

AWS Labs 4-6

Aaron Yu

Lab Partners: None

Cisco CCNP

Introduction

# Purpose

The purpose of these labs is to continue to put knowledge of Amazon Web Services (AWS) learned through the course’s videos into the actual console through preconfigured environments to learn aspects of the different web services provided by AWS such as EBS, RDS, and ELB.

# What is AWS?

Amazon Web Services is a service made by Amazon that provides pay-as-you-use cloud computing. This model is useful as this allows for people in need of these services but are unfamiliar with how these things work to be able to invest less at first and increase infrastructure as needed, as AWS allows for easy maintenance and scalability. In essence, AWS allows for a very dynamic infrastructure, able to adapt new applications or systems in minutes, and allow for no money being wasted on excess equipment.

Lab 4 – Working with EBS

# Purpose

This lab teaches and goes more in depth on AWS Elastic Block Store (EBS), which is a very important aspect of Amazon EC2.

# Background Information

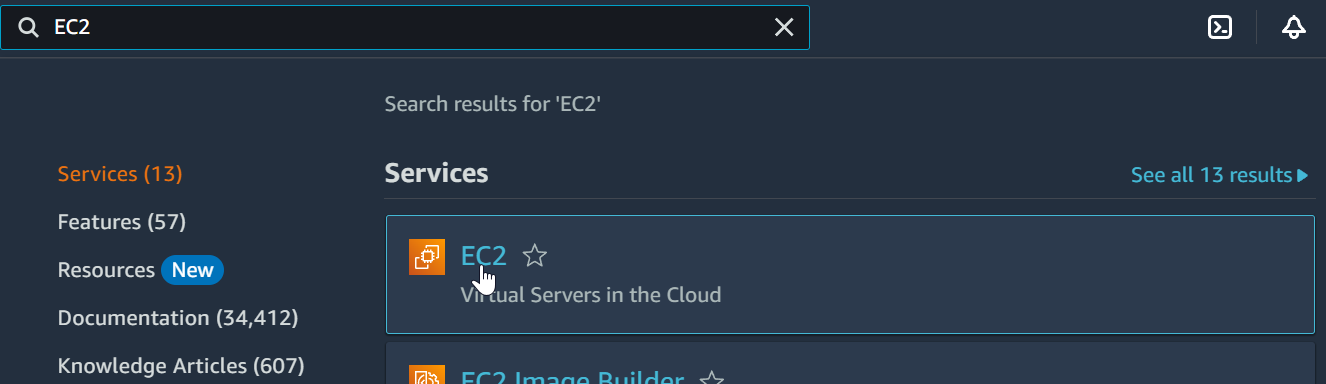
AWS EBS is how AWS stores data and files. It is an easily scalable block storage service, in which these blocks can be mounted onto EC2 instances to be used. EBS is for data that should be easy access and kept for later user. EBS works with EC2 to provide scalable block storage volumes. EBS also has the ability to create snapshots, which can be used to clone or back up data, which is very useful to keep information safe or to reduce busy work tasks. EBS also provides different types of volumes for specific use cases, such as General Purpose SSD or a Provisioned IOPS SSD.

# Lab Summary

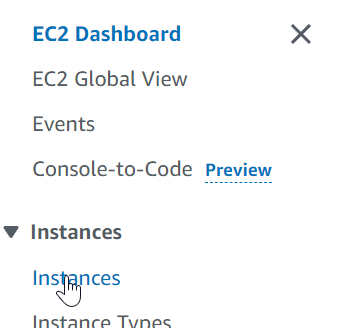
In this lab we create an EBS volume through an EC2 instance to display a webpage when mounted on EC2, modify the volume using a Linux shell and learn several Linux commands, attach the volume to an EC2 instance, and create and restore a snapshot of the volume to simulate restoring deleted files.

# Configurations

Open EC2 from the search bar.

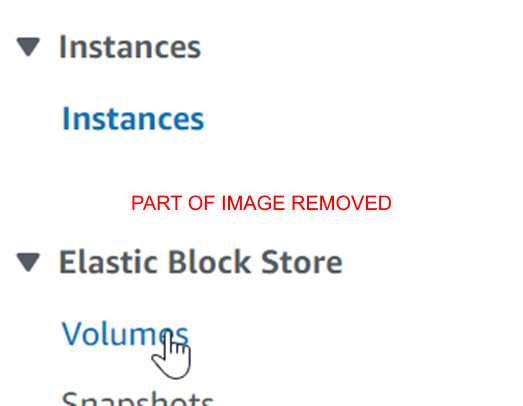


From the left sidebar go to Instances.



There is already an instance, wait for its status to become 2/2 checks passed.

Go to volumes from the left sidebar.

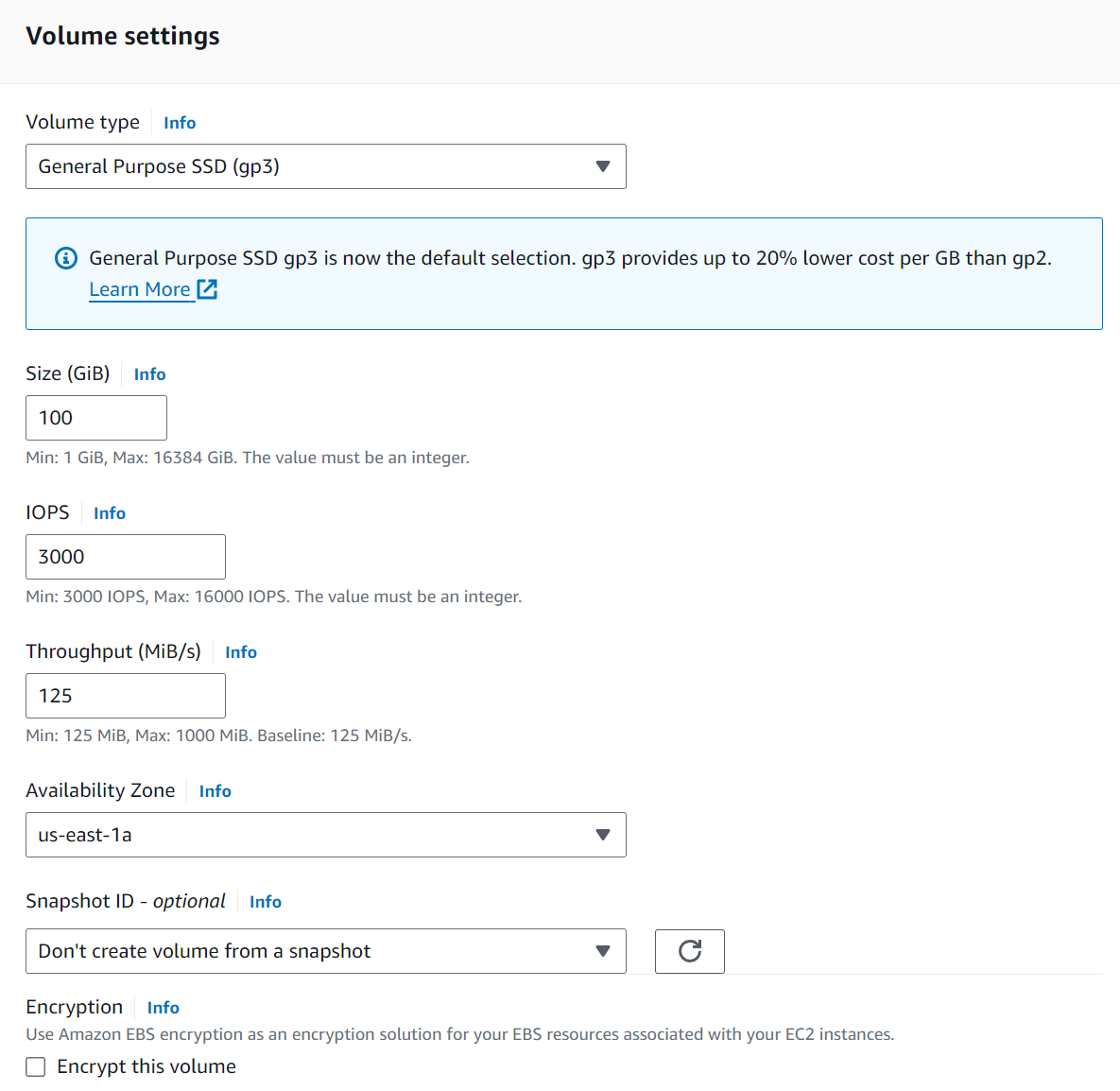


(Part of image removed for visual purpose)

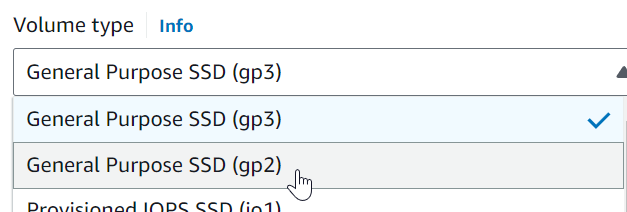
Press the “Create volume” button to create a EBS volume.



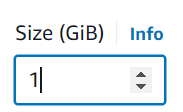
This page will show up.



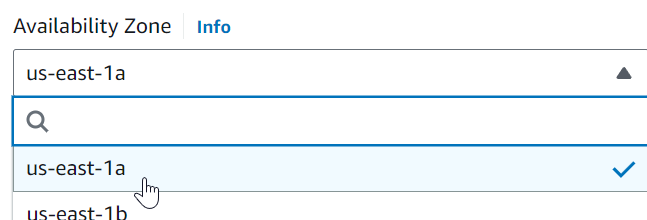
Set the volume to gp2.



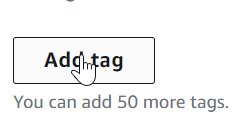
Set the size to 1GB.

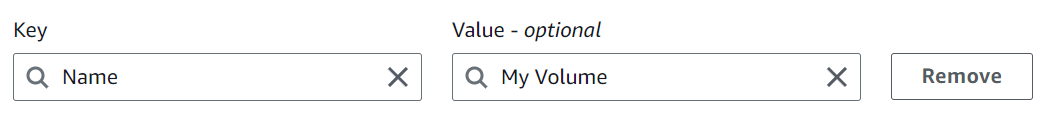


Set the AZ to us-east-1a.

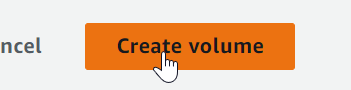


Add a tag with the key Name with the value My Volume.





Press the “Create Volume” button to create the EBS volume.







Reload the page periodically until the volume is marked Available.

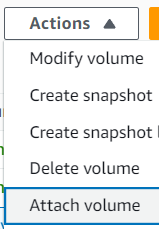
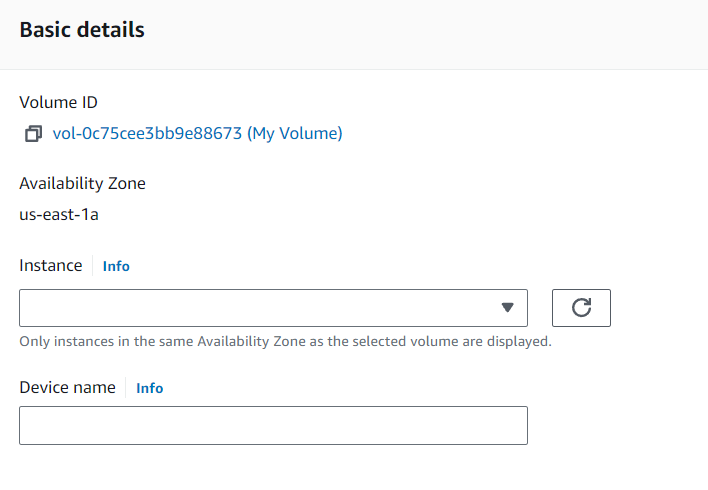




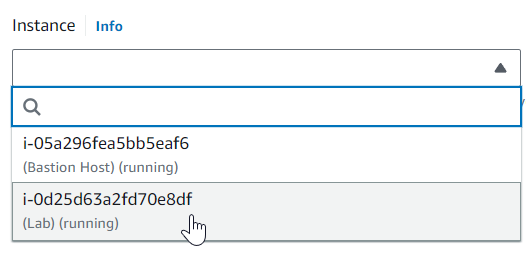
Select the My Volume volume, open the Actions dropdown menu, and select Attach volume to attach the volume to EC2.





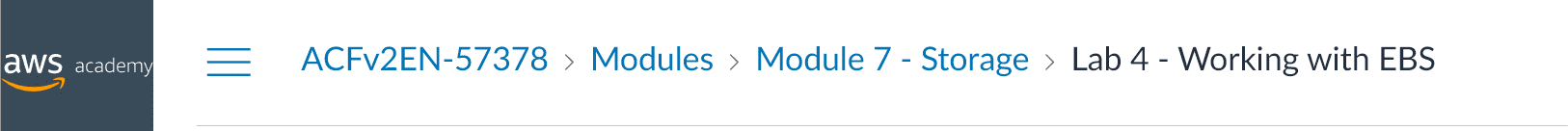
  


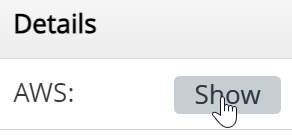
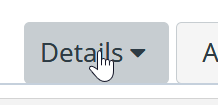
Set the instance to the Lab instance and the Device name to /dev/sdf.



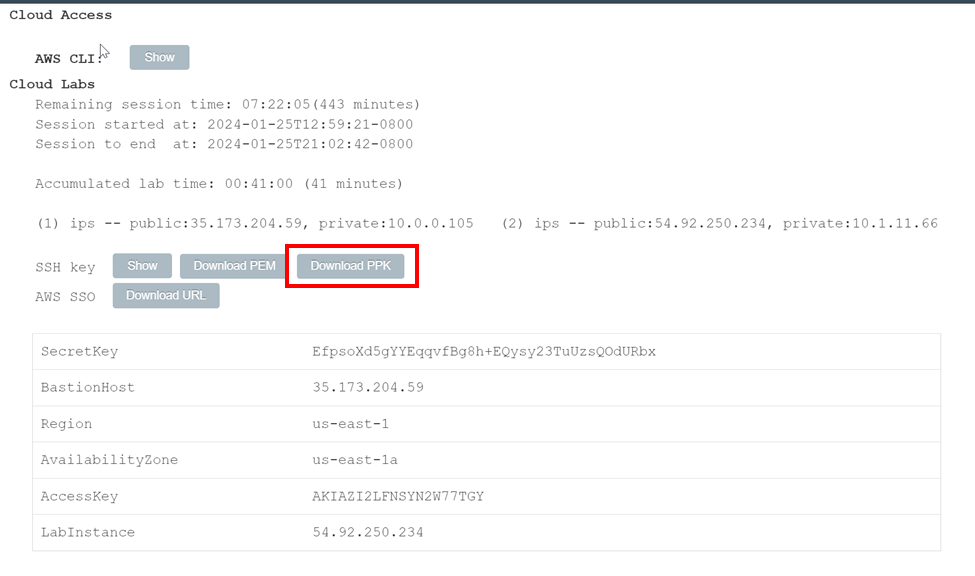


Go back to AWS Academy, open the Details dropdown menu, and press the “Show” button.





From here, download the PPK with the Download PPK button.

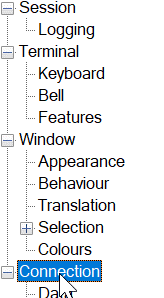


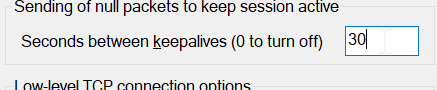


Save the file, then open PuTTY.

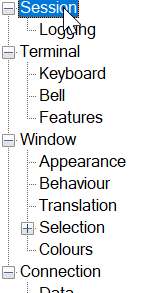


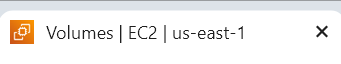
Navigate to Connections and change the Seconds between keepalives to 30





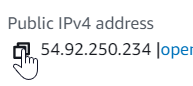
Navigate to Session and return to the AWS console.



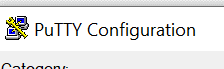


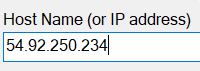
Go to Instances, select the Lab instance, and copy the Public IPv4 address from the bottom pane.



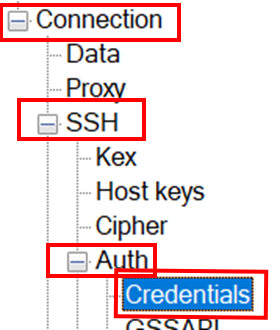


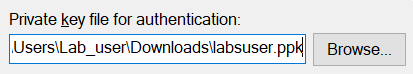
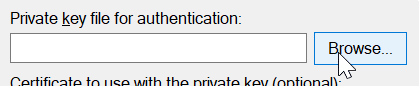
Copy the IP to the Host name in PuTTY.

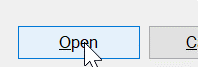


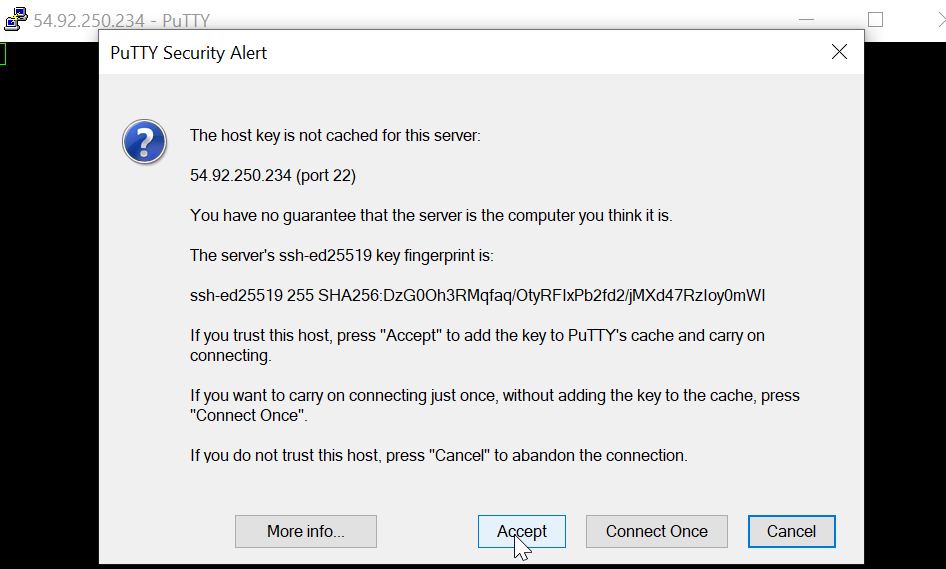


Navigate to Credentials, and set the PPK file as the private key file for authentication, then press the “Open” button to start the SSH connection. Press Accept.







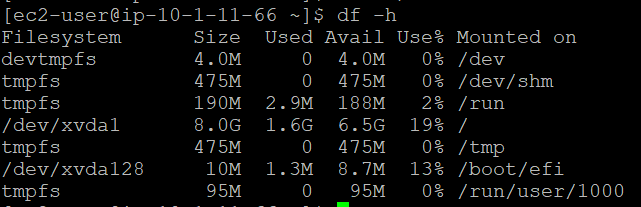


Login as ec2-user, and the authentication will be done by the PPK file.





Use the df -h command to show information about where each file resides on the file system.



Use the sudo mkfs -t ext3 /dev/sdf command to make an ext3 filesystem named /dev/sdf with root permissions.

Use the sudo mkdir /mnt/data-store command to make a director named data-store at /mnt/ with root permissions.

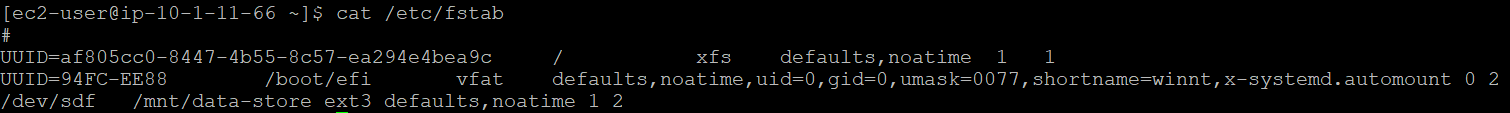


Use the sudo mount /dev/sdf /mnt/data-store to mount the /dev/sdf/ filesystem to the data-store directory with root permissions.

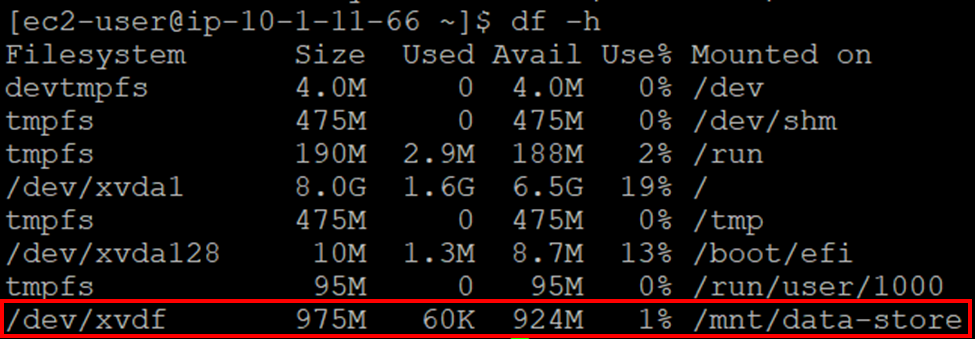
Use the echo "/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2" | sudo tee -a /etc/fstab to append the line "/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2" to the /etc/fstab file with root permissions.



Use the cat /etc/fstab command to print the fstab file to make sure changes have been made successfully.



Use the df -h command again to view changes



Use the sudo sh -c "echo some text has been written > /mnt/data-store/file.txt" command to write the text “some text has been written” to a new file named file.txt in data-store.

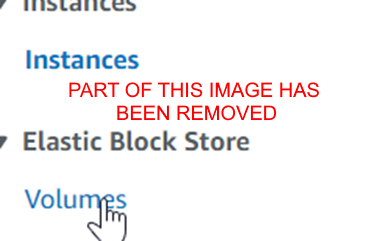


Use the cat /mnt/data-store/file.txt command to print this file and make sure changes have been made successfully.



Return to AWS and go to Volumes.

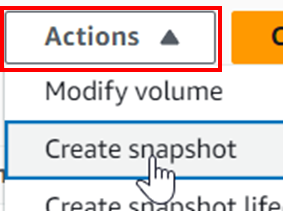


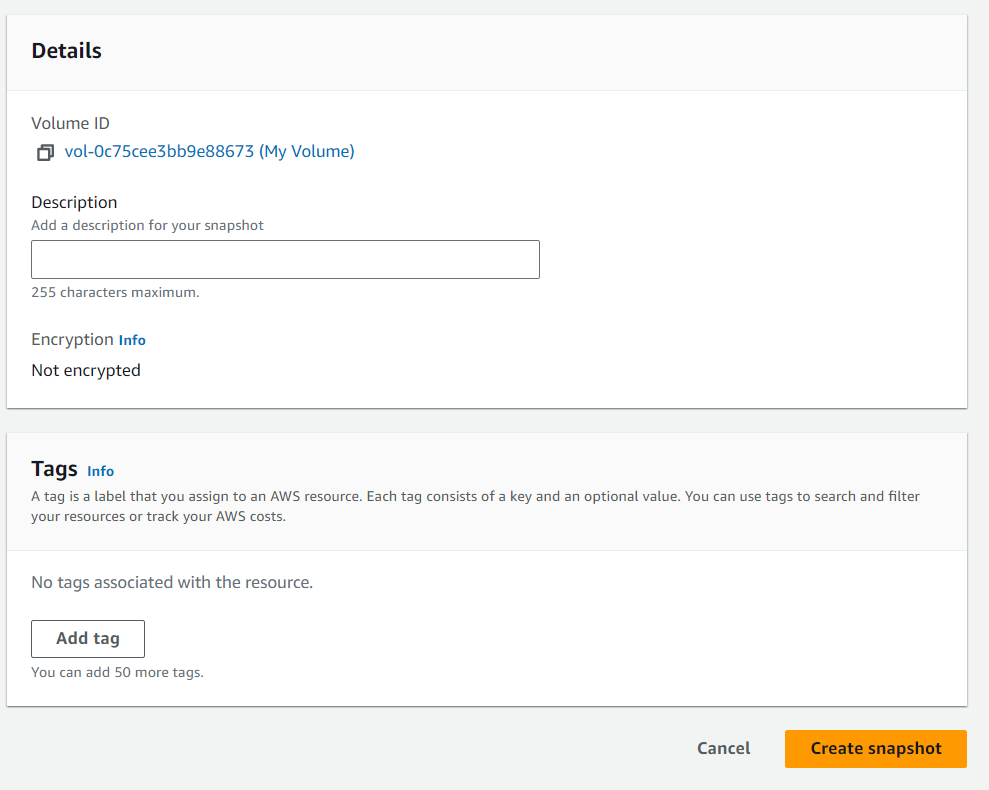


(Part of this image has been removed for visual purpose)

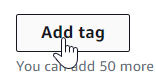
Select the My Volume volume and select “Create snapshot” through the Actions dropdown menu.

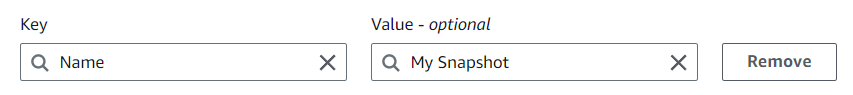


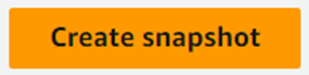




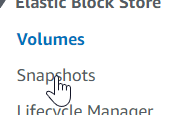
Add a tag with the key Name and the value My Snapshot and press the “Create snapshot” button to create the snapshot.

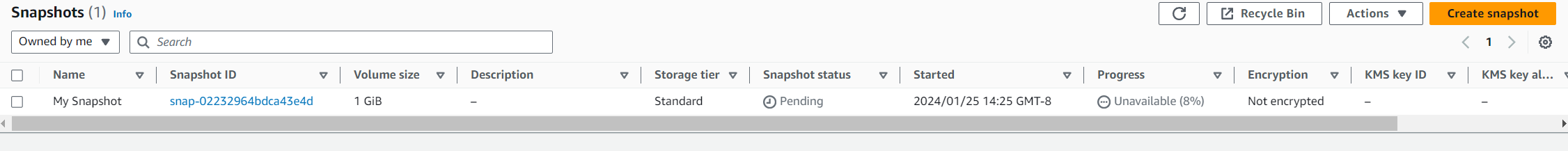






The snapshot will be created and can be viewed from Snapshots in the left sidebar.





Reload till the snapshot’s status is Available.







Return to the EBS console.



Use the sudo rm /mnt/data-store/file.txt command to delete file.txt

Use the ls /mnt/data-store to view non-hidden files in /mnt/data-store to verify file.txt has been removed.

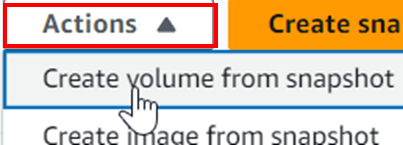


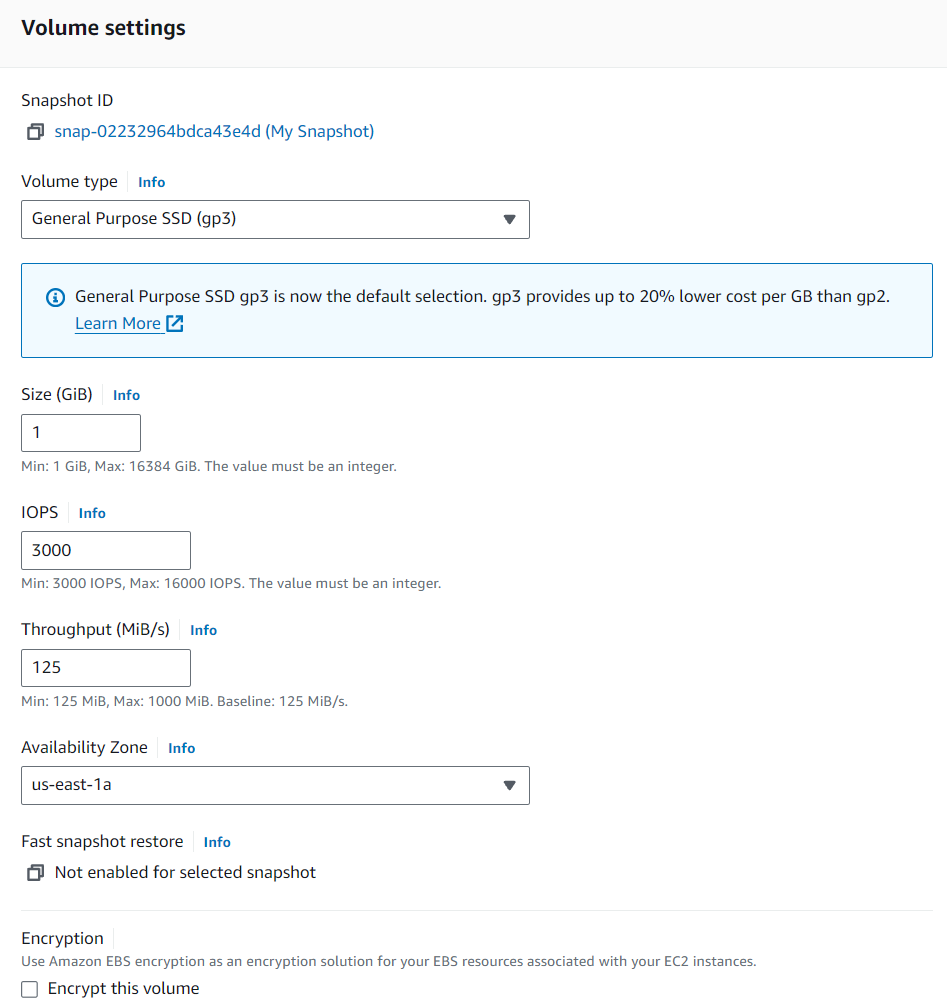
Return to the EC2 console to restore the snapshot.



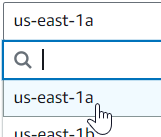
Select the snapshot and from the Actions dropdown menu select Create volume from snapshot.

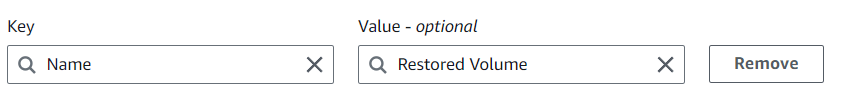






Set the AZ to us-east-1a and add a tag with the key Name and the value Restored Volume.



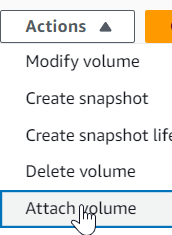


Create the volume.

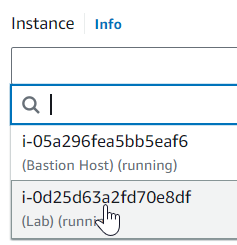


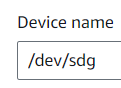
Return to Volumes, select the Restored Volume volume, and attach it.





Select the Lab instance and set the device name to /dev/sdg and attach the volume.





Return to the PuTTY terminal.



Use the sudo mkdir /mnt/data-store2 command to create a folder named data-store2 in /mnt/.



Use the sudo mount /dev/sdg /mnt/data-store2 command to mount the /dev/sdg filesystem to the /mnt/data-store2 folder.



Use the ls /mnt/data-store2/ command to list files in data-store2/ to verify that file.txt has been restored.



Lab 5 – Build a Database Server

# Purpose

In this lab we will be learning how to make a database with AWS, using RDS, and learning how to modify and access it through a web browser with VPC.

# Background Information

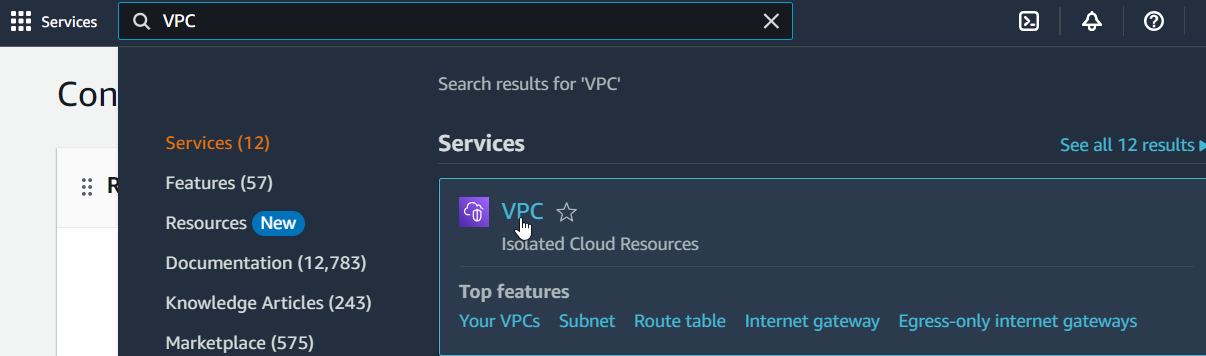
AWS Relational Database Service (RDS) is an AWS service to make and manage relational databases efficiently on the AWS cloud. It supports several well known database engines: Amazon Aurora, Oracle, Microsoft SQL Server, PostgreSQL, MySQL and MariaDB. It is easy to deploy through multiple AZ and has integration with AWS CloudWatch to provide metrics and monitoring abilities.

# Lab Summary

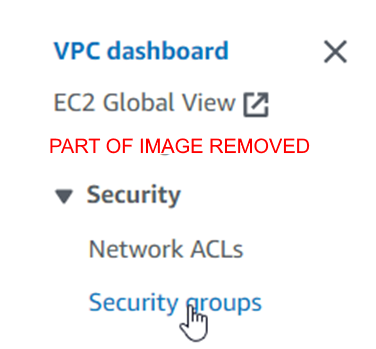
In this lab we make a security group for VPC for our database, then set up a DB subnet group so we connect a database that we also set up to the VPC through RDS. Finally we open up our VPC in the browser and access and perform CRUD operations on the RDS database we created.

# Configurations

Open VPC from search.



Go to Security groups

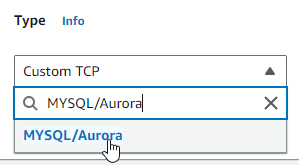


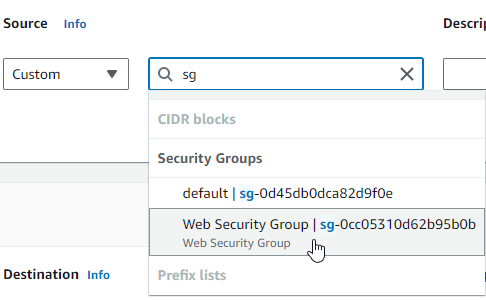
(Part of this image has been removed for visual purpose)

Create a security group with the name DB Security Group, the description Permit access from Web Security Group, and in the VPC Lab VPC, then create an inbound rule.



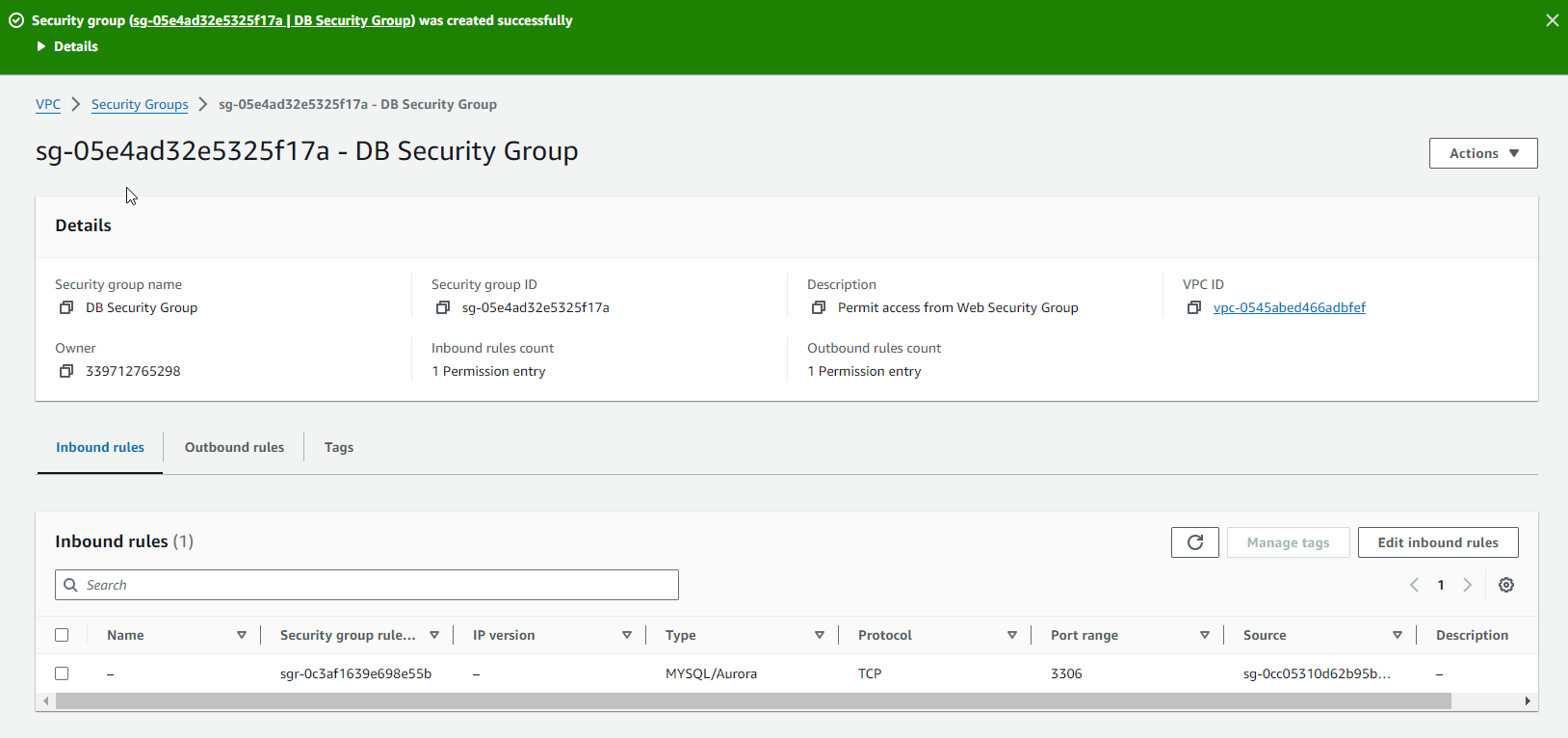
This rule Type should be MySQL/Aurora, and the source should be the Web Security Group.



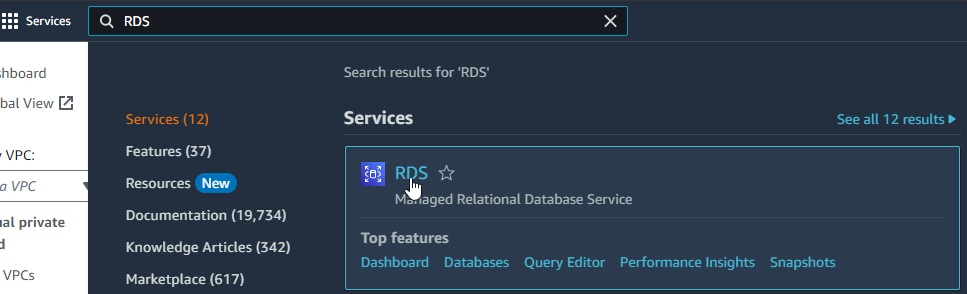


Create the group.

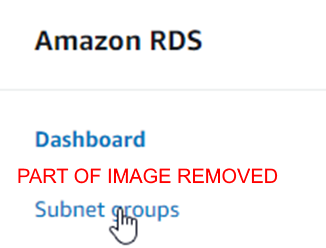




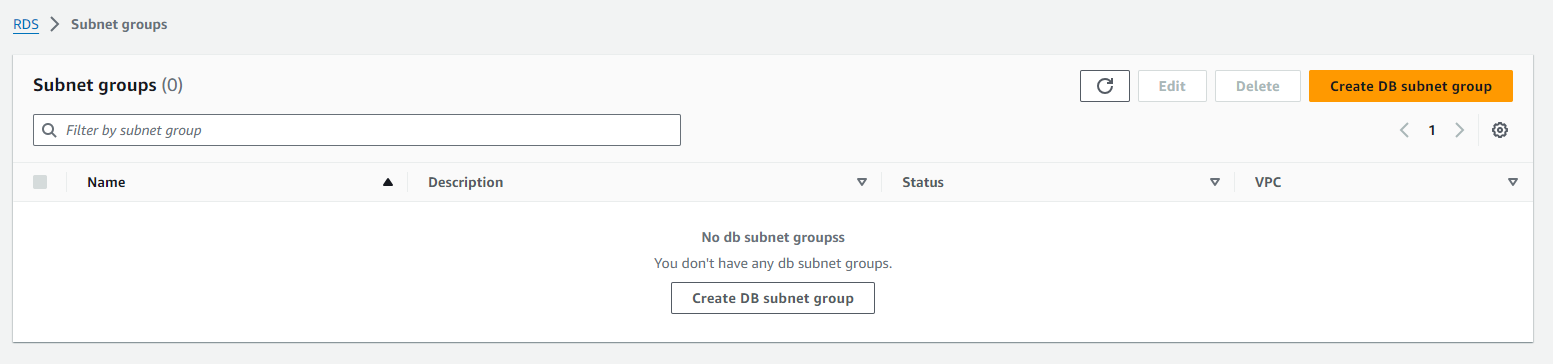
Go to RDS from search.



Go to Subnet Groups.

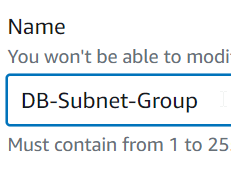


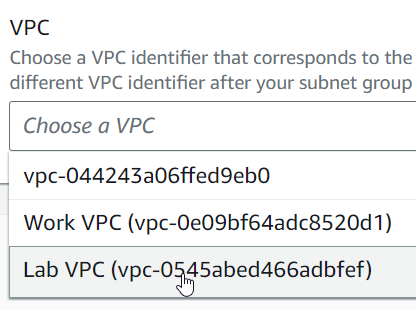
(Part of this image has been removed for visual purpose)



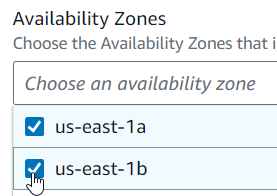
Create a DB subnet group with the name DB-Subnet-Group, the description DB Subnet Group, in the VPC Lab VPC.

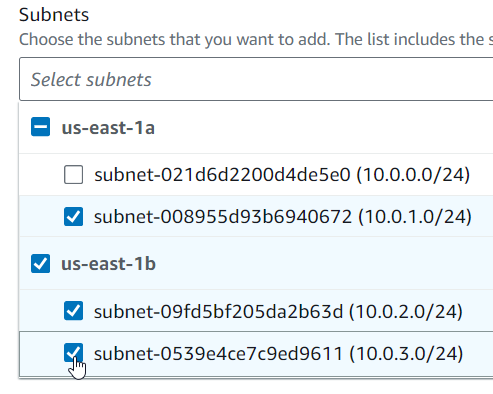




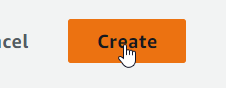


Under Add Subnets, Set the AZ to us-east-1a and b and select subnets from 10.0.1.0/24 to 10.0.3.0/24.

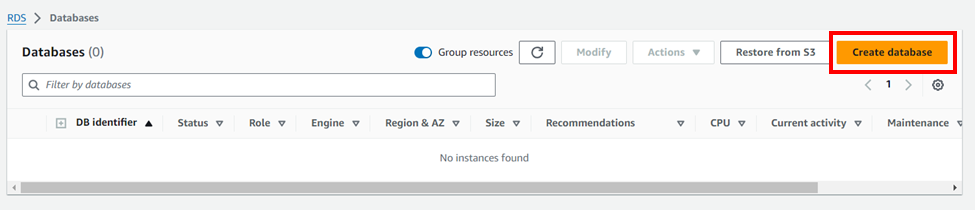


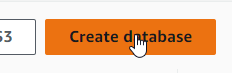


Create the subnet.

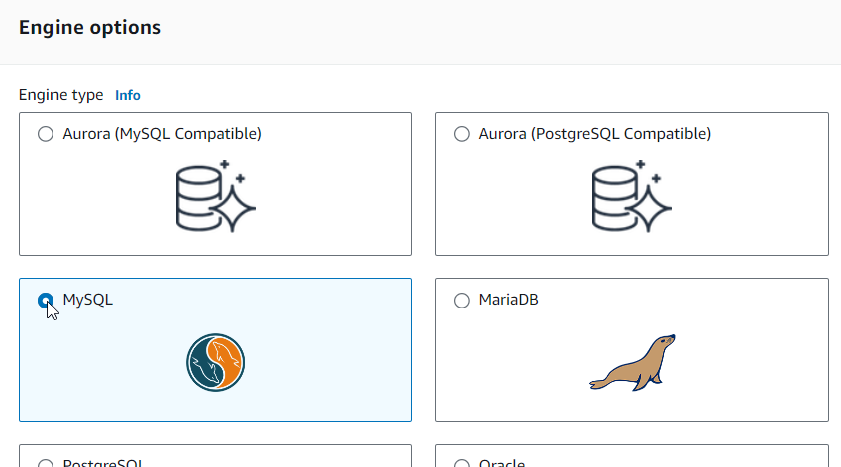


Go to Databases and create a database.

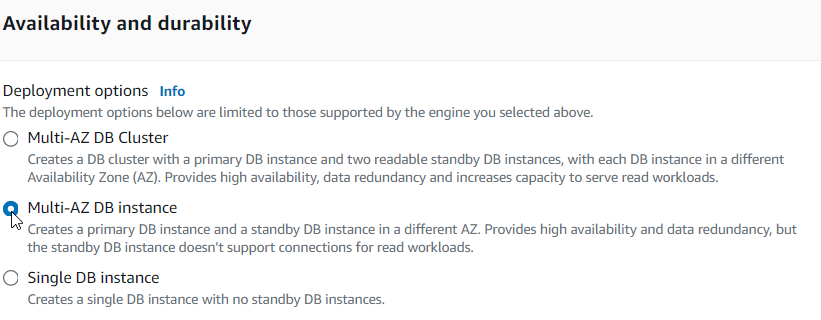




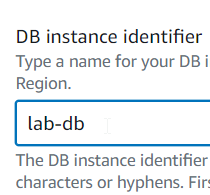
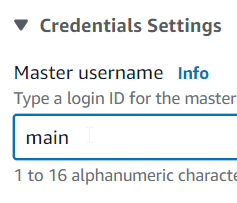
Change the engine to MySQL, set the template to Dev/Test, and make the deployment Multi-AZ DB instance.

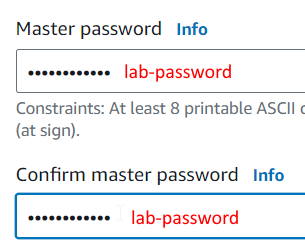




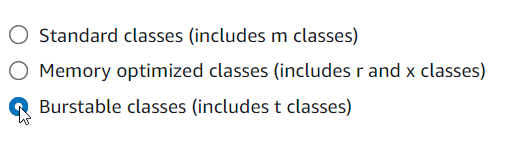


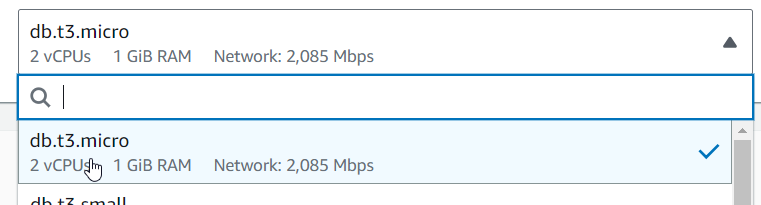
Configure the database identifier as lab-db, the master username as main, and the lab password as lab-password.



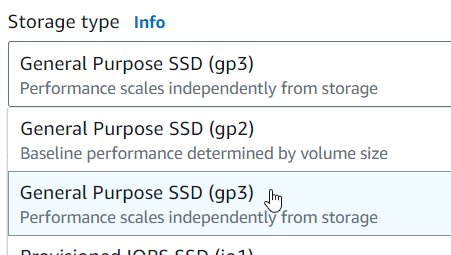


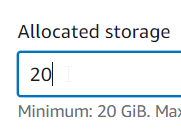
Set the instance class to Burstable classes and db.t3.micro.



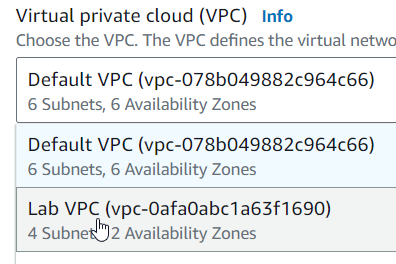


Set the storage type to General Purpose SSD and allocated storage to 20GB.

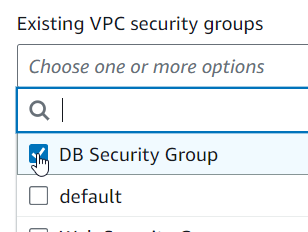




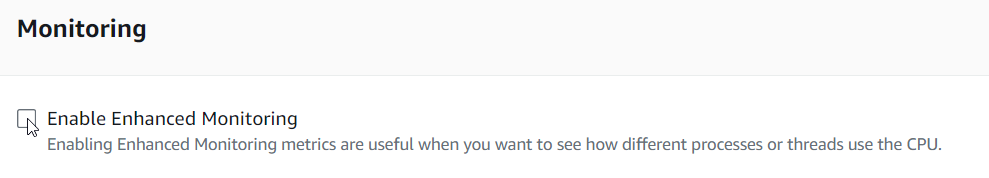
Under Connectivity, change the VPC to Lab VPC.



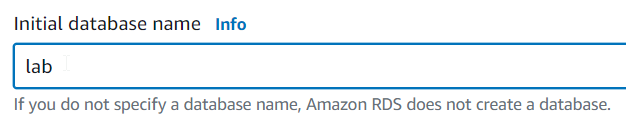
Choose an existing security group and select DB Security Group.

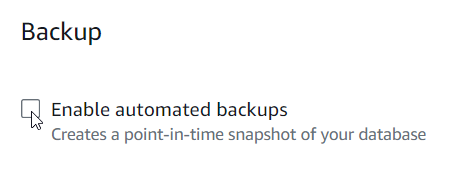


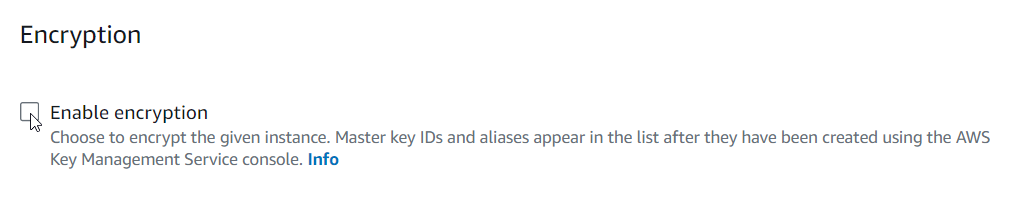
Under Monitoring, disable Enhanced Monitoring.



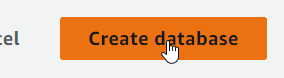
Expand Additional Information, and set the initial database name to lab, and disable automatic backups and encryption.



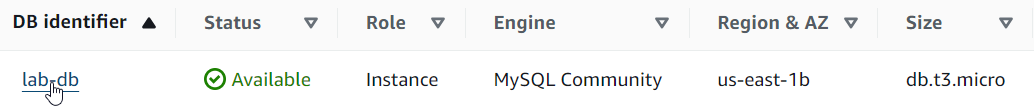


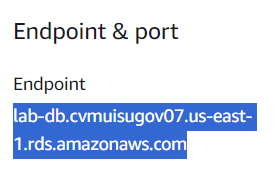
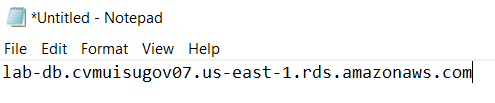


Create the database.



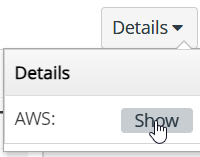
Click the entry’s DB identifier then copy the endpoint under the Endpoint and Port column. Store this in a text editor.

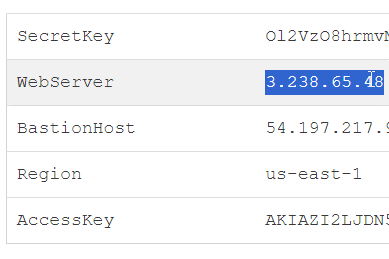




Return to the AWS Academy and show AWS details, then copy the WebServer IP.

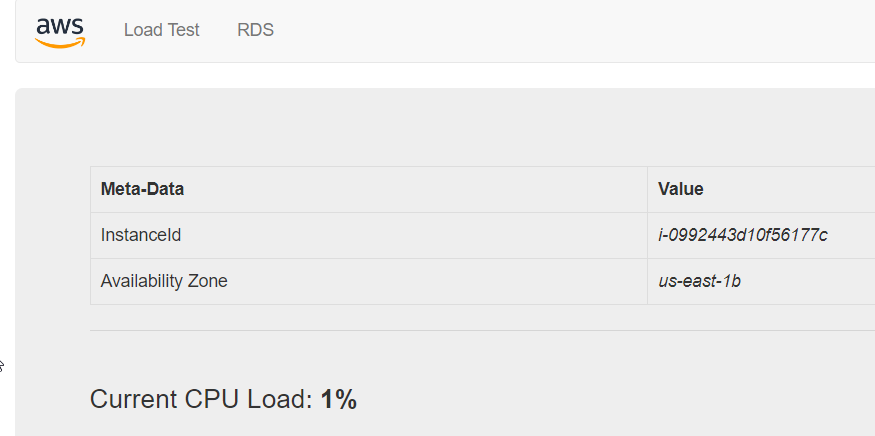






Enter and load the WebServer IP into a new tab.

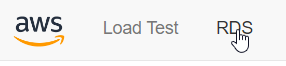


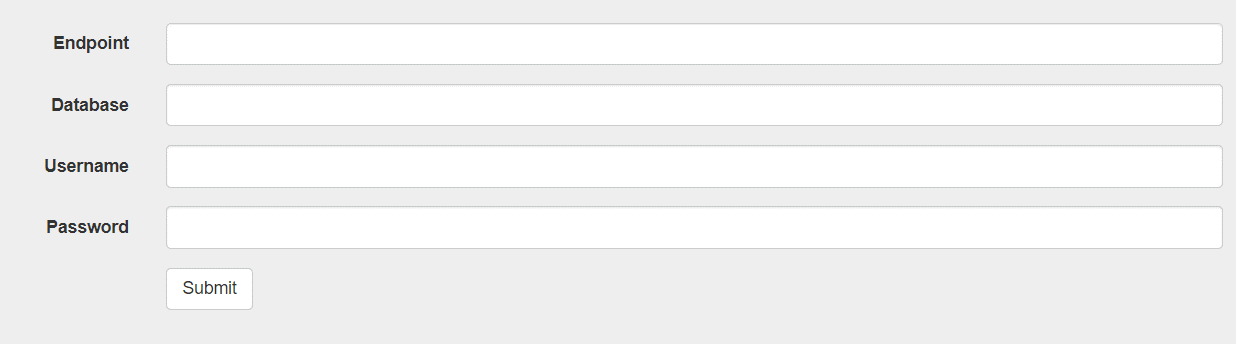


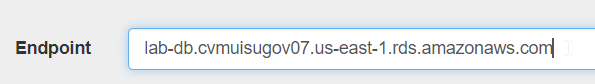
Go to RDS, and paste the endpoint into the endpoint, and enter the information configured earlier:

* Database: lab
* Username: main
* Password: lab-password

Then submit.



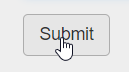




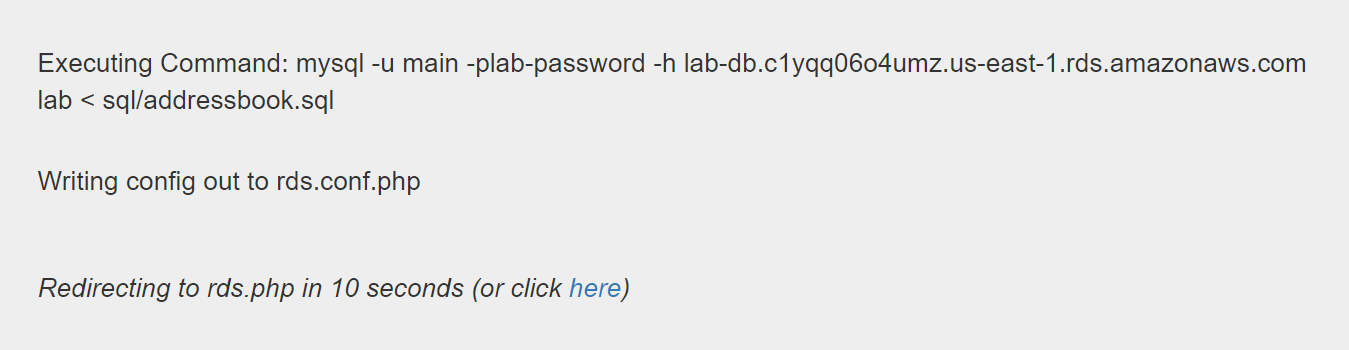


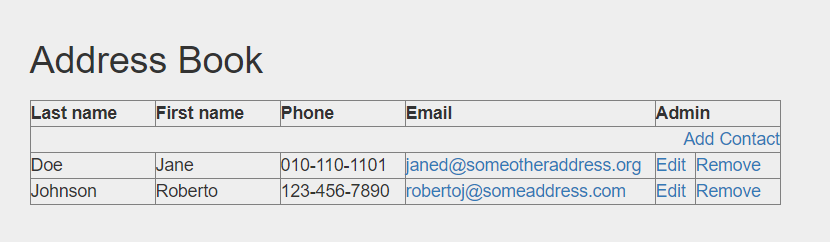






A message will appear essentially saying that the site is loading the SQL database.





Now we can do CRUD operations to this database from the browser.

Create:



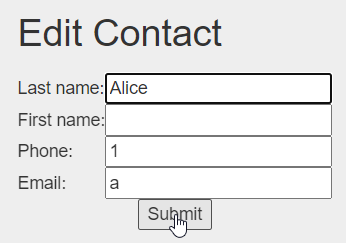


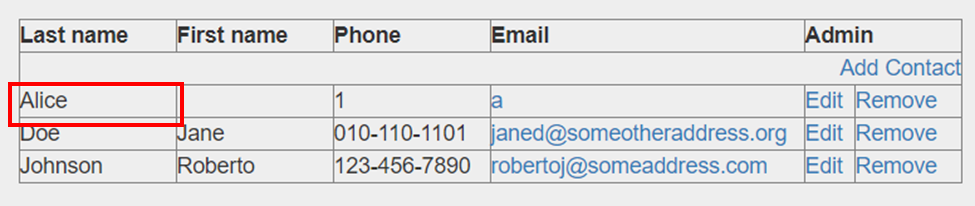


Read: the database can be read.

Update:

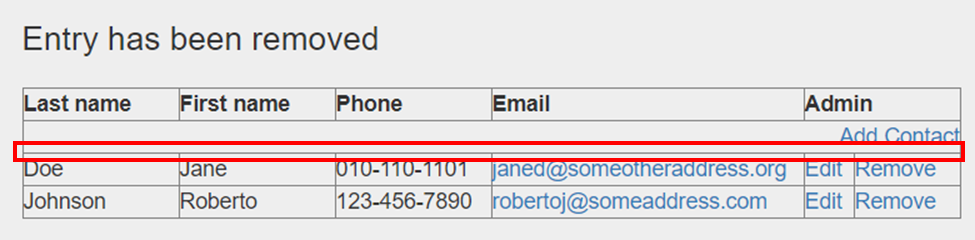






Delete:





Lab 6 – Scale and Load Balance Your Architecture

# Purpose

The purpose of this lab is to learn about AWS Elastic Load Balancing (ELB) to automatically scale resources based on performance needs.

# Background Information

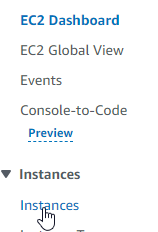
AWS ELB is an AWS service that automatically distributes application traffic among multiple EC2 instances. It improves quality of service with increased repetition and efficient routing of application traffic. It does this using a load balancer and an auto scaling group which creates and terminates EC2 instances as needed and load balances to these instances. It keeps track of how many instances are needed using alarms.

# Lab Summary

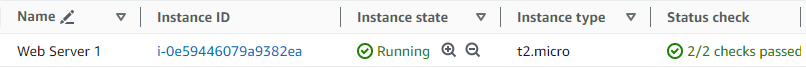
In this lab we create an Amazon Machine Image (AMI), a load balancer, a launch template and auto scaling group, and make new instances automatically scaled, and finally created CloudWatch alarms and use EC2 to monitor performance of infrastructure.

# Configurations

Go to EC2 Instances.



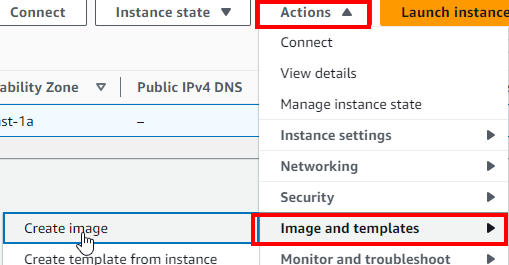
Wait for both status checks to pass on Web Server 1.



Now we will make an AMI for Web Server 1.

Select Web Server 1 and from Actions select Image and templates and Create image.

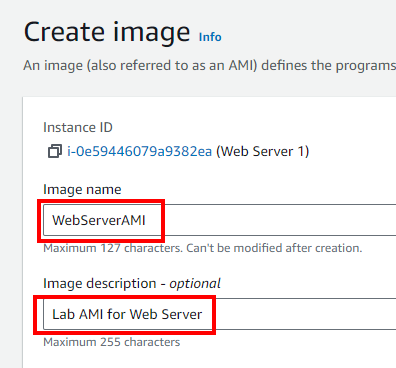




Configure as follows:

* Name: WebServerAMI.
* Description: Lab AMI for Web Server.

Create the image.







Now we will create a load balancer so traffic can be distributed among multiple EC2 instances and AZ.

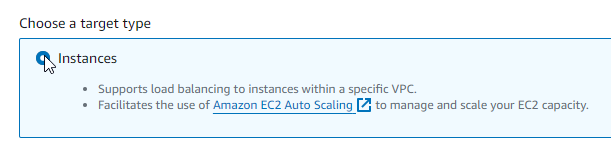
Scroll down and go to Target Groups in the left sidebar(these define where to send traffic from load balancer).

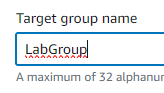
Create and configure a new target group as follows:

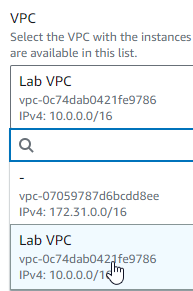
* Target type: Instances.
* Name: LabGroup.
* VPC: Lab VPC.

Press Next.











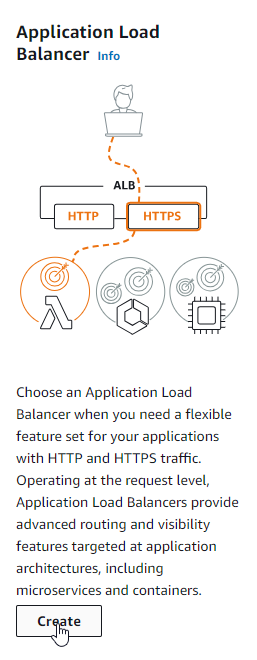
Scroll down and create another target group.



Go to Load Balancers and create a new load balancer.



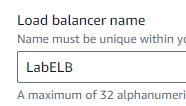
Create an application load balancer.

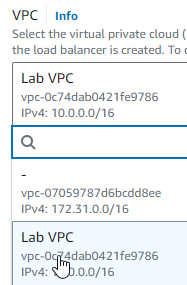


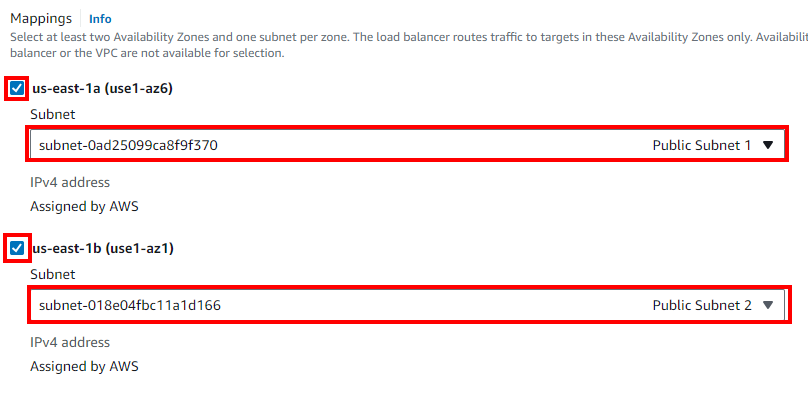
Configure as follows:

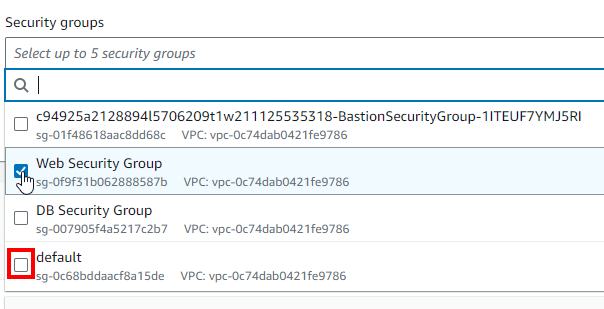
* Name: LabELB.
* VPC: Lab VPC.
* Select both AZ and select the Public Subnets for both.
* Select the Web Security Group security group and deselect the default security group.
* Select the LabGroup target group for the default action for the HTTP:80 listener.

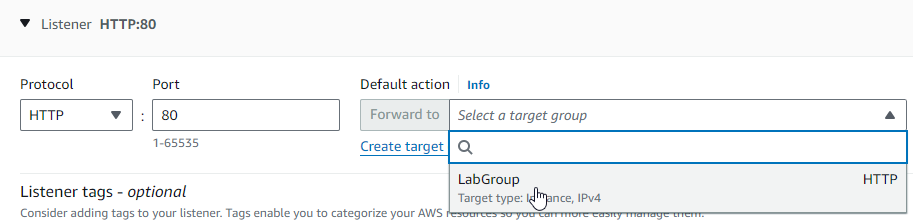
Scroll down and create load balancer







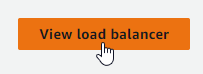


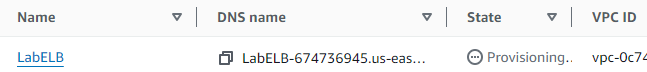






View the load balancer.





We do not need to wait for the load balancer’s state to be ready.

Go to Launch Templates and create a launch template.

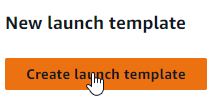
Configure as follows:

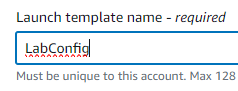
* Name: LabConfig.
* Check “Provide guidance to help me set up a template that I can use with EC2 Auto Scaling”.
* Go to My AMIs and make sure Web Server AMI is selected.
* Instance Type: t2.micro.
* Key pair name: vockey.
* Choose Select existing security group.
* Security Group: Web Security Group.

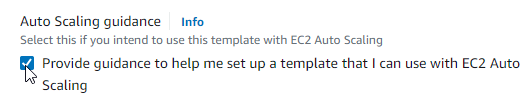
Expand Advanced details

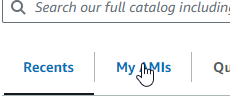
* Detailed CloudWatch monitoring: Enabled

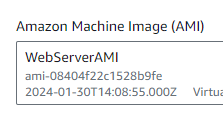
Create the launch template.

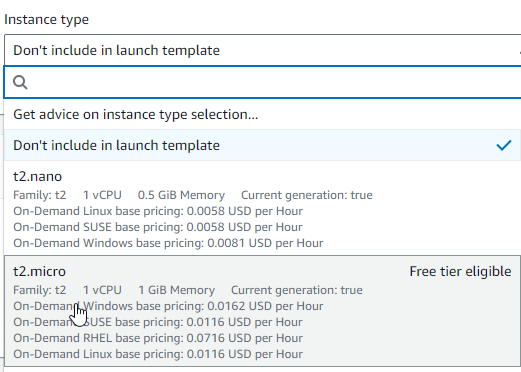


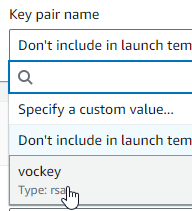


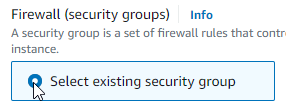


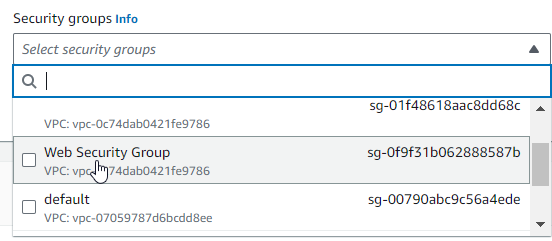


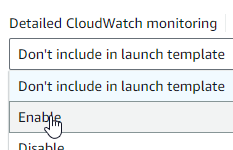












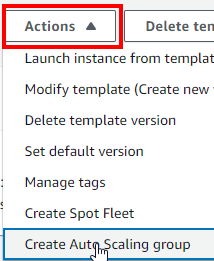




Go to LabConfig.



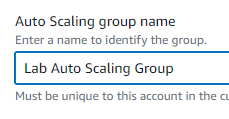
Create an auto scaling group from Actions.

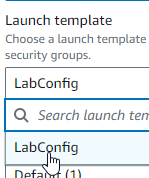


Configure as follows:

* Name: Lab Auto Scaling Group.
* Launch Template: LabConfig.

Press Next.



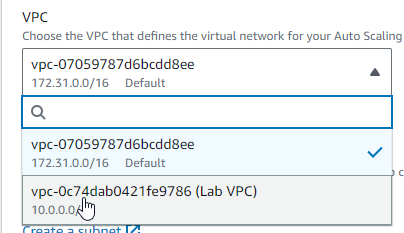


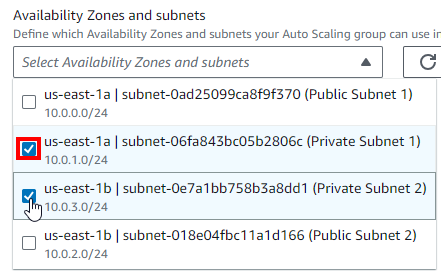


Configure as follows:

* VPC: Lab VPC.
* AZ and subnets: Select Private Subnets

Press Next.







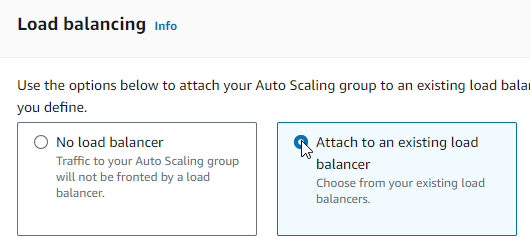
Configure as follows:

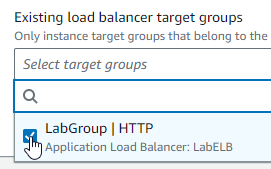
* Attach to existing load balancer
* Existing load balancer target groups: LabGroup.

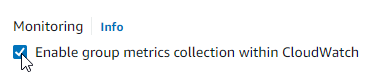
Scroll to Additional Settings

* Check “Enable group metrics collection within CloudWatch”.

Press Next.





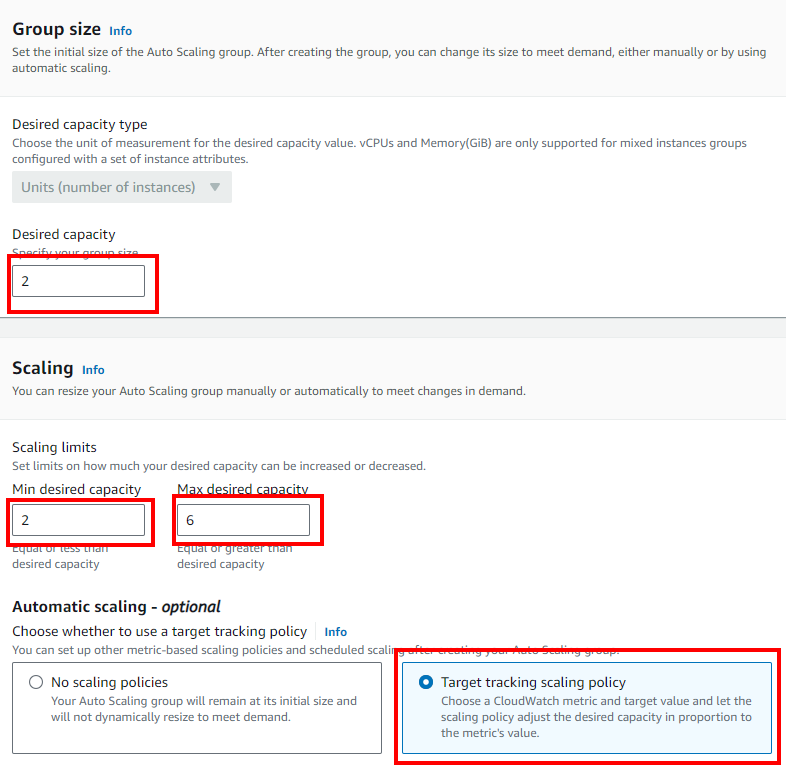


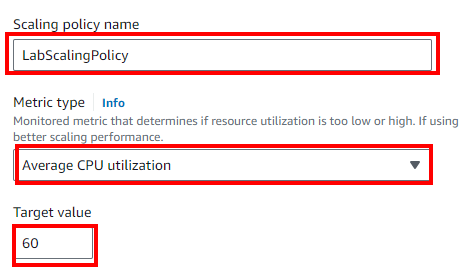


Configure as follows:

* Group size:
  + Desired Capacity: 2.
  + Min Capacity: 2.
  + Max Capacity: 6.
* Automatic scaling: Target tracking scaling policy.
  + Name: LabScalingPolicy.
  + Metric Type: Average CPU utilization
  + Target value: 60

Press Next.



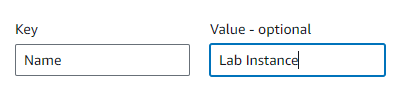




Press Next.



Add a tag named Name with value Lab Instance.



Press Next.



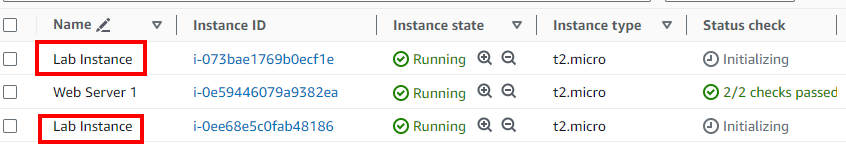
Review settings and create auto scaling group.



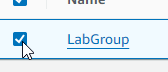


Now to check to make sure everything is working.

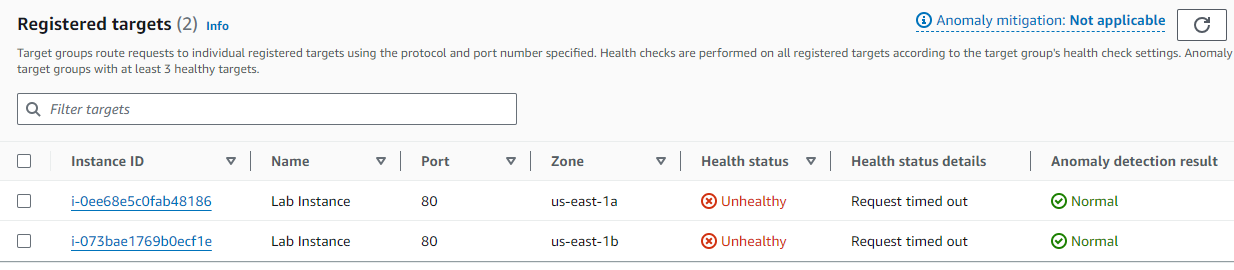
Go to Instances, there will be two new instances made by the auto scaling group.



Go to Target Groups and select LabGroup.



There should be two targets in the Targets tab.

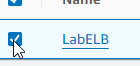


Refresh until the targets are healthy.

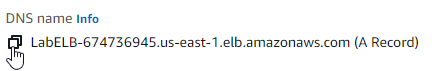
A screenshot of a computer

Description automatically generated

Go to Load Balancers and select LabELB.



Copy DNS name and load in new tab.





A screenshot of a computer

Description automatically generated

If the page loads, the load balancer worked.

Finally, we will test the auto scaling.

Go to AWS CloudWatch.

A screenshot of a computer

Description automatically generated

Go to All alarms in the Alarms category. There are two alarms made by auto scaling to maintain the CPU load.

A screenshot of a computer

Description automatically generated

Notice the conditions.

A screenshot of a computer

Description automatically generated

Go to EC2 and navigate to Auto Scaling Groups.

Select Lab Auto Scaling Group, go to the Automatic Scaling tab, and select LabScalingPolicy from the bottom pane.

A screenshot of a computer

Description automatically generated

A red square with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

From the Actions dropdown menu in the bottom pane, select Edit, and modify the target value to 50. Press Update.

A screenshot of a computer

Description automatically generated

A blue rectangle with black text

Description automatically generated

A close up of a sign

Description automatically generated



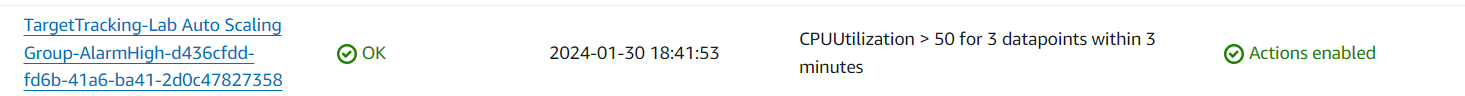
Go back to CloudWatch All alarms.

A screenshot of a computer

Description automatically generated

The CPUUtilization has been reduced to 50, as expected.

Note how the alarm’s state is OK.



Go back to the AWS website and select Load Test.

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

Return to CloudWatch, and shortly the higher alarm will be in alarm, and the lower alarm will be OK.

A group of numbers and letters

Description automatically generated

If we go to EC2 instances, we can see more instances have been created to load balance with the higher load needed.

A screenshot of a computer

Description automatically generated

Finally, to terminate this instance, select the Web Server 1 instance and select Terminate instance from Instance state and terminate it.

A screenshot of a computer

Description automatically generated

A screenshot of a computer error

Description automatically generated

Conclusion

# Problems

I did not encounter any problems that significantly affected my progress.

# Conclusion

These labs continued to teach essential AWS concepts and services to new AWS customers. These labs taught coursetakers about the EBS, RDS, and ELB services and how they function to improve workflow. The labs were also a lot of busy work, but they were effective in getting a good grip on using AWS.