CLOUD COMPUTING LABS SECTION II

Daniel Ge
Period 678

CISCO

CCNP LAB REPORT

Purpose

These labs continue with more exercises of the AWS concepts covered in later sections of the AWS Cloud Foundations course to train users in familiarity with using the AWS management system in sample corporate scenarios. By going through more mock scenarios with more AWS services, we expand our experience in implementing AWS Cloud solutions to tackle more advanced business goals.

Background Information

When an EC2 (Elastic Compute Cloud) instance is deleted, all the data associated with the instance would typically be deleted. To prevent this from happening, we may use Amazon EBS (Elastic Block Storage) by attaching the instance to Amazon Elastic Block Storage such that data is preserved even when the instance is unexpectedly terminated, such as by a power outage.

The advantage of AWS EBS over normal Object storage would be that block storage allows us to change a small part of a large file without updating the entire file through virtue of dividing the file into many blocks. This advantage significantly reduces the latency and cost if EBS, while improving throughput.

EBS Volumes also offer additional features for resiliency through automatic replication within the Availability Zones and optional backups to Amazon S3. Through snapshots, we may recreate a previous stable version of the volume in case the current version has an error that cannot be recovered.

EBS Volumes are divided into Solid State Drives (SSD) and Hard Disk Drives (HDD) then further subdivided into options such as General Purpose or Provisioned IOPS, each offering different Volume Sizes, Volume Maximums, and Throughputs. Selecting the correct EBS Volume for a given situation is key to saving on costs while achieving the desired result.

Though AWS' storage services like AWS S3 (Simple Storage System) or EBS (Elastic Block Storage) allow us transfer large amounts of unstructured data with high throughput and low latency, organizations may occasionally need to achieve more complex tasks on their data such as filtering or data analytics. For this purpose, AWS provides various Database services, including AWS RDS (Relational Database Service)

AWS RDS allows users to create multiple databases in each DB instance. These instances can be deployed across multiple AZs (availability zones) and are best fit for applications without too much information stored but frequently accessed. RDS offers advantages including high throughput, low-cost, data security, and automatic scaling.

AWS RDS DBs are best used for high durability complex querying with high query and write rates as opposed to use cases with massive read rates or GET requests done without SQL.

After setting up the RDS DB, instances are charged either on-demand or reserved for a multi-year term. The first backup to the storage can be made without additional charge.

Given the fluctuation of traffic for web organizations throughout a given day or year, it is often inefficient to upkeep many EC2 instances, RDS databases, or other applications to account for the maximum spike in traffic usage every year. Instead, AWS allows us to automatically scale many of our services used according to demand.

In our lab, we will be large interacting with EC2 Auto Scaling. Using AWS CloudWatch, we may set alarms to detect when our AWS traffic gets too high or too low. When these alarms are triggered, we may then configure for EC2 Auto Scaling to automatically increase or decrease the number of EC2 instances run.

To distribute traffic across new and old instances in a balanced manner, we use Elastic Load Balancing. During periods of high traffic volume, load balancing automatically assigns each user traffic to some instance such that, overall, each instance is used at a similar level. Load balancing can occur at the Application level (OSI Layer 7) for HTTP/HTTPS traffic, at the network level (OSI Layer 4) for TCP/UDP/TLS traffic, or the classic load balancer to balance traffic across EC2 instances.

Lab Summary

In Lab 4, we practice using Amazon EBS (Elastic Block Storage). We begin by creating a new EBS Volume, then attach the volume to an EC2 instance. After the connection is established, we will then use EBS Snapshots to preserve the current file state. To test EBS Snapshots, we will make changes on our EBS Volume by deleting a file, then use snapshots to restore it.

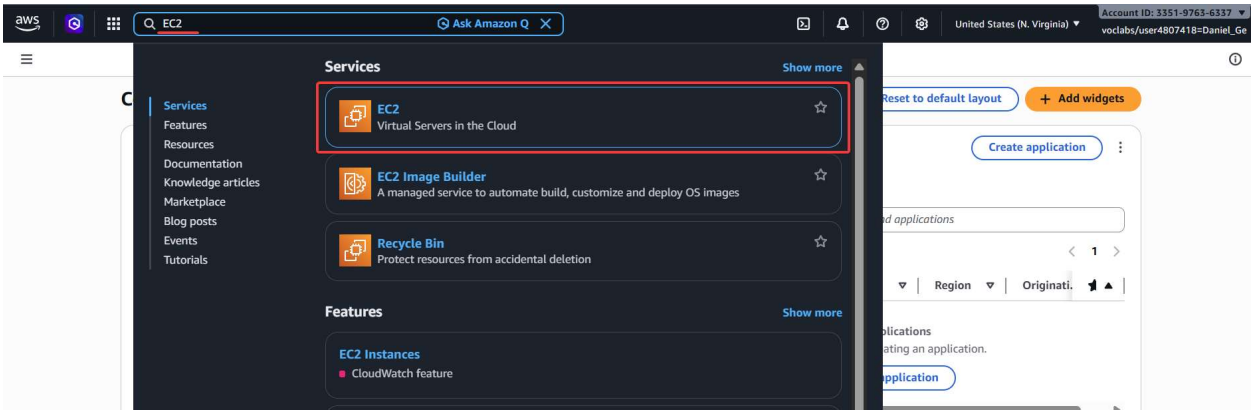
In Lab 5, we practice creating and using an Amazon RDS (Relational Database System) DB instance. We begin by configuring the appropriate security groups and subnet group for the instance, then create the instance itself through the AWS Management Console. We may then test the RDS DB instance by interacting with the database through a browser window.

In Lab 6, we practice with creating load balancers and auto scaling groups. The lab begins with creating a load balancer for the incoming web traffic. Afterwards, we create an Auto-Scaling group with a desired capacity of 2 but max capacity of 6 set to scale at Average CPU Utilization greater than 60%. We test the load balancer by spamming the EC2 instance with traffic, and observing the new instances being added.

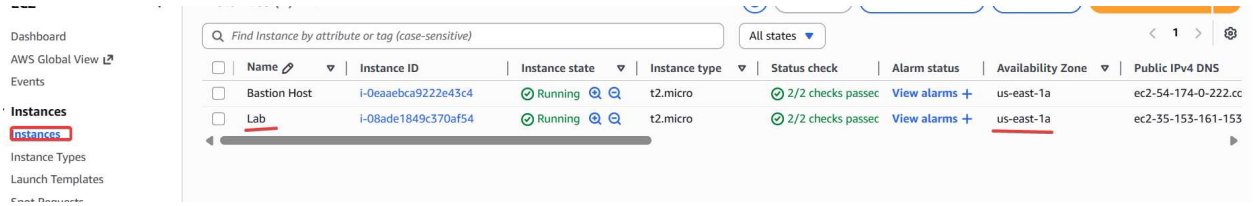
Lab Procedure

Working with EBS (Lab 4)

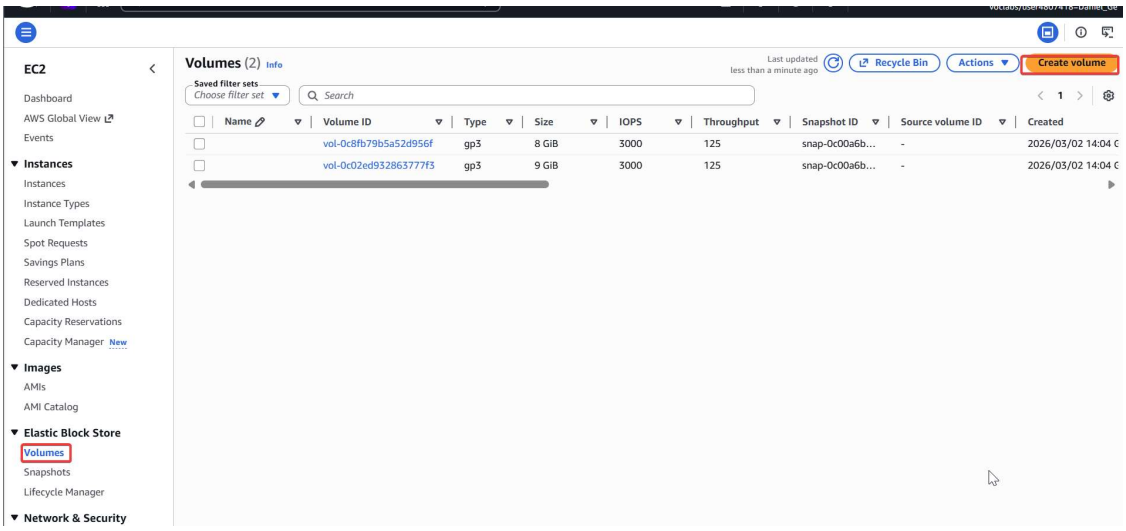
To begin creating a new EBS Volume, using the search bar, open the EC2 menu



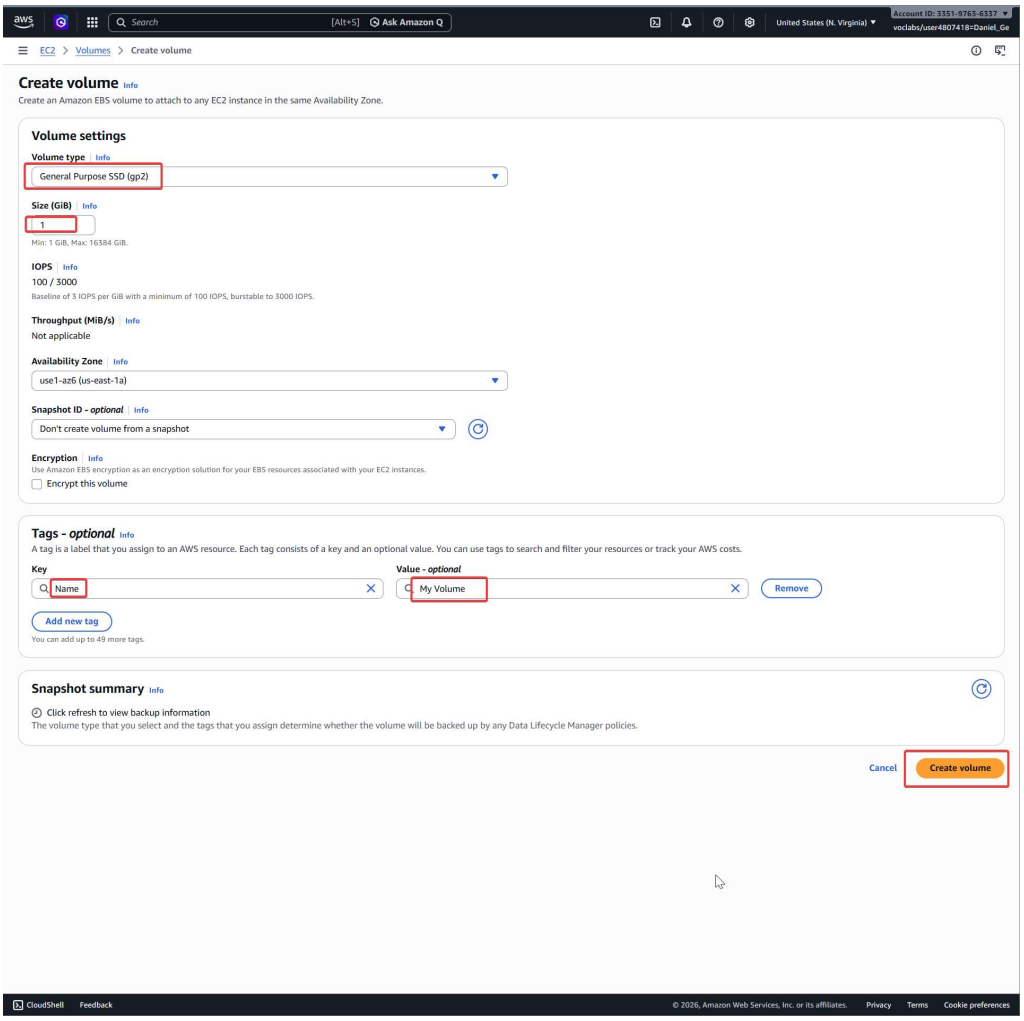
Click the instances tab and verify that there is an instance Lab already started running in us-east-1a



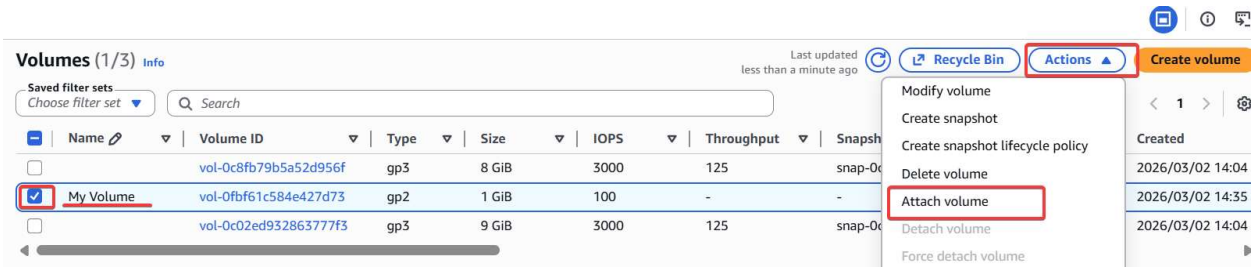
Click the Volumes tab on the left menu, then click “Create Volume”



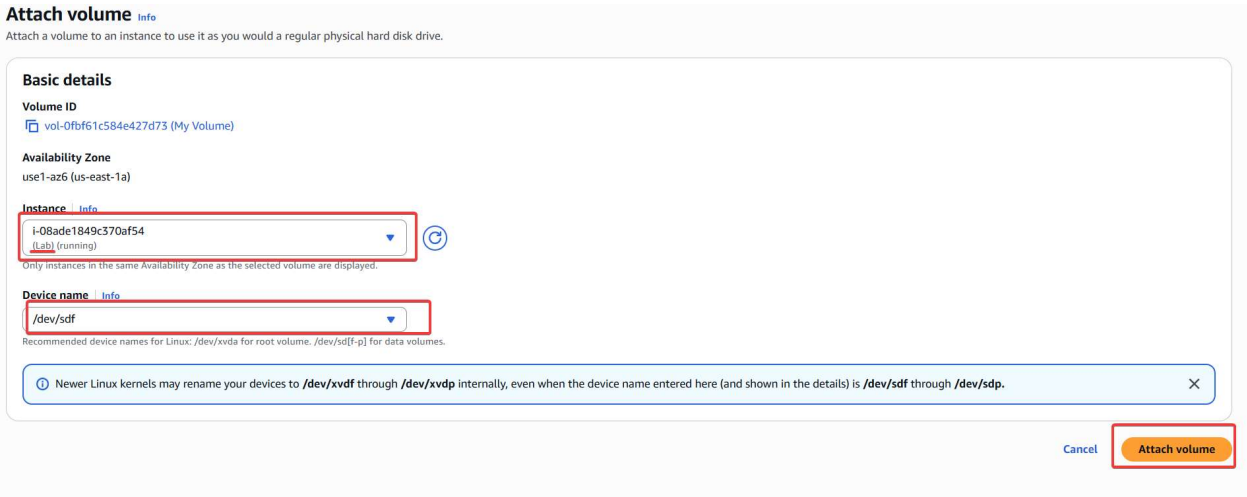
Confirm the Volume type is set to gp2, set the size to 1 GiB, and add a Tag with key “Name” and Value “My Volume”



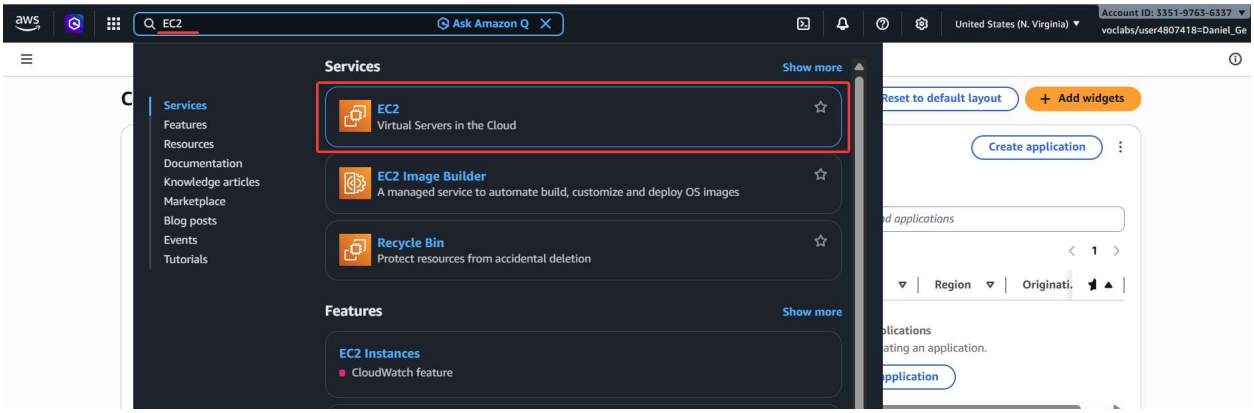
Select the checkbox next to the Volume with name “My Volume”, then click the Actions drop bar and click Attach Volume



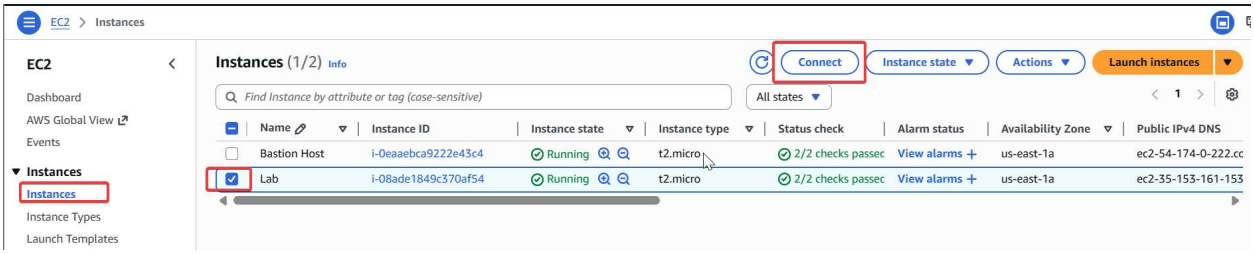
Configure the instance to the one labeled “Lab” and device name to “/dev/sdf”, then click Attach Volume



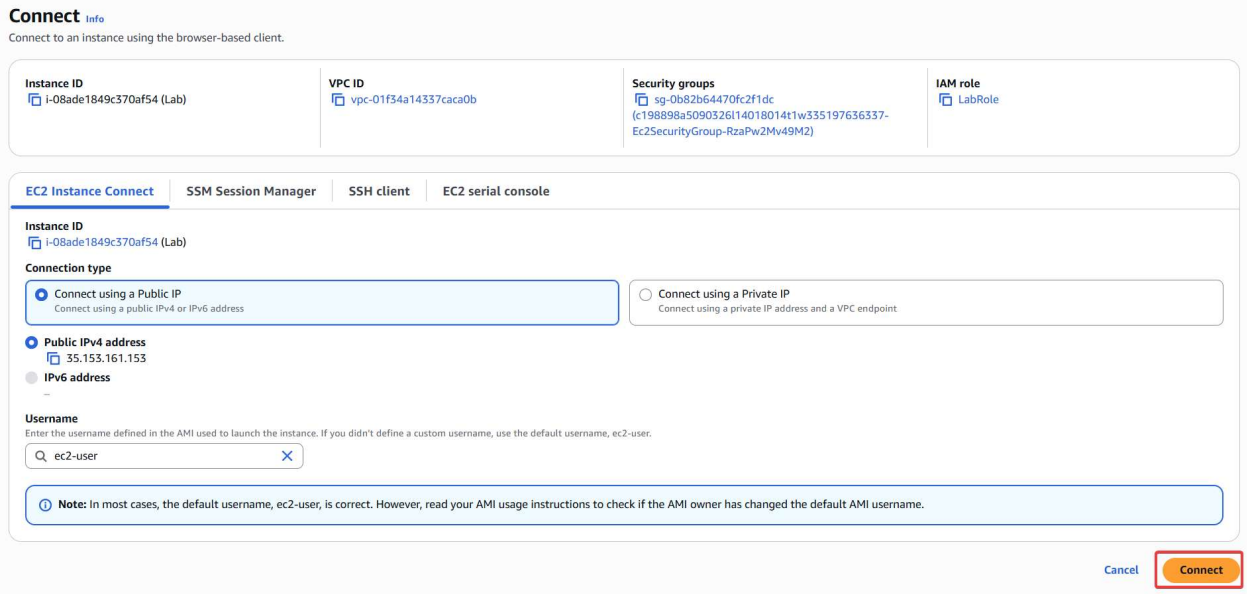
Using the search bar, open the EC2 menu



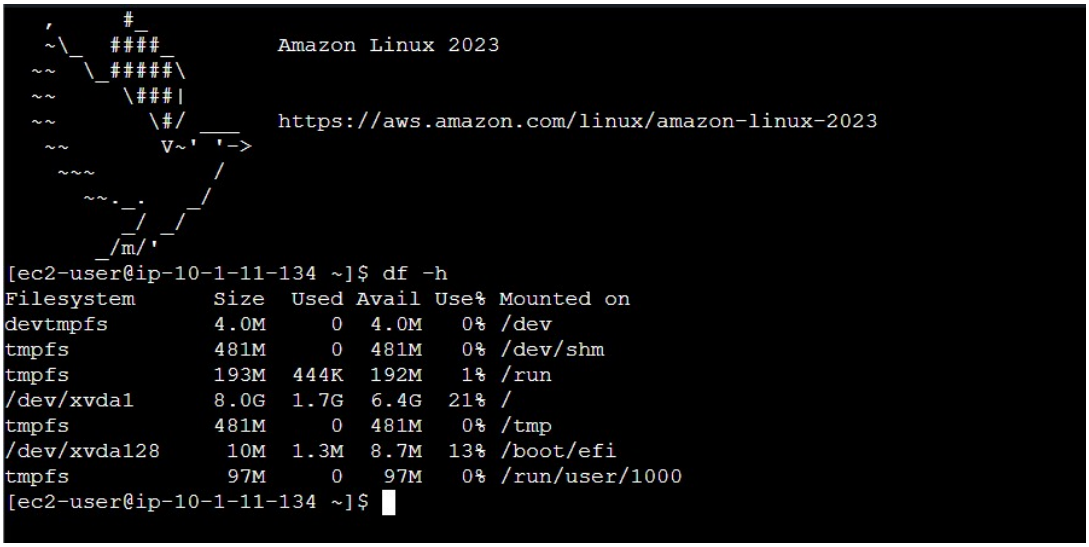
Open “Instances” from the left menu, then click the checkbox next to “lab”. Click “Connect” on the top



Keep the default settings and click Connect



After the terminal opens, type in “df -h” and click enter



This outputs the disk volume from /dev/xvda1 mounted. We will mount a new volume, create an ext3 file system using the command,

```
sudo mkfs -t ext3 /dev/sdf
```

```
[ec2-user@ip-10-1-11-134 ~]$ sudo mkfs -t ext3 /dev/sdf
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 262144 4k blocks and 65536 inodes
Filesystem UUID: ba09c403-fe6d-4184-8297-baaa9c8287c4
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

Now create a directory for mounting the new volume using the command,

```
sudo mkdir /mnt/data-store
```

```
[ec2-user@ip-10-1-11-134 ~]$ sudo mkdir /mnt/data-store
```

Mount the new volume to the EC2 instance using the command,

```
sudo mount /dev/sdf /mnt/data-store
```

```
[ec2-user@ip-10-1-11-134 ~]$ sudo mount /dev/sdf /mnt/data-store
```

To ensure the instance mounts this volume upon startup, add the following command,

```
echo "/dev/sdf   /mnt/data-store ext3 defaults,noatime 1 2" | sudo tee
-a /etc/fstab
```

```
[ec2-user@ip-10-1-11-134 ~]$ echo "/dev/sdf   /mnt/data-store ext3 defaults,noatime 1 2" | sudo tee -a /etc/fstab
/dev/sdf   /mnt/data-store ext3 defaults,noatime 1 2
```

View the configuration file using the command,

```
cat /etc/fstab
```

```
[ec2-user@ip-10-1-11-134 ~]$ cat /etc/fstab
#
UUID=96e83033-9c05-4912-8900-93256ded3e51     /           xfs    defaults,noatime  1   1
UUID=C42B-0705        /boot/efi       vfat    defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2
/dev/sdf   /mnt/data-store ext3 defaults,noatime 1 2
```

Verify the changes by typing in “df -h” and click enter. Observe the new /dev/xvdf mounted onto the data store

```
[ec2-user@ip-10-1-11-134 ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        4.0M     0  4.0M   0% /dev
tmpfs           481M     0  481M   0% /dev/shm
tmpfs           193M  440K  192M   1% /run
/dev/xvda1      8.0G  1.7G  6.4G  21% /
tmpfs           481M     0  481M   0% /tmp
/dev/xvda128     10M  1.3M  8.7M  13% /boot/efi
tmpfs            97M     0   97M   0% /run/user/1000
/dev/xvdf       975M   60K  924M   1% /mnt/data-store
```

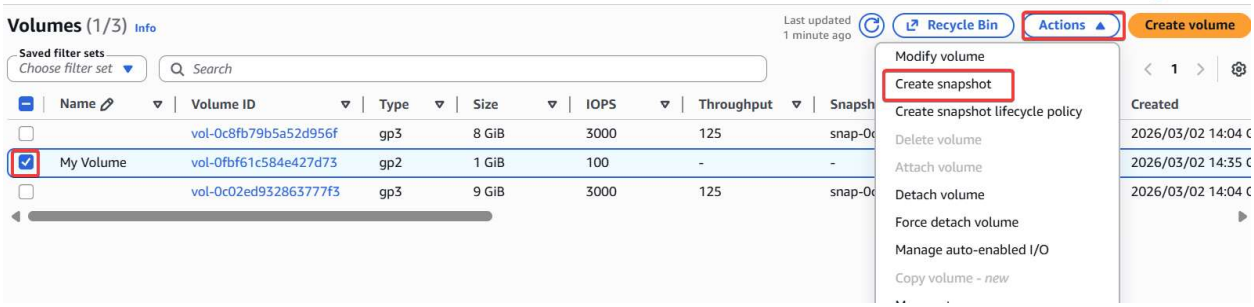
To test, create file.txt on the data-store folder and write a text on there using the command

```
sudo sh -c "echo some text has been written > /mnt/data-store/file.txt"
```

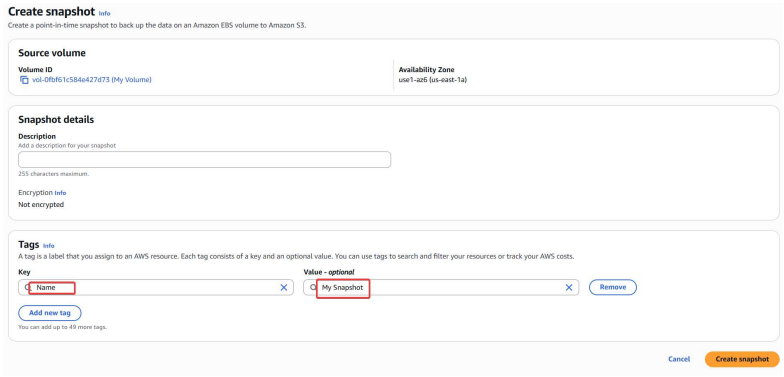
Then verify the text has been added

```
[ec2-user@ip-10-1-11-134 ~]$ sudo sh -c "echo some text has been written > /mnt/data-store/file.txt"
[ec2-user@ip-10-1-11-134 ~]$ cat /mnt/data-store/file.txt
some text has been written
```

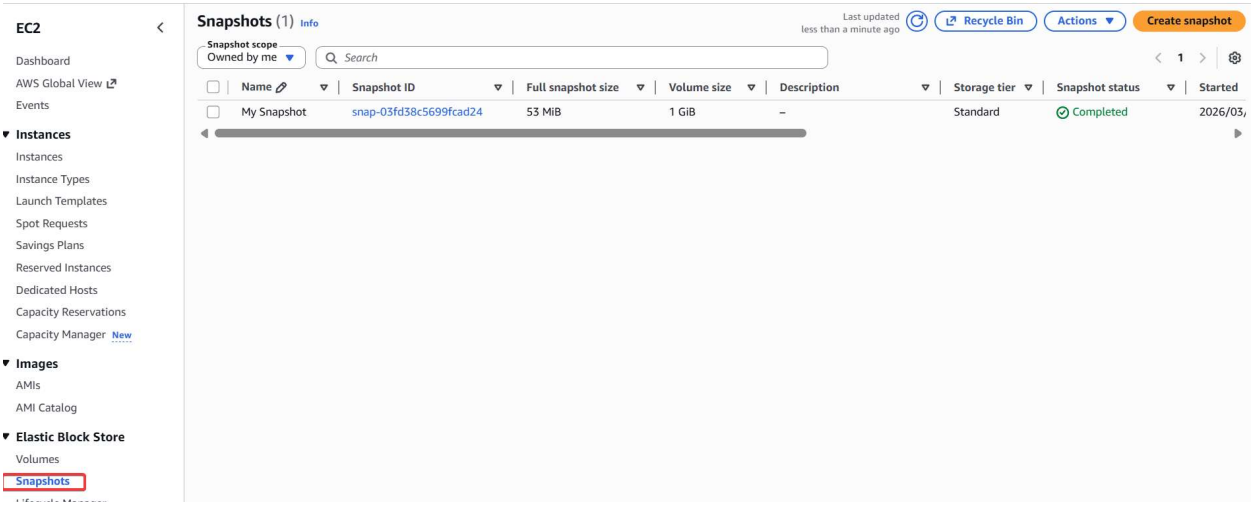
Return to the Volumes tab on EC2 console. Select “My Volumes” then “Create snapshot” under the “Actions” menu



Add a tag with Key “Name” and Value “My Snapshot”, then click Create Snapshot



Open Snapshots using the left menu and verify that a new snapshot has been created



Return to EC2 Connect Console and delete the file.txt file using the command,

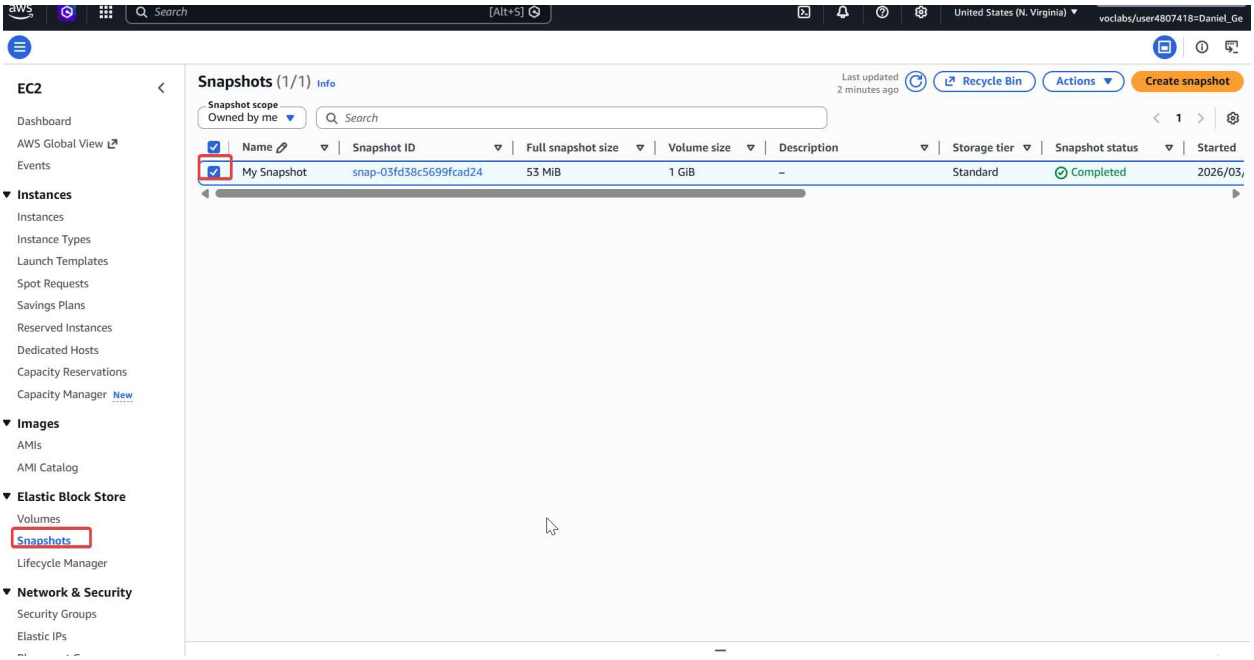
```
sudo rm /mnt/data-store/file.txt
```

Verify the deletion using the command,

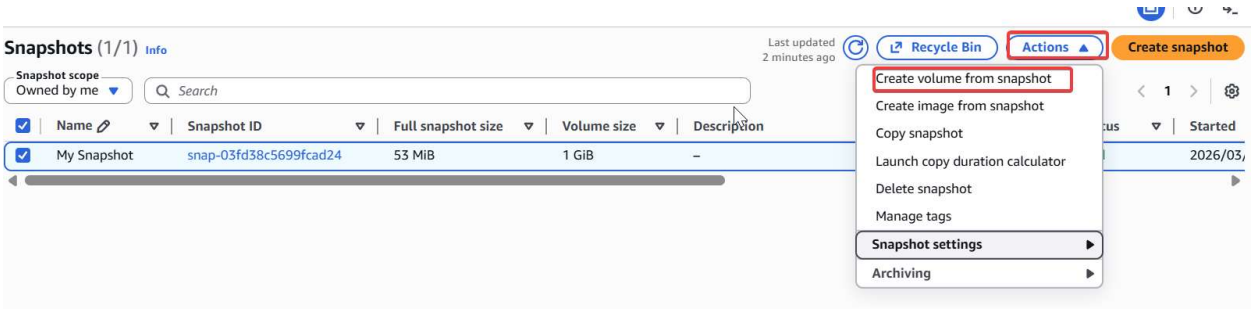
```
ls /mnt/data-store/
```

```
[ec2-user@ip-10-1-11-134 ~]$ sudo rm /mnt/data-store/file.txt
[ec2-user@ip-10-1-11-134 ~]$ ls /mnt/data-store/
lost+found
```

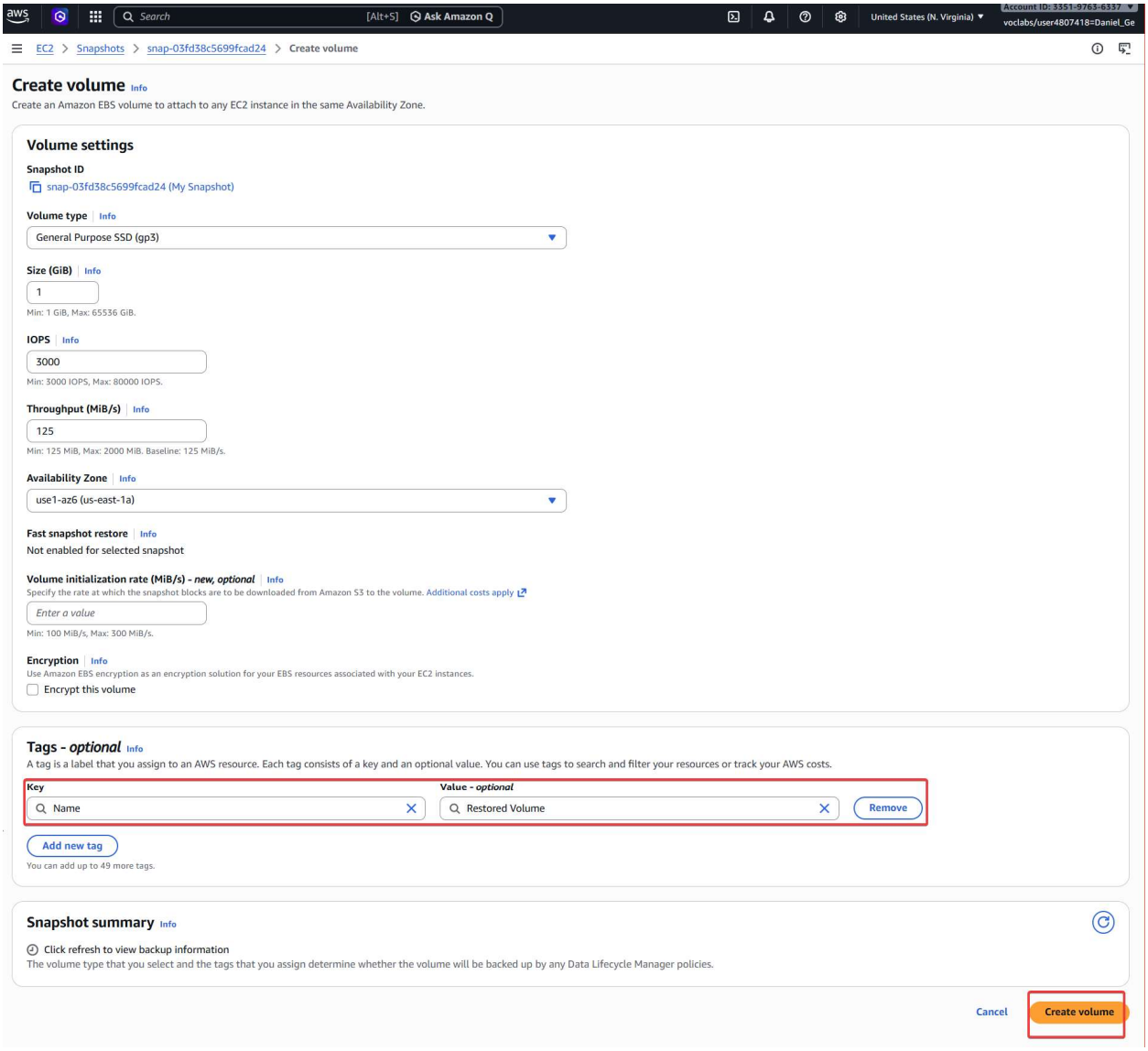
To restore this file using the snapshot, return to the Snapshots tab and click “My Snapshot”



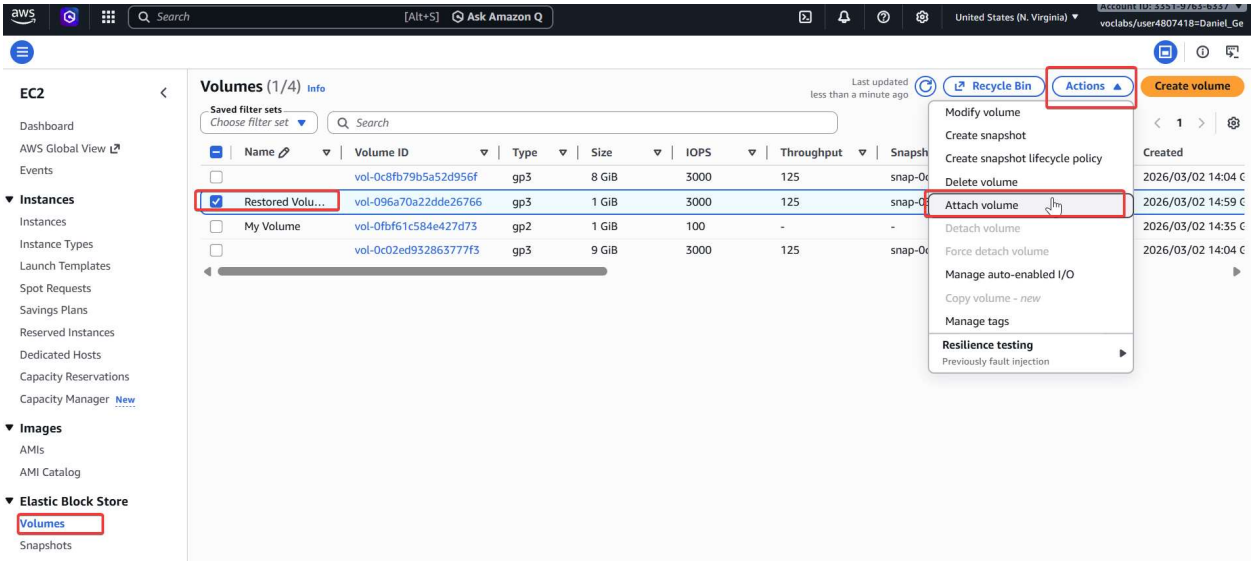
Under the Actions drop down, click “Create volume from snapshot”



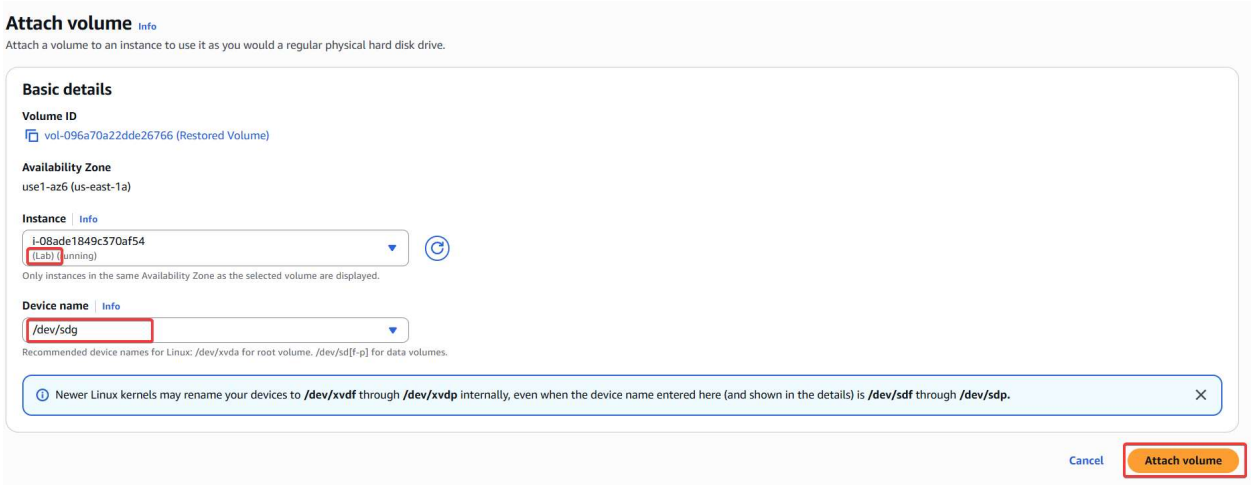
Configure the Tag as follows, then “Create volume”



Navigate to the Volumes tab, click checkbox next to the Restored Volume, and click “Attach Volume” under the Actions drop down



Configure the Instance to lab and Device name to “/dev/sdg”, then click “Attach volume”



Mount the backup using the console through the commands,

```
sudo mkdir /mnt/data-store2
```

```
sudo mount /dev/sdg /mnt/data-store2
```

Confirm the volume has been mounted using,

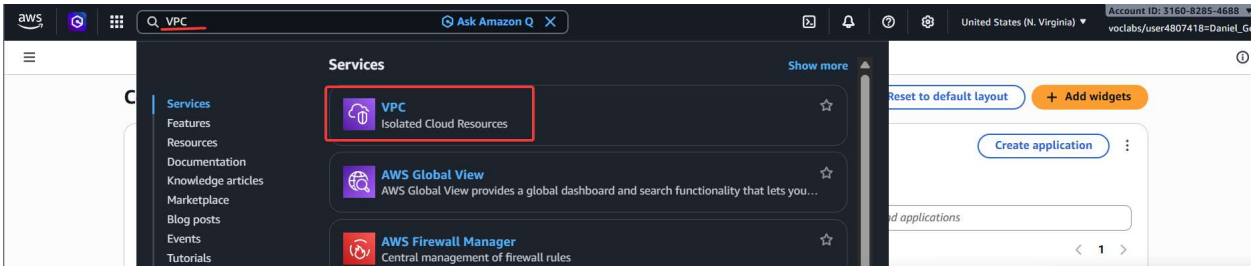
```
sudo mount /dev/sdg /mnt/data-store2
```

file.txt has been restored.

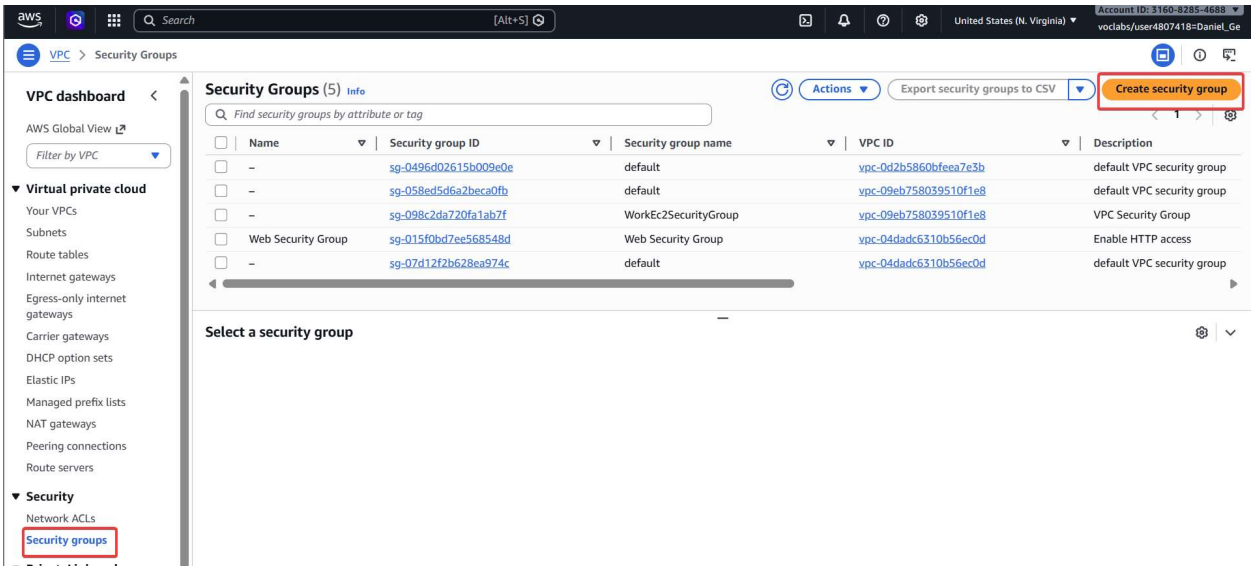
```
[ec2-user@ip-10-1-11-134 ~]$ sudo mkdir /mnt/data-store2
[ec2-user@ip-10-1-11-134 ~]$ sudo mount /dev/sdg /mnt/data-store2
[ec2-user@ip-10-1-11-134 ~]$ ls /mnt/data-store2/
file.txt  lost+found
```

Build a Database Server (Lab 5)

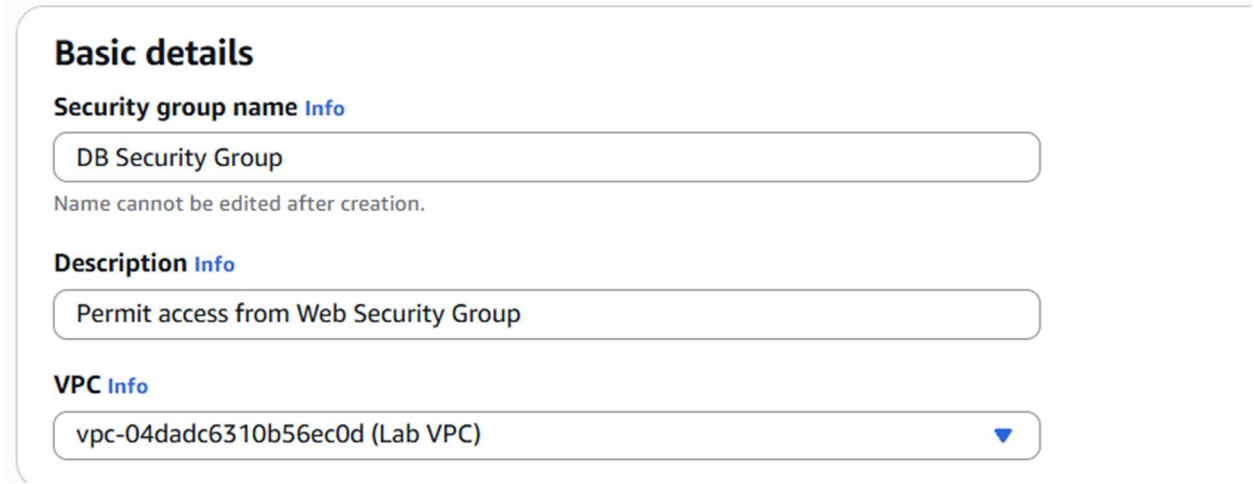
Using the search bar, open search for and open the VPC console



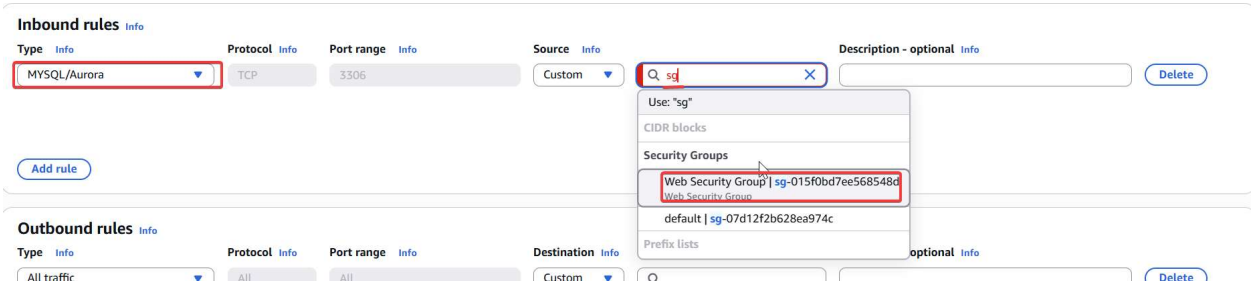
Click “Security groups” on the left menu, then the “Create security group” button



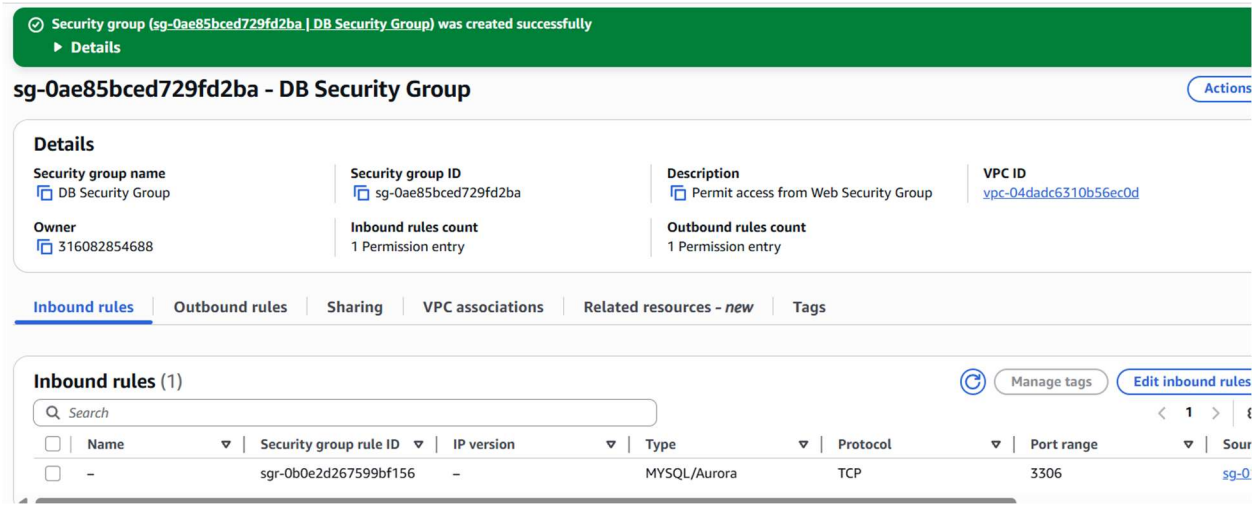
Configure the security group name and description as follows, then set the VPC to the pre-made Lab VPC



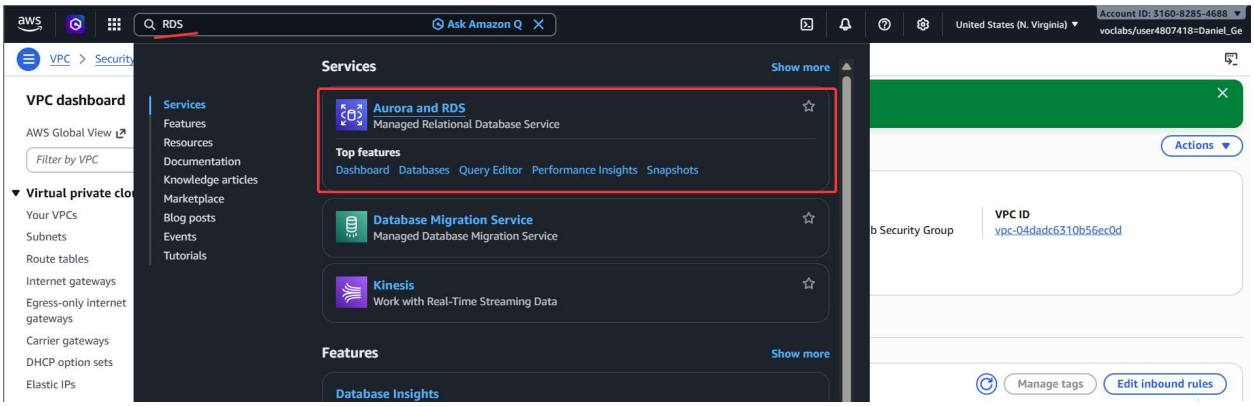
Set the type to MYSQL/Aurora, then type sg in the search bar to the right of Source to find security group “Web Security Group”



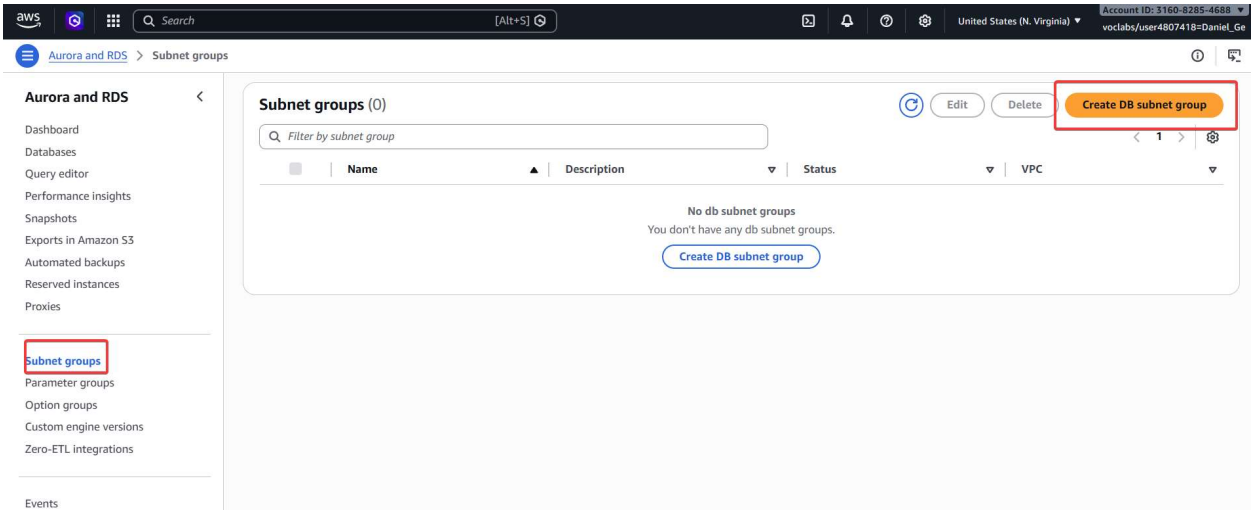
Click Create Security Group



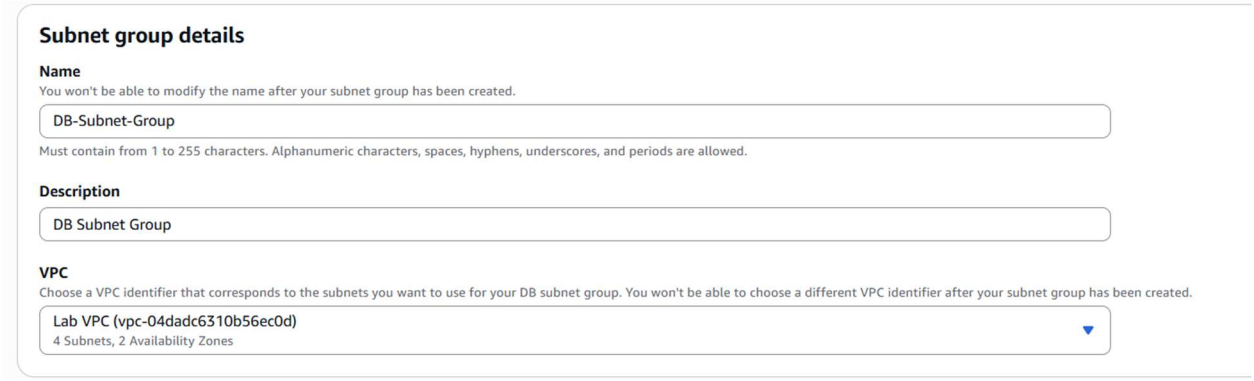
Using the search bar, navigate to the Aurora and RDS console



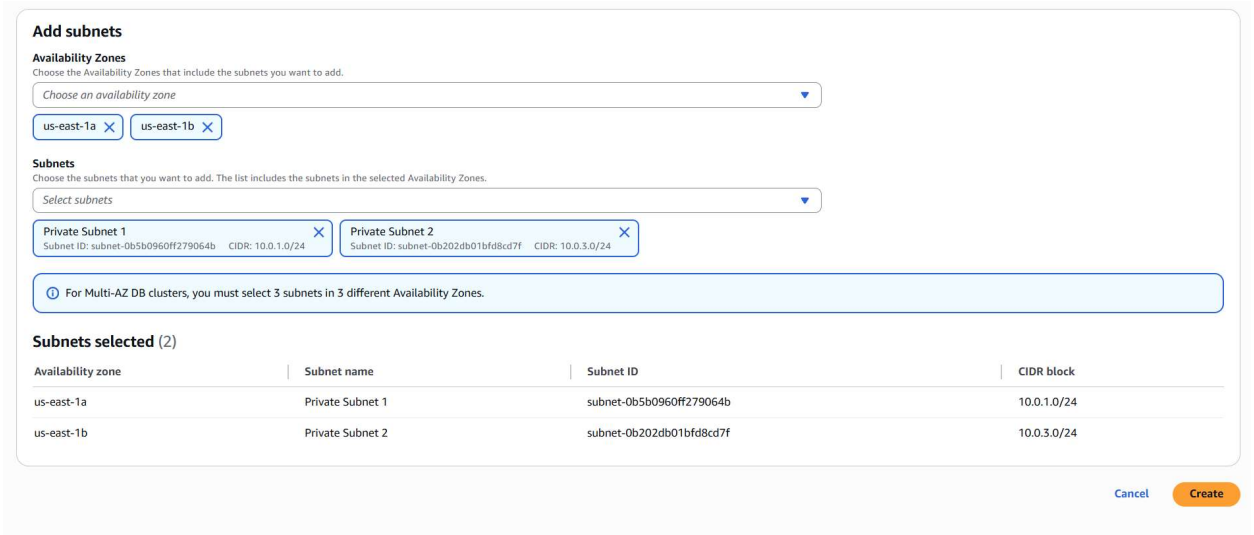
Click “Subnet groups”, then “Create DB subnet group”



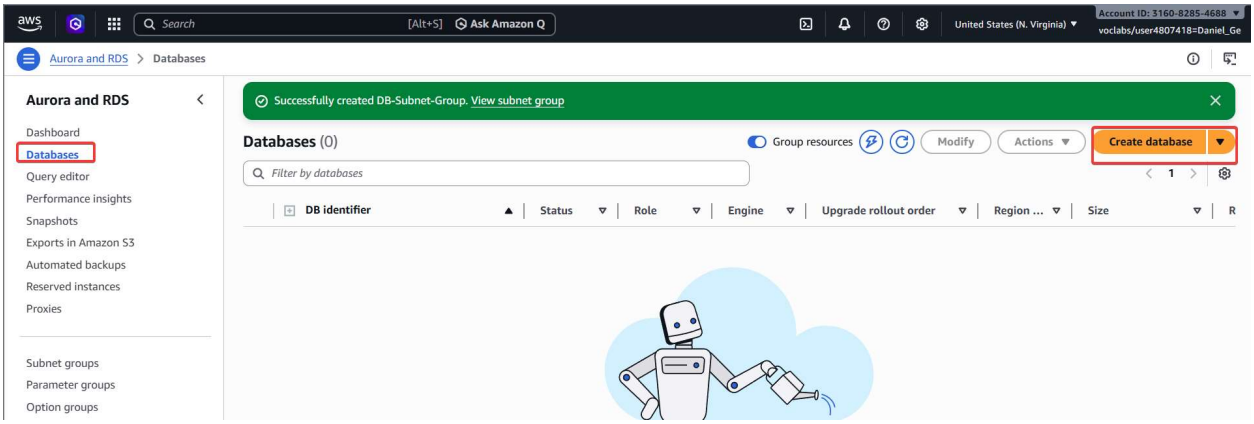
Configure the Name and Description as follows, then choose Lab VPC



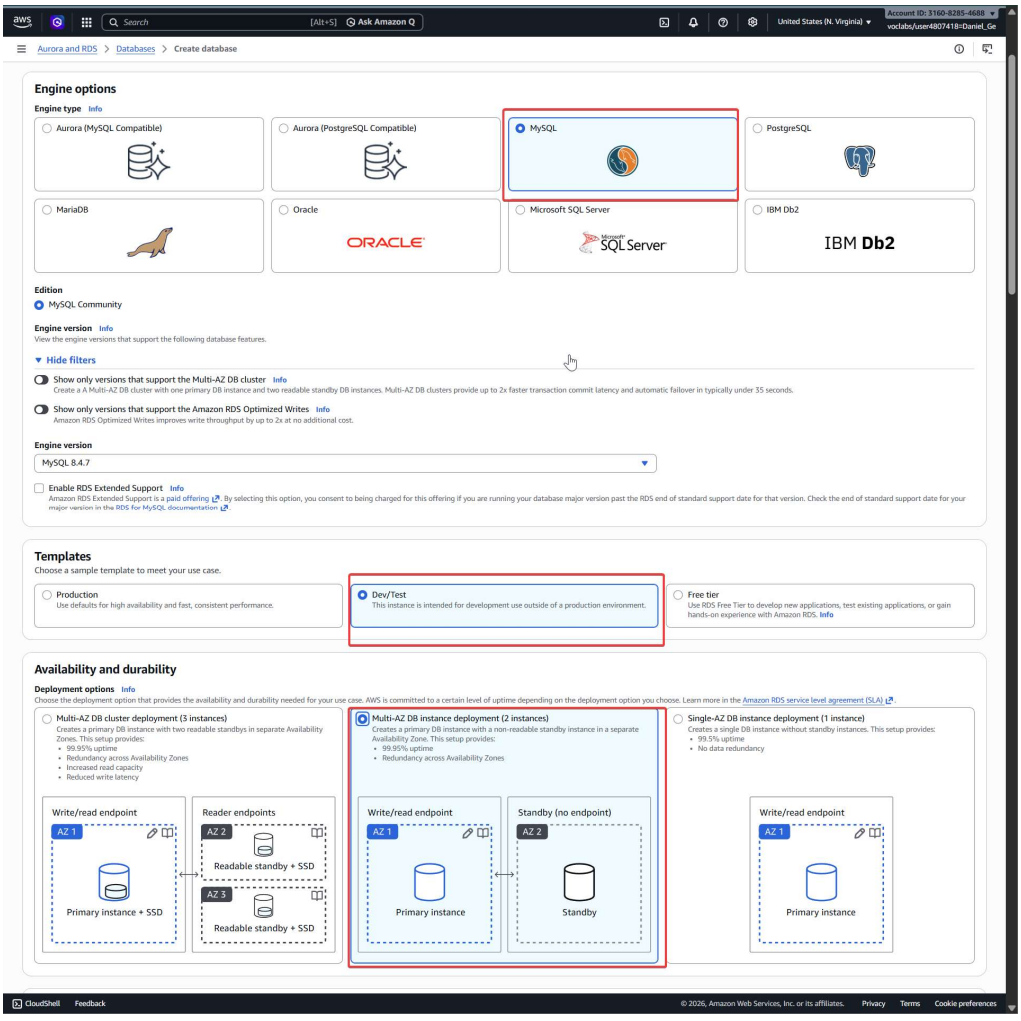
Add Availability zones “us-east-1a” and “us-east-1b” and subnets “Private Subnet 1” and “Private Subnet 2”, then click Create



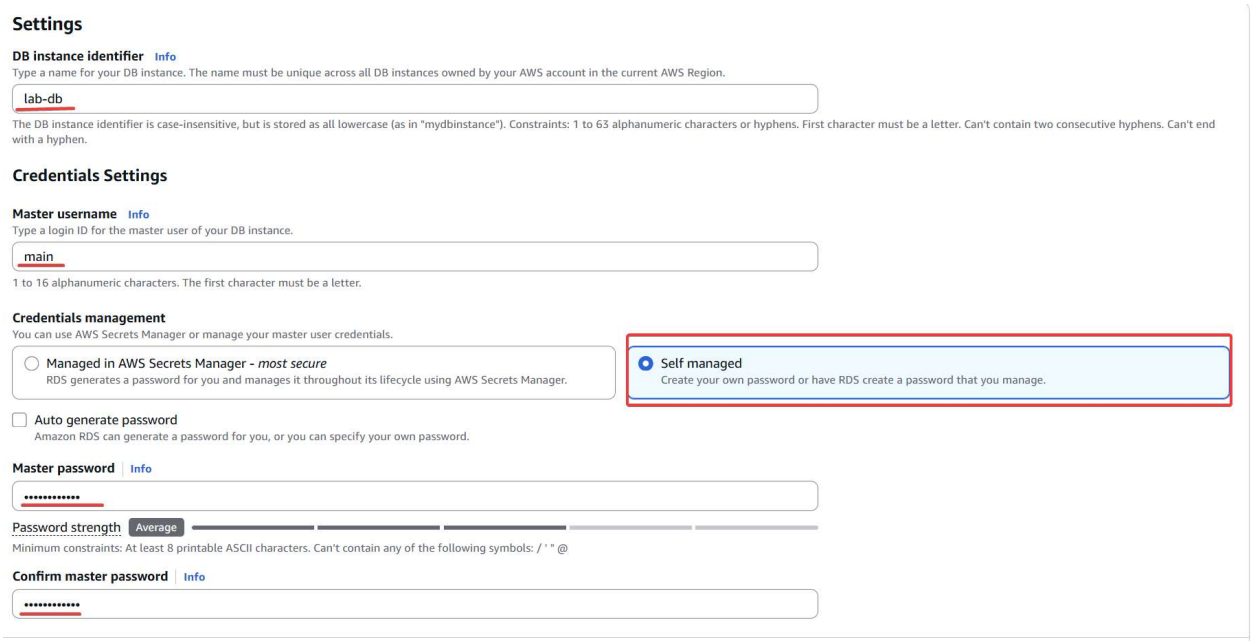
Click the “Databases” tab, then “Create database”



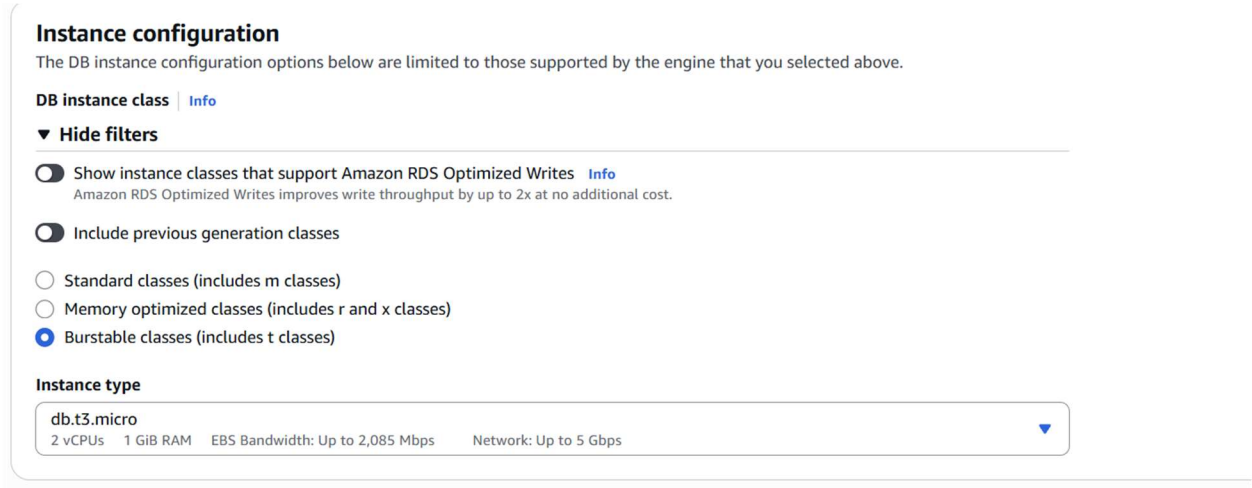
Choose MySQL, Dev/Test, and 2 instances



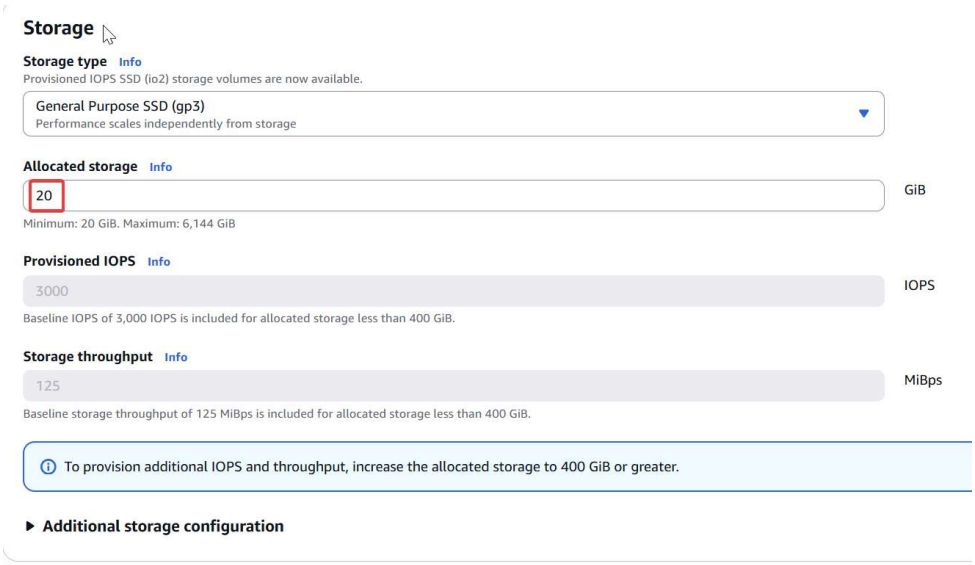
Configure the DB instance identifier and Master username as follows, then select “Self managed” and set the Master password to “lab-password”



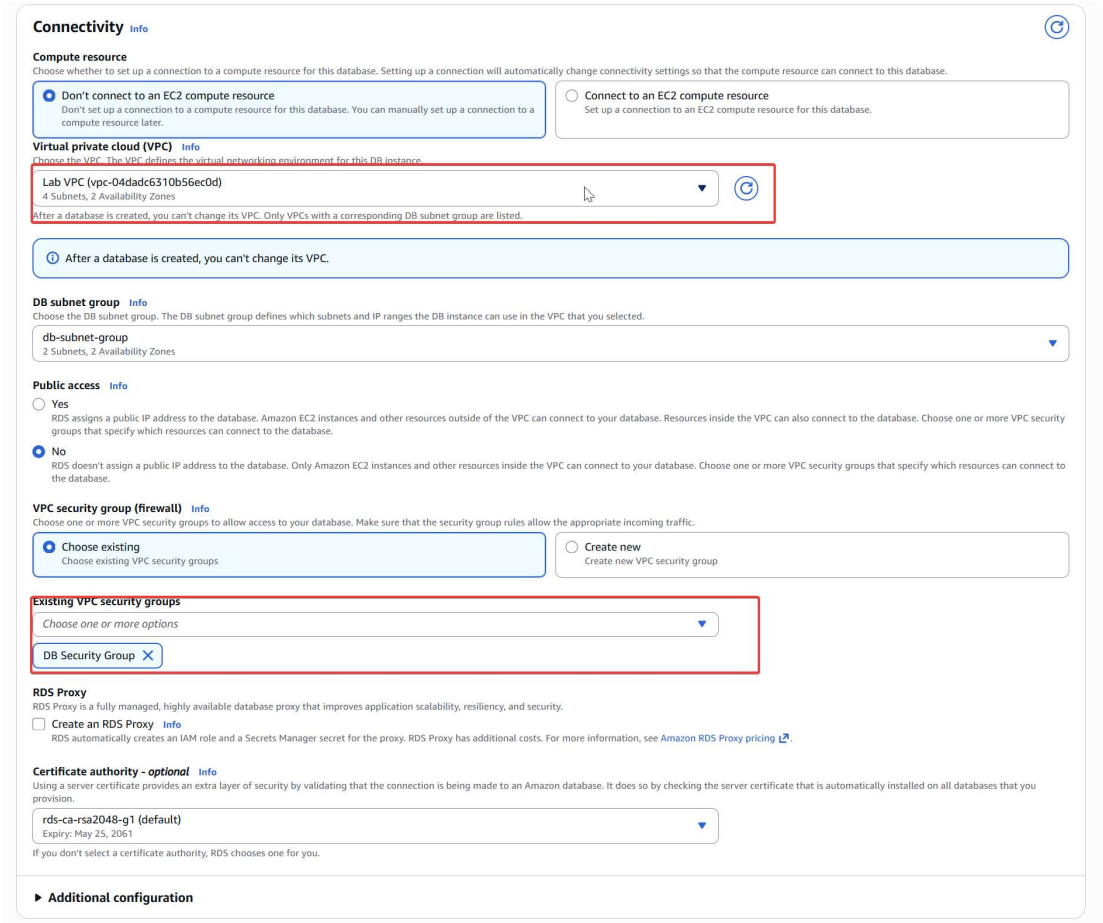
Select Burstable classes under Instance configuration



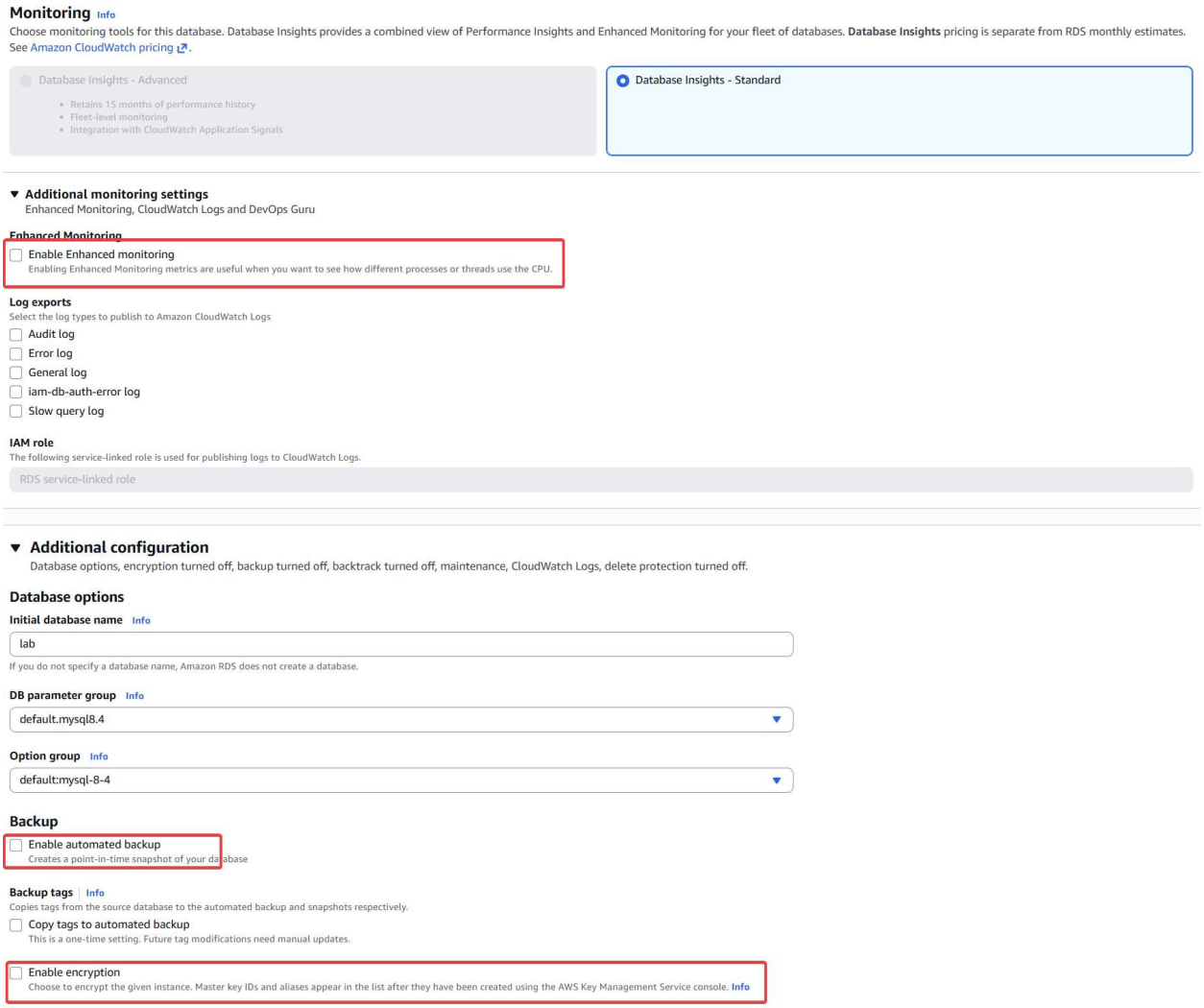
Change the Allocated storage to 20



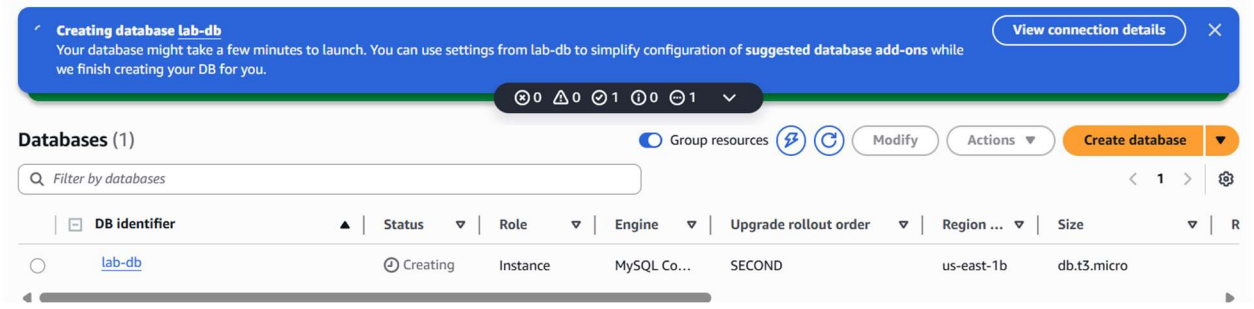
Select “Lab VPC” for the Virtual private cloud, and configure the Existing VPC security group to only DB Security Group



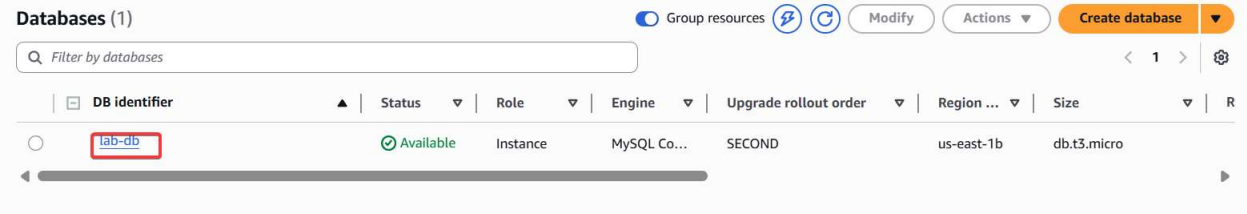
Under monitoring, uncheck “Enable Enhanced monitoring”, uncheck “Enable automated backup”, and uncheck “Enable encryption”



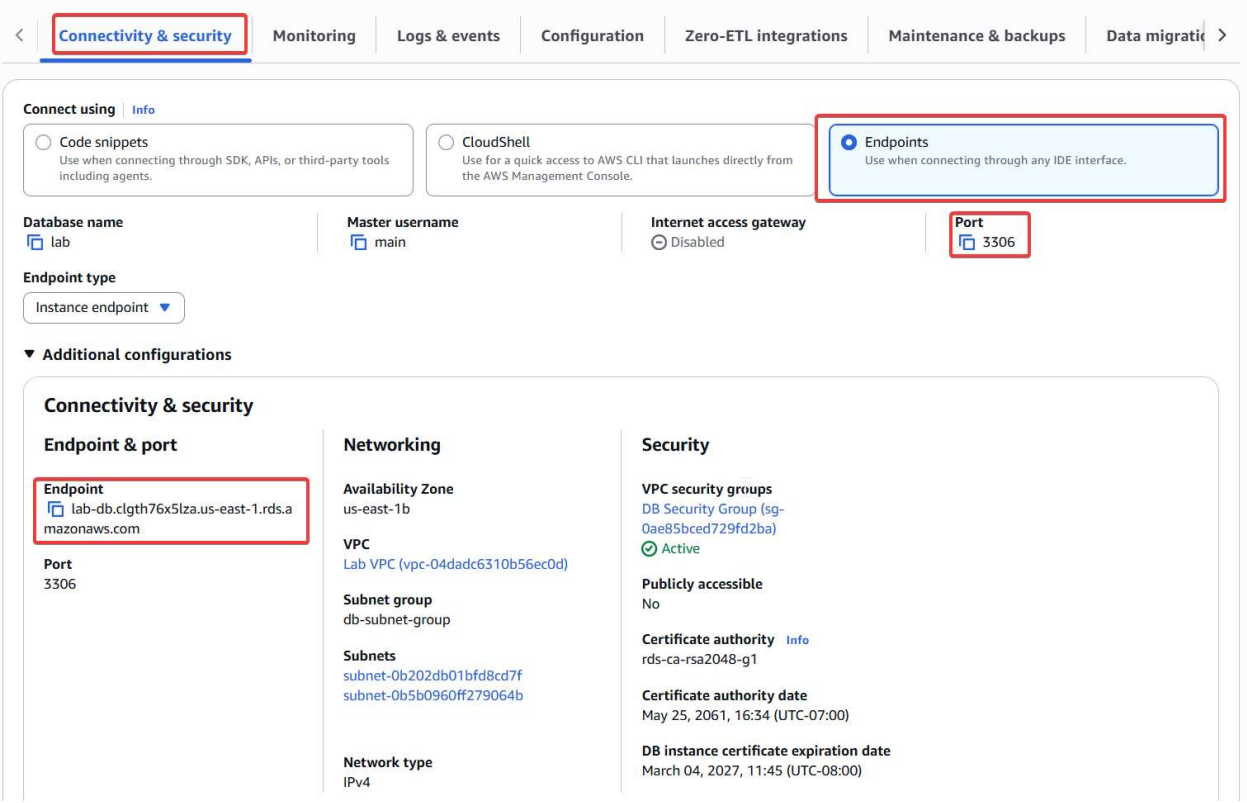
Click “Create database” and wait approximately 5 minutes for the database to deploy



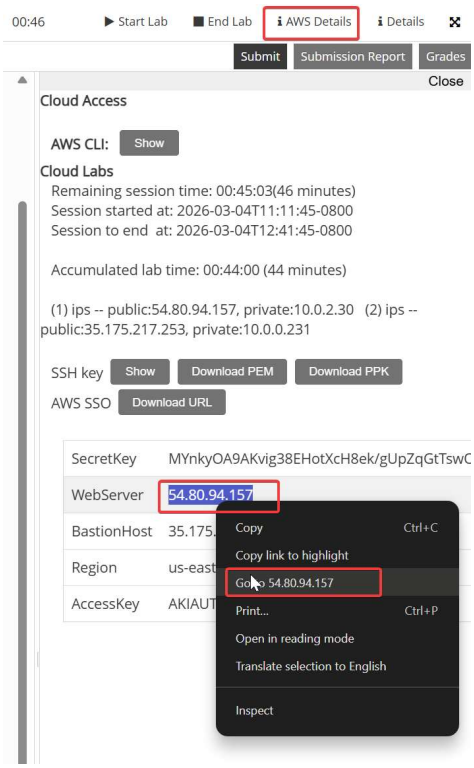
Click the blue lab-db link to open the lab-db page



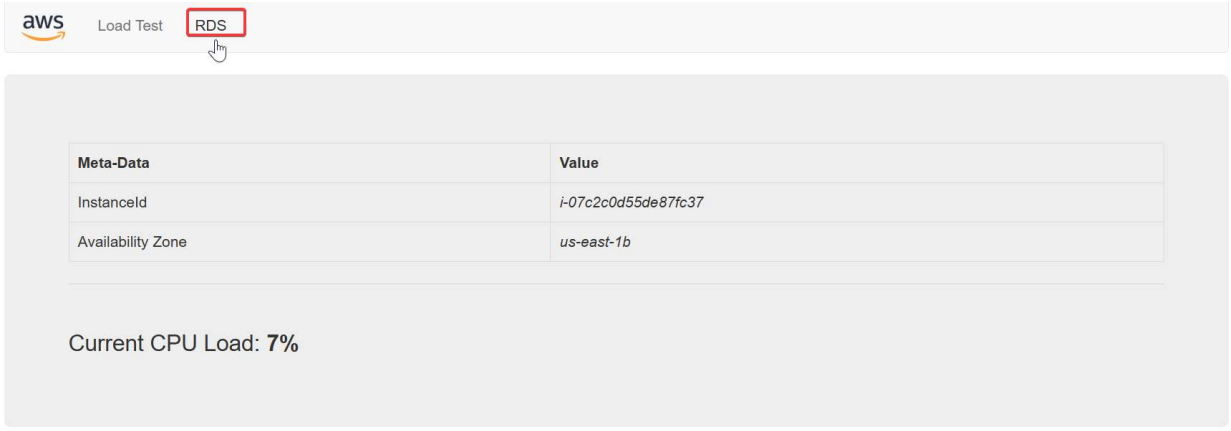
Under the connectivity & security tab, click “Endpoints” and copy the Endpoint URL



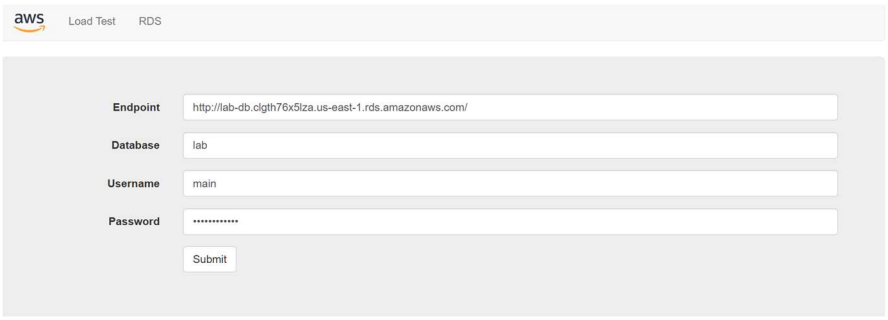
Return to the lab instruction page used to start the lab. Click “AWS Details” at the top right, then paste the WebServer IP into the browser



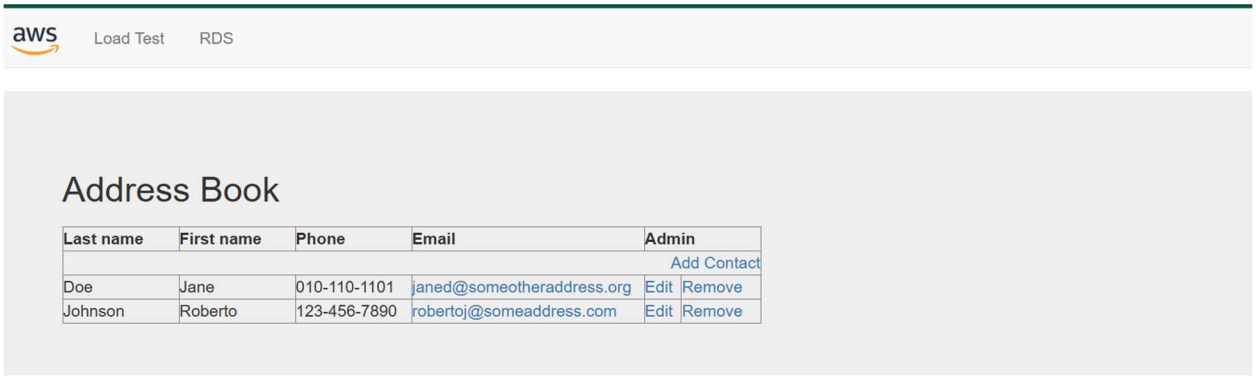
The page displays information about the EC2 instance. Click the RDS link



Paste the previously copied URL into Endpoint, enter “lab” for Database, “main” for Username, and “lab-password” for Password. Click Submit.

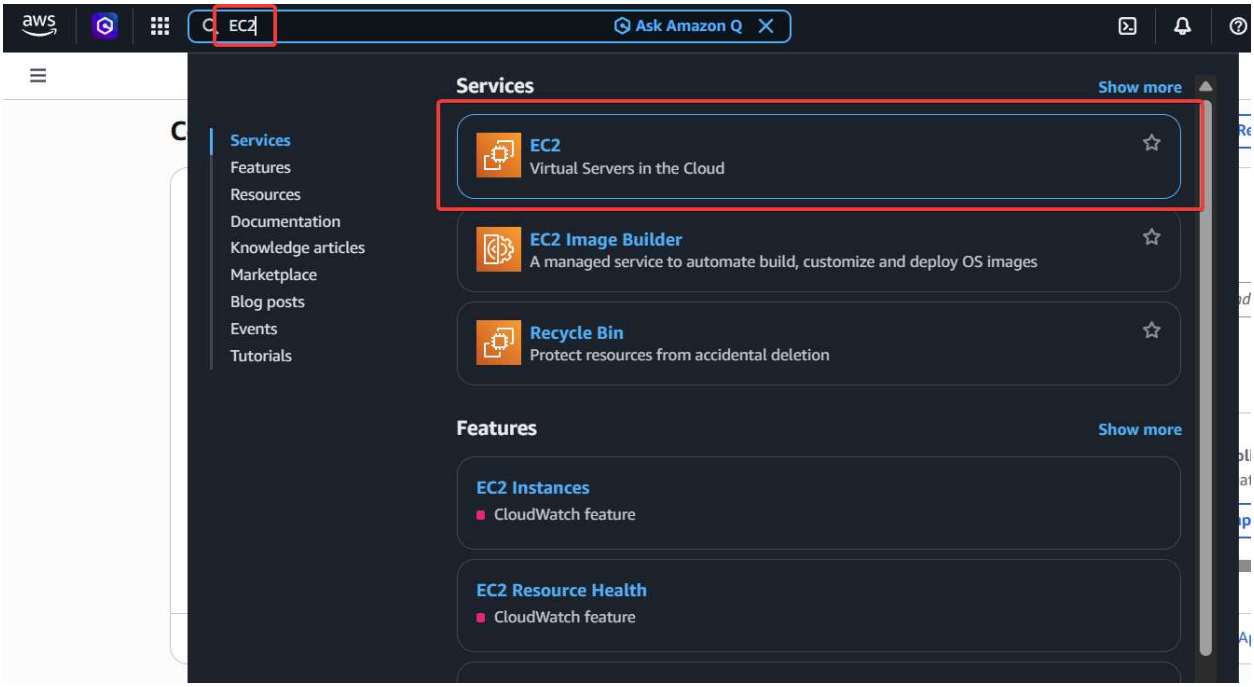


The page will now display an address book from the App, which can be edited. The changes will be reflected in the RDS database across multiple regions

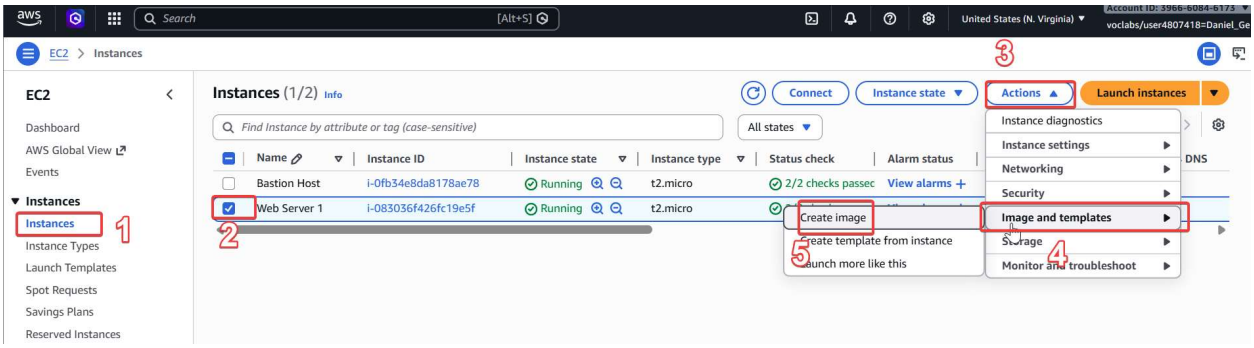


Load Balancing (Lab 6)

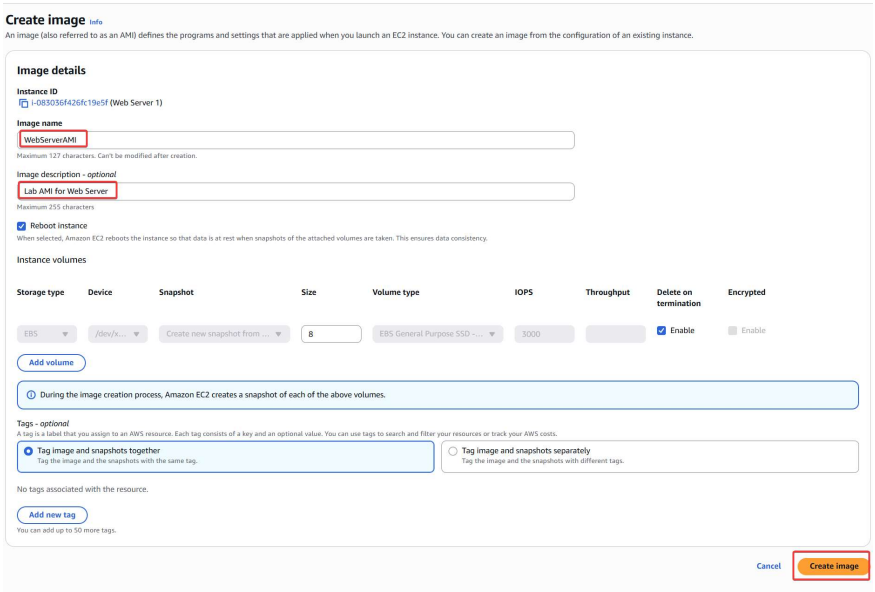
Using the search bar, navigate to AWS EC2 menu



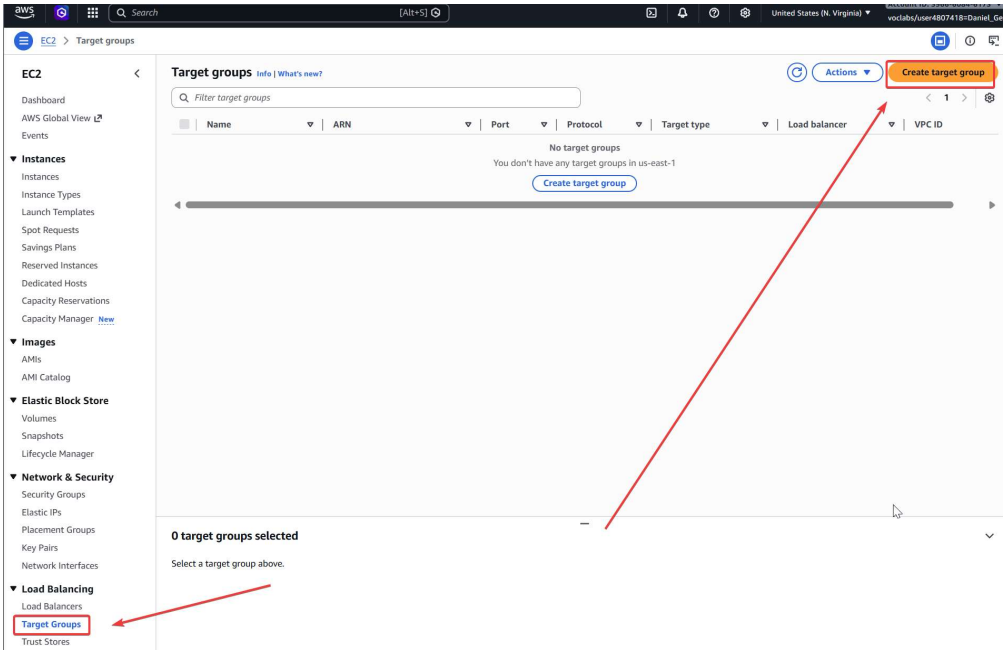
Go to the Instances tab using the left menu. Click Web Server 1, Actions > Images and templates > Create image



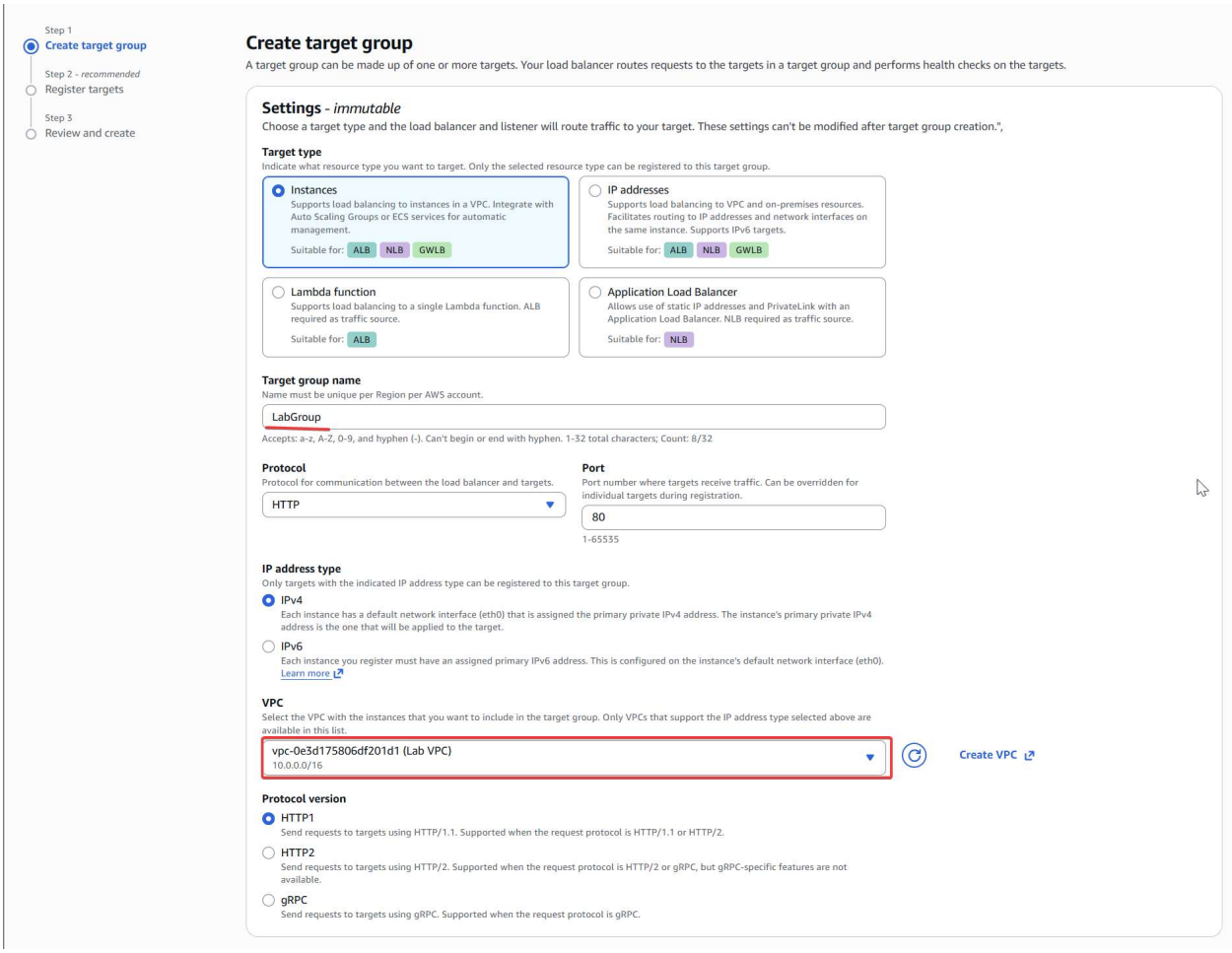
Name the image “WebServerAMI” and configure description “Lab AMI for Web Server”, then click Create Image



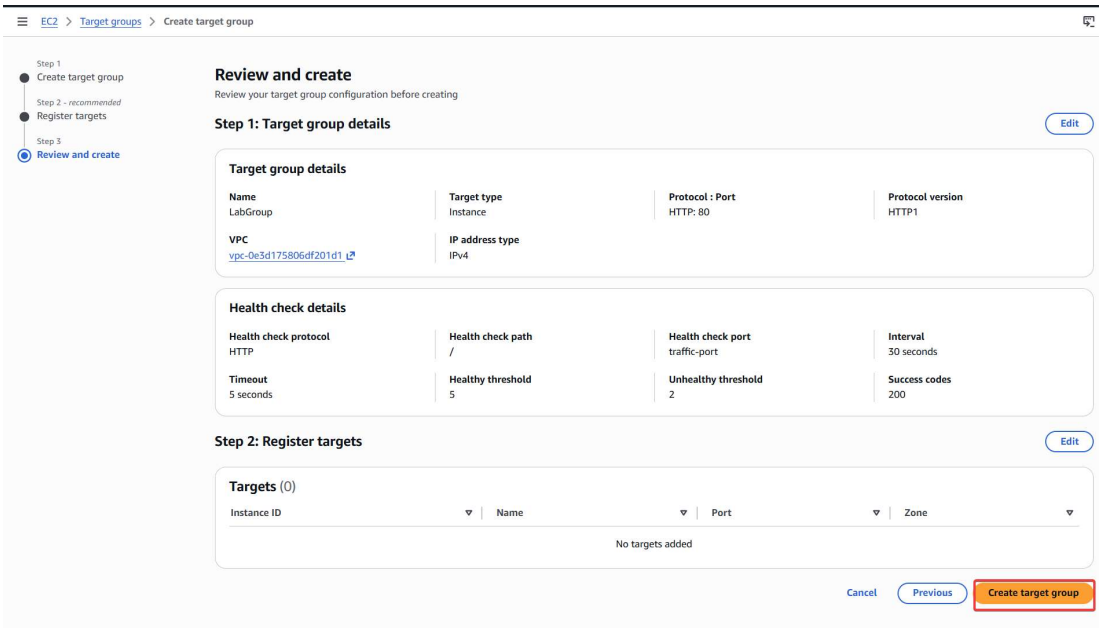
We now create a Load Balancer. In the left menu, go to the Target Groups tab. Then, click “Create target group”



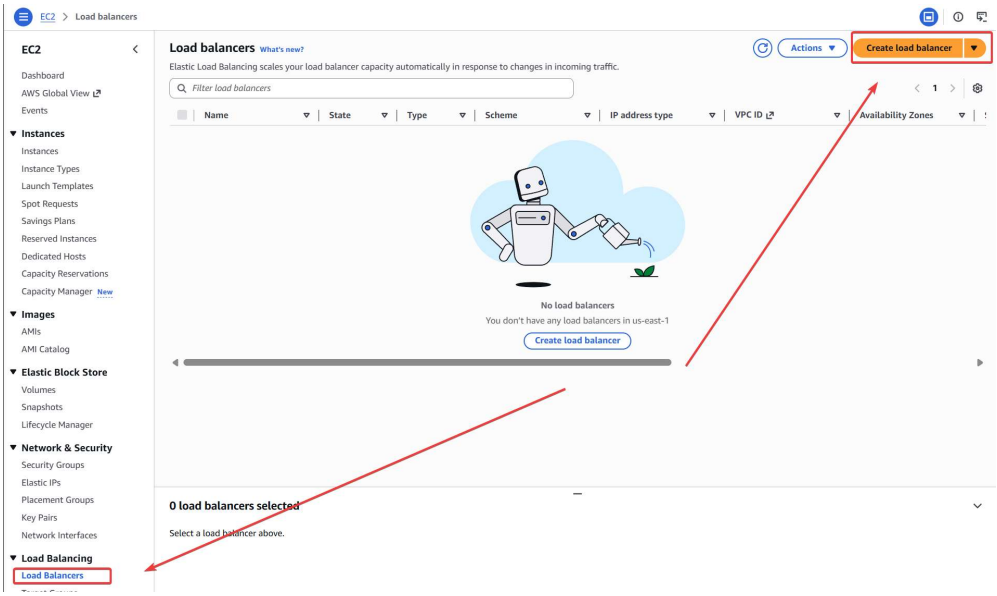
Enter group name “LabGroup”; select Lab VPC. Then, click “Next” at bottom of the screen to proceed.



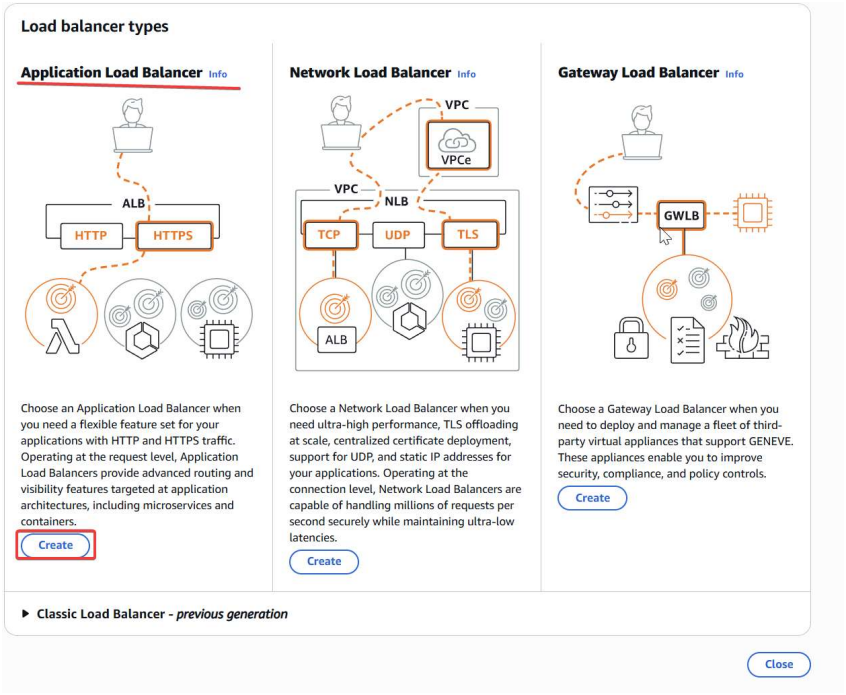
Click “Next” and skip the Register targets step, then click “Create target group”



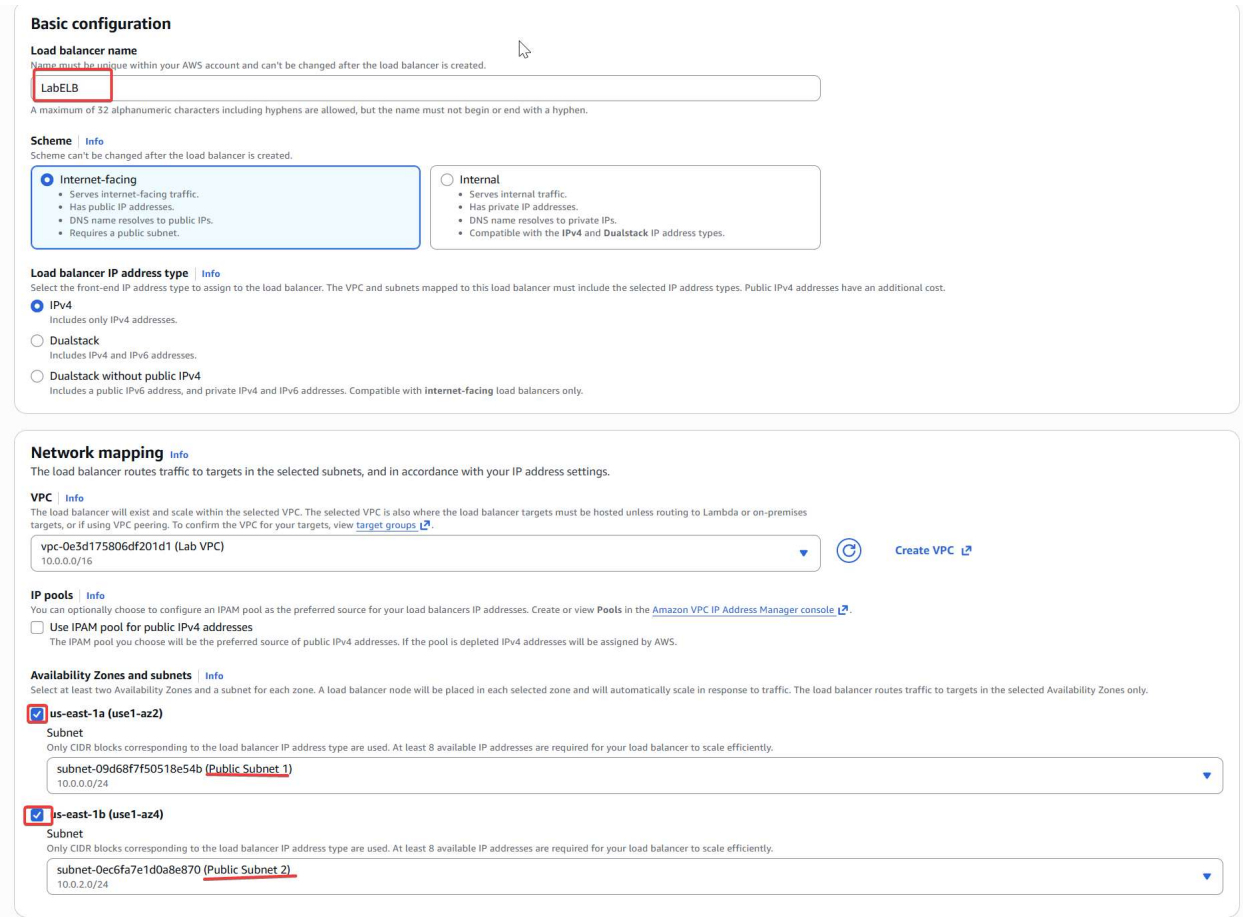
Now click the “Load Balancers” tab on the left sidebar, then click “Create load balancer”



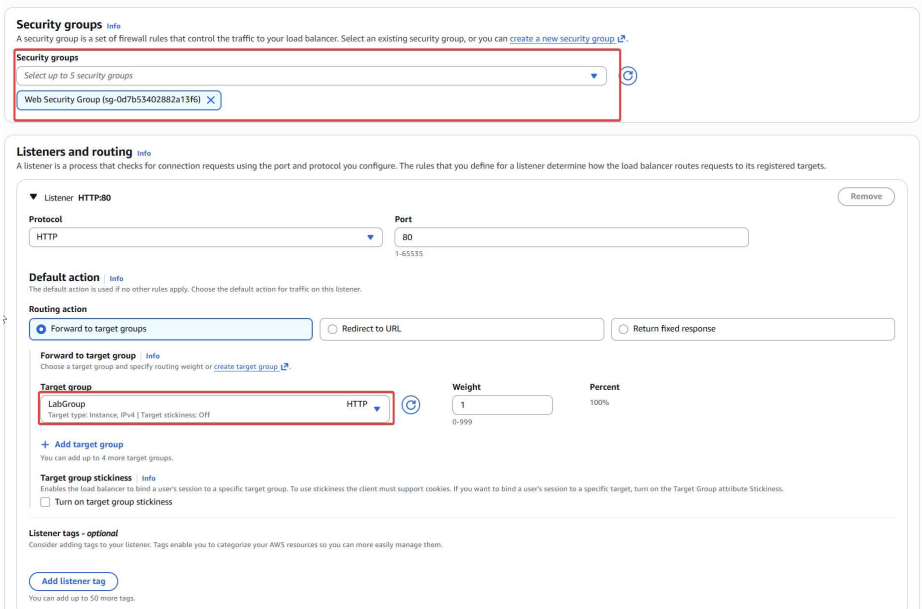
Click Create under Application Load Balancer



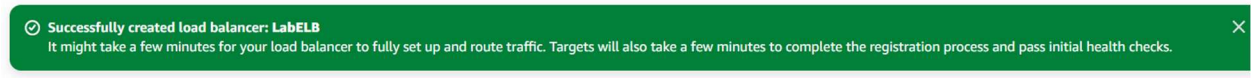
Name the Load balancer “LabELB”, select Lab VPC, then select Public Subnet 1 and Public Subnet 2 under the two Availability zones



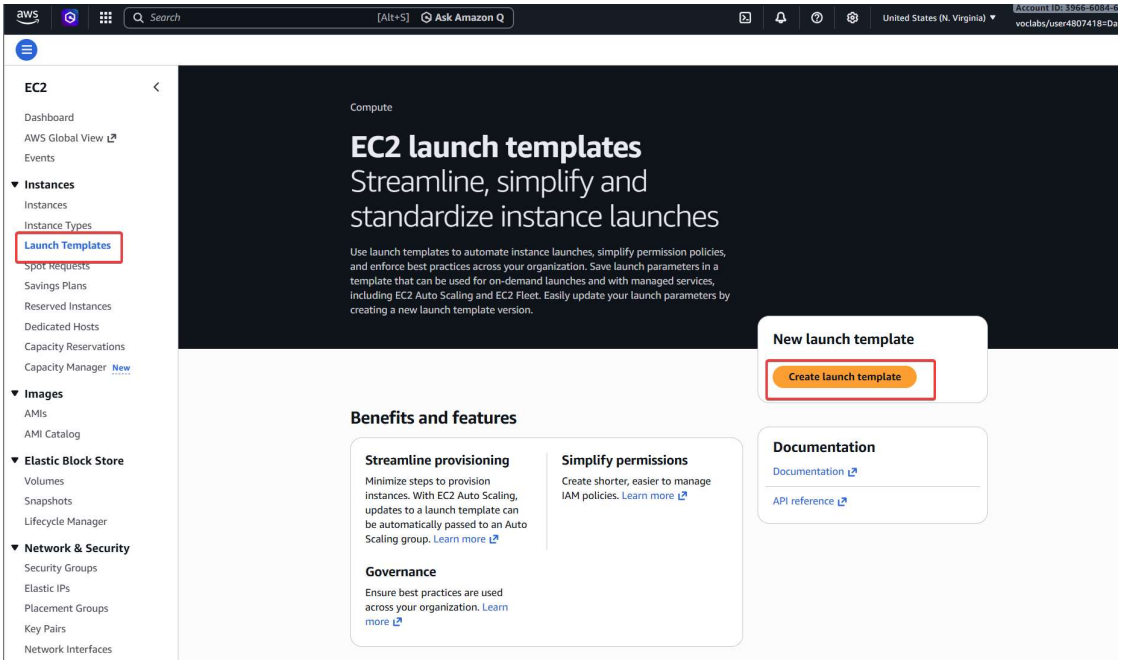
Under Security groups, deselect Default and select Web Security Group; under “Target group” for Default action of forward to target groups, select LabGroup



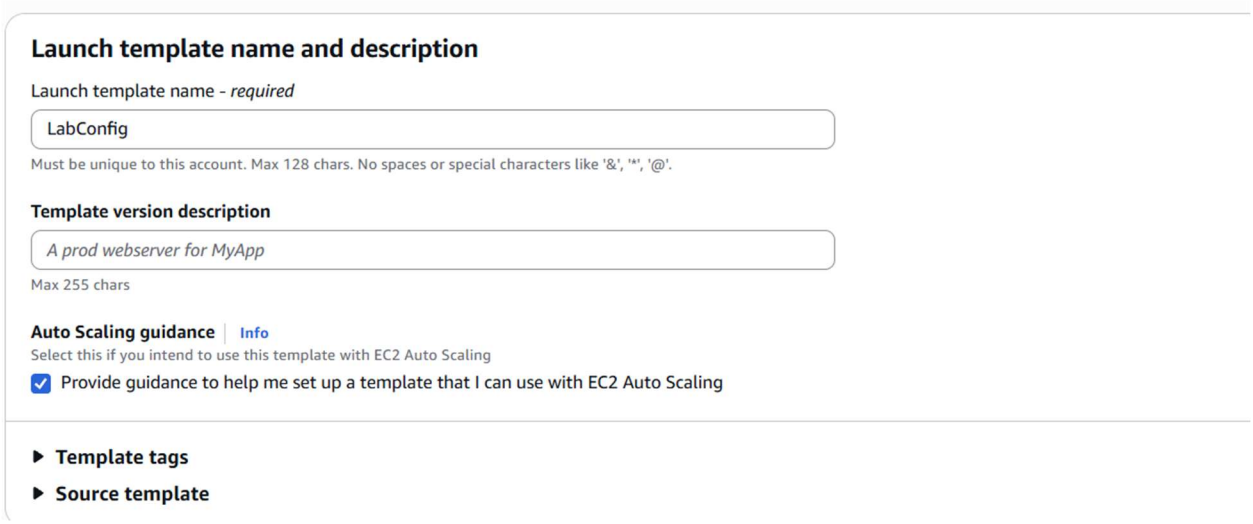
Scroll to bottom and click Create load balancer



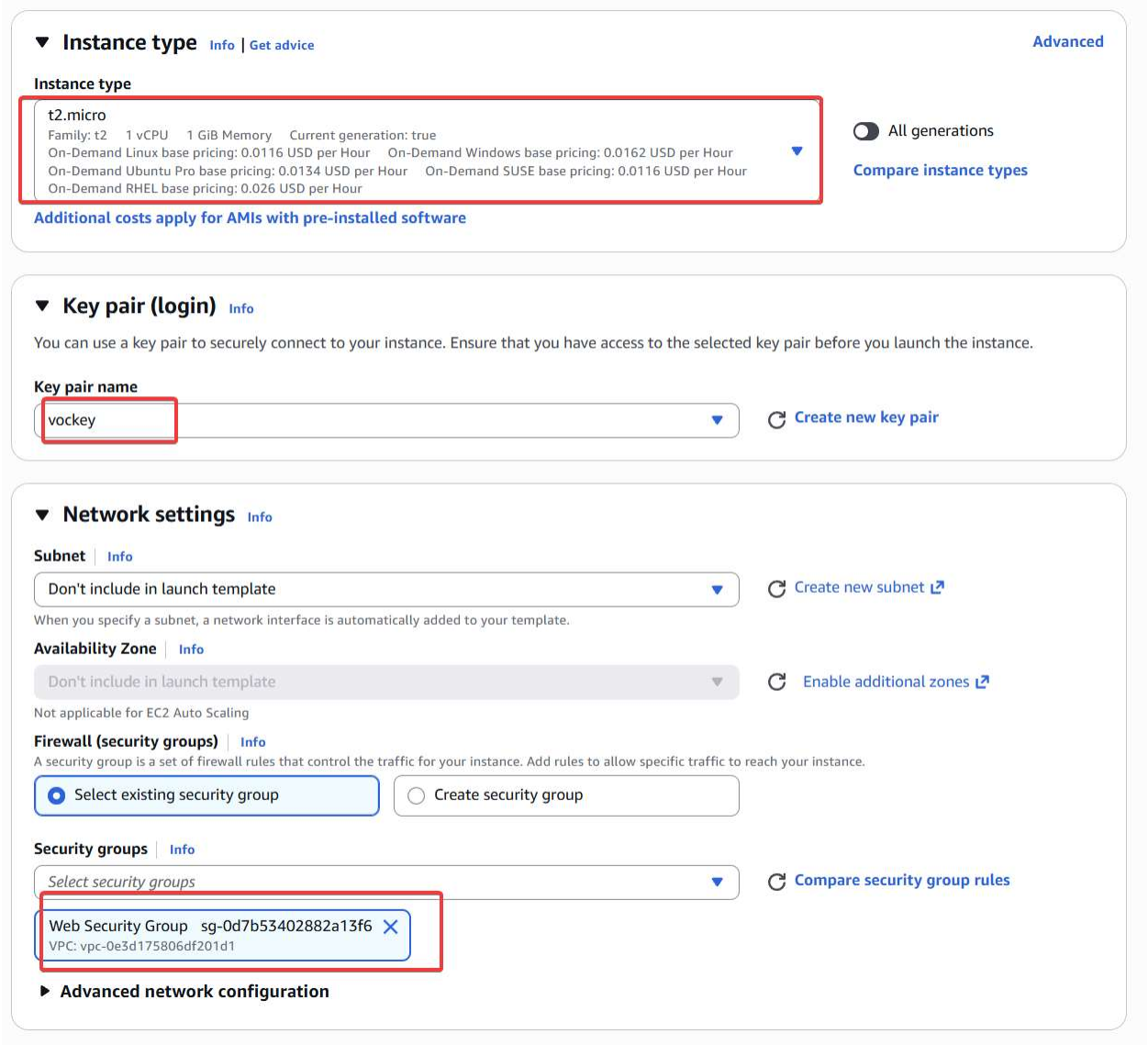
In the left menu, choose the Launch Templates tab. Then, click “Create launch template”



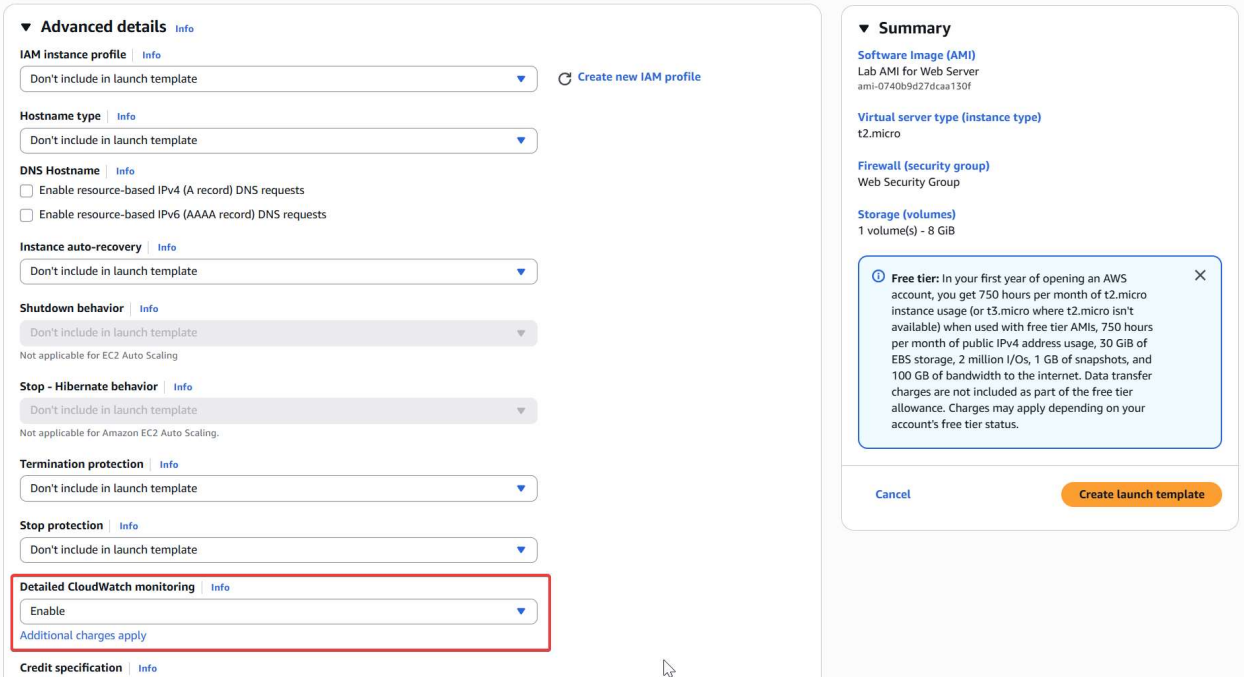
Name the template “LabConfig” and select Auto Scaling guidance



Select t2.micro for instance type, vockey for key pair name, and “Web Security Group” for security group

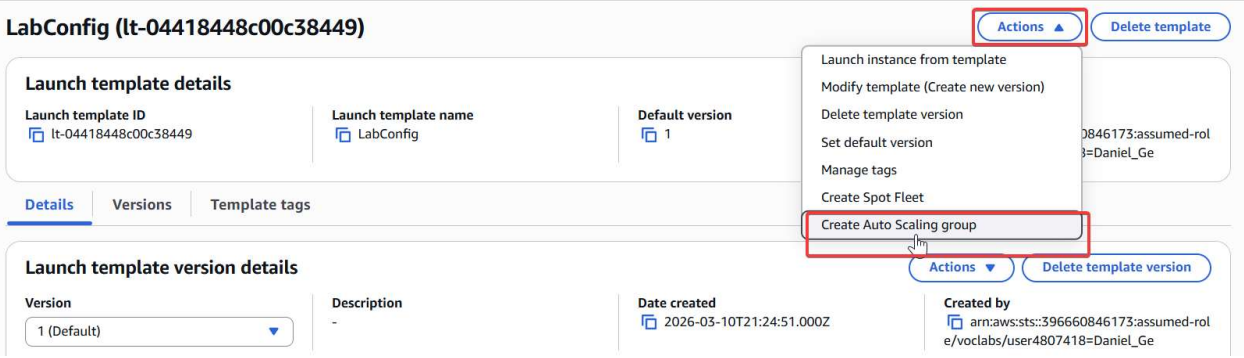


Enable Detailed CloudWatch monitoring and Create Launch template

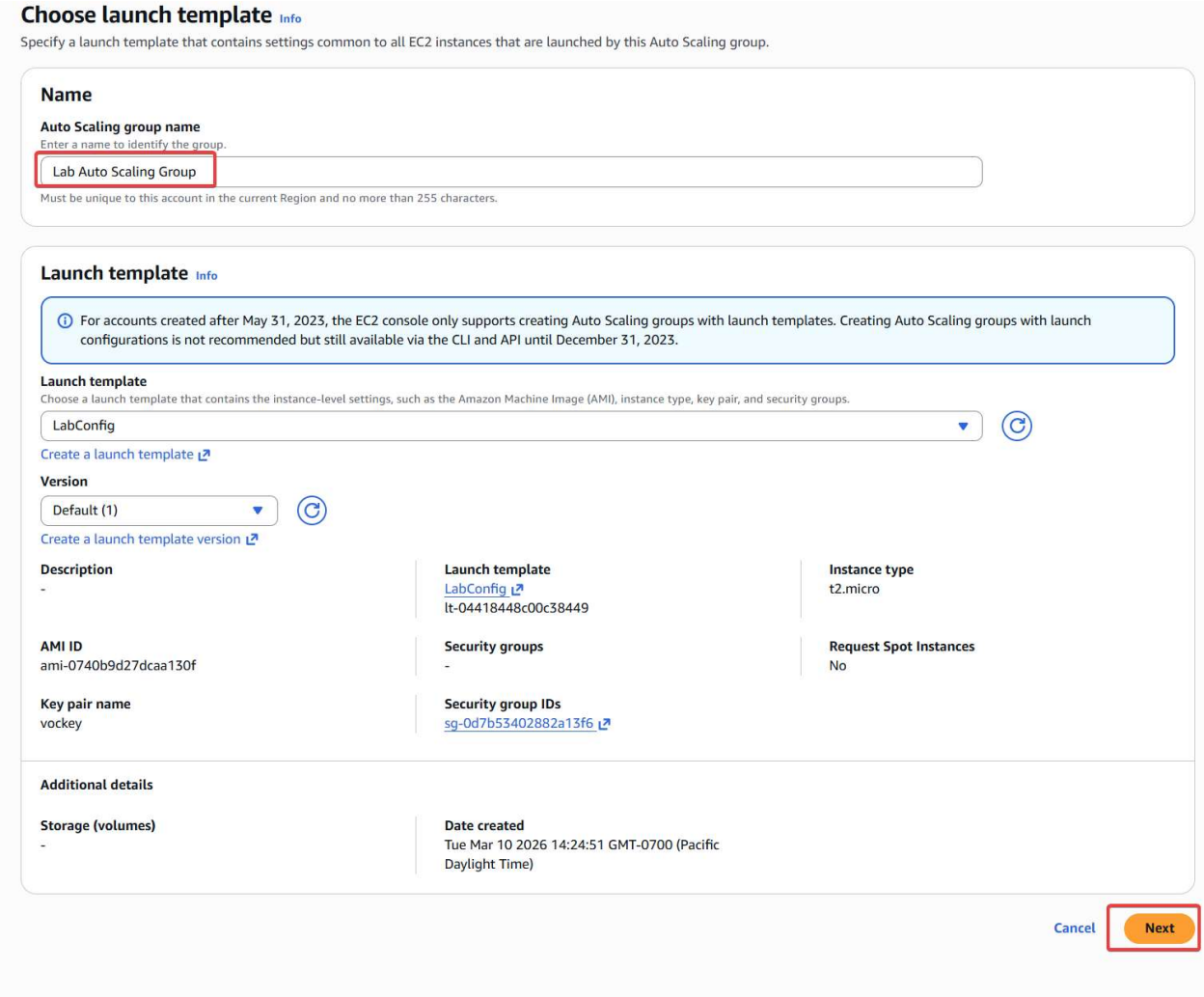


Click LabConfig in the success message

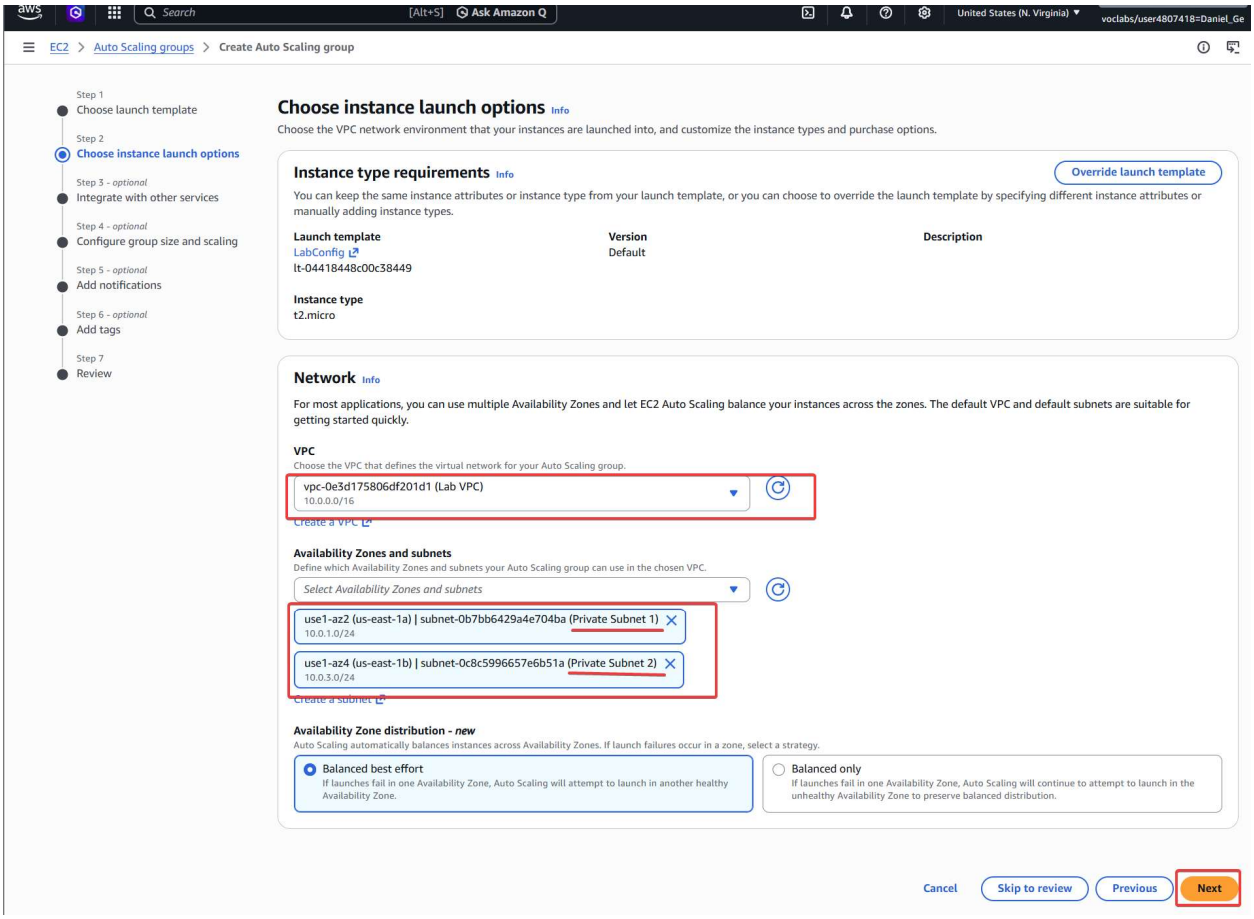
Under Actions, click “Create Auto Scaling Group”



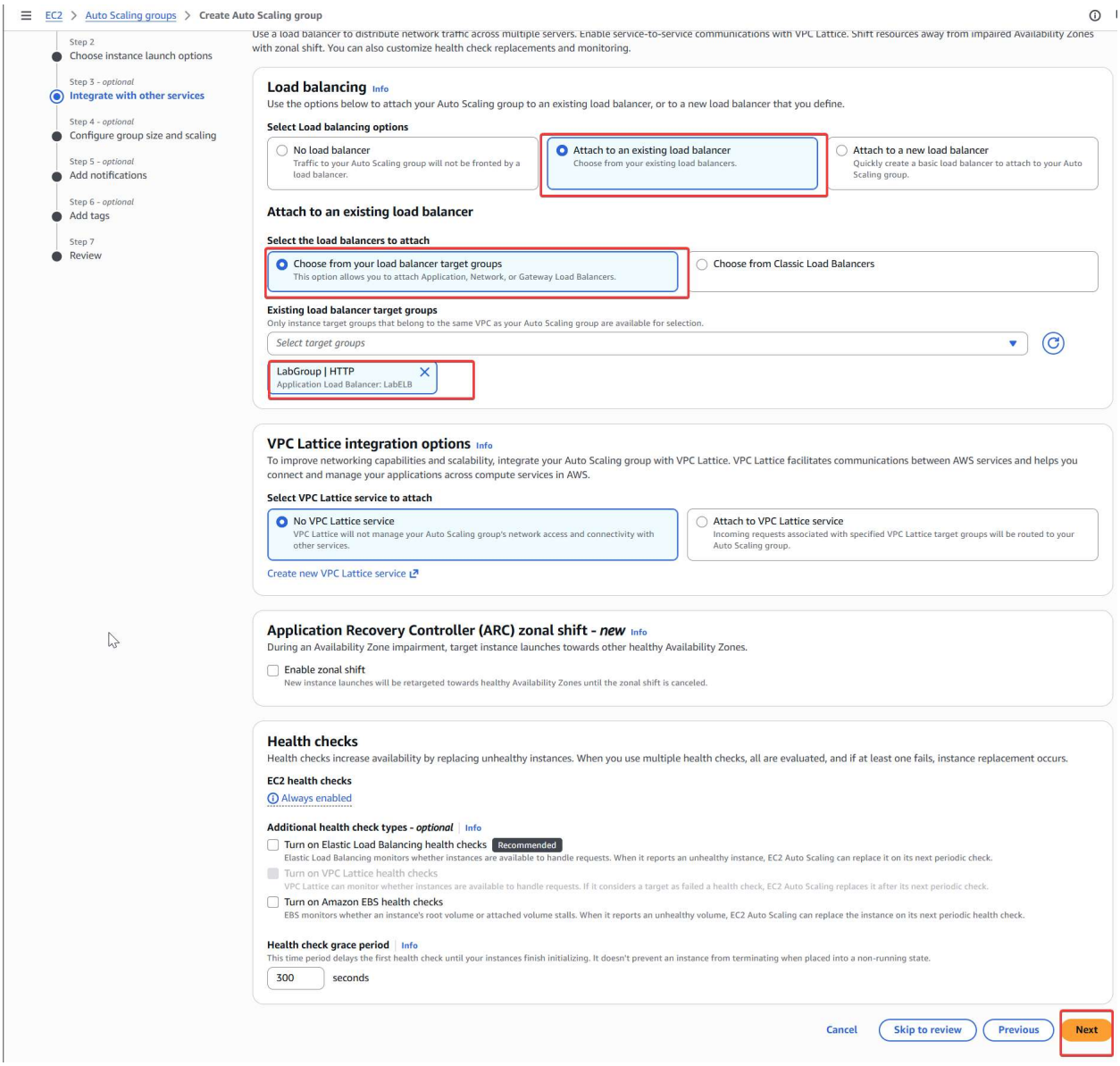
Name the group “Lab Auto Scaling Group”, then click “Next”



Choose Lab VPC for VPC. Choose Private Subnet 1 and Private Subnet 2 for Availability Zones and subnets, then click Next.



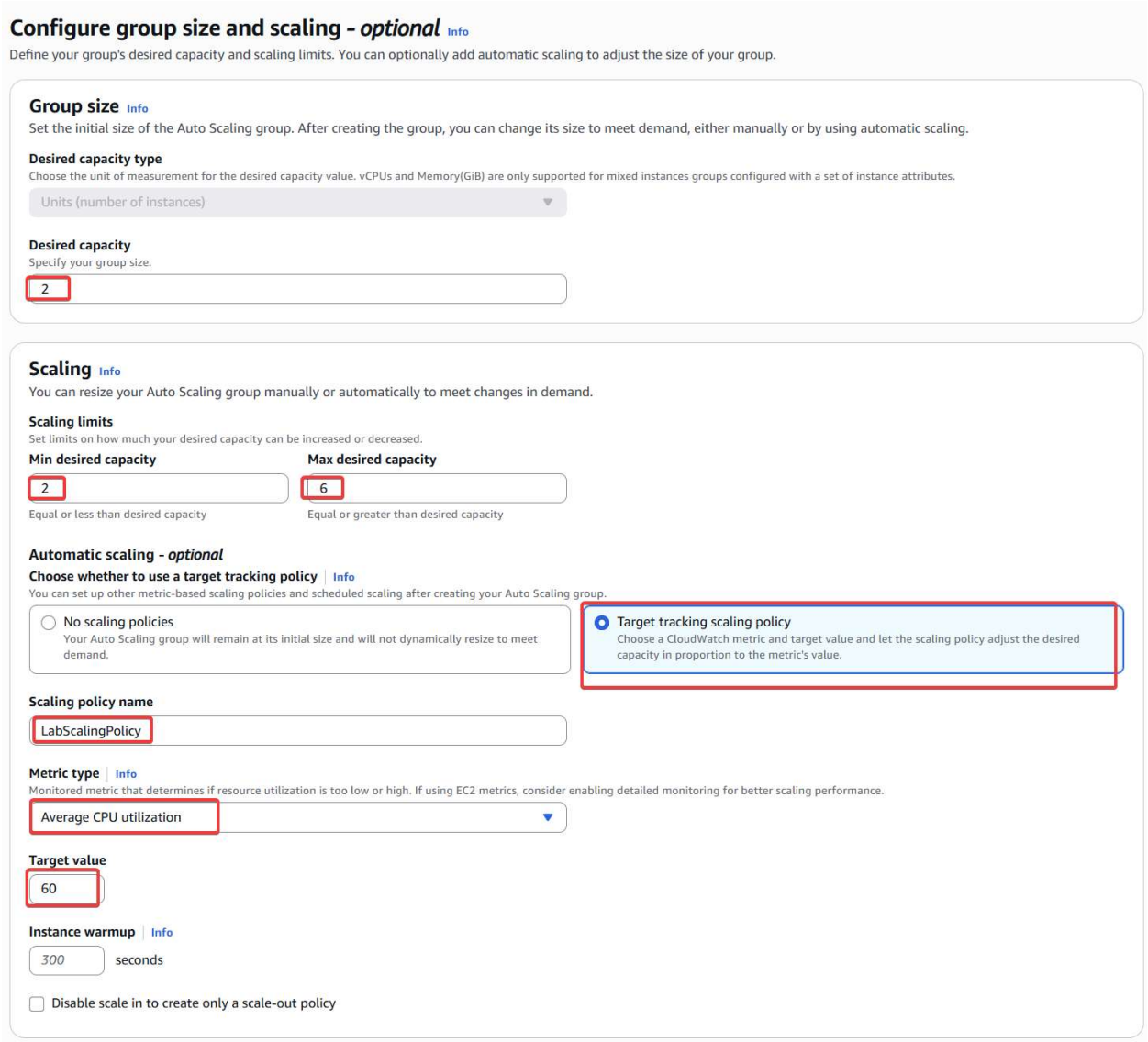
Choose “Attach to an existing load balancer” and select LabGroup as the target group. Click “Next” to continue



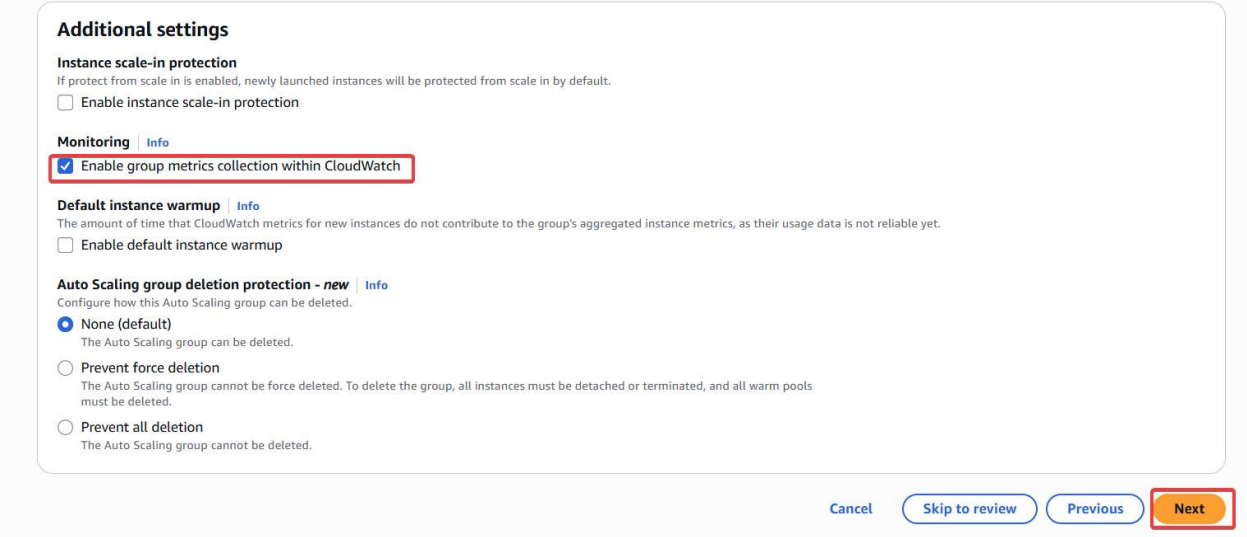
Make the following configurations

- Desired capacity: 2
- Min desired capacity: 2
- Max desired capacity: 6

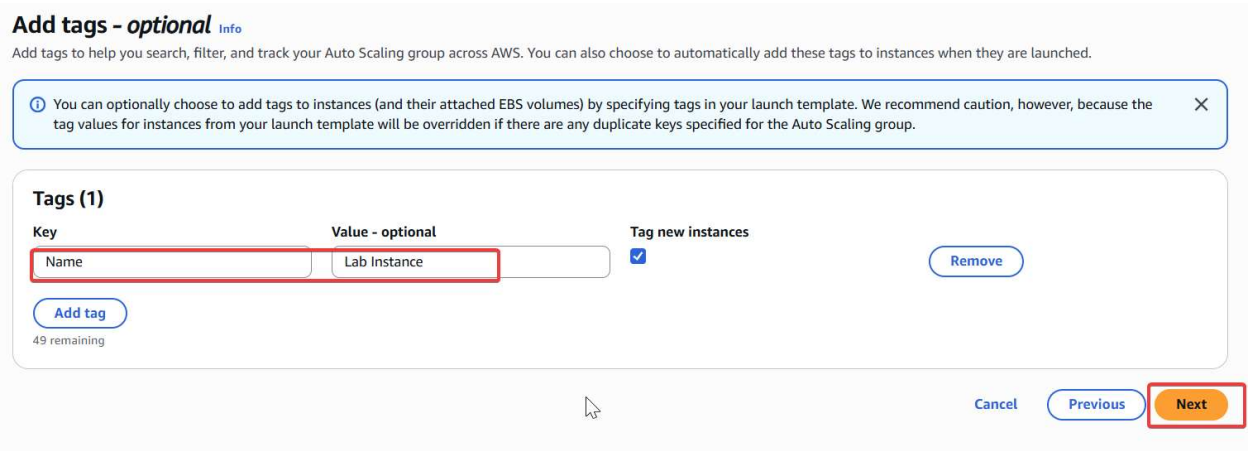
Select “Target tracking scaling policy”, name the policy “LabScalingPolicy”, then set Metric type to Average CPU utilization with target value 60



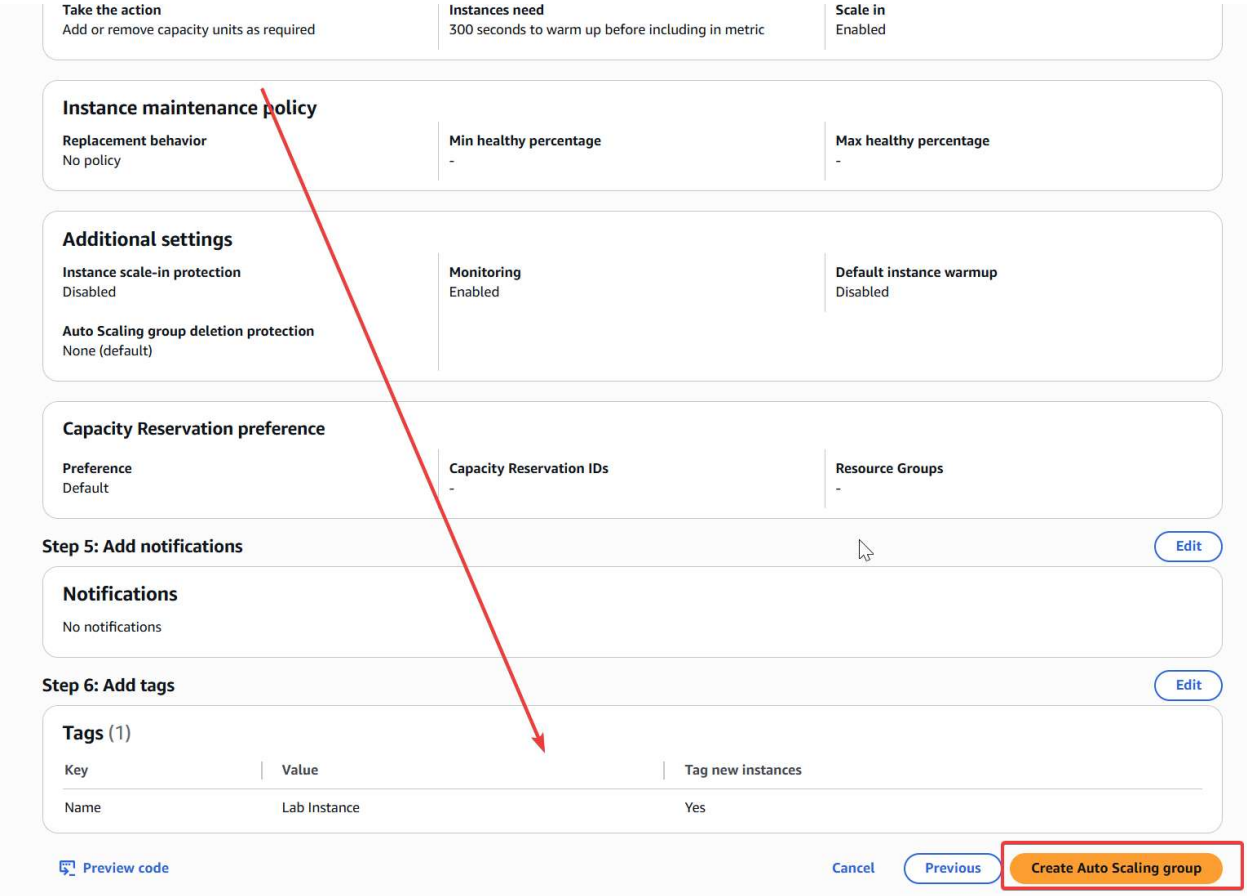
Enable group metrics collection within CloudWatch, then click Next



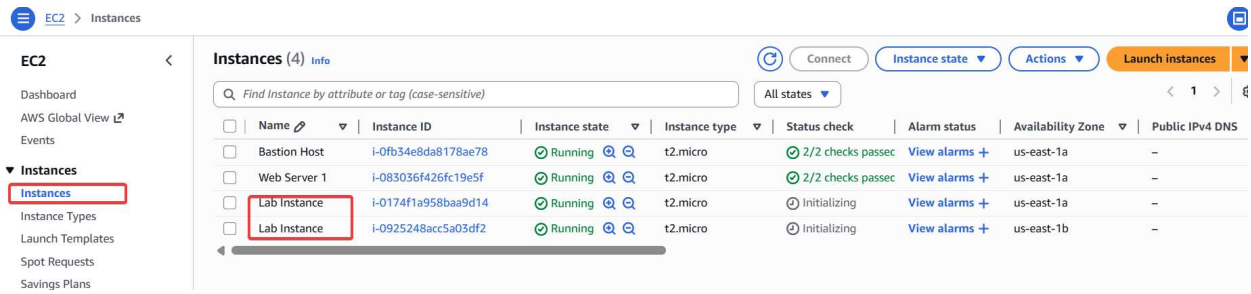
Click Next to skip Step 5, then click “Add tag” with Key “Name” and Value “Lab Instance” in Step 6. Click Next.



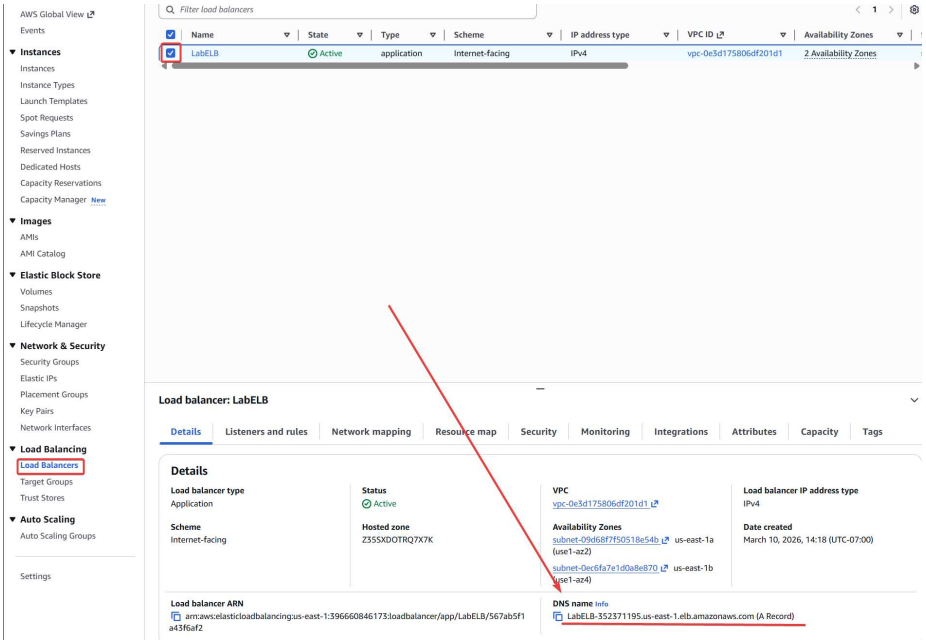
Review the configuration, then click “Create Auto Scaling group”



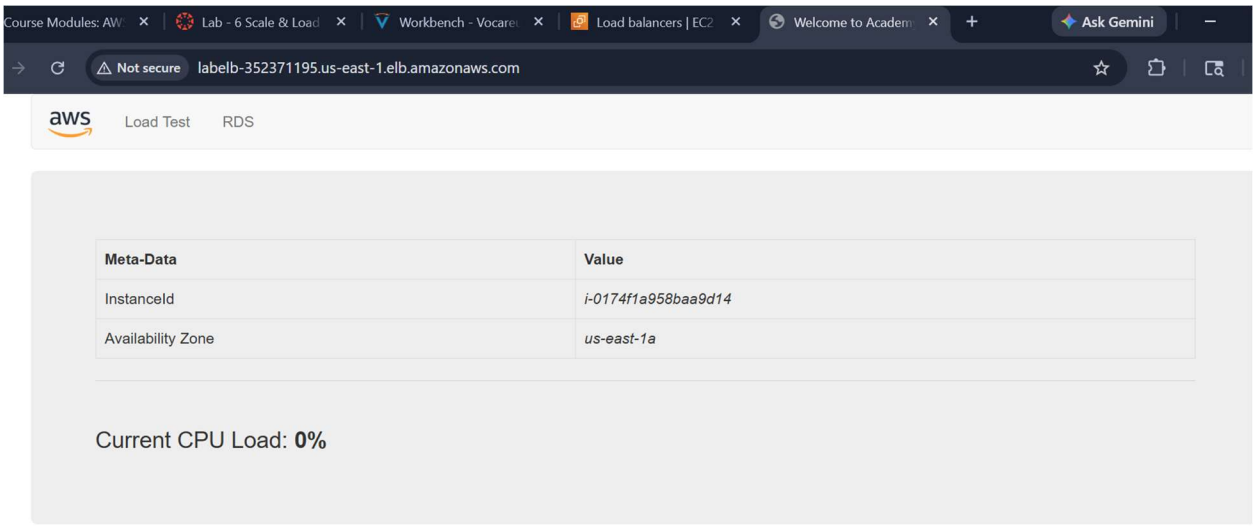
Go to the Instances tab using the left menu to check there are now two instances named “Lab Instance” launched by Auto Scaling



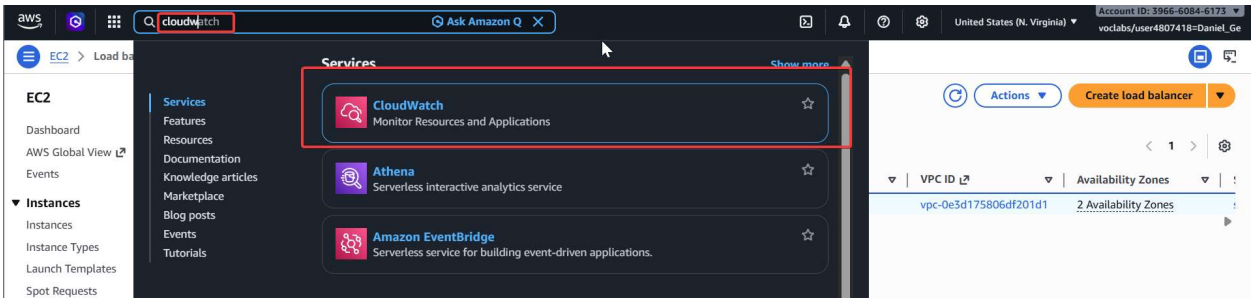
Select the Load Balancers tab on the left menu, then select the LabELB load balancer. Find and copy the DNS name



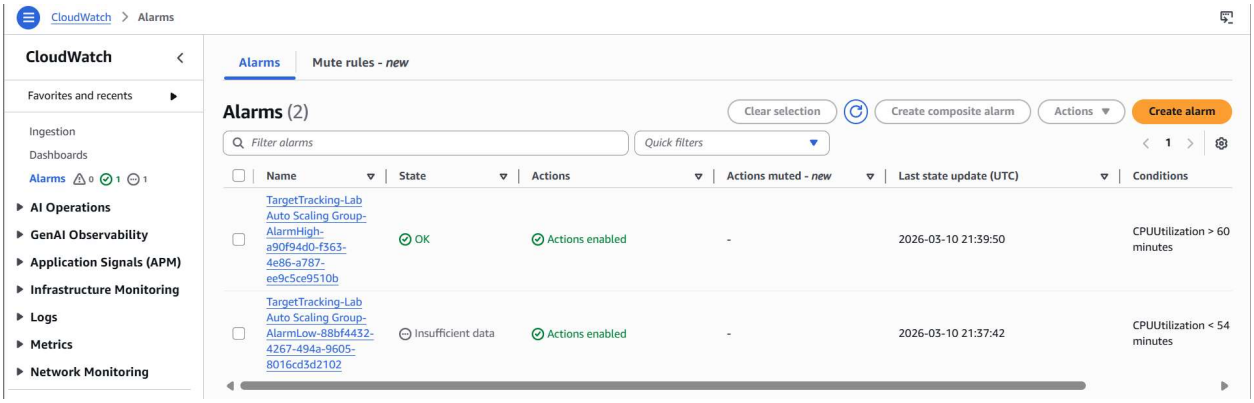
Paste the DNS name into a browser. It will display the CPU Load



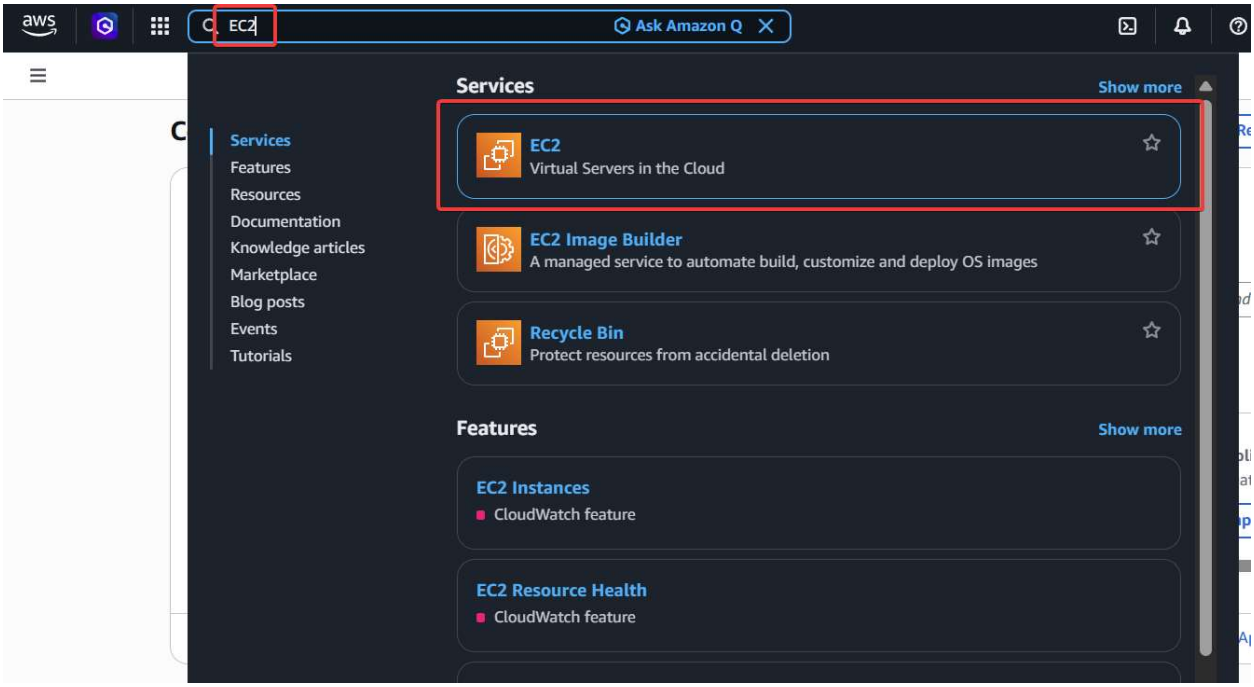
Using the search bar, go to “CloudWatch”



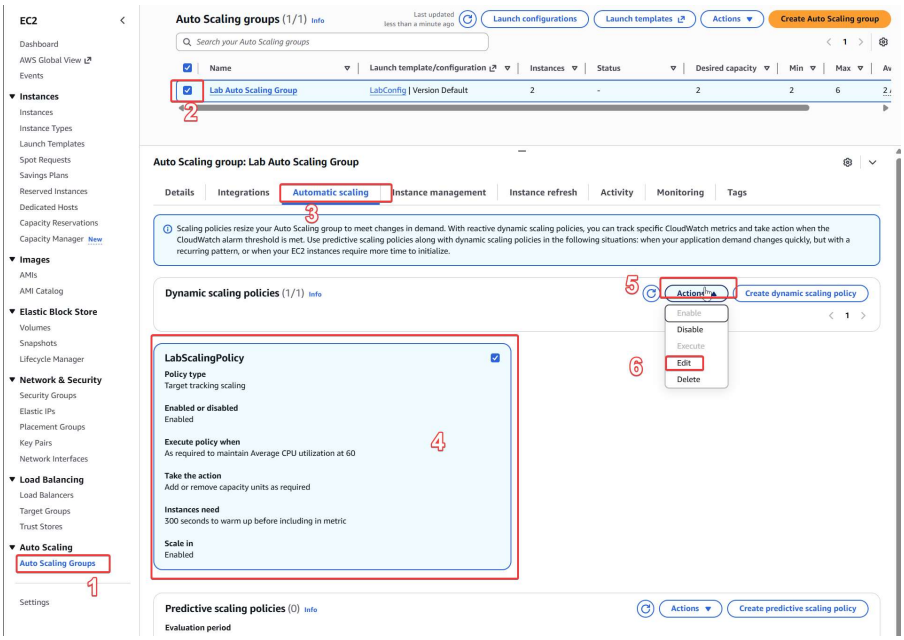
In the Alarms tab, we may find two alarms



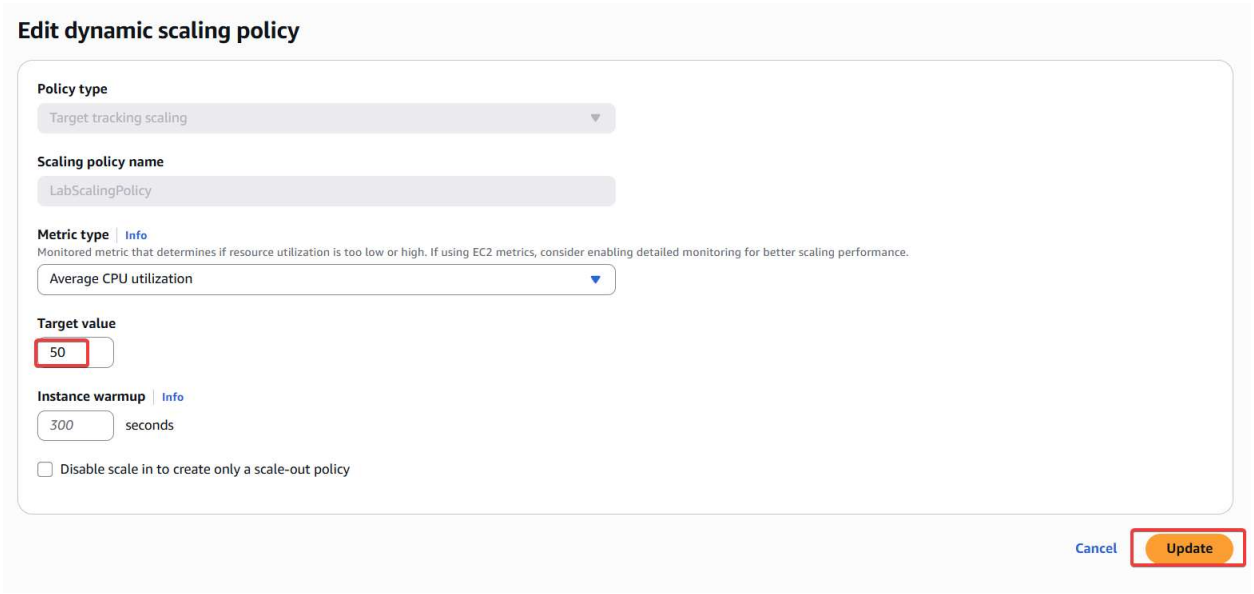
Using the search bar, go to the EC2 Console



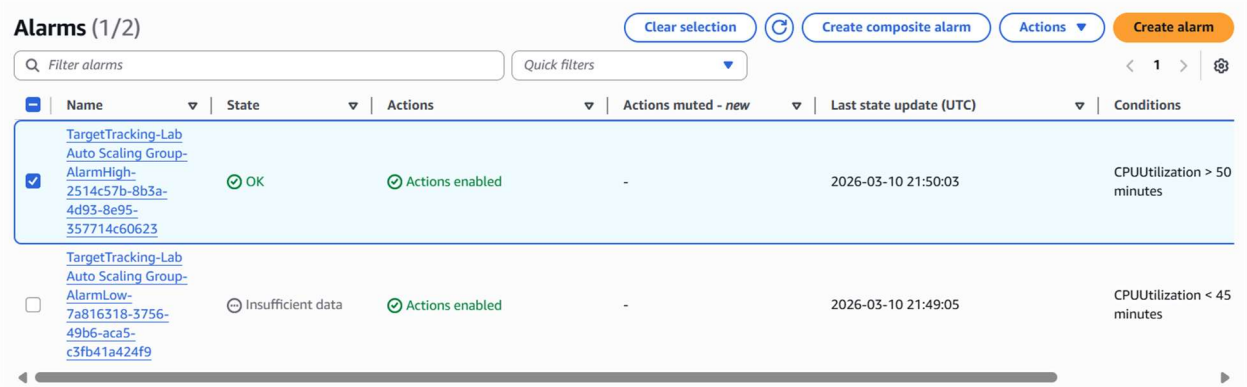
Go to the Auto Scaling Group, check the Lab Auto Scaling Group, and follow the steps below to edit the LabScalingPolicy



Set the Target value to 50, then click “Update”

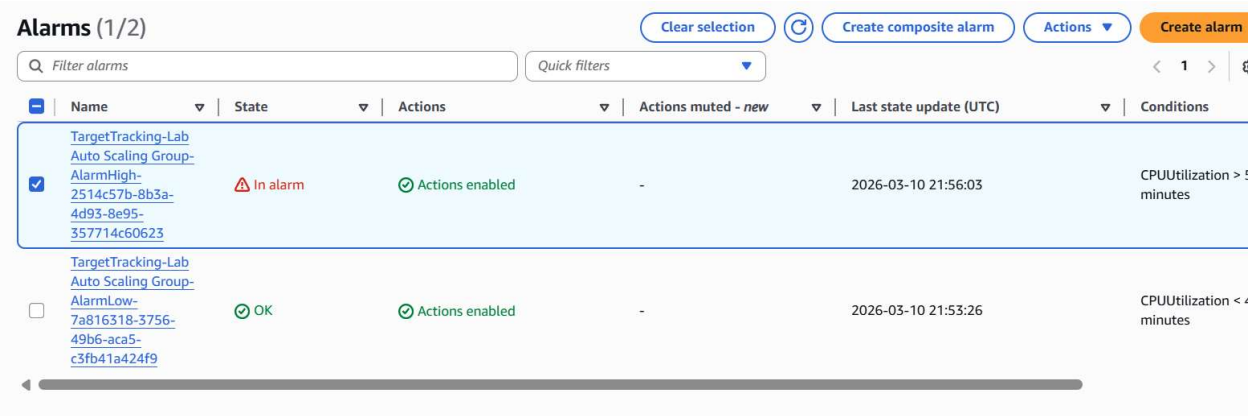


Return to the CloudWatch alarms page and select the alarm displaying “OK”

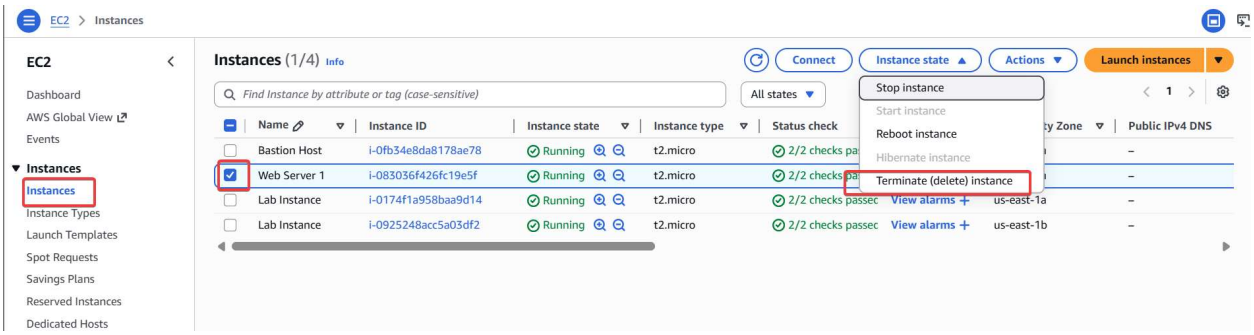


Return to the web browser tab from earlier, then click “Load Test”

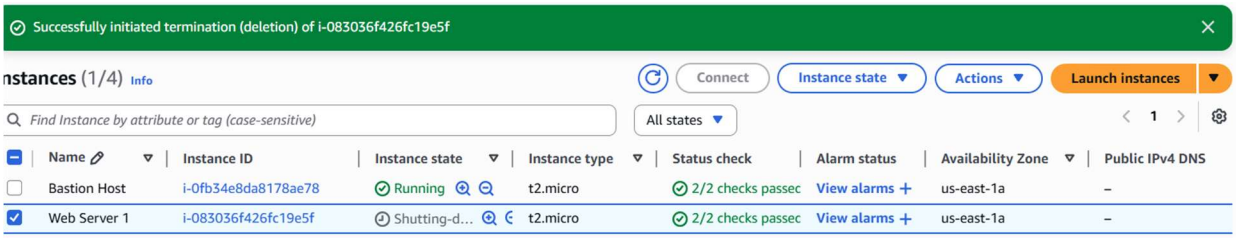
Return to the CloudWatch console tab and wait until the state change to “Alarm High”. We can see that additional instances are getting launched in response



Return to the EC2 tab and go to Instances. Click Web Server 1, Instance state > Terminate



Choose Terminate to terminate the instance



Conclusion

Through completing the two parts of the Cloud Computing AWS mock activities, including 6 labs and 2 activities, we have experienced setting up different cloud computing solutions on AWS through a variety of AWS features. In the future, we are prepared to apply the AWS services appropriately through our training in the videos, then replicate these steps through our experience in the above labs.