CLOUDYETI / FREE PDF / CCDV-F

Claude Certified Developer - Foundations (CCDV-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-developer-foundations-practice-test

DOMAIN 1 / AGENTS AND WORKFLOWS

1. A document pipeline always performs the same three steps: extract fields, validate them against a schema, and write an approved record. Which design is the best default?

A) A multi-agent debate for every document

B) A single prompt that performs extraction and directly writes to production

C) A free-form autonomous agent that chooses any next action

D) A deterministic workflow that calls Claude only for the extraction step and uses code for validation and writing

Answer: D. The sequence and control logic are known, so deterministic orchestration is simpler and safer. Claude can handle the ambiguous extraction while code enforces schema and write rules.

2. A custom agent receives an assistant response whose stop reason indicates tool use. What should the application do next?

A) Convert the tool request into plain text for the user

B) End the loop because the assistant already responded

C) Execute the requested allowed tool, append the tool result to the conversation, and call the model again

D) Discard prior messages and start a new conversation

Answer: C. A tool-use stop indicates that the model is waiting for a tool result. The client must execute the authorized tool, return a correctly formed result block, and continue the loop.

3. A coordinator delegates a long security-log investigation to a specialist agent. What should the specialist return to preserve the coordinator's context budget?

A) The entire raw log and full internal transcript

B) A compact evidence-backed finding with severity, cited log locations, uncertainty, and recommended next action

C) Only a one-word verdict

D) A new unrelated task for another agent

Answer: B. A structured, evidence-backed handoff preserves the information needed for the next decision without flooding the coordinator with raw context.

4. A consumer electronics retailer operates in a jurisdiction with strict data residency rules: all customer data must remain within a specific geographic boundary and must never be transmitted to external services without explicit governance approval. The retailer is building a Claude-powered agent loop that can look up order history, initiate returns, and escalate to a human agent. Which governance control most directly enforces the data-residency constraint within the agent loop design?

A) Restrict every tool definition to functions that query only on-premises or in-region data stores, and require a human-approval handoff before any tool call that would transmit customer PII outside the approved boundary.

B) Set the Claude API temperature to zero so the model always produces deterministic tool-call arguments, preventing accidental data leakage through probabilistic outputs.

C) Enable prompt caching on all system prompts so that customer data is stored in Anthropic's cache layer, reducing repeated transmission of sensitive fields across turns.

D) Use the Message Batches API to queue all customer interactions offline and process them in bulk during a nightly window when compliance monitoring is inactive.

Answer: A. Data residency is enforced at the tool layer: by restricting tool definitions to in-region endpoints and inserting a human-approval handoff before any cross-boundary transmission, the architecture never moves PII outside the approved zone. Temperature zero affects output determinism, not data routing. Prompt caching stores content in Anthropic's infrastructure, which would violate residency requirements. The Message Batches API defers processing but does not constrain where data travels; batching during low-monitoring windows is a compliance anti-pattern, not a control.

5. A developer expects the Messages API to remember a user's previous request, but the second call has no context. What is the correct fix?

A) Put the history in the model name

B) Wait for server-side memory to synchronize

C) Send the relevant prior user and assistant turns again in the messages array

D) Reuse only the request ID

Answer: C. Messages API calls are stateless with respect to conversation history. The application owns state and must include the relevant prior turns in each request.

6. A chat UI must show text as it is generated and still record final token usage. Which implementation is appropriate?

A) Poll the same non-streaming request repeatedly

B) Consume the event stream, render content deltas, and read the final message metadata when the stream completes

C) Estimate usage from the number of UI characters

D) Open a new request for every token

Answer: B. Streaming exposes incremental content events and a final completed message with authoritative metadata. Character estimates and repeated requests are inaccurate and wasteful.

7. A weather tool times out during an agent loop. How should the application report this to Claude?

A) Return a tool-result block tied to the original tool-use ID, marked as an error with a concise recoverable message

B) Delete the tool request from history

C) Throw away the whole conversation

D) Pretend the tool returned clear weather

Answer: A. A structured error result preserves protocol continuity and lets Claude explain, retry appropriately, or choose another path. Fabricating success corrupts the agent state.

8. A company needs overnight summaries for 80,000 archived tickets and does not need interactive latency. Which API pattern best fits?

A) A realtime stream kept open overnight

B) One prompt containing every ticket

C) One synchronous request from a browser for all tickets

D) Message Batches with independently identified requests and asynchronous result retrieval

Answer: D. Batch processing is intended for large asynchronous workloads that do not require immediate responses. Independent IDs support reconciliation and retries.

9. A worker may retry after a network timeout while executing a payment-refund tool. What application control is essential?

A) Automatic retries with no limit

B) A higher temperature

C) An idempotency key and server-side duplicate check around the consequential operation

D) A longer tool description only

Answer: C. Network ambiguity can cause a successful action to be repeated. Idempotency must be enforced by the application or tool service, not inferred by the model.

10. Every API request fails immediately with an authentication error before any output tokens are produced. What should the developer inspect first?

A) Prompt wording

B) API credentials, request headers, account access, and endpoint configuration

C) The tool schema descriptions

D) The model's reasoning quality

Answer: B. An authentication failure occurs before model generation. Debug transport and credential configuration before changing prompts or application logic.

11. An application needs a response that downstream code can safely parse into a fixed object. What is the strongest design?

A) Use a supported structured-output schema and still validate the received object before side effects

B) Extract fields with regular expressions from an essay

C) Retry forever when parsing fails

D) Ask for valid JSON in prose and trust the result

Answer: A. Schema-constrained output reduces formatting variance, while application validation remains necessary before using data in consequential operations.

12. A construction engineering firm runs a Claude-powered structural-analysis assistant. During a live session, a tool call to an external load-calculation service returns an HTTP 503 after 30 seconds. The firm has very limited labeled failure data, so automated retry classification is unreliable. What is the most appropriate recovery action to keep the agent loop from stalling indefinitely?

A) Immediately terminate the session and instruct the user to restart, because a 503 means the downstream service is permanently unavailable.

B) Return a tool_result block with is_error set to true and a descriptive message, then let Claude decide whether to retry, use an alternative tool, or ask the user for guidance.

C) Silently drop the tool_result turn and resend the original user message so Claude regenerates the tool_use block from scratch.

D) Cache the partial tool output already received and pass it as a valid tool_result so Claude can continue reasoning on incomplete data without signaling an error.

Answer: B. The Messages API requires that every tool_use block receives a corresponding tool_result block; omitting it or silently re-issuing the user turn breaks the conversation contract. When the result is an error, the spec explicitly supports setting is_error: true with a descriptive message so Claude can reason about the failure and decide next steps - ideal when labeled data for automated classification is scarce. Silently dropping the result leaves the conversation in an invalid state. Terminating the session discards all context unnecessarily. Passing incomplete data as a valid result hides the error from Claude, preventing correct recovery reasoning.

DOMAIN 3 / CLAUDE CODE

13. A repository has build commands and coding conventions that every Claude Code session should follow. Where should the team place them?

A) Only in one developer's chat history

B) In a concise repository CLAUDE.md committed with the project

C) In comments copied into every source file

D) In an untracked temporary file outside the repository

Answer: B. A repository CLAUDE.md provides versioned, shared project instructions. It should stay concise and point to deeper documentation where needed.

DOMAIN 4 / EVAL, TESTING, AND DEBUGGING

14. A prompt change improves five hand-picked examples but user complaints rise after release. What test gap most likely caused this?

A) The eval set was not representative of production inputs and failure modes

B) The prompt was too short

C) The API response was streamed

D) The model was not asked to self-score

Answer: A. Hand-picked successes do not estimate production behavior. A representative regression set should include common cases, edge cases, and known failures.

15. An app handles simple intent classification and occasional complex contract analysis. Which design best balances quality and cost?

A) Use one random model per request

B) Route by input length only

C) Use the most expensive model for every request

D) Use a smaller model for validated routine classification and route complex or uncertain cases to a more capable model

Answer: D. Validated task-based routing applies capability where needed while controlling routine cost. Escalation should use measurable difficulty or confidence signals, not randomness.

16. Thousands of requests share the same long policy manual but have different user questions. What optimization should the developer evaluate?

A) Create a new model for each user

B) Repeat the manual in a different order each time

C) Place the stable reusable prefix first and use prompt caching for that repeated content

D) Increase output tokens

Answer: C. Prompt caching benefits repeated stable prompt prefixes. Keeping reusable content stable and early improves cacheability without changing the task.

17. A team wants to cut inference cost without lowering task success. What should it do first?

A) Downgrade every request immediately

B) Measure cost, latency, and task-quality by request class, then test routing, prompt reduction, caching, or batching against the eval suite

C) Remove all context

D) Judge quality from one demo

Answer: B. Optimization requires a baseline and protected quality metric. Different request classes may need different interventions, so blanket downgrades are risky.

18. A global logistics company runs a nightly compliance pipeline that classifies 50,000 shipment records against a 12,000-token regulatory policy document. Auditors require a traceable record showing exactly which policy text drove each classification decision. The engineering team is evaluating whether to use the Message Batches API with prompt caching enabled on the shared policy prefix. Which integration tradeoff most directly supports the auditors' explainability requirement while controlling cost?

A) Use streaming responses for each record so partial outputs can be captured incrementally, providing auditors with a token-by-token trace of the model's reasoning against the policy.

B) Embed a summarized version of the policy document in every prompt to reduce token count, then submit records synchronously so response latency stays low enough for real-time audit dashboards.

C) Use the Message Batches API with the policy document placed in a cache_control breakpoint; store the batch request ID and per-record custom_id alongside each response to reconstruct which policy version and input produced each output.

D) Submit all 50,000 records as individual synchronous Messages API calls so each response can be logged in real time, avoiding the asynchronous complexity of the Batches API.

Answer: C. The Message Batches API assigns a unique custom_id to every request and returns results keyed to that ID, creating a durable, auditable link between each input record and its output. Pairing this with a cache_control breakpoint on the shared policy prefix caches that prefix across the batch, cutting cost while ensuring every request uses the identical policy text - satisfying both explainability and cost goals. Synchronous individual calls lack the batch-level traceability structure and cost savings. Summarizing the policy destroys fidelity needed for audit. Streaming captures token order, not a structured evidence trail linking policy text to decisions.

DOMAIN 6 / PROMPT AND CONTEXT ENGINEERING

19. A support assistant includes 400 pages of documentation in every request and often misses the relevant rule. What is the best redesign?

A) Shuffle the documents on every request

B) Add more unrelated documents

C) Retrieve a small set of relevant passages, preserve source metadata, and pass those with the question

D) Ask for a longer answer

Answer: C. Targeted retrieval improves relevance and reduces distraction. Source metadata also supports citations and post-generation verification.

20. A summarizer processes customer documents that may contain text such as 'ignore previous instructions.' How should the prompt handle this?

A) Treat document text as trusted instructions

B) Clearly delimit document content as untrusted data and state that instructions inside it must not override system or developer rules

C) Delete every sentence containing the word ignore

D) Ask the document what rules to follow

Answer: B. Retrieved and user-supplied content must be treated as data, not policy. Clear trust boundaries reduce prompt-injection risk, though additional controls may still be needed.

21. A developer is building a Claude-powered research assistant that must answer questions about a 150-page technical specification document. The document is identical for every user session. The developer wants to include the full document in context on every request while keeping response latency and cost manageable across many concurrent sessions. Which prompt-structuring approach best achieves reliable, low-latency, cost-efficient responses when the same large document is shared across all requests?

A) Paste the full document into the user turn of every request, immediately before the user's question, so that Claude always has access to the complete text.

B) Split the document into 10-page chunks and send only the chunk most likely to contain the answer on each request, selected by a keyword match on the user's question before the API call.

C) Place the full document in the system prompt with a cache-control breakpoint positioned at the end of the document text, so that the stable document prefix is stored and reused across requests rather than re-processed each time.

D) Summarize the document down to a few hundred words and place the summary in the system prompt, relying on the condensed version to capture all details needed for accurate answers.

Answer: C. Prompt caching is designed for exactly this pattern: large, stable content that is reused across many requests. Placing the full document in the system prompt and marking the end of that prefix with a cache-control breakpoint allows the API to store the processed key-value representation of the document. Subsequent requests that share the same prefix up to that breakpoint are served from cache, avoiding re-processing the document tokens and reducing both latency and cost. The document's full fidelity is preserved, supporting reliable answers to any question. Placing the document in the user turn immediately before the question means the document appears in a position that changes with every new question; because the prefix up to the cache breakpoint must be identical across requests for a cache hit to occur, this placement makes consistent caching architecturally difficult and typically results in the full document being billed as fresh input tokens on every call. Summarizing the document reduces token cost but sacrifices detail, making the assistant unreliable for questions that depend on precise technical content present only in the original text. Chunking with keyword matching can miss relevant context that spans multiple sections or uses different terminology than the query, introducing reliability failures that undermine the goal of accurate answers across the full document.

DOMAIN 7 / SECURITY AND SAFETY

22. An internal agent can search documents and delete records. Most users only need search. What is the safest authorization model?

A) Give every user both tools and rely on the prompt

B) Enforce least privilege in application and tool authorization, exposing delete only to approved users with confirmation and audit logging

C) Hide the delete tool name

D) Let Claude decide each user's role

Answer: B. Authorization must be enforced outside the model. Least privilege, explicit confirmation, and audit records protect consequential operations.

23. A fleet-maintenance provider stores vehicle inspection records in a regulated database. Their Claude-powered agent can both read inspection histories and permanently delete records flagged as duplicates. Regulations require that deletion events be auditable and reversible for 90 days. A developer proposes embedding the database service-account password directly inside the agent's system prompt so that the model can reference it when constructing deletion queries. Which security boundary decision best addresses both the regulatory requirement and the credential exposure risk?

A) Remove delete capability from the agent entirely and require technicians to run all deletions manually through the database UI, with no agent involvement in the deletion workflow.

B) Store the service-account password in the system prompt as proposed, but hash it with SHA-256 before embedding it so the plaintext is not directly visible in logs.

C) Store the service-account password in a dedicated secrets manager outside the prompt, and gate every delete action behind an explicit human-approval step that logs the approver identity and timestamp before execution.

D) Rotate the service-account password every 24 hours and include the current password in the system prompt, relying on short credential lifetime to limit the impact of any exposure.

Answer: C. Credentials must never appear in prompts. Prompt content can be logged by the application layer, cached by the API, echoed back in model outputs, or exposed through prompt-injection attacks - any of these paths would leak the password. A secrets manager keeps credentials entirely outside the conversation, injecting them only at the application layer when a specific, authorized action is about to execute. Separately, permanent deletion is a consequential and difficult-to-reverse action. Requiring an explicit human-approval step before each deletion - with the approver's identity and a timestamp recorded - creates the auditable, reversible-within-90-days trail the regulation demands. These two controls together satisfy both requirements. Hashing the password before embedding it in the system prompt does not protect the credential: the hash is still present in prompt logs and caches, and a hash of a known password can be used to verify guesses or, for weak passwords, reversed. The plaintext is also still needed by the database driver, so the application must store it somewhere anyway. Rotating the password daily reduces the window of exposure after a leak but does not prevent the leak itself. The password is still written into the prompt on every request and is therefore still exposed to every logging and caching pathway. Removing delete capability entirely eliminates the credential-in-prompt risk but also eliminates a legitimate operational function. The correct approach is to govern the capability with proper secrets handling and human approval, not to abandon it.

DOMAIN 8 / TOOLS AND MCPS

24. Claude repeatedly chooses `search_customers` when it should use `get_customer_by_id`. What is the best first improvement?

A) Make both tool names, descriptions, parameters, and use boundaries precise and non-overlapping

B) Increase maximum output length

C) Retry the same call silently

D) Add random examples to the user prompt

Answer: A. Tool descriptions and schemas are part of the model's selection interface. Clear, mutually distinct contracts directly address systematic misselection.

25. Several Claude clients need standardized access to the same read-only asset catalog. Which approach best reduces duplicated integration code?

A) Copy the catalog into every repository

B) Give each client direct database administrator access

C) Embed catalog credentials in every prompt

D) Expose the catalog through a centrally maintained MCP server with scoped authentication and read-only tools or resources

Answer: D. MCP provides a reusable integration boundary across compatible clients. Centralized scoped authentication and read-only capabilities reduce duplicated code and excessive privilege.

Want the rest?

The full CCDV-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 CCDV-F practice test