AWS mini-project

**SnapBot:**

**Snapshot Automation System**

**By**

**Aravind U R**

**‘SnapBot’** is an automated snapshot management system designed to streamline the creation and deletion of snapshots within AWS environments. Utilizing AWS Lambda, Amazon EventBridge, and SNS notifications; SnapBot provides a fully automated solution for handling snapshots of Amazon EC2 instances and EBS volumes. This serverless system ensures consistent data backup, reduces manual intervention, and optimizes storage usage and costs, while keeping users informed with real-time notifications.

**Architecture Overview:**

SnapBot operates using three key AWS services:

* **AWS Lambda**: Executes functions to automate snapshot creation and deletion in response to scheduled or event-driven triggers.
* **Amazon EventBridge**: Handles scheduling and event-based rules to trigger Lambda functions at predefined intervals or based on system activity.
* **Amazon SNS (Simple Notification Service)**: Provides real-time alerts and notifications for snapshot events, including creation, deletion, and any potential errors.

**AWS EBS Snapshot**

Amazon Elastic Block Store (EBS) snapshots are point-in-time backups of EBS volumes. Snapshots are stored in Amazon S3, but they are not visible as S3 objects. They capture all the data that has been written to the volume at the time the snapshot is taken, which makes them useful for backup, disaster recovery, and data migration. Here are some key details about EBS snapshots:

**Features of EBS Snapshots:**

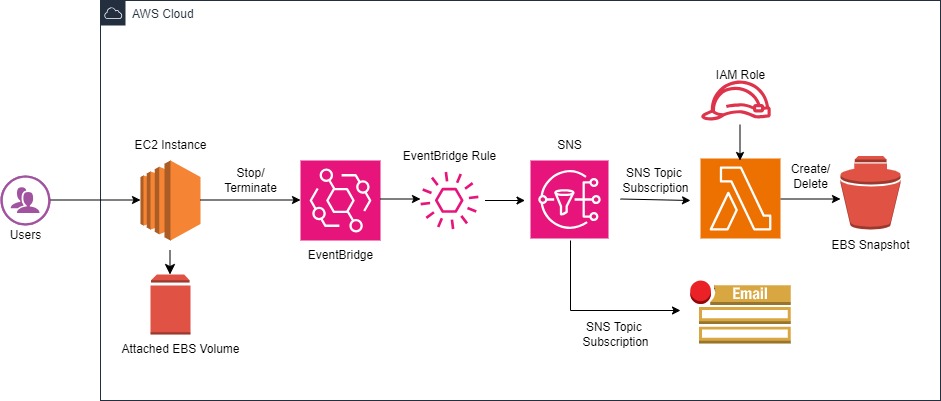
1. **Incremental Backups**: EBS snapshots are incremental, meaning only the data that has changed since the last snapshot is saved, reducing storage costs.
2. **Ease of Use**: You can create a snapshot of an EBS volume at any time. Snapshots can be copied across AWS regions.
3. **Encryption**: Snapshots of encrypted volumes are automatically encrypted. When you create a volume from an encrypted snapshot, the new volume is also encrypted.
4. **Restoration**: You can create a new EBS volume from a snapshot, allowing for easy recovery or cloning of your volumes.
5. **Automation**: AWS supports the automation of snapshot creation using tools like **Data Lifecycle Manager** or scripting with AWS CLI or SDKs.

**Common Use Cases:**

* **Backup**: Regular snapshots provide a way to back up critical data.
* **Disaster Recovery**: Snapshots can be copied to different regions for redundancy and recovery in case of regional failures.
* **Data Migration**: Snapshots can be used to clone or move volumes to different regions or accounts.
* **Scaling**: Snapshots allow quick replication of volumes for horizontal scaling of infrastructure.

Let’s Begin!

**Architectural Diagram**



**Solution Configuration**

Let’s start!

**1. Launching an EC2 instance**

launching an EC2instance with the following specifications:

* Name: DemoInstance
* AMI: Amazon Linux 2
* Instance Type: t2.micro
* Proceed without a keypair (we will not be SSHing)
* Auto-Assign public IP: Enable
* Create new Security group (DemoInstance -SG) allow SSH from Anywhere.

copy and save the**Instance ID**as we are going to need it later on.

**2. IAM Role Creation**

**Role for Snapshot Creation Lambda Function**

Go to IAM and create a policy named “**Snapshot-Create-Policy”** via the JSON tab with the following:



Json link for **Snapshot-Create-Policy:**

***<https://github.com/cloud-aravind/My-AWS-Projects/blob/main/SnapShot%20-%20Snapshot%20Automation%20System/Delete-Snapshot-Policy.json>***

We need to create a role now. Go to **Roles** then on**Create role**. For Trusted entity type select **AWS service** and for Use case select **Lambda**.

Click**Next**. In the **Add permissions** screen search for the policy we just created. Select it and click**Next**. In the next screen type in “**Snapshot-Create-Role”** for the name then click on **Create role**.

**Role for Snapshot Deletion Lambda Function**

Go to IAM and create a policy named “**Snapshot-Delete-Policy”** via the JSON tab with the following:



Json link for **Snapshot-Delete-Policy:**

[***https://github.com/cloud-aravind/My-AWS-Projects/blob/main/SnapShot%20-%20Snapshot%20Automation%20System/Delete-Snapshot-Policy.json***](https://github.com/cloud-aravind/My-AWS-Projects/blob/main/SnapShot%20-%20Snapshot%20Automation%20System/Delete-Snapshot-Policy.json)

Go to **Roles** then on**Create role**. For Trusted entity type select **AWS service** and for Use case select **Lambda**.

Click**Next**. In the **Add permissions** screen search for the policy we just created. Select it and click**Next**. In the next screen type in “**Snapshot-Create-Role”** for the name then click on **Create role**.

**3. Create Lambda Functions**

**For Snapshot creation:**

Now navigate to**Lambda** and start creating a function for Snapshot creation. Give the function the name of “**EBSSnapshotCreate”** with a Runtime of **Python 3.9**.

For Execution role select **Use an existing role**. In the **Existing role** box, select the **Snapshot-Create-Role**. Click on **Create function**.

Now, Paste the Python code in the lambda function.

Python Code link:

[***https://github.com/cloud-aravind/My-AWS-Projects/blob/main/SnapShot%20-%20Snapshot%20Automation%20System/EBSSnapshotCreate.py***](https://github.com/cloud-aravind/My-AWS-Projects/blob/main/SnapShot%20-%20Snapshot%20Automation%20System/EBSSnapshotCreate.py)

Paste the following into the Lambda function code block then click the **Deploy** button***.***

Copy the **Lambda function ARN** and save it for later.

**For Snapshot Deletion:**

Let’s create another Lambda function for Snapshot deletion.

Give the function the name of “**EBSSnapshotCreate”** with a Runtime of **Python 3.9**.

For Execution role select **Use an existing role**. In the **Existing role** box, select the **Snapshot-Delete-Role**. Click on **Create function**.

Paste the Python code in the lambda function.

Python Code link:

[***https://github.com/cloud-aravind/My-AWS-Projects/blob/main/SnapShot%20-%20Snapshot%20Automation%20System/EBSSnapshotDelete.py***](https://github.com/cloud-aravind/My-AWS-Projects/blob/main/SnapShot%20-%20Snapshot%20Automation%20System/EBSSnapshotDelete.py)

Paste the following into the Lambda function code block then click the **Deploy** button***.***

Copy the **Lambda function ARN** and save it for later.

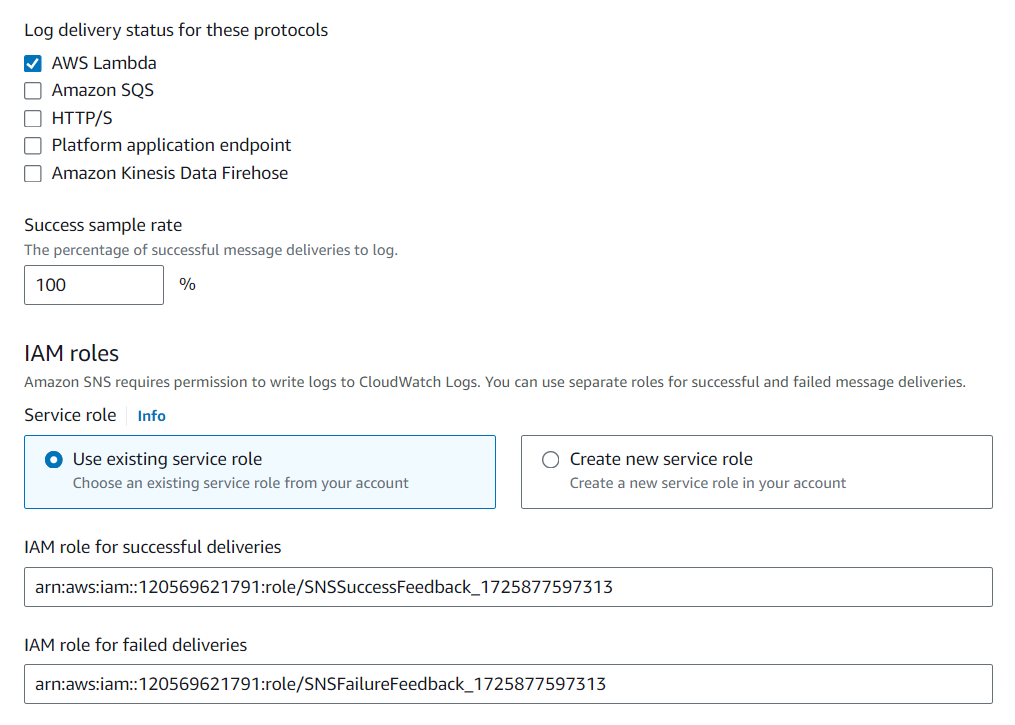
**4. SNS Topic creation**

**For Snapshot creation:**

Now go to the **SNS** console and click on**Topics**, **Create Topic.**

For Type select **Standard** and name it **“Snapshot-Create-Topic”.**

Scroll down and expand the **Delivery status logging** section. In there, select **AWS Lambda**,**Create new service role** then click on **Create new roles**. Clicking on**Create new roles** will redirect you to another tab for**IAM**. In that tab, leave everything as default and click**Create role**.



You can click on **Create topic** now.

**For Snapshot Deletion:**

Create another topic for Snapshot deletion and name it **“Snapshot-Delete-Topic”**

Follow the same steps as mentioned above.

**5. Create Subscriptions**

On this Subscriptions section we will be creating **4 subscriptions**. 2 subscriptions for each topic;

1. For invoke lambda functions
2. For Email notifications

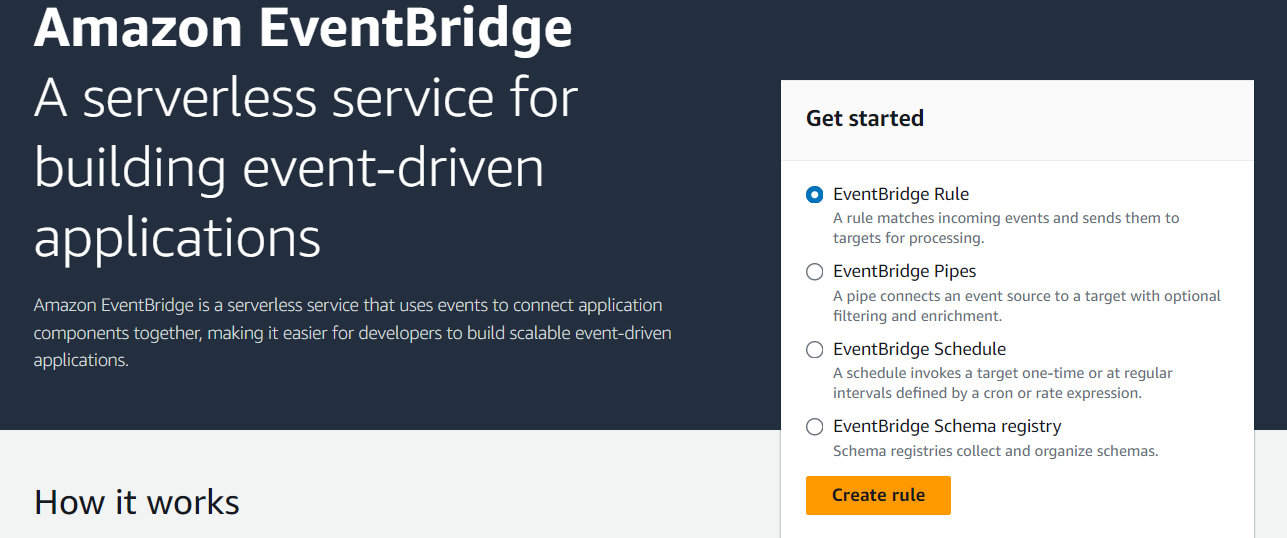
Let’s create the first one.

Click on **Create subscription**. For**Topic ARN** choose the topic we created- **Snapshot-Create-Topic and Snapshot-Delete-Topic**. For Protocol select **AWS Lambda** and paste in the **Lambda function ARN** into the Endpoint box – choose specific functions for corresponding topics. Click on **Create subscription**.

Let’s create the second subscription. We will choose the same topics**ARN**. This time for Protocol, choose**email** and enter your email address in the **Endpoint** box. After you create this second subscription, you should receive an email asking if you would like to subscribe for both the topics. **Please confirm the subscription**.

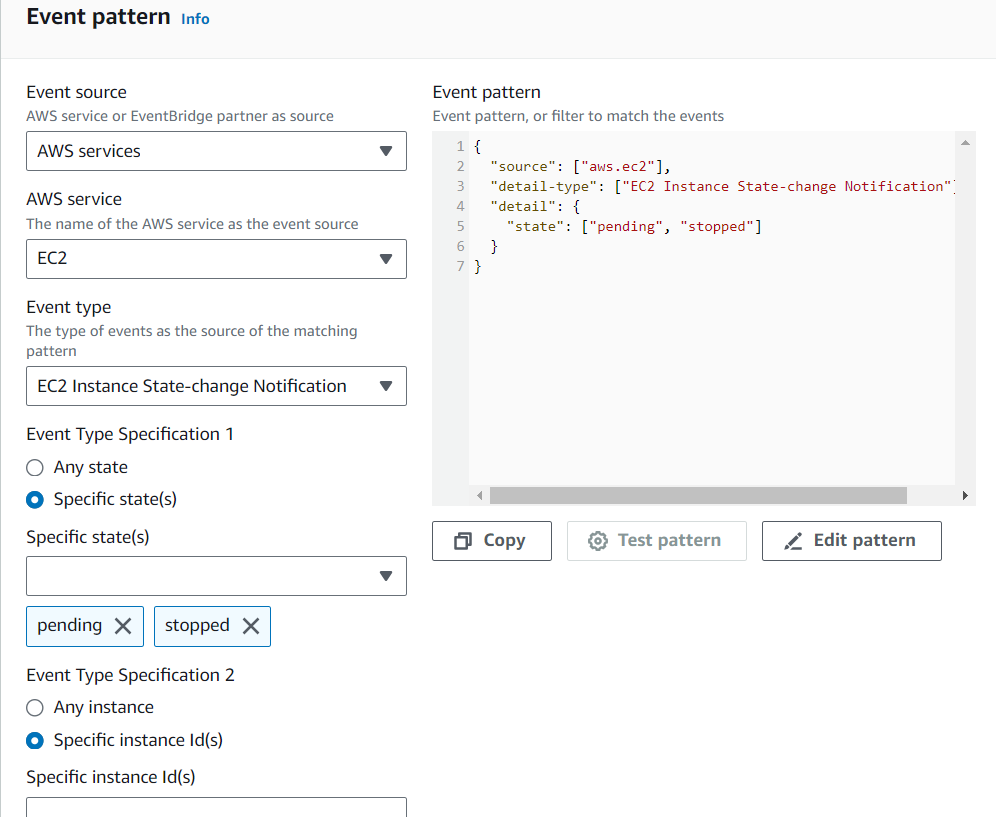
**6. Create EventBridge rule**

* The last step we have to perform is creating a rule within **Amazon EventBridge**. On the **EventBridge** screen, select **EventBridge Rule** then click **Create rule**.



* On the next screen name the rule **Snapshot\_Create\_Rule**, select **Rule with an event pattern** then click**Next**.
* For **Snapshot\_Create\_Rule.**

In the **Event pattern** section, select **AWS services**, **EC2**, and **EC2 Instance State-change Notification**. In the area below, click on **Specific state(s)** and select **stopped** and **pending**. For**Specific instance Id(s)**, paste in the **instance ID** from earlier then click **Next**.



* For **Snapshot\_Delete\_Rule.**

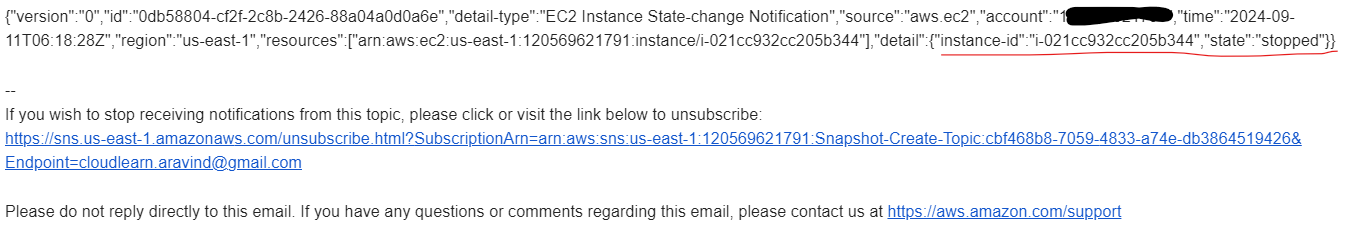
In the **Event pattern** section, select **AWS services**, **EC2**, and **EC2 Instance State-change Notification**. In the area below, click on **Specific state(s)** and select **Terminated**. For**Specific instance Id(s)**, paste in the **instance ID** from earlier then click **Next**.

* In the following screen, select**SNS Topic** for the target and then choose the Topic we created. Keep clicking next and **Create the rule**.
* In the following screen, select**the specific SNS Topic** for the target and then choose the Topic we created. Keep clicking next and **Create the rule**.

The configuration parts were completed here.

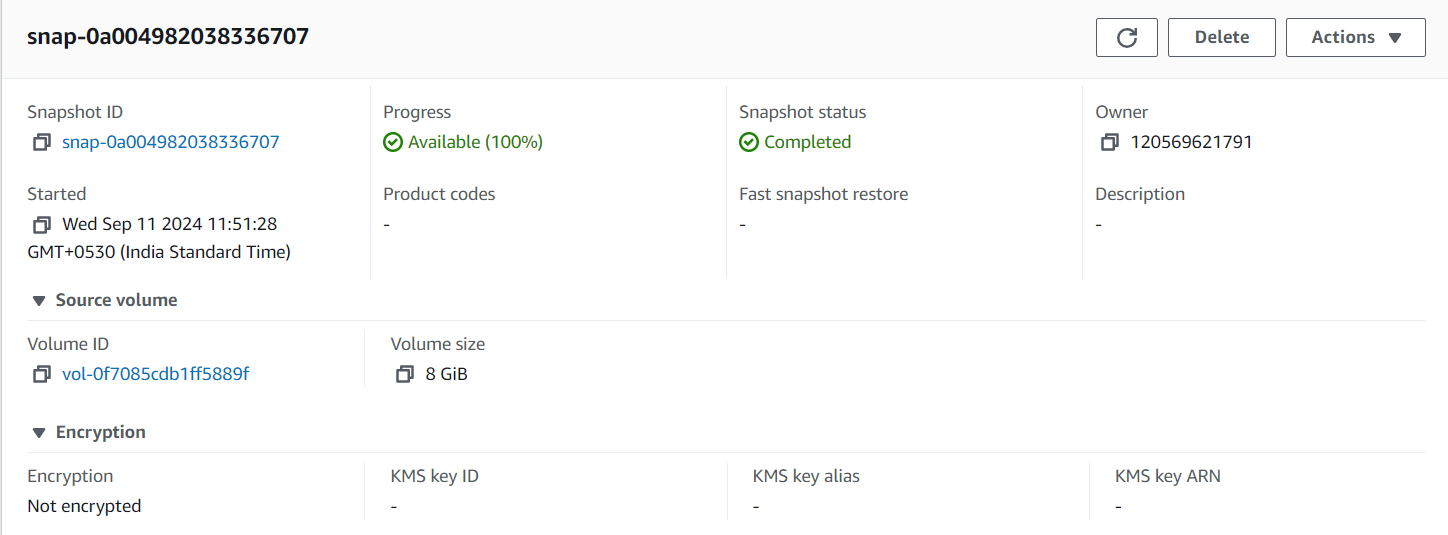
We can test our **Lambda function** and **EventBridge** rule by going over to the **EC2** console and **stopping** the instance.

If everything was done correctly, 2 things should happen. The first is that we should receive an email stating that the**EC2** has been **stopped**.

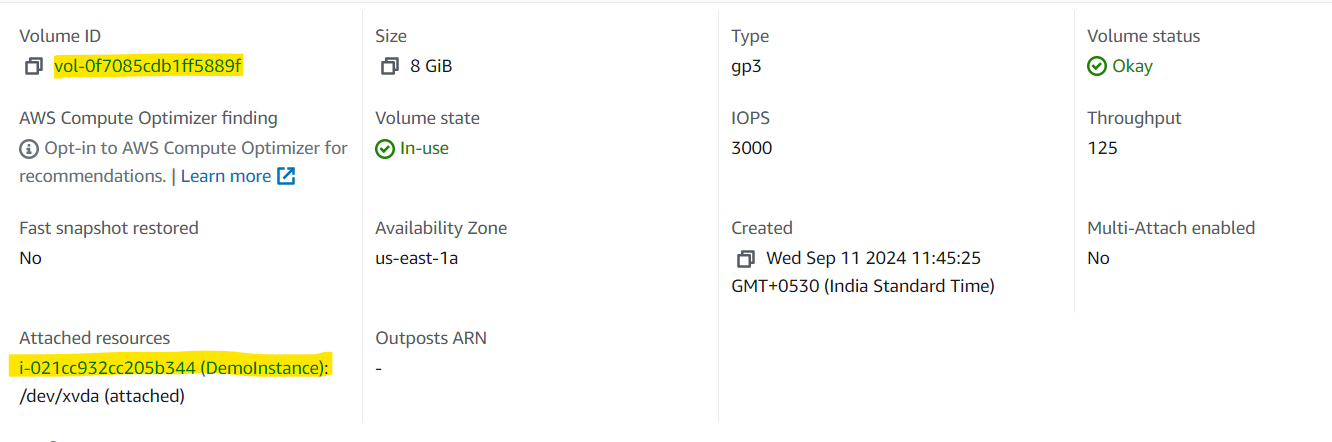


Secondly, a snapshot of the instance should be in**progress** or **completed**.





We can verify here the source volume id is same as the volume id of our **DemoInstance**.

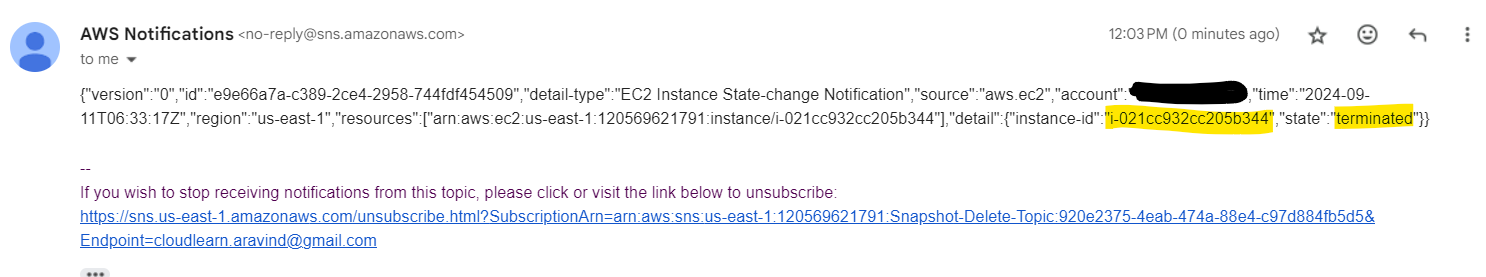


Here is also our Lambda function creates and CloudWatch logs about the Snapshot creation process.

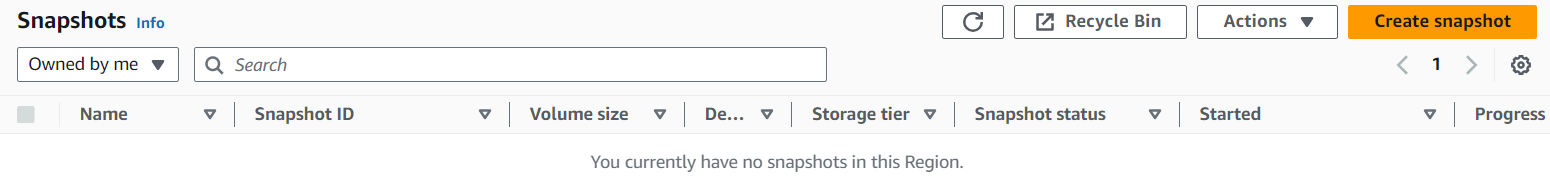


We can test our **Lambda function** and **EventBridge** rule by going over to the **EC2** console and **Terminate the Instance** the instance.

2 things should happen. The first is that we should receive an email stating that the**EC2** has been **Terminated**.



Secondly, the snapshot of the instance is deleted.



If both actions took place for you then congratulations, our automation system is working fine.

**Conclusion**

SnapBot is a powerful and cost-efficient solution for AWS snapshot management, integrating seamlessly with AWS Lambda and EventBridge to automate the entire snapshot lifecycle. By leveraging serverless technologies, SnapBot provides robust, scalable, and highly customizable snapshot creation and deletion, ensuring optimal resource management and cost control within AWS environments.

**Use Case Scenarios:**

* **Data Backup and Recovery**: SnapBot ensures regular snapshots of critical AWS resources, providing a reliable recovery point in case of failure or disaster.
* **Cost Management**: By automating snapshot deletion, SnapBot reduces the risk of high storage costs caused by forgotten or unnecessary backups.
* **Compliance and Audit**: Organizations with strict data retention policies can use SnapBot to enforce automatic deletion of snapshots after a compliance-defined duration, ensuring data management aligns with regulatory requirements.

**Benefits:**

* **Operational Efficiency**: Automates both creation and deletion of snapshots, reducing manual work and human errors.
* **Cost Optimization**: Serverless architecture with AWS Lambda eliminates the need for dedicated infrastructure, while automated deletion prevents excess storage costs.
* **Scalability**: SnapBot easily handles environments of any size, from small development setups to large enterprise-scale infrastructures.
* **Reliability**: EventBridge and Lambda provide reliable event-driven execution, ensuring snapshots are consistently created and deleted according to policy.

References**:**

* <https://medium.com/towards-aws/aws-devops-pro-lab-5-automating-ebs-snapshots-with-amazon-eventbridge-e50e85a4d823>
* <https://medium.com/@amanpathakdevops/unleashing-cloud-savings-turbo-charge-your-aws-cost-optimization-with-ebs-snapshot-management-ab4c20bd470d>
* <https://github.com/cloud-aravind/My-AWS-Projects/tree/main/SnapShot%20-%20Snapshot%20Automation%20System>