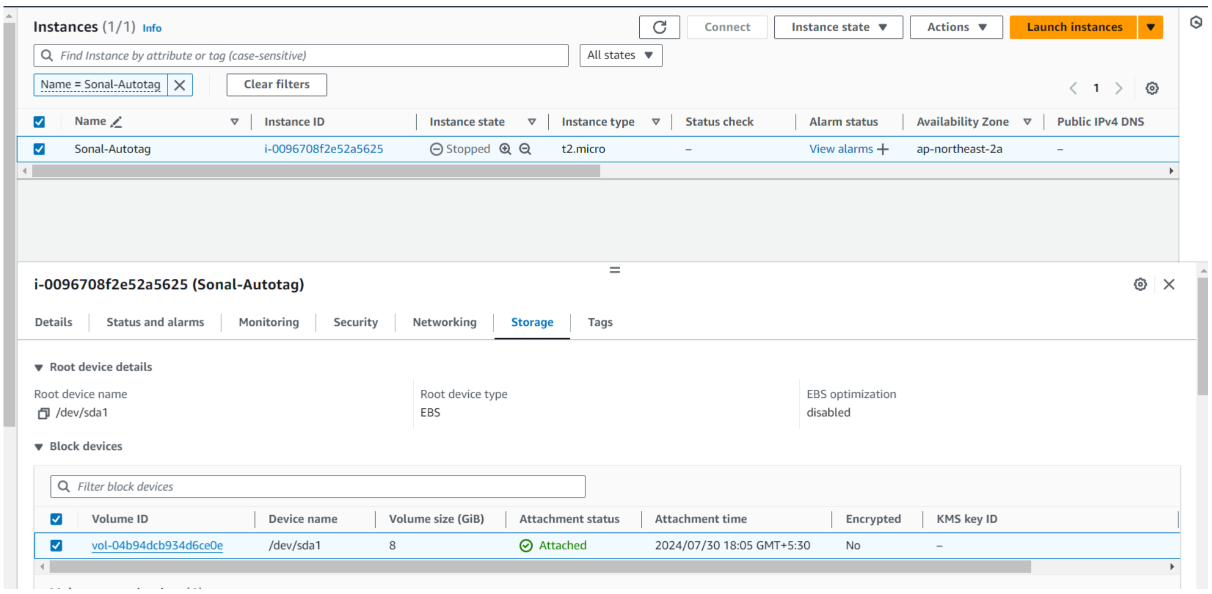
Assignment 4: Automatic EBS Snapshot and Cleanup Using AWS Lambda and Boto3

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.

Step 1. EBS Setup

- Navigate to the EC2 Dashboard and Identify or Create an EBS Volume
- Note down the volume ID



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Sonal-Autotag	i-0096708f2e52a5625	Stopped	t2.micro	–	View alarms	ap-northeast-2a	–

i-0096708f2e52a5625 (Sonal-Autotag)

Root device details

Root device name	Root device type	EBS optimization
/dev/sda1	EBS	disabled

Block devices

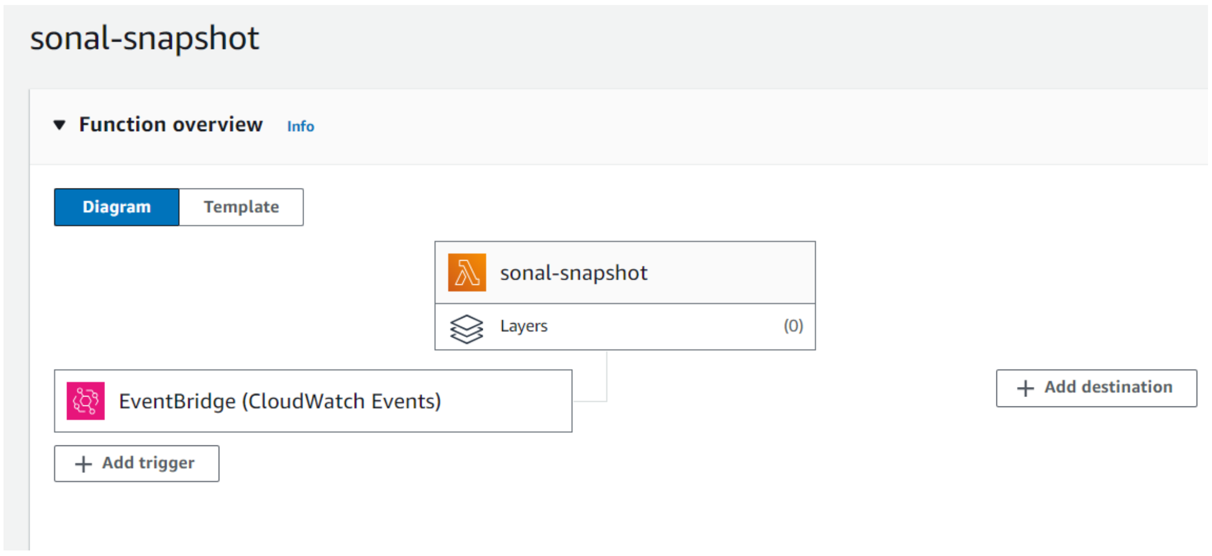
Volume ID	Device name	Volume size (GiB)	Attachment status	Attachment time	Encrypted	KMS key ID
vol-04b94dcb934d6ce0e	/dev/sda1	8	Attached	2024/07/30 18:05 GMT+5:30	No	–

Step 2. Lambda IAM Role

- Navigate to the IAM Dashboard and create role for Lambda
- Attach the AmazonEC2FullAccess policy

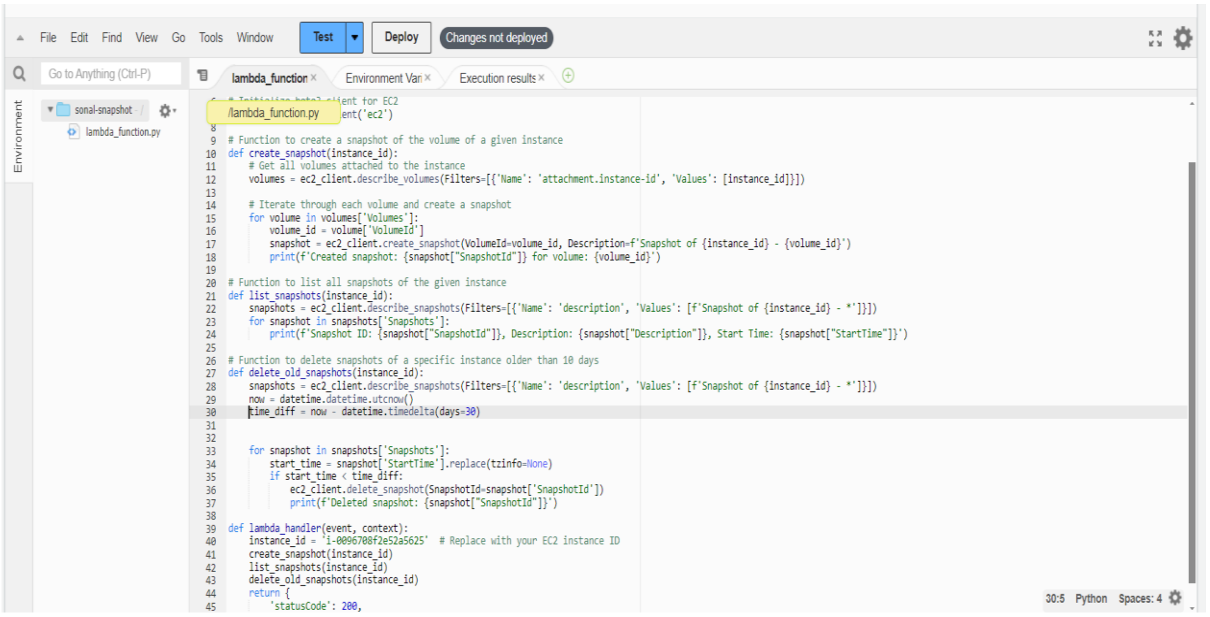
Step 3. Lambda Function

- Navigate to the Lambda Dashboard and Create a New Function
- Choose Python 3.11 as the runtime.
- Under Permissions, select Use an existing role and choose the role created in the previous step



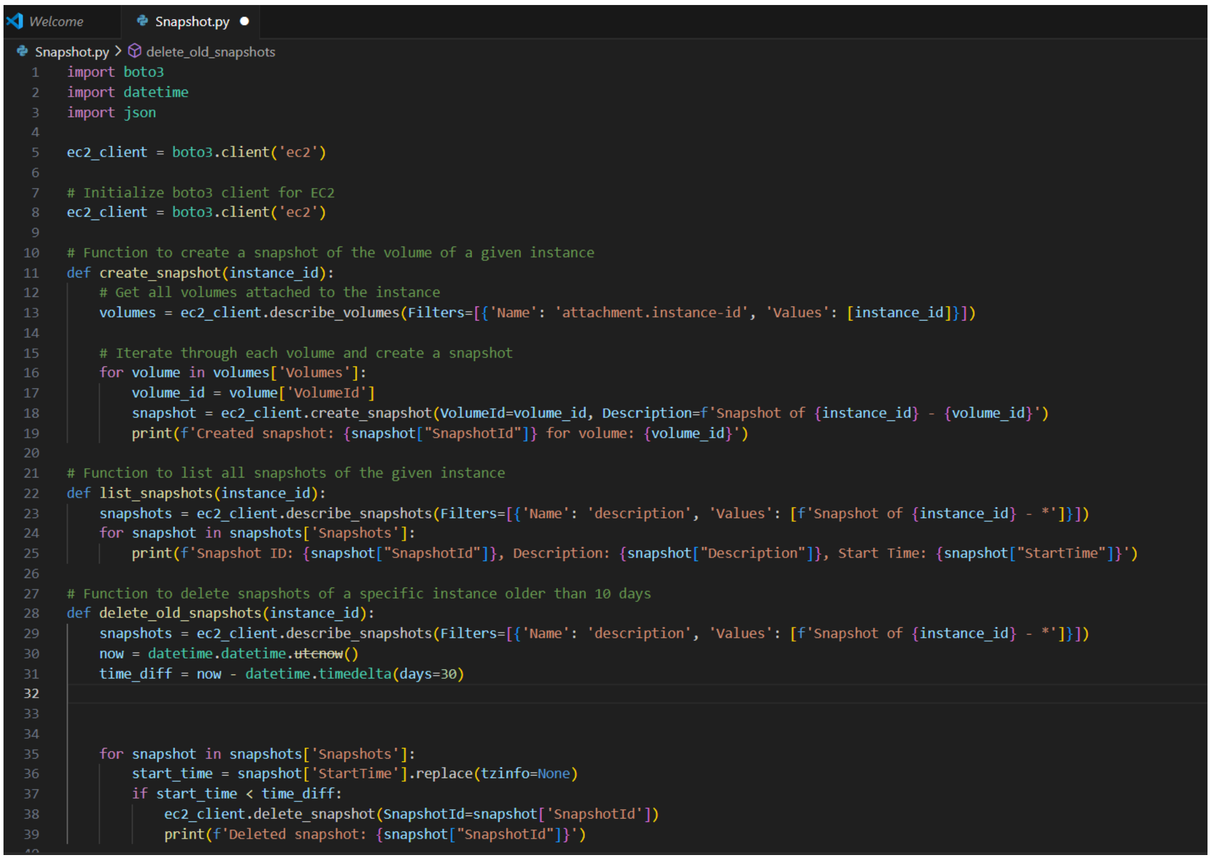
Step 4. Write the Boto3 Python script to:

- Initialize a boto3 EC2 client.
- Create a snapshot for the specified EBS volume.
- List snapshots and delete those older than 30 days.
- Print the IDs of the created and deleted snapshots for logging purposes.

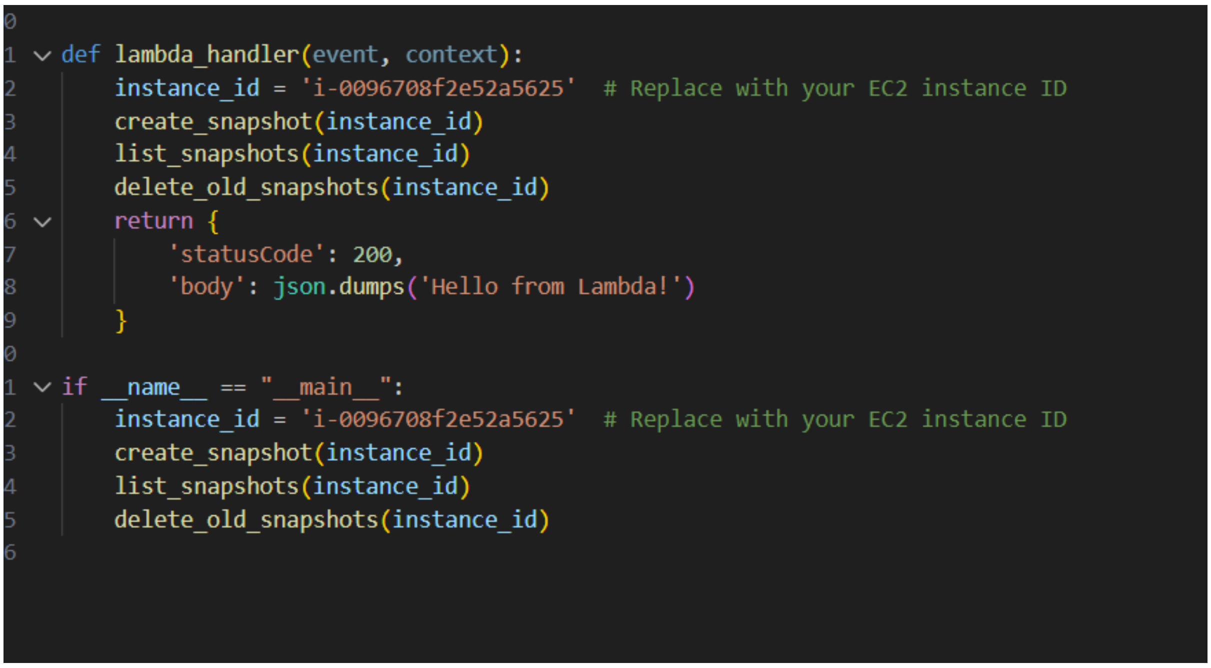


```python
# Initialize boto3 client for EC2
ent('ec2')

# Function to create a snapshot of the volume of a given instance
def create_snapshot(instance_id):
    # Get all volumes attached to the instance
    volumes = ec2_client.describe_volumes(Filters=[{'Name': 'attachment.instance-id', 'Values': [instance_id]}])

    # Iterate through each volume and create a snapshot
    for volume in volumes['Volumes']:
        volume_id = volume['VolumeId']
        snapshot = ec2_client.create_snapshot(VolumeId=volume_id, Description=f'Snapshot of {instance_id} - {volume_id}')
        print(f'Created snapshot: {snapshot["SnapshotId"]} for volume: {volume_id}')

# Function to list all snapshots of the given instance
def list_snapshots(instance_id):
    snapshots = ec2_client.describe_snapshots(Filters=[{'Name': 'description', 'Values': [f'Snapshot of {instance_id} - *']}])
    for snapshot in snapshots['Snapshots']:
        print(f'Snapshot ID: {snapshot["SnapshotId"]}, Description: {snapshot["Description"]}, Start Time: {snapshot["StartTime"]}')

# Function to delete snapshots of a specific instance older than 10 days
def delete_old_snapshots(instance_id):
    snapshots = ec2_client.describe_snapshots(Filters=[{'Name': 'description', 'Values': [f'Snapshot of {instance_id} - *']}])
    now = datetime.datetime.utcnow()
    time_diff = now - datetime.timedelta(days=30)


    for snapshot in snapshots['Snapshots']:
        start_time = snapshot['StartTime'].replace(tzinfo=None)
        if start_time < time_diff:
            ec2_client.delete_snapshot(SnapshotId=snapshot['SnapshotId'])
            print(f'Deleted snapshot: {snapshot["SnapshotId"]}')

def lambda_handler(event, context):
    instance_id = 'i-0096708f2e52a5625'  # Replace with your EC2 instance ID
    create_snapshot(instance_id)
    list_snapshots(instance_id)
    delete_old_snapshots(instance_id)
    return {
        'statusCode': 200,
```

Python script:



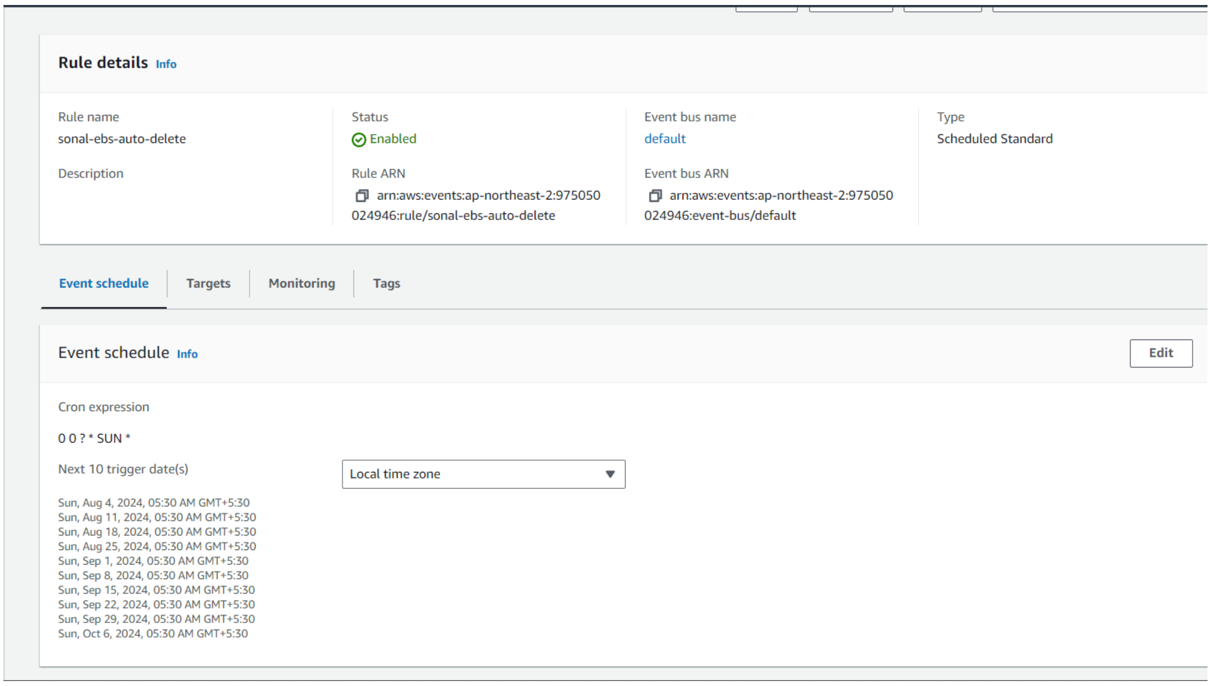
```python
import boto3
import datetime
import json

ec2_client = boto3.client('ec2')

# Initialize boto3 client for EC2
ec2_client = boto3.client('ec2')

# Function to create a snapshot of the volume of a given instance
def create_snapshot(instance_id):
    # Get all volumes attached to the instance
    volumes = ec2_client.describe_volumes(Filters=[{'Name': 'attachment.instance-id', 'Values': [instance_id]}])

    # Iterate through each volume and create a snapshot
    for volume in volumes['Volumes']:
        volume_id = volume['VolumeId']
        snapshot = ec2_client.create_snapshot(VolumeId=volume_id, Description=f'Snapshot of {instance_id} - {volume_id}')
        print(f'Created snapshot: {snapshot["SnapshotId"]} for volume: {volume_id}')

# Function to list all snapshots of the given instance
def list_snapshots(instance_id):
    snapshots = ec2_client.describe_snapshots(Filters=[{'Name': 'description', 'Values': [f'Snapshot of {instance_id} - *']}])
    for snapshot in snapshots['Snapshots']:
        print(f'Snapshot ID: {snapshot["SnapshotId"]}, Description: {snapshot["Description"]}, Start Time: {snapshot["StartTime"]}')

# Function to delete snapshots of a specific instance older than 10 days
def delete_old_snapshots(instance_id):
    snapshots = ec2_client.describe_snapshots(Filters=[{'Name': 'description', 'Values': [f'Snapshot of {instance_id} - *']}])
    now = datetime.datetime.~~utcnow~~()
    time_diff = now - datetime.timedelta(days=30)



    for snapshot in snapshots['Snapshots']:
        start_time = snapshot['StartTime'].replace(tzinfo=None)
        if start_time < time_diff:
            ec2_client.delete_snapshot(SnapshotId=snapshot['SnapshotId'])
            print(f'Deleted snapshot: {snapshot["SnapshotId"]}')
```



```python

def lambda_handler(event, context):
    instance_id = 'i-0096708f2e52a5625'  # Replace with your EC2 instance ID
    create_snapshot(instance_id)
    list_snapshots(instance_id)
    delete_old_snapshots(instance_id)
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }

if __name__ == "__main__":
    instance_id = 'i-0096708f2e52a5625'  # Replace with your EC2 instance ID
    create_snapshot(instance_id)
    list_snapshots(instance_id)
    delete_old_snapshots(instance_id)

```

Step 5. Event Source:

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



Rule name sonal-ebs-auto-delete	Status ✓ Enabled	Event bus name default	Type Scheduled Standard
Description	Rule ARN arn:aws:events:ap-northeast-2:975050024946:rule/sonal-ebs-auto-delete	Event bus ARN arn:aws:events:ap-northeast-2:975050024946:event-bus/default	

Cron expression

0 0 ? * SUN *

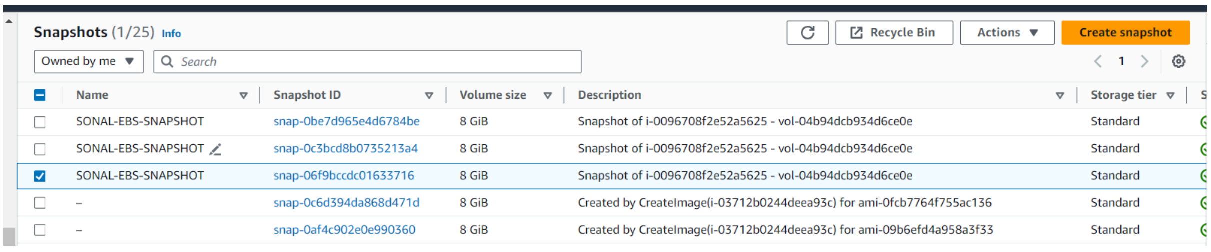
Next 10 trigger date(s)

Local time zone

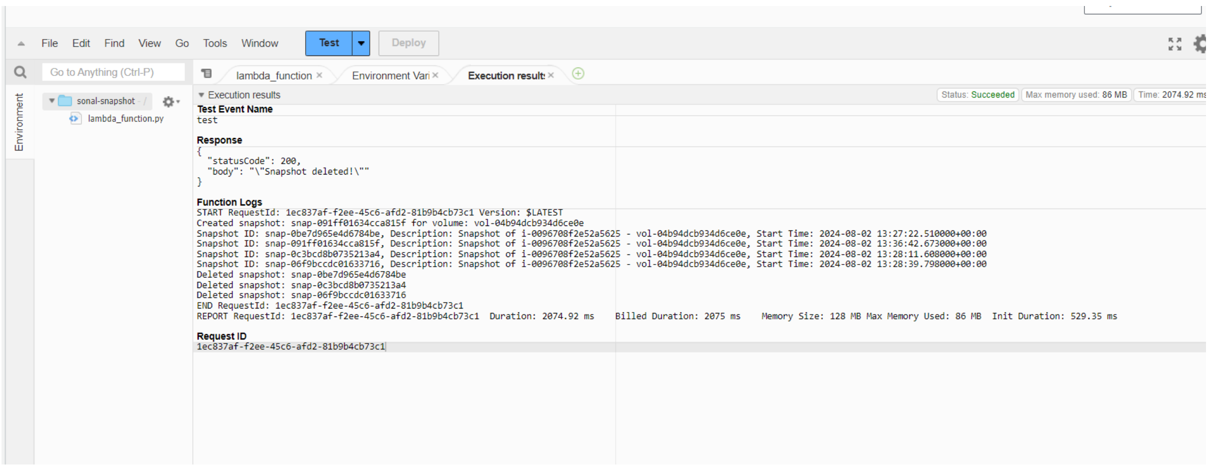
```
Sun, Aug 4, 2024, 05:30 AM GMT+5:30
Sun, Aug 11, 2024, 05:30 AM GMT+5:30
Sun, Aug 18, 2024, 05:30 AM GMT+5:30
Sun, Aug 25, 2024, 05:30 AM GMT+5:30
Sun, Sep 1, 2024, 05:30 AM GMT+5:30
Sun, Sep 8, 2024, 05:30 AM GMT+5:30
Sun, Sep 15, 2024, 05:30 AM GMT+5:30
Sun, Sep 22, 2024, 05:30 AM GMT+5:30
Sun, Sep 29, 2024, 05:30 AM GMT+5:30
Sun, Oct 6, 2024, 05:30 AM GMT+5:30
```

Step 6. 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.



	Name	Snapshot ID	Volume size	Description	Storage tier
☐	SONAL-EBS-SNAPSHOT	snap-0be7d965e4d6784be	8 GiB	Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e	Standard
☐	SONAL-EBS-SNAPSHOT	snap-0c3bcd8b0735213a4	8 GiB	Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e	Standard
☑	SONAL-EBS-SNAPSHOT	snap-06f9bccdc01633716	8 GiB	Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e	Standard
☐	–	snap-0c6d394da868d471d	8 GiB	Created by CreateImage(i-03712b0244deea93c) for ami-0fcb7764f755ac136	Standard
☐	–	snap-0af4c902e0e990360	8 GiB	Created by CreateImage(i-03712b0244deea93c) for ami-09b6efd4a958a3f33	Standard



Status: Succeeded | Max memory used: 86 MB | Time: 2074.92 ms

Test Event Name

```
test
```

Response

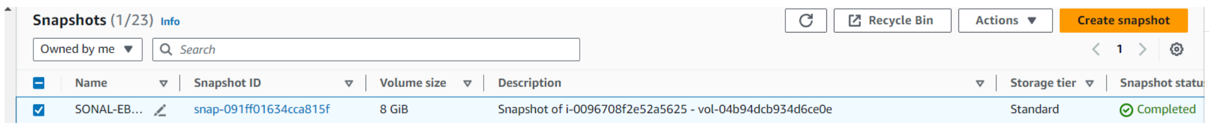
```
{
  "statusCode": 200,
  "body": "\"Snapshot deleted!\""
}
```

Function Logs

```
START RequestId: 1ec837af-f2ee-45c6-afd2-81b9b4cb73c1 Version: $LATEST
Created snapshot: snap-091ff01634cca815f for volume: vol-04b94dcb934d6ce0e
Snapshot ID: snap-0be7d965e4d6784be, Description: Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e, Start Time: 2024-08-02 13:27:22.510000+00:00
Snapshot ID: snap-091ff01634cca815f, Description: Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e, Start Time: 2024-08-02 13:36:42.673000+00:00
Snapshot ID: snap-0c3bcd8b0735213a4, Description: Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e, Start Time: 2024-08-02 13:28:11.608000+00:00
Snapshot ID: snap-06f9bccdc01633716, Description: Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e, Start Time: 2024-08-02 13:28:39.798000+00:00
Deleted snapshot: snap-0be7d965e4d6784be
Deleted snapshot: snap-0c3bcd8b0735213a4
Deleted snapshot: snap-06f9bccdc01633716
END RequestId: 1ec837af-f2ee-45c6-afd2-81b9b4cb73c1
REPORT RequestId: 1ec837af-f2ee-45c6-afd2-81b9b4cb73c1  Duration: 2074.92 ms    Billed Duration: 2075 ms    Memory Size: 128 MB Max Memory Used: 86 MB  Init Duration: 529.35 ms
```

Request ID

```
1ec837af-f2ee-45c6-afd2-81b9b4cb73c1
```



	Name	Snapshot ID	Volume size	Description	Storage tier	Snapshot statu
✓	SONAL-EB...	snap-091ff01634cca815f	8 GiB	Snapshot of i-0096708f2e52a5625 - vol-04b94dcb934d6ce0e	Standard	✓ Completed