PassQueen
a.k.a dumpsqueen.com

NVIDIA Agentic AI

NVIDIA NCP-AAI

Version Demo

Total Demo Questions: 15

Total Premium Questions: 155

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Agentic AI Foundations	8
Topic 2, Agentic AI Architecture and Design	45
Topic 3, Agentic AI Development	13
Topic 4, Agentic AI Evaluation	40
Topic 5, Agentic AI Operations	48
Topic 6, Mix Questions	1
Total	155

QUESTION NO: 1

Which two deployment patterns are MOST suitable for scaling agentic workloads on NVIDIA Infrastructure? (Choose two.)

A. Bare metal deployment with manual resource allocation

B. Static virtual machine deployment with fixed resources

C. Serverless deployment without GPU acceleration

D. Containerized deployment with NIM (NVIDIA Inference Microservices)

E. Kubernetes orchestration with Horizontal Pod Autoscaling (HPA)

ANSWER: D E

Explanation:

Containerized deployment with NIM (NVIDIA Inference Microservices) is suitable because NIM packages optimized, production-ready inference services into containers that can be consistently deployed across NVIDIA-accelerated environments. This supports the modular architecture common in agentic systems, where an application may use separate model services for reasoning, embeddings, reranking, retrieval, or guardrails. Containerized services provide repeatable deployment, versioning, health management, and independent resource allocation, allowing GPU-backed components to be operated as scalable services rather than as tightly coupled application processes. NVIDIA documents NIM deployment and operation on Kubernetes environments in its NIM Kubernetes deployment guide.

Kubernetes orchestration with Horizontal Pod Autoscaling (HPA) is also suitable because agentic workload demand can vary substantially with concurrent users, multi-step reasoning, tool calls, and token generation length. Kubernetes provides scheduling, service discovery, rolling updates, and recovery for containerized inference endpoints, while HPA can adjust replica counts based on observed utilization or custom metrics. This enables inference capacity to expand and contract operationally while preserving availability. In practice, GPU-aware capacity planning and relevant application or inference metrics should accompany HPA policies. NVIDIA's NIM Operator documentation describes Kubernetes-native lifecycle management for NIM services, reinforcing this scalable deployment pattern.

QUESTION NO: 2

In a ReAct (Reasoning-Acting) agent architecture, what is the correct sequence of operations when the agent encounters a complex multi-step problem requiring external tool usage?

A. Thought -- > Answer -- > Action -- > Observation

B. Action -- > Thought -- > Observation -- > Action -- > Thought -- > Observation -- > Answer

C. Observation -- > Thought -- > Action -- > Observation -- > Thought -- > Action -- > Answer

D. Thought -- > Action -- > Observation -- > Thought -- > Action -- > Observation -- > Answer

ANSWER: D

Explanation:

The sequence "Thought -- > Action -- > Observation -- > Thought -- > Action -- > Observation -- > Answer" correctly represents the iterative control loop used by a ReAct agent. The agent first forms a reasoning step that identifies what information or operation is needed. It then performs an action, such as calling a search service, database, API, calculator, or another approved tool. The tool's returned result becomes an observation, which supplies grounded information for the next reasoning step. For a complex task, this loop repeats: each new thought incorporates the latest observation, selects the next useful action, and evaluates the resulting evidence. Once the observations provide sufficient support to complete the task, the agent produces its final answer. This alternation is central to ReAct because it combines explicit reasoning with interaction in an environment, allowing the agent to adapt its plan as tool results arrive rather than committing to an answer before obtaining needed evidence. The original ReAct work describes trajectories as interleaved reasoning traces and actions, with observations from those actions informing subsequent reasoning. See the ReAct paper and NVIDIA's NIM function-calling documentation for the tool-invocation pattern used in agent workflows.

QUESTION NO: 3

An agent can use a document-analysis tool to summarize uploaded files. Some uploaded files contain text instructing the agent to reveal its system instructions and send confidential records to an external address.

Which two design practices are MOST effective for reducing the risk that untrusted document content controls the agent workflow? (Choose two.)

A. Treat uploaded and retrieved document text as untrusted data that cannot override system or workflow instructions.

B. Require explicit authorization and schema validation for sensitive tool actions such as external data transmission.

C. Give the document-analysis tool access to all available tools so it can resolve embedded requests autonomously.

D. Increase the model temperature so it is less likely to follow any one instruction in a document.

E. Remove system instructions after the document has been retrieved to reduce competing context.

ANSWER: A B

Explanation:

Retrieved and uploaded content must be handled as untrusted data, not as instructions with authority equal to system policy or user-approved workflow rules. Clear separation of instruction sources helps the agent and orchestration layer avoid treating embedded text as permission to alter goals or execute actions.

Tool actions should also be constrained by explicit authorization and schemas. An agent may summarize a document, but sending records externally should require an approved capability, validated parameters, and, where appropriate, human confirmation. Output filtering alone is insufficient because the unsafe behavior can occur through a tool call before a final response is generated.

QUESTION NO: 4

In a production agentic system handling thousands of concurrent conversations, which state management strategy provides optimal performance while ensuring context preservation?

A. Global shared state with locks for concurrent access

B. Session-isolated state with serialization and lazy loading

C. Stateless design with context reconstruction from message history

ANSWER: B

Explanation:

Session-isolated state with serialization and lazy loading is the appropriate strategy for a high-concurrency agentic application because each conversation retains an independent, durable record of the context required to continue its work. Isolation prevents one user's messages, tool outputs, plans, or preferences from being exposed to or overwritten by another concurrent session. Serialization allows that state to be safely persisted outside an individual application worker, so requests can be distributed across replicas and a conversation can resume after failover or routing to a different worker. Lazy loading further improves production performance: the system retrieves only the relevant state when a session becomes active rather than holding every conversation's full context in process memory. In practice, the agent can maintain a compact session record, summaries, and references to retrieved artifacts, then assemble the relevant working context for the current turn. This supports scalable serving while preserving multi-turn continuity and controlling prompt-context cost. NVIDIA's agent-development guidance treats memory, retrieval, tools, and workflow state as orchestration concerns, and its deployment guidance emphasizes scalable, resilient inference services. See the NVIDIA AIQ Toolkit documentation and NVIDIA NIM LLM documentation.

QUESTION NO: 5

You are implementing a RAG (Retrieval-Augmented Generation) solution.

What is the primary purpose of implementing semantic guardrails within a RAG system?

A. To establish rules and constraints based on the meaning of user queries and generated responses.

B. To eliminate all potential harmful entries from the vector database.

C. To automatically translate all LLM responses into multiple languages for improved user comprehension.

D. To filter out all queries containing specific keywords that have been flagged as problematic.

ANSWER: A

Explanation:

To establish rules and constraints based on the meaning of user queries and generated responses. is correct because semantic guardrails assess intent, context, and meaning rather than relying only on literal word matches. In a RAG workflow, this enables the application to determine whether an incoming request is within an allowed scope, whether retrieved context is appropriate to use, and whether a proposed response complies with organizational safety, privacy, factuality, and topic-control requirements. Guardrails can therefore intercept, redirect, or constrain interactions before an unsafe or out-of-policy answer is delivered.

This meaning-aware control is especially important for RAG because the system combines user input, externally retrieved material, and model-generated text. Each stage can introduce risks that are not detectable through simple string filtering: a request may be indirectly attempting to bypass policy, retrieved content may be irrelevant or untrusted for the requested task, or an answer may be inconsistent with the available evidence. NVIDIA NeMo Guardrails supports configurable rails for input and output handling, as well as retrieval-related controls in RAG applications. These controls provide a policy enforcement layer around the model interaction while preserving the ability to use retrieval for grounded answers. See the NVIDIA NeMo Guardrails documentation and its guardrails library guide.

QUESTION NO: 6

When analyzing memory-related performance degradation in agents handling extended customer support sessions, which evaluation methods effectively identify optimization opportunities for context retention? (Choose two.)

A. Clear memory after each interaction and reset session state, removing historical context needed for personalized tasks to identify optimization opportunities.

B. Profile memory access patterns by measuring retrieval latency, relevance scoring accuracy, and storage efficiency while monitoring context window utilization to identify optimization opportunities.

C. Use fixed memory allocation including all conversation types, topic changes, and user needs, allowing adaptive-free observation of interaction patterns to identify optimization opportunities.

D. Implement sliding window analysis comparing context compression strategies, summarization quality, and information preservation rates across varying conversation lengths to identify optimization opportunities.

E. Store all conversation history including all interactions, allowing adaptive-free observation of data to identify optimization opportunities.

ANSWER: B D

Explanation:

“Profile memory access patterns by measuring retrieval latency, relevance scoring accuracy, and storage efficiency while monitoring context window utilization to identify optimization opportunities.” is correct because long-running agent sessions require measurement of both memory-system behavior and the effective use of the model’s finite context. Retrieval latency exposes responsiveness bottlenecks, relevance accuracy shows whether recalled memories help the current support task, and storage and context-window measurements reveal inefficient retention policies. These are actionable operational and quality signals for tuning retrieval, ranking, expiration, and memory selection.

“Implement sliding window analysis comparing context compression strategies, summarization quality, and information preservation rates across varying conversation lengths to identify optimization opportunities.” is also correct because support conversations often outgrow the available context window. Evaluating summaries and compression over increasing session lengths tests whether durable customer facts, prior troubleshooting steps, constraints, and unresolved issues remain available without excessive token use. A sliding-window approach makes it possible to compare retained-information quality against context cost as the conversation evolves. This aligns with NVIDIA guidance to evaluate retrieval quality and end-to-end RAG behavior using measurable metrics, while managing context efficiently for agent workflows. See the NVIDIA RAG evaluation documentation and the NVIDIA enterprise RAG pipeline guidance.

QUESTION NO: 7

When evaluating optimization opportunities between NeMo Guardrails, NIM microservices, and TensorRT-LLM in a production healthcare agent, which analysis approach best identifies optimization opportunities across the NVIDIA stack?

A. Conduct stress testing of individual microservices and guardrails to measure peak throughput and determine theoretical performance limits of each module.

B. Use default configurations to establish a deployment baseline, focusing on stability before conducting deeper performance profiling.

C. Create end-to-end latency waterfalls that capture guardrail overhead, NIM queuing delays, and TensorRT optimization benefits while assessing overall pipeline efficiency.

D. Tune each component individually, focusing primarily on local performance metrics with secondary attention to integration patterns.

ANSWER: C

Explanation:

Create end-to-end latency waterfalls that capture guardrail overhead, NIM queuing delays, and TensorRT optimization benefits while assessing overall pipeline efficiency. This is the appropriate analysis because a healthcare agent request traverses multiple dependent stages, and the user-visible result is determined by the accumulated latency and behavior of the complete request path rather than inference execution alone. A waterfall view can attribute time to guardrail checks, service-to-service communication, request queueing and dynamic batching, model execution, and response processing. That attribution enables engineers to identify the stage that limits the production service and to evaluate whether a change improves both the relevant component and the overall agent experience. NVIDIA Triton exposes metrics for request, queue, compute-input, compute-infer, and compute-output timing, providing the measurements needed to distinguish scheduling pressure from inference work. NIM is designed to deploy optimized inference microservices, while TensorRT-LLM supplies optimized LLM inference capabilities; assessing them in the full pipeline ensures their benefits are measured in the context in which users experience them. This approach also supports realistic latency objectives for healthcare workflows, where guardrail processing and operational reliability are integral parts of serving an agent. See the NVIDIA Triton metrics documentation and NVIDIA NIM for LLMs documentation.

QUESTION NO: 8

When analyzing user feedback patterns to improve a technical documentation agent, which evaluation methods effectively translate feedback into actionable optimization strategies? (Choose two.)

A. Collect broad user feedback as-is, enabling rapid accumulation of suggestions and diverse perspectives for potential future analysis.

B. Design iterative feedback loops with version tracking, A/B testing of improvements, and regression monitoring to ensure changes enhance rather than degrade performance

C. Incorporate user suggestions rapidly to maximize responsiveness and demonstrate continuous adaptation to evolving user needs.

D. Implement feedback categorization systems grouping issues by type (accuracy, clarity, completeness) with quantitative impact scoring and improvement prioritization matrices

ANSWER: B D

Explanation:

“Design iterative feedback loops with version tracking, A/B testing of improvements, and regression monitoring to ensure changes enhance rather than degrade performance” is effective because it turns feedback-driven changes into measurable, controlled releases. Version tracking preserves the relationship between a specific agent configuration, its prompts or retrieval settings, and observed user outcomes. Controlled comparisons establish whether a proposed revision improves the intended experience, while regression monitoring protects previously successful behaviors as the documentation agent evolves. This supports a repeatable optimization cycle rather than relying on anecdotal feedback.

“Implement feedback categorization systems grouping issues by type (accuracy, clarity, completeness) with quantitative impact scoring and improvement prioritization matrices” is equally essential because raw feedback must be structured before teams can make reliable engineering decisions. Categorization identifies the underlying failure pattern, and impact scoring helps prioritize changes that affect user success, response quality, or operational risk most significantly. Together, these practices connect feedback collection to an auditable backlog, measurable evaluation criteria, and validated agent updates. NVIDIA’s evaluation guidance emphasizes systematic assessment of model and application behavior, while its agent tooling supports observability and iterative improvement across agent workflows. See NVIDIA NeMo Evaluator documentation and NVIDIA NIM for large language models documentation.

QUESTION NO: 9

When analyzing an agent’s failure to complete multi-step financial analysis tasks, which evaluation approach best identifies prompt engineering improvements needed for reliable task decomposition and execution?

A. Implement systematic prompt testing with chain-of-thought reasoning templates, step-by-step decomposition analysis, and success rate tracking across tasks of varying complexity.

B. Focus primarily on response speed optimization as a primary focus over reasoning quality, step completion accuracy, and prompt clarity for complex analytical requirements.

C. Test only final output accuracy as this will automatically include intermediate reasoning steps, decomposition quality, and prompt structure effectiveness for complex workflows.

D. Rely on generic prompt templates which are by default already optimized for general use, instead of tailoring them to financial terminology, calculation needs, or specialized multi-step analysis patterns.

ANSWER: A

Explanation:

Implement systematic prompt testing with chain-of-thought reasoning templates, step-by-step decomposition analysis, and success rate tracking across tasks of varying complexity. This approach evaluates the behavior that matters for an agentic, multi-step workflow: whether the prompt reliably elicits a usable plan, preserves the required financial context and constraints across steps, selects and sequences actions appropriately, and produces a completed result. A representative test set spanning straightforward calculations, multi-document analysis, exception handling, and longer task chains makes it possible to identify where reliability declines as task complexity rises. Tracking completion and success metrics over repeated runs also provides evidence that a prompt change improves the workflow rather than merely helping an isolated example. Examining the task decomposition and intermediate execution trace is especially valuable because an agent can fail before its final response—for example, by omitting a required calculation, using an incomplete subtask plan, or failing to recover from a tool-result issue. This diagnostic evidence supports targeted prompt refinements, such as clarifying subtask boundaries, specifying validation checks, and requiring structured intermediate outputs. NVIDIA’s agentic-AI guidance emphasizes evaluation, tracing, and observability as core practices for understanding and improving agent behavior in production workflows. See the NVIDIA NIM agent documentation and NVIDIA NeMo Evaluator overview.

QUESTION NO: 10

You’re evaluating the RAG pipeline by comparing its responses to synthetic questions. You’ve collected a large set of similarity scores.

What’s the primary benefit of aggregating these scores into a single metric (e.g., average similarity)?

A. Aggregation identifies the specific chunks within the RAG pipeline that are contributing to the highest similarity scores.

B. Aggregation reduces the complexity of the evaluation process and allows for a more overall assessment of the pipeline's effectiveness.

C. Aggregation provides a more accurate representation of the RAG pipeline's performance.

D. Aggregation eliminates the need for qualitative analysis of the RAG pipeline's responses.

ANSWER: B

Explanation:

Aggregation reduces the complexity of the evaluation process and allows for a more overall assessment of the pipeline's effectiveness. A large collection of per-question similarity scores is useful for diagnosis, but it is difficult to interpret quickly, compare consistently across runs, or use as a concise release-quality signal. Summarizing the scores into a metric such as mean similarity produces a repeatable benchmark that can be tracked over time. This makes it practical to compare alternative embedding models, chunking strategies, retrieval settings, prompts, or model versions using the same synthetic evaluation set. A single aggregate score also helps teams detect regressions and communicate overall evaluation results without requiring every stakeholder to inspect each individual response.

The metric should be calculated against a fixed, representative dataset and retained with the pipeline configuration and experiment results. In operational RAG evaluation, aggregate metrics provide the high-level health signal, while the underlying individual scores remain available for investigating outliers and failure patterns. NVIDIA's RAG documentation describes the RAG pipeline components whose configuration and performance can be evaluated systematically, and NVIDIA's evaluation guidance emphasizes measurable, repeatable assessment workflows. See the NVIDIA RAG documentation and NVIDIA NeMo Evaluator overview.

QUESTION NO: 11

When analyzing suboptimal agent response quality after deployment, which parameter tuning evaluation methods effectively identify the optimal configuration adjustments? (Choose two.)

A. Design ablation studies systematically varying individual parameters while holding others constant to isolate each parameter's impact on agent behavior and performance.

B. Apply identical parameter settings across all agent types and tasks, promoting consistency and simplifying comparison across different use cases.

C. Implement A/B testing frameworks comparing temperature, top-k, and top-p variations while measuring task-specific quality metrics and user satisfaction scores.

D. Use production traffic directly for parameter experiments, enabling real-world insights and faster identification of impactful settings.

E. Randomly adjust all parameters simultaneously, allowing for broader exploration of the parameter space in a shorter time frame.

ANSWER: A C

Explanation:

Design ablation studies systematically varying individual parameters while holding others constant to isolate each parameter's impact on agent behavior and performance. This method establishes a causal basis for tuning: by changing one decoding or agent parameter at a time against a stable task set, evaluators can attribute observed differences in answer quality, tool-use reliability, groundedness, latency, or cost to the parameter being tested. That makes the resulting configuration decision repeatable rather than anecdotal.

Implement A/B testing frameworks comparing temperature, top-k, and top-p variations while measuring task-specific quality metrics and user satisfaction scores. Controlled comparisons validate whether a promising configuration produces a meaningful improvement under representative deployment conditions. For an agentic workflow, the measurements should be aligned to the actual task outcome, such as successful task completion and correct tool invocation, and can be

complemented by user feedback. NVIDIA's evaluation tooling emphasizes using defined evaluators and datasets to measure AI-system quality systematically, while NVIDIA NIM documentation describes configurable generation behavior at inference time. Together, ablation and controlled A/B evaluation provide both isolated diagnosis and deployment-relevant validation for configuration adjustments. See the NVIDIA NeMo Evaluator documentation and NVIDIA NIM LLM inference documentation.

QUESTION NO: 12

A support agent uses long-term memory for customer preferences and prior case history. The team wants to reduce irrelevant recalls and prevent obsolete issue details from affecting future interactions.

Which two memory-management practices are MOST appropriate? (Choose two.)

A. Attach provenance, timestamps, and expiry or review policies to persisted memory records.

B. Retrieve memory using customer identity, current-task context, and metadata filters.

C. Append every conversation turn verbatim to the long-term memory store.

D. Delete all customer memory whenever any support case is marked resolved.

E. Select memories solely by embedding similarity without applying customer or status constraints.

ANSWER: A B

Explanation:

Attaching provenance, timestamps, and expiry or review policies to memory records enables the system to distinguish durable preferences from time-sensitive case details. Retrieving with customer identity, task context, and metadata filters limits recall to records relevant to the current interaction. Together, these controls improve relevance while reducing the risk that old case information is presented as current.

Appending every conversation verbatim increases storage and retrieval noise. Deleting all memory after a resolved case removes durable preferences that may remain useful, and selecting memories only by embedding similarity ignores important ownership and lifecycle constraints.

QUESTION NO: 13

You are designing an AI-powered drafting assistant for contract lawyers. The assistant suggests standard clauses and highlights potential risks based on past agreements. Senior attorneys must review, accept, modify, or reject each suggestion, see why a clause was recommended, and provide feedback to help improve the assistant.

Which design feature is most critical for enabling effective human-in-the-loop oversight, transparency, and trust?

A. Display suggested clauses with links to additional details about provenance and risk highlighting in a side panel, allowing users to access more context as needed.

B. Insert suggested clauses into the draft and highlight changes for review at the end, inviting users to provide detailed feedback on clauses they wish to flag for improvement.

C. Present batch "accept all" or "reject all" controls for suggested clauses, with explanations and feedback collected in a summary report after draft review.

D. Show inline "why" explanations for each suggestion, highlight precedent and risk factors, and include accept/modify/reject controls with immediate feedback capture for model refinement.

ANSWER: D

Explanation:

Show inline "why" explanations for each suggestion, highlight precedent and risk factors, and include accept/modify/reject controls with immediate feedback capture for model refinement. This design embeds meaningful human control at the exact

point where an AI-generated recommendation could affect a legal document. Attorneys can inspect the rationale and supporting precedent, assess the relevant risk indicators, and make a deliberate decision for each individual clause rather than reviewing opaque outputs after the fact. This supports accountability because the attorney's action—accepting, editing, or rejecting a recommendation—is explicit and can be recorded alongside the system's rationale. Immediate feedback capture also turns expert corrections into high-quality signals for evaluation and iterative improvement, while preserving the lawyer's authority over the final draft. NVIDIA's responsible AI guidance emphasizes human oversight, transparency, and traceability as core elements of trustworthy AI systems; user-facing explanations and review controls operationalize those principles in a high-stakes workflow. NVIDIA's AI Enterprise security and governance guidance also stresses monitoring, governance, and controls throughout deployment. See NVIDIA Trustworthy AI and NVIDIA AI Enterprise Security.

QUESTION NO: 14

A multi-agent claims-processing application runs several worker replicas to handle variable demand. Workers must survive restarts without losing a claim's current workflow state, and duplicate delivery of a work message must not result in the same claim action being performed twice.

Which two architectural practices are MOST appropriate? (Choose two.)

A. Persist workflow state in a durable shared store that workers can recover after a restart.

B. Use idempotency keys or deduplication records for state-changing claim actions.

C. Store each claim's current state only in the memory of the worker processing it.

D. Assume duplicate message delivery cannot occur when workers are deployed on Kubernetes.

E. Increase the number of worker replicas without changing workflow-state handling.

ANSWER: A B

Explanation:

Persist workflow state outside ephemeral worker containers in a durable, shared state store. This allows a replacement worker to recover the claim state after a restart and supports horizontal scaling without tying correctness to one running replica.

Idempotency keys or deduplication records should be applied to state-changing claim actions. Message systems can redeliver work after failures, so a worker must detect that an action was already completed before repeating it. Autoscaling alone improves capacity but does not provide durable state or protect against duplicate side effects.

QUESTION NO: 15

A financial-services agent retrieves policy documents before recommending a customer action. During an audit, the team finds that the final answer cites a relevant-looking document, but the document was superseded three weeks earlier.

Which design change most effectively prevents this type of grounded but outdated response?

A. Increase the model context window so that more policy documents can be compared during generation.

B. Store document version, effective date, and supersession status as retrieval metadata, then filter or rank results using those fields.

C. Fine-tune the reasoning model periodically on all historical policy documents.

D. Lower the vector-search similarity threshold so that the agent retrieves a broader set of documents.

ANSWER: B

Explanation:

Store document version, effective date, and supersession status as retrieval metadata, then filter or rank retrieval results using those fields. Semantic similarity alone can retrieve an older document that is topically relevant. Applying authoritative lifecycle metadata ensures that the generation stage receives the current approved policy rather than merely the most similar passage. Re-embedding the corpus without controlling document status would not reliably prevent retrieval of superseded content, while increasing the model context window does not establish source validity.