MockCert

Study guide | Google Cloud | Professional Agentic Architect

Google Cloud Professional Agentic Architect

Notes for every domain of the Google Cloud Professional Agentic Architect exam, organized by the official exam guide's domains and weights.

By MockCert about 54 min read Updated November 18, 2026

CONTENTS

DOMAIN 1 · 13%
Building agents using low-code tools 2

DOMAIN 2 · 17%
Using coding agents for application development 9

DOMAIN 3 · 33%
Developing custom agents 16

DOMAIN 4 · 22%
Evaluating and deploying agentic workflows 27

DOMAIN 5 · 15%
Securing and governing agentic workflows 34

MockCert is an independent certification preparation platform. It is not affiliated with, endorsed by or operated by Google Cloud. Exam domains and weights follow the official exam guide. These notes were last updated on November 18, 2026; the notes at mockcert.com are kept up to date as the exam changes.

Domain 1 | 13% of the exam | 9 min read

Building agents using low-code tools

Overview

This domain is about 13% of the exam, roughly 8 of its 60 questions.

Every point below comes from Google Cloud's official documentation; each section lists its sources.

Configuring agentic workflows and behavior using low-code tools

Start here

THINK FIRST

In a CX Agent Studio app, when should an agent steer the conversation itself, and when should it hand off to a deterministic flow?

Why it matters: Questions describe a conversation design and ask how to split work among agents, flows, transfer rules, callbacks and instructions.

LLM-driven agents steer the conversation and route between agents and flows. Flows run strict, step-by-step business logic as sealed black boxes. Instructions and model settings shape how each agent behaves.

Key terms

Term	Meaning
Steering agent	A CX Agent Studio agent that routes the conversation between other agents and flows.
Flow-based agent	An agent that hands the conversation to a Dialogflow CX flow until the flow reaches END_SESSION.
Transfer rule	A rule on an agent that moves the conversation to a target agent, such as from parent to child.
Callback	A step the agent runs at set points, such as before a model call; a list of callbacks runs in order.
Global instruction	An app-level instruction that applies to every agent and can set one stable identity or personality.

Agents steer the conversation

CX Agent Studio agents act as the steering layer. The root agent is the entry point of the app. Steering agents then route between other agents and flows. Their LLMs, not flows, should decide what the user wants.

EXAM TRAP

Do not build intent classification and detection into a flow. Use LLMs for that job instead.

When a flow fits

Flows fit highly deterministic business logic. Pick a flow for strict logic that callbacks cannot handle, such as collecting and validating data in a fixed sequence. The agent treats the flow as a black box. It passes everything the flow needs through explicit input parameters.

Handing off to a flow

A flow-based agent keeps control until the flow reaches END_SESSION. At that point, the conversation returns to the agent that started the hand-off. User input reaches the flow through the DetectIntent API method. When you import a flow-based agent, the flow environment defaults to draft.

EXAM TRAP

Do not assume the agent application can call a flow out of the box. It needs you to give the Customer Engagement Suite Service Agent the Dialogflow API Client role.

Passing data in and out

Variable mappings carry data across the flow hand-off. Input mapping turns agent variables into flow session parameters, and output mapping does the reverse. By default, every parameter a flow gathers moves to session scope, so later flows can read it.

Input and output variable mapping compared

Feature	Input mapping	Output mapping
Direction	Agent application to flow	Flow to agent application
Arrives as	Session parameters in the flow	Variables in the agent application

Transfer rules

Transfer rules move the conversation between parent and child agents. A deterministic rule transfers to the target agent at once when its condition is met. That condition can be a Python code block, and the transfer happens when it evaluates to true.

The two transfer directions compared

Feature	PARENT_TO_CHILD	CHILD_TO_PARENT
Moves from	Parent agent	Child agent
Moves to	Child agent	Parent agent

EXAM TRAP

Matching rules do not all fire. When several transfer rules match, only the first rule in the list is used.

Callback order

Callbacks run one at a time, in list order. If a callback returns an overridden response, the remaining callbacks are skipped. A callback marked disabled is ignored by the agent.

EXAM TRAP

Before-model callbacks do not run just once per turn. They run before every model call, so they can run several times.

Writing agent instructions

An agent's instruction field guides how its LLM behaves. A global instruction covers every agent in the app, which keeps one identity or personality. For a built-in system tool, the instructions should say when to use it and supply its arguments.

EXAM TRAP

You cannot edit a system tool. You shape how the agent uses it through instructions and tool arguments.

Model settings

App model settings are defaults that agents can override. An agent with no model set inherits the model of its parent agent. A lower temperature makes answers more predictable, while a higher one makes them more creative.

Key takeaways

- Agents steer and route; flows run strict, deterministic logic as black boxes.
- A flow-based agent keeps the conversation until END_SESSION, then hands it back.
- When several transfer rules match, only the first one in the list is used.
- A global instruction gives every agent in the app one stable identity or personality.

Sources: Flow-based agents | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · MCP Tools Reference: ces.googleapis.com | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · MCP Tools Reference: ces.googleapis.com | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · MCP Tools Reference: ces.googleapis.com | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · System tools | CX Agent Studio | Google Cloud Documentation (opens in a new tab)

Connecting enterprise data to Gemini Enterprise

Start here

THINK FIRST

How would you let an agent query enterprise systems safely and also search images, audio or text by meaning?

Why it matters: Questions describe an enterprise data need and ask which connector, identity, search or endpoint option fits it.

Agents reach structured business data through connector tools that run actions on a connection, under their own identity. Unstructured data such as text and images becomes embeddings that Vector Search compares by meaning.

Key terms

Term	Meaning
Connector tool	An agent tool made of one Integration Connectors connection plus one or more actions.
Agent Identity	A unique SPIFFE-formatted ID that gets its own IAM permissions instead of a shared service account.
Hybrid search	Vector Search that mixes dense and sparse embeddings, using a weight you set for each type.
Restrict	A Vector Search feature, also called filtering, that limits a search to part of the index with Boolean rules.
Endpoint	Where a Vector Search index is deployed: a public endpoint or a private one inside a VPC.

Connector tools

A connector tool pairs one connection with one or more actions. In CX Agent Studio, the connection comes from Integration Connectors. Each connection exposes entities that match the objects of the connected system. For BigQuery an entity is a table; for Salesforce it is an object such as Order or Case.

Querying and trimming output

Many connectors offer the `ExecuteCustomQuery` action for SQL queries. This action lets the agent run SQL directly against the connected system. To keep responses small, choose a set of output fields on the tool. A smaller response only carries the fields the agent needs.

EXAM TRAP

Returning every field is a poor choice when token limits are a concern. Picking output fields shrinks the tool response and exposes only relevant fields.

Secure agent access

Give each agent its own Agent Identity, not a shared service account. Administrators grant that identity specific IAM permissions, such as on a BigQuery dataset. Agent Identity Auth Manager handles OAuth and refresh tokens, so developers never manage those credentials. If a connection allows authentication override, a connector tool can pass credentials from session parameters.

EXAM TRAP

A shared service account is the option Agent Identity replaces. Agent Identity gives one agent its own granular permissions in IAM.

Grounding with Google Search

Excluded domains always win over other domain settings. This applies in the CX Agent Studio Google Search tool. An excluded domain stays out even if it is also a preferred domain or a URL for context. When a citation sets `requires_attribution` to true, your client app must show attribution and link to the cited URI.

Domain settings in the Google Search tool

Feature	Excluded domains	Preferred domains
What it does	Removes the domain from results	Restricts results to the listed domains
Precedence	Highest; beats preferred domains and URLs for context	Overridden by an exclusion
Limit	Not stated	At most **20** domains

EXAM TRAP

Listing a domain as preferred does not bring back a domain that is also excluded. Exclusion has the highest precedence.

Reranking results

The Agent Search ranking API reorders documents by relevance. It scores each document against a query with a pre-trained language model. Use it when you already have candidate documents and need them in precise relevance order.

Embedding multimodal data

Clean and preprocess data before you generate embeddings. The data must be in a format the embedding model can use. Embeddings are not limited to text, and Vector Search can compare text and

image data together. Next, pick a ready-to-use model or a custom one.

Ready-to-use and custom embedding models compared

Feature	Ready-to-use model	Custom model
Best fit	Standard semantic matching of text to text or text to images	Matching that needs a model tuned on your data
Training	None; no training or tuning needed	Must be trained on your data, which can take a long time
Typical choice	Text embeddings, from dense embedding models	Tuned model; a pretrained Model Garden model skips training

EXAM TRAP

A ready-to-use model does not need training for standard text or image matching. Only a custom model must be trained on your data.

Search types and filters

Vector Search supports dense, sparse and hybrid search. Hybrid search blends dense and sparse embeddings by the weight you give each type. To search only part of the index, use Restrict, which applies Boolean rules and is also called filtering.

Choosing an endpoint

Pick the endpoint type before you create the index. A public endpoint needs no network setup, but its latency is slightly higher. A private endpoint in a VPC requires network setup first and supports **two** private network types.

Vector Search endpoint options compared

Feature	Public endpoint	VPC Network Peering	Private Service Connect
Network setup	None needed	Required first	Required first
Main benefit	Faster to set up and easier to maintain	Reduced network latency	Private use across VPCs of other teams, projects or organizations

EXAM TRAP

A public endpoint is not the low-latency choice. It is simpler to run, but private networks have slightly lower latency.

Key takeaways

- A connector tool is **one** connection plus one or more actions; entities map to tables or objects.
- Agent Identity replaces shared service accounts with granular IAM permissions for each agent.
- In the Google Search tool, excluded domains beat preferred domains, which are capped at **20**.

- Ready-to-use embedding models need no training; custom models must be trained on your data.
- Choose a public or private Vector Search endpoint before creating the index.

Sources: Integration Connector ??? | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · Agents overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Vector Search | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Before you begin | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Google Search tool | CX Agent Studio | Google Cloud Documentation (opens in a new tab)

Domain 2 | 17% of the exam | 9 min read

Using coding agents for application development

Overview

This domain is about 17% of the exam, roughly 10 of its 60 questions.

Every point below comes from Google Cloud's official documentation; each section lists its sources.

Using coding agents effectively

Start here

THINK FIRST

How do you give a coding agent useful tools while keeping its code, credentials and clusters safe?

Why it matters: Questions describe a coding agent setup and ask which tool setting, sandbox control or patching approach fits it.

An agent's tools carry risk hints that clients use to decide when to ask the user. Untrusted workloads run in isolated sandboxes, and Workload Identity Federation is the most secure way to give their Pods access. GKE then keeps clusters patched through staged release channels.

Key terms

Term	Meaning
Tool annotation	A hint sent to MCP clients that describes a tool's basic risk, such as whether it is read-only.
GKE Sandbox	A GKE feature, built on gVisor, that isolates untrusted workloads in sandboxes on the node.
Workload Identity Federation	Lets a Kubernetes service account run as an IAM service account, so its Pods get that account's permissions.
Network policy	A namespace rule that limits which Pods can send and receive traffic, based on Pod labels.
Soak time	The time a patch version spends in earlier release channels before it reaches later ones.

Tool annotations signal risk

Annotations tell MCP clients how risky a tool is. Most clients treat these hints as untrusted. Still, a client can use them to decide when to show the user a confirmation prompt before a tool runs.

Annotation defaults

With no annotation, a tool is assumed able to cause harm. The `destructiveHint` defaults to true and the `readOnlyHint` defaults to false. A tool must say otherwise to look safe. For `openWorldHint`, a web search tool is open world, while a memory tool is not.

Two MCP annotations compared

Feature	destructiveHint	readOnlyHint
Meaning when true	The tool can perform destructive actions	The tool does not modify its environment
Default value	true	false
Value on the list_agents tool	false	true

EXAM TRAP

A tool with no annotations is not treated as safe. Its destructive hint defaults to true and its read-only hint defaults to false.

Calling an MCP tool

A client runs a tool with the JSON-RPC tools/call method. The request names the tool inside `params`. For example, the read-only `list_agents` tool is called this way.

Example: Asks an MCP server to run the list_agents tool.

```
{
  "method": "tools/call",
  "params": { "name": "list_agents" }
}
```

- `"name": "list_agents"`: the tool to run

Sandboxing untrusted workloads

Protect the node's operating system from untrusted Pods. That is the good practice for clusters running unknown or untrusted workloads. Separately, you can enable GKE Sandbox on a cluster to isolate untrusted workloads in sandboxes on the node. It is built on the open gVisor project.

Credentials inside a sandbox

Workload Identity Federation is the most secure way to grant Pod access. It also works with GKE Sandbox, unlike the node's Compute Engine service account. Giving Pods a service account key is strongly discouraged, because keys are hard to manage securely. Autopilot clusters always have Workload Identity

Federation turned on.

Two ways for Pods to get Google Cloud credentials

Feature	Workload Identity Federation	Node service account
Works with GKE Sandbox	Yes	No, GKE Sandbox blocks the Compute Engine metadata server
How Pods get permissions	A Kubernetes service account runs as an IAM service account	From the node's Compute Engine service account

EXAM TRAP

In Standard clusters, Workload Identity Federation is not on by default. Pods can reach instance metadata until you enable it manually.

Network policy defaults

With no network policy, all Pod traffic is allowed. Ingress and egress flow freely into and out of every Pod. Once you apply a network policy in a namespace, traffic to and from Pods that don't match its labels is dropped.

EXAM TRAP

A network policy does not only affect the Pods it names. Pods in that namespace that don't match its labels lose all traffic.

How GKE patches clusters

GKE upgrades control planes and nodes automatically. Google scans Kubernetes and Google-managed containers and starts patching when a fix exists. Patches reach the Rapid channel first, then Regular, then Stable. Some fixes need only a control plane upgrade, while others also need node upgrades.

EXAM TRAP

An automatic control plane upgrade does not fix every vulnerability. Some fixes also require node upgrades.

Speeding up security patches

Accelerated patch auto-upgrades skip the in-channel soak time. First, test the patch on other GKE clusters with rollout sequencing before production gets it. In Regular or Stable, you can then manually upgrade to that exact patch version. Google recommends upgrading clusters at least **once a month**.

Key takeaways

- MCP clients treat tool annotations as untrusted hints, but can use them to decide when to ask the user.
- A tool is assumed destructive and not read-only unless its annotations say otherwise.
- GKE Sandbox isolates untrusted workloads and blocks the metadata server, so use Workload Identity

Federation for credentials.

- Applying a network policy drops traffic for Pods in its namespace that don't match its labels.
- Patches move from Rapid to Regular to Stable, and accelerated auto-upgrades skip that soak time.

Sources: MCP Tools Reference: ces.googleapis.com | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · GKE security overview | Google Cloud Documentation (opens in a new tab) · Security patching | GKE security | Google Cloud Documentation (opens in a new tab) · GKE overview | Google Kubernetes Engine (GKE) | Google Cloud Documentation (opens in a new tab)

Customizing coding agents for enterprise workflows

Start here

THINK FIRST

How would you package a new agent skill, then build, govern, and safely stop the agents that use it?

Why it matters: Questions describe an enterprise need and ask which skill, registry, CLI, or governance feature meets it.

Skills are reusable packages that extend what an agent can do, stored in a private registry. Deployed agents move through a set lifecycle, from scaffold to observe. Gateway policies govern what deployed agents may do.

Key terms

Term	Meaning
Skill	A self-contained package of instructions and executable code that extends an agent's capabilities.
Skill revision	An immutable snapshot of one skill version, with a name, a description, and a link to its parent skill.
Agents CLI	A tool that walks agents through the lifecycle: scaffold, evaluate, deploy, publish, and observe.
Agent Registry	The central catalog for governing, discovering, and searching agents, tools, MCP servers, and skills.
Semantic governance policy	A plain-language rule that controls how an agent uses tools at runtime.

Skills and Skill Registry

Skill Registry stores agent skills privately and securely. Each skill is a self-contained package that adds a capability to an agent. A built-in skill named gcp-skill-registry lets agents create, search for, and manage skills themselves.

A skill compared with a skill revision

Feature	Skill	Skill revision
Can change?	Yes, it is mutable	No, it is an immutable snapshot
What it holds	Metadata, the default revision, and the content	A name, a description, and a parent skill reference

EXAM TRAP

A skill revision cannot be edited. To change a skill, you work with the mutable skill, which points to a default revision.

Packaging a valid skill

A skill zip needs SKILL.md, no symbolic links, and at most 10 MB. Inside SKILL.md, the YAML front matter must have a valid name and description. Skill Registry checks every payload on create or update, and these checks run asynchronously.

The two required SKILL.md fields compared

Feature	name	description
Maximum length	**64** characters	**1024** characters
If the field is missing	Validation fails	Validation fails

EXAM TRAP

A name like My-Skill fails because names allow only lowercase letters, numbers, and hyphens. A name that starts or ends with a hyphen also fails.

The Agents CLI lifecycle

The Agents CLI gives one path: scaffold, evaluate, deploy, publish, observe. It serves both coding agents and developers. It manages infrastructure with Terraform and deploys through CI/CD pipelines on Cloud Build. Observability is built in through Cloud Trace and Cloud Logging.

Building and scaling with ADK

ADK agents run locally or scale globally on three platforms. The platforms are Agent Runtime, Cloud Run, and Google Kubernetes Engine. ADK comes in Python, TypeScript, Go, and Java. You can build predictable pipelines with workflow agents. For more adaptive behavior, let agents coordinate dynamic routing instead.

Discovery in Agent Registry

Agents deployed to Agent Platform join Agent Registry automatically. The registry is one catalog for governance, discovery, and search. It holds MCP servers, tools, standalone skills, and AI agents across your organization.

Governing with Agent Gateway

Agent Gateway blocks every agent connection unless IAM grants it. For Agent Runtime agents, Google runs the gateway proxy, so no proxy setup is needed. In dry run mode, the Semantic governance engine logs verdicts on tool calls but never blocks them.

Three kinds of policy for governing agents compared

Feature	IAM Unified Access Policy	Semantic governance policy	Token management policy
Controls	Which tools and endpoints an agent may reach	How an agent uses tools	Shared LLM token budgets
Example use	Grant one agent access to an approved tool	Stop an unsafe combination of tools	Prevent runaway token spending

EXAM TRAP

A tool call can be denied even with no constraints set. By default, the Semantic governance engine also checks that the call matches what the user asked for.

Stopping a deployed agent

Deleting a deployed agent is the kill switch. Deletion stops all running containers at once and invalidates the agent's identity. Any IAM Unified Access Policy binding for it stays active until you update or delete it.

EXAM TRAP

Revoking a policy alone is not an immediate halt. Policy changes take **2** to **7** minutes to spread, so pair revocation with deletion or traffic rerouting.

Observing and optimizing

Deploying with ADK collects traces, logs, and metrics by default. An agent's observability dashboards include an Evaluation view. It tracks response quality, safety, hallucination rates, and tool-use quality. To improve an agent, the adk optimize command refines its root system instructions against your test suite with the GEPA algorithm.

Key takeaways

- A skill is mutable, but each skill revision is an immutable snapshot.
- SKILL.md limits are **64** characters for the name and **1024** for the description.
- Agent Gateway denies all agent connections by default until an IAM policy grants access.
- For an immediate stop, delete the agent instead of relying only on policy revocation.

Sources: Skill Registry overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent Runtime | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent Gateway overview | Gemini Enterprise Agent Platform | Google Cloud Documentation

(opens in a new tab) · Agent Registry | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · ADK | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agents overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Configure semantic governance policies | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Manage deployed agents | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab)

Domain 3 | 33% of the exam | 16 min read

Developing custom agents

Overview

This domain is about 33% of the exam, roughly 20 of its 60 questions.

Every point below comes from Google Cloud's official documentation; each section lists its sources.

Designing and building agentic workflows in code

Start here

THINK FIRST

What should drive your choice of model, agent framework and memory setup when you build an agent in code?

Why it matters: Questions describe an agent's needs and ask which model option, build path, or session and memory setting fits them.

A coded agent rests on three layers. You pick and serve a model from Model Garden, build the agent logic with ADK, and give it context. Sessions hold one conversation, while memories carry user facts across sessions.

Key terms

Term	Meaning
Model Garden	Platform to discover, test, customize and deploy Google proprietary models and select open-source models.
Model as a Service (MaaS)	Serverless, managed API for using open and partner models without managing infrastructure.
Agent Development Kit (ADK)	Framework for building complex agents in custom code, with fine-grained control over logic and tools.
Session	The ordered messages and actions (events) of one ongoing interaction between a user and the agent.
Memory Bank	Store of user memories that agents read and write; it evolves with the context the agent provides.

Finding a model in Model Garden

Model Garden is where you discover, test and deploy models. It offers more than **200** foundation models from Google, partners and the open-source community. Partner models include Anthropic Claude, Grok and Mistral AI. Gemma 3 is an open model with text and image input and a **128K** context window. Gemini 3.1 Flash-Lite is the most cost-efficient model, built for low-latency, high-volume, cost-sensitive traffic. Pay-as-you-go use comes in **3** options: Flex, Priority and Standard.

Managed API or self-deployment

MaaS removes infrastructure work; self-deployment fits custom models. Partner models used as MaaS are serverless, yet requests still go to Gemini Enterprise Agent Platform endpoints. Self-deployment is also cost-effective for production apps with predictable, high-volume traffic.

MaaS and self-deployment compared

Feature	MaaS	Self-deployment
How the model runs	Serverless, behind a managed API	Deployed by you from Model Garden
Best fit	Partner and open models with no infrastructure to manage	Apps that need custom weights or a fine-tuned version

EXAM TRAP

Self-deployment is not only for custom models. For production apps with predictable, high-volume traffic, it is also a cost-effective option.

Making models smaller

Quantization and distillation both yield smaller, more efficient models. Pick quantization when you want lower inference latency and power use. Pick distillation when a large model must serve devices with limited resources.

Quantization and distillation compared

Feature	Quantization	Distillation
How it works	Lowers the numeric precision of model parameters	Trains a small student model to mimic a larger teacher
Main benefit	Smaller model, less power, lower inference latency	More efficient models for devices with limited resources

Controlling model access and data

Organization policies limit which models and endpoints teams use. A Model Garden policy at organization, folder or project level can allow only vetted models and deny all others. The constraint `constraints/gcp.restrictEndpointUsage` blocks the global endpoint to enforce regional ones. Use a regional endpoint for data residency needs. Prompts and responses for partner models are not shared with third parties.

EXAM TRAP

Hugging Face models flagged as suspicious are not blocked. Only models flagged as unsafe are blocked; suspicious ones are marked but can still be deployed.

Building agents with ADK

ADK is the custom-code path for multi-agent systems. It gives fine-grained control over logic, tools and environment simulation. You use it to build, test and deploy a prototype agent to a Google Cloud runtime. Agent Runtime is a fully managed place to host, deploy and scale agents built with ADK or other tools.

EXAM TRAP

You do not have to build session handling yourself. Once an ADK agent is deployed to Gemini Enterprise Agent Platform, session management is automatic.

Sessions, state and memory

A session covers one conversation; memory spans many sessions. Every session needs a user ID, and memories extracted from it map to that user. Sessions are also the input from which long-term memories are generated.

Three places an agent keeps context

Feature	Session	State	Memory
What it holds	Messages and actions (events) in order	Temporary data	Personalized information about a user
How long it matters	One ongoing interaction	Only the current conversation	Across multiple sessions

How Memory Bank stores memories

Memory Bank evolves with agent context, unlike RAG's static store. It keeps only extracted facts that match at least one of the instance's memory topics. By default, it generates memories with gemini-3.5-flash, in the first person. You can switch to third person, or set a TTL so stale memories are deleted automatically. Region and encryption settings are fixed once the instance exists.

EXAM TRAP

Memories do not expire on their own by default. Memory Bank sets no TTL on created or updated memories unless you configure one.

Reading and writing memories

ADK agents read and write Memory Bank with built-in tools. The VertexAiMemoryBankService handles these calls, and ADK can orchestrate fetches automatically. Use CreateMemory when the agent needs more control over which facts are saved. Each memory has a scope, set when the memory is generated or created and fixed after that. A retrieval request names a scope too. RetrieveMemories looks memories up by that

scope; ListMemories skips scope filtering and is not recommended for low-latency retrieval. Agents on Agent Runtime already have permission to read, write and generate memories.

EXAM TRAP

RetrieveMemories does not return partial scope matches. It returns only memories with exactly the same scope as the request, in any key order.

Key takeaways

- Use MaaS for open and partner models without infrastructure work; self-deploy for custom weights or fine-tuned versions.
- A session holds one conversation, while memory follows one user across many sessions.
- ADK agents deployed to Gemini Enterprise Agent Platform get session management automatically.
- Memory Bank sets no TTL by default, so add one if stale memories should be deleted automatically.

Sources: Google models | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · ?????? | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent Platform Sessions overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · ?????????? | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Overview of models on Agent Platform | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent Platform Memory Bank | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · ?????? ?? ?????? ?-Gemini Enterprise Agent Platform ?-MaaS | Google Cloud Documentation (opens in a new tab) · Generative AI glossary | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab)

Integrating enterprise domain knowledge

Start here

THINK FIRST

How would you ground an agent in enterprise data while keeping its access to tools and credentials tightly controlled?

Why it matters: Questions describe a retrieval or tool-access need and ask which search setup, permission control or toolset fits it.

Grounding an agent in enterprise knowledge has two sides. Retrieval finds the right content, using embeddings, filters and reranking in Vector Search. Access control decides what the agent may reach, using a sandbox, least-privilege credentials and Agent Registry policies.

Key terms

Term	Meaning
Embedding	A vector representation of text, used for semantic search, classification and clustering.

Term	Meaning
Hybrid search	Search that uses dense and sparse embeddings together, weighted as you specify.
Restrict	A feature, also called filtering, that limits a search to part of the index with Boolean rules.
Agents API	The control plane that creates agents and configures their environments, data mounts and network allowlists.
Agent Registry	A unified catalog of Model Context Protocol (MCP) servers, tools, skills and AI agents across an organization.

Embeddings and Vector Search

Embeddings turn text into vectors that Vector Search can compare. The Embeddings for Text model produces vectors for semantic search, classification and clustering. For standard text-to-text or text-to-image matching, a ready-to-use model needs no training or tuning. Vector Search itself is built on the ScaNN algorithm.

EXAM TRAP

Do not plan a training or tuning job for plain text-to-text or text-to-image matching. A ready-to-use model already covers that standard case.

Dense, sparse and hybrid

Vector Search supports dense, sparse and hybrid search. Pick by what must match. Dense embeddings find results by meaning, while sparse embeddings suit keyword searches. Hybrid search uses both types, weighted the way you choose.

The three Vector Search modes compared

Feature	Dense	Sparse	Hybrid
Captures	Semantic meaning of text	Text syntax	Both, through two embedding types
Array values	Mostly non-zero	Very few non-zero	Uses both kinds of embeddings
Typical use	Results by semantic similarity	Keyword searches	Blending both, by weights you set

EXAM TRAP

Hybrid search is not a switch between dense and sparse. It uses both embedding types, weighted as you specify.

Filtering and reranking

Restrict narrows the index; the ranking API reorders results. Restrict applies Boolean rules so a query searches only a subset of the index. The ranking API then reranks documents by relevance to the query, using a pre-trained language model. Recall measures index quality: the share of returned neighbors that are true nearest neighbors.

EXAM TRAP

Reranking does not shrink what gets searched. Restrict limits the index with Boolean rules; the ranking API only reorders and scores the documents returned.

Public endpoints

A public endpoint needs no network setup. Choose it when fast setup and easy maintenance matter more than the lowest latency. Public networks have slightly higher latency than private endpoints.

Sandbox by default

A managed agent starts with no outside access at all. By default, each agent works in a sandbox without external systems, networks or credentials. You must configure any connectivity beyond it explicitly. The Agents API does this by mounting outside data, such as Cloud Storage, and defining network allowlists. With network access enabled, the agent can reach external APIs, clone public repositories and download packages. At runtime, you talk to deployed agents through the Interactions API, the data plane.

Least-privilege credentials

An agent may use any credential it can access. That is why every credential should be least privilege. Create service accounts or API keys with only the permissions the agent needs, and avoid broad or admin access. Prefer short-lived, expiring tokens over long-lived API keys. Configure tools with minimum permissions, using read-only mode unless write access is necessary.

EXAM TRAP

Do not assume an agent uses only the credentials its task seems to need. It may use any credential it has access to.

Governing access with Agent Registry

Agent access policies apply to resources in Agent Registry. Register MCP servers, tools, skills, agents and endpoints to govern centrally what agents can connect to. IAM allow policies then control agent communication with registered resources. Agent Gateway enforces these policies through Identity-Aware Proxy.

Example: An IAM policy binding that grants agent principals egress access to registered resources.

```
{
  "policy": {
    "bindings": [
      {
        "role": "roles/iap.egressor",
        "members": ["AGENT_PRINCIPALS"]
      }
    ]
  }
}
```

- `roles/iap.egressor`: the role that grants agent egress access

EXAM TRAP

A policy cannot be bound to a resource that is not registered. Register the resource in Agent Registry first, or the binding fails with NOT_FOUND.

Toolsets for custom capabilities

Toolsets come from MCP servers, OpenAPI schemas or Integration Connectors. The Agents API connects external tools and APIs, such as MCP servers, to extend a managed agent. When you build tools in CX Agent Studio, each toolset type generates its tools differently.

CX Agent Studio toolset types compared

Feature	MCP toolset	OpenAPI toolset	Connector toolset
Tools come from	The list an MCP server offers	An OpenAPI schema	An Integration Connectors connection
Key rule	Server must use Streamable HTTP transport	An unnamed tool takes its operationId as its name	Admin auth by default; end-user auth can override it

EXAM TRAP

An MCP server on another transport will not work. CX Agent Studio MCP tools support only servers that use Streamable HTTP.

Key takeaways

- Dense embeddings match meaning, sparse embeddings match keywords, and hybrid search weights both.
- Restrict filters the index with Boolean rules; the ranking API reorders results by relevance.
- Agents start sandboxed, so every outside connection must be configured explicitly.
- Give agents least-privilege, short-lived credentials and read-only tools where possible.
- Agent access policies bind only to resources registered in Agent Registry.

Sources: Vector Search | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new

tab) · Managed Agents API on Agent Platform overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Configura políticas del agente de IAM | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Google models | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Before you begin | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent Registry | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · MCP Tools Reference: ces.googleapis.com | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · MCP Tools Reference: ces.googleapis.com | CX Agent Studio | Google Cloud Documentation (opens in a new tab)

Orchestrating and coordinating agentic workflows

Start here

THINK FIRST

How would you connect, hand off between, and govern several cooperating agents on Google Cloud?

Why it matters: Questions describe a multi-agent design and ask which protocol, handoff model, or governance service fits it.

Agents rely on two open protocols: MCP connects them to external systems, and A2A lets them coordinate with other agents. On Google Cloud, each deployed agent gets an Agent Identity and a place in Agent Registry. Agent Gateway then enforces policies on its calls.

Key terms

Term	Meaning
Model Context Protocol (MCP)	An open standard that connects AI applications to external systems.
Agent2Agent (A2A)	An open messaging standard that lets autonomous agents talk and coordinate tasks across separate systems.
Agent Identity	A unique, SPIFFE-formatted ID for one agent, used in place of shared service accounts.
Agent Registry	One catalog to store, discover, and manage MCP servers, tools, skills, and agents across an organization.
Agent Gateway	The runtime checkpoint that intercepts an agent's calls to tools or other agents to enforce access policies.

MCP and A2A at a glance

MCP links agents to systems; A2A links agents to agents. Both are open standards. MCP connects an AI application to external systems. A2A lets autonomous agents exchange messages and share tasks across systems that differ.

MCP and A2A compared

Feature	MCP	A2A
Connects	An AI application to external systems	Autonomous agents to each other across systems
Session state	Stateless starting with version **2026-07-28**	Remote third-party sessions are stateless by default
Wire options	HTTP endpoint (remote server) or stdio (local server)	JSONRPC, GRPC, or HTTP+JSON bindings

Local and remote MCP servers

Remote MCP servers offer an HTTP endpoint; local ones use stdio. A local server usually runs on your own machine and talks over standard input and output. A remote server runs on the provider's infrastructure. Under stateless MCP, every request carries what it needs in HTTP headers or the `_meta` parameter. CX Agent Studio accepts only StreamableHttpTransport servers, not SSE ones.

> **EXAM TRAP**
> Do not plan an initialize handshake or an `Mcp-Session-Id` for stateless MCP. Each request already holds all the information the server needs.

A2A agent cards and messages

An AgentCard is the manifest an agent uses to describe itself. It lists the agent's identity, capabilities, skills, supported ways to communicate, and security requirements. When a task reaches `TASK_STATE_INPUT_REQUIRED`, it is interrupted until the user supplies more input.

> **EXAM TRAP**
> Sending embeddings or raw audio to a remote agent over A2A is wrong. Neither one crosses the A2A network interface.

Delegation vs pass-through

Delegation keeps control with the primary agent; pass-through hands it off. Choose delegation when the primary agent should stay in charge and sum up the answer. Choose pass-through, a Preview feature, when the remote agent should take over the conversation.

The two A2A handoff models

Feature	Delegation	Pass-through
Session control	Stays with the primary agent	Moves to the remote A2A agent
Remote agent's role	Called by the primary agent as a tool	Its replies go back to the user as they are
Handling the result	Primary agent summarizes it	No intermediate model call

EXAM TRAP

Pass-through does not summarize the remote agent's reply. The primary application returns it without any intermediate model call.

Wiring remote agents in CX Agent Studio

Agent as a tool stays inside one app; an A2A Protocol tool reaches out. An A2A Protocol tool calls external agents, remote services, or third-party platforms. A RemoteA2aConfig names the remote agent with an inline `agentCard` or an `agentRegistry` reference, never both. You can also list a RemoteA2aAgent under a parent's `childAgents` and hand off with a `PARENT_TO_CHILD` transfer rule. When several rules match, the first wins.

Building and deploying workflows

Agent Runtime deploys agents from many languages and frameworks. It also supports the A2A protocol; an A2A agent lists `google-adk[a2a]` in its requirements. ADK is the custom-code path for complex multi-agent orchestration. LangGraph models a workflow as a graph, with human-in-the-loop and rewind or replay. Agents CLI offers ReAct, RAG, and multi-agent templates.

Agent Identity and Agent Registry

Deployed agents carry a unique identity and join Agent Registry automatically. Agent Identity is a SPIFFE-formatted ID that is more granular than a shared service account. It follows the agent resource ID, not the framework you built with. Agents deployed to Agent Platform are registered without extra steps.

EXAM TRAP

Patching an already deployed agent to add `identity_type` or `agent_gateway_config` does not work. Both fields cannot change after deployment, so you must redeploy the agent.

Agent policies at Agent Gateway

Agent Gateway intercepts agent calls to enforce access policies. It enforces IAM allow policies through Identity-Aware Proxy (IAP). Egress policies cover **four** interactions: agent to registry, agent to agent, agent to MCP server, and agent to endpoint. Conditions can limit destructive actions, like deleting resources, or close off open-world tools, like web search. Begin in dry-run mode.

EXAM TRAP

A policy cannot be bound to a resource missing from Agent Registry. The binding fails right away with NOT_FOUND, not later at runtime.

Key takeaways

- MCP connects AI applications to external systems, while A2A lets agents coordinate across systems.
- In delegation the primary agent keeps control; in pass-through the remote A2A agent takes over.
- An AgentCard lists an agent's identity, skills, communication methods, and security requirements.

- Agent Gateway intercepts agent calls to enforce policies, and policies bind only to registered resources.
- Agent Identity is a unique SPIFFE ID tied to the agent resource, not to its framework.

Sources: MCP ??? | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · A2A Protocol tools | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · Agents overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent Registry | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Use the CX Agent Studio MCP server | Google Cloud Documentation (opens in a new tab) · Package lf.a2a.v1 | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · REST Resource: projects.locations.apps.agents | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · Agent Runtime | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab)

Domain 4 | 22% of the exam | 10 min read

Evaluating and deploying agentic workflows

Overview

This domain is about 22% of the exam, roughly 13 of its 60 questions.

Every point below comes from Google Cloud's official documentation; each section lists its sources.

Evaluating agents in development and in production

Start here

THINK FIRST

How would you build test sets and pick tools to evaluate an agent both before and after it goes live?

Why it matters: Questions describe an agent and a testing need, then ask which test set, metric or evaluation tool fits.

Agent evaluation is a continuous cycle: evaluate, analyze failures, then optimize. Eval cases define the tasks, and traces record what the agent did. Offline evaluation scores traces against metrics, and Online Monitors keep scoring on a schedule.

Key terms

Term	Meaning
Eval case	A specification that defines the task the agent must perform.
Trace	An immutable record of agent behavior: model inputs, responses and tool calls.
Conversation plan	A hidden instruction that tells the simulated user its goals and how to react.
Golden expectation	A check on a golden turn, such as which tool was called and with what parameters.
Quality drift	A drop in agent performance over time caused by changes in user behavior or external data.

The Quality Flywheel

Evaluation is a loop: evaluate, analyze failures, then optimize. You score the agent, group its failures into clusters, and fix those specific issues. Then the cycle starts again. An eval case specifies the task. A trace is the immutable record of what the agent did, including model inputs, responses and tool calls.

Reference-based vs reference-free

Some metrics need a reference answer; others do not. Match the metric to your test set. If your eval cases lack reference answers, Exact Match cannot score them. Use a reference-free metric such as Helpfulness instead.

Two kinds of metrics compared

Feature	Reference-based	Reference-free
Example metric	Exact Match	Helpfulness
Needs a reference answer	Yes, in the eval case	No, it judges the trace on its own

EXAM TRAP

Helpfulness does not need a reference answer. It is reference-free and evaluates the trace by itself.

Simulated test sets

Simulation builds a test suite even without production data. LLMs generate diverse multi-turn scenarios from the agent's instructions and tool definitions. They then roleplay as the user to stress-test the conversation logic. Each case has a starting prompt and a conversation plan. Simulation needs at least **one** agent version in the Agent Registry.

EXAM TRAP

You do not have to wait for live traffic to build a test set. Simulation can create cases from scratch with LLMs.

Edge cases and failures

Environment simulation injects errors into tool calls. It intercepts chosen tool calls and returns mocked data or simulated errors, such as HTTP **503** errors or latency spikes. A persona's speaking rate can be raised to test fast talkers. In user simulation, Max Turn caps invocations to stop endless loops.

EXAM TRAP

Max Turn is not unlimited by default. It defaults to **5** invocations, which stops a runaway conversation.

ADK evaluation tooling

Evaluate an Agent Development Kit (ADK) agent locally to iterate fast. Local runs let you tune prompts and tool configurations quickly. Built-in and partner tools test execution trajectories. The agents-cli eval

commands give a CLI workflow to evaluate and optimize. The `adk optimize` command applies the GEPA algorithm to refine root system instructions against your test suite.

Scoring traces and sessions

Score single traces or whole multi-turn sessions. Offline evaluation scores individual traces or full sessions against predefined or custom metrics. A session is a multi-turn conversation history. Multi-turn autoraters evaluate entire conversation histories automatically. In the console, a new offline evaluation adds all **four** core metrics by default.

Golden and scenario evaluations

A CX Agent Studio evaluation uses golden steps or a scenario, never both. A golden evaluation replays a golden conversation, and expectations check each turn. An expectation can require a specific tool with given parameters, or no tool calls at all. Tool invocation correctness passes at a threshold from **0** to **1** that defaults to **1.0**. A scenario instead simulates the conversation and scores it against rubrics.

Golden and scenario evaluations compared

Feature	Golden evaluation	Scenario evaluation
Input	Golden steps	Scenario config
Turn limit	Replays at most **100** turns	Simulates at most **100** turns, the default
Checked against	Golden expectations for each turn	Required rubrics

EXAM TRAP

A mocked tool response does not fill gaps safely. Any parameters you leave out of mockToolResponse are hallucinated by the LLM.

Continuous online monitoring

Online Monitors evaluate on a schedule, typically every 10 minutes. This loop helps you detect quality drift early. Results go to Cloud Logging, and numeric scores go to Cloud Monitoring. The Max samples per run setting caps evaluation costs.

EXAM TRAP

Max samples per run does not set how often a monitor runs. It caps evaluation costs; the schedule sets the timing.

Key takeaways

- Evaluation is a continuous cycle of evaluating, analyzing failure clusters and optimizing.
- Exact Match needs a reference answer in the eval case; Helpfulness does not.
- Simulation creates multi-turn test cases with LLMs, even without production data.
- A golden expectation can check that a specific tool was called, or that no tool was.
- Online Monitors typically run every **10** minutes to catch quality drift.

Sources: Optimize agent prompts | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent evaluation | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Simulate agent behavior | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Run offline evaluations | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Continuous evaluation with online monitors | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Evaluation.GoldenExpectation | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · MCP Tools Reference: ces.googleapis.com | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · ADK | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab)

Deploying and scaling production workloads

Start here

THINK FIRST

How would you pick a runtime for an agent, then keep it fast, reliable, and affordable once it runs in production?

Why it matters: Questions describe an agent workload or failure and ask which runtime, monitoring signal, or fix fits it.

Agent Runtime is a fully managed home for agents, while Cloud Run runs containers as immutable revisions. Once deployed, telemetry shows latency, errors, and the agent's reasoning path. Monitoring alerts and a kill switch let you catch drift and stop bad behavior.

Key terms

Term	Meaning
Agent Runtime	Fully managed runtime, formerly Agent Engine, that hosts, deploys, and scales agents.
Runtime contract	HTTP API spec that sets a custom container agent's endpoints and request and response structures.
Revision	An immutable Cloud Run version, created whenever any setting changes.
OpenTelemetry	Protocols used to collect agent traces, logs, and metrics such as latency and token usage.
Drift	A significant change over time in the feature data a model sees in production.

Agent Runtime basics

Agent Runtime runs agents without you managing servers. It was formerly named Agent Engine. It hosts, deploys, and scales agents built with ADK or other tools, so you focus on agent logic. It adds session management, a long-term Memory Bank, VPC-SC compliance, and IAM settings. Agents deployed to Agent Platform also join the Agent Registry automatically.

Custom agents and the contract

Any language works if the agent is containerized and meets the contract. The runtime contract defines the HTTP endpoints and the request and response shapes. ADK is fully integrated. LangChain, LangGraph, AG2, and LlamaIndex get managed templates in the Agent Platform SDK. With that SDK, query code is identical for local and deployed agents.

EXAM TRAP

Agent Runtime is not limited to ADK agents. Any language or framework can run, as long as the agent is containerized and follows the runtime contract.

Agent Runtime vs Cloud Run

Match the runtime to the use case, requirements, and cost. The exam names Agent Runtime, Cloud Run, and GKE as options. Agent Runtime hides the infrastructure from you. Cloud Run services cost nothing until they receive requests.

Agent Runtime and Cloud Run compared

Feature	Agent Runtime	Cloud Run
What it runs	Containerized agents in any language that meet the runtime contract	Container images; any setting change makes a new immutable revision
Extra features	Integrated session management and a long-term Memory Bank	Continuous deployment, if set up: a Cloud Build trigger on new commits to one branch
Fast recovery	Delete the agent or reroute revision traffic	Roll back to a previous, known-good revision

EXAM TRAP

Changing only a Cloud Run setting, with the same image, still creates a new revision. Every revision is immutable.

Telemetry for deployed agents

ADK deployments send telemetry to Cloud Observability by default. The suite includes Cloud Trace, Cloud Logging, Cloud Monitoring, and Topology. It uses OpenTelemetry to collect traces, logs, and metrics such as latency and token usage. That lets you debug failures, watch costs, and follow an agent's full reasoning path.

Agent dashboard tabs

Overview tracks the whole agent; Tools tracks its tool calls. Switch on OpenTelemetry trace and log instrumentation to populate these pages. Check the Tools tab when tool invocation latency is the suspect. With traces on, the Playground also shows Trace, Event, and State details, where Event graphs the invoked APIs.

Metrics on the Overview and Tools tabs

Feature	Overview tab	Tools tab
Volume	Agent request count	Number of tool calls
Errors	Agent error rate	Tool error rate
Speed	Agent latency	Tool latency

EXAM TRAP

Agent latency on the Overview tab does not isolate slow tools. Tool latency appears on the Tools tab.

Skew and drift detection

Use skew detection with training data, drift detection without it. Model Monitoring alerts when a score passes a threshold, so you can re-evaluate or retrain. By default, v1 watches all features with a threshold of **0.3**. To cut cost, set a sampling rate that monitors only a subset of requests.

Skew detection and drift detection compared

Feature	Skew detection	Drift detection
Compares	Production feature data against the training data	Changes in production feature data over time
Use when	The original training data is available	The original training data is not available

EXAM TRAP

Drift detection does not need the training data. Skew detection is the one that does.

Halting a misbehaving agent

Deleting the deployed agent is the immediate kill switch. Deletion stops every running container and invalidates the agent's SPIFFE identity. You can also reroute traffic or revoke access through Agent Gateway and IAM. Policy changes take **2** to **7** minutes to spread, so pair them with deletion or rerouting.

EXAM TRAP

Revoking IAM access alone does not stop an agent right away. Policy changes need **2** to **7** minutes or more to propagate.

Debugging Cloud Run errors

Reproduce locally first to split code bugs from platform issues. Running the image locally shows whether the problem is in the code or only in Cloud Run. Error Reporting groups errors by message across revisions and services. Cloud Logging also shows errors that lack a recognized stack trace. For a quick fix, roll back to a previous healthy revision.

EXAM TRAP

Error Reporting does not list every error. Check Cloud Logging for messages that lack a recognized stack trace.

Key takeaways

- Agent Runtime is fully managed and runs any containerized agent that meets the runtime contract.
- Any Cloud Run setting change creates a new immutable revision, and a previous revision is your rollback.
- Use the Tools tab to find slow or failing tool calls.
- Pick skew detection when training data exists and drift detection when it does not.
- Deleting an agent halts it at once, while IAM revocation takes **2** to **7** minutes.

Sources: Agent Runtime | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Deploy container images to Cloud Run services | Google Cloud Documentation (opens in a new tab) · Agents overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Introduzione a Model Monitoring | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Manage deployed agents | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Professional Agentic Architect Certification exam guide A Google Cloud Certified Professional Agentic Architect is a tec (opens in a new tab) · Use an agent | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Tutorial: Local troubleshooting of a Cloud Run service | Google Cloud Documentation (opens in a new tab)

Domain 5 | 15% of the exam | 10 min read

Securing and governing agentic workflows

Overview

This domain is about 15% of the exam, roughly 9 of its 60 questions.

Every point below comes from Google Cloud's official documentation; each section lists its sources.

Configuring agent security and governance

Start here

THINK FIRST

How do agent identity, Agent Gateway policies and content filters work together to keep an agent's actions safe?

Why it matters: Questions describe an agent's access or safety need and ask which identity, policy, auth method or filter meets it.

Every agent has its own identity, and Agent Gateway denies its connections unless IAM allows them. The gateway also reports on traffic. Model Armor and semantic governance policies then check content and tool calls.

Key terms

Term	Meaning
SPIFFE ID	The unique ID that Agent Identity gives each agent for authentication, access control and audit.
Agent Gateway	Enforces IAM allow policies on agent traffic to resources registered in Agent Registry, using Identity-Aware Proxy (IAP).
Egress policy	An IAM policy that grants `roles/iap.egressor` on registered agents, MCP servers and endpoints.
Model Armor	A content filter on a gateway that scans user prompts and tool responses in real time.
Semantic governance policy	Business rules in natural language that a policy engine checks each agent tool call against.

Agent identity

Each agent gets a unique SPIFFE ID from Agent Identity. That ID works as the agent's digital signature for authentication, access control and audit. Identity and Access Management (IAM) policies then decide what the agent may reach.

Tool auth methods

A CX Agent Studio tool's auth config sets at most one method. API key, OAuth, service agent ID token, service account and bearer token are mutually exclusive, so you pick only one. An API key can travel in an HTTP header or in the query string.

Three auth methods for tool API calls compared

Feature	OAuth	Service account	Bearer token
Where the credential comes from	Client secret, named by a Secret Manager secret version	The service account, used to get an access token	A session context variable
Key setting	The provider's token endpoint, used to get an access token	Scopes; the cloud-platform scope is the default	Token given as $context.variables plus the variable name
Notes	Supports only the client credential grant	Access token is sent in the Authorization header	Suits a token collected from the end user

EXAM TRAP

OAuth for tool calls does not accept every grant type. Only the client credential grant is supported.

Tool execution identity

OpenAPI tool actions run as the CX Agent Studio service account. They do not use the end user's own permissions. Deploy agent applications in a dedicated project to limit what that service account can reach. A conditional IAM role binding can also restrict a principal to one agent application.

EXAM TRAP

A tool action does not run with the permissions of the user chatting with the agent. It runs with the CX Agent Studio service account's permissions.

Deny by default

Agent Gateway blocks every connection unless an IAM policy allows it. You write IAM allow policies for resources registered in Agent Registry. The gateway enforces them through IAP, so enable the IAP API (`iap.googleapis.com`) in your project. It also terminates mTLS handshakes and handles OAuth 2.0 and workload identity federation for you.

EXAM TRAP

An agent cannot reach a registered tool just because both are registered. With no explicit IAM policy, the gateway blocks the connection.

Egress policies

Egress policies cover four kinds of agent interaction. They control traffic from an agent to a registry, another agent, an MCP server or an endpoint. One policy condition, ReadOnly, limits the agent to reading data and prevents writes. A condition's CEL expression must evaluate to true to allow access.

EXAM TRAP

Binding a policy to an unregistered resource does not wait until runtime to fail. It is checked at bind time and fails at once with a **NOT_FOUND** error.

Monitoring agent traffic

The gateway produces telemetry for every agent interaction. It captures this telemetry at the network level and exports it to Agent Observability. That gives you a full view of what your agents do. One gateway instance can govern up to **5,000** resources registered in Agent Registry.

Model Armor and semantic governance

Model Armor filters content; the policy engine judges tool calls. Semantic governance constraints are business rules in plain language, up to **5,000** characters. In DRY_RUN mode, the engine evaluates tool calls and logs verdicts but never blocks them.

Model Armor and semantic governance compared

Feature	Model Armor	Semantic governance
What it checks	User prompts and tool responses, in real time	Agent tool calls, against your constraints
What it stops	Prompt injection, sensitive data leaks and harmful content	Tool calls that break a constraint or do not match the user's request
Where it applies	Attached to gateways	Agents linked to an Agent Gateway

EXAM TRAP

A tool call that breaks no constraint can still be denied. With user intent matching on, the engine rejects calls that do not match the user's request.

Semantic governance setup

Governed agents need two deployment fields set. Deploy the agent with `identity_type=AGENT_IDENTITY` and `agent_gateway_config`, or it is left out of the policy's agent selector. The agent must also call its model without streaming. With several projects, define policies in the

management project, because the engine evaluates only its own project's policies.

EXAM TRAP

You cannot add the two fields to an existing agent. Both are immutable, so an agent deployed without them must be redeployed.

Key takeaways

- Agent Gateway denies every agent connection until an explicit IAM policy allows it.
- OpenAPI tool actions use the CX Agent Studio service account's permissions, not the end user's.
- Each semantic governance evaluation logs an overall verdict, plus a verdict and rationale per tool or action, in Cloud Logging.
- To use a Secret Manager secret for tool auth, grant the Customer Engagement Suite Service Agent the Secret Accessor role on it.

Sources: Panoramica di Agent Gateway | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Configura políticas de administración semántica | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Configure IAM agent policies | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Agent Registry | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Authenticate to CX Agent Studio | Google Cloud Documentation (opens in a new tab) · ApiAuthentication | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · OpenAPI tools | CX Agent Studio | Google Cloud Documentation (opens in a new tab) · Access control with IAM | CX Agent Studio | Google Cloud Documentation (opens in a new tab)

Implementing secure agent behavior and execution

Start here

THINK FIRST

How would you layer identity, access checks and content filtering so an agent can only do what it should?

Why it matters: Questions describe an agent deployment and ask which control blocks a risk, proves identity or approves a destination.

Agent Gateway is the checkpoint for agent traffic: it denies every connection by default and checks who is asking and where they are going. Model Armor then screens the content of prompts and responses for attacks and data leaks.

Key terms

Term	Meaning
Agent Registry	The central directory of approved agents, tools, MCP servers and endpoints in an organization.

Term	Meaning
SPIFFE ID	The unique, secure ID that Agent Identity gives each agent, acting as its digital signature.
Model Armor	Content security filters that scan prompts and responses to block prompt injection, data leaks and harmful content.
Floor settings	Minimum protection rules that every Model Armor template created at the project level must meet.
Semantic governance policy	A plain-language rule that controls how agents use tools and is enforced at runtime.

Deny by default

Agent Gateway blocks every connection unless IAM allows it. Access starts closed. An agent can reach a destination only after an explicit IAM policy grants that access. Before allowing a connection, the gateway also checks permissions against Agent Registry.

EXAM TRAP

Registering a tool in Agent Registry is not enough on its own. Without an explicit IAM policy that grants access, the connection stays blocked.

Identity and approved destinations

Agent Identity answers who; Agent Registry answers where. Each agent's SPIFFE ID acts as its digital signature, so authentication, access control and audits tie back to one agent. The registry lists which Model Context Protocol (MCP) servers, tools and endpoints are approved targets.

The two identity and access building blocks of Agent Gateway

Feature	Agent Identity	Agent Registry
Question it answers	Who made the request?	What destinations are approved?
What it holds	A unique SPIFFE ID for each agent	Approved agents, tools, MCP servers and endpoints
What it is used for	Authentication, access control and auditing	Checking permissions before a connection is allowed

Identity propagation

The gateway handles OAuth and Workload Identity Federation for agents. Agent Gateway works with the Agent Identity Auth Manager. Together they handle OAuth handshakes and Workload Identity Federation transparently, as part of secure transport between agents and their destinations.

Rules for tool use

Semantic governance policies stop unintended agent actions at runtime. These rules are written in plain language and enforce business rules while the agent runs. For example, a policy can stop an agent from running an unsafe combination of tools.

Screening prompts and responses

Model Armor scans both what goes into a model and what comes out. Attached to a gateway, it checks user prompts and tool responses in real time. It covers traffic going to the agent and traffic leaving it. Some integrations, such as Apigee, also clean intermediate steps like grounding data.

Inspect before you block

Start Model Armor in Inspect only, then move to Inspect and block. Watching first shows likely block rates for your use case. After you review the logs and tune settings, switch modes for active protection.

Model Armor enforcement types compared

Feature	Inspect only	Inspect and block
On a violation	Logs details to Cloud Logging; processing continues	Gives active protection
When to use it	First, to learn block rates and efficacy	After analyzing logs and adjusting settings

EXAM TRAP

Calling Model Armor through its REST API does not block anything by itself. It only detects and reports violations, so the application must decide whether to block.

Floor settings and templates

Floor settings set the minimum bar for every project template. A floor setting violation blocks the prompt or response and sends a finding to Security Command Center. Floor settings also protect MCP tool calls and responses. Keep separate templates for user prompts and model responses.

What Model Armor misses

Model Armor checks each turn alone and cannot read encoded text. It does not track conversation history across turns. It does not decode Base64, hexadecimal, URL-encoded or encrypted prompts. Only the Gemini Enterprise integration scans documents and images; every other integration scans text only.

EXAM TRAP

An attack split across several turns is not caught by conversation context, because Model Armor treats each prompt and response as a single-turn request.

Key takeaways

- Agent Gateway blocks all connections until an explicit IAM policy grants access.
- Agent Identity gives each agent a SPIFFE ID; Agent Registry lists approved destinations.
- Model Armor screens prompts and responses, in real time, on both ingress and egress paths.
- Begin with Inspect only, read the logs, then switch to Inspect and block.
- Model Armor sees one turn at a time and skips encoded content.

Sources: Agent Gateway overview | Gemini Enterprise Agent Platform | Google Cloud Documentation (opens in a new tab) · Model Armor overview | Google Cloud Documentation (opens in a new tab) · Overview | Model Armor | Google Cloud Documentation (opens in a new tab)

MockCert

Take a Google Cloud Professional Agentic Architect mock exam the way the real one feels

5 timed mock exams · 60 questions in 120 minutes each · one is free

- Free to start: study notes, explained quizzes, flashcards and the first mock exam
- Timed mock exams built to the official exam guide
- A verifiable Certificate of Assessment when you pass a premium mock

Explore Google Cloud Professional Agentic Architect on MockCert



Open it on your phone

mockcert.com/google-agentic-architect/

Scan the code with your phone's camera.