Vendor: Amazon

Exam Code: DVA-C02

Exam Name: AWS Certified Developer - Associate (DVA-C02)

Version: DEMO

QUESTION 1

A developer is creating an application that will store personal health information (PHI). The PHI needs to be encrypted at all times. An encrypted Amazon RDS for MySQL DB instance is storing the data. The developer wants to increase the performance of the application by caching frequently accessed data while adding the ability to sort or rank the cached datasets.
Which solution will meet these requirements?

A. Create an Amazon ElastiCache for Redis instance. Enable encryption of data in transit and at rest. Store frequently accessed data in the cache.
B. Create an Amazon ElastiCache for Memcached instance. Enable encryption of data in transit and at rest. Store frequently accessed data in the cache.
C. Create an Amazon RDS for MySQL read replica. Connect to the read replica by using SSL. Configure the read replica to store frequently accessed data.
D. Create an Amazon DynamoDB table and a DynamoDB Accelerator (DAX) cluster for the table. Store frequently accessed data in the DynamoDB table.

Answer: A
Explanation:
You can use Amazon Elasticache for Redis Sorted Sets to easily implement a dashboard that keeps a list of sorted data by their rank.
https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/elasticache-use-cases.html#elasticache-for-redis-use-cases-gaming
https://aws.amazon.com/elasticache/redis-vs-memcached/

QUESTION 2

A developer is creating an application that will be deployed on IoT devices. The application will send data to a RESTful API that is deployed as an AWS Lambda function. The application will assign each API request a unique identifier. The volume of API requests from the application can randomly increase at any given time of day.
During periods of request throttling, the application might need to retry requests. The API must be able to handle duplicate requests without inconsistencies or data loss.
Which solution will meet these requirements?

A. Create an Amazon RDS for MySQL DB instance. Store the unique identifier for each request in a database table. Modify the Lambda function to check the table for the identifier before processing the request.
B. Create an Amazon DynamoDB table. Store the unique identifier for each request in the table. Modify the Lambda function to check the table for the identifier before processing the request.
C. Create an Amazon DynamoDB table. Store the unique identifier for each request in the table. Modify the Lambda function to return a client error response when the function receives a duplicate request.
D. Create an Amazon ElastiCache for Memcached instance. Store the unique identifier for each request in the cache. Modify the Lambda function to check the cache for the identifier before processing the request.

Answer: B
Explanation:
The resolution is to make the Lambda function idempotent.
https://repost.aws/knowledge-center/lambda-function-idempotent
https://aws.amazon.com/builders-library/making-retries-safe-with-idempotent-APIs/

QUESTION 3

A company hosts a client-side web application for one of its subsidiaries on Amazon S3. The web

application can be accessed through Amazon CloudFront from https://www.example.com. After a successful rollout, the company wants to host three more client-side web applications for its remaining subsidiaries on three separate S3 buckets. To achieve this goal, a developer moves all the common JavaScript files and web fonts to a central S3 bucket that serves the web applications. However, during testing, the developer notices that the browser blocks the JavaScript files and web fonts.

What should the developer do to prevent the browser from blocking the JavaScript files and web fonts?

A. Create four access points that allow access to the central S3 bucket. Assign an access point to each web application bucket.
B. Create a bucket policy that allows access to the central S3 bucket. Attach the bucket policy to the central S3 bucket.
C. Create a cross-origin resource sharing (CORS) configuration that allows access to the central S3 bucket. Add the CORS configuration to the central S3 bucket.
D. Create a Content-MD5 header that provides a message integrity check for the central S3 bucket. Insert the Content-MD5 header for each web application request.

Answer: C
Explanation:
This is a frequent trouble. Web applications cannot access the resources in other domains by default, except some exceptions. You must configure CORS on the resources to be accessed.
https://docs.aws.amazon.com/AmazonS3/latest/userguide/cors.html

QUESTION 4
An application is processing clickstream data using Amazon Kinesis. The clickstream data feed into Kinesis experiences periodic spikes. The PutRecords API call occasionally fails and the logs show that the failed call returns the response shown below:

```
{
    "FailedRecordCount": 1,
    "Records": [
        {
            "SequenceNumber": "21269319889900637946712965403778482371",
            "ShardId": "shardId-000000000001"
        },
        {
            "ErrorCode":"ProvisionedThroughputExceededException",
            "ErrorMessage": "Rate exceeded for shard shardId-000000000001 in

                stream exampleStreamName under account 123456789."
        },
        {
            "SequenceNumber": "21269319889999637946712965403778482985",
            "ShardId": "shardId-000000000002"
        }
    ]
}
```

Which techniques will help mitigate this exception? (Choose two.)

A. Which techniques will help mitigate this exception? (Choose two.)

B. Use a PutRecord API instead of PutRecords.
C. Reduce the frequency and/or size of the requests.
D. Use Amazon SNS instead of Kinesis.
E. Reduce the number of KCL consumers.

Answer: AC
Explanation:
https://aws.amazon.com/premiumsupport/knowledge-center/kinesis-data-stream-throttling-errors/

QUESTION 6
A developer is creating a template that uses AWS CloudFormation to deploy an application. The application is serverless and uses Amazon API Gateway, Amazon DynamoDB, and AWS Lambda. Which AWS service or tool should the developer use to define serverless resources in YAML?

A. CloudFormation serverless intrinsic functions
B. AWS Elastic Beanstalk
C. AWS Serverless Application Model (AWS SAM)
D. AWS Cloud Development Kit (AWS CDK)

Answer: C
Explanation:
The AWS Serverless Application Model(SAM) is an open-source framework for building serverless applications. It provides shorthand syntax to express functions, APIs, databases, and event source mappings. With just a few lines per resource, you can define the application you want and model it using YAML.

QUESTION 7
A developer wants to insert a record into an Amazon DynamoDB table as soon as a new file is added to an Amazon S3 bucket.
Which set of steps would be necessary to achieve this?

A. Create an event with Amazon EventBridge that will monitor the S3 bucket and then insert the records into DynamoDB.
B. Configure an S3 event to invoke an AWS Lambda function that inserts records into DynamoDB.
C. Create an AWS Lambda function that will poll the S3 bucket and then insert the records into DynamoDB.
D. Create a cron job that will run at a scheduled time and insert the records into DynamoDB.

Answer: B
Explanation:
Configure S3 event as trigger to lambda.
Allow lambda to insert records in Dynamo DB.

QUESTION 8
A development team wants to build a continuous integration/continuous delivery (CI/CD) pipeline. The team is using AWS CodePipeline to automate the code build and deployment. The team wants to store the program code to prepare for the CI/CD pipeline.
Which AWS service should the team use to store the program code?

A. AWS CodeDeploy

B. AWS CodeArtifact
C. AWS CodeCommit
D. Amazon CodeGuru

Answer: C
Explanation:
AWS CodeCommit - A fully-managed source control service that hosts secure Git-based repositories. CodeCommit makes it easy for teams to collaborate on code in a secure and highly scalable ecosystem. This solution uses CodeCommit to create a repository to store the application and deployment codes.
https://aws.amazon.com/blogs/devops/complete-ci-cd-with-aws-codecommit-aws-codebuild-aws-codedeploy-and-aws-codepipeline/

QUESTION 9
A developer is testing a new file storage application that uses an Amazon CloudFront distribution to serve content from an Amazon S3 bucket. The distribution accesses the S3 bucket by using an origin access identity (OAI). The S3 bucket's permissions explicitly deny access to all other users. The application prompts users to authenticate on a login page and then uses signed cookies to allow users to access their personal storage directories. The developer has configured the distribution to use its default cache behavior with restricted viewer access and has set the origin to point to the S3 bucket. However, when the developer tries to navigate to the login page, the developer receives a 403 Forbidden error.
The developer needs to implement a solution to allow unauthenticated access to the login page. The solution also must keep all private content secure.
Which solution will meet these requirements?

A. Add a second cache behavior to the distribution with the same origin as the default cache behavior. Set the path pattern for the second cache behavior to the path of the login page, and make viewer access unrestricted. Keep the default cache behavior's settings unchanged.
B. Add a second cache behavior to the distribution with the same origin as the default cache behavior. Set the path pattern for the second cache behavior to *, and make viewer access restricted. Change the default cache behavior's path pattern to the path of the login page, and make viewer access unrestricted.
C. Add a second origin as a failover origin to the default cache behavior. Point the failover origin to the S3 bucket. Set the path pattern for the primary origin to *, and make viewer access restricted. Set the path pattern for the failover origin to the path of the login page, and make viewer access unrestricted.
D. Add a bucket policy to the S3 bucket to allow read access. Set the resource on the policy to the Amazon Resource Name (ARN) of the login page object in the S3 bucket. Add a CloudFront function to the default cache behavior to redirect unauthorized requests to the login page's S3 URL.

Answer: A
Explanation:
https://docs.aws.amazon.com/cloudfront/latest/APIReference/API_CacheBehavior.html
The path pattern for the default cache behavior is * and cannot be changed. If the request for an object does not match the path pattern for any cache behaviors, CloudFront applies the behavior in the default cache behavior.

QUESTION 10
A developer is using AWS Amplify Hosting to build and deploy an application. The developer is receiving an increased number of bug reports from users. The developer wants to add end-to-end testing to the application to eliminate as many bugs as possible before the bugs reach production.

Which solution should the developer implement to meet these requirements?

A. Run the amplify add test command in the Amplify CLI.
B. Create unit tests in the application. Deploy the unit tests by using the amplify push command in the Amplify CLI.
C. Add a test phase to the amplify.yml build settings for the application.
D. Add a test phase to the aws-exports.js file for the application.

Answer: C
Explanation:
You can run end-to-end (E2E) tests in the test phase of your Amplify app to catch regressions before pushing code to production. The test phase can be configured in the build specification YML.
https://docs.aws.amazon.com/amplify/latest/userguide/running-tests.html

QUESTION 11
An ecommerce company is using an AWS Lambda function behind Amazon API Gateway as its application tier. To process orders during checkout, the application calls a POST API from the frontend. The POST API invokes the Lambda function asynchronously. In rare situations, the application has not processed orders. The Lambda application logs show no errors or failures.
What should a developer do to solve this problem?

A. Inspect the frontend logs for API failures. Call the POST API manually by using the requests from the log file.
B. Create and inspect the Lambda dead-letter queue. Troubleshoot the failed functions. Reprocess the events.
C. Inspect the Lambda logs in Amazon CloudWatch for possible errors. Fix the errors.
D. Make sure that caching is disabled for the POST API in API Gateway.

Answer: B
Explanation:
https://docs.aws.amazon.com/lambda/latest/dg/invocation-retries.html
Asynchronous invocation - Lambda retries function errors twice. If the function doesn't have enough capacity to handle all incoming requests, events might wait in the queue for hours or days to be sent to the function. You can configure a dead-letter queue on the function to capture events that weren't successfully processed.