Claude Certified Architect - Foundations (CCAR-F) Questions and Answers

25 practice questions with answers and explanations, covering every exam domain. Taken from our bank of 150 reviewed questions.

Looking for exam dumps? Use these original practice questions instead. These are original practice questions written by CloudYeti. They are NOT real exam questions and are not affiliated with or endorsed by Anthropic.

Take the full timed practice test free:
https://learn.cloudyeti.io/free/claude-certified-architect-foundations-practice-test

DOMAIN 1 / AGENTIC ARCHITECTURE AND ORCHESTRATION

1. A claims process has a fixed sequence and each transition must be auditable: intake, extraction, rule validation, human approval, and payment. Which architecture best fits?

A) A swarm of agents that vote on payment

B) A prompt that asks Claude to remember every policy

C) A single autonomous agent with unrestricted tools

D) A deterministic state machine that uses Claude for ambiguous extraction and explanation while code enforces transitions and approval

Answer: D. The process has known stages and high-impact transitions. Deterministic orchestration creates auditability while reserving model judgment for the genuinely ambiguous steps.

2. A due-diligence system must independently analyze legal, financial, and security evidence before a final synthesis. What topology is most appropriate?

A) A chain where each specialist sees and rewrites all prior raw evidence

B) One agent reads everything in one growing context

C) A coordinator delegates bounded parallel analyses to specialists, then synthesizes their structured evidence-backed reports

D) Three agents edit the same document simultaneously

Answer: C. The domains are separable and benefit from specialist context. Structured parallel handoffs preserve independence and keep the coordinator's context focused.

3. In a multi-agent research system, two specialists discover contradictory revenue figures. Who should resolve the conflict?

A) Whichever agent finished first

B) The coordinator, using cited evidence and a defined conflict-resolution policy, with human escalation if material uncertainty remains

C) Both specialists should silently average the figures

D) The final writer should choose the larger figure

Answer: B. Cross-specialist conflicts belong at the orchestration layer, where evidence can be compared consistently and material uncertainty can trigger human review.

4. An account agent may draft changes freely but must never suspend a customer without authorization. Which safeguard provides the strongest guarantee?

A) A tool-level policy that blocks suspension unless a validated approval token is present

B) Several examples where suspension is avoided

C) A lower temperature

D) A system-prompt sentence asking it to be careful

Answer: A. A deterministic authorization check at the action boundary enforces the invariant. Prompting can guide behavior but cannot guarantee authorization.

5. A production agent can repeatedly search when evidence is weak. What architecture prevents unbounded loops while preserving useful autonomy?

A) End after every first tool call

B) Remove all observability to reduce latency

C) Let it run until it feels finished

D) Track explicit stop reasons, tool-call budgets, elapsed time, repeated-call detection, and an escalation path

Answer: D. Agent loops need explicit termination and budget policies. Repetition detection and escalation handle failure modes that model instructions alone may not stop.

6. An airport operator runs a gate-assignment agent on a fully air-gapped private network with no internet egress. The agent must autonomously reassign gates during irregular operations, but any action that closes a concourse - affecting hundreds of passengers - must require explicit human approval before execution. The team wants the strongest architectural guarantee that concourse-closure actions are never executed autonomously, even if the agent's reasoning concludes they are safe. Which architecture best enforces this lifecycle boundary?

A) Use a high-temperature sampling setting so the agent produces diverse plans and is statistically unlikely to select a concourse-closure action without considering alternatives.

B) Place a deterministic pre-execution hook that intercepts any tool call matching a concourse-closure signature, halts the agent turn, and routes the proposed action to a human-approval queue before the tool is invoked.

C) Wrap the concourse-closure tool in a separate specialist sub-agent that requires a second API call, adding latency that gives operators time to intervene manually.

D) Implement a prompt instruction telling the agent to always ask a human before closing a concourse, relying on the model's instruction-following to prevent autonomous closure.

Answer: B. A pre-execution hook intercepts tool calls before they are dispatched, providing a deterministic, code-enforced boundary that cannot be bypassed by model reasoning - exactly the guarantee needed for irreversible actions. Prompt instructions rely on model compliance and can be overridden by unexpected reasoning paths, making them insufficient as a hard boundary.
High-temperature sampling is a generation parameter unrelated to action gating and introduces unpredictability rather than safety. A second sub-agent call adds latency but does not enforce approval; the sub-agent could still execute autonomously. Hooks are the correct mechanism for enforcing lifecycle boundaries on dangerous tool invocations.

7. A fleet-maintenance provider wants to build a multi-agent system that ingests raw vehicle telemetry CSV files, normalizes units, detects anomalies, and generates maintenance work orders - all with minimal custom code. The team is evaluating how to divide coordinator and specialist responsibilities for the data-preparation stage. Which division of responsibility best minimizes custom code while keeping the coordinator's role appropriately scoped?

A) Embed all data preparation, anomaly detection, and work-order generation inside a single monolithic agent with no sub-agents, eliminating inter-agent communication overhead.

B) Assign a dedicated data-preparation specialist - invoked by the coordinator via a structured tool call - to handle parsing, unit normalization, and schema validation, while the coordinator only sequences specialist invocations and aggregates results.

C) Use the coordinator to write and execute ad-hoc shell scripts for each new telemetry format, dynamically generating transformation code at runtime to avoid maintaining static parsers.

D) Have the coordinator parse and normalize every CSV file itself using inline Python, then pass clean records to specialists, keeping all data transformation logic in one place.

Answer: B. Effective multi-agent design delegates focused, bounded tasks to specialists while the coordinator manages sequencing and aggregation - not implementation details. Assigning data preparation to a specialist invoked via structured tool calls leverages reusable, testable specialist logic and keeps the coordinator thin, minimizing custom glue code. Having the coordinator perform inline transformations bloats its responsibilities and duplicates logic. A monolithic agent eliminates modularity and makes the system harder to extend or debug. Dynamic shell-script generation at runtime maximizes custom code and operational risk. The specialist-coordinator split is the canonical pattern for minimizing bespoke code in multi-agent pipelines.

DOMAIN 2 / TOOL DESIGN AND MCP INTEGRATION

8. A tool named `manage_account` accepts a free-form action string and a large untyped payload. Claude frequently forms unsafe calls. What redesign is best?

A) Add more optional fields

B) Make the description longer but keep the interface

C) Split it into narrow intent-revealing tools with typed parameters, validation, and explicit error contracts

D) Allow arbitrary shell commands instead

Answer: C. Narrow tools reduce ambiguity and make authorization and validation specific to each operation. A broad free-form mutation surface is difficult to reason about safely.

9. A remote MCP server is operated by a third party and returns content that enters the model context. What assumption should the architecture make?

A) All server content is trusted because it uses MCP

B) Server output is untrusted; scope credentials, validate responses, delimit content, and restrict exposed capabilities

C) MCP removes the need for authentication

D) The model can identify every malicious instruction by itself

Answer: B. MCP standardizes connectivity, not trust. Third-party output can be malicious or malformed, so normal authentication, validation, and prompt-injection defenses still apply.

10. A local developer tool needs low-latency access to a process running on the same workstation, while a shared enterprise connector serves many remote clients. What deployment split is reasonable?

A) Use a local process transport for the workstation tool and an authenticated remote transport for the shared service

B) Put shared credentials in CLAUDE.md

C) Expose the local filesystem to every remote client

D) Use the same unauthenticated public endpoint for both

Answer: A. Transport and authentication should match the deployment boundary. Local process communication and shared remote service access have different threat and operational models.

11. A booking tool rejects a request because the selected date is unavailable but suggests three alternatives. What should its error response contain?

A) A stack trace with credentials

B) A fabricated successful booking

C) Only 'failed'

D) A stable machine-readable error code, safe human-readable explanation, and structured alternative dates

Answer: D. Structured recoverable errors let the agent choose a valid next step while preserving safety and observability. Internal secrets and ambiguous messages should not leak.

12. A monorepo has organization-wide security rules and service-specific test commands. How should Claude Code instructions be structured?

A) Put every detail in one global file

B) Keep universal rules at the repository level and place service-specific guidance near each service, avoiding duplication and conflicts

C) Rely on each developer's personal memory

D) Generate instructions anew for every session

Answer: B. Layered instructions mirror scope. Universal guidance remains visible while local files keep service context relevant and maintainable.

13. A team must prevent Claude Code from modifying generated migration files under any circumstances. What is the strongest control?

A) A deterministic pre-action hook or permission rule that rejects writes to the protected path

B) A few examples of safe edits

C) A post-commit reminder

D) A comment asking Claude not to edit them

Answer: A. A deterministic pre-action control enforces the protected-path invariant before mutation. Written guidance alone is advisory.

14. Five repositories repeat the same release-audit procedure, while their build commands differ. What should become a reusable skill?

A) One developer's unreviewed chat transcript

B) All permissions set to unrestricted

C) Every repository's full source tree

D) The common release-audit method, with repository-local instructions supplying build-specific commands

Answer: D. A skill should capture the reusable workflow. Repository-specific operational details remain local so the shared procedure does not become brittle.

15. Claude Code opens a pull request after implementing a change. Which CI design best supports reliable agent-assisted delivery?

A) Skip tests to preserve model context

B) Merge automatically when the diff is non-empty

C) Run deterministic tests, linters, security checks, and policy gates, then require appropriate review based on change risk

D) Ask Claude whether its code is correct

Answer: C. External verification provides evidence independent of the agent's self-assessment. Review depth can scale with risk while CI enforces repeatable gates.

16. A sports analytics team is rolling out a Claude Code agent that auto-generates post-game statistical summaries. To avoid a fleet-wide regression, the team wants to expose the new model version to only 10 % of games before a full release. They have a labeled set of 200 historical game summaries graded by analysts. Which data-preparation strategy best supports this canary evaluation while minimizing the risk of distributional mismatch between the canary slice and the full dataset?

A) Use the 10 % of summaries that received the lowest analyst scores so the canary test stresses the hardest cases first.

B) Split the 200 summaries by sport type and route only one sport's games through the canary model to isolate domain variance.

C) Reserve the 20 most recent game summaries as the canary set because recency ensures the slice reflects current team rosters and rule changes.

D) Randomly sample 10 % of the historical summaries, run the new model on that slice, and compare aggregate quality metrics against the baseline model on the same slice before widening the rollout.

Answer: D. A random sample preserves the statistical distribution of the full labeled set, so canary metrics generalize to the broader population - this is the standard evaluation practice described in Anthropic's agent-evaluation guidance, which stresses representative sampling for reliable signal. Recency-only slicing introduces temporal bias (roster/rule changes skew results). Selecting only low-scoring summaries creates a difficulty-biased slice that inflates apparent improvement. Sport-type routing confounds domain variance with model variance, making it impossible to attribute metric changes to the model alone.

17. A compliance extractor must return one of three statuses and cited evidence spans. Which prompt/output design is strongest?

A) Define a strict schema with enumerated statuses and evidence fields, supply representative edge cases, and validate the result

B) Ask for CSV without defining columns

C) Infer status from response tone

D) Request a persuasive essay

Answer: A. A constrained schema makes valid states explicit and supports deterministic validation. Examples clarify edge cases without replacing the contract.

18. A prompt contains policy, task instructions, examples, user data, and retrieved documents in one undifferentiated block. What is the best redesign?

A) Randomize the order each request

B) Remove the policy

C) Add more prose at the end

D) Separate instruction layers and clearly delimit examples and untrusted data so precedence and purpose are explicit

Answer: D. Clear structural boundaries reduce instruction ambiguity and help prevent untrusted data from being treated as higher-priority guidance.

19. Two prompts perform similarly on average, but one fails catastrophically on rare high-risk inputs. Which should the architecture select?

A) Ignore rare cases because they are uncommon

B) The one with the slightly higher average score

C) The one that meets the risk-weighted release threshold on critical cases, even if its average is marginally lower

D) Choose randomly

Answer: C. Risk-weighted evals prevent average performance from hiding unacceptable failure modes. Release criteria should reflect consequence, not frequency alone.

20. A prompt has accumulated forty examples and now behaves inconsistently when the examples conflict. What is the best next step?

A) Add forty more examples

B) Curate a small representative non-conflicting set, move enforceable rules into explicit instructions or schemas, and rerun evals

C) Remove all validation

D) Increase randomness

Answer: B. Examples should clarify behavior, not become an unmaintainable policy store. Explicit contracts and curated examples reduce contradictions and context cost.

21. A property-management company runs a tenant-communication assistant in production. The team wants to improve the prompt without any service interruption. They plan to test a revised prompt that better handles maintenance-request escalations. The system serves 50,000 active users and cannot tolerate a full cutover until confidence is established. Which evaluation and rollout strategy best satisfies the zero-downtime constraint while generating reliable performance data on the new prompt?

A) Deploy the revised prompt to all users simultaneously, monitor error rates for 24 hours, and roll back if degradation exceeds a threshold.

B) Replace the baseline prompt with the revised prompt in a staging environment only, collect developer feedback for two weeks, and then perform a full production cutover.

C) Run the revised prompt in a shadow or canary configuration - routing a small percentage of live traffic to it - while comparing output quality metrics against the baseline prompt before widening the rollout.

D) Evaluate the revised prompt exclusively on a static offline benchmark dataset, then promote it to production once benchmark scores improve.

Answer: C. Canary or shadow deployment routes a controlled slice of live traffic to the new prompt while the baseline continues serving the majority, enabling real-distribution quality comparisons with no downtime risk. A full simultaneous cutover violates the zero-downtime constraint by exposing all users to an unvalidated change. Static offline benchmarks miss distribution shift on live traffic and cannot confirm production performance. A staging-only evaluation with a hard cutover still creates a single risky promotion event rather than a gradual, data-driven rollout.

22. A long-running support agent loses key commitments after its history is compacted. What should the design change?

A) Repeat the entire transcript forever

B) Ask users to remember commitments for the agent

C) Disable every context limit

D) Maintain an external structured state for commitments and decisions, and rehydrate only the relevant state each turn

Answer: D. Critical state should not live only in an eventually compressed transcript. An external structured record survives compaction and supports targeted context assembly.

23. A model often overlooks a critical rule buried in a long retrieved packet. Which mitigation is best to test first?

A) Increase output length

B) Add more unrelated chunks

C) Retrieve fewer higher-relevance chunks and place the key rule near the task with an explicit citation requirement

D) Hide the rule in metadata

Answer: C. Relevant context and deliberate placement reduce distraction and lost-in-the-middle effects. Citation requirements make use of the rule observable.

24. An agent fails after completing two of four external actions. What information is required for safe recovery?

A) Only the final error message

B) Durable action state, idempotency keys, tool inputs and outcomes, and a policy for retry, compensate, or escalate

C) The model's hidden reasoning

D) A larger prompt

Answer: B. Recovery depends on authoritative external state and idempotent operations. Logs and explicit compensation policies prevent duplicate or contradictory side effects.

25. A semiconductor design team runs a multi-agent pipeline where a routing agent dispatches sub-agents for netlist analysis, timing closure, and DRC verification. Each sub-agent receives a full copy of the evolving chip specification document, which changes schema every sprint as new IP blocks are added. Token costs are escalating and latency is increasing. Which data-preparation strategy best controls context budget while tolerating frequent schema changes?

A) Extract and inject only the schema-relevant fields each sub-agent actually needs at dispatch time, using a thin adapter layer that maps the current schema version to each agent's required interface.

B) Convert the specification to a fixed canonical schema once per sprint and distribute that frozen snapshot to all sub-agents for the duration of the sprint, ignoring mid-sprint schema updates.

C) Cache the full specification at the orchestrator and re-send it only when a sub-agent explicitly requests a refresh, relying on the sub-agent to signal when its cached copy is stale.

D) Serialize the entire chip specification into a single JSON blob and pass it verbatim to every sub-agent so each has complete context regardless of schema version.

Answer: A. Effective context engineering prescribes injecting only the information a specific agent needs rather than broadcasting full documents. An adapter layer that maps the live schema to each agent's required fields keeps token usage proportional to actual need and isolates sub-agents from schema churn - each adapter absorbs version differences without requiring agent rewrites. Passing the full blob ignores selective injection and inflates cost. Relying on sub-agents to request refreshes shifts budget responsibility to the wrong layer and risks stale reads. Freezing a snapshot per sprint cannot handle mid-sprint schema changes that affect active runs.

Want the rest?

The full CCAR-F practice test on learn.cloudyeti.io has 150 questions, a timed exam mode, and a score by domain so you know what to study next.

Start the free CCAR-F practice test