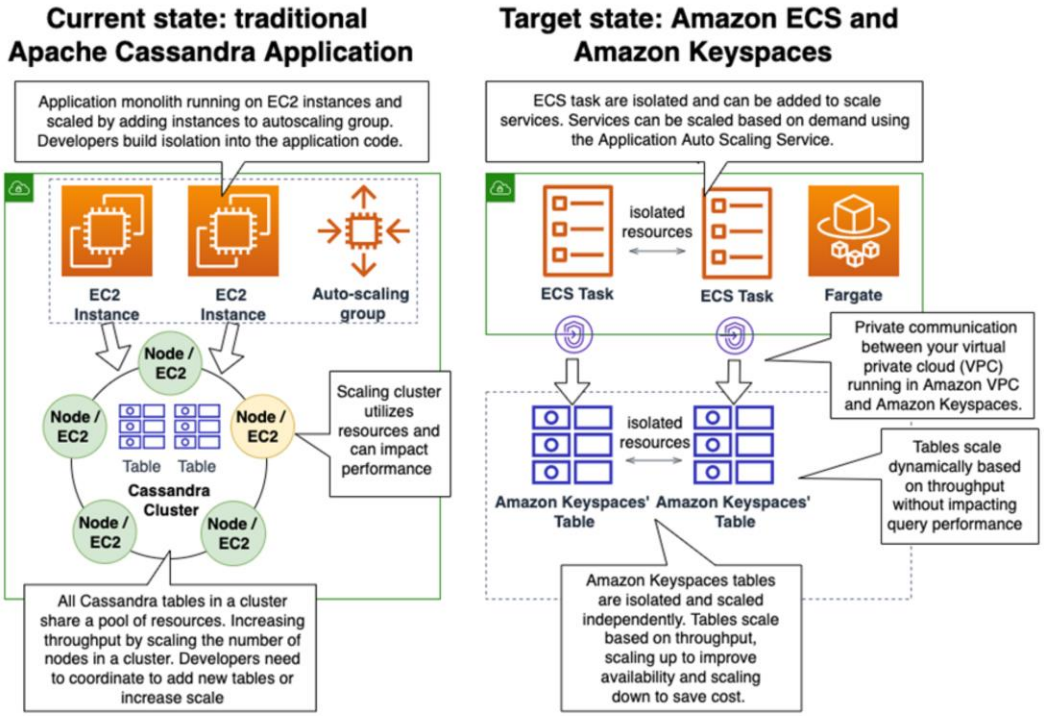
Visit www.kodekloud.com to learn more.

Designing for Reliability – Keyspaces (for Apache Cassandra)

Keyspaces (for Apache Cassandra)



Current state: traditional Apache Cassandra Application

Application monolith running on EC2 instances and scaled by adding instances to autoscaling group. Developers build isolation into the application code.

Scaling cluster utilizes resources and can impact performance

All Cassandra tables in a cluster share a pool of resources. Increasing throughput by scaling the number of nodes in a cluster. Developers need to coordinate to add new tables or increase scale

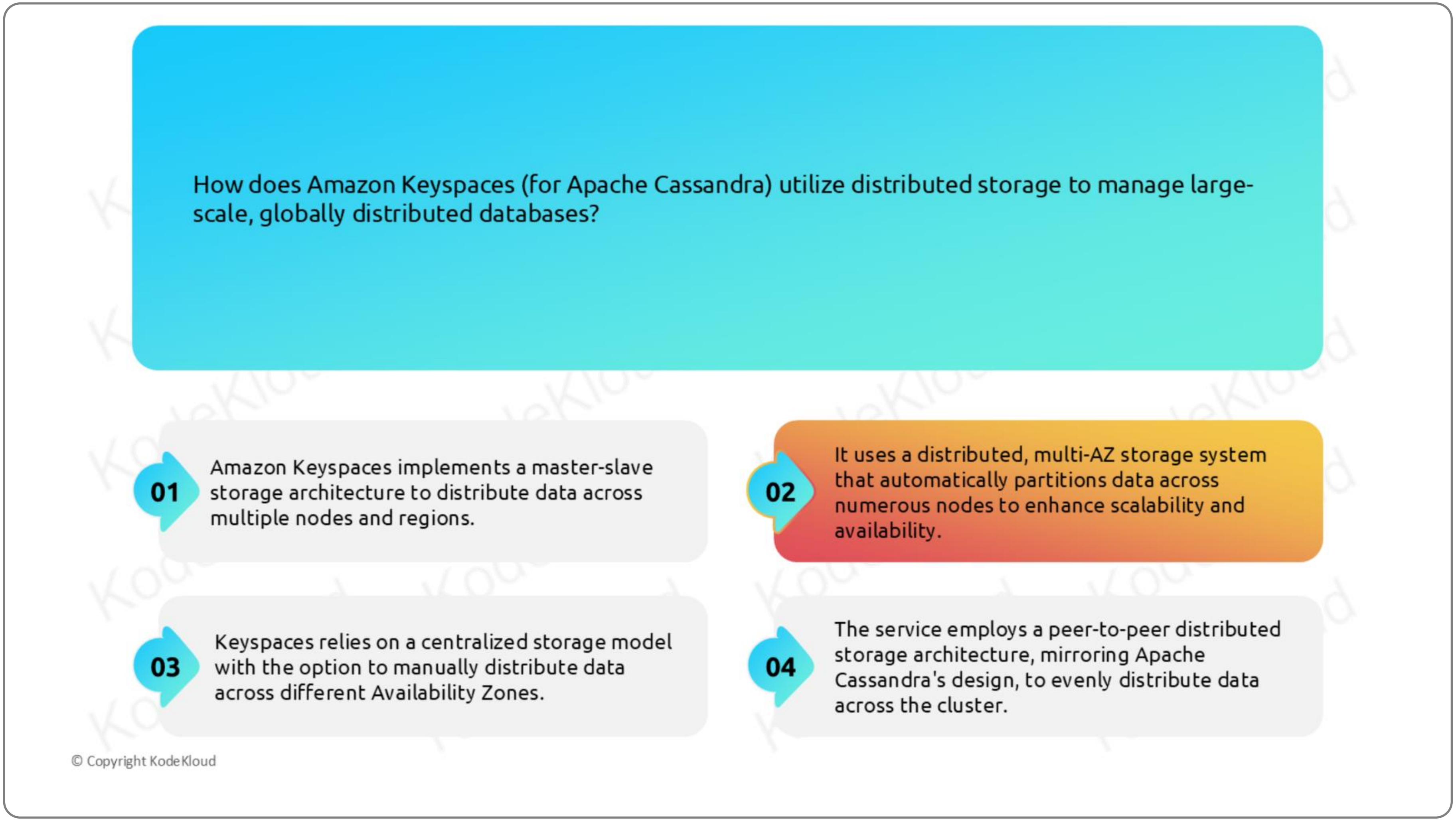
Target state: Amazon ECS and Amazon Keyspaces

ECS task are isolated and can be added to scale services. Services can be scaled based on demand using the Application Auto Scaling Service.

Private communication between your virtual private cloud (VPC) running in Amazon VPC and Amazon Keyspaces.

Tables scale dynamically based on throughput without impacting query performance

Amazon Keyspaces tables are isolated and scaled independently. Tables scale based on throughput, scaling up to improve availability and scaling down to save cost.



How does Amazon Keyspaces (for Apache Cassandra) utilize distributed storage to manage large-scale, globally distributed databases?

01 Amazon Keyspaces implements a master-slave storage architecture to distribute data across multiple nodes and regions.

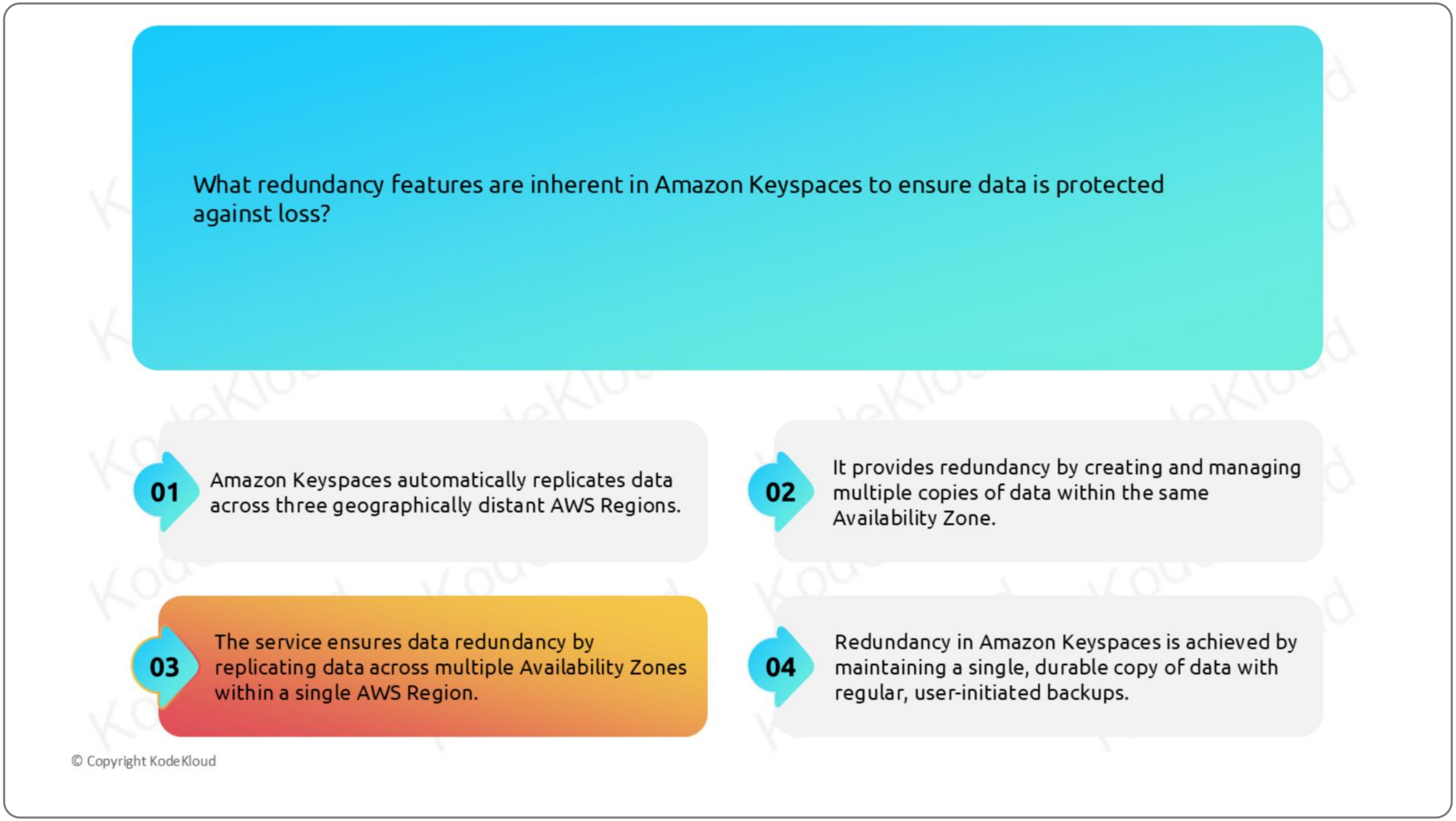
02 It uses a distributed, multi-AZ storage system that automatically partitions data across numerous nodes to enhance scalability and availability.

03 Keyspaces relies on a centralized storage model with the option to manually distribute data across different Availability Zones.

04 The service employs a peer-to-peer distributed storage architecture, mirroring Apache Cassandra's design, to evenly distribute data across the cluster.

Correct Answer: B
Explanation: Amazon Keyspaces (for Apache Cassandra) uses a distributed, multi-AZ storage system that automatically partitions data across many nodes within an AWS Region. This design allows for seamless scalability and high availability, as data is distributed and replicated across multiple physical locations within the region.
Incorrect Answers:
A) Amazon Keyspaces does not use a master-slave architecture; it is designed to be a serverless, scalable, and highly

available service that manages the underlying infrastructure for you.
C) Keyspaces does not use a centralized storage model; it is built on a distributed storage system.
D) While Amazon Keyspaces is compatible with Apache Cassandra, it abstracts the underlying storage architecture to provide a managed service experience.
References:
Amazon Keyspaces (for Apache Cassandra)



What redundancy features are inherent in Amazon Keyspaces to ensure data is protected against loss?

01 Amazon Keyspaces automatically replicates data across three geographically distant AWS Regions.

02 It provides redundancy by creating and managing multiple copies of data within the same Availability Zone.

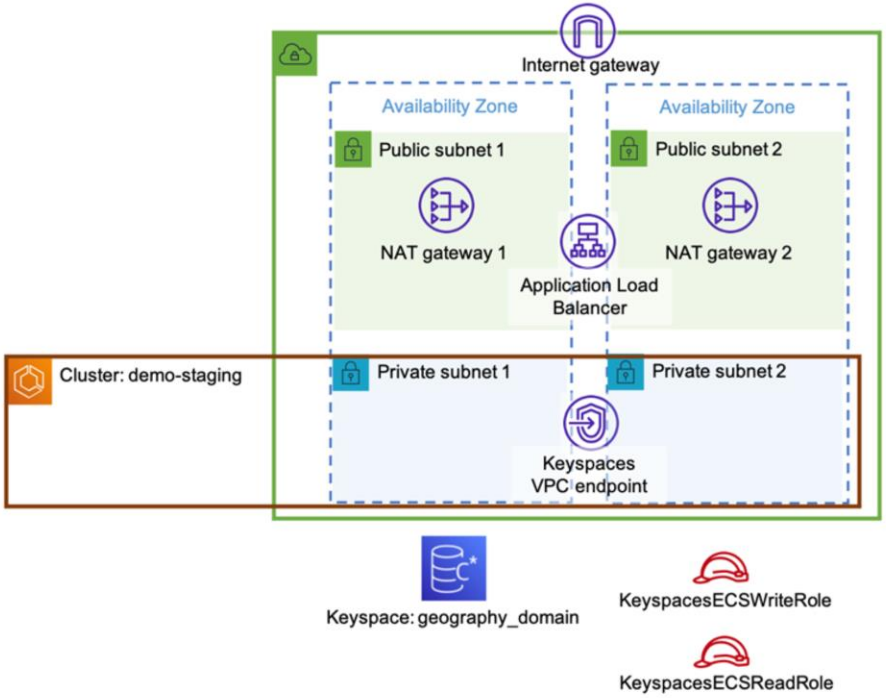
03 The service ensures data redundancy by replicating data across multiple Availability Zones within a single AWS Region.

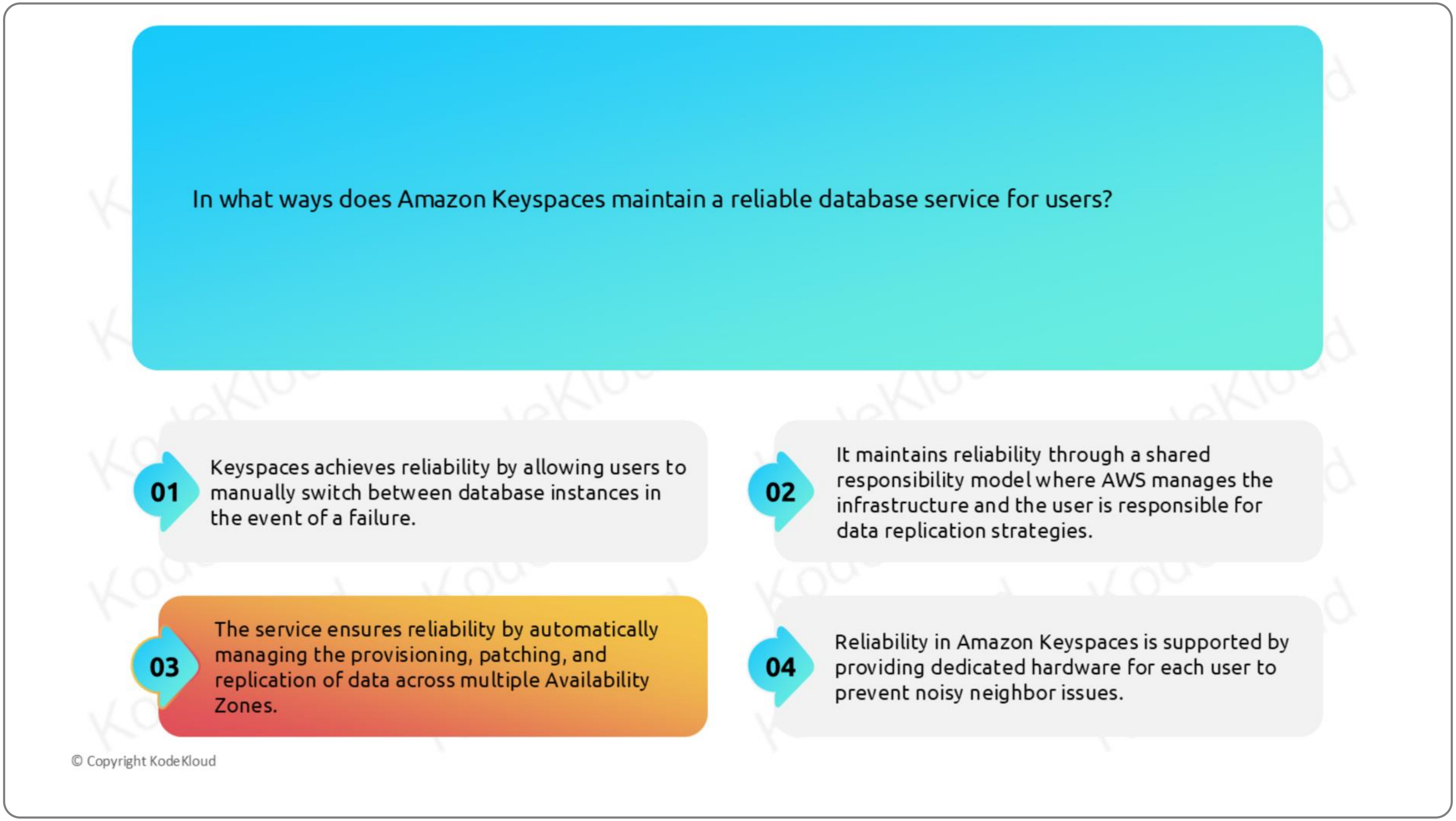
04 Redundancy in Amazon Keyspaces is achieved by maintaining a single, durable copy of data with regular, user-initiated backups.

Correct Answer: C
Explanation: Amazon Keyspaces (for Apache Cassandra) ensures data redundancy by replicating data automatically across multiple Availability Zones within the same AWS Region. This replication strategy is designed to provide high durability and availability for your data, protecting it against the loss of an Availability Zone.
Incorrect Answers:
A) Amazon Keyspaces does not automatically replicate data across multiple regions; its redundancy is focused within a

single region across Availability Zones.
B) Redundancy in Keyspaces is not limited to a single Availability Zone; it spans multiple AZs.
D) While backups are supported in Keyspaces, redundancy is not solely dependent on backups; it is provided by real-time replication across AZs.
References:
Amazon Keyspaces (for Apache Cassandra) Features

Keyspaces (for Apache Cassandra)





In what ways does Amazon Keyspaces maintain a reliable database service for users?

01 Keyspaces achieves reliability by allowing users to manually switch between database instances in the event of a failure.

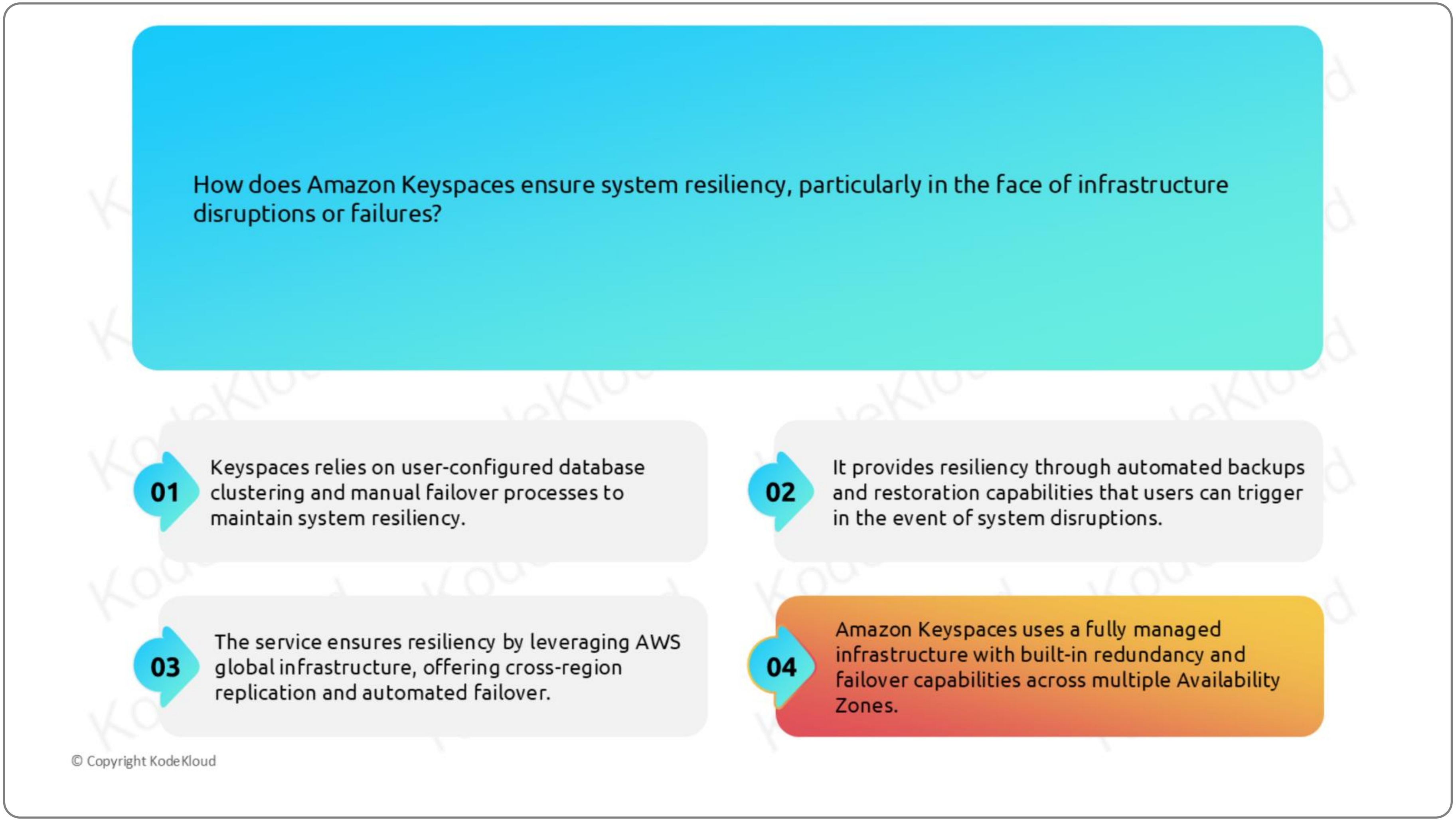
02 It maintains reliability through a shared responsibility model where AWS manages the infrastructure and the user is responsible for data replication strategies.

03 The service ensures reliability by automatically managing the provisioning, patching, and replication of data across multiple Availability Zones.

04 Reliability in Amazon Keyspaces is supported by providing dedicated hardware for each user to prevent noisy neighbor issues.

Correct Answer: C
Explanation: Amazon Keyspaces (for Apache Cassandra) maintains a reliable database service by automatically handling the provisioning, patching, and replication of data. It abstracts the management of the underlying infrastructure, including data replication across multiple Availability Zones, which is crucial for achieving high availability and fault tolerance.
Incorrect Answers:
A) Users do not need to manually switch between instances; Amazon Keyspaces manages failover and replication

automatically.
B) While AWS does follow a shared responsibility model, the management of infrastructure and data replication within Keyspaces is fully managed by AWS.
D) Amazon Keyspaces is a serverless offering, so the concept of dedicated hardware for each user does not apply.
References:
[Amazon Keyspaces (for Apache Cassandra) – How It Works]()



How does Amazon Keyspaces ensure system resiliency, particularly in the face of infrastructure disruptions or failures?

01 Keyspaces relies on user-configured database clustering and manual failover processes to maintain system resiliency.

02 It provides resiliency through automated backups and restoration capabilities that users can trigger in the event of system disruptions.

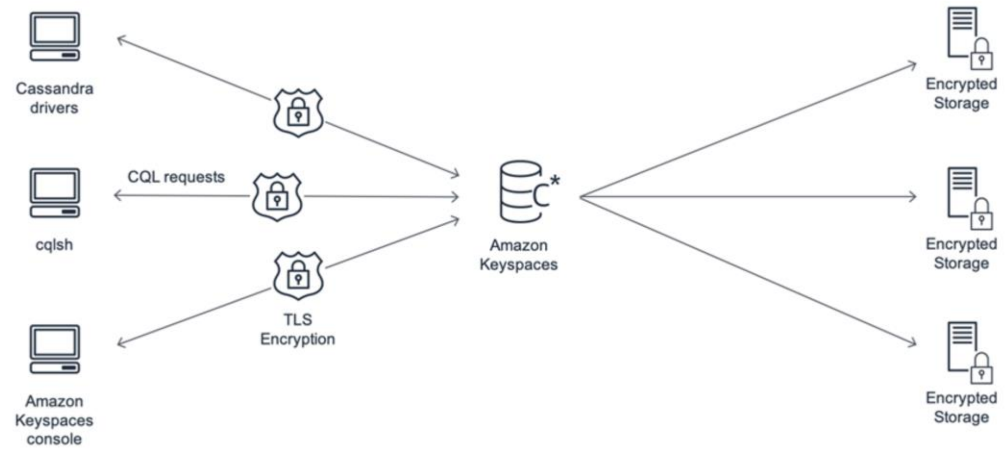
03 The service ensures resiliency by leveraging AWS global infrastructure, offering cross-region replication and automated failover.

04 Amazon Keyspaces uses a fully managed infrastructure with built-in redundancy and failover capabilities across multiple Availability Zones.

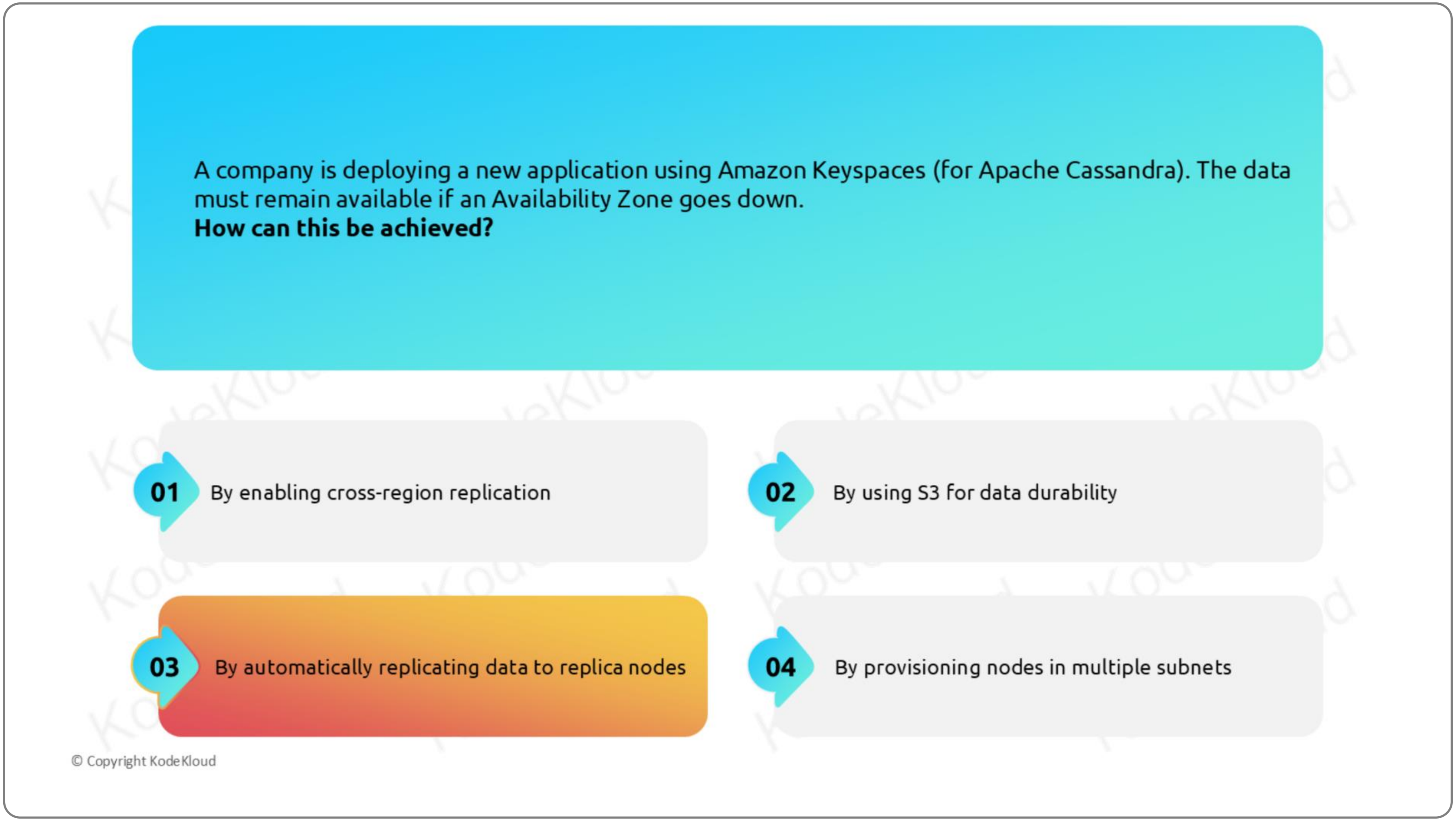
Correct Answer: D
Explanation: Amazon Keyspaces is designed to ensure system resiliency by leveraging AWS's highly available infrastructure. It provides built-in redundancy and automated failover capabilities across multiple Availability Zones within an AWS Region. This means that in the event of a failure or disruption in one Availability Zone, the service can continue to operate seamlessly, as it is designed to withstand such incidents without any data loss or significant performance impact.
Incorrect Answers:

A) Amazon Keyspaces is a fully managed service, so users do not need to configure database clustering or manual failover processes; these are handled automatically by the service.
B) While automated backups and restoration capabilities do contribute to resiliency, the primary mechanism for resiliency in Amazon Keyspaces is its multi-AZ replication and failover, not just backups.
C) As of my last update, Amazon Keyspaces does not offer cross-region replication as a built-in feature. Resiliency is provided primarily through multi-AZ deployment within a single region.
Explanation Continued:
Resiliency in Amazon Keyspaces is further supported by its serverless nature, which automatically scales capacity up or down to match your application's needs without requiring manual intervention. This scalability ensures that the database can handle varying loads and maintain performance even during unexpected surges in demand.
Moreover, Amazon Keyspaces offers continuous backups to Amazon S3, enabling point-in-time recovery to protect against accidental data loss. The combination of these features allows Amazon Keyspaces to provide a resilient, scalable, and highly available database service suitable for a wide range of applications, from development and test environments to production workloads.
References:
Amazon Keyspaces (for Apache Cassandra) Features
Amazon Keyspaces (for Apache Cassandra) FAQs

Keyspaces (for Apache Cassandra)



Redundancy is through redundant synchronized storage.



A company is deploying a new application using Amazon Keyspaces (for Apache Cassandra). The data must remain available if an Availability Zone goes down.

How can this be achieved?

01 By enabling cross-region replication

02 By using S3 for data durability

03 By automatically replicating data to replica nodes

04 By provisioning nodes in multiple subnets

Correct Answer: C

Explanation: Amazon Keyspaces provides high availability across AZs by automatically replicating data to multiple replica nodes placed in different AZs. If one AZ becomes unavailable, reads and writes can be served from replicas in other AZs.

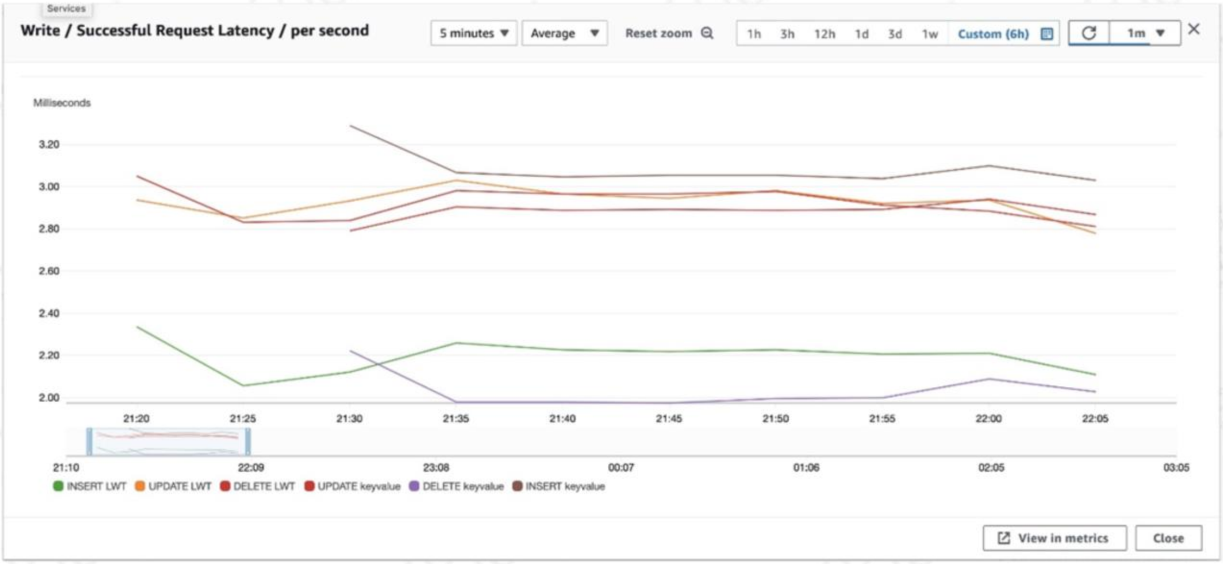
Incorrect Answers:

A) Cross-region replication is for disaster recovery, not high availability within a region.

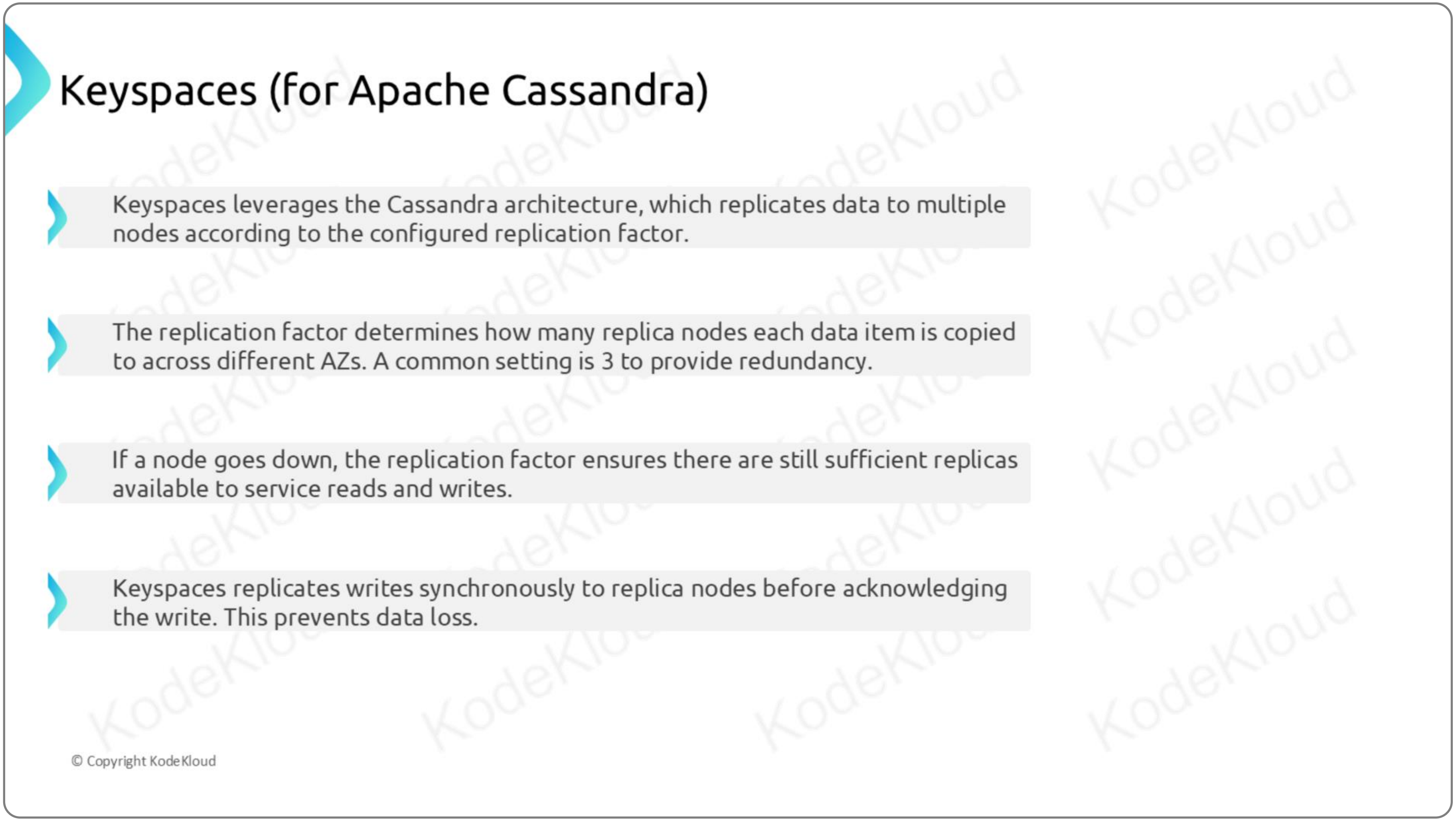
B) S3 stores backups but does not provide real-time data replication.

D) Deploying nodes in different subnets does not automatically replicate data.
References:
https://docs.aws.amazon.com/keyspaces/latest/devguide/high-availability.html

Designing for Reliability – Keyspaces (for Apache Cassandra)



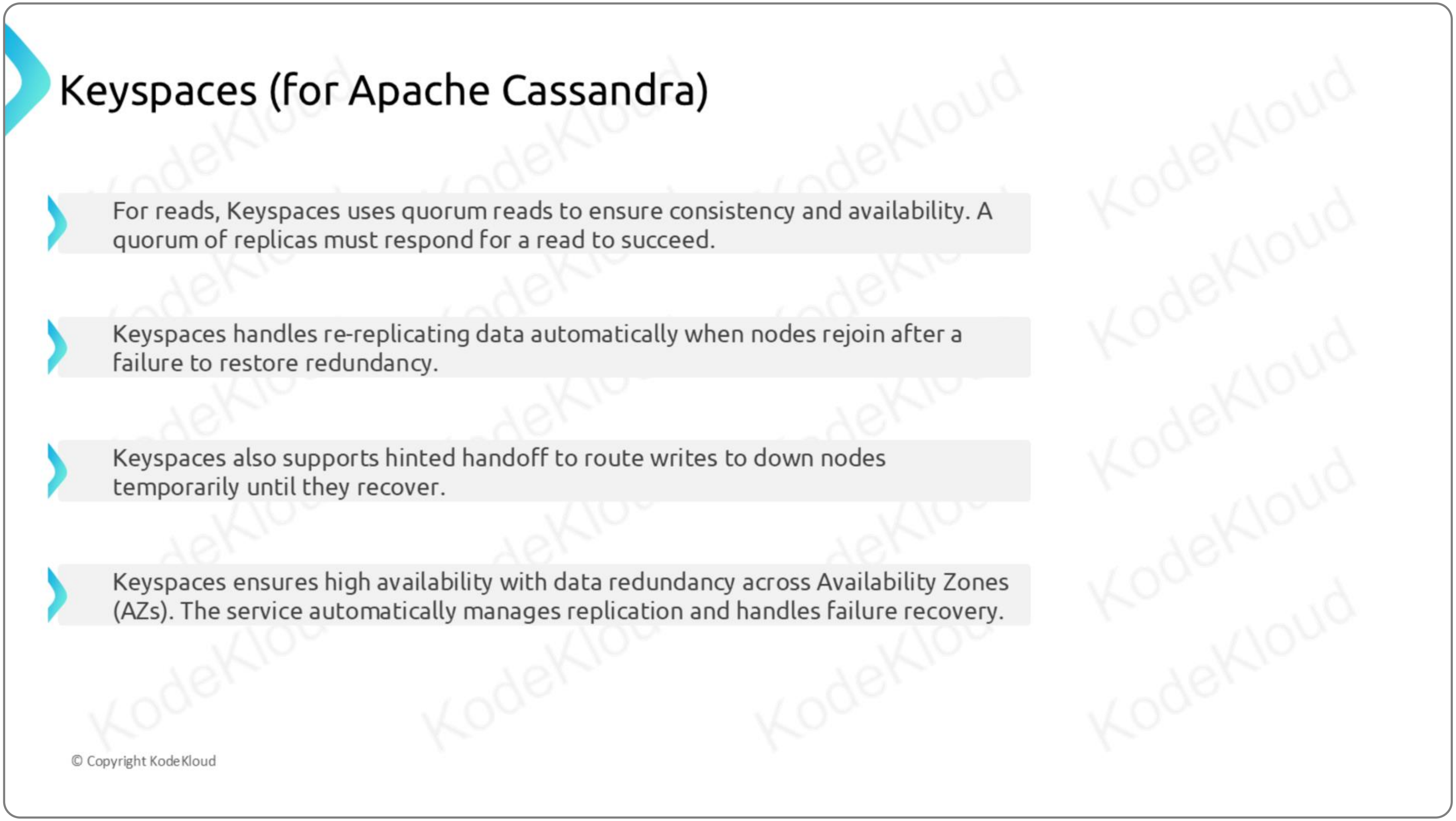
Keyspaces is highly managed, so per standard, CloudWatch and CloudTrail will contain metrics and logs relevant to Reliability.



Keyspaces (for Apache Cassandra)

- Keyspaces leverages the Cassandra architecture, which replicates data to multiple nodes according to the configured replication factor.
- The replication factor determines how many replica nodes each data item is copied to across different AZs. A common setting is 3 to provide redundancy.
- If a node goes down, the replication factor ensures there are still sufficient replicas available to service reads and writes.
- Keyspaces replicates writes synchronously to replica nodes before acknowledging the write. This prevents data loss.

Keyspaces leverages the Cassandra architecture, which replicates data to multiple nodes according to the configured replication factor.
The replication factor determines how many replica nodes each data item is copied to across different AZs. A common setting is 3 to provide redundancy.
If a node goes down, the replication factor ensures there are still sufficient replicas available to service reads and writes.
Keyspaces replicates writes synchronously to replica nodes before acknowledging the write. This prevents data loss.



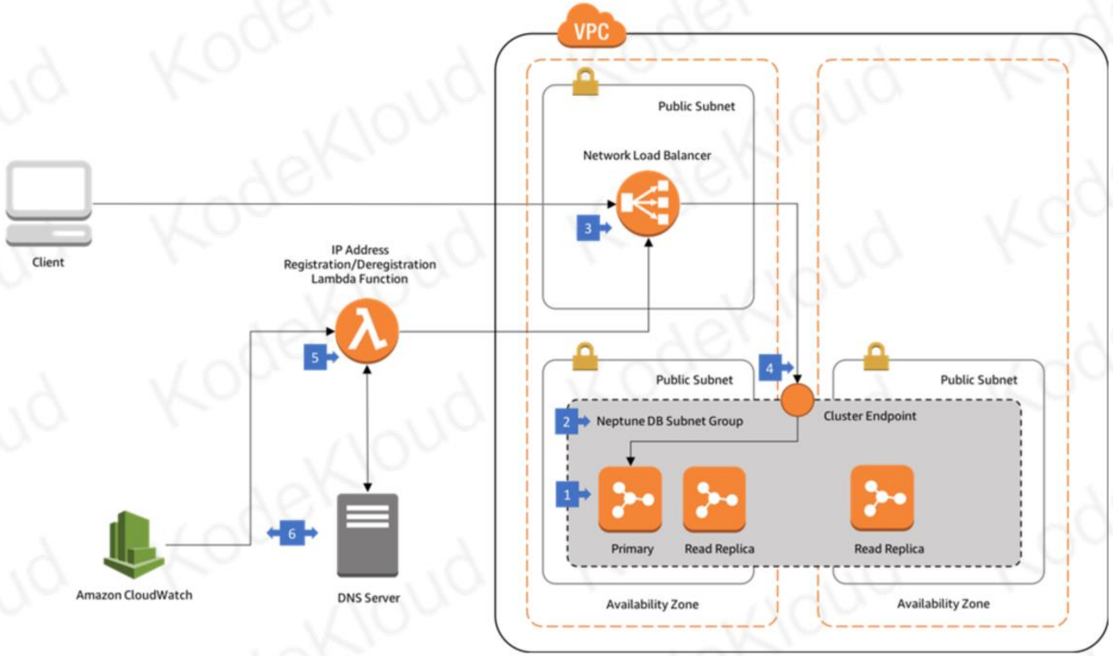
Keyspaces (for Apache Cassandra)

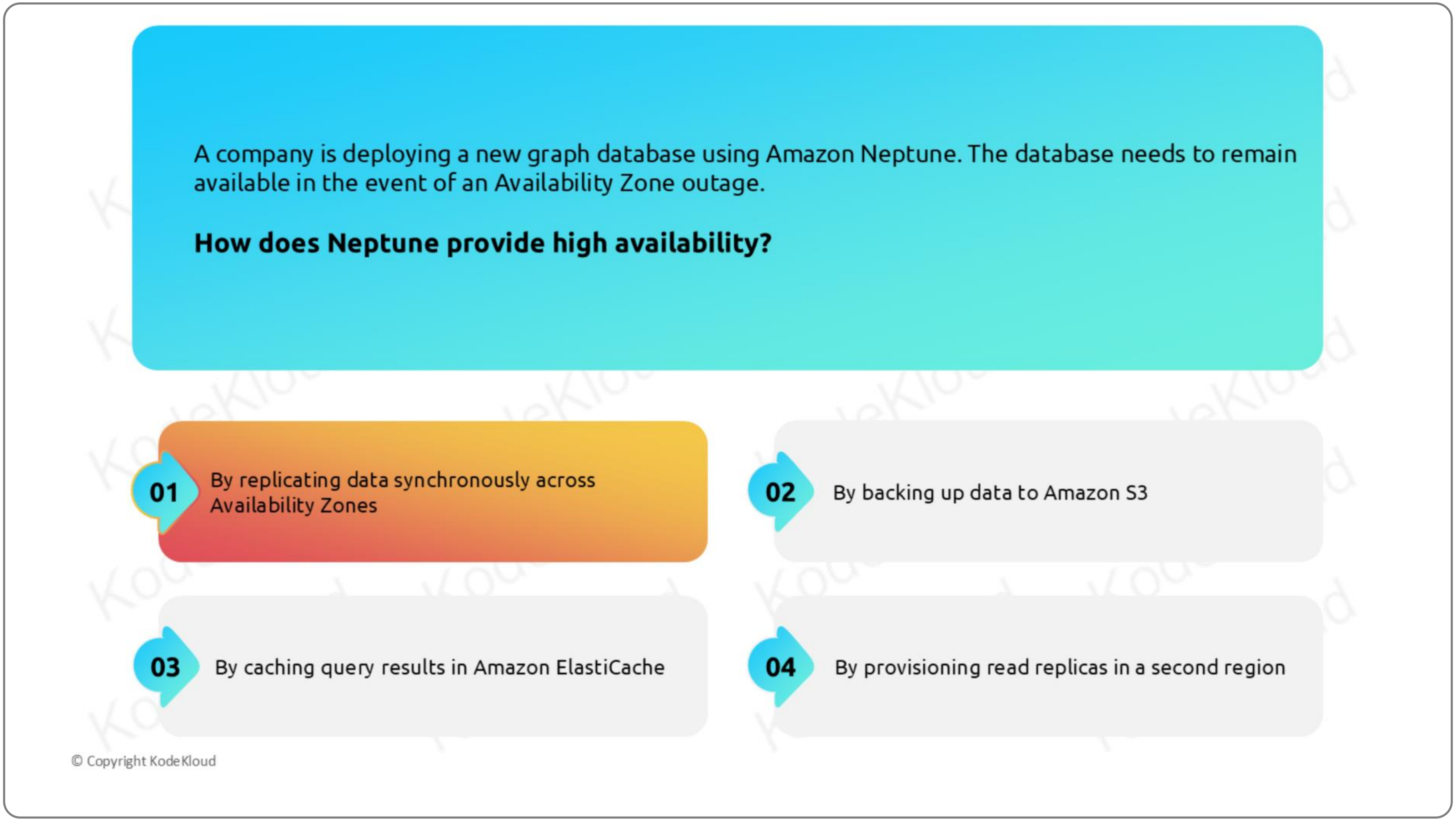
- For reads, Keyspaces uses quorum reads to ensure consistency and availability. A quorum of replicas must respond for a read to succeed.
- Keyspaces handles re-replicating data automatically when nodes rejoin after a failure to restore redundancy.
- Keyspaces also supports hinted handoff to route writes to down nodes temporarily until they recover.
- Keyspaces ensures high availability with data redundancy across Availability Zones (AZs). The service automatically manages replication and handles failure recovery.

For reads, Keyspaces uses quorum reads to ensure consistency and availability. A quorum of replicas must respond for a read to succeed.
Keyspaces handles re-replicating data automatically when nodes rejoin after a failure to restore redundancy.
Keyspaces also supports hinted handoff to route writes to down nodes temporarily until they recover.
Together, these capabilities allow Keyspaces to deliver high availability by maintaining redundant copies of data across AZs. The service handles replication and failure recovery automatically.

Designing for Reliability – Neptune

Neptune (neo4j)





A company is deploying a new graph database using Amazon Neptune. The database needs to remain available in the event of an Availability Zone outage.

How does Neptune provide high availability?

01 By replicating data synchronously across Availability Zones

02 By backing up data to Amazon S3

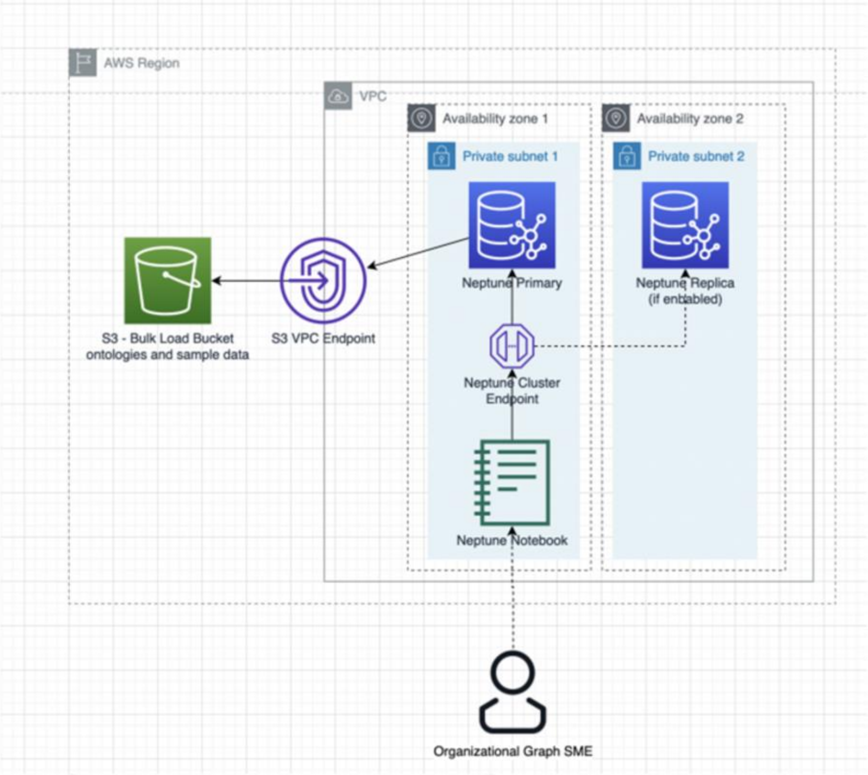
03 By caching query results in Amazon ElastiCache

04 By provisioning read replicas in a second region

Correct Answer: A
Explanation: Amazon Neptune provides high availability by synchronously replicating data across three Availability Zones. If one AZ becomes unavailable, Neptune can redirect reads and writes to the remaining AZs.
Incorrect Answers:
B) Backups to S3 provide disaster recovery, not high availability.
C) Caching improves read performance but does not replicate data.

D) Read replicas in a second region are for disaster recovery.
References:
https://docs.aws.amazon.com/neptune/latest/userguide/feature-overview-availability.html

Neptune (neo4j)



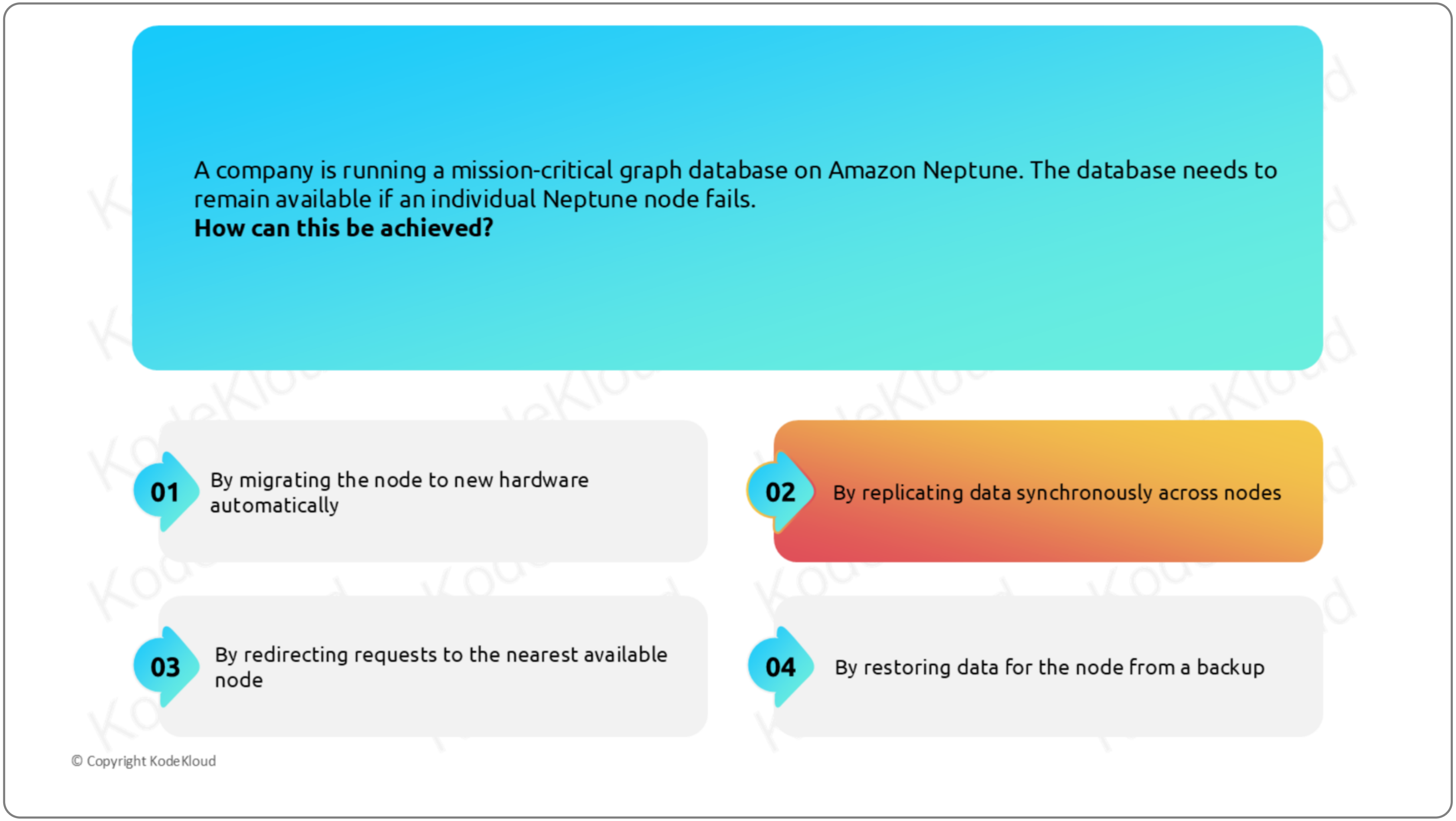
Neptune uses synchronous replication to copy data across three Availability Zones. Writes are committed to all AZs before being acknowledged.
Reads use quorum consistency to require responses from a majority of replicas before returning results. This prevents stale reads if an AZ goes down.
Neptune storage is spread evenly across the AZs at the shard level for balanced capacity and performance.
If an AZ becomes unavailable, Neptune redirects reads/writes to the remaining available AZs automatically.

Neptune's replication protocol keeps the replicas in sync and handles re-replication of data when a failed node recovers.

Neptune can also leverage read replicas in other AZs to scale read capacity. Read replicas are kept synchronized via asynchronous replication.

In addition to AZ redundancy, Neptune replicates data across three physical servers per AZ for redundancy against individual server failures.

Neptune continuously monitors the cluster and initiates repairs and re-replication automatically in case of node failures.



A company is running a mission-critical graph database on Amazon Neptune. The database needs to remain available if an individual Neptune node fails.
How can this be achieved?

01 By migrating the node to new hardware automatically

02 By replicating data synchronously across nodes

03 By redirecting requests to the nearest available node

04 By restoring data for the node from a backup

Correct Answer: B
Explanation: Neptune synchronously replicates data across nodes in multiple Availability Zones. If a single node fails, Neptune will continue serving requests from the remaining replicas and automatically re-replicate data when the failed node recovers.
Incorrect Answers:
A) Neptune does not migrate individual failed nodes to new hardware.

C) Requests are not redirected on a node failure since data is replicated.
D) Node failures do not require restoring data from backup.
References:
https://docs.aws.amazon.com/neptune/latest/userguide/feature-overview-availability.html

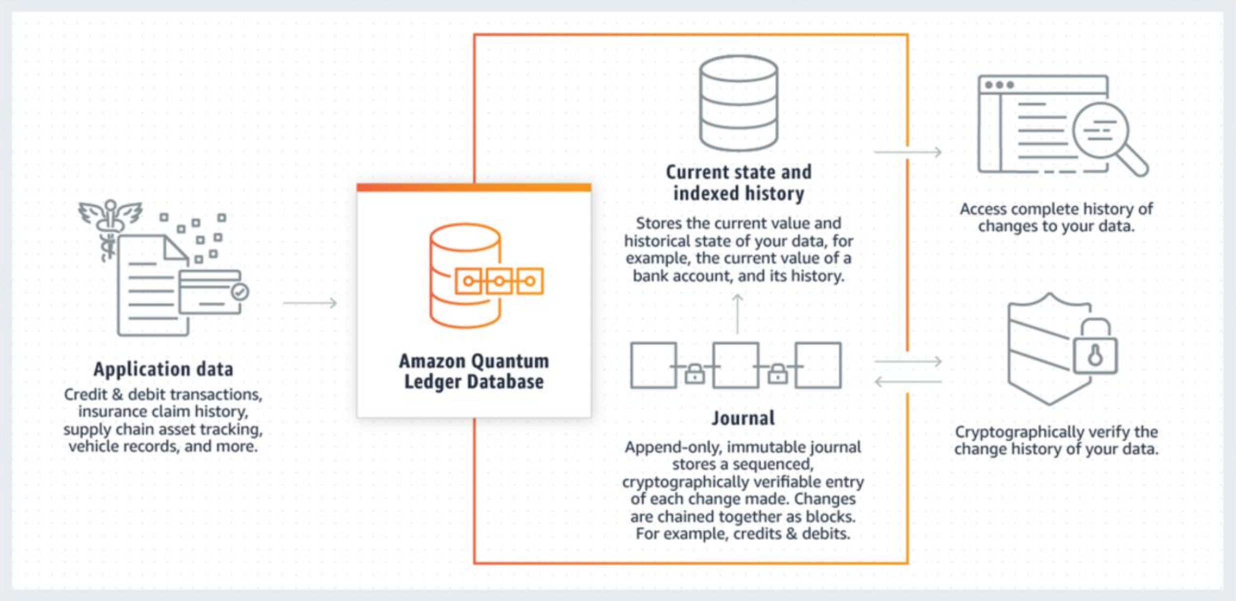
Designing for Reliability

Turning up Reliability on Database Services

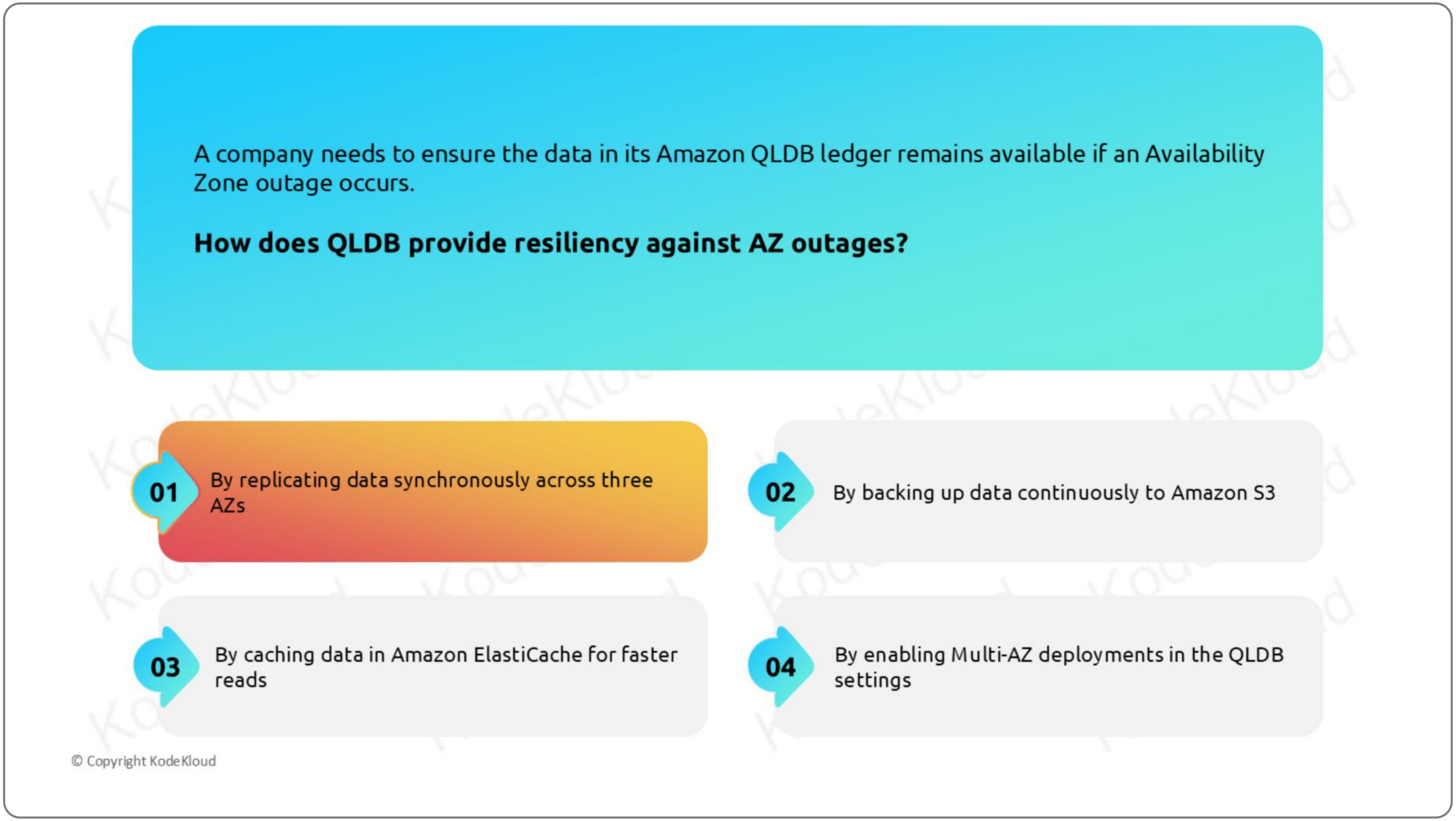
Immutability and Time

Designing for Reliability – QLDB

QLDB


Application data
Credit & debit transactions, insurance claim history, supply chain asset tracking, vehicle records, and more.
Amazon Quantum Ledger Database
Current state and indexed history
Stores the current value and historical state of your data, for example, the current value of a bank account, and its history.
Journal
Append-only, immutable journal stores a sequenced, cryptographically verifiable entry of each change made. Changes are chained together as blocks. For example, credits & debits.
Access complete history of changes to your data.
Cryptographically verify the change history of your data.

The quantum ledger database works like DocumentDB.



A company needs to ensure the data in its Amazon QLDB ledger remains available if an Availability Zone outage occurs.

How does QLDB provide resiliency against AZ outages?

01 By replicating data synchronously across three AZs

02 By backing up data continuously to Amazon S3

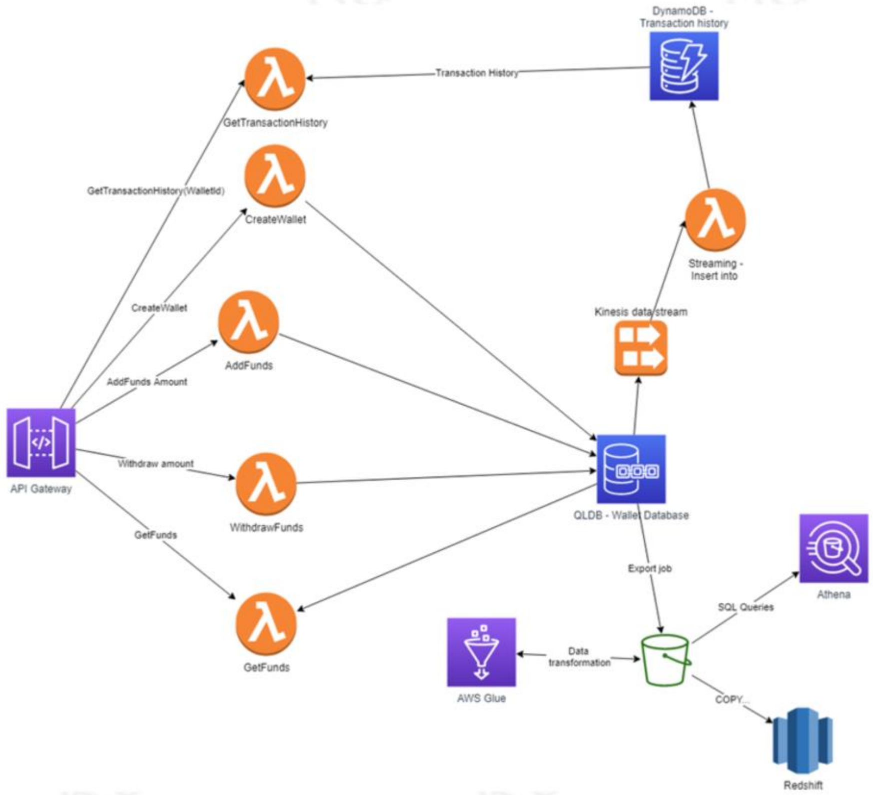
03 By caching data in Amazon ElastiCache for faster reads

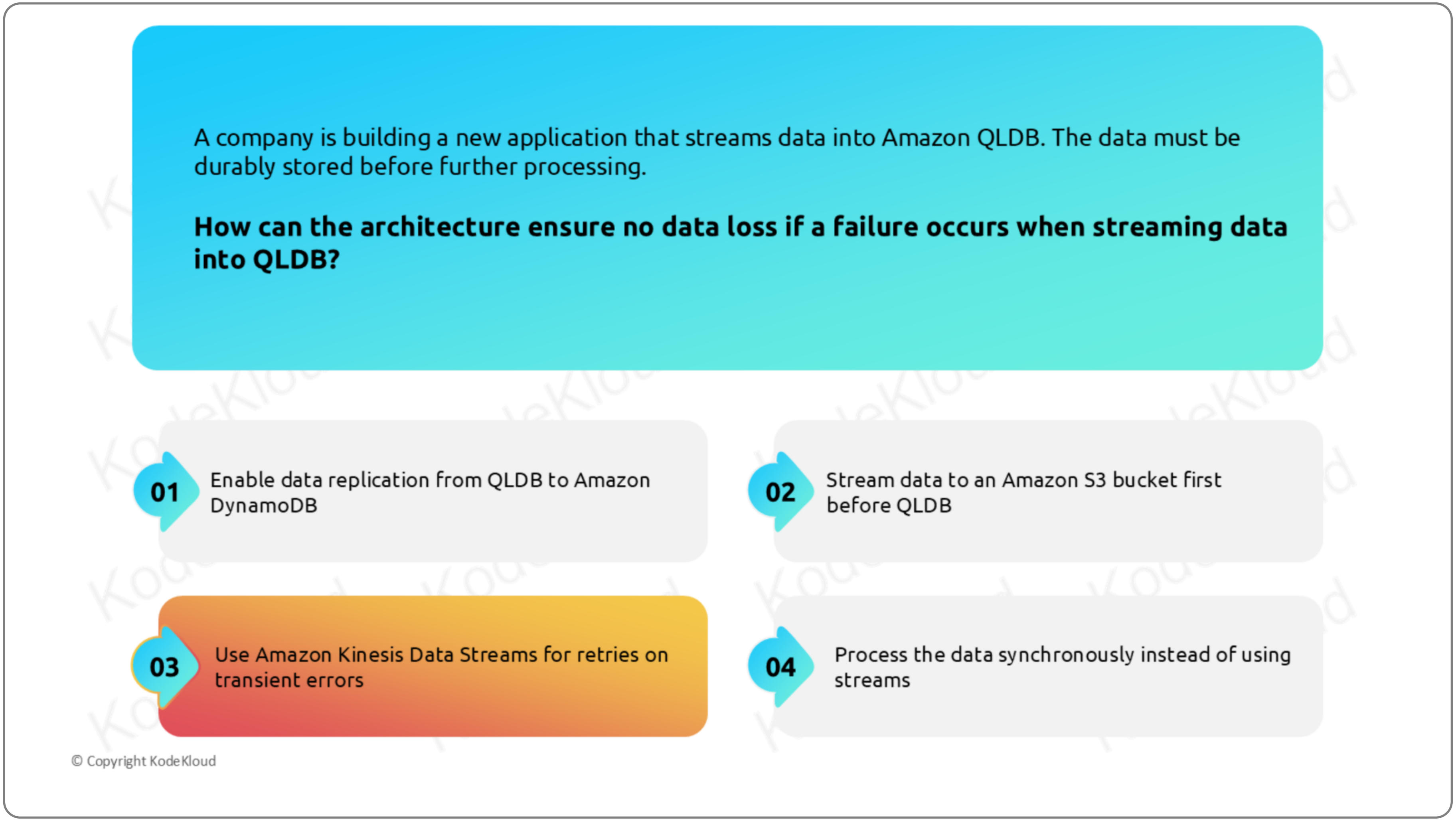
04 By enabling Multi-AZ deployments in the QLDB settings

Correct Answer: A
Explanation: Amazon QLDB synchronously replicates data across three Availability Zones to remain available during an AZ outage. The ledger will operate uninterrupted if one AZ becomes unavailable.
Incorrect Answers:
B) Backups help recover data but do not provide high availability.
C) Caching improves read performance but does not replicate data.

D) Multi-AZ redundancy is automatically built-in, not a configurable setting.
References:
https://docs.aws.amazon.com/qldb/latest/developerguide/disaster-recovery-resiliency.html

QLDB


DynamoDB - Transaction history
Transaction History
GetTransactionHistory
GetTransactionHistory(WalletId)
CreateWallet
CreateWallet
Streaming - Insert into
Kinesis data stream
AddFunds
AddFunds Amount
API Gateway
Withdraw amount
WithdrawFunds
QLDB - Wallet Database
GetFunds
Export job
Athena
SQL Queries
GetFunds
AWS Glue
Data transformation
COPY...
Redshift



A company is building a new application that streams data into Amazon QLDB. The data must be durably stored before further processing.

How can the architecture ensure no data loss if a failure occurs when streaming data into QLDB?

01 Enable data replication from QLDB to Amazon DynamoDB

02 Stream data to an Amazon S3 bucket first before QLDB

03 Use Amazon Kinesis Data Streams for retries on transient errors

04 Process the data synchronously instead of using streams

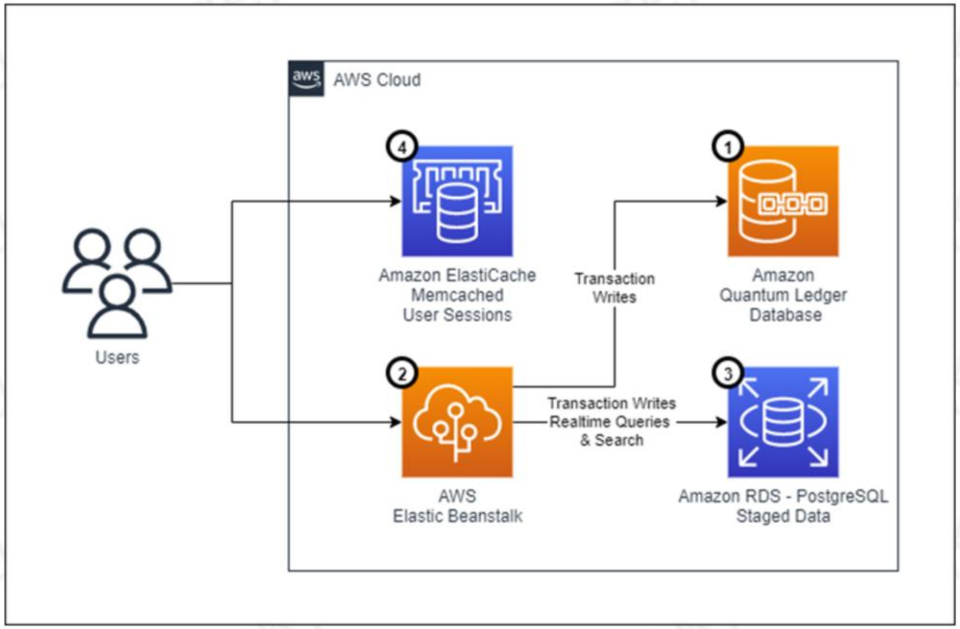
Correct Answer: C. Amazon QLDB integrates with IAM, allowing you to define granular permissions for specific ledger operations using IAM policies.
Incorrect: A. QLDB supports IAM roles and also allows granular permissions for specific ledger operations. B. QLDB supports both IAM users and roles and also allows the use of IAM policies for access control. D. QLDB fully supports IAM integration for access control and permissions.
Explanation: Amazon QLDB integrates seamlessly with AWS Identity and Access Management (IAM). This integration

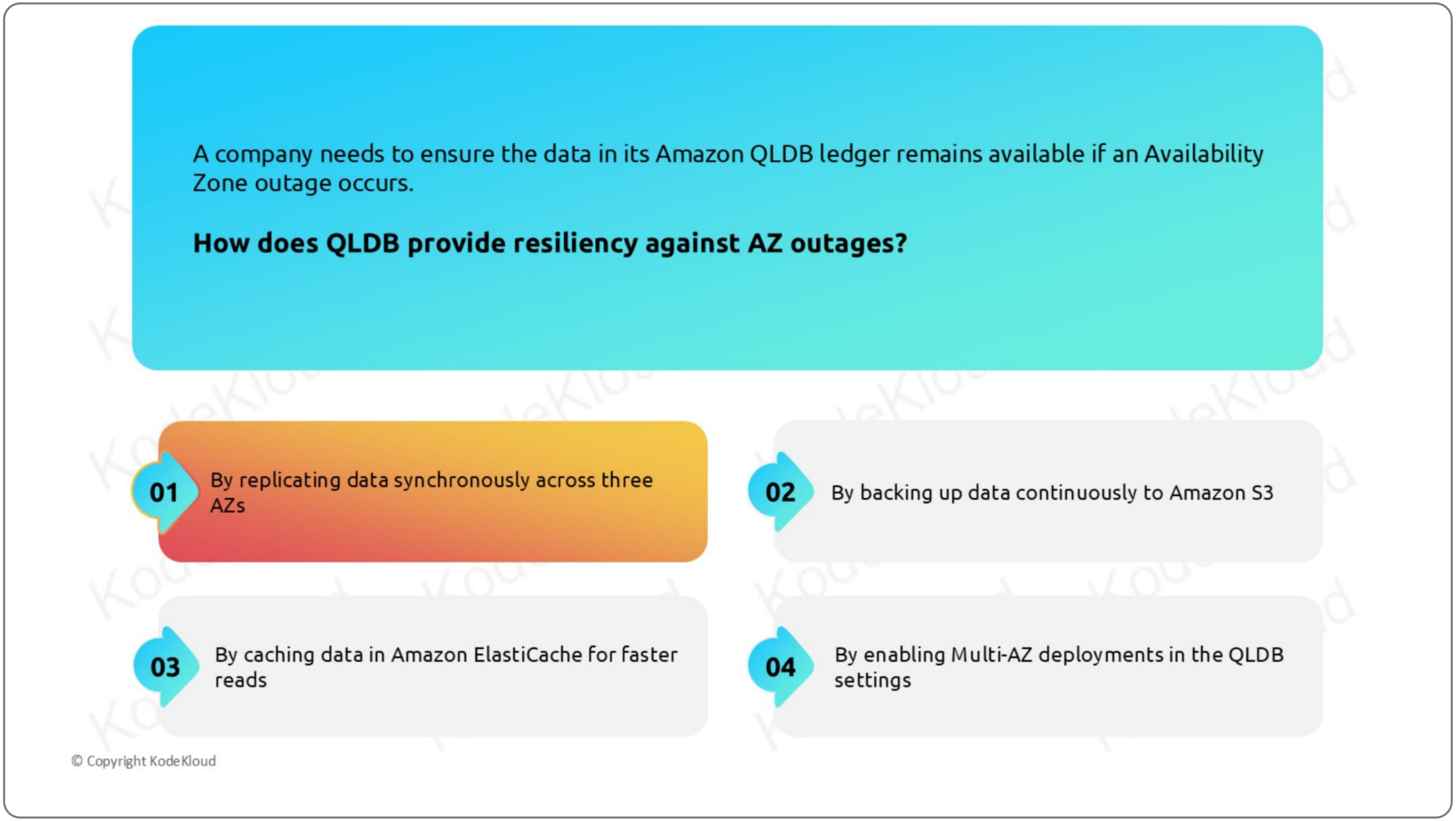
allows organizations to define granular permissions for specific operations on the ledger using IAM policies. By leveraging IAM, organizations can ensure that only authorized users and roles can perform specific actions on the QLDB ledger, enhancing the Reliability of their data.

References:

[Amazon QLDB: Identity and Access Management in Amazon QLDB]()

QLDB


AWS Cloud
Users
4
Amazon ElastiCache Memcached User Sessions
Transaction Writes
1
Amazon Quantum Ledger Database
2
AWS Elastic Beanstalk
Transaction Writes Realtime Queries & Search
3
Amazon RDS - PostgreSQL Staged Data



A company needs to ensure the data in its Amazon QLDB ledger remains available if an Availability Zone outage occurs.

How does QLDB provide resiliency against AZ outages?

01 By replicating data synchronously across three AZs

02 By backing up data continuously to Amazon S3

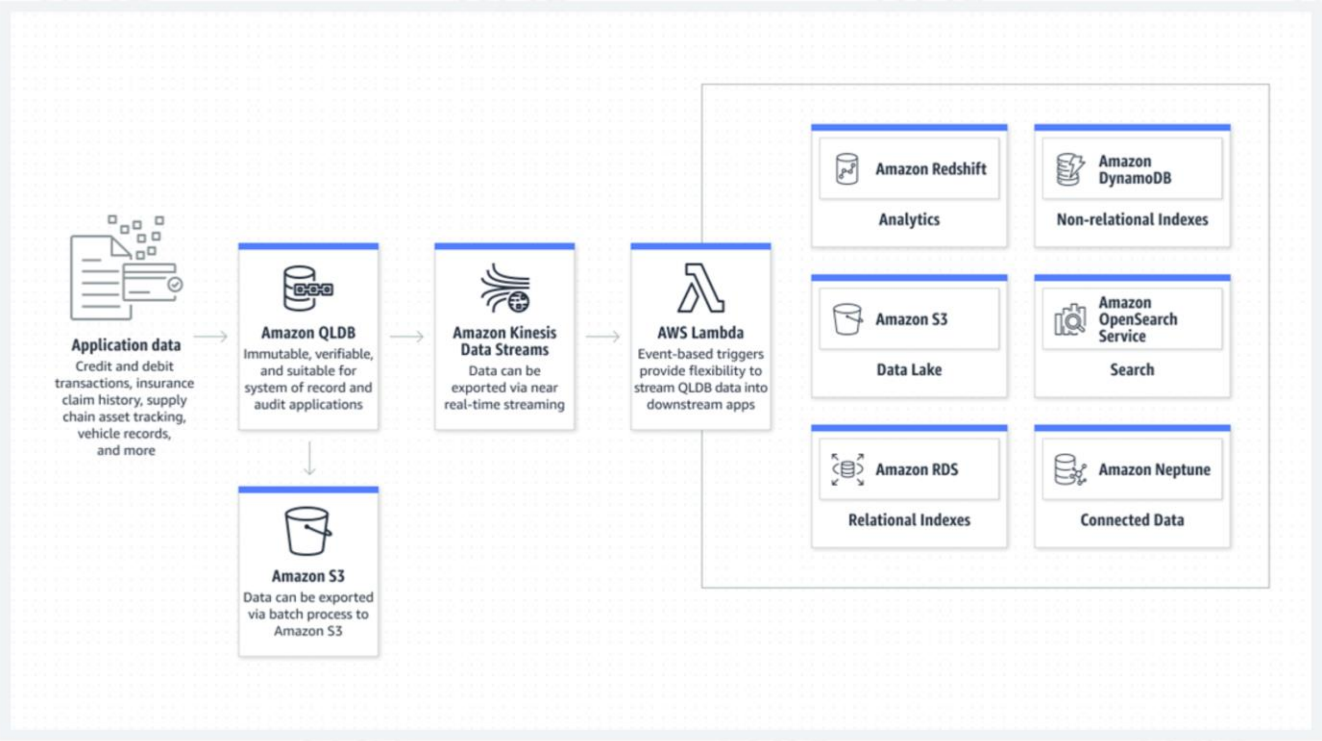
03 By caching data in Amazon ElastiCache for faster reads

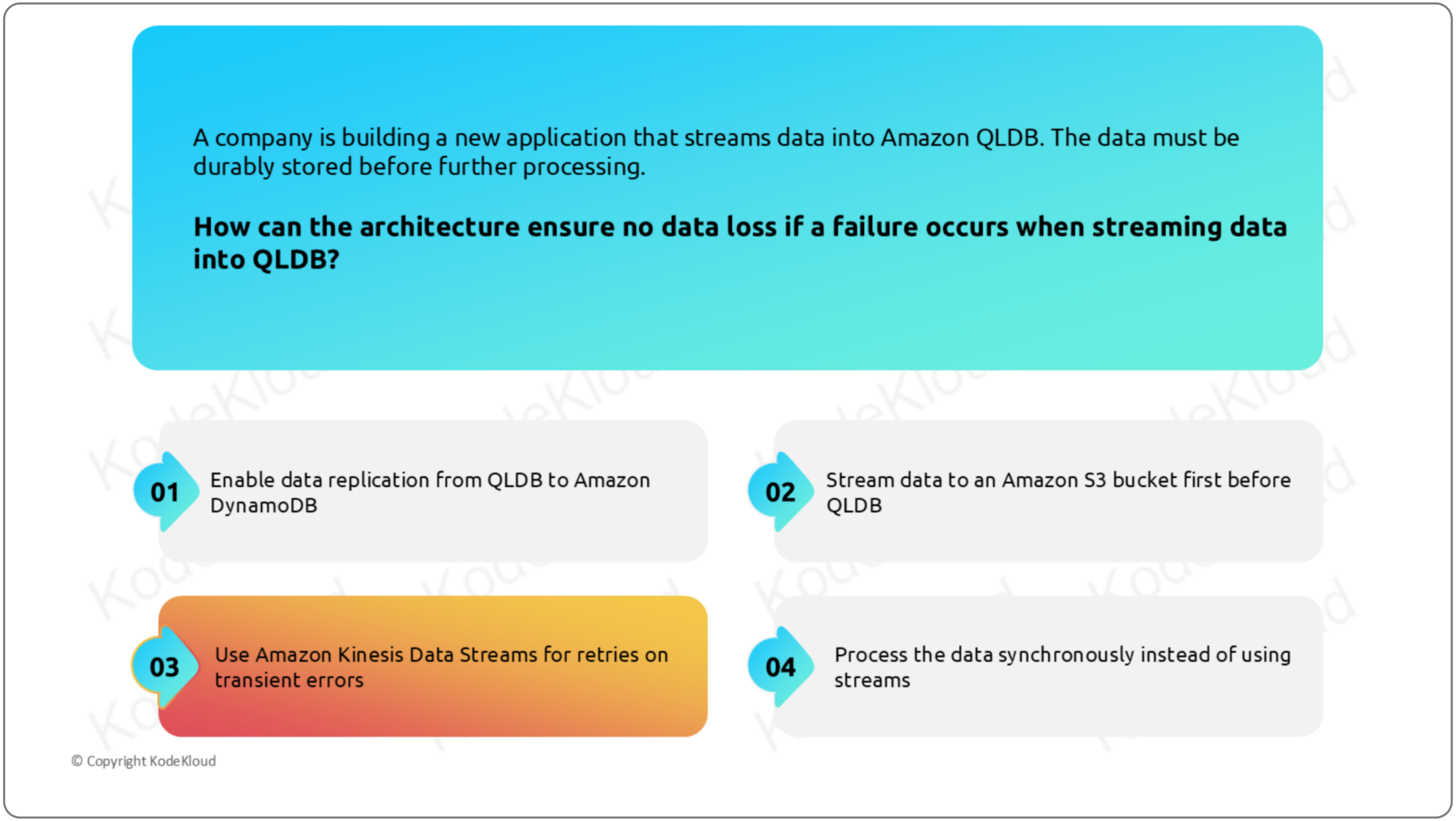
04 By enabling Multi-AZ deployments in the QLDB settings

Correct Answer: A
Explanation: Amazon QLDB synchronously replicates data across three Availability Zones to remain available during an AZ outage. The ledger will operate uninterrupted if one AZ becomes unavailable.
Incorrect Answers:
B) Backups help recover data but do not provide high availability.
C) Caching improves read performance but does not replicate data.

D) Multi-AZ redundancy is automatically built-in, not a configurable setting.
References:
https://docs.aws.amazon.com/qldb/latest/developerguide/disaster-recovery-resiliency.html

QLDB


Application data
Credit and debit transactions, insurance claim history, supply chain asset tracking, vehicle records, and more
Amazon QLDB
Immutable, verifiable, and suitable for system of record and audit applications
Amazon Kinesis Data Streams
Data can be exported via near real-time streaming
AWS Lambda
Event-based triggers provide flexibility to stream QLDB data into downstream apps
Amazon S3
Data can be exported via batch process to Amazon S3
Amazon Redshift
Analytics
Amazon DynamoDB
Non-relational Indexes
Amazon S3
Data Lake
Amazon OpenSearch Service
Search
Amazon RDS
Relational Indexes
Amazon Neptune
Connected Data



A company is building a new application that streams data into Amazon QLDB. The data must be durably stored before further processing.

How can the architecture ensure no data loss if a failure occurs when streaming data into QLDB?

01 Enable data replication from QLDB to Amazon DynamoDB

02 Stream data to an Amazon S3 bucket first before QLDB

03 Use Amazon Kinesis Data Streams for retries on transient errors

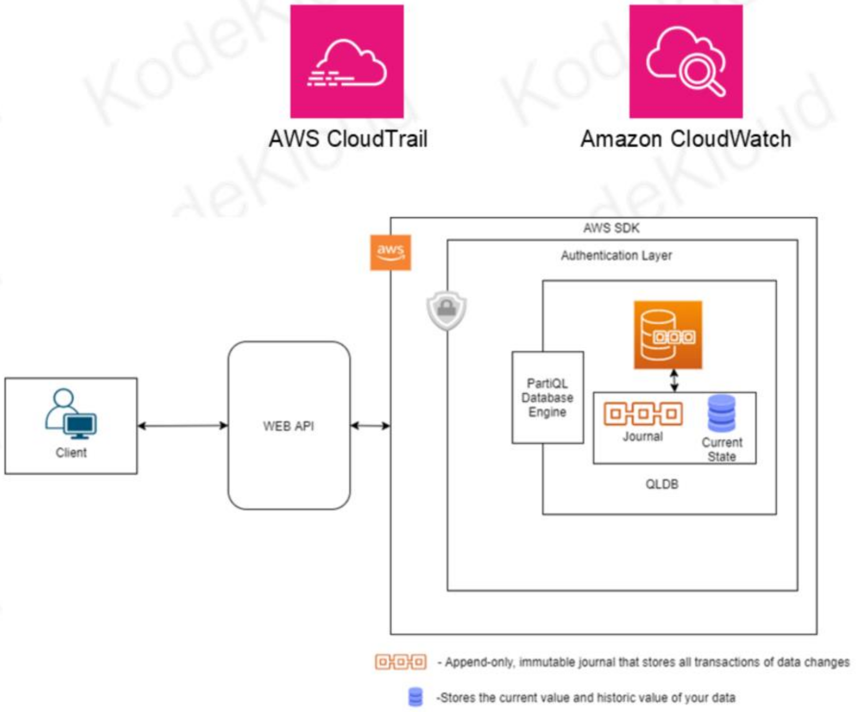
04 Process the data synchronously instead of using streams

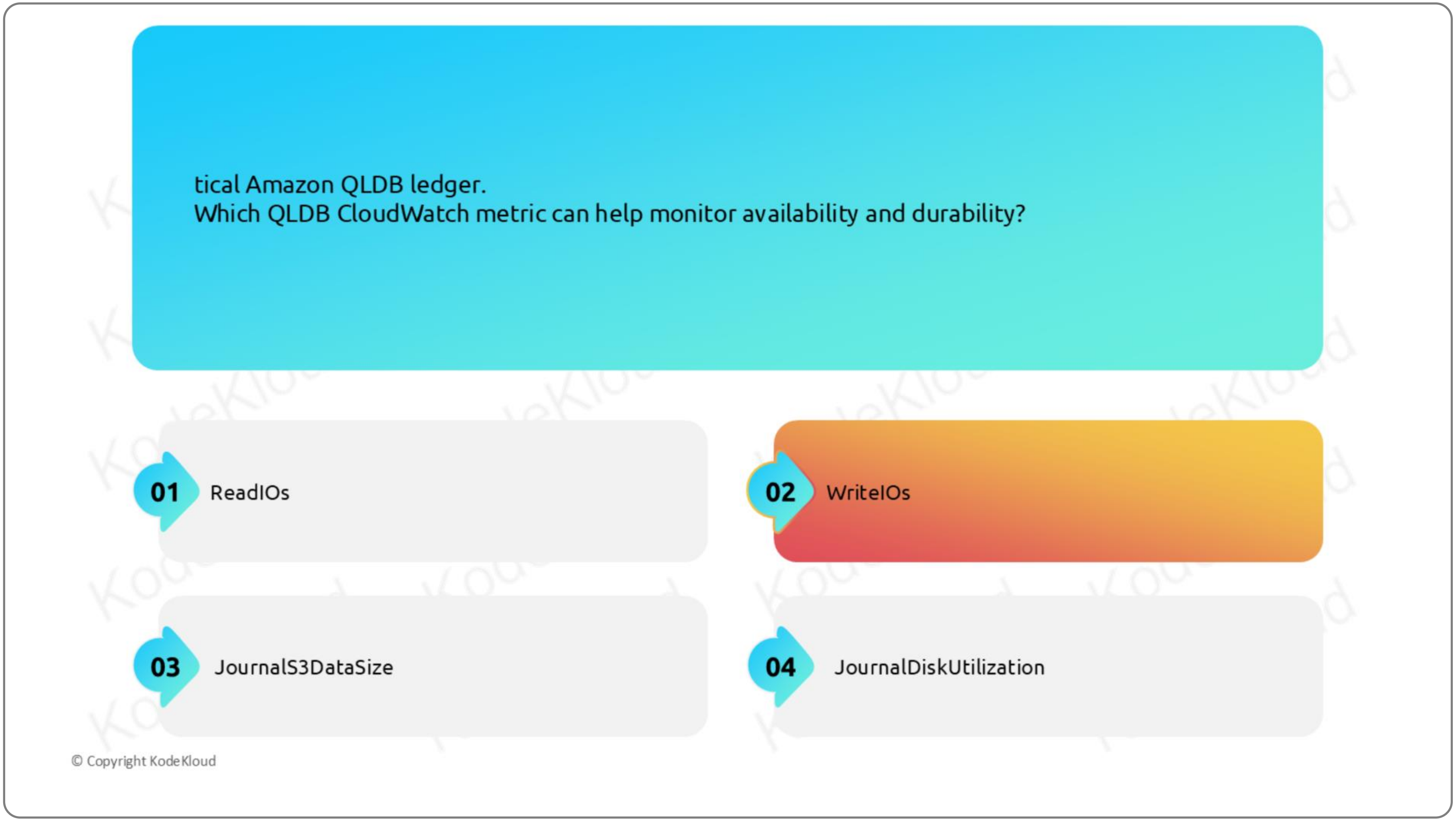
Correct Answer: C
Explanation: Streaming data into QLDB via Amazon Kinesis Data Streams allows configuring retries for transient errors, ensuring the data is eventually persisted durably in QLDB without loss.
Incorrect Answers:
A) QLDB replication to DynamoDB is for analytics, not streaming resilience.
B) S3 temporarily buffers data but does not provide retry capabilities.

D) Synchronous processing limits scalability and does not provide retries.
References:
https://docs.aws.amazon.com/qldb/latest/developerguide/kinesis-integrator.html

QLDB

QLDB uses the standard CloudWatch and CloudTrail pair to track metrics and API calls, respectively.


AWS CloudTrail
Amazon CloudWatch
AWS SDK
aws
Authentication Layer
PartiQL Database Engine
Journal
Current State
QLDB
WEB API
Client
- Append-only, immutable journal that stores all transactions of data changes
-Stores the current value and historic value of your data



tical Amazon QLDB ledger.
Which QLDB CloudWatch metric can help monitor availability and durability?

01 ReadIOs

02 WriteIOs

03 JournalS3DataSize

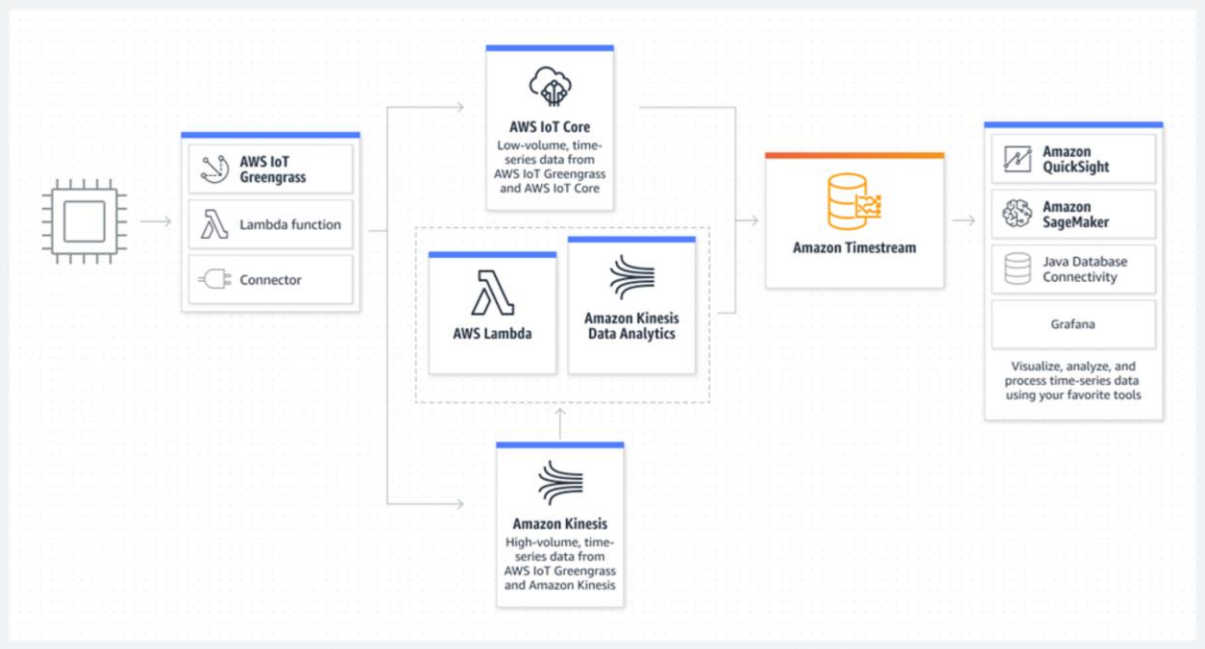
04 JournalDiskUtilization

Correct Answer: B
Explanation: WriteIOs measures writes to the QLDB ledger, which can help monitor availability.
Incorrect Answers:
A) ReadIOs measures read traffic but does not directly indicate availability issues.
C) JournalS3DataSize monitors storage usage but not availability.
D) JournalDiskUtilization is for disk space monitoring unrelated to availability.

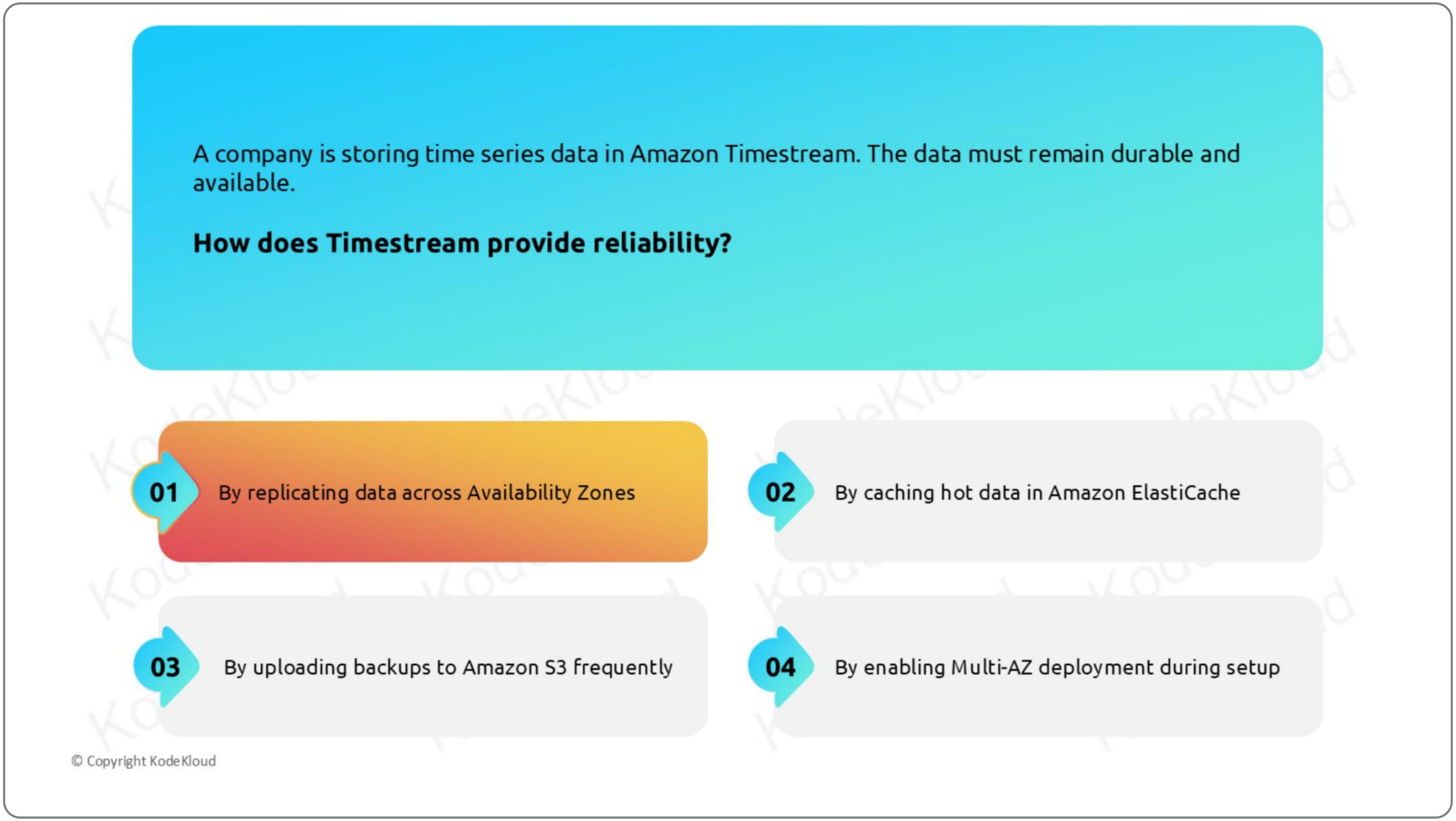
References:
https://docs.aws.amazon.com/qldb/latest/developerguide/metrics.html

Designing for Reliability – TimeStream

Timestream


AWS IoT Greengrass
Lambda function
Connector
AWS IoT Core
Low-volume, time-series data from AWS IoT Greengrass and AWS IoT Core
AWS Lambda
Amazon Kinesis Data Analytics
Amazon Kinesis
High-volume, time-series data from AWS IoT Greengrass and Amazon Kinesis
Amazon Timestream
Amazon QuickSight
Amazon SageMaker
Java Database Connectivity
Grafana
Visualize, analyze, and process time-series data using your favorite tools

TimeStream is built to be immutable around time-captured data.



A company is storing time series data in Amazon Timestream. The data must remain durable and available.

How does Timestream provide reliability?

01 By replicating data across Availability Zones

02 By caching hot data in Amazon ElastiCache

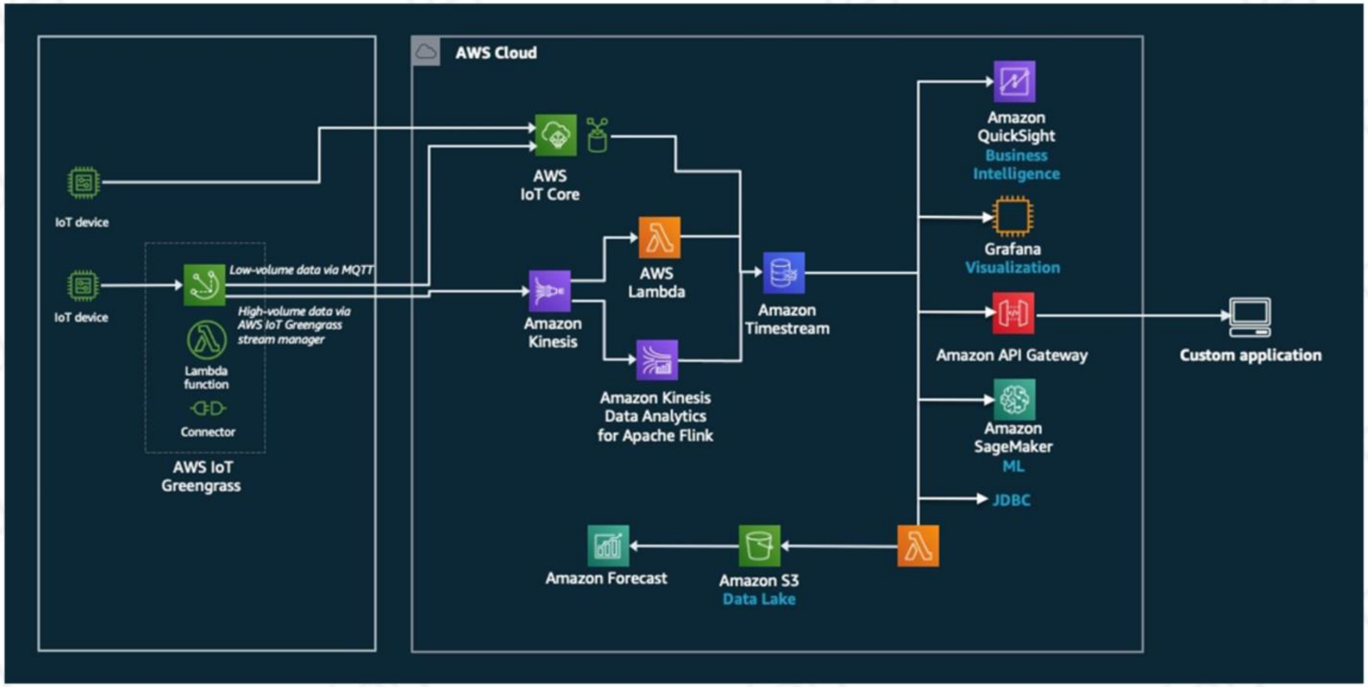
03 By uploading backups to Amazon S3 frequently

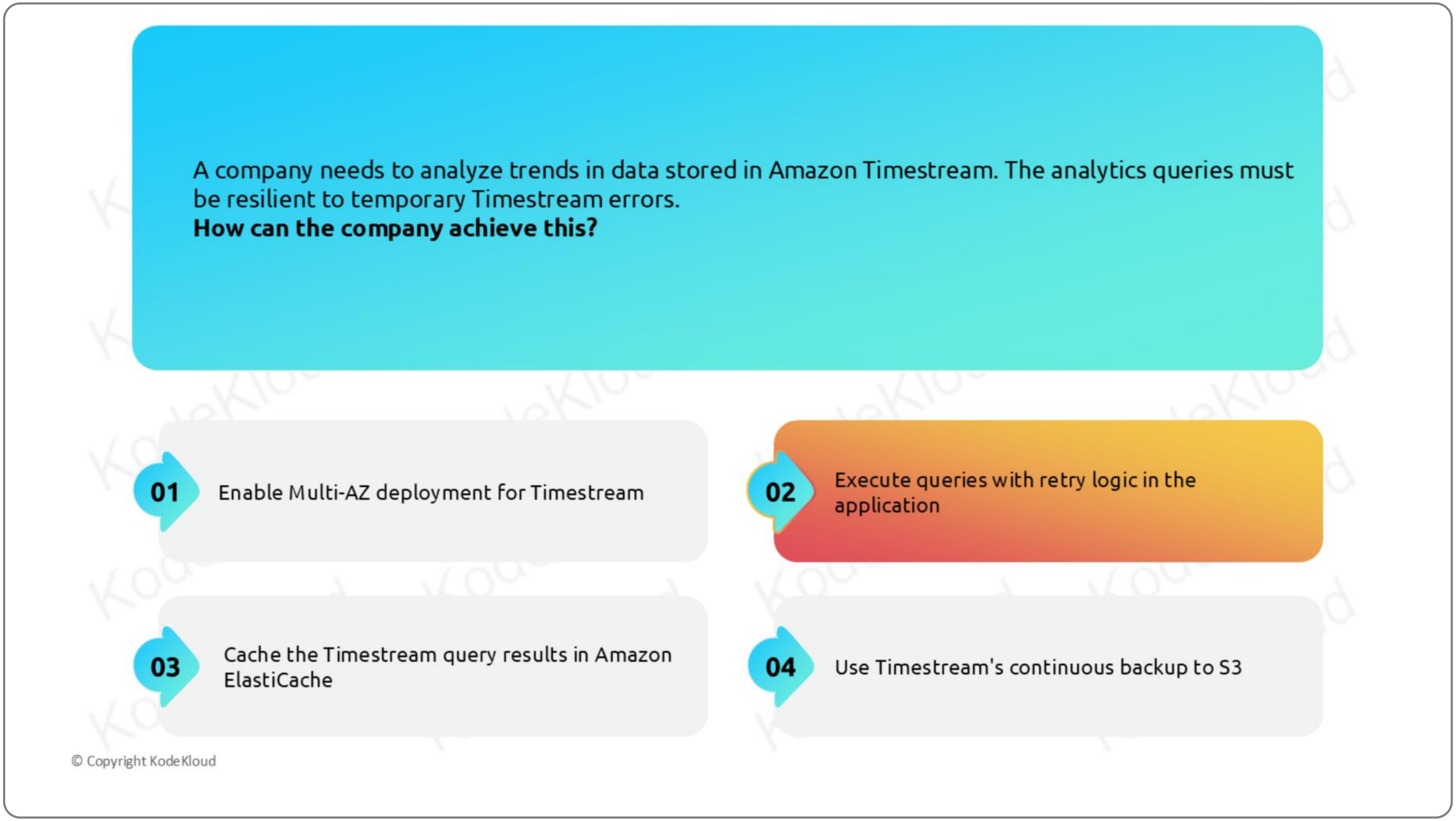
04 By enabling Multi-AZ deployment during setup

Correct Answer: A
Explanation: Amazon Timestream replicates data across Availability Zones to provide durable data storage and automatic failover in case of AZ outages.
Incorrect Answers:
B) Caching in ElastiCache does not provide data durability.
C) S3 backups help with restores but do not provide real-time reliability.

D) Multi-AZ is built-in, not a configurable deployment option.
References:
https://docs.aws.amazon.com/timestream/latest/developerguide/high-availability.html

Timestream


AWS Cloud
IoT device
IoT device
Low-volume data via MQTT
High-volume data via AWS IoT Greengrass stream manager
Lambda function
Connector
AWS IoT Greengrass
AWS IoT Core
AWS Lambda
Amazon Kinesis
Amazon Kinesis Data Analytics for Apache Flink
Amazon Timestream
Amazon QuickSight
Business Intelligence
Grafana
Visualization
Amazon API Gateway
Custom application
Amazon SageMaker
ML
JDBC
Amazon Forecast
Amazon S3
Data Lake



A company needs to analyze trends in data stored in Amazon Timestream. The analytics queries must be resilient to temporary Timestream errors.

How can the company achieve this?

01 Enable Multi-AZ deployment for Timestream

02 Execute queries with retry logic in the application

03 Cache the Timestream query results in Amazon ElastiCache

04 Use Timestream's continuous backup to S3

Correct Answer: B

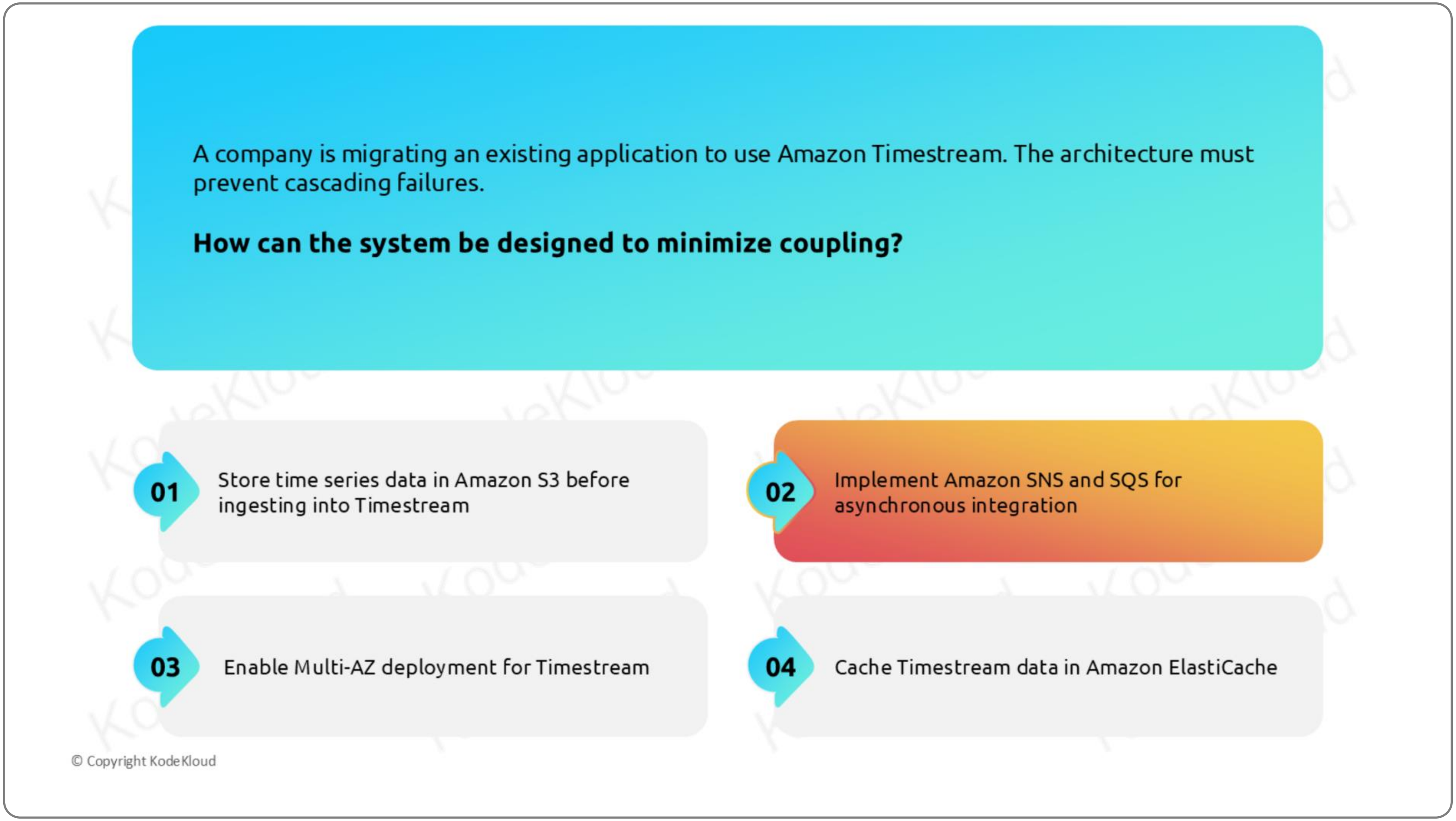
Explanation: Implementing retries in the application logic allows queries to Timestream to be resilient to transient errors and temporary service disruptions.

Incorrect Answers:

A) Multi-AZ is automatically built into Timestream and cannot be configured.

C) Caching results does not provide resiliency to source query errors.

D) Backups to S3 provide disaster recovery, not query resilience.
References:
https://docs.aws.amazon.com/timestream/latest/developerguide/best-practices.html#retry
Domain: Design Resilient z



A company is migrating an existing application to use Amazon Timestream. The architecture must prevent cascading failures.

How can the system be designed to minimize coupling?

01 Store time series data in Amazon S3 before ingesting into Timestream

02 Implement Amazon SNS and SQS for asynchronous integration

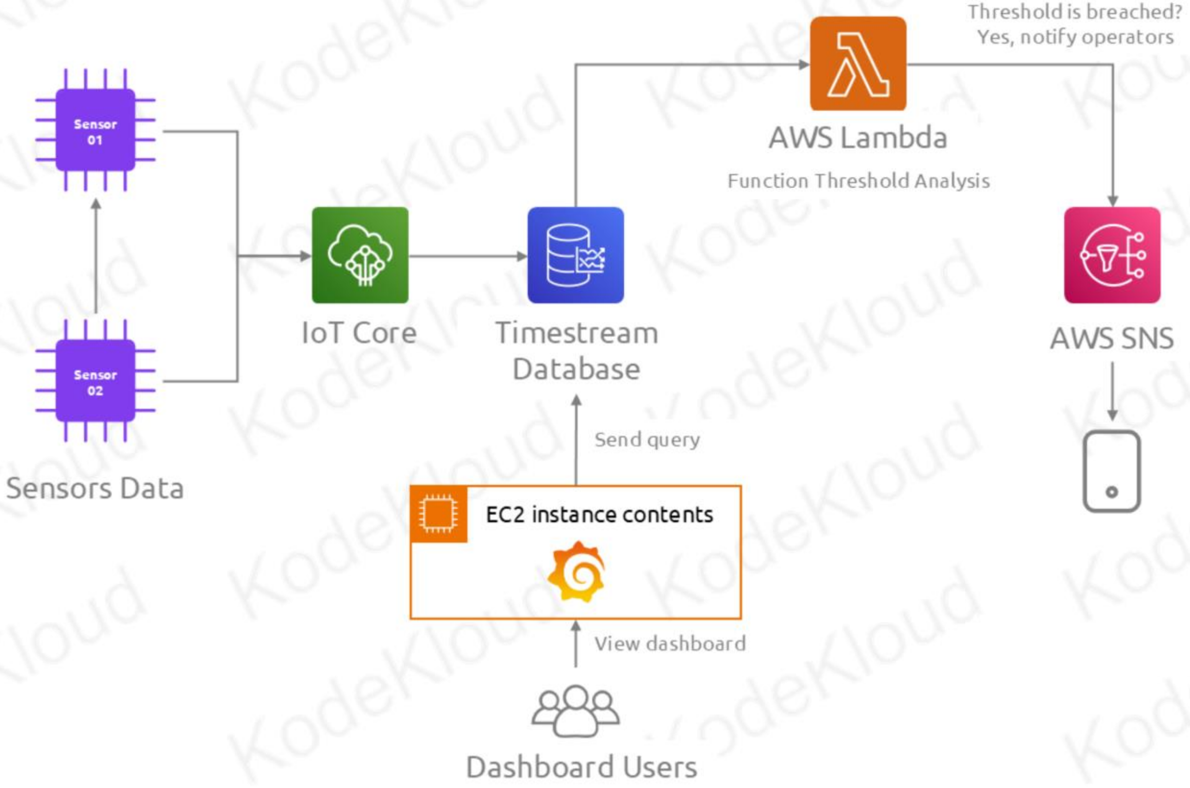
03 Enable Multi-AZ deployment for Timestream

04 Cache Timestream data in Amazon ElastiCache

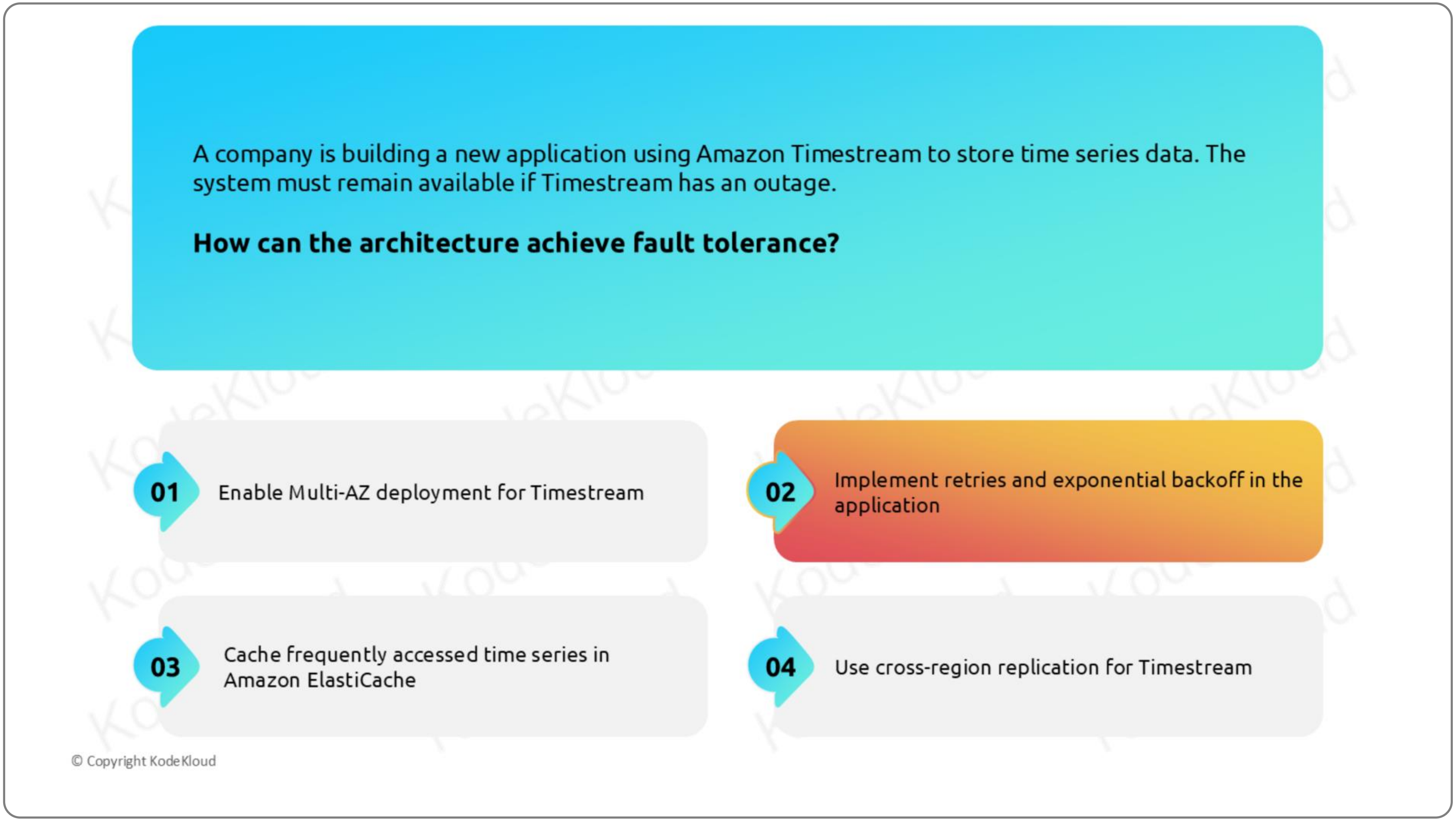
Correct Answer: B
Explanation: Using Amazon SNS and SQS allows loosely coupled asynchronous integration between the application and Timestream. This prevents cascading failures across services.
Incorrect Answers:
A) S3 provides durable storage but does not decouple components.
C) Multi-AZ is automatically built into Timestream and does not reduce coupling.

D) Caching in ElastiCache does not reduce coupling with Timestream.
References:
https://docs.aws.amazon.com/timestream/latest/developerguide/best-practices.html#decouple-with-sqs-sns

Timestream



Sensor 01
Sensor 02
Sensors Data
IoT Core
Timestream Database
AWS Lambda
Function Threshold Analysis
Threshold is breached? Yes, notify operators
AWS SNS
Send query
EC2 instance contents
View dashboard
Dashboard Users



A company is building a new application using Amazon Timestream to store time series data. The system must remain available if Timestream has an outage.

How can the architecture achieve fault tolerance?

01 Enable Multi-AZ deployment for Timestream

02 Implement retries and exponential backoff in the application

03 Cache frequently accessed time series in Amazon ElastiCache

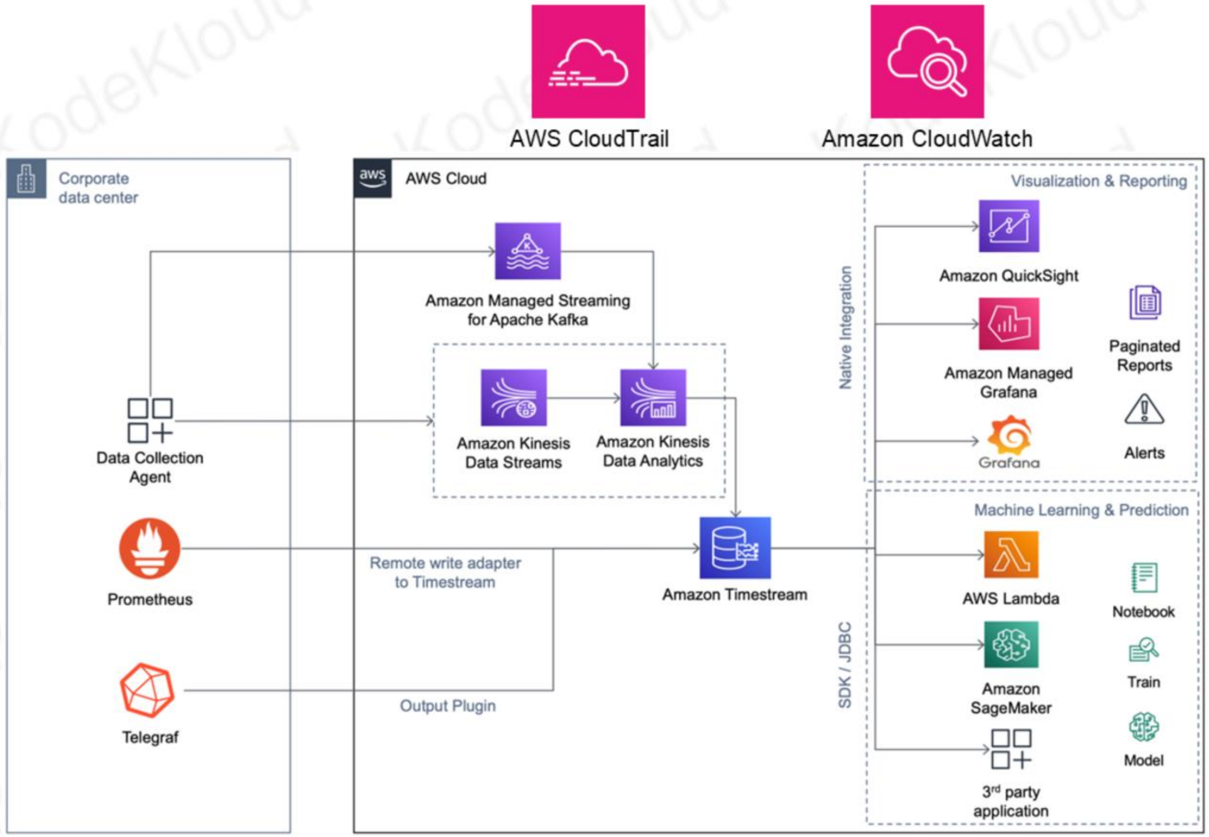
04 Use cross-region replication for Timestream

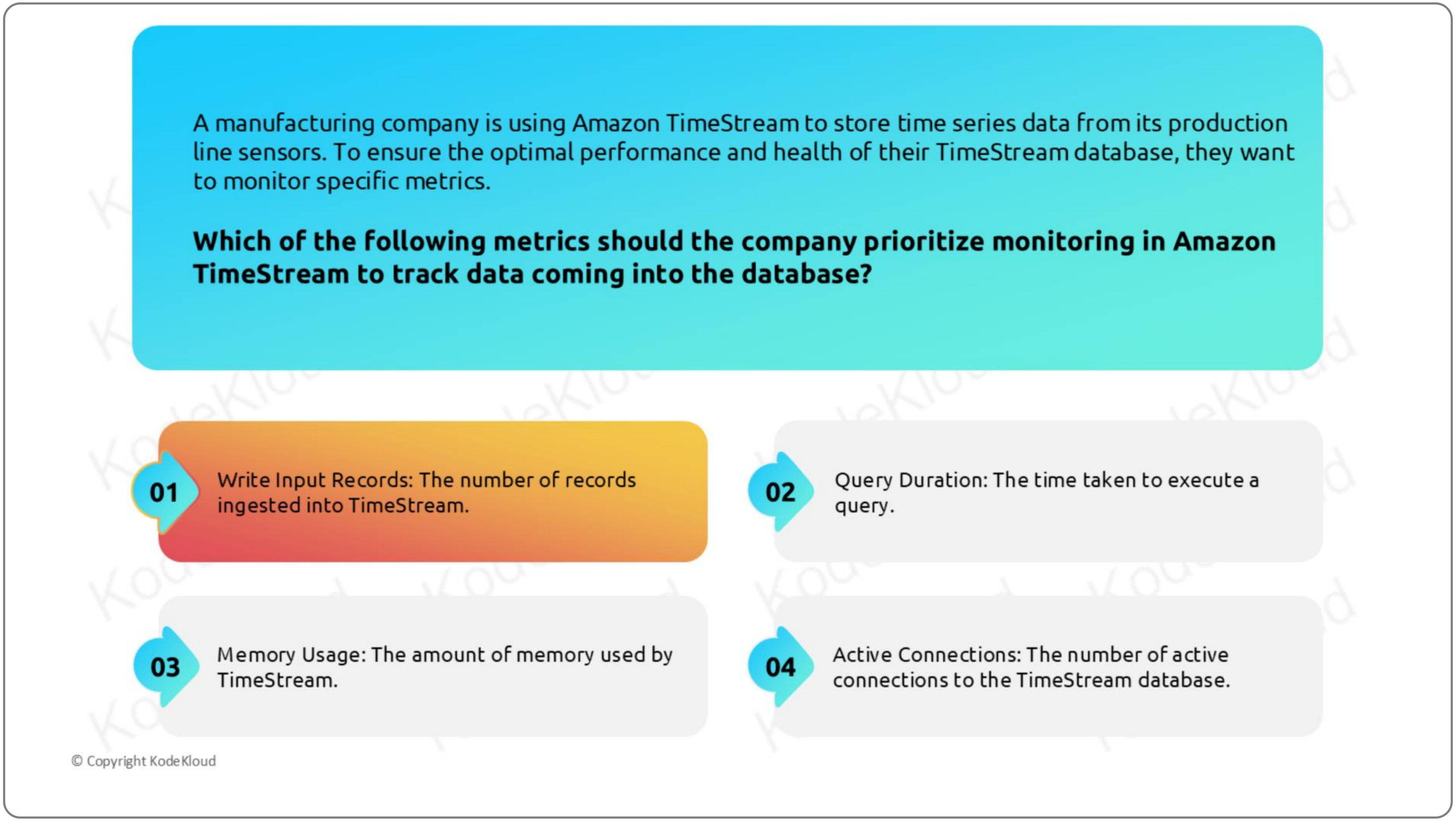
Correct Answer: B
Explanation: Implementing retries with exponential backoff in the application logic allows it to gracefully handle Timestream errors and outages.
Incorrect Answers:
A) Multi-AZ deployment is automatically built into Timestream.
C) Caching does not provide fault tolerance to Timestream outages.

D) Cross-region replication provides disaster recovery, not fault tolerance.
References:
https://docs.aws.amazon.com/timestream/latest/developerguide/best-practices.html#retry

Timestream

Timestream mainly reports through CloudWatch for host status and through CloudTrail to see API calls.





A manufacturing company is using Amazon TimeStream to store time series data from its production line sensors. To ensure the optimal performance and health of their TimeStream database, they want to monitor specific metrics.

Which of the following metrics should the company prioritize monitoring in Amazon TimeStream to track data coming into the database?

01 Write Input Records: The number of records ingested into TimeStream.

02 Query Duration: The time taken to execute a query.

03 Memory Usage: The amount of memory used by TimeStream.

04 Active Connections: The number of active connections to the TimeStream database.

Correct Answer: A. Write Input Records: The number of records ingested into TimeStream.
Incorrect: B. While monitoring the duration of queries can provide insights into performance, the ingestion rate (Write Input Records) is more critical for time-series databases like TimeStream. C. Amazon TimeStream is a serverless database, so users don't need to manage the underlying infrastructure or monitor memory usage. D. Active Connections might be relevant for traditional databases, but for a serverless database like TimeStream, the ingestion rate is more critical.
Explanation: Amazon TimeStream is optimized for time-series data. Monitoring the ingestion rate, represented by the

Write Input Records metric, is crucial to understand the data flow and ensure that all time-series data points are being captured correctly. This helps in identifying any anomalies or issues with data ingestion and ensures data integrity.

References:

Amazon TimeStream: Monitoring with Amazon CloudWatch

Summary

01 This section covers all of the datastore services in AWS and options for reliability

02 Reliability in this space has more to do with ensuring redundancy if the DB is node based, otherwise it is inherently redundant

03 The new database options has tons of new options like replication or automated recovery/failover

04 Many of the reliability metrics were covered in Security, but some were mentioned in this section as well for completeness

Designing for Reliability

Turning up Reliability on Application Integration Services

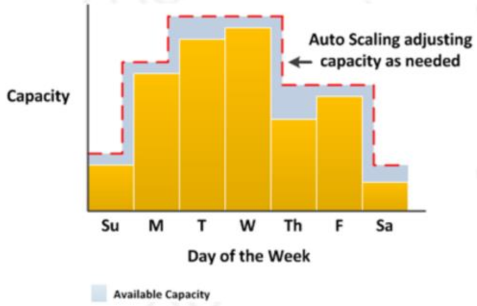
Our Approach to Design

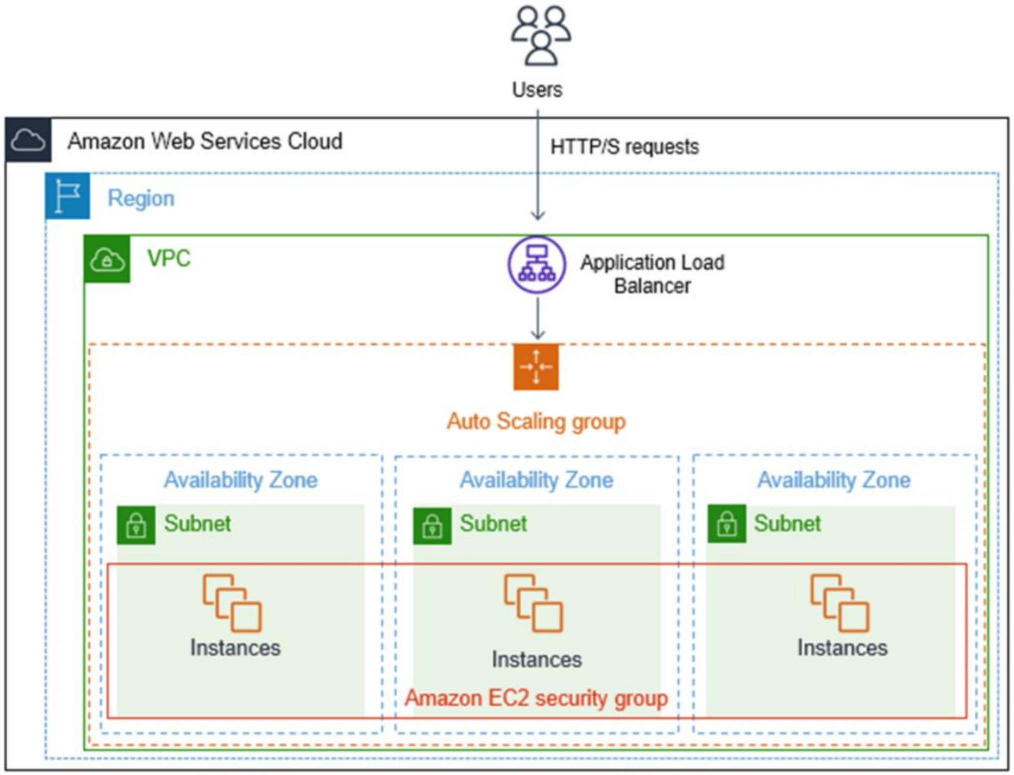
The Power of Scaling

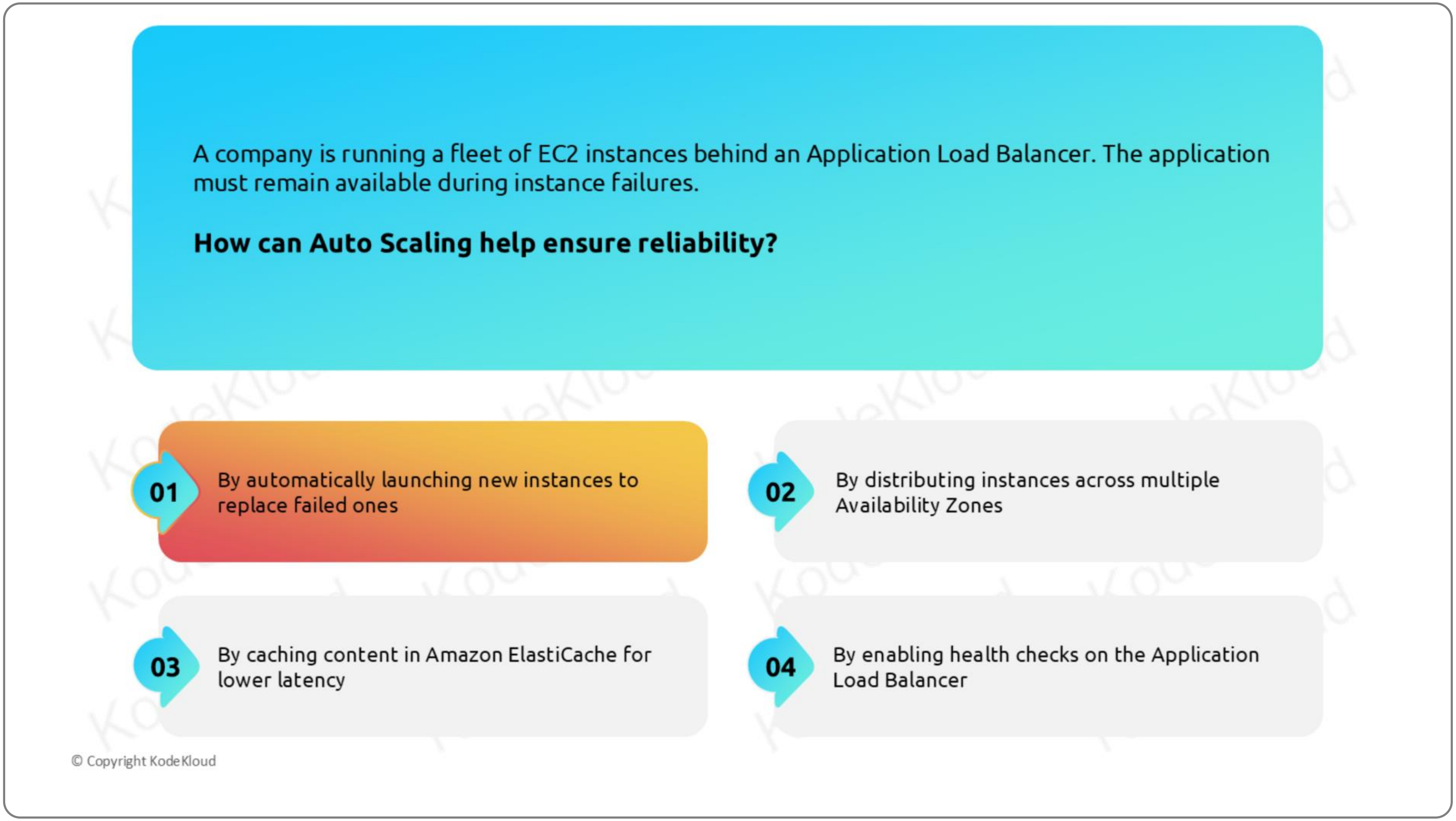
Designing for Reliability – Autoscaling

Autoscaling

Autoscaling doesn't have much in Reliability because it is an enabler service. It looks like a single point of failure, but is actually the entry point for a bunch of devices that AWS doesn't let you see.







A company is running a fleet of EC2 instances behind an Application Load Balancer. The application must remain available during instance failures.

How can Auto Scaling help ensure reliability?

01 By automatically launching new instances to replace failed ones

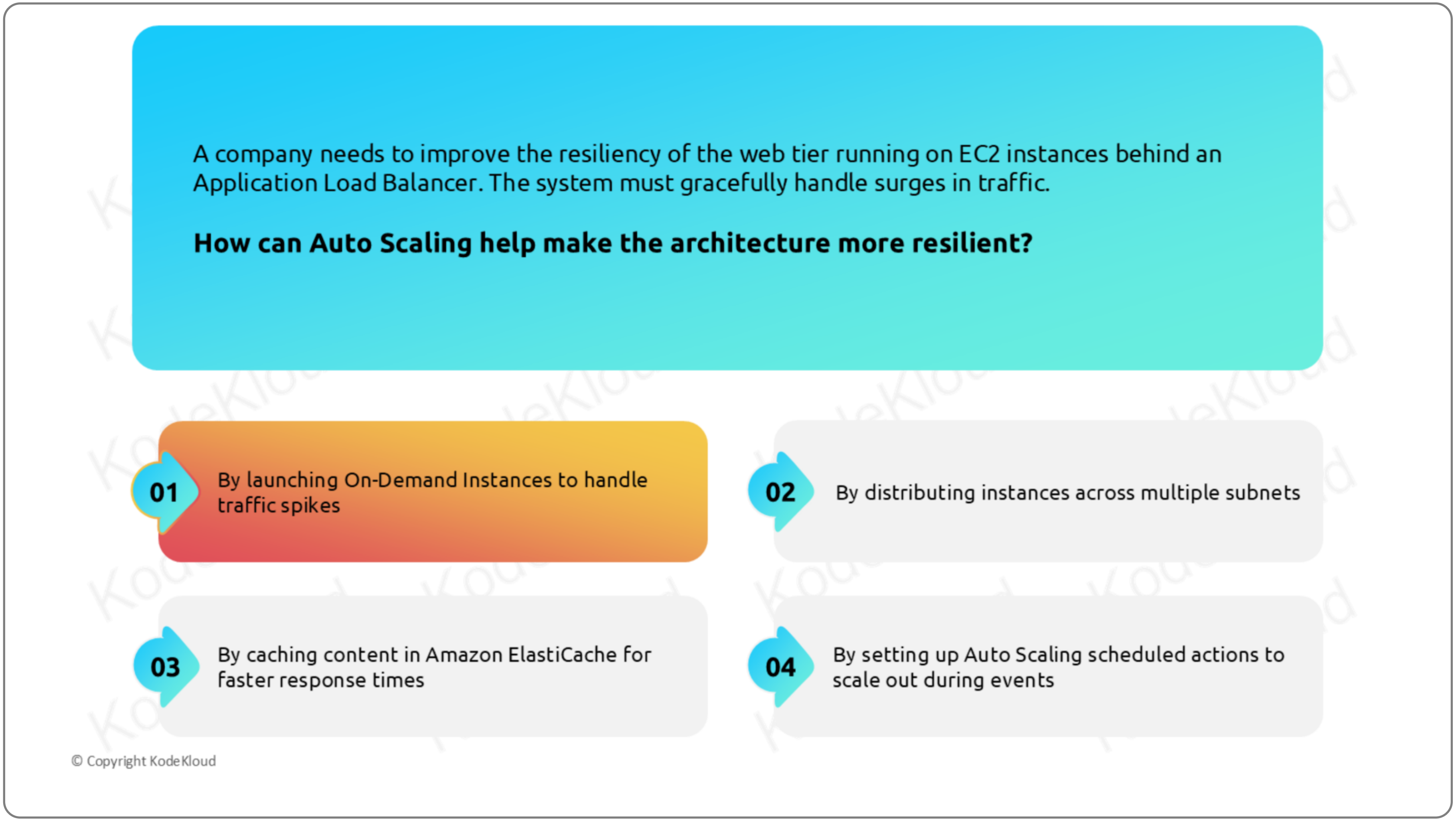
02 By distributing instances across multiple Availability Zones

03 By caching content in Amazon ElastiCache for lower latency

04 By enabling health checks on the Application Load Balancer

Correct Answer: A
Explanation: Auto Scaling automatically launches new instances to replace failed ones, ensuring a minimum number of healthy instances are always available to serve traffic.
Incorrect Answers:
B) Distributing instances is good practice but does not directly replace failed ones.
C) Caching content in ElastiCache does not replace failed instances.

D) Health checks enable load balancer to route around failures but do not launch new instances.
References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html



A company needs to improve the resiliency of the web tier running on EC2 instances behind an Application Load Balancer. The system must gracefully handle surges in traffic.

How can Auto Scaling help make the architecture more resilient?

01 By launching On-Demand Instances to handle traffic spikes

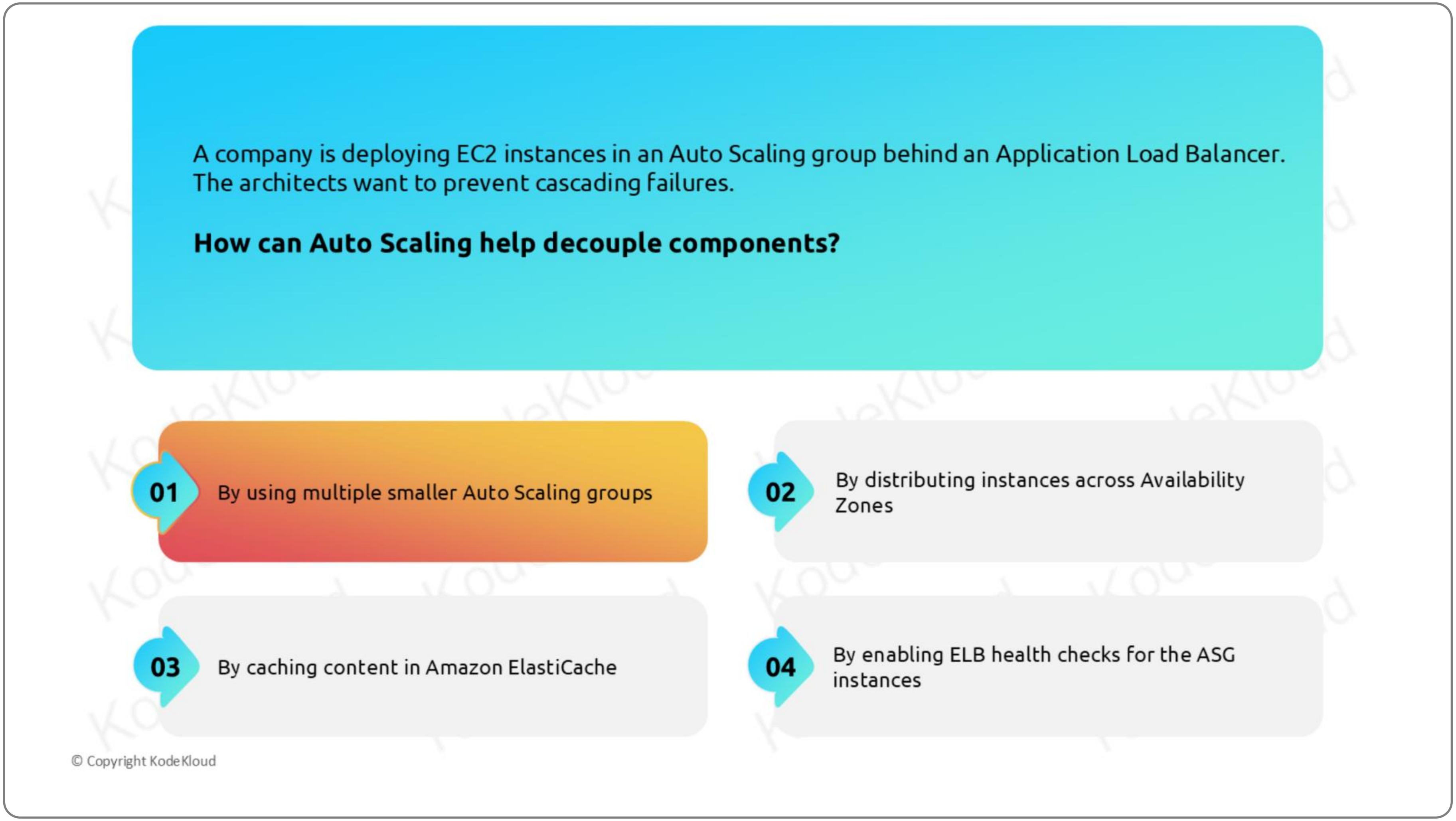
02 By distributing instances across multiple subnets

03 By caching content in Amazon ElastiCache for faster response times

04 By setting up Auto Scaling scheduled actions to scale out during events

Correct Answer: A
Explanation: Auto Scaling can automatically launch On-Demand instances to rapidly scale out and handle traffic spikes, improving architecture resilience.
Incorrect Answers:
B) Distributing instances provides availability but does not directly improve resilience to traffic spikes.
C) Caching content in ElastiCache can improve performance but does not increase capacity.

D) Scheduled actions scale on a fixed schedule which does not adapt to real-time traffic changes.
References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scale-based-on-demand.html



A company is deploying EC2 instances in an Auto Scaling group behind an Application Load Balancer. The architects want to prevent cascading failures.

How can Auto Scaling help decouple components?

01 By using multiple smaller Auto Scaling groups

02 By distributing instances across Availability Zones

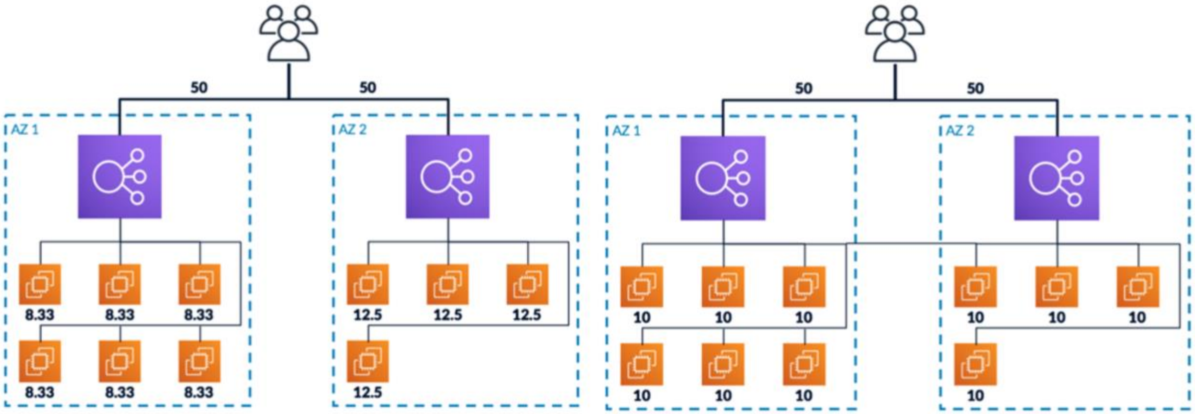
03 By caching content in Amazon ElastiCache

04 By enabling ELB health checks for the ASG instances

Correct Answer: A
Explanation: Using multiple independent Auto Scaling groups decouples components to prevent cascading failures across layers.
Incorrect Answers:
B) Distributing instances provides high availability but does not decouple components.
C) Caching content does not reduce coupling between application components.

D) Health checks monitor instance health but do not decouple ASGs.
References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/auto-scaling-benefits.html

Autoscaling



Cross-Zone Load Balancing Disabled

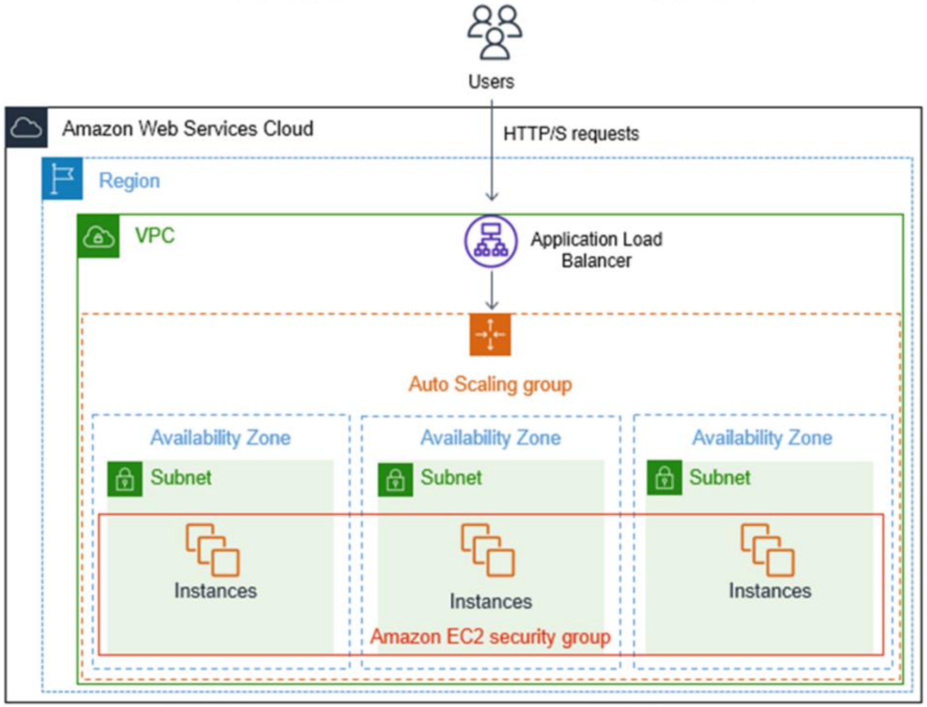
Cross-Zone Load Balancing Enabled

Autoscaling usually crosses AZs by default, but remember that if Cross Zone Load Balancing is not enabled on the Load Balancer, then you are going to see uneven connection numbers.

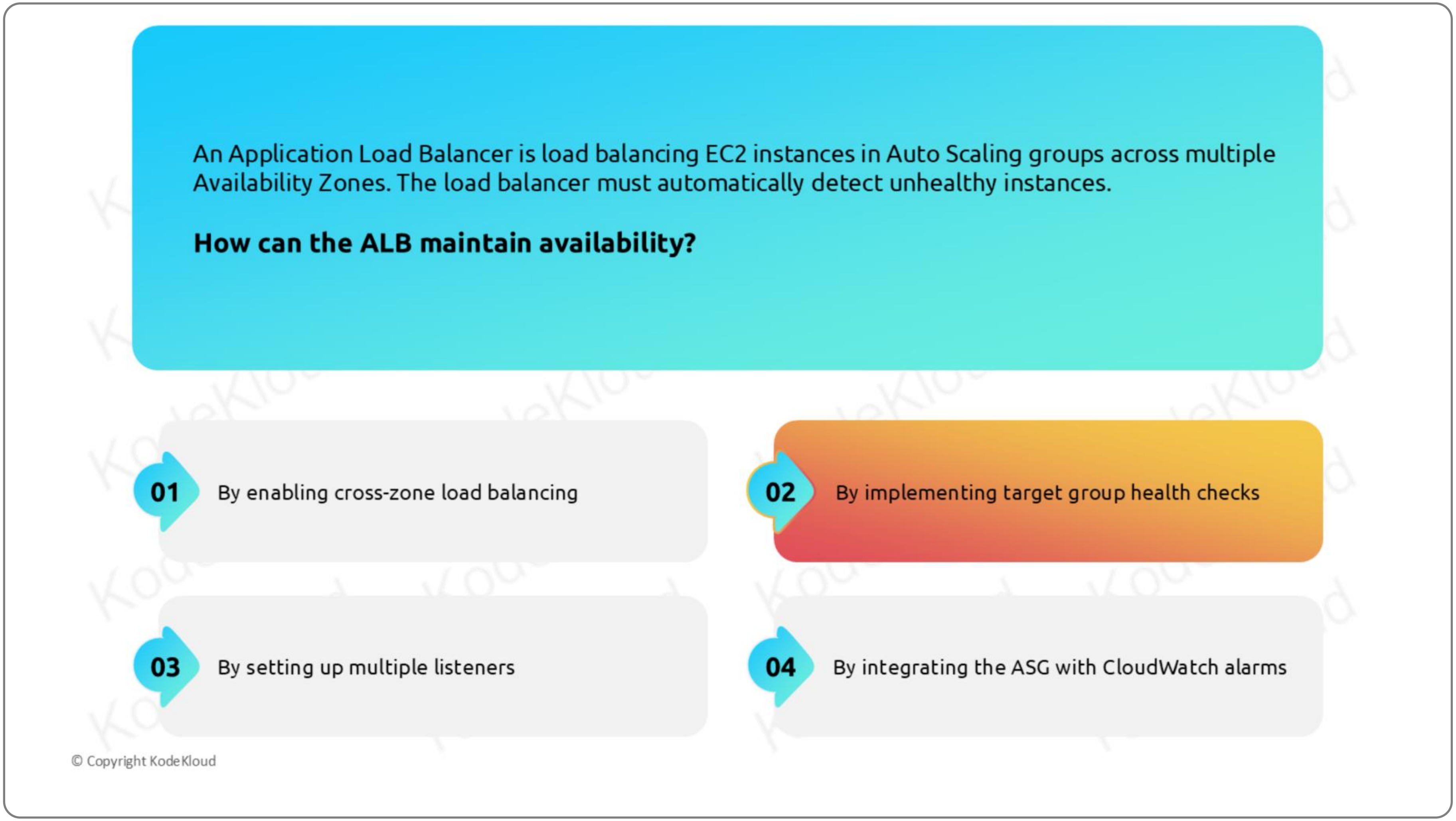
Designing for Reliability – Elastic Load Balancing

Application Load Balancer

Application Load Balancer



ALB is also an enabler service, but it has a few Reliability pieces.



An Application Load Balancer is load balancing EC2 instances in Auto Scaling groups across multiple Availability Zones. The load balancer must automatically detect unhealthy instances.

How can the ALB maintain availability?

01 By enabling cross-zone load balancing

02 By implementing target group health checks

03 By setting up multiple listeners

04 By integrating the ASG with CloudWatch alarms

Correct Answer: B

Explanation: Application Load Balancer health checks can automatically detect unhealthy instances and stop routing traffic to them.

Incorrect Answers:

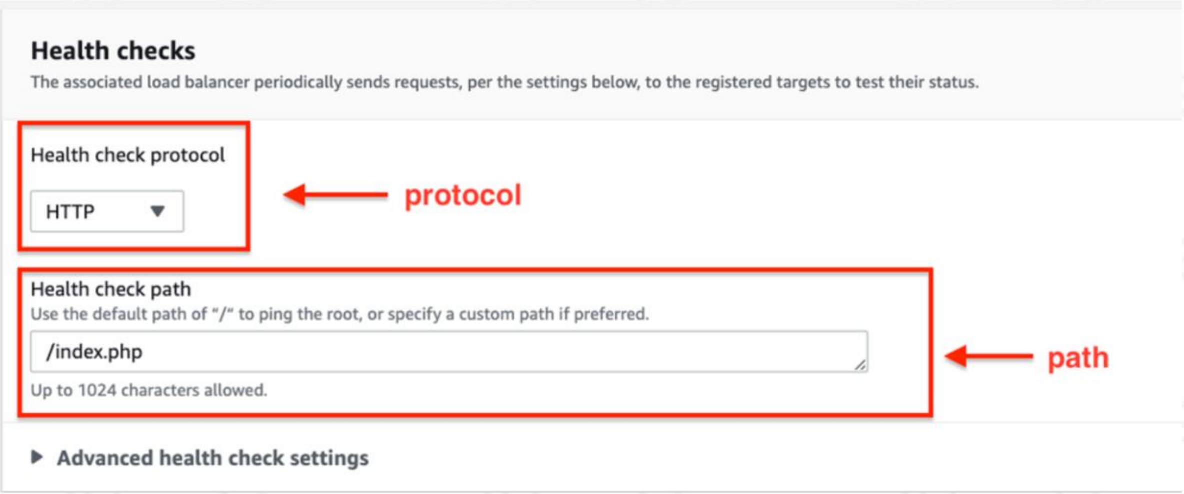
A) Cross-zone load balancing provides AZ resiliency but does not detect unhealthy instances.

C) Multiple listeners allow segmented traffic but do not check instance health.

D) CloudWatch alarms can trigger autoscaling but do not detect unhealthy instances.

References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/target-group-health-checks.html

Application Load Balancer



Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

HTTP ▼

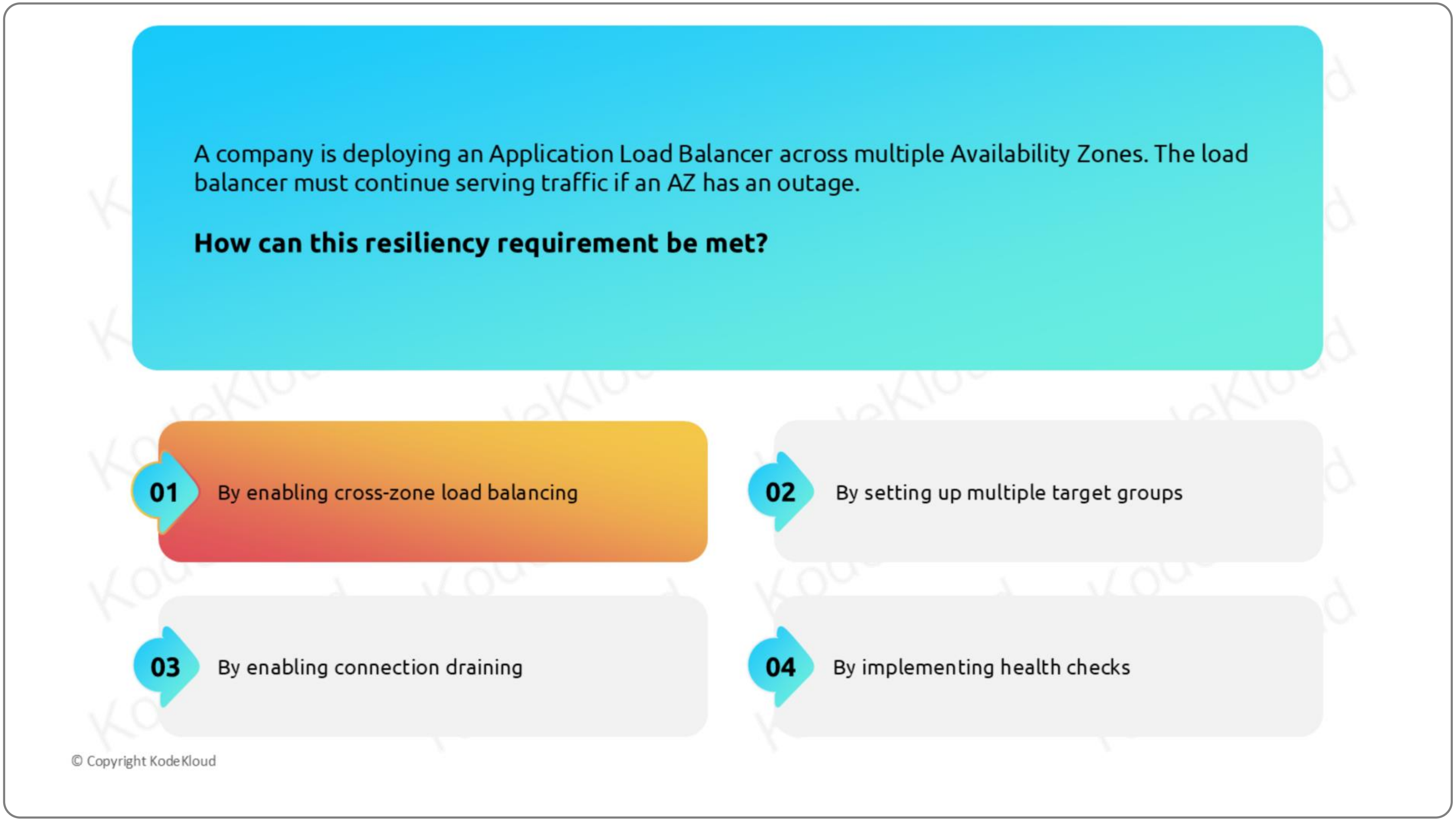
Health check path

Use the default path of "/" to ping the root, or specify a custom path if preferred.

/index.php

Up to 1024 characters allowed.

▶ Advanced health check settings



A company is deploying an Application Load Balancer across multiple Availability Zones. The load balancer must continue serving traffic if an AZ has an outage.

How can this resiliency requirement be met?

01 By enabling cross-zone load balancing

02 By setting up multiple target groups

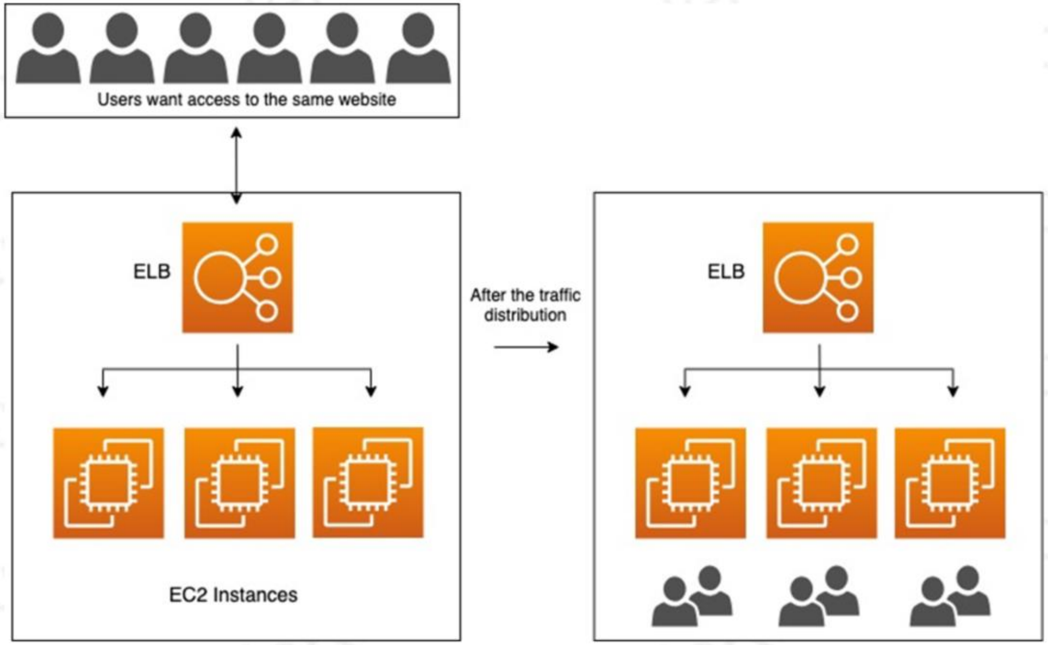
03 By enabling connection draining

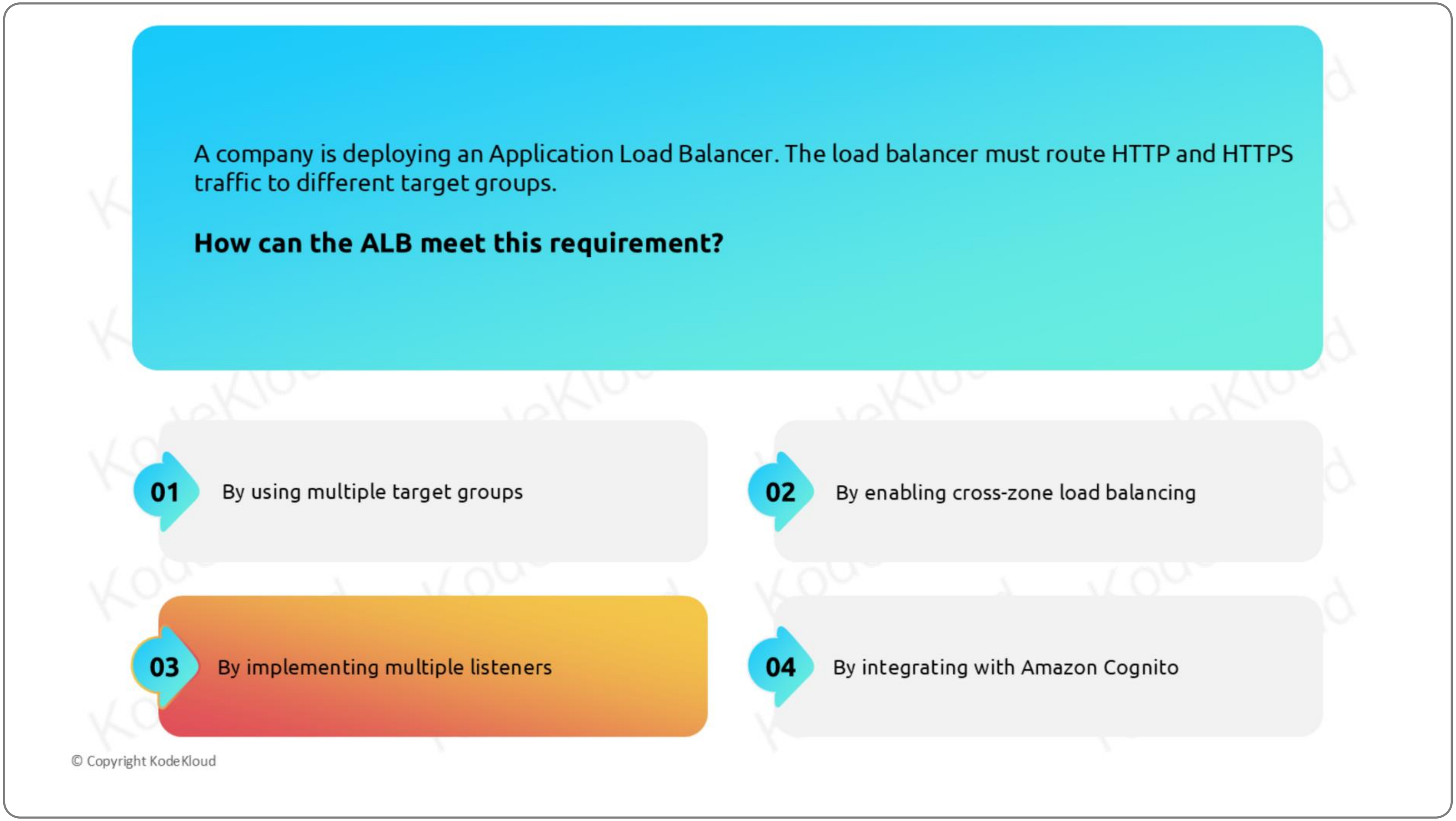
04 By implementing health checks

Correct Answer: A
Explanation: Enabling cross-zone load balancing allows the ALB to continue routing traffic to instances in the remaining healthy AZs during an AZ outage.
Incorrect Answers:
B) Multiple target groups help segment traffic but do not provide AZ redundancy.
C) Connection draining facilitates graceful shutdowns but does not provide AZ resiliency.

D) Health checks enable routing around unhealthy instances but do not provide AZ redundancy.
References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-subnets.html

Application Load Balancer


Users want access to the same website
ELB
EC2 Instances
After the traffic distribution
ELB



A company is deploying an Application Load Balancer. The load balancer must route HTTP and HTTPS traffic to different target groups.

How can the ALB meet this requirement?

01 By using multiple target groups

02 By enabling cross-zone load balancing

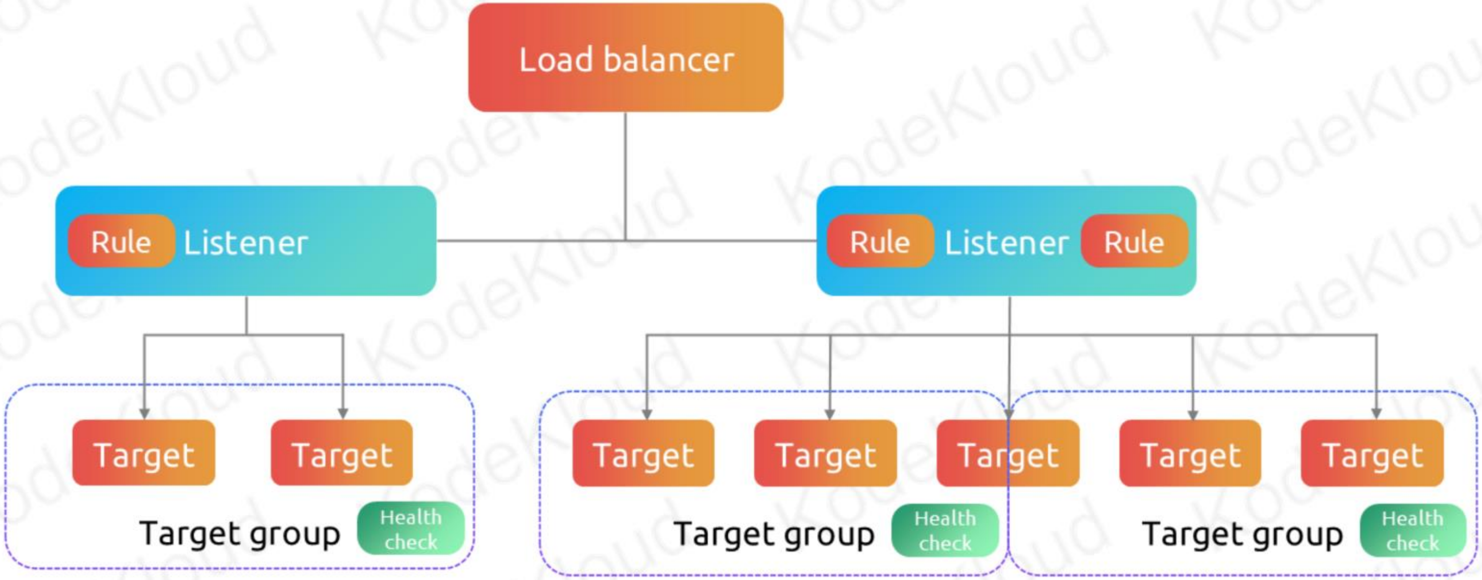
03 By implementing multiple listeners

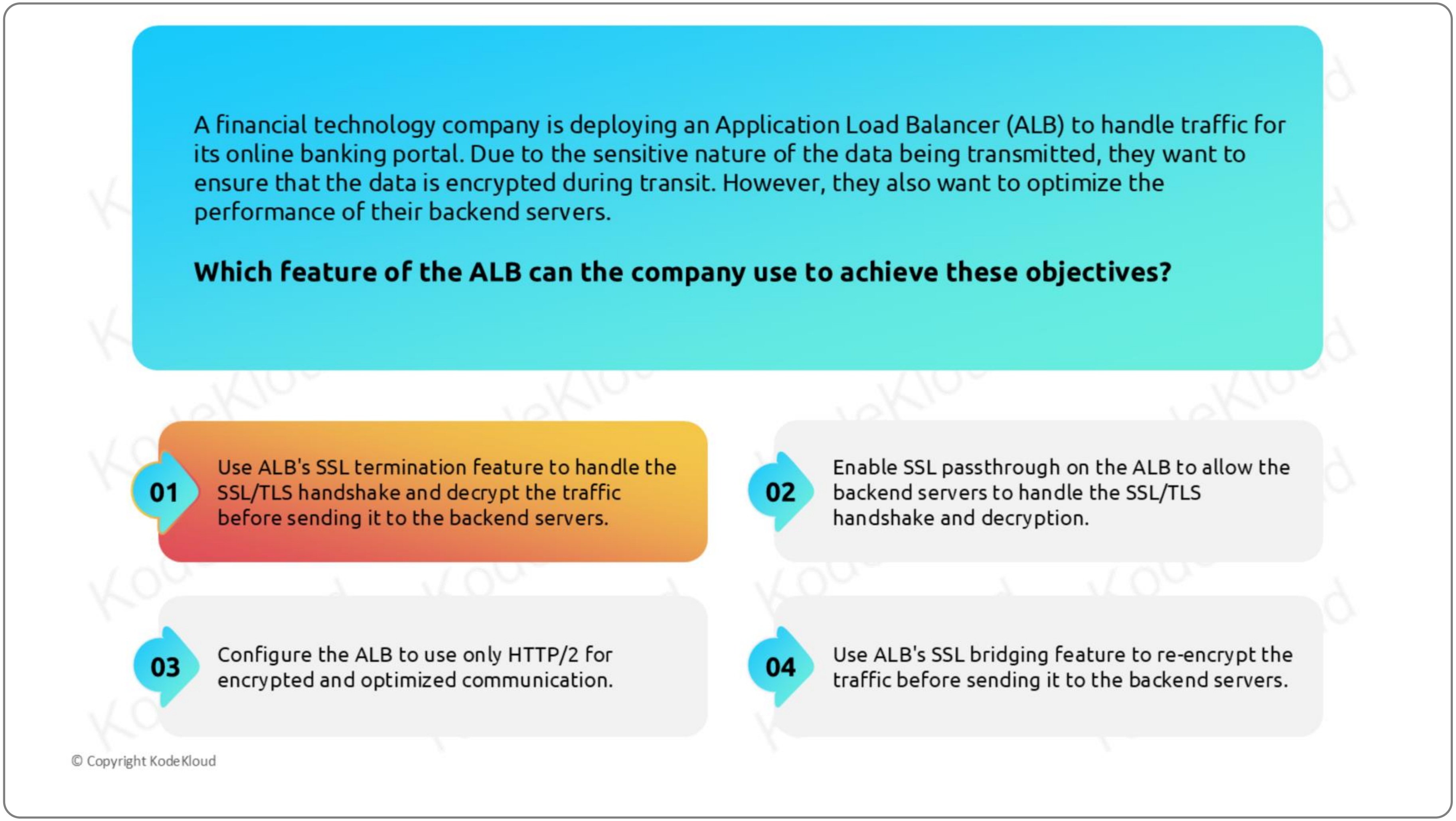
04 By integrating with Amazon Cognito

Correct Answer: C
Explanation: Application Load Balancer supports multiple listeners, allowing different listeners for HTTP and HTTPS traffic mapped to different target groups.
Incorrect Answers:
A) Multiple target groups allows segmenting instance groups but multiple listeners would still be needed.
B) Cross-zone load balancing provides high availability but does not enable multiple listeners.

D) Amazon Cognito enables user authentication but does not support multiple listeners.
References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-listeners.html

Application Load Balancer





A financial technology company is deploying an Application Load Balancer (ALB) to handle traffic for its online banking portal. Due to the sensitive nature of the data being transmitted, they want to ensure that the data is encrypted during transit. However, they also want to optimize the performance of their backend servers.

Which feature of the ALB can the company use to achieve these objectives?

01 Use ALB's SSL termination feature to handle the SSL/TLS handshake and decrypt the traffic before sending it to the backend servers.

02 Enable SSL passthrough on the ALB to allow the backend servers to handle the SSL/TLS handshake and decryption.

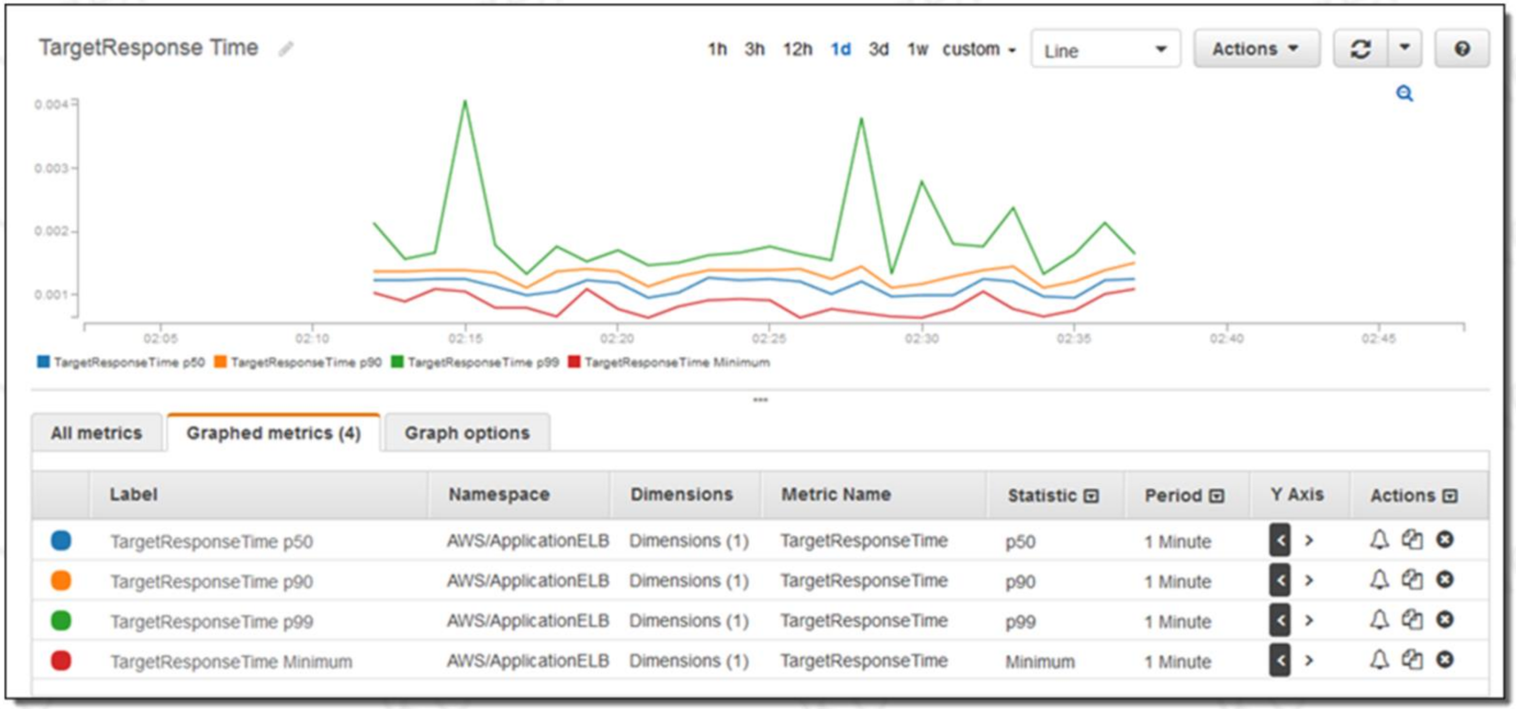
03 Configure the ALB to use only HTTP/2 for encrypted and optimized communication.

04 Use ALB's SSL bridging feature to re-encrypt the traffic before sending it to the backend servers.

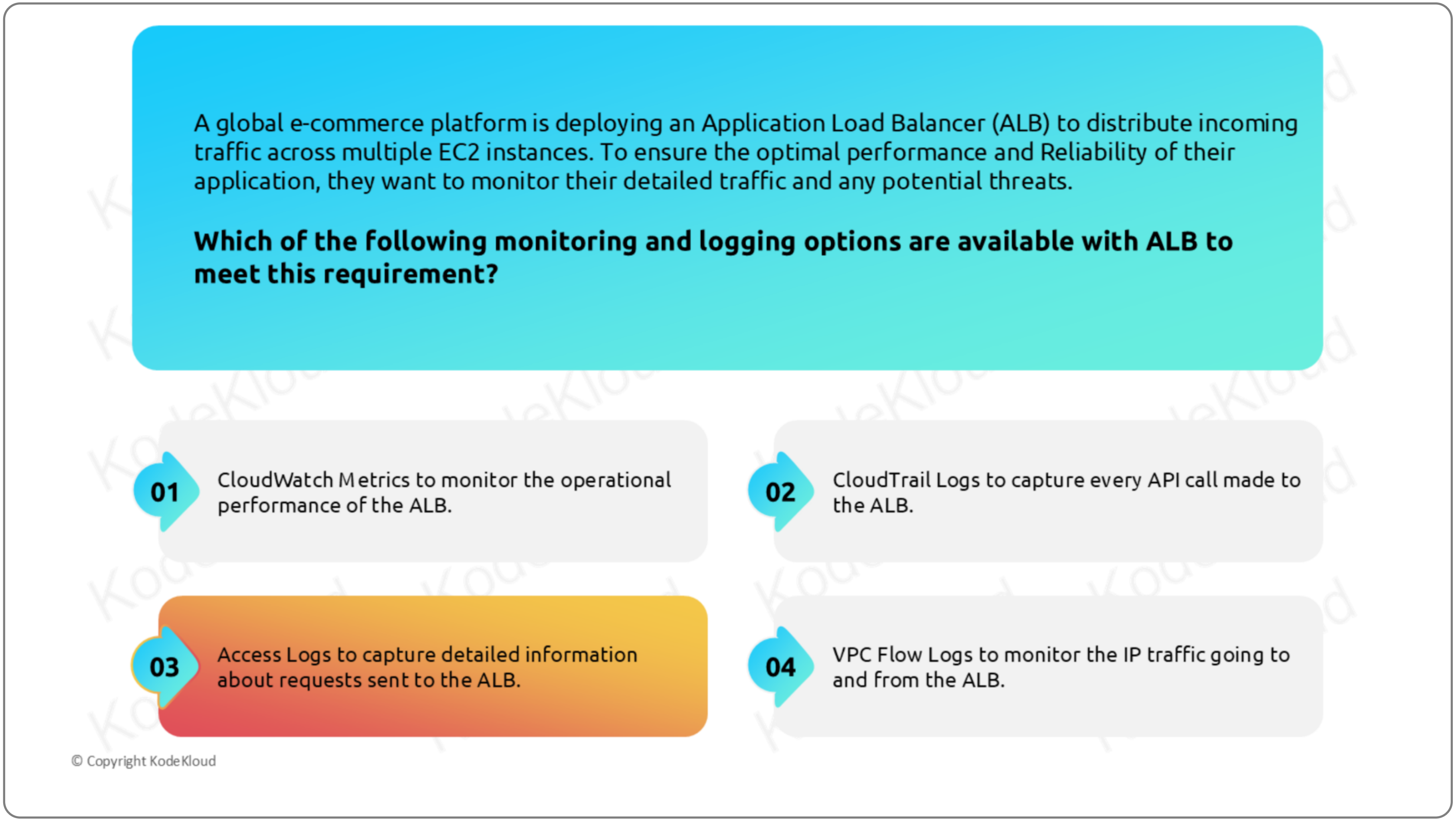
Correct Answer: A. Use ALB's SSL termination feature to handle the SSL/TLS handshake and decrypt the traffic before sending it to the backend servers.
Incorrect: B. SSL passthrough would mean the backend servers handle the decryption, which doesn't offload the SSL processing from them. C. While HTTP/2 can provide performance benefits, it doesn't address the SSL offloading requirement. D. ALB doesn't have a feature specifically called "SSL bridging." The primary method for offloading SSL processing is SSL termination.

Explanation: SSL offloading, also known as SSL termination, refers to the process of decrypting SSL traffic at the load balancer level rather than at the backend servers. By using the ALB's SSL termination feature, the company can offload the CPU-intensive process of handling the SSL/TLS handshake and decrypting traffic from the backend servers. This ensures that data is encrypted during transit while also optimizing the performance of the backend servers.
References: Application Load Balancers: SSL Termination

Application Load Balancer



	Label	Namespace	Dimensions	Metric Name	Statistic	Period	Y Axis	Actions
🔵	TargetResponseTime p50	AWS/ApplicationELB	Dimensions (1)	TargetResponseTime	p50	1 Minute	‹ ›	🔔 ⧉ ✖
🟠	TargetResponseTime p90	AWS/ApplicationELB	Dimensions (1)	TargetResponseTime	p90	1 Minute	‹ ›	🔔 ⧉ ✖
🟢	TargetResponseTime p99	AWS/ApplicationELB	Dimensions (1)	TargetResponseTime	p99	1 Minute	‹ ›	🔔 ⧉ ✖
🔴	TargetResponseTime Minimum	AWS/ApplicationELB	Dimensions (1)	TargetResponseTime	Minimum	1 Minute	‹ ›	🔔 ⧉ ✖



A global e-commerce platform is deploying an Application Load Balancer (ALB) to distribute incoming traffic across multiple EC2 instances. To ensure the optimal performance and Reliability of their application, they want to monitor their detailed traffic and any potential threats.

Which of the following monitoring and logging options are available with ALB to meet this requirement?

01 CloudWatch Metrics to monitor the operational performance of the ALB.

02 CloudTrail Logs to capture every API call made to the ALB.

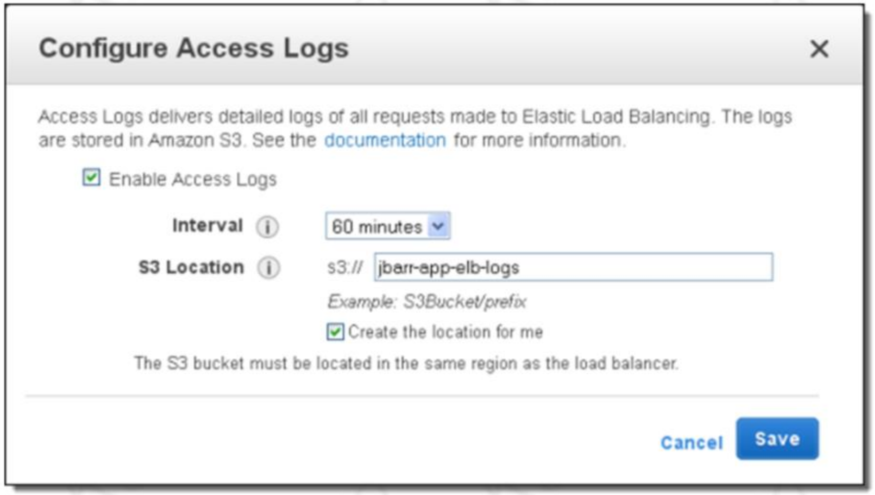
03 Access Logs to capture detailed information about requests sent to the ALB.

04 VPC Flow Logs to monitor the IP traffic going to and from the ALB.

Correct Answer: C. Access Logs to capture detailed information about requests sent to the ALB.
Incorrect: A. While CloudWatch Metrics can be used to monitor the operational performance of various AWS services, it doesn't provide detailed logging of requests made to the ALB. B. CloudTrail captures API calls made on AWS services, but it doesn't provide detailed request-level information for traffic sent to the ALB. D. VPC Flow Logs capture IP traffic information for VPCs, but they don't provide detailed request-level logs like the ALB's Access Logs.
Explanation: Access Logs for the Application Load Balancer provide detailed information about requests sent to the load

balancer. Each log contains information such as the client's IP address, request path, response code, and more. This detailed logging is crucial for analyzing traffic patterns, troubleshooting issues, and detecting potential Reliability threats.
References: Application Load Balancer: Access Logs

Application Load Balancer



Configure Access Logs

Access Logs delivers detailed logs of all requests made to Elastic Load Balancing. The logs are stored in Amazon S3. See the documentation for more information.

☑ Enable Access Logs

Interval 60 minutes

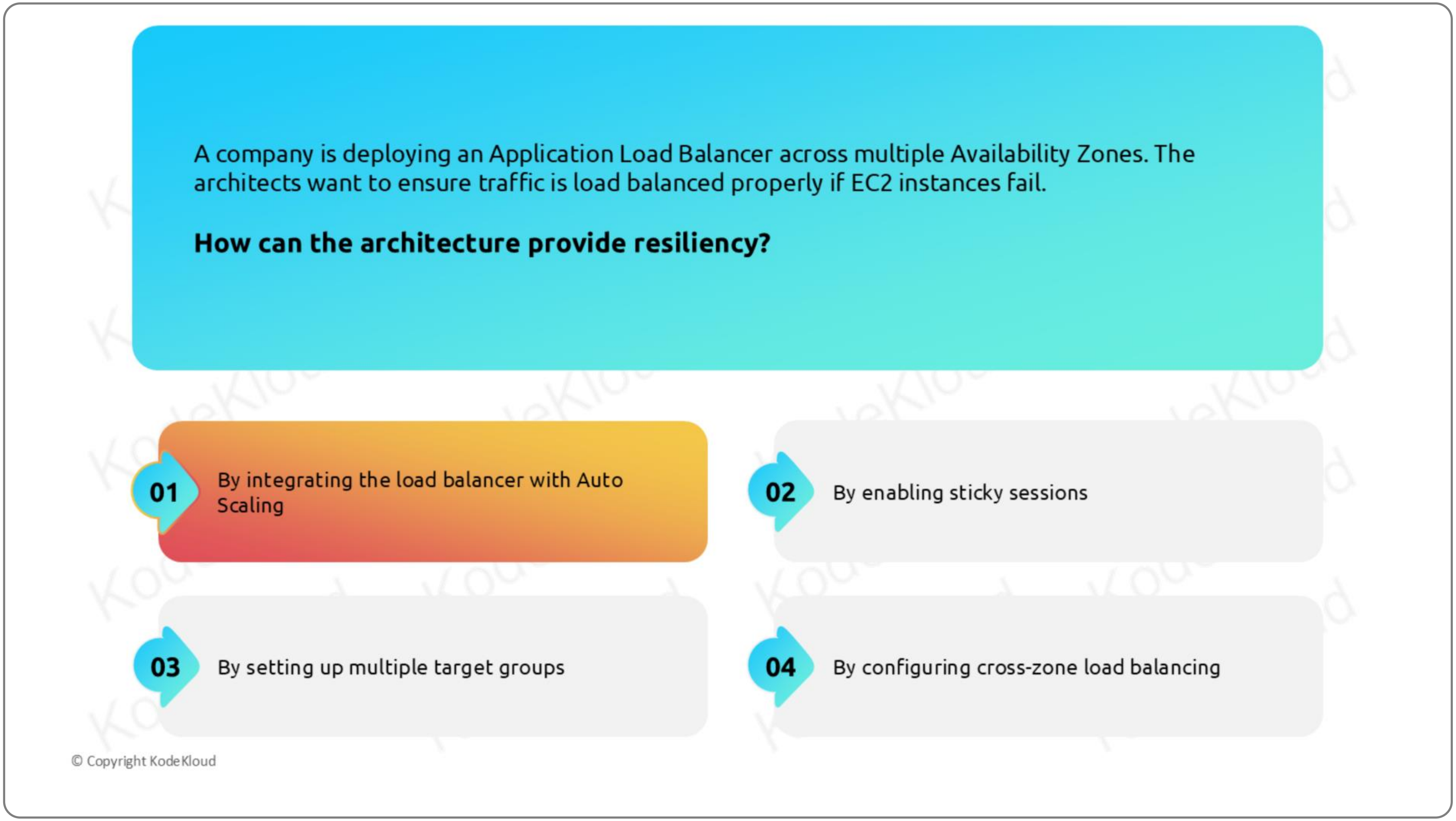
S3 Location s3:// jbarr-app-elb-logs

Example: S3Bucket/prefix

☑ Create the location for me

The S3 bucket must be located in the same region as the load balancer.

Logging following the standards of CloudTrail and CloudWatch, but there are also ELB Access Logs for details.



A company is deploying an Application Load Balancer across multiple Availability Zones. The architects want to ensure traffic is load balanced properly if EC2 instances fail.

How can the architecture provide resiliency?

01 By integrating the load balancer with Auto Scaling

02 By enabling sticky sessions

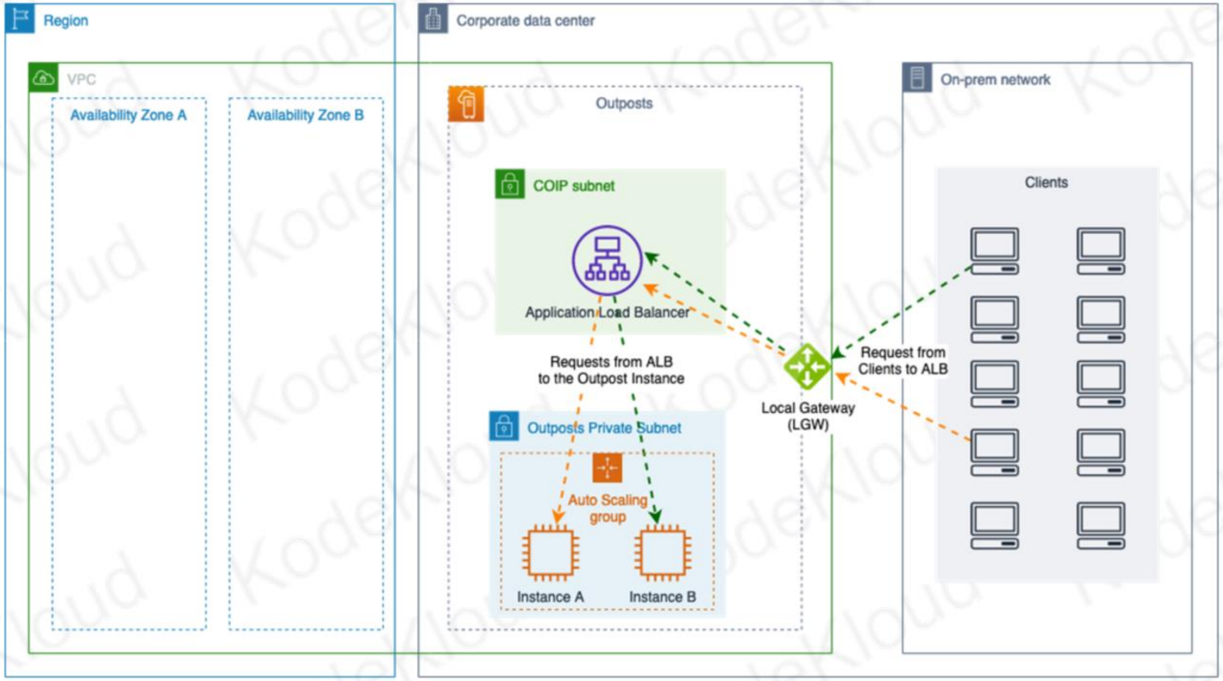
03 By setting up multiple target groups

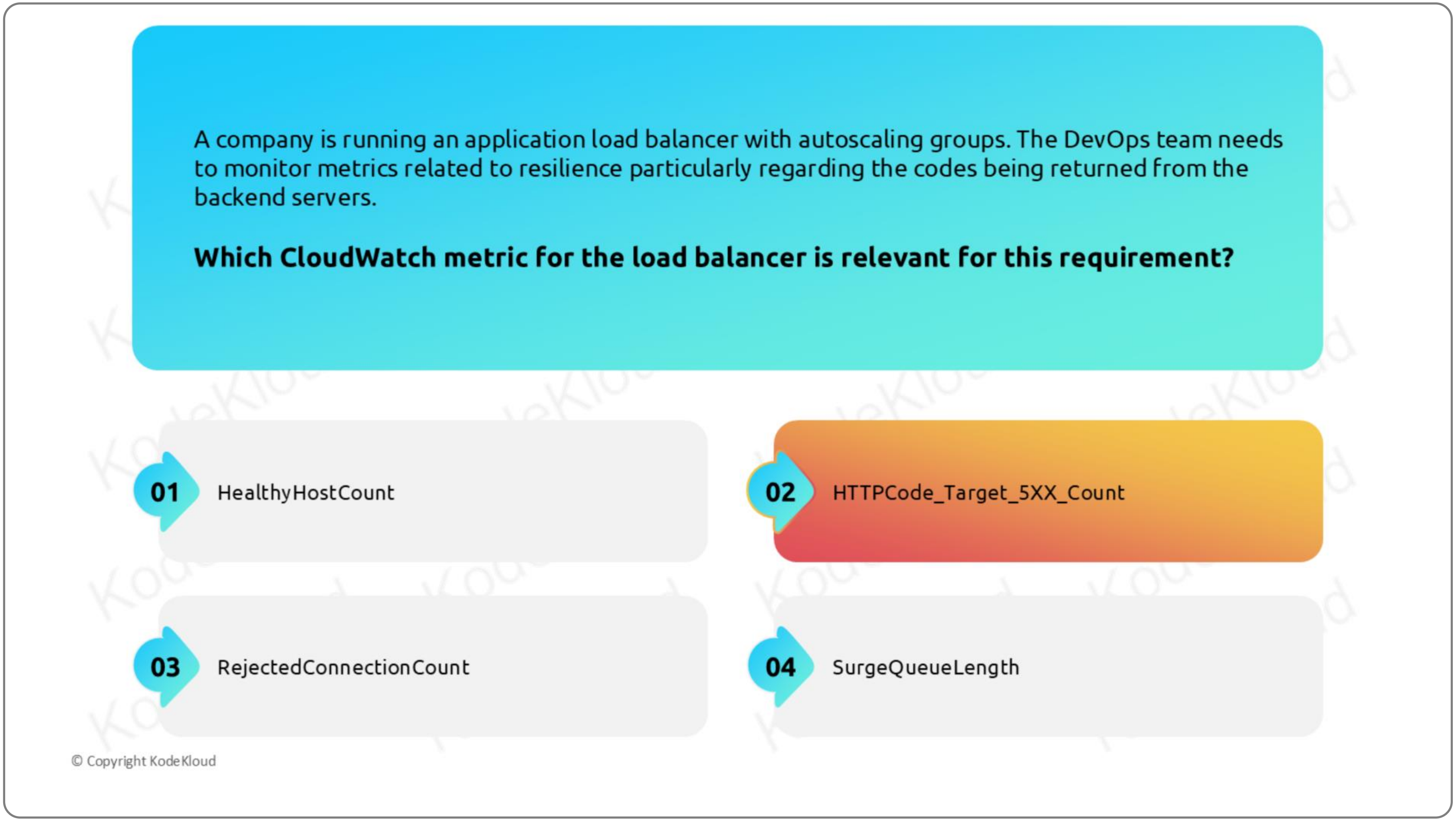
04 By configuring cross-zone load balancing

Correct Answer: A
Explanation: Integrating the ALB with Auto Scaling allows failed instances to automatically be replaced, maintaining capacity.
Incorrect Answers:
B) Sticky sessions increase affinity but do not replace failed instances.
C) Multiple target groups segment traffic but do not ensure capacity.

D) Cross-zone load balancing provides AZ resiliency but does not replace instances.
References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/autoscaling-load-balancer.html

Application Load Balancer





A company is running an application load balancer with autoscaling groups. The DevOps team needs to monitor metrics related to resilience particularly regarding the codes being returned from the backend servers.

Which CloudWatch metric for the load balancer is relevant for this requirement?

01 HealthyHostCount

02 HTTPCode_Target_5XX_Count

03 RejectedConnectionCount

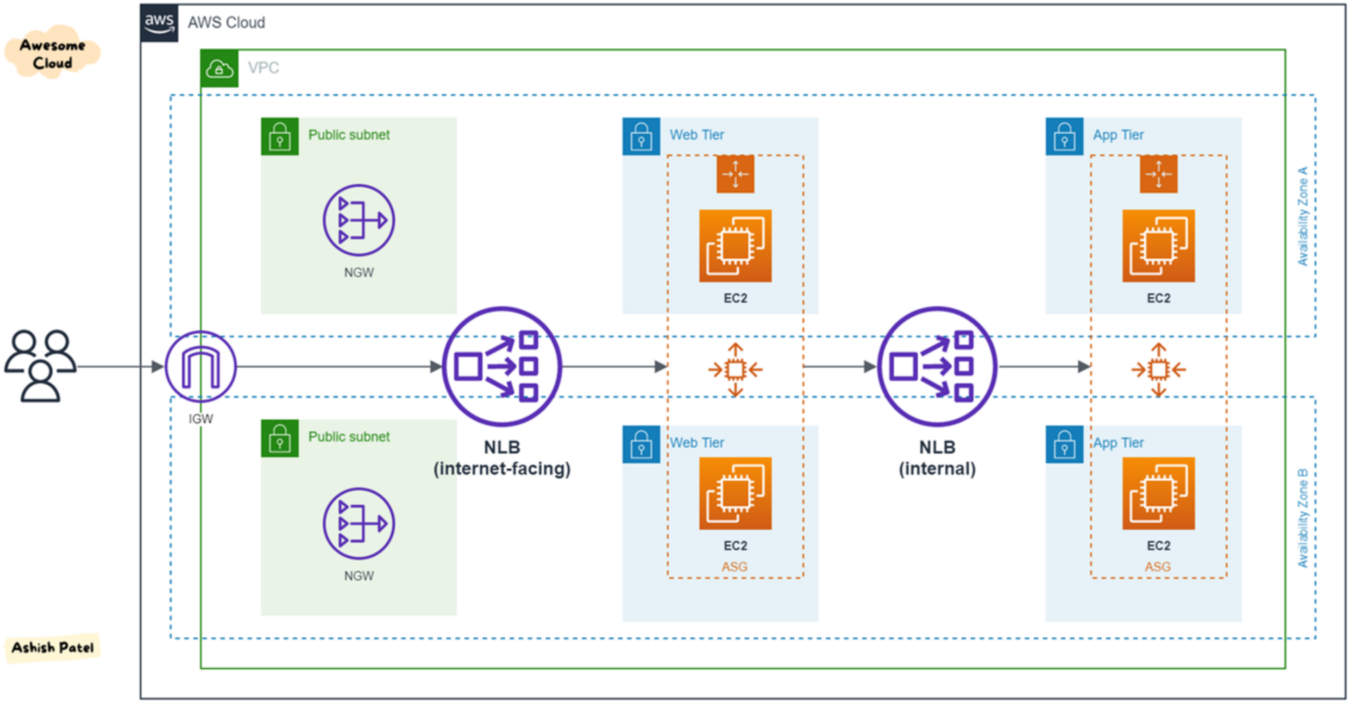
04 SurgeQueueLength

Correct Answer: B
Explanation: HTTPCode_Target_5XX_Count monitors 5XX errors from targets which helps identify issues reaching the backend instances.
Incorrect Answers:
A) HealthyHostCount provides insight into instance health but not direct backend reachability.
C) RejectedConnectionCount monitors client errors, not backend target errors.

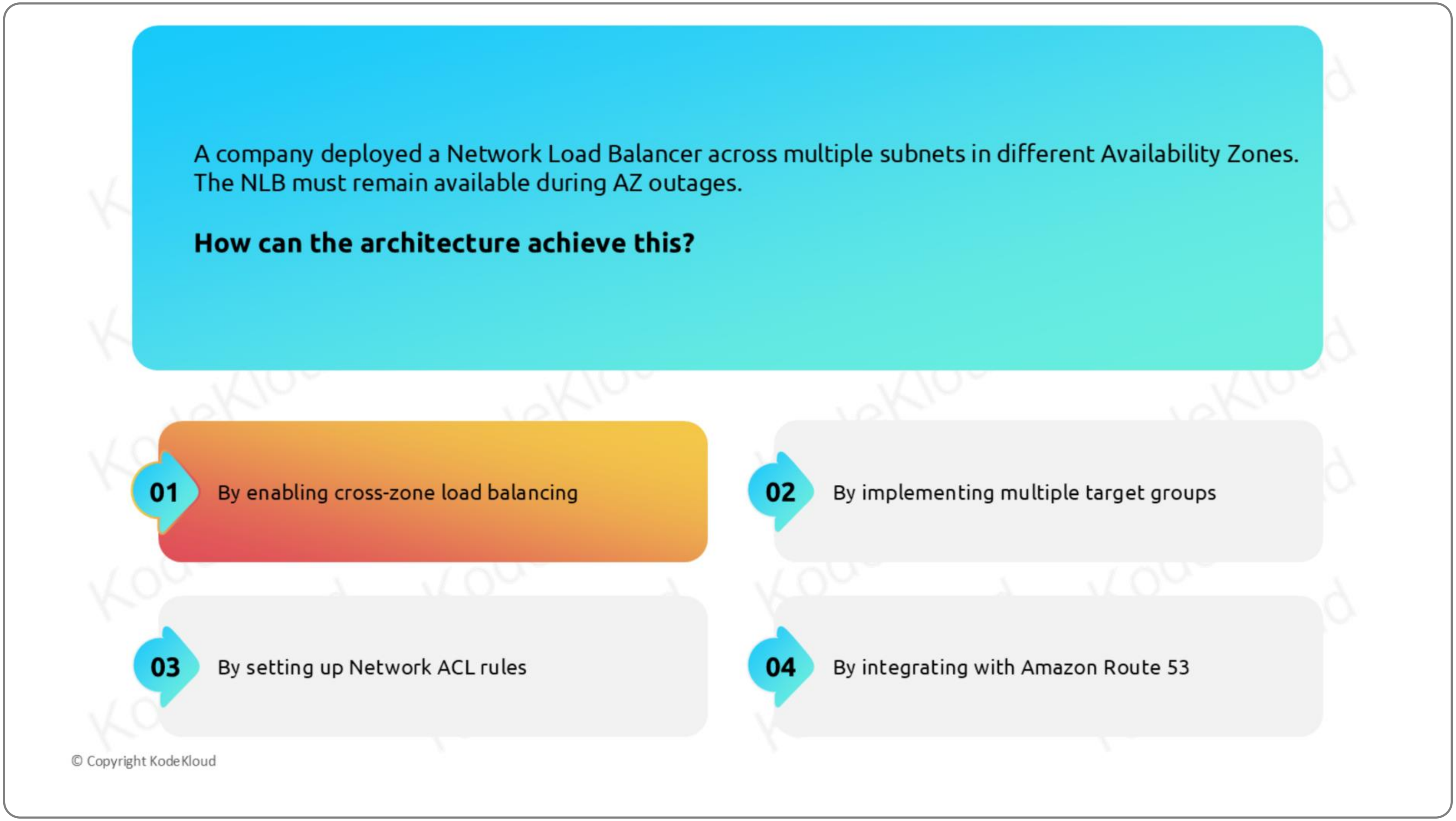
D) SurgeQueueLength monitors traffic backlog but does not indicate backend issues.
References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-cloudwatch-metrics.html

Network Load Balancer

Network Load Balancer



NLBs are like ALBs, but more simple. Reliability is about the same.



A company deployed a Network Load Balancer across multiple subnets in different Availability Zones. The NLB must remain available during AZ outages.

How can the architecture achieve this?

01 By enabling cross-zone load balancing

02 By implementing multiple target groups

03 By setting up Network ACL rules

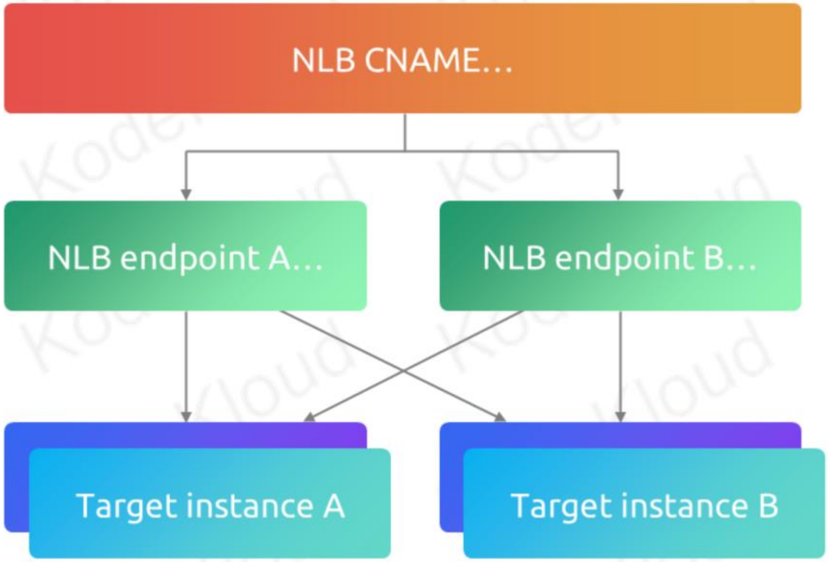
04 By integrating with Amazon Route 53

Correct Answer: A
Explanation: Enabling cross-zone load balancing for the NLB allows it to remain available and route traffic across the remaining healthy AZs during an AZ outage.
Incorrect Answers:
B) Multiple target groups help segment traffic but do not provide AZ redundancy.
C) Network ACLs control VPC traffic flows but do not provide NLB redundancy.

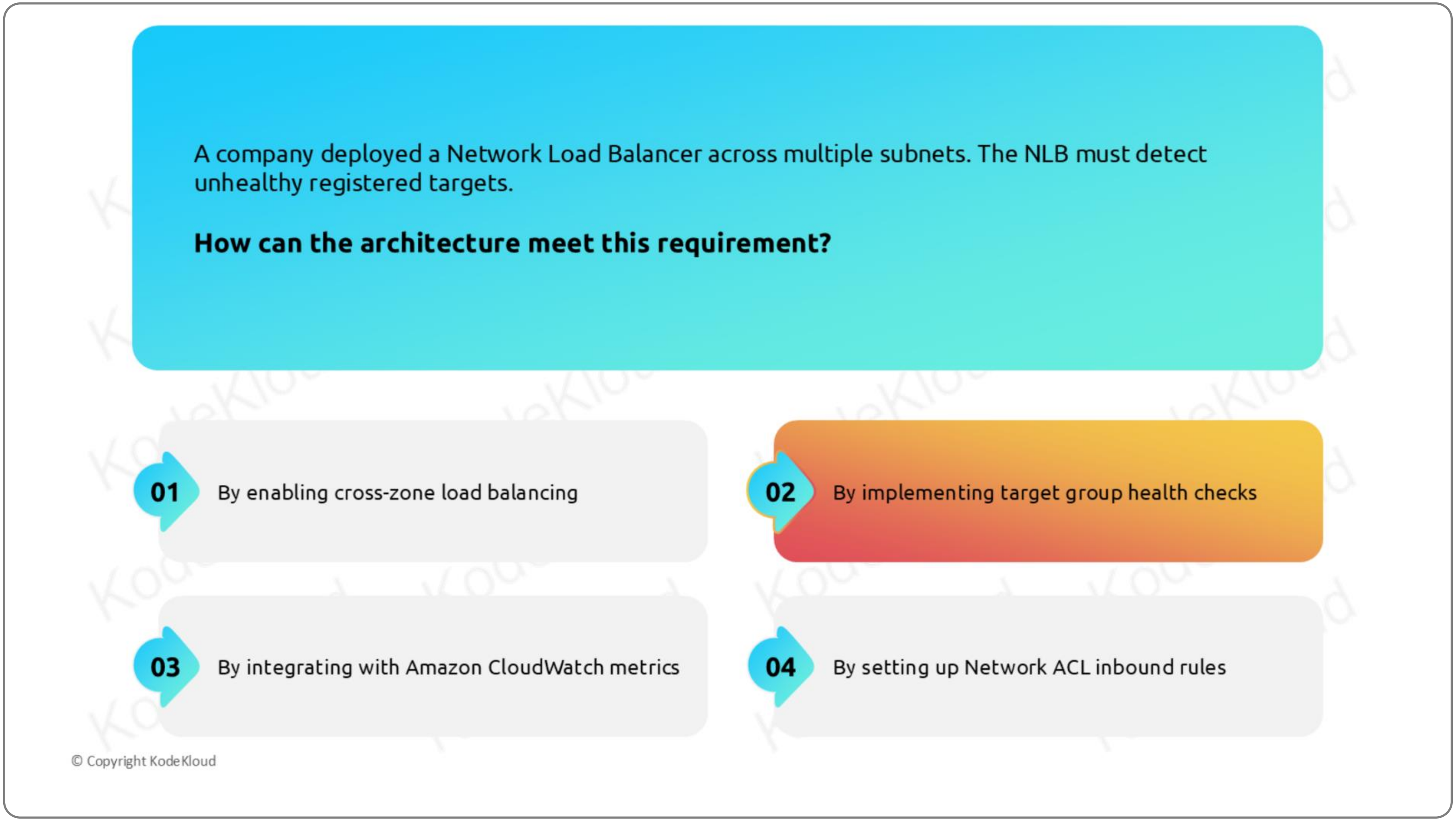
D) Amazon Route 53 can route traffic but does not provide the NLB redundancy across AZs.
References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/network/load-balancer-subnets.html

Network Load Balancer

Cross-zone load balancing enabled



Viewer does not support full SVG 1.1



A company deployed a Network Load Balancer across multiple subnets. The NLB must detect unhealthy registered targets.

How can the architecture meet this requirement?

01 By enabling cross-zone load balancing

02 By implementing target group health checks

03 By integrating with Amazon CloudWatch metrics

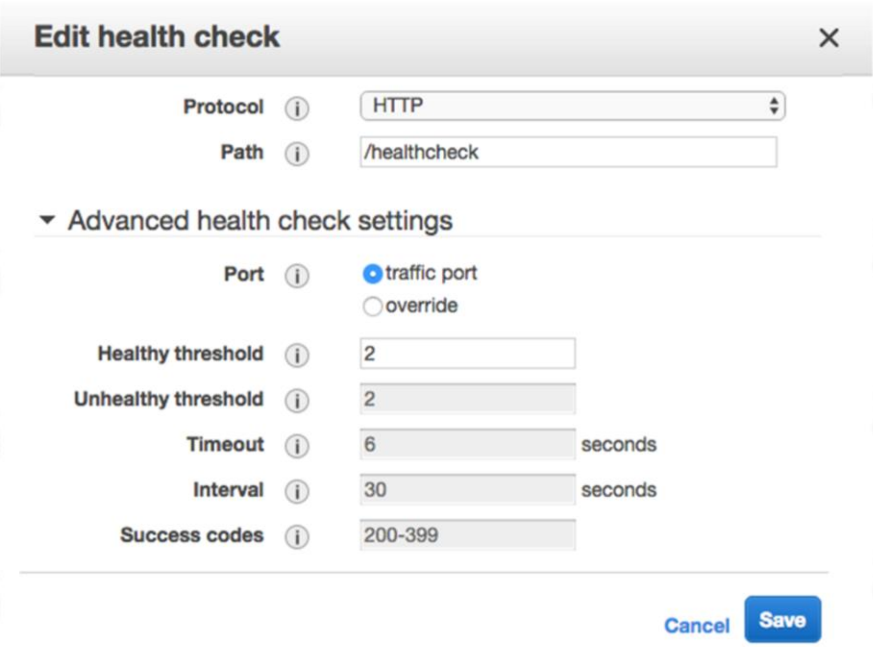
04 By setting up Network ACL inbound rules

Correct Answer: B
Explanation: NLB target group health checks can automatically detect and stop routing traffic to unhealthy registered targets.
Incorrect Answers:
A) Cross-zone load balancing provides HA across AZs but does not check target health.
C) CloudWatch metrics provide monitoring but do not check target health.

D) Network ACLs control VPC traffic but do not check target health.
References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/network/target-group-health-checks.html

Network Load Balancer

Health checks are applicable here.



Edit health check

Protocol: HTTP

Path: /healthcheck

▾ Advanced health check settings

Port: ● traffic port ○ override

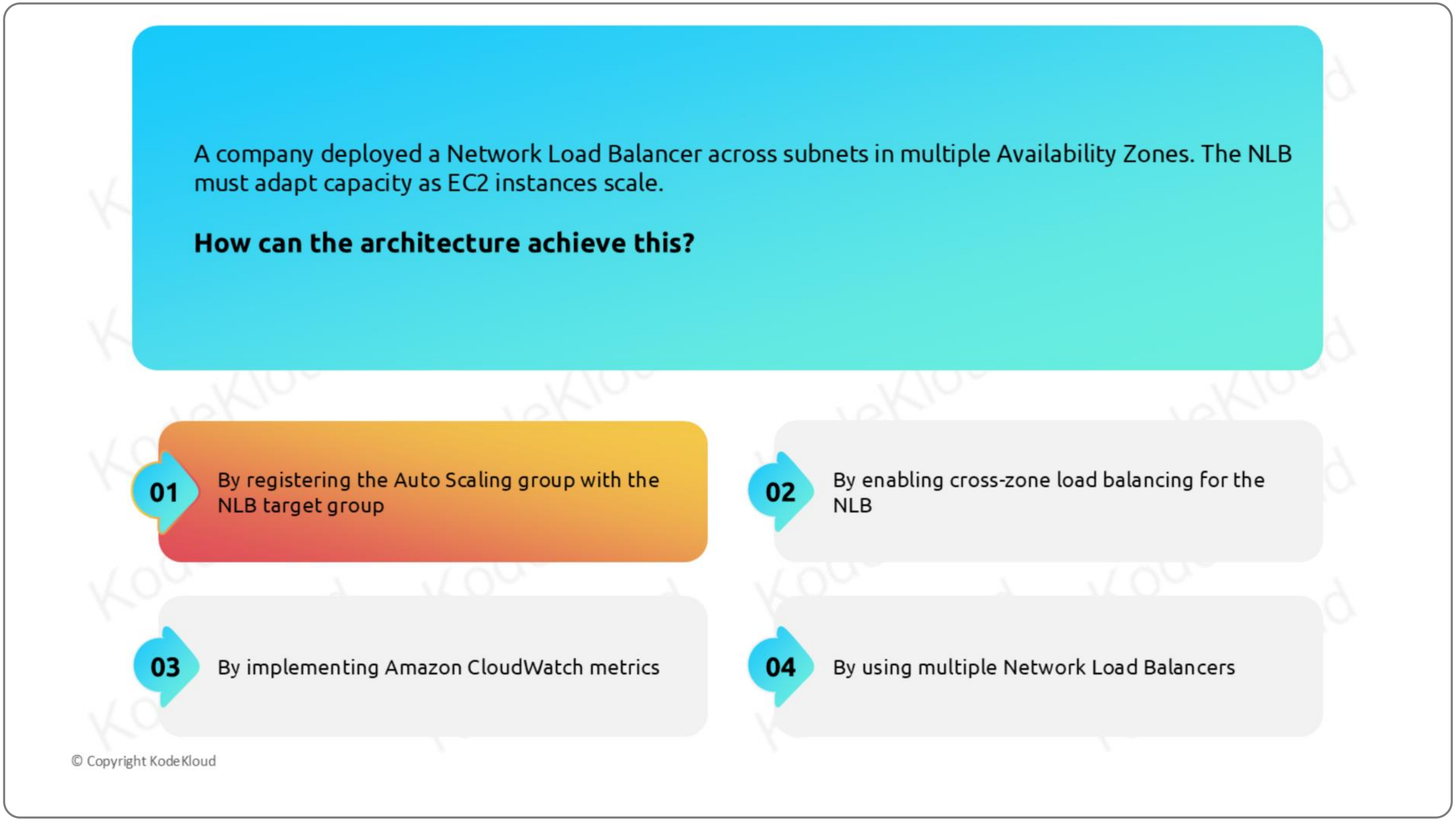
Healthy threshold: 2

Unhealthy threshold: 2

Timeout: 6 seconds

Interval: 30 seconds

Success codes: 200-399



A company deployed a Network Load Balancer across subnets in multiple Availability Zones. The NLB must adapt capacity as EC2 instances scale.

How can the architecture achieve this?

01 By registering the Auto Scaling group with the NLB target group

02 By enabling cross-zone load balancing for the NLB

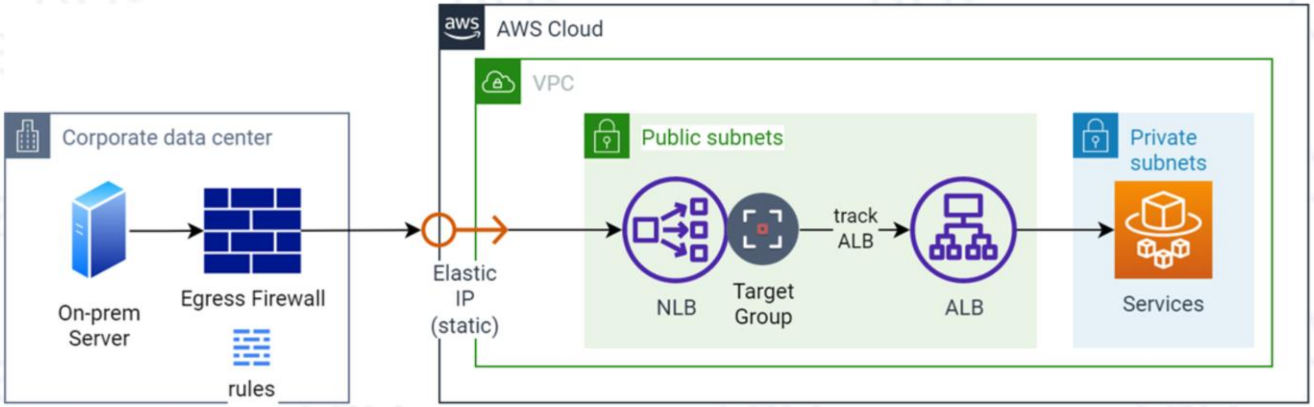
03 By implementing Amazon CloudWatch metrics

04 By using multiple Network Load Balancers

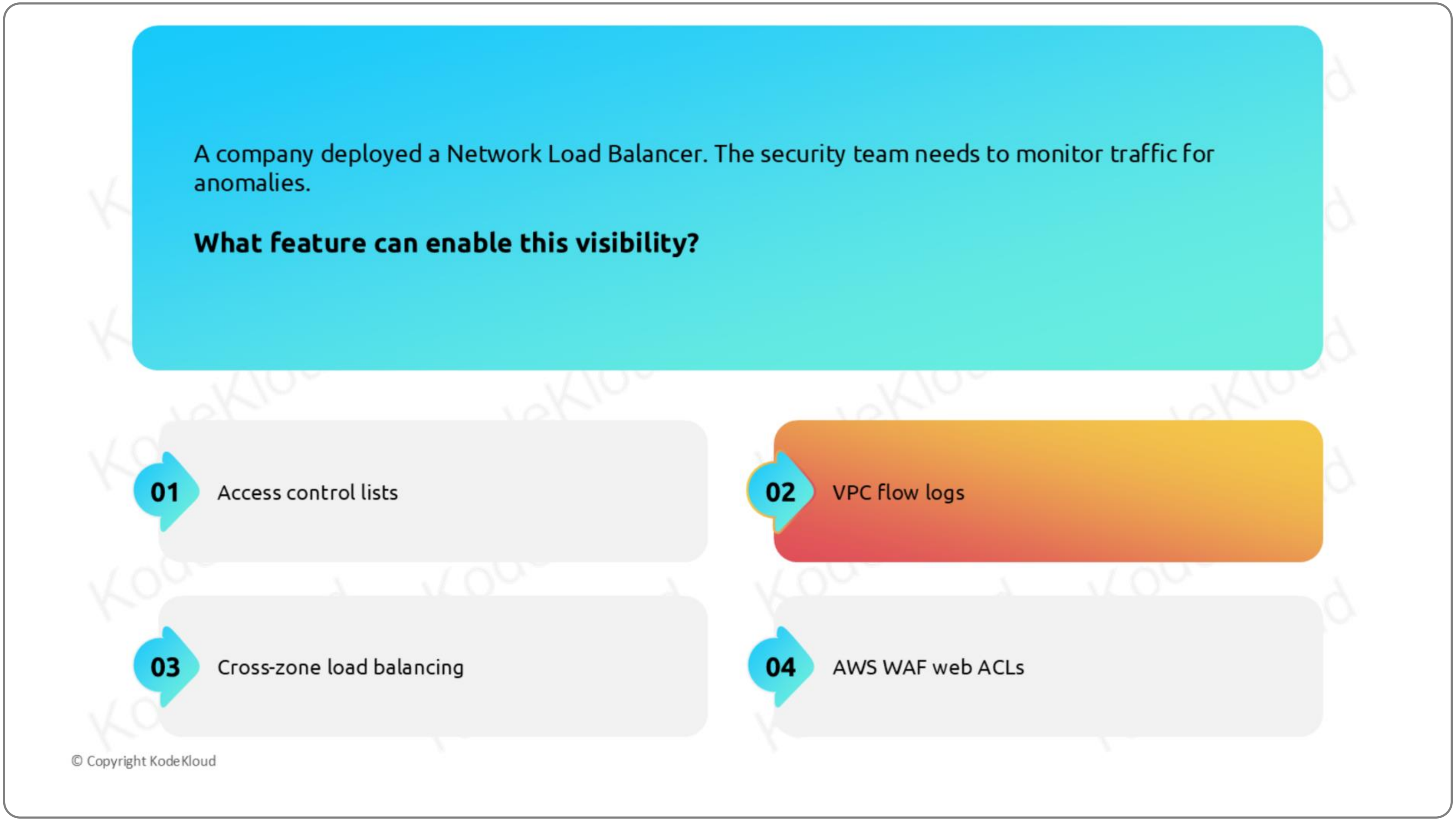
Correct Answer: A
Explanation: Registering the ASG with the NLB target group allows the load balancer to automatically scale up and down with the Auto Scaling group.
Incorrect Answers:
B) Cross-zone load balancing provides AZ redundancy but does not integrate with ASGs.
C) CloudWatch metrics enable monitoring but do not integrate the ASG with the NLB.

D) Using multiple NLBs can increase capacity but does not integrate with ASGs.
References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/autoscaling-load-balancer.html

Network Load Balancer



They support IP addresses per Zone for static usage.



A company deployed a Network Load Balancer. The security team needs to monitor traffic for anomalies.

What feature can enable this visibility?

01 Access control lists

02 VPC flow logs

03 Cross-zone load balancing

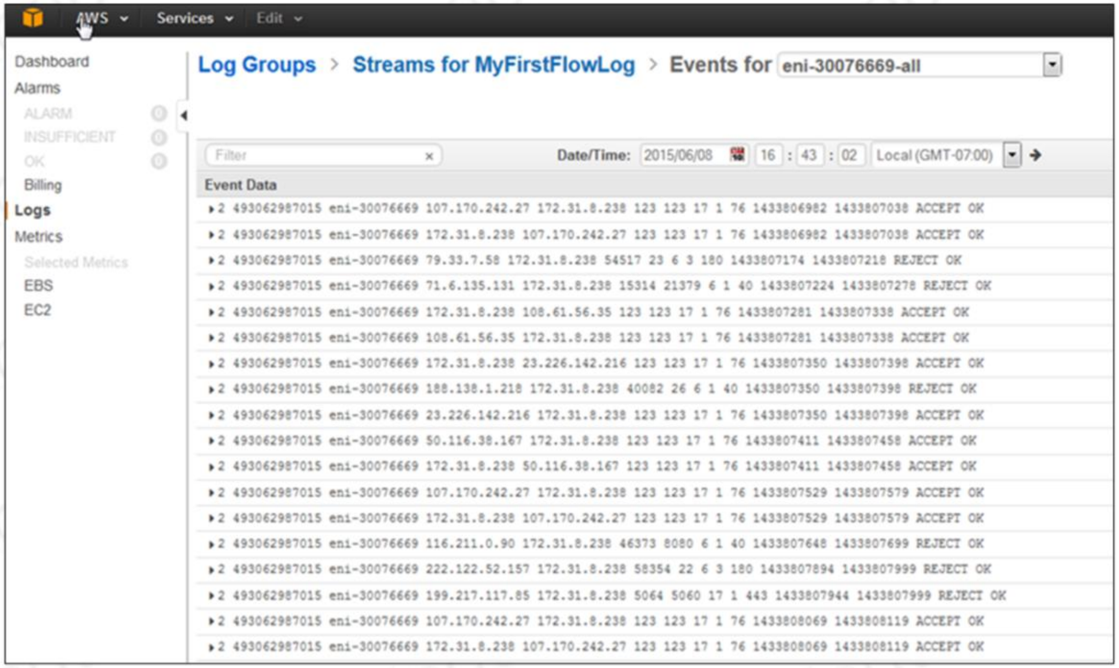
04 AWS WAF web ACLs

Correct Answer: B
Explanation: Enabling VPC Flow Logs on the NLB subnets allows logging IP traffic metadata for monitoring and analysis.
Incorrect Answers:
A) ACLs control traffic flows but do not provide logs.
C) Cross-zone load balancing provides HA across AZs but does not log traffic.
D) WAF ACLs inspect web requests but do not log at the VPC networking layer.

References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/network/load-balancer-cloudwatch-metrics.html

Network Load Balancer

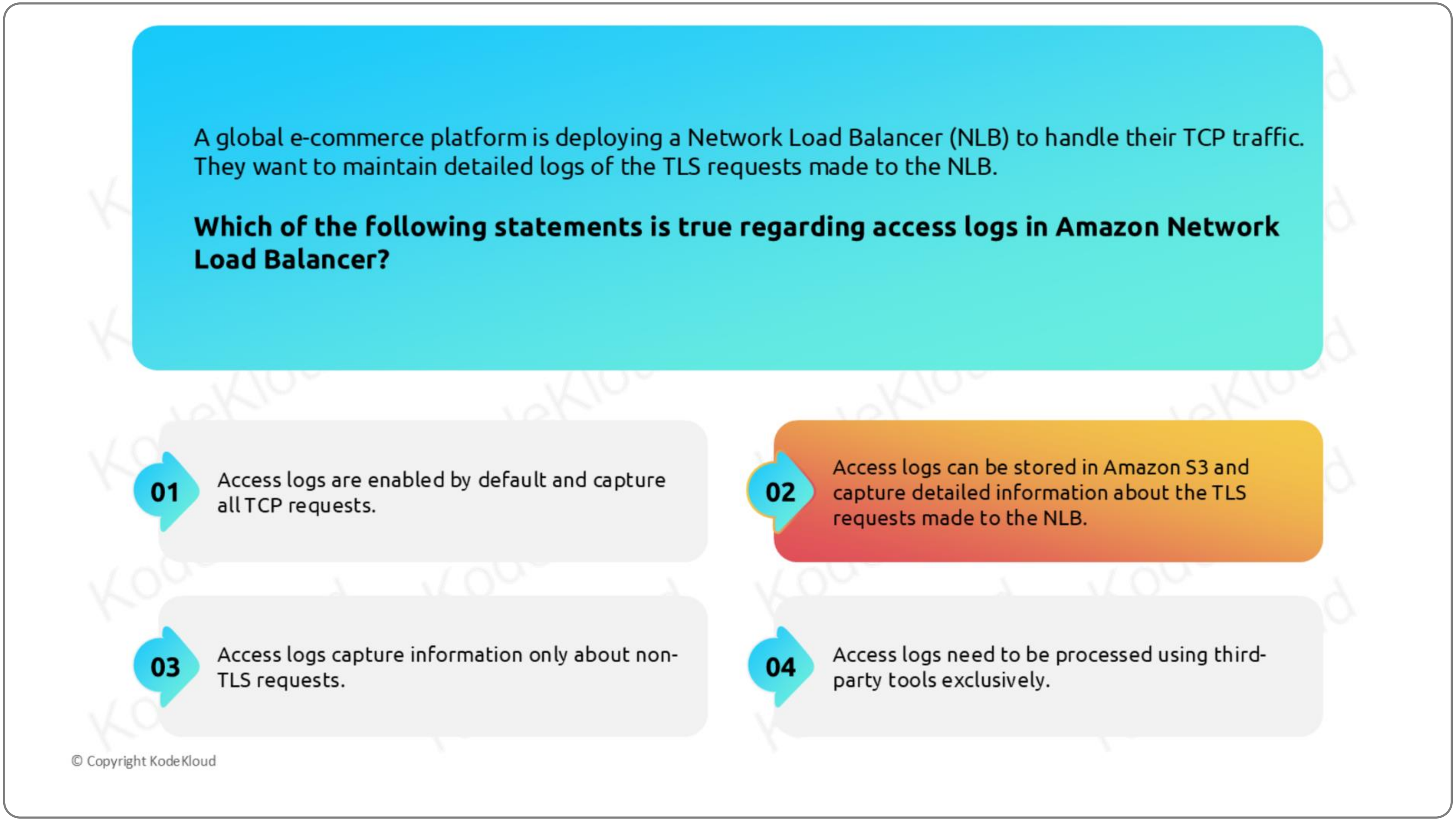
VPC Flow logs will capture traffic for NLBS as well.



Log Groups > Streams for MyFirstFlowLog > Events for eni-30076669-all

Date/Time: 2015/06/08 16 : 43 : 02 Local (GMT-07:00)

```
Event Data
2 493062987015 eni-30076669 107.170.242.27 172.31.8.238 123 123 17 1 76 1433806982 1433807038 ACCEPT OK
2 493062987015 eni-30076669 172.31.8.238 107.170.242.27 123 123 17 1 76 1433806982 1433807038 ACCEPT OK
2 493062987015 eni-30076669 79.33.7.58 172.31.8.238 54517 23 6 3 180 1433807174 1433807218 REJECT OK
2 493062987015 eni-30076669 71.6.135.131 172.31.8.238 15314 21379 6 1 40 1433807224 1433807278 REJECT OK
2 493062987015 eni-30076669 172.31.8.238 108.61.56.35 123 123 17 1 76 1433807281 1433807338 ACCEPT OK
2 493062987015 eni-30076669 108.61.56.35 172.31.8.238 123 123 17 1 76 1433807281 1433807338 ACCEPT OK
2 493062987015 eni-30076669 172.31.8.238 23.226.142.216 123 123 17 1 76 1433807350 1433807398 ACCEPT OK
2 493062987015 eni-30076669 188.138.1.218 172.31.8.238 40082 26 6 1 40 1433807350 1433807398 REJECT OK
2 493062987015 eni-30076669 23.226.142.216 172.31.8.238 123 123 17 1 76 1433807350 1433807398 ACCEPT OK
2 493062987015 eni-30076669 50.116.38.167 172.31.8.238 123 123 17 1 76 1433807411 1433807458 ACCEPT OK
2 493062987015 eni-30076669 172.31.8.238 50.116.38.167 123 123 17 1 76 1433807411 1433807458 ACCEPT OK
2 493062987015 eni-30076669 107.170.242.27 172.31.8.238 123 123 17 1 76 1433807529 1433807579 ACCEPT OK
2 493062987015 eni-30076669 172.31.8.238 107.170.242.27 123 123 17 1 76 1433807529 1433807579 ACCEPT OK
2 493062987015 eni-30076669 116.211.0.90 172.31.8.238 46373 8080 6 1 40 1433807648 1433807699 REJECT OK
2 493062987015 eni-30076669 222.122.52.157 172.31.8.238 58354 22 6 3 180 1433807894 1433807999 REJECT OK
2 493062987015 eni-30076669 199.217.117.85 172.31.8.238 5064 5060 17 1 443 1433807944 1433807999 REJECT OK
2 493062987015 eni-30076669 107.170.242.27 172.31.8.238 123 123 17 1 76 1433808069 1433808119 ACCEPT OK
2 493062987015 eni-30076669 172.31.8.238 107.170.242.27 123 123 17 1 76 1433808069 1433808119 ACCEPT OK
```



A global e-commerce platform is deploying a Network Load Balancer (NLB) to handle their TCP traffic. They want to maintain detailed logs of the TLS requests made to the NLB.

Which of the following statements is true regarding access logs in Amazon Network Load Balancer?

01 Access logs are enabled by default and capture all TCP requests.

02 Access logs can be stored in Amazon S3 and capture detailed information about the TLS requests made to the NLB.

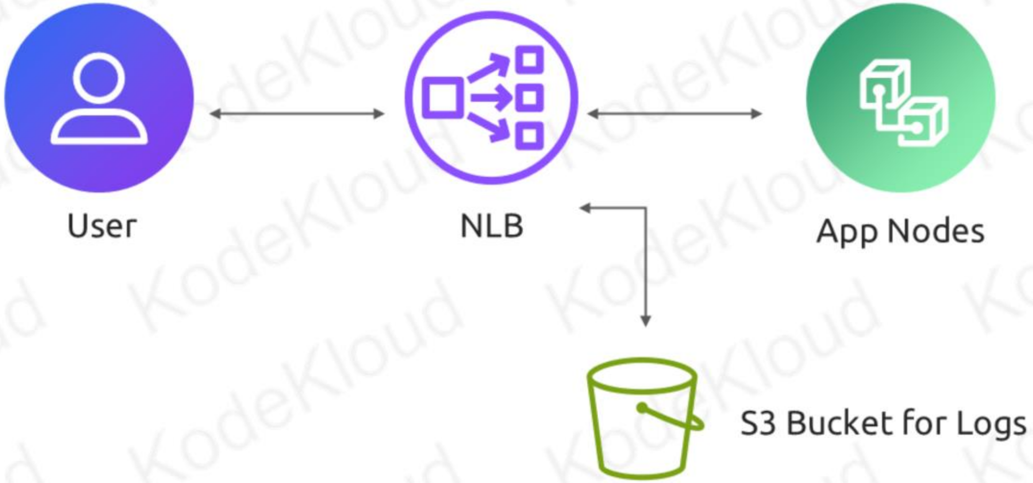
03 Access logs capture information only about non-TLS requests.

04 Access logs need to be processed using third-party tools exclusively.

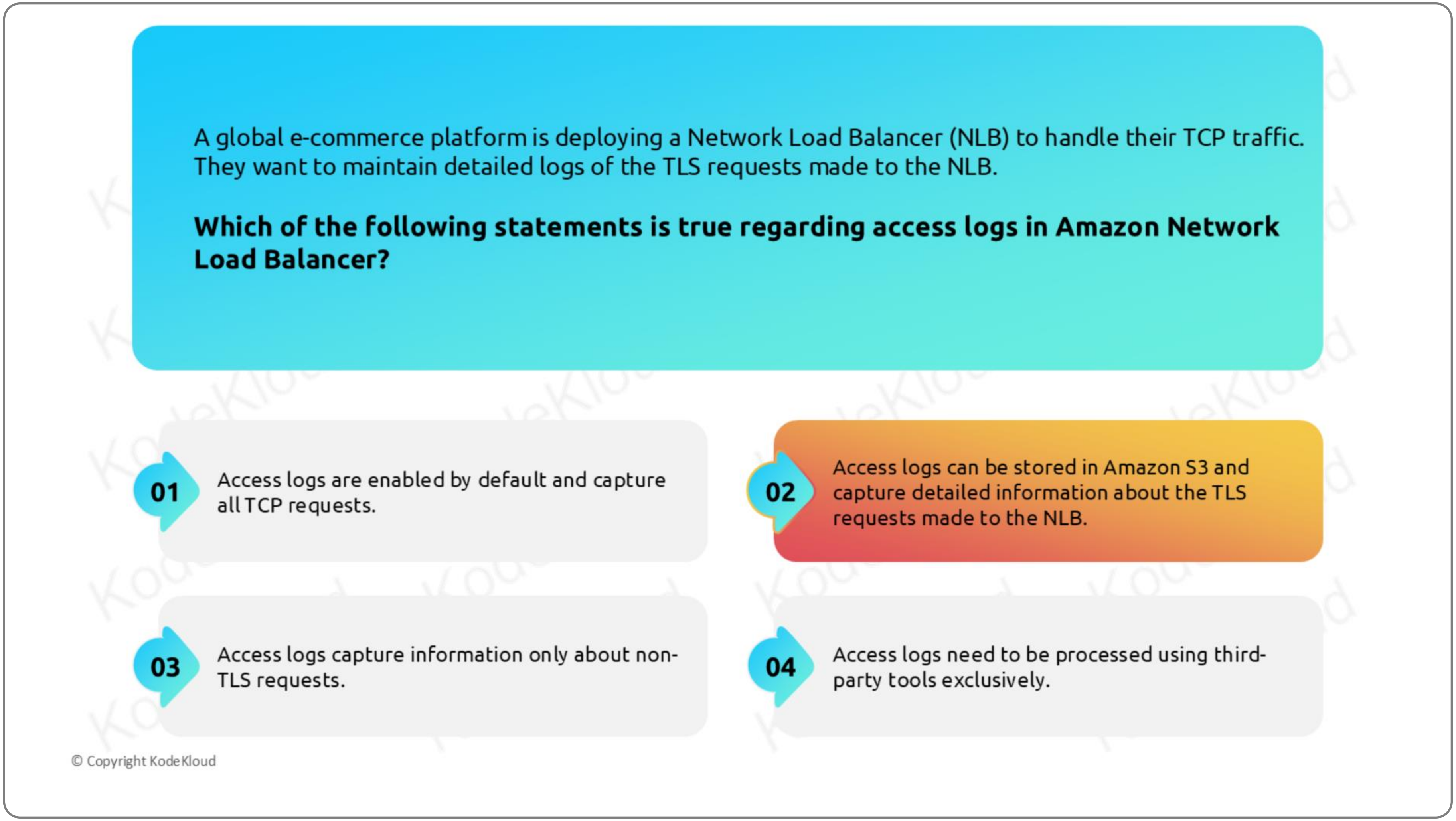
Correct Answer: B. Access logs can be stored in Amazon S3 and capture detailed information about the TLS requests made to the NLB.
Incorrect: A. Access logs are not enabled by default. C. Access logs are created only if the load balancer has a TLS listener and they contain information only about TLS requests. D. While third-party tools can be used to process access logs, AWS also offers tools like Amazon Athena for analyzing the logs.
Explanation: Amazon Network Load Balancer provides access logs that capture detailed information about the TLS

requests sent to the NLB. These logs can be stored in Amazon S3 for further analysis. Access logs are particularly useful for analyzing traffic patterns and troubleshooting issues. It's important to note that access logs are created only if the NLB has a TLS listener.
References: Amazon Elastic Load Balancing - Network Load Balancers

Network Load Balancer



Also, NLBs have the standard metrics and logging, but also ACCESS LOGS for TLS listeners only.



A global e-commerce platform is deploying a Network Load Balancer (NLB) to handle their TCP traffic. They want to maintain detailed logs of the TLS requests made to the NLB.

Which of the following statements is true regarding access logs in Amazon Network Load Balancer?

01 Access logs are enabled by default and capture all TCP requests.

02 Access logs can be stored in Amazon S3 and capture detailed information about the TLS requests made to the NLB.

03 Access logs capture information only about non-TLS requests.

04 Access logs need to be processed using third-party tools exclusively.

Correct Answer: B. Access logs can be stored in Amazon S3 and capture detailed information about the TLS requests made to the NLB.
Incorrect: A. Access logs are not enabled by default. C. Access logs are created only if the load balancer has a TLS listener and they contain information only about TLS requests. D. While third-party tools can be used to process access logs, AWS also offers tools like Amazon Athena for analyzing the logs.
Explanation: Amazon Network Load Balancer provides access logs that capture detailed information about the TLS

requests sent to the NLB. These logs can be stored in Amazon S3 for further analysis. Access logs are particularly useful for analyzing traffic patterns and troubleshooting issues. It's important to note that access logs are created only if the NLB has a TLS listener.
References: Amazon Elastic Load Balancing - Network Load Balancers

Gateway Load Balancer

The key and important reliability features of Elastic Load Balancing Gateway Load Balancers include:
Automatic Scaling: Gateway Load Balancers can automatically scale to handle varying levels of incoming traffic. This ensures that the load balancer can handle increased traffic without any manual intervention.
Health Monitoring: Gateway Load Balancers continuously monitor the health of the registered targets. They only route traffic to the healthy targets, ensuring that any unhealthy targets are automatically removed from the load balancing rotation.

Fault Tolerance: Gateway Load Balancers are designed to be highly available and fault-tolerant. They distribute traffic across multiple targets in different Availability Zones, ensuring that if one target or Availability Zone becomes unavailable, traffic is automatically routed to the remaining healthy targets.

Redundancy: Gateway Load Balancers provide redundancy by automatically distributing traffic across multiple targets. This helps to prevent any single point of failure and ensures that traffic can be handled even if one or more targets become unavailable.

Elasticity: Gateway Load Balancers can scale up or down based on the demand. They can handle increased traffic by automatically adding more targets and distributing the traffic evenly across them. Similarly, they can scale down when the traffic decreases, reducing the number of targets and optimizing resource utilization.

Load Balancing Algorithms: Gateway Load Balancers use advanced load balancing algorithms to distribute traffic evenly across the registered targets. This helps to optimize resource utilization and ensure that no single target is overwhelmed with traffic.

Connection Draining: Gateway Load Balancers support connection draining, which allows in-flight requests to complete even when a target is being deregistered or becomes unhealthy. This ensures that there is no disruption to the user experience during target changes or failures.

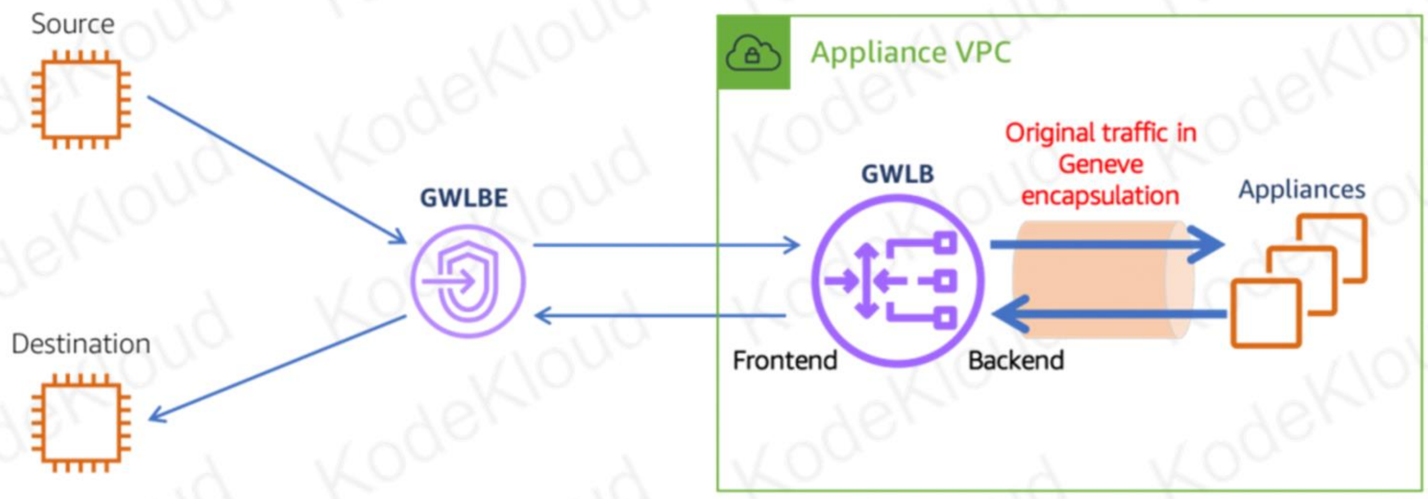
Monitoring and Logging: Gateway Load Balancers provide detailed monitoring and logging capabilities, allowing you to track the performance and health of your load balancer. This helps in troubleshooting and optimizing the load balancing configuration.

SOURCES:

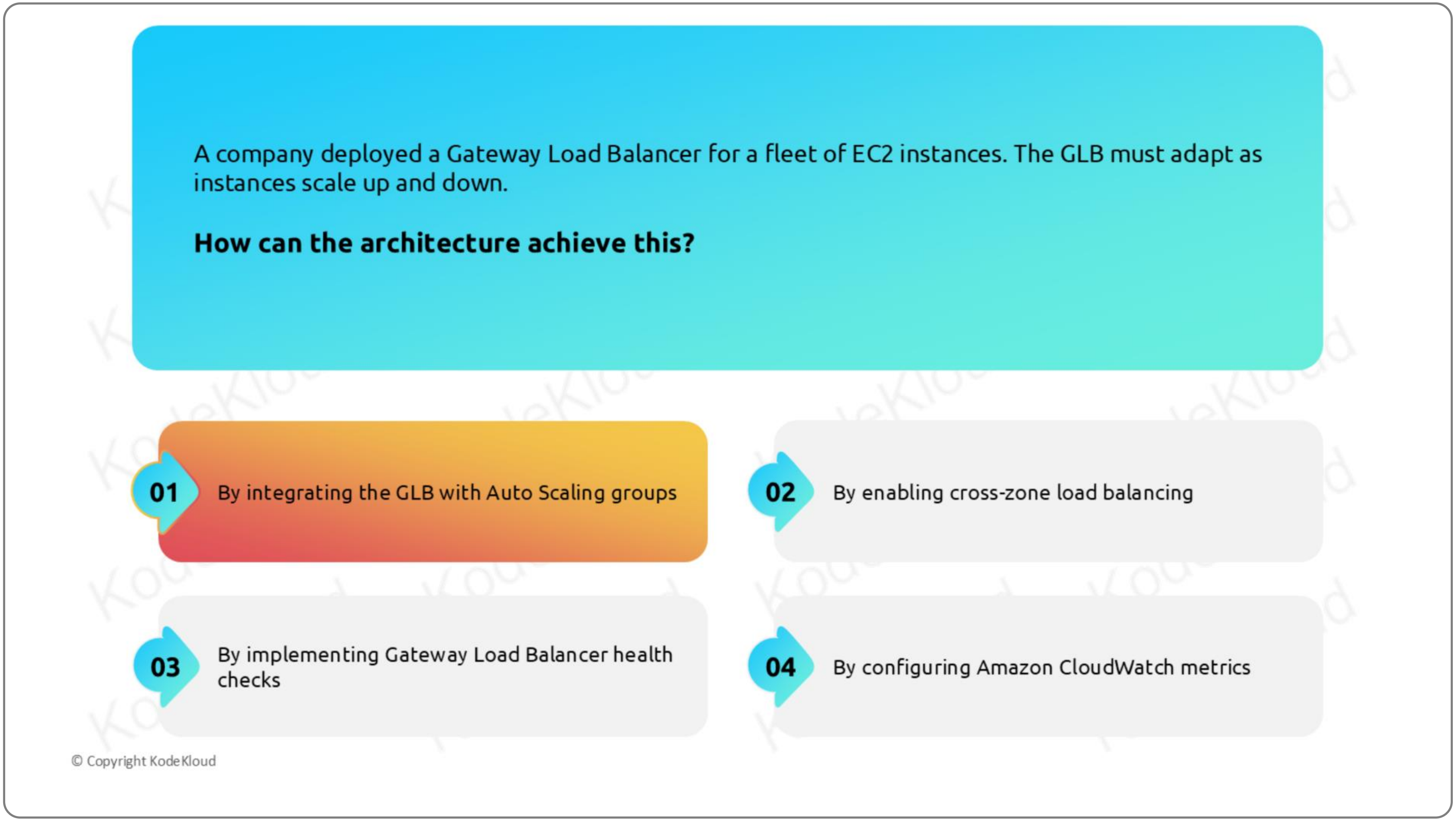
Page 1: Elastic Load Balancing Gateway Load Balancers

Page 5: Elastic Load Balancing Gateway Load Balancers

Gateway Load Balancer



Gateway load balancer is kind of an autoscaling device.



A company deployed a Gateway Load Balancer for a fleet of EC2 instances. The GLB must adapt as instances scale up and down.

How can the architecture achieve this?

01 By integrating the GLB with Auto Scaling groups

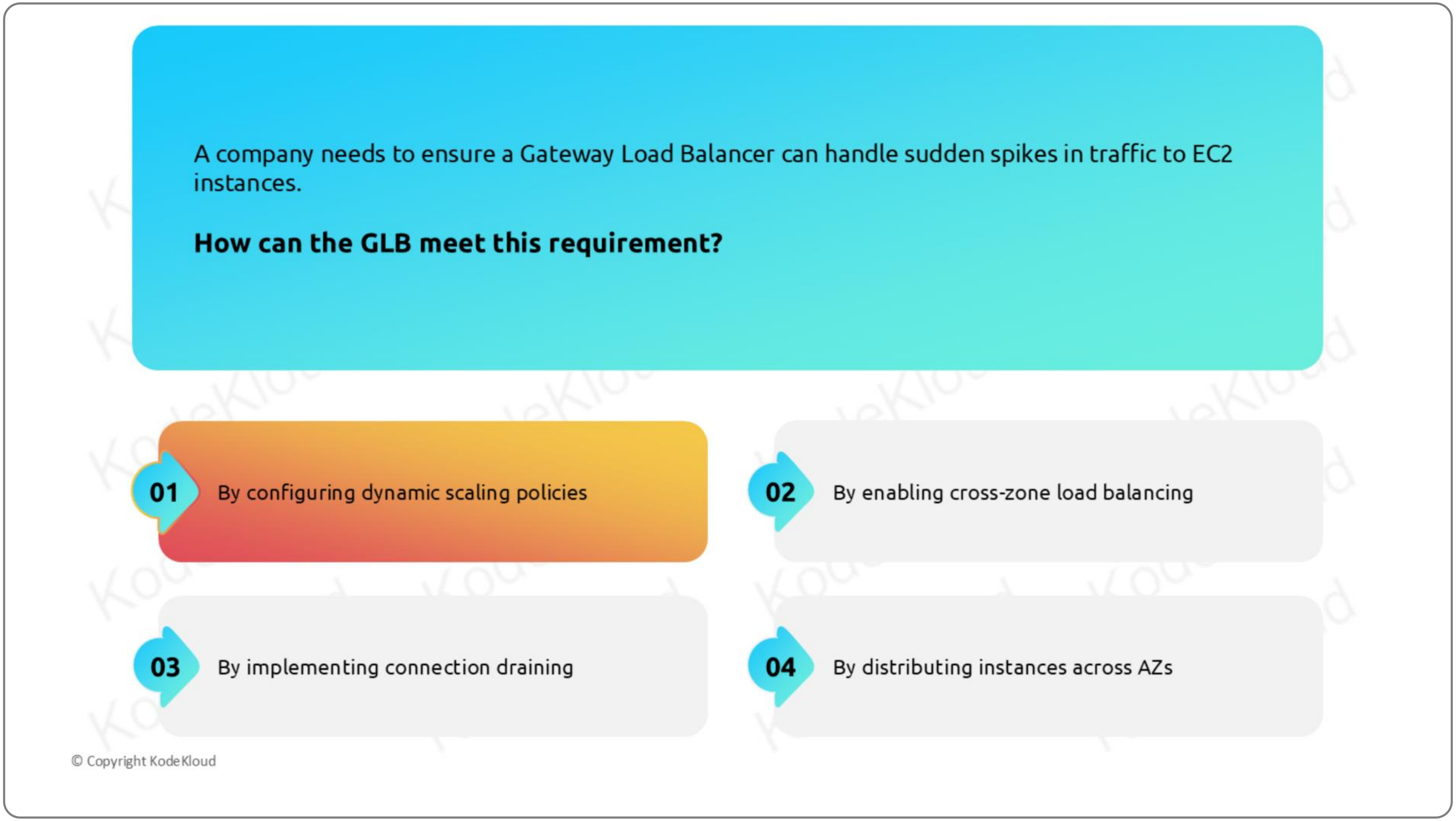
02 By enabling cross-zone load balancing

03 By implementing Gateway Load Balancer health checks

04 By configuring Amazon CloudWatch metrics

Correct Answer: A
Explanation: Integrating the Gateway Load Balancer with Auto Scaling groups allows it to automatically scale up and down as instances are added or removed.
Incorrect Answers:
B) Cross-zone load balancing provides resilience across AZs but does not integrate with Auto Scaling.
C) Health checks enable routing around unhealthy instances but do not integrate with Auto Scaling.

D) CloudWatch metrics provide visibility but do not integrate the GLB with Auto Scaling.
References:
https://docs.aws.amazon.com/autoscaling/ec2/userguide/autoscaling-load-balancer.html



A company needs to ensure a Gateway Load Balancer can handle sudden spikes in traffic to EC2 instances.

How can the GLB meet this requirement?

01 By configuring dynamic scaling policies

02 By enabling cross-zone load balancing

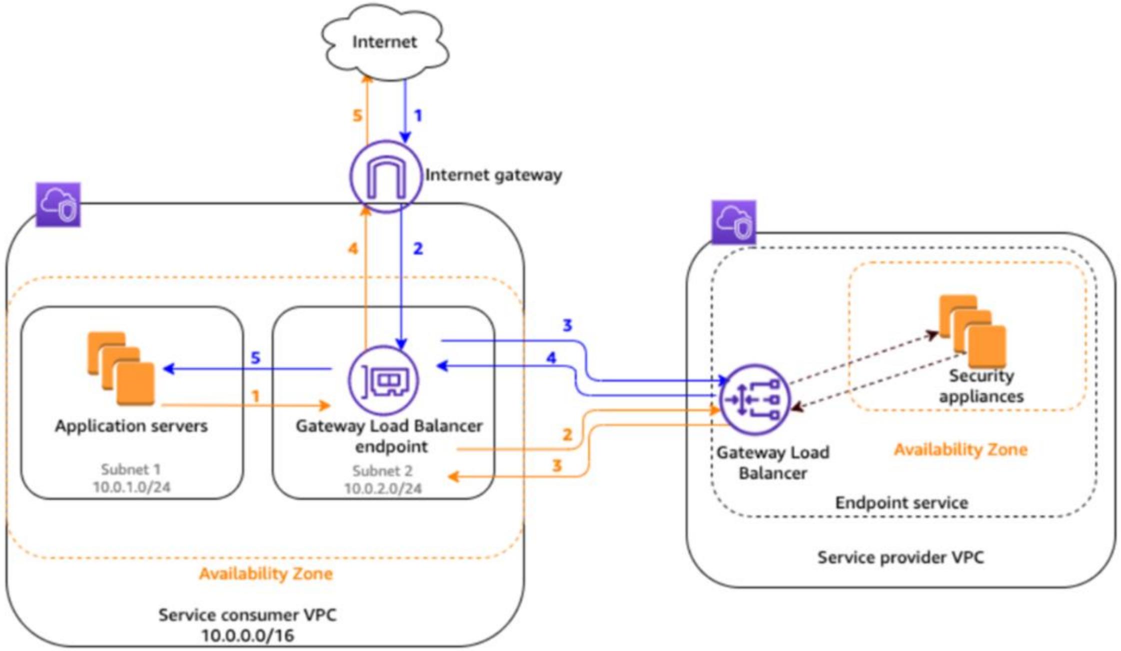
03 By implementing connection draining

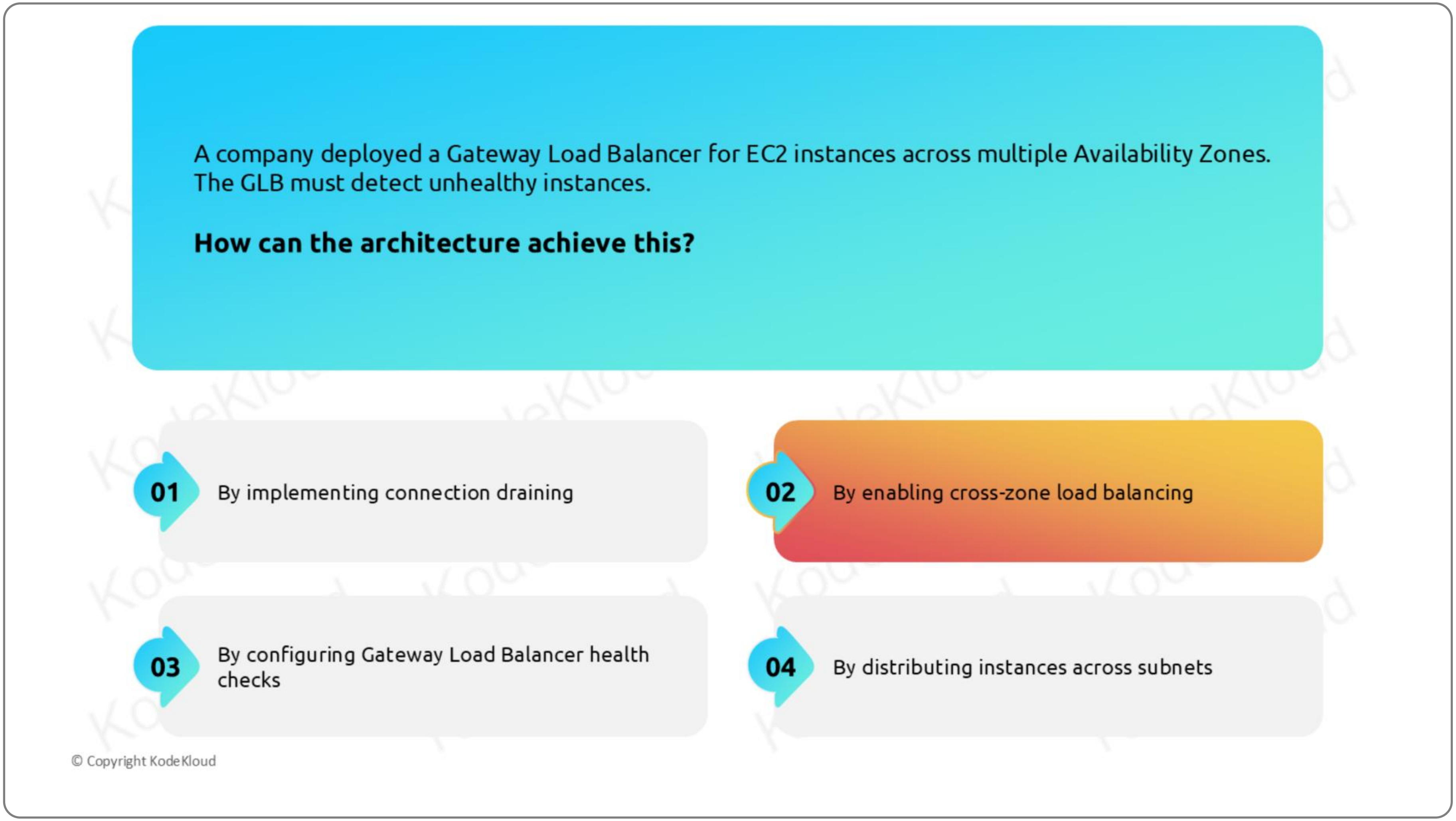
04 By distributing instances across AZs

Correct Answer: A
Explanation: Gateway Load Balancer supports dynamic scaling to automatically scale capacity up and down based on traffic to handle spikes.
Incorrect Answers:
B) Cross-zone load balancing provides AZ resiliency but does not dynamically scale.
C) Connection draining facilitates graceful instance shutdowns but does not dynamically scale.

D) Distributing instances provides high availability but does not dynamically scale capacity.
References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/gateway/dynamic-scaling.html

Gateway Load Balancer





A company deployed a Gateway Load Balancer for EC2 instances across multiple Availability Zones. The GLB must detect unhealthy instances.

How can the architecture achieve this?

01 By implementing connection draining

02 By enabling cross-zone load balancing

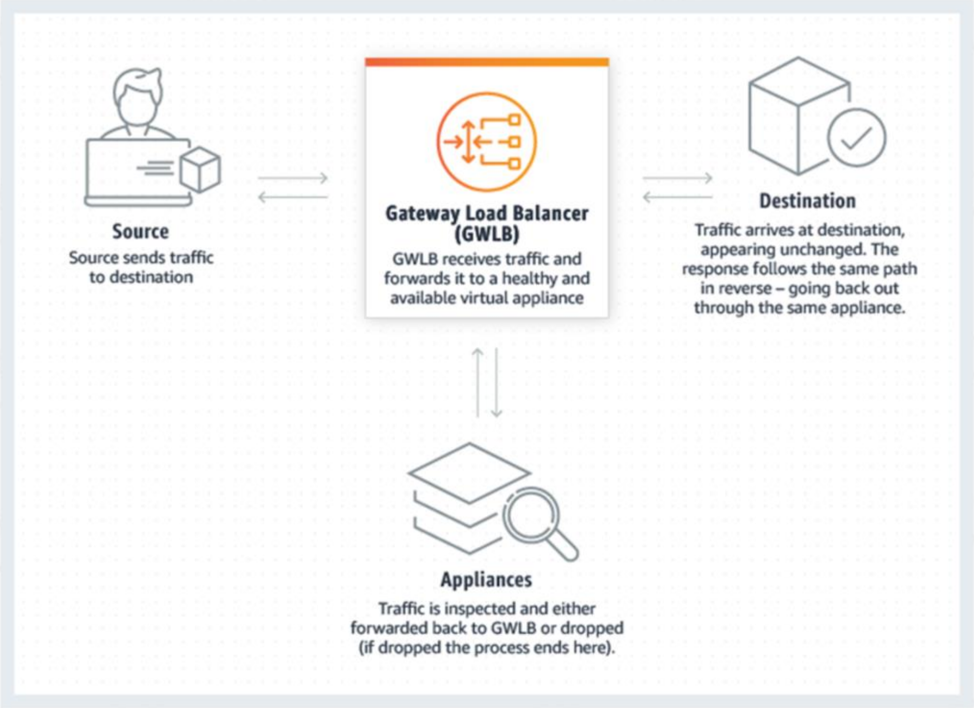
03 By configuring Gateway Load Balancer health checks

04 By distributing instances across subnets

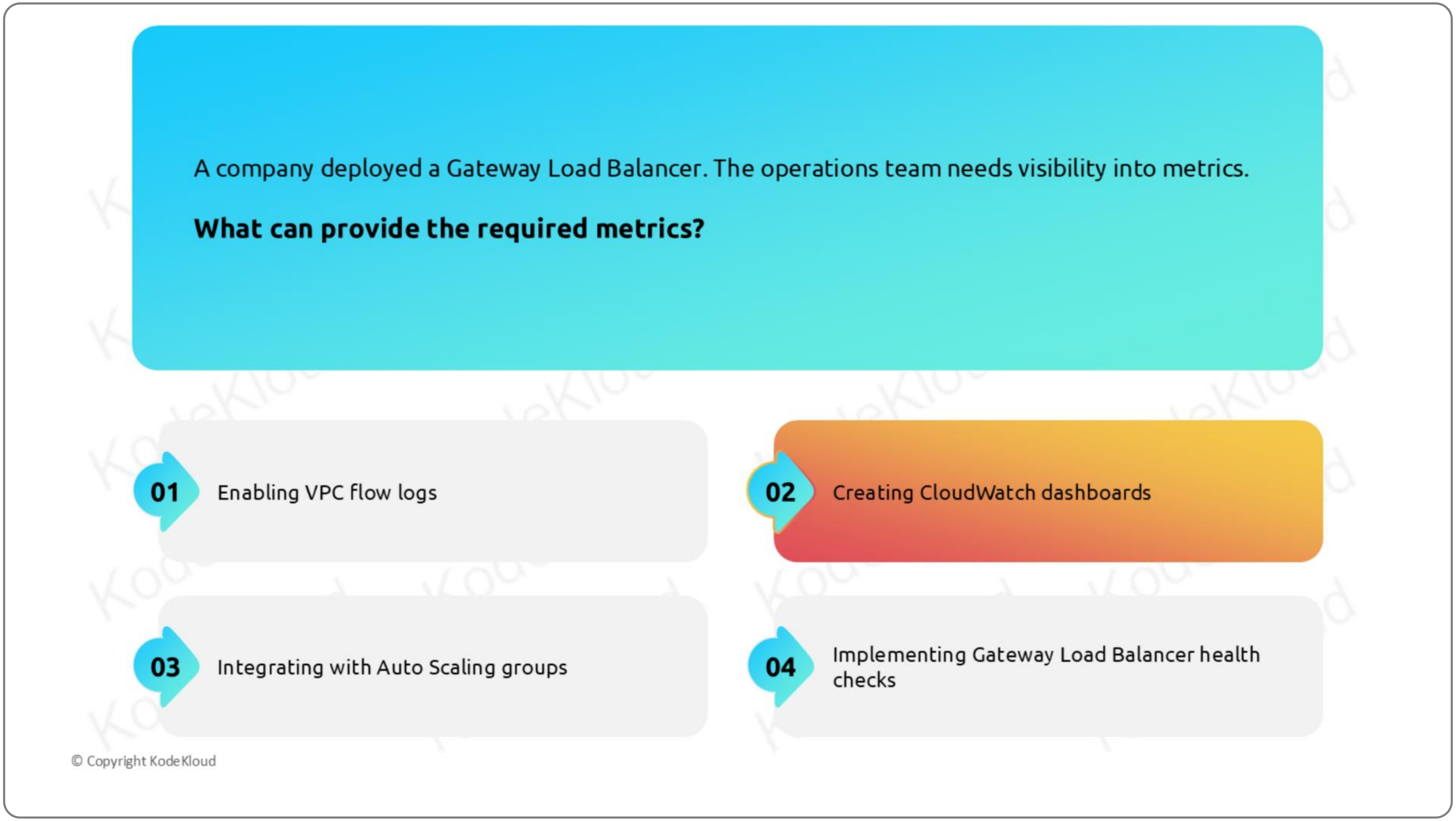
Correct Answer: B. By navigating to the GLB, choosing Actions, selecting "Allow principals," and then entering the ARNs of the service consumers that are allowed to create an endpoint to the service.
Incorrect: A. Security Groups work based on IP protocols, ports, and source/destination IP addresses or CIDR blocks, not AWS accounts or IAM roles/users. C. There isn't a direct "CreateEndpoint" IAM action for GLB. D. VPC peering allows for routing between VPCs and doesn't control who can create endpoints to a service.
Explanation: To control which AWS accounts, IAM roles, and users can create endpoints to the service behind a Gateway

Load Balancer, you can use the "Allow principals" action. By specifying the ARNs of the allowed service consumers, you can ensure that only those entities can establish an endpoint connection to your service.
References: Amazon Gateway Load Balancer - Allowing Principals

Gateway Load Balancer



Source
Source sends traffic to destination
Gateway Load Balancer (GWLB)
GWLB receives traffic and forwards it to a healthy and available virtual appliance
Destination
Traffic arrives at destination, appearing unchanged. The response follows the same path in reverse – going back out through the same appliance.
Appliances
Traffic is inspected and either forwarded back to GWLB or dropped (if dropped the process ends here).



A company deployed a Gateway Load Balancer. The operations team needs visibility into metrics.

What can provide the required metrics?

01 Enabling VPC flow logs

02 Creating CloudWatch dashboards

03 Integrating with Auto Scaling groups

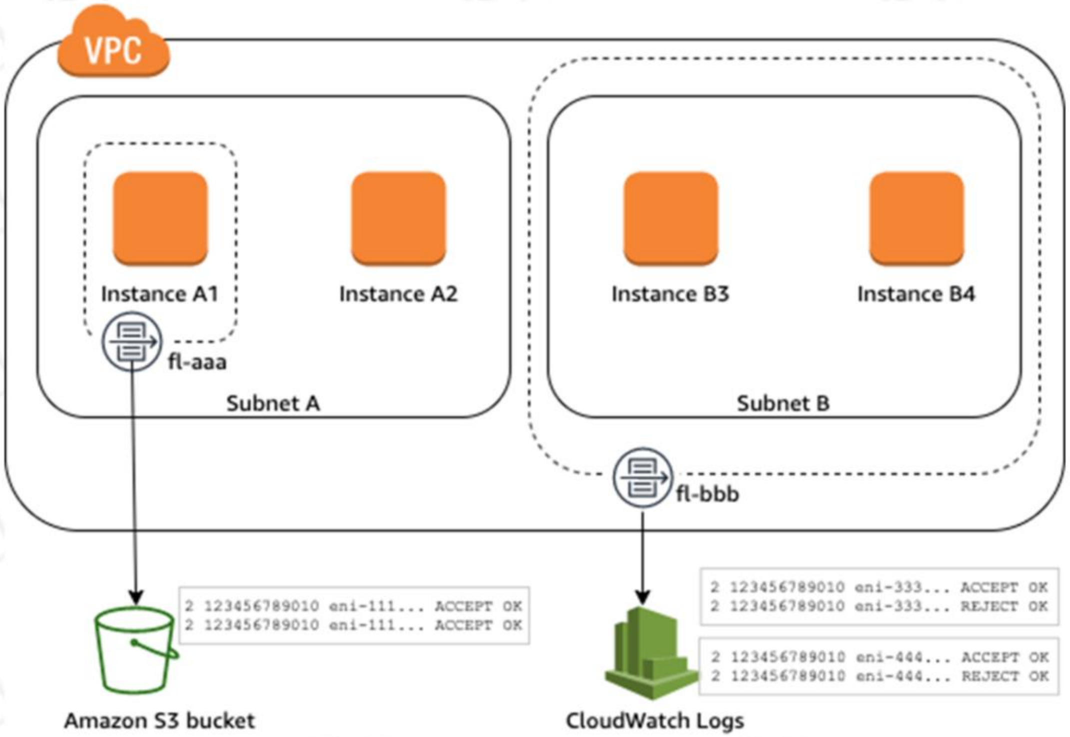
04 Implementing Gateway Load Balancer health checks

Correct Answer: B
Explanation: Creating CloudWatch dashboards allows visibility into key Gateway Load Balancer metrics for monitoring.
Incorrect Answers:
A) VPC flow logs provide network traffic metadata but not full GLB metrics.
C) Integrating with Auto Scaling groups provides scaling but not metrics.
D) Health checks determine instance health but do not provide metrics.

References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/gateway/cloudwatch-metrics.html

Gateway Load Balancer

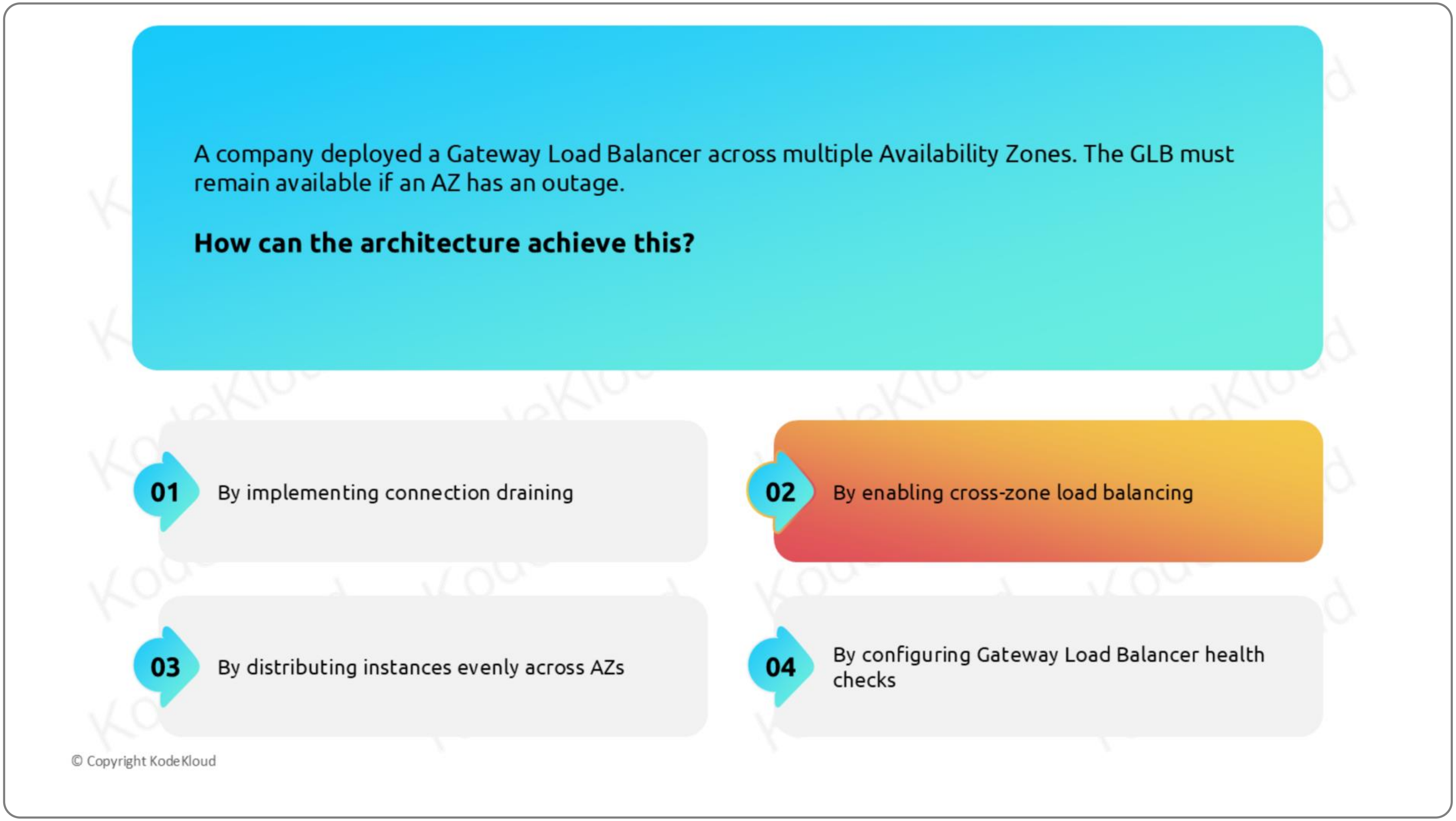
GLB uses CloudWatch, CloudTrail, and VPC flow logs for monitoring.



```
2 123456789010 eni-111... ACCEPT OK
2 123456789010 eni-111... ACCEPT OK
```

```
2 123456789010 eni-333... ACCEPT OK
2 123456789010 eni-333... REJECT OK
```

```
2 123456789010 eni-444... ACCEPT OK
2 123456789010 eni-444... REJECT OK
```



A company deployed a Gateway Load Balancer across multiple Availability Zones. The GLB must remain available if an AZ has an outage.

How can the architecture achieve this?

01 By implementing connection draining

02 By enabling cross-zone load balancing

03 By distributing instances evenly across AZs

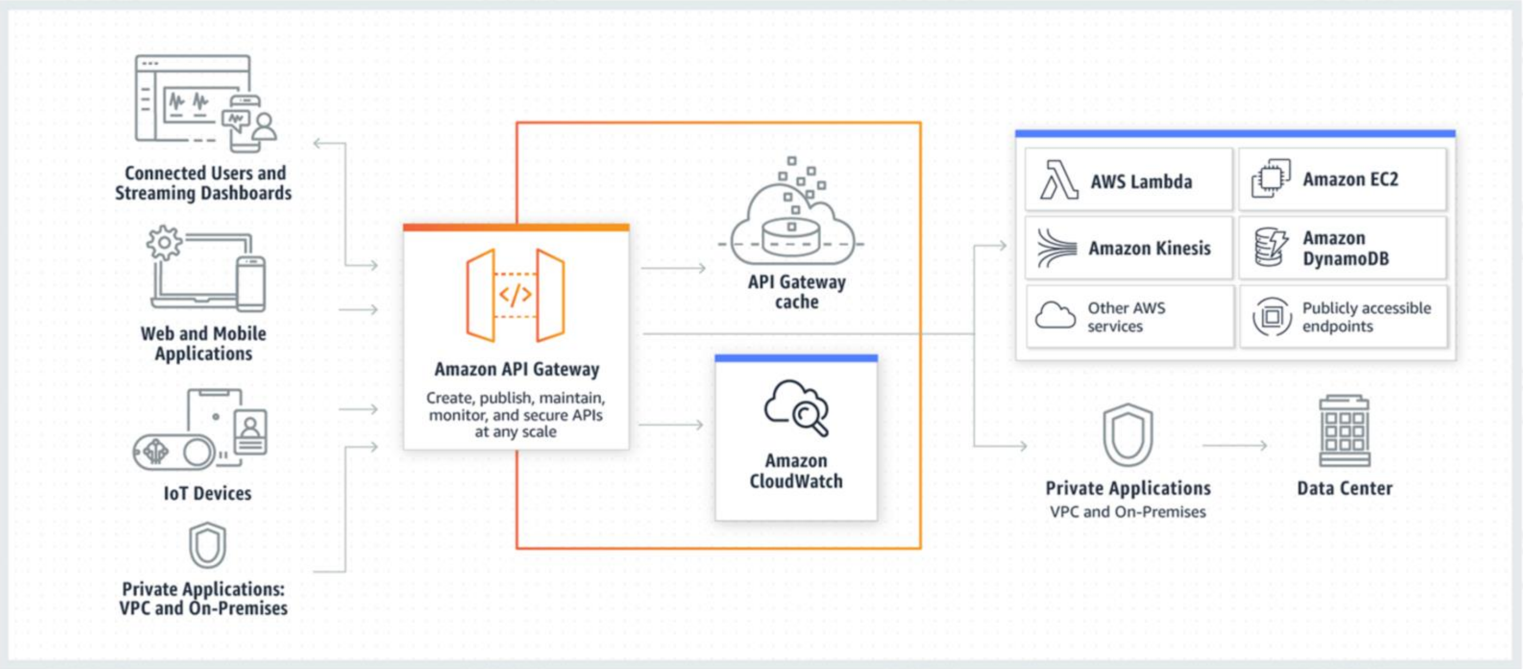
04 By configuring Gateway Load Balancer health checks

Correct Answer: B
Explanation: Enabling cross-zone load balancing for the Gateway Load Balancer allows it to remain available and route traffic across the remaining healthy AZs during an AZ outage.
Incorrect Answers:
A) Connection draining facilitates graceful instance shutdowns but does not provide AZ redundancy.
C) Distributing instances provides high availability but does not maintain GLB redundancy across AZs.

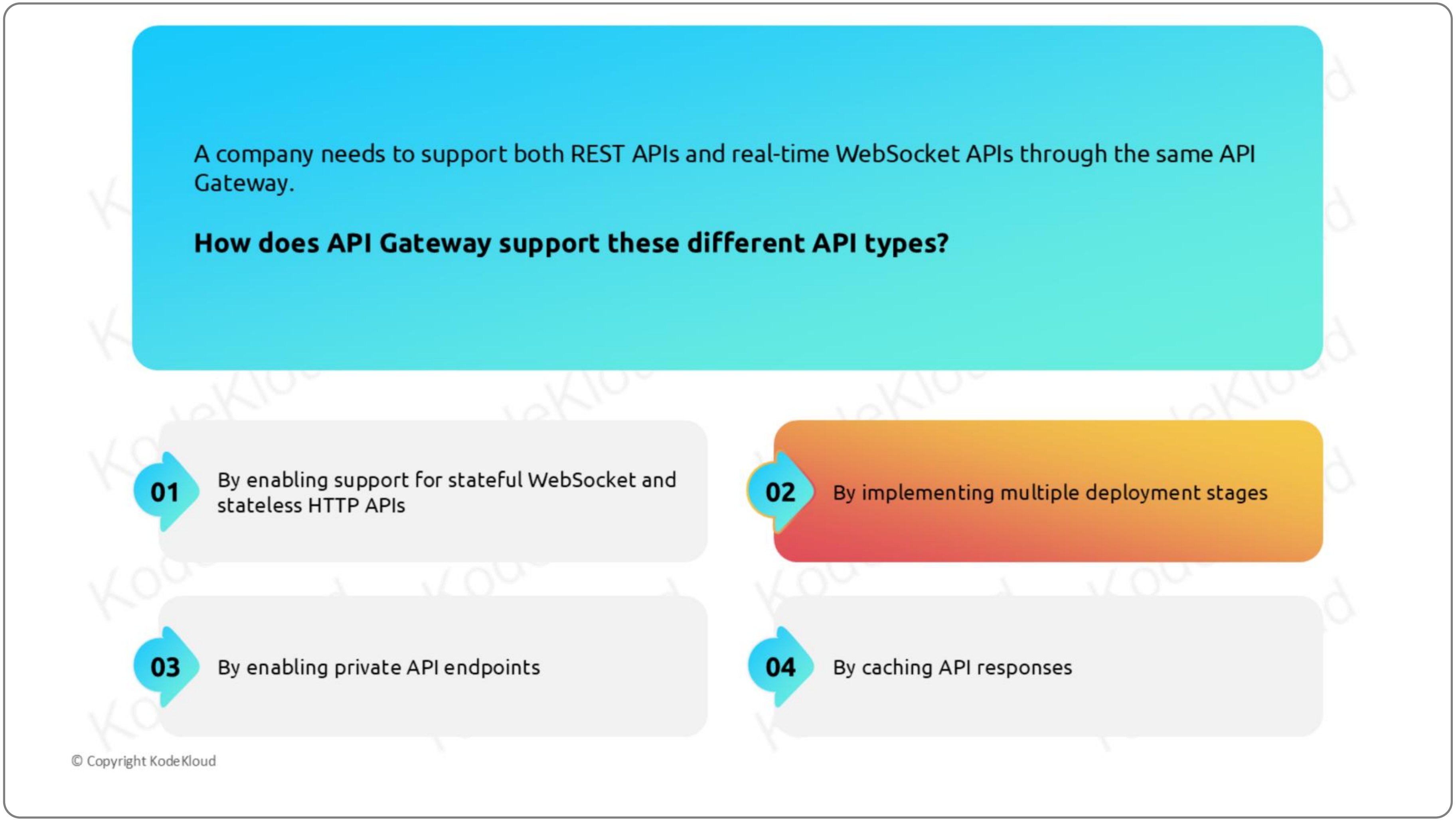
D) Health checks enable routing around unhealthy instances but do not provide AZ resiliency.
References:
https://docs.aws.amazon.com/elasticloadbalancing/latest/gateway/cross-zone-load-balancing.html

Designing for Reliability – API Gateway

API Gateway



Connected Users and Streaming Dashboards
Web and Mobile Applications
IoT Devices
Private Applications: VPC and On-Premises
Amazon API Gateway
Create, publish, maintain, monitor, and secure APIs at any scale
API Gateway cache
Amazon CloudWatch
AWS Lambda
Amazon EC2
Amazon Kinesis
Amazon DynamoDB
Other AWS services
Publicly accessible endpoints
Private Applications
VPC and On-Premises
Data Center



A company needs to support both REST APIs and real-time WebSocket APIs through the same API Gateway.

How does API Gateway support these different API types?

01 By enabling support for stateful WebSocket and stateless HTTP APIs

02 By implementing multiple deployment stages

03 By enabling private API endpoints

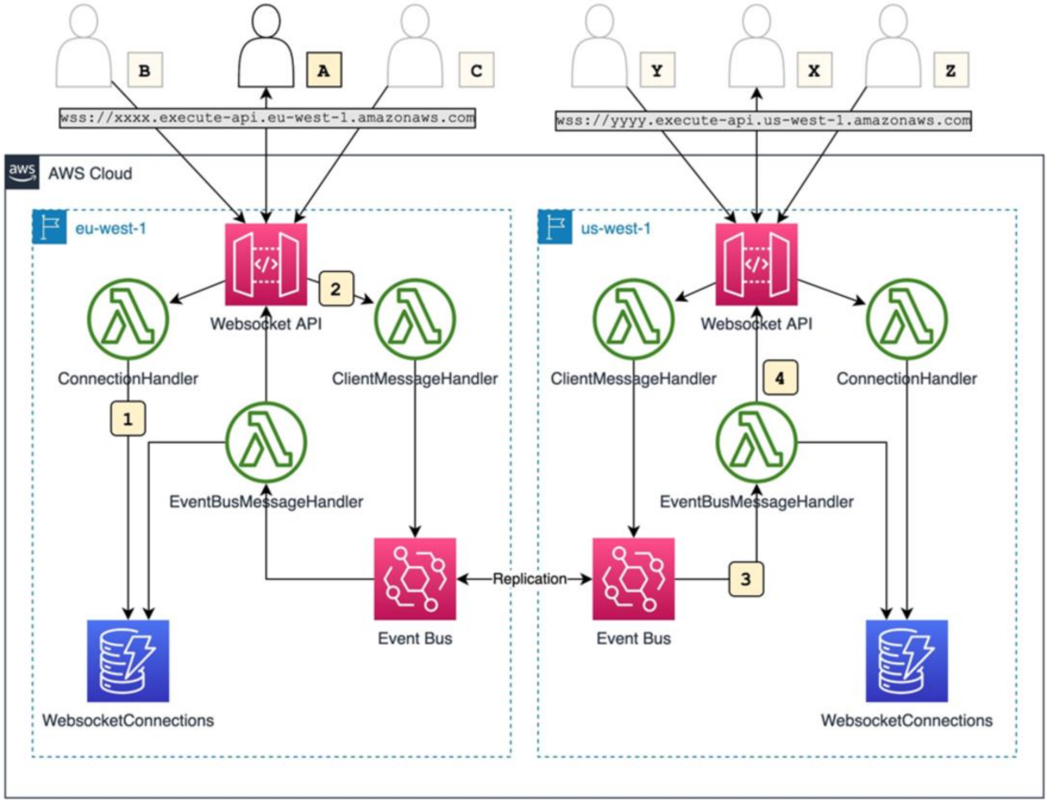
04 By caching API responses

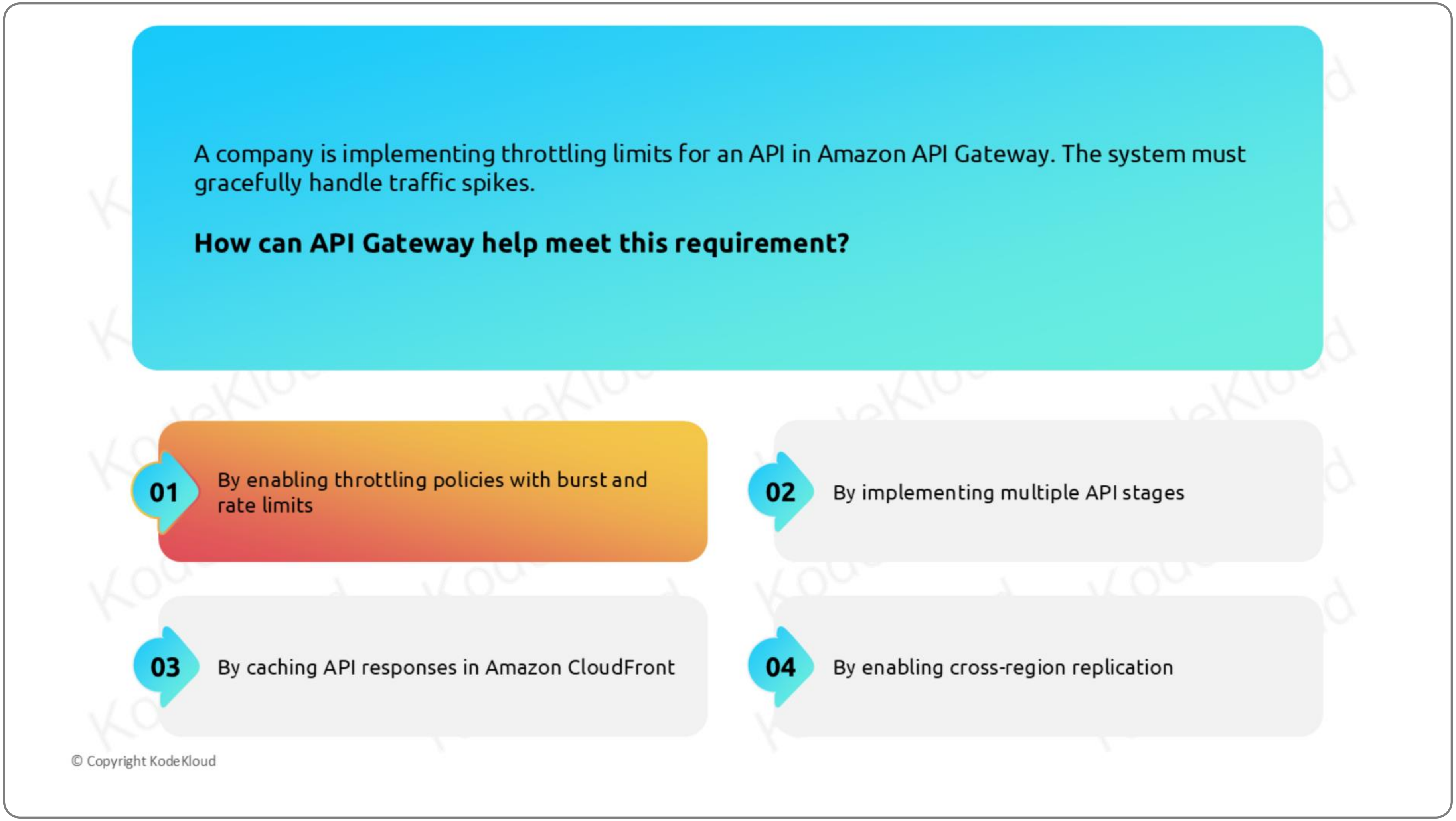
Correct Answer: B. API Gateway acts as a "front door" for applications to access data, business logic, or functionality from your backend services, such as workloads running on EC2, Lambda, or any web application.
Incorrect: A. API Gateway is designed for microservices and serverless architectures, not monolithic applications. C. API Gateway is designed to manage APIs and does support integration with various backend services. D. API Gateway is a fully managed service, eliminating the need for manual setup and maintenance of physical hardware.
Explanation: Amazon API Gateway is a fully managed service that makes it easy for developers to create, publish, maintain,

monitor, and secure APIs at any scale. It acts as an entry point for applications, allowing them to access various backend services seamlessly. With API Gateway, you can efficiently manage traffic, handle bursts of traffic, and even connect to services like AWS Lambda and Amazon EC2.
References: Amazon API Gateway - Developer Guide

API Gateway

API Gateway has a few Reliability features such as Stateful versus Stateless.





A company is implementing throttling limits for an API in Amazon API Gateway. The system must gracefully handle traffic spikes.

How can API Gateway help meet this requirement?

01 By enabling throttling policies with burst and rate limits

02 By implementing multiple API stages

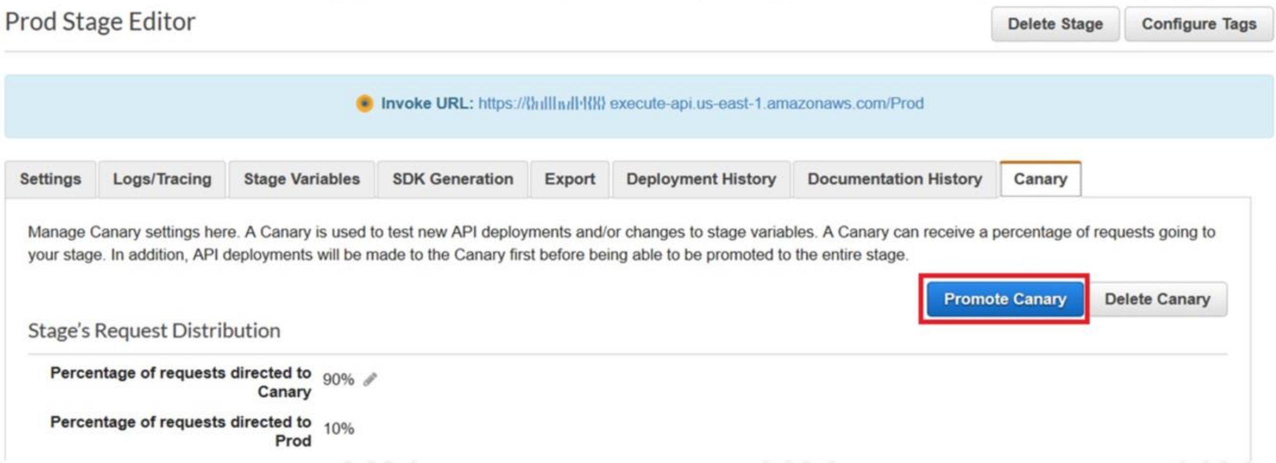
03 By caching API responses in Amazon CloudFront

04 By enabling cross-region replication

Correct Answer: A
Explanation: API Gateway throttling allows configuring burst and rate limits to smoothly handle traffic spikes and prevent system overload.
Incorrect Answers:
B) Multiple stages help manage deployments but do not throttle traffic.
C) CloudFront caching improves performance but does not throttle requests.

D) Cross-region replication provides HA but does not throttle traffic.
References:
https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-request-throttling.html

API Gateway



Prod Stage Editor

Invoke URL: https://[redacted] execute-api.us-east-1.amazonaws.com/Prod

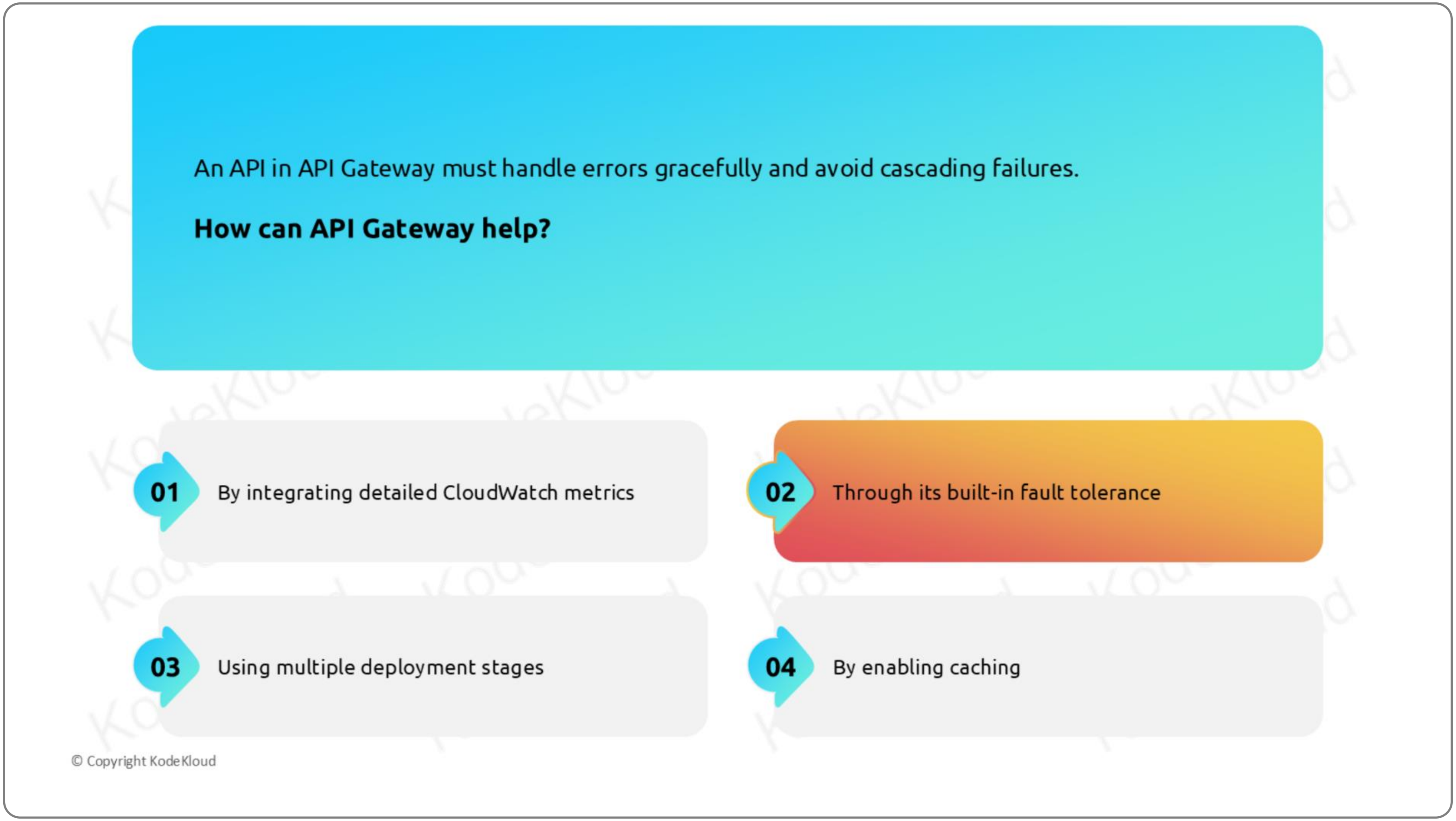
Manage Canary settings here. A Canary is used to test new API deployments and/or changes to stage variables. A Canary can receive a percentage of requests going to your stage. In addition, API deployments will be made to the Canary first before being able to be promoted to the entire stage.

Stage's Request Distribution

Percentage of requests directed to Canary 90%

Percentage of requests directed to Prod 10%

Canary deployments allow you to deploy forward or back depending on what you need.



An API in API Gateway must handle errors gracefully and avoid cascading failures.

How can API Gateway help?

01 By integrating detailed CloudWatch metrics

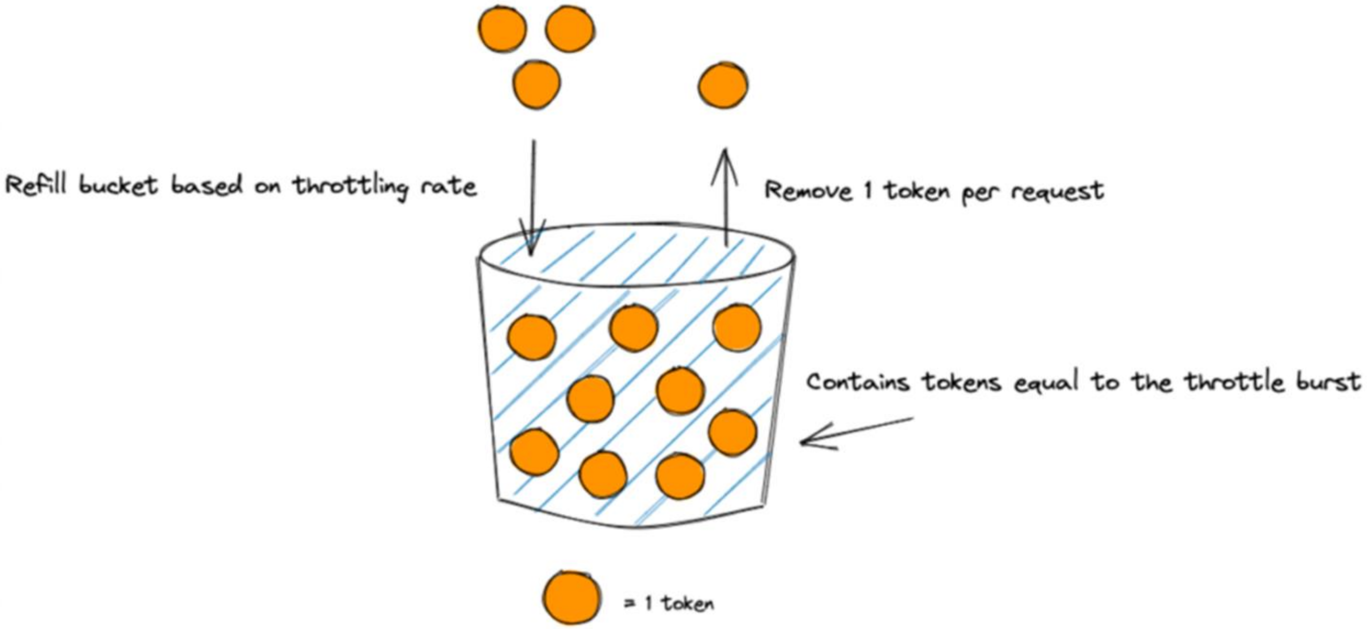
02 Through its built-in fault tolerance

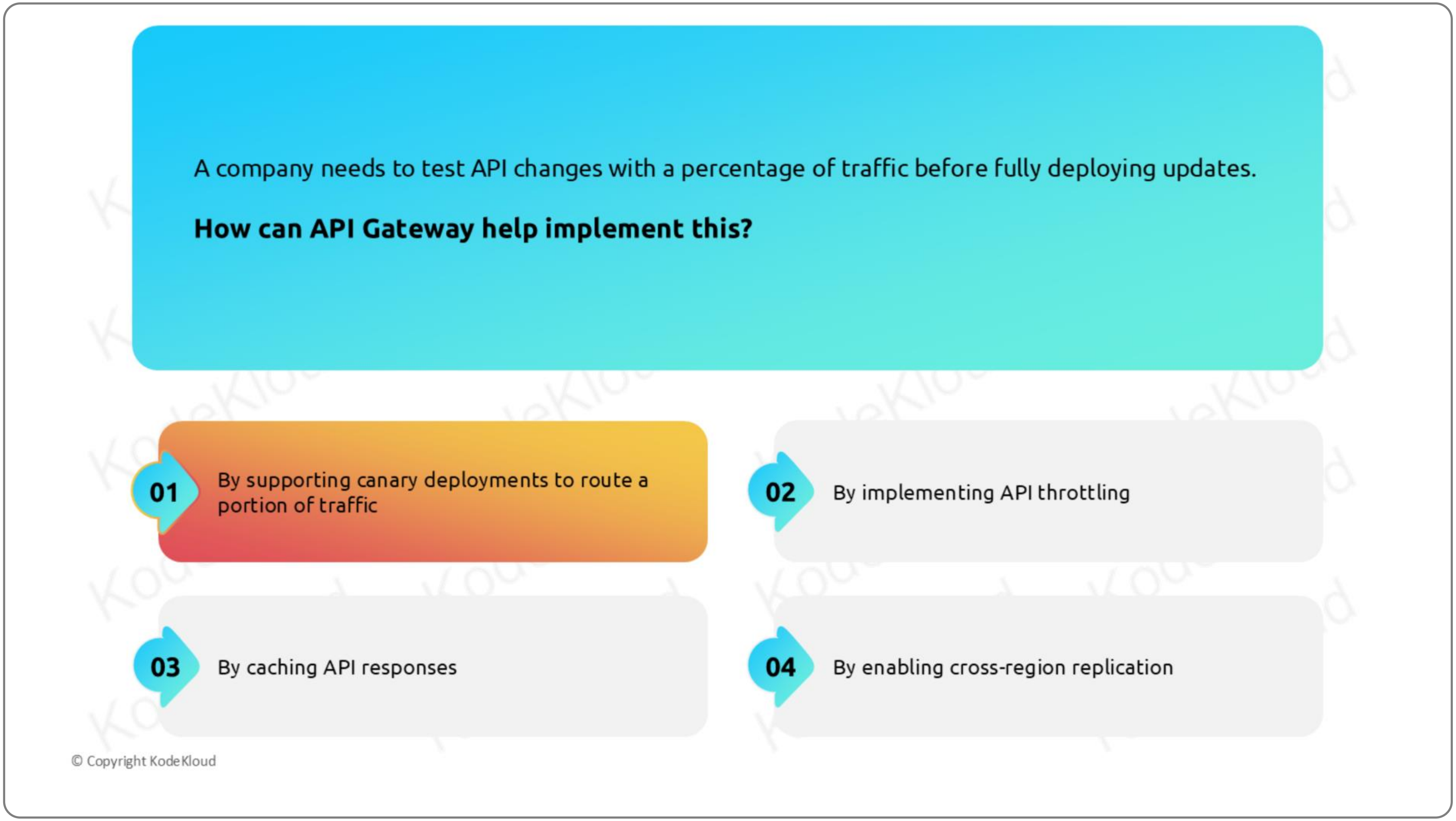
03 Using multiple deployment stages

04 By enabling caching

Answer: B
Explanation: API Gateway has robust fault tolerance built-in to handle errors and anomalies without failing end users.
Domain: Design Resilient Architectures

API Gateway


Refill bucket based on throttling rate
Remove 1 token per request
Contains tokens equal to the throttle burst
= 1 token



A company needs to test API changes with a percentage of traffic before fully deploying updates.

How can API Gateway help implement this?

01 By supporting canary deployments to route a portion of traffic

02 By implementing API throttling

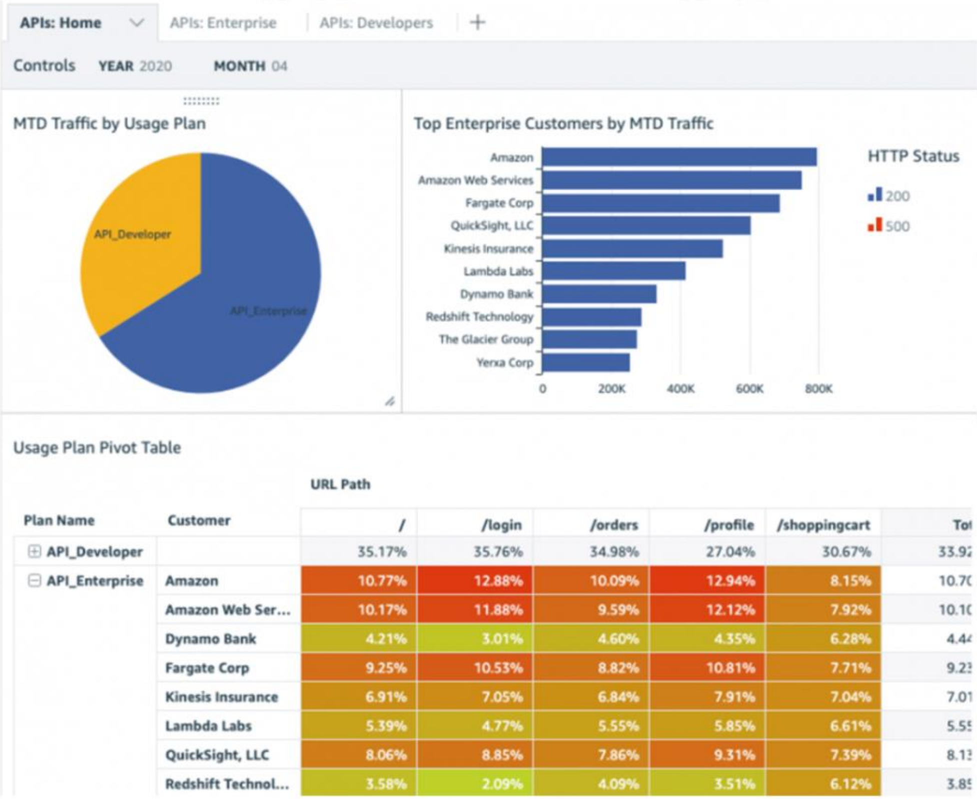
03 By caching API responses

04 By enabling cross-region replication

Explanation:
A) Correct. API Gateway canary deployments allow splitting traffic between two stages to gradually shift a percentage of users to new API versions.
B) Incorrect. Throttling helps manage traffic spikes but does not relate to canary deployments.
C) Incorrect. Caching improves performance but does not relate to canary deployments.
D) Incorrect. Cross-region replication provides HA but does not relate to canary deployments.

References:
https://docs.aws.amazon.com/apigateway/latest/developerguide/deploy-canary-release.html

API Gateway



APIs: Home | APIs: Enterprise | APIs: Developers | +

Controls YEAR 2020 MONTH 04

MTD Traffic by Usage Plan

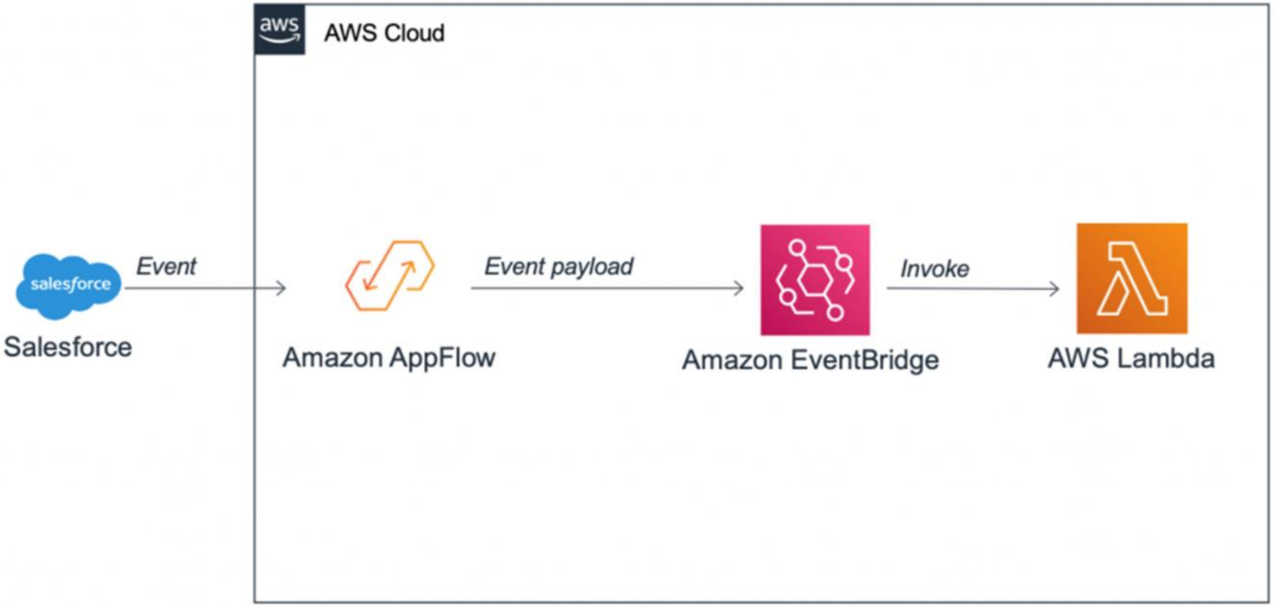
Top Enterprise Customers by MTD Traffic

Usage Plan Pivot Table

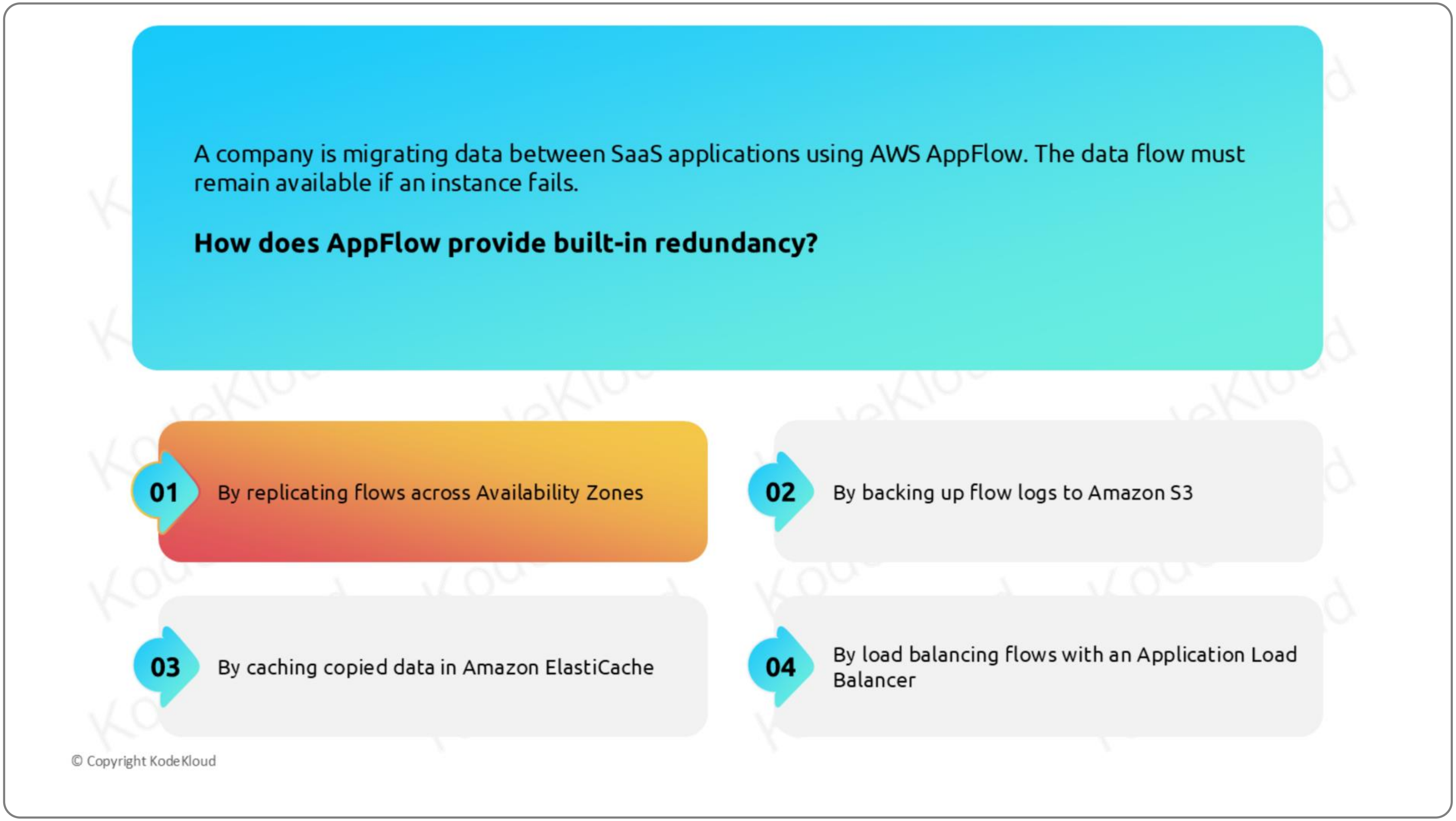
		URL Path					
Plan Name	Customer	/	/login	/orders	/profile	/shoppingcart	Tot
⊞ API_Developer		35.17%	35.76%	34.98%	27.04%	30.67%	33.9
⊟ API_Enterprise	Amazon	10.77%	12.88%	10.09%	12.94%	8.15%	10.7
	Amazon Web Ser...	10.17%	11.88%	9.59%	12.12%	7.92%	10.1
	Dynamo Bank	4.21%	3.01%	4.60%	4.35%	6.28%	4.4
	Fargate Corp	9.25%	10.53%	8.82%	10.81%	7.71%	9.2
	Kinesis Insurance	6.91%	7.05%	6.84%	7.91%	7.04%	7.01
	Lambda Labs	5.39%	4.77%	5.55%	5.85%	6.61%	5.5
	QuickSight, LLC	8.06%	8.85%	7.86%	9.31%	7.39%	8.1
	Redshift Technol...	3.58%	2.09%	4.09%	3.51%	6.12%	3.8

Designing for Reliability – AppFlow

AppFlow



Amazon AppFlow is just about receiving and sending data from third parties, so Reliability, being a highly managed service, is integrated naturally.



A company is migrating data between SaaS applications using AWS AppFlow. The data flow must remain available if an instance fails.

How does AppFlow provide built-in redundancy?

01 By replicating flows across Availability Zones

02 By backing up flow logs to Amazon S3

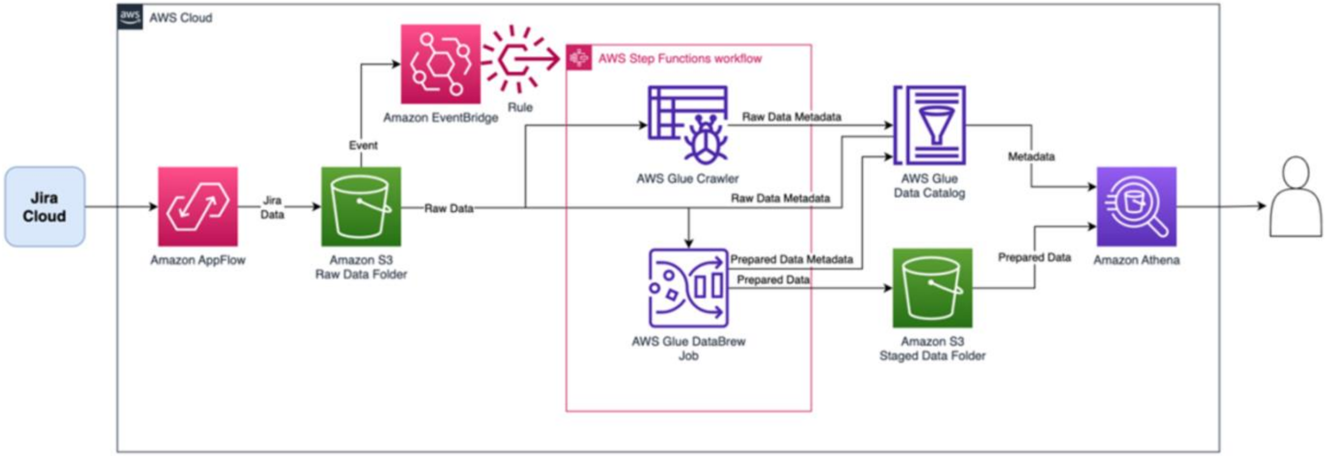
03 By caching copied data in Amazon ElastiCache

04 By load balancing flows with an Application Load Balancer

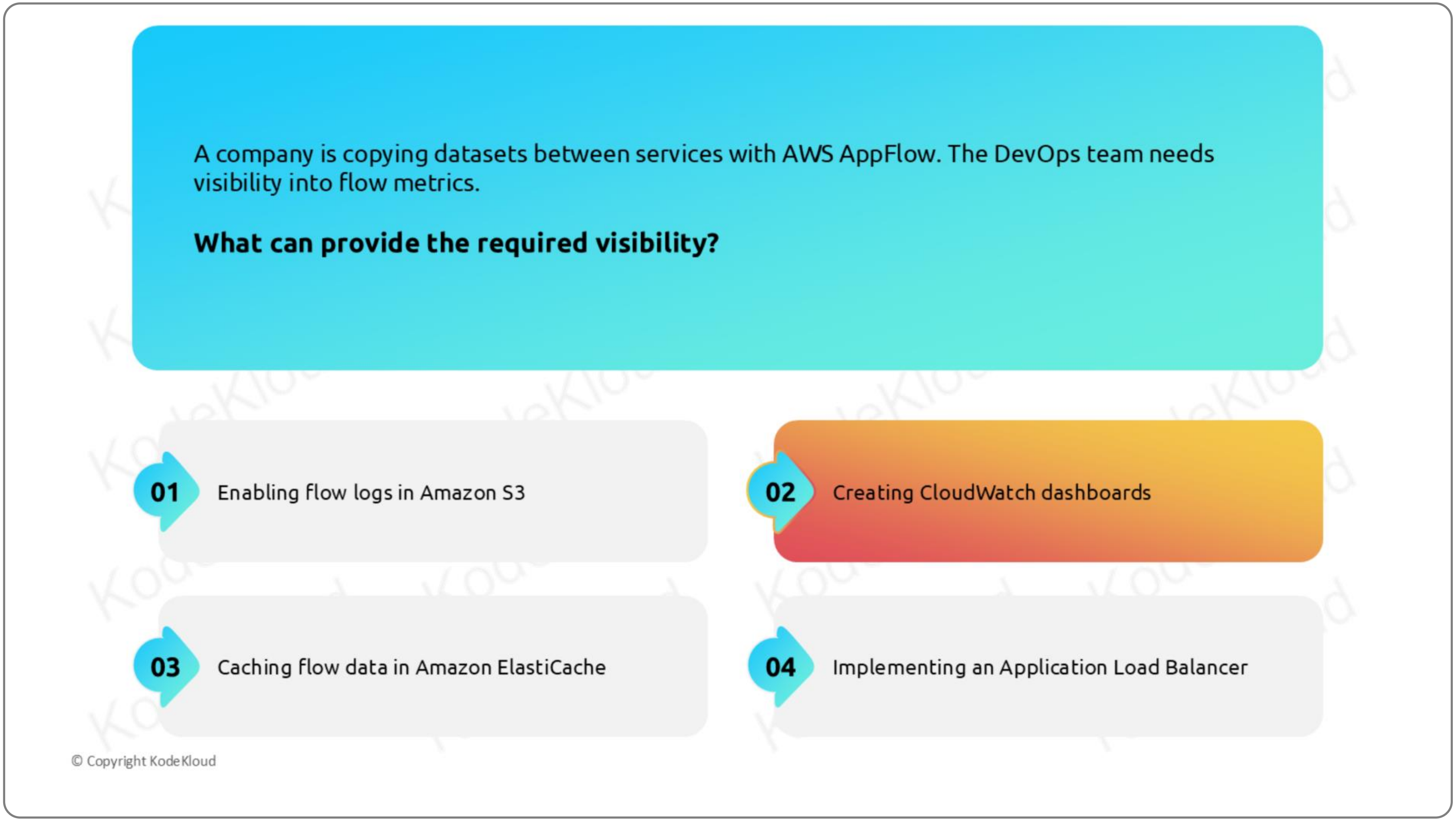
Correct Answer: A
Explanation: AWS AppFlow automatically runs flows in a highly available configuration across Availability Zones to remain resilient to AZ outages.
Incorrect Answers:
B) Flow logs capture execution history but do not provide redundancy.
C) Caching copied data improves performance but does not provide redundancy.

D) An Application Load Balancer helps scale traffic but AppFlow provides built-in redundancy.
References:
https://docs.aws.amazon.com/appflow/latest/userguide/service-understands.html

Designing for Reliability – AppFlow



AppFlow relies on Built-in Redundancy to ensure transfer reliability.



A company is copying datasets between services with AWS AppFlow. The DevOps team needs visibility into flow metrics.

What can provide the required visibility?

01 Enabling flow logs in Amazon S3

02 Creating CloudWatch dashboards

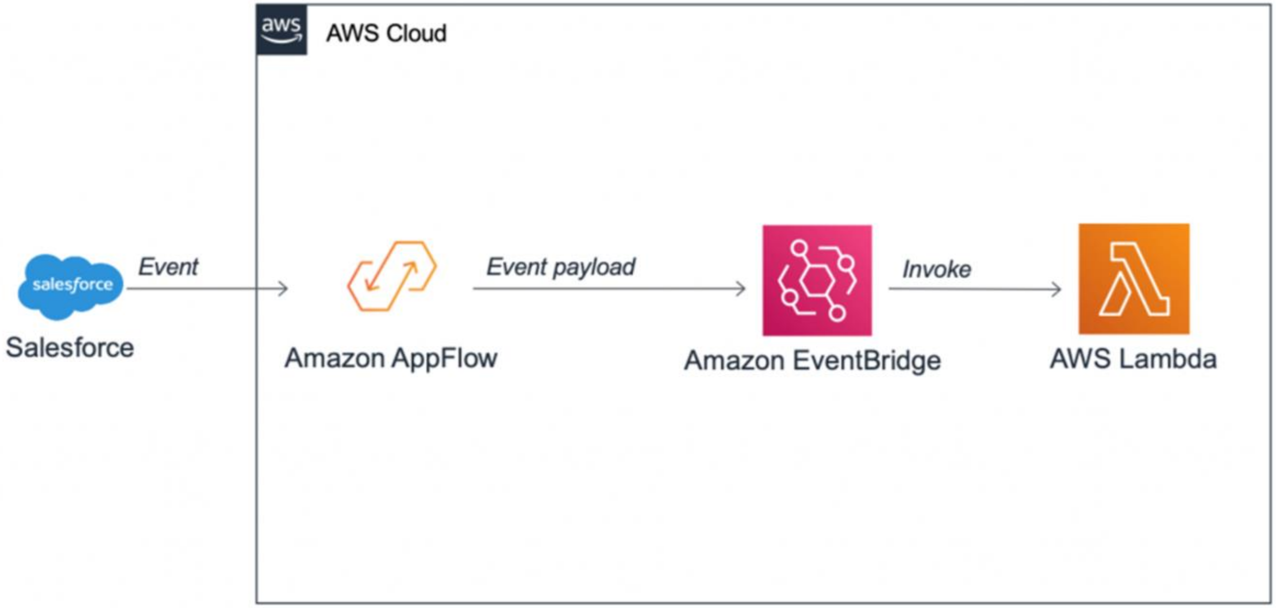
03 Caching flow data in Amazon ElastiCache

04 Implementing an Application Load Balancer

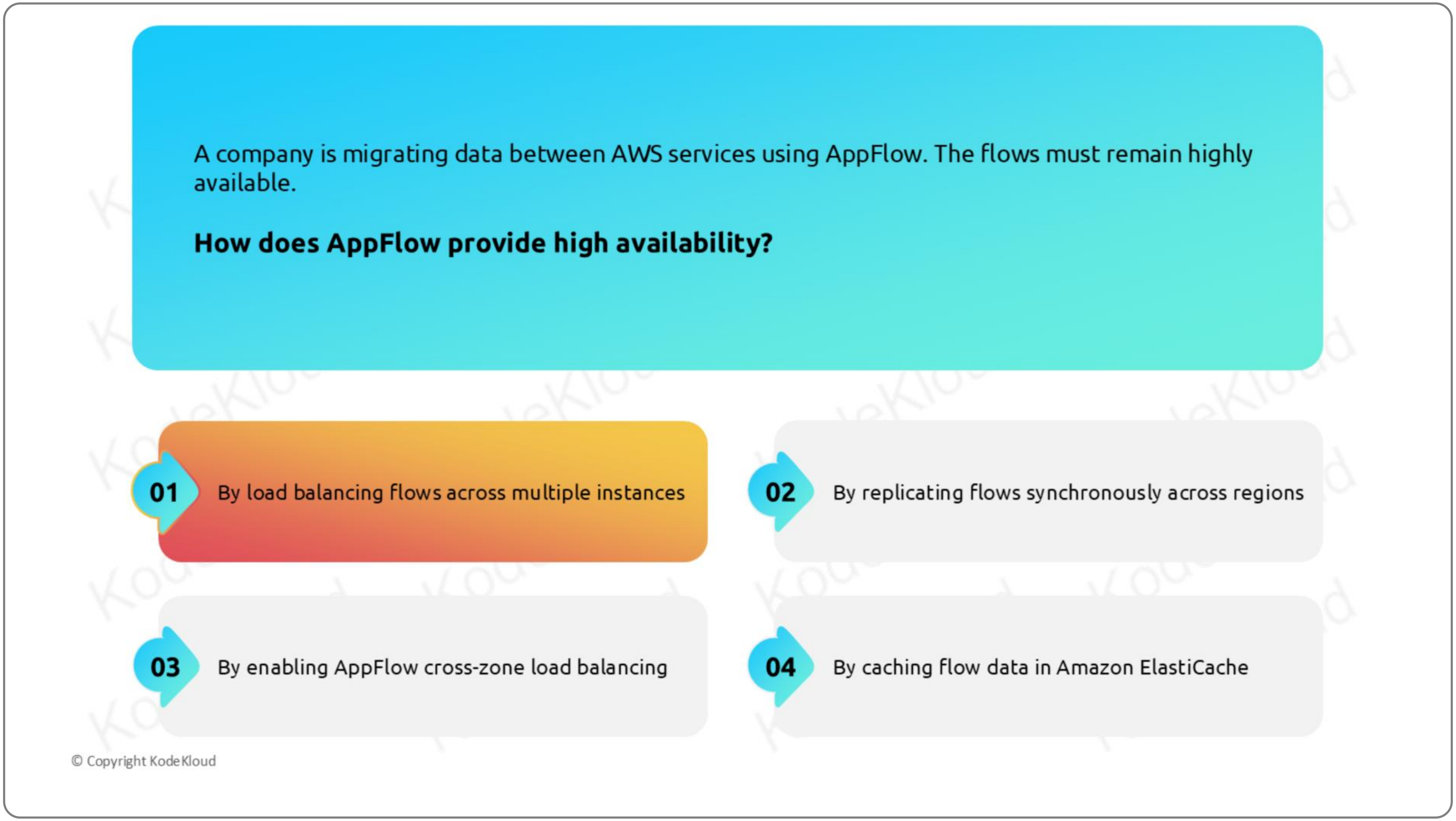
Correct Answer: B
Explanation: Creating CloudWatch dashboards provides visibility into key AppFlow metrics like failed executions, data errors, and flow latency.
Incorrect Answers:
A) Flow logs capture execution history but do not provide operational metrics visibility.
C) Caching flow data locally does not provide visibility into AppFlow metrics.

D) Using a load balancer helps distribute traffic but does not provide visibility.
References:
https://docs.aws.amazon.com/appflow/latest/userguide/monitoring-cloudwatch.html

AppFlow



Several Monitoring and Flow Management tools are available that allow visibility into the progress of jobs.



A company is migrating data between AWS services using AppFlow. The flows must remain highly available.

How does AppFlow provide high availability?

01 By load balancing flows across multiple instances

02 By replicating flows synchronously across regions

03 By enabling AppFlow cross-zone load balancing

04 By caching flow data in Amazon ElastiCache

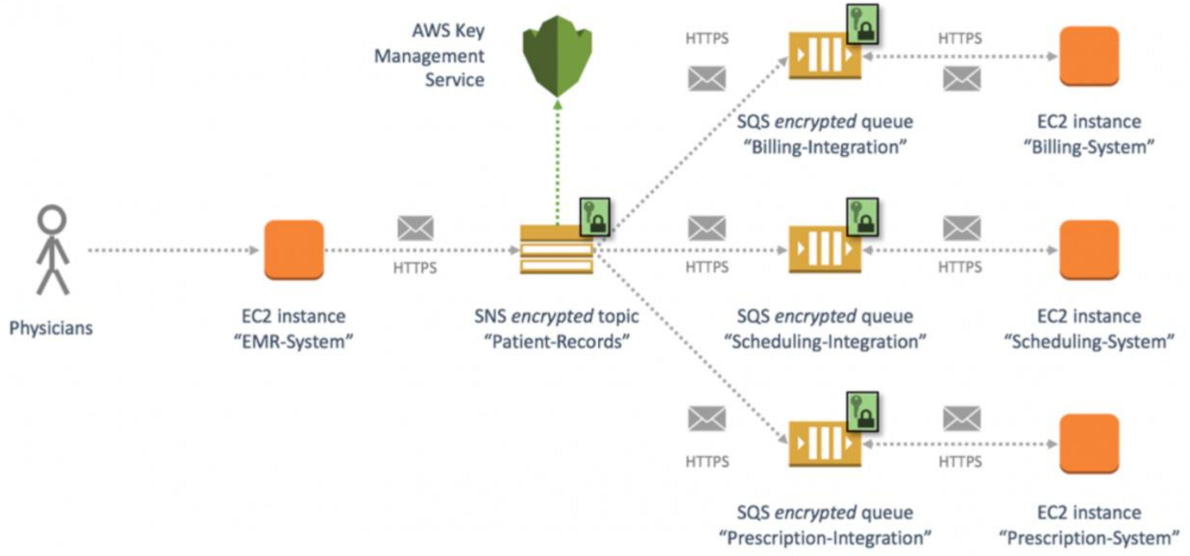
Correct Answer: A
Explanation: AppFlow runs flows across multiple instances in a high availability configuration spanning AZs. If an instance fails, other instances process the flows.
Incorrect Answers:
B) AppFlow does not replicate flows across regions.
C) Cross-zone load balancing is not an AppFlow feature.

D) Caching flow data does not provide redundancy.
References:
https://docs.aws.amazon.com/appflow/latest/userguide/service-understands.html

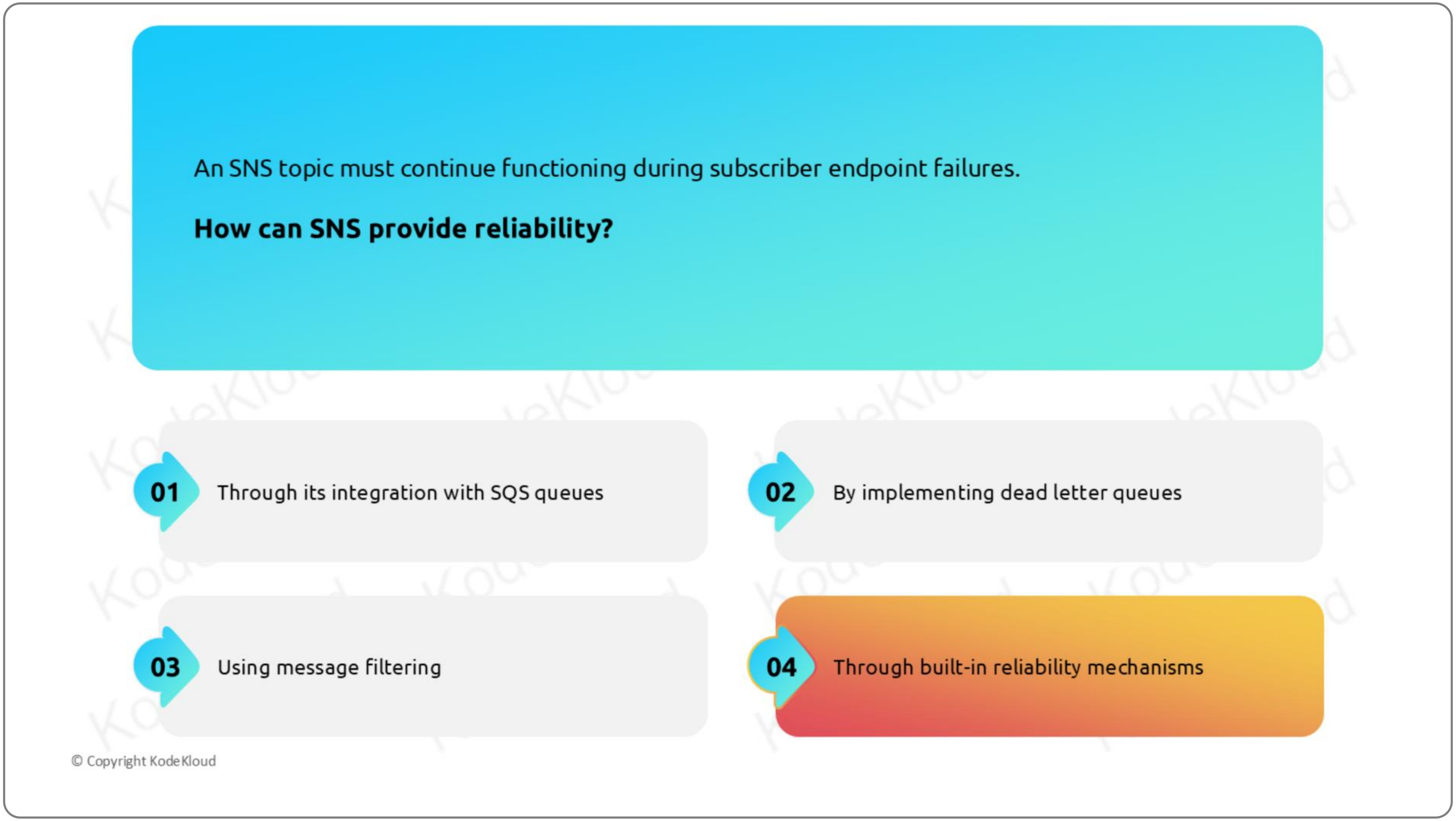
The Power of Messaging and Events

Designing for Reliability – Simple Notification Service

Simple Notification Service


AWS Key Management Service
HTTPS
HTTPS
SQS *encrypted* queue "Billing-Integration"
EC2 instance "Billing-System"
Physicians
EC2 instance "EMR-System"
HTTPS
SNS *encrypted* topic "Patient-Records"
HTTPS
HTTPS
SQS *encrypted* queue "Scheduling-Integration"
EC2 instance "Scheduling-System"
HTTPS
HTTPS
SQS *encrypted* queue "Prescription-Integration"
EC2 instance "Prescription-System"

Simple Notification Service is a notification system. Reliability happens by default as it is a highly managed system.



An SNS topic must continue functioning during subscriber endpoint failures.

How can SNS provide reliability?

01 Through its integration with SQS queues

02 By implementing dead letter queues

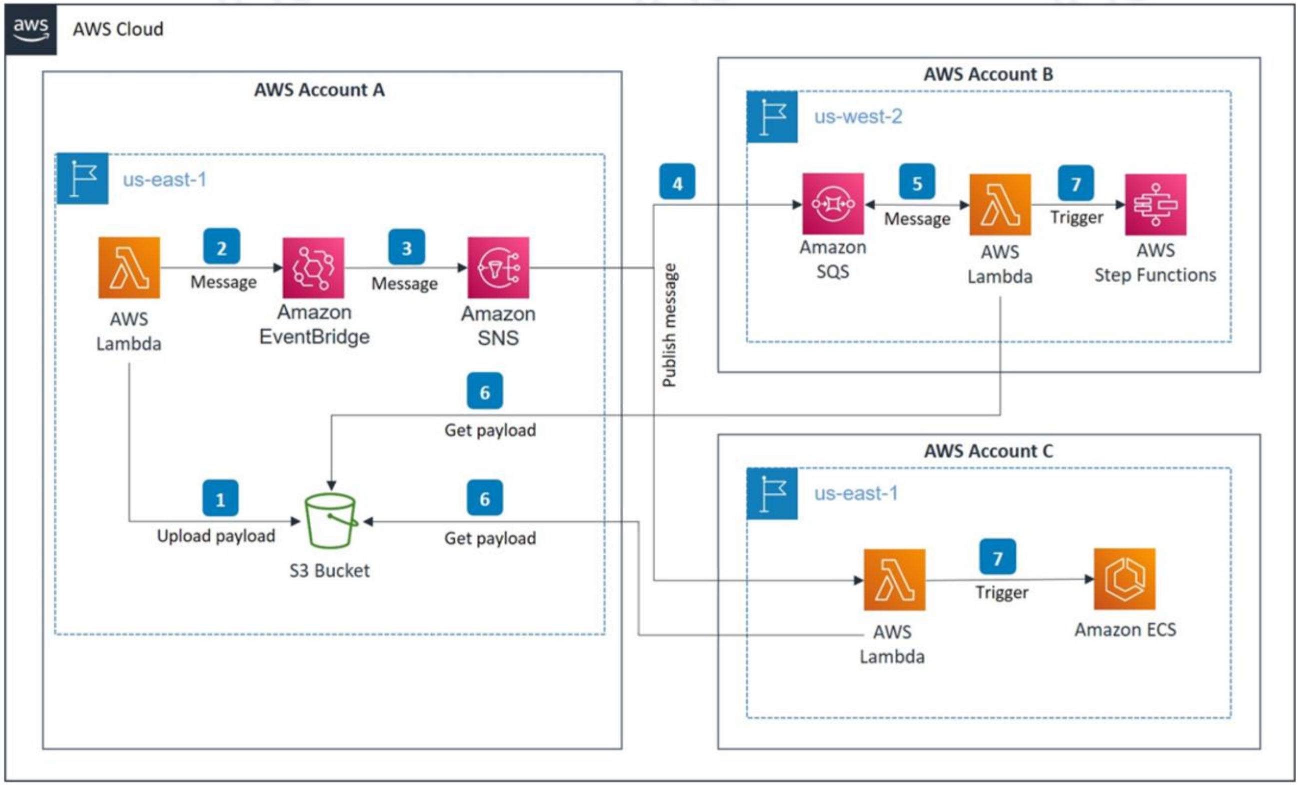
03 Using message filtering

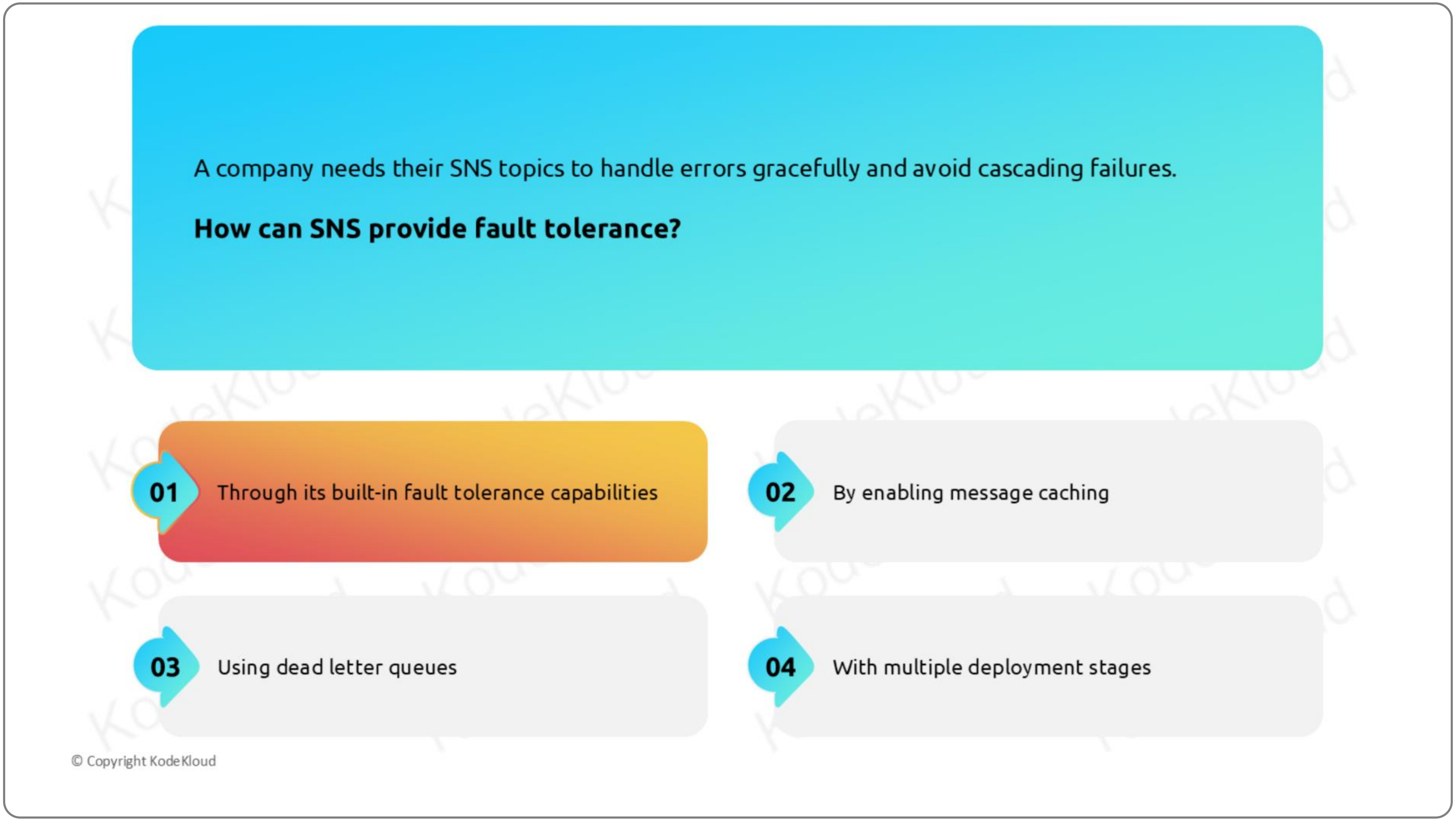
04 Through built-in reliability mechanisms

Correct Answer: D
Explanation: SNS has built-in reliability mechanisms like retries and robust message handling that maintain uptime despite subscriber endpoints failing.

Simple Notification Service

Simple Notification Service is scalable and fault tolerant by default





A company needs their SNS topics to handle errors gracefully and avoid cascading failures.

How can SNS provide fault tolerance?

01 Through its built-in fault tolerance capabilities

02 By enabling message caching

03 Using dead letter queues

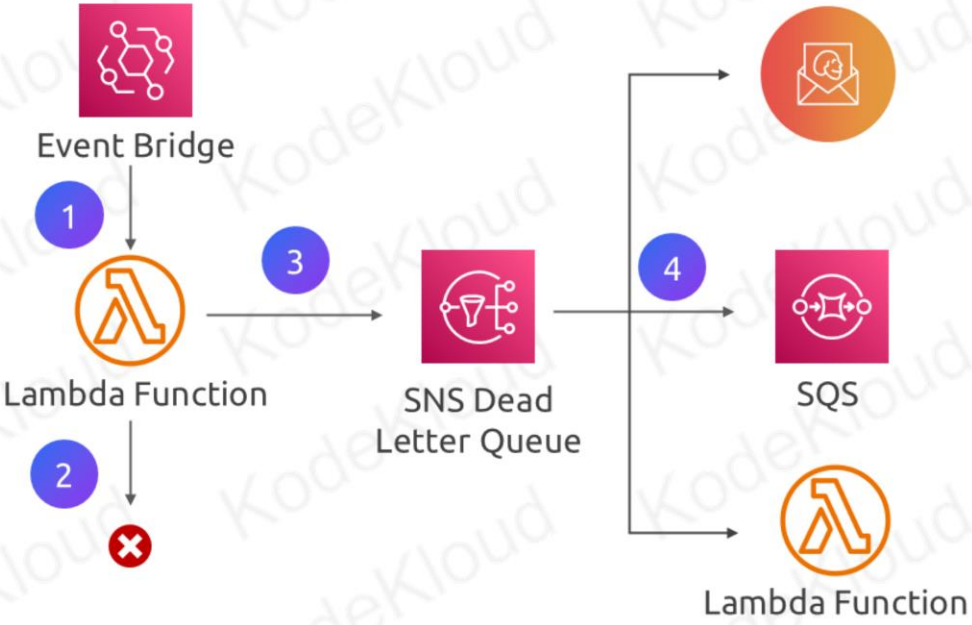
04 With multiple deployment stages

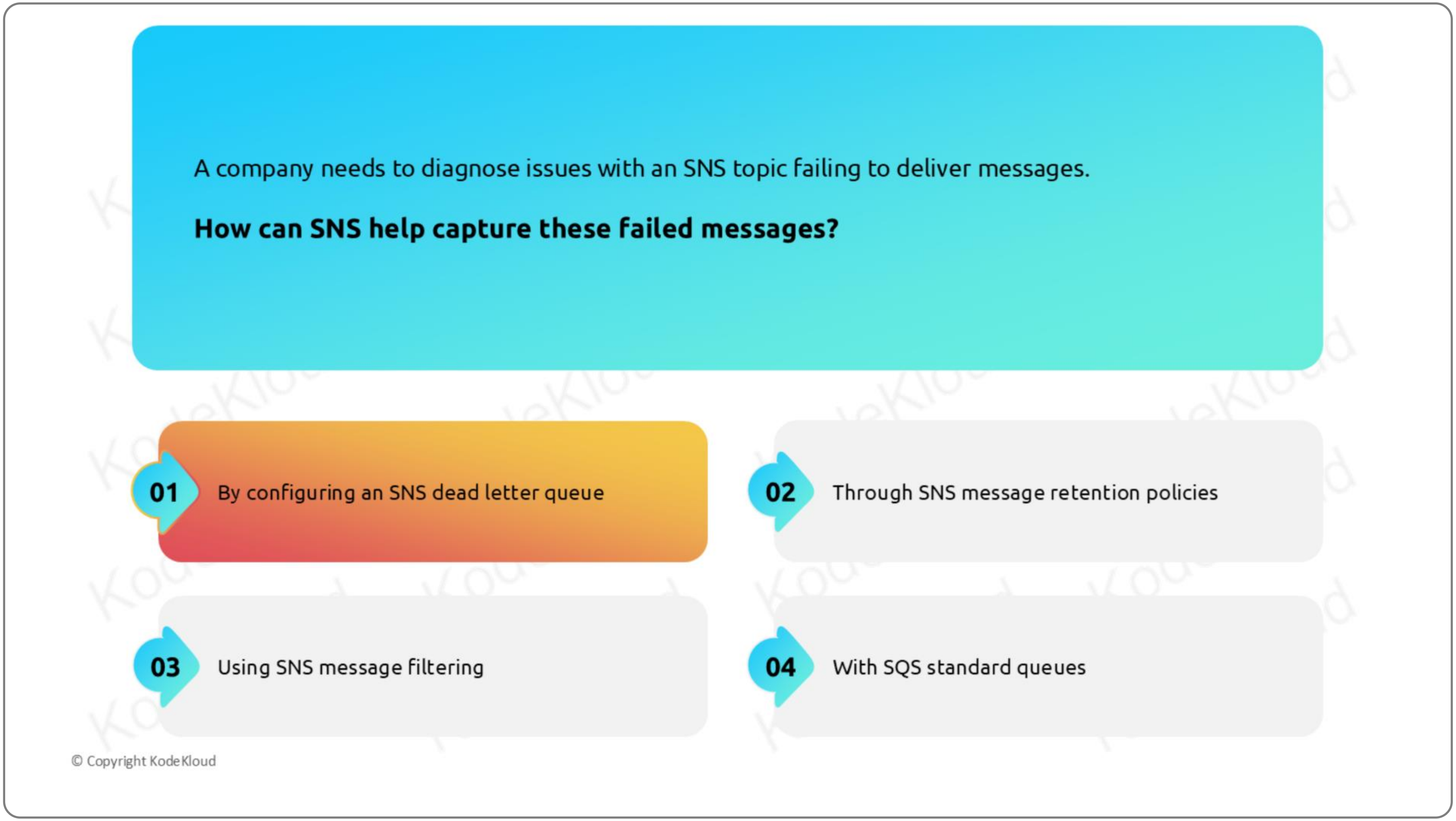
Correct Answer: A
Explanation: SNS has robust built-in fault tolerance to handle errors and anomalies without failing publishers or subscribers.
Incorrect Answers:
B) Message caching improves performance but does not provide fault tolerance.
C) Dead letter queues capture failed messages but do not provide fault tolerance.

D) Multiple stages help deploy changes but do not provide fault tolerance.
References:
https://docs.aws.amazon.com/sns/latest/dg/sns-high-throughput-failure-resiliency.html

Simple Notification Service

SNS supports a Dead Letter Queue for items that were undeliverable, thus increasing reliability.





A company needs to diagnose issues with an SNS topic failing to deliver messages.

How can SNS help capture these failed messages?

01 By configuring an SNS dead letter queue

02 Through SNS message retention policies

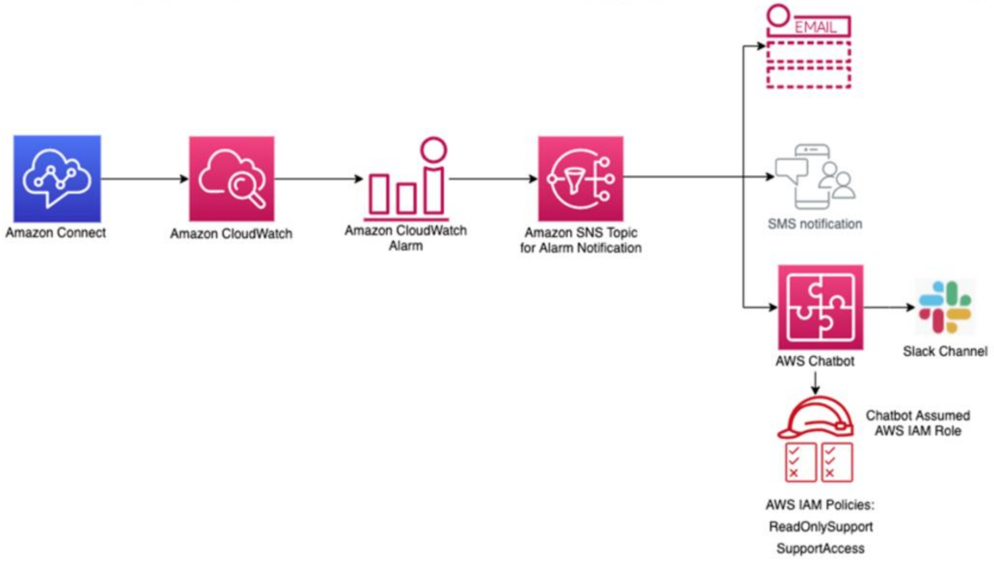
03 Using SNS message filtering

04 With SQS standard queues

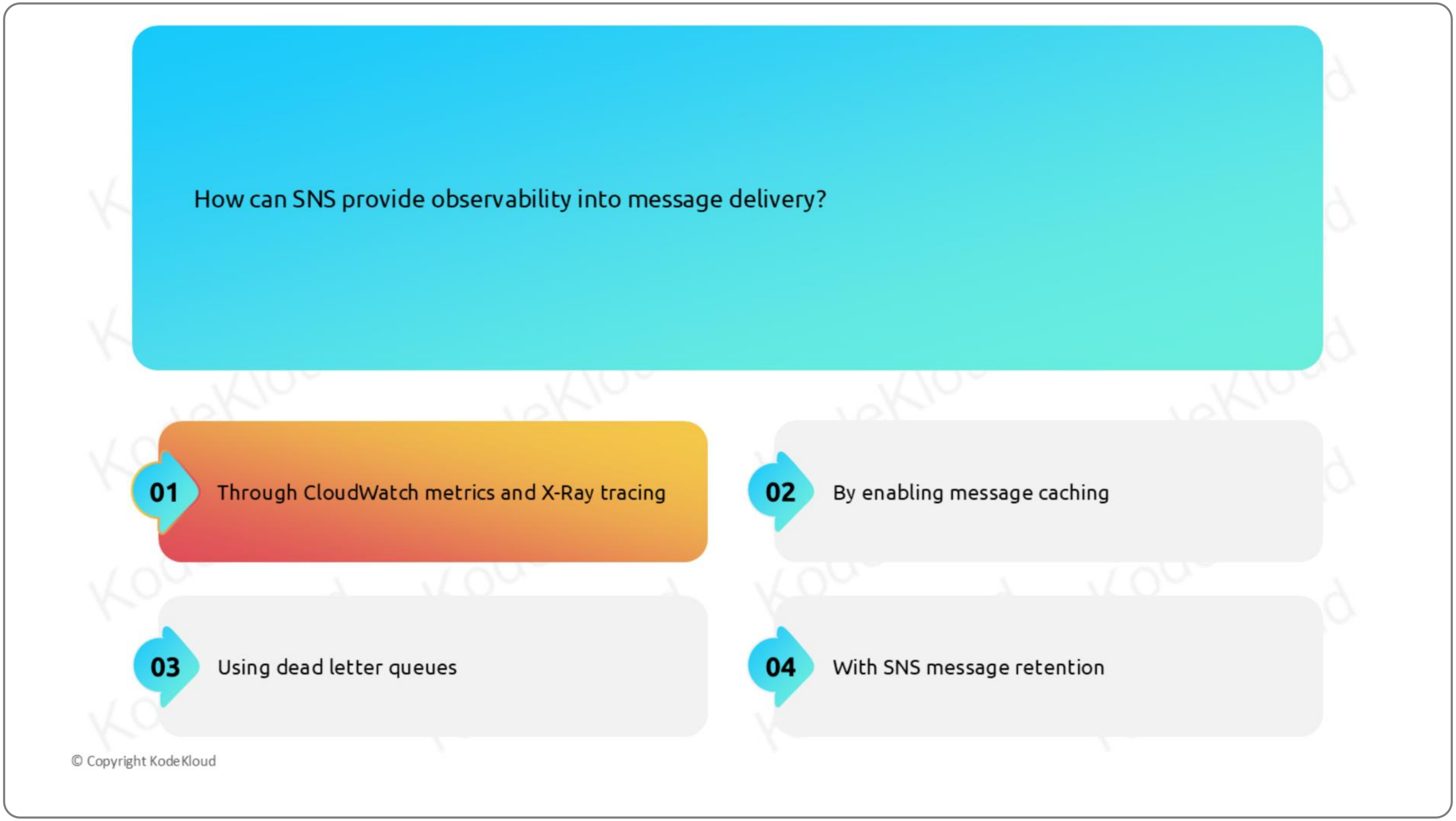
Correct Answer: A
Explanation: SNS dead letter queues can capture messages that failed processing or delivery to aid in diagnosing issues.
Incorrect Answers:
B) Message retention retains messages but does not capture failures.
C) Message filtering drops unwanted messages but does not log failures.
D) SQS standard queues provide reliability but do not capture SNS dead letters.

References:
https://docs.aws.amazon.com/sns/latest/dg/sns-dead-letter-queues.html

Simple Notification Service



SNS uses CloudTrail and CloudWatch for standard metrics and auditing.



How can SNS provide observability into message delivery?

01 Through CloudWatch metrics and X-Ray tracing

02 By enabling message caching

03 Using dead letter queues

04 With SNS message retention

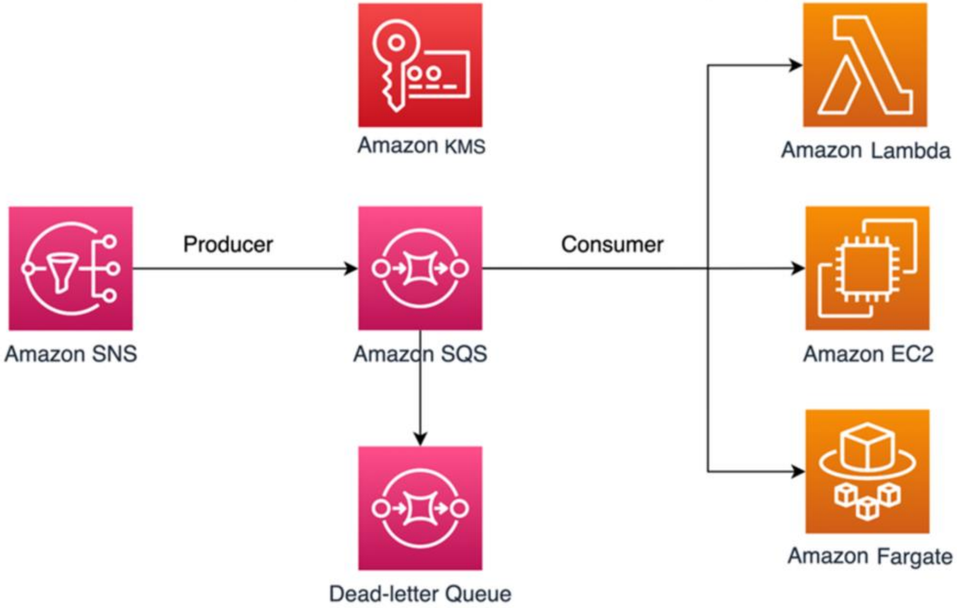
Correct Answer: A
Explanation: CloudWatch and X-Ray offer detailed monitoring and tracing for SNS message delivery.
Incorrect Answers:
B) Caching improves performance but does not provide observability.
C) Dead letter queues capture failures but do not provide tracing.
D) Message retention retains messages but does not provide observability.

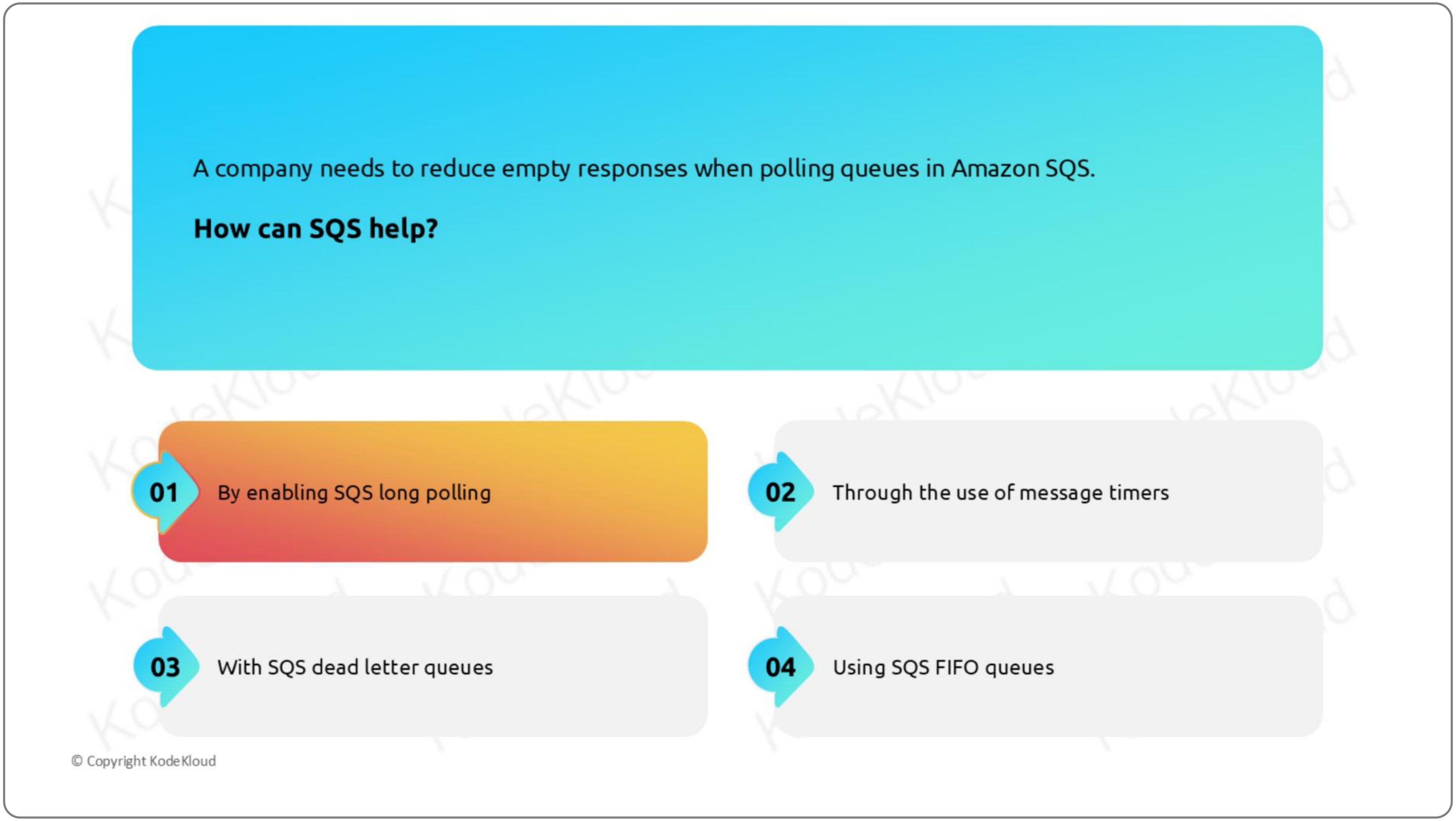
References:
https://docs.aws.amazon.com/sns/latest/dg/sns-monitoring-using-cloudwatch.html

Designing for Reliability – Simple Queue Service

Simple Queue Service

SQS hosts data, but it is a highly managed service, so everything can be encrypted server-side.





A company needs to reduce empty responses when polling queues in Amazon SQS.

How can SQS help?

01 By enabling SQS long polling

02 Through the use of message timers

03 With SQS dead letter queues

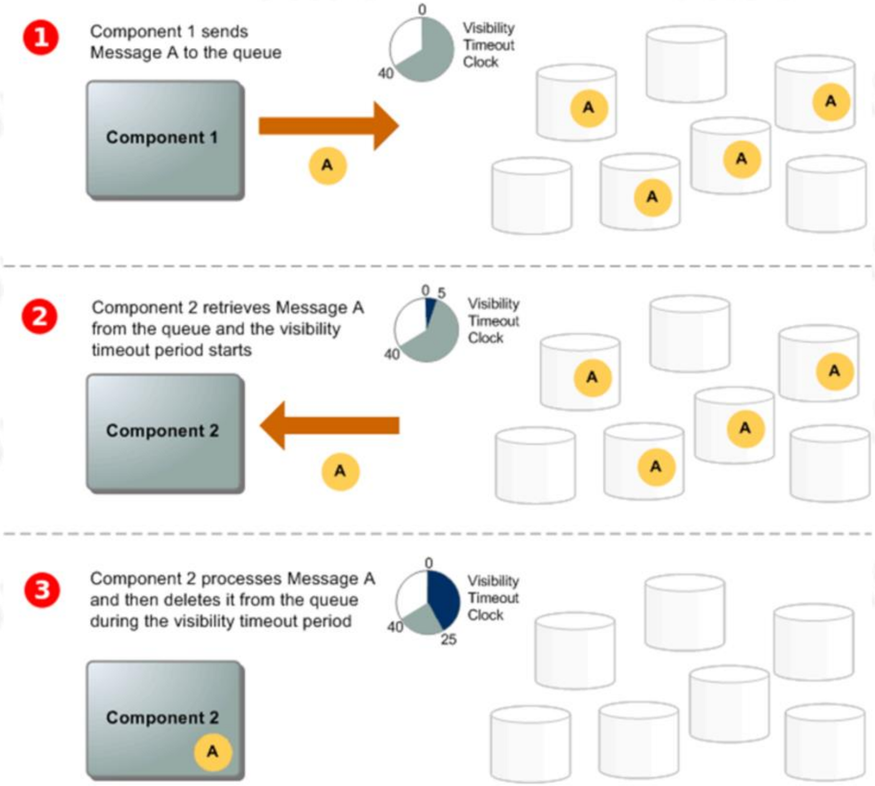
04 Using SQS FIFO queues

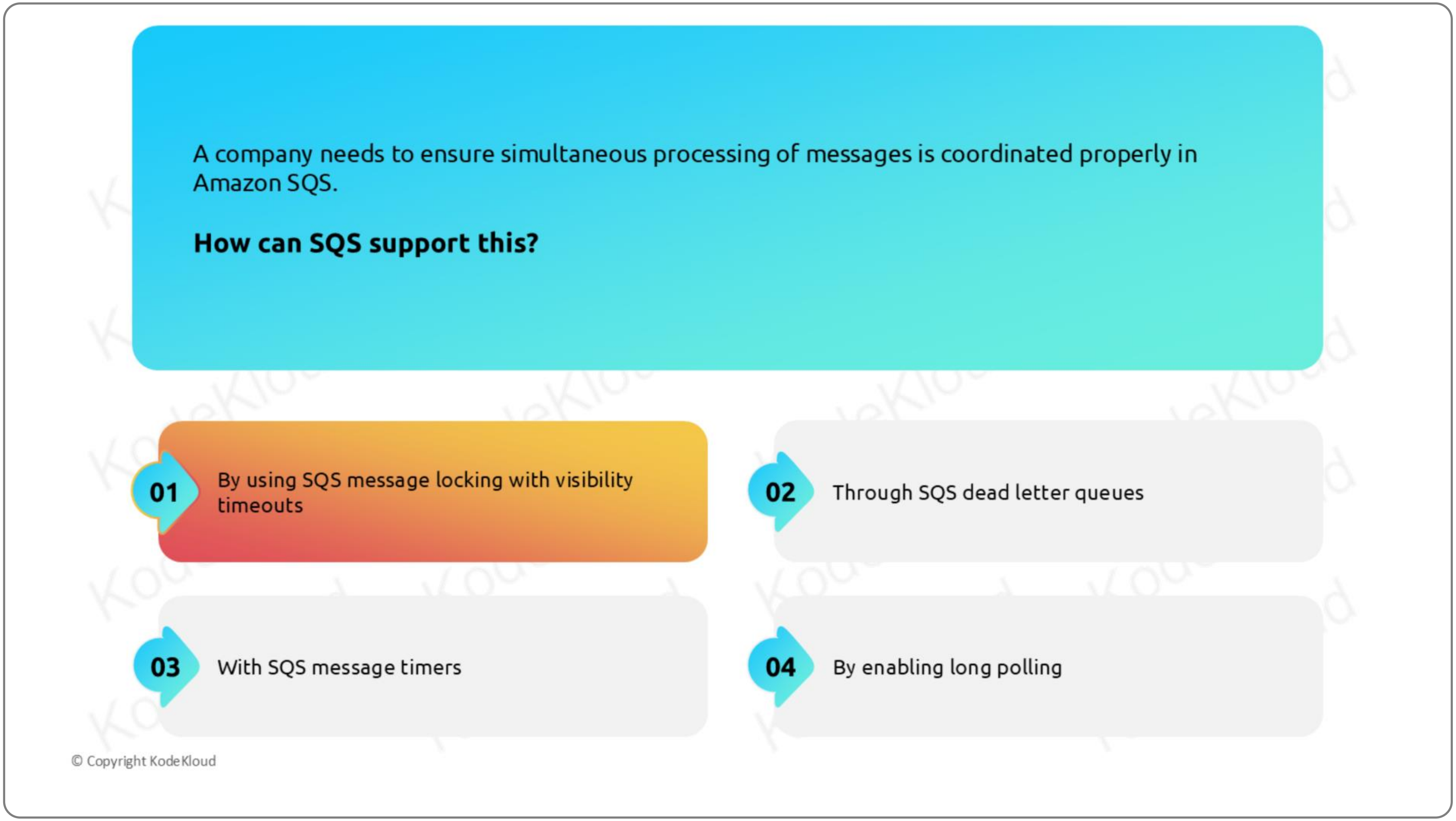
Correct Answer: A
Explanation: SQS long polling reduces empty responses by allowing longer waits for messages when polling.
Incorrect Answers:
B) Message timers set retry delays but do not reduce empty responses.
C) Dead letter queues capture failed messages but do not reduce empty responses.
D) FIFO queues order messages but do not reduce empty responses.

References:
https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-long-polling.html

Simple Queue Service

To ensure at least once processing, SQS has a locking mechanism.


1
Component 1 sends Message A to the queue
0
40
Visibility Timeout Clock
Component 1
A
2
Component 2 retrieves Message A from the queue and the visibility timeout period starts
0
5
40
Visibility Timeout Clock
Component 2
A
3
Component 2 processes Message A and then deletes it from the queue during the visibility timeout period
0
40
25
Visibility Timeout Clock
Component 2
A



A company needs to ensure simultaneous processing of messages is coordinated properly in Amazon SQS.

How can SQS support this?

01 By using SQS message locking with visibility timeouts

02 Through SQS dead letter queues

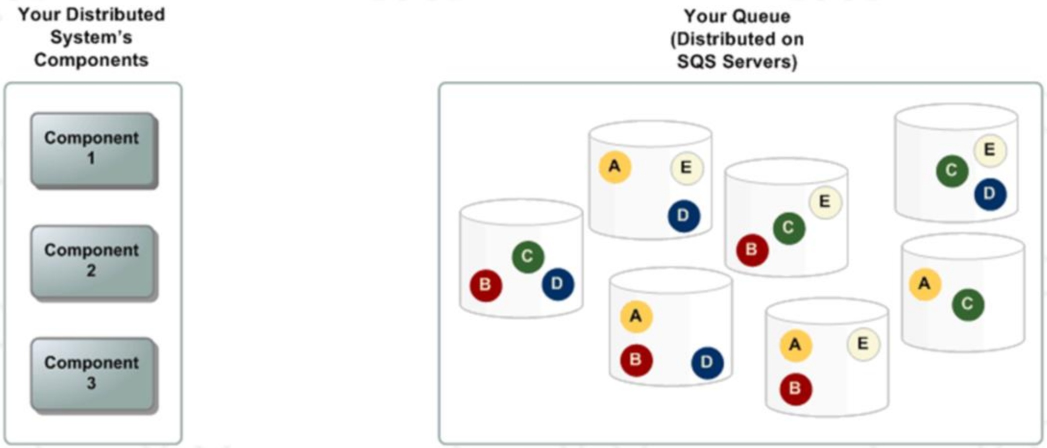
03 With SQS message timers

04 By enabling long polling

Correct Answer: A
Explanation: SQS locking with visibility timeouts helps coordinate parallel processing by locking messages during processing.
Incorrect Answers:
B) Dead letter queues capture failed messages but do not coordinate processing.
C) Message timers set retry delays but do not coordinate processing.

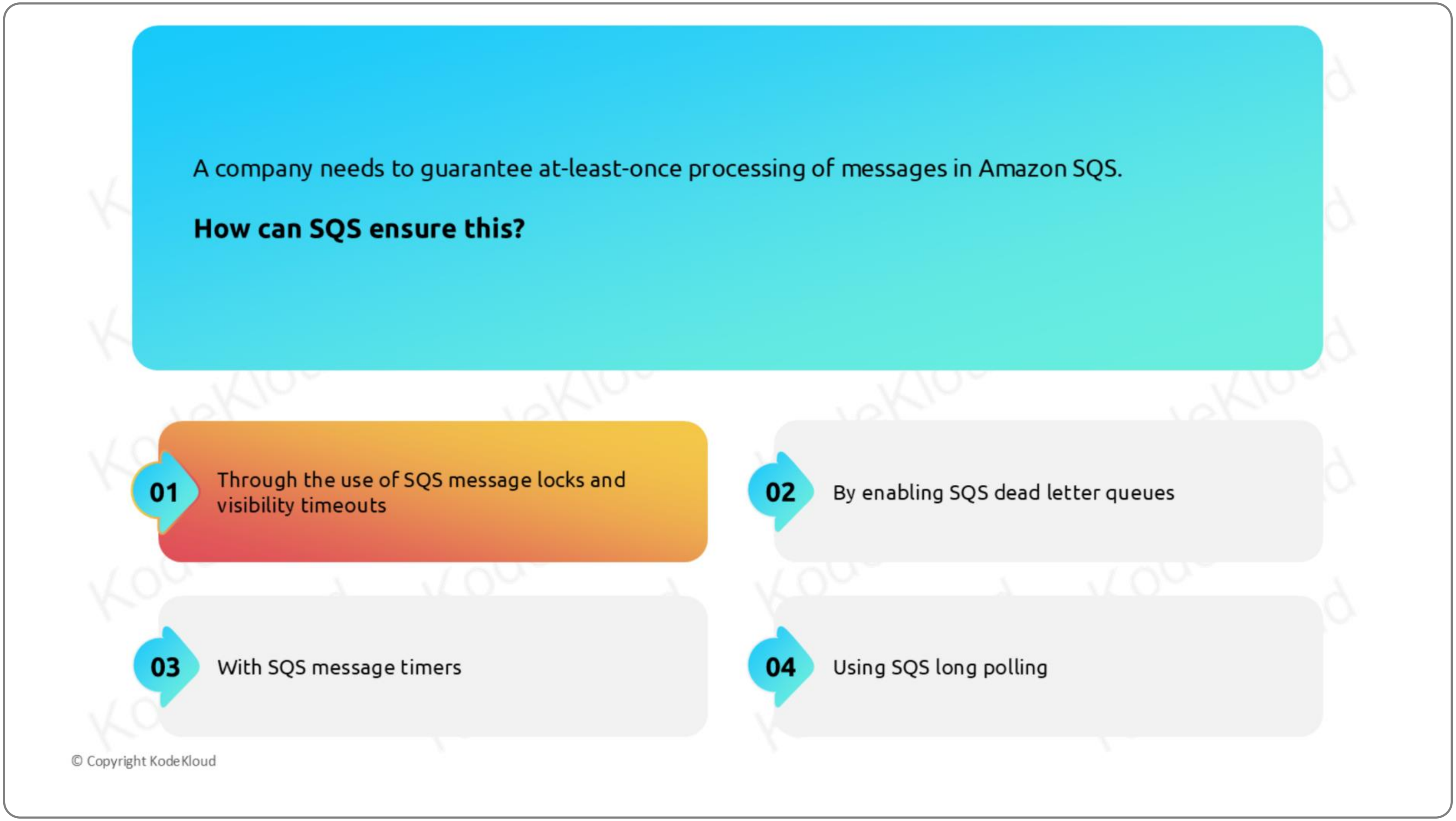
D) Long polling reduces empty responses but does not coordinate processing.
References:
https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-visibility-timeout.html

Simple Queue Service



Your Distributed System's Components
Component 1
Component 2
Component 3
Your Queue (Distributed on SQS Servers)
A
E
D
B
C

SQS has built-in scalability and uses many “hidden” servers to server the single queue you see in the system.



A company needs to guarantee at-least-once processing of messages in Amazon SQS.

How can SQS ensure this?

01 Through the use of SQS message locks and visibility timeouts

02 By enabling SQS dead letter queues

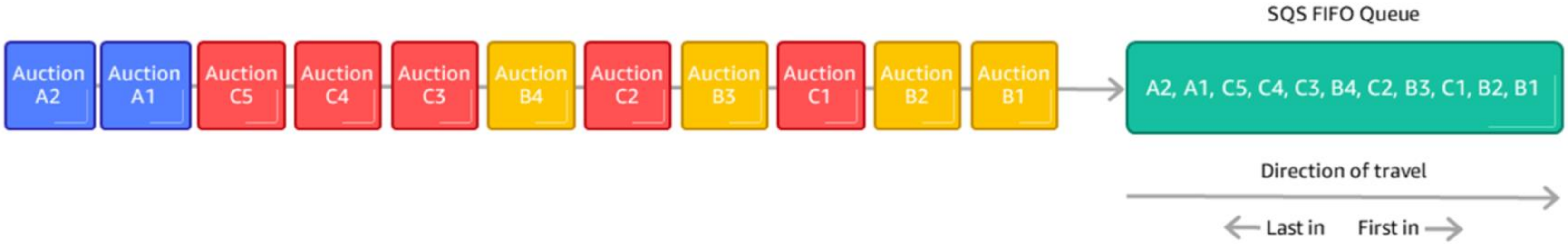
03 With SQS message timers

04 Using SQS long polling

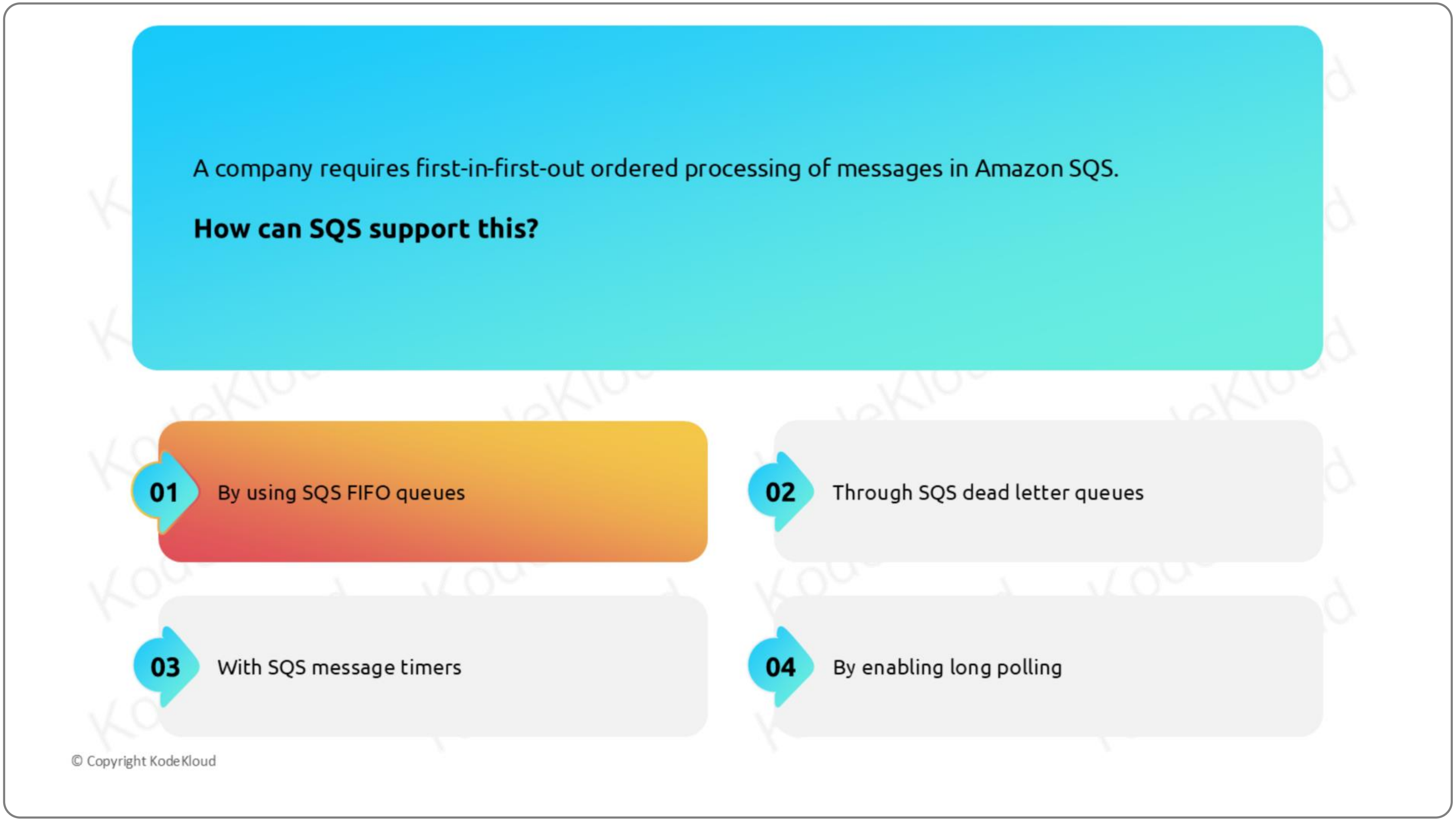
Correct Answer: A
Explanation: SQS locks and visibility timeouts guarantee at-least-once processing by preventing message loss during processing.
Incorrect Answers:
B) Dead letter queues capture failed messages but do not ensure at-least-once delivery.
C) Message timers set retry delays but do not ensure at-least-once delivery.

D) Long polling reduces empty responses but does not ensure at-least-once delivery.
References:
https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-visibility-timeout.html

Simple Queue Service



You can also enable First In First Out if necessary to increase ordering reliability.



A company requires first-in-first-out ordered processing of messages in Amazon SQS.

How can SQS support this?

01 By using SQS FIFO queues

02 Through SQS dead letter queues

03 With SQS message timers

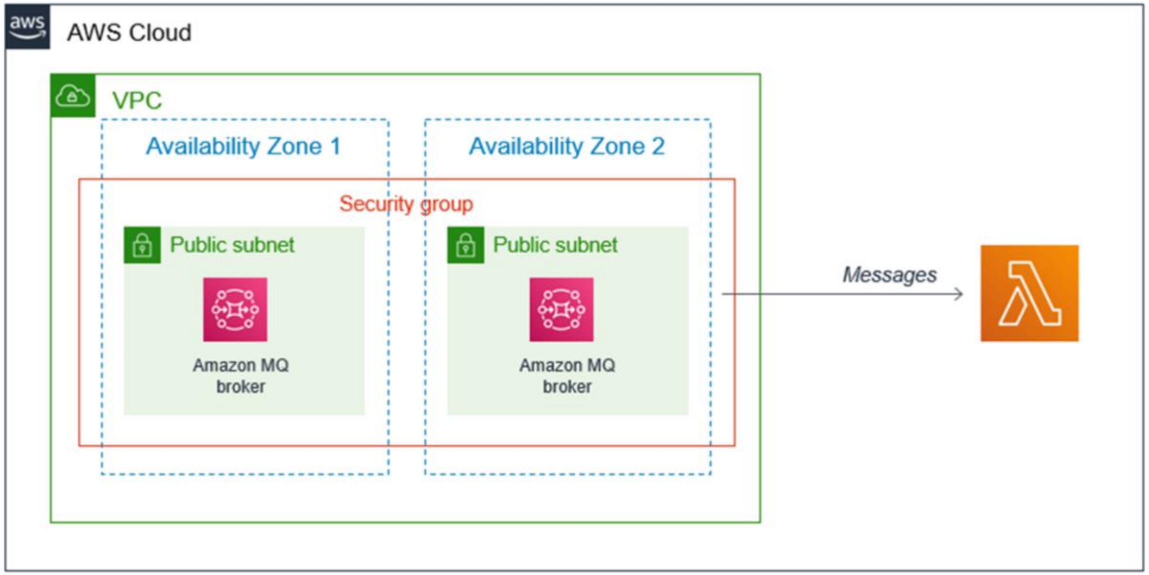
04 By enabling long polling

Correct Answer: A
Explanation: SQS FIFO queues provide strict first-in-first-out ordering of messages.
Incorrect Answers:
B) Dead letter queues capture failed messages but do not order messages.
C) Message timers set retry delays but do not order messages.
D) Long polling reduces empty responses but does not order messages.

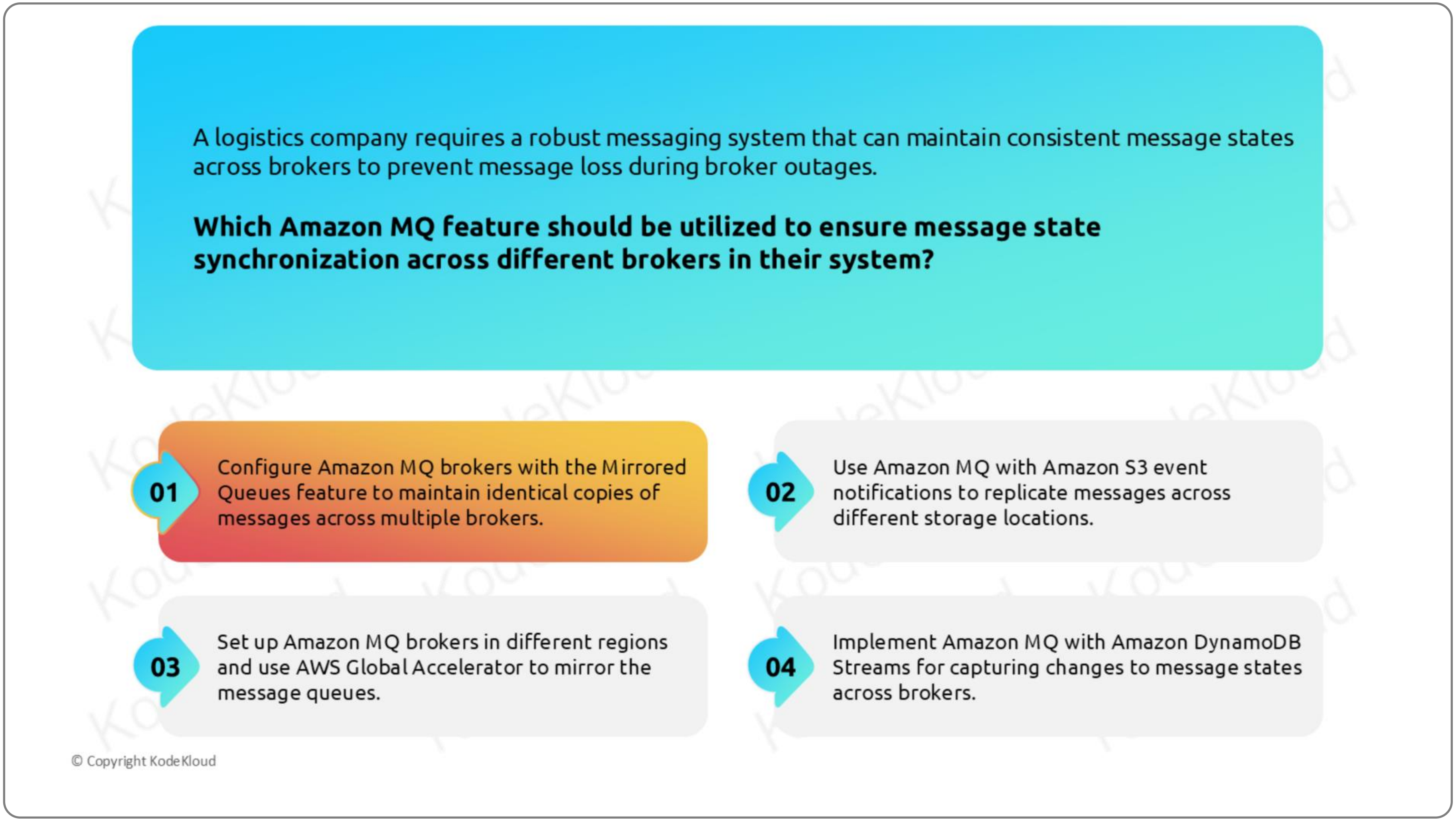
References:
https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/fifo-queues.html

Designing for Reliability – Amazon MQ

Amazon MQ



ActiveMQ achieves reliability mainly through Active/Passive setups for multiple machines called Mirror Queues.



A logistics company requires a robust messaging system that can maintain consistent message states across brokers to prevent message loss during broker outages.

Which Amazon MQ feature should be utilized to ensure message state synchronization across different brokers in their system?

01 Configure Amazon MQ brokers with the Mirrored Queues feature to maintain identical copies of messages across multiple brokers.

02 Use Amazon MQ with Amazon S3 event notifications to replicate messages across different storage locations.

03 Set up Amazon MQ brokers in different regions and use AWS Global Accelerator to mirror the message queues.

04 Implement Amazon MQ with Amazon DynamoDB Streams for capturing changes to message states across brokers.

Correct Answer: A
Explanation:
Amazon MQ's Mirrored Queues feature is designed to synchronize the state of messages across a pair of brokers. When this feature is enabled, it ensures that messages are mirrored across active and standby brokers, providing high availability and preventing message loss in case of broker outages. This feature is crucial for the logistics company's messaging system, as it ensures that even during failovers, the message state is consistent and no messages are lost, which is

essential for tracking logistics operations reliably.

Incorrect Answers:

B) Amazon S3 event notifications are not used for message replication across brokers; they are designed for integrating S3 events with other AWS services.

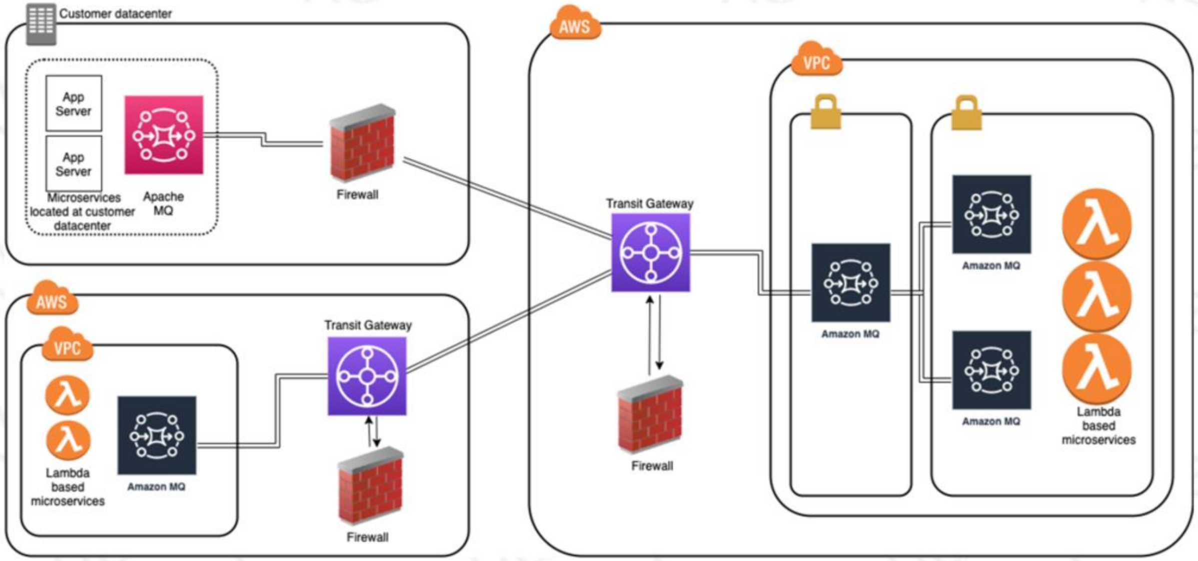
C) AWS Global Accelerator improves global application performance but does not mirror message queues across Amazon MQ brokers.

D) Amazon DynamoDB Streams captures changes to DynamoDB tables, not Amazon MQ message states, and therefore would not be suitable for this purpose.

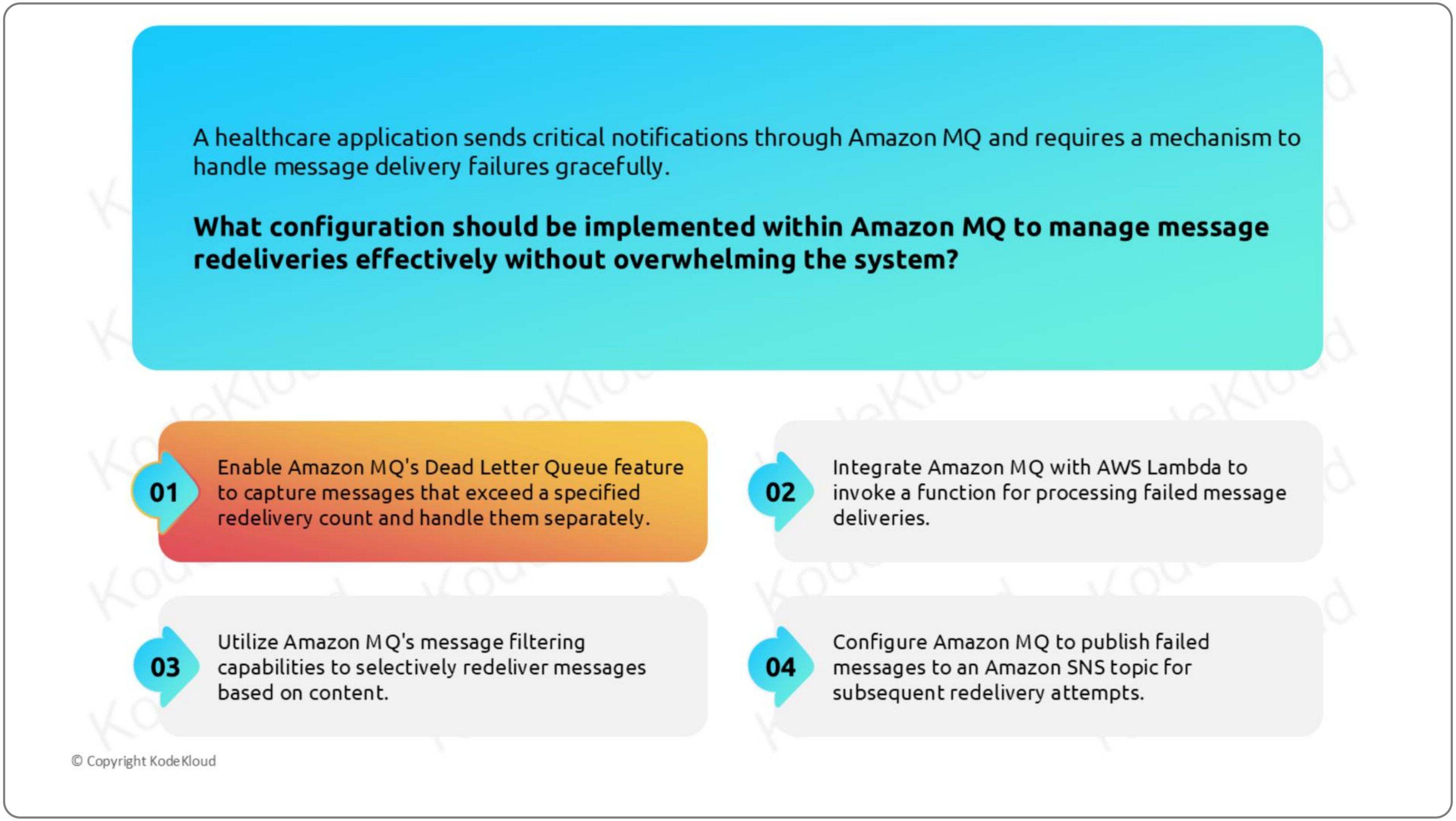
References:

Amazon MQ Broker Architecture

Amazon MQ



Reliability for Amazon MQ can also be increased by Redelivery Policies as well as Priority Dispatch Policies for higher priority items.



A healthcare application sends critical notifications through Amazon MQ and requires a mechanism to handle message delivery failures gracefully.

What configuration should be implemented within Amazon MQ to manage message redeliveries effectively without overwhelming the system?

01 Enable Amazon MQ's Dead Letter Queue feature to capture messages that exceed a specified redelivery count and handle them separately.

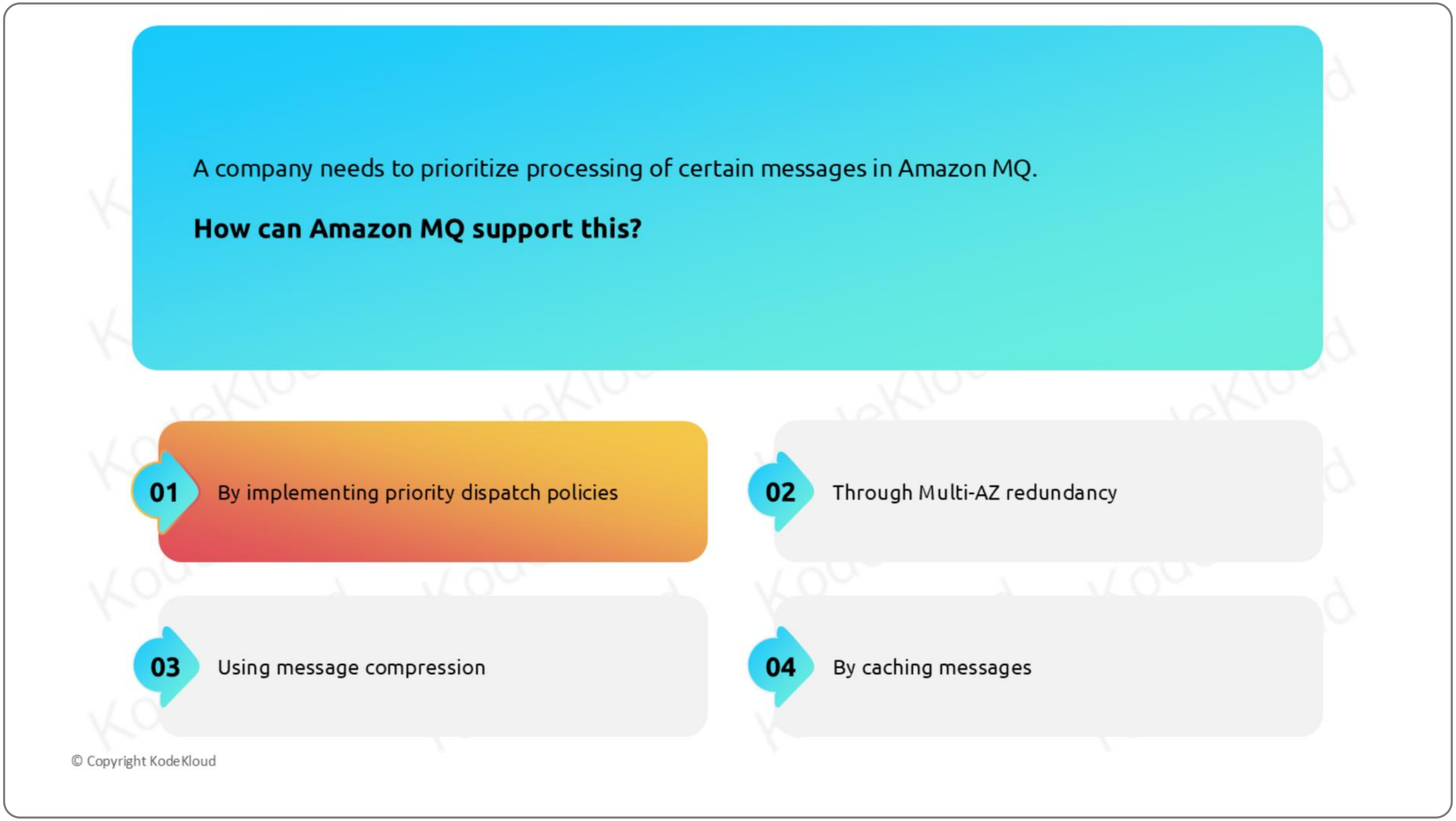
02 Integrate Amazon MQ with AWS Lambda to invoke a function for processing failed message deliveries.

03 Utilize Amazon MQ's message filtering capabilities to selectively redeliver messages based on content.

04 Configure Amazon MQ to publish failed messages to an Amazon SNS topic for subsequent redelivery attempts.

Correct Answer: A
Explanation:
Amazon MQ's Dead Letter Queue (DLQ) feature allows messages that cannot be delivered successfully after a certain number of redelivery attempts (as specified by the redelivery policy) to be sent to a separate dead-letter queue. This mechanism enables the application to handle message delivery failures systematically and ensures that messages that cannot be processed are not lost but can be reviewed and handled appropriately. This is particularly important in

healthcare applications where critical notifications must not be missed, and system integrity is paramount.
Incorrect Answers:
B) While AWS Lambda can be used for processing messages, it does not provide a built-in mechanism for managing redeliveries within Amazon MQ.
C) Message filtering in Amazon MQ is used to selectively receive messages based on criteria but does not manage the redelivery of failed messages.
D) Publishing failed messages to an Amazon SNS topic is not a native feature of Amazon MQ for handling message redelivery.
References:
Amazon MQ Redelivery Policy



A company needs to prioritize processing of certain messages in Amazon MQ.

How can Amazon MQ support this?

01 By implementing priority dispatch policies

02 Through Multi-AZ redundancy

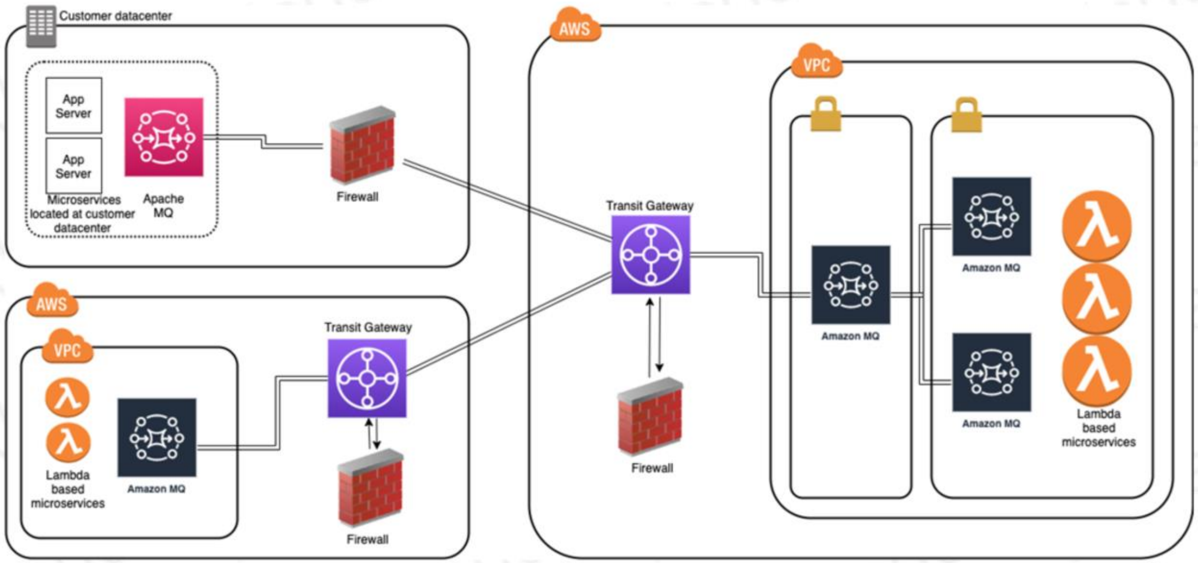
03 Using message compression

04 By caching messages

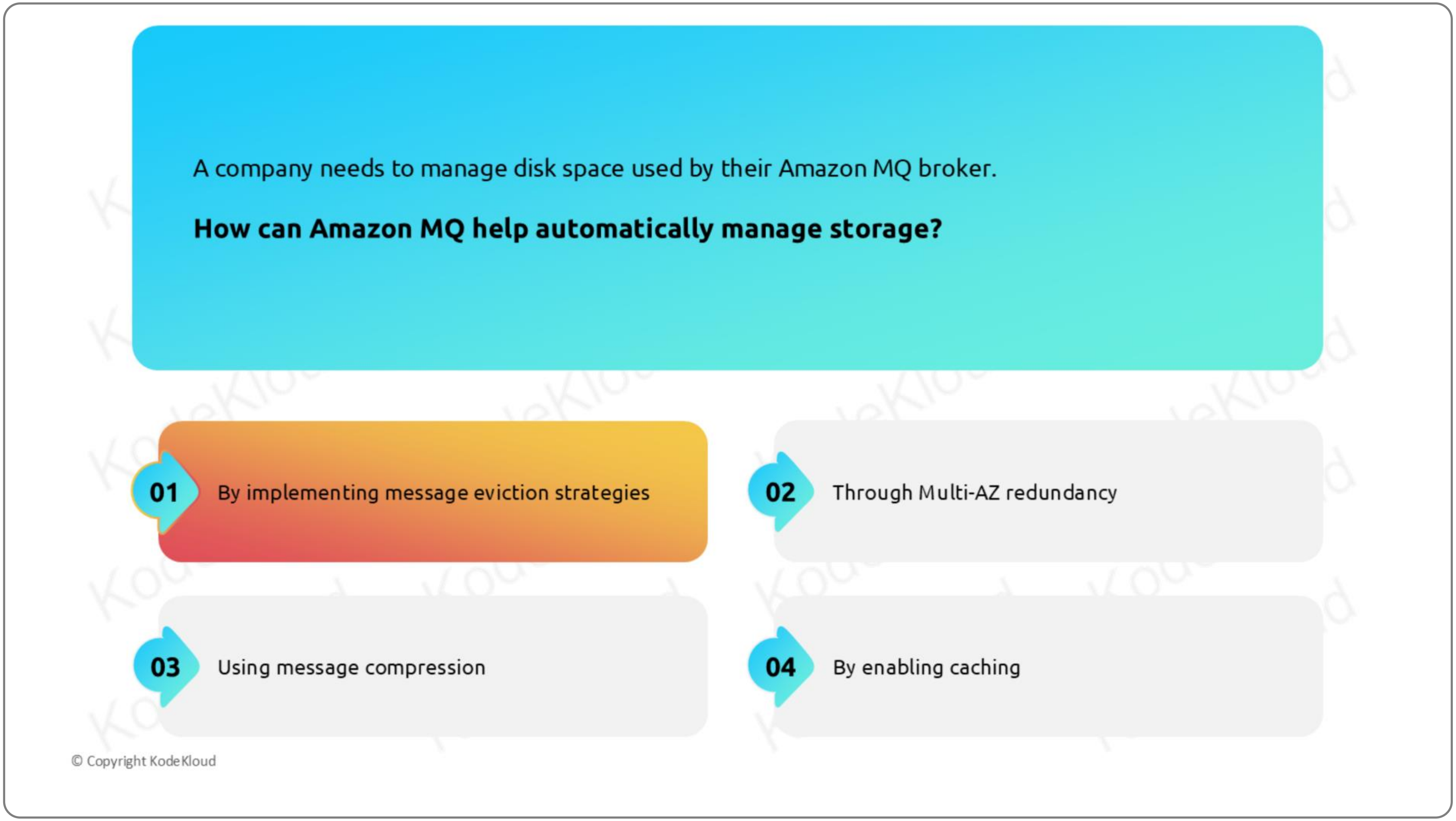
Correct Answer: A
Explanation: Amazon MQ priority dispatch policies allow setting priority levels on messages to control order of processing.
Incorrect Answers:
B) Multi-AZ provides high availability but does not prioritize message dispatch order.
C) Message compression reduces payload size but does not enable priority processing.
D) Caching improves performance but does not enable priority processing.

References:
https://docs.aws.amazon.com/amazon-mq/latest/developer-guide/amazon-mq-configuring-wire-level.html#priority-dispatch-policy

Amazon MQ



Reliability for Amazon MQ also has message eviction strategies like Oldest message first or low priority last that also ensures availability for messages when storage is constrained.



A company needs to manage disk space used by their Amazon MQ broker.

How can Amazon MQ help automatically manage storage?

01 By implementing message eviction strategies

02 Through Multi-AZ redundancy

03 Using message compression

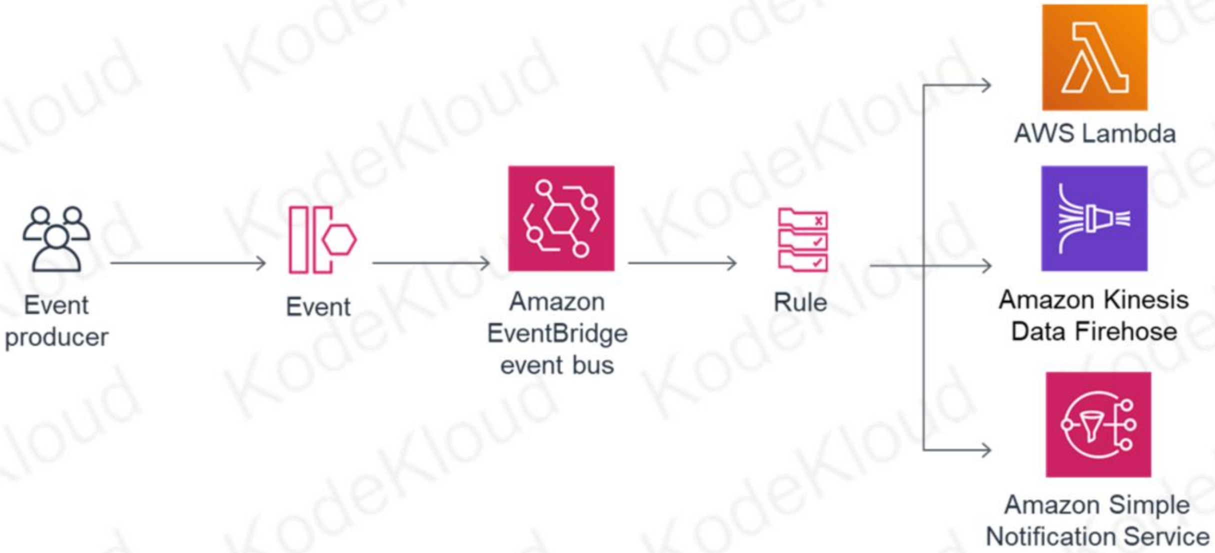
04 By enabling caching

Correct Answer: A
Explanation: Amazon MQ supports message eviction strategies like First In First Out (FIFO) and Last In First Out (LIFO) to automatically remove messages from storage when limits are reached.
Incorrect Answers:
B) Multi-AZ provides high availability but does not manage message storage.
C) Message compression reduces payload size but does not automatically remove messages.

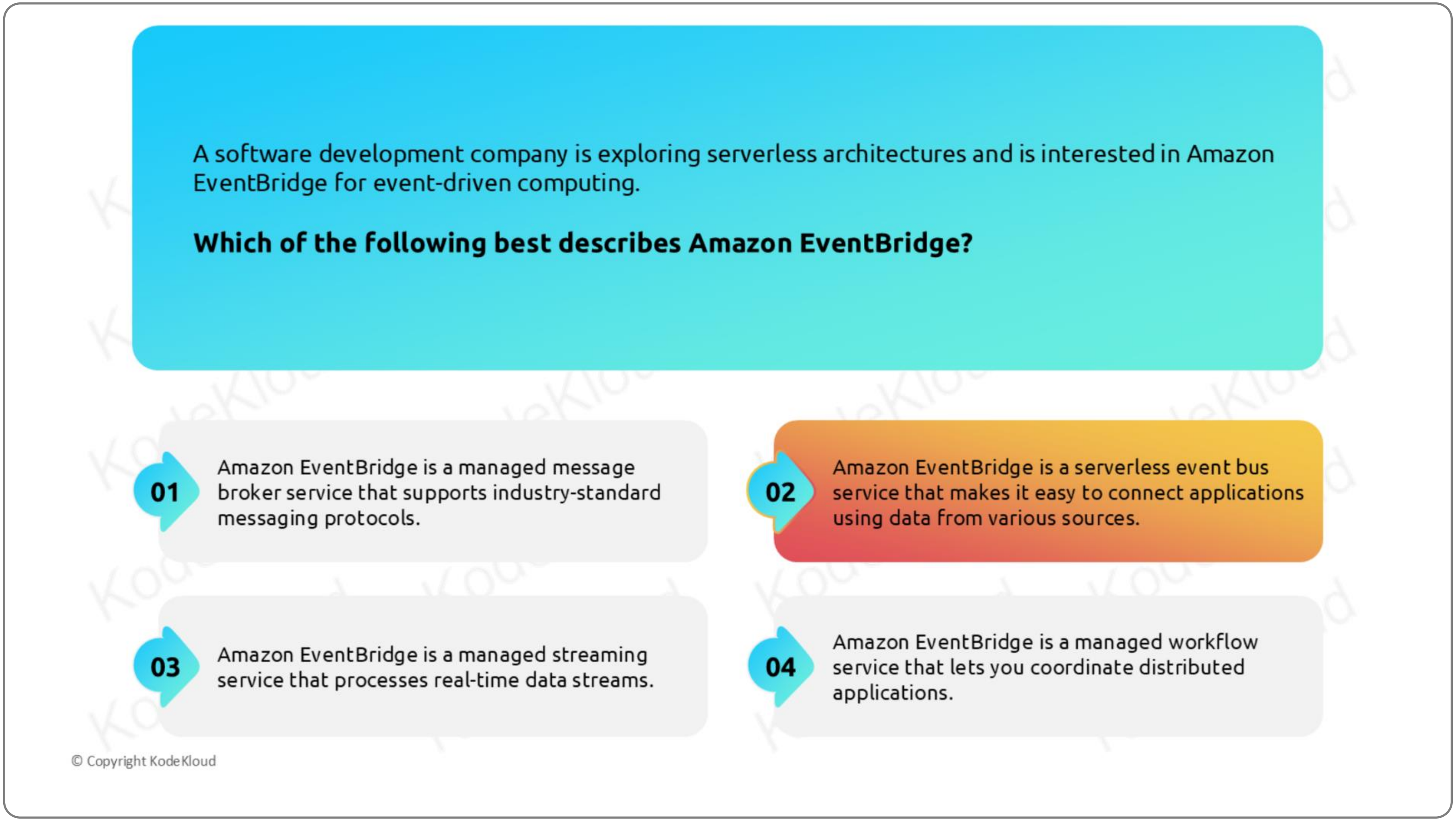
D) Caching improves performance but does not manage broker disk space.
References:
https://docs.aws.amazon.com/amazon-mq/latest/developer-guide/amazon-mq-broker-config-eviction.html

Designing for Reliability – Amazon EventBridge

Amazon EventBridge


Event producer
Event
Amazon EventBridge event bus
Rule
AWS Lambda
Amazon Kinesis Data Firehose
Amazon Simple Notification Service

EventBridge is highly managed, so everything is reliable by default as it is "hidden"; there are some knobs to turn though.



A software development company is exploring serverless architectures and is interested in Amazon EventBridge for event-driven computing.

Which of the following best describes Amazon EventBridge?

01 Amazon EventBridge is a managed message broker service that supports industry-standard messaging protocols.

02 Amazon EventBridge is a serverless event bus service that makes it easy to connect applications using data from various sources.

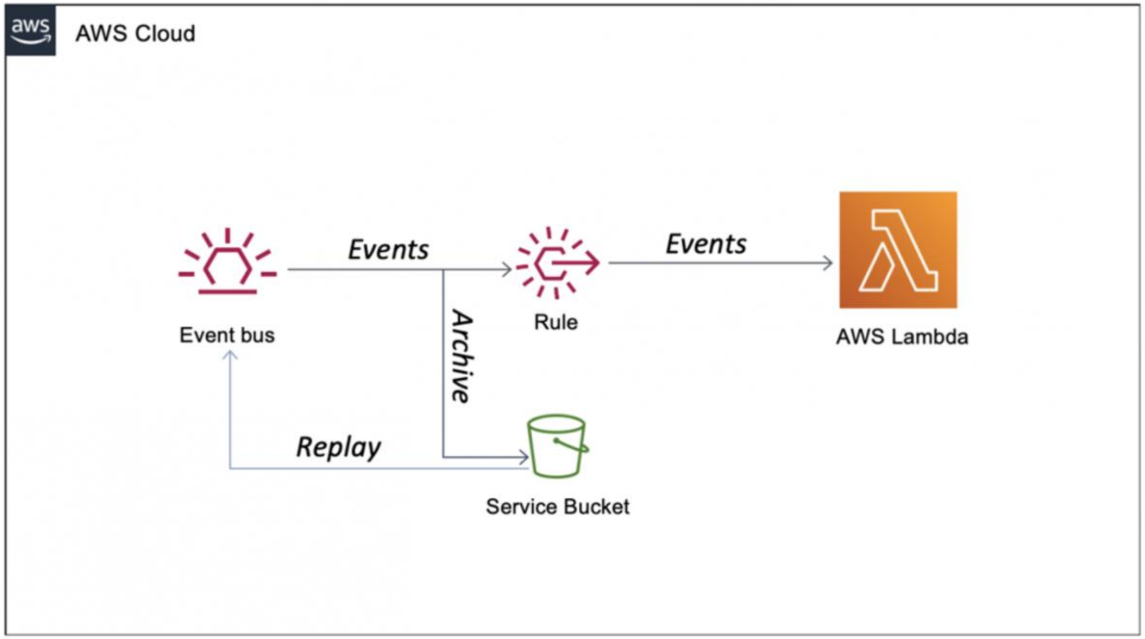
03 Amazon EventBridge is a managed streaming service that processes real-time data streams.

04 Amazon EventBridge is a managed workflow service that lets you coordinate distributed applications.

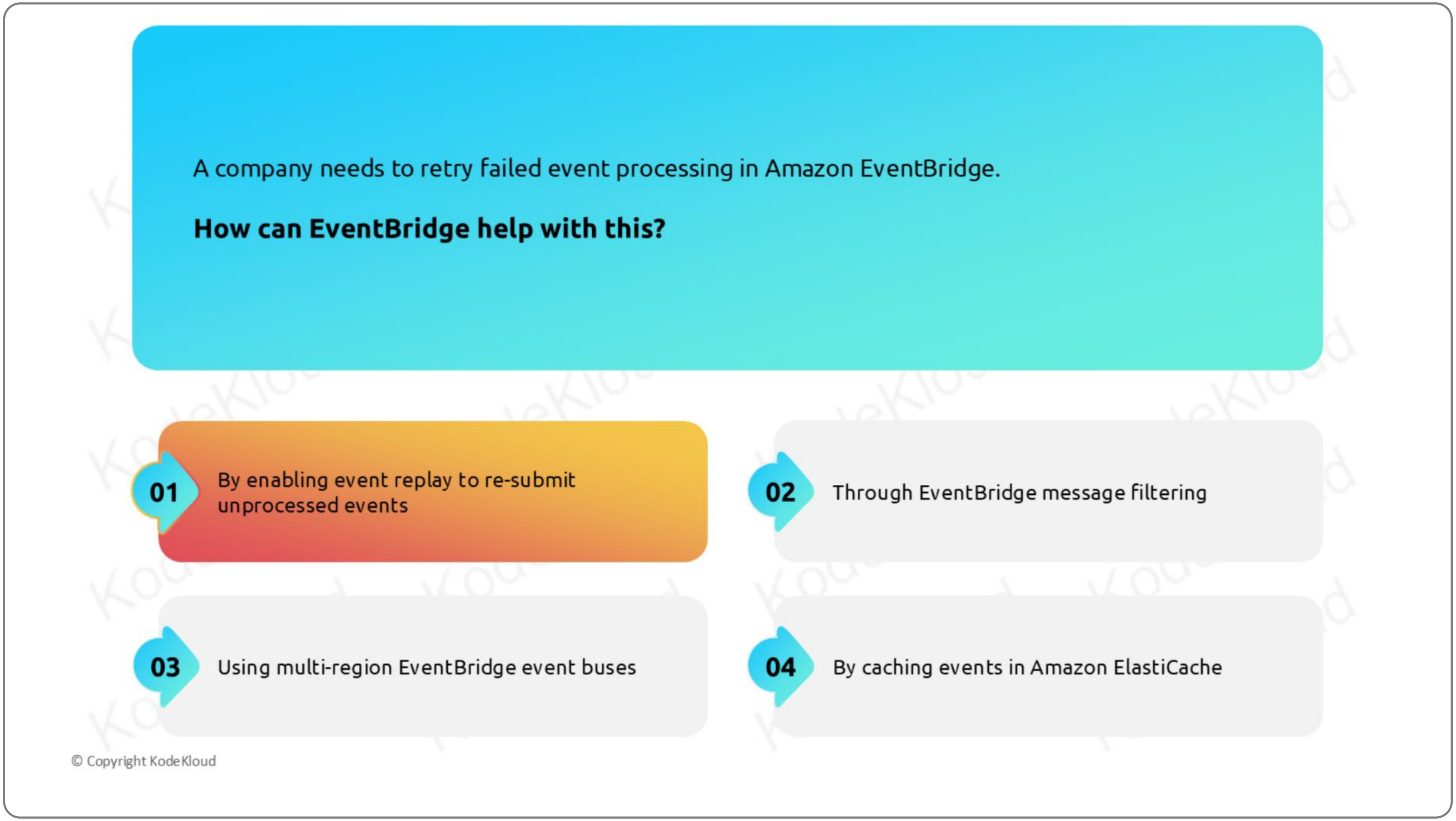
Correct Answer: B. Amazon EventBridge is a serverless event bus service that makes it easy to connect applications using data from various sources.
Incorrect: A. This describes Amazon MQ. C. This describes Amazon Kinesis. D. This describes AWS Step Functions.
Explanation: Amazon EventBridge is a serverless event bus that makes it easy to ingest, filter, and deliver events from various sources to different AWS services and on-premises applications. With EventBridge, developers can build event-driven architectures that react in real-time to data from custom applications, integrated Software-as-a-Service (SaaS)

applications, and AWS services.
References: Amazon EventBridge: What is Amazon EventBridge?

Amazon EventBridge



Event replay is one



A company needs to retry failed event processing in Amazon EventBridge.

How can EventBridge help with this?

01 By enabling event replay to re-submit unprocessed events

02 Through EventBridge message filtering

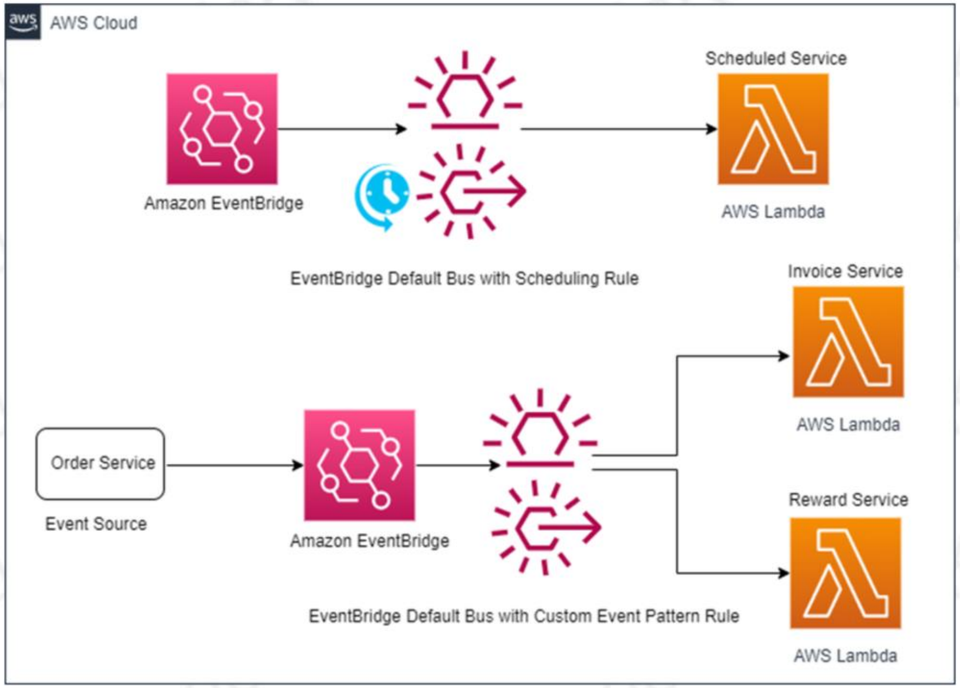
03 Using multi-region EventBridge event buses

04 By caching events in Amazon ElastiCache

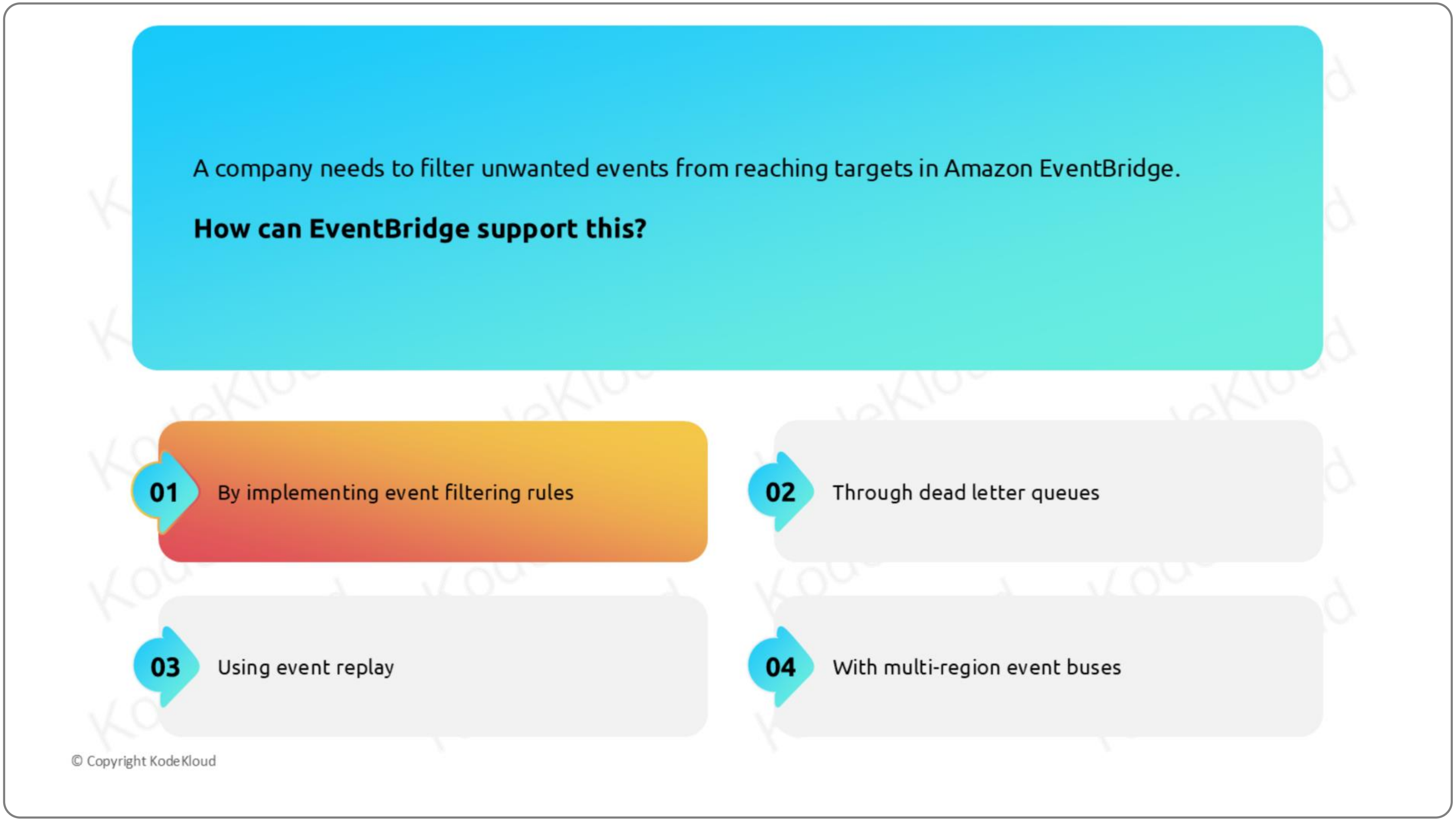
Correct Answer: A
Explanation: Amazon EventBridge event replay automatically retries failed events, providing at-least-once event delivery semantics.
Incorrect Answers:
B) Message filtering drops unwanted events but does not retry failed events.
C) Multi-region event buses provide cross-region redundancy but do not retry failed events.

D) Caching events in ElastiCache improves performance but does not enable retries.
References:
https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-replay-events.html

Amazon EventBridge



Retry Policies and Event Filters



A company needs to filter unwanted events from reaching targets in Amazon EventBridge.

How can EventBridge support this?

01 By implementing event filtering rules

02 Through dead letter queues

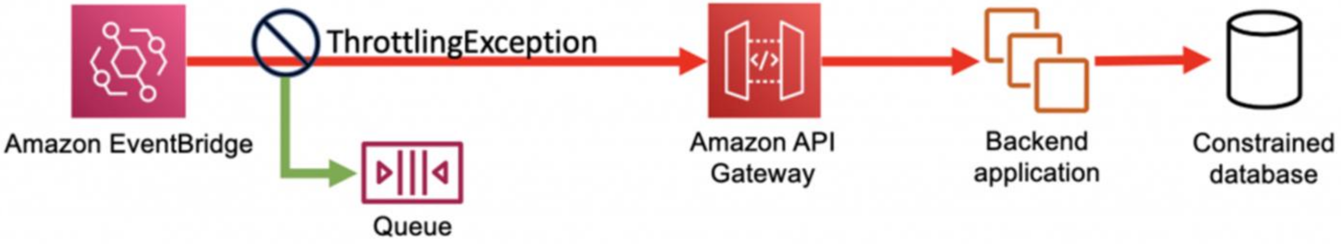
03 Using event replay

04 With multi-region event buses

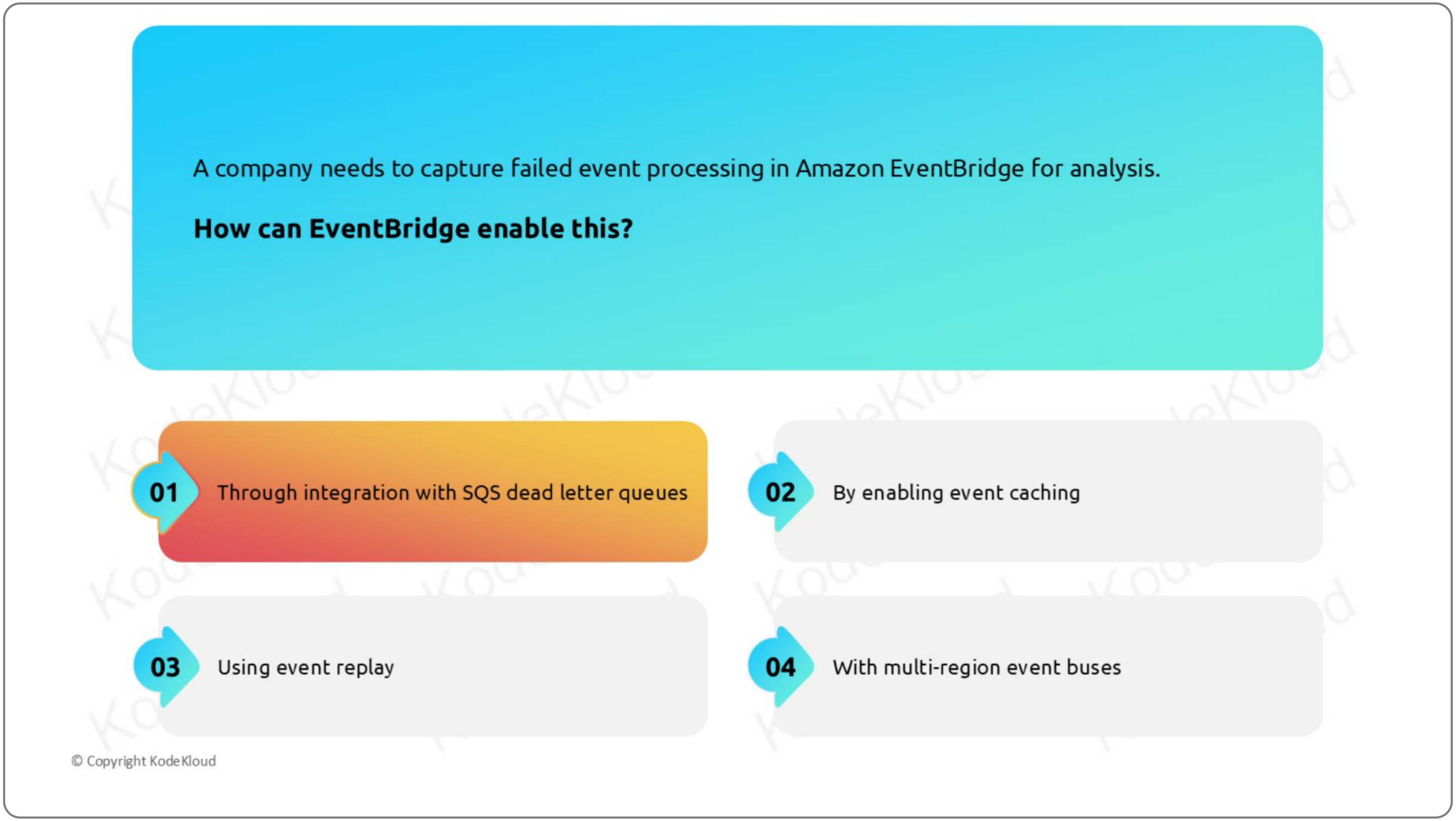
Correct Answer: A
Explanation: Amazon EventBridge event filtering allows specifying rules to filter unwanted events before they reach targets.
Incorrect Answers:
B) Dead letter queues capture failed events but do not filter events.
C) Event replay retries failed events but does not filter events.

D) Multi-region event buses provide redundancy but do not filter events.
References:
https://docs.aws.amazon.com/eventbridge/latest/userguide/filter-events.html

Amazon EventBridge



Dead Letter Queues are another for retry and catchall.



A company needs to capture failed event processing in Amazon EventBridge for analysis.

How can EventBridge enable this?

01 Through integration with SQS dead letter queues

02 By enabling event caching

03 Using event replay

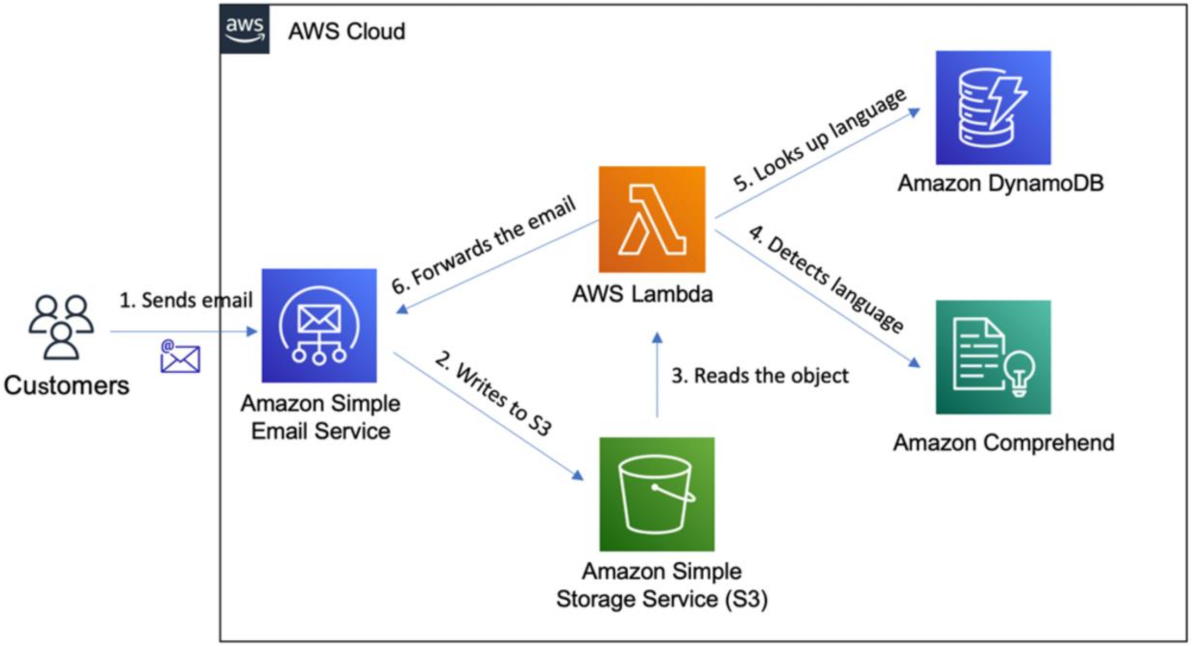
04 With multi-region event buses

Correct Answer: A
Explanation: Amazon EventBridge can integrate with SQS dead letter queues to capture failed events for troubleshooting.
Incorrect Answers:
B) Caching improves performance but does not capture failed events.
C) Event replay retries failed events but does not capture them.
D) Multi-region event buses provide redundancy but do not capture failed events.

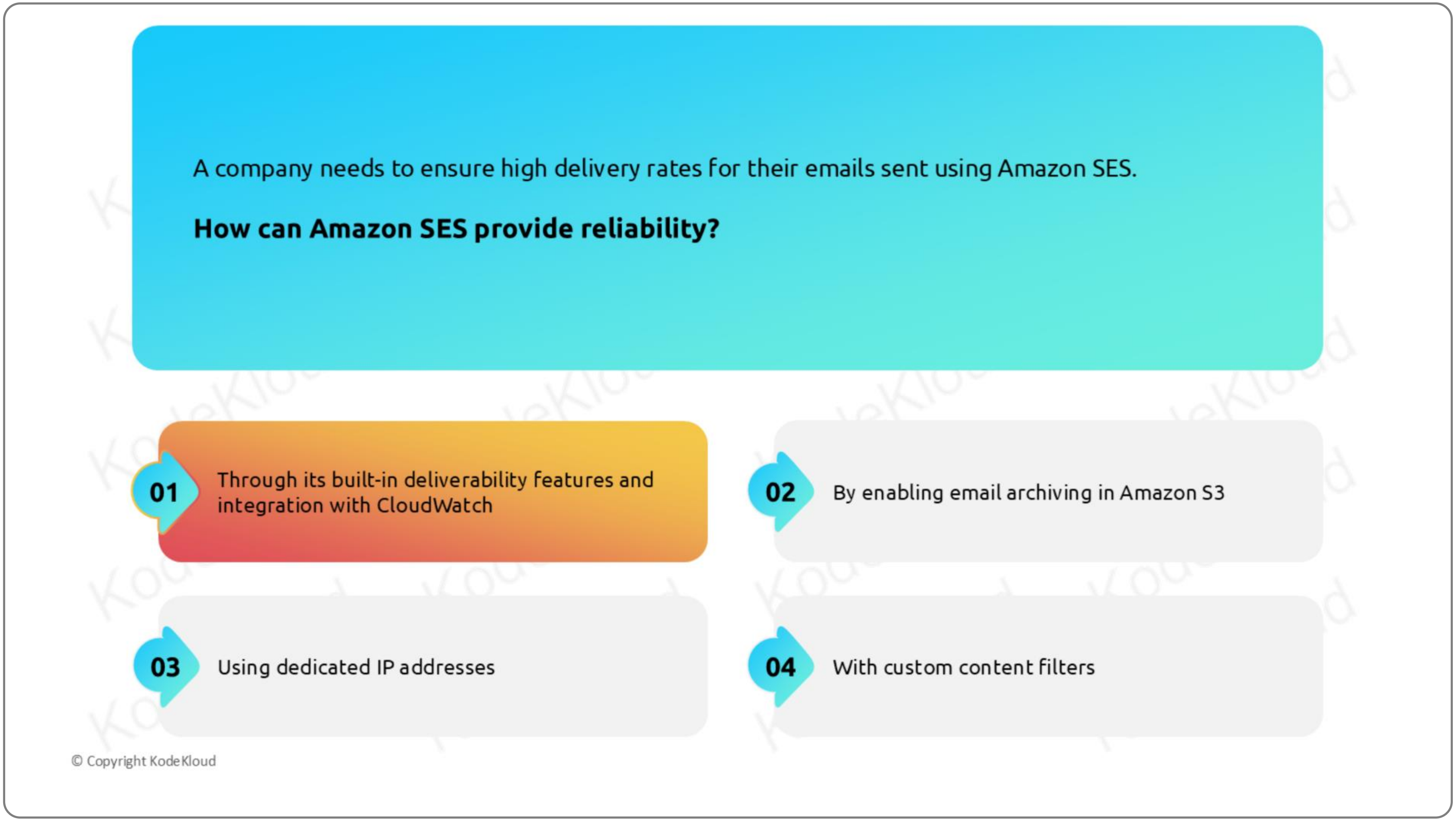
References:
https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-dlq.html

Designing for Reliability – Simple Email Service

Simple Email Service



The Simple Email Service (or SES) is highly managed and thus highly redundant.



A company needs to ensure high delivery rates for their emails sent using Amazon SES.

How can Amazon SES provide reliability?

01 Through its built-in deliverability features and integration with CloudWatch

02 By enabling email archiving in Amazon S3

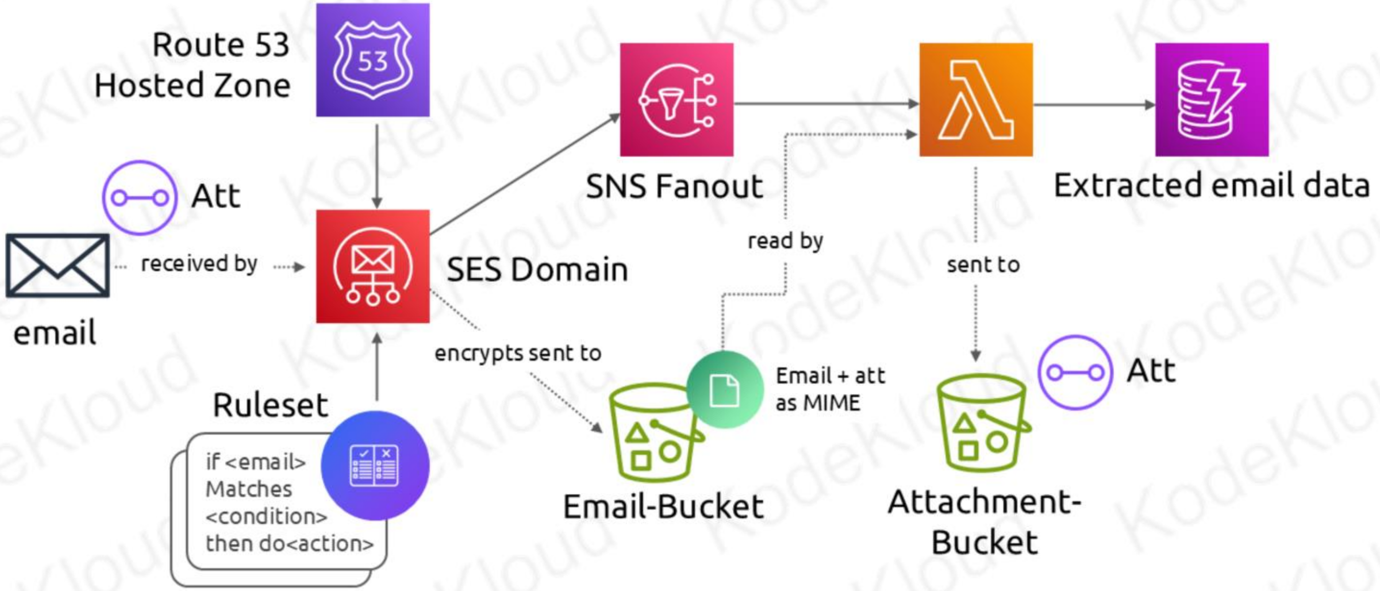
03 Using dedicated IP addresses

04 With custom content filters

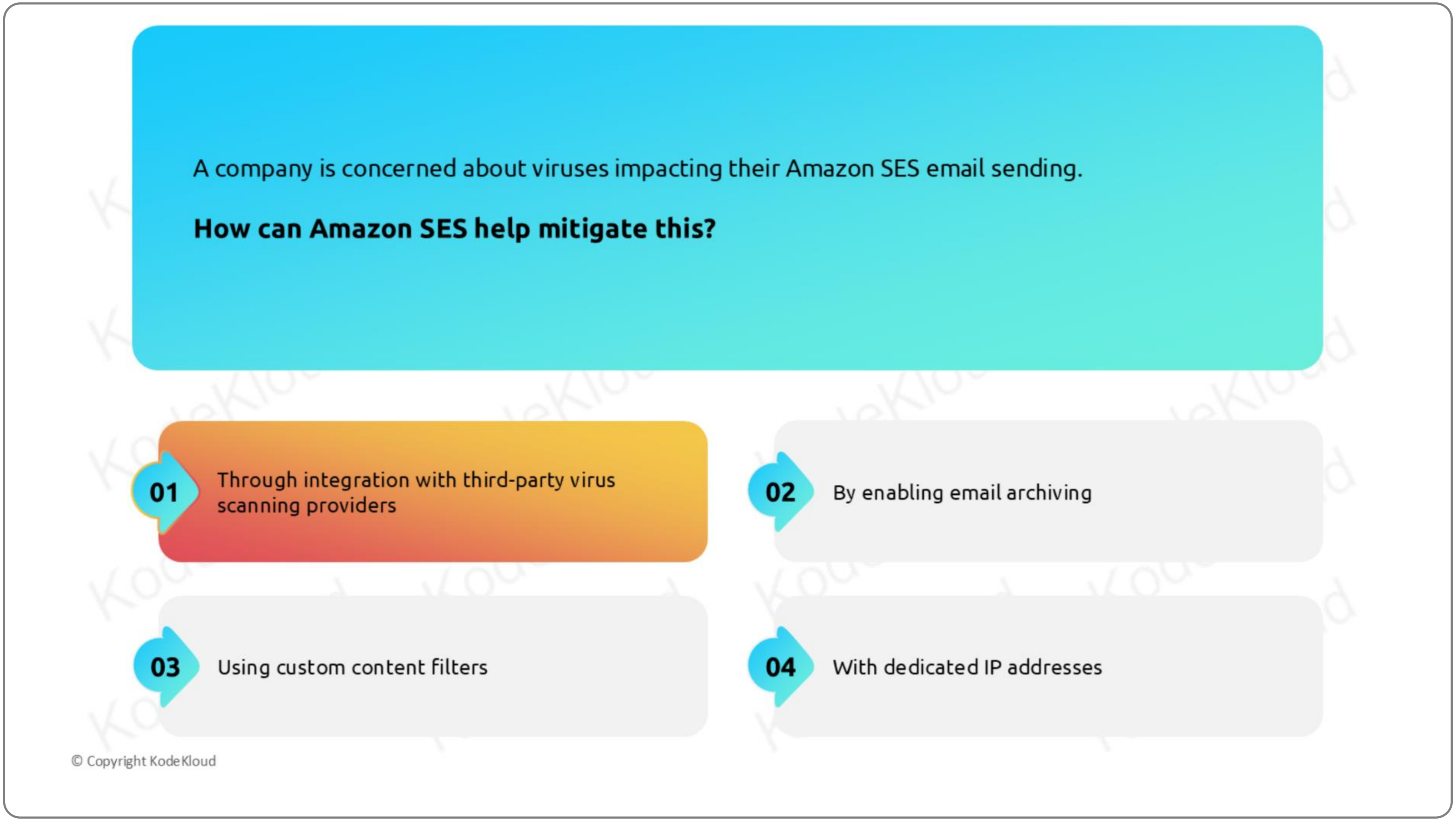
Correct Answer: A
Explanation: Amazon SES optimizes deliverability through automatic retries, reputation monitoring, and integration with CloudWatch metrics for visibility.
Incorrect Answers:
B) Archiving emails in S3 provides persistence but does not improve deliverability.
C) Dedicated IP addresses help with reputation but do not directly provide reliability.

D) Custom content filters screen emails but do not improve deliverability.
References:
https://aws.amazon.com/ses/features/deliverability/
https://docs.aws.amazon.com/ses/latest/DeveloperGuide/monitor-sending-activity.html

Simple Email Service



SES supports inline virus scanning and leverages AWS' customer experience sending out large numbers of emails.



A company is concerned about viruses impacting their Amazon SES email sending.

How can Amazon SES help mitigate this?

01 Through integration with third-party virus scanning providers

02 By enabling email archiving

03 Using custom content filters

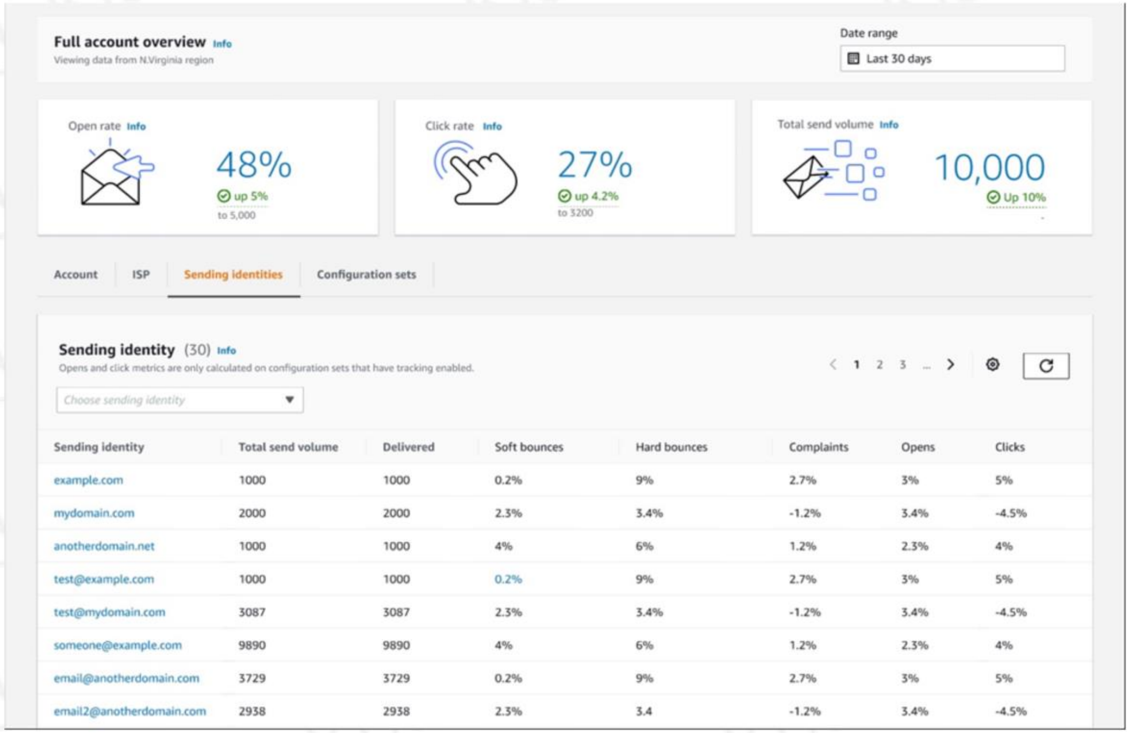
04 With dedicated IP addresses

Correct Answer: A
Explanation: Amazon SES can integrate with third-party virus scanning services to scan emails before sending.
Incorrect Answers:
B) Email archiving provides persistence but does not scan for viruses.
C) Custom content filters screen email content but do not scan for viruses.
D) Dedicated IP addresses help with deliverability but do not scan emails.

References:
https://docs.aws.amazon.com/ses/latest/DeveloperGuide/pre-sending-actions.html
https://aws.amazon.com/blogs/messaging-and-targeting/help-stop-the-spread-of-disease-viruses-and-malware/

Simple Email Service

SES uses the Virtual Deliverability Manager for regular infrastructure notification that could impact deliverability.



Full account overview

Viewing data from N.Virginia region

Date range: Last 30 days

Open rate: 48% — up 5% to 5,000

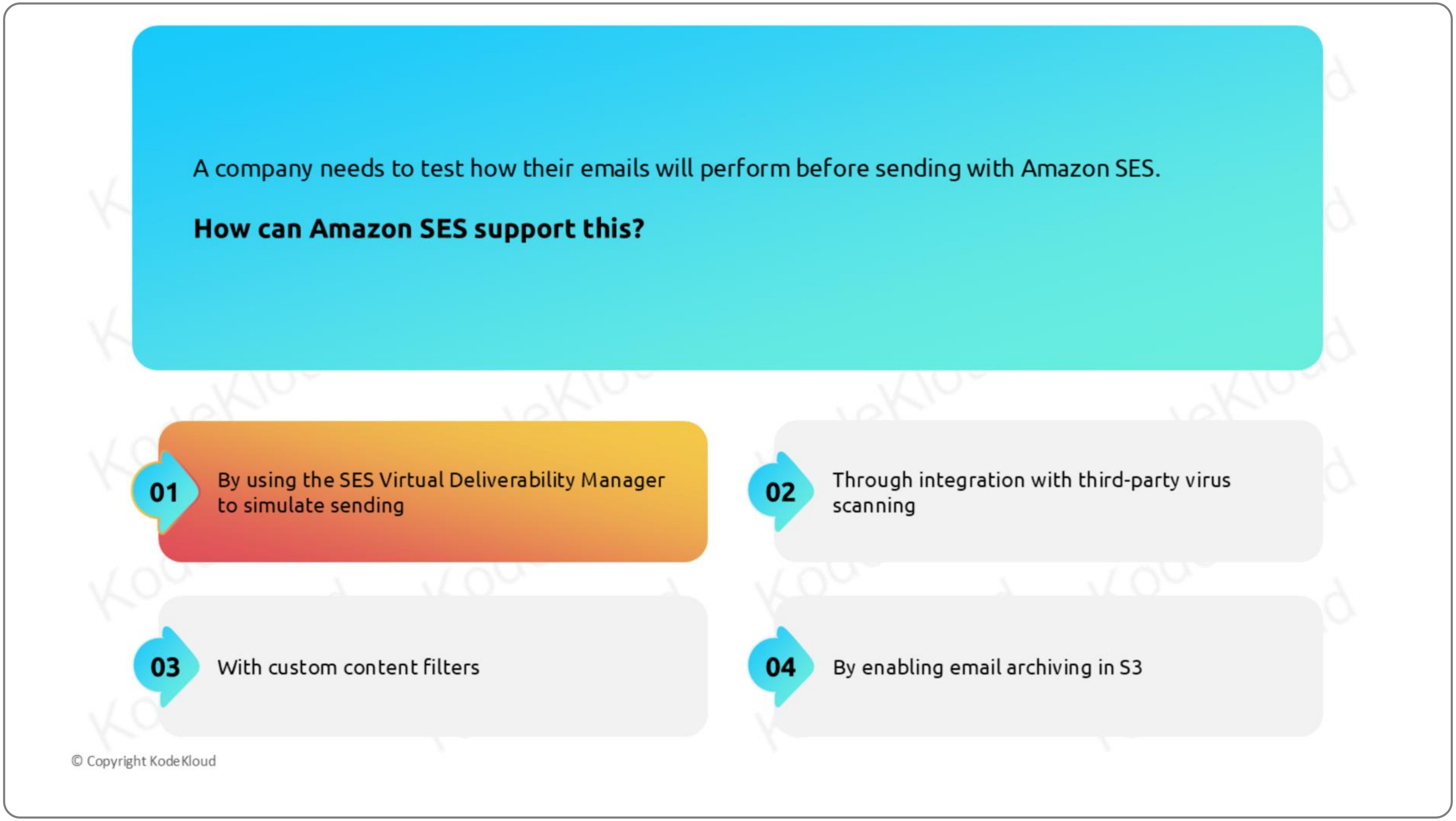
Click rate: 27% — up 4.2% to 3200

Total send volume: 10,000 — Up 10%

Sending identity (30)

Opens and click metrics are only calculated on configuration sets that have tracking enabled.

Sending identity	Total send volume	Delivered	Soft bounces	Hard bounces	Complaints	Opens	Clicks
example.com	1000	1000	0.2%	9%	2.7%	3%	5%
mydomain.com	2000	2000	2.3%	3.4%	-1.2%	3.4%	-4.5%
anotherdomain.net	1000	1000	4%	6%	1.2%	2.3%	4%
test@example.com	1000	1000	0.2%	9%	2.7%	3%	5%
test@mydomain.com	3087	3087	2.3%	3.4%	-1.2%	3.4%	-4.5%
someone@example.com	9890	9890	4%	6%	1.2%	2.3%	4%
email@anotherdomain.com	3729	3729	0.2%	9%	2.7%	3%	5%
email2@anotherdomain.com	2938	2938	2.3%	3.4	-1.2%	3.4%	-4.5%



A company needs to test how their emails will perform before sending with Amazon SES.

How can Amazon SES support this?

01 By using the SES Virtual Deliverability Manager to simulate sending

02 Through integration with third-party virus scanning

03 With custom content filters

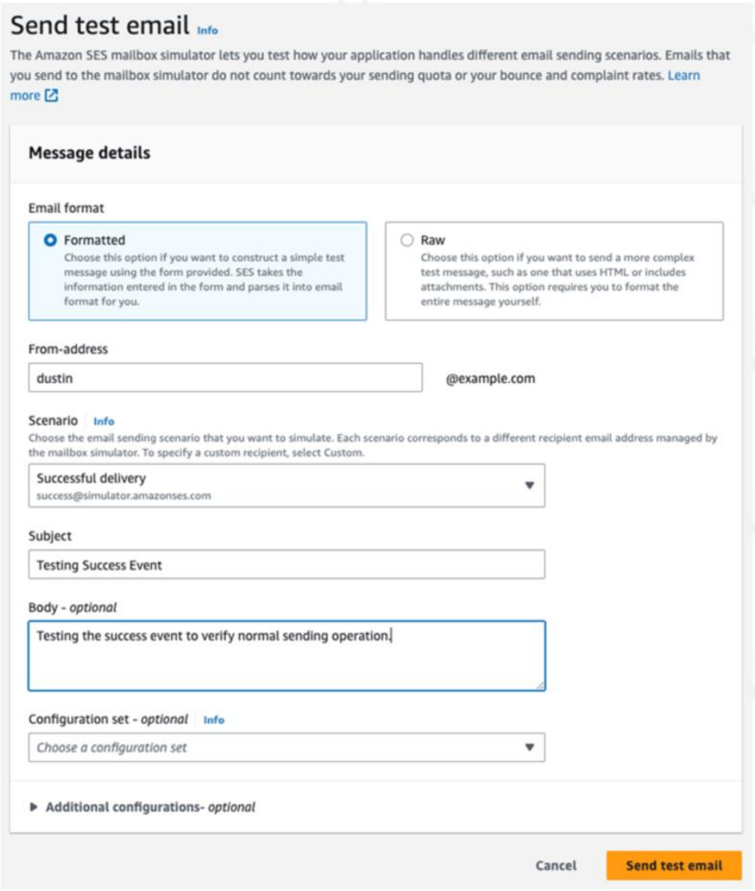
04 By enabling email archiving in S3

Correct Answer: A
Explanation: The SES Virtual Deliverability Manager allows testing deliverability by simulating sending without impacting sender reputation.
Incorrect Answers:
B) Virus scanning services scan for malware but do not test deliverability.
C) Custom content filters screen email content but do not test deliverability.

D) Email archiving in S3 provides persistence but does not test deliverability.
References:
https://aws.amazon.com/blogs/messaging-and-targeting/test-email-deliverability-with-the-new-ses-virtual-deliv-manager/

Simple Email Service

SES mailbox simulator



Send test email Info

The Amazon SES mailbox simulator lets you test how your application handles different email sending scenarios. Emails that you send to the mailbox simulator do not count towards your sending quota or your bounce and complaint rates. Learn more

Message details

Email format

● Formatted
Choose this option if you want to construct a simple test message using the form provided. SES takes the information entered in the form and parses it into email format for you.

○ Raw
Choose this option if you want to send a more complex test message, such as one that uses HTML or includes attachments. This option requires you to format the entire message yourself.

From-address: dustin @example.com

Scenario Info
Choose the email sending scenario that you want to simulate. Each scenario corresponds to a different recipient email address managed by the mailbox simulator. To specify a custom recipient, select Custom.

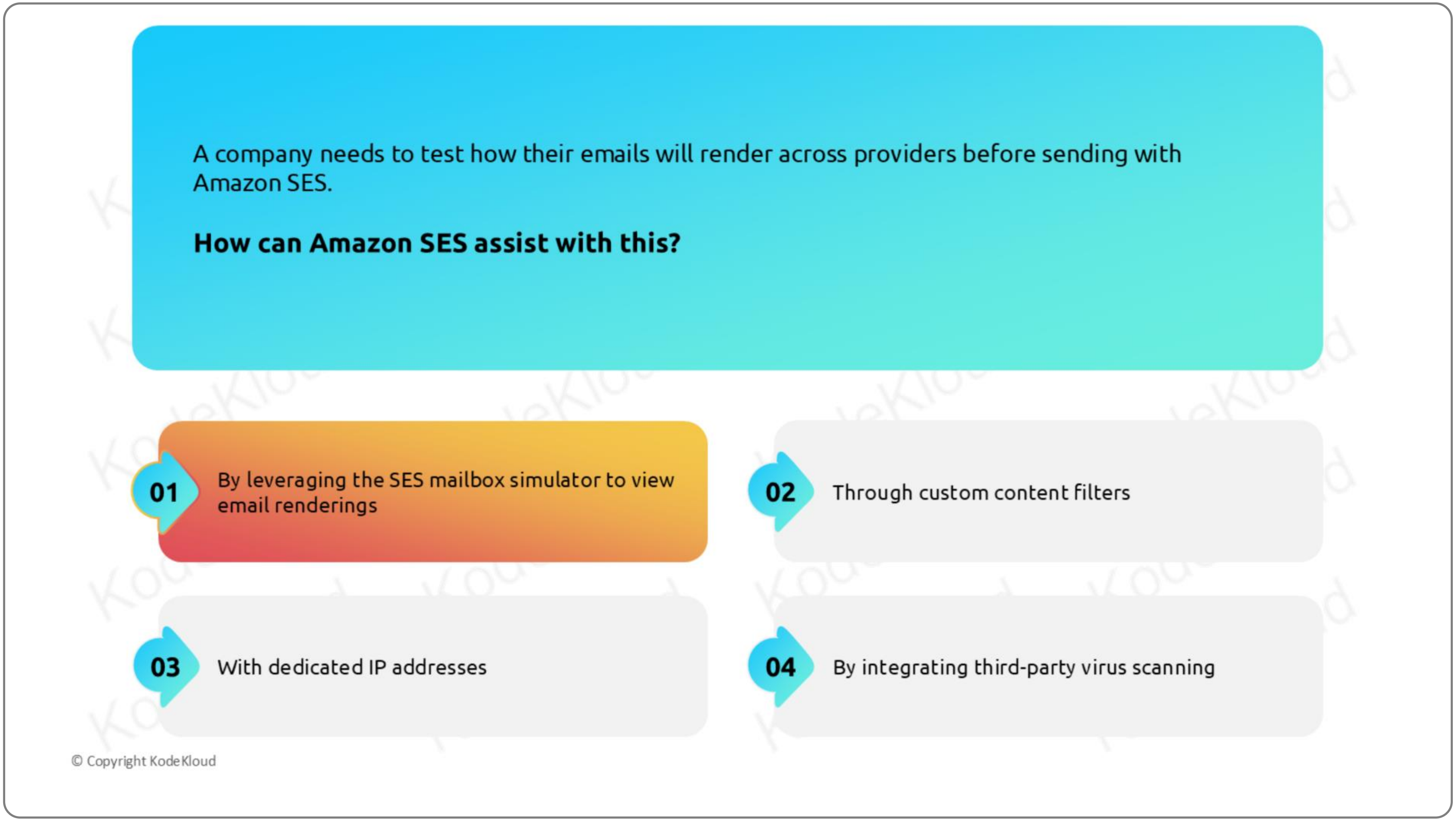
Successful delivery ▼
success@simulator.amazonses.com

Subject: Testing Success Event

Body - *optional*: Testing the success event to verify normal sending operation.

Configuration set - *optional* Info: *Choose a configuration set* ▼

▶ Additional configurations- *optional*



A company needs to test how their emails will render across providers before sending with Amazon SES.

How can Amazon SES assist with this?

01 By leveraging the SES mailbox simulator to view email renderings

02 Through custom content filters

03 With dedicated IP addresses

04 By integrating third-party virus scanning

Correct Answer: A
Explanation: The SES mailbox simulator allows previewing how emails will render across major providers before sending.
Incorrect Answers:
B) Custom content filters screen emails but do not simulate renderings.
C) Dedicated IP addresses help with deliverability but do not simulate renderings.
D) Virus scanning services scan for malware but do not simulate renderings.

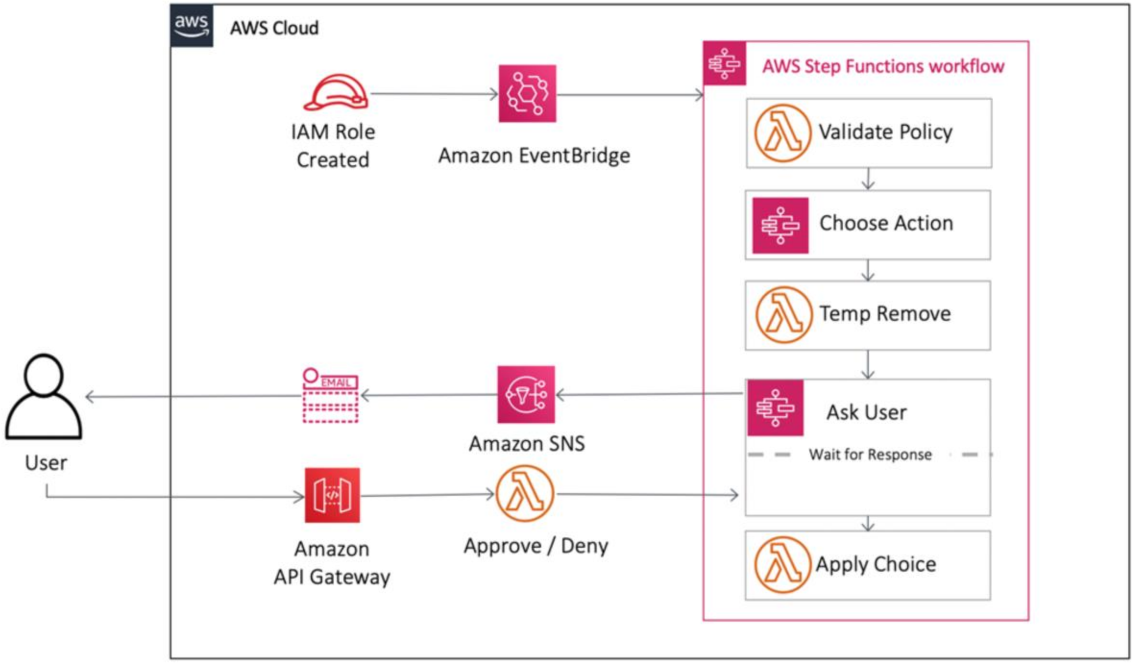
References:
https://docs.aws.amazon.com/ses/latest/DeveloperGuide/send-email-mailbox-simulator.html

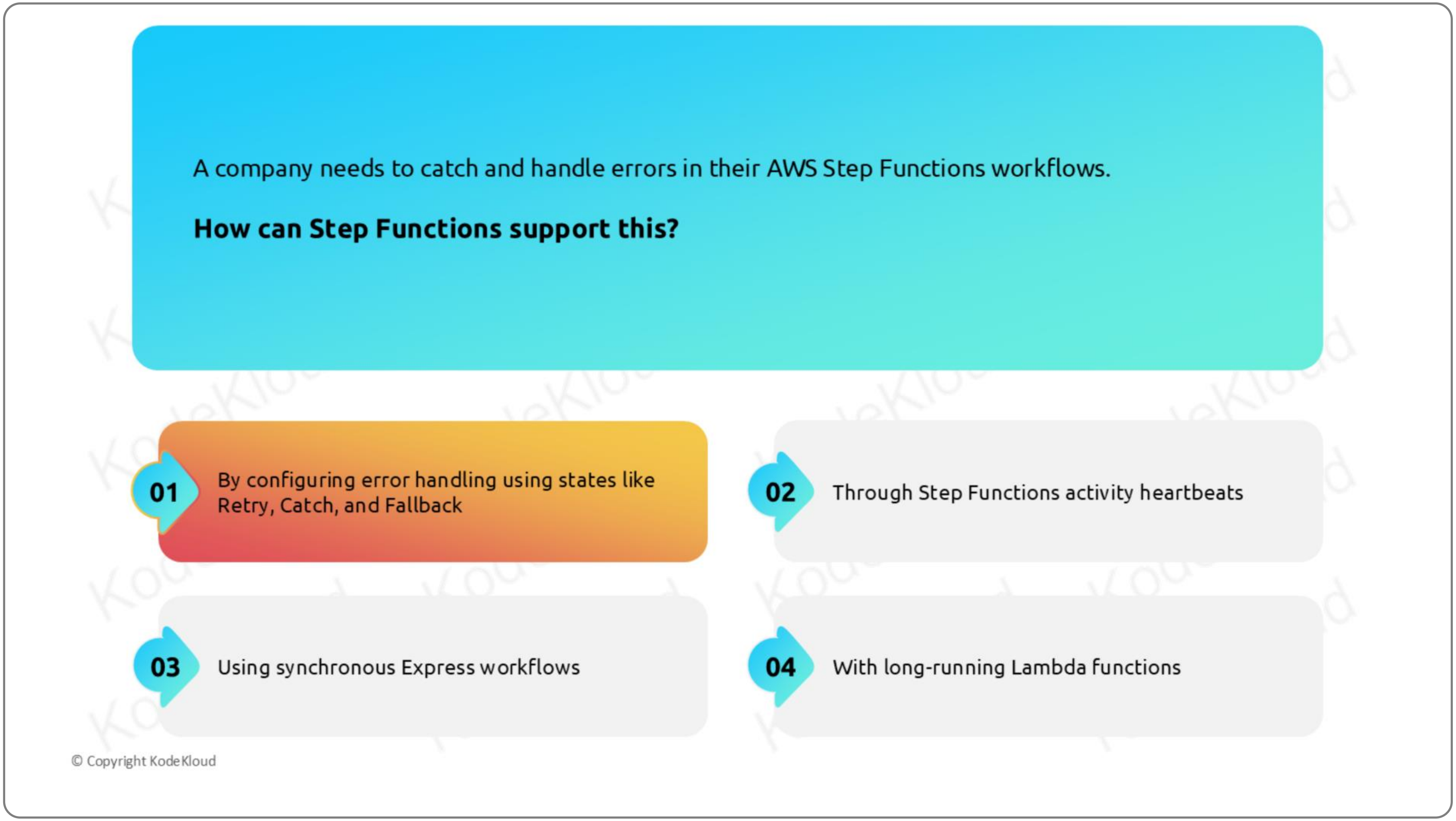
Power of Workflows and Orchestration

Designing for Reliability – Step Functions

Step Functions

Step Functions is already inherently redundant and reliable due to its nature.





A company needs to catch and handle errors in their AWS Step Functions workflows.

How can Step Functions support this?

01 By configuring error handling using states like Retry, Catch, and Fallback

02 Through Step Functions activity heartbeats

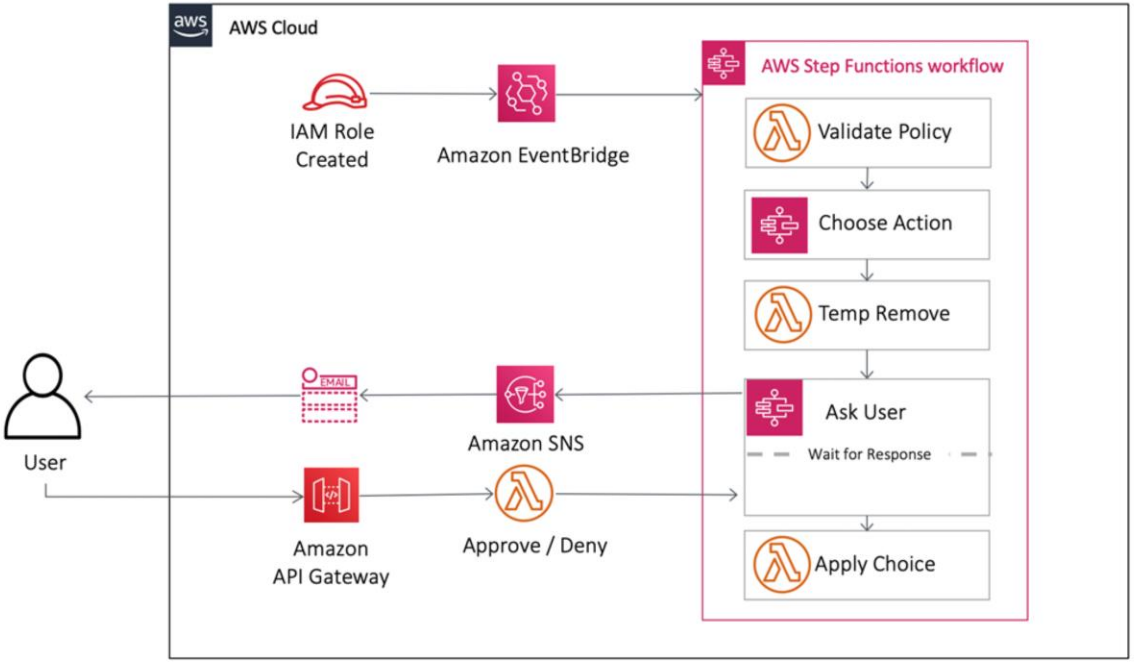
03 Using synchronous Express workflows

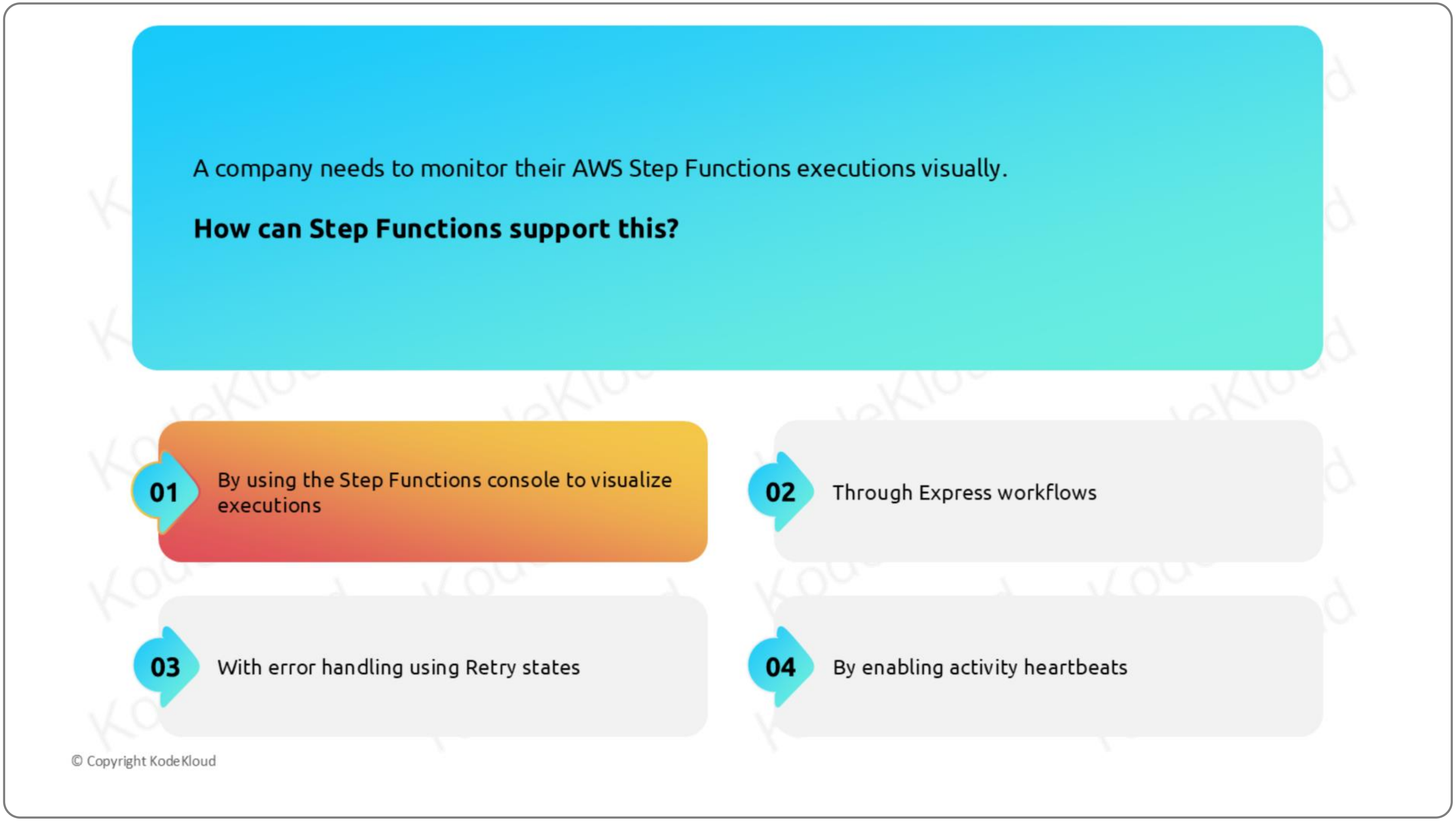
04 With long-running Lambda functions

Correct Answer: A
Explanation: AWS Step Functions allows implementing robust error handling logic using states like Retry, Catch, and Fallback to catch and recover from errors.
Incorrect Answers:
B) Activity heartbeats monitor execution but do not enable error handling.
C) Express workflows optimize for speed but do not affect error handling.

D) Long-running Lambda functions may be needed but do not relate to Step Functions error handling.
References:
https://docs.aws.amazon.com/step-functions/latest/dg/concepts-error-handling.html

Step Functions





A company needs to monitor their AWS Step Functions executions visually.

How can Step Functions support this?

01 By using the Step Functions console to visualize executions

02 Through Express workflows

03 With error handling using Retry states

04 By enabling activity heartbeats

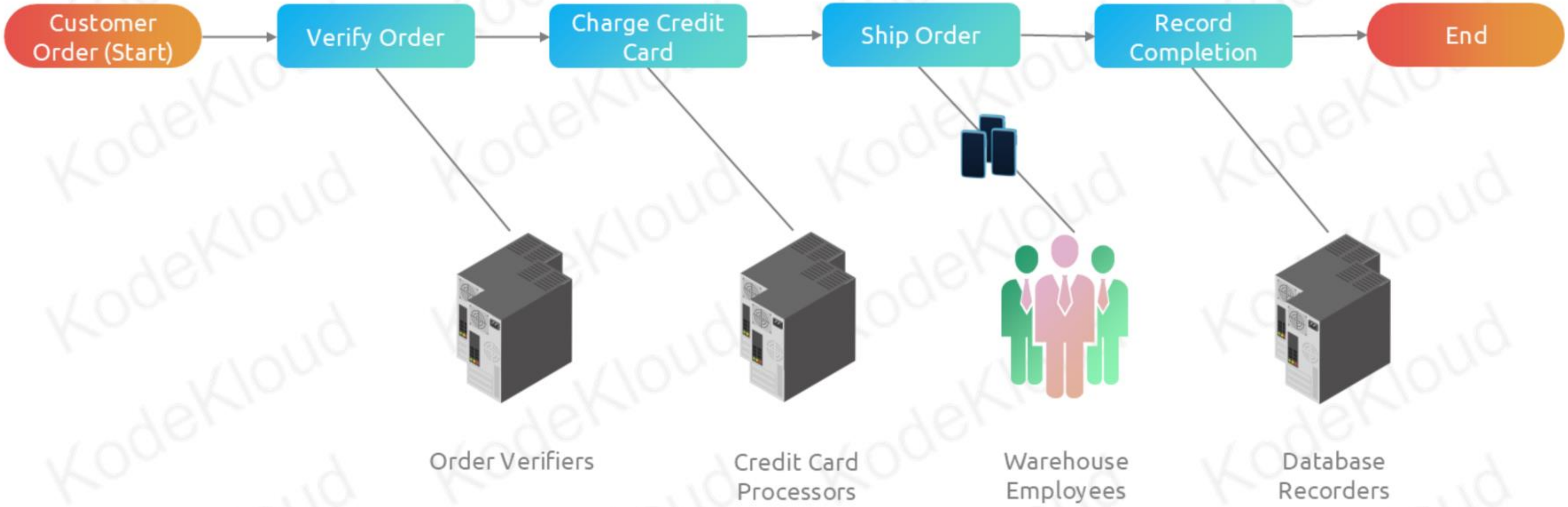
Correct Answer: A
Explanation: The AWS Step Functions console provides visualization of state machine executions, helping monitor workflow progress.
Incorrect Answers:
B) Express workflows optimize for speed but do not visualize workflows.
C) Retry states handle errors but do not visualize workflows.
D) Activity heartbeats monitor execution but do not visually depict workflows.

References:

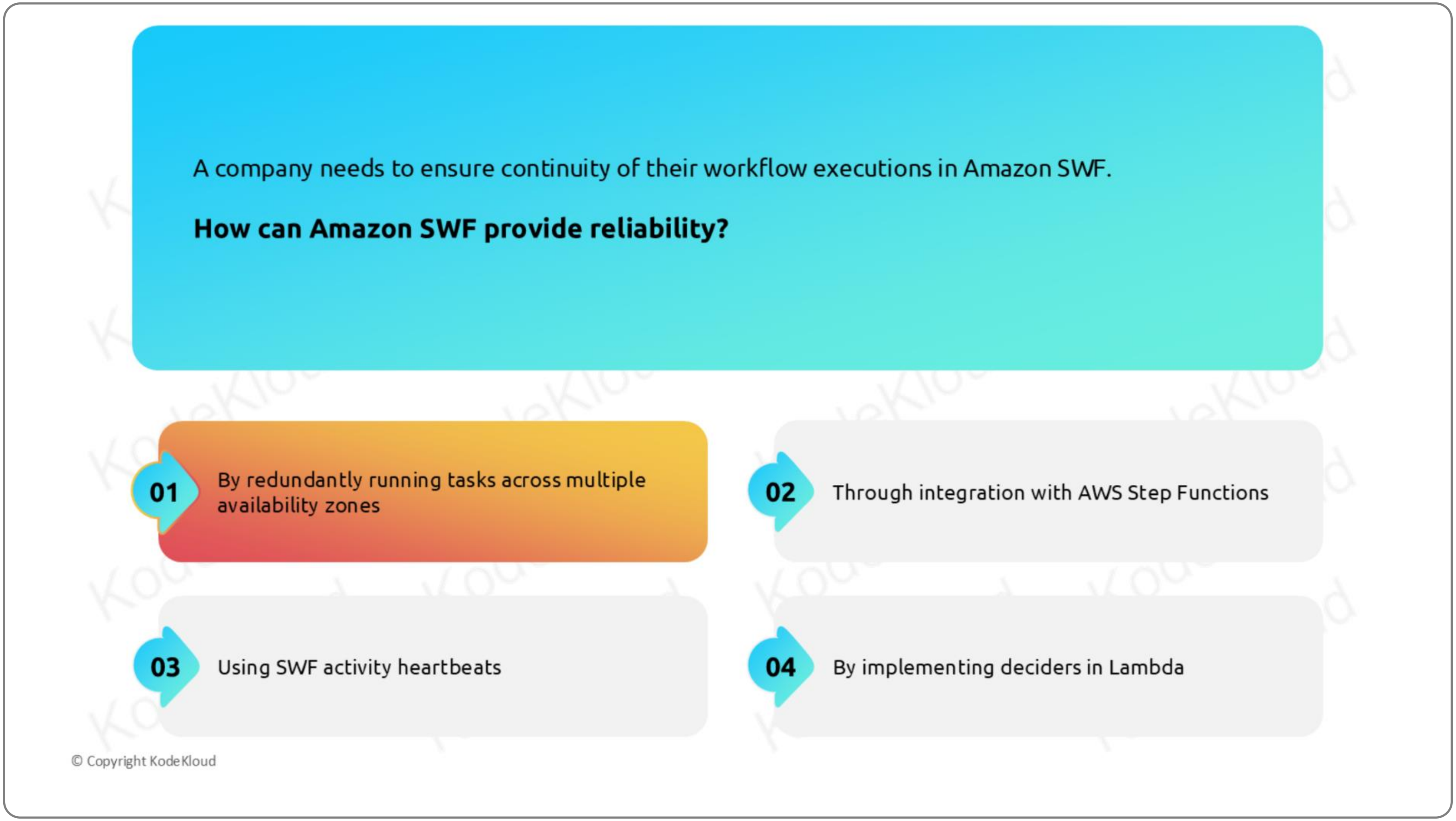
https://docs.aws.amazon.com/step-functions/latest/dg/procedure-view-execution-history.html

Designing for Reliability – Simple Workflow Service

Simple Workflow Service



SWF is an older service for workflow orchestration. It works like step functions but isn't serverless or unmanaged, so AZs must be taken into account.



A company needs to ensure continuity of their workflow executions in Amazon SWF.

How can Amazon SWF provide reliability?

01 By redundantly running tasks across multiple availability zones

02 Through integration with AWS Step Functions

03 Using SWF activity heartbeats

04 By implementing deciders in Lambda

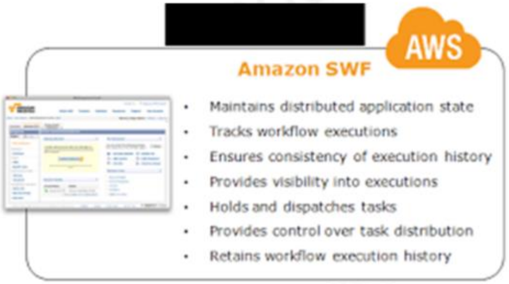
Correct Answer: A
Explanation: Amazon SWF provides redundancy by allowing tasks to be run across multiple availability zones to provide high reliability.
Incorrect Answers:
B) Integration with Step Functions enables orchestration but does not directly provide SWF redundancy.
C) Activity heartbeats provide execution visibility but do not enable redundancy.

D) Implementing deciders in Lambda enables programming flexibility but does not provide redundancy.
References:
https://docs.aws.amazon.com/amazonswf/latest/developerguide/swf-dev-zones.html

Simple Workflow Service

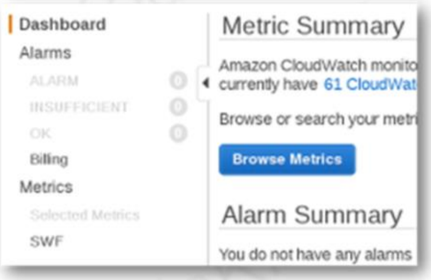
Step Functions, like most AWS services, is encrypted and tracked using standard AWS Services. CloudWatch, CloudTrail, and AWS Config are used for this.

[redacted]



Amazon SWF

- Maintains distributed application state
- Tracks workflow executions
- Ensures consistency of execution history
- Provides visibility into executions
- Holds and dispatches tasks
- Provides control over task distribution
- Retains workflow execution history



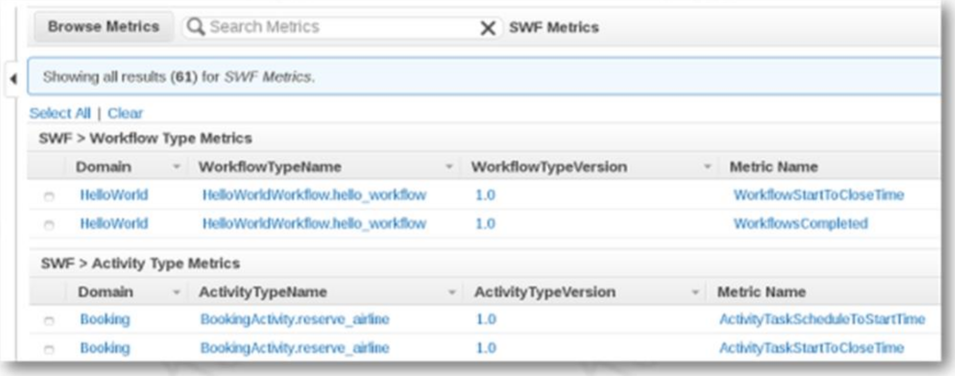
Metric Summary

Amazon CloudWatch monito
currently have 61 CloudWat

Browse or search your metri

Alarm Summary

You do not have any alarms



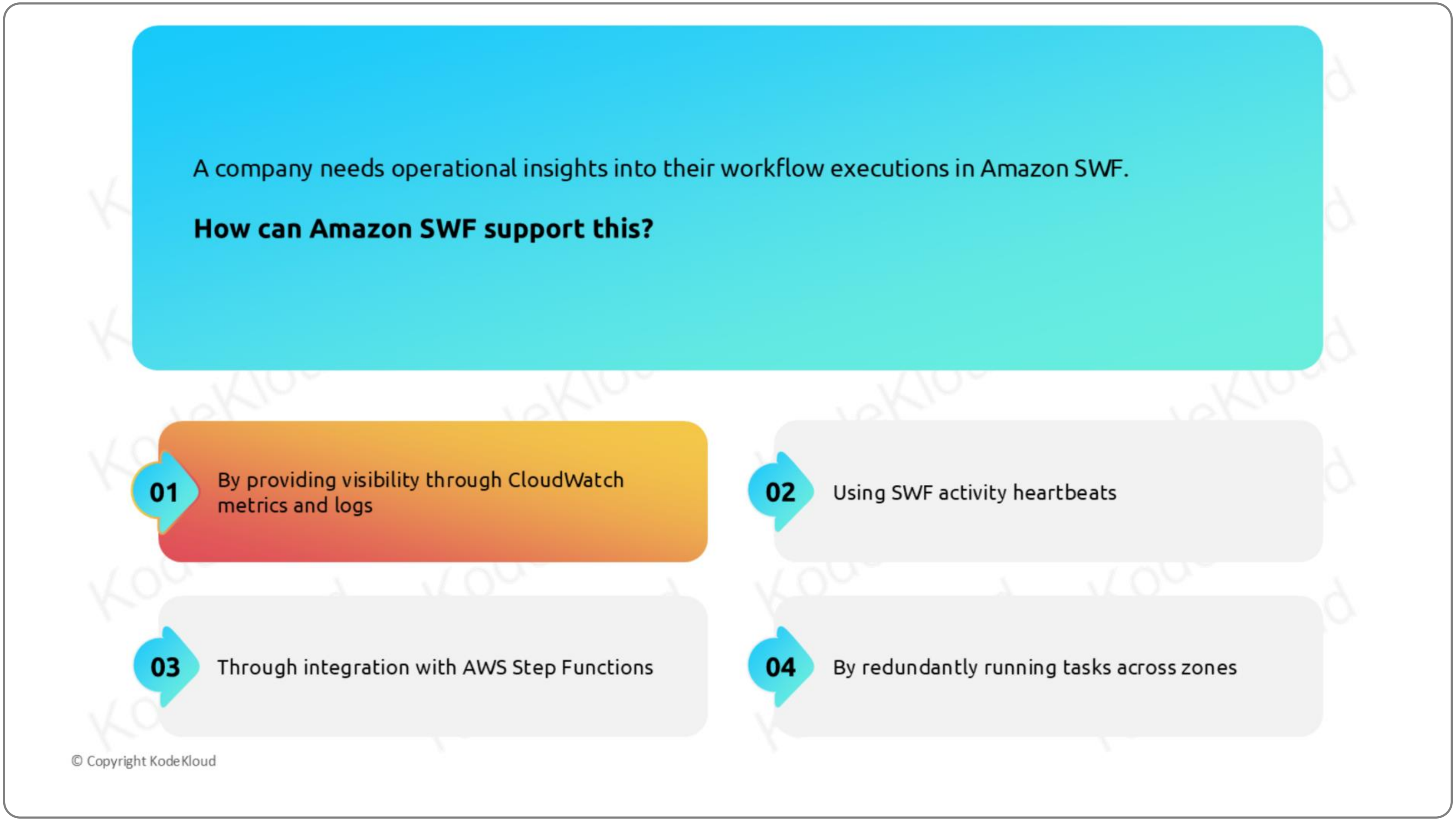
Showing all results (61) for *SWF Metrics*.

SWF > Workflow Type Metrics

	Domain	WorkflowTypeName	WorkflowTypeVersion	Metric Name
□	HelloWorld	HelloWorldWorkflow.hello_workflow	1.0	WorkflowStartToCloseTime
□	HelloWorld	HelloWorldWorkflow.hello_workflow	1.0	WorkflowsCompleted

SWF > Activity Type Metrics

	Domain	ActivityTypeName	ActivityTypeVersion	Metric Name
□	Booking	BookingActivity.reserve_airline	1.0	ActivityTaskScheduleToStartTime
□	Booking	BookingActivity.reserve_airline	1.0	ActivityTaskStartToCloseTime



A company needs operational insights into their workflow executions in Amazon SWF.

How can Amazon SWF support this?

01 By providing visibility through CloudWatch metrics and logs

02 Using SWF activity heartbeats

03 Through integration with AWS Step Functions

04 By redundantly running tasks across zones

Correct Answer: A
Explanation: Amazon SWF offers visibility into workflow executions via CloudWatch metrics and logs.
Incorrect Answers:
B) Activity heartbeats provide execution visibility but are not the primary monitoring method.
C) Integration with Step Functions enables orchestration but does not directly provide visibility.
D) Redundant tasks provide reliability but not workflow visibility.

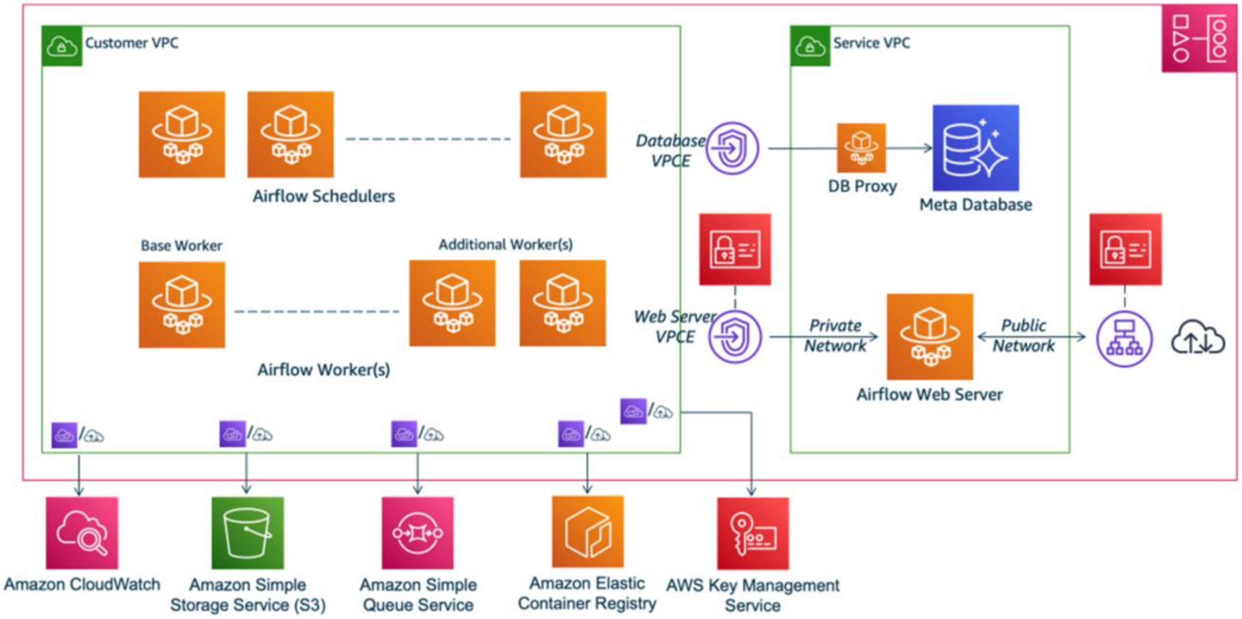
References:
https://docs.aws.amazon.com/amazonswf/latest/developerguide/swf-dev-cloudwatch.html

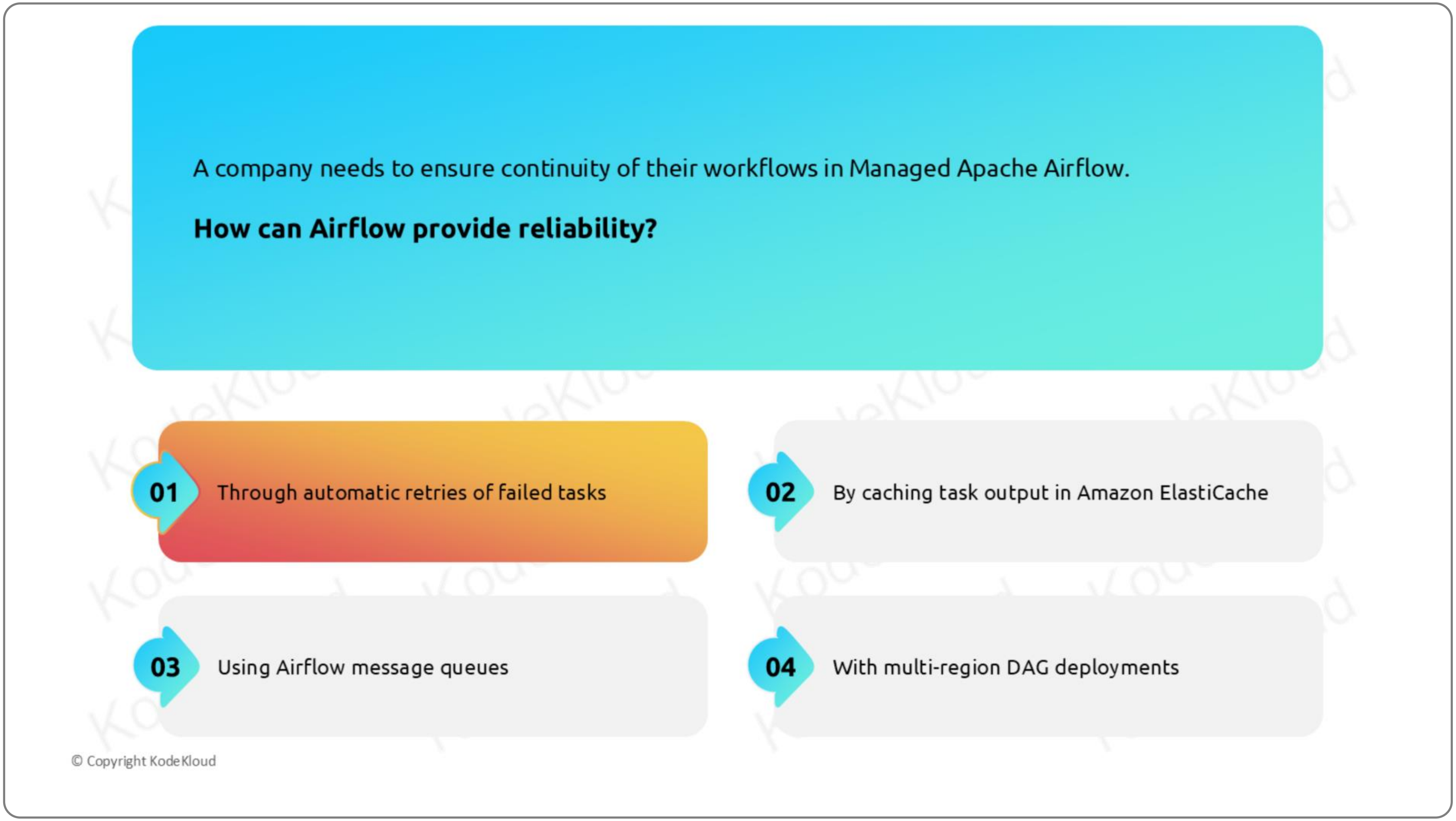
Designing for Reliability – Managed Apache AirFlow

Managed Apache Airflow

MWAA is a complicated beast of a service, but highly managed. So reliability is baked in. Scaling, management, patching, etc.

Amazon MWAA Architecture





A company needs to ensure continuity of their workflows in Managed Apache Airflow.

How can Airflow provide reliability?

01 Through automatic retries of failed tasks

02 By caching task output in Amazon ElastiCache

03 Using Airflow message queues

04 With multi-region DAG deployments

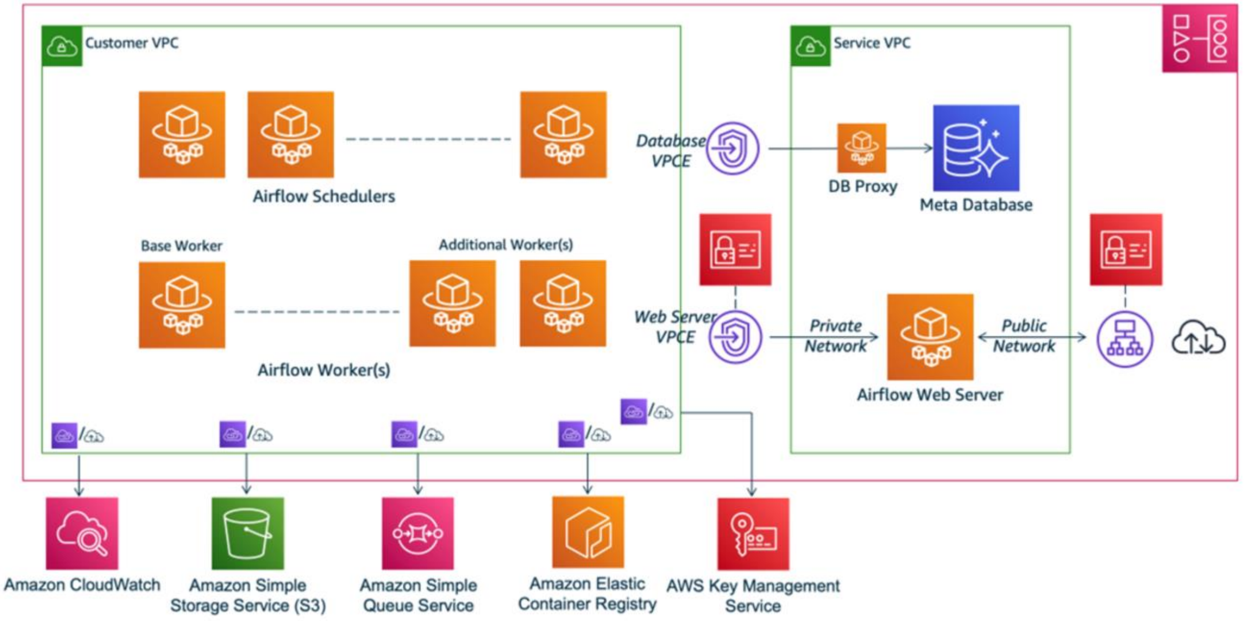
Correct Answer: A
Explanation: Managed Apache Airflow provides reliability by automatically retrying failed tasks up to 3 times.
Incorrect Answers:
B) Caching in ElastiCache improves performance but does not directly provide reliability.
C) Message queues enable loose coupling but do not provide reliability.
D) Multi-region DAG deployments enable disaster recovery but do not directly provide reliability.

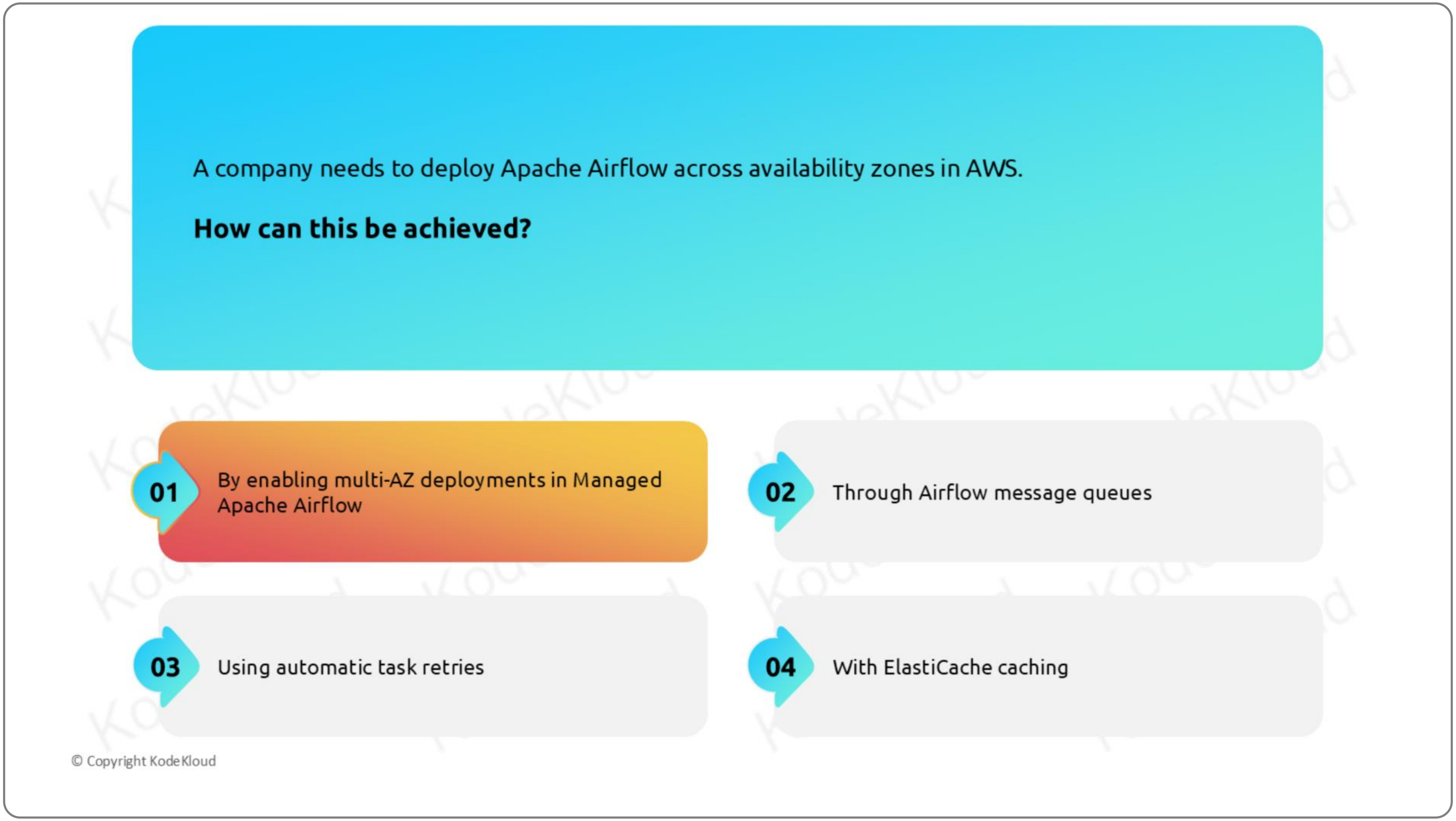
References:
https://docs.aws.amazon.com/mwaa/latest/userguide/mwaa-tasks.html

Managed Apache Airflow

MWAA uses standard logging and infrastructure protection. Since it uses other services, Multi-AZ is a concern.

Amazon MWAA Architecture





A company needs to deploy Apache Airflow across availability zones in AWS.

How can this be achieved?

01 By enabling multi-AZ deployments in Managed Apache Airflow

02 Through Airflow message queues

03 Using automatic task retries

04 With ElastiCache caching

Correct Answer: A
Explanation: Managed Apache Airflow supports deploying airflow environments across multiple availability zones for redundancy.
Incorrect Answers:
B) Message queues enable loose coupling but do not provide redundancy.
C) Automatic task retries provide reliability but not redundancy.

D) Caching in ElastiCache improves performance but does not provide redundancy.
References:
https://docs.aws.amazon.com/mwaa/latest/userguide/mwaa-high-availability.html

Summary

01 This section covers all the application enhancers and application integration services in AWS

02 This ranges from Autoscaling to Load Balancing, to email to Serverless Orchestration

03 Most of these services are Highly Managed, which means redundancy is just a matter of one or two features

04 Many services in application integration are aimed at enabling loose coupling, a key aspect of App Integration

KodeKloud

Visit www.kodekloud.com to learn more.