Cloud Dev Lab 2

Lab Objectives

By the end of this lab, you will have built a serverless, event-driven, scalable, and decoupled architecture to calculate quiz statistics for a class. A general pipeline for any simple transformation pipeline.

Proposed Services

- API Gateway
- AWS Lambda
- SQS (Simple Queue Service)
- SNS (Simple Notification Service)
- S3 (Simple Storage Service)
- CloudWatch

Region: us-east-1

IaaC Tool: Terraform.

Expected Workflow

1. **API Gateway**:
    - Exposes the first Lambda function to the internet.
    - Ensure `authorization = "NONE"` for now so that the API Gateway is accessible without credentials.
    - In a real solution, it would be integrated with AWS Cognito to enable key based authentication to avoid misuse.
2. **Request-Receiver-Lambda**:
    - Receives CSV data via an API request (refer to the example `curl` request below).
    - Parses and stores the data as a CSV file in an S3 bucket (e.g. raw-data-bucket).
    - Place a message in an SQS queue (you need to determine the message format).
    - In a more complex solution, this would be handling very large file uploads using multipart upload.

3. **SQS Queue 1**:
    - Receives messages from the first Lambda function.
    - Triggers **Analytics-Calculator-Lambda**.
4. **Analytics-Calculator-Lambda**:
    - Retrieves the CSV file from the S3 bucket.
    - Parses the file and calculates statistics for each quiz (**minimum, maximum, mean** scores).
    - Stores the computed statistics in another S3 bucket.
    - Send a message to **SQS Queue 2**.
    - In a more complex solution, this would be doing multiple transformations based on business logic and doing quality checks.
5. **SQS Queue 2**:
    - Triggers **Result-Notifier-Lambda**.
6. **Result-Notifier-Lambda**:
    - Retrieves the statistics file from the second S3 bucket.
    - Send an email with the quiz statistics report to students using SNS.
    - In a more complex solution, this would be handling downstream system(s) integration.

API post request example: (you can use postman or curl)

```
curl -X POST "<Your_API_Endpoint>" \
     -H "Content-Type: application/json" \
     -d '{
           "data": [
           "Quiz1,85,76,90",
           "Quiz2,78,88,85",
           "Quiz3,92,90,87"
           ]
     }'
```

In total, you should have 3 lambda functions, 2 SQS queues, 1 API gateway, 1 SNS service and 2 S3 buckets. Remember to assign policies based on “least access privileges” and for debugging your lambdas, remember to allow them to print their logs to cloudwatch.

Checkpoints

1. **Checkpoint 1**: Verify that your API request successfully creates a CSV file in the S3 bucket.
2. **Checkpoint 2**: Ensure that the computed statistics file is stored in the second S3 bucket.
3. **Checkpoint 3**: Confirm that you receive the email with the quiz statistics report.

Submission Guidelines:

You are required to submit:

1) Terraform and Lambda functions Code files In a zip folder. Remember to follow proper file naming convention for terraform files.
2) An architecture diagram of the entire process. Please incorporate feedback from lab 1 diagrams shared on slack.
3) Screenshots of each of the checkpoints
4) Explanations of what “Event Driven Architecture” and “Decoupled Systems” are, their advantages and how they are used in this lab.

Add these in a pdf and rename that as **<YourRollNumber>_Lab2.pdf** and submit on the LMS assignments tab.

Note: For any topic, refer to Amazon's or Terraform documentation for detail, examples are shown also in class. Remember to look at service pricing and limits. Be sure to add relevant tags to all of the services that you use for this lab. To maintain fairness, the TAs will be marking as per the curl request shared.

Good luck and start early.