**LAMBDA FUNCTIONS FOR COST-OPTIMIZATION**

**Problems:**

If there are 100 of servers running in an organizations. Each server will have its own volume and there will be many backups created from those servers. If the work completed by the employees they will terminate those servers. While terminating those servers volumes will also deleted along with the servers or done manually. But the snapshots will be retained in the storage. There might be chance of forgetting to delete the snapshots due to huge number of workloads. Then the service provider will charge for the snapshot stored. It will increase the cost of cloud in the organization.

**Solution:**

To use the services and resources efficiently and optimize the cost we can use lambda function to trigger an event driven execution.

**Overview:**

To implement cost optimization on AWS using Lambda functions using python as a runtime which is done manually. But we can do the same task as event driven also by triggering it.

**CODE:**

**import boto3**

**def lambda\_handler(event, context):**

**ec2 = boto3.client('ec2')**

**# Get all EBS snapshots**

**response = ec2.describe\_snapshots(OwnerIds=['self'])**

**# Get all active EC2 instance IDs**

**instances\_response = ec2.describe\_instances(Filters=[{'Name': 'instance-state-name', 'Values': ['running']}])**

**active\_instance\_ids = set()**

**for reservation in instances\_response['Reservations']:**

**for instance in reservation['Instances']:**

**active\_instance\_ids.add(instance['InstanceId'])**

**# Iterate through each snapshot and delete if it's not attached to any volume or the volume is not attached to a running instance**

**for snapshot in response['Snapshots']:**

**snapshot\_id = snapshot['SnapshotId']**

**volume\_id = snapshot.get('VolumeId')**

**if not volume\_id:**

**# Delete the snapshot if it's not attached to any volume**

**ec2.delete\_snapshot(SnapshotId=snapshot\_id)**

**print(f"Deleted EBS snapshot {snapshot\_id} as it was not attached to any volume.")**

**else:**

**# Check if the volume still exists**

**try:**

**volume\_response = ec2.describe\_volumes(VolumeIds=[volume\_id])**

**if not volume\_response['Volumes'][0]['Attachments']:**

**ec2.delete\_snapshot(SnapshotId=snapshot\_id)**

**print(f"Deleted EBS snapshot {snapshot\_id} as it was taken from a volume not attached to any running instance.")**

**except ec2.exceptions.ClientError as e:**

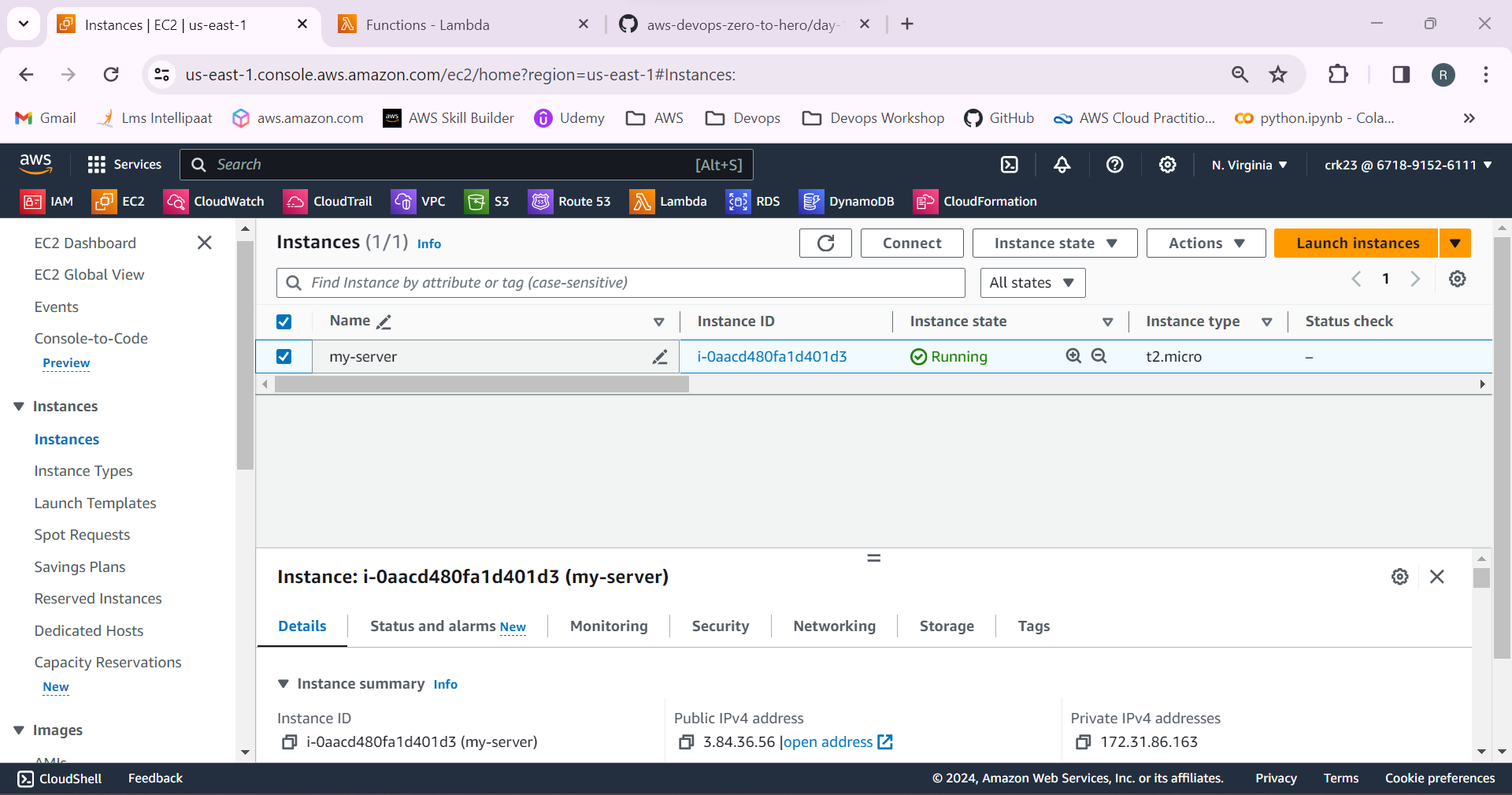
**if e.response['Error']['Code'] == 'InvalidVolume.NotFound':**

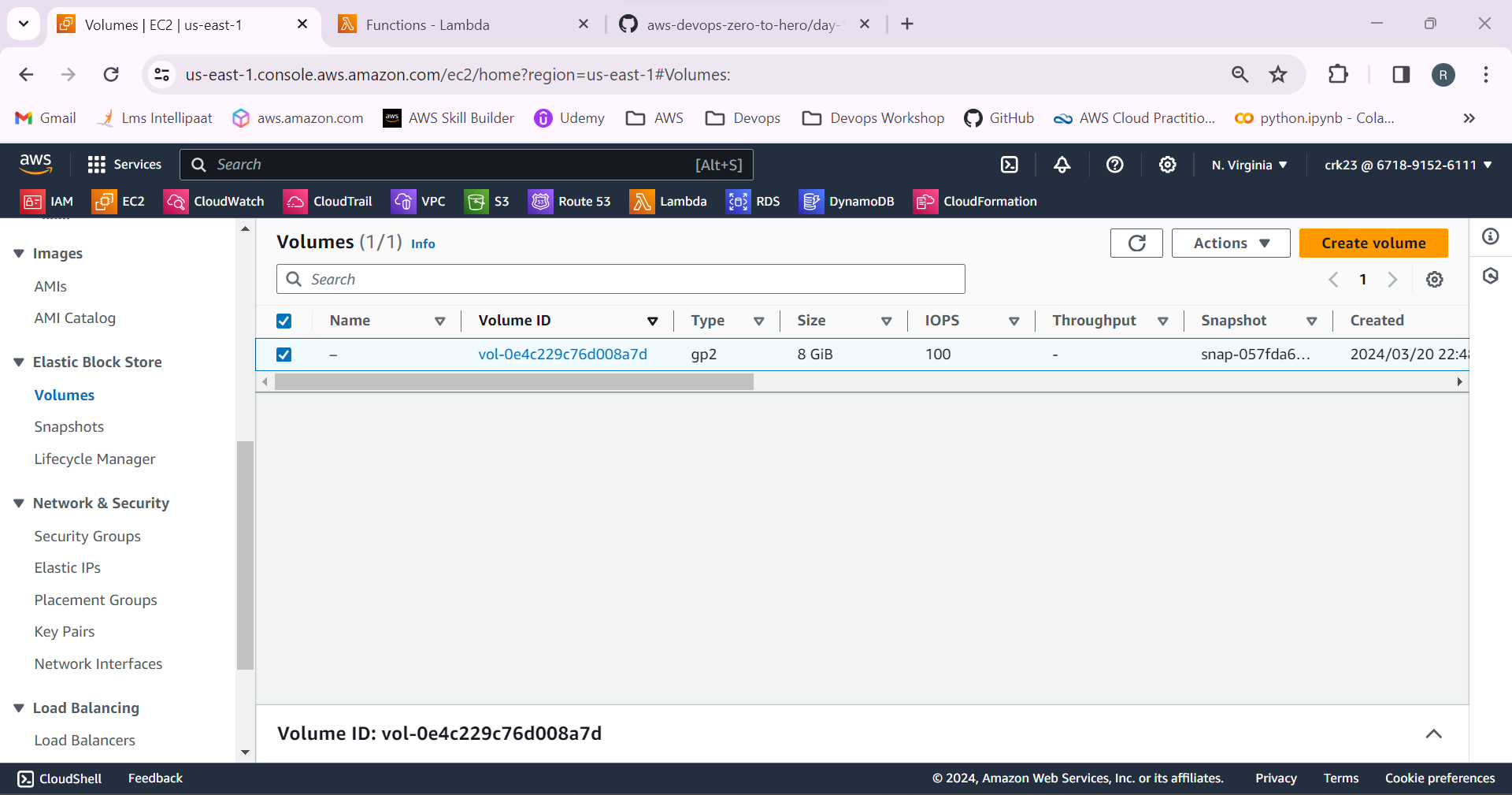
**# The volume associated with the snapshot is not found (it might have been deleted)**

**ec2.delete\_snapshot(SnapshotId=snapshot\_id)**

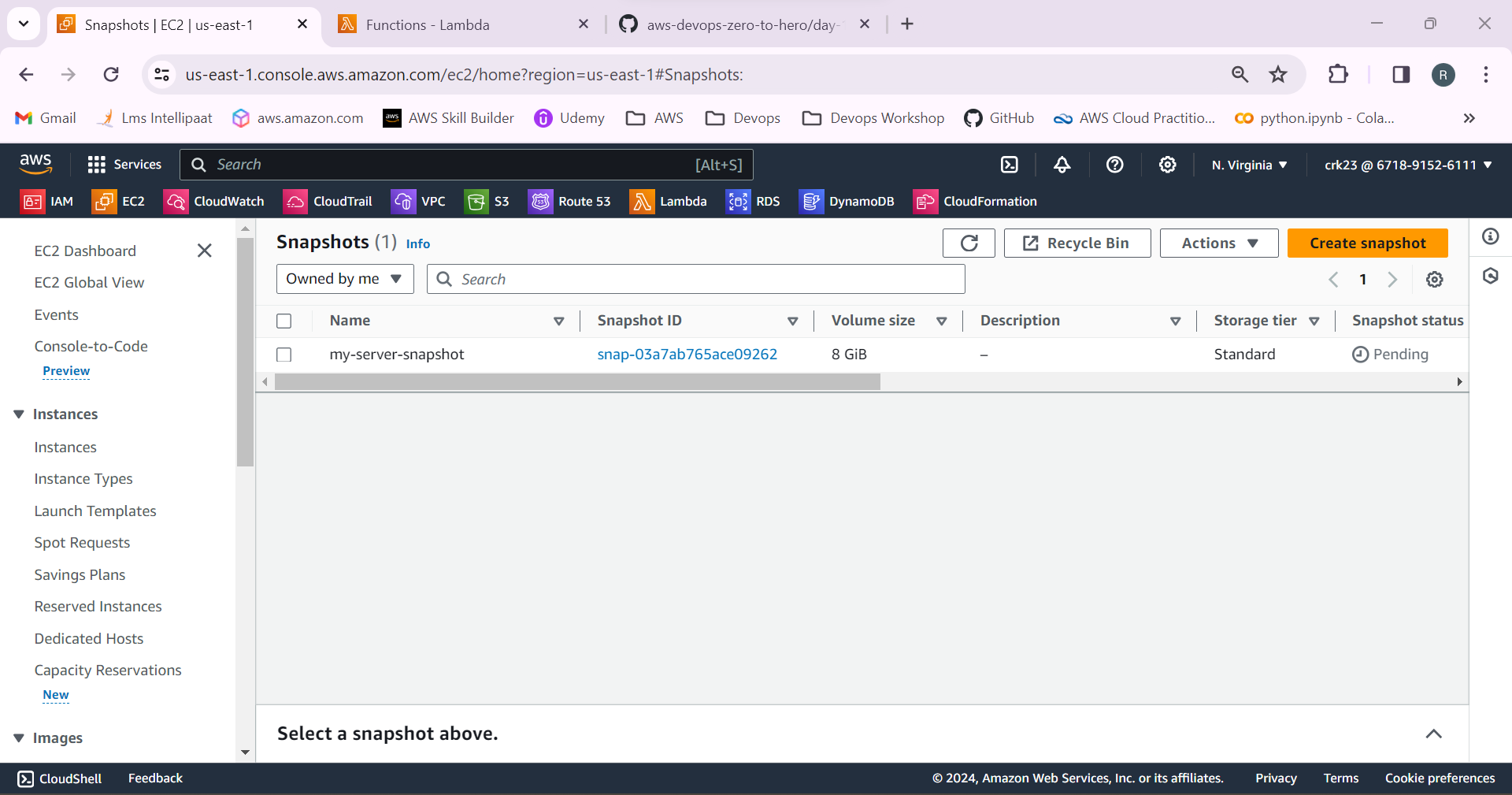
**print(f"Deleted EBS snapshot {snapshot\_id} as its associated volume was not found.")**

1.Create a EC2 instance

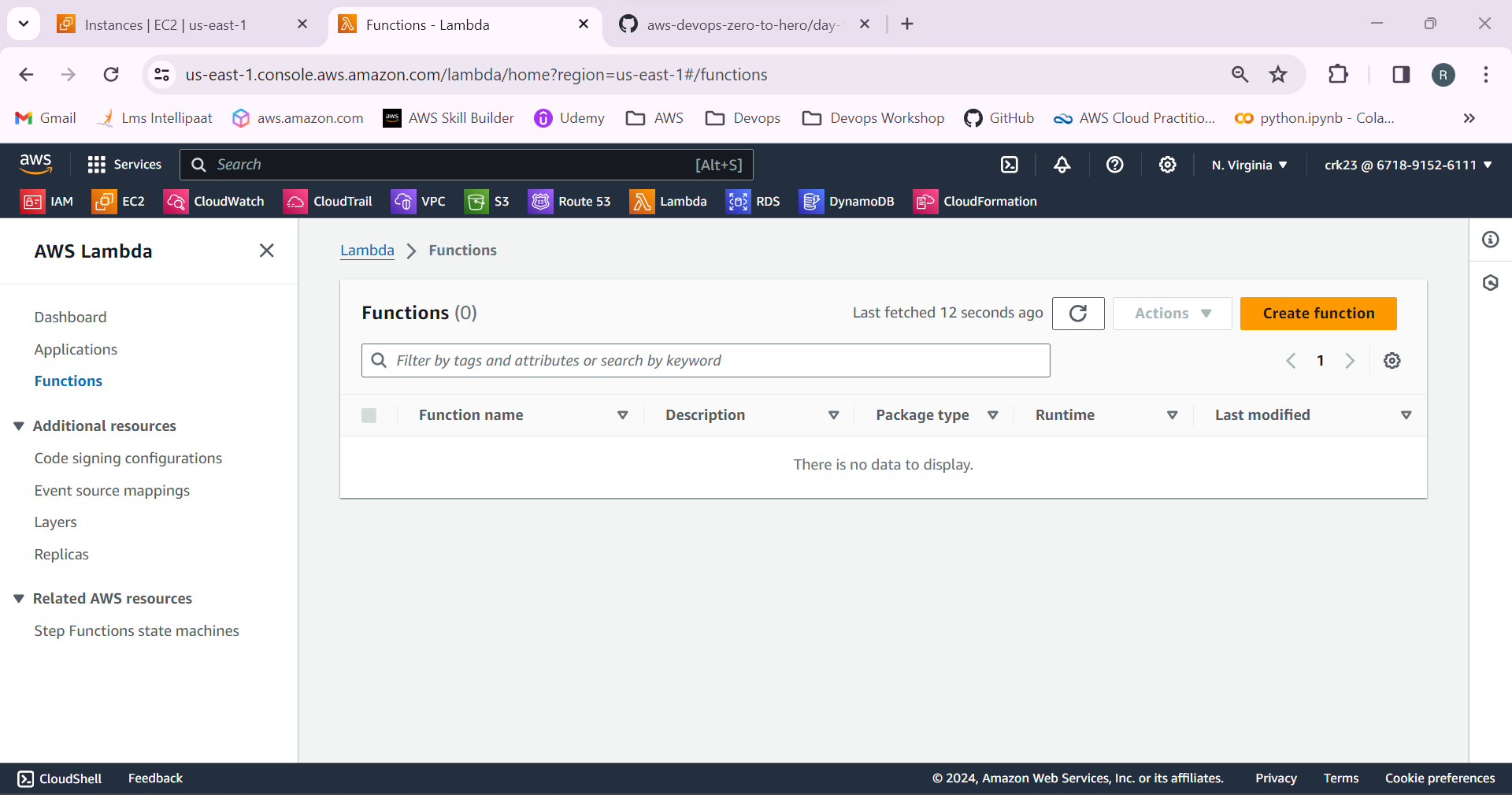




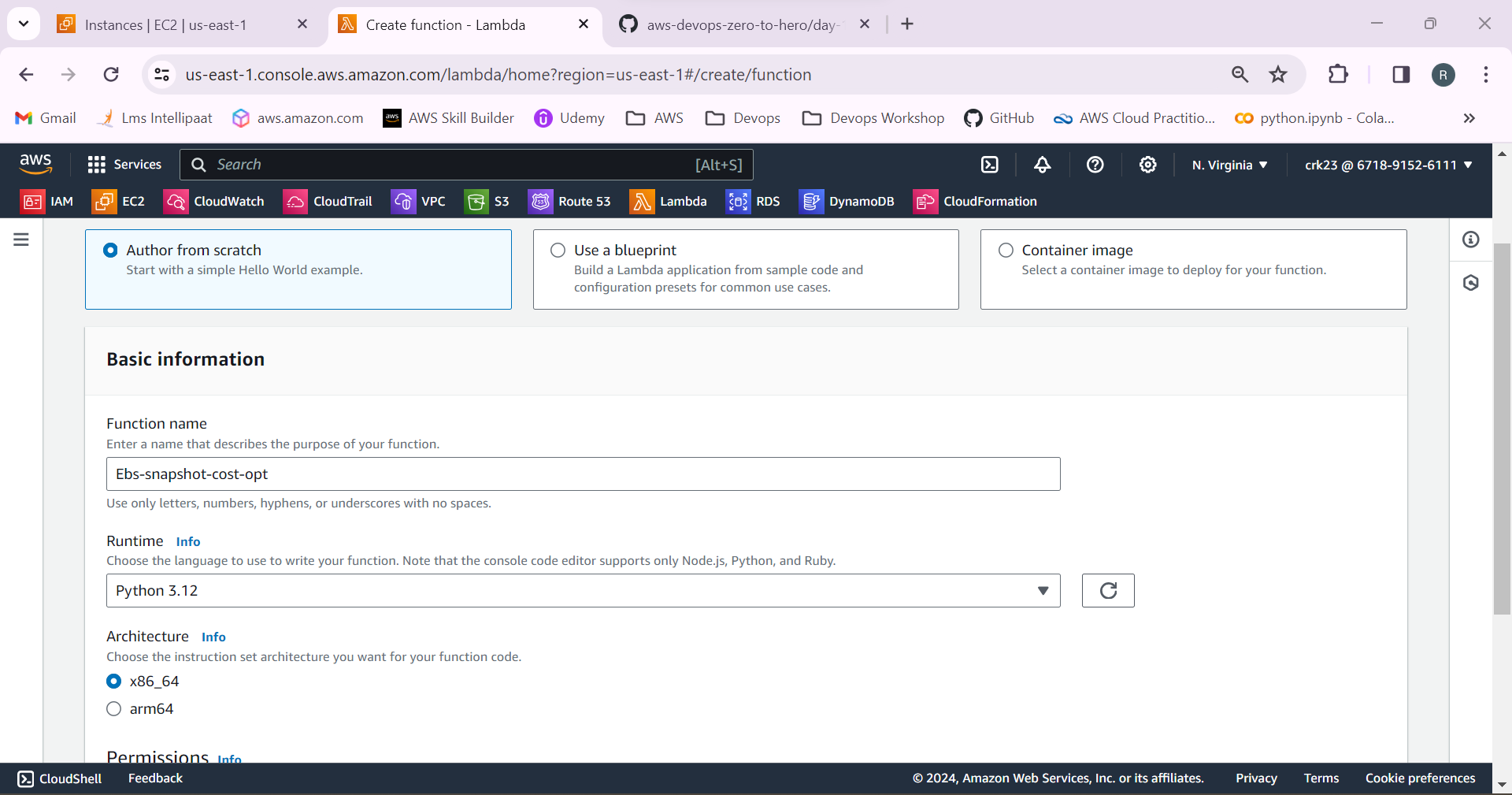
2.Create a snapshot from the EBS volume of the EC2 instance.



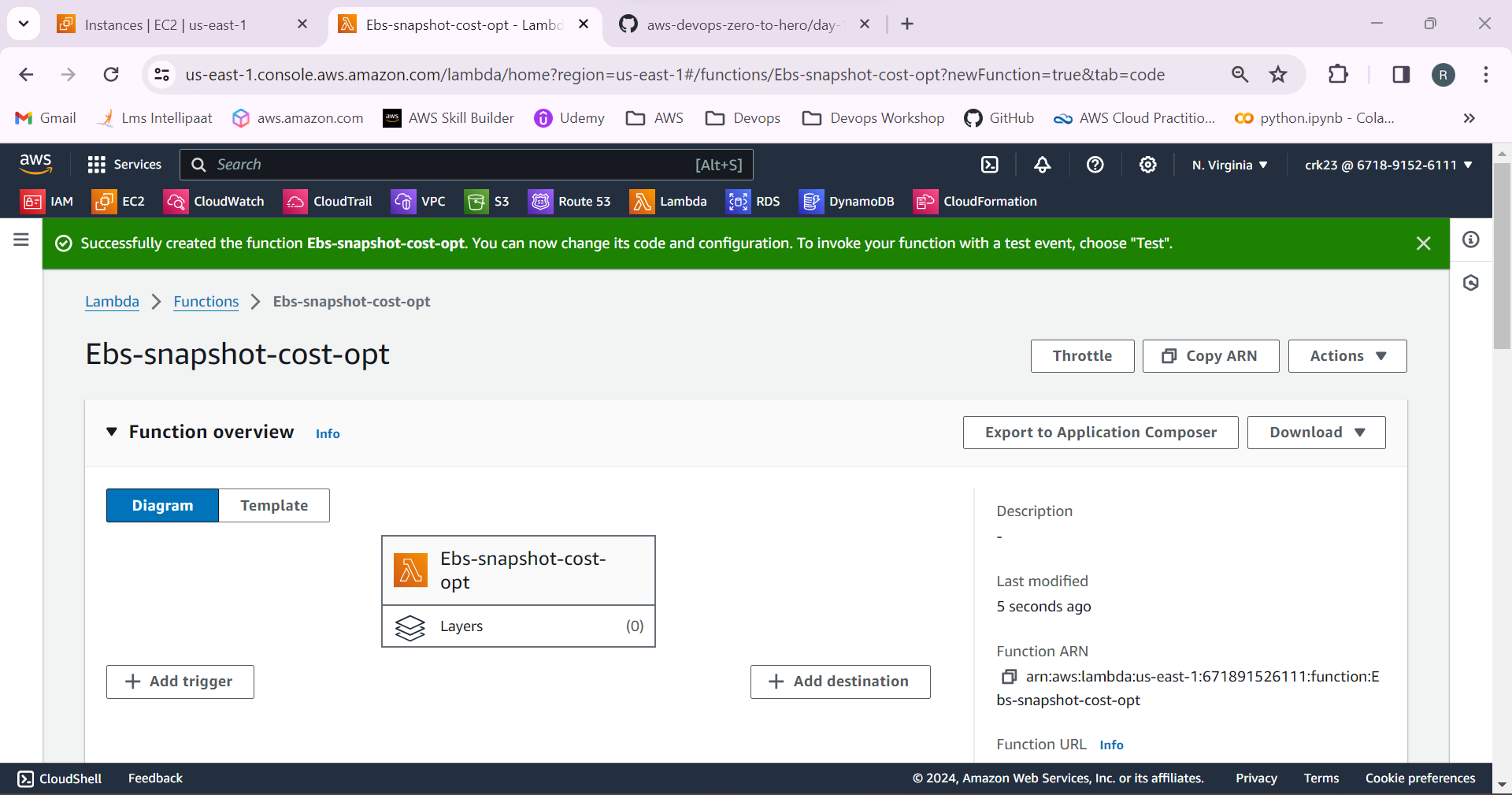
3.Select Lambda from AWS services and create a lambda function.



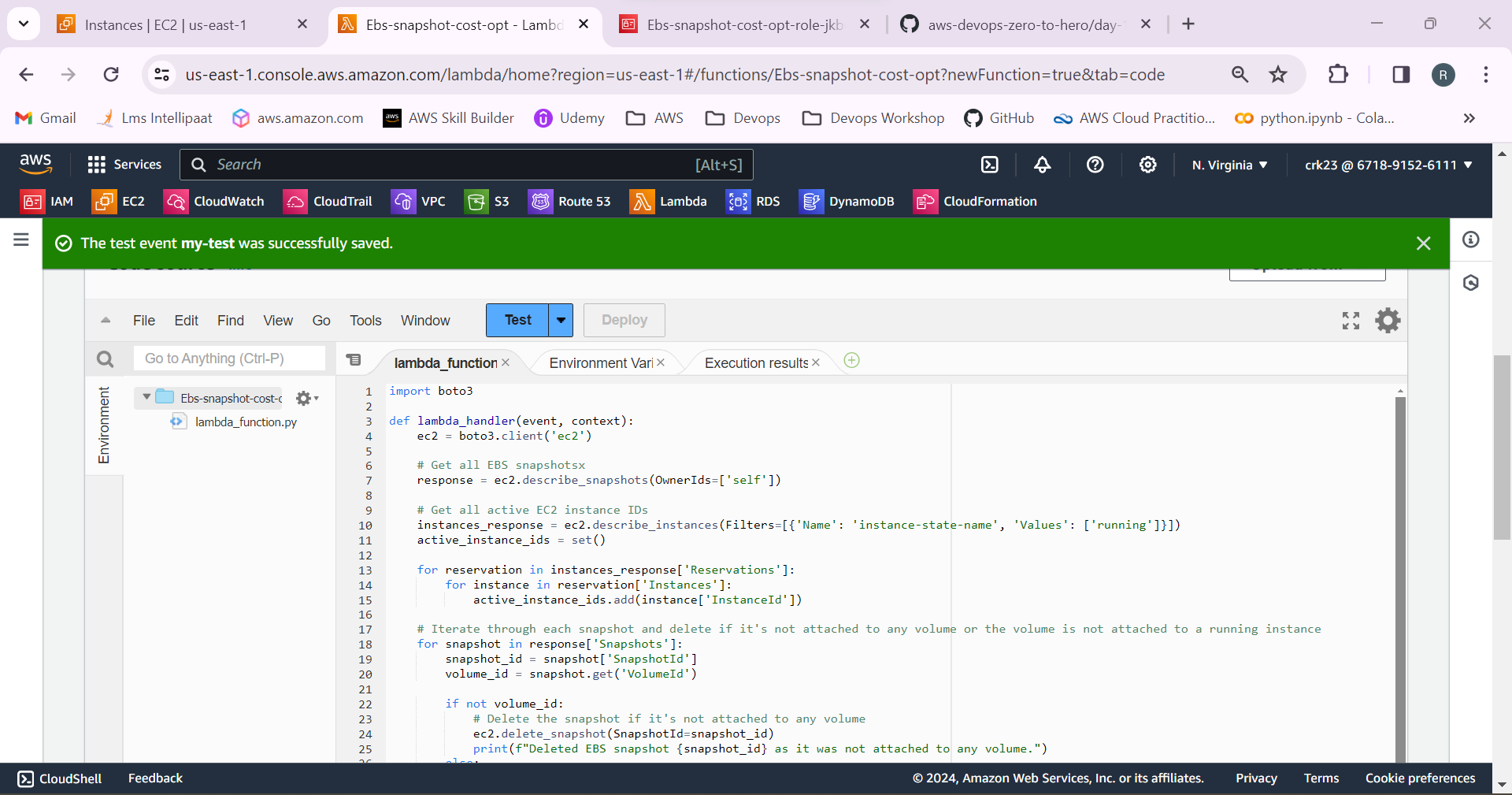
4.Select Runtime as python



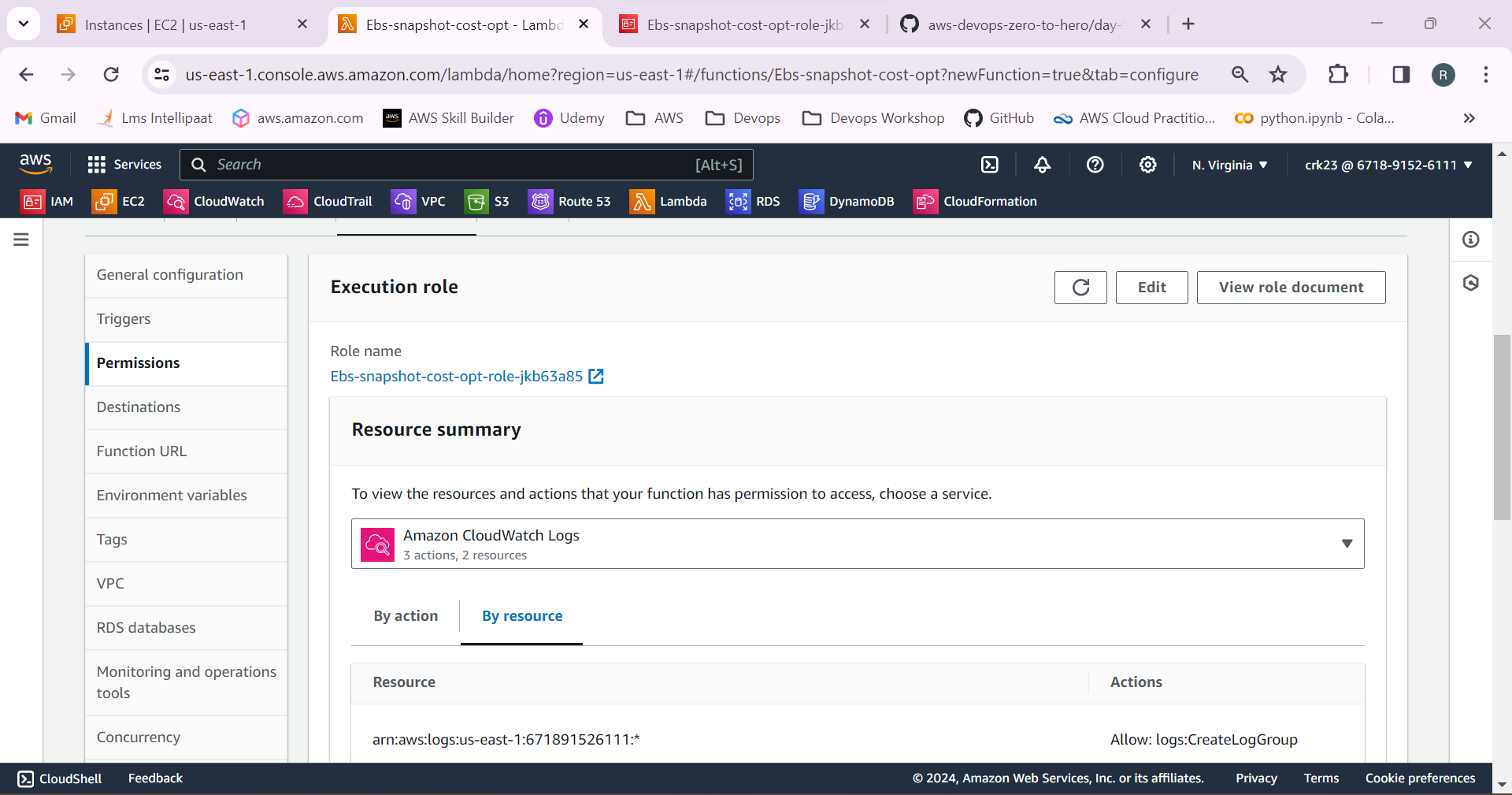
5.Lambda function is created successfully.

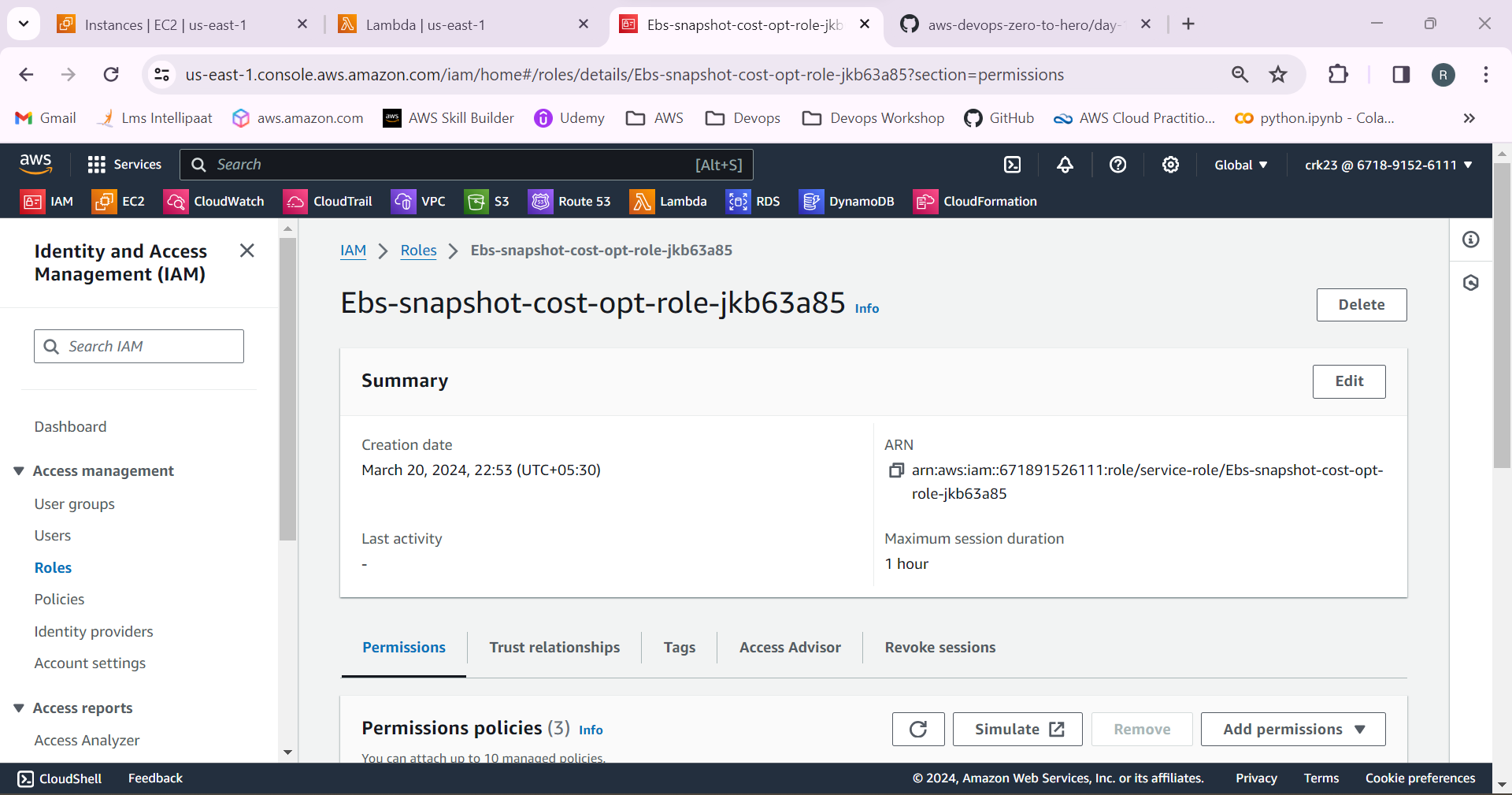


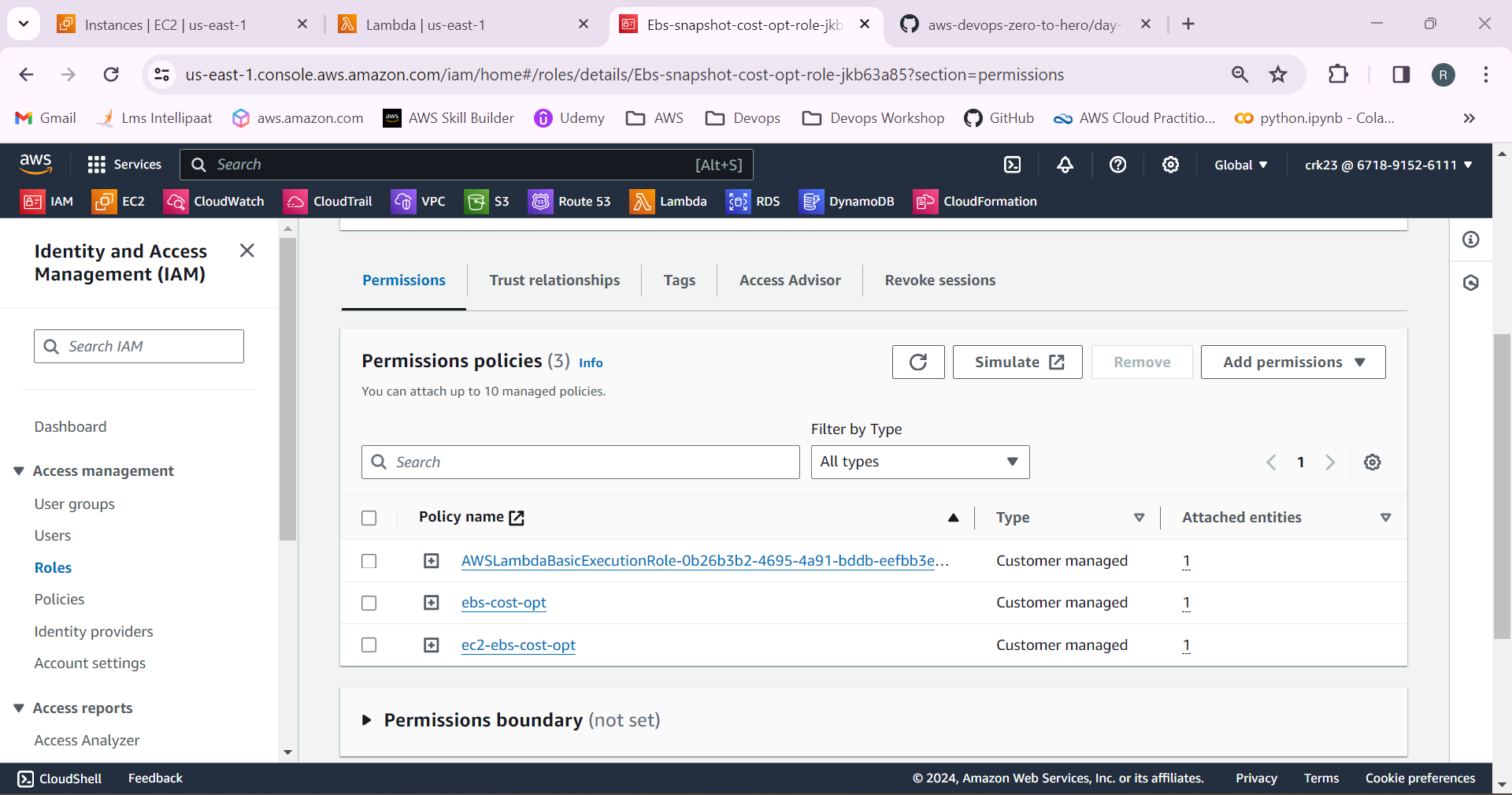
6.Now copy the code mentioned above and paste it in the lambda function. Deploy and test the function.



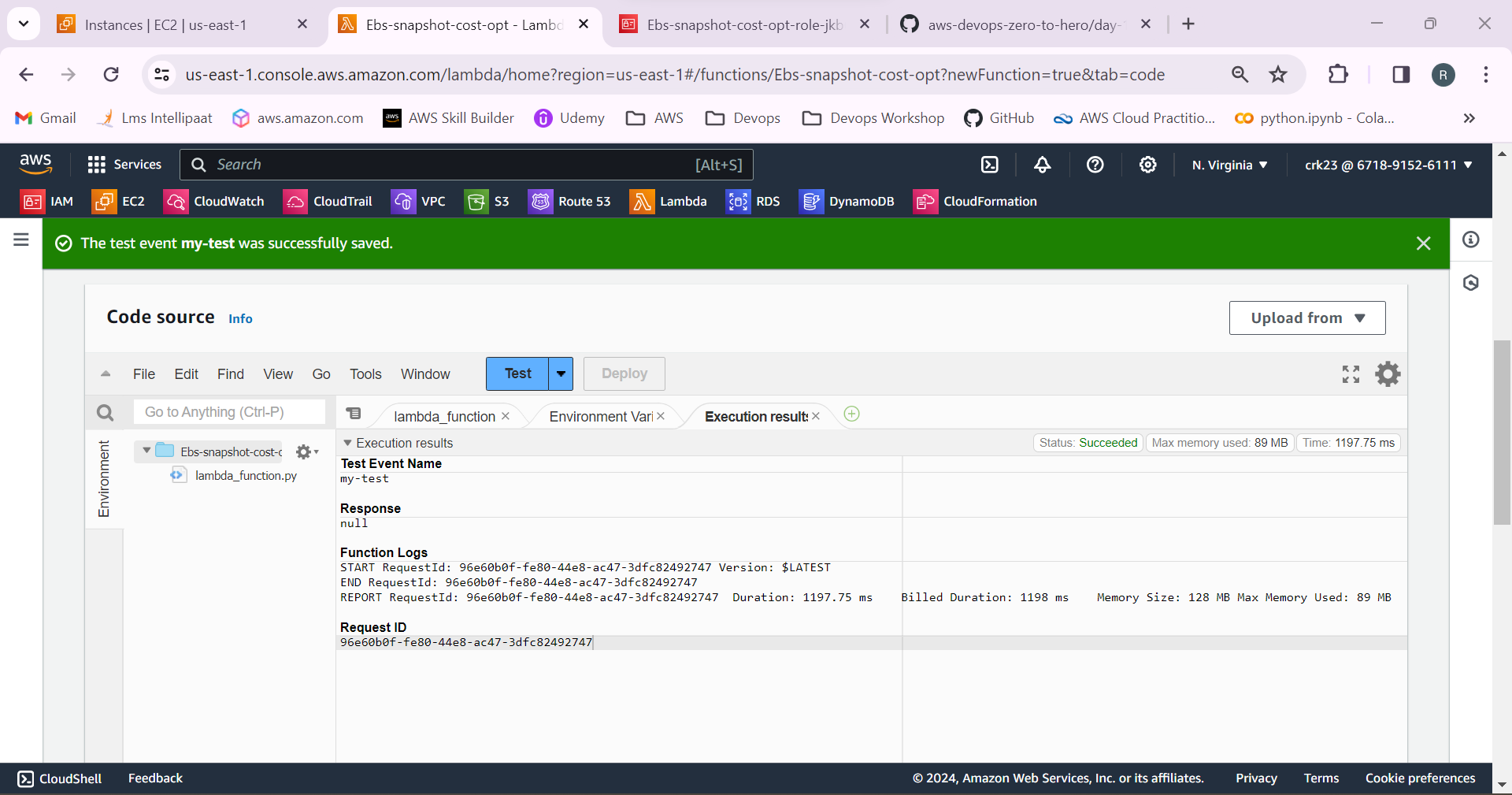
7.Select the IAM role and add the **describe snapshots, delete snapshots, describe volumes, describe instances**  permissions to the IAM role



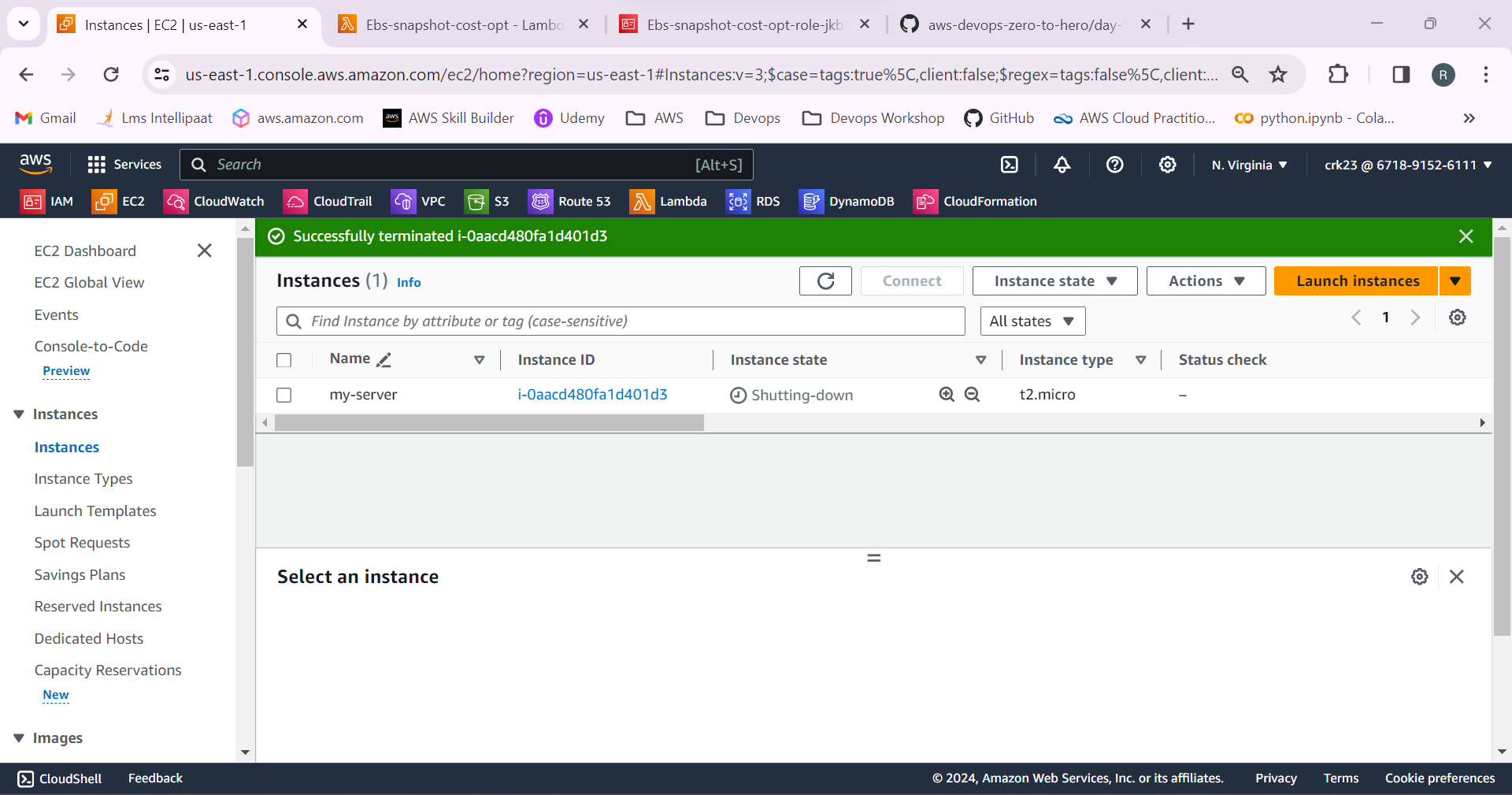




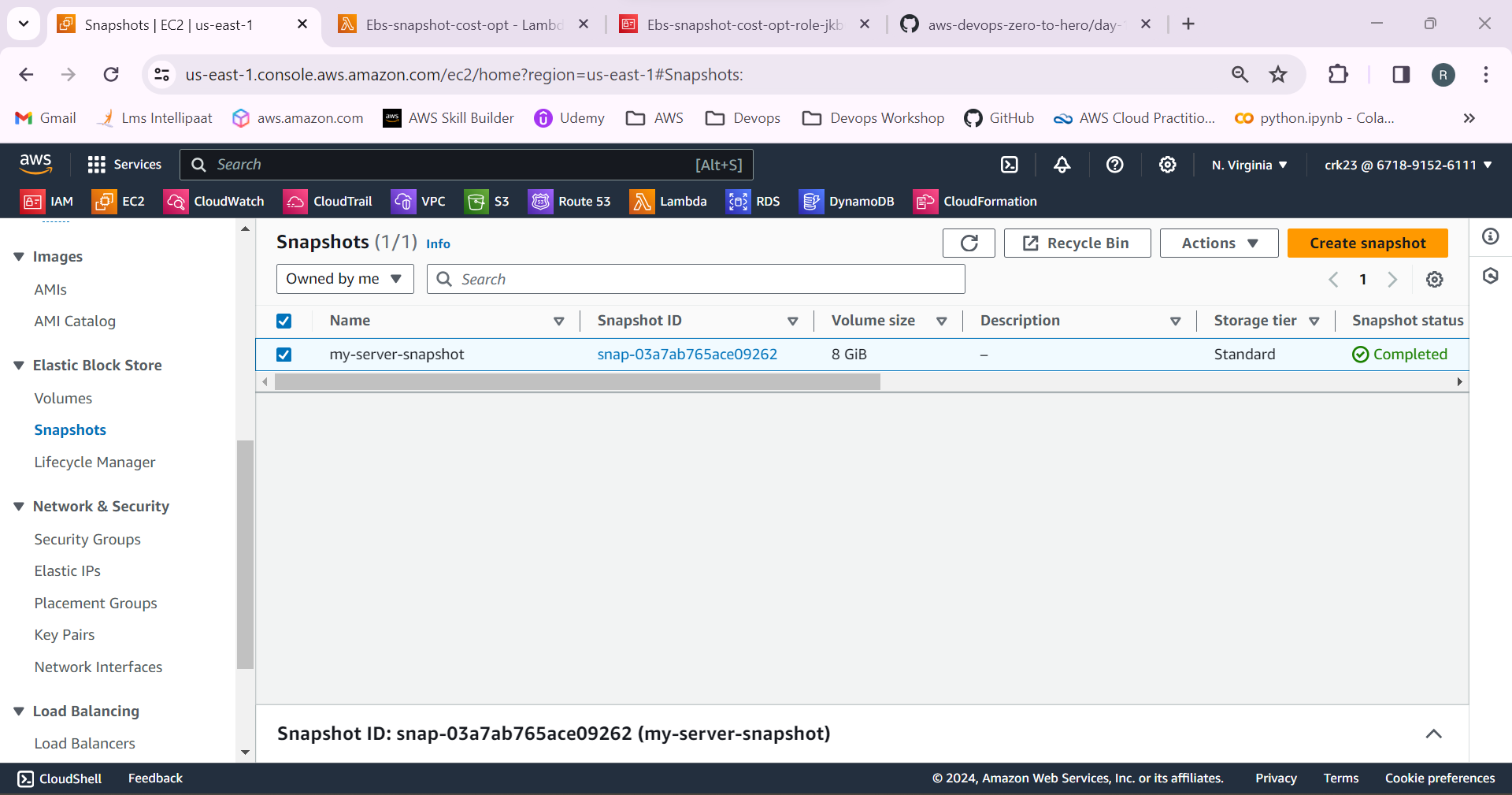
8.Now run the lambda function



9.Delete the EC2 instance.



10.But the snapshot is not deleted



11.Now test the lambda function. The snapshot will be deleted.

