Cover Image Attribution: Joe Mabel CC-BY-SA 4.0

AWS Academy Cloud Foundations: Configuring Elastic Block Store, Relational Database Service, and Elastic Cloud Compute Auto Scaling

Colin J. Faletto, CCNA

aws academy

Purpose

This write-up is intended to document and explain the fourth, fifth, and sixth in the AWS Academy Cloud Foundations course. These labs are intended to provide additional knowledge in the field of cloud computing, such as block-level file storage for virtual machines, basic relational database configuration, and automatic infrastructure scaling to meet user demand. These concepts are essential for new cloud engineers to provide parallels to more advanced facets of traditional IT. These labs also provide a strong foundation of knowledge for the AWS management console, as they all take place primarily inside this console.

Background

Amazon is a company based in Seattle that runs the biggest e-commerce platform in the world. They were started in 1994 by former CEO Jeff Bezos, and have grown from a small online bookstore to a giant online store offering a wide variety of products. Amazon also has a strong physical retail presence in the grocery space with their Amazon Fresh and Whole Foods chains, and has a strong online media presence through Twitch, Prime Video, and Amazon Music, which provide entertainment in the form of livestreams, movies and television shows, and music respectively. Amazon also has a popular line of e-readers and tablets with their Kindle brand and has a successful brand of artificial intelligence assistants with their Amazon Alexa A.I. and their Amazon Echo line of smart speakers.

Amazon Web Services, or AWS, is Amazon's cloud computing division. It was created in 2002 to provide simple web services to customers and expanded to cloud storage and computing in 2006. It is the leading cloud service provider and is popular for its pay-as-you-go service model. AWS provides services to everyone from small businesses to massive companies like Coca-Cola and Apple, and even provides web infrastructure to government branches. AWS takes the responsibility and cost of managing a data center out of the hands of businesses and maintains a massive global network of Amazon data centers that split customer traffic among them. AWS currently has 34 geographic regions, each of which have multiple availability zones which themselves contain multiple data centers. These data centers are in undisclosed locations for security reasons, though their general position is published. AWS offers services for virtual machines, cloud storage, database management, machine learning, IoT services, cloud networking, and much more.

Amazon Elastic Compute Cloud, or EC2, is an AWS service that allows customers to create virtual machines in the AWS cloud. These machines are very versatile, as they can be allocated as many or as few resources (CPU, RAM, GPU) as needed and can run nearly any operating system. By default, EC2 instances will run Amazon Linux, which is a version of Linux optimized for AWS servers. Amazon's e-commerce platform, its primary source of revenue, has been running on EC2 instances for over a decade. EC2 instances have a variety of different types, which are optimized for different purposes such as memory (R series, X series), compute (C series), and storage (H series, I series, D series).

Amazon Virtual Private Cloud, or VPC, is an AWS service that provides a virtual network inside the AWS cloud. This service allows AWS objects, such as EC2 instances, to communicate with each other. In a VPC, each EC2 machine is assigned a unique private IPv4 address, which is then connected via NAT to a public address on an internet gateway. Using this gateway, machines in the VPC can communicate with other AWS VPCs and other Internet-connected machines. Amazon VPC is provided at no additional charge to customers using EC2 instances.

Amazon Identity and Access Management, or IAM, is an AWS service that provides a layer of security to customers by limiting the resources different users can access. IAM follows the principle of least privilege, meaning that by default, all AWS controls are blocked for users unless they have been explicitly granted permissions. IAM represents a portion of the customer responsibilities in the AWS shared responsibility model, which is a model outlining that AWS is responsible for the physical security of data centers and networks while the customer is responsible for keeping their customer data and configurations safe. One of IAM's unique features is its role feature, which creates identities with elevated permissions that can be temporarily assigned to users. This feature works similarly to the "sudo" command in unix-based operating systems.

Amazon Elastic Block Store, or EBS, is an AWS service that allows virtual block-level storage to be attached to an EC2 instance. These drives can be formatted with any number of different file systems, such as ext3, ext4 (used with Linux), NTFS (used with Windows), and APFS (used with macOS). Depending on the partitioning scheme, EBS drives can be up to 16 tebibytes in size. EBS Volumes come as either solid-state or hard disk storage, with SSDs coming in high-performance IOPS and general-purpose GP variants, and HDDs coming in throughput-optimized (ST) and cold low-cost (SC) variants. EBS volumes are set up in a way that ensures redundancy, meaning that component failure in AWS data centers doesn't result in data loss for the customer.

A relational database is a type of database that stores information in terms of keys and values. A value is a specific data point that can be accessed by referencing its corresponding key. Relational databases can be divided further into any number of different tables. Amazon Relational Database Service, or RDS, is an AWS service that allows relational databases to be created and managed in the AWS cloud. RDS supports several database types, such as MySQL, Oracle Database, MariaDB, PostgreSQL, and Microsoft SQL Server. Amazon also offers their own database type, Aurora, which is compatible with MySQL and optimized for performance and availability. RDS databases can also be deployed across multiple availability zones, which can optionally create multiple instances of an RDS database in different locations.

AWS Elastic Load balancing, or ELB, is an AWS service that automatically balances traffic across multiple devices in the AWS cloud. It is often used in conjunction with EC2 instances. This service creates more consistency in how AWS resources are distributed, which ensures a more stable experience for end users.

AWS Auto Scaling is an AWS service that allows applications to automatically be created or deleted based on a variety of targets, such as CPU utilization or bandwidth. This service facilitates scalability of cloud computing services and adapts AWS

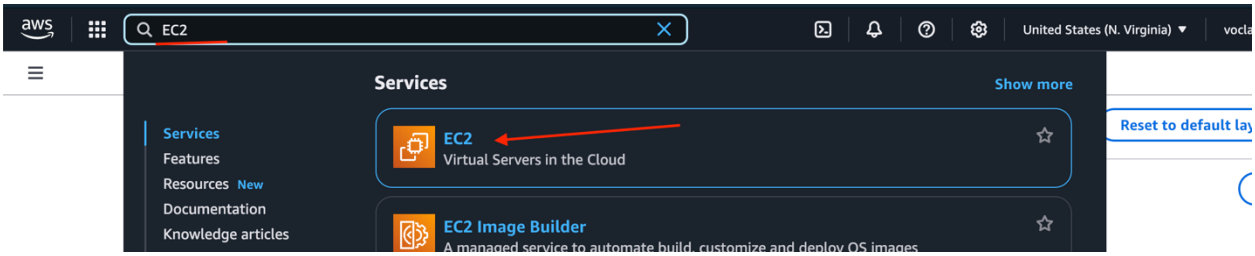
resources to better fit the needs of customers at any given time. This service is also economical for AWS users as it allows them to pay only for resources required for any given moment in time. Auto Scaling can automatically create and delete resources, EC2 instances, DynamoDB databases, and Aurora RDS databases.

Lab Summary

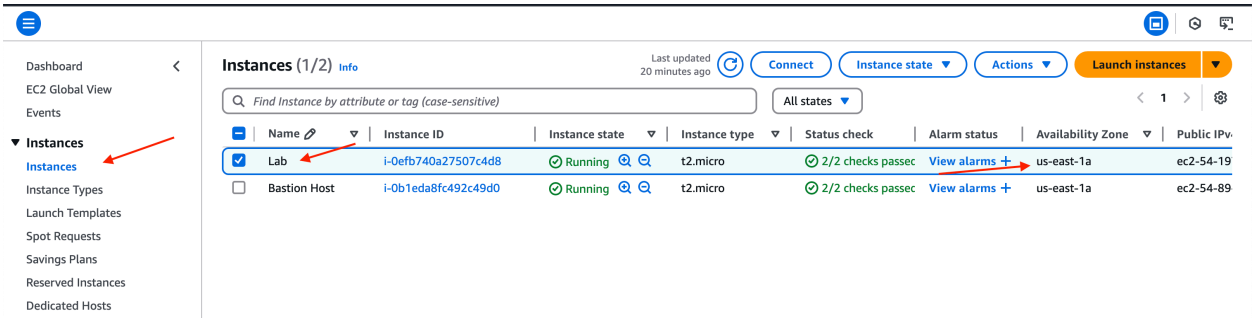
This write-up covers three different AWS labs. The first lab involves creating an EBS volume, attaching it to an EC2 instance, instantiating a new file system on the volume, then creating and restoring a snapshot of the volume. The second lab involves building and launching an RDS address book database, connecting it to a pre-existing web server, then manipulating the database with a web server GUI. The third lab involves creating a load balancer and auto scaling group based on an EC2 instance AMI, then testing the auto scaling feature by simulating excess CPU usage.

Lab Commands (EBS)

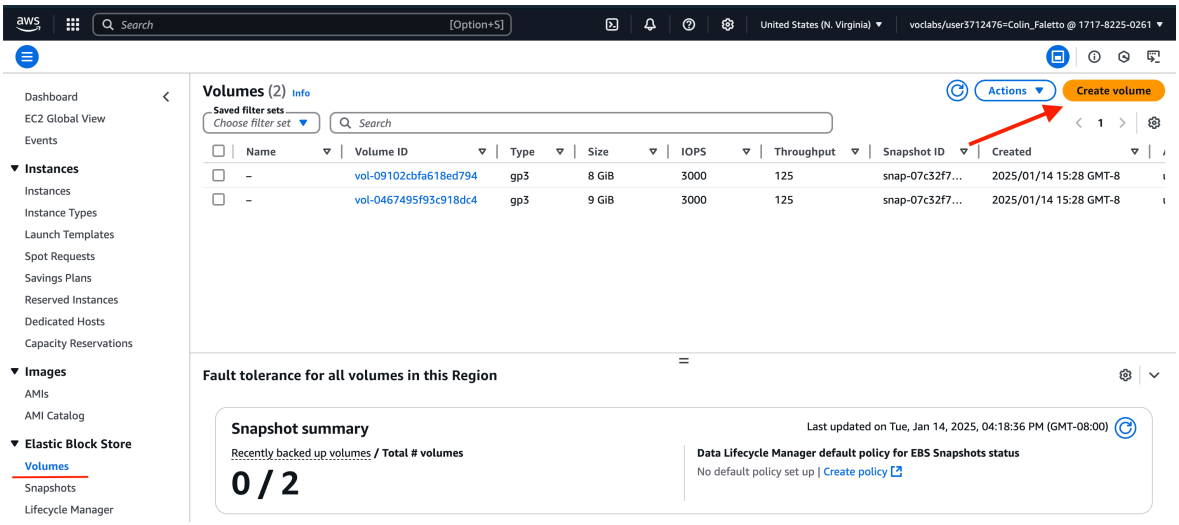
From the AWS search bar, type in and launch “EC2”.



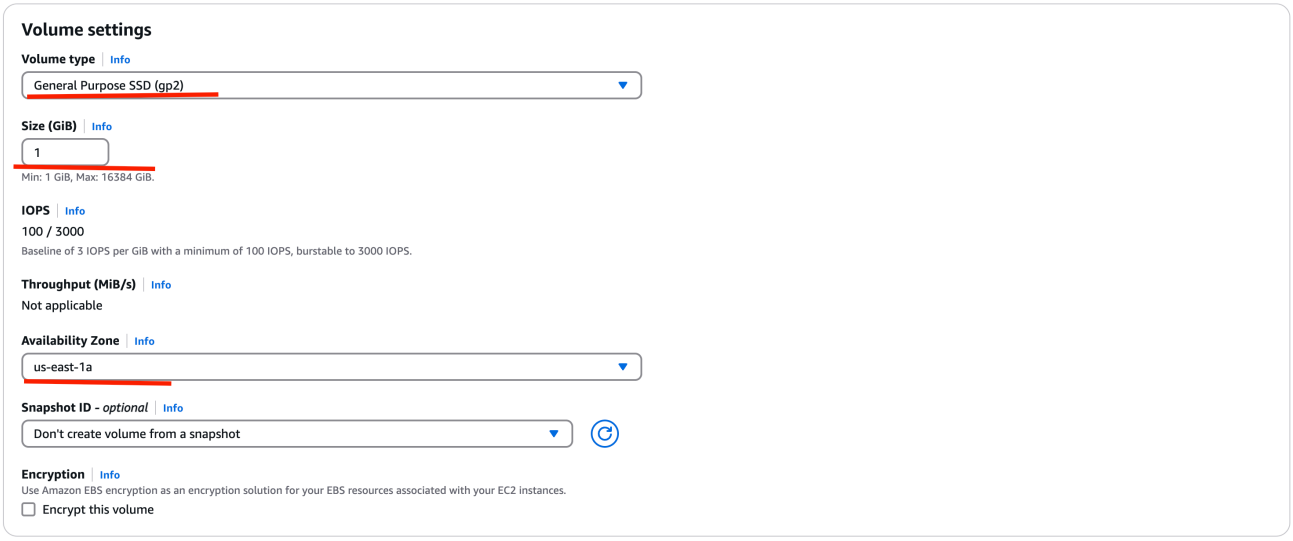
From the Instances tab, make sure that the EC2 instance named “Lab” has been created. Make sure to write down the EC2 instance’s availability zone, which is “us-east-1a” in this case.



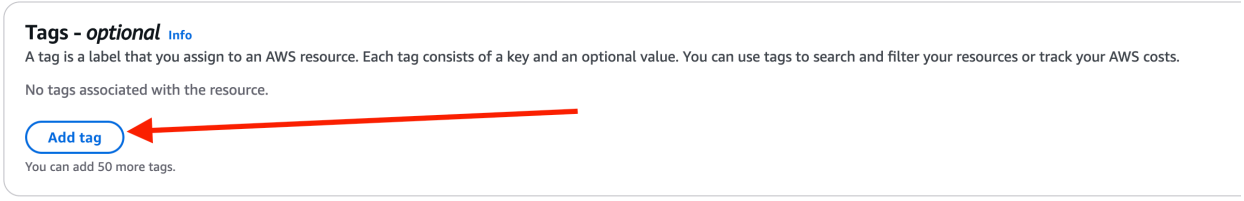
Go to the “Volumes” section and click “Create Volume”.



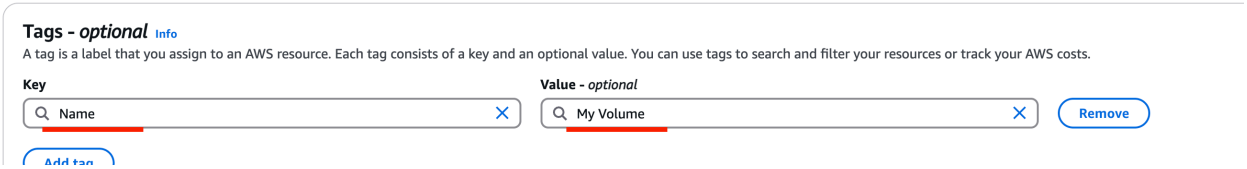
Set the volume type to “gp2”, the size to 1 gigabyte, and the availability zone to the same zone as the Lab instance.



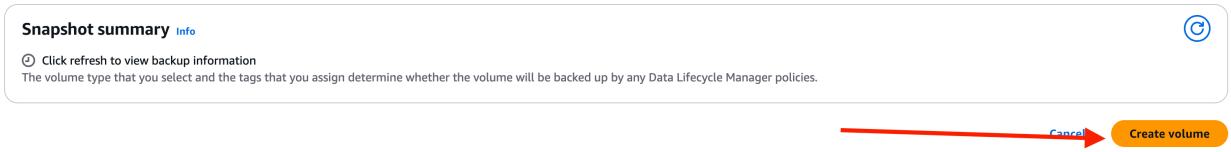
Under “Tags”, click “Add Tag”.



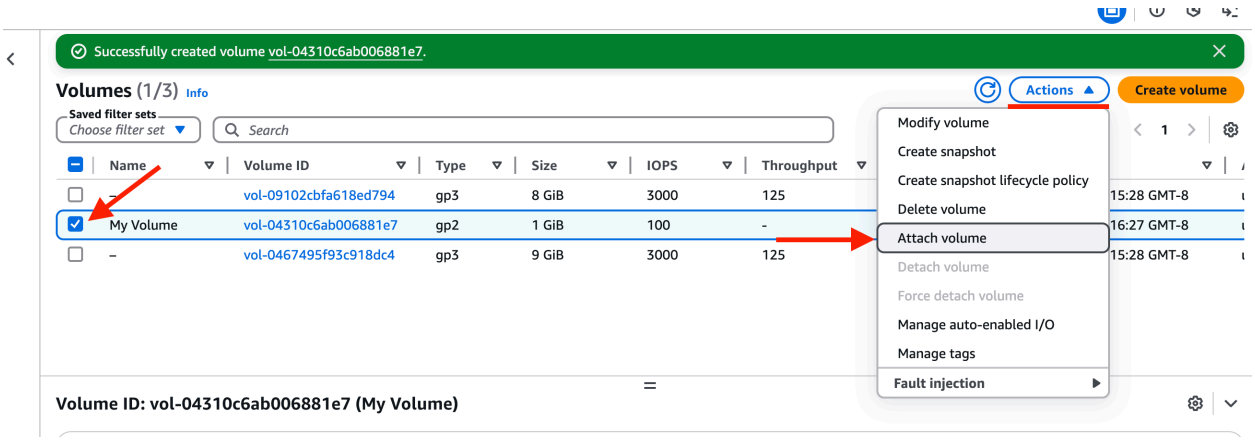
Set the key to “Name” and the value to “My Volume”.



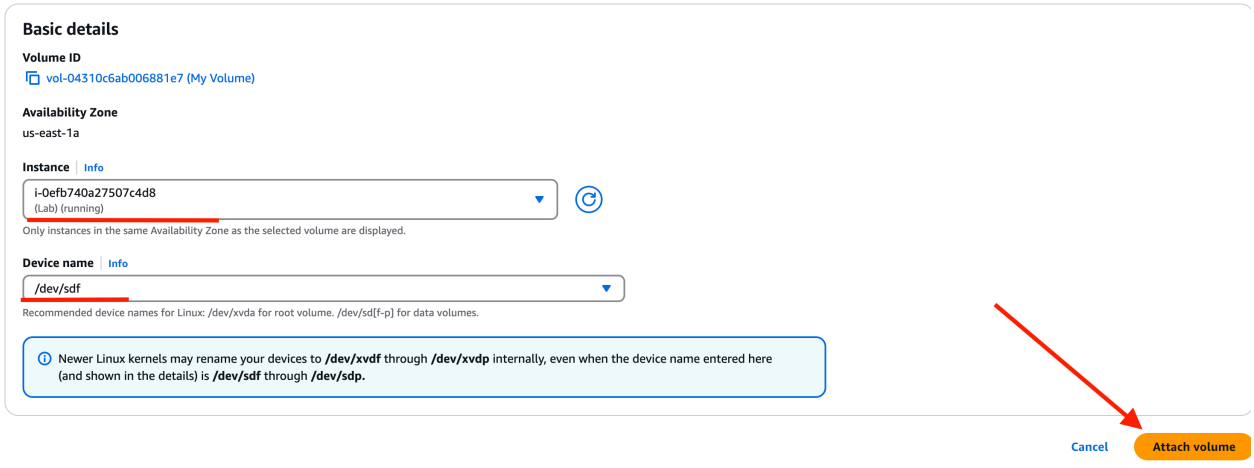
Click “Create Volume”.



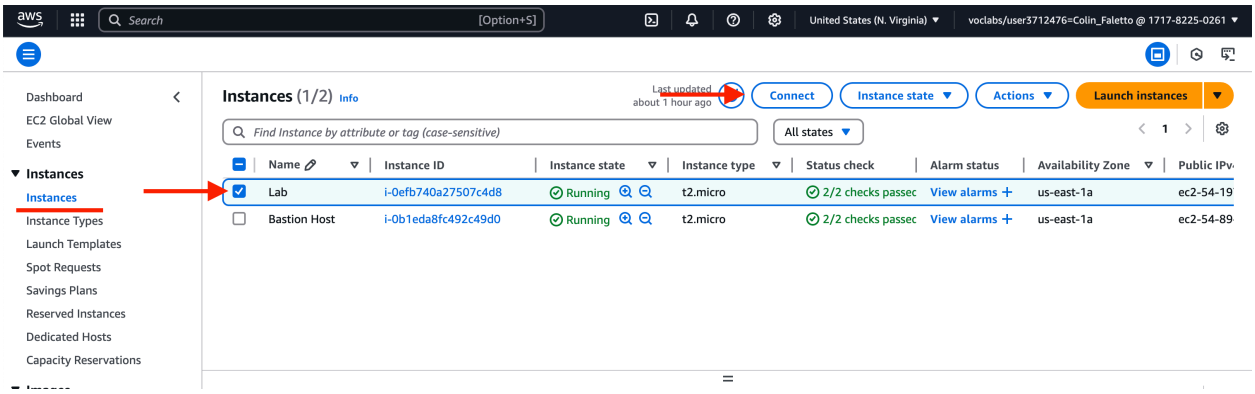
You should now be redirected to the Volumes page. Click the checkbox next to My Volume, then under Actions, click Attach Volume.



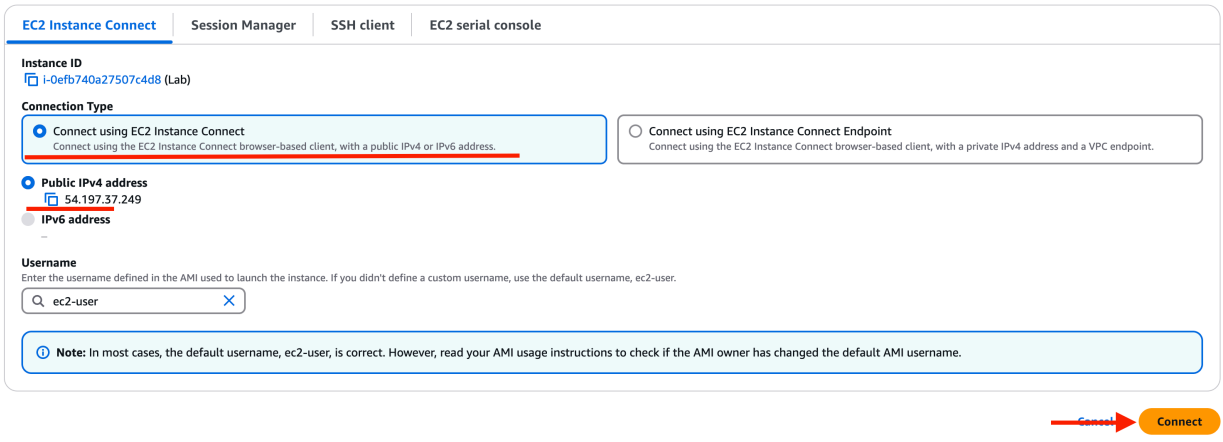
Set the instance to the Lab instance and the device name to /dev/sdf, then click Attach volume.



Go to the “Instances” section, click the checkbox next to “Lab”, and click “Connect”.



Select “EC2 instance connect”, select “Public IPv4 address”, then click “Connect”.



Run the command `df -h` to list the mounted volumes; you should see output like this:

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-1-11-198 ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        4.0M     0  4.0M   0% /dev
tmpfs           475M     0  475M   0% /dev/shm
tmpfs           190M  452K  190M   1% /run
/dev/xvda1      8.0G  1.6G  6.4G  20% /
tmpfs           475M     0  475M   0% /tmp
/dev/xvda128     10M  1.3M  8.7M  13% /boot/efi
tmpfs            95M     0   95M   0% /run/user/1000
```

Run the command `sudo mkfs -t ext3 /dev/sdf` to create an ext3 file system on the volume you created.

```
[ec2-user@ip-10-1-11-198 ~]$ sudo mkfs -t ext3 /dev/sdf
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 262144 4k blocks and 65536 inodes
Filesystem UUID: d545580a-928e-4063-ad5d-bda7ee246ba3
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
```

Run the command `sudo mkdir /mnt/data-store` to create a new folder at /mnt/data-store.

```
[ec2-user@ip-10-1-11-198 ~]$ sudo mkdir /mnt/data-store
```

Run the command `sudo mount /dev/sdf /mnt/data-store` to mount your volume at the created folder.

```
[ec2-user@ip-10-1-11-198 ~]$ sudo mount /dev/sdf /mnt/data-store
```

Run the command `echo "/dev/sdf  /mnt/data-store ext3 defaults,noatime 1 2" | sudo tee -a /etc/fstab` to add this mount binding to /etc/fstab, a file that automatically mounts volumes at startup.

```
[ec2-user@ip-10-1-11-198 ~]$ echo "/dev/sdf  /mnt/data-store ext3 defaults,noatime 1 2" | sudo tee -a /etc/fstab
```

Run the command `cat /etc/fstab` to confirm that the line was added.

```
[ec2-user@ip-10-1-11-198 ~]$ cat /etc/fstab
#
UUID=73e034f4-2887-4ec9-8b40-0d35c0091a37     /           xfs    defaults,noatime  1   1
UUID=9F37-3C35      /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
```

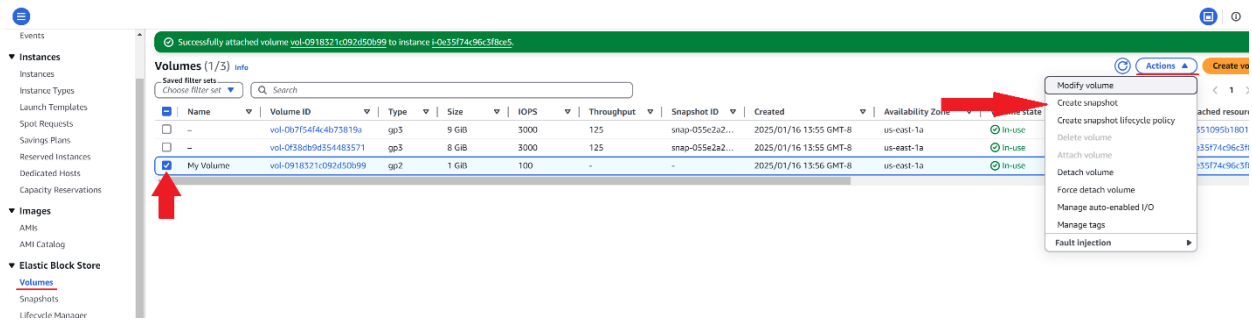
Run the command `sudo sh -c "echo some text has been written > /mnt/data-store/file.txt"` to create a file with the text “some text has been written” at the specified location.

```
[ec2-user@ip-10-1-11-50 ~]$ sudo sh -c "echo some text has been written > /mnt/data-store/file.txt"
```

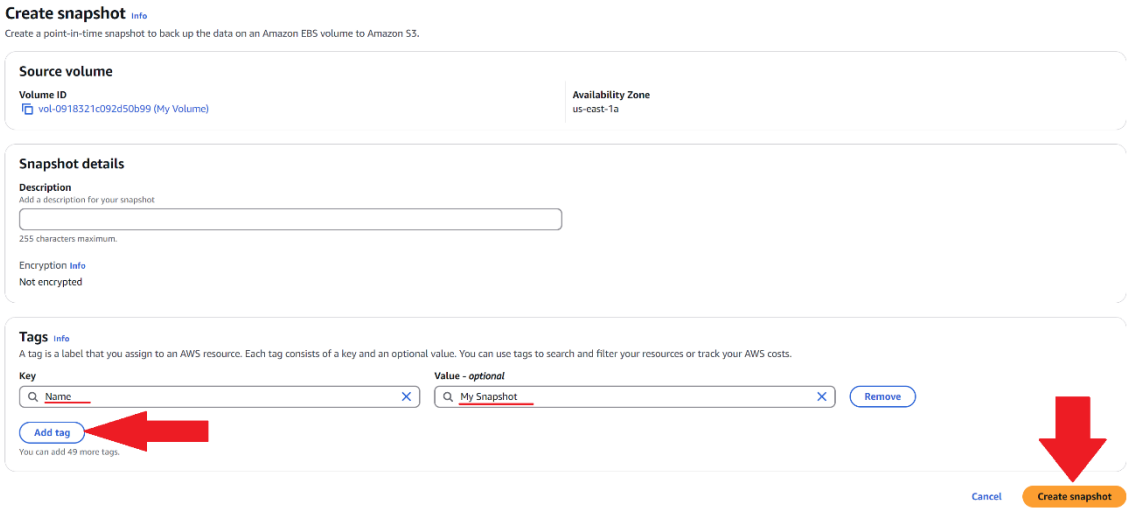
Run the command `cat /mnt/data-store/file.txt` to confirm that the file was written.

```
[ec2-user@ip-10-1-11-50 ~]$ cat /mnt/data-store/file.txt
some text has been written
```

Go back to the main AWS console tab. Under Volumes, check My Volume, then click Actions > Create Snapshot.



Under Tags, click Add Tag, set the key to Name, set the value to My Snapshot, then click Create Snapshot.



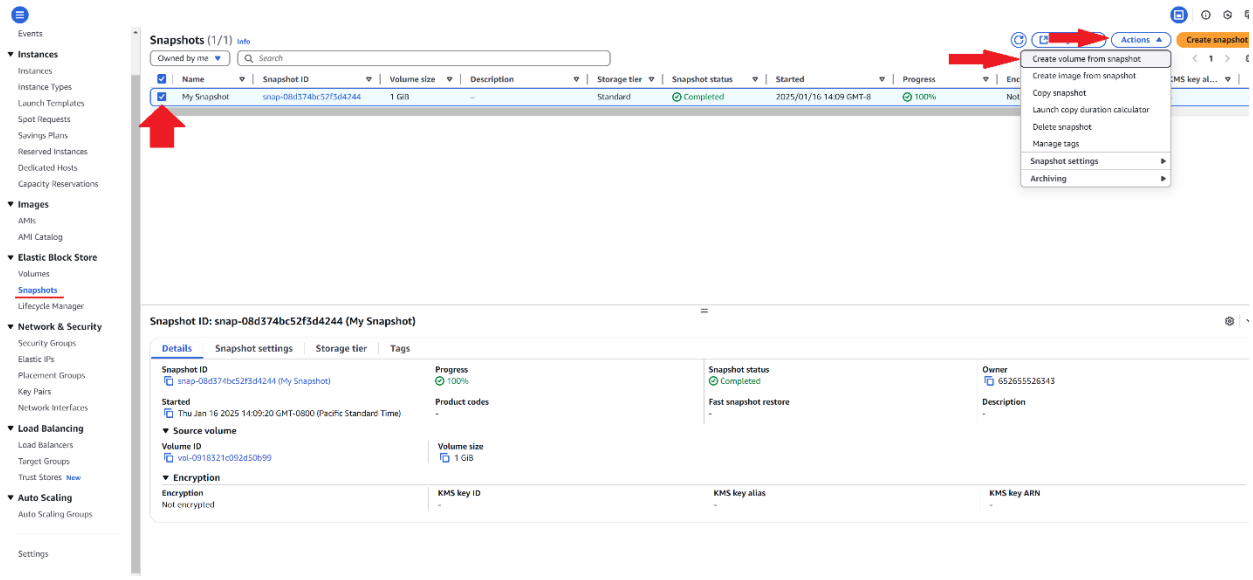
Switch back to the console and run the command `sudo rm /mnt/data-store/file.txt` to remove the file you created earlier.

```
[ec2-user@ip-10-1-11-50 ~]$ sudo rm /mnt/data-store/file.txt
```

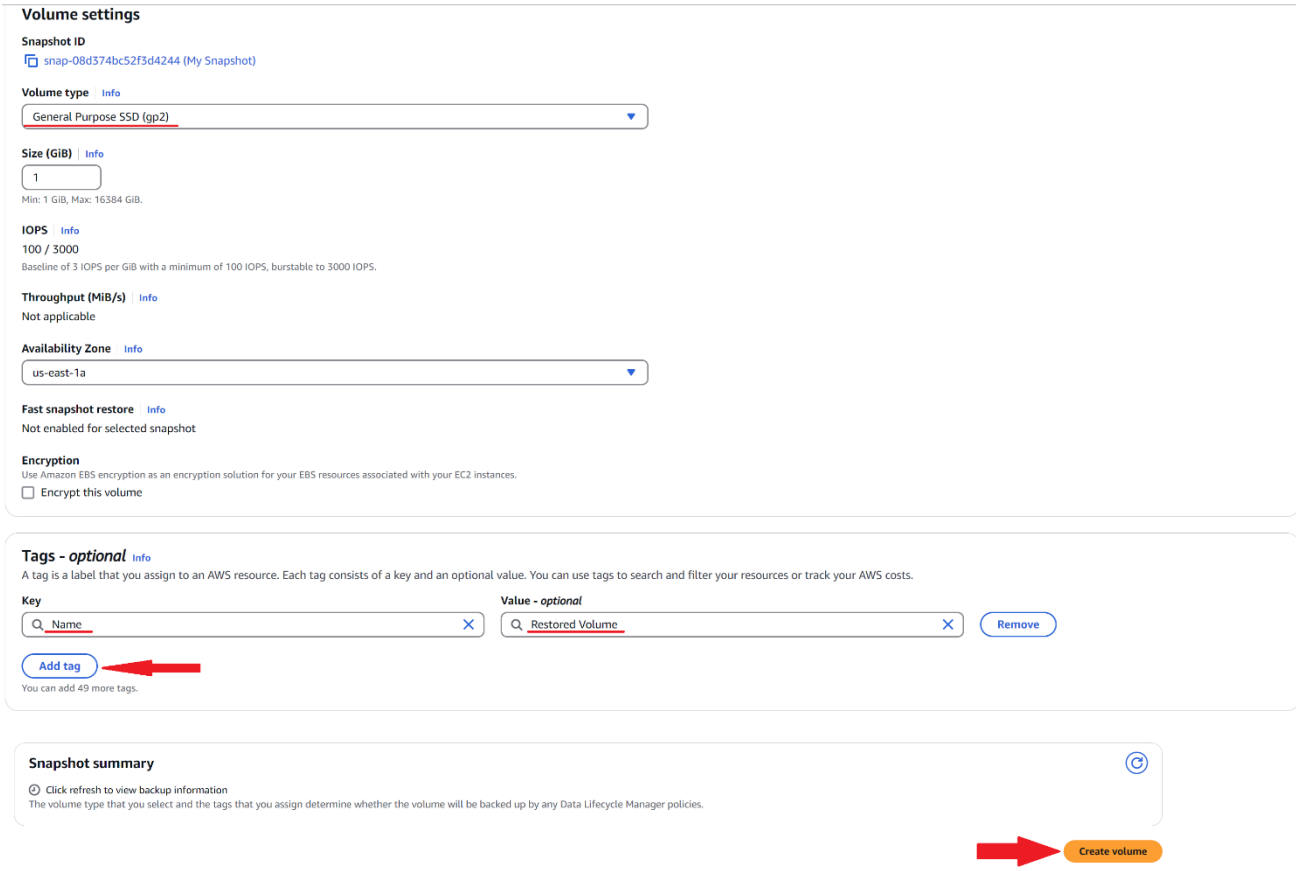
Run the command `ls /mnt/data-store` to confirm that the file has been deleted.

```
[ec2-user@ip-10-1-11-50 ~]$ ls /mnt/data-store
lost+found      no mention of file.txt
```

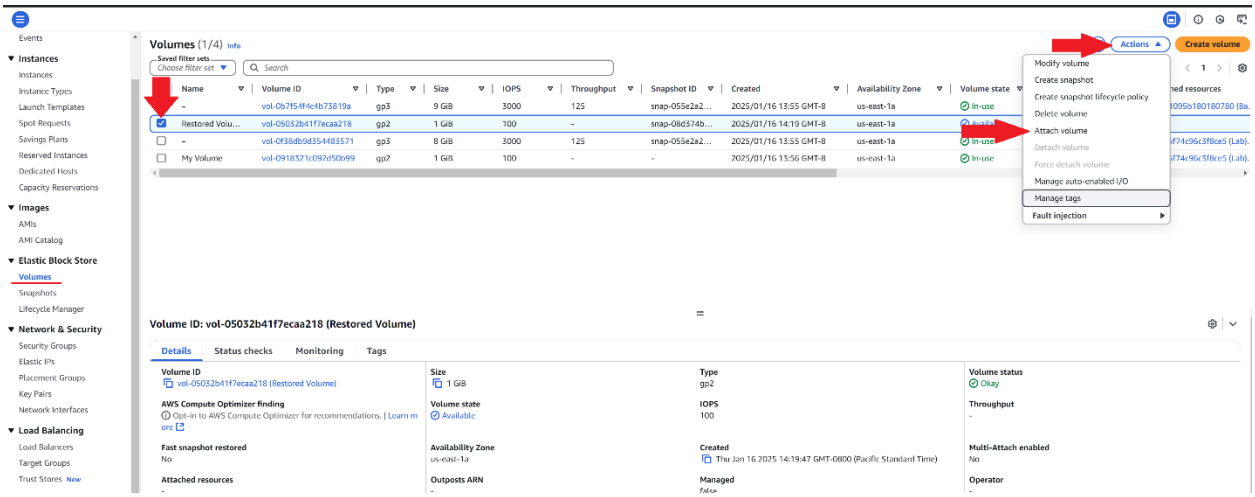
Back in the AWS console, go to Snapshots > Check My Snapshot > Actions > Create volume from snapshot.



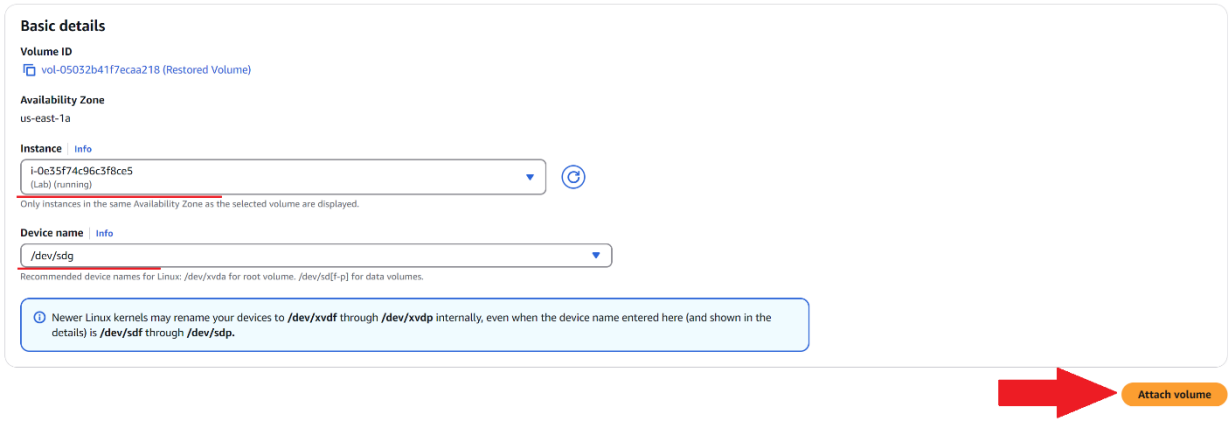
Set the volume type to gp2 (everything else should be correct by default), then click Add Tag, set the key to Name, set the value to Restored Volume, then click Create Volume.



Under Volumes, check Restored Volume, then click Actions > Attach Volume



Set the instance to Lab, the device name to /dev/sdg, then click Attach Volume.



Run the command `sudo mkdir /mnt/data-store2` to make a new folder for the restored drive.

```
[ec2-user@ip-10-1-11-50 ~]$ sudo mkdir /mnt/data-store2
```

Run the command `sudo mount /dev/sdg /mnt/data-store2` to mount the restored volume at the specified point.

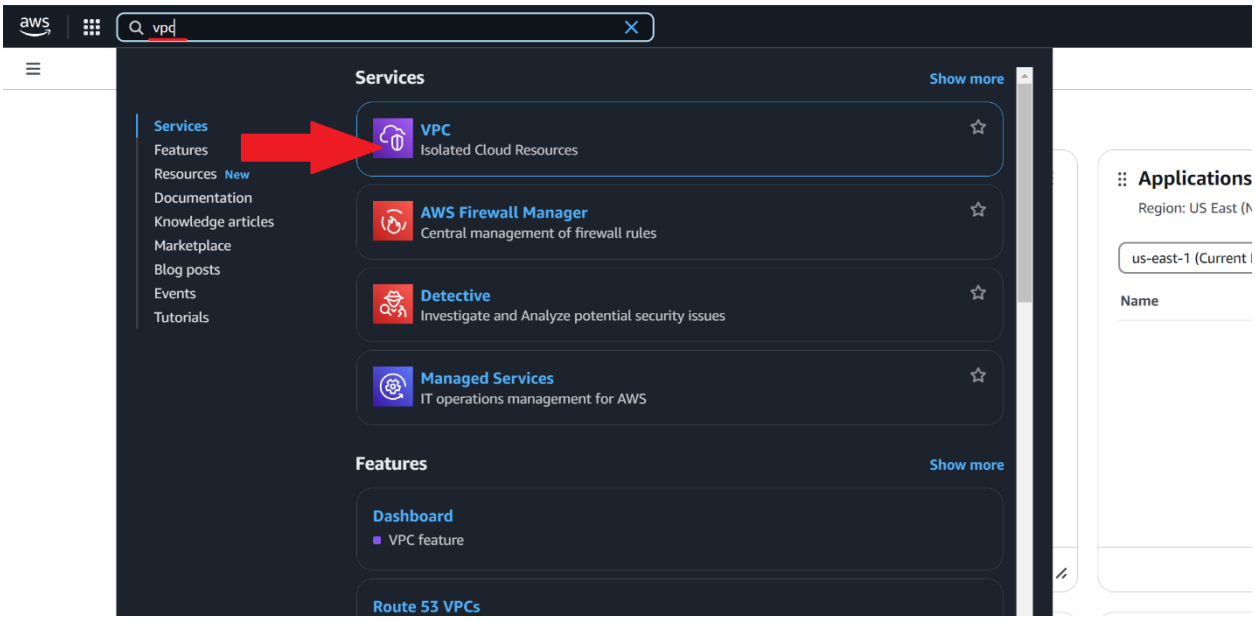
```
[ec2-user@ip-10-1-11-50 ~]$ sudo mount /dev/sdg /mnt/data-store2
```

Run the command `ls /mnt/data-store2/` and ensure that file.txt shows up.

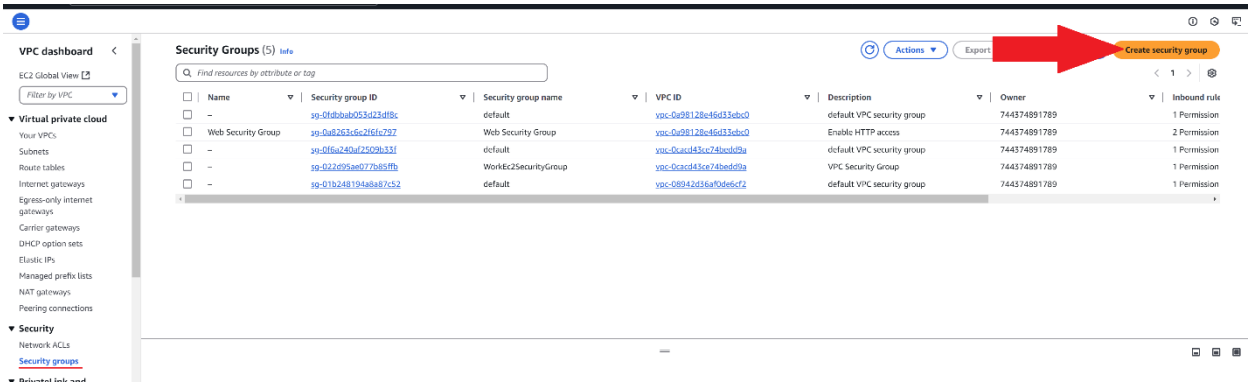
```
[ec2-user@ip-10-1-11-50 ~]$ ls /mnt/data-store2/
file.txt  lost+found
```

Lab Commands (RDS)

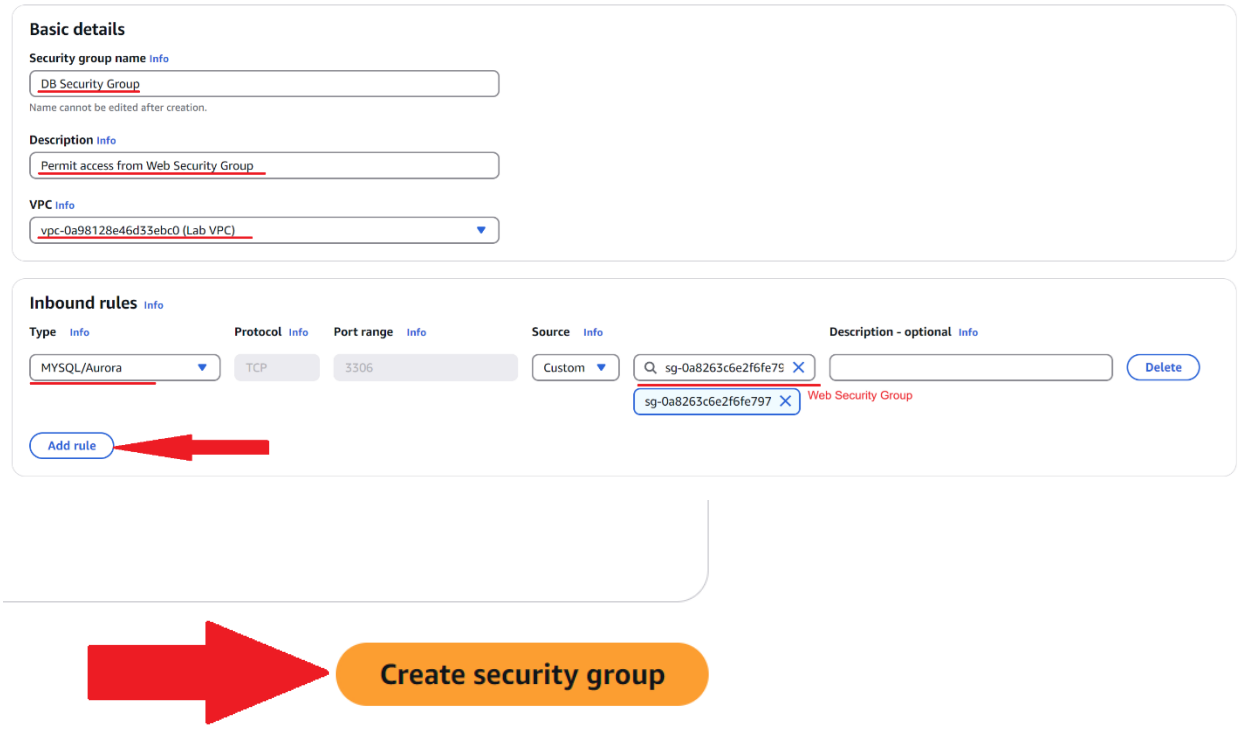
From the AWS console search bar, open VPC.



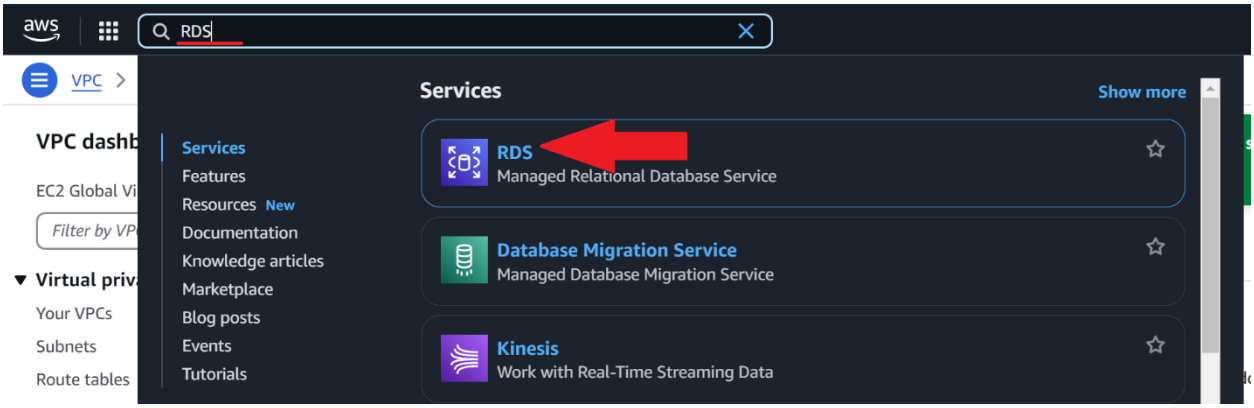
Under Security Groups, click Create Security Group.



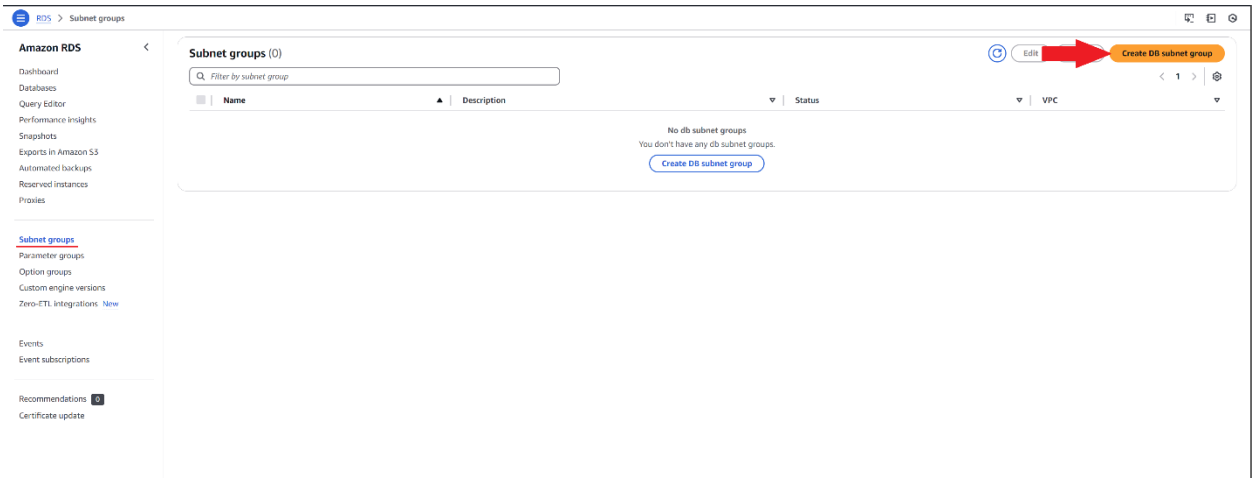
Set the name to DB Security Group, the description to Permit access from Web Security Group, and the VPC to Lab VPC. Under Inbound Rules, click Add Rule, set the type to MySQL/Aurora, and set the source to the Web Security Group. Click Create security group.



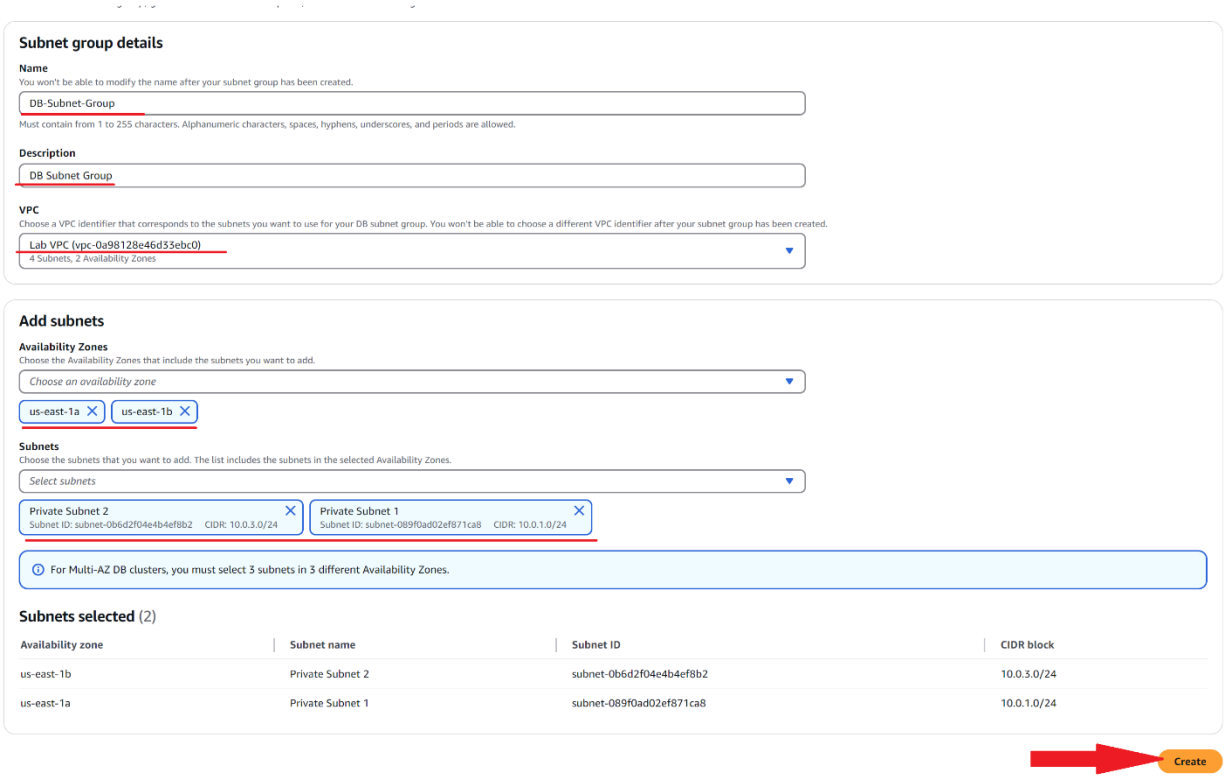
Go to RDS from the search bar.



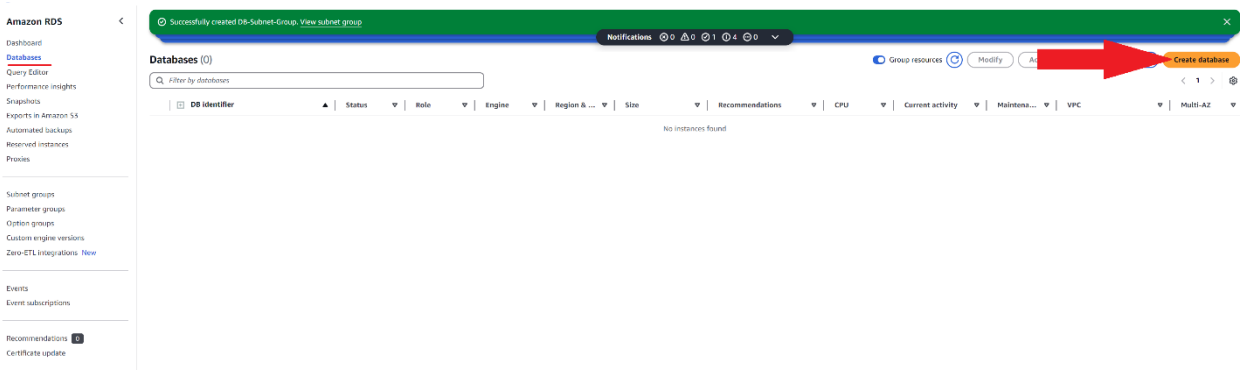
Go to Subnet groups > Create DB Subnet Group.



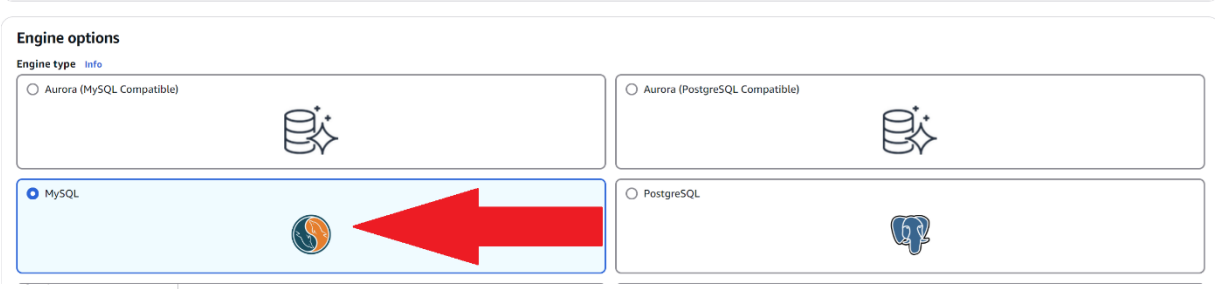
Set the name to DB-Subnet-Group, the description to DB Subnet Group, and the VPC to Lab VPC. Set the availability zones to us-east-1a and us-east-1b, and set the subnets to Private Subnet 1 and Private Subnet 2. Click Create.



Go to Databases > Create Database.

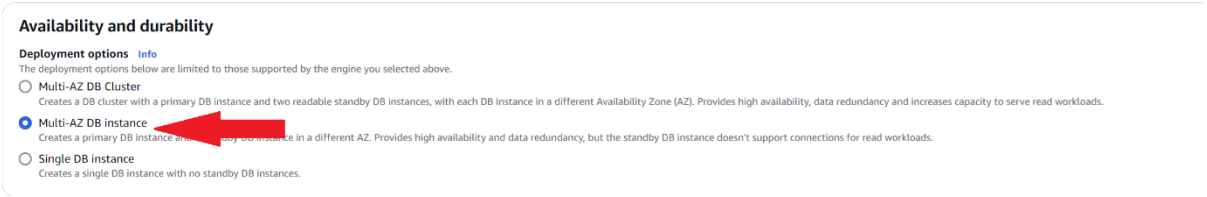


Set the engine type to MySQL.

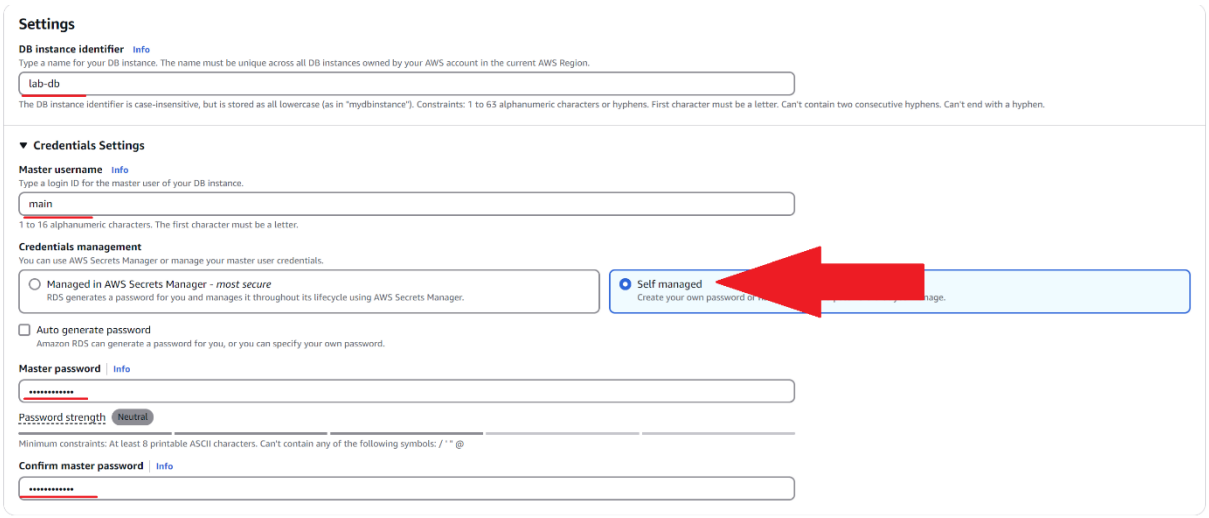


Set the template to Dev/Test.

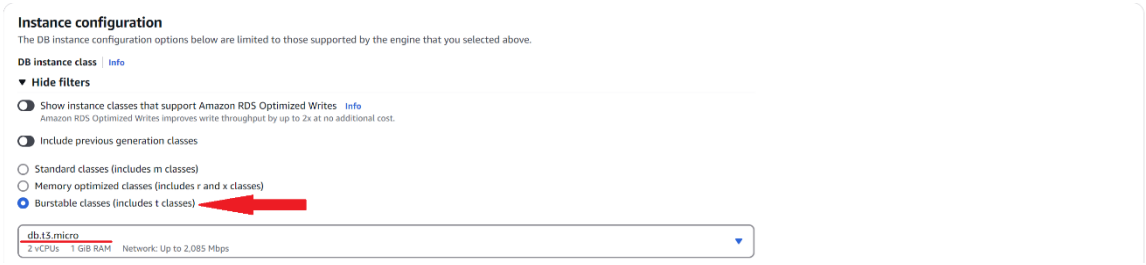
Set the deployment options to Multi-AZ DB instance.



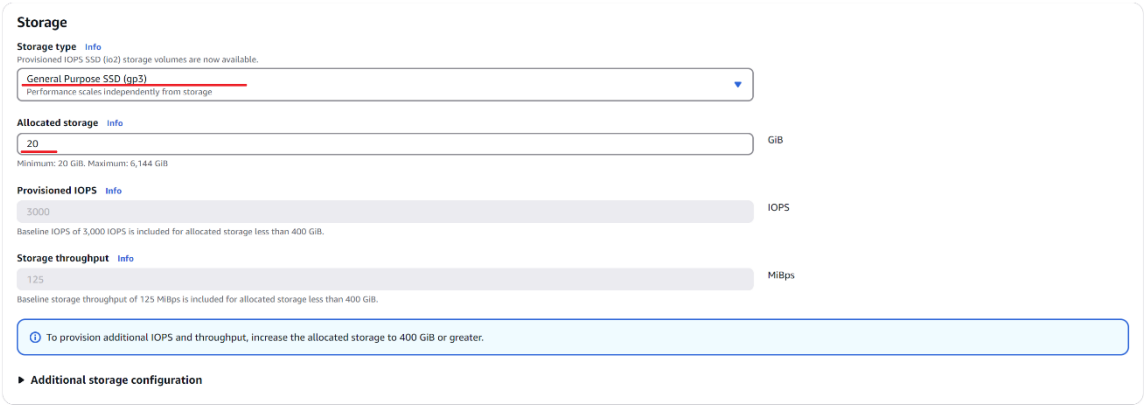
Set the DB instance identifier to lab-db, the master username to main, the credentials management to self-managed, and the master password to lab-password.



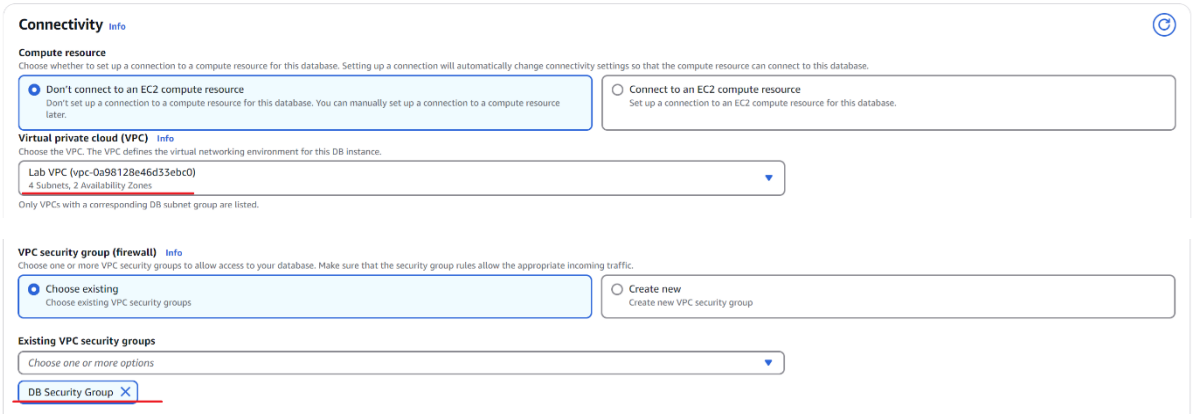
Under Instance configuration, set the instance class to Burstable classes, and select db.t3.micro.



Under storage, set the storage type to General Purpose SSD and the allocated storage to 20 GiB.

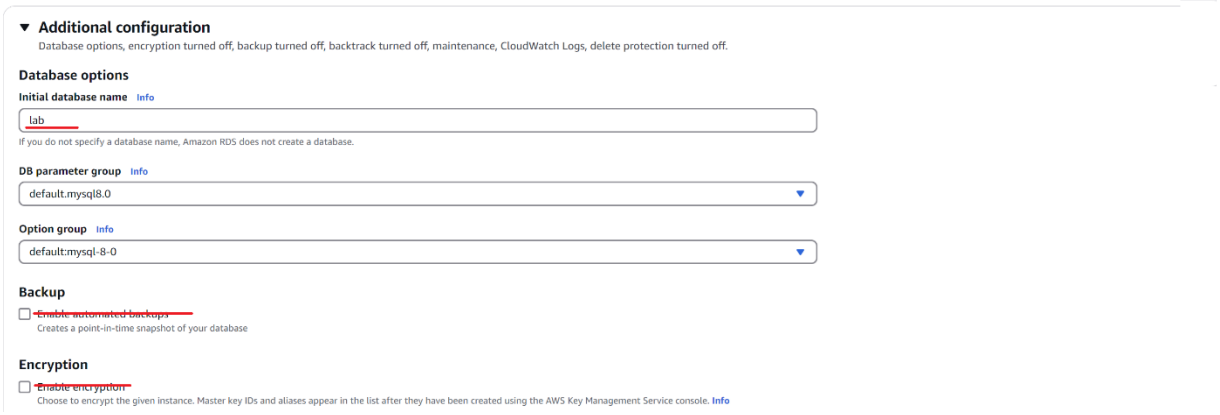


Under connectivity, set the VPC to Lab VPC and set the existing security groups to DB security group.

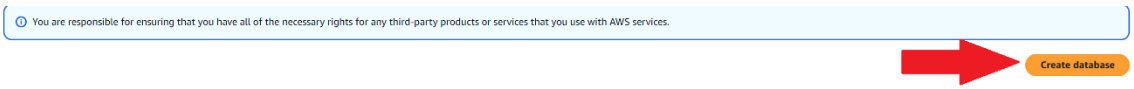


Under monitoring, disable Enhanced monitoring.

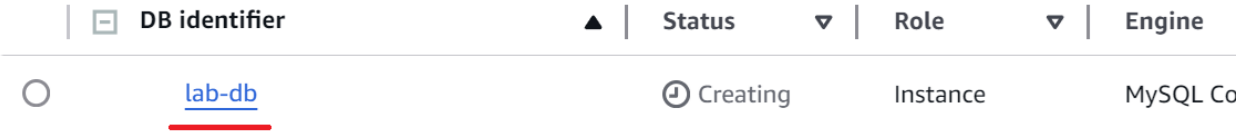
Under additional configuration, set the initial database name to lab, and disable automated backups and encryption.



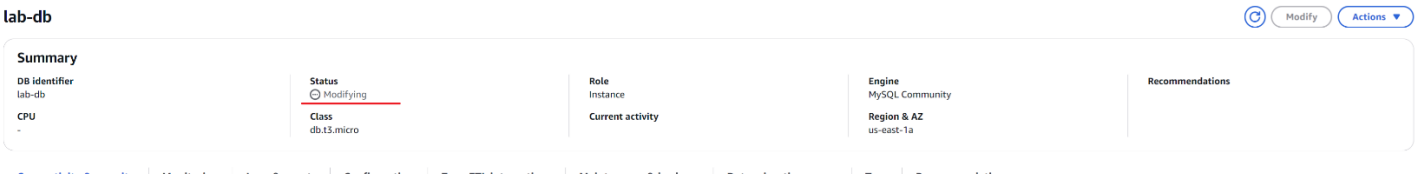
Click create database.



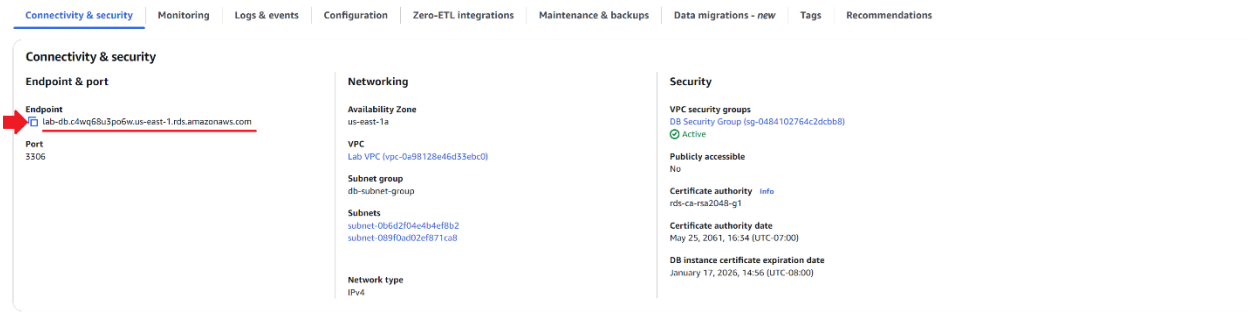
Click on lab-db.



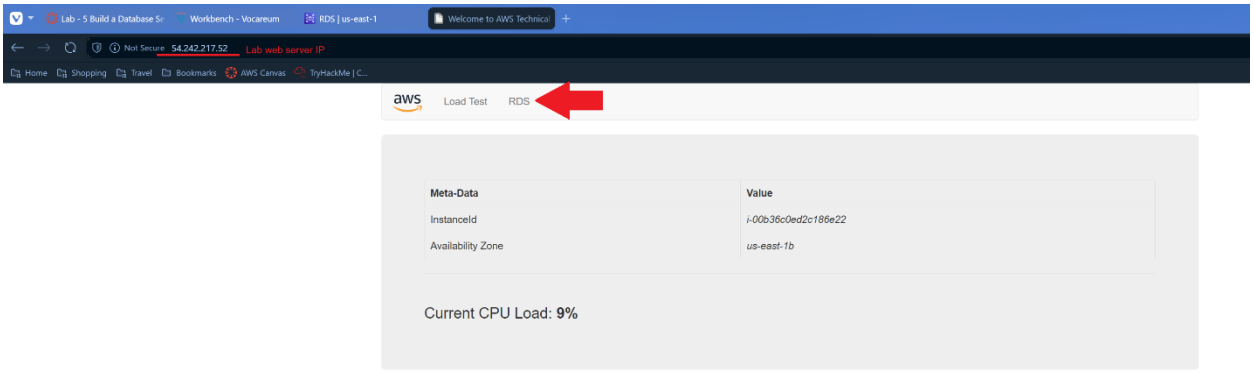
Wait until the status says Modifying or Available



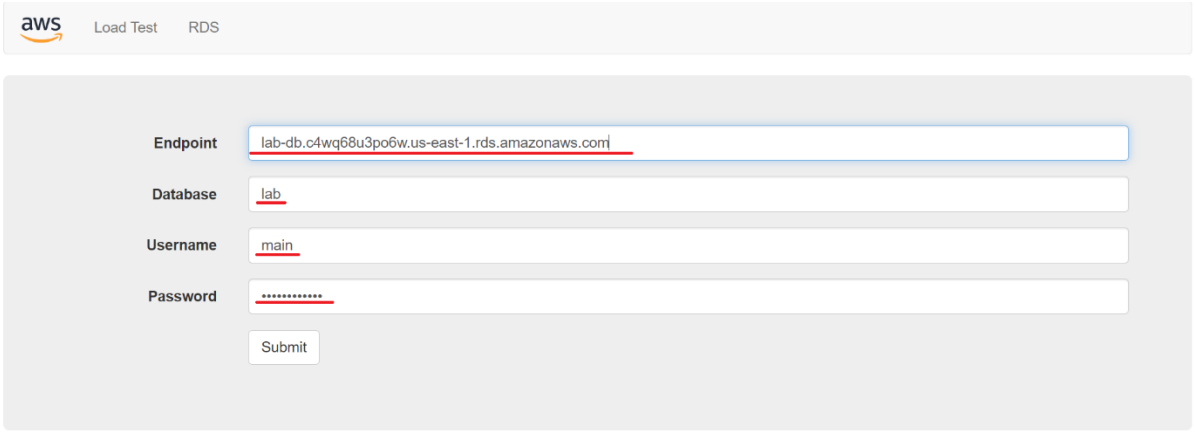
Scroll down to the Connectivity and security seciton and copy the Endpoint value.



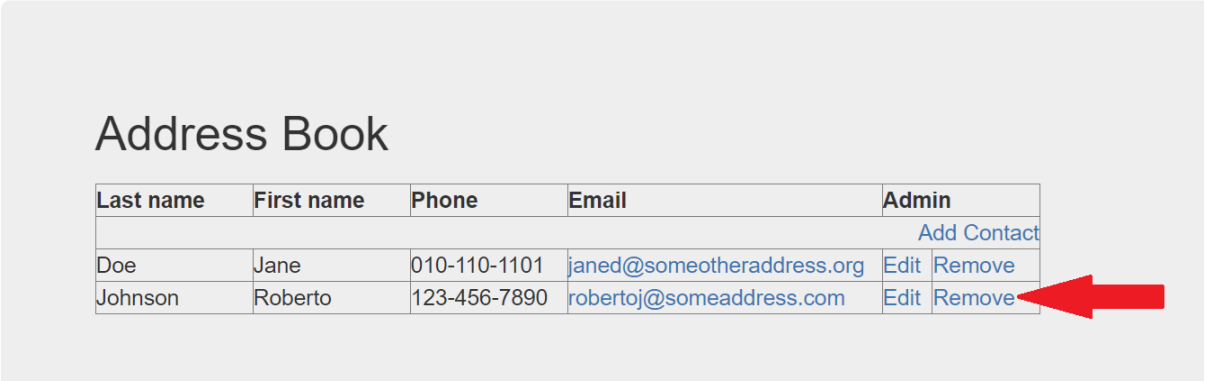
Connect to the Lab's web server IP, and click on RDS.



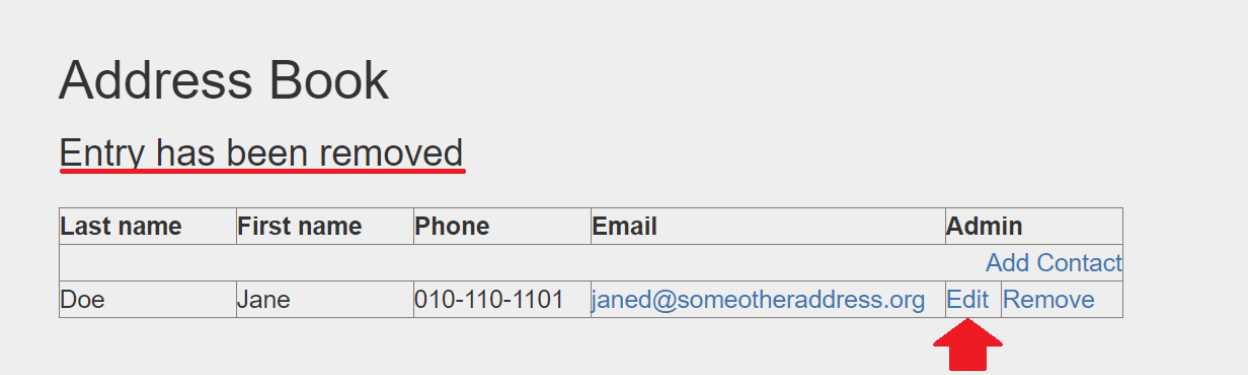
Paste the endpoint URL, and enter the database name, username, and password from earlier.



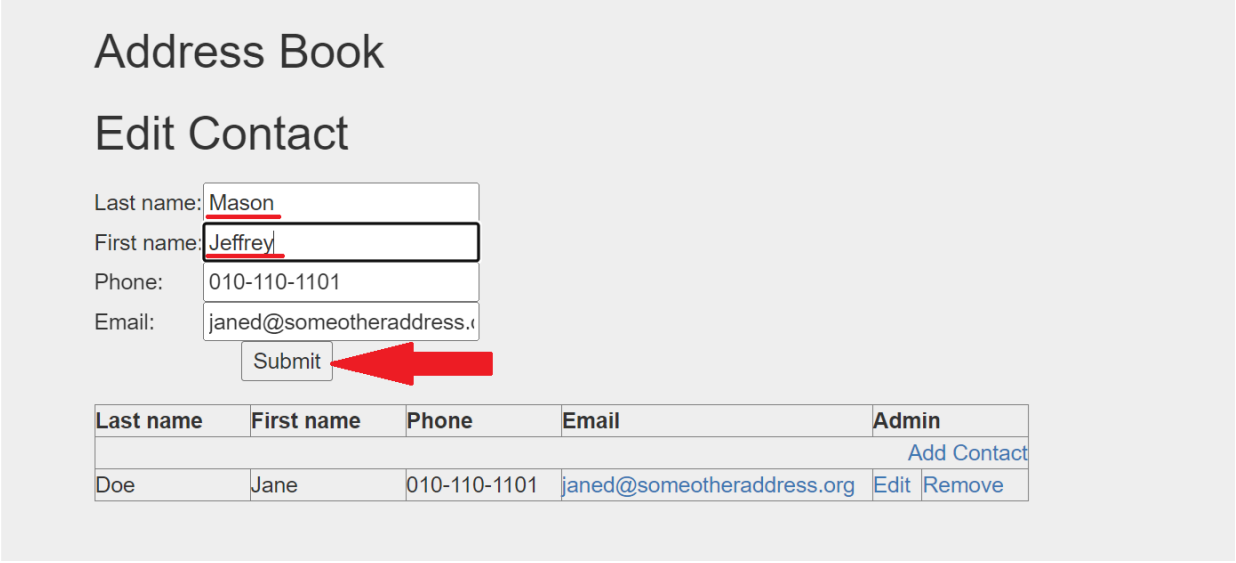
If done correctly, you should see an address book database. Click "Remove" to remove an entry.



You should see a message saying that the entry has been removed. Next, click the "Edit" button next to the remaining entry.

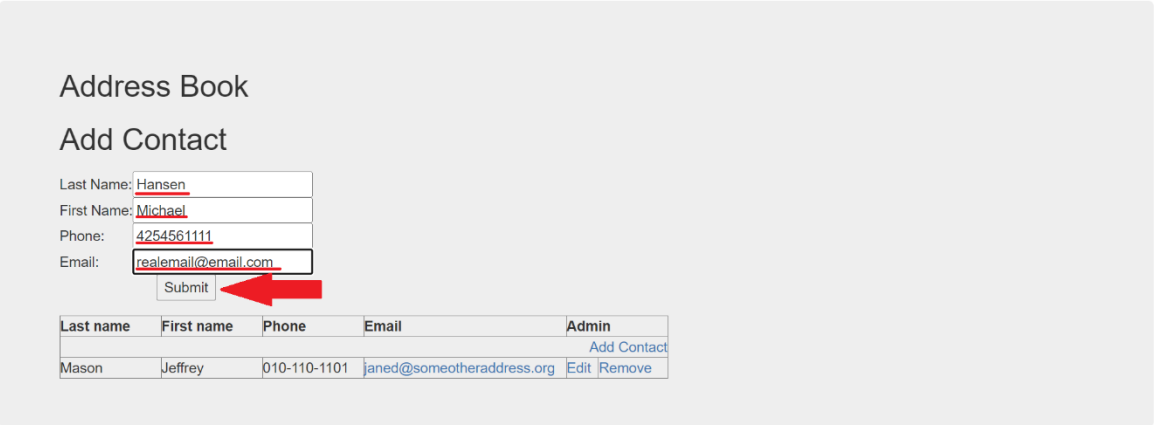


Change the details of the entry and click "Submit".



You should see a success message. Click "Add contact" to add a new contact.

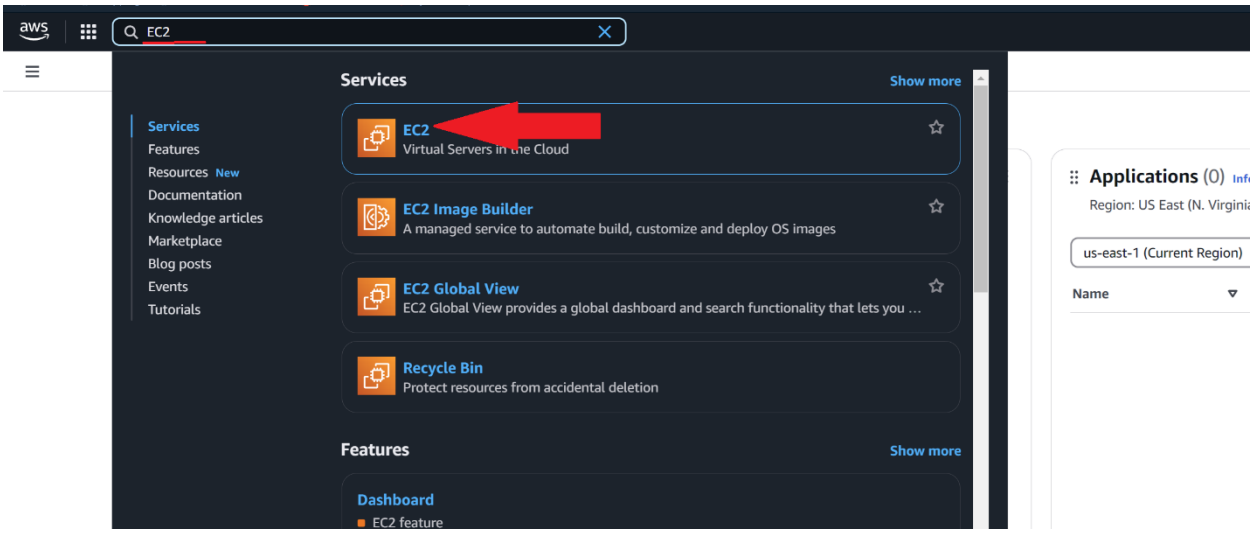
Add some information for the new contact and click “Submit”.



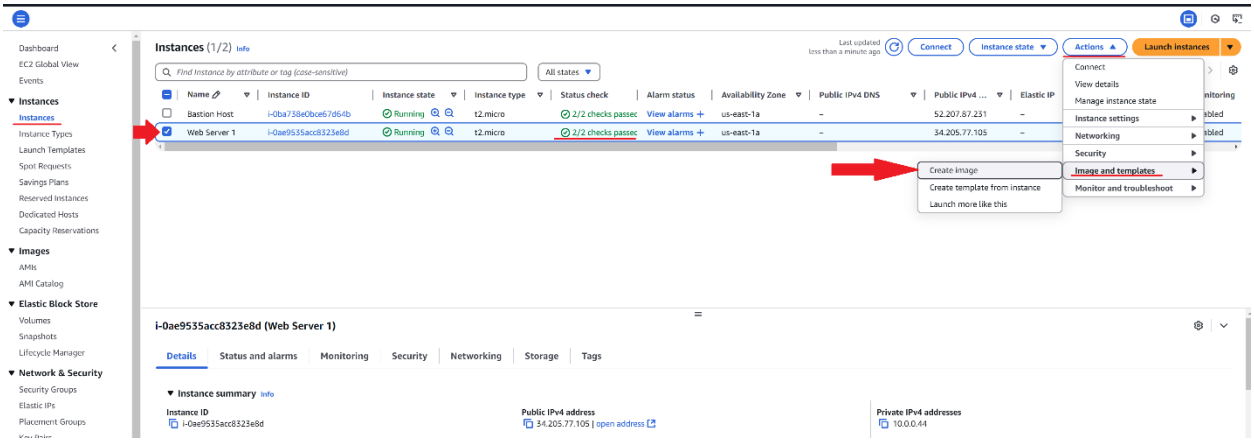
You should see both your newly created and newly modified entry in the address book.

Lab Commands (Auto Scaling)

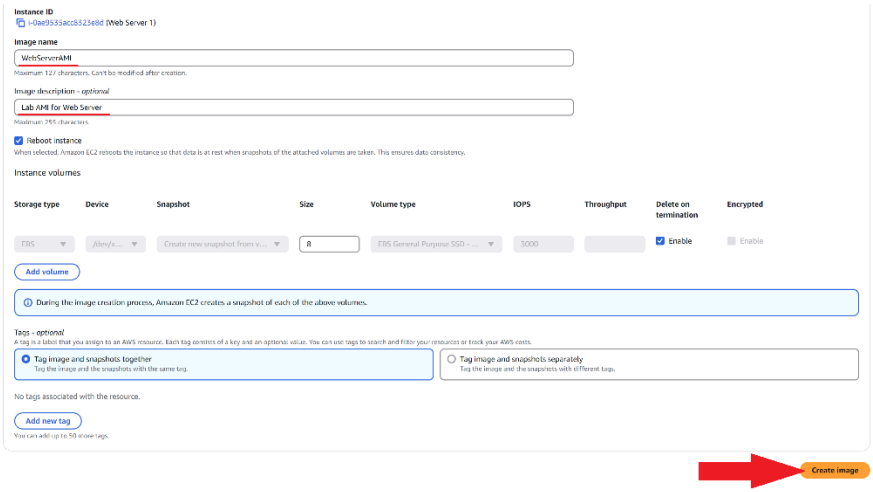
Go to the AWS management console and select “EC2”.



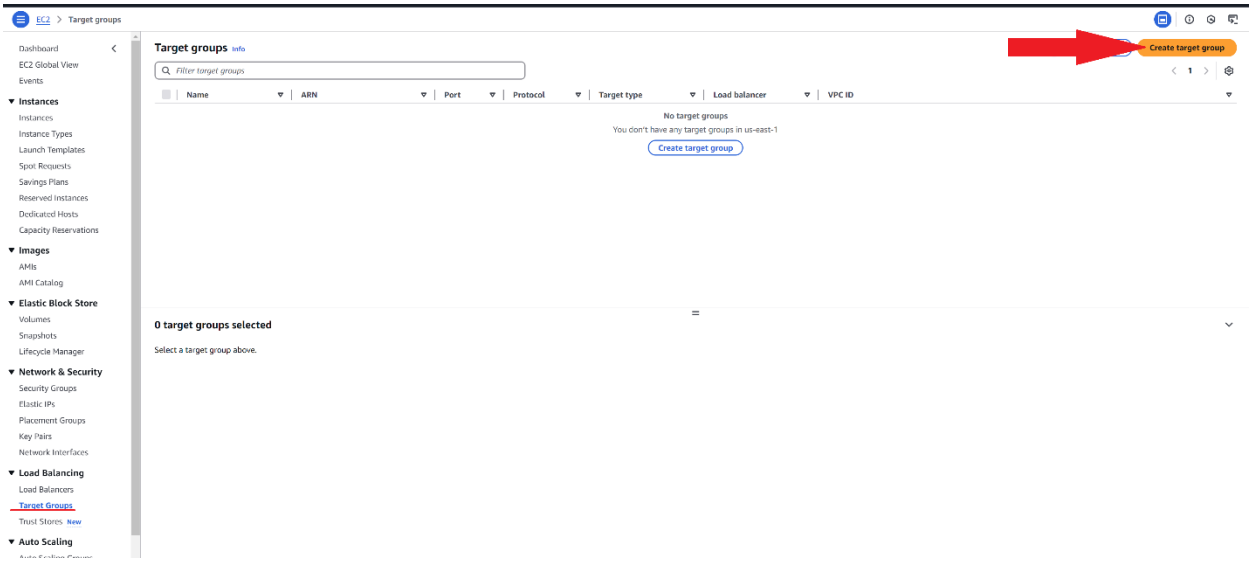
Go to Instances, ensure that Web Server 1's status is 2/2 checks passed, then click on the checkbox next to the web server and click Actions > Image and Templates > Create Image.



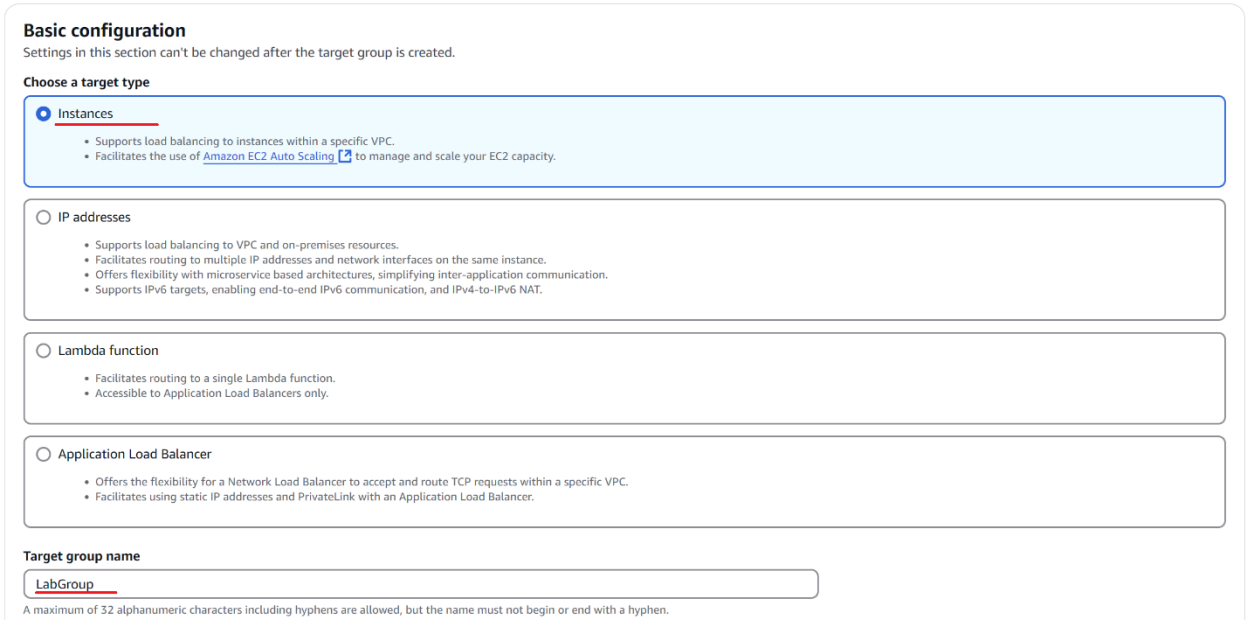
Set the image name to WebServerAMI and the image description to Lab AMI for Web Server, then click Create image.



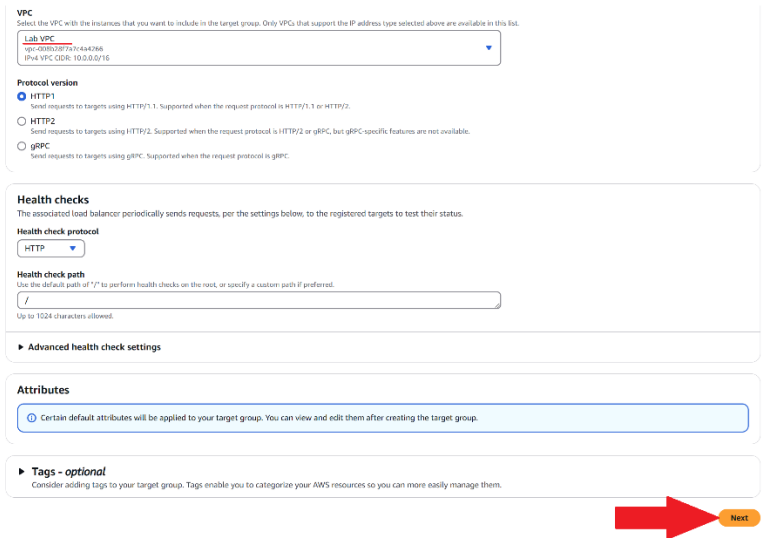
Go to Target Groups > Create target group.



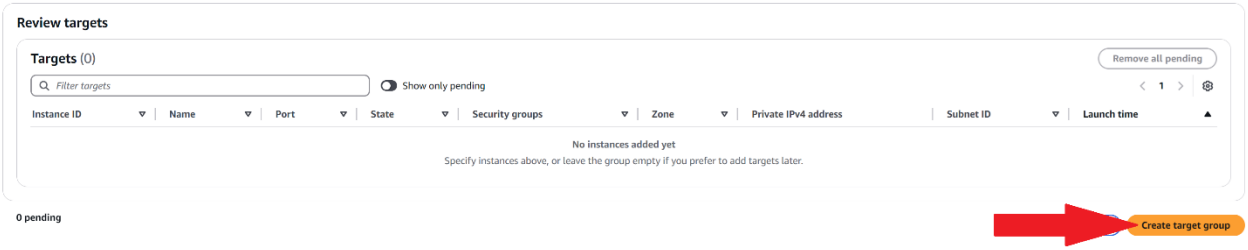
Set the target type to Instances and the target group name to LabGroup.



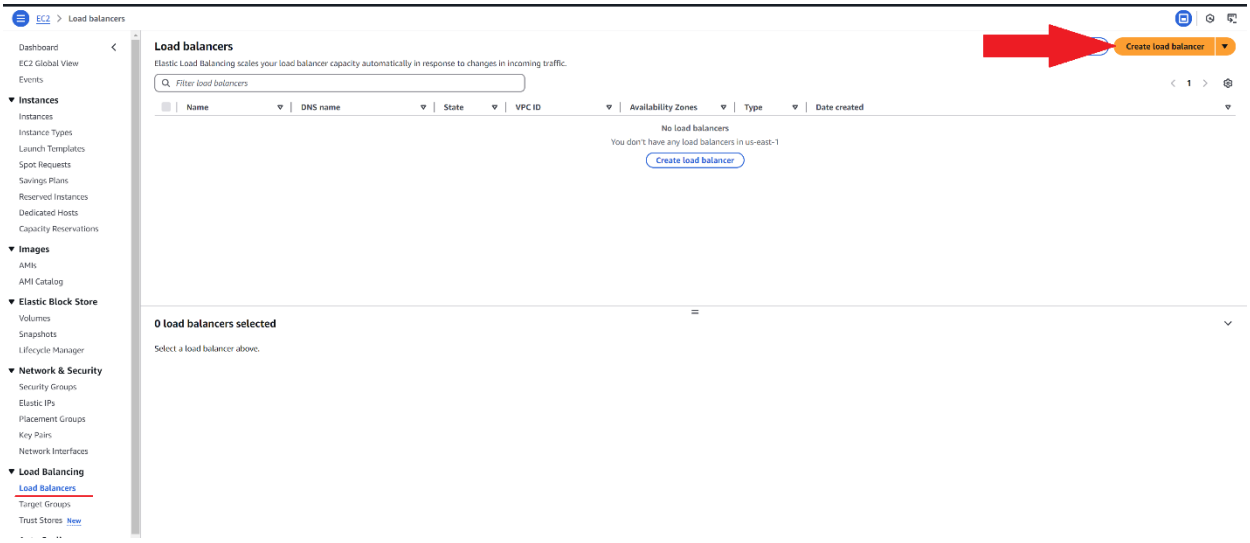
Set the VPC to Lab VPC, then click Next.



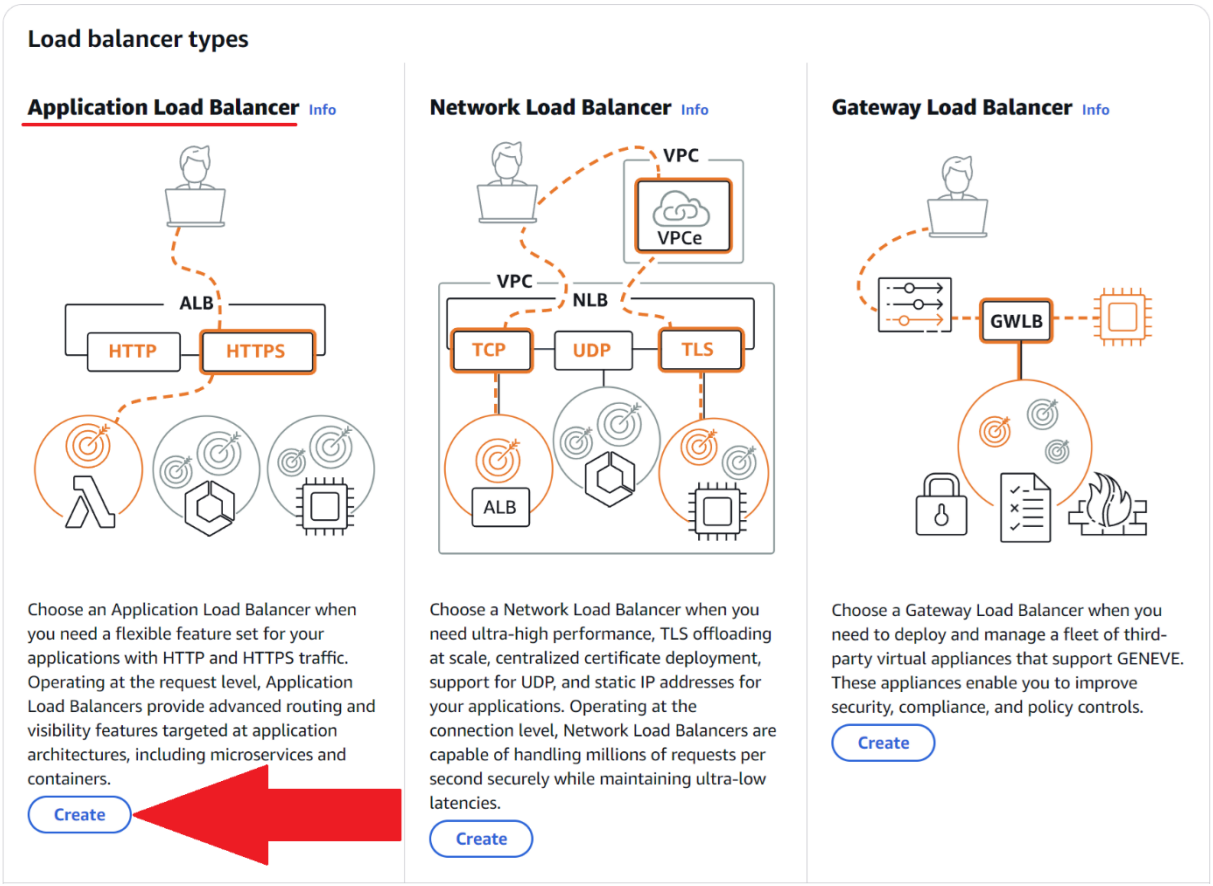
Click “Create target group”.



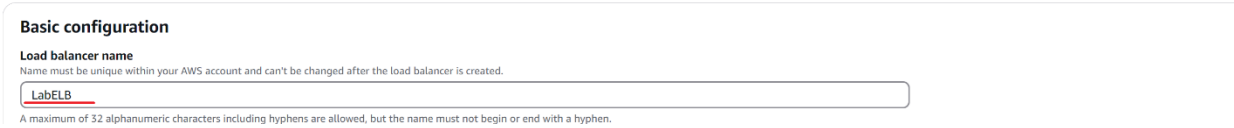
Go to Load Balancers > Create load balancer.



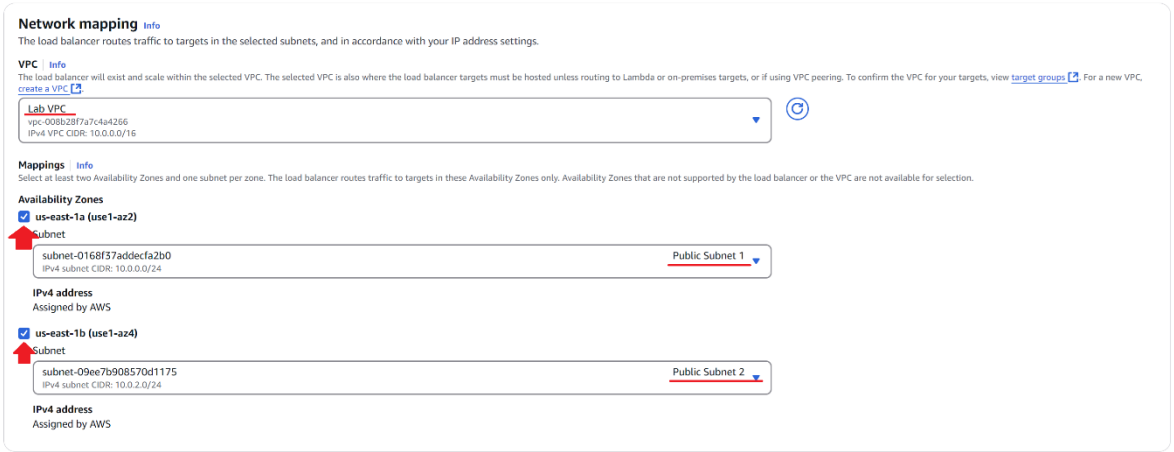
Under Application Load Balancer, click Create.



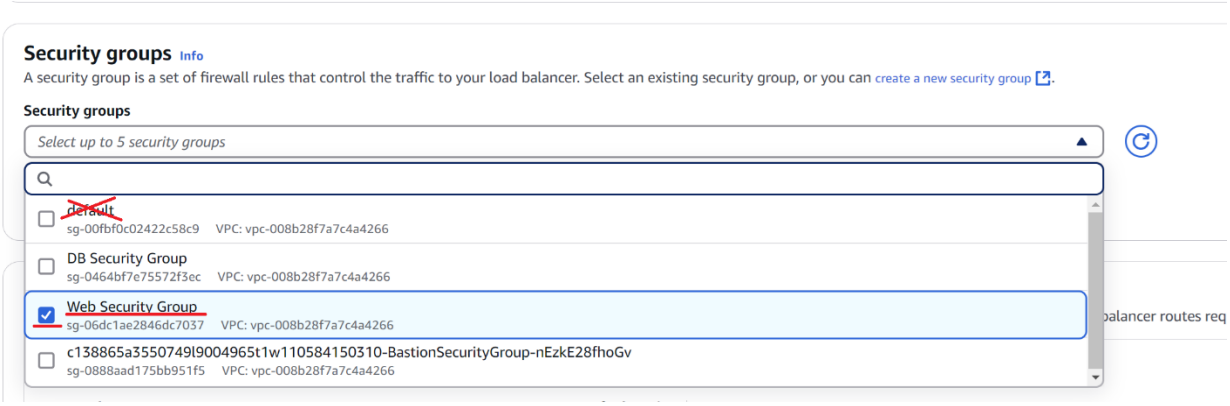
Under Basic configuration, set the load balancer name to LabELB.



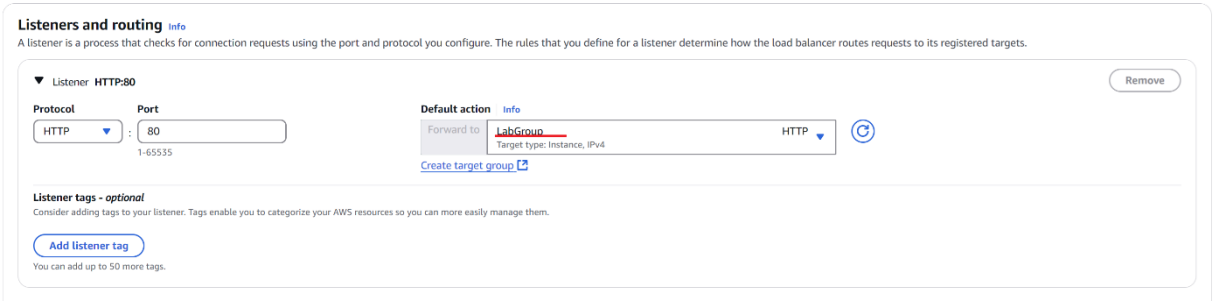
Under Network mapping, set the VPC to Lab VPC, check both availability zones, and set them to Public Subnets 1 and 2 respectively.



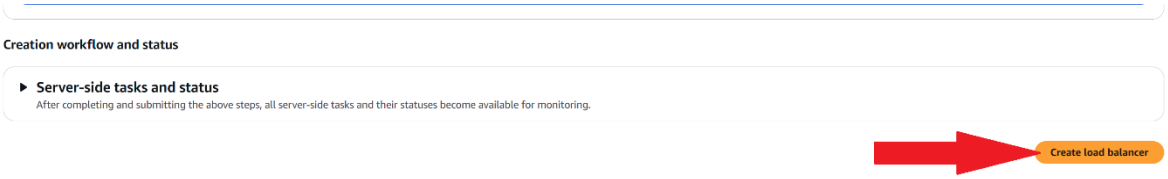
Under Security groups, select Web Security Group and unselect default.



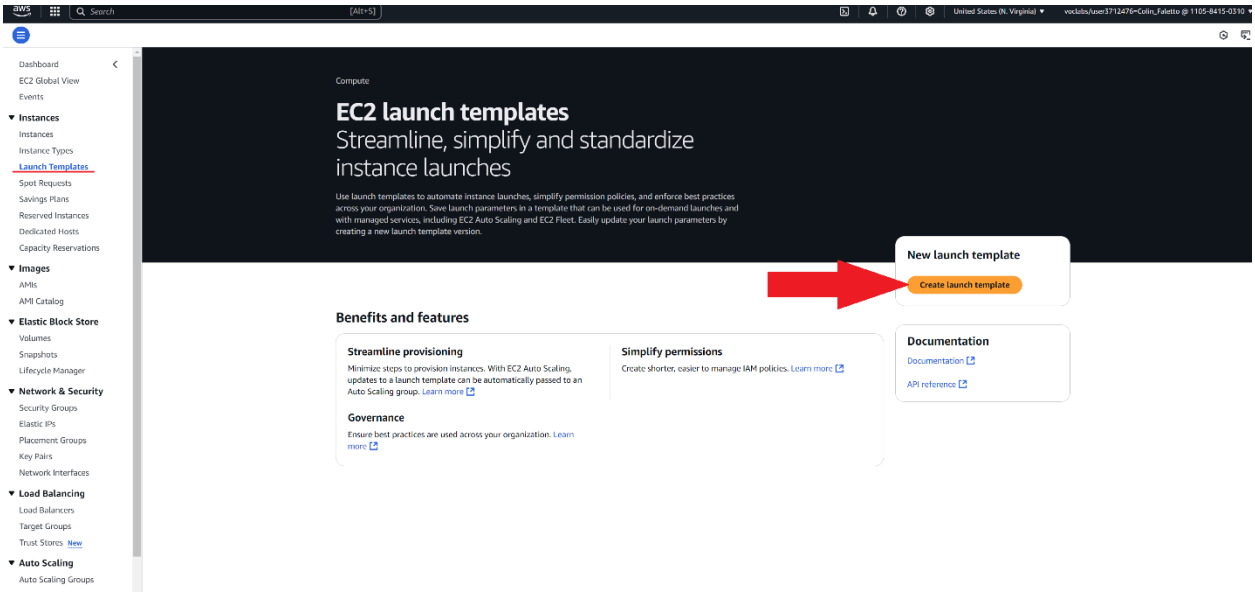
Under Listeners and routing, set the default action to LabGroup



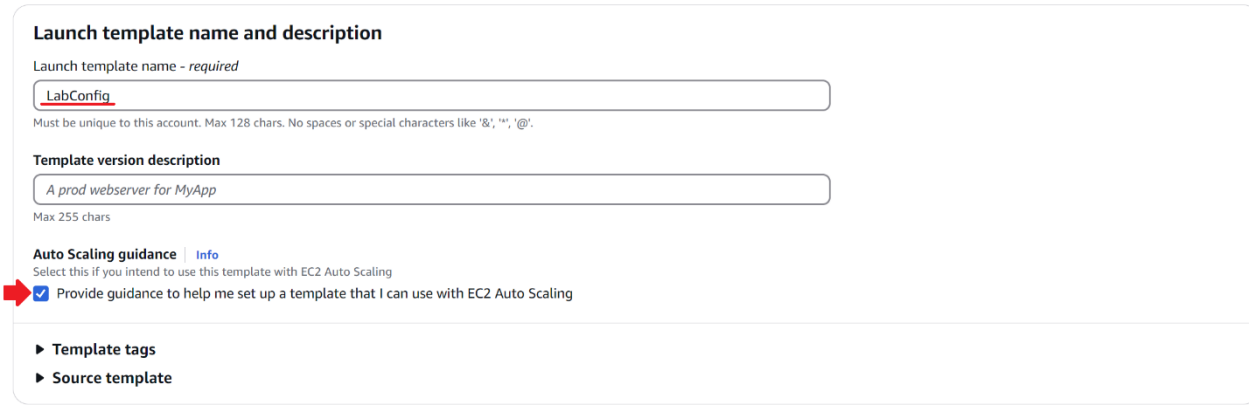
Click Create load balancer.



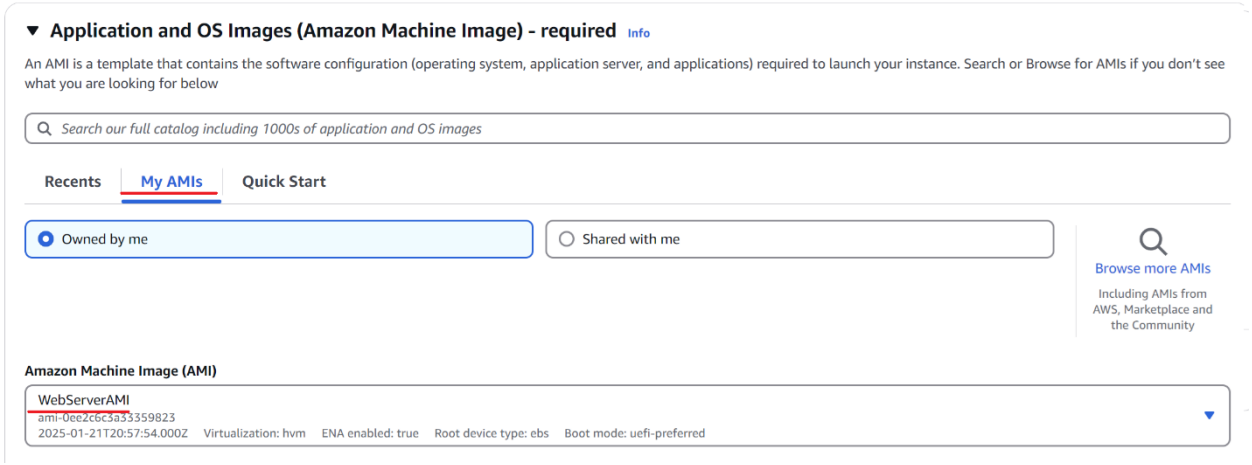
Go to Launch Templates > Create launch template.



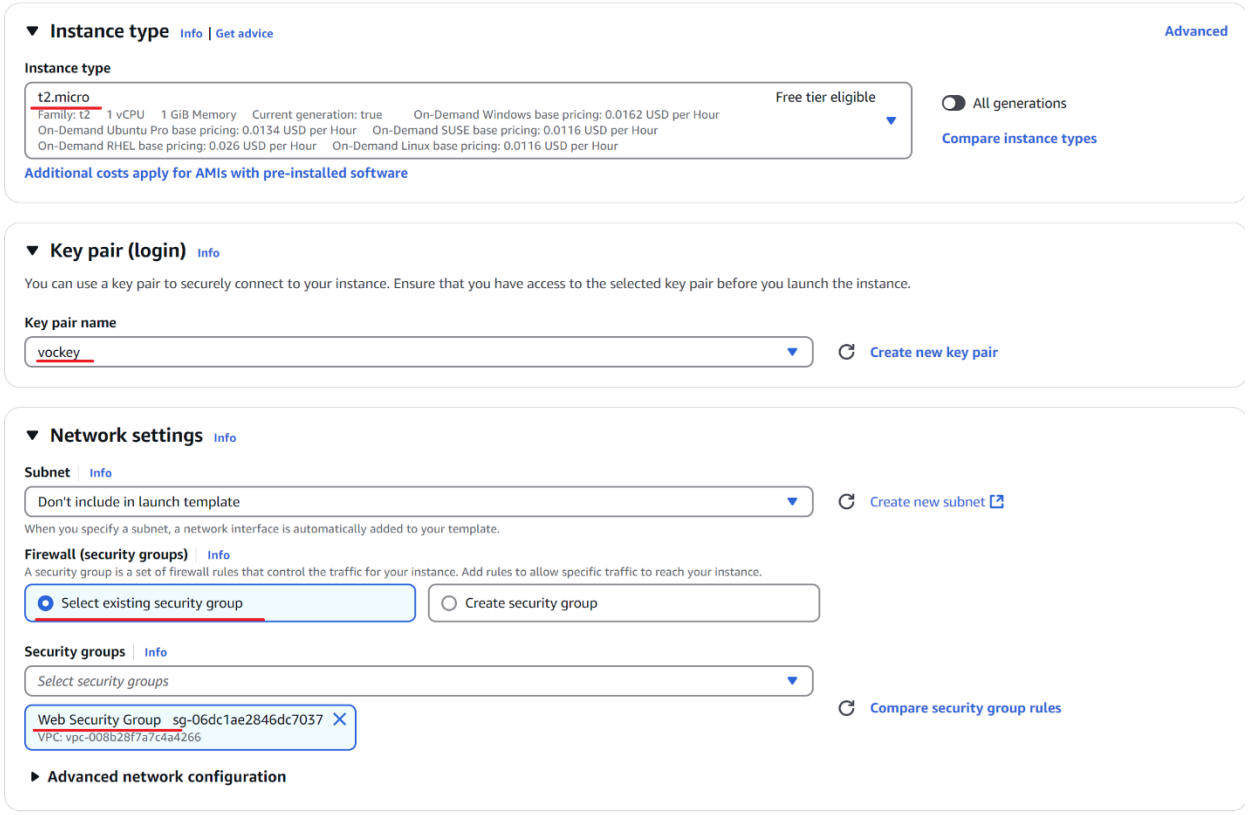
Set the launch template name to LabConfig and click the checkbox under Auto Scaling Guidance.



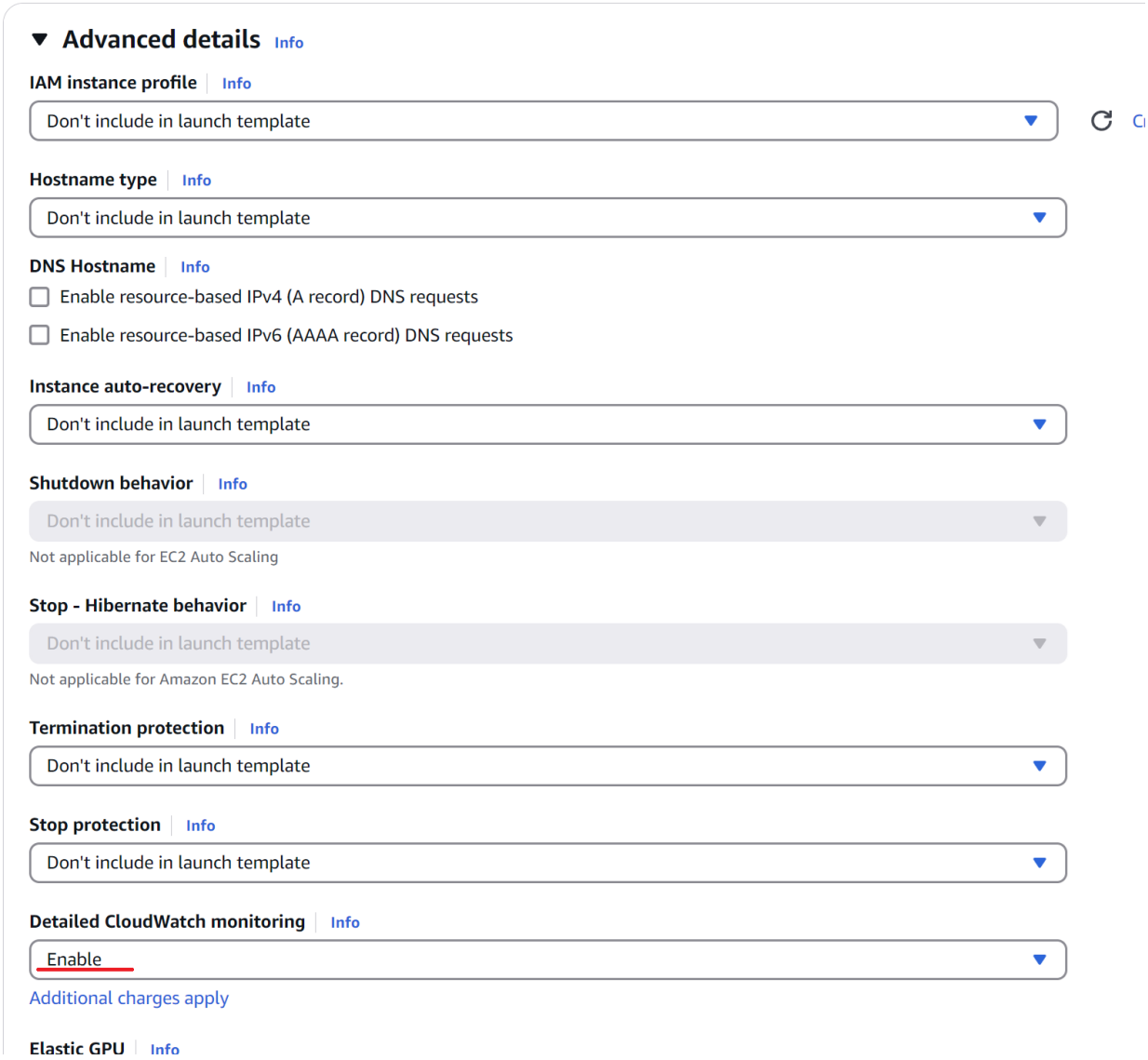
Under the Application and OS images section, click My AMIs and select the AMI you created earlier.



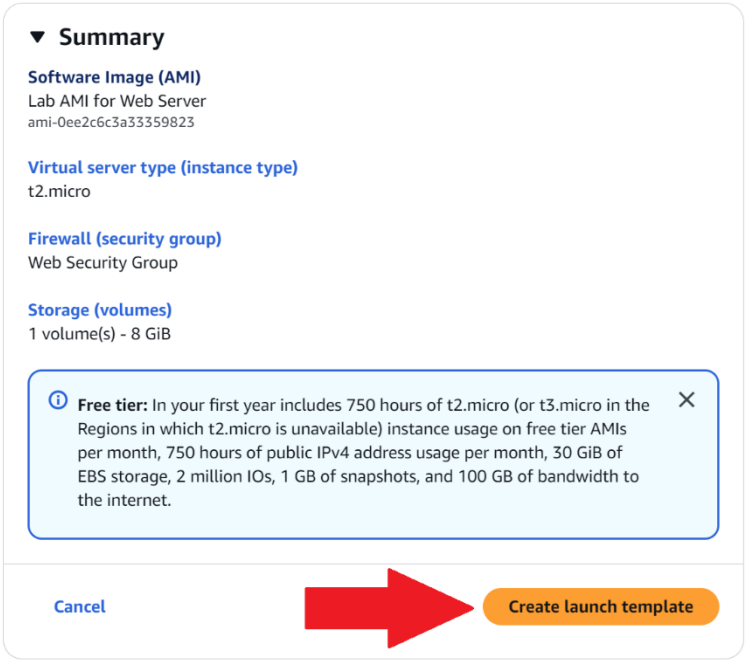
Set the instance type to t2.micro, the Key pair name to vockey, the Firewall (security groups) section to Select existing security group, and the Security groups dropdown to Web Security Group.



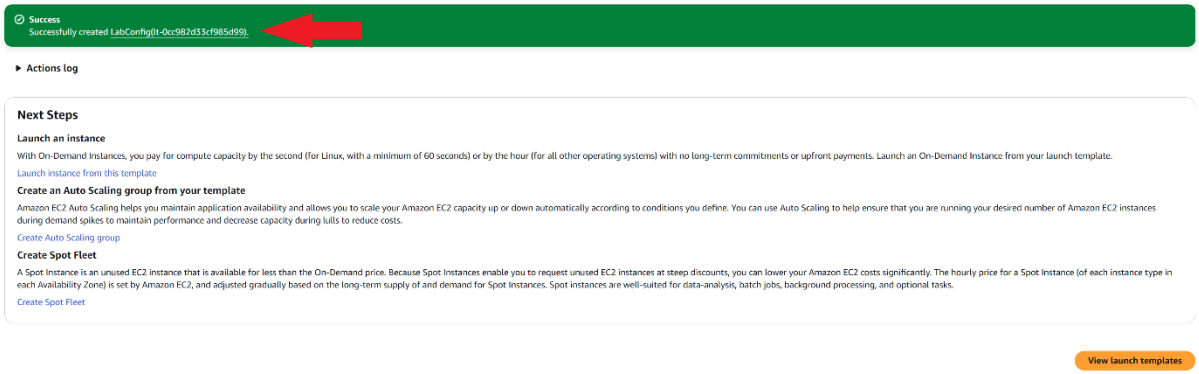
Under Advanced details, enable Detailed CloudWatch monitoring.



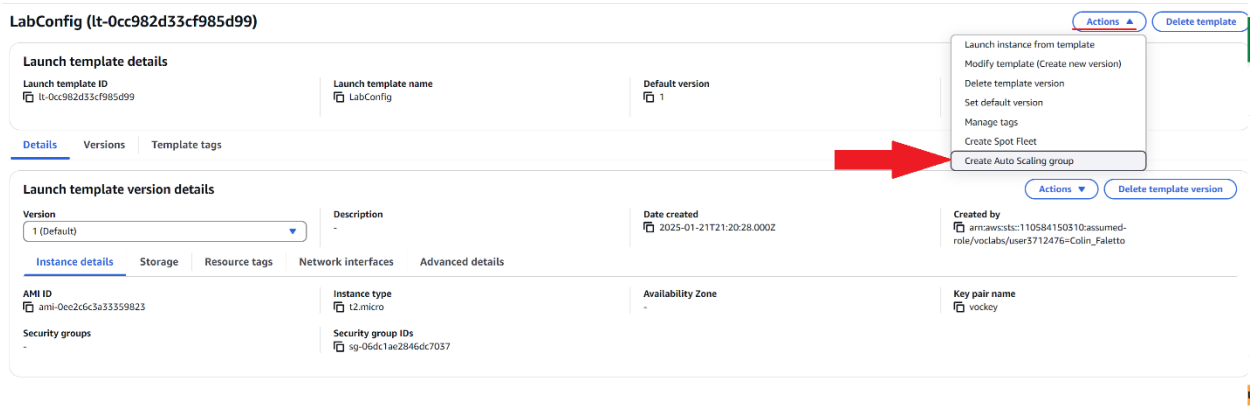
Under Summary, click Create launch template.



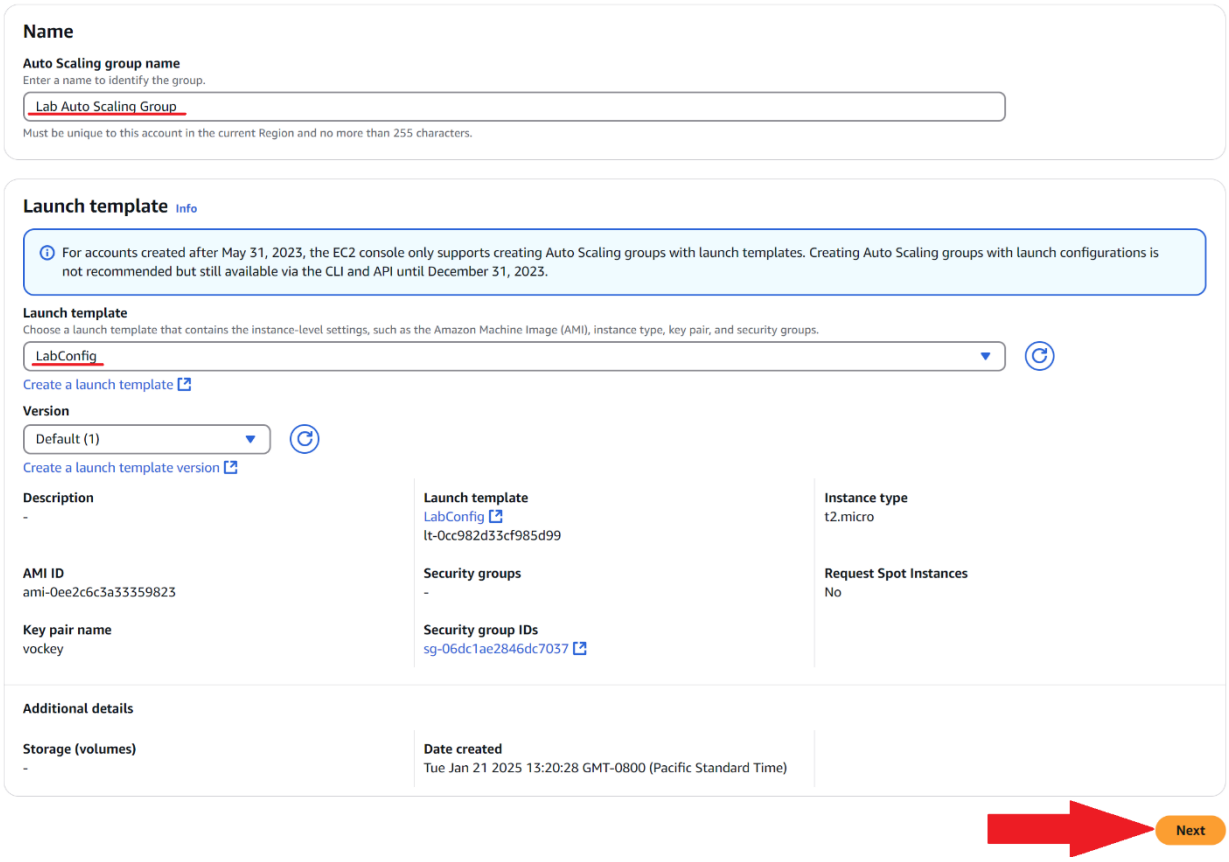
Click the hyperlink in the success dialog.



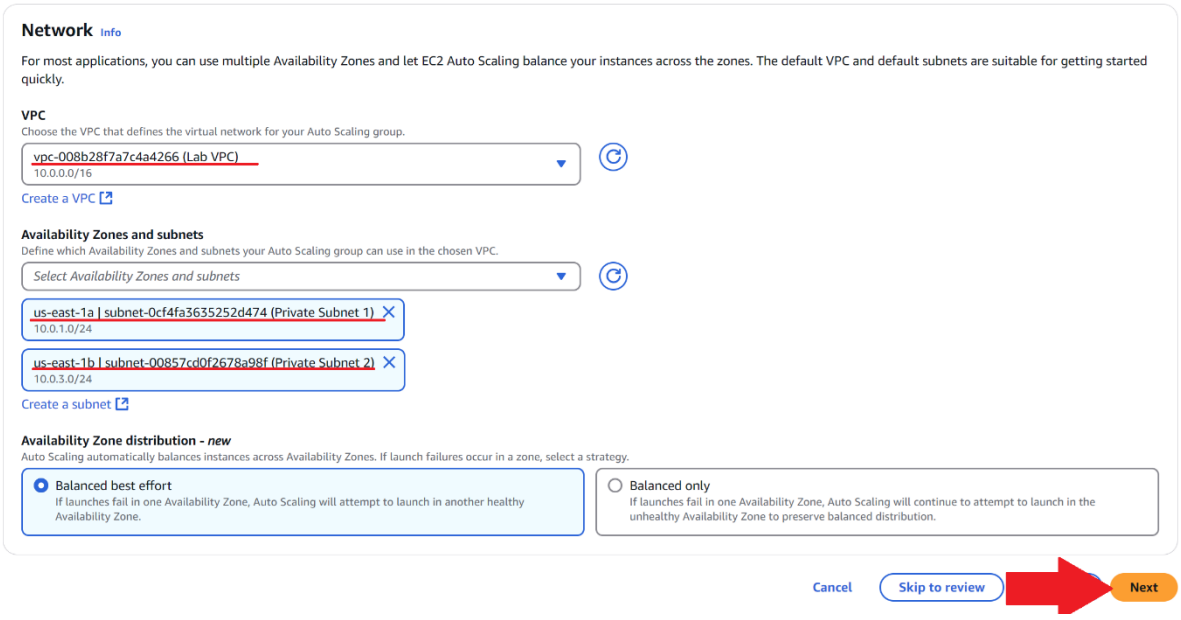
Click Actions > Create auto scaling group.



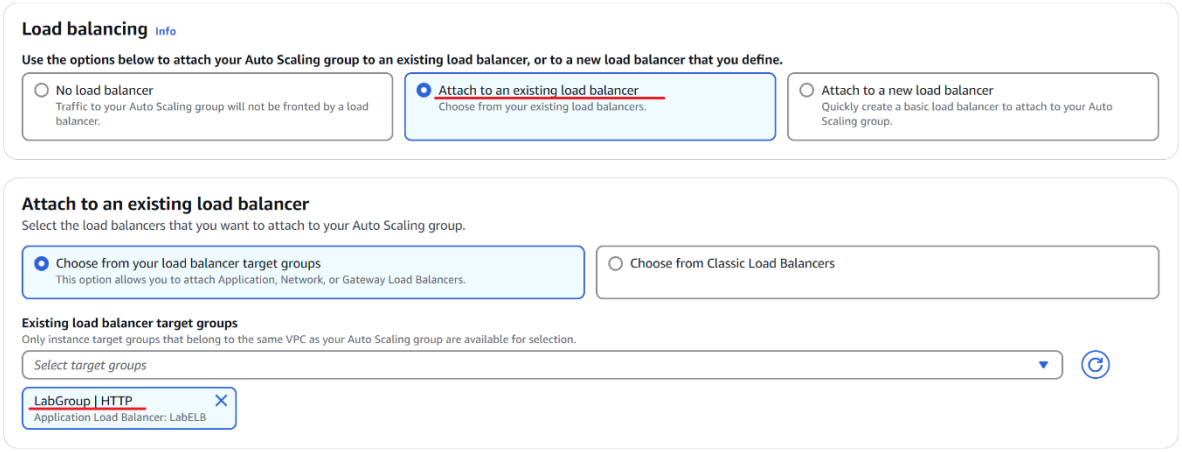
Set the name to Lab Auto Scaling Group, ensure that the Launch temlate is set to LabConfig, and click Next.



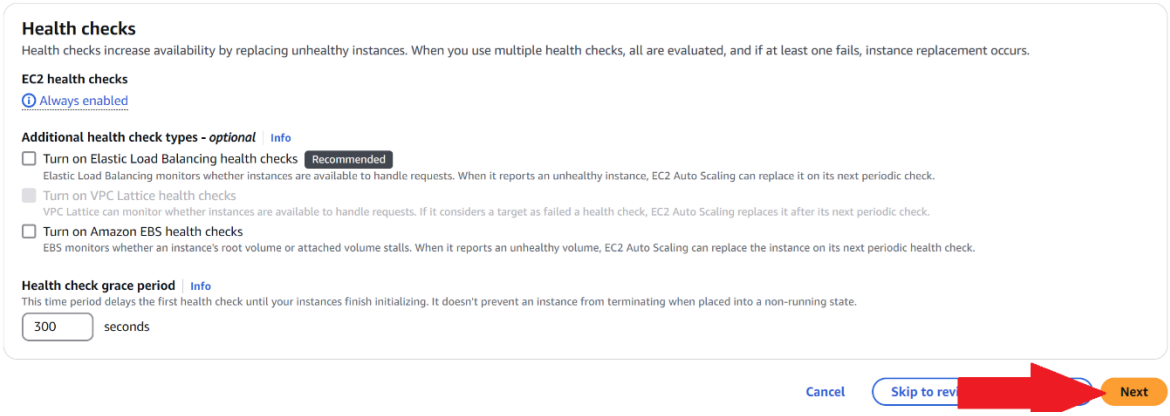
Set the VPC to Lab VPC, select both private subnets under Availability Zones and subnets, then click Next.



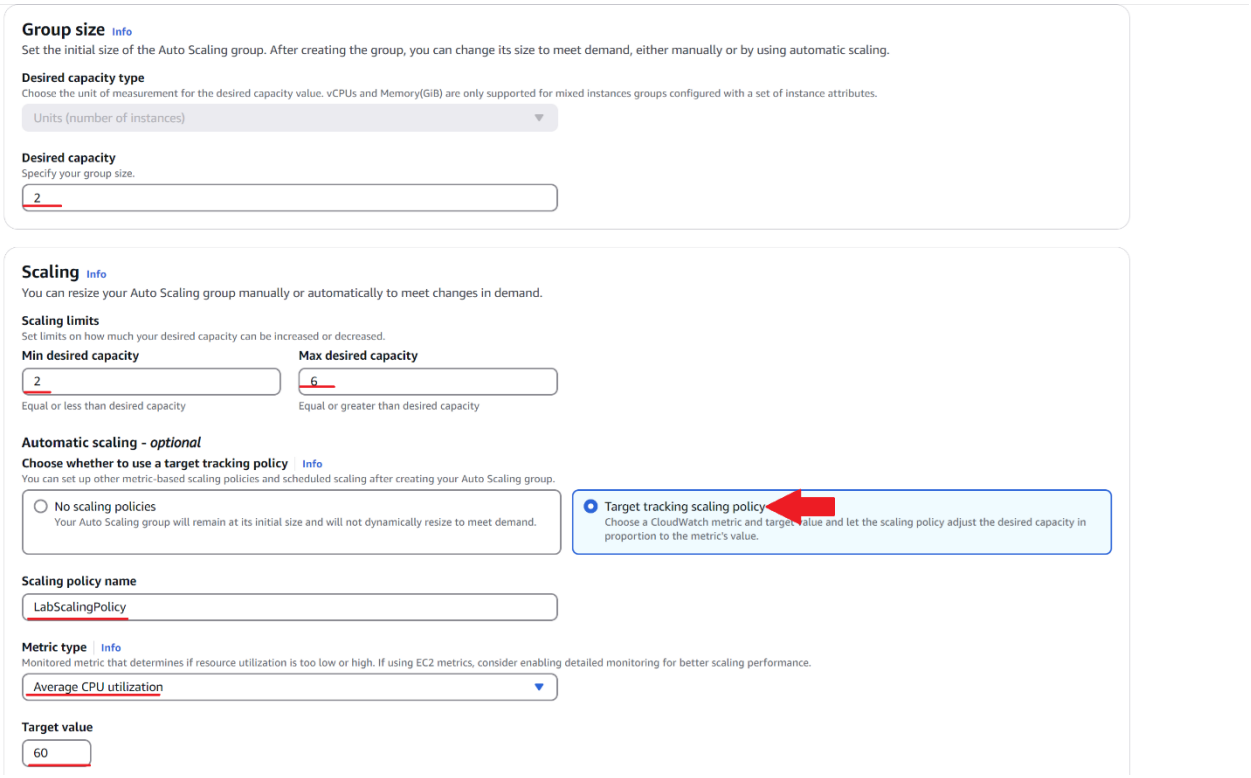
Under load balancing, select attach to an existing load balancer, and under existing load balancer target groups, select LabGroup



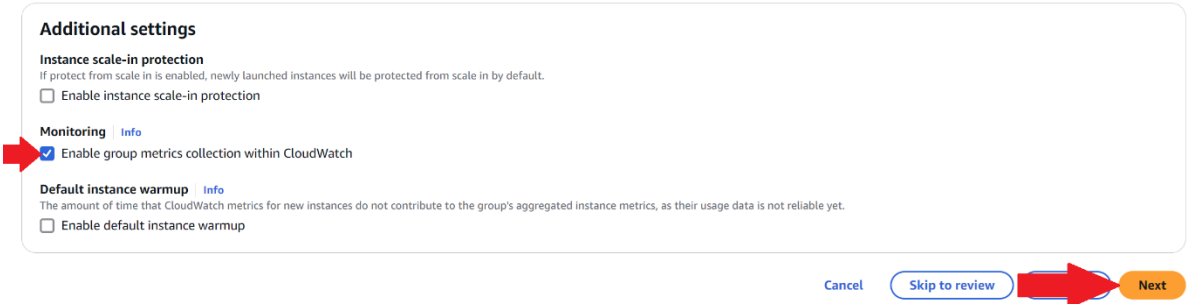
Click Next.



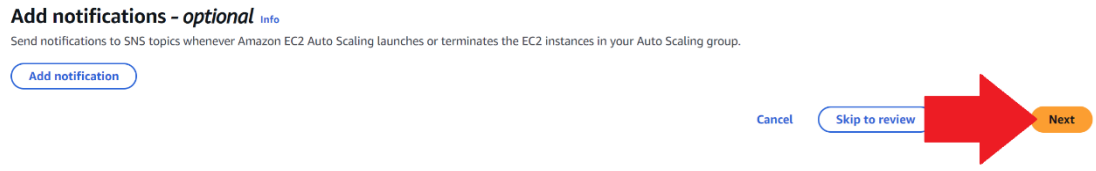
Set the the Desired capacity to 2, the minimum capacity to 2, and the maximum capacity to 6. Set the automatic scaling policy to target tracking scaling policy, set the name to LabScalingPolicy, the metric type to Average CPU utilization, and the target value to 60. This setting will automatically scale the number of EC2 instances to maintain an average of 60% CPU utilization across instances.



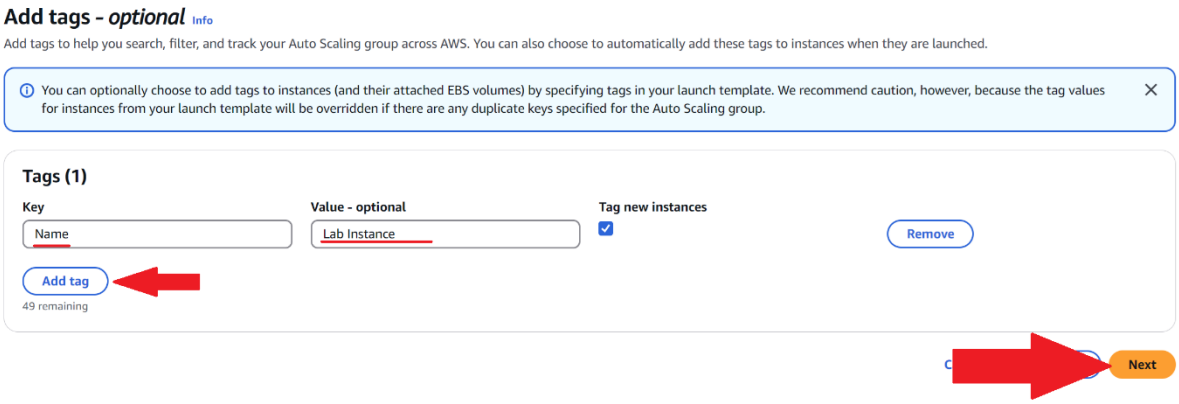
Under Additional settings, click Enable group metrics collection within CloudWatch, then click Next.



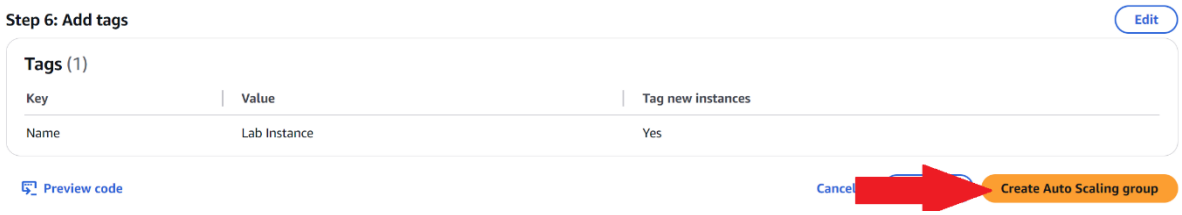
Under Add notifications, click Next.



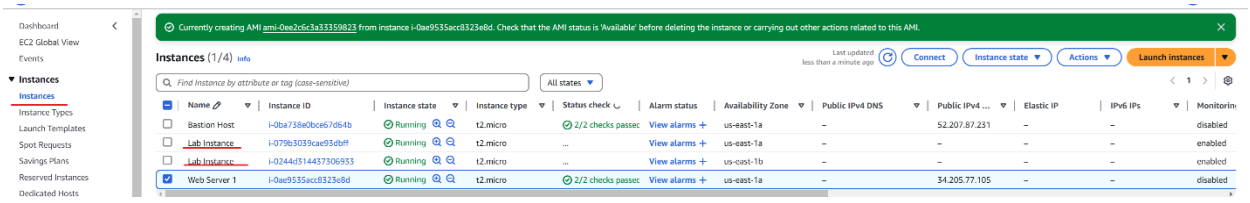
Under Add tags, click Add tag, set the key to Name, an



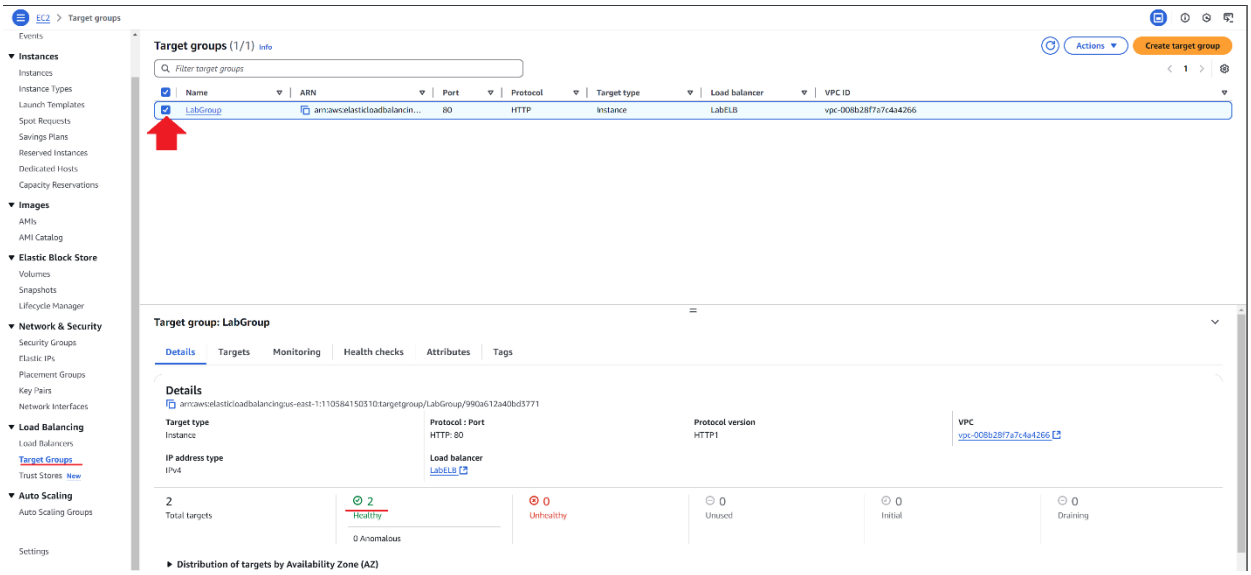
Click Create Auto Scaling Group.



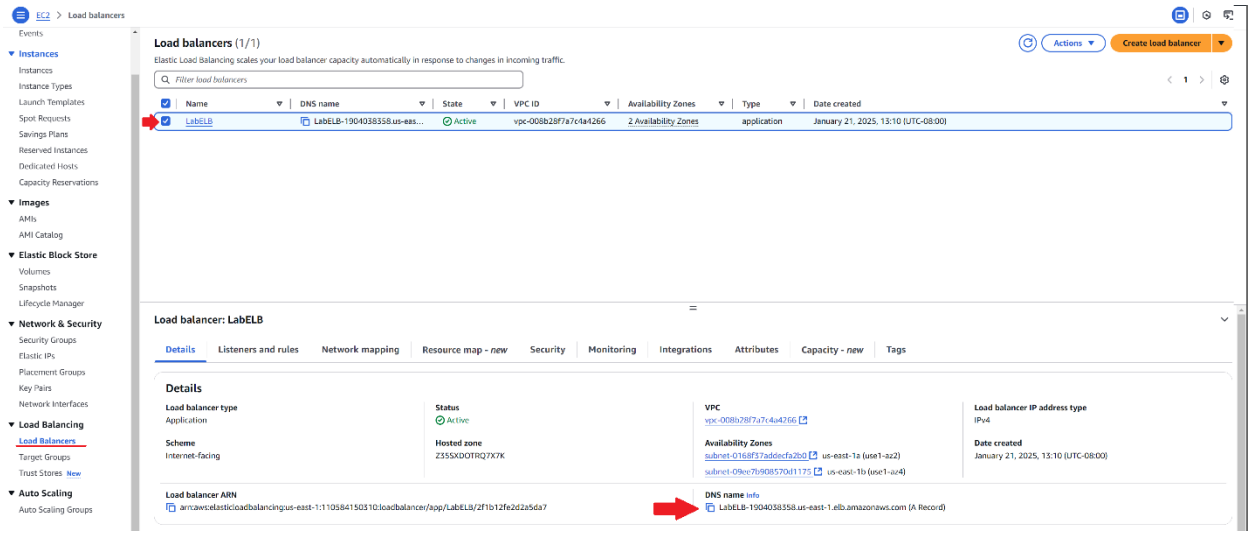
Click Instances and ensure that two copies of Lab Instance have been launched.



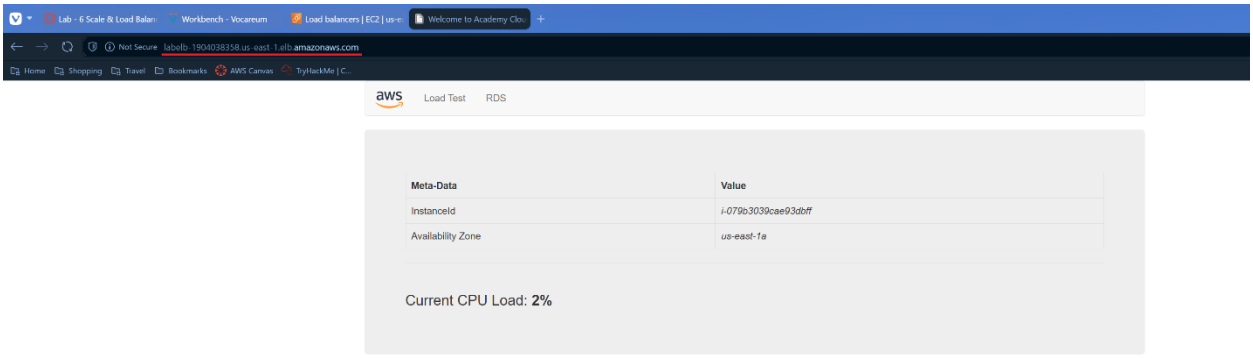
Go to Target Groups, check LabGroup, and ensure that both targets are healthy.



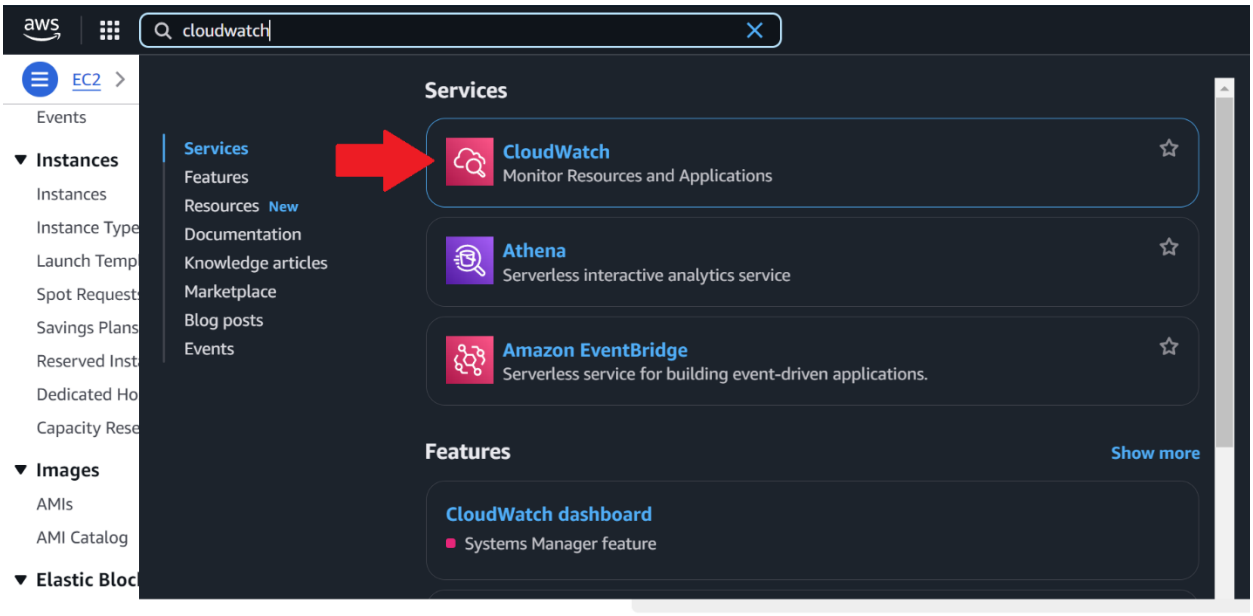
Go to Load balancers, select LabELB, then copy the DNS name.



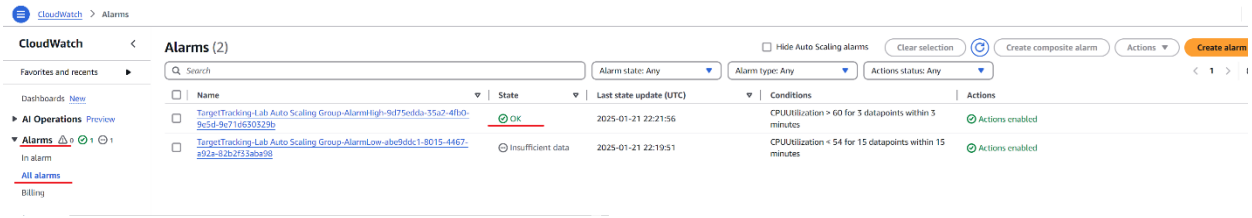
Open the DNS server in a new browser tab and ensure that you see the EC2 instance web server as shown below.



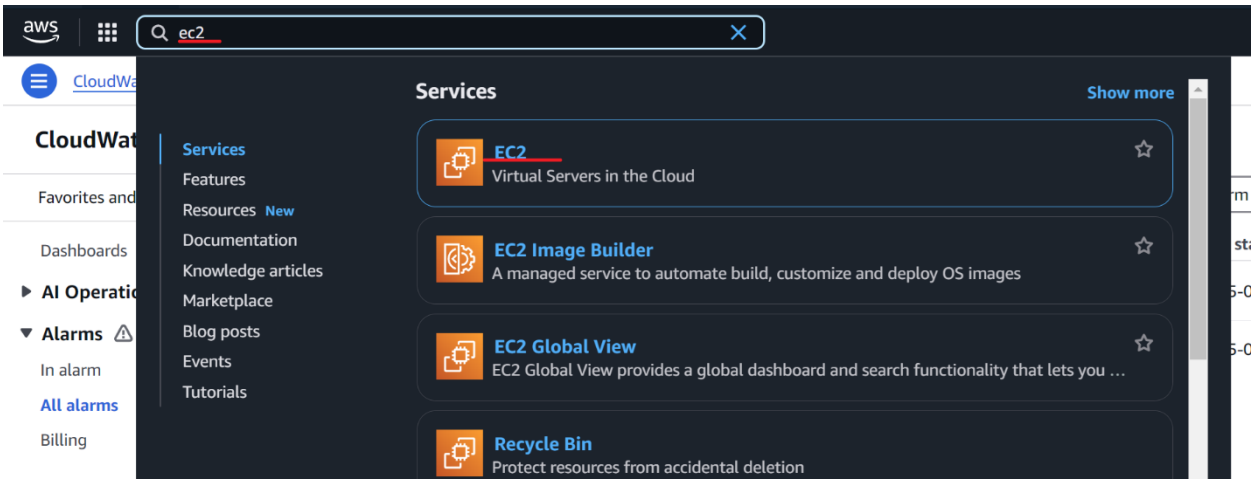
In the search bar, search for and open Cloudwatch.



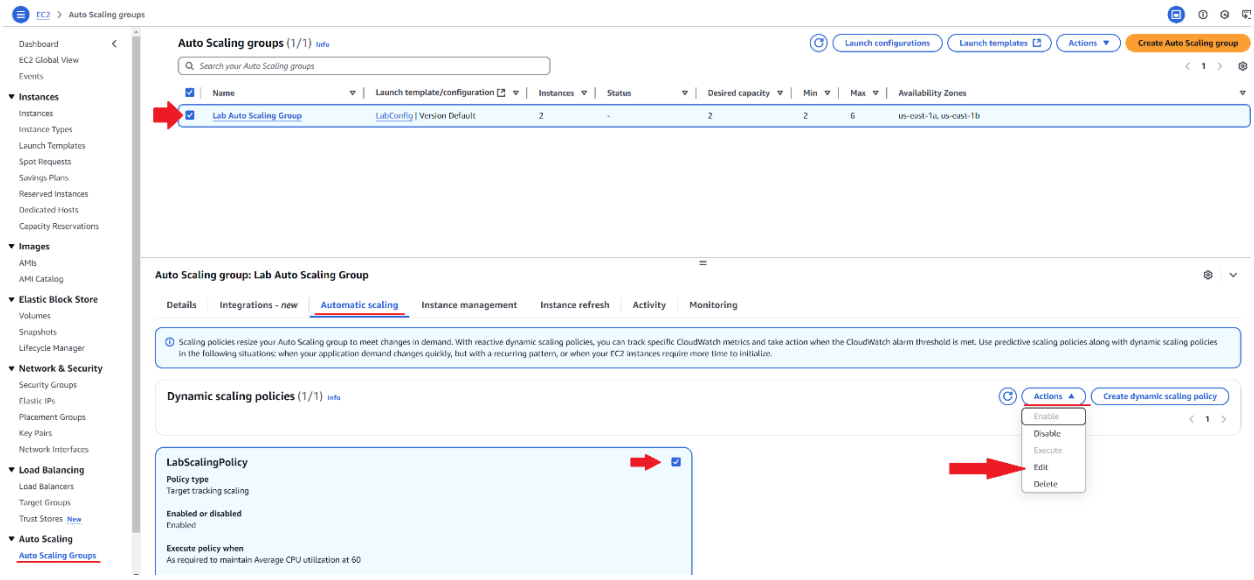
Click Alarms > All alarms, and ensure that the AlarmHigh alarm reports OK.



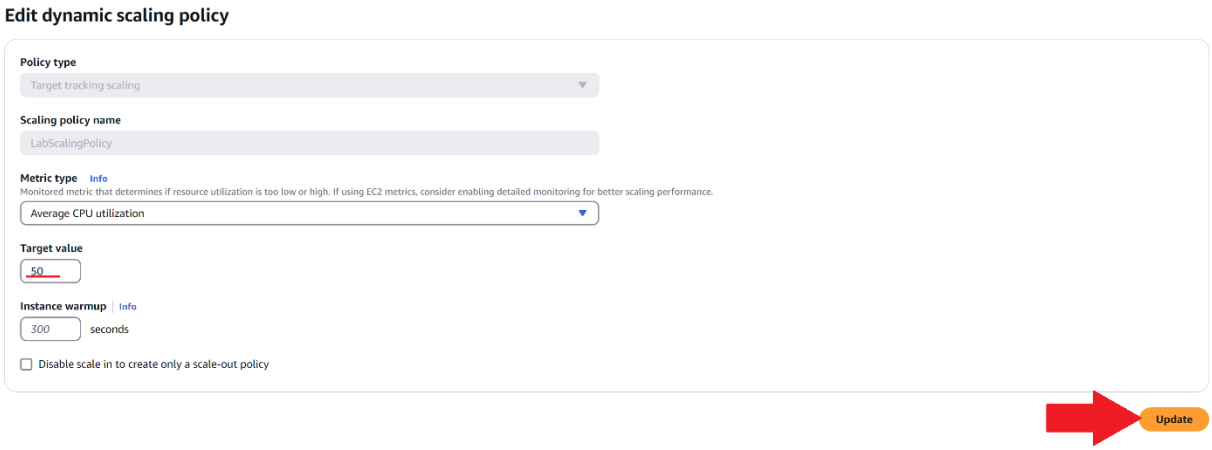
Go back to EC2 from the search bar.



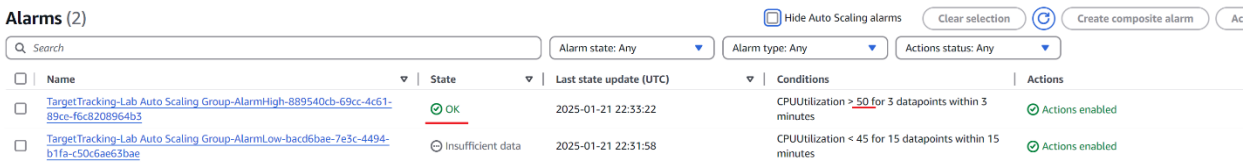
Select Auto Scaling groups, select Lab Auto Scaling group, click the Automatic Scaling tab, select LabScalingPolicy, then click Actions > Edit.



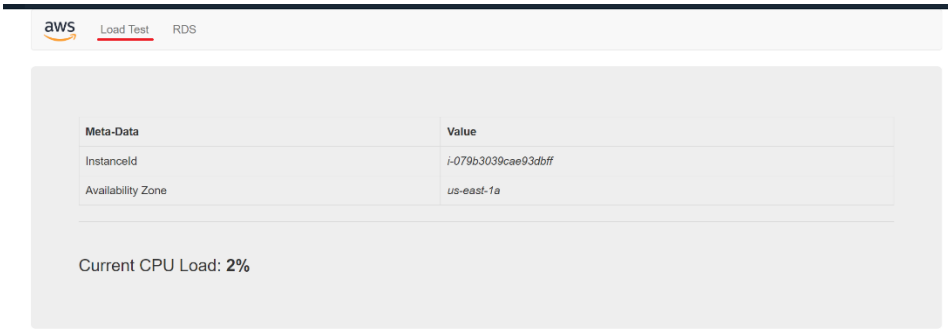
Set the target value to 50 and click Update.



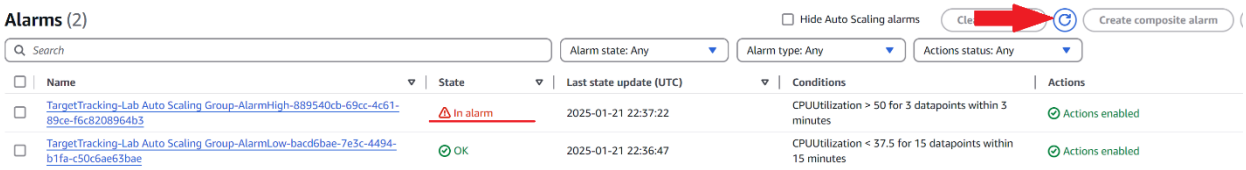
Return to Cloudwatch > Alarms > All alarms and ensure the updated AlarmHigh alarm still reports OK.



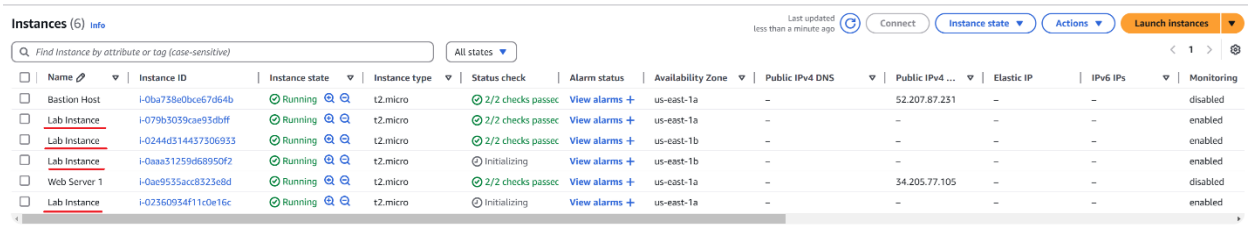
Return to the EC2 web server tab and click Load Test.



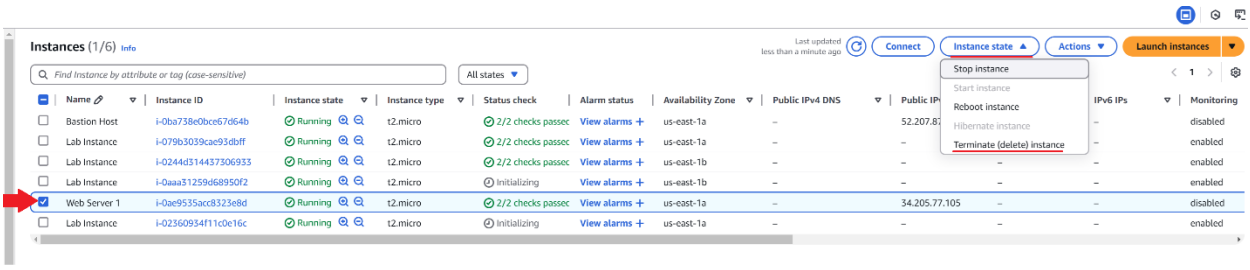
Return to the AWS console and continually press the refresh button on the alarms page. Eventually, the AlarmHigh will report that it is in alarm.



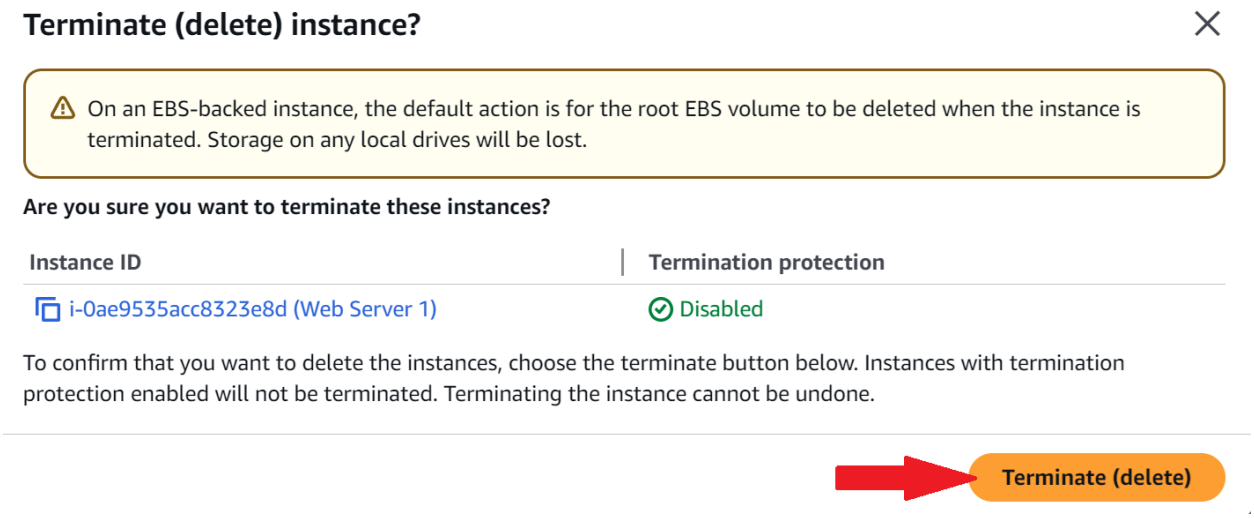
Return to EC2 > Instances. You should see that the Auto Scaling Group has automatically created more instances of the Lab Instance AMI.



Next, to clean up, select Web Server 1 and click Actions > Terminate (delete) instance.

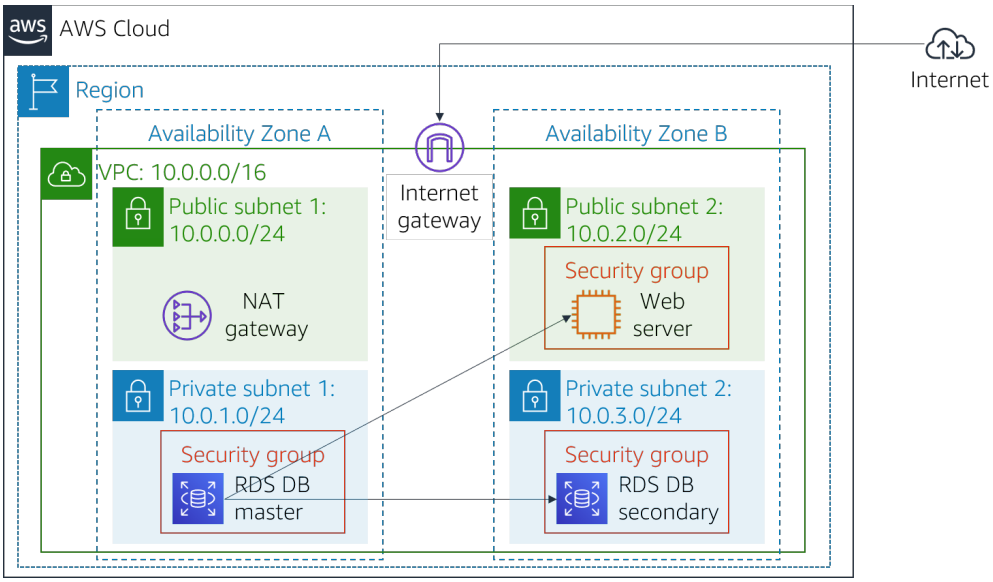


Click Terminate.

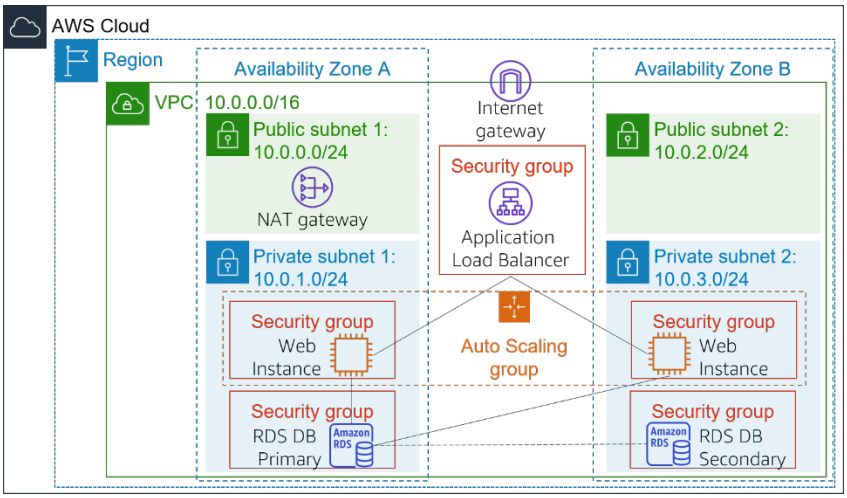


Topology (EBS)

Topology (RDS)



Topology (Auto Scaling)



Problems

No problems were encountered in this lab.

Conclusion

To wrap up, these three labs were a great addition to my understanding of the AWS cloud and management console. Through learning about Elastic Block store, I now have a deeper understanding of the inner workings of Elastic Compute Cloud. Through learning about RDS, I now have a solid foundation of knowledge about how large amounts of data with a consistent structure are stored. Through learning about auto scaling, I'm now confident that I could scale AWS services for use at a company with heavy reliance on cloud resources.