost?
- **Ensuring governance, security, and compliance.** How do you govern your multi-agent systems and ensure strong control for data privacy, ethics, and regulatory compliance?
- **Integrating with existing systems.** How will your AI agents access your data, applications, and workflows?

This technical guide will help answer questions like these. It provides a systematic, operations-driven roadmap for developers and teams to navigate the new landscape with confidence.

You'll learn how to build efficient, scalable, and secure AI-driven solutions without sacrificing enterprise robustness. And you'll soon be on your way to creating complex, multi-agent systems that can seamlessly collaborate to reimagine enterprise workflows and customer experiences, all powered by Google Cloud.

Core concepts

Need a quick primer on models, grounding, orchestration, and more?

Explore AI agents

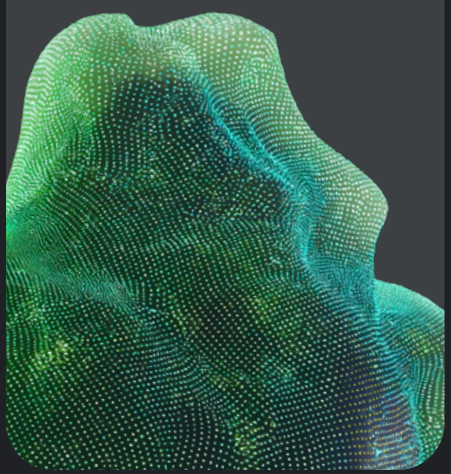
1. Google Cloud, The ROI of AI, 2025

3 phases of the agentic AI lifecycle

In this enterprise guide to agentic AI, we explore the three phases of the agent lifecycle, honing in on Vertex AI Agent Builder—the unified platform that provides the secure, open foundation needed to transform workflows into powerful, multi-agent systems at global scale.

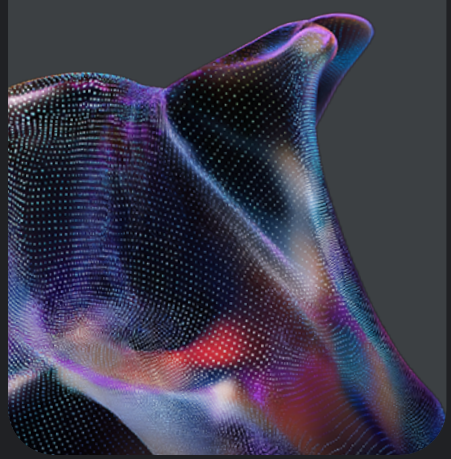
1 Build

With openness and choice at the core, Agent Builder gives developers the option of using Agent Development Kit (ADK) or other open frameworks. It also offers an expansive range of models.



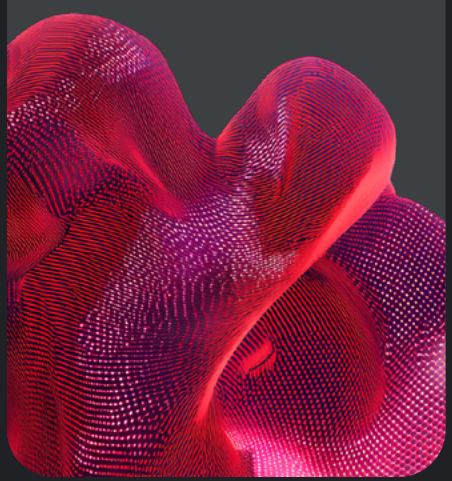
2 Scale

Powered by Agent Engine, enterprises can move agents into production with predictable performance and reliability assured.



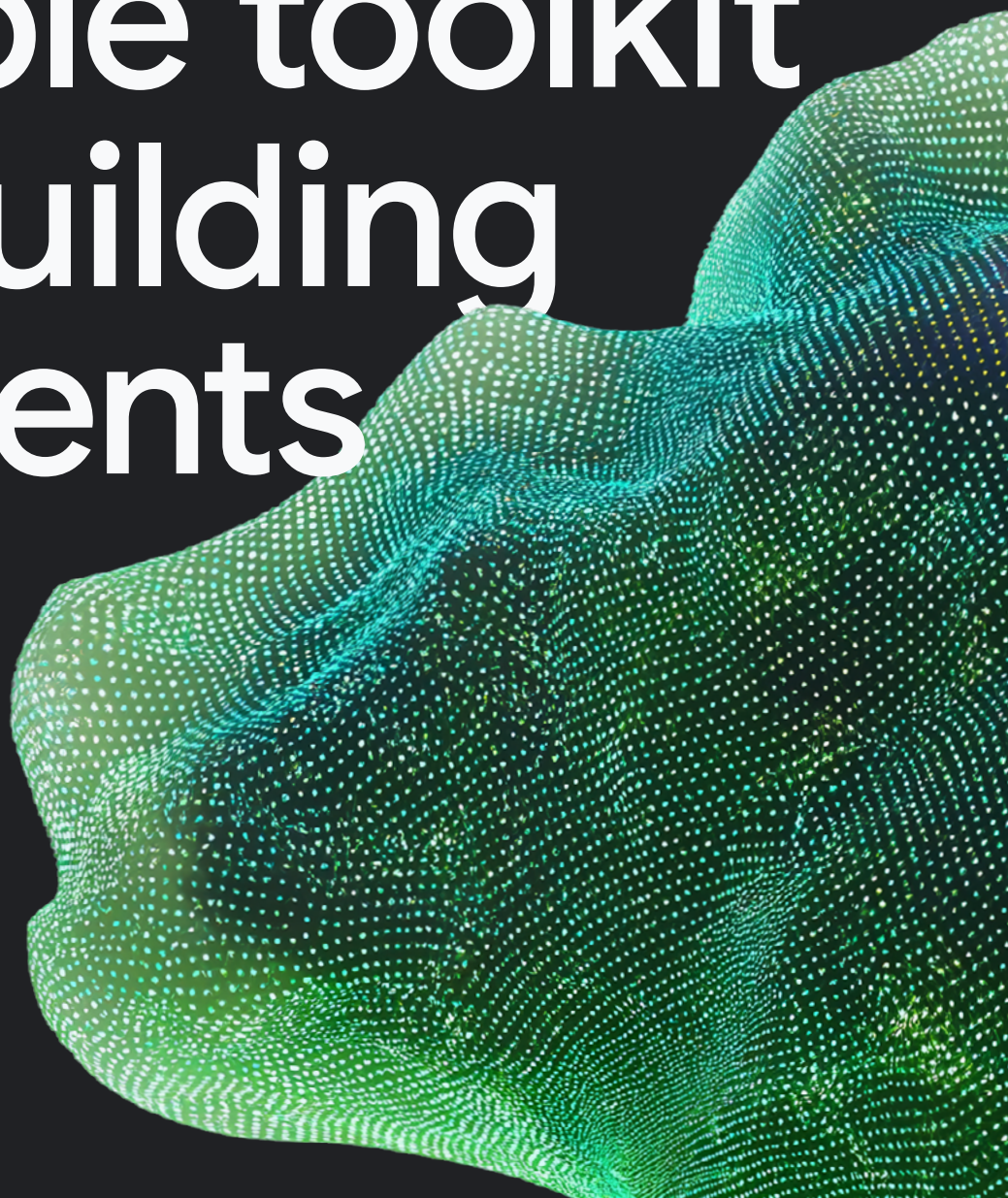
3 Govern

Security and trust are critical to enterprise adoption. Agent Builder ensures AI agents adhere to compliance mandates, with platform-enforced controls like identity, auditability, and security baked in.



1 Build

An open, flexible toolkit for building AI agents



When building AI agents in the enterprise, developers need to balance security, reliability, and efficiency. Vertex AI Agent Builder delivers. The open, unified platform is designed to give your developers more choice and flexibility on the build journey.

At its heart is ADK, Google Cloud's open-source toolkit for developing and deploying AI agents (explored in detail below). Developers can also use other open-source frameworks like LangGraph, CrewAI, AG2, and LlamaIndex—enabling them to build within their preferred ecosystem knowing that the agents they build will run seamlessly on Google Cloud's managed platform.

Agent Builder offers expansive model choice, with the option of using Gemini models or any other model through Model Garden. To ensure agents can collaborate across a diverse ecosystem and execute complex tasks across enterprise systems, developers can use Model Context Protocol (MCP) and Agent2Agent (A2A) protocol. They can also execute agent payments through the Agent Payments Protocol (AP2) or any commercial transaction through the higher order Universal Commerce Protocol (UCP), which is wired into Gemini and Google Search AI Mode.

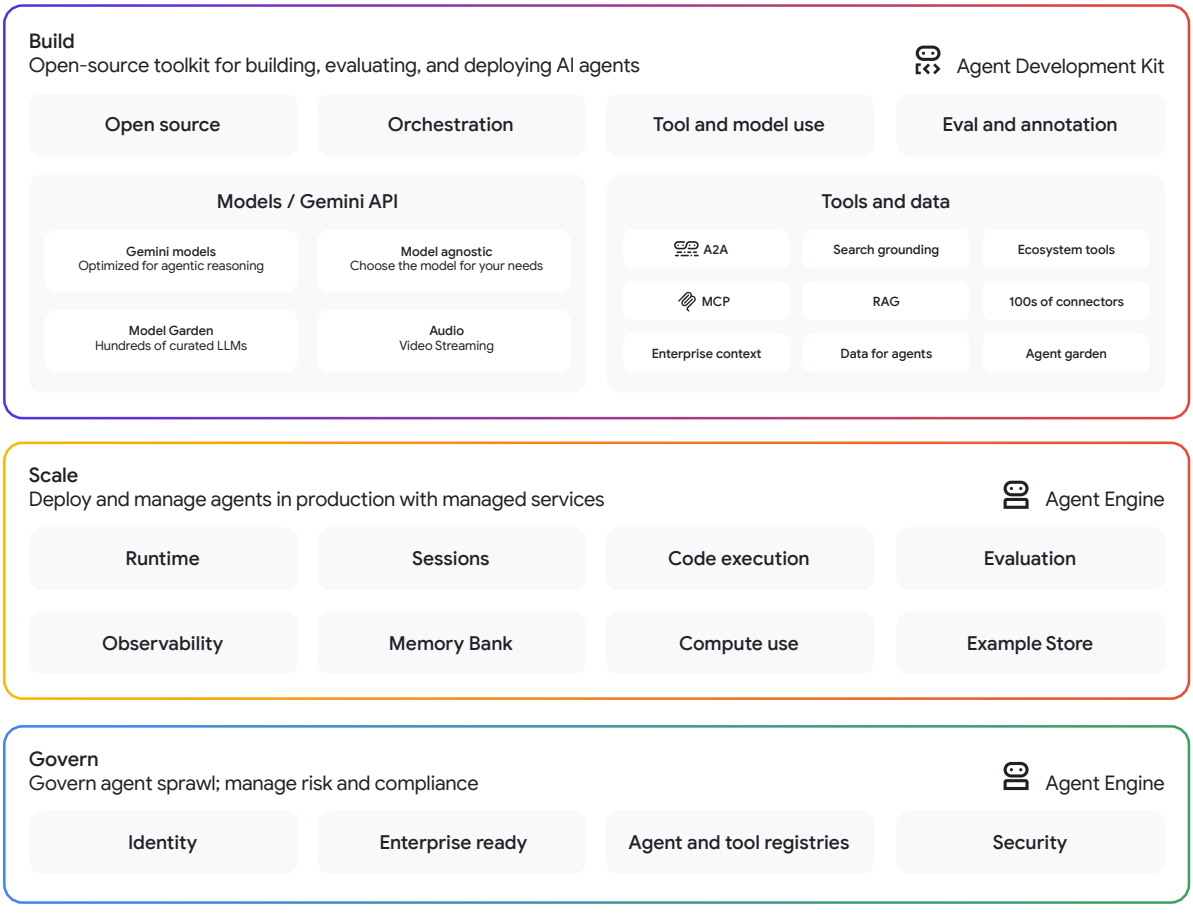
It all combines to deliver a seamless, secure, and robust approach to building enterprise-grade AI agents.

Core concepts

Think of the model as your agent's brain. Learn how to pick the right one to balance smarts, speed, and cost for your specific needs.

Explore models

Core components of Vertex AI Agent Builder



Develop multi-agent applications with ADK

ADK is an open-source toolkit to orchestrate agents, choose models, and manage agent development locally in Python, Java, Go, and TypeScript. It's designed for speed, enabling developers to get started in under 100 lines of code. It's also designed for flexibility—so you can choose the right foundation for your specific operational needs.

With ADK, you can:

1. Build complex, collaborative AI systems

ADK is multi-agent by design. It's easy to build highly specialized AI solutions that automate complex, multi-step workflows. And, with flexible orchestration (sequential, parallel, or dynamic), you can start with simple automations and evolve to highly adaptive systems.

2. Keep using the tools you already own

As an open ecosystem, ADK allows your agents to interact with all the tools you already own. You can connect your agents to productivity tools you already use, like Notion, Slack, an ERP, or a CRM, as well as tool frameworks like LangChain and LlamaIndex, or agent frameworks like LangGraph or CrewAI.

3. Ensure quality and reliability from day one

ADK's built-in observability and evaluation tools help you systematically test, debug, and benchmark your AI agents. This moves your process beyond simple "vibe-testing," allowing you to iterate and launch professional-grade agents quickly, while building user trust through proven reliability.

4. Scale AI with confidence

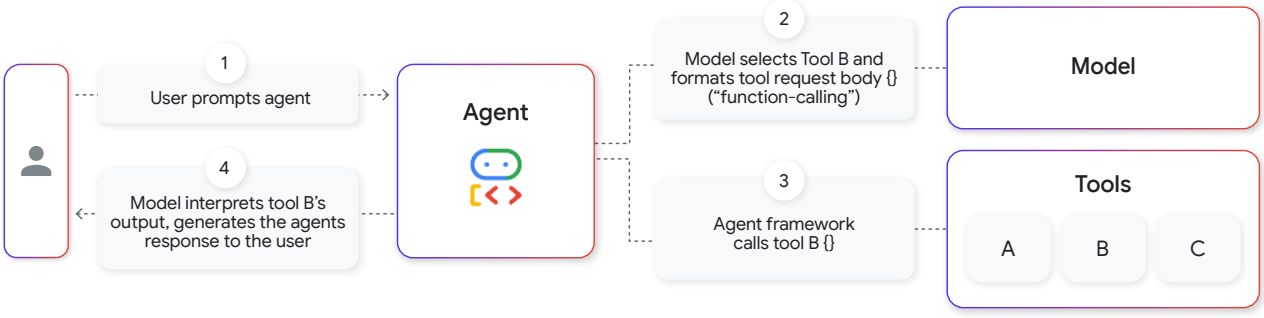
Your AI solutions should be designed to scale efficiently, preventing bottlenecks as your organization grows. ADK accelerates the path to production by bridging the gap between local development and deployment. Agents can deploy anywhere, from local testing to fully managed, auto-scaling runtimes like Vertex AI Agent Engine or Cloud Run.

Core concepts

Tools allow your agent to go beyond just "thinking" so it can actually "do." Learn how they help agents connect to external systems, fetch data, and complete real-world tasks.

Explore tools

Building complex workflows gets easier with ADK



ADK core: Selecting the agent architecture

A foundational step in building with ADK is selecting the right agent architecture. Distinct agent classes are designed for different execution patterns, and your choice will determine how your agent reasons and operates.

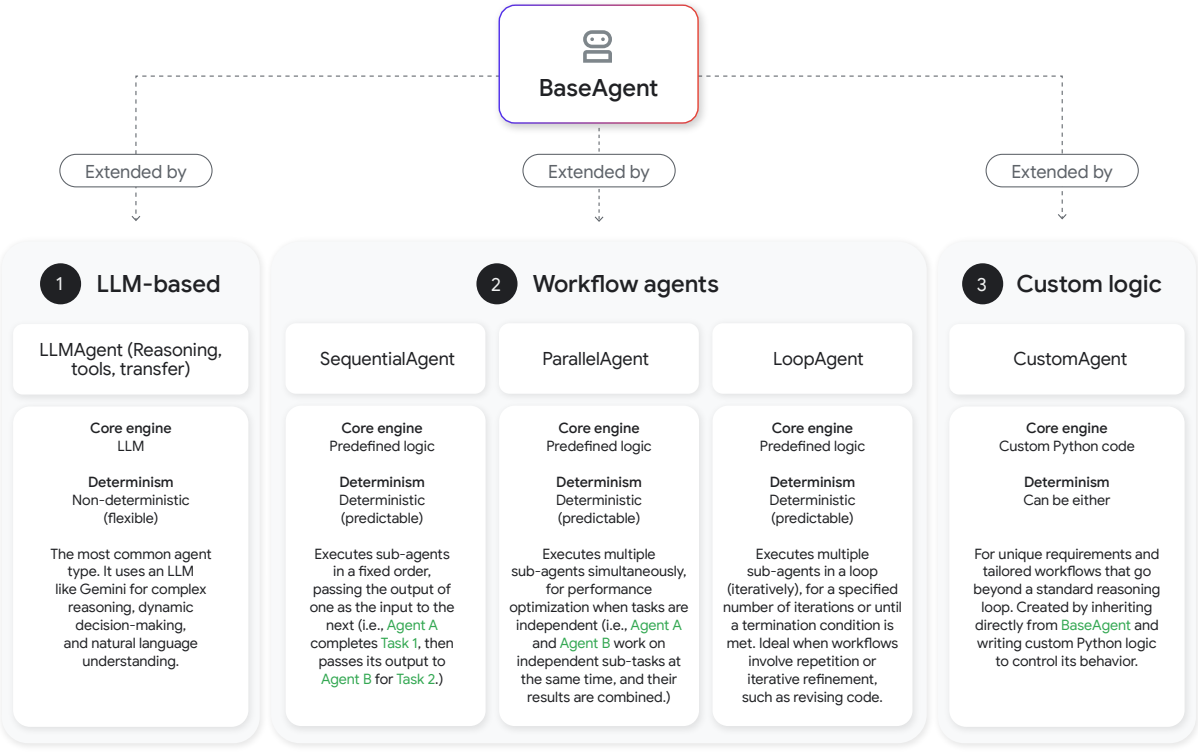
Developers typically face a trade-off between the flexible, non-deterministic power of an LLM and the predictable, deterministic control of hard-coded logic. Understanding the interplay between these agent classes is key to building robust and effective AI systems.

Core concepts

Agents use different types of data storage to manage information effectively. Learn how they balance long-term knowledge, short-term conversation context, and a permanent record of their actions.

Explore data architecture

ADK's agent types are organized into three categories



ADK orchestration: Implementing the ReAct loop

For an agent to act reliably and predictably in a high-stakes enterprise system, its tool must be defined by a clear, unassailable contract. ADK provides the core abstractions needed to implement the foundational ReAct paradigm in a structured way. The agent executes the loop, handling the transitions between fundamental stages:

- **Reasoning (thought):** The agent analyzes the user's prompt and its current internal state, then calls the underlying language model to form a hypothesis and determine the next best action.
- **Action (tool use and agent delegation):** When the agent decides to act, it can invoke a simple function or, for more complex tasks, delegate the work to another specialized sub-agent using a collaborative delegation pattern.
- **Observation:** The system automatically captures the information returned by the tool or sub-agent and passes it back to the reasoning engine. This output is then fed into the next reasoning step of the cycle.

By providing a native implementation of this essential pattern, ADK abstracts away the boilerplate code, so you can quickly translate the powerful concept of a ReAct loop into a working, multi-step agent.

Core concepts

Orchestration is the logic that guides an agent through a multi-step task. Learn how it helps an agent decide which tools to use and what order to follow to complete complex workflows.

Explore orchestration

ADK tools: Defining a framework for action

In ADK, an agent can use tools to perform actions beyond the native capabilities of its core reasoning model. These tools bridge the gap between internal reasoning and external execution, whether that involves making API calls, using MCP to access data, or collaborating with other agents via A2A.

For a model to use a tool correctly, its definition must serve as a clear and unambiguous API contract, composed of:

- **Function signature:** Using descriptive names for tools and their parameters. Explicit type hints are mandatory, as they provide the structural schema the model uses to generate valid arguments.
- **Docstring (the semantic core):** This is the model's primary source of semantic information. It must precisely define the tool's purpose, usage criteria, parameters, and expected return schema.
- **Return schema:** A tool must return a structured response. Best practice is to include a status key (e.g., 'success' or 'error') so the agent can reliably distinguish between successful outcomes and failures in its observation step and reason about how to proceed.
- **Stateful tools:** All tools have read or write access to a persistent context. Tools can be simple or sophisticated, and context is organized hierarchically in state, session, and memory. The agent acts as a compiled view on top of these primitives for every LLM call, allowing a developer to dynamically filter and format the most relevant information.

Pro tip

For more on tools, including code examples and advanced usage patterns, refer to the ADK documentation.

For best practices and examples of how to define parameters and tool schemas, structure effective prompts, and implement complex, multi-agent workflows, check out the ADK Samples repository.

Design an interoperable, open agent ecosystem

So your agents can collaborate across a diverse ecosystem, you need to ensure interoperability and prevent vendor lock-in. Agent Builder comes with native support for open protocols that enable agent-to-tool and agent-to-agent communication.

Standardize with MCP or OpenAPI

MCP is an emerging open standard for connecting AI and LLMs with external data sources and tools. You can plug your AI applications into various data sources and tools without the hassle of building custom point-to-point integrations for each one.

With ADK, your agents can:

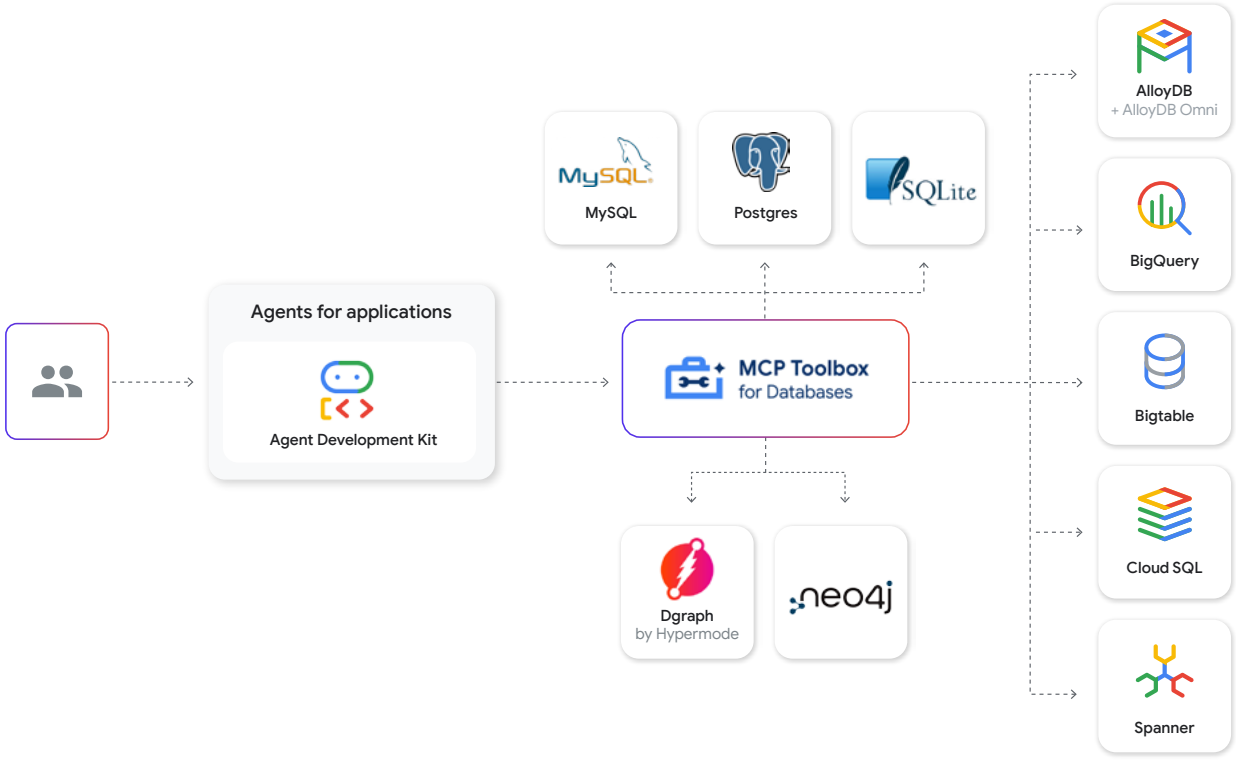
- **Consume external tools:** An ADK agent can act as an MCP client, allowing it to use any tool exposed by a third-party MCP server.
- **Expose native tools:** Developers can wrap their ADK tools in an MCP server, making them securely available to any other MCP-compliant agent or application.

Pro tip

Use the open-source MCP Toolbox for Databases to easily and securely connect your agents to a large array of popular data sources.

For production environments, utilize Google's fully-managed, remote MCP servers to upgrade existing API infrastructure into a unified, remote layer across Google Cloud services—eliminating the overhead of maintaining community-built servers while ensuring built-in security.

MCP is like a universal adapter for an agent's data sources and tools



Connect agents with the A2A protocol

Collaboration unlocks the true power of specialized agents. It's why Google champions the A2A protocol, an open standard that ensures your agents can discover, communicate with, and securely coordinate actions with other agents, regardless of who built them or what framework they use. This robust, standardized communication protocol transforms a collection of isolated agents into a truly interoperable ecosystem.

To facilitate this open ecosystem, the A2A protocol includes:

- **Agent card:** A digital "business card" (typically a JSON file at a well-known endpoint) that an agent uses to advertise its capabilities, endpoint URL, and authentication requirements—so other agents can easily discover it.
- **Task-oriented architecture:** Interactions are framed as "tasks." A client agent sends a task request to a server agent, which processes it and returns a response. An agent can act as both a client and a server.
- **Agnostic modalities:** A2A supports text, audio, and video communication, reflecting the evolving, multimodal nature of agent interactions.

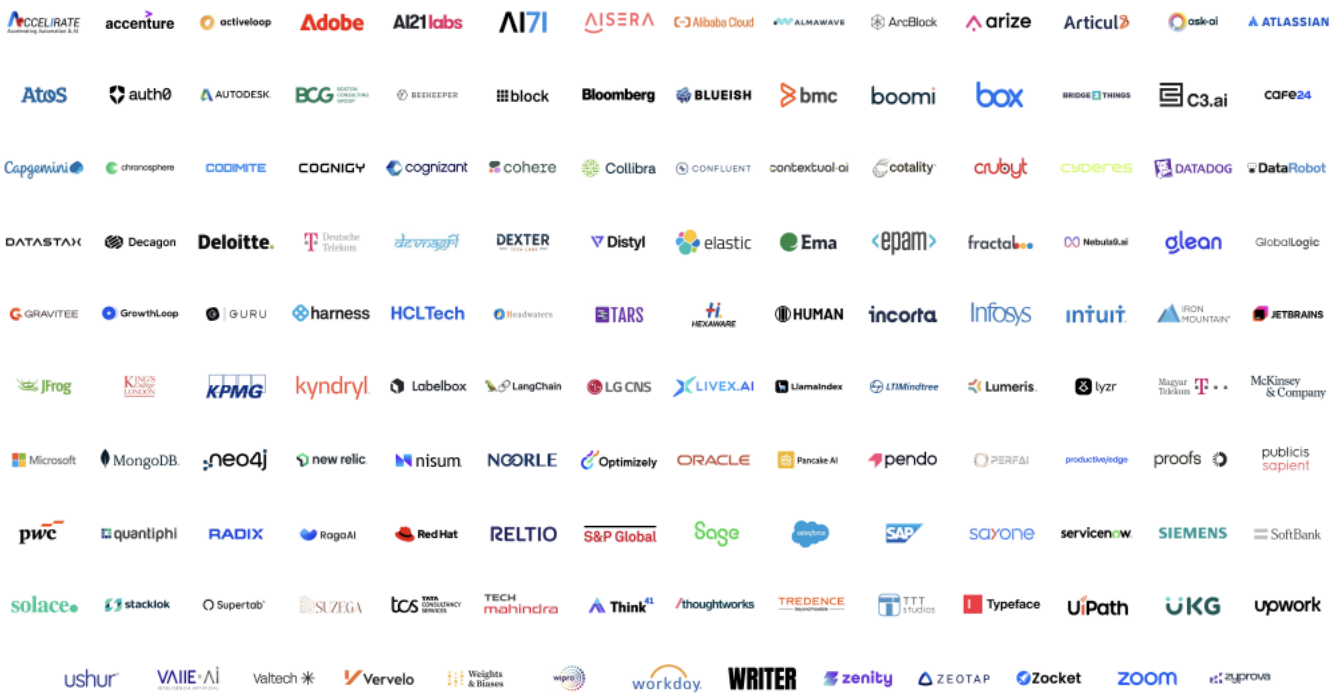
ADK agents can natively participate in this ecosystem. They expose a standard HTTP endpoint and an agent.json file, allowing them to be discovered and to communicate with any other A2A-compliant agent.

Pro tip

To get started, explore these A2A resources:

- A2A Project GitHub
- A2A protocol docs
- A2A protocol specification

The A2A protocol's rich partner ecosystem



Customer spotlight

How Box uses ADK and the A2A protocol to accelerate content development

Box is an Intelligent Content Management platform that enables organizations to fuel collaboration, manage the entire content lifecycle, secure critical content, and transform business workflows.

Situation

Critical business processes like compliance checks, contract management, and loan approvals are slowed by employees having to search and interpret vast amounts of information stored across documents in Box. This creates inefficiency and delays critical decisions.

Solution

Box introduced an A2A-enabled agent, built with Google's ADK and using Gemini. The agent connects directly to the Box Intelligent Content Management platform, allowing users to ask complex questions in natural language and receive summarized, contextual answers and insights from their documents instantly. The agent integrates with Gemini Enterprise and is available on Google Cloud Marketplace.

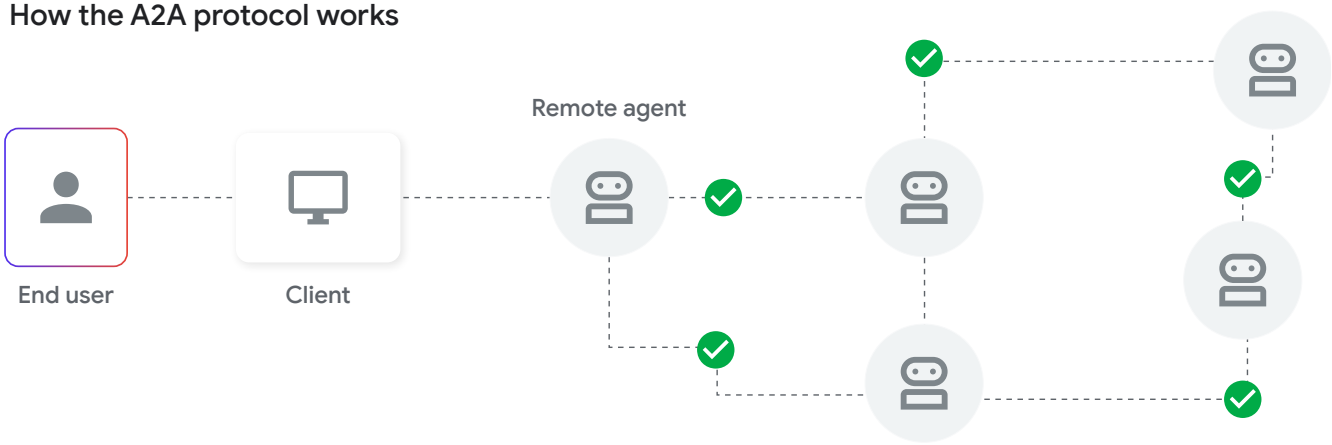
Impact

This dramatically accelerates content-centric workflows and improves the quality of decision-making. It lays the foundation for more advanced transactional agents that can govern, manage, and initiate processes like e-signatures and approvals, fundamentally transforming how work gets done in the company.

> "We're entering a new era where AI agents will transform how work gets done—and content is at the center of it all. With Box as the secure content layer and Google Cloud's A2A protocol enabling seamless collaboration across our ecosystem, we're unlocking powerful new ways to automate business processes, accelerate decision-making, and drive real outcomes for our customers."

box

How the A2A protocol works



Execute payments through AP2

AP2 is an open protocol developed with leading payments and technology companies to initiate and transact agent-led payments across platforms.

With support for different payment types—from credit and debit cards to stablecoins and real-time bank transfers—it helps establish a common foundation to securely authenticate, validate, and convey an agent's authority to transact.

The protocol can be used as an extension of A2A and MCP, establishing a payment-agnostic framework for users, merchants, and payments providers to transact with confidence across all types of payment methods.

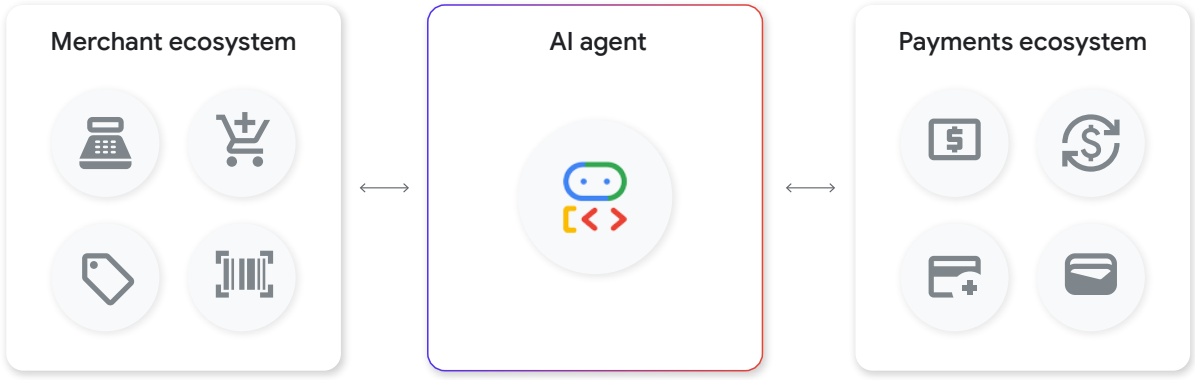
AP2 helps to address:

- **Authorization:** Proving that a user gave an agent the specific authority to make a particular purchase.
- **Authenticity:** Enabling a merchant to be sure that an agent's request accurately reflects the user's true intent.
- **Accountability:** Determining accountability if a fraudulent or incorrect transaction occurs.

Pro tip

Check out the public GitHub repository to see the complete technical specification, documentation, and reference implementations for AP2.

AP2 is an open protocol to handle agentic payments



2 Scale

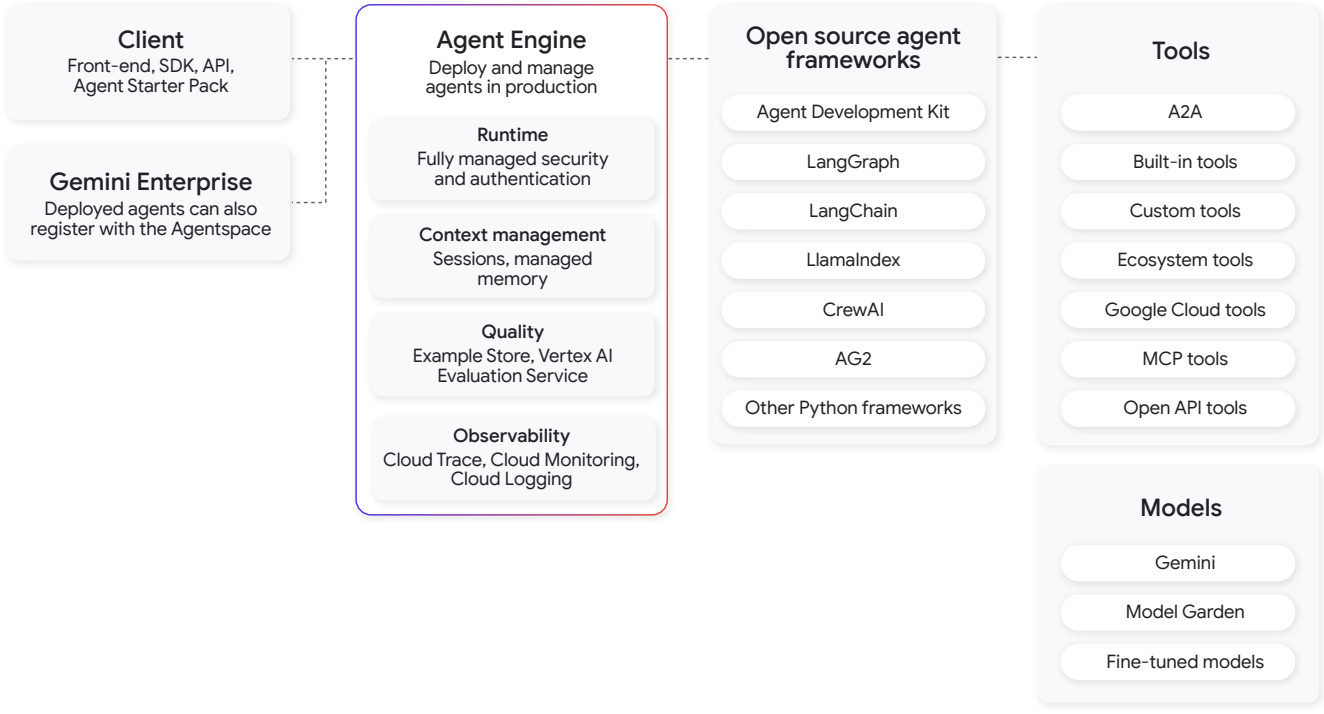
Reliable and efficient performance in production

Once AI agents have been built, Agent Builder helps developers scale them into production—with built-in testing, release management, and reliability at a global and secure scale.

These powerful capabilities are packaged up for easy access in Vertex AI Agent Engine. This set of managed services includes a managed runtime, sessions, memory, and sandboxes that developers can use individually or in combination.

Agent Engine provides the easiest and most direct path to a scalable, production-ready agent, abstracting the underlying infrastructure and freeing your engineers to focus on core agent logic rather than operational overhead.

Agent Engine helps enterprises scale AI agents in production



Deploy to the managed runtime

ADK is deployment-agnostic by design. The core agent logic you define in Python is decoupled from the serving infrastructure so you can develop and test locally, then deploy the same agent to various production environments. Here, we'll look at how to deploy to Agent Engine, which provides deep integration with the Vertex AI ecosystem for MLOps, monitoring, and security.

As a service designed for agentic workloads, Agent Engine provides several key benefits in deployment. Core capabilities include:

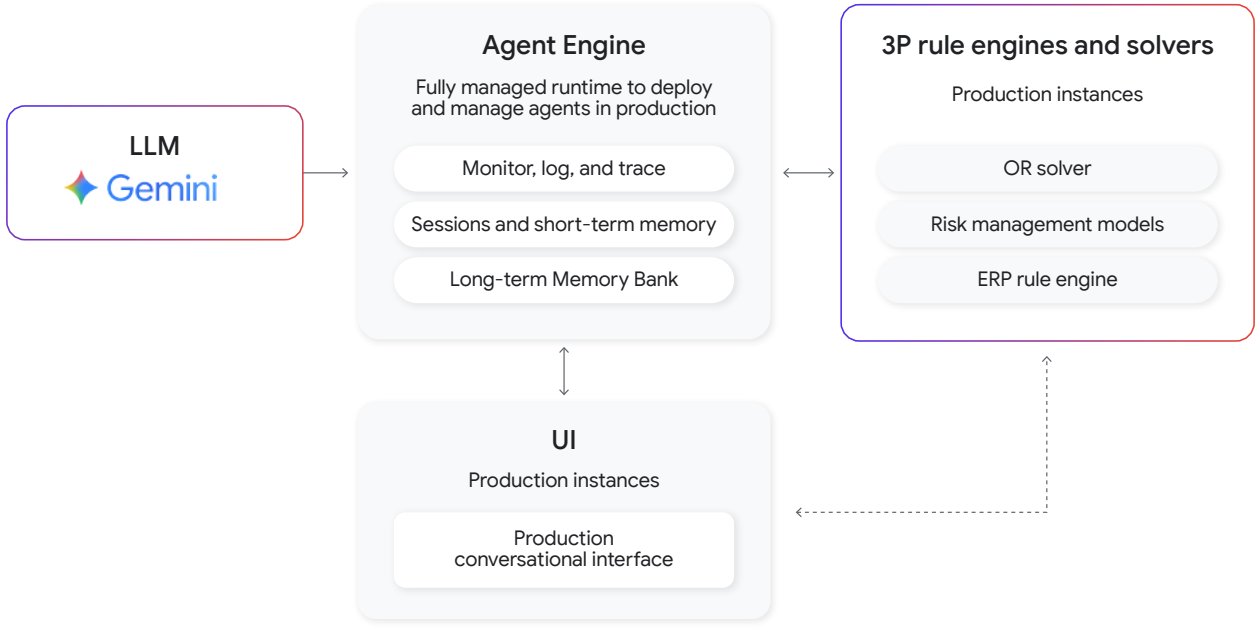
- **Automated scalability:** Automatically handles scaling to meet varying user loads.
- **Security and authentication:** Provides integrated identity and access management.
- **Framework agnostic:** Supports agents built with various frameworks, not just ADK.
- **Agent lifecycle management:** Provides APIs for creating, reading, updating, and deleting your deployed agents.

Core concepts

A runtime environment provides the stable foundation needed to move an agent from a prototype to a real-world application. Learn how it manages scaling, security, and monitoring to keep your agents running reliably at scale.

Explore runtime

System architecture for an agent engine built with Gemini



Measure and continuously improve performance

With AI agents successfully deployed in production, enterprises need robust ways to ensure their ongoing reliability and performance.

Agent Builder enables you to:

- **Test, monitor, and trace:** Example Store provides a centralized repository to store and dynamically serve few-shot examples. This allows developers to steer agent performance and improve accuracy on specific tasks without the need for additional model tuning or retraining.
- **Iteratively improve agent behavior:** Evaluation Service provides a feedback loop system to guide and refine agent behavior by enabling scaled review of agent responses and trajectories. This allows teams to benchmark performance against specific quality metrics and proactively identify failure modes before they reach production.

Execute complex, dynamic tasks safely

For autonomous AI agents, a sandbox environment isn't just a best practice—it is an essential component for scaling responsibly.

Agent Builder enables you to:

- **Run sandboxes:** With Code Execution, agents can run code in a secure, isolated, and managed sandbox environment. Sandboxes can be created in under a second, empowering the agent to solve complex computational problems, perform data analysis, and automate scriptable tasks directly.
- **Run computer operations:** With Computer Use, agents can interact with and control a computer's graphical interface—allowing it to automate tasks on virtually any software by mimicking human-like interactions with the screen, mouse, and keyboard.

3 Govern

Security and trust baked in

Strong governance is key to the success of AI agent adoption. Robust enterprise-grade controls are required to secure AI agents, along with proven methodologies for managing data privacy, ethics, and regulatory compliance.

Agent Builder plays a key role here—eliminating the "black box" problem of autonomous systems and enforcing accountability via a three-stage process of identity, auditability, and security.

It starts with a secure-by-design foundation built on Google Cloud, centered on Agent Identity. This gives every agent its own unique, native Google Cloud identity for granular permissions. On top of this, built-in observability gives you full traces, logs, and monitoring to see exactly what your agents are doing.

Finally, you can add enhanced security with integrated Google Cloud Security solutions. This includes Model Armor to protect your agents at runtime from threats like prompt injection, and our Security Command Center to get a unified view of your entire agent ecosystem security posture and detect agentic threats.

Enterprise-grade identity and auditability

Moving an AI agent from a prototype to production environment transforms it from a controlled experiment into a live system with real-world consequences. Strong governance is essential, providing a critical framework to establish accountability, ensure regulatory compliance, and mitigate risks.

Enterprises can use key tools within Agent Builder to maintain effective governance of agentic systems.

These include:

- **Agent Identity:** Native to Google Cloud, Agent Identity provides a unique, managed identity for each agent. It helps enterprises manage the complexity and risk of fragmented authentication by delivering an unambiguous audit trail and enables true least-privilege security.
- **Observability Suite:** This provides verifiable proof of agent behavior, with monitoring dashboards, detailed tracing, and an interactive playground for debugging. You can use it to proactively find policy violations, conduct in-depth forensic investigations, and simplify compliance auditing.
- **Agent and tool registry:** This is a centralized system to manage, version, and discover approved agents and tools. It promotes reusability, consistency, and control across your organization, ensuring developers use vetted and compliant components.

Customer spotlight

> **Geotab uses Vertex AI Agent Builder to rapidly build and deploy agents in production. Specifically, we use Google's Agent Development Kit (ADK) as the framework for our AI Agent Center of Excellence. It provides the flexibility to orchestrate various frameworks under a single, governable path to production, while offering an exceptional developer experience that dramatically accelerates our build-test-deploy cycle. For Geotab, ADK is the foundation that allows us to rapidly and safely scale our agentic AI solutions across the enterprise."**

Mike Branch
Vice President, Data & Analytics, GeoTab

Enhanced security features

Scaling your AI agents increases your risk of malicious attack. Implementing robust security measures is essential to protect the agent's integrity and ensure the trustworthiness of the system at scale.

Key security features within Agent Builder include Model Armor and Security Command Center.

Model Armor

To enhance the security and safety of your AI applications, Model Armor proactively screens LLM prompts and responses, protects against threats like prompt injection and data exfiltration, and ensures responsible AI practices.

As well as helping to defend against threats, Model Armor offers a variety of filters to help you provide safe and secure AI models, including:

- Responsible AI safety filter
- Prompt injection and jailbreak detection
- Sensitive data protection
- Malicious URL detection

Security Command Center

With visibility across your inventory of agentic assets and associated risks, Security Command Center delivers enterprise-grade security services such as continuous vulnerability scanning, real-time threat detection, and security posture monitoring for your agents. It gives you a unified view of agentic AI security, helping you identify misconfigurations and high risk issues, and respond to threats.

Use cases

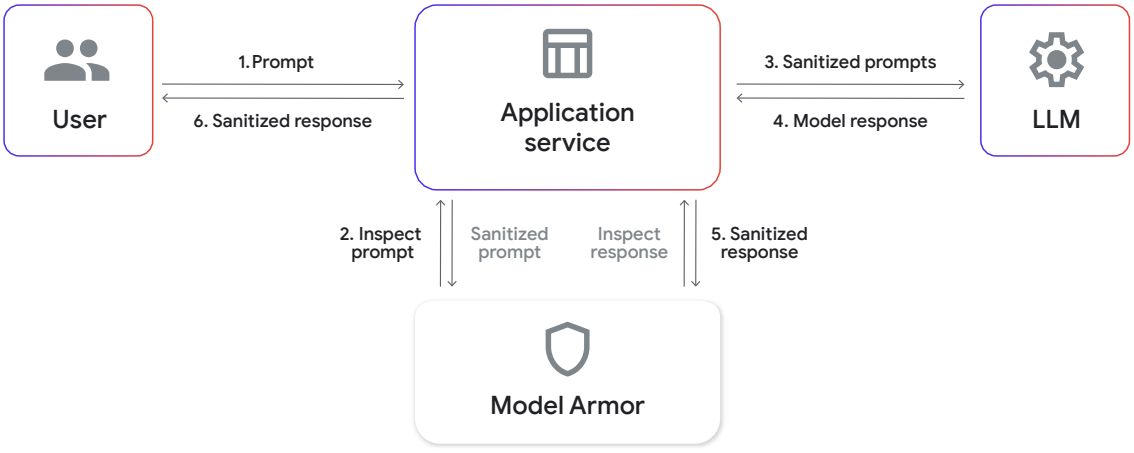
Security

- Mitigate the risk of leaking sensitive intellectual property (IP) and personally identifiable information (PII) from being included in LLM prompts or responses.
- Protect against prompt injection and jailbreak attacks, preventing malicious actors from manipulating AI systems to perform unintended actions.
- Scan text in PDFs for sensitive or malicious content.

Safety and responsible AI

- Prevent your chatbot from recommending competitor solutions, maintaining brand integrity and customer loyalty.
- Organizations can filter social media posts generated by their AI containing harmful messaging, such as dangerous or hateful content.

An application using Model Armor to protect an LLM and a user



Conclusion

We have entered a new era where AI agents are poised to transform enterprise workflows and customer experiences. The path to production is not through complexity—rather, it's through a platform built on trust and transparency.

Vertex AI Agent Builder is the definitive, integrated solution designed to eliminate the choice between performance and protection. It delivers a unique blend of openness and robust platform capabilities that ensures your agents perform reliably, securely, and seamlessly at global scale.

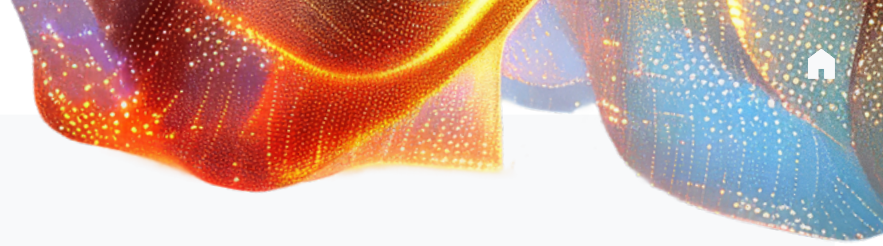
This guide has demonstrated that the building blocks for designing tomorrow's enterprise-grade, multi-agent systems are here. The challenge is no longer if agents deliver value, but how to deploy them with enterprise confidence.

Your next step is strategic, not technical

To accelerate innovation and establish a foundation of unambiguous accountability for your agent fleet, engage our specialists today. We'll show you how the Vertex AI platform can help you:

- **Govern and mitigate risk:** Review how Agent Identity and the integrated Observability Suite establish a permanent, verifiable audit trail for compliance and forensic investigations.
- **Scale and optimize cost:** Discuss different capacity reservation options (such as Provisioned Throughput) to ensure guaranteed model capacity for your entire agent fleet (including popular open-source models). This is essential for delivering consistent performance and predictable expenditure under your largest financial commitments.

Don't just build a prototype. Build a protected, production-ready system on Agent Builder.



Resources

- AdkApp: Develop and deploy agents on Vertex AI Agent Engine.
- Advent of Agents: A 25-day educational journey to master AI agent development—from building local prototypes with ADK to deploying production-ready agents on Agent Engine.
- Agent Development Kit (ADK): ADK is a flexible and modular framework for developing and deploying AI agents.
- Agent2Agent (A2A): An open protocol enabling communication and interoperability between opaque agentic applications.
- Agent Starter Pack: Get production-ready agents on Google Cloud, faster. Go from idea to deployment faster with pre-built templates and tools.
- BigQuery: BigQuery is Google Cloud's fully managed, petabyte-scale, and cost-effective analytics data warehouse that lets you run analytics over vast amounts of data in near real time.
- Check grounding API: As part of your RAG experience in AI Applications, you can check grounding to determine how grounded a piece of text (called an "answer candidate") is in a given set of reference texts (called "facts").
- Cloud Functions API: This API manages lightweight user-provided functions executed in response to events.
- Cloud Run: Run frontend and backend services, batch jobs, host LLMs, and queue processing workloads without the need to manage infrastructure.
- Cloud Spanner: Acts as a globally consistent backend for mission-critical agent actions. In advanced implementations, a tool representing a core business process (e.g., process_global_order) would trigger a transaction in a Spanner-backed system to ensure global integrity.
- Cloud Storage: Acts as the agent's durable file system, and is used as the source of truth for raw documents (e.g., PDFs, images) that are then indexed by services like Vertex AI Search.
- Cloud Storage bucket: The basic containers that hold your data. Everything that you store in Cloud Storage must be contained in a bucket.
- Cloud SQL: Serves as the agent's reliable system of record. Tools log their actions to Cloud SQL, creating a permanent, ACID-compliant audit trail for every important agent-driven action.
- Colab Enterprise: A collaborative, managed notebook environment with the security and compliance capabilities of Google Cloud.
- Example Store: Store and dynamically retrieve few-shot examples.
- Firestore: A highly scalable NoSQL database for your web and mobile applications.
- Gemini 2.5 Flash: Gemini 2.5 Flash is designed to control the trade-off between quality, cost, and speed.
- Gemini 3 Pro Image (Nano Banana Pro): Gemini can generate and process images conversationally. You can prompt Gemini with text, images, or a combination of both allowing you to create, edit, and iterate on visuals with unprecedented control.
- Gemini 3 Pro: Our most advanced reasoning Gemini model, capable of solving complex problems.
- Gemini 3 Flash: Our latest model optimized for speed and high-frequency agentic workflows. It combines Pro-grade reasoning with next-generation efficiency, delivering elite performance at a fraction of the cost.
- Gemma: A collection of lightweight, state-of-the-art open models built from the same technology that powers our Gemini models.
- Gen AI evaluation service: Evaluate any generative model or application and benchmark the evaluation results against your own judgment, using your own evaluation criteria.
- Google Cloud Observability: Observability services that help you to understand the behavior, health, and performance of your applications.
- Google Kubernetes Engine (GKE): The most scalable and fully automated Kubernetes service. Put your containers on autopilot and securely run your enterprise workloads at scale – with little to no Kubernetes expertise required.
- GraphRAG: Combines knowledge graphs with Retrieval-Augmented Generation (RAG) to enhance the accuracy, context, and explainability of LLMs.

- Imagen: Brings Google's state-of-the-art image generative AI capabilities to application developers.
- MCP Toolbox for Databases: An open source MCP server that helps you build generative AI tools so that your agents can access data in your database.
- Memorystore: Provides a high-speed cache for the agent. It stores the results of frequent or expensive tool calls, drastically reducing latency and operational costs.
- Model Context Protocol (MCP): An open source protocol that standardizes how applications provide context to LLMs.
- Model evaluation in Vertex AI: A predictive AI evaluation service that lets you evaluate model performance across specific use cases.
- Model Garden on Vertex AI: A single, centralized platform to discover, customize, and deploy a wide variety of models from Google and Google partners.
- Model tuning: A crucial process in adapting Gemini to perform specific tasks with greater precision and accuracy.
- ReAct: Orchestration with a ReAct (reasoning + action) agent involves a multi-turn interaction between an application and a model (or models) where the agent manages conversations, transactions, and LLM instructions.
- Responsible AI: To aid developers, Vertex AI Studio has built-in content filtering, and our generative AI APIs have safety attribute scoring to help customers test Google's safety filters and define confidence thresholds that are right for their use case and business.
- Retrieval-Augmented Generation: RAG is an AI framework that combines the strengths of traditional information retrieval systems (such as search and databases) with the capabilities of generative LLMs.
- Vector database: Any database that allows you to store, index, and queryvector embeddings, or numerical representations of unstructured data, such as text, images, or audio.
- Veo: A tool to generate new videos from a text prompt or an image prompt.
- Vertex AI Agent Engine: A set of services that enables developers to deploy, manage, and scale AI agents in production.
- Vertex AI Memory Bank: A managed service on Agent Engine that dynamically generates and retrieves long-term, personalized memories based on users' conversations. It can implement automated or agent-direct distillation.
- Vertex AI notebooks: Access every capability in Vertex AI Platform to work across the entire data science workflow–from data exploration to prototype to production.
- Vertex AI Platform: A fully managed, unified AI development platform for building and using generative AI.
- Vertex AI RAG Engine: A data framework for developing context-augmented LLM applications.
- Vertex AI Search: Brings together the power of deep information retrieval, state-of-the-art natural language processing, and the latest in LLM processing to understand user intent and return the most relevant results for the user.
- Vertex AI Studio: Streamline your foundation model workflows with Vertex AI Studio. Rapidly prototype, refine, and seamlessly deploy models to your applications.

Google Cloud

Questions? Ask our enterprise team.

Contact us