CLOUD COMPUTING LABS SECTION I

Daniel Ge
Period 678

CISCO

CCNP LAB REPORT

Purpose

To practice the AWS concepts covered in the AWS Cloud Foundations course and train to use the AWS system in our companies, we must go through mock scenarios where we implement AWS Cloud solutions to address business goals our company may have. Through these three labs, we practice using the AWS Management Console accessed through a web browser to achieve setting up many services or networks, such as virtual machines, virtual networks, and security policies for a hypothetical company. At the end of these exercises, we are better prepared to replicate these steps to real-life business scenarios.

Background Information

Under the AWS Shared Responsibility Model, although AWS will protect the security of the cloud, including the computing resources and servers, it is the user's responsibility to ensure security in the cloud by protecting elements like their own data and configuring firewalls. The AWS IAM (Identity and Access Management) Service provides a framework to achieve this for AWS users. AWS IAM is split into two halves corresponding to the two pillars of web security: authorization and authentication.

With AWS IAM authentication, we may verify that users accessing their AWS account are who they claim to be. To achieve this, IAM allows us to configure which login credentials employees are required to set, from passwords to access keys. We additionally have the ability to apply MFA (multi-factor authentication) to combine multiple authentication methods during log-in.

Through AWS IAM authorization, we may configure users, groups, and roles. Users and groups of users are usually static, but roles can be applied to users or applications temporarily. After these are defined, we may then configure policies to be applied to these subjects. The policies will rule what the users, groups, and roles may complete.

Using AWS IAM, administrators can configure access permissions to the company's web resources. Additional tools may be applied to encrypt data at rest and in transit. Given that Amazon continues to properly secure the AWS cloud itself, with the correct security settings, intruders and malicious actors will be prevented from accessing, viewing, or altering company resources.

Just as breakthroughs in communications allowed for the construction of networks, where devices with 32-bit IPv4 or 128-bit IPv6 addresses may communicate with each other, we may build networks on AWS as well using Amazon VPC (Virtual Private Cloud). VPCs are networks run on AWS logically isolated from VPCs of other accounts and dedicated to one AWS account. Unlike traditional networks, however, AWS allows us to deploy VPCs in one region but multiple availability zones within the region.

Within each availability zone, we may deploy subnets. Subnets are divided into public and private types. Public subnets may communicate with the public internet through an IGW (internet gateway) while private subnets can be accessed through a NAT gateway. AWS allows us to construct routes between the networks and IP addresses of applications within the networks. There also exist additional services such as VPC sharing, VPC peering, or VPC site-to-site VPNs to better connect VPC subnets or VPCs themselves, but they are beyond the scope of this lab.

One of the most common uses of Amazon VPCs is using them to launch Amazon EC2 instances. Since VPCs provides the networking environment and security configurations beyond that of what EC2 itself can provide, we isolate EC2 instances in the VPCs and assign them with IP addresses to use them. Just as we cannot use a real-life computer to provide a service to customers without connecting it to a network, we cannot use virtual EC2 machines without having a VPC network.

To handle the IT load of running a large organization including businesses or research institutes, we often need to employ the use of large data servers. However, given the difficulty and cost in maintaining, scaling, and running these servers independently, Amazon AWS offers the opportunity to migrate away from traditional company data centers to virtual machines on the cloud. Using Amazon EC2 (Elastic Compute Cloud), we may use AWS virtual machines' computing power to run services like web servers, gaming servers, or file servers for our users.

Amazon EC2 instances are launched from AMIs (Amazon Machine Images) based on operating systems like Linux or Windows and packages them in such a way they can be used for AWS. Users then have the option to configure instance size, with various levels of memory, CPU processing power, disk space, and network performance. To optimize costs, users should consider their priorities and select appropriately sized instances.

Additional configurations that users will set when creating and using an Amazon EC2 instance includes network settings (selecting the VPC), IAM role, user data (what the virtual machine contains), storage options (storage volume size), tags, security groups, and key pair.

Lab Summary

In Lab 1, the Introduction to IAM lab, we will open a sample AWS account with 3 pre-existing IAM Users and Groups set with pre-existing policies: EC2-Admin, EC2-Support, and S3-Support. These groups are respectively configured with permissions to modify EC2 instances, view EC2 instances, and view S3 information. We will then configure three users with the correct permissions, then log in through their usernames and passwords to verify that each user is configured to be able to do precisely what they are meant to do.

In Lab 2, we practice building and using a VPC (Virtual Private Cloud) on AWS. We follow steps to configure public and private subnets in the VPC across multiple Availability Zones, including NAT Gateways and Internet Gateways to allow for communication with the VPC. We will additionally configure the VPC with other necessary components including

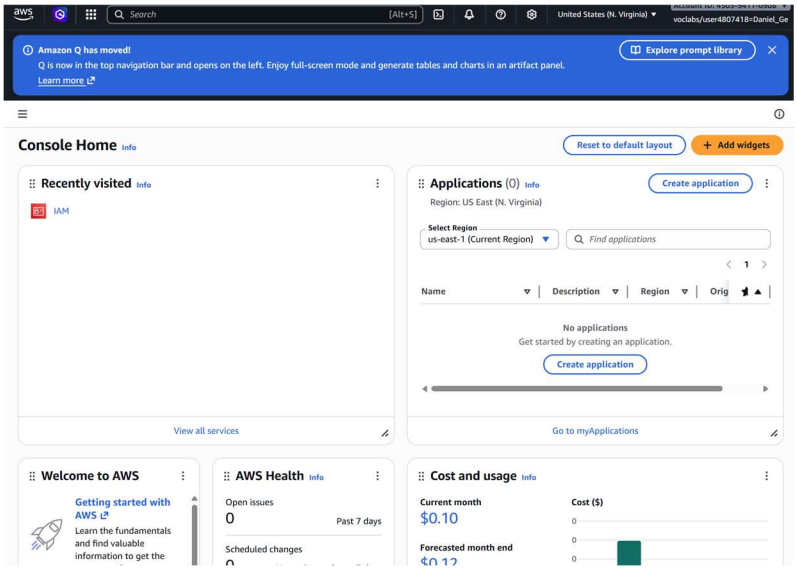
security groups. In the end, we will practice using the VPC by launching an EC2 instance into the VPC.

Lab 3 focuses on launching an EC2 instance. We will launch the instance with the appropriate configurations in security to allow for HTTP traffic and practice monitoring the instance. Afterwards, we will experiment with features such as stopping the EC2 instance to adjust its size or stop protection, which prevents the instance from getting stopped on accident.

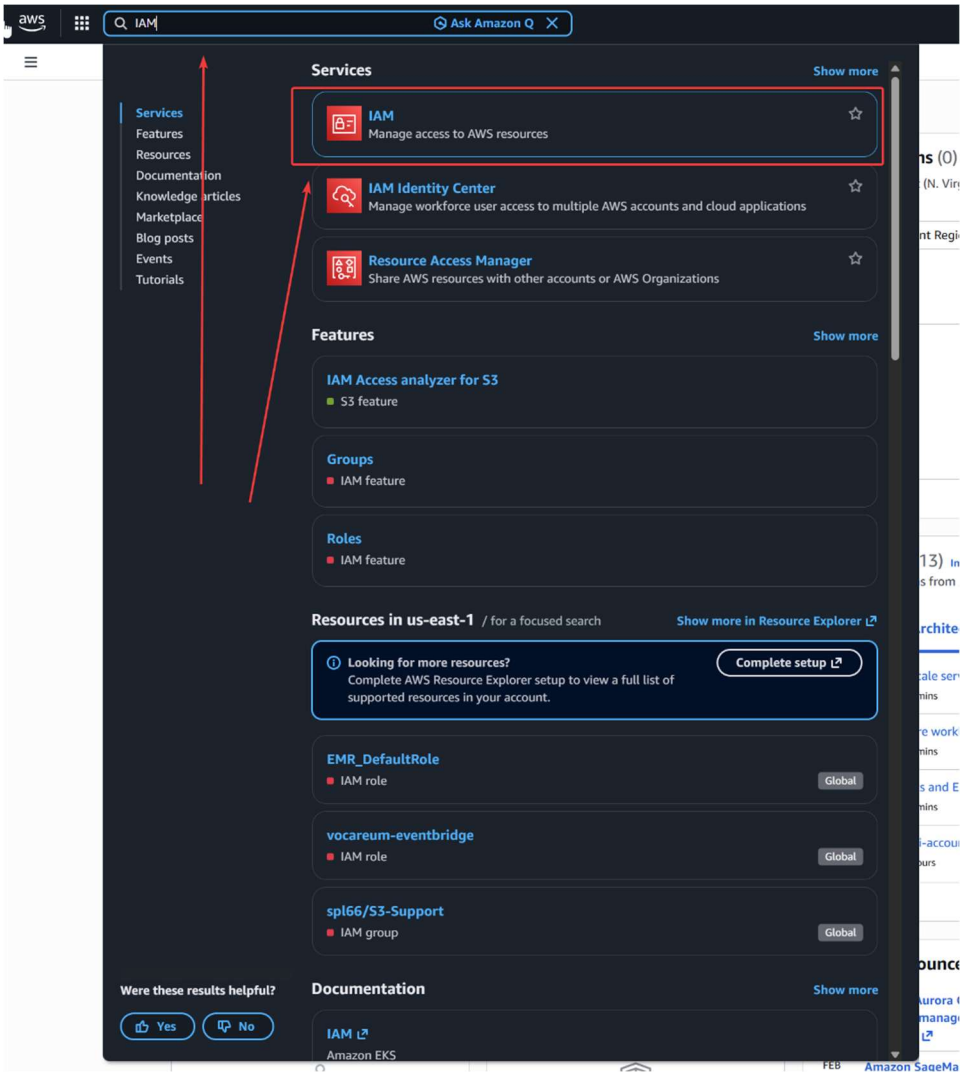
Lab Procedure

Introduction to IAM (Lab 1)

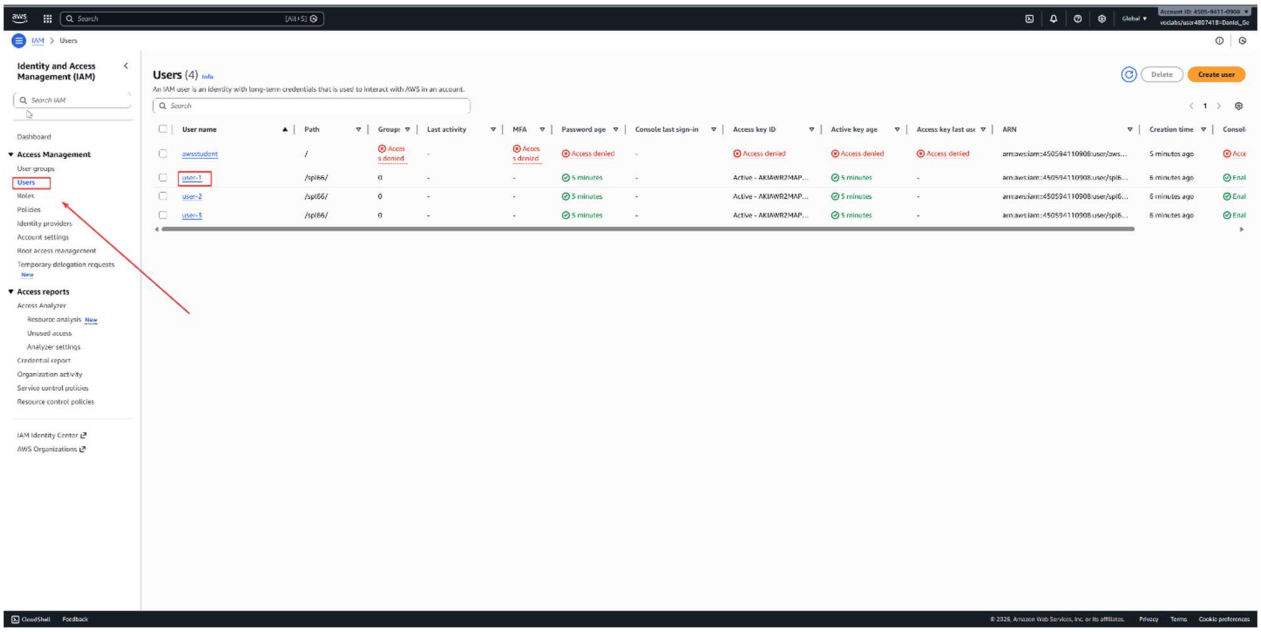
Start the lab and enter the main menu



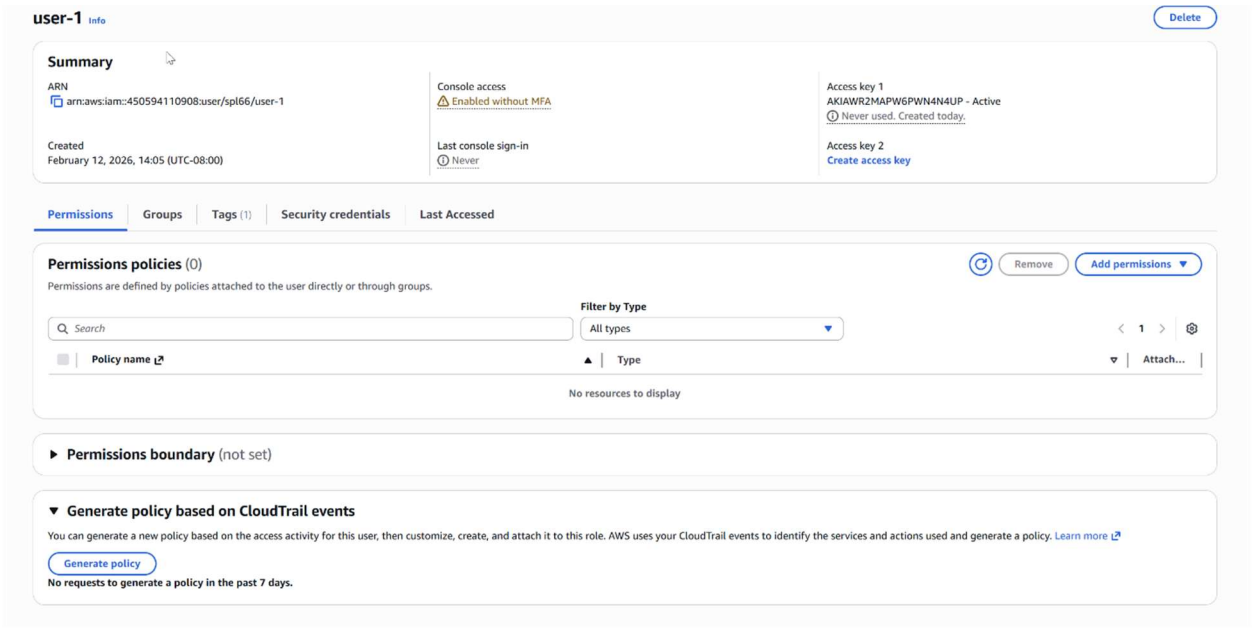
Search “IAM” in the search bar and click the IAM option to open the IAM menu



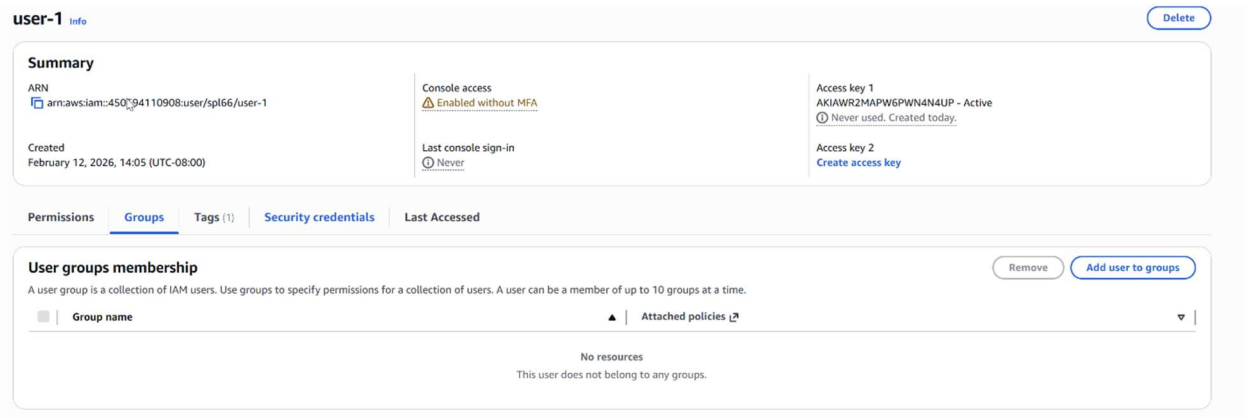
Click “Users” on the left menu and observe how there are three existing users currently. Click “user-1” the first option



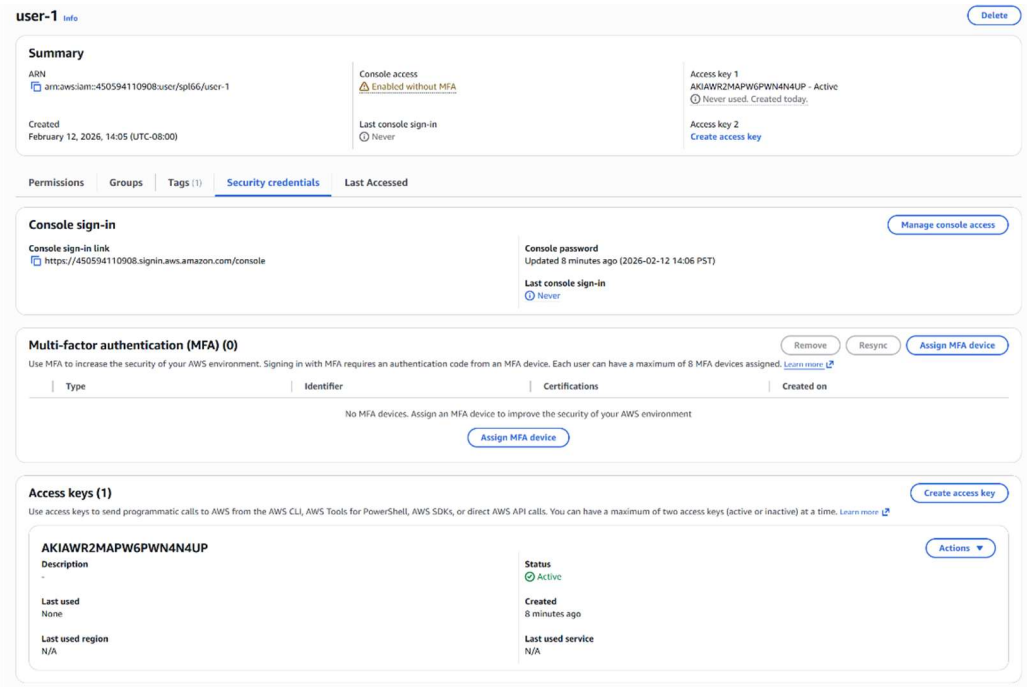
Observe that user-1 does not have any permissions



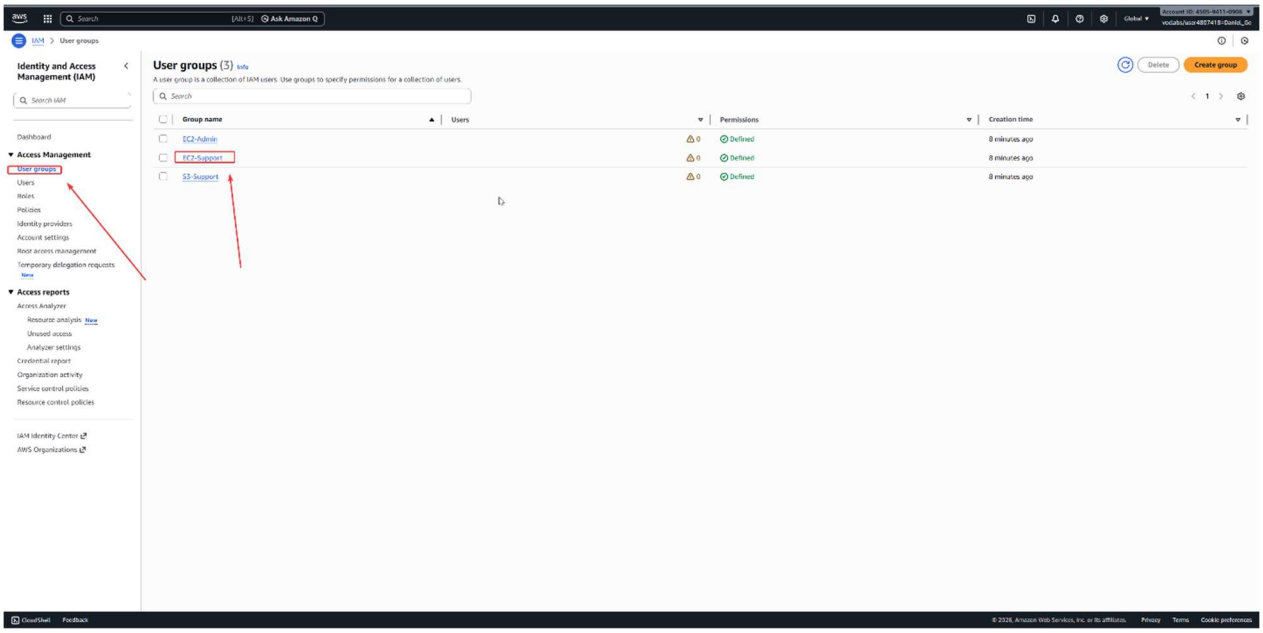
Click the “Groups” tab to observe that user-1 is not a member of any group



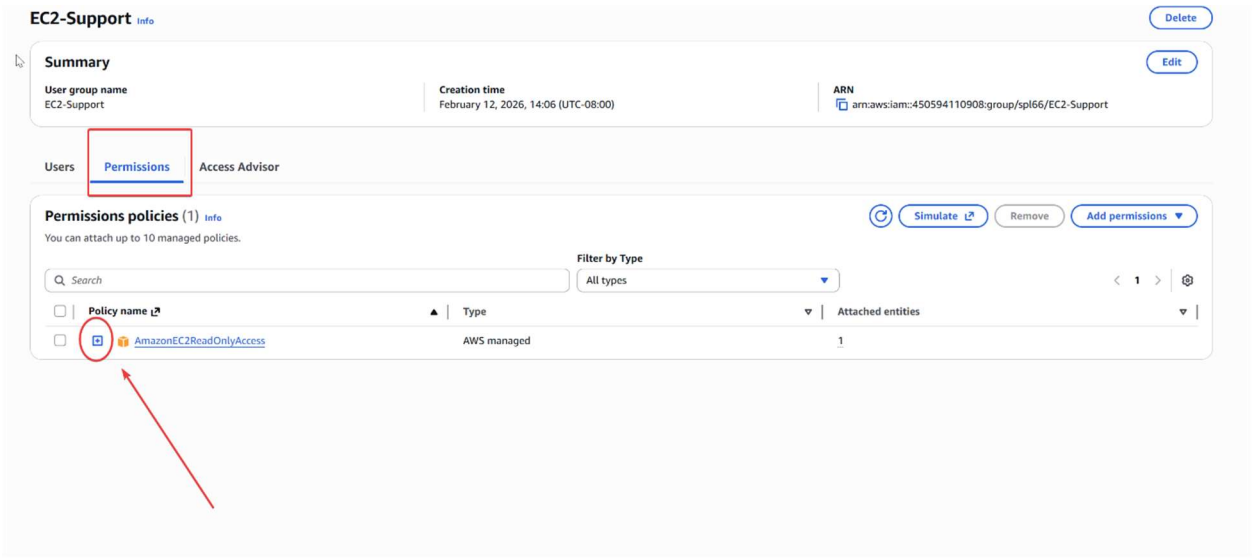
Click the “Security credentials” tab to observe that user-1 has a console password



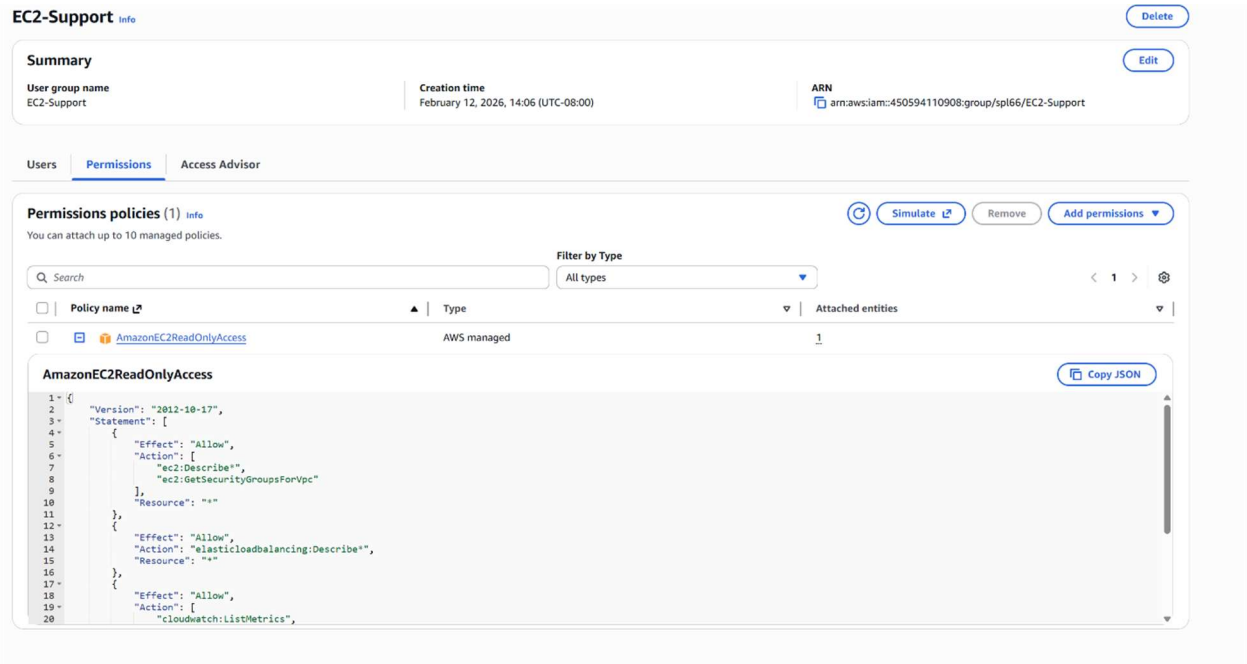
Click the User Groups option on the left menu to find the three pre-made groups



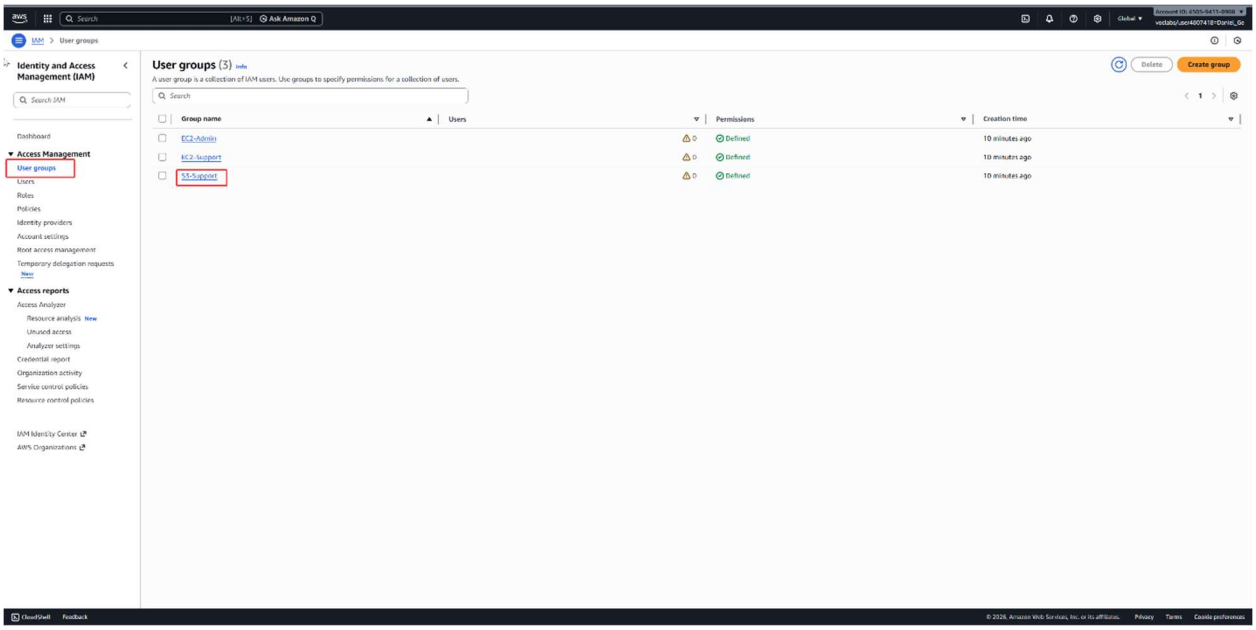
Click EC2-Support group, then the tab Permissions. Click the blue plus next to "AmazonEC2ReadOnlyAccess"



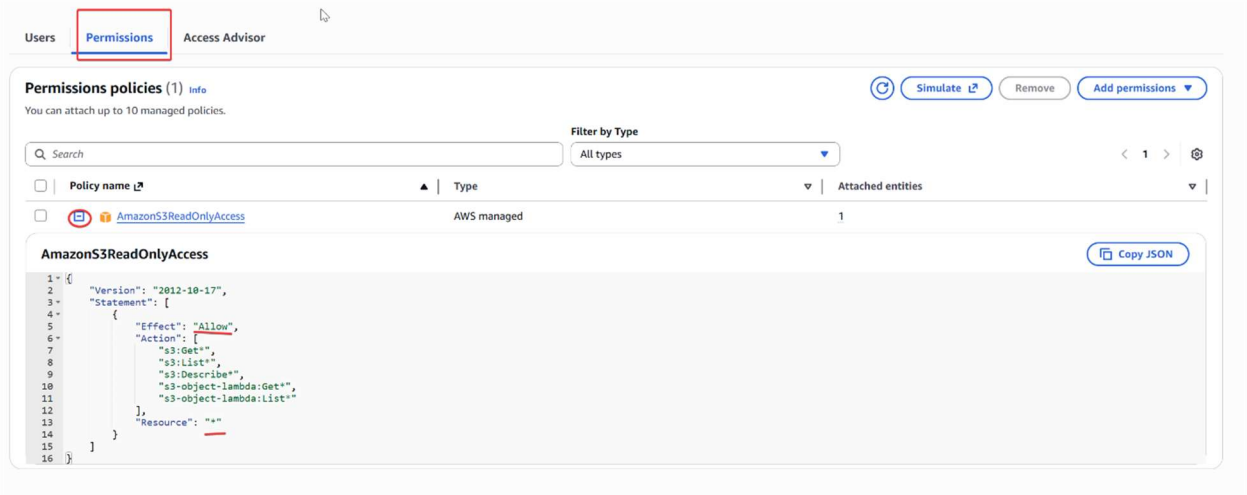
The AmazonEC2ReadOnlyAccess policy has been configured with multiple Allow and Deny statements which allow the user to read EC2 information, but denies them the ability to affect the operation



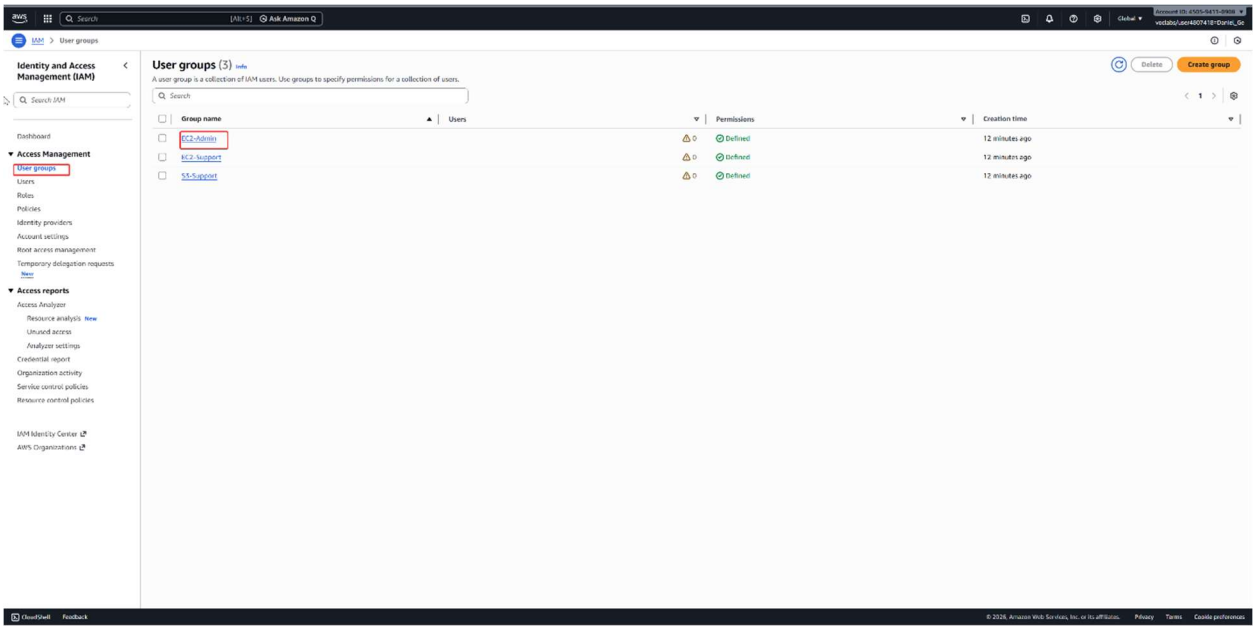
Return to the User Groups tab, and now click S3-Support



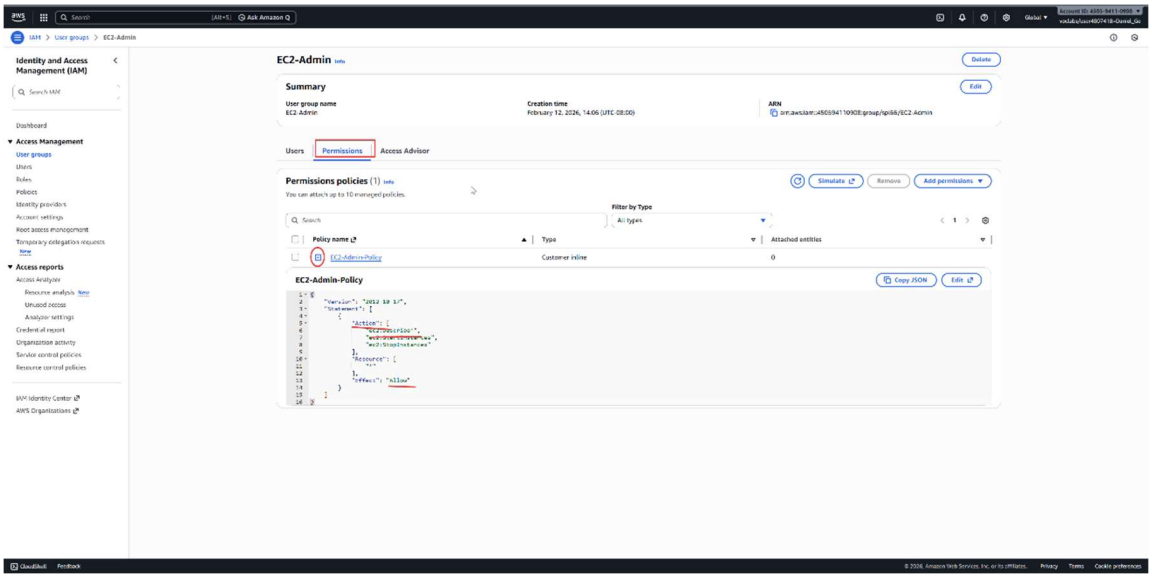
Click Permissions then the blue square to expand and view the configuration for “AmazonS3ReadOnlyAccess”



Return to the User Groups tab, and now click EC2-Admin

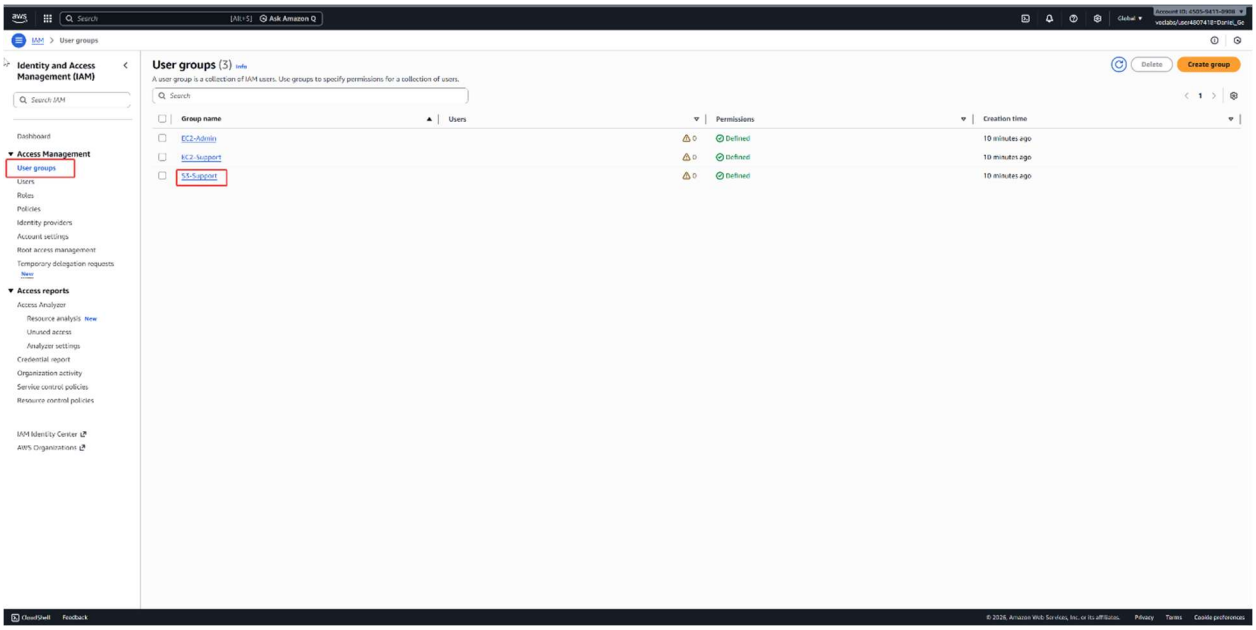


Click Permissions then the blue square to expand and view the configuration for “EC2-Admin-Policy”. This group is configured with an Inline Policy which allows the user to stop and start the EC2 instances.

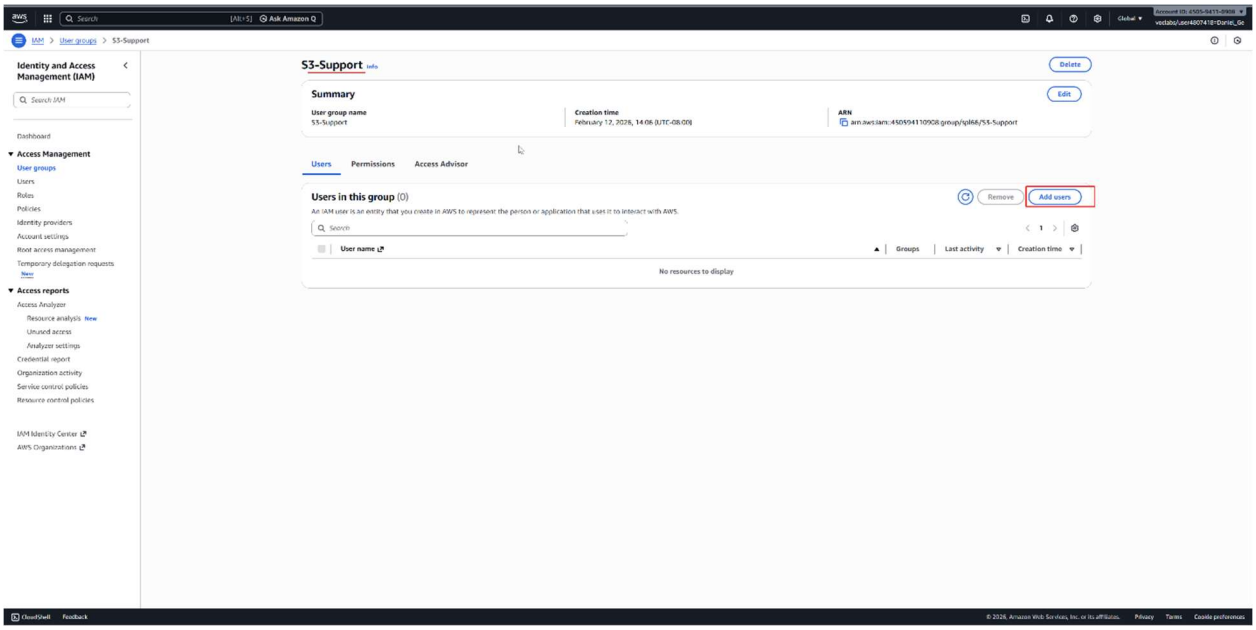


Configuring User Permissions through IAM (Lab 1 continued)

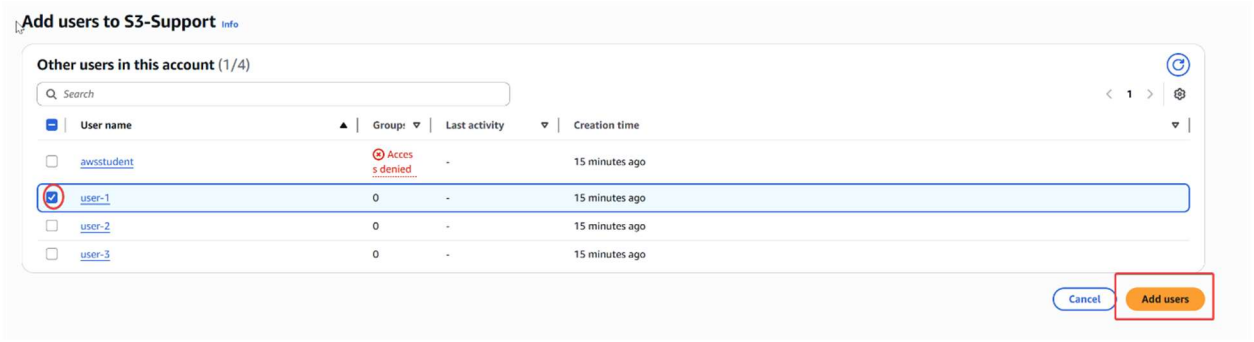
Click “User Groups” on the left menu to return to S3-Support group menu



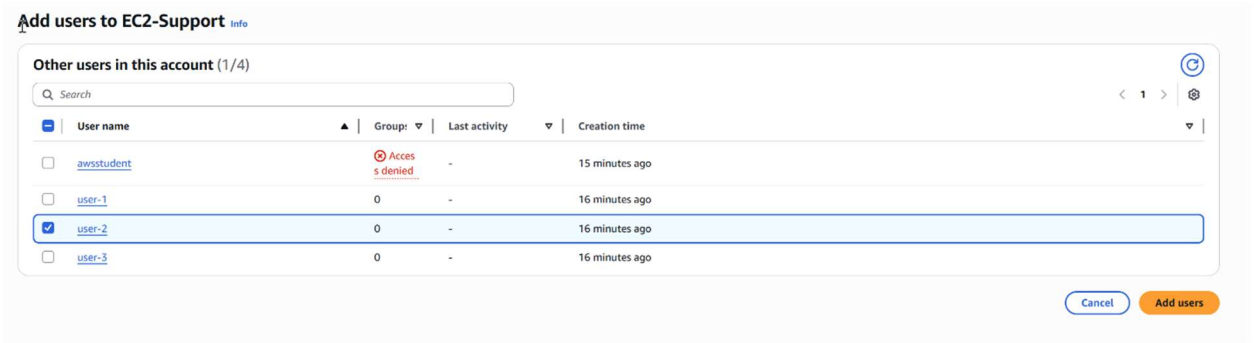
Click “Add users”



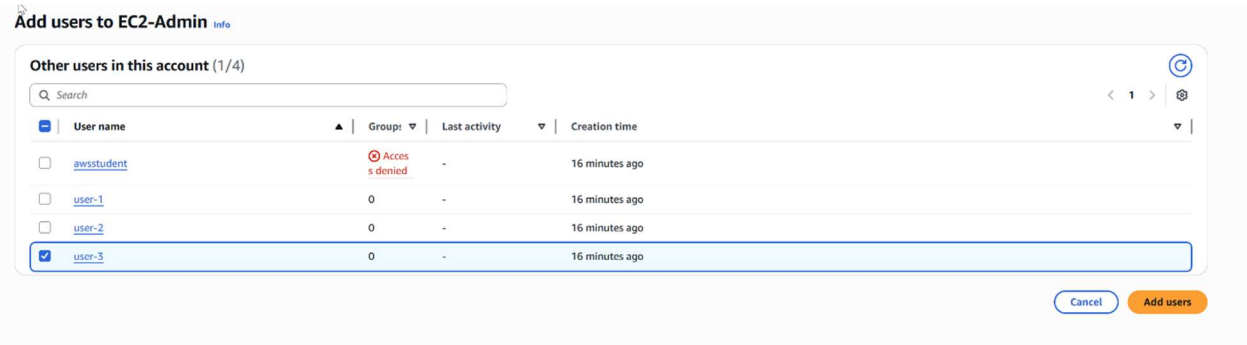
Check the box next to “user-1”, then click the orange “Add users” button to add user-1 to S3-Support



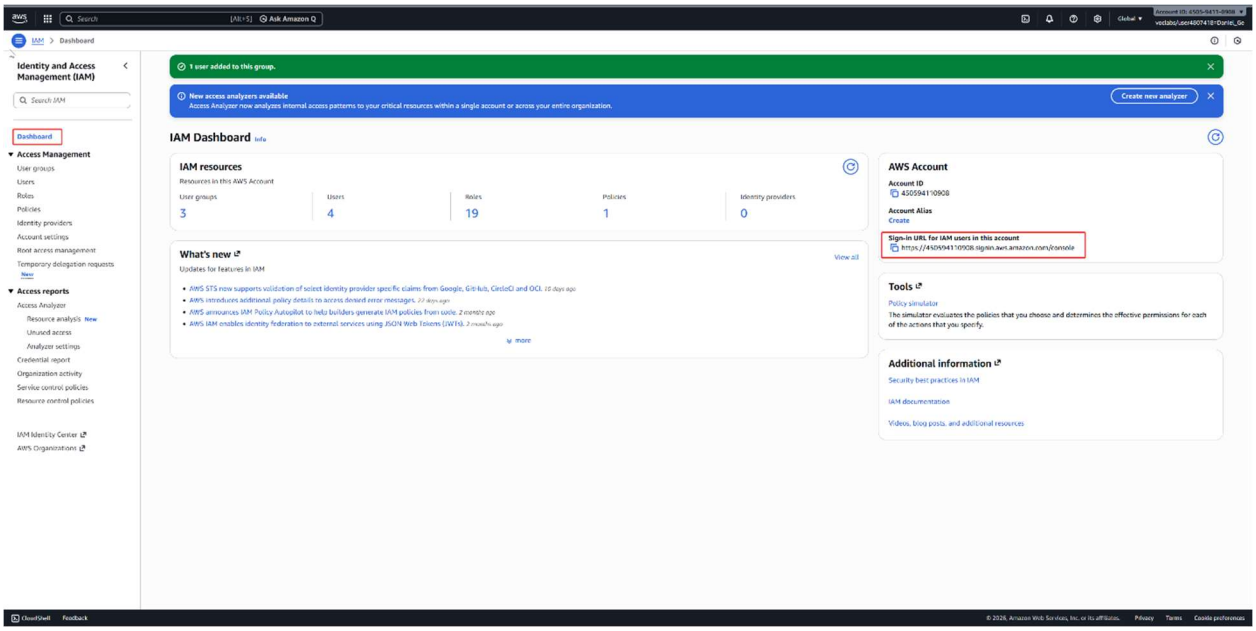
Follow the analogous steps to add “user-2” to EC2-Support



Follow the analogous steps to add “user-3” to EC2-Admin

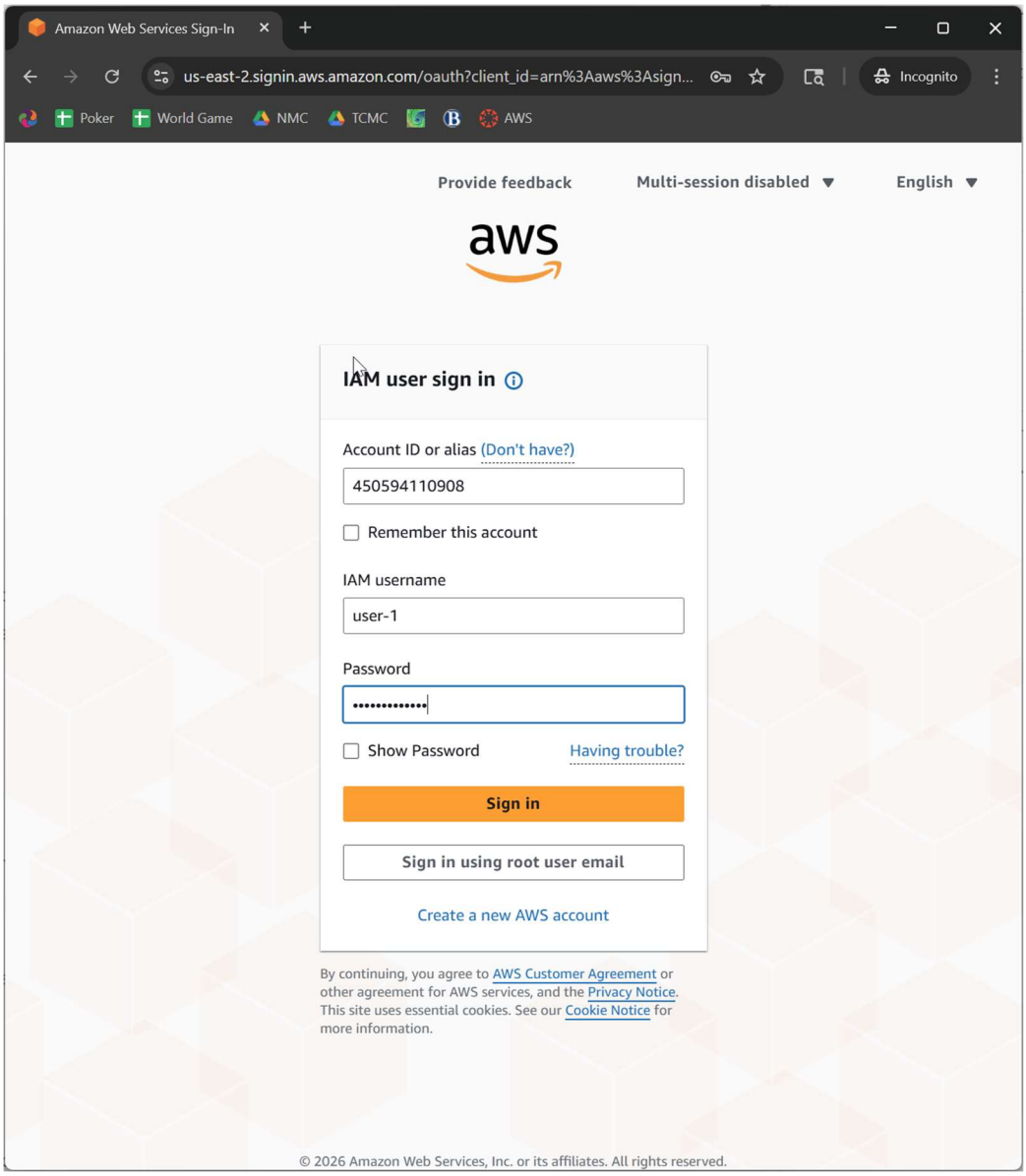


Click “Dashboard” on the left menu, then find the Sign-in URL for IAM users under AWS Account

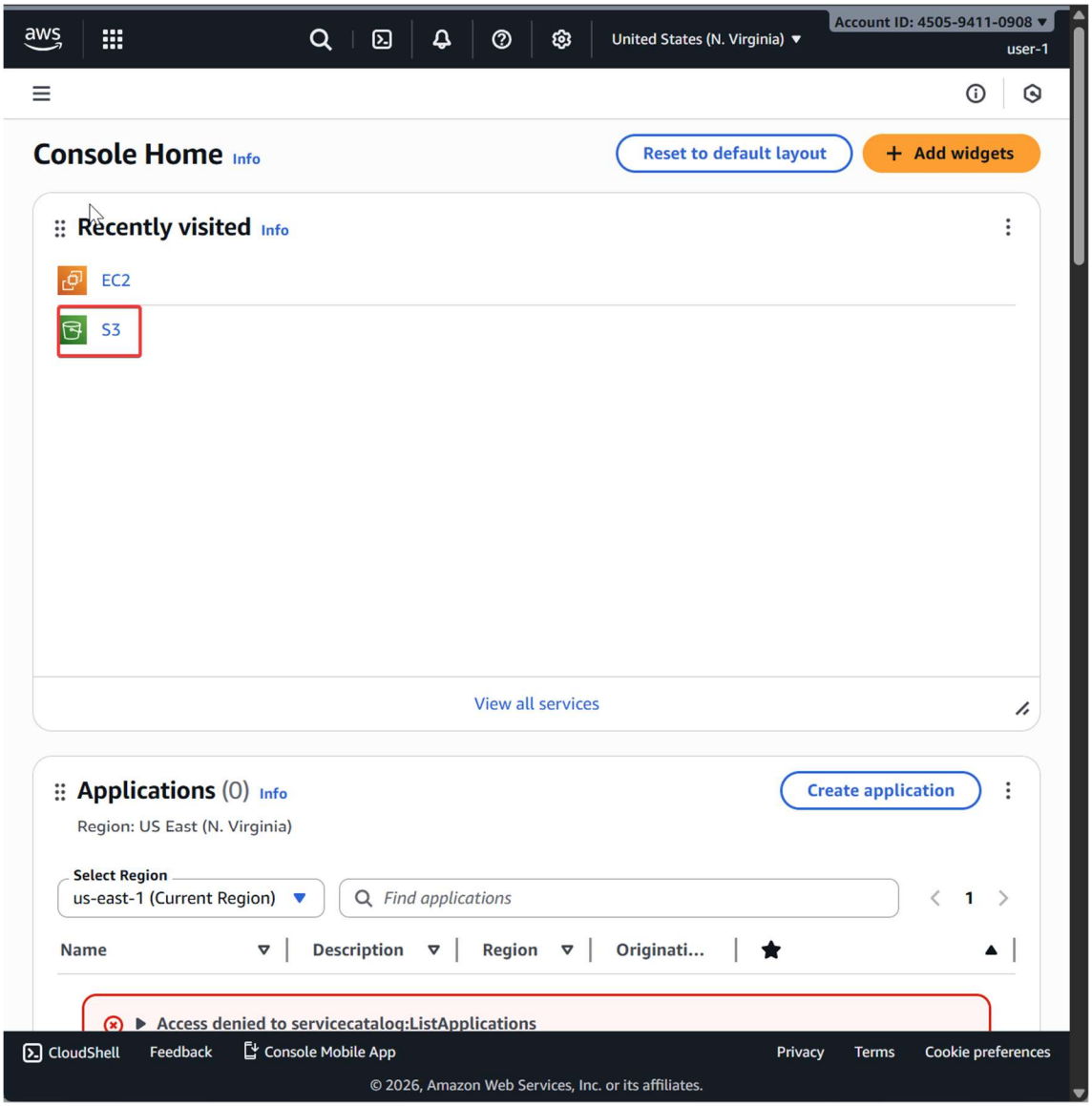


Open an Incognito tab using Ctrl+Shift+N, then paste the above sign-in URL into the search bar. Sign in through user-1’s credentials

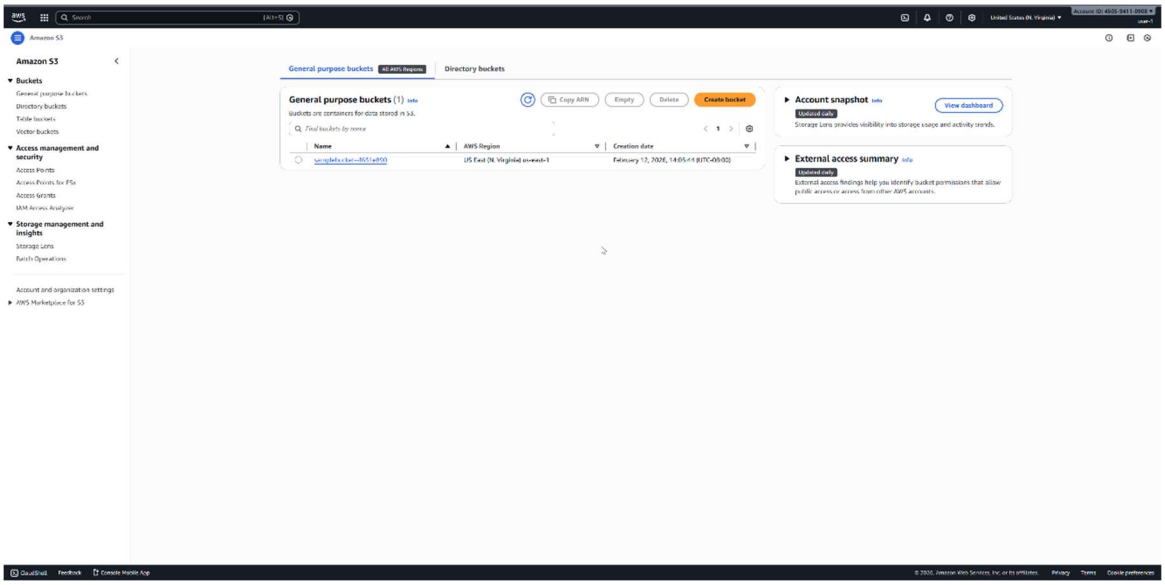
- Username: user-1
- Password: Lab-Password1



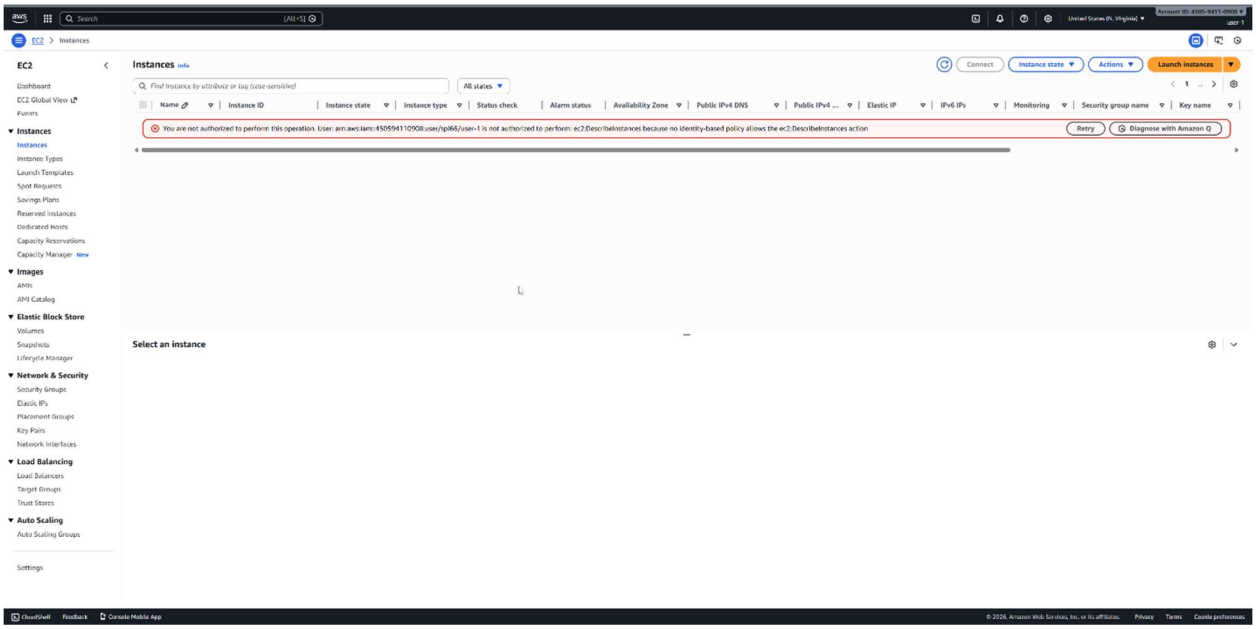
Once signed in, click S3 to enter the Simple Storage System console



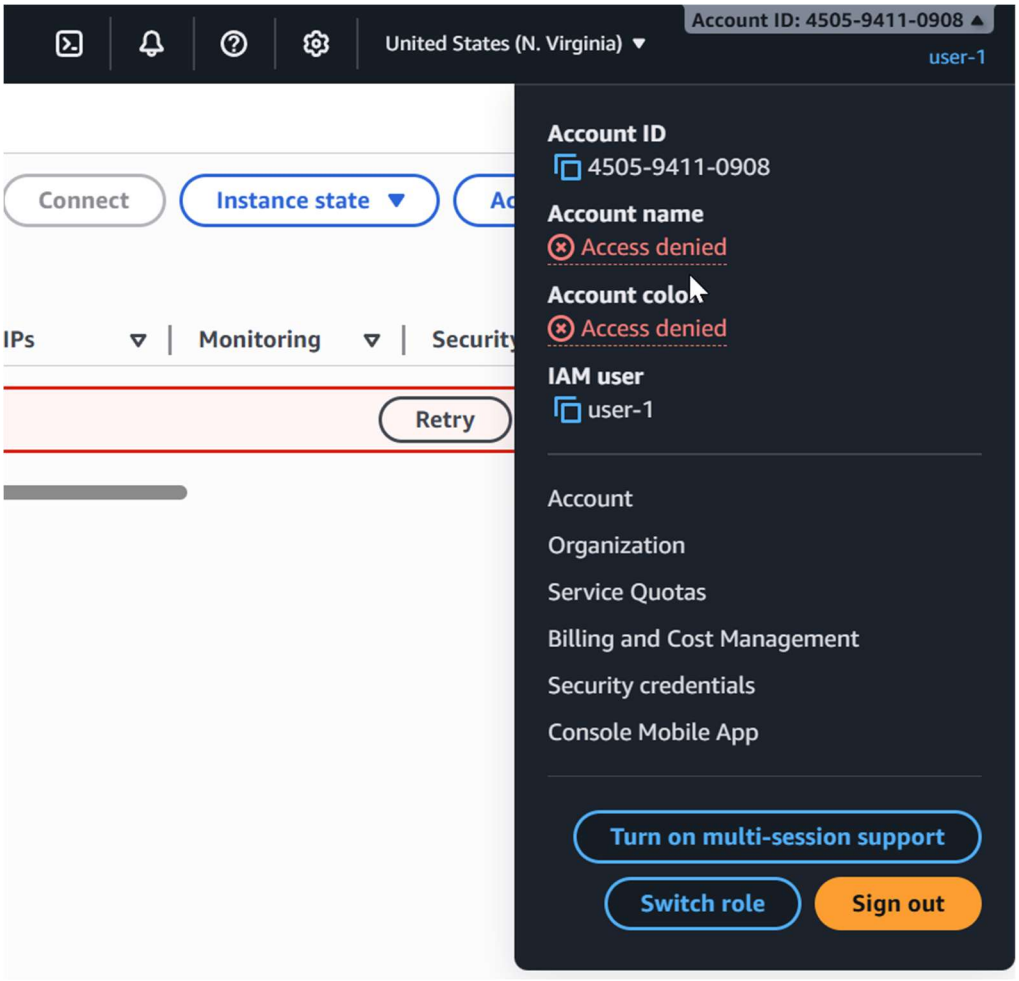
Since User-1 has been configured with the permissions to view S3 configurations, observe that they can see the Amazon S3 buckets associated with the company account



Navigate to EC2 through Services, and observe that User-1 cannot see the EC2 instances since they do not have permissions to do so



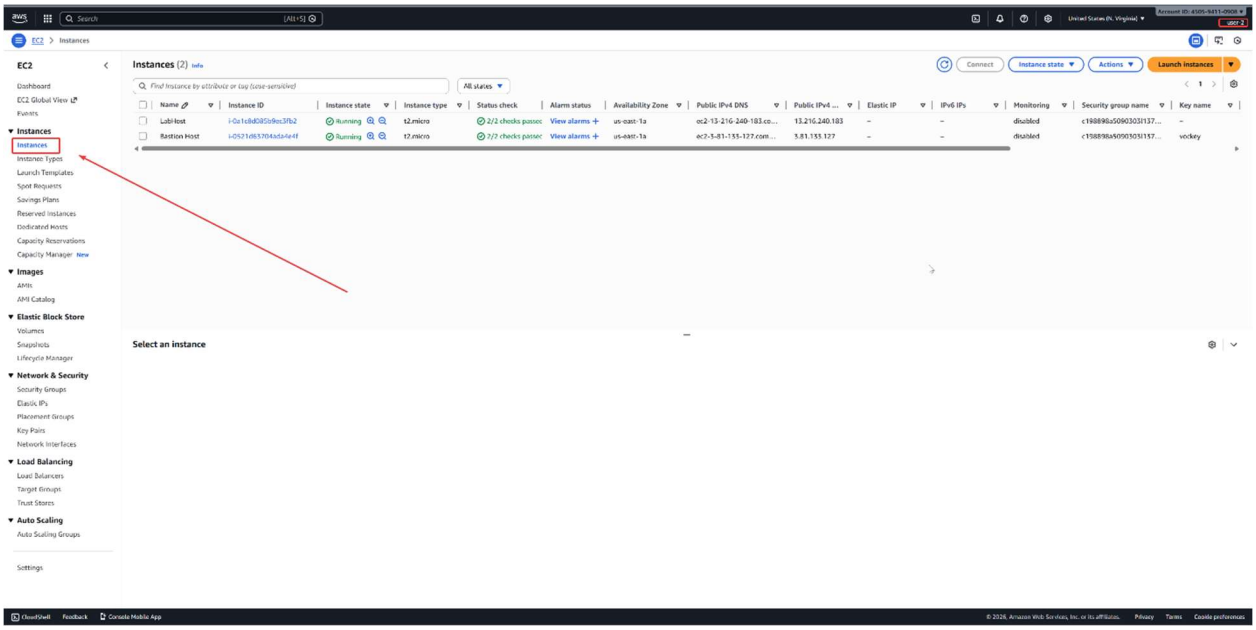
Click “user-1” on the top-right and “sign out”



Follow the same steps to sign in with User-2

- Username: user-2
- Password: Lab-Password2

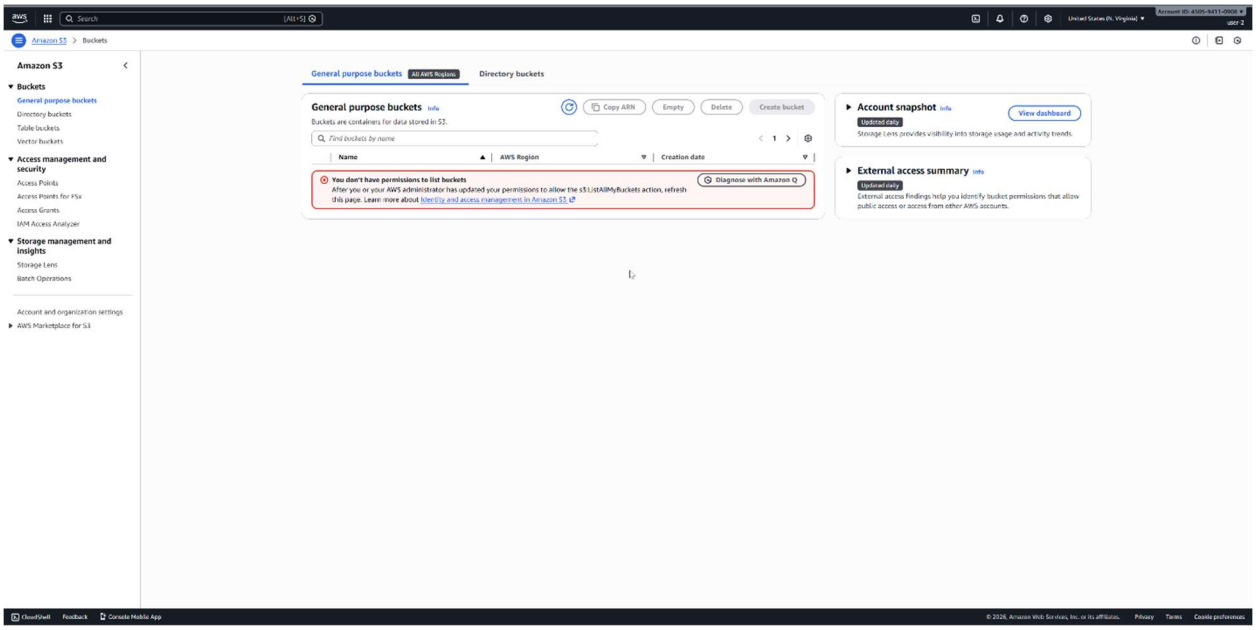
Enter the EC2 console, then click “Instances” on the left menu.



Click Instance-state in the top-right, then “Stop instance”

Observe the failure message. Although user-2 does have permissions to view EC2 instances, they do not have permissions to stop or start those instances

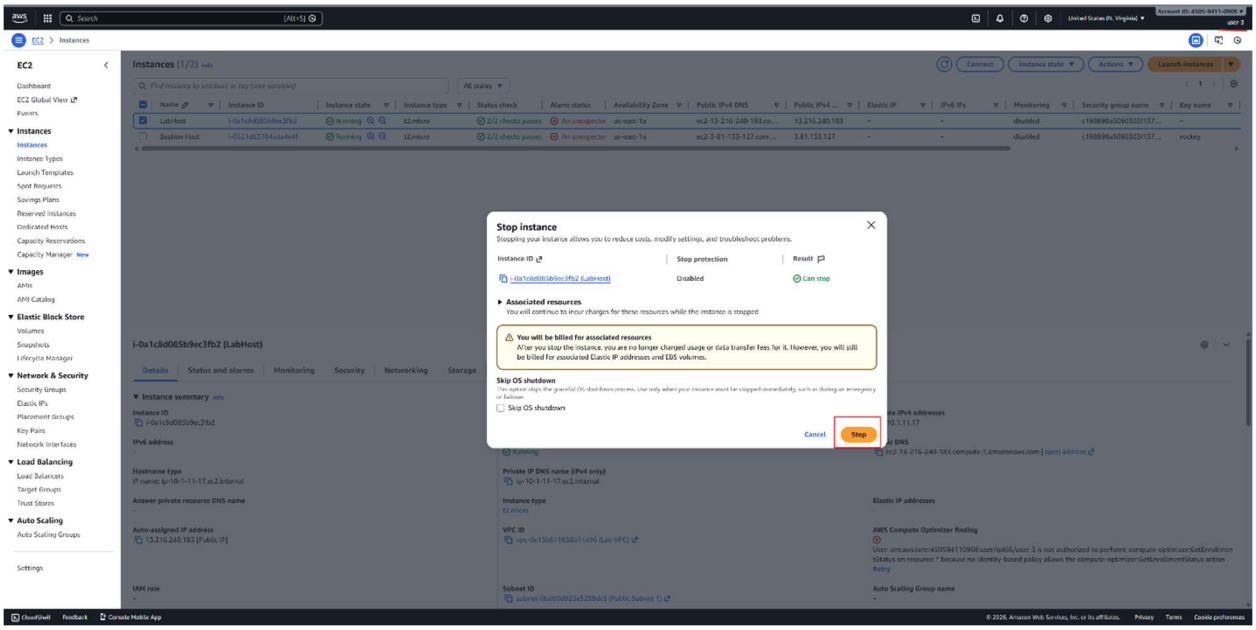
Open the AWS S3 console on user-2. User-2 does not have permissions to view those buckets



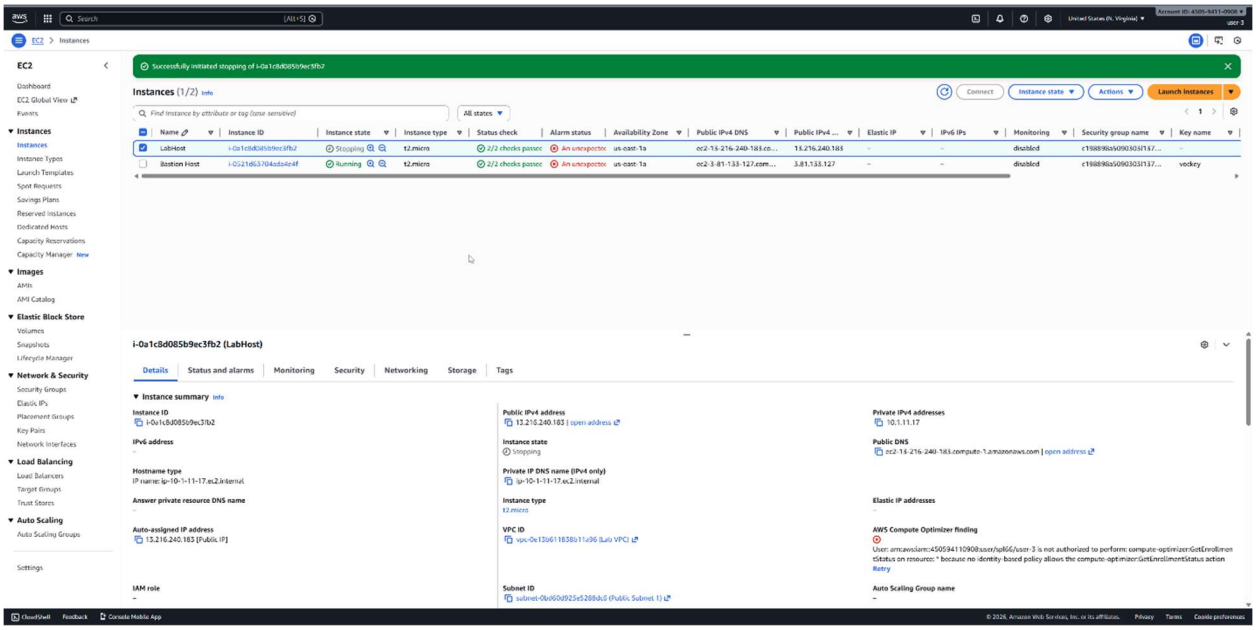
Sign out and log in to User-3's account

- Username: User-3
- Password: Lab-Password3

Follow the same steps as before to attempt to stop the instance

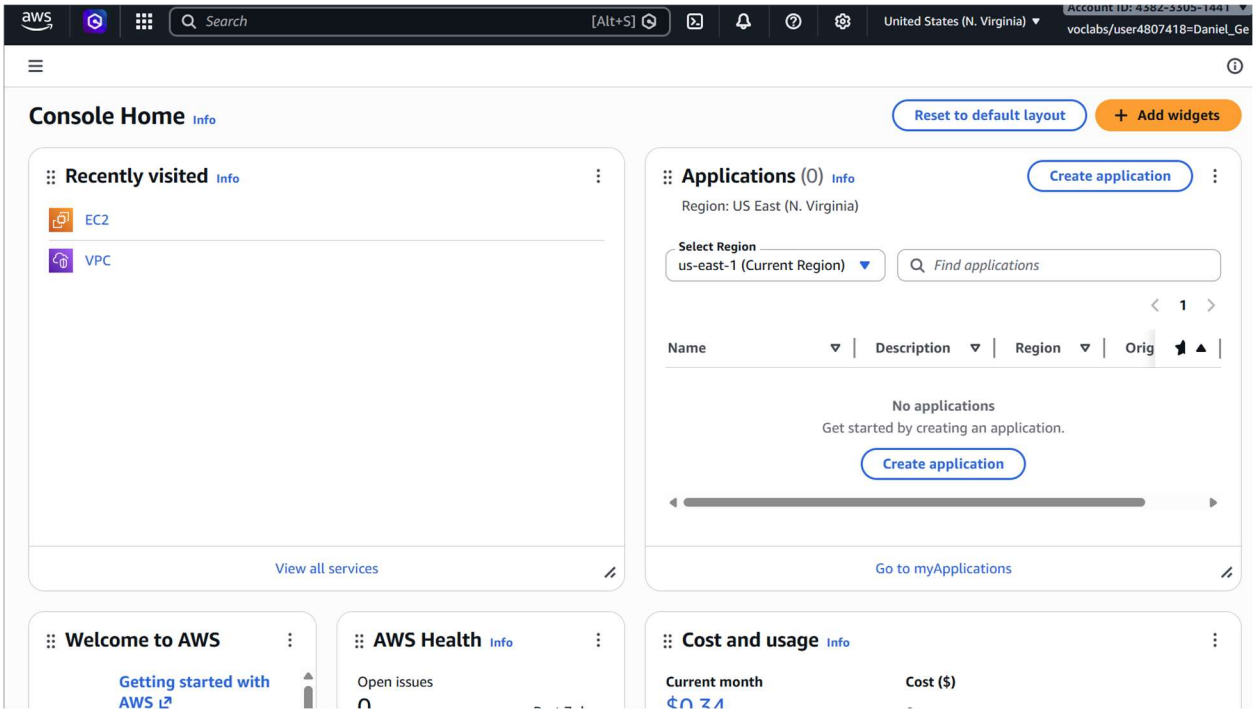


This time, the stop should be successful, since User-3 has been configured with permissions to stop the EC2 instance

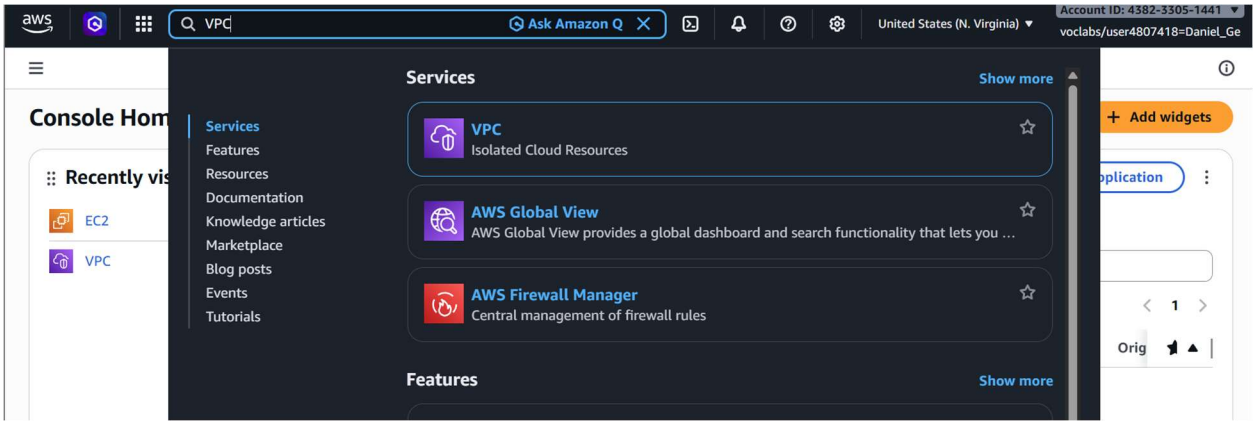


Building a Virtual Private Cloud and Launching a Web Server (Lab 2)

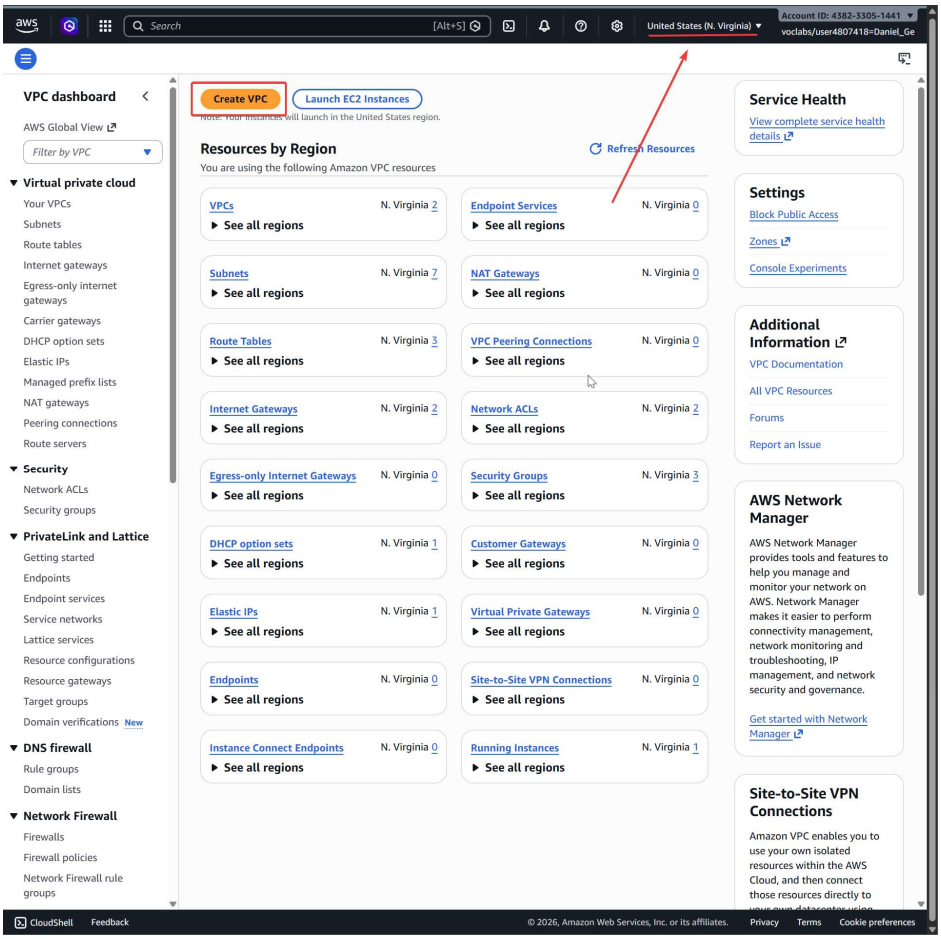
Start the lab and enter the main menu



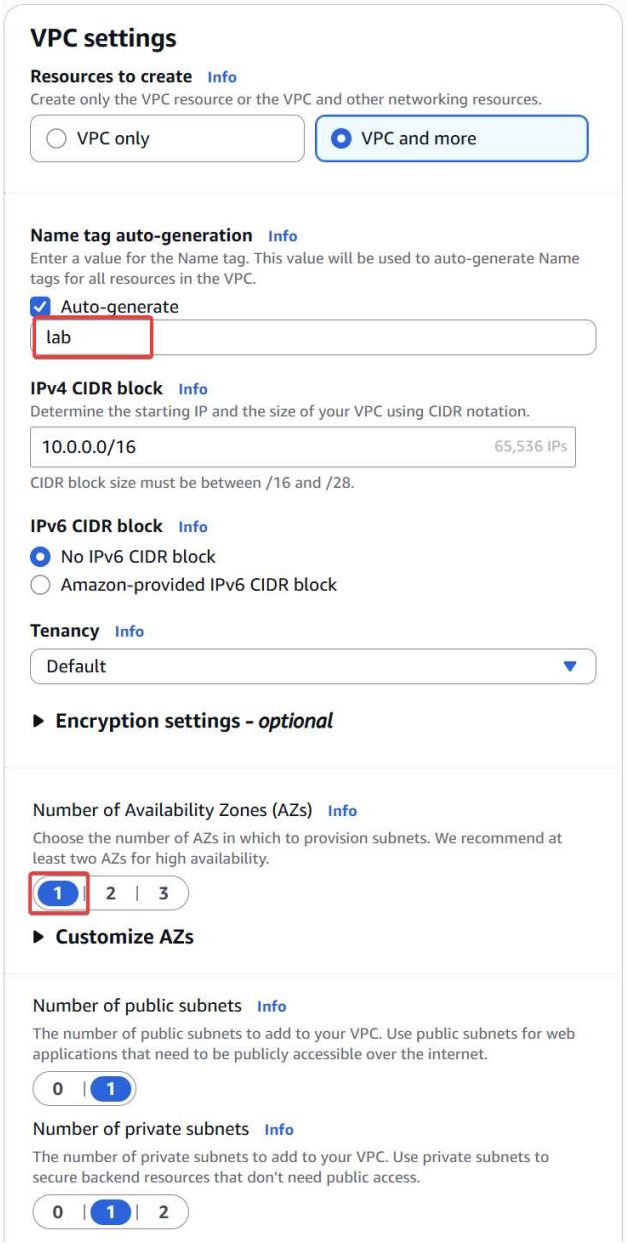
Search “VPC” in the search bar and click the VPC option to open the VPC menu



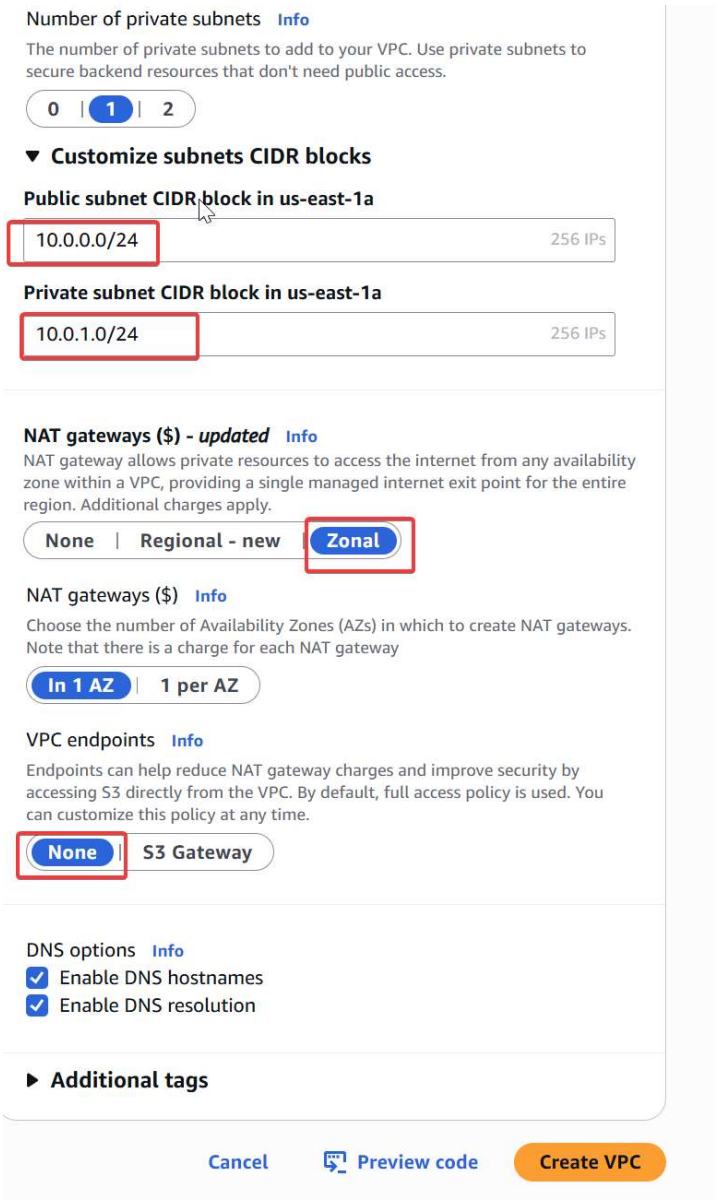
Read the top right to verify that the server is N. Virginia, then click “Create VPC”



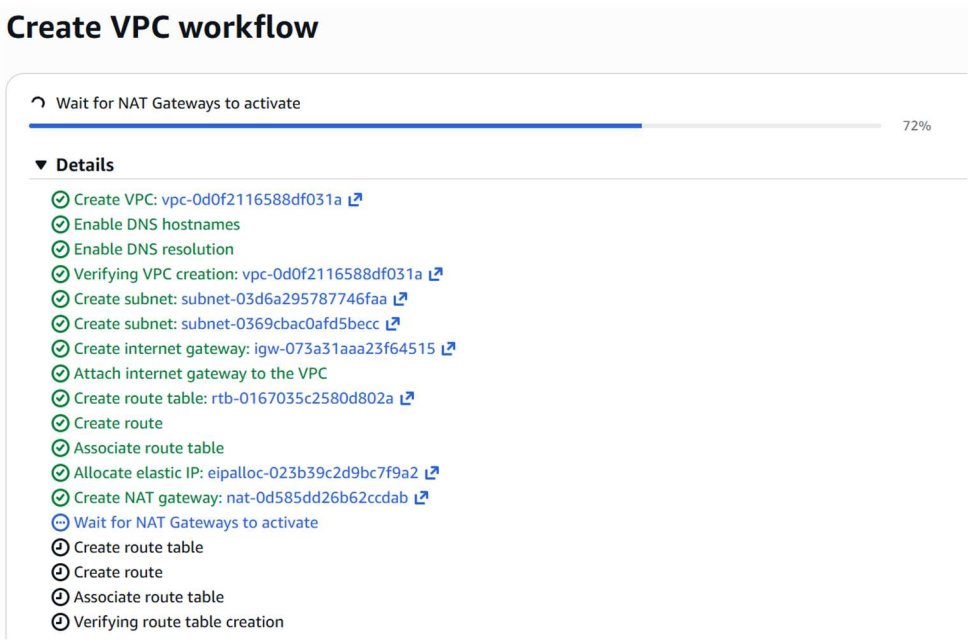
Under VPC Settings, change the Name tag to “lab” and Number of Availability Zones to 1



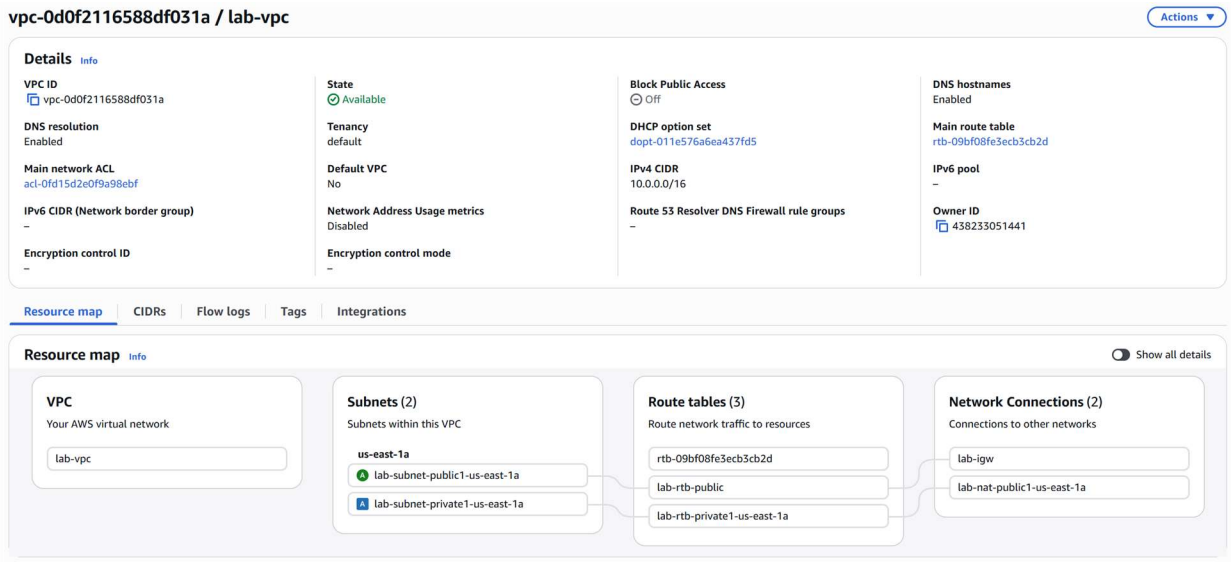
Edit the Public and Private subnet CIDRs as follows, change NAT gateways to Zonal, and set VPC endpoints to None. After verifying the configuration, click “Create VPC”



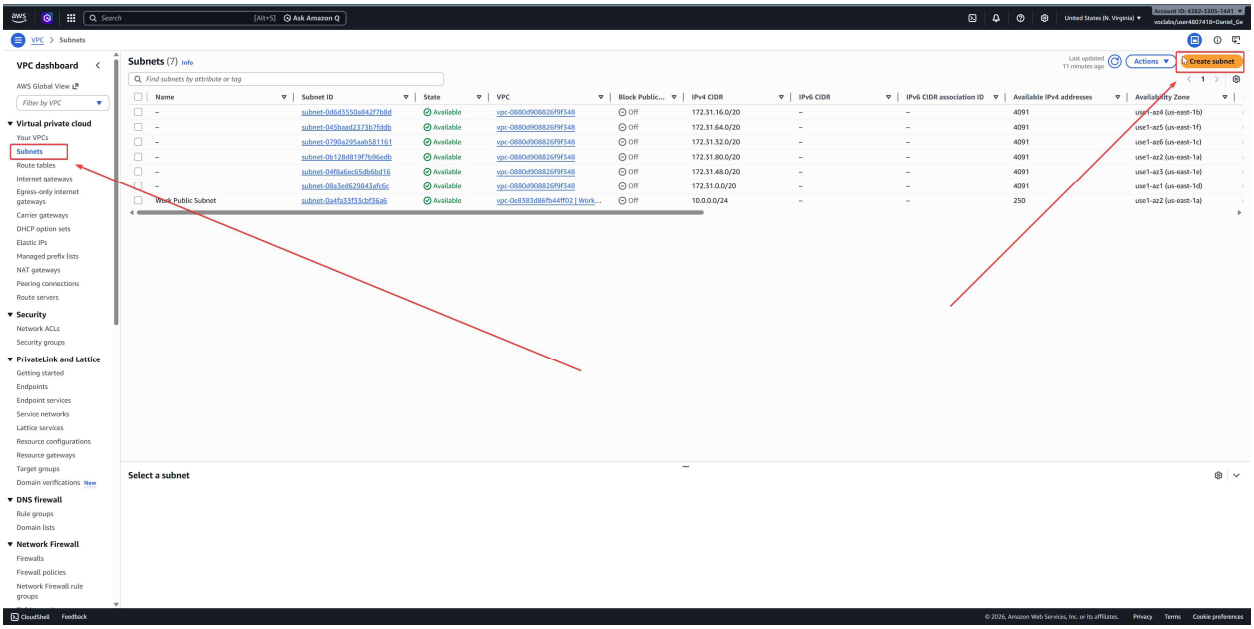
Wait for everything to activate and complete



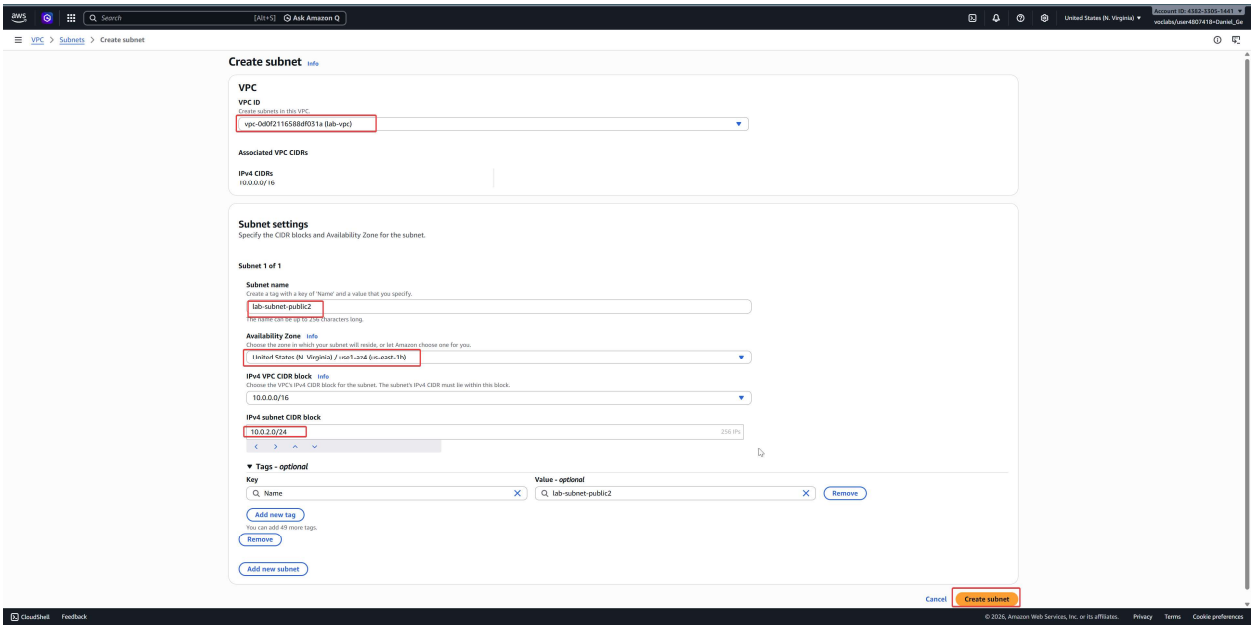
Click “View VPC”. The VPC contains all the addresses in the network 10.0.0.0/16. We have further configured the Lab subnets 10.0.0.0/24 and 10.0.1.0/24



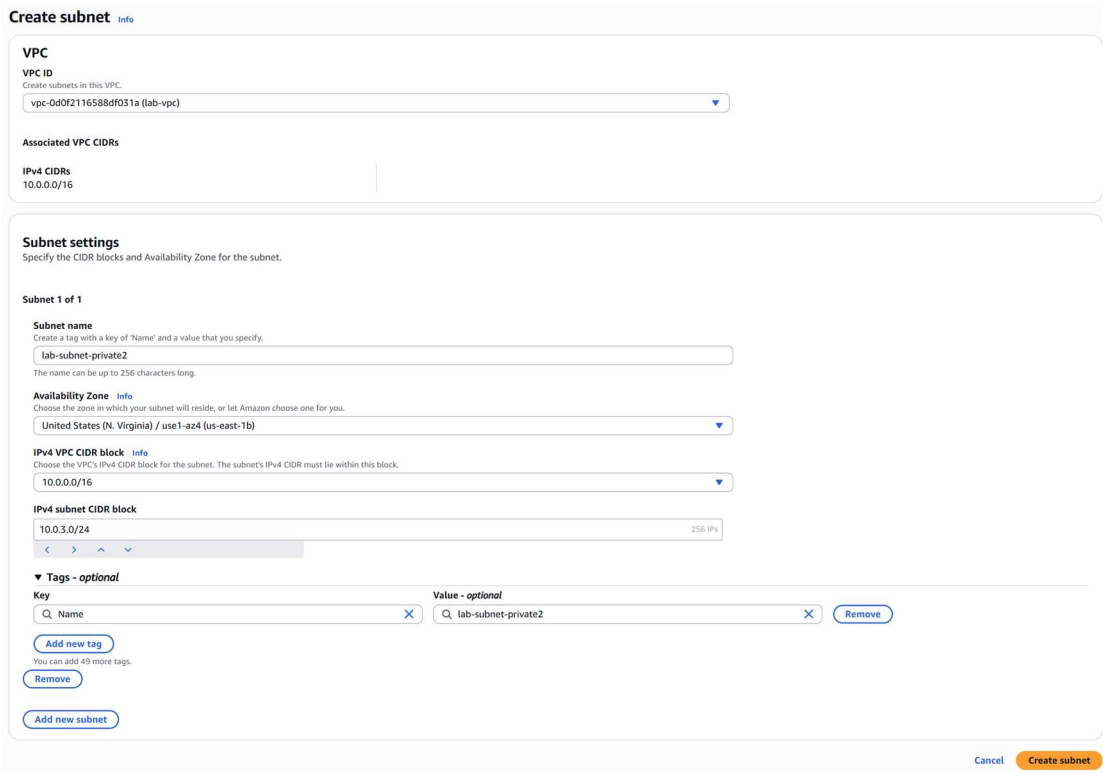
Click “Subnets” and “Create Subnets” to create more subnets



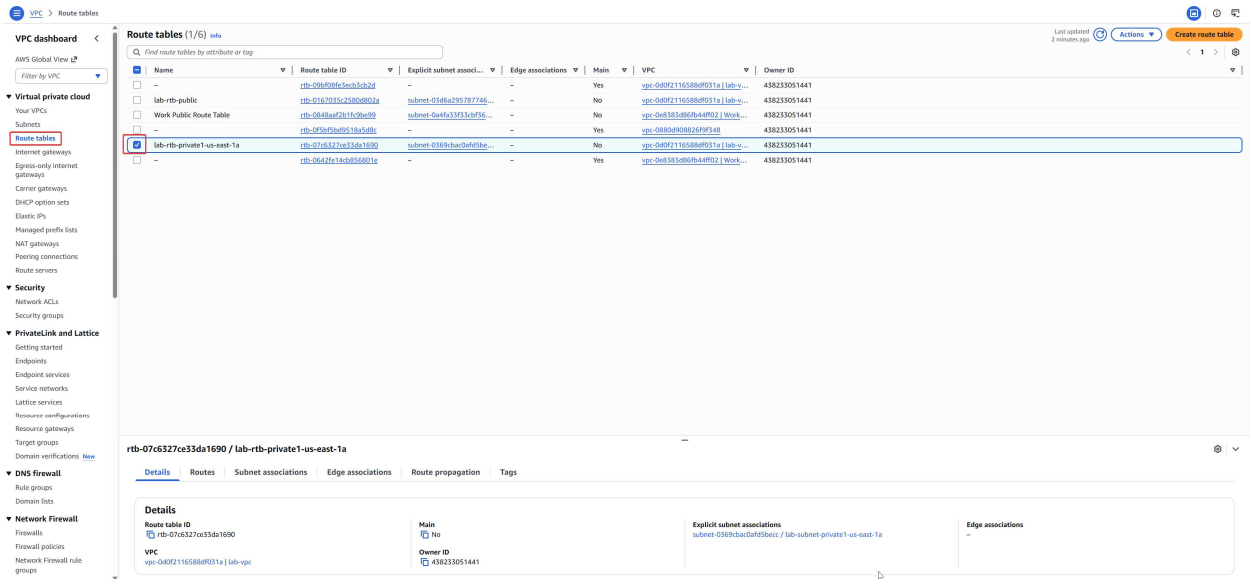
Set the VPC ID to “lab-vpc”, the Subnet name, and the IPv4 subnet CIDR block as follows:



Create the corresponding private subnet using the following configurations

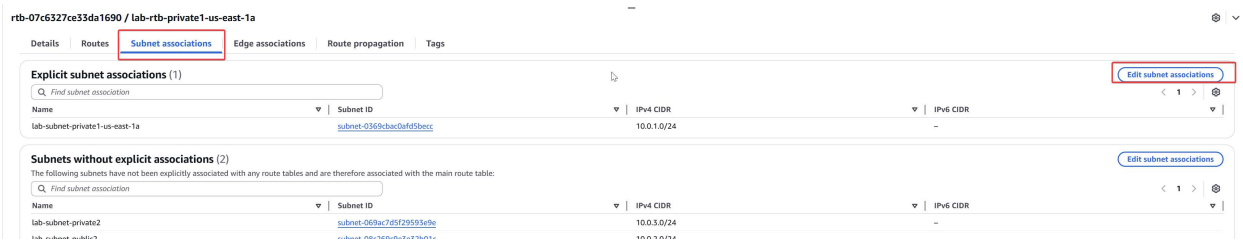


To begin configuring the routes, click “Route Table” on the left menu, and select the lab-rtb-private1-us-east-1a route table

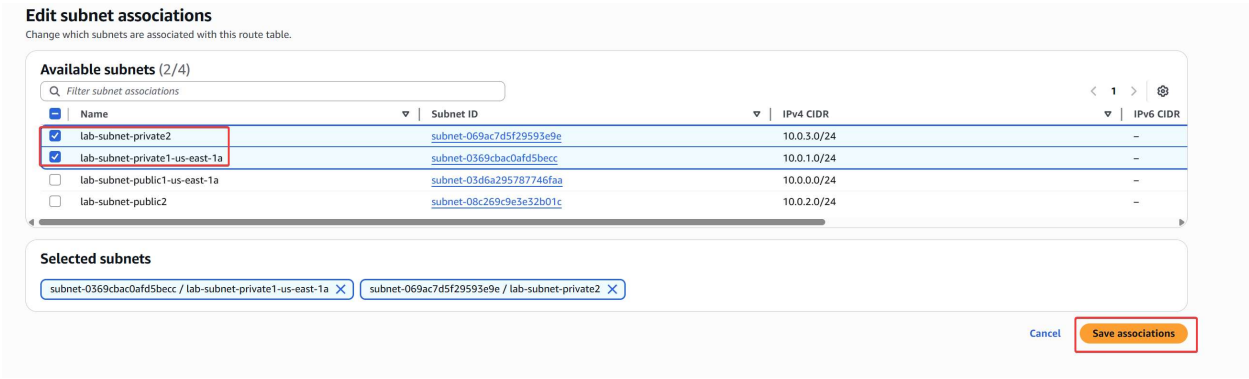


Click “Routes” on the bottom menu, and observe how traffic to the internet (0.0.0.0/0) is sent through the nat gateway

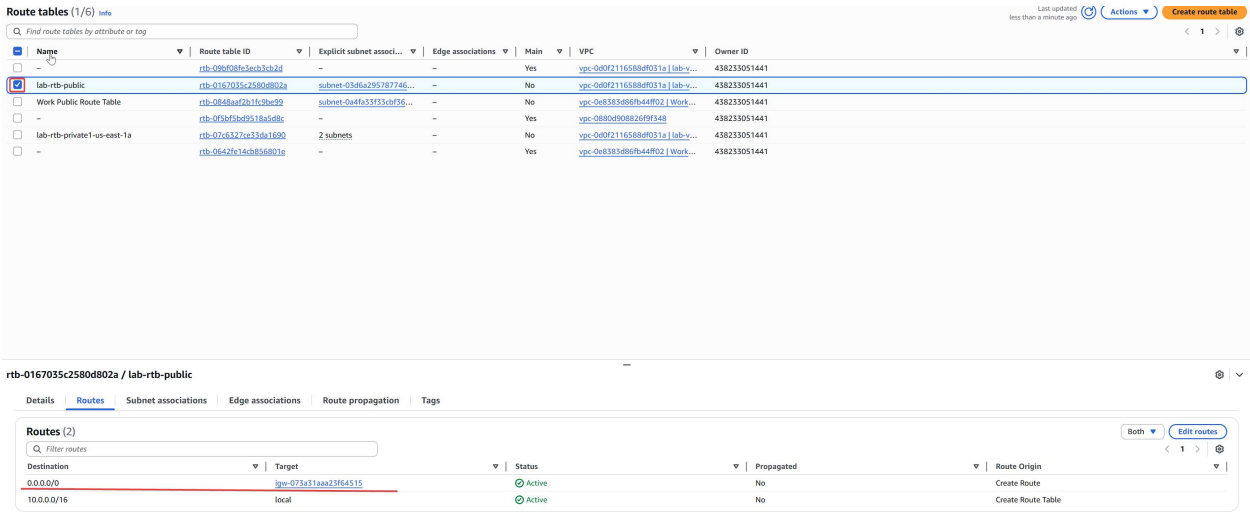
Click “Subnet associations”, then “Edit subnet associations” under Explicit subnet associations



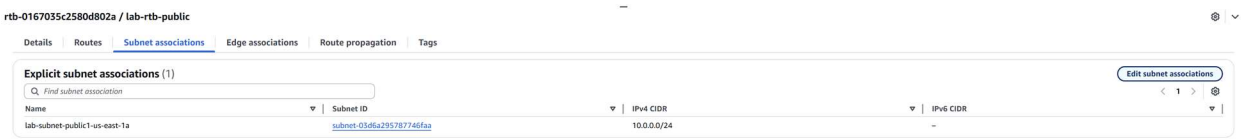
Select both lab-subnet-private1-us-east-1a and lab-subnet-private2, then “Save Assocations” such that both subnets receive this Route Table



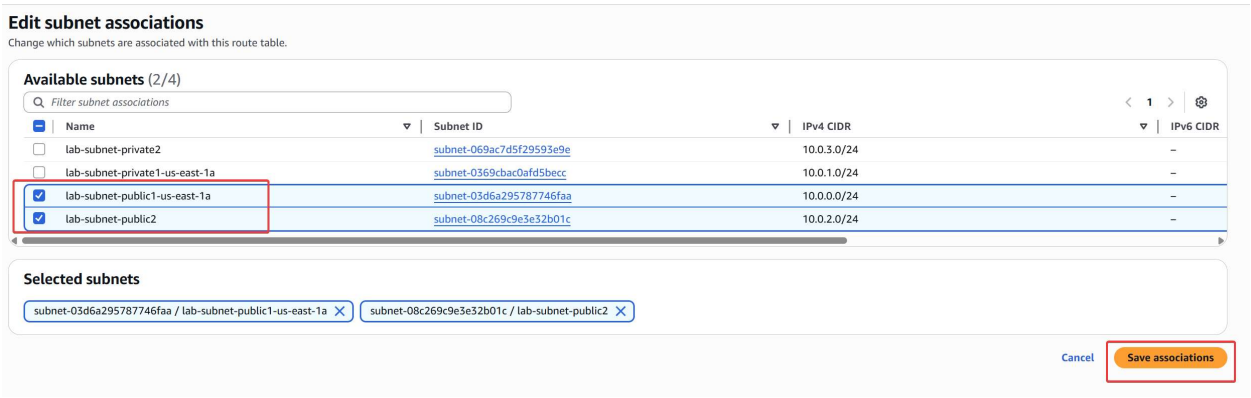
Return to the Route Table and select lab-rtb-public, then the “Routes” tab. Observe that for the public subnet, traffic to the internet have a direct route through the IGW (internet gateway)



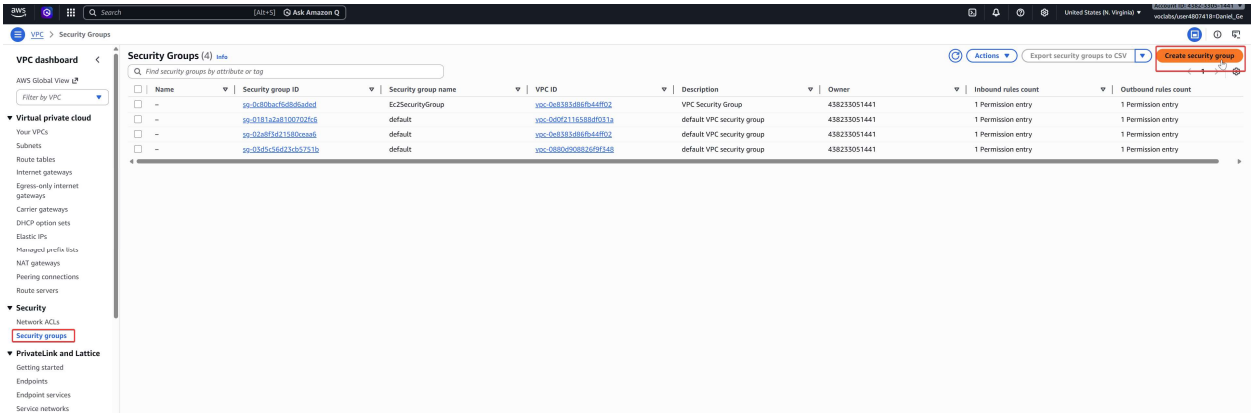
Click “Subnet associations”, then “Edit subnet associations” under Explicit subnet associations



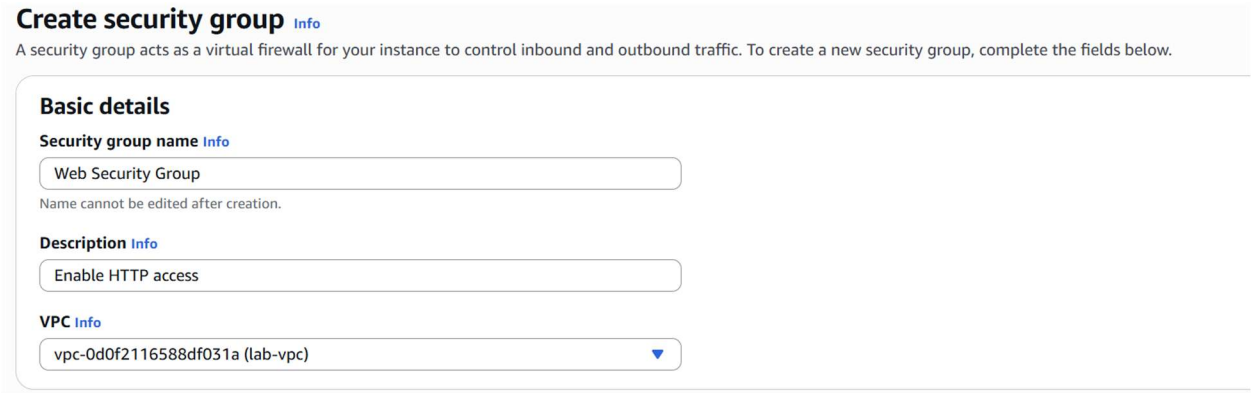
Select both public subnets, then “Save Associations”



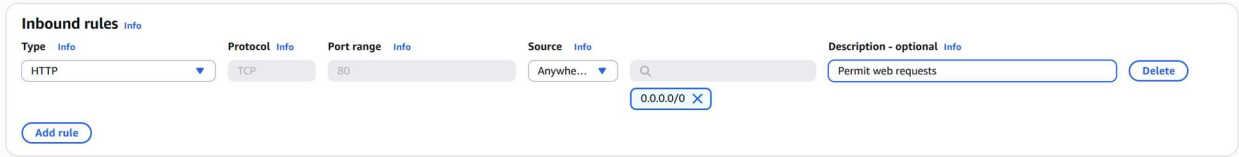
Click “Security Groups” on the left menu, then “Create Security Group”



Configure the name, description and VPC of the group as follows:

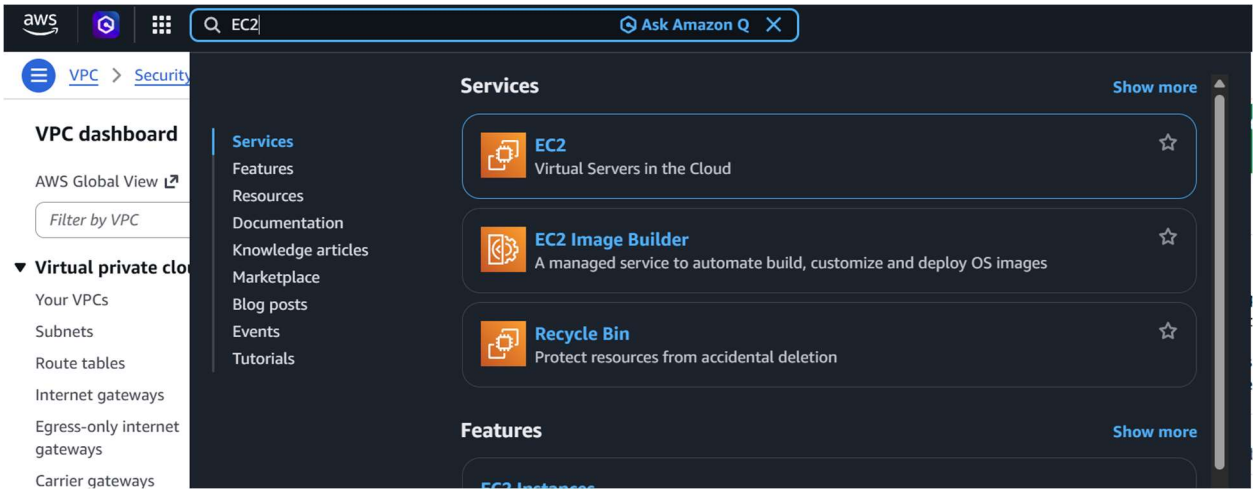


Click “Add rule” under Inbound rules to add an inbound rule. Set the Type to HTTP, configure Source as “Anywhere IPv4”, and configure the description

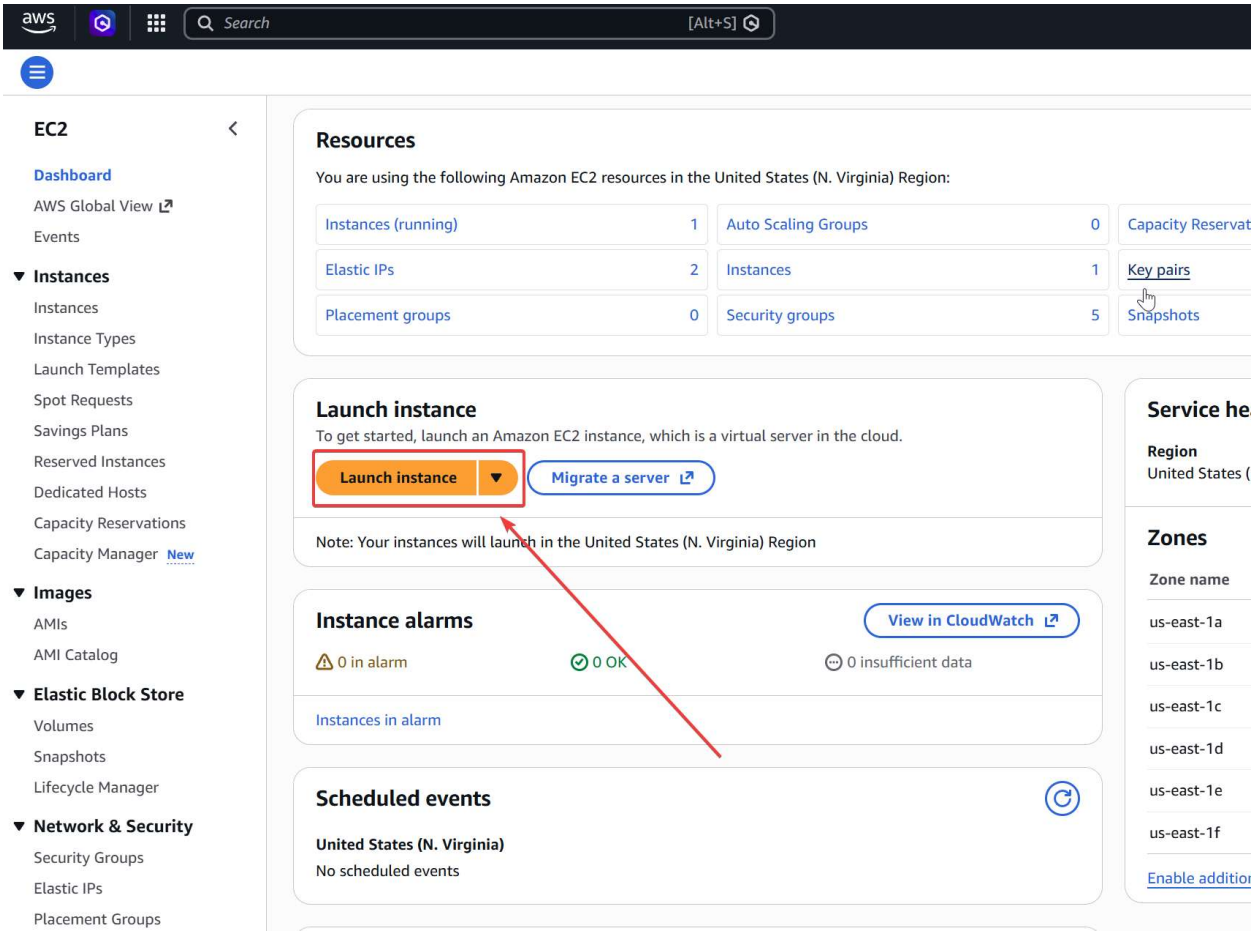


Click Create Security Group

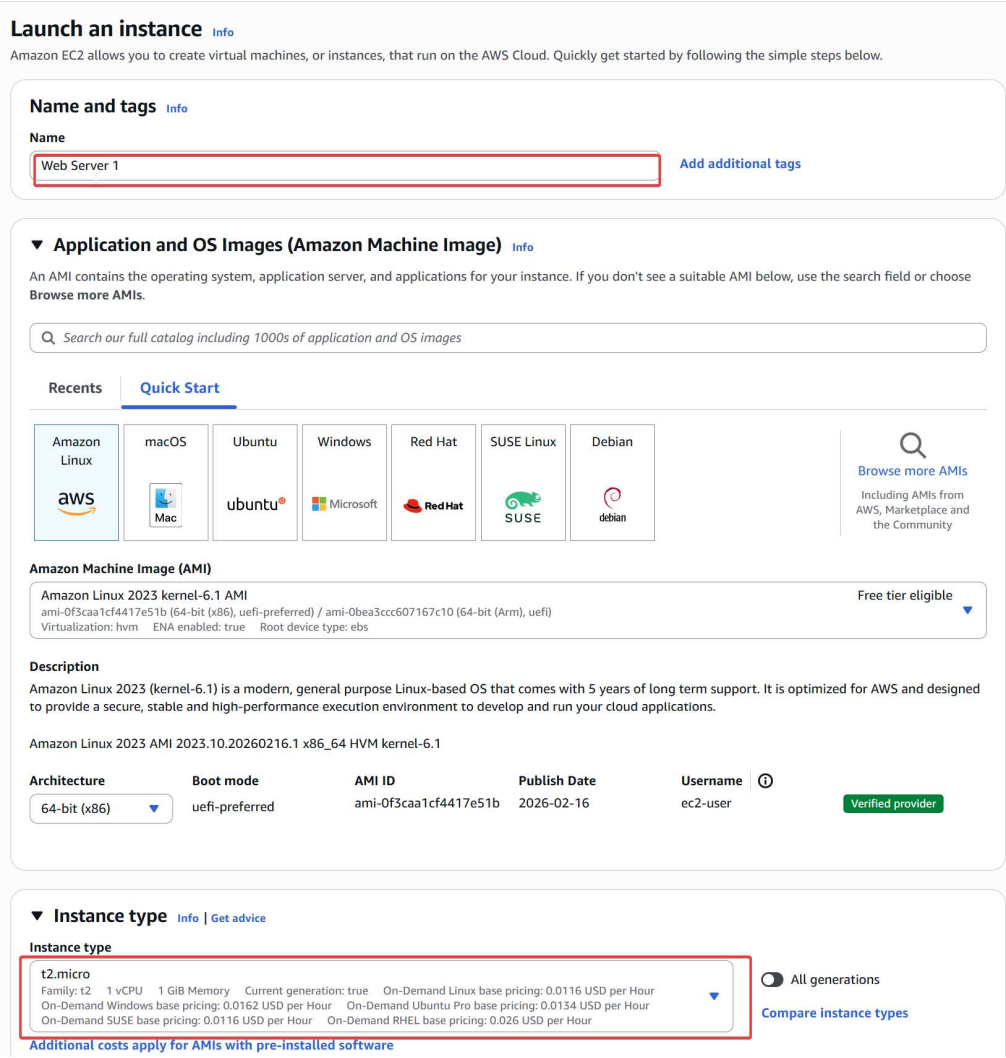
Search EC2 in the top search bar and open the EC2 console



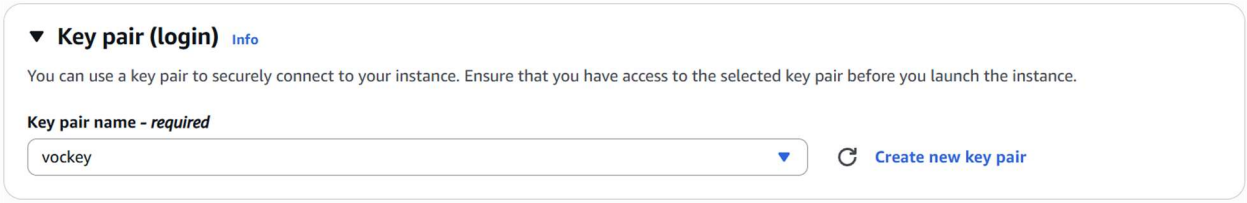
Click “Launch Instance”



Name the instance “Web Server 1” and the Instance type to t2.micro

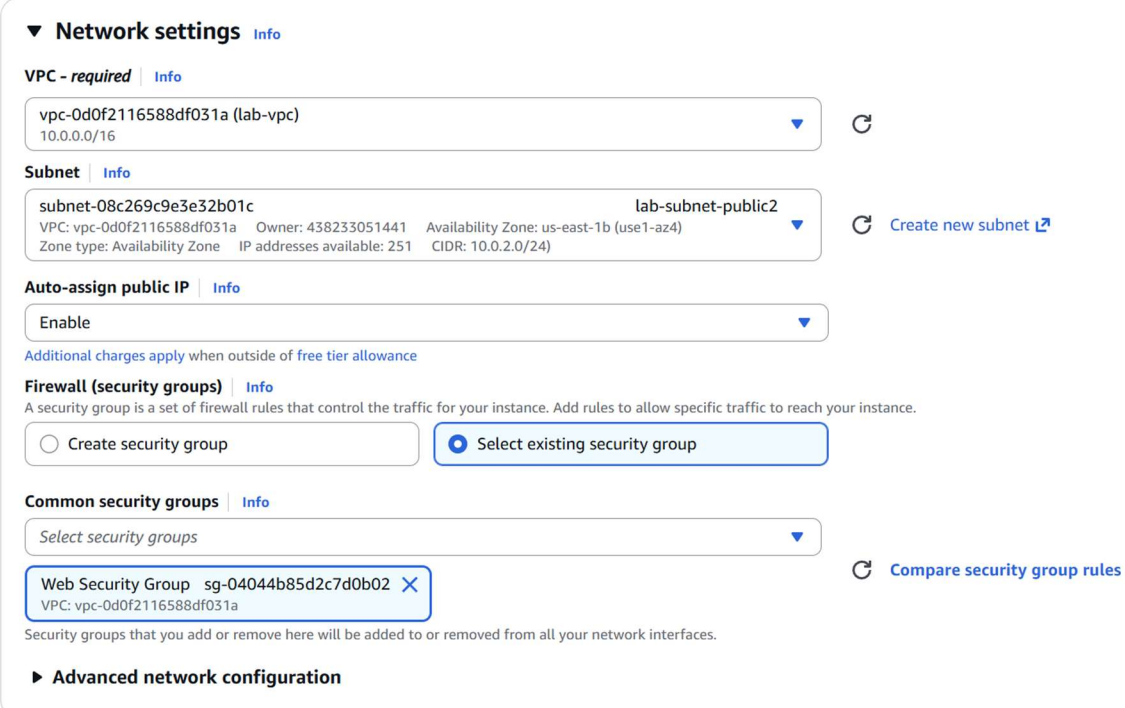


Set the Key pair name to “vockey”



Click the “Edit” button next to Network settings, and add the instance to the subnet and VPC network we created by

- selecting “lab-vpc” under VPC
- selecting “lab-subnet-public2” under Subnet
- enabling auto-assigning public IP
- Set Firewall to security group “Web Security Group”

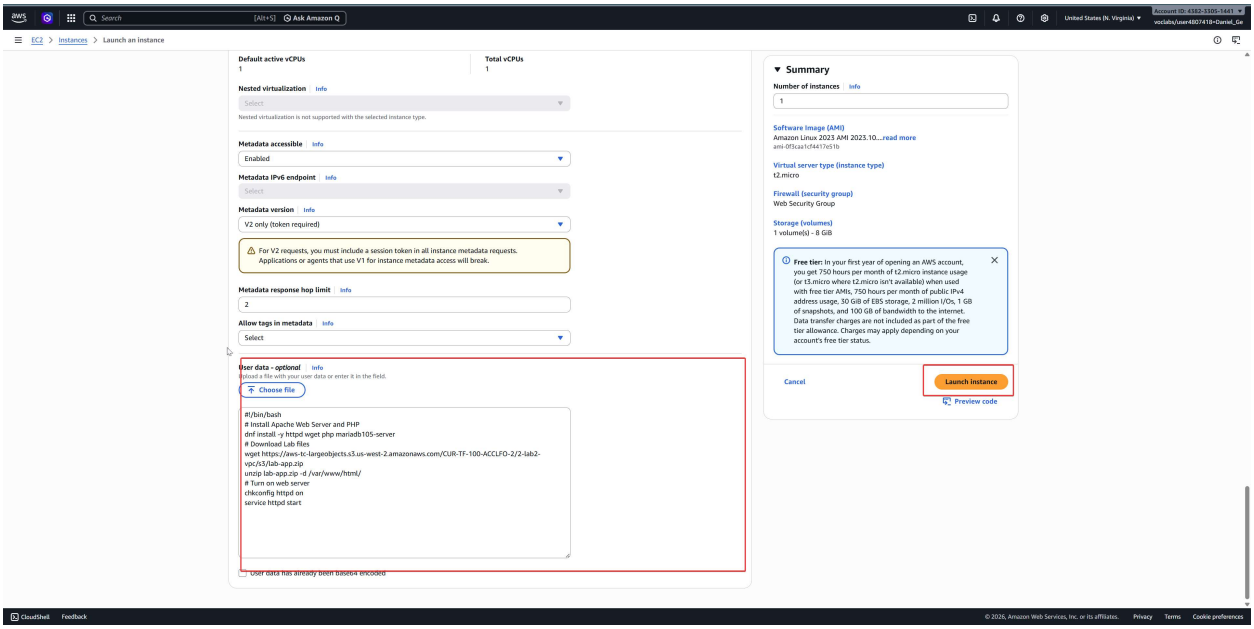


Open “Advanced details”, scroll all the way to the bottom, and paste the following code

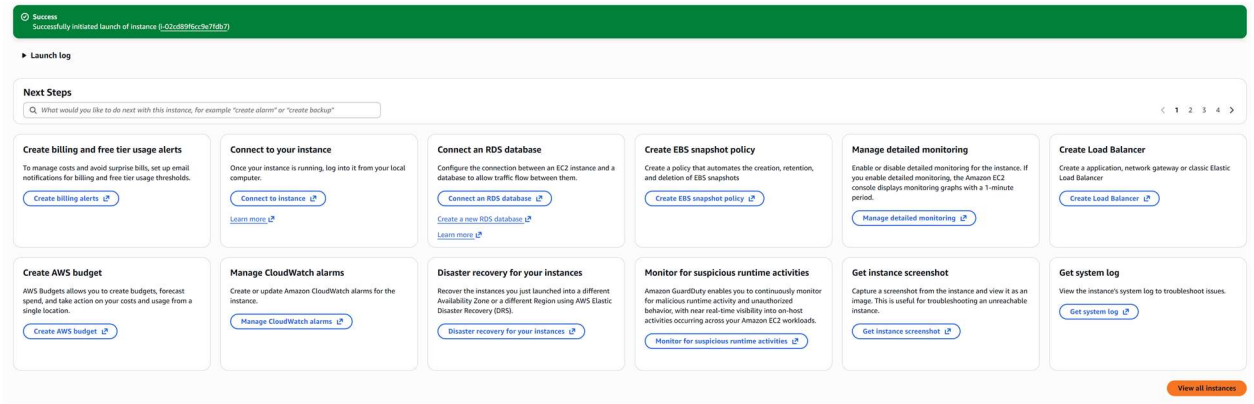
```
#!/bin/bash

# Install Apache Web Server and PHP

dnf install -y httpd wget php mariadb105-server

# Download Lab files

wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-ACCLFO-2/2-lab2-vpc/s3/lab-app.zip

unzip lab-app.zip -d /var/www/html/

# Turn on web server

chkconfig httpd on

service httpd start
```

Into the “User data” box

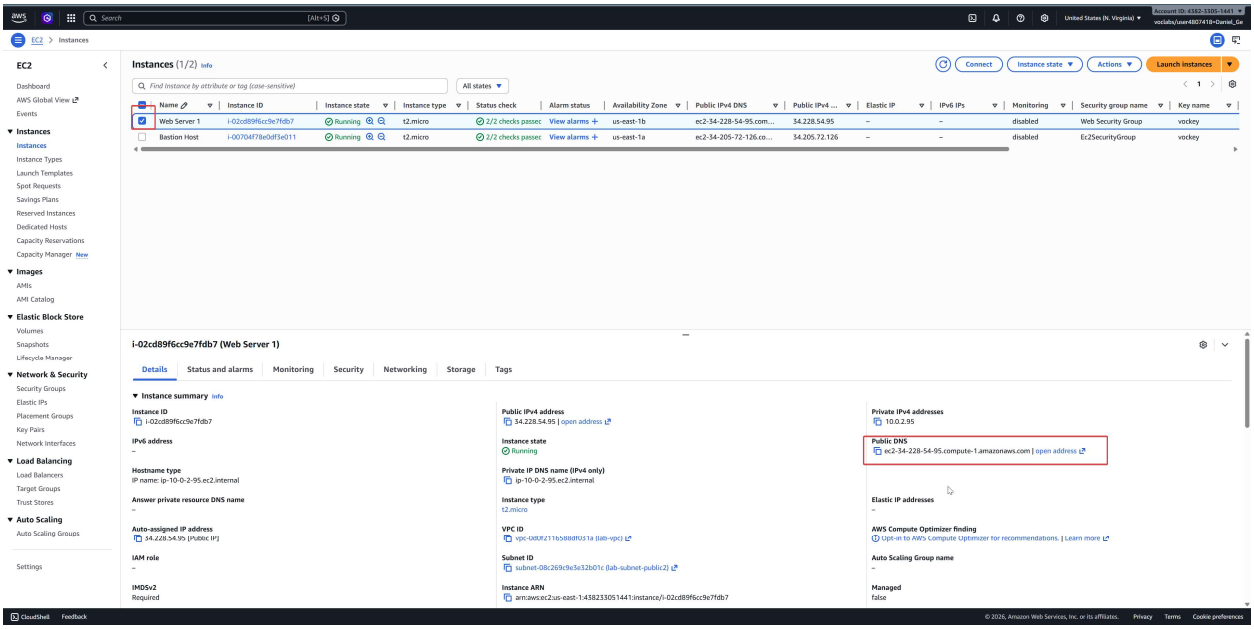
Click “Launch Instance”



Upon seeing the Success message, click “View all instances” at the bottom of the screen



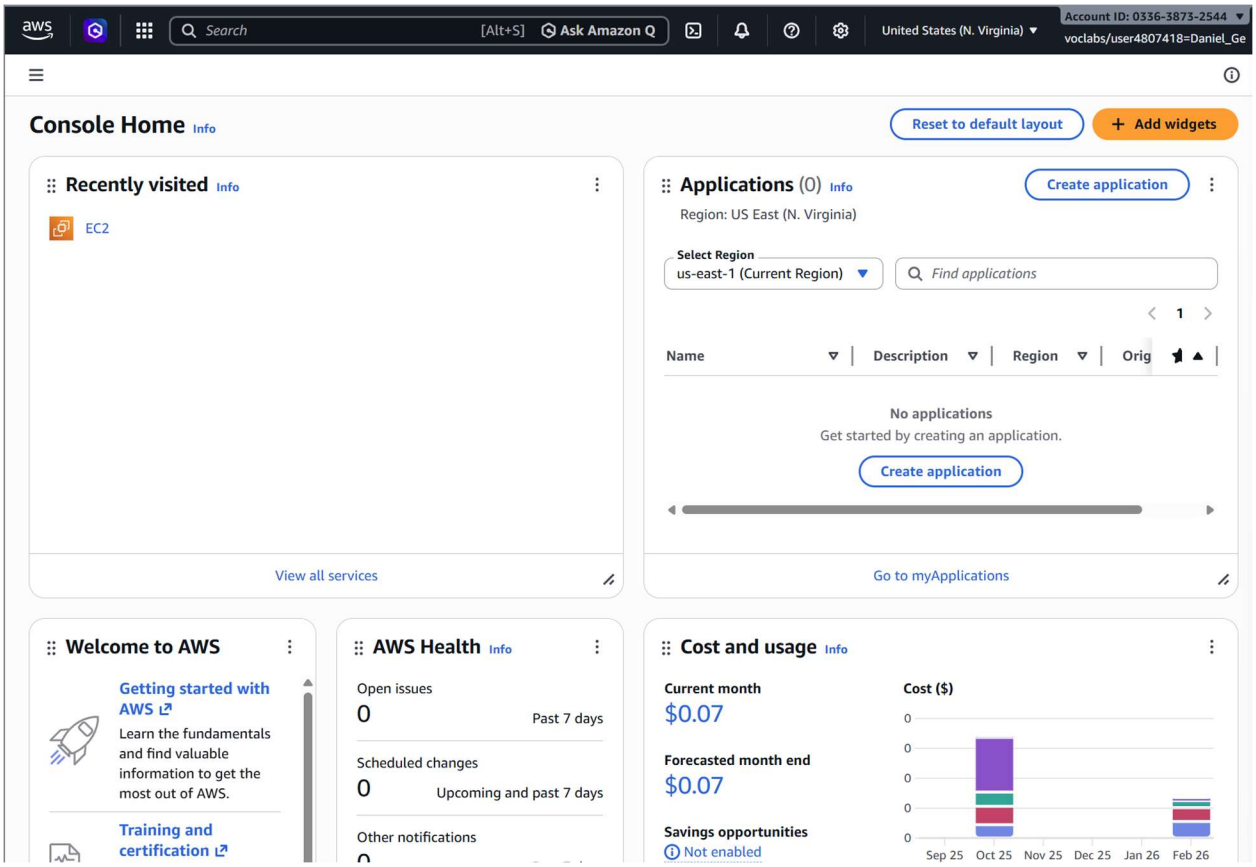
After the status displays 2/2 checks passed next to Web Server 1, check the Web Server 1 box



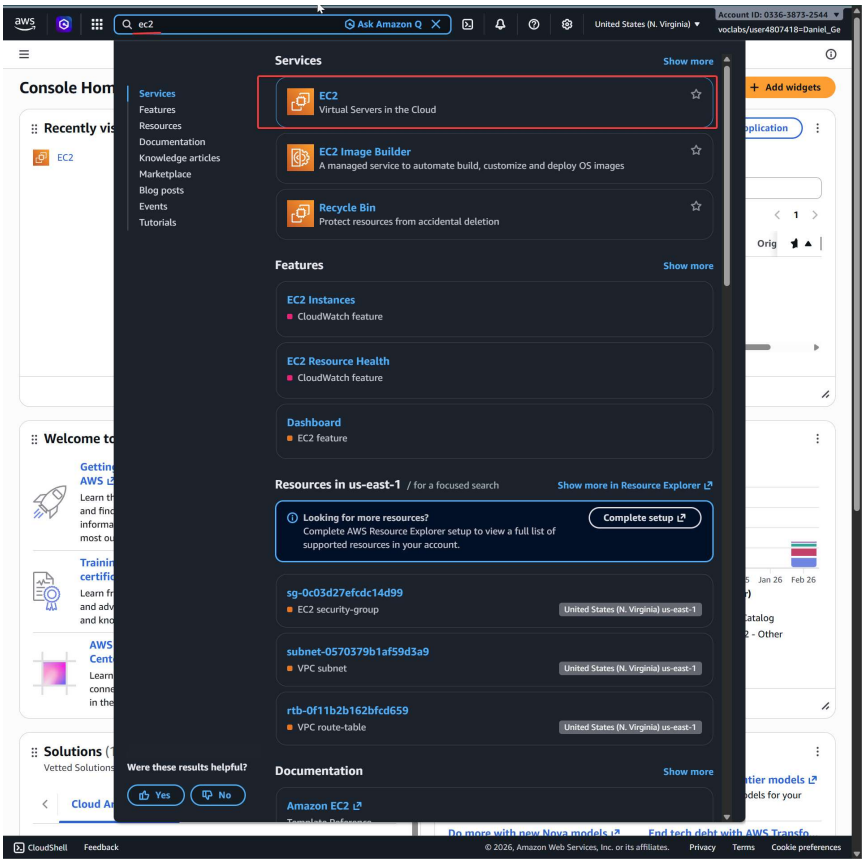
Check the details tab to find the Public IPv4 DNS address, and paste the address into the browser

Introduction to Amazon EC2 (Lab 3)

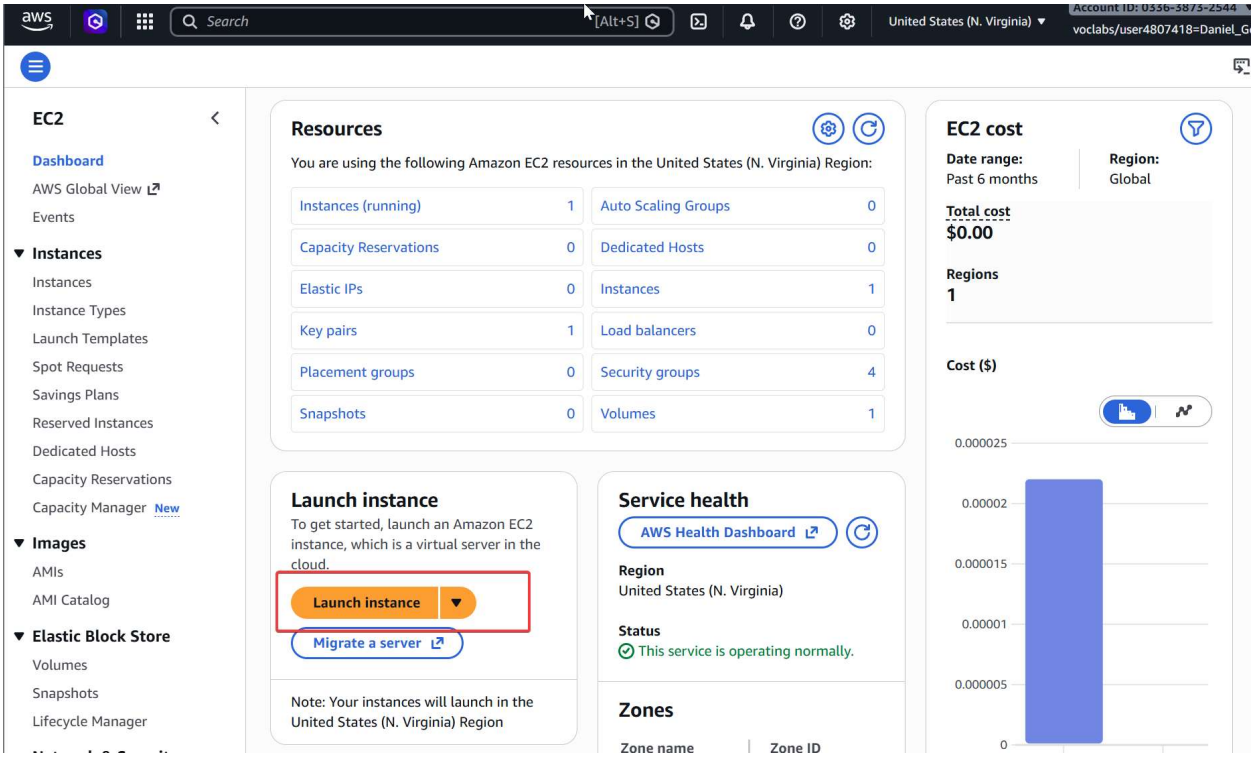
Start the lab and enter the main menu



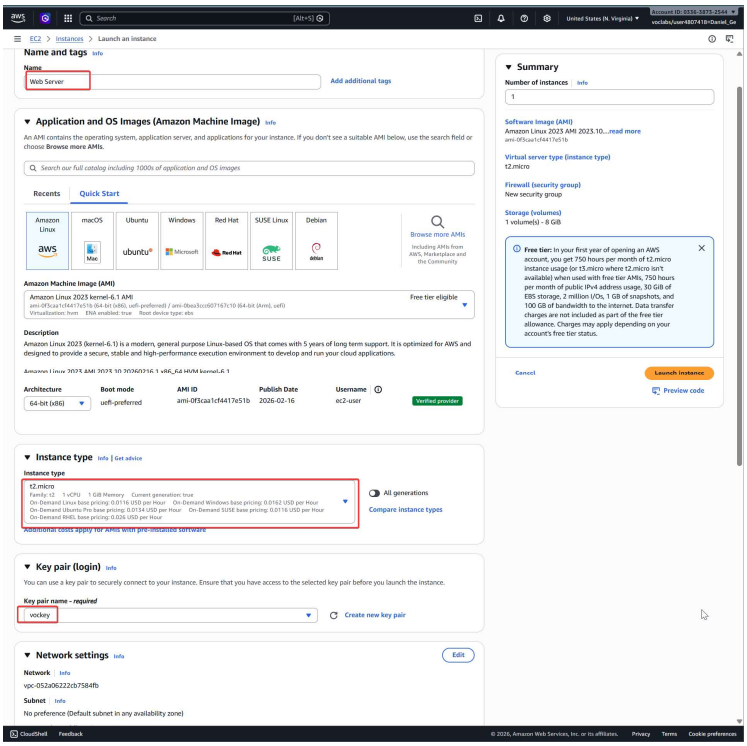
Search “EC2” in the search bar and click the EC2 option to open the EC2 menu



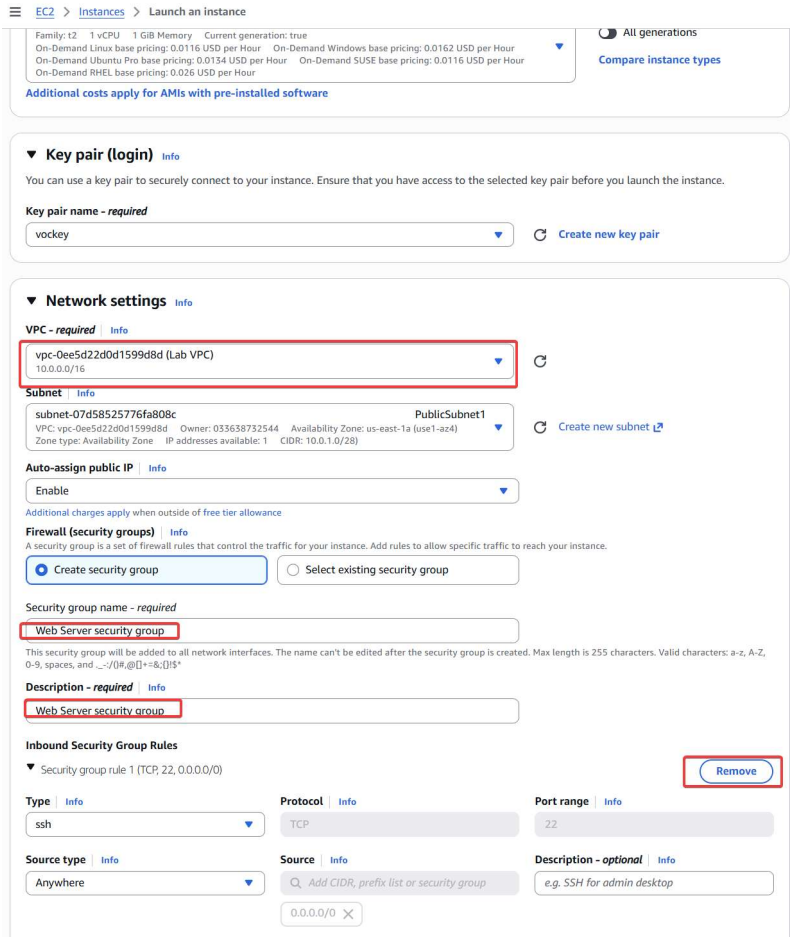
Click “Launch Instance”



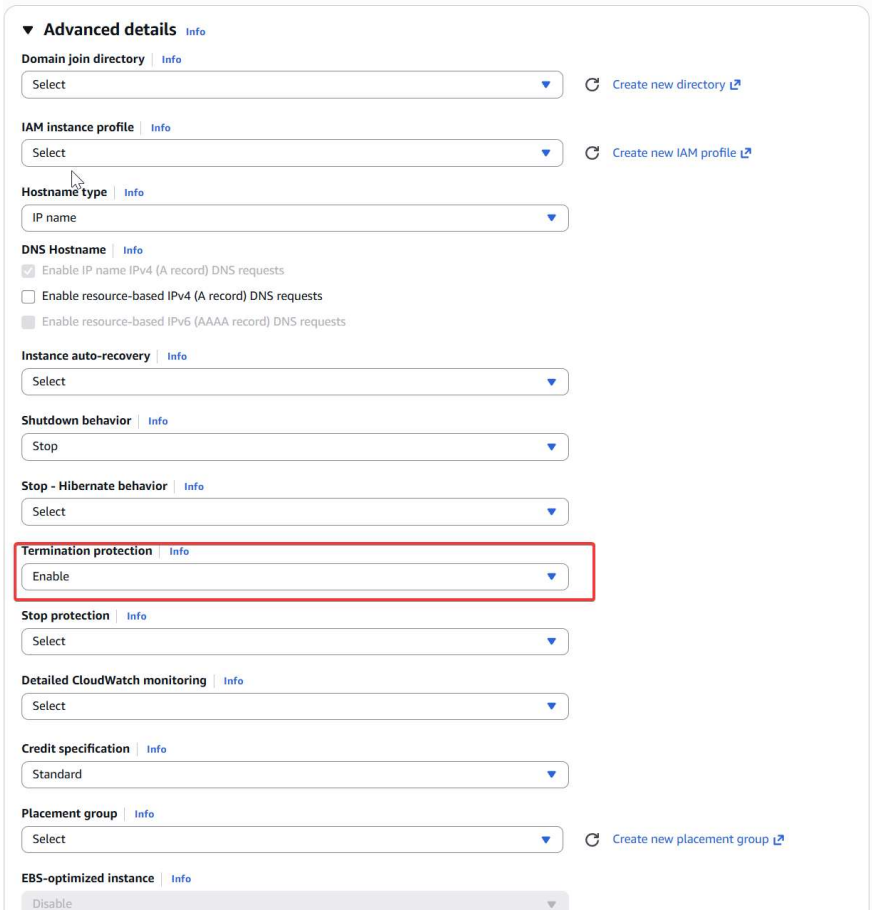
Name the server “Web Server”, set the instance type to t2.micro, and the key pair to “vockey”. Click the blue “edit” button to the right of network settings



Configure the VPC to “Lab VPC”, fill out the security group name and description as follows, and click “Remove” to remove the current inbound security group rule

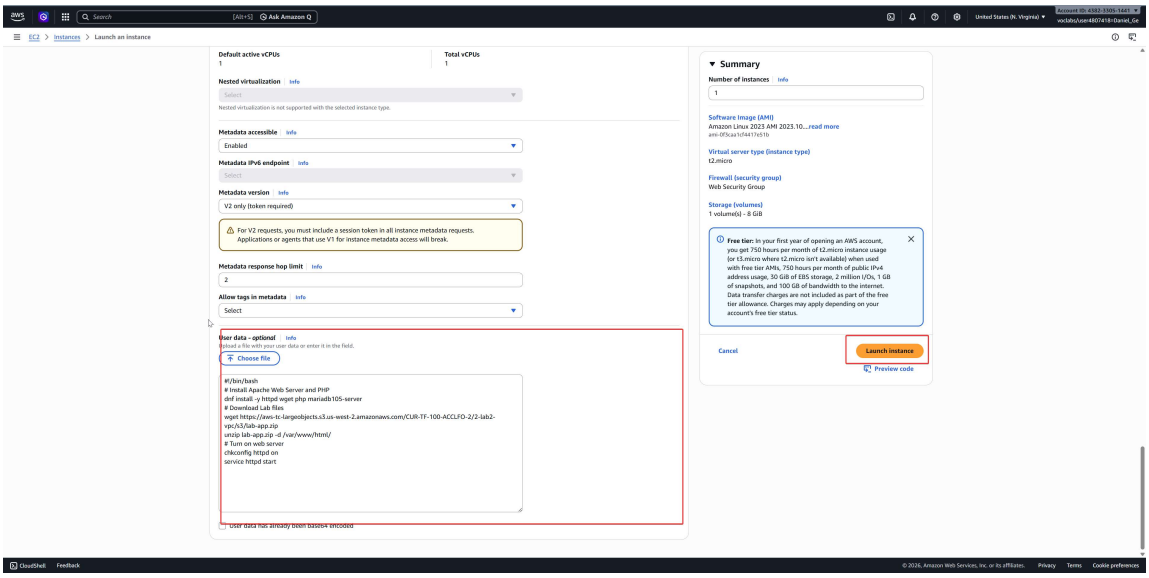


Under Advanced details, enable Termination protection

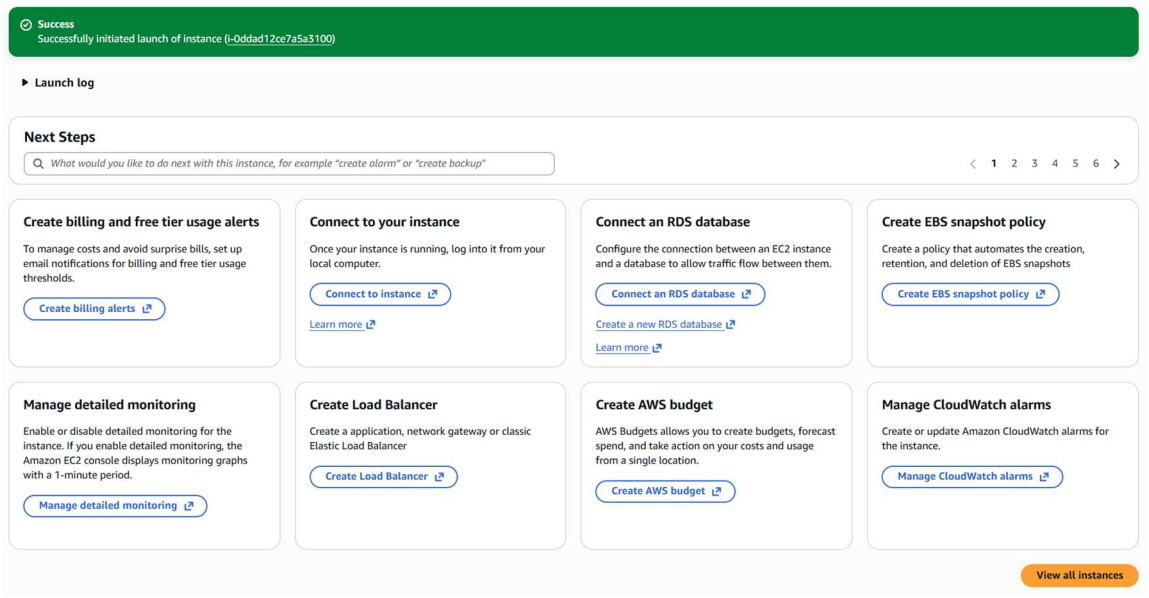


Paste the following code into the User data box, then click Launch instance.

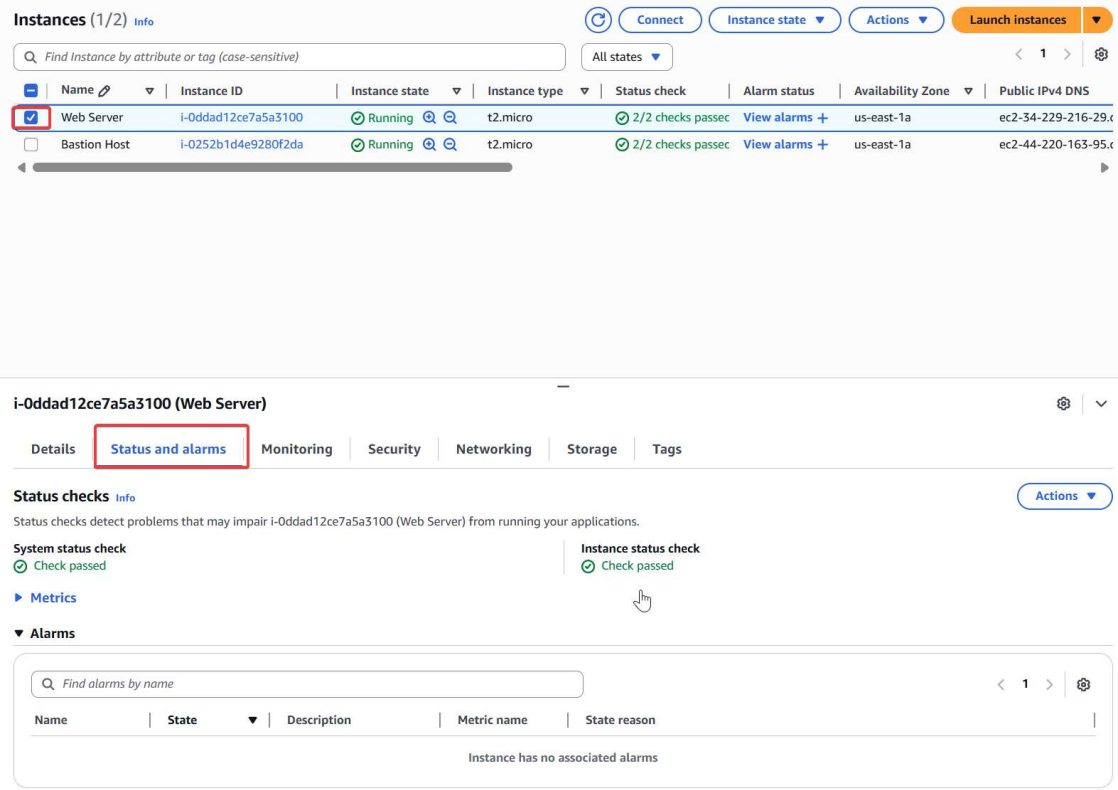
```
#!/bin/bash
dnf install -y httpd
systemctl enable httpd
systemctl start httpd
echo '<html><h1>Hello From Your Web Server!</h1></html>' > /var/www/html/index.html
```



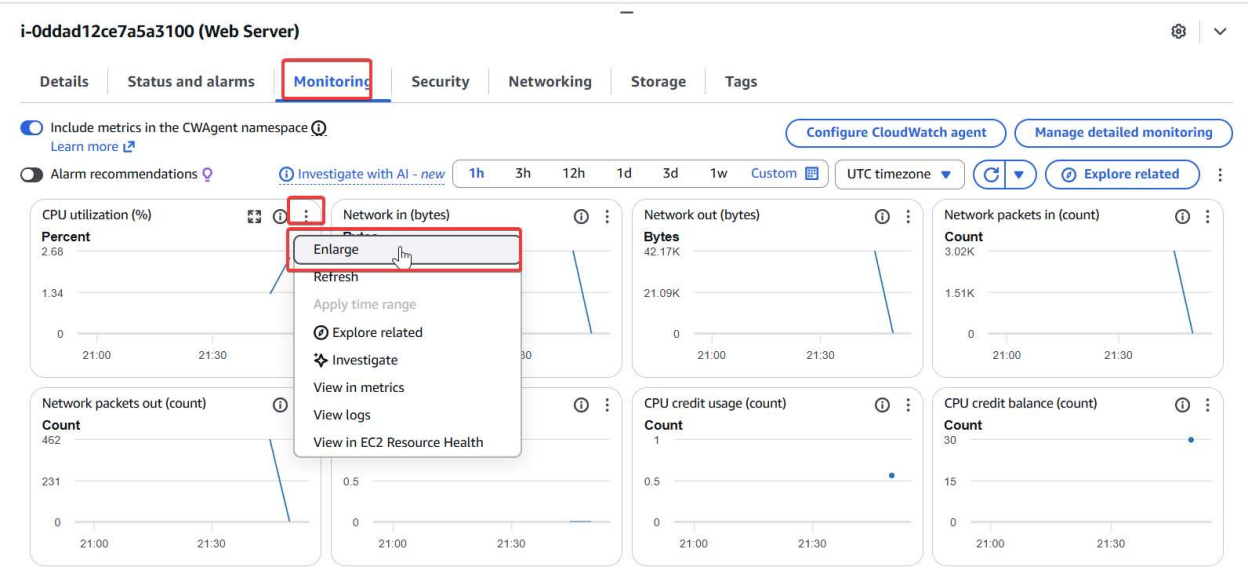
Click “View all Instances”

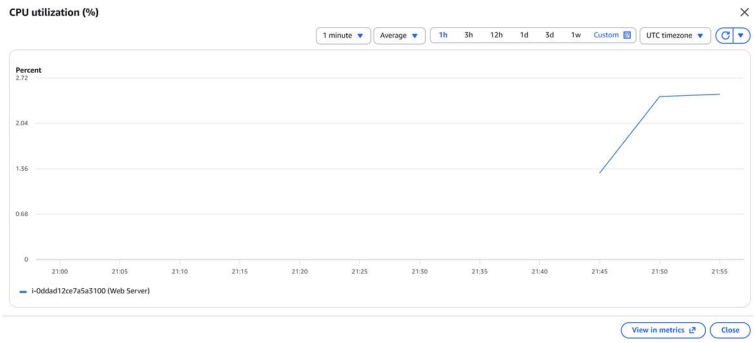


After the status check displays 2/2 checks passed, click the checkbox next to “Web Server”, then the “Status and alarms” tab to observe that both status checks have passed

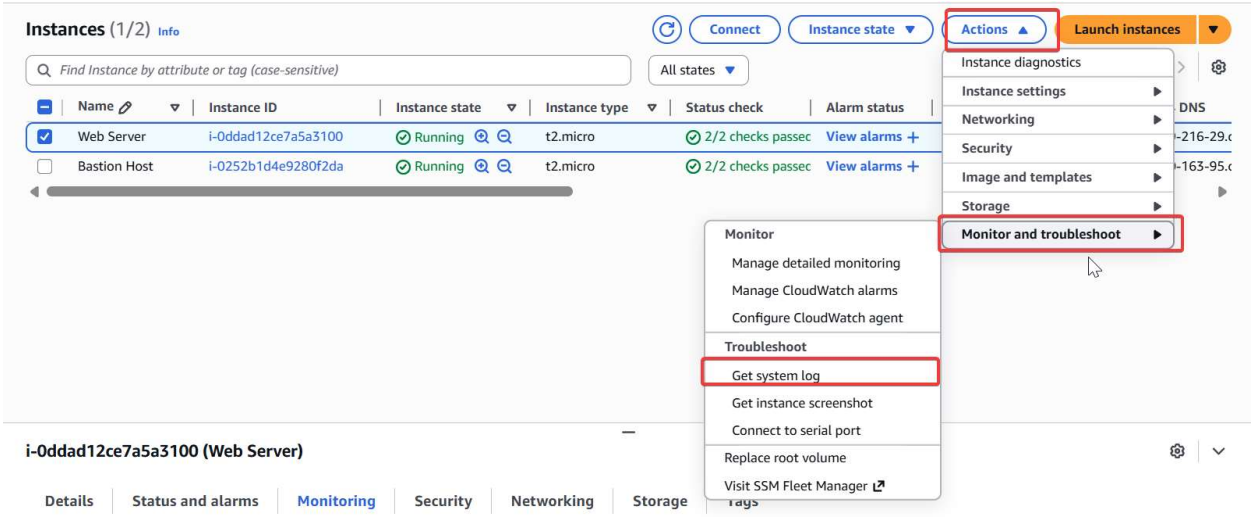


Click “Monitoring” to go to the monitoring tab, then, to enlarge one of the graphs, click the three dots then “Enlarge”

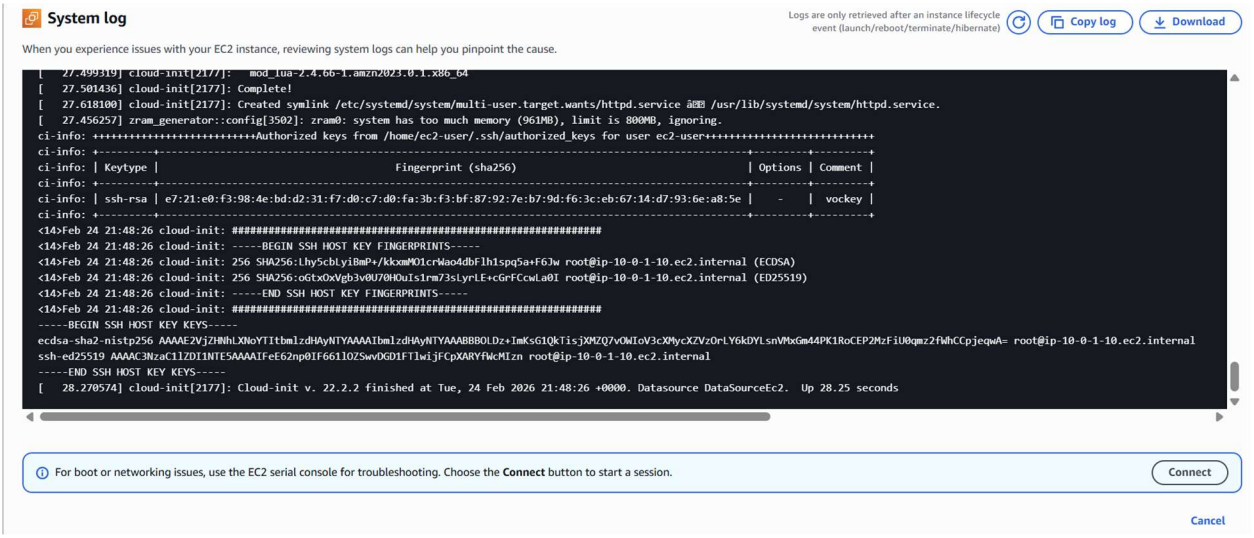




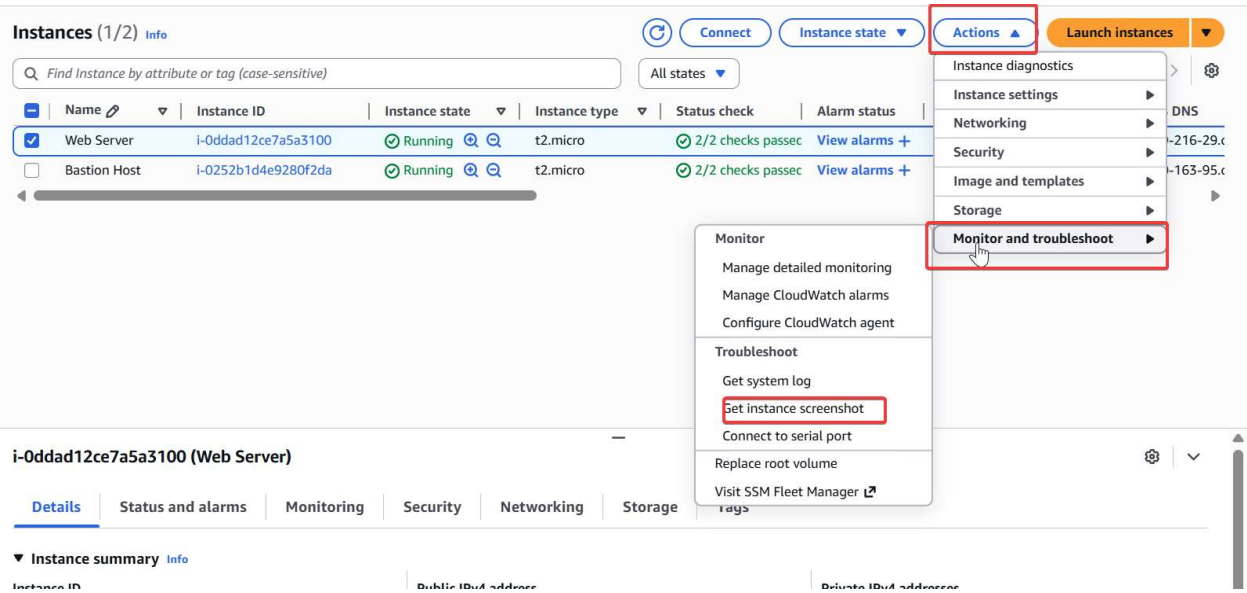
To find the system log, a feature which supports diagnosing the issue during troubleshooting, click “Actions”, “Monitor and troubleshoot”, then “Get system log”



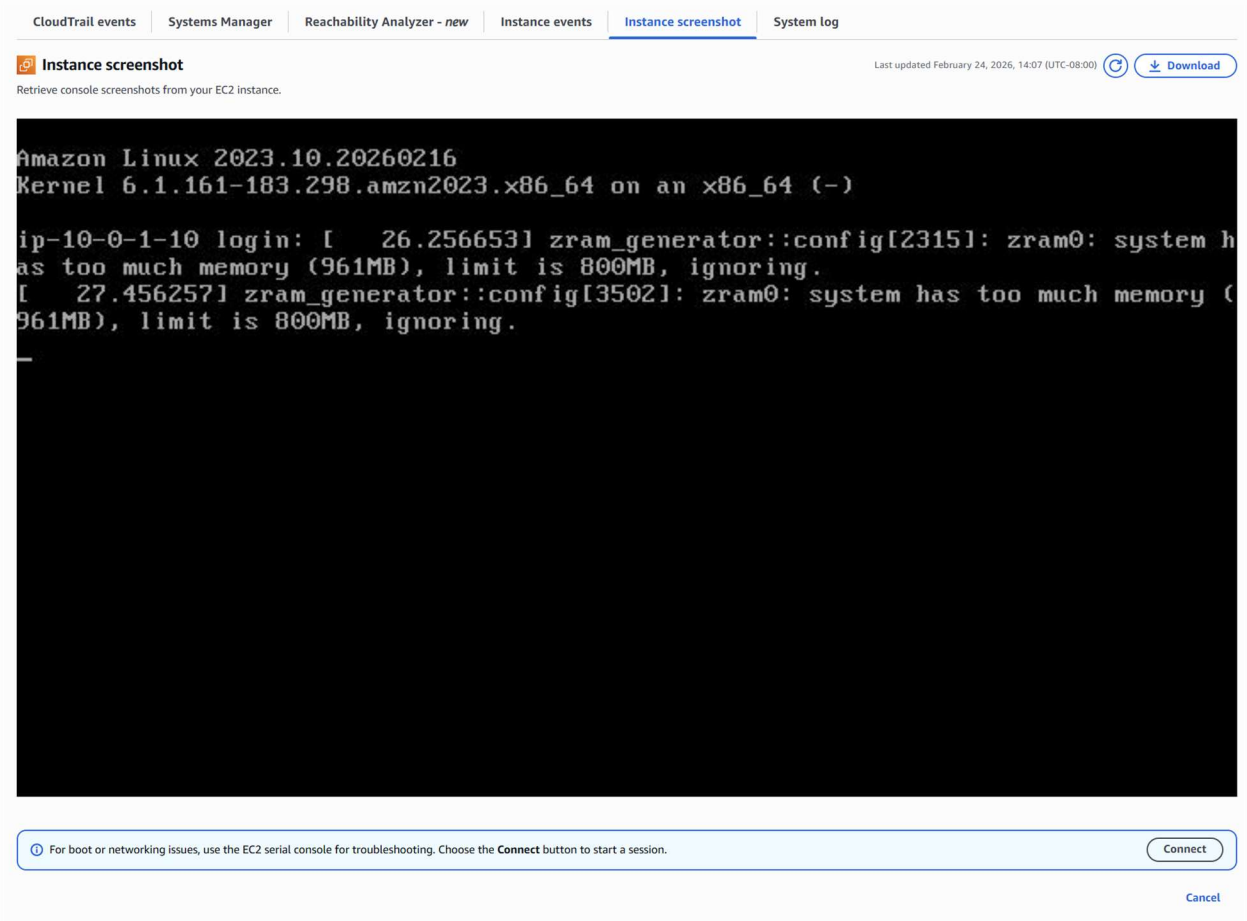
After reading the system log, click cancel at the bottom of the page to exit out



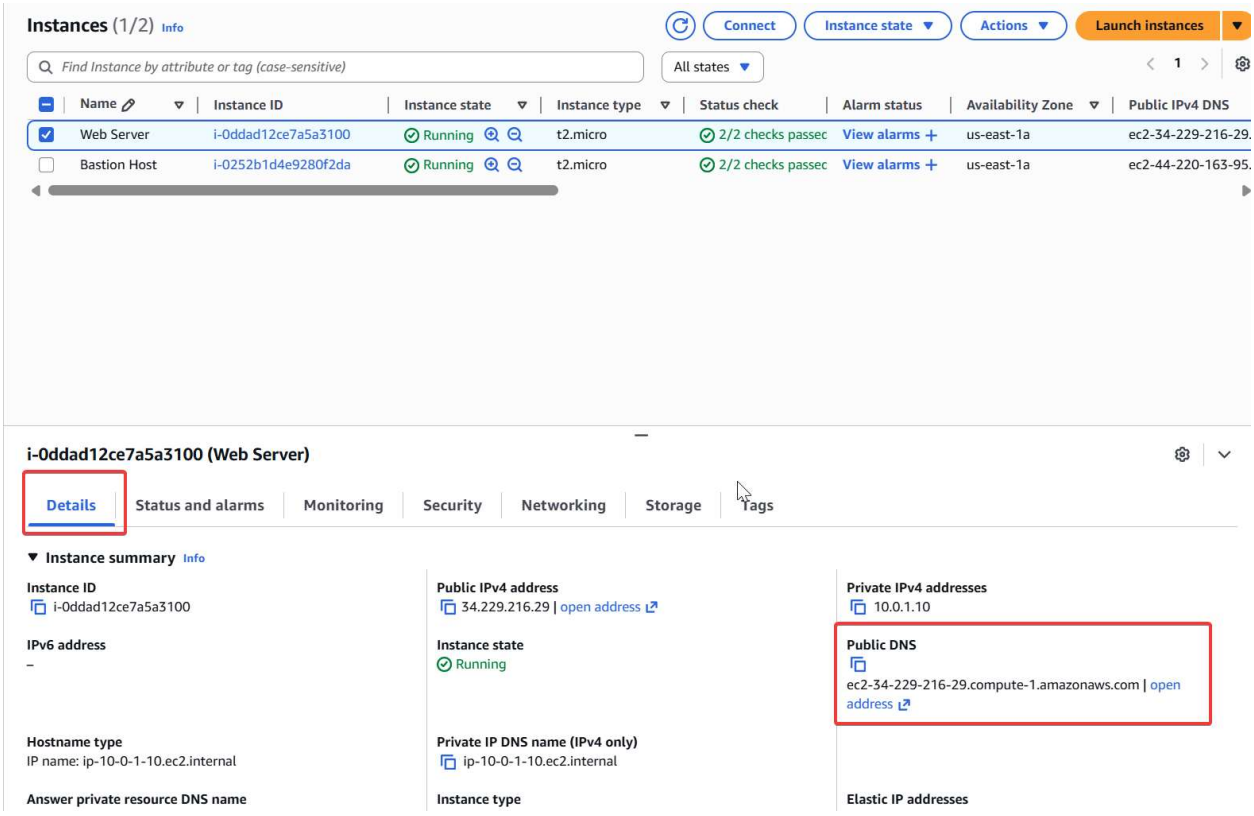
To get an image of what the server would look like if a screen was attached, click “Actions”, “Monitor and troubleshoot”, then “Get instance screenshot”



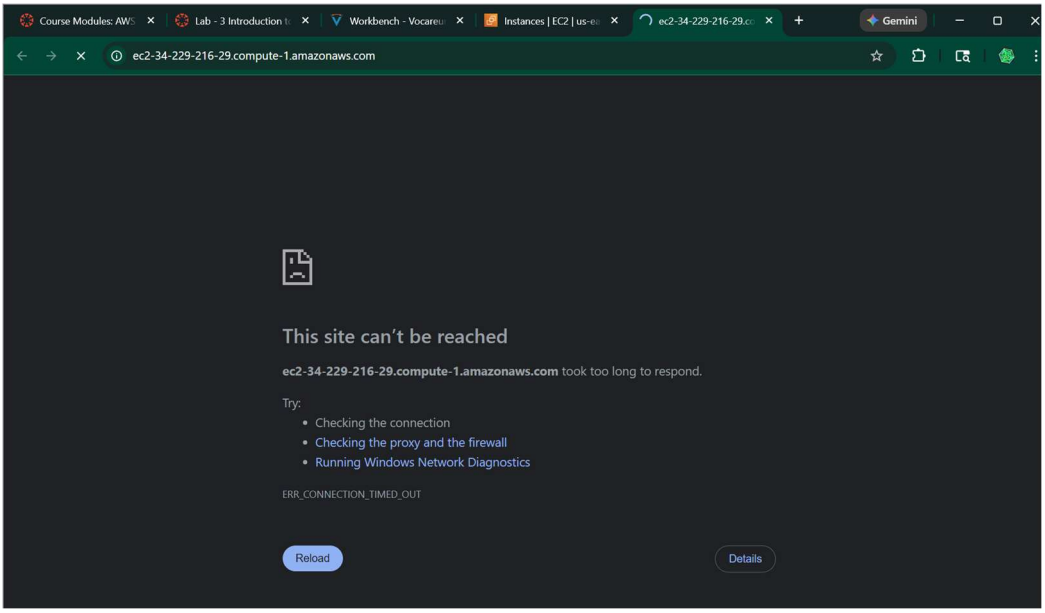
After reading the screenshot and checking for errors, click cancel at the bottom of the page to exit out



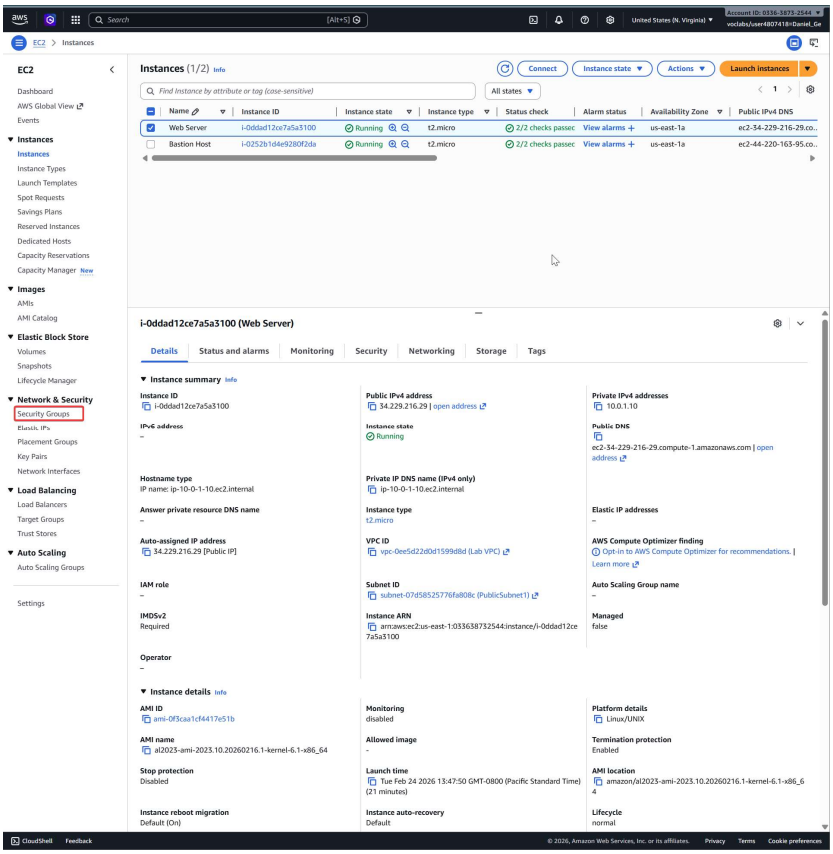
Click “Details” then click “open address” under “Public DNS” to attempt to access the website right now



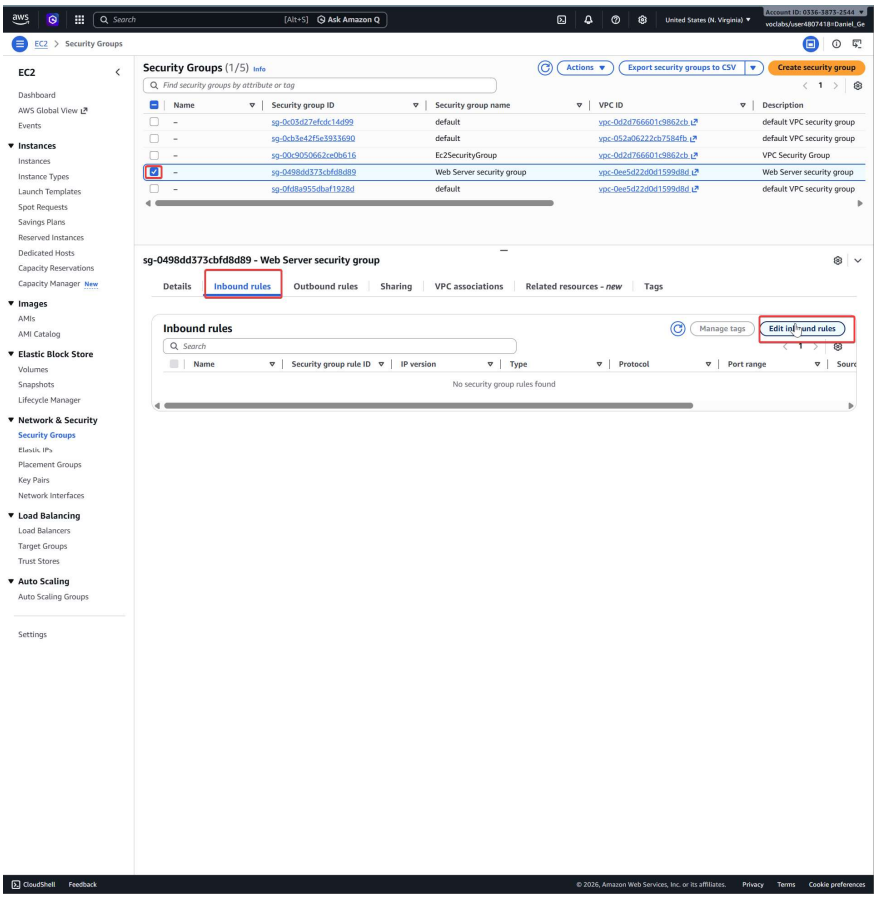
Observe that the website is currently inaccessible since we have not created any inbound security rule allowing traffic to visit the website



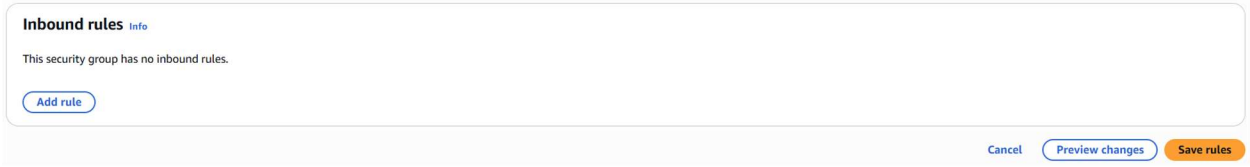
With the browser tab still open, return to the previous EC2 Console tab and click “Security Groups” on the left menu



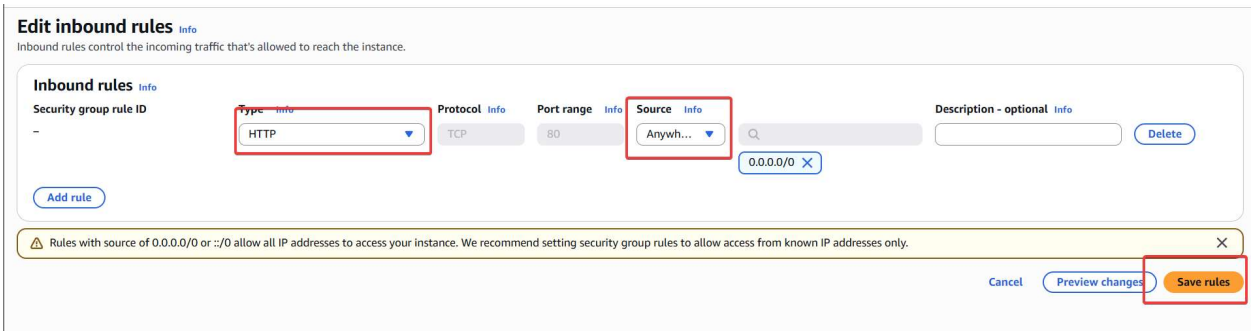
Check the box for “Web Server security group”, click the “Inbound rules” tab, then “Edit inbound rules”



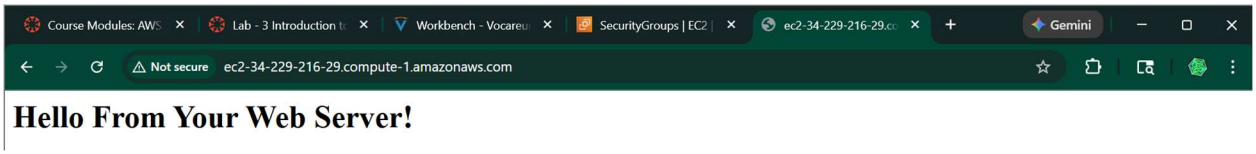
Click “Add rule”



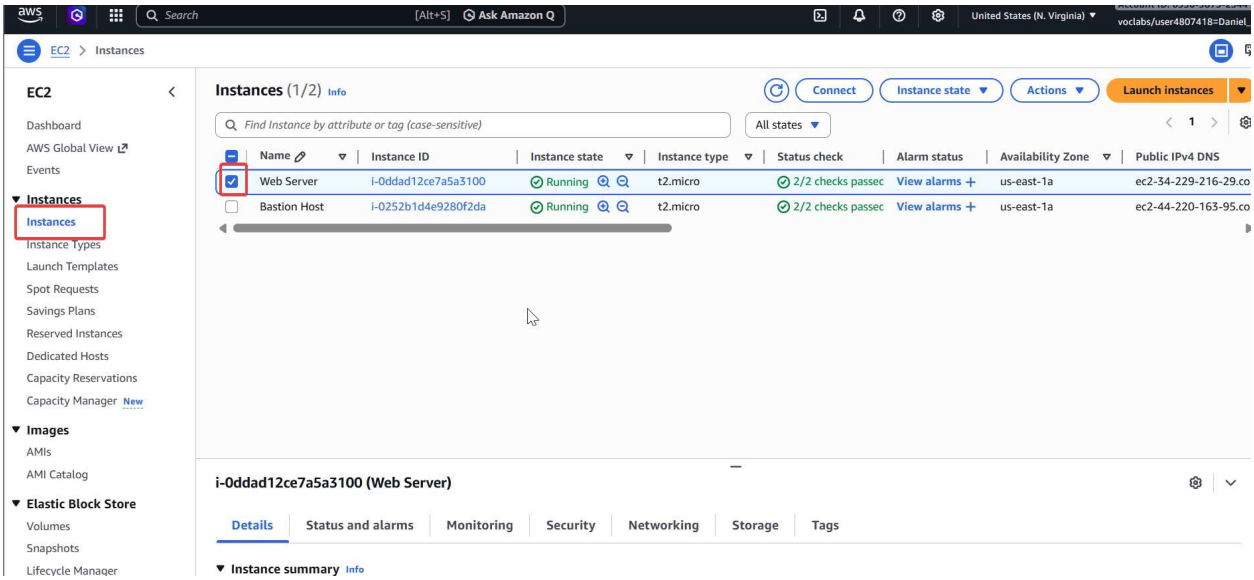
Set the Type to HTTP, Source to “Anywhere IPV4”, then Save rules



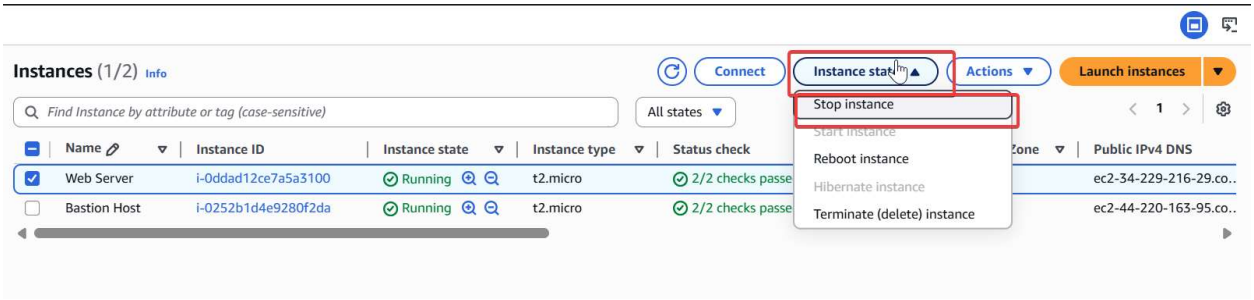
Returning to the tab with the web server, we are now able to access it and see the message



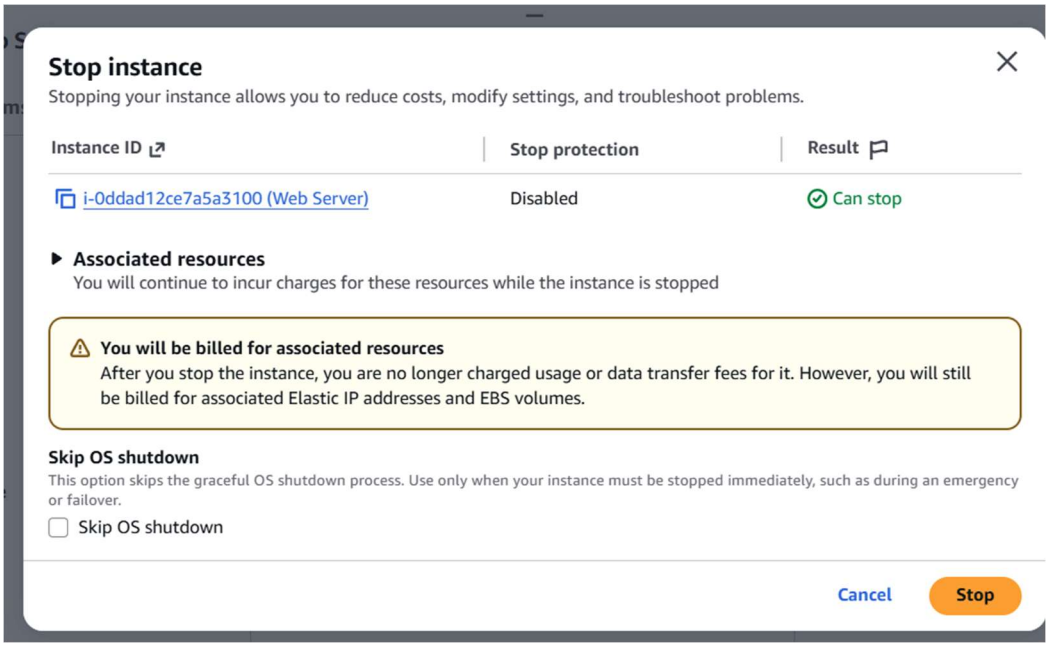
To change the size of the instance, we first must stop the instance. Begin by clicking “Instances” on the left menu, then checkbox the Web Server



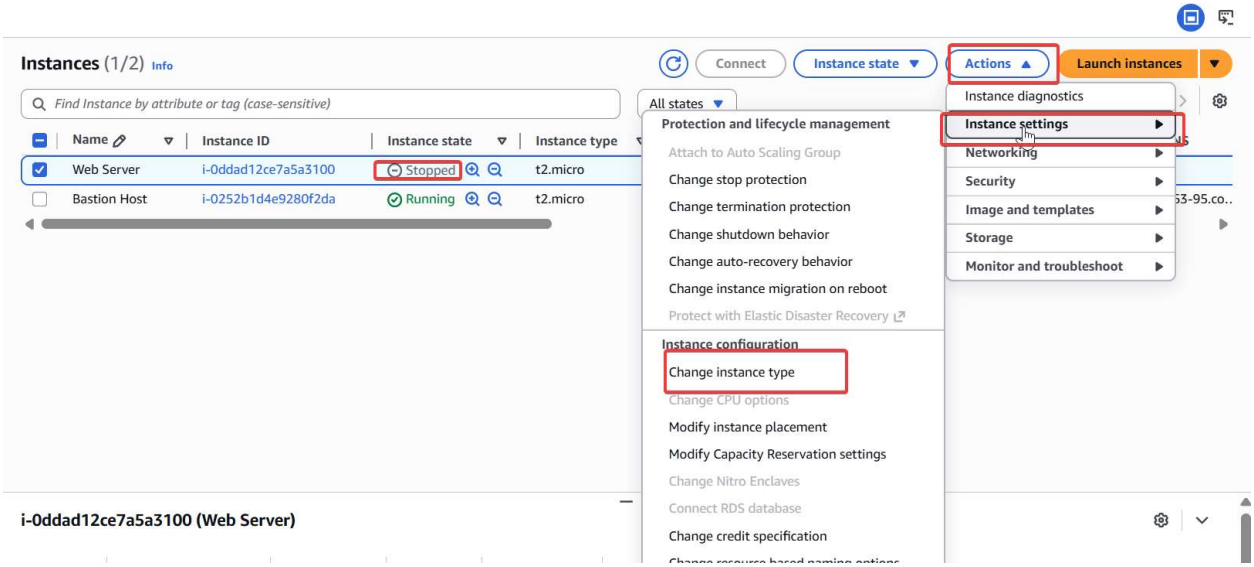
Click “Instance state” then “Stop instance” on the top



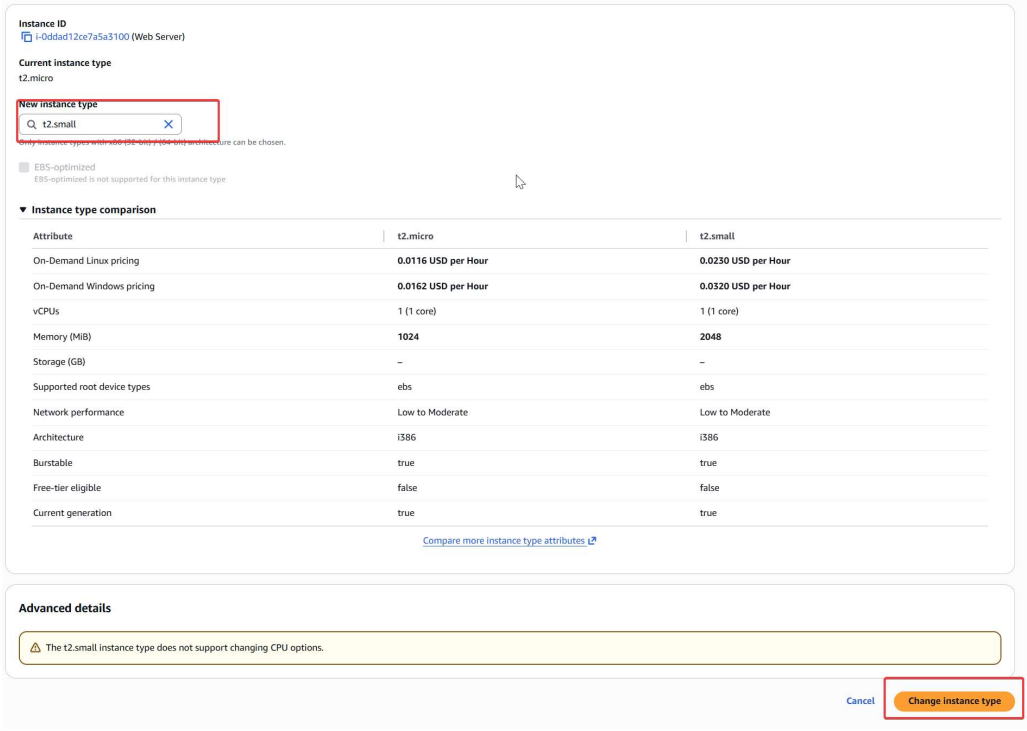
Click “Stop”



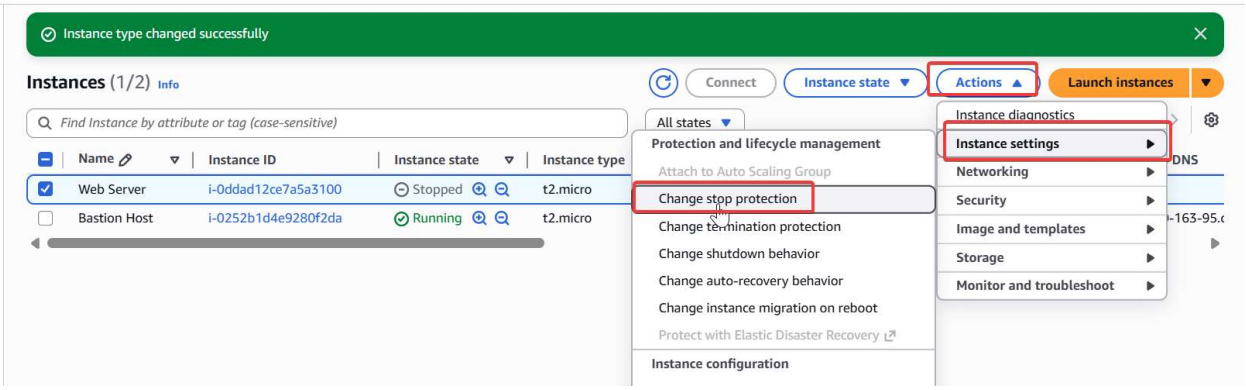
After the Instance state displays “Stopped”, click “Actions” then “Instance settings” then “Change instance type”



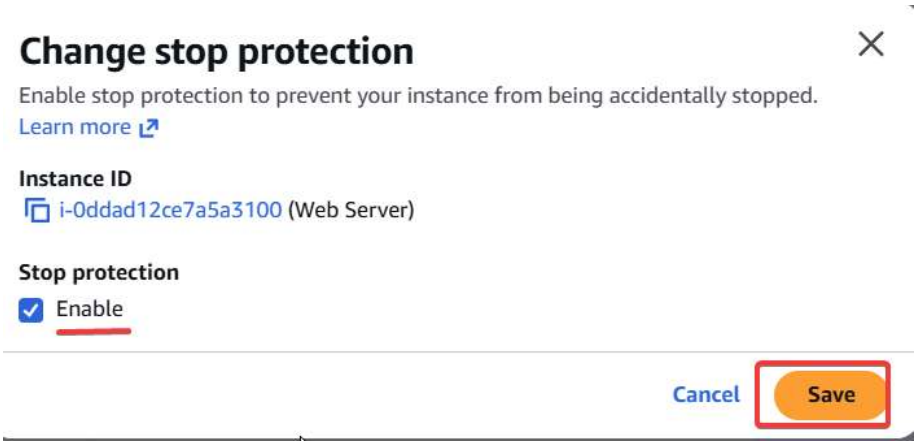
Change the instance type to “t2.small”, then click “Change instance type”



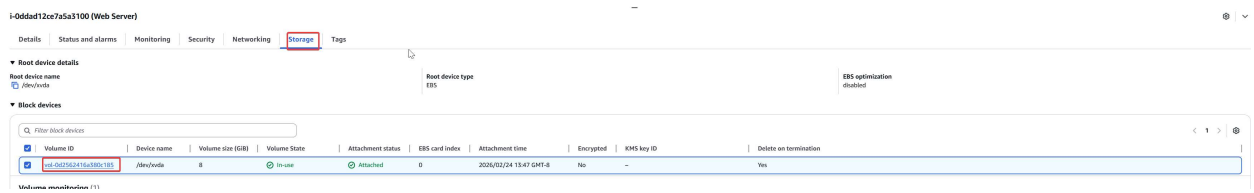
Click “Actions” then “Instance settings” then “Change stop protection”



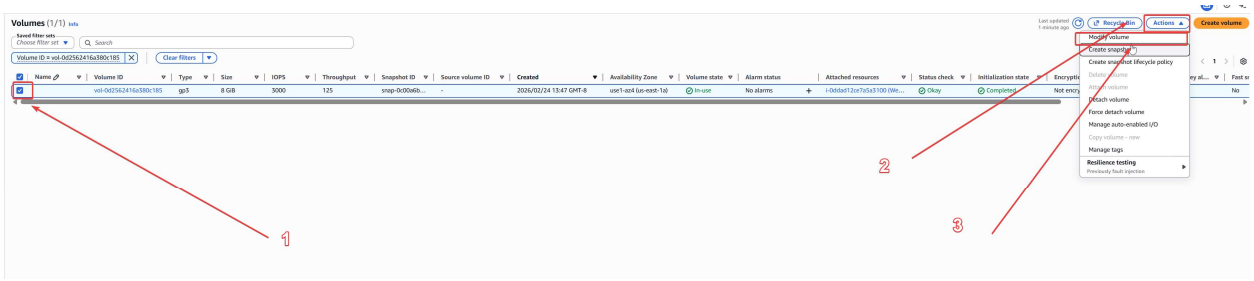
Click “Enable” then “Save”



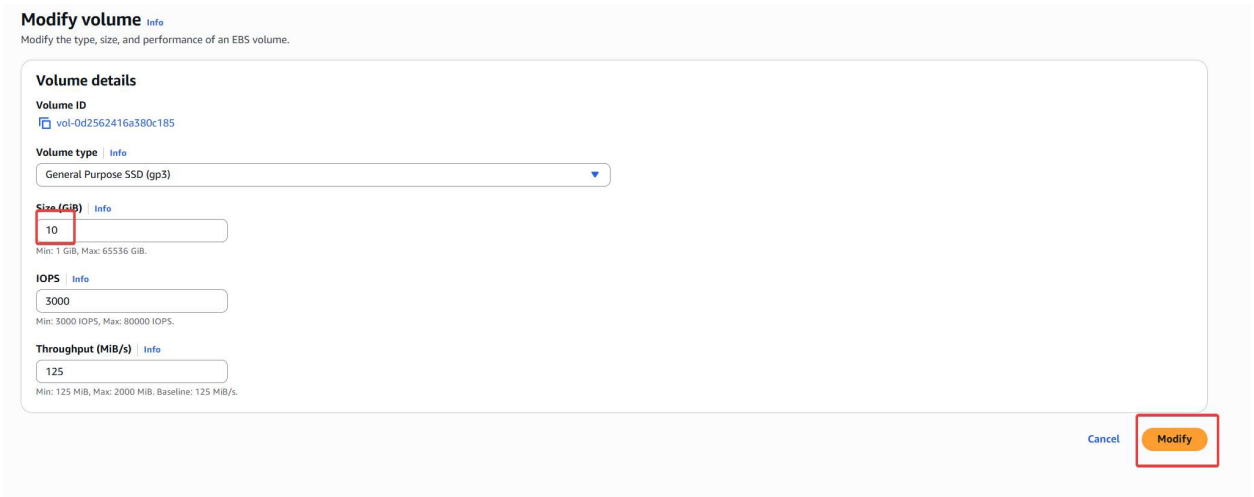
Next, to resize the EBS Volume, click the Storage tab, then select the Volume ID name



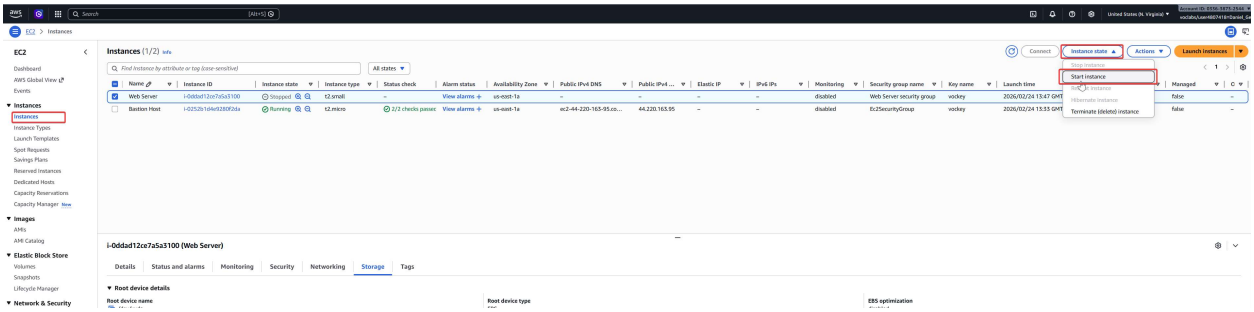
Click the checkbox next to the first volume displayed, then “Actions” and “Modify Volume”



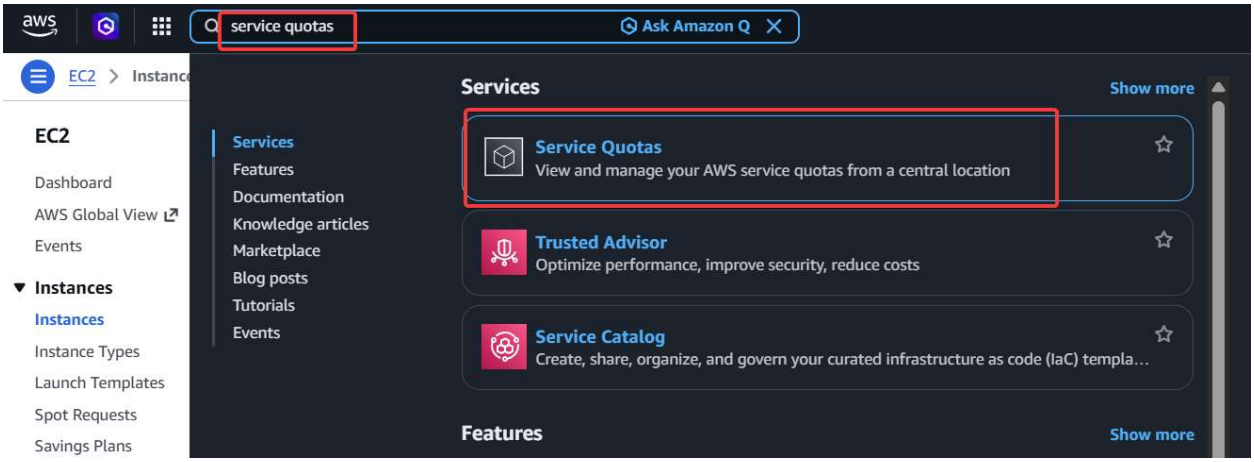
Set the Size to 10 GiB, and click “Modify”



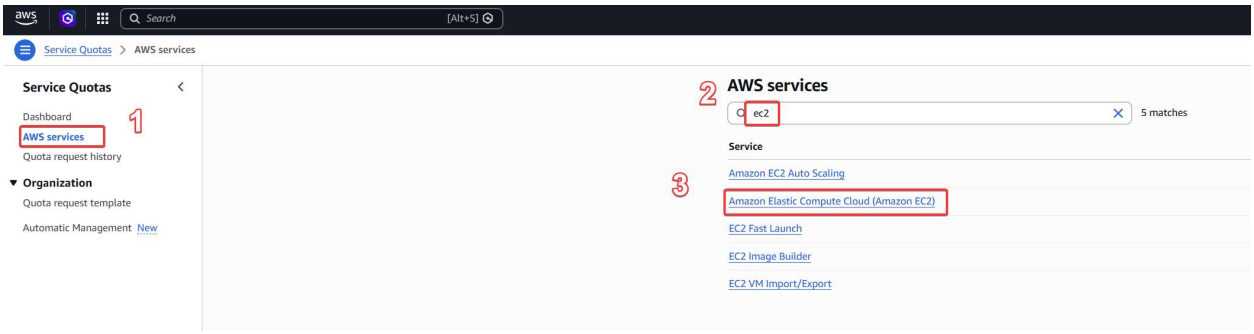
Return to the Instances tab using the left menu, then click “Instance state” and “Start instance”



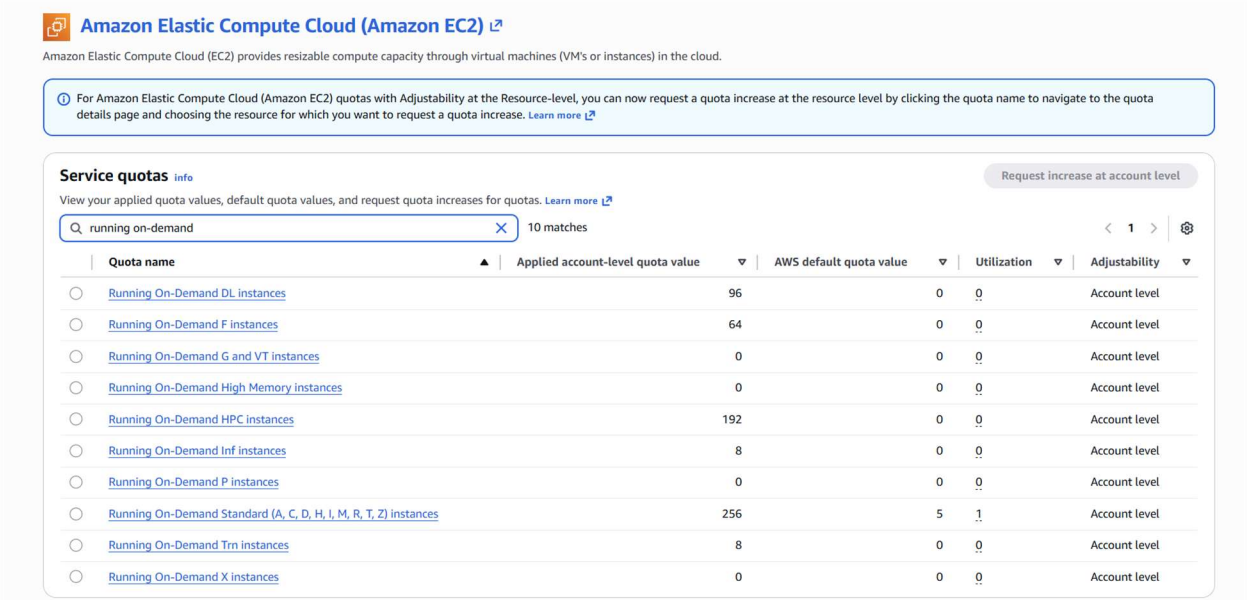
To find information on quotas, search “service quotas” in the search bar



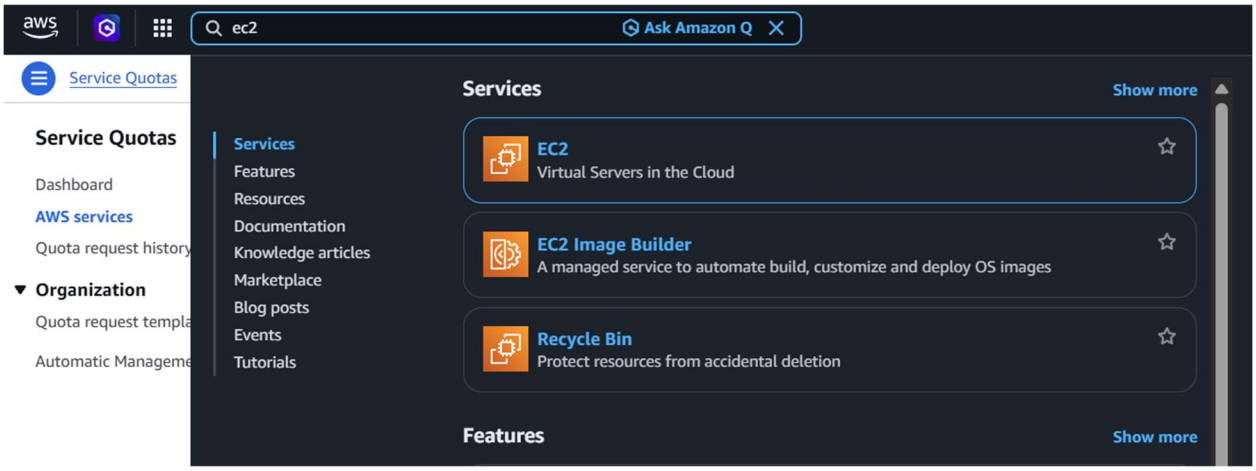
Click “AWS Services” in the left menu, then search for EC2



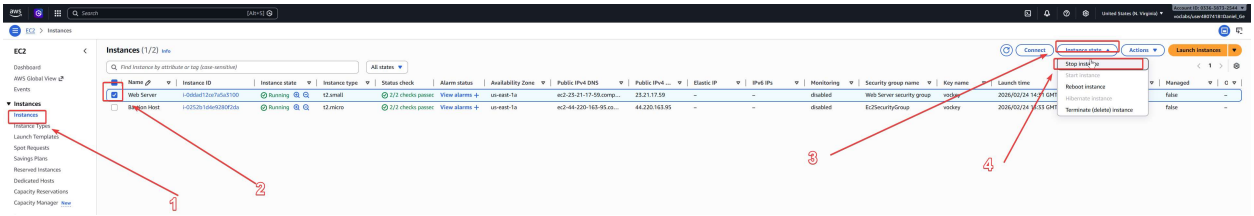
As the EC2 account owner, you can search and modify the Service quotas. For example, to modify the running on-demand quotas, simply search running on-demand in the search bar



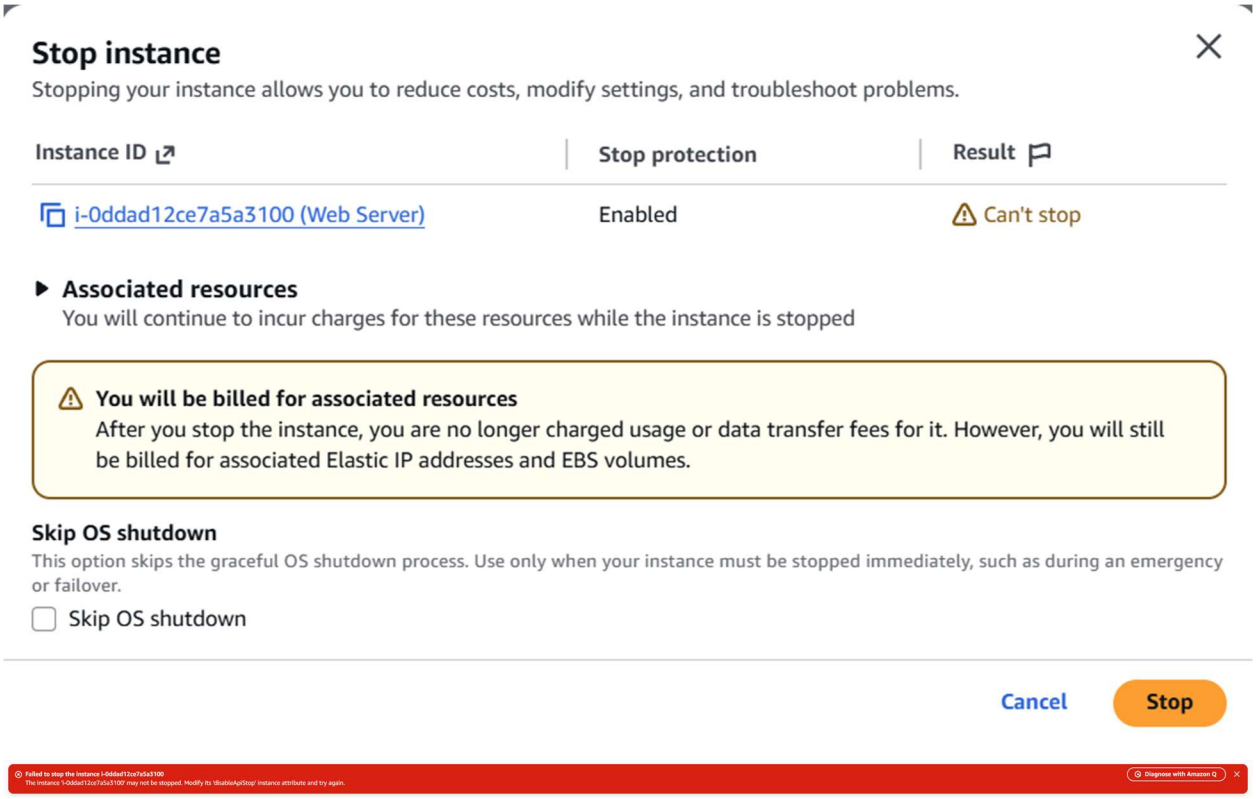
To test the stop protection, return to the EC2 menu



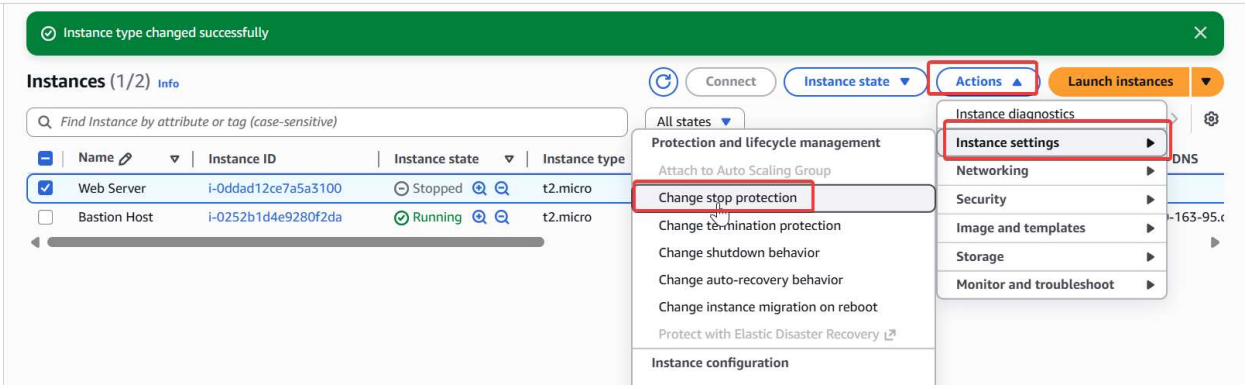
Return to the instances tab, check the checkbox next to Web server, and click “Instance state” then Stop instance



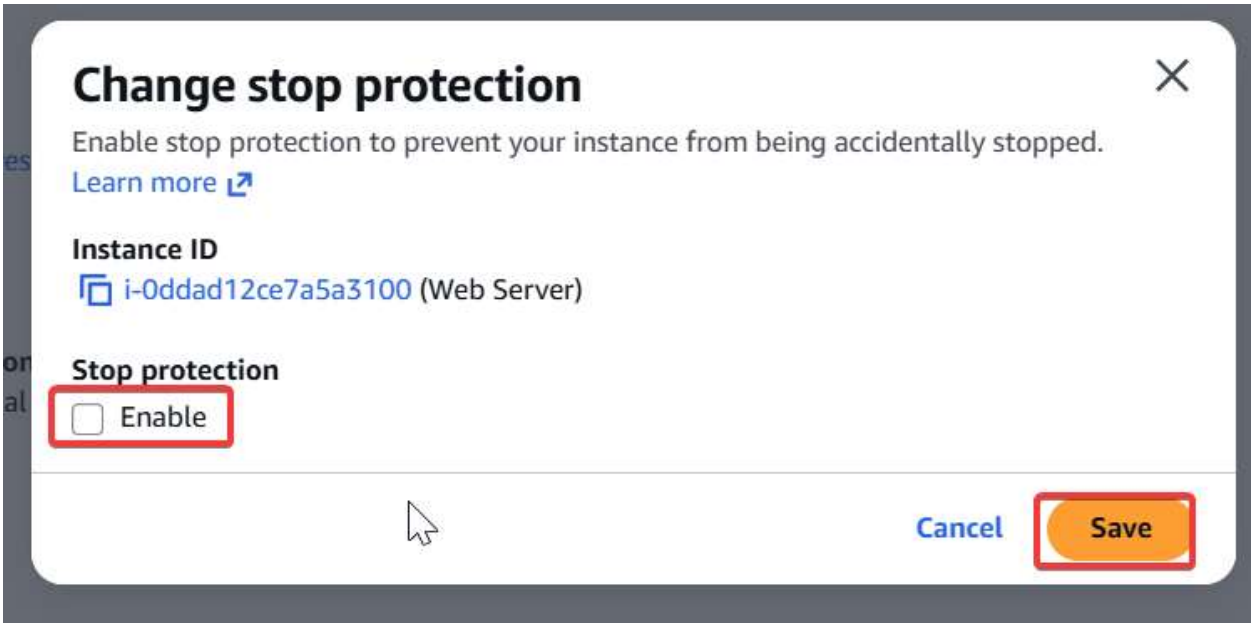
Click “Stop”. Observe that the instance cannot be stopped since stop protection is enabled



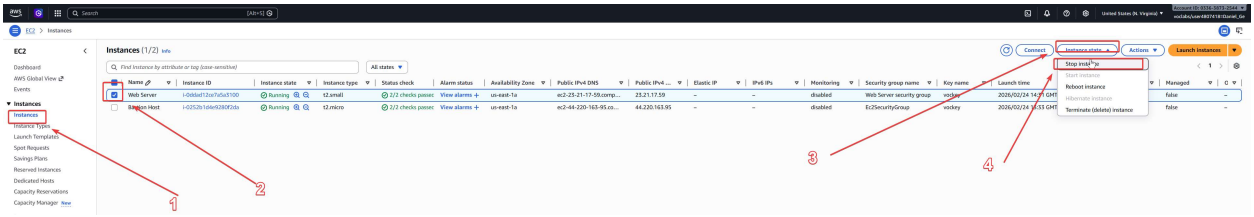
Click “Actions” then “Instance settings” then “Change stop protection”



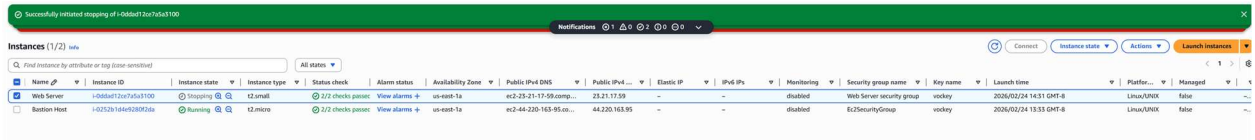
Uncheck the “Enable” box for stop protection, then click “Save”



Follow the same steps to stop the instance

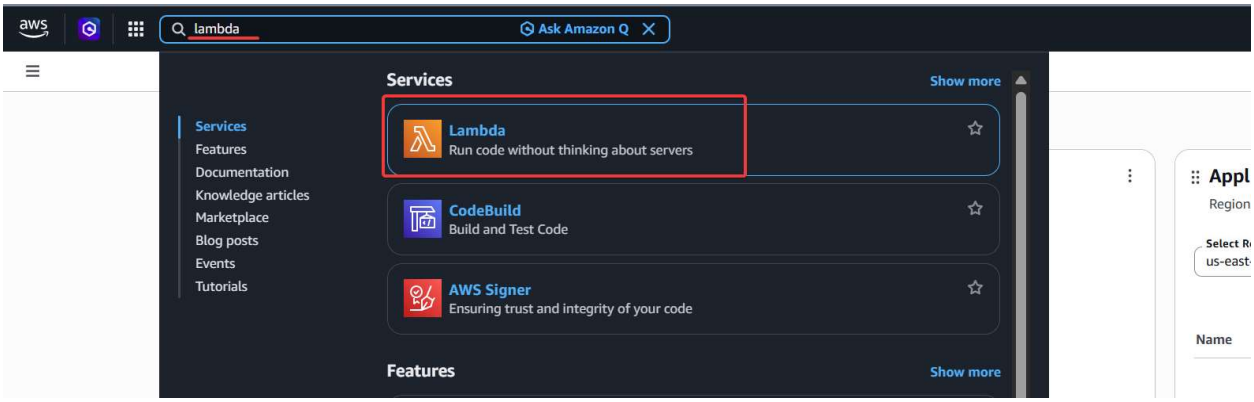


This time, the stop should be successful since stop protection is disabled

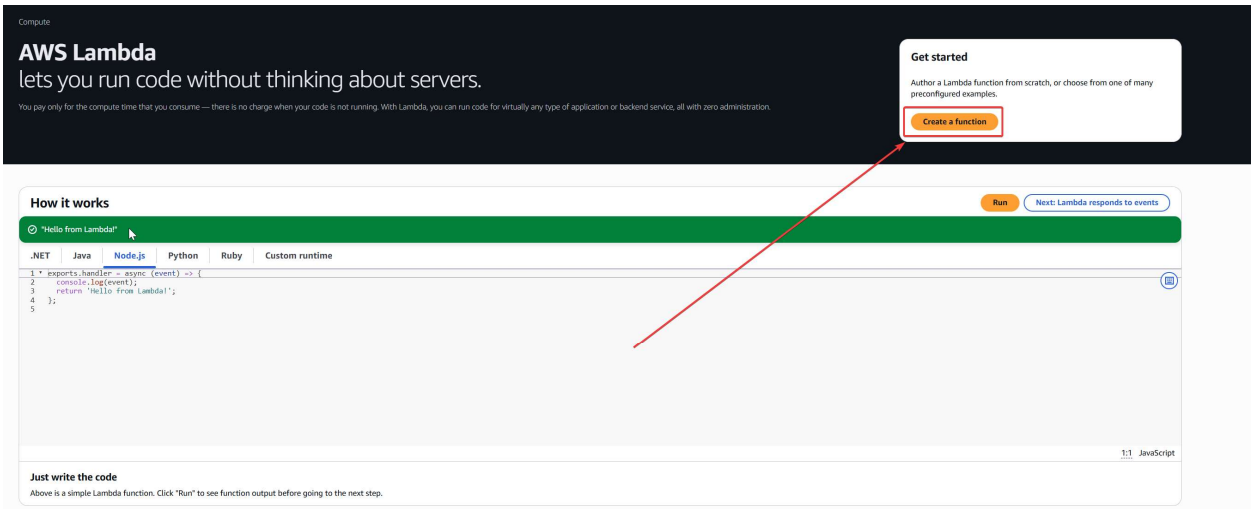


AWS Lambda

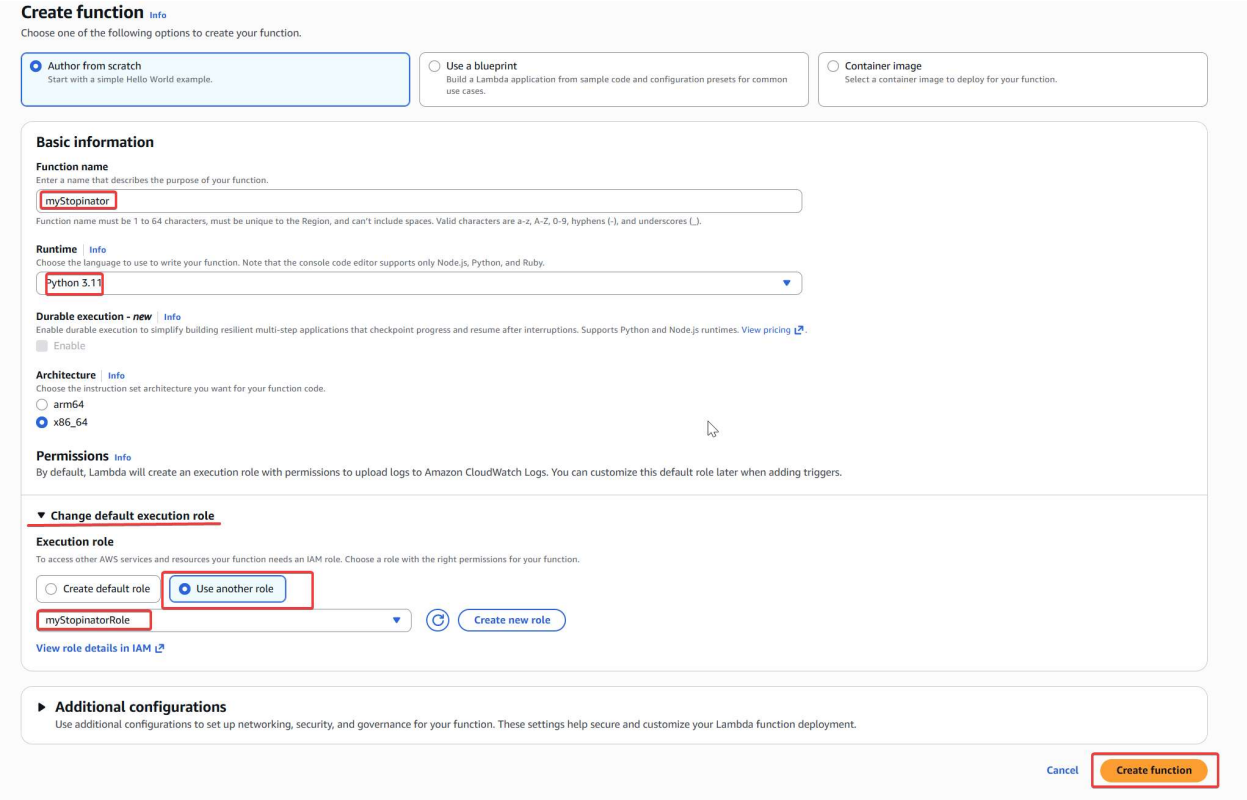
Using the search bar next to services, search for “Lambda” and open to AWS Lambda console



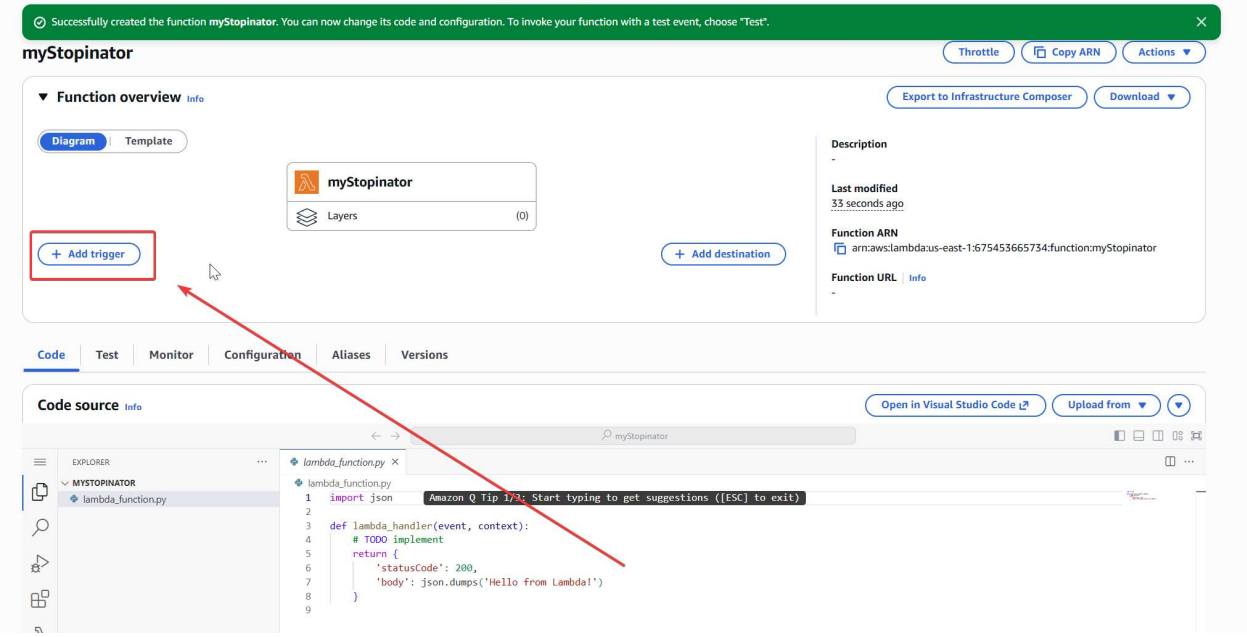
Click “Create a Function”



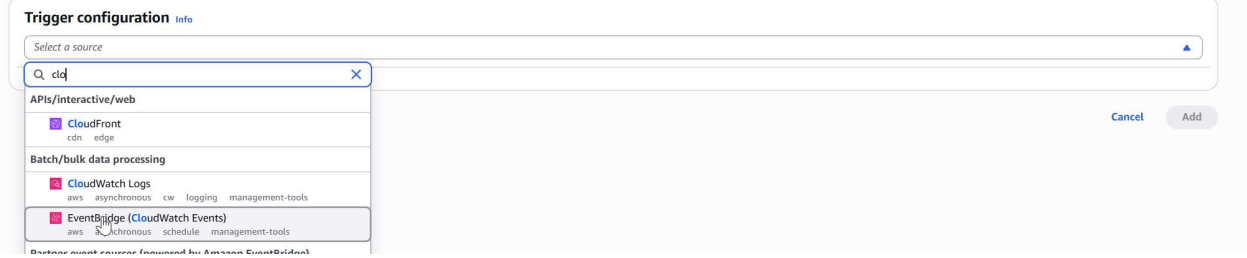
Configure the Python function with the following settings and permissions



To create a trigger which automatically activates the function, click “Add trigger”



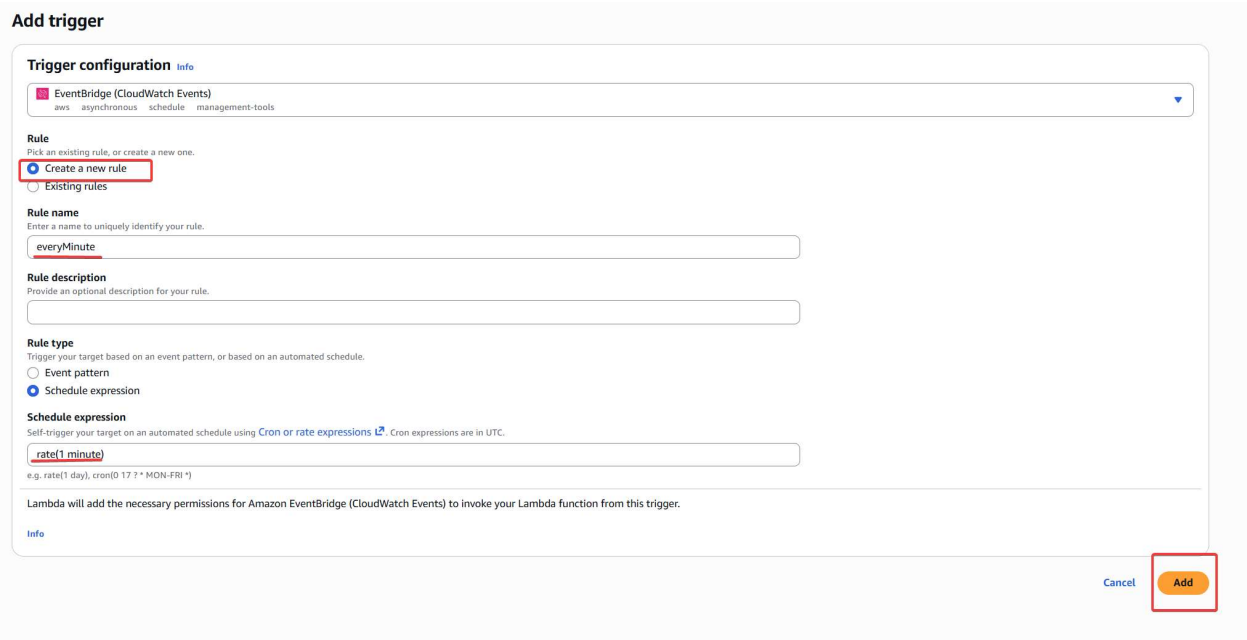
Link the trigger to CloudWatch by selecting **“EventBridge (CloudWatch Events)”** from the source options



Click “Create a new rule” to create a rule with,

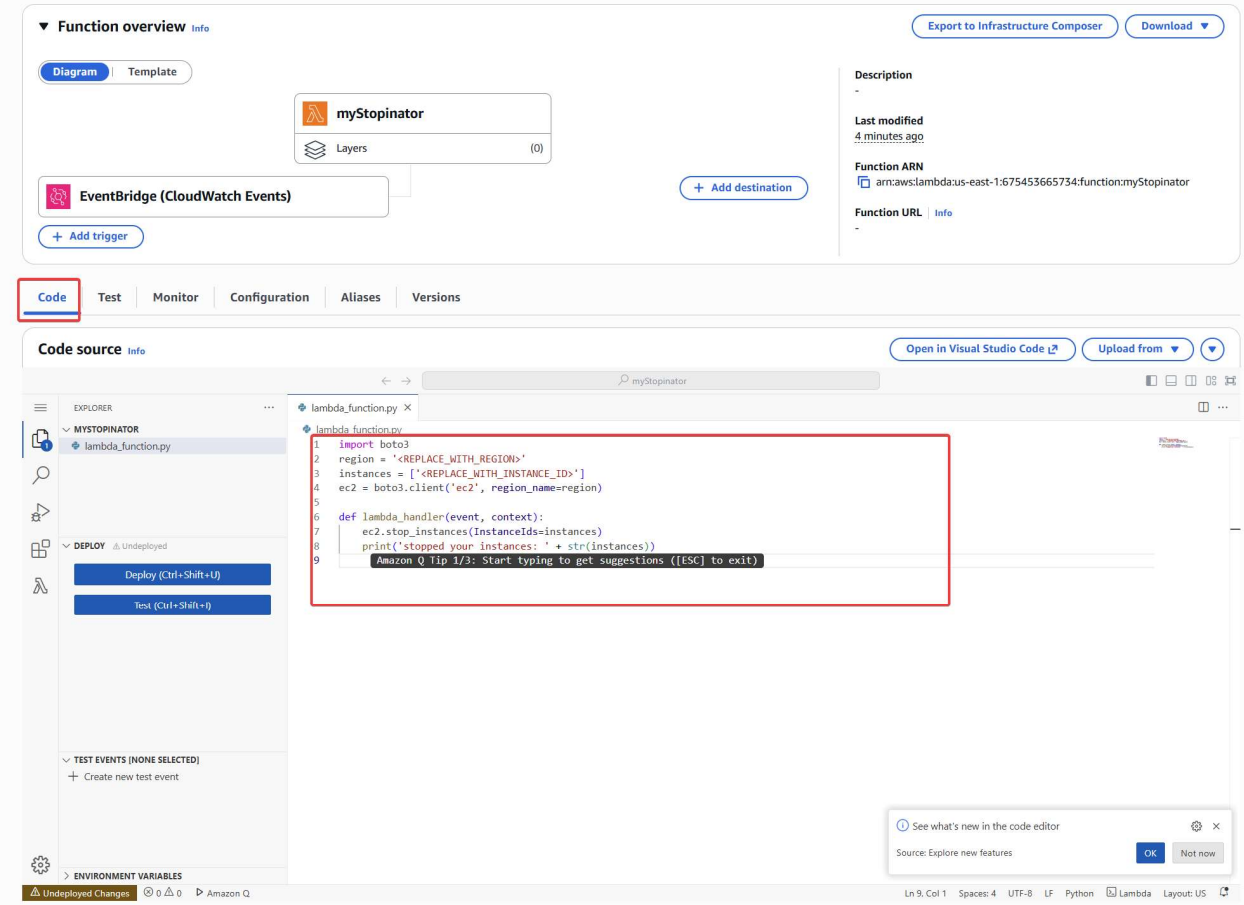
- Name: everyMinute
- Schedule expression: rate(1 minute)

Click Add to add the rule

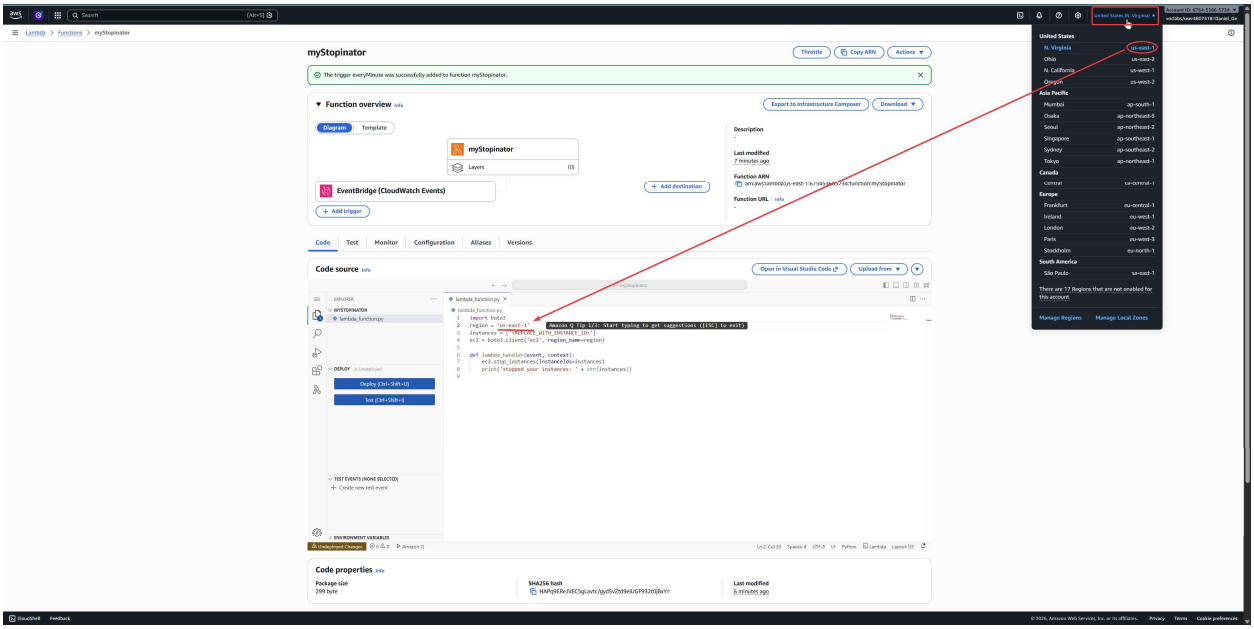


Click the “Code” tab and replace the current content of the text editor with the following code

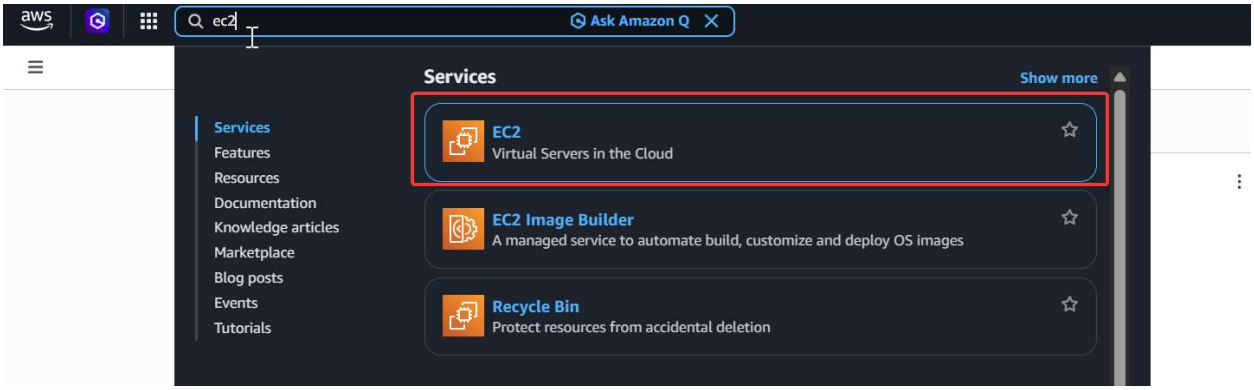
```
import boto3
region = '<REPLACE_WITH_REGION>'
instances = ['<REPLACE_WITH_INSTANCE_ID>']
ec2 = boto3.client('ec2', region_name=region)

def lambda_handler(event, context):
    ec2.stop_instances(InstanceIds=instances)
    print('stopped your instances: ' + str(instances))
```



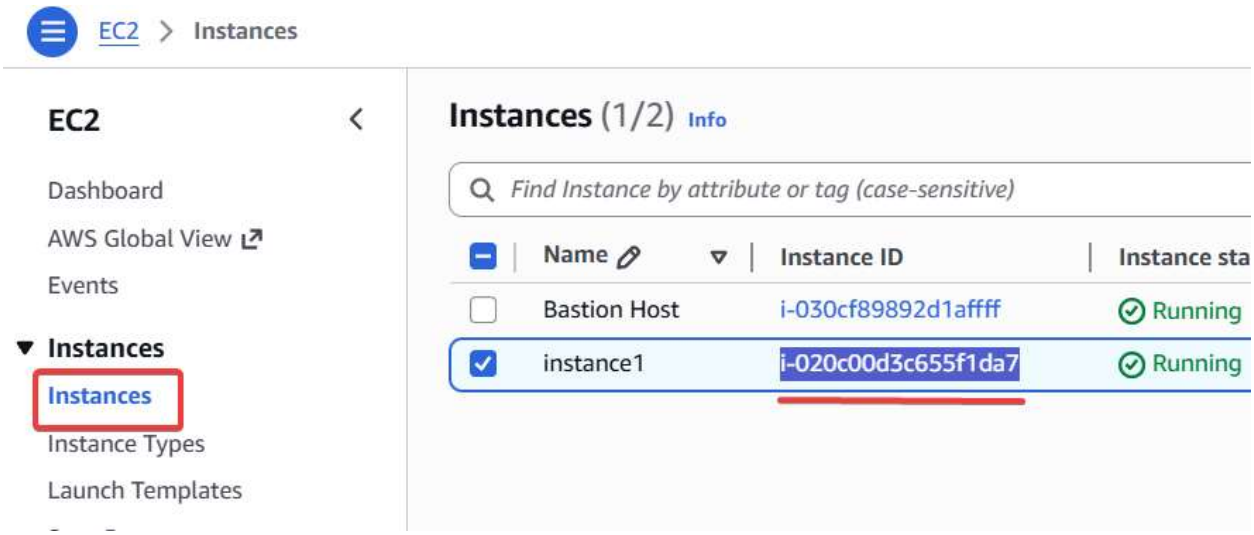
Click the tab on the top right displaying the user's current region to find the region name, then replace "`<REPLACE_WITH_REGION>`" with this name



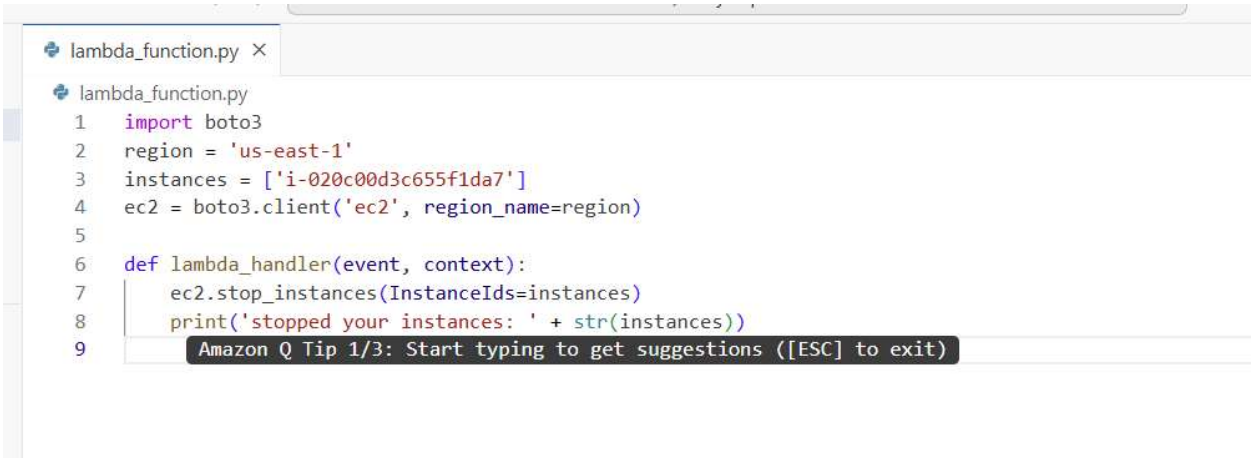
Open the EC2 menu in a new tab



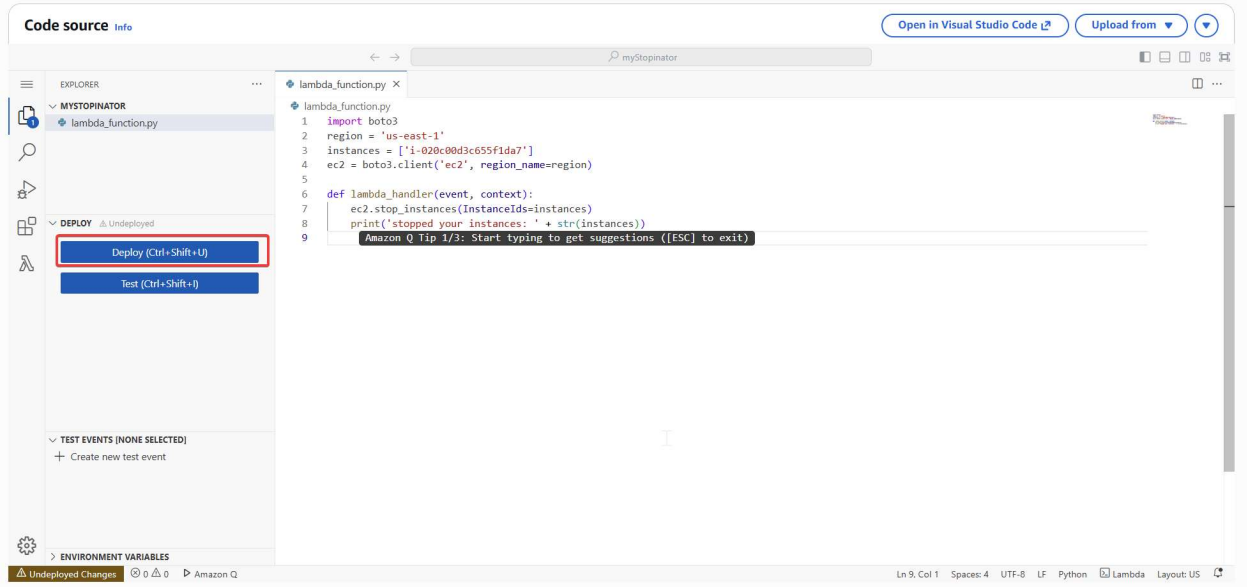
Click “Instances” on the left menu, then copy the Instance ID of “instance1”



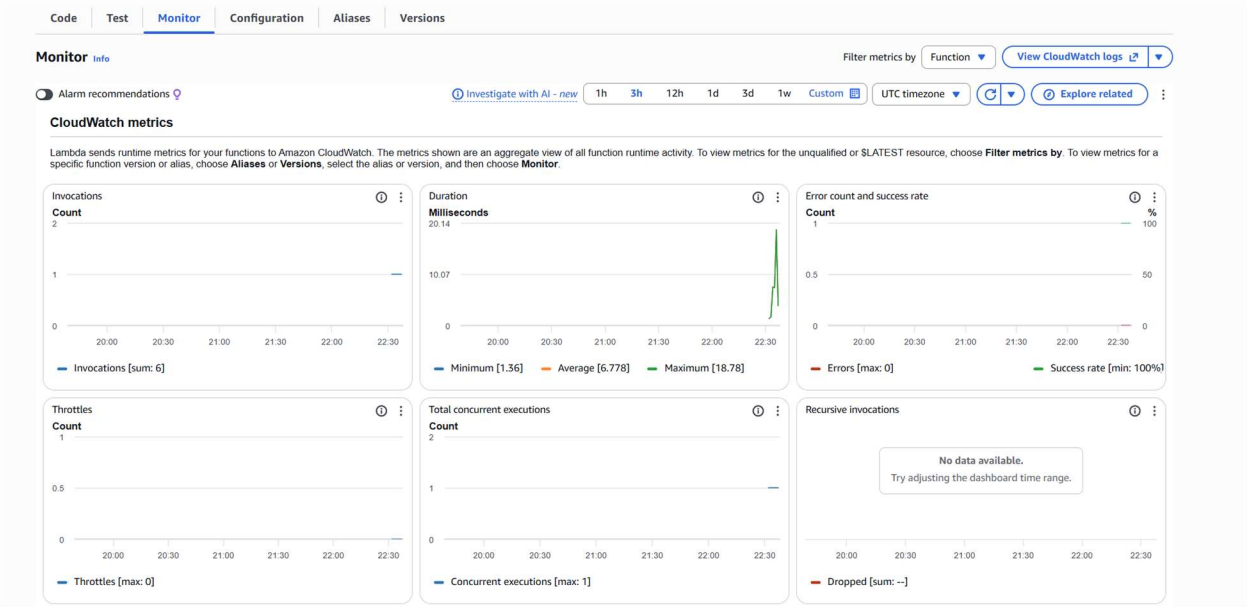
Copy this instance ID to the code



Click “Deploy”

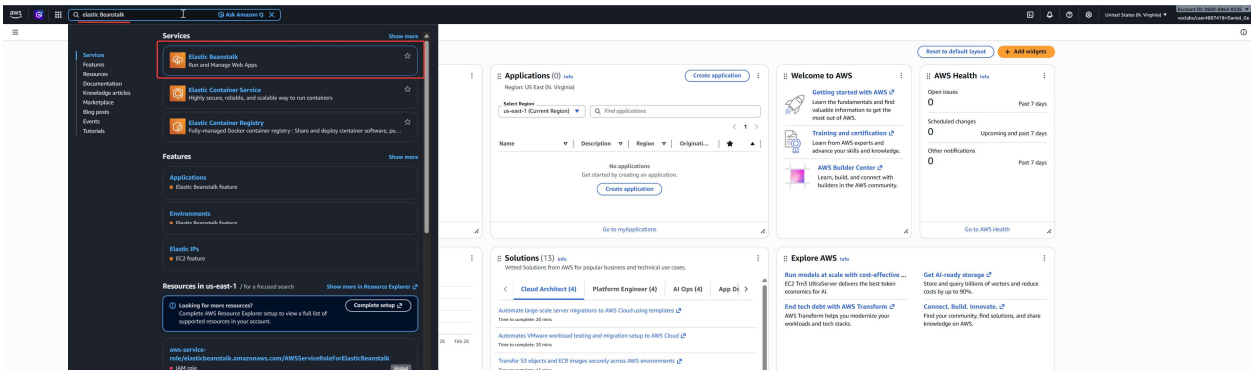


Click the Monitor tab to see if there are any issues reported, as well as other telemetry about how the code is running

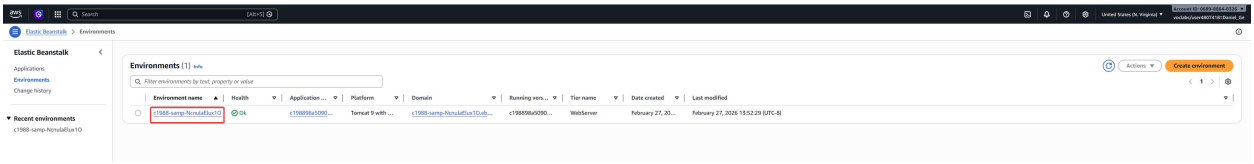


AWS Elastic Beanstalk Activity

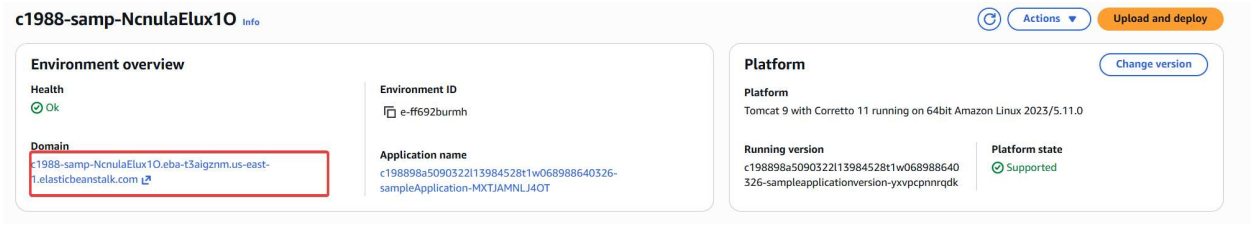
In the top-left search bar, search for “Elastic Beanstalk” and click its option to open up the AWS Elastic Beanstalk console



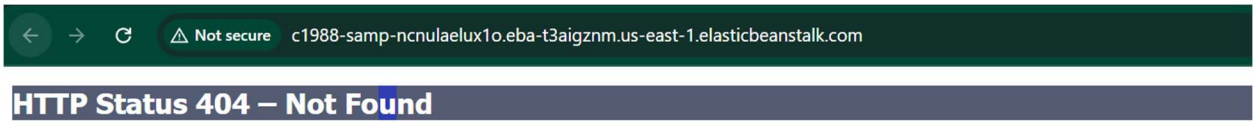
Click the environment name to open the environment menu



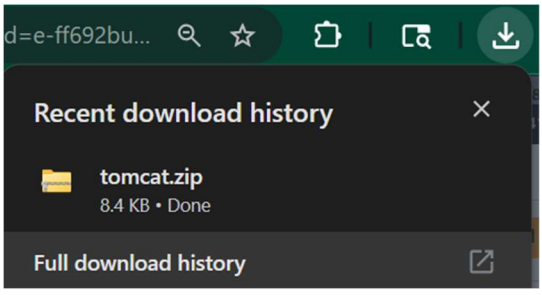
Open the Domain link



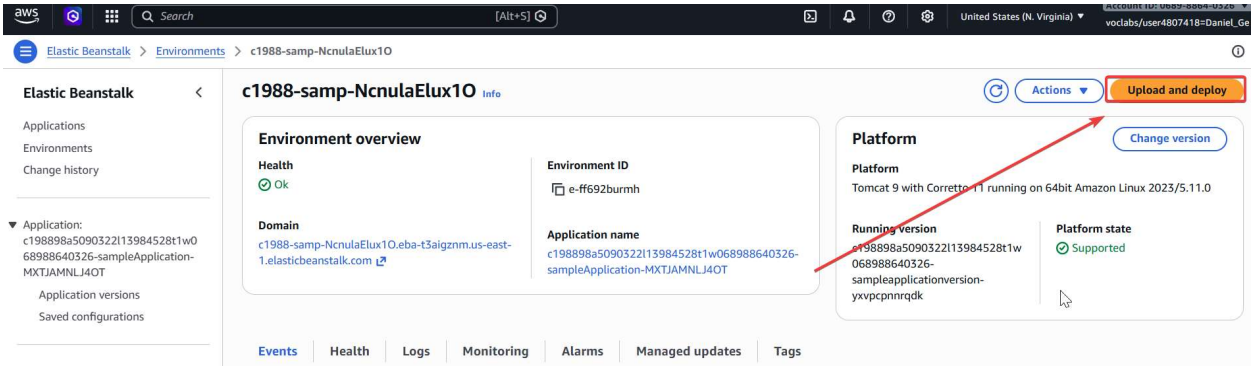
Since the server doesn't have any application yet, the webpage should display Error 404



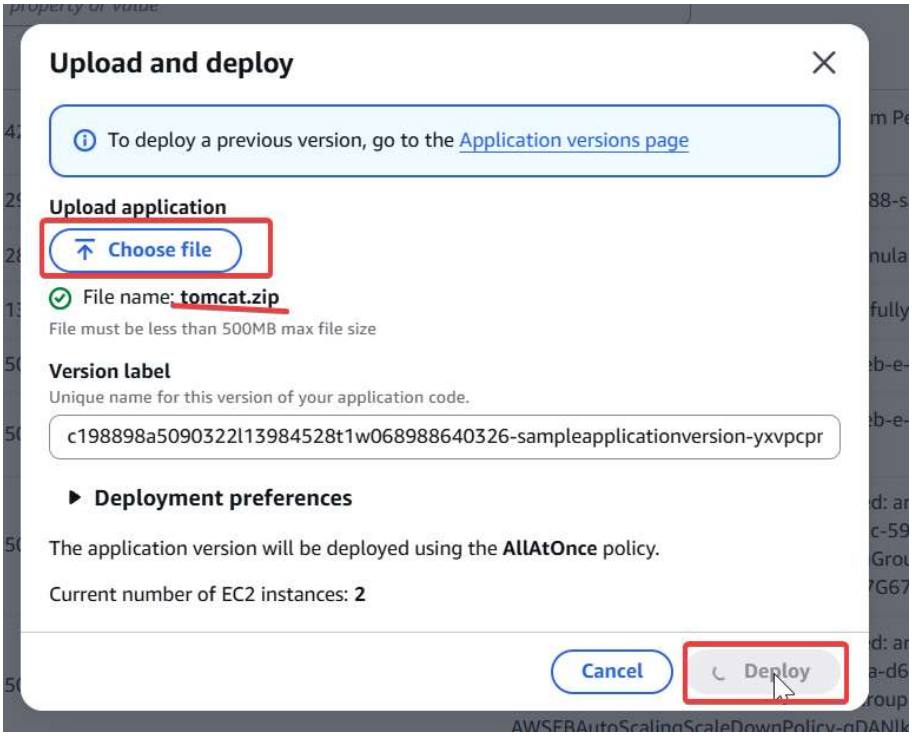
To practice deploying application on AWS Elastic Beanstalk, download a sample application from docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/tomcat.zip



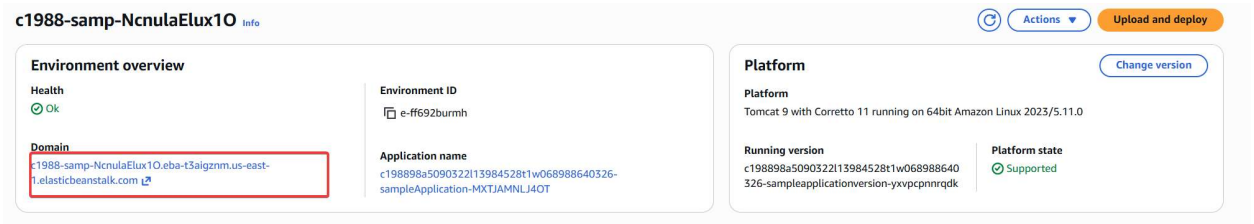
Click "Upload and Deploy"

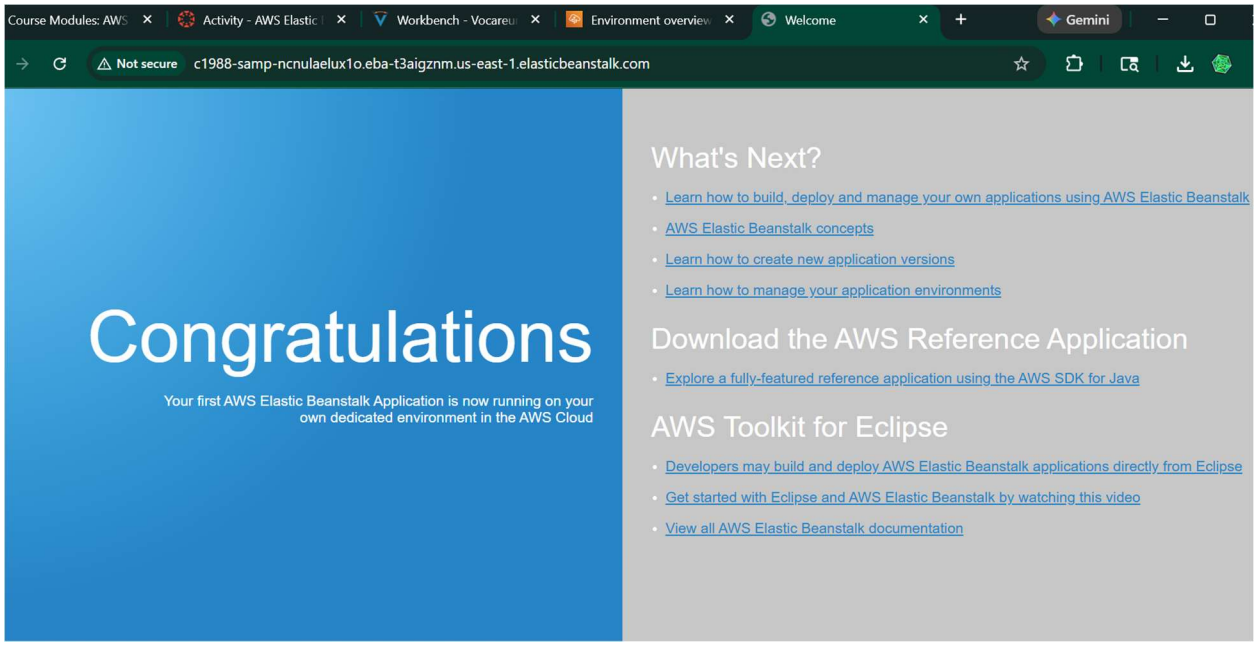


Click “Choose file” and select the recently downloaded tomcat.zip file, then click “Deploy”

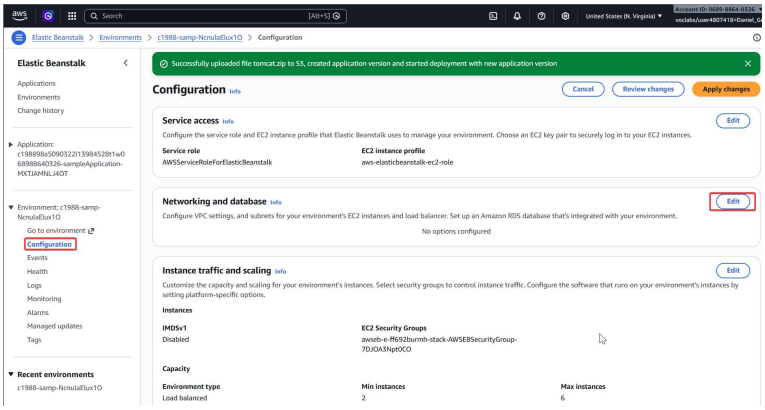


Reopen the domain link and verify that the website now shows a successful page

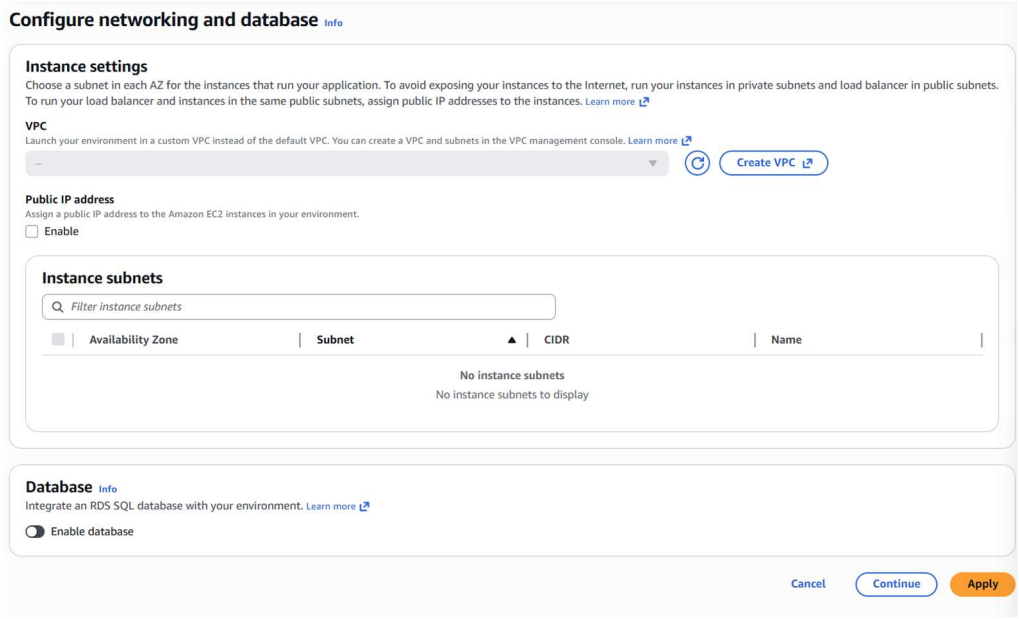




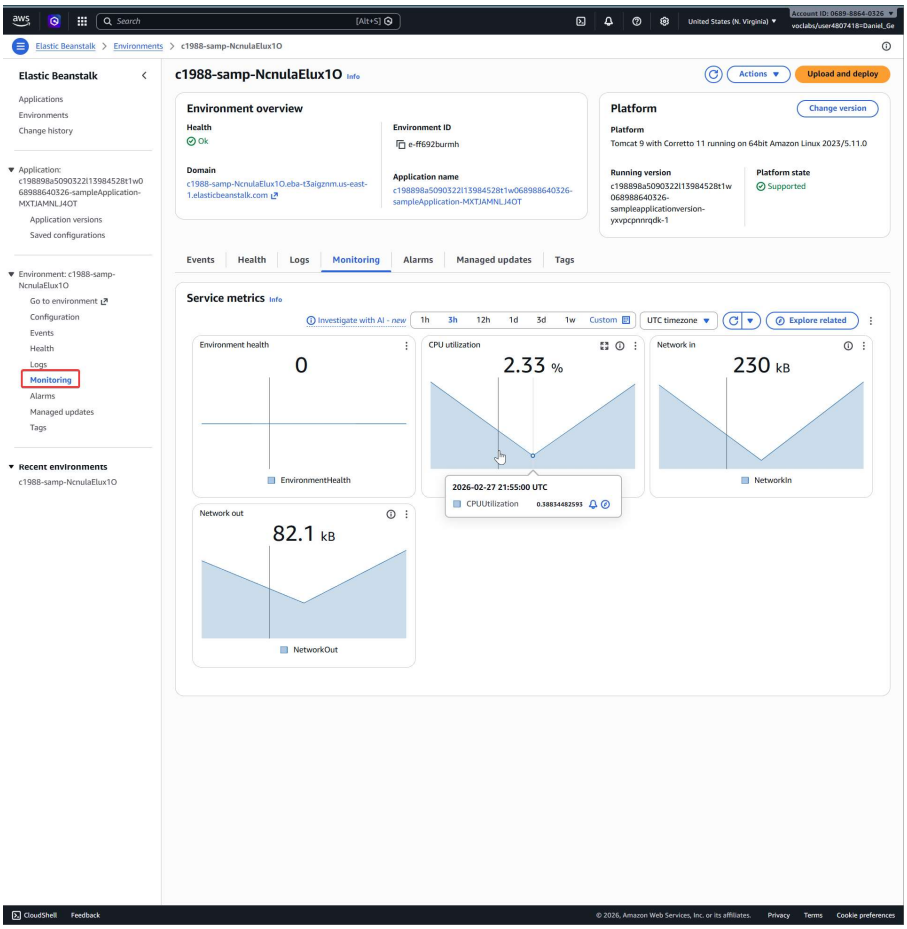
Go to the Configuration tag menu, then click “Edit”



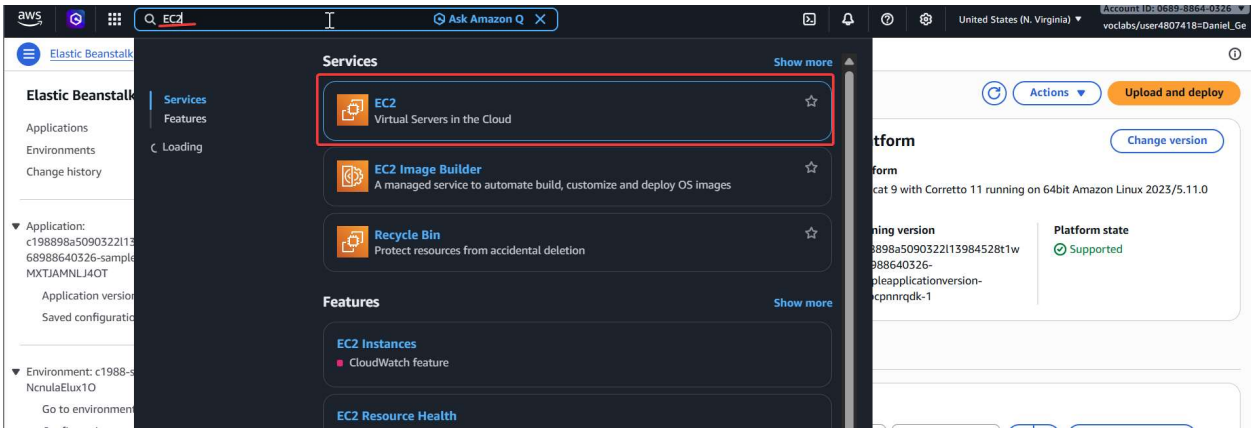
We may configure the Instance setting in this tab. Click “Cancel” to return to the menu.



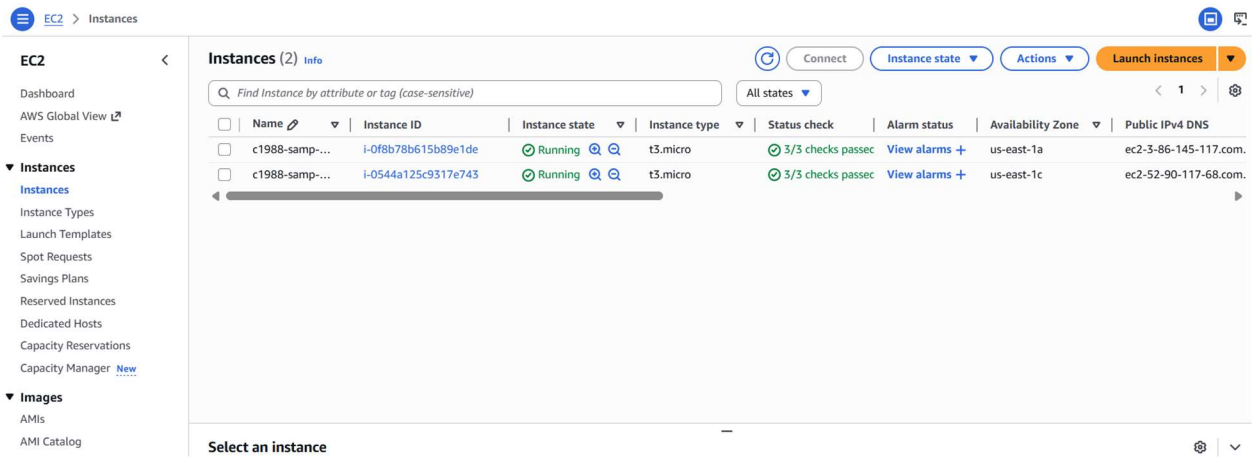
To view telemetry about the application, click the Monitoring tab



Using the search bar, navigate to the EC2 dashboard



Here, we may find more information about the instances launched



Problems

At the end of lab 2, I observed that I could not access the website we launched, receiving a request timeout error. This was due to the permissions being set up in such a way that requests from HTTPS were not permitted, hence an expected failure. In a later lab, this was addressed and I was able to access the desired website.

When attempting to complete multiple labs on the same day, I encountered an error when trying to open the second lab. This problem was due to AWS refusing to open multiple test accounts in rapid succession. The problem was fixed once I moved to a different browser application.

Conclusion

Through the experience of completing Labs 1, 2, and 3, including the encounter of occasional errors which needed troubleshooting, we gain a better understanding of how web services and cloud services operating on AWS are run. We are better prepared to apply the tools we learned about through the AWS instruction videos in AWS IAM, VPC, and EC2 to real-life applications.