



Vendor: Amazon

Exam Code: AIP-C01

Exam Name: AWS Certified Generative AI Developer -
Professional (AIP-C01)

Version: DEMO

QUESTION 1

A financial services company is building a customer support application that retrieves relevant financial regulation documents from a database based on semantic similarity to user queries. The application must integrate with Amazon Bedrock to generate responses. The application must search documents in English, Spanish, and Portuguese. The application must filter documents by metadata such as publication date, regulatory agency, and document type.

The database stores approximately 10 million document embeddings. To minimize operational overhead, the company wants a solution that minimizes management and maintenance effort while providing low-latency responses for real-time customer interactions.

Which solution will meet these requirements?

- A. Use Amazon OpenSearch Serverless to provide vector search capabilities and metadata filtering. Integrate with Amazon Bedrock Knowledge Bases to enable Retrieval Augmented Generation (RAG) using an Anthropic Claude foundation model.
- B. Deploy an Amazon Aurora PostgreSQL database with the pgvector extension. Store embeddings and metadata in tables. Use SQL queries for similarity search and send results to Amazon Bedrock for response generation.
- C. Use Amazon S3 Vectors to configure a vector index and non-filterable metadata fields. Integrate S3 Vectors with Amazon Bedrock for RAG.
- D. Set up an Amazon Neptune Analytics database with a vector index. Use graph-based retrieval and Amazon Bedrock for response generation.

Answer: A

Explanation:

Option A is the optimal solution because it provides scalable semantic search, rich metadata filtering, and tight integration with Amazon Bedrock while minimizing operational overhead. Amazon OpenSearch Serverless is designed for high-volume, low-latency search workloads and removes the need to manage clusters, capacity planning, or scaling policies.

With support for vector search and structured metadata filtering, OpenSearch Serverless enables efficient similarity search across 10 million embeddings while applying constraints such as language, publication date, regulatory agency, and document type. This is critical for financial services use cases where relevance and compliance depend on precise filtering.

Integrating OpenSearch Serverless with Amazon Bedrock Knowledge Bases enables a fully managed RAG workflow. The knowledge base handles embedding generation, retrieval, and context assembly, while Amazon Bedrock generates responses using a foundation model. This significantly reduces custom glue code and operational complexity.

Multilingual support is handled at the embedding and retrieval layer, allowing documents in English, Spanish, and Portuguese to be searched semantically without language-specific query logic. OpenSearch's distributed architecture ensures consistent low-latency responses for real-time customer interactions.

QUESTION 2

An ecommerce company is using Amazon Bedrock to build a generative AI (GenAI) application. The application uses AWS Step Functions to orchestrate a multi-agent workflow to produce detailed product descriptions. The workflow consists of three sequential states: a description generator, a technical specifications validator, and a brand voice consistency checker. Each state produces intermediate reasoning traces and outputs that are passed to the next state. The application uses an Amazon S3 bucket for process storage and to store outputs.

During testing, the company discovers that outputs between Step Functions states frequently exceed the 256 KB quota and cause workflow failures. A GenAI Developer needs to revise the application architecture to efficiently handle the Step Functions 256 KB quota and maintain workflow observability. The revised architecture must preserve the existing multi-agent reasoning and acting (ReAct) pattern.

Which solution will meet these requirements with the LEAST operational overhead?

- A. Store intermediate outputs in Amazon DynamoDB. Pass only references between states. Create a Map state that retrieves the complete data from DynamoDB when required for each agent's processing step.
- B. Configure an Amazon Bedrock integration to use the S3 bucket URI in the input parameters for large outputs. Use the ResultPath and ResultSelector fields to route S3 references between the agent steps while maintaining the sequential validation workflow.
- C. Use AWS Lambda functions to compress outputs to less than 256 KB before each agent state. Configure each agent task to decompress outputs before processing and to compress results before passing them to the next state.
- D. Configure a separate Step Functions state machine to handle each agent's processing. Use Amazon EventBridge to coordinate the execution flow between state machines. Use S3 references for the outputs as event data.

Answer: B

Explanation:

Option B is the best solution because it directly addresses the Step Functions 256 KB state payload quota by externalizing large intermediate artifacts to Amazon S3 and passing only lightweight references (URLs/keys) between states. This is a standard AWS pattern for workflows that produce large intermediate results, and it avoids introducing additional databases, compression logic, or cross-state-machine coordination that increases operational overhead.

In a multi-agent ReAct workflow, intermediate reasoning traces can be verbose and grow quickly as each agent produces chain-of-thought style artifacts, structured outputs, and supporting evidence. Step Functions is designed to orchestrate state transitions and pass JSON payloads, but large payloads should be stored outside the state machine and referenced by pointer values. Using Amazon S3 for intermediate outputs is operationally efficient because the application already uses S3 for storage, and S3 provides durable, low-cost storage with simple access patterns.

ResultPath and ResultSelector allow each state to store or reshape results so that only the required reference fields (such as s3Uri, object key, metadata, trace IDs) are forwarded to subsequent states. This preserves observability because the workflow can still log trace references, correlate steps with S3 objects, and store structured metadata for debugging. It also preserves the sequential validation design, keeping the existing ReAct pattern intact while preventing failures due to oversized payloads.

QUESTION 3

A specialty coffee company has a mobile app that generates personalized coffee roast profiles by using Amazon Bedrock with a three-stage prompt chain. The prompt chain converts user inputs into structured metadata, retrieves relevant logs for coffee roasts, and generates a personalized roast recommendation for each customer.

Users in multiple AWS Regions report inconsistent roast recommendations for identical inputs, slow inference during the retrieval step, and unsafe recommendations such as brewing at excessively high temperatures. The company must improve the stability of outputs for repeated inputs. The company must also improve app performance and the safety of the app's outputs.

The updated solution must ensure 99.5% output consistency for identical inputs and achieve inference latency of less than 1 second. The solution must also block unsafe or hallucinated recommendations by using validated safety controls.

Which solution will meet these requirements?

- A. Deploy Amazon Bedrock with provisioned throughput to stabilize inference latency. Apply Amazon Bedrock guardrails with semantic denial rules to block unsafe outputs. Use Amazon Bedrock Prompt Management to manage prompts by using approval workflows.
- B. Use Amazon Bedrock Agents to manage chaining. Log model inputs and outputs to Amazon CloudWatch Logs. Use logs from CloudWatch to perform A/B testing for prompt versions.
- C. Cache prompt results in Amazon ElastiCache. Use AWS Lambda functions to pre-process metadata and to trace end-to-end latency. Use AWS X-Ray to identify and remediate performance bottlenecks.
- D. Use Amazon Kendra to improve roast log retrieval accuracy. Store normalized prompt metadata within Amazon DynamoDB. Use AWS Step Functions to orchestrate multi-step prompts.

Answer: A

Explanation:

Option A is the only choice that simultaneously addresses all three requirements: (1) higher output consistency for identical inputs, (2) sub-1-second performance, and (3) validated safety controls that block unsafe or hallucinated recommendations.

Provisioned throughput in Amazon Bedrock reserves capacity for the chosen model, which helps stabilize latency and reduces the chance of throttling or variable response times across Regions. This is important for a mobile app with strict latency goals and users distributed across multiple Regions. While provisioned throughput primarily improves performance predictability, it also reduces variability caused by contention during peak demand.

Amazon Bedrock guardrails provide validated safety controls to filter or block unsafe content. Semantic denial rules are appropriate for preventing dangerous brewing guidance (for example, excessively high temperatures) and for reducing hallucinated instructions that violate safety policies. Guardrails can be enforced consistently regardless of prompt-chain complexity, providing a uniform safety layer around the model outputs.

Amazon Bedrock Prompt Management supports controlled prompt versioning and approval workflows. By standardizing prompts, controlling changes, and ensuring the same prompt version is used for identical inputs, the company improves output stability and reduces drift caused by unmanaged prompt edits. Combined with strict configuration control (including fixed inference parameters such as temperature where appropriate), this improves repeatability and increases the likelihood of achieving the 99.5% consistency target.

QUESTION 4

A media company must use Amazon Bedrock to implement a robust governance process for AI-generated content. The company needs to manage hundreds of prompt templates. Multiple teams use the templates across multiple AWS Regions to generate content. The solution must provide version control with approval workflows that include notifications for pending reviews. The solution must also provide detailed audit trails that document prompt activities and consistent prompt parameterization to enforce quality standards. Which solution will meet these requirements?

- A. Configure Amazon Bedrock Studio prompt templates. Use Amazon CloudWatch dashboards to display prompt usage metrics. Store approval status in Amazon DynamoDB. Use AWS Lambda functions to enforce approvals.

- B. Use Amazon Bedrock Prompt Management to implement version control. Configure AWS CloudTrail for audit logging. Use AWS Identity and Access Management policies to control approval permissions. Create parameterized prompt templates by specifying variables.
- C. Use AWS Step Functions to create an approval workflow. Store prompts in Amazon S3. Use tags to implement version control. Use Amazon EventBridge to send notifications.
- D. Deploy Amazon SageMaker Canvas with prompt templates stored in Amazon S3. Use AWS CloudFormation for version control. Use AWS Config to enforce approval policies.

Answer: B

Explanation:

Option B is the correct solution because Amazon Bedrock Prompt Management is purpose-built to manage, govern, and standardize prompt usage at scale across teams and Regions. It provides native version control, allowing teams to track prompt changes over time and ensure that only approved versions are used in production workflows.

Prompt Management supports approval workflows that align with enterprise governance requirements. Approval permissions can be enforced through IAM policies, ensuring that only authorized reviewers can approve or publish prompt versions. This removes the need for custom workflow engines or external storage systems, significantly reducing operational overhead.

Parameterized prompt templates enable consistent prompt structure while allowing controlled variation through defined variables. This ensures consistent quality standards and reduces prompt drift, which is critical when hundreds of prompts are reused across multiple applications and teams.

AWS CloudTrail integrates natively with Amazon Bedrock to provide immutable audit logs for prompt creation, updates, approvals, and usage. These detailed audit trails satisfy compliance requirements and allow security and governance teams to trace prompt activity across Regions and users.

QUESTION 5

A company is developing a generative AI (GenAI) application that uses Amazon Bedrock foundation models. The application has several custom tool integrations. The application has experienced unexpected token consumption surges despite consistent user traffic.

The company needs a solution that uses Amazon Bedrock model invocation logging to monitor InputTokenCount and OutputTokenCount metrics. The solution must detect unusual patterns in tool usage and identify which specific tool integrations cause abnormal token consumption. The solution must also automatically adjust thresholds as traffic patterns change.

Which solution will meet these requirements?

- A. Use Amazon CloudWatch Logs to capture model invocation logs. Create CloudWatch dashboards for token metrics. Configure static CloudWatch alarms with fixed thresholds for each tool integration.
- B. Store model invocation logs in Amazon S3. Use AWS Glue and Amazon Athena to analyze token usage trends.
- C. Use Amazon CloudWatch Logs to capture model invocation logs. Create CloudWatch metric filters to extract tool-specific invocation patterns. Apply CloudWatch anomaly detection alarms that automatically adjust baselines for each tool's token metrics.
- D. Store model invocation logs in an Amazon S3 bucket. Use AWS Lambda to process logs in real time. Manually update CloudWatch alarm thresholds based on trends identified by the Lambda function.

Answer: C

Explanation:

Option C best meets the requirements by combining native Amazon Bedrock logging with adaptive monitoring and minimal operational overhead. Amazon Bedrock model invocation logging can be sent directly to CloudWatch Logs, where detailed fields such as InputTokenCount, OutputTokenCount, and tool invocation metadata are captured for each request.

CloudWatch metric filters allow extraction of structured metrics from logs, including tool-specific token consumption patterns. By defining filters per tool integration, the company can isolate which tools are responsible for increased token usage without building custom log-processing pipelines.

CloudWatch anomaly detection provides automatic baseline modeling and dynamic thresholds based on historical traffic patterns. Unlike static alarms, anomaly detection adapts as usage evolves, making it ideal for applications with changing workloads or seasonal usage patterns. This directly satisfies the requirement to automatically adjust thresholds as traffic patterns change.

When abnormal token consumption occurs, anomaly detection alarms trigger immediately, enabling rapid investigation and remediation. Because this solution uses fully managed AWS services without custom analytics jobs or manual threshold tuning, it significantly reduces operational effort.

QUESTION 6

A financial services company needs to pre-process unstructured data such as customer transcripts, financial reports, and documentation. The company stores the unstructured data in Amazon S3 to support an Amazon Bedrock application.

The company must validate data quality, create auditable metadata, monitor data metrics, and customize text chunking to optimize foundation model (FM) performance.

Which solution will meet these requirements with the LEAST development effort?

- A. Use Amazon SageMaker Data Wrangler to create a data flow. Configure Amazon CloudWatch metrics and alarms to monitor data quality. Use a custom AWS Lambda function to pre-process the data. Load processed data into Amazon Bedrock.
- B. Set up an AWS Glue crawler to catalog data sources. Create AWS Glue ETL jobs to run custom transformation scripts. Use AWS Glue Data Quality to validate and monitor data quality. Load processed data into Amazon Bedrock.
- C. Use Amazon Comprehend to extract entities. Create an AWS Lambda function to chunk text. Run Amazon Athena to query and validate data quality. Load processed data into Amazon Bedrock.
- D. Create an AWS Step Functions workflow to orchestrate data pre-processing tasks. Run custom code on Amazon EC2 instances. Use Amazon SageMaker Model Monitor to monitor data quality. Load processed data into Amazon Bedrock.

Answer: B

Explanation:

Option B is the most appropriate solution because it uses AWS-native, purpose-built data engineering and governance services to address data quality validation, metadata creation, monitoring, and transformation with minimal custom development. AWS Glue is designed specifically for large-scale data preparation and integrates seamlessly with Amazon S3, making it ideal for preprocessing unstructured datasets for downstream GenAI applications.

AWS Glue crawlers automatically infer schemas and populate the AWS Glue Data Catalog, creating auditable, queryable metadata for all datasets. This satisfies the requirement for

traceability and governance, which is especially critical in financial services environments. Glue ETL jobs allow teams to implement customizable transformation logic, including text normalization and chunking strategies optimized for foundation model context windows.

AWS Glue Data Quality provides built-in rulesets for validating completeness, accuracy, and consistency. It also publishes quality metrics that can be monitored over time, meeting the requirement for ongoing data quality monitoring without building custom validation frameworks.

Because AWS Glue is fully managed, it eliminates the need to manage infrastructure, scaling, or orchestration. This significantly reduces development and operational effort compared to custom Lambda pipelines or EC2-based processing. The processed and validated data can then be safely ingested into Amazon Bedrock workflows or knowledge bases.

QUESTION 7

An ecommerce company is developing a generative AI (GenAI) solution that uses Amazon Bedrock with Anthropic Claude to recommend products to customers. Customers report that some recommended products are not available for sale or are not relevant. Customers also report long response times for some recommendations.

The company confirms that most customer interactions are unique and that the solution recommends products not present in the product catalog.

Which solution will meet this requirement?

- A. Increase grounding within Amazon Bedrock Guardrails. Enable automated reasoning checks. Set up provisioned throughput.
- B. Use prompt engineering to restrict model responses to relevant products. Use streaming inference to reduce perceived latency.
- C. Create an Amazon Bedrock Knowledge Bases and implement Retrieval Augmented Generation (RAG). Set the PerformanceConfigLatency parameter to optimized.
- D. Store product catalog data in Amazon OpenSearch Service. Validate model recommendations against the catalog. Use Amazon DynamoDB for response caching.

Answer: C

Explanation:

Option C is the correct solution because it directly addresses both correctness and performance issues by grounding the model's responses in authoritative product data using Retrieval Augmented Generation. Amazon Bedrock Knowledge Bases are designed to connect foundation models to trusted enterprise data sources, ensuring that generated responses are constrained to known, validated content.

By ingesting the product catalog into a knowledge base, the GenAI application retrieves only products that actually exist in the catalog. This prevents hallucinated or unavailable recommendations, which is a common issue when models rely solely on prompt instructions without retrieval grounding. RAG ensures that the model's output is based on retrieved facts rather than learned generalizations.

Setting the PerformanceConfigLatency parameter to optimized enables Bedrock to prioritize lower-latency retrieval and inference paths, improving responsiveness for real-time recommendation scenarios. This directly addresses the reported performance issues without requiring provisioned throughput or caching strategies that are ineffective for mostly unique interactions.

QUESTION 8

A financial services company is developing a real-time generative AI (GenAI) assistant to support human call center agents. The GenAI assistant must transcribe live customer speech, analyze context, and provide incremental suggestions to call center agents while a customer is still speaking. To preserve responsiveness, the GenAI assistant must maintain end-to-end latency under 1 second from speech to initial response display. The architecture must use only managed AWS services and must support bidirectional streaming to ensure that call center agents receive updates in real time. Which solution will meet these requirements?

- A. Use Amazon Transcribe streaming to transcribe calls. Pass the text to Amazon Comprehend for sentiment analysis. Feed the results to Anthropic Claude on Amazon Bedrock by using the InvokeModel API. Store results in Amazon DynamoDB. Use a WebSocket API to display the results.
- B. Use Amazon Transcribe streaming with partial results enabled to deliver fragments of transcribed text before customers finish speaking. Forward text fragments to Amazon Bedrock by using the InvokeModelWithResponseStream API. Stream responses to call center agents through an Amazon API Gateway WebSocket API.
- C. Use Amazon Transcribe batch processing to convert calls to text. Pass complete transcripts to Anthropic Claude on Amazon Bedrock by using the ConverseStream API. Return responses through an Amazon Lex chatbot interface.
- D. Use the Amazon Transcribe streaming API with an AWS Lambda function to transcribe each audio segment. Call the Amazon Titan Embeddings model on Amazon Bedrock by using the InvokeModel API. Publish results to Amazon SNS.

Answer: B

Explanation:

Option B is the only solution that satisfies all strict real-time, streaming, and latency requirements. Amazon Transcribe streaming with partial results allows transcription fragments to be delivered before the speaker finishes a sentence. This significantly reduces perceived latency and enables downstream processing to begin immediately, which is essential for maintaining sub-1-second end-to-end response times.

Using Amazon Bedrock's InvokeModelWithResponseStream API enables token-level or chunk-level streaming responses from the foundation model. This allows the GenAI assistant to begin delivering suggestions to call center agents incrementally instead of waiting for a full model response. This streaming inference capability is critical for interactive, real-time agent assistance use cases.

Amazon API Gateway WebSocket APIs provide fully managed, bidirectional communication between backend services and agent dashboards. This ensures that updates flow continuously to agents as new transcription fragments and model outputs become available, preserving real-time responsiveness without requiring custom socket infrastructure.

QUESTION 9

A company is using Amazon Bedrock to build a customer-facing AI assistant that handles sensitive customer inquiries. The company must use defense-in-depth safety controls to block sophisticated prompt injection attacks. The company must keep audit logs of all safety interventions. The AI assistant must have cross-Region failover capabilities. Which solution will meet these requirements?

- A. Configure Amazon Bedrock guardrails with content filters set to high to protect against prompt injection attacks. Use a guardrail profile to implement cross-Region guardrail inference. Use Amazon CloudWatch Logs with custom metrics to capture detailed guardrail intervention events.
- B. Configure Amazon Bedrock guardrails with content filters set to high. Use AWS WAF to block

- suspicious inputs. Use AWS CloudTrail to log API calls.
- C. Deploy Amazon Comprehend custom classifiers to detect prompt injection attacks. Use Amazon API Gateway request validation. Use CloudWatch Logs to capture intervention events.
 - D. Configure Amazon Bedrock guardrails with custom content filters and word filters set to high. Configure cross-Region guardrail replication for failover. Store logs in AWS CloudTrail for compliance auditing.

Answer: A

Explanation:

Option A provides the most complete, AWS-native defense-in-depth solution for protecting against prompt injection attacks while meeting audit and resiliency requirements. Amazon Bedrock guardrails are designed specifically to enforce safety policies on both user inputs and model outputs, including protections against prompt injection and jailbreak attempts.

Setting content filters to high increases sensitivity to malicious or manipulative inputs. Guardrail profiles allow the same guardrail configuration to be applied consistently across multiple Regions, enabling cross-Region inference and failover without configuration drift. This directly satisfies the requirement for regional resilience.

Amazon CloudWatch Logs captures detailed guardrail intervention events, including when content is blocked, modified, or flagged. Custom metrics derived from these logs enable fine-grained auditing, alerting, and reporting on safety enforcement actions. This provides a more detailed audit trail of safety interventions than API-level logs alone.

QUESTION 10

A company upgraded its Amazon Bedrock-powered foundation model (FM) that supports a multilingual customer service assistant. After the upgrade, the assistant exhibited inconsistent behavior across languages. The assistant began generating different responses in some languages when presented with identical questions.

The company needs a solution to detect and address similar problems for future updates. The evaluation must be completed within 45 minutes for all supported languages. The evaluation must process at least 15,000 test conversations in parallel. The evaluation process must be fully automated and integrated into the CI/CD pipeline. The solution must block deployment if quality thresholds are not met.

Which solution will meet these requirements?

- A. Create a distributed traffic simulation framework that sends translation-heavy workloads to the assistant in multiple languages simultaneously. Use Amazon CloudWatch metrics to monitor latency, concurrency, and throughput. Run simulations before production releases to identify infrastructure bottlenecks.
- B. Deploy the assistant in multiple AWS Regions with Amazon Route 53 latency-based routing and AWS Global Accelerator to improve global performance. Store multilingual conversation logs in Amazon S3. Perform weekly post-deployment audits to review consistency.
- C. Create a pre-processing pipeline that normalizes all incoming messages into a consistent format before sending the messages to the assistant. Apply rule-based checks to flag potential hallucinations in the outputs. Focus evaluation on normalized text to simplify testing across languages.
- D. Set up standardized multilingual test conversations with identical meaning. Run the test conversations in parallel by using Amazon Bedrock model evaluation jobs. Apply similarity and hallucination thresholds. Integrate the process into the CI/CD pipeline to block releases that fail.

Answer: D

Explanation:

Option D is the correct solution because it directly evaluates multilingual output consistency and quality in an automated, scalable, and deployment-gating workflow. Amazon Bedrock model evaluation jobs are designed to run large-scale, repeatable evaluations against defined datasets and to produce quantitative metrics that can be used as objective release criteria.

The core issue is semantic inconsistency across languages for equivalent inputs. The most reliable way to detect this is to create standardized test conversations where each language version expresses the same intent and constraints. Running those tests through the updated model and comparing results with similarity metrics (for example, semantic similarity between expected and actual answers, or between language variants) surfaces regressions that infrastructure testing cannot detect.

Bedrock evaluation jobs support running evaluations at scale and are well suited for processing large datasets quickly. By parallelizing evaluation runs across languages and conversations, the company can meet the 45-minute requirement while executing at least 15,000 conversations. Because the process is standardized, it also allows consistent baseline comparisons across releases.

Applying hallucination thresholds ensures that answers remain grounded and do not introduce fabricated details, which is particularly important when language-specific behavior shifts after a model upgrade. Integrating evaluation jobs into the CI/CD pipeline enables fully automated execution on every model or configuration update. The pipeline can enforce a hard quality gate that blocks deployment if thresholds are not met, preventing regressions from reaching production.

QUESTION 11

A financial services company is developing a Retrieval Augmented Generation (RAG) application to help investment analysts query complex financial relationships across multiple investment vehicles, market sectors, and regulatory environments. The dataset contains highly interconnected entities that have multi-hop relationships. Analysts must examine relationships holistically to provide accurate investment guidance. The application must deliver comprehensive answers that capture indirect relationships between financial entities and must respond in less than 3 seconds. Which solution will meet these requirements with the LEAST operational overhead?

- A. Use Amazon Bedrock Knowledge Bases with GraphRAG and Amazon Neptune Analytics to store financial data. Analyze multi-hop relationships between entities and automatically identify related information across documents.
- B. Use Amazon Bedrock Knowledge Bases and an Amazon OpenSearch Service vector store to implement custom relationship identification logic that uses AWS Lambda to query multiple vector embeddings in sequence.
- C. Use Amazon OpenSearch Serverless vector search with k-nearest neighbor (k-NN). Implement manual relationship mapping in an application layer that runs on Amazon EC2 Auto Scaling.
- D. Use Amazon DynamoDB to store financial data in a custom indexing system. Use AWS Lambda to query relevant records. Use Amazon SageMaker to generate responses.

Answer: A

Explanation:

Option A best satisfies the requirement to capture multi-hop, highly interconnected relationships with minimal operational overhead. Traditional vector similarity search excels at finding semantically similar text but is not optimized for reasoning over explicit entity-to-entity relationships, especially when analysts need indirect, multi-hop connections. Graph-based retrieval is designed specifically for these kinds of relationship traversals.

GraphRAG combines retrieval-augmented generation with graph-aware context selection. By representing entities and their relationships in a graph store, the system can traverse multiple hops to assemble a holistic set of relevant facts. This improves completeness and reduces the chance that the model misses indirect relationships that are essential for accurate investment guidance.

Amazon Neptune Analytics provides a managed graph analytics environment capable of efficiently traversing and analyzing complex relationship networks. When integrated with Amazon Bedrock Knowledge Bases, it reduces custom engineering by providing managed ingestion, retrieval, and orchestration patterns suitable for GenAI applications. This lowers operational overhead compared to building and maintaining custom multi-stage retrieval logic.

Meeting the sub-3-second requirement is also more feasible with a graph-optimized engine because multi-hop traversals can be executed efficiently compared to chaining multiple vector searches and joining results in an application layer. The managed nature of Knowledge Bases and Neptune Analytics reduces maintenance, scaling, and operational burden while enabling strong performance.

QUESTION 12

A company is developing a customer support application that uses Amazon Bedrock foundation models (FMs) to provide real-time AI assistance to the company's employees. The application must display AI-generated responses character by character as the responses are generated. The application needs to support thousands of concurrent users with minimal latency. The responses typically take 15 to 45 seconds to finish. Which solution will meet these requirements?

- A. Configure an Amazon API Gateway WebSocket API with an AWS Lambda integration. Configure the WebSocket API to invoke the Amazon Bedrock `InvokeModelWithResponseStream` API and stream partial responses through WebSocket connections.
- B. Configure an Amazon API Gateway REST API with an AWS Lambda integration. Configure the REST API to invoke the Amazon Bedrock standard `InvokeModel` API and implement frontend client-side polling every 100 ms for complete response chunks.
- C. Implement direct frontend client connections to Amazon Bedrock by using IAM user credentials and the `InvokeModelWithResponseStream` API without any intermediate gateway or proxy layer.
- D. Configure an Amazon API Gateway HTTP API with an AWS Lambda integration. Configure the HTTP API to cache complete responses in an Amazon DynamoDB table and serve the responses through multiple paginated GET requests to frontend clients.

Answer: A

Explanation:

This requirement explicitly calls for character-by-character streaming, long-running responses, low latency, and massive concurrency, which aligns directly with Amazon Bedrock streaming inference patterns.

Amazon Bedrock provides the `InvokeModelWithResponseStream` API specifically for streaming partial model outputs as tokens are generated. This enables near-instant feedback to users instead of waiting for the full response to complete, which is essential when responses last up to 45 seconds.

Amazon API Gateway WebSocket APIs are purpose-built for bidirectional, low-latency, server-initiated communication, allowing the backend to push characters or tokens to clients in real time. This eliminates inefficient polling and supports thousands of concurrent open connections.

AWS Lambda integrates natively with WebSocket APIs and scales automatically with connection

volume, enabling a fully managed, serverless architecture. This approach maintains security, centralized authentication, throttling, and observability while avoiding direct client access to Bedrock APIs.

QUESTION 13

An elevator service company has developed an AI assistant application by using Amazon Bedrock. The application generates elevator maintenance recommendations to support the company's elevator technicians. The company uses Amazon Kinesis Data Streams to collect the elevator sensor data.

New regulatory rules require that a human technician must review all AI-generated recommendations. The company needs to establish human oversight workflows to review and approve AI recommendations. The company must store all human technician review decisions for audit purposes.

Which solution will meet these requirements?

- A. Create a custom approval workflow by using AWS Lambda functions and Amazon SQS queues for human review of AI recommendations. Store all review decisions in Amazon DynamoDB for audit purposes.
- B. Create an AWS Step Functions workflow that has a human approval step that uses the `waitForTaskToken` API to pause execution. After a human technician completes a review, use an AWS Lambda function to call the `SendTaskSuccess` API with the approval decision. Store all review decisions in Amazon DynamoDB.
- C. Create an AWS Glue workflow that has a human approval step. After the human technician review, integrate the application with an AWS Lambda function that calls the `SendTaskSuccess` API. Store all human technician review decisions in Amazon DynamoDB.
- D. Configure Amazon EventBridge rules with custom event patterns to route AI recommendations to human technicians for review. Create AWS Glue jobs to process human technician approval queues. Use Amazon ElastiCache to cache all human technician review decisions.

Answer: B

Explanation:

AWS Step Functions provides native support for human-in-the-loop workflows, making it the best fit for regulatory oversight requirements. The `waitForTaskToken` integration pattern is explicitly designed to pause a workflow until an external actor--such as a human reviewer--completes a task.

In this architecture, AI-generated recommendations are sent to a human technician for review. The workflow pauses execution using a task token. Once the technician approves or rejects the recommendation, an AWS Lambda function calls `SendTaskSuccess` or `SendTaskFailure`, allowing the workflow to continue deterministically.

This approach ensures full auditability, as Step Functions records every state transition, timestamp, and execution path. Storing review outcomes in Amazon DynamoDB provides durable, queryable audit records required for regulatory compliance.

Thank You for Trying Our Product

Lead2pass Certification Exam Features:

- ★ More than **99,900** Satisfied Customers Worldwide.
- ★ Average **99.9%** Success Rate.
- ★ **Free Update** to match latest and real exam scenarios.
- ★ **Instant Download** Access! No Setup required.
- ★ Questions & Answers are downloadable in **PDF** format and **VCE** test engine format.
- ★ Multi-Platform capabilities - **Windows, Laptop, Mac, Android, iPhone, iPod, iPad**.
- ★ **100%** Guaranteed Success or **100%** Money Back Guarantee.
- ★ **Fast**, helpful support **24x7**.



View list of all certification exams: <http://www.lead2pass.com/all-products.html>



Microsoft



ORACLE



JUNIPER
NETWORKS



EMC²
where information lives

10% Discount Coupon Code: ASTR14