**Description**

**General Notes:**

**Submission:**

- Provide the Python code in GitHub and share the GitHub link as a submission.

- Provide brief documentation on the steps followed.

- Take a screenshot of the steps and put them into the documentation.

**Evaluation Criteria:**

- Correctness of the code.

- The ability to complete the task.

- Clarity and organization of documentation.

**You are only required to do any 4 of the below for the grading. However, it is recommended that you practice as much as possible, which will help you build your expertise with automation and various AWS services.**

**Tasks:**

**Assignment 1: Automated Instance Management Using AWS Lambda and Boto3**

**Objective:** In this assignment, you will gain hands-on experience with AWS Lambda and Boto3, Amazon's SDK for Python. You will create a Lambda function that will automatically manage EC2 instances based on their tags.

**Task:** You're tasked to automate the stopping and starting of EC2 instances based on tags. Specifically:

1. Setup:

   - Create two EC2 instances.

   - Tag one of them as `Auto-Stop` and the other as `Auto-Start`.

2. Lambda Function Creation:

   - Set up an AWS Lambda function.

   - Ensure that the Lambda function has the necessary IAM permissions to describe, stop, and start EC2 instances.

3. Coding:

   - Using Boto3 in the Lambda function:

     - Detect all EC2 instances with the `Auto-Stop` tag and stop them.

     - Detect all EC2 instances with the `Auto-Start` tag and start them.

4. Testing:

   - Manually invoke the Lambda function.

   - Confirm that the instance tagged `Auto-Stop` stops and the one tagged `Auto-Start` starts.

**Instructions:**

1. EC2 Setup:

   - Navigate to the EC2 dashboard and create two new t2.micro instances (or any other available free-tier type).

   - Tag the first instance with a key `Action` and value `Auto-Stop`.

   - Tag the second instance with a key `Action` and value `Auto-Start`.

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach the `AmazonEC2FullAccess` policy to this role. (Note: In a real-world scenario, you would want to limit permissions for better security.)

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Initialize a boto3 EC2 client.

     2. Describe instances with `Auto-Stop` and `Auto-Start` tags.

     3. Stop the `Auto-Stop` instances and start the `Auto-Start` instances.

     4. Print instance IDs that were affected for logging purposes.

4. Manual Invocation:

   - After saving your function, manually trigger it.

   - Go to the EC2 dashboard and confirm that the instances' states have changed according to their tags.

**Assignment 2: Automated S3 Bucket Cleanup Using AWS Lambda and Boto3**

**Objective**: To gain experience with AWS Lambda and Boto3 by creating a Lambda function that will automatically clean up old files in an S3 bucket.

**Task:** Automate the deletion of files older than 30 days in a specific S3 bucket.

**Instructions:**

1. S3 Setup:

   - Navigate to the S3 dashboard and create a new bucket.

   - Upload multiple files to this bucket, ensuring that some files are older than 30 days (you may need to adjust your system's date temporarily for this or use old files).

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach the `AmazonS3FullAccess` policy to this role. (Note: For enhanced security in real-world scenarios, use more restrictive permissions.)

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Initialize a boto3 S3 client.

     2. List objects in the specified bucket.

     3. Delete objects older than 30 days.

     4. Print the names of deleted objects for logging purposes.

4. Manual Invocation:

   - After saving your function, manually trigger it.

   - Go to the S3 dashboard and confirm that only files newer than 30 days remain.

**Assignment 3: Monitor Unencrypted S3 Buckets Using AWS Lambda and Boto3**

**Objective:**To enhance your AWS security posture by setting up a Lambda function that detects any S3 bucket without server-side encryption.

**Task:** Automate the detection of S3 buckets that don't have server-side encryption enabled.

**Instructions:**

1. S3 Setup:

   - Navigate to the S3 dashboard and create a few buckets. Ensure that a couple of them don't have server-side encryption enabled.

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach the `AmazonS3ReadOnlyAccess` policy to this role.

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Initialize a boto3 S3 client.

     2. List all S3 buckets.

     3. Detect buckets without server-side encryption.

     4. Print the names of unencrypted buckets for logging purposes.

4. Manual Invocation:

   - After saving your function, manually trigger it.

   - Review the Lambda logs to identify the buckets without server-side encryption.

**Assignment 4: Automatic EBS Snapshot and Cleanup Using AWS Lambda and Boto3**

**Objective:** To automate the backup process for your EBS volumes and ensure that backups older than a specified retention period are cleaned up to save costs.

**Task:** Automate the creation of snapshots for specified EBS volumes and clean up snapshots older than 30 days.

**Instructions:**

1. EBS Setup:

   - Navigate to the EC2 dashboard and identify or create an EBS volume you wish to back up.

   - Note down the volume ID.

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach policies that allow Lambda to create EBS snapshots and delete them (`AmazonEC2FullAccess` for simplicity, but be more restrictive in real-world scenarios).

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Initialize a boto3 EC2 client.

     2. Create a snapshot for the specified EBS volume.

     3. List snapshots and delete those older than 30 days.

     4. Print the IDs of the created and deleted snapshots for logging purposes.

4. Event Source (Bonus):

   - Attach an event source, like Amazon CloudWatch Events, to trigger the Lambda function at your desired backup frequency (e.g., every week).

5. Manual Invocation:

   - After saving your function, either manually trigger it or wait for the scheduled event.

   - Go to the EC2 dashboard and confirm that the snapshot is created and old snapshots are deleted.

**Assignment 5: Auto-Tagging EC2 Instances on Launch Using AWS Lambda and Boto3**

**Objective**: Learn to automate the tagging of EC2 instances as soon as they are launched, ensuring better resource tracking and management.

**Task:** Automatically tag any newly launched EC2 instance with the current date and a custom tag.

**Instructions:**

1. EC2 Setup:

   - Ensure you have the capability to launch EC2 instances.

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach the `AmazonEC2FullAccess` policy to this role.

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Initialize a boto3 EC2 client.

     2. Retrieve the instance ID from the event.

     3. Tag the new instance with the current date and another tag of your choice.

     4. Print a confirmation message for logging purposes.

4. CloudWatch Events:

   - Set up a CloudWatch Event Rule to trigger the EC2 instance launch event.

   - Attach the Lambda function as the target.

5. Testing:

   - Launch a new EC2 instance.

   - After a short delay, confirm that the instance is automatically tagged as specified.

**Assignment 6: Monitor and Alert High AWS Billing Using AWS Lambda, Boto3, and SNS**

**Objective:** Create an automated alerting mechanism for when your AWS billing exceeds a certain threshold.

**Task:** Set up a Lambda function to check your AWS billing amount daily, and if it exceeds a specified threshold, send an alert via SNS.

**Instructions:**

1. SNS Setup:

   - Navigate to the SNS dashboard and create a new topic.

   - Subscribe your email to this topic.

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach policies that allow reading CloudWatch metrics and sending SNS notifications.

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Initialize boto3 clients for CloudWatch and SNS.

     2. Retrieve the AWS billing metric from CloudWatch.

     3. Compare the billing amount with a threshold (e.g., $50).

     4. If the billing exceeds the threshold, send an SNS notification.

     5. Print messages for logging purposes.

4. Event Source (Bonus):

   - Attach an event source, like Amazon CloudWatch Events, to trigger the Lambda function daily.

5. Testing:

   - Manually trigger the Lambda function or wait for the scheduled event.

   - If your billing is over the threshold, you should receive an email alert.

**Assignment 7: DynamoDB Item Change Alert Using AWS Lambda, Boto3, and SNS**

**Objective:** Automate the process to receive an alert whenever an item in a DynamoDB table gets updated.

**Task**: Set up a Lambda function that gets triggered when an item in a DynamoDB table is updated and sends an alert via SNS.

**Instructions:**

1. DynamoDB Setup:

   - Navigate to the DynamoDB dashboard and create a table.

   - Add a few items to the table.

2. SNS Setup:

   - Navigate to the SNS dashboard and create a new topic.

   - Subscribe your email to this topic.

3. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach policies that allow Lambda to read DynamoDB Streams and send SNS notifications.

4. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. Extract the modified DynamoDB item from the event.

     2. Send an SNS notification detailing the change.

     3. Log messages for tracking.

5. DynamoDB Stream:

   - Enable DynamoDB Streams on your table and set the view type to "New and old images".

   - Attach the Lambda function to the DynamoDB Stream.

6. Testing:

   - Update an item in your DynamoDB table.

   - Confirm that you receive an SNS alert detailing the change.

**Submission:**

- Provide the Python code used in the Lambda function.

- Document the steps followed.

- Share screenshots of the SNS alert and Lambda logs.

**Assignment 8: Analyze Sentiment of User Reviews Using AWS Lambda, Boto3, and Amazon Comprehend**

**Objective**: Automatically analyze and categorize the sentiment of user reviews using Amazon Comprehend.

**Task:** Set up a Lambda function to receive user reviews, analyze their sentiment using Amazon Comprehend, and log the results.

**Instructions:**

1. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach policies that allow Lambda to use Amazon Comprehend.

2. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created previously.

   - Write the Boto3 Python script to:

     1. Extract the user review from an event.

     2. Use Amazon Comprehend to analyze the sentiment of the review.

     3. Log the sentiment result.

3. Testing:

   - Manually trigger the Lambda function with sample reviews.

   - Confirm the sentiment analysis results in the Lambda logs.

**Assignment 9: Archive Old Files from S3 to Glacier Using AWS Lambda and Boto3**

**Objective:** Automate the archival of files older than a certain age from an S3 bucket to Amazon Glacier for cost-effective storage.

**Task:** Automatically move files in an S3 bucket older than 6 months to Glacier storage class.

**Instructions:**

1. S3 Setup:

   - Navigate to the S3 dashboard and create a bucket.

   - Upload a mix of old and new files to this bucket.

2. Lambda IAM Role:

   - In the IAM dashboard, create a new role for Lambda.

   - Attach the `AmazonS3FullAccess` policy to this role.

3. Lambda Function:

   - Navigate to the Lambda dashboard and create a new function.

   - Choose Python 3.x as the runtime.

   - Assign the IAM role created in the previous step.

   - Write the Boto3 Python script to:

     1. List objects in the S3 bucket.

     2. Identify files older than 6 months.

     3. Change the storage class of identified files to Glacier.

     4. Log the archived files.

4. Testing:

   - Manually trigger the Lambda function or set it to run periodically.

   - Confirm that older files in the S3 bucket are moved to the Glacier storage class.

**Assignment 10: Notify When ELB 5xx Errors Spike Using AWS Lambda, Boto3, and SNS**

**Objective:** To automatically receive notifications when your Elastic Load Balancer (ELB) encounters an unusually high number of 5xx errors.

**Task:** Set up a Lambda function that checks the ELB's 5xx error metrics and sends an SNS notification if errors exceed a certain threshold.

**Instructions:**

1. SNS Setup:

   - In the SNS dashboard, create a new topic.

   - Subscribe your email or phone to this topic.

2. Lambda IAM Role:

   - Create a new role for Lambda with permissions to read CloudWatch metrics and send SNS notifications.

3. Lambda Function:

   - Create a function and assign the above IAM role.

   - Use Boto3 to:

     1. Fetch the 5xx error count metric from CloudWatch for your ELB.

     2. If the count exceeds a threshold (e.g., 10 in the last 5 minutes), trigger an SNS notification.

4. CloudWatch Events:

   - Schedule your Lambda function to run every 5 minutes.

5. Testing:

   - Simulate or wait for a spike in 5xx errors.

   - Confirm receipt of the SNS notification.

**Submission:**

- Provide the Python code for the Lambda function.

- Screenshots of the ELB error metrics and received SNS notifications.

**Assignment 11: EC2 Backup and File Cleanup Using Lambda, Boto3, and S3**

**Objective**: Automatically back up EC2 instance data to S3 and delete backups older than 30 days.

**Task:** Set up a Lambda function to periodically back up specific EC2 instance files/folders to S3 and clear old backups.

**Instructions**:

1. Lambda IAM Role:

   - Create a role with permissions to access EC2 instances, read files, and write to S3.

2. Lambda Function:

   - Create a function with the above IAM role.

   - Use Boto3 to:

     1. Connect to a specific EC2 instance.

     2. Zip and backup specified files/folders to an S3 bucket.

     3. Check for backups older than 30 days in the bucket and delete them.

3. CloudWatch Events:

   - Schedule your Lambda function to run once a day.

4. Testing:

   - Verify that backups are being created in the S3 bucket and that old ones are removed.

**Assignment 12: Auto-Scale EC2 Instances Based on Load Using AWS Lambda, Boto3, and SNS**

**Objective:** Automatically scale up or down the number of EC2 instances based on network load.

**Task:** Set up a Lambda function that checks the network load on an ELB and scales the number of EC2 instances accordingly.

**Instructions:**

1. SNS Setup:

   - Create a new SNS topic.

   - Subscribe to the topic.

2. Lambda IAM Role:

   - Create a role with permissions to read ELB metrics, start/terminate EC2 instances, and send SNS notifications.

3. Lambda Function:

   - Create a function with the above IAM role.

   - Use Boto3 to:

     1. Fetch the network load metric from ELB.

     2. If the load is above a high threshold (e.g., 80% for 5 minutes), launch a new EC2 instance.

     3. If the load is below a low threshold (e.g., 20% for 5 minutes), terminate an EC2 instance.

     4. Notify via SNS about the scaling action taken.

4. CloudWatch Events:

   - Schedule your Lambda function to run every 5 minutes.

5. Testing:

   - Simulate varying network loads.

   - Confirm that EC2 instances are correctly scaled and SNS notifications are received.

**Assignment 13: Audit S3 Bucket Permissions and Notify for Public Buckets**

**Objective:**Automatically audit S3 bucket permissions and send notifications if any buckets have public read or write permissions.

**Task**: Set up a Lambda function to regularly audit S3 bucket permissions and send SNS notifications for any buckets that are publicly accessible.

**Instructions:**

1. SNS Setup:

   - Create a new SNS topic.

   - Subscribe to the topic with your email.

2. Lambda IAM Role:

   - Create a role with permissions to list S3 bucket permissions and send SNS notifications.

3. Lambda Function:

   - Create a function with the above IAM role.

   - Use Boto3 to:

     1. List all S3 buckets.

     2. For each bucket, check its permission settings.

     3. If a bucket has public read or write permissions, send a notification with its name via SNS.

4. CloudWatch Events:

   - Schedule your Lambda function to run daily.

5. Testing:

   - Make one or two of your S3 buckets public.

   - Run the Lambda function and ensure you receive appropriate SNS notifications.

**Assignment 14: Monitor EC2 Instance State Changes Using AWS Lambda, Boto3, and SNS**

**Objective:** Automatically monitor changes in EC2 instance states and send notifications whenever an instance is started or stopped.

**Task:** Set up a Lambda function that listens to EC2 state change events and sends SNS notifications detailing the state changes.

**Instructions:**

1. SNS Setup:

   - Navigate to the SNS dashboard and create a new topic.

   - Subscribe to this topic with your email.

2. Lambda IAM Role:

   - Create a role with permissions to read EC2 instance states and send SNS notifications.

3. Lambda Function:

   - Create a function and assign the above IAM role.

   - Use Boto3 to:

     1. Extract details from the event regarding the EC2 state change.

     2. Send an SNS notification with details about which EC2 instance changed state and the new state (e.g., started, stopped).

4. EC2 Event Bridge (formerly CloudWatch Events):

   - Set up an Event Bridge rule to trigger your Lambda function whenever an EC2 instance state changes.

5. Testing:

   - Start or stop one of your EC2 instances.

   - Confirm you receive an SNS notification about the state change.

**Assignment 15: Implement a Log Cleaner for S3**

**Objective:**Create a Lambda function that automatically deletes logs in a specified S3 bucket that are older than 90 days.

**Instructions:**

1. Create a new Lambda function.

2. Using Boto3, configure the function to:

   1. Access the specified S3 bucket.  
   2. List all the log files.  
   3. Check the age of each log.  
   4. Delete logs older than 90 days.

3. Schedule this function to run weekly using AWS EventBridge.

**Assignment 16: Automated SNS Alerts for EC2 Disk Space Utilization**

**Objective:** Set up a Lambda function that checks EC2 instances for disk space utilization, sending an SNS alert if utilization exceeds 85%.

**Instructions:**

1. Create a Lambda function.

2. Using Boto3, the function should:

   1. Connect to EC2 instances.  
   2. Check disk space utilization.  
   3. If disk space utilization exceeds 85%, publish a message to an SNS topic.

3. Set up a CloudWatch event to trigger this Lambda function daily.

**Assignment 17: Restore EC2 Instance from Snapshot**

**Objective:** Automate the process of creating a new EC2 instance from the latest snapshot using a Lambda function.

**Instructions:**

1. Create a Lambda function.

2. Using Boto3, the function should:

   1. Fetch the most recent snapshot of a given EC2 instance.  
   2. Create a new EC2 instance using the fetched snapshot.

3. Trigger this Lambda function manually or on a schedule, depending on your recovery requirements.

**Assignment 18: Autosave EC2 Instance State Before Shutdown**

**Objective:**Before an EC2 instance is shut down, automatically save its current state to an S3 bucket.

**Instructions:**

1. Create a Lambda function.

2. Using Boto3, the function should:

   1. Detect when an EC2 instance is about to be terminated.  
   2. Save the current state or necessary files from the EC2 instance to an S3 bucket.

3. Use CloudWatch Events to trigger this Lambda function whenever an EC2 termination command is detected.

**Assignment 19: Load Balancer Health Checker**

**Objective:**Design a Lambda function that checks the health of registered instances behind an Elastic Load Balancer (ELB) and notifies via SNS if any instances are unhealthy.

**Instructions:**

1. Create a Lambda function.

2. With Boto3, configure the function to:

1. Check the health of registered instances behind a given ELB.

2. If any instances are found to be unhealthy, publish a detailed message to an SNS topic.

3. Set up a CloudWatch event to trigger this Lambda function every 10 minutes.

**Submission Instructions:**

To submit your assignment, please follow these guidelines:

- Ensure that your assignment is fully completed.

- Push your code to a GitHub repository.

- Share the repository link by including it in a text, Word, or PDF file format.

Submit the file/text containing the repository link via Vlearn.