AZUBI AFRICA

Generation GHANA

CAPSTONE PROJECT

Designing And Maintaining A Scalable Web Application in AWS

OBJECTIVES

01 Design and deploy a VPC

- ✓ Set up VPC and subnets.
- ✓ Attach Internet and NAT Gateways.
- ✓ Enforce security through Security Groups and Network ACLs.

02 Implement Auto Scaling and load balancing

- ✓ Configure an Auto Scaling Group with EC2 instances across multiple availability zones.
- ✓ Configure the ALB to span across multiple availability zones, improving fault tolerance..
- ✓ Define scaling policies to automatically add or remove instances based on CPU utilization or other custom metrics.

03 Configuring a DynamoDB Table in AWS

- ✓ Create and configure a DynamoDB table using the AWS Management Console.
- ✓ Enable interaction with DynamoDB using the Boto3 SDK in Python.
- ✓ Add and manage records (items) using attributes via the console.

BASIC TERMINOLOGIES

VPC

A logically isolated section of the public cloud.

Subnet

A segment of the VPC for isolating resources and controlling access within VPC.

Internet Gateway

Gateway that enables communication between your VPC and the internet.

NAT Gateway

Allows instances in a private subnet to connect to the internet without exposing them to inbound connection from the internet.

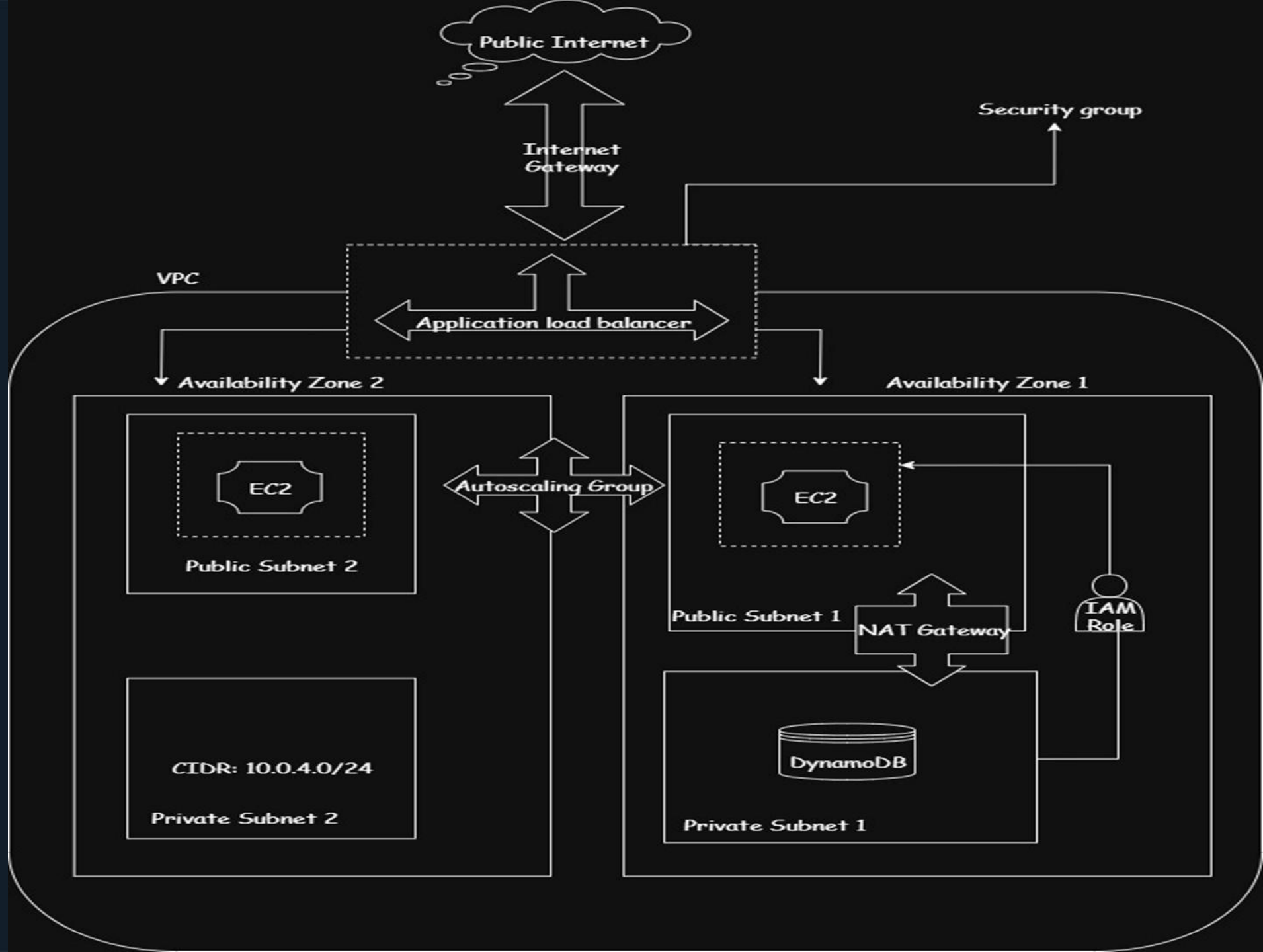
Security Group

a virtual firewall that controls inbound and outbound traffic for your EC2 instances.

NACL

a stateless firewall that acts as an optional layer of security for subnets within a VPC, controlling inbound and outbound traffic.

THE ARCHITECTURE



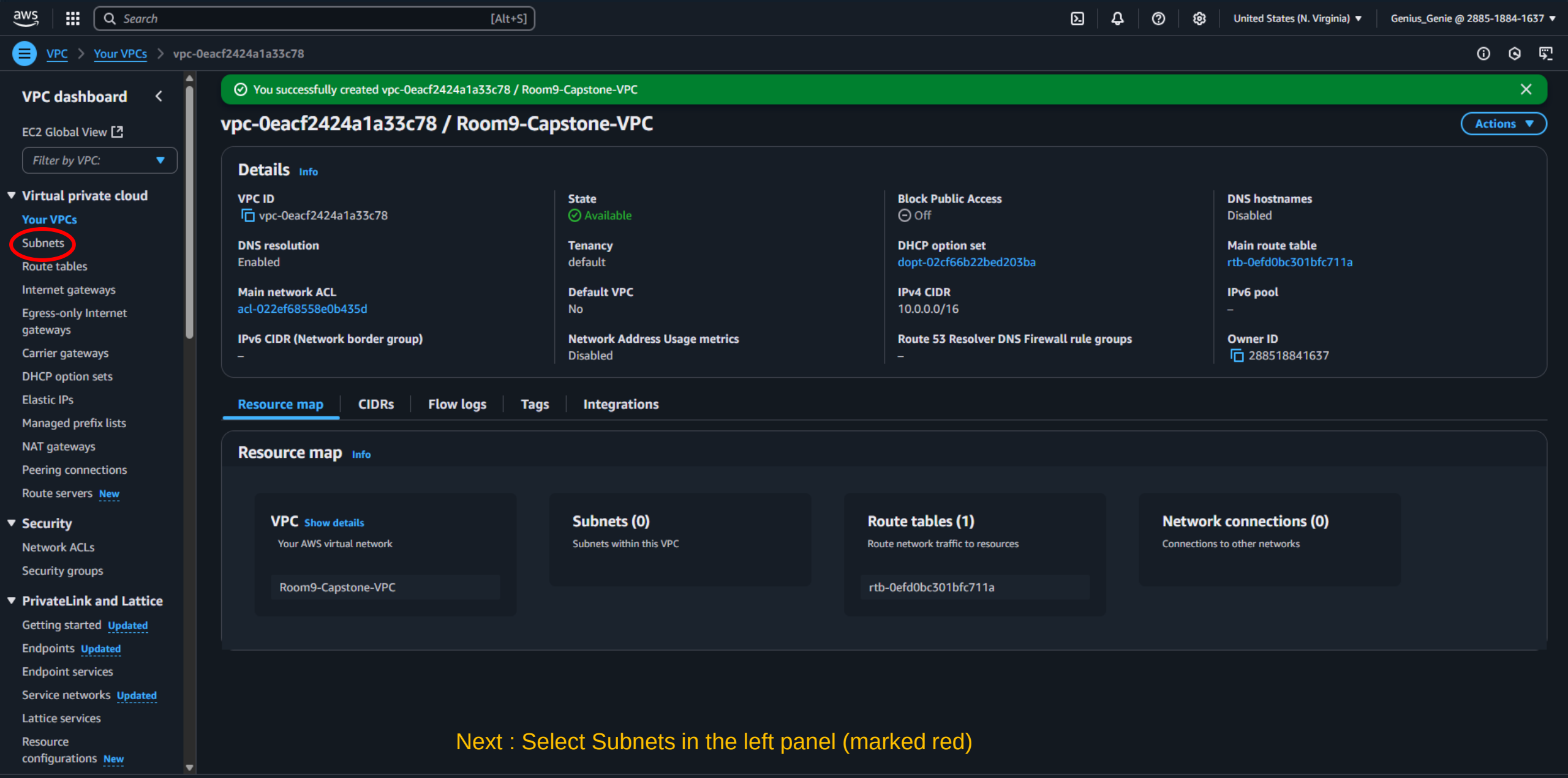
Step1: Create a VPC

- Log in to AWS Console
- Navigate to the **VPC Dashboard** by searching for "VPC" in the **AWS Services** search bar.
- Click **“VPCs”** on the left panel.
- Click the **“Create VPC”** button.
- Choose **“VPC only”** or “VPC with subnets” if you want AWS to auto-generate other components.
- Configure VPC Settings
- Click **“Create VPC”**

NOTES:

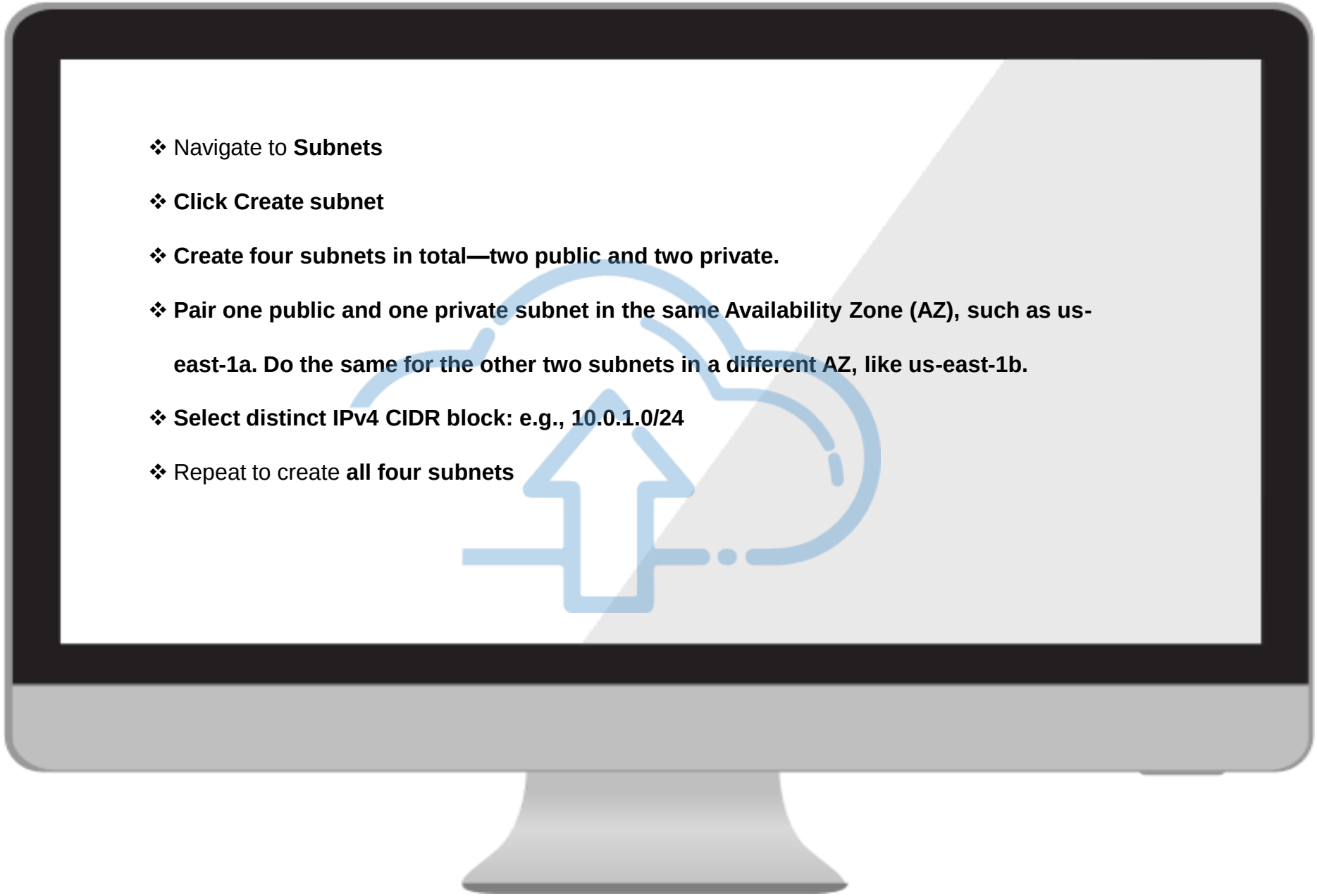
- **A well-chosen CIDR block, like 10.0.0.0/16, ensures sufficient IP addresses for all subnets and future scalability.**
- **Defines whether instances launched in your VPC will run on shared hardware (multitenancy) or a dedicated hardware**

VPC Created



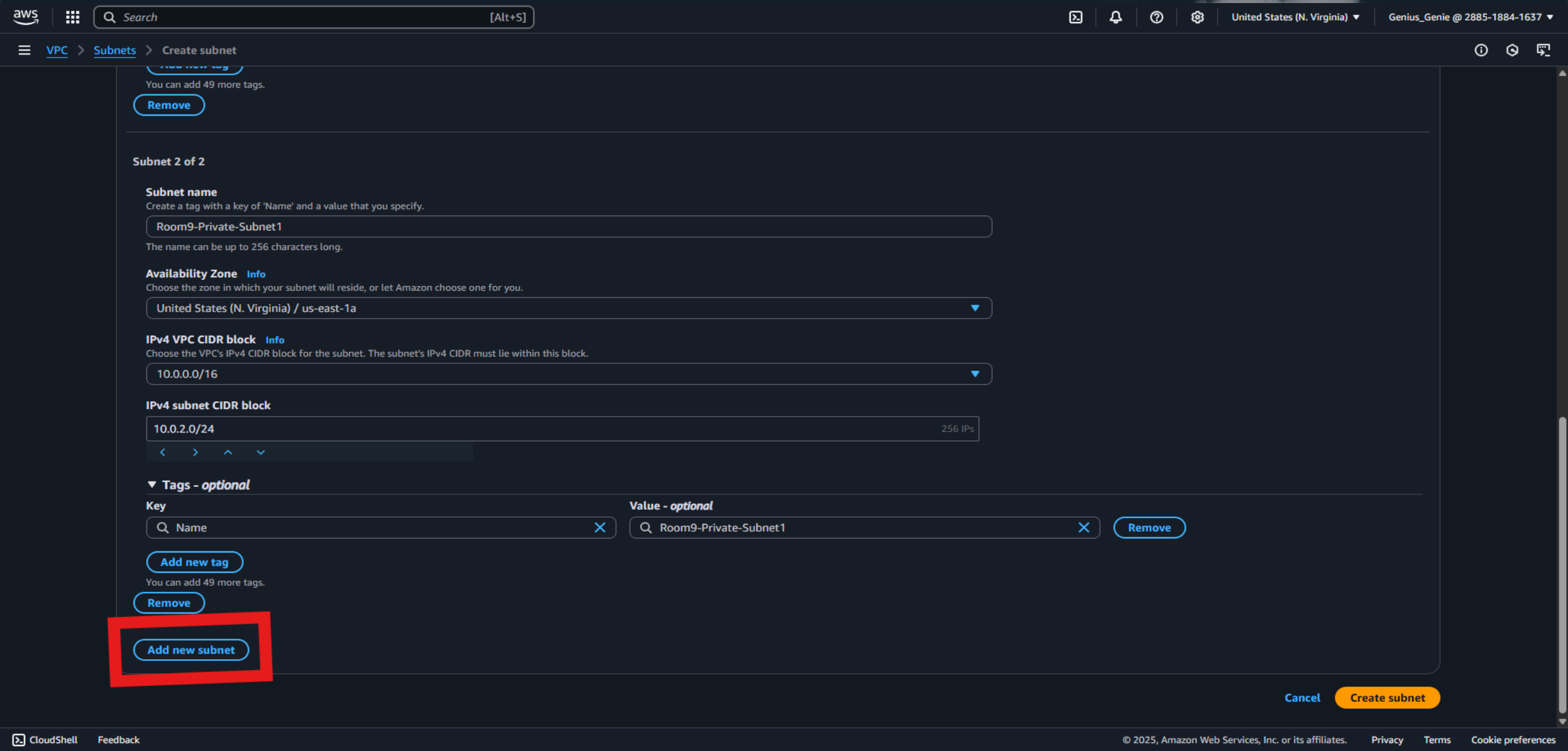
Next : Select Subnets in the left panel (marked red)

Step2: Create 4 subnets

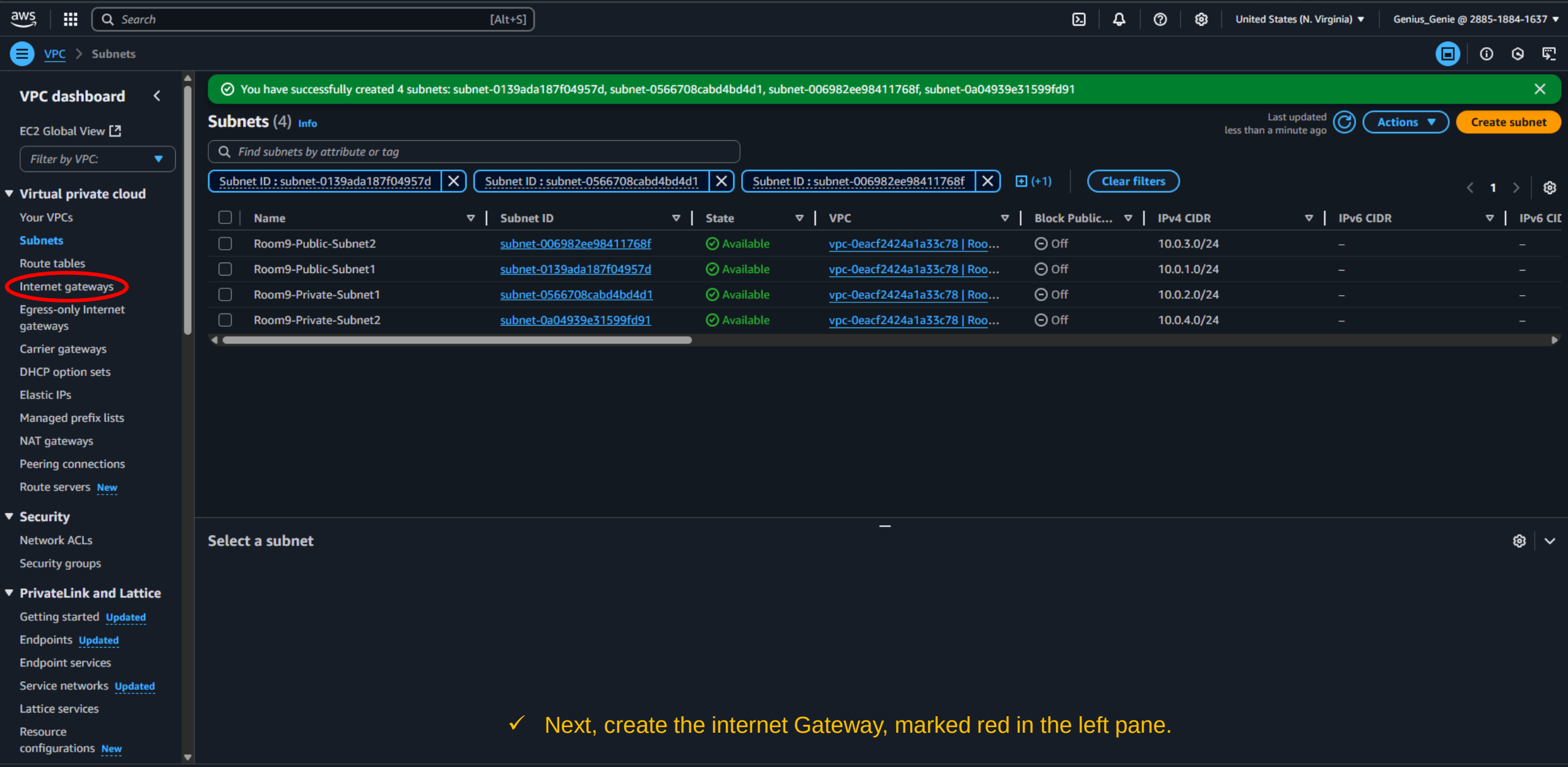


Private subnet in AZ 1

- ✓ To create all subnets at a goal, use the add new subnet button
- ✓ Subnets are assigned distinct CIDR blocks (10.0.1.0/24, 10.0.2.0/24…) to segment the VPC's IP range and avoid overlap.

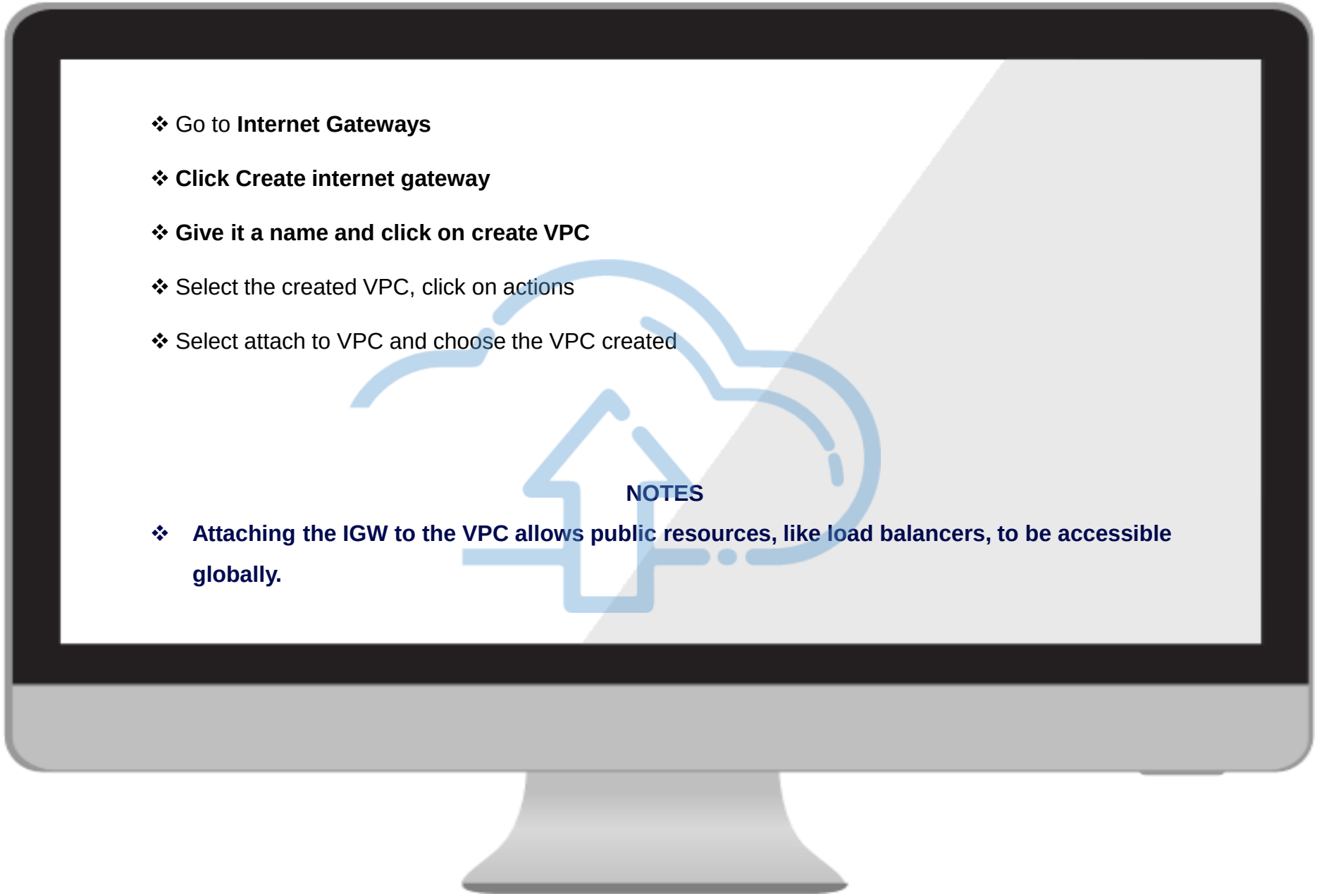


Subnets Created



✓ Next, create the internet Gateway, marked red in the left pane.

Step3: Create Internet Gateway



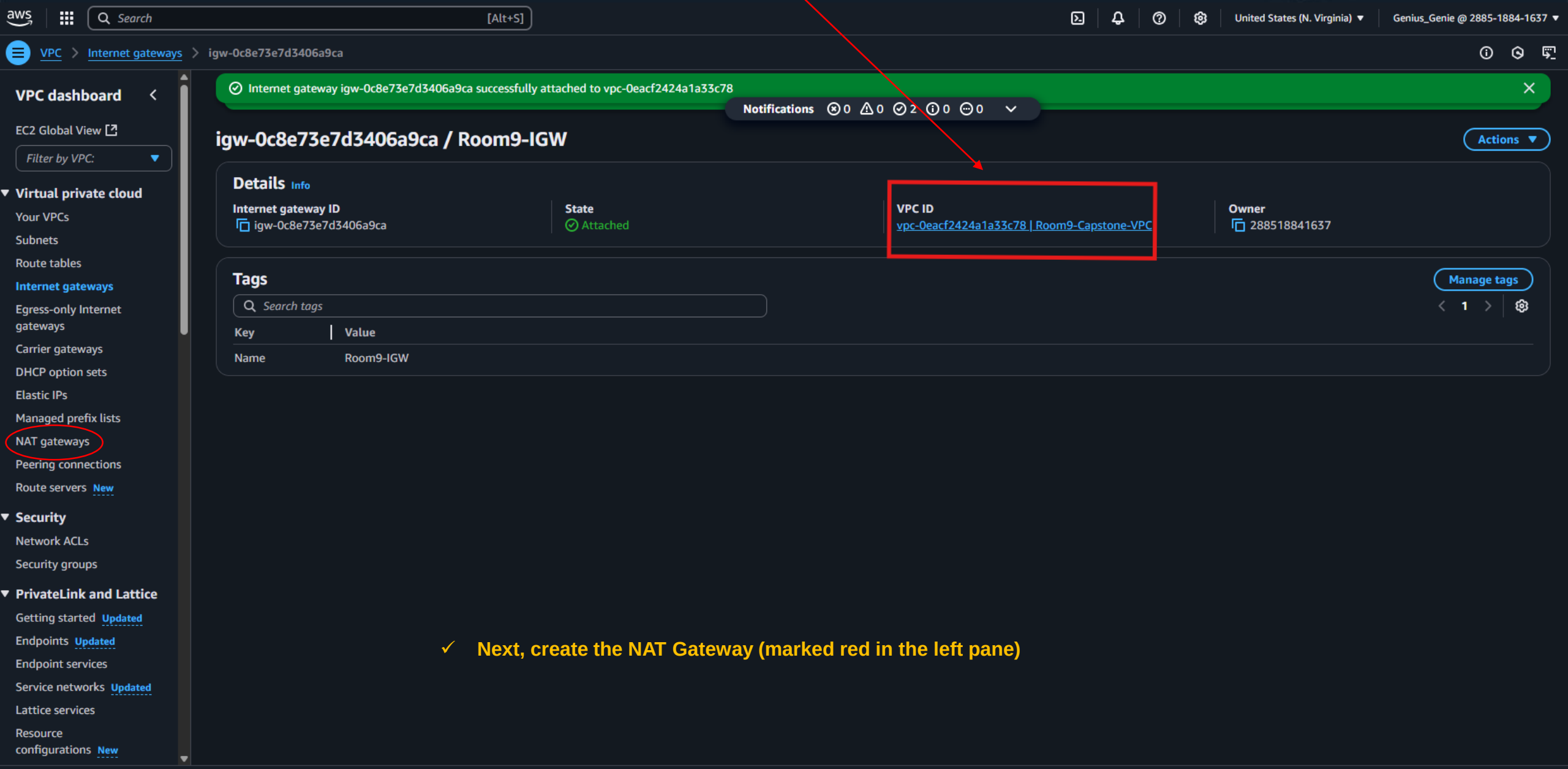
- Go to **Internet Gateways**
- **Click Create internet gateway**
- **Give it a name and click on create VPC**
- Select the created VPC, click on actions
- Select attach to VPC and choose the VPC created

NOTES

- **Attaching the IGW to the VPC allows public resources, like load balancers, to be accessible globally.**

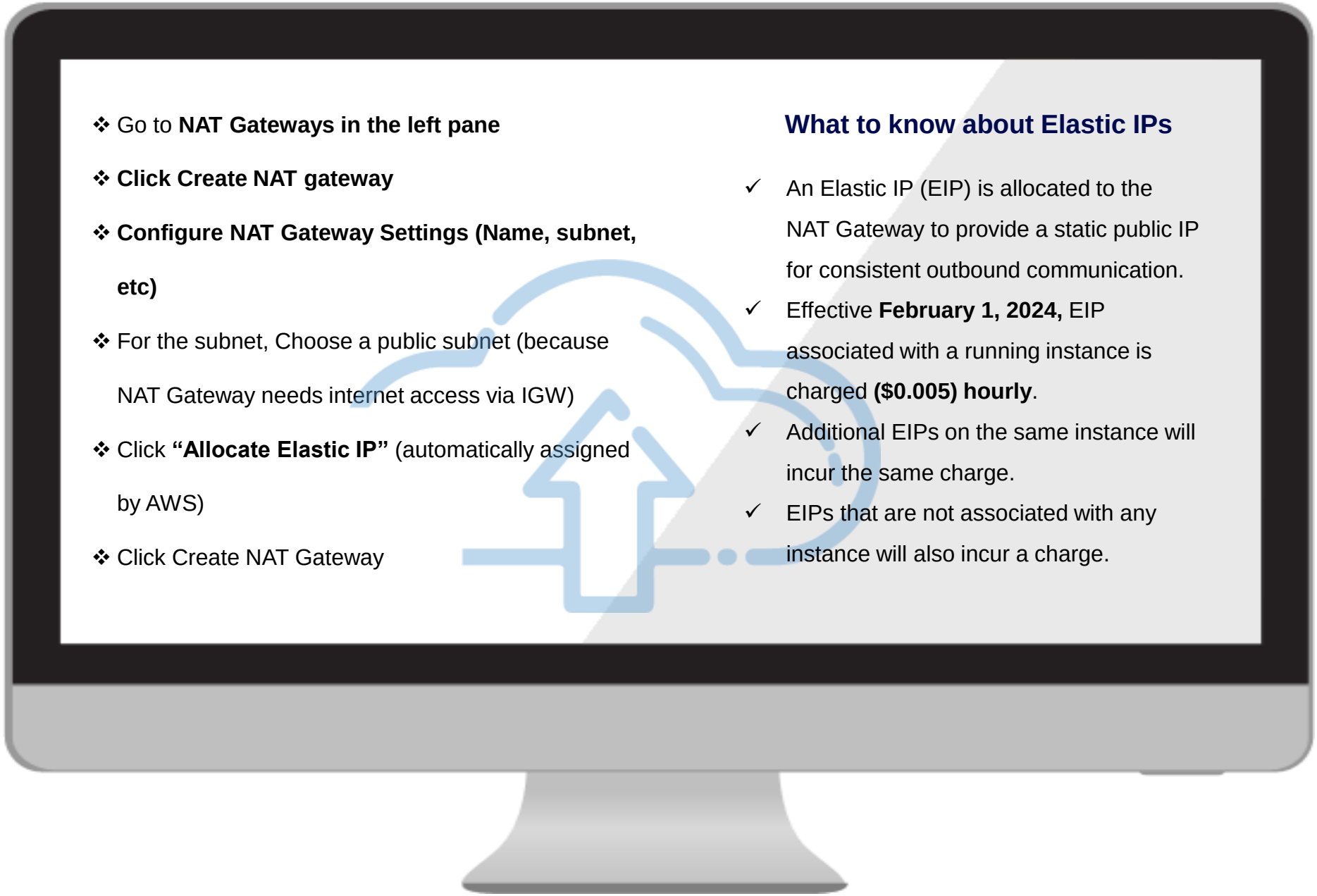
Internet Gateway Created

✓ Notice that a VPC id has been added in the Details section of your IGW



✓ Next, create the NAT Gateway (marked red in the left pane)

Step4: Create NAT Gateway

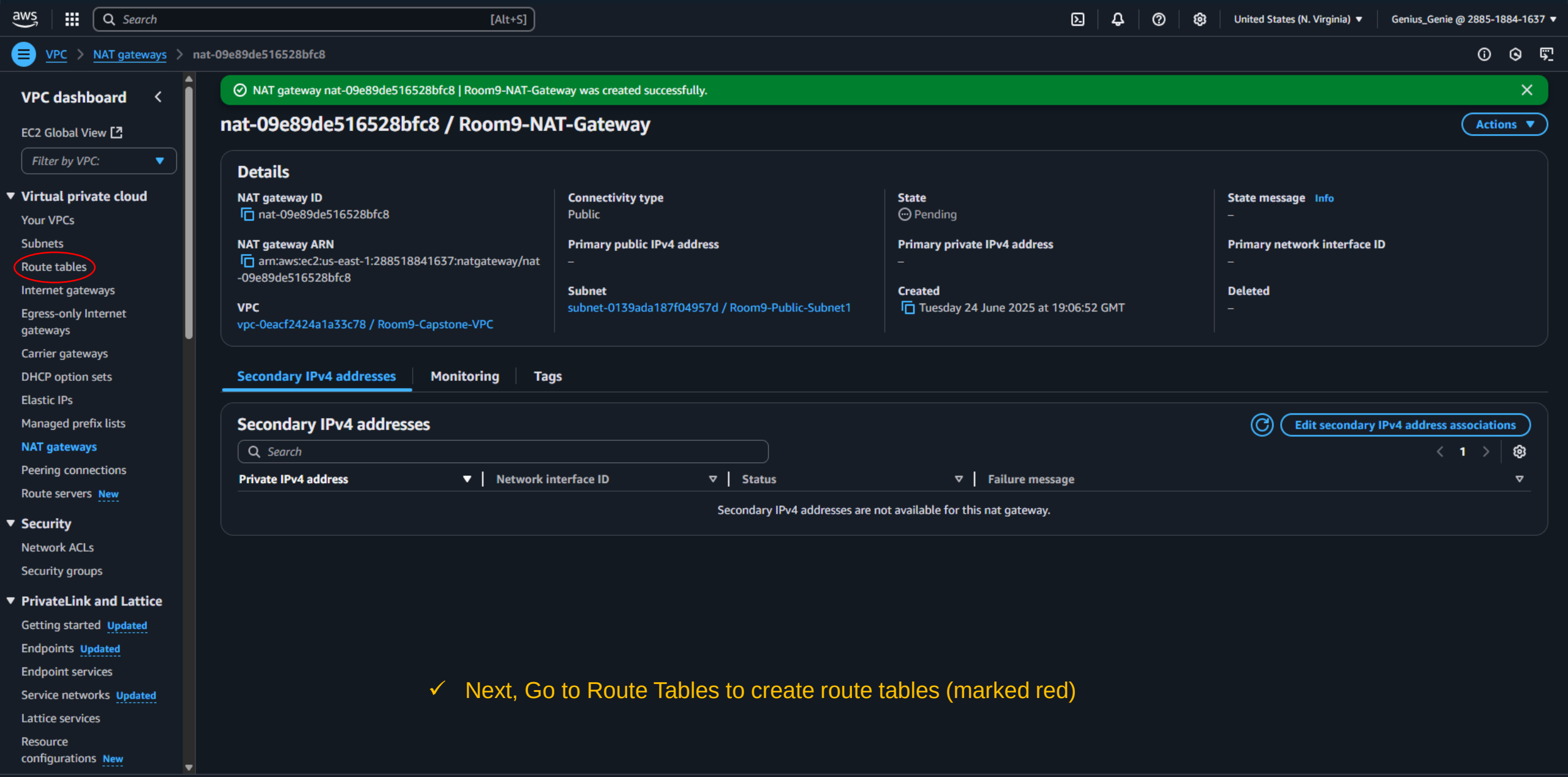


- Go to **NAT Gateways in the left pane**
- **Click Create NAT gateway**
- **Configure NAT Gateway Settings (Name, subnet, etc)**
- For the subnet, Choose a public subnet (because NAT Gateway needs internet access via IGW)
- Click **“Allocate Elastic IP”** (automatically assigned by AWS)
- Click Create NAT Gateway

What to know about Elastic IPs

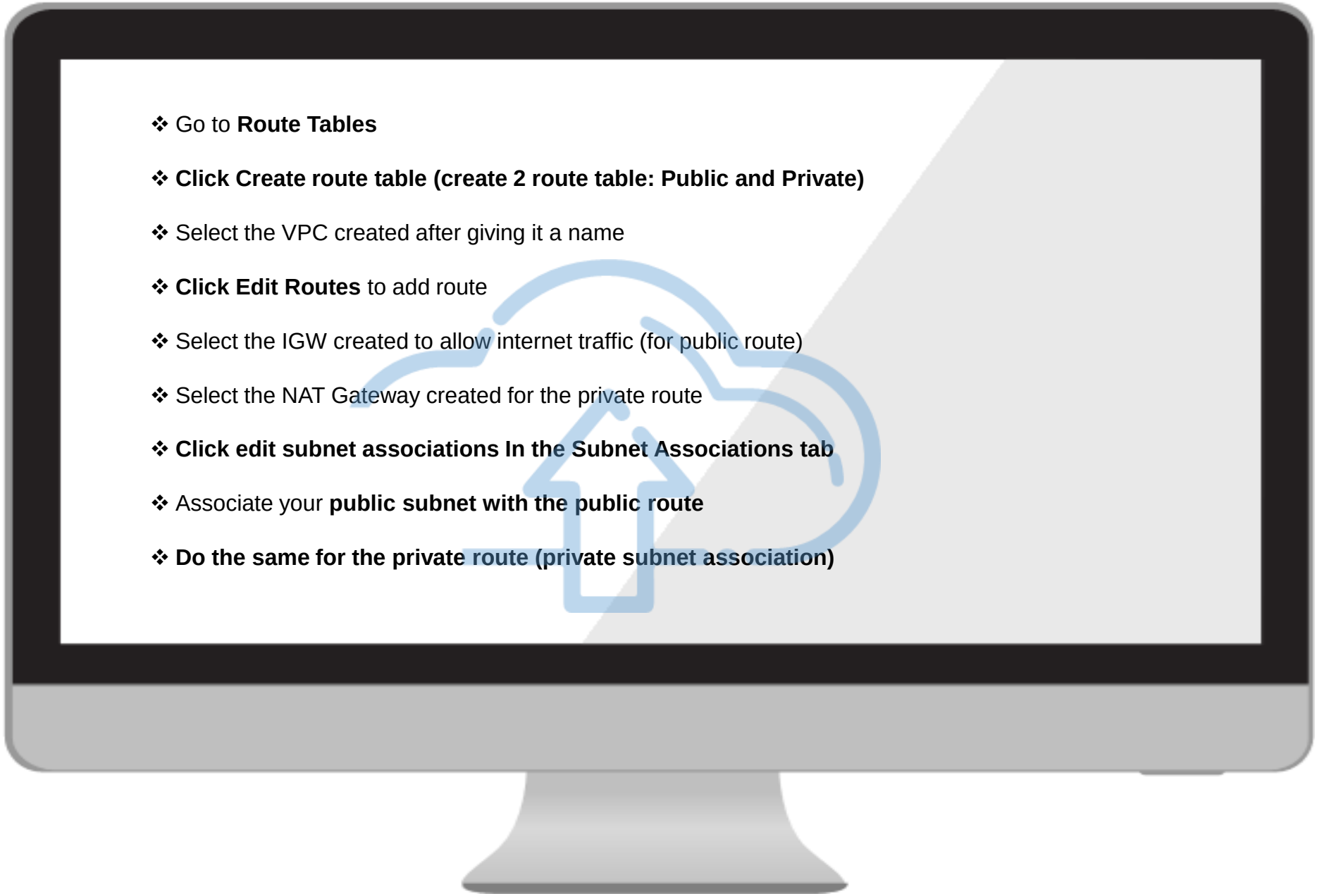
- ✓ An Elastic IP (EIP) is allocated to the NAT Gateway to provide a static public IP for consistent outbound communication.
- ✓ Effective **February 1, 2024,** EIP associated with a running instance is charged **($0.005) hourly**.
- ✓ Additional EIPs on the same instance will incur the same charge.
- ✓ EIPs that are not associated with any instance will also incur a charge.

NAT Gateway Created



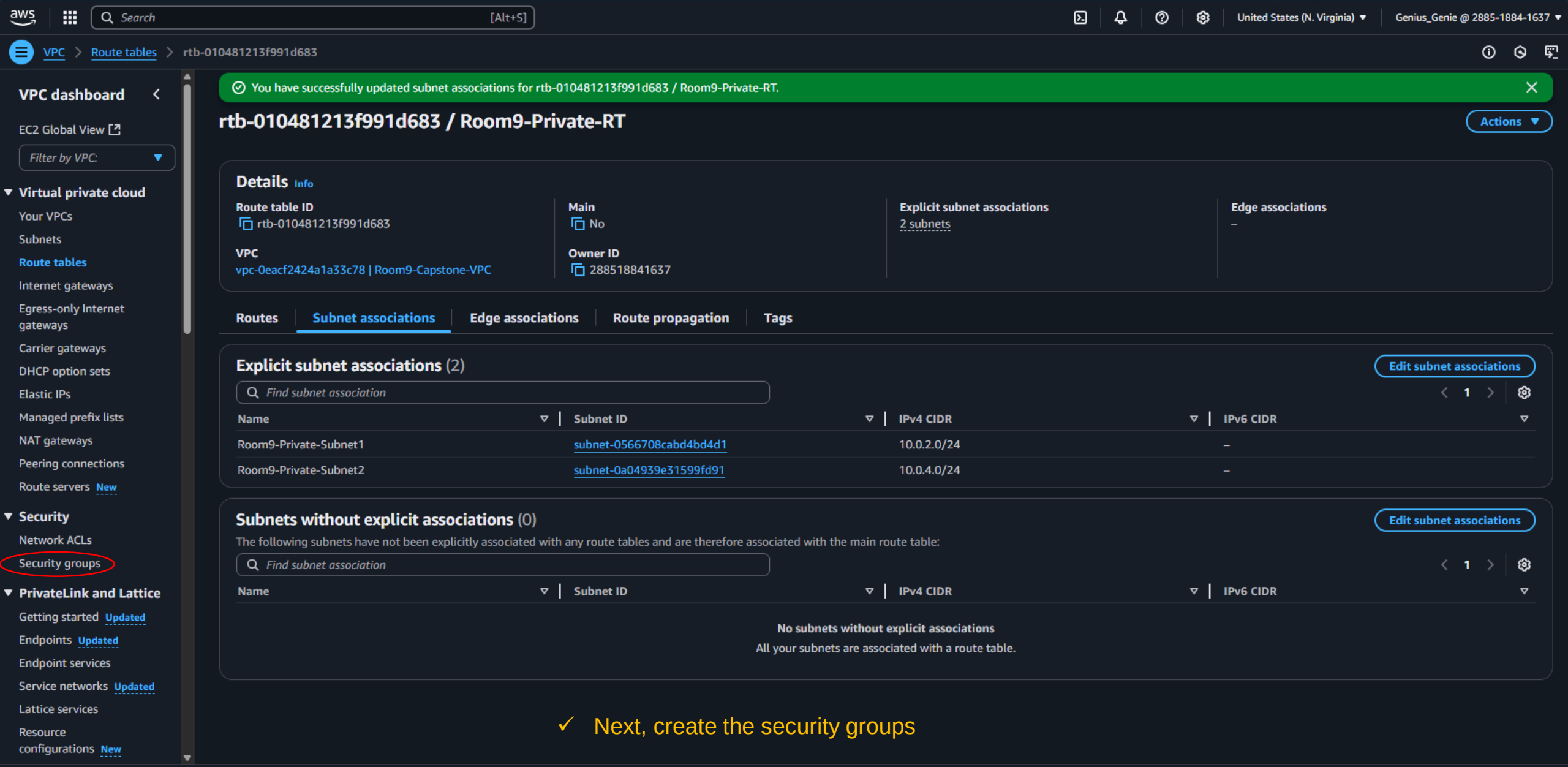
✓ Next, Go to Route Tables to create route tables (marked red)

Step5: Create Route Tables



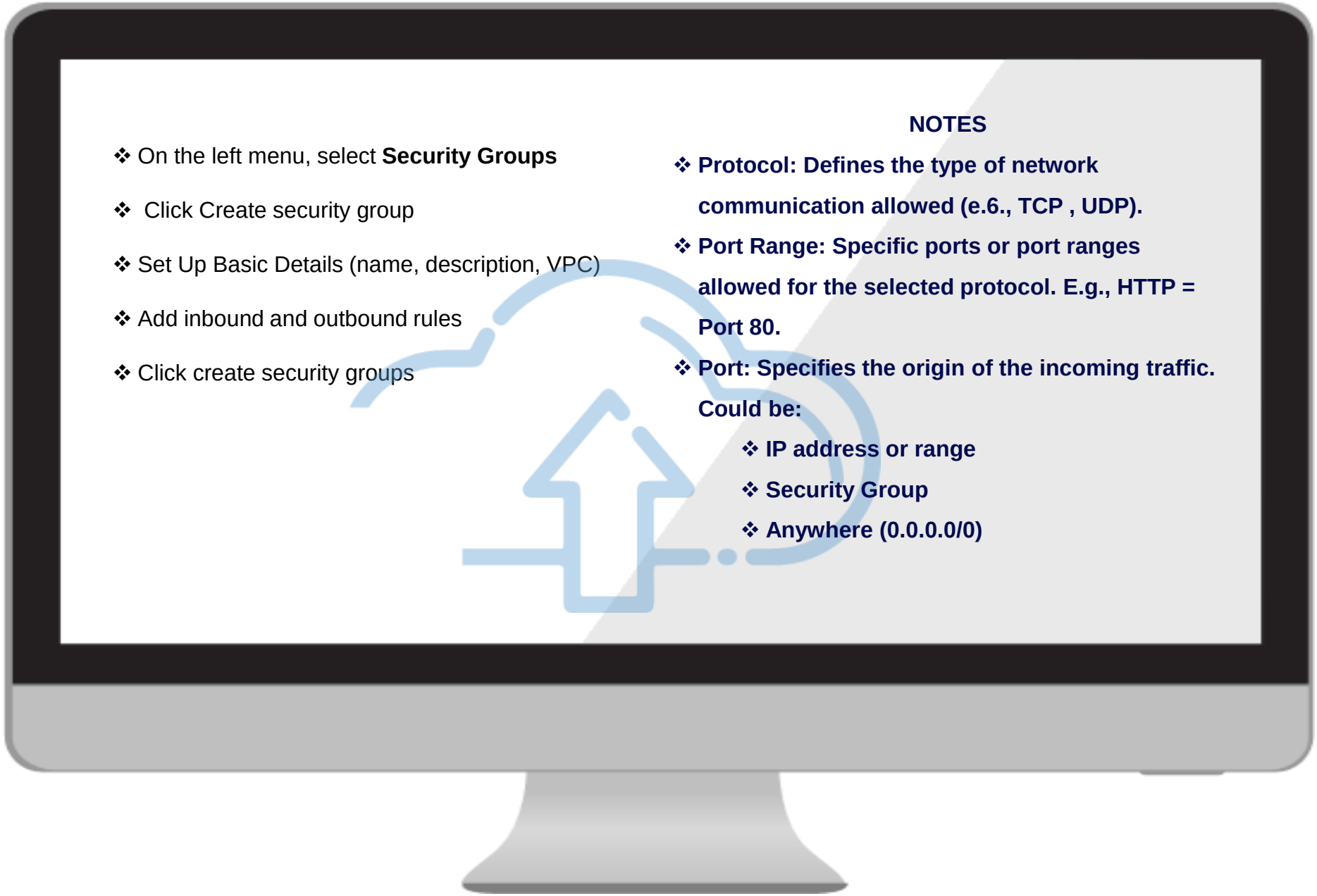
- Go to **Route Tables**
- **Click Create route table (create 2 route table: Public and Private)**
- Select the VPC created after giving it a name
- **Click Edit Routes** to add route
- Select the IGW created to allow internet traffic (for public route)
- Select the NAT Gateway created for the private route
- **Click edit subnet associations In the Subnet Associations tab**
- Associate your **public subnet with the public route**
- **Do the same for the private route (private subnet association)**

Route Table Created (Private Route Table)



✓ Next, create the security groups

Step6: Create and Configure Security Groups



- On the left menu, select **Security Groups**
- Click Create security group
- Set Up Basic Details (name, description, VPC)
- Add inbound and outbound rules
- Click create security groups

NOTES

- **Protocol: Defines the type of network communication allowed (e.6., TCP , UDP).**
- **Port Range: Specific ports or port ranges allowed for the selected protocol. E.g., HTTP = Port 80.**
- **Port: Specifies the origin of the incoming traffic. Could be:**
  - **IP address or range**
  - **Security Group**
  - **Anywhere (0.0.0.0/0)**

Step 6: Inbound and Outbound Rules

SSH	TCP	22	Anywhe...	0.0.0.0/0	Allow traffic from anywhere Allow traffic from anywhere	Delete
HTTP	TCP	80	Anywhe...	0.0.0.0/0	Allow traffic from anywhere Allow traffic from anywhere	Delete
HTTPS	TCP	443	Anywhe...	0.0.0.0/0	Allow traffic from anywhere Allow traffic from anywhere	Delete

Add rule

⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Outbound rules Info

Type Info	Protocol Info	Port range Info	Destination Info		Description - optional Info	
All traffic	All	All	Custom	0.0.0.0/0		Delete

Add rule

⚠ Rules with destination of 0.0.0.0/0 or ::/0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses.

Tags - *optional*

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

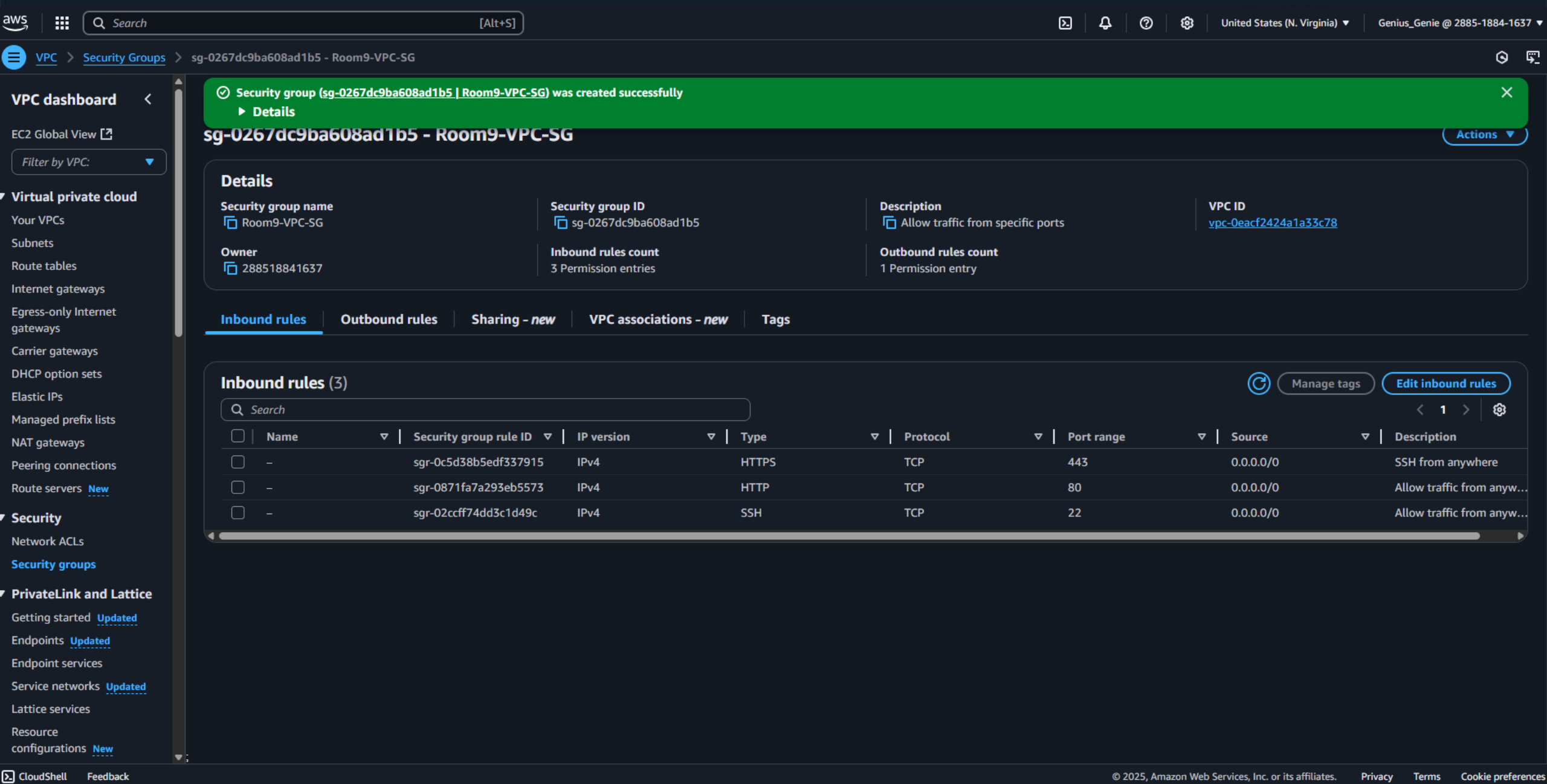
Add new tag

You can add up to 50 more tags

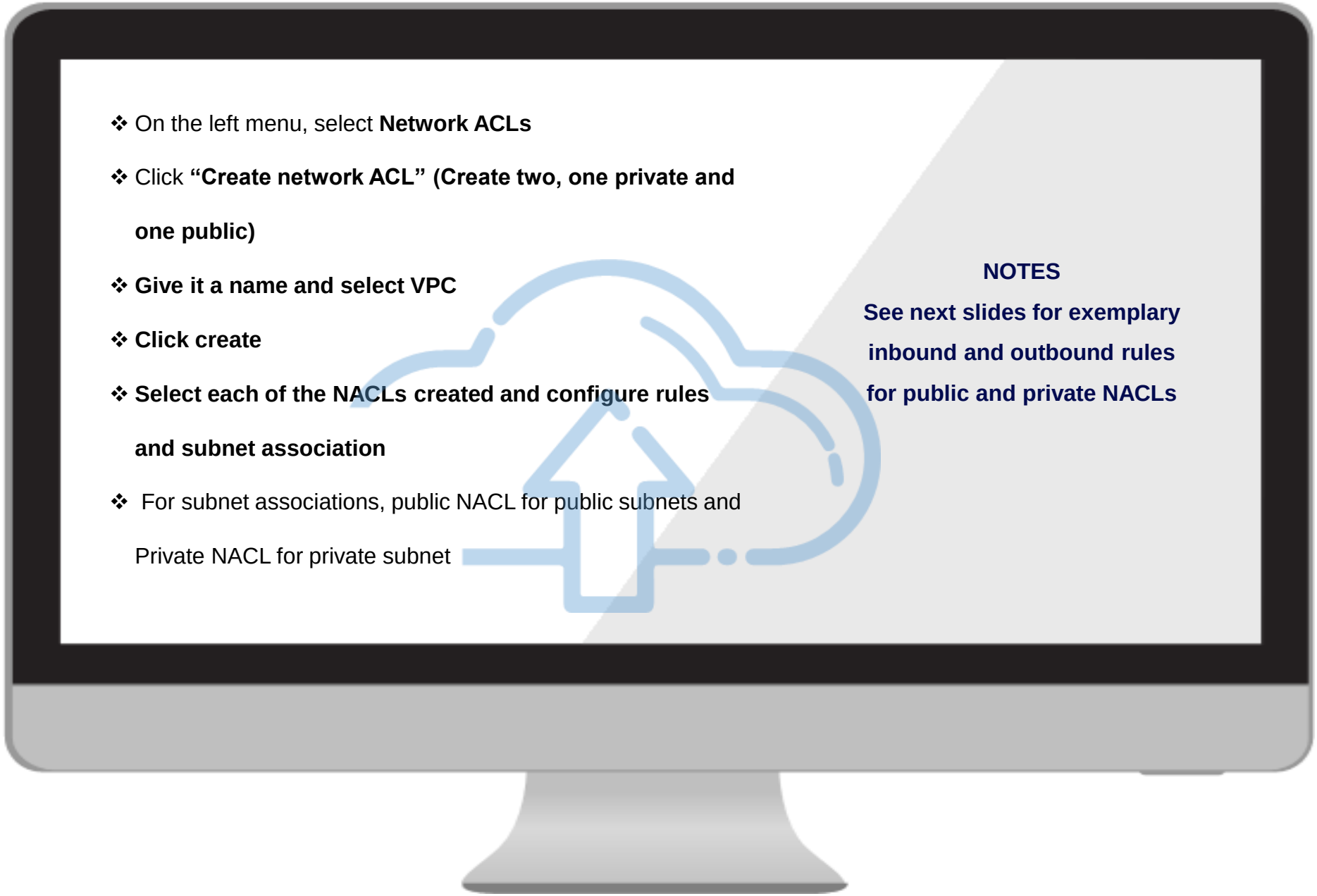
Cancel | Create security group

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Security Group Created



Step7: Create and configure NACLs



- On the left menu, select **Network ACLs**
- Click **“Create network ACL” (Create two, one private and one public)**
- **Give it a name and select VPC**
- **Click create**
- **Select each of the NACLs created and configure rules and subnet association**
- For subnet associations, public NACL for public subnets and Private NACL for private subnet

NOTES

See next slides for exemplary inbound and outbound rules for public and private NACLs

Step 12: Inbound Rules for Public NACL

Edit outbound rules Info

Outbound rules control the outgoing traffic that's allowed to leave the VPC.

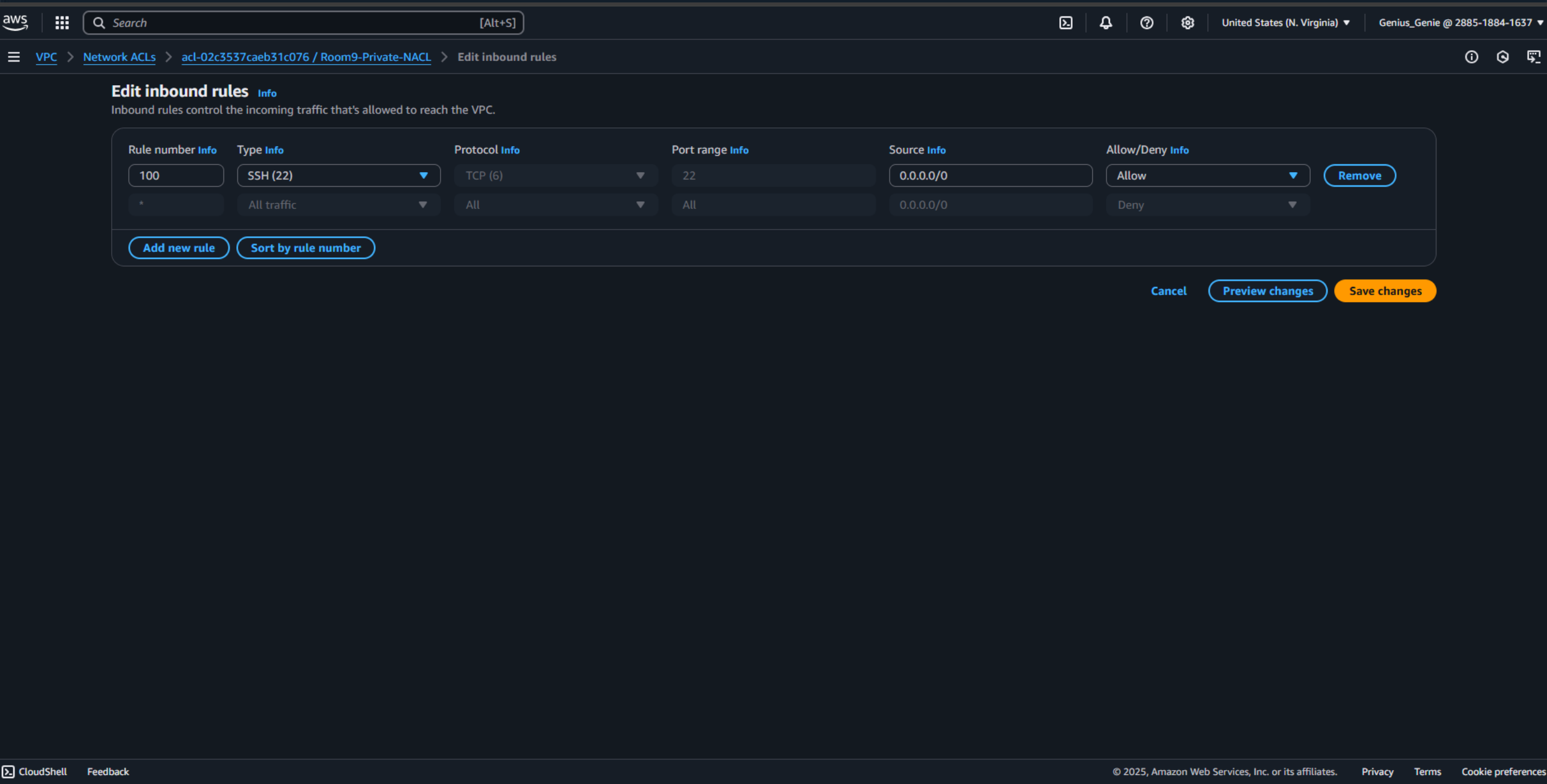
Rule number Info	Type Info	Protocol Info	Port range Info	Destination Info	Allow/Deny Info	
100	HTTP (80)	TCP (6)	80	0.0.0.0/0	Allow	Remove
110	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow	Remove
120	SSH (22)	TCP (6)	22	0.0.0.0/0	Allow	Remove
*	All traffic	All	All	0.0.0.0/0	Deny	

Add new rule | Sort by rule number

Cancel | Preview changes | Save changes

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 12: Configure Inbound Rules for Private NACL



Step 12: Configure Outbound Rules for Private NACL

Edit outbound rules Info

Outbound rules control the outgoing traffic that's allowed to leave the VPC.

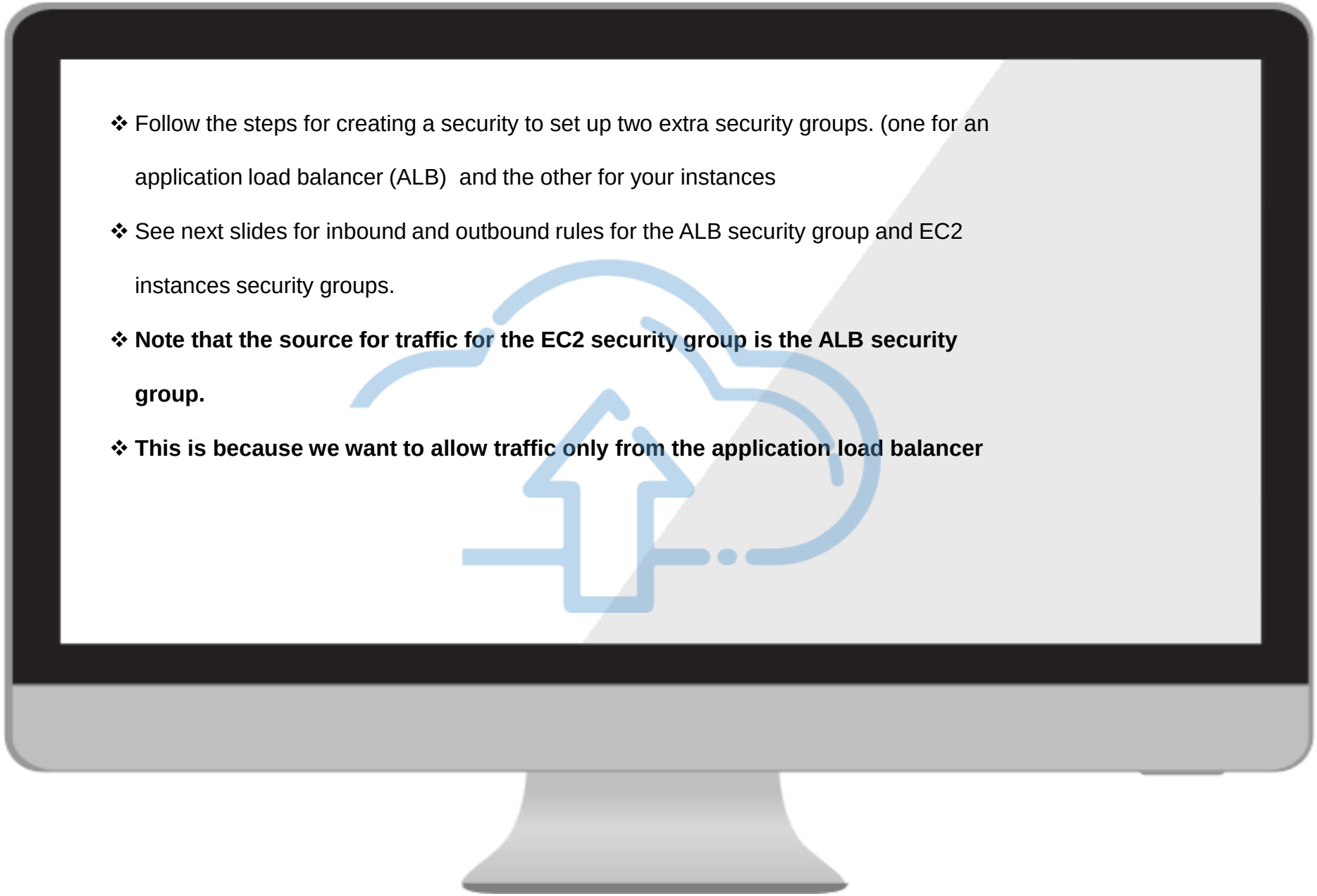
Rule number Info	Type Info	Protocol Info	Port range Info	Destination Info	Allow/Deny Info	
100	HTTP (80)	TCP (6)	80	0.0.0.0/0	Allow	Remove
110	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow	Remove
120	SSH (22)	TCP (6)	22	0.0.0.0/0	Allow	Remove
*	All traffic	All	All	0.0.0.0/0	Deny	

Add new rule | Sort by rule number

Cancel | Preview changes | Save changes

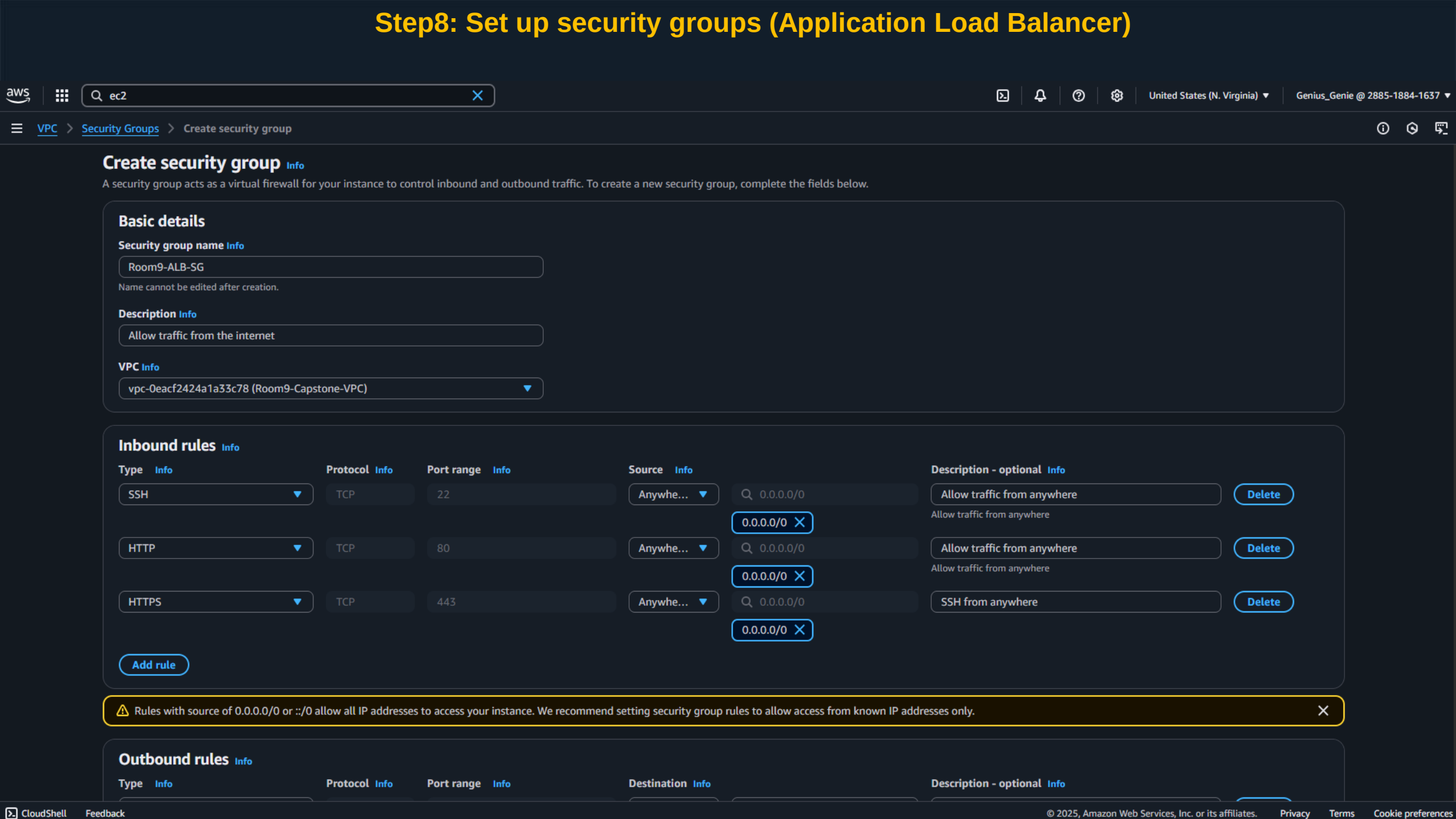
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step8: Set up extra security groups

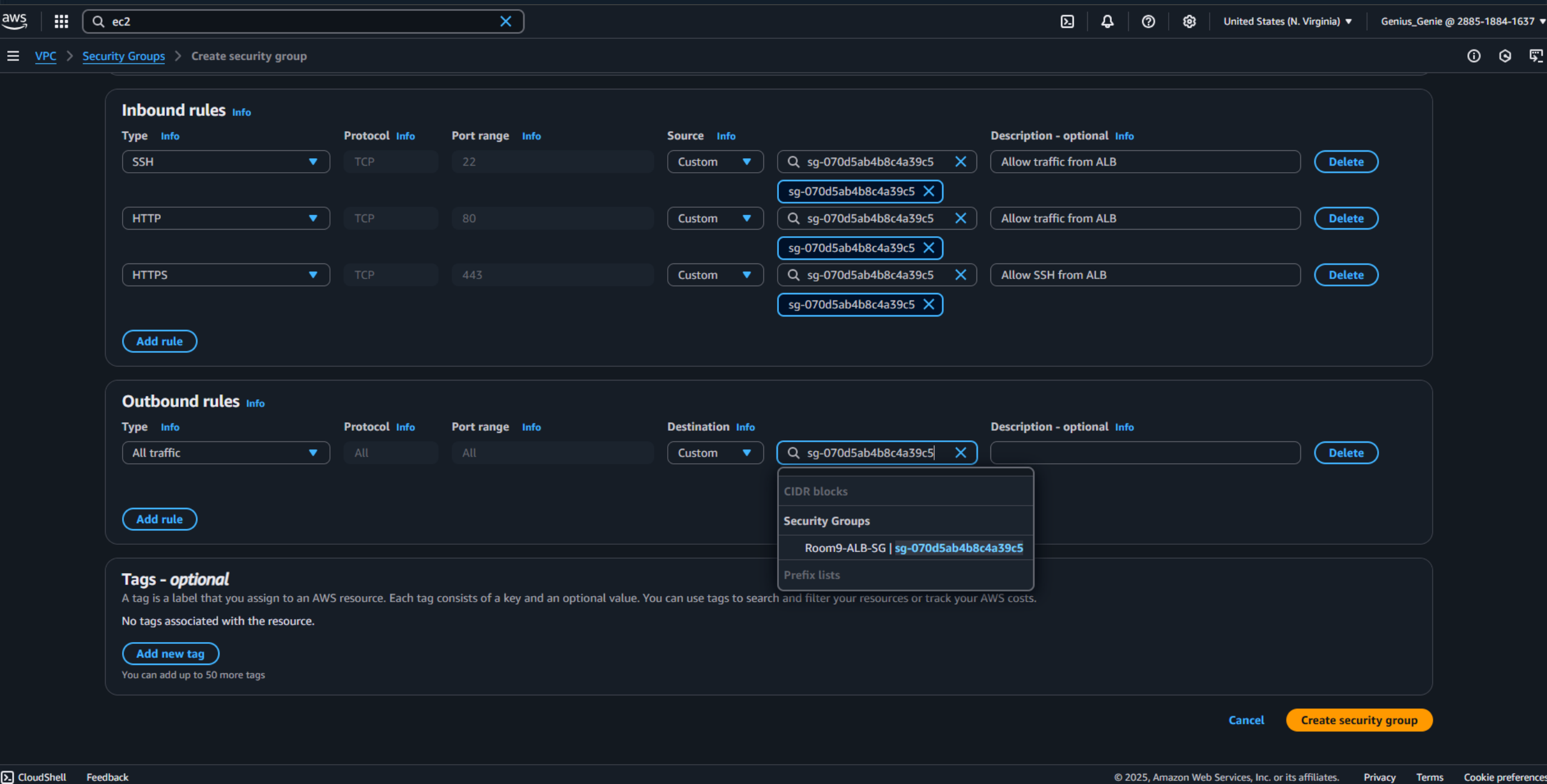


- Follow the steps for creating a security to set up two extra security groups. (one for an application load balancer (ALB) and the other for your instances
- See next slides for inbound and outbound rules for the ALB security group and EC2 instances security groups.
- **Note that the source for traffic for the EC2 security group is the ALB security group.**
- **This is because we want to allow traffic only from the application load balancer**

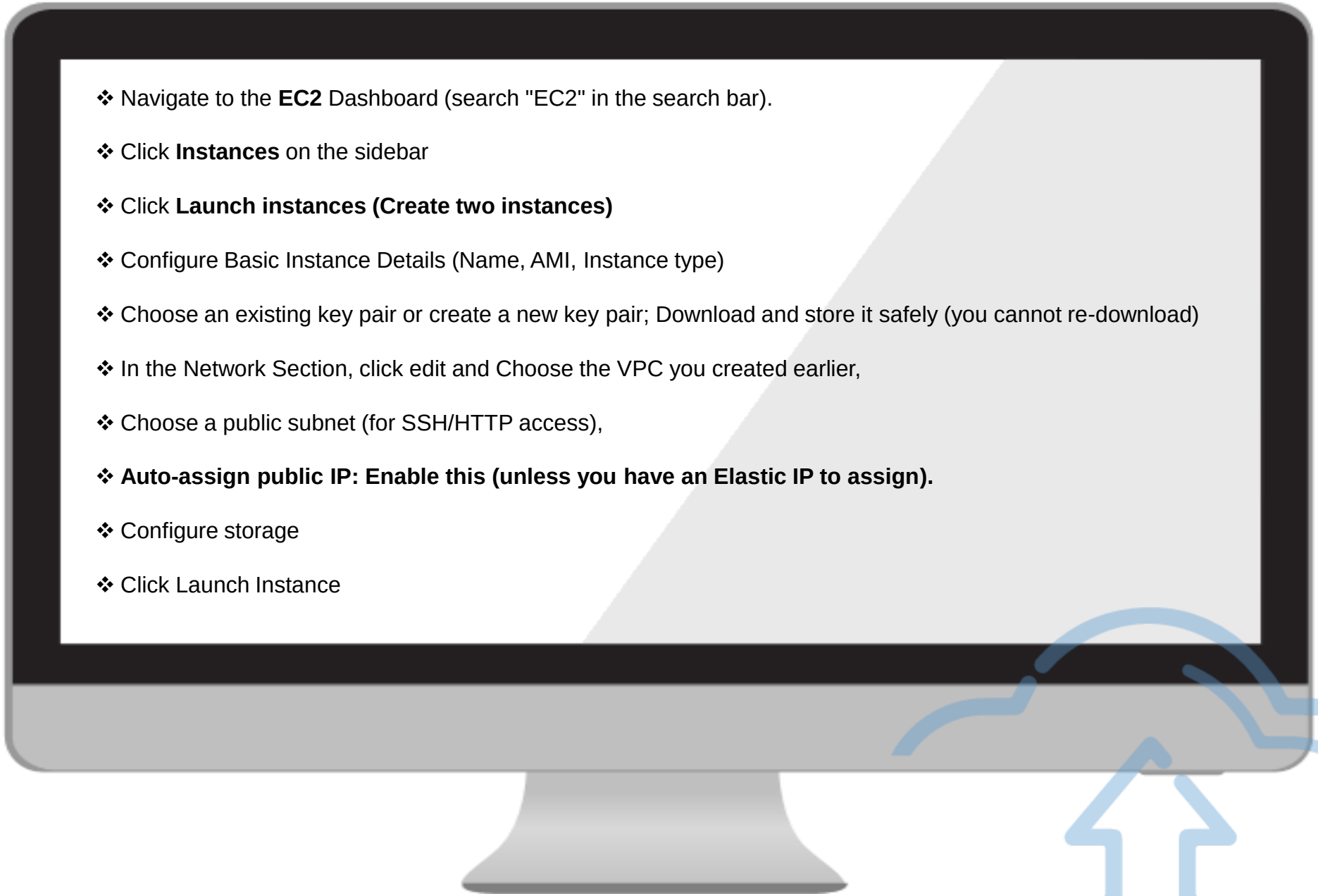
Step8: Set up security groups (Application Load Balancer)



Step8: Set up security groups (EC2 Instances)



Step9: Launch an EC2 Instance



- ❖ Navigate to the **EC2** Dashboard (search "EC2" in the search bar).
- ❖ Click **Instances** on the sidebar
- ❖ Click **Launch instances (Create two instances)**
- ❖ Configure Basic Instance Details (Name, AMI, Instance type)
- ❖ Choose an existing key pair or create a new key pair; Download and store it safely (you cannot re-download)
- ❖ In the Network Section, click edit and Choose the VPC you created earlier,
- ❖ Choose a public subnet (for SSH/HTTP access),
- ❖ **Auto-assign public IP: Enable this (unless you have an Elastic IP to assign).**
- ❖ Configure storage
- ❖ Click Launch Instance

Step9: Create and launch an instance

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name

Room9 Web Server

Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Linux | Debian

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 kernel-6.1 AMI — Free tier eligible

ami-09e6f87a47903347c (64-bit (x86), uefi-preferred) / ami-0db36bcbb6bf68b98 (64-bit (Arm), uefi)

Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.7.20250609.0 x86_64 HVM kernel-6.1

Architecture | Boot mode | AMI ID | Publish Date | Username

Summary

Number of instances Info

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.7.2...read more

ami-09e6f87a47903347c

Virtual server type (instance type)

t2.micro

Firewall (security group)

Room9-VPC-SG

Storage (volumes)

1 volume(s) - 8 GiB

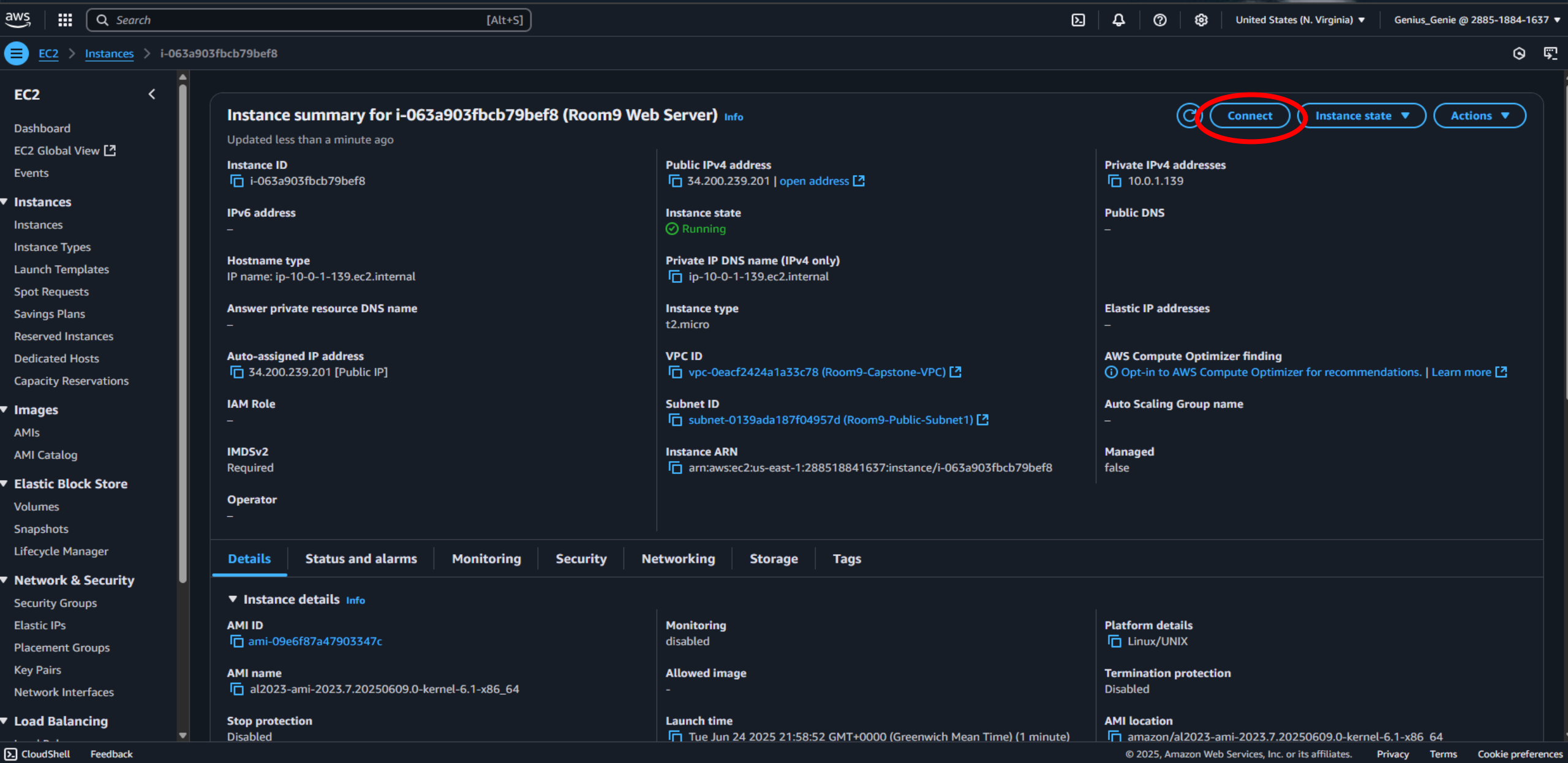
Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel | Launch instance | Preview code

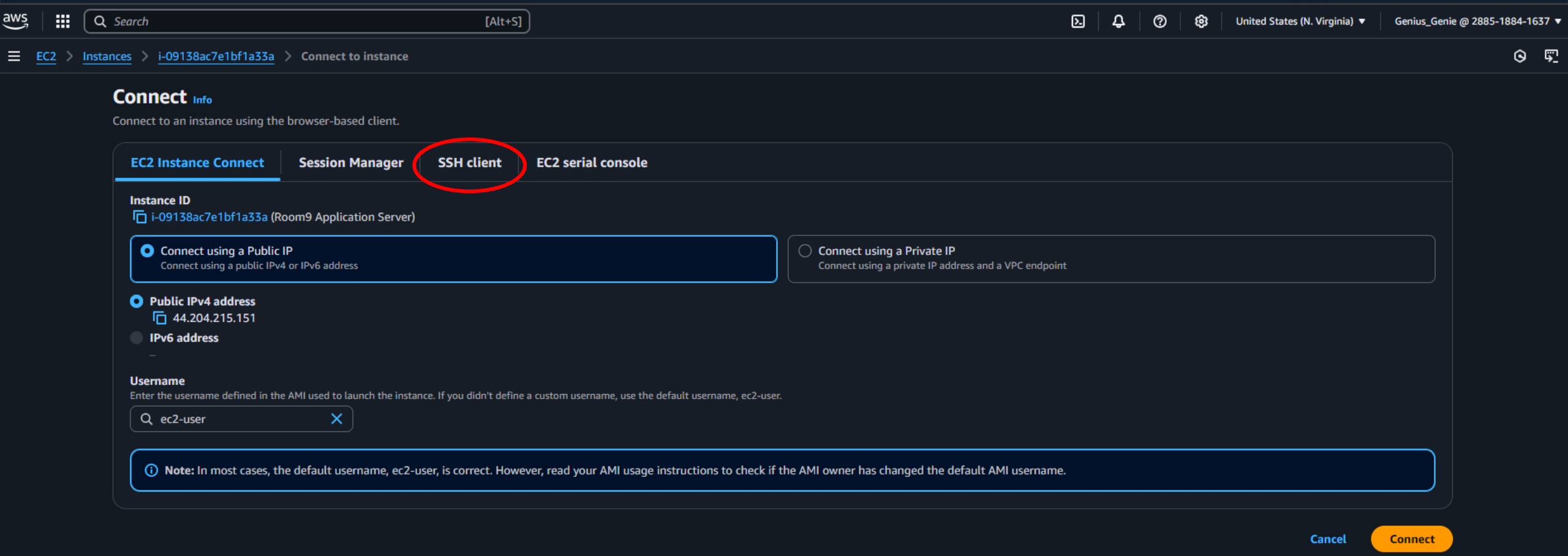
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step9: Instance launched

✓ To connect to your Instance, click the connect button as shown in this image



Step9: Instance launched



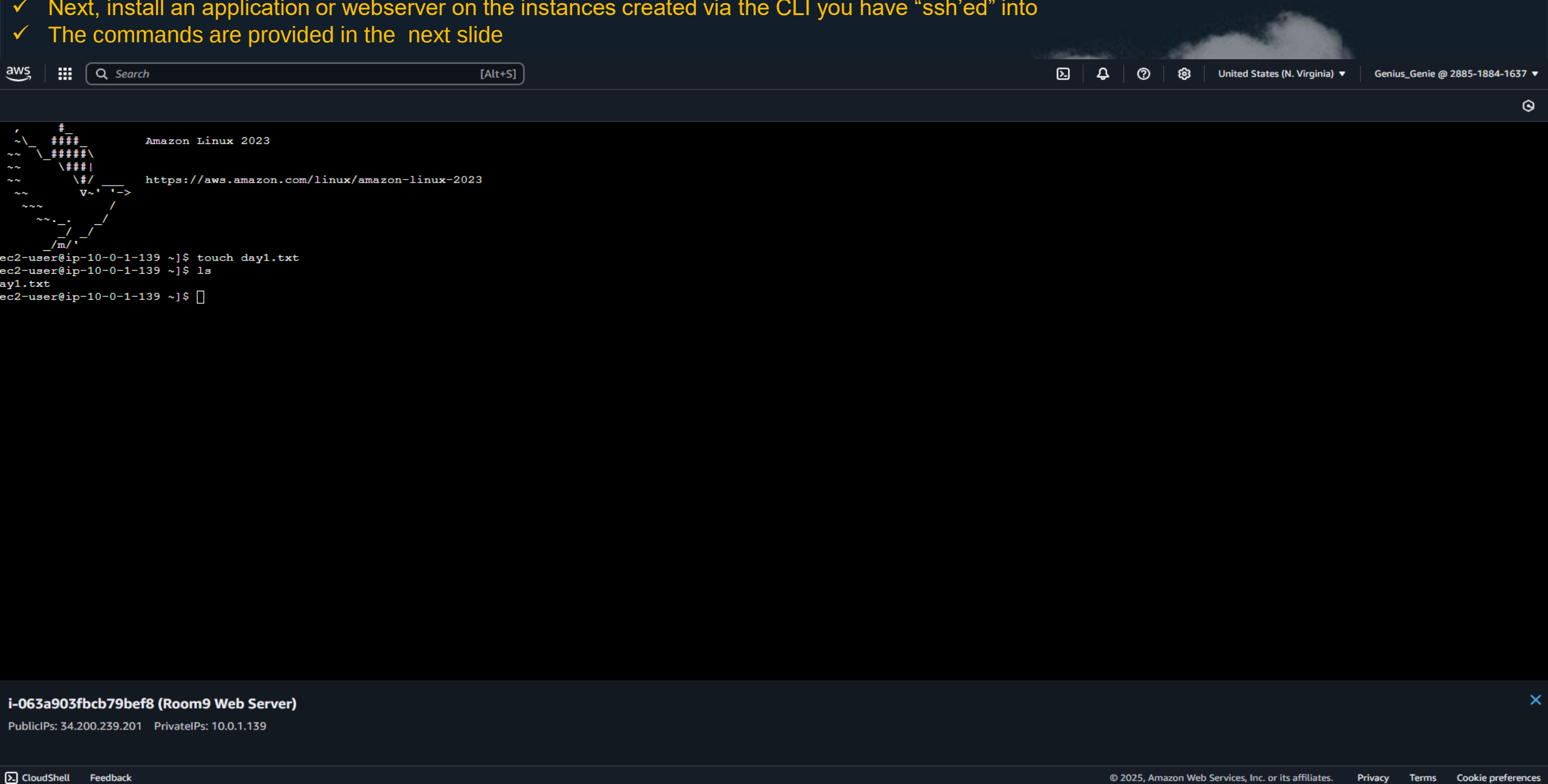
NOTES

- ✓ Connect to instance in the browser OR
- ✓ Connect via preferred CLI with the downloaded .pem file created.
- ✓ Copy the command for ssh via the SSH Client tab marked red

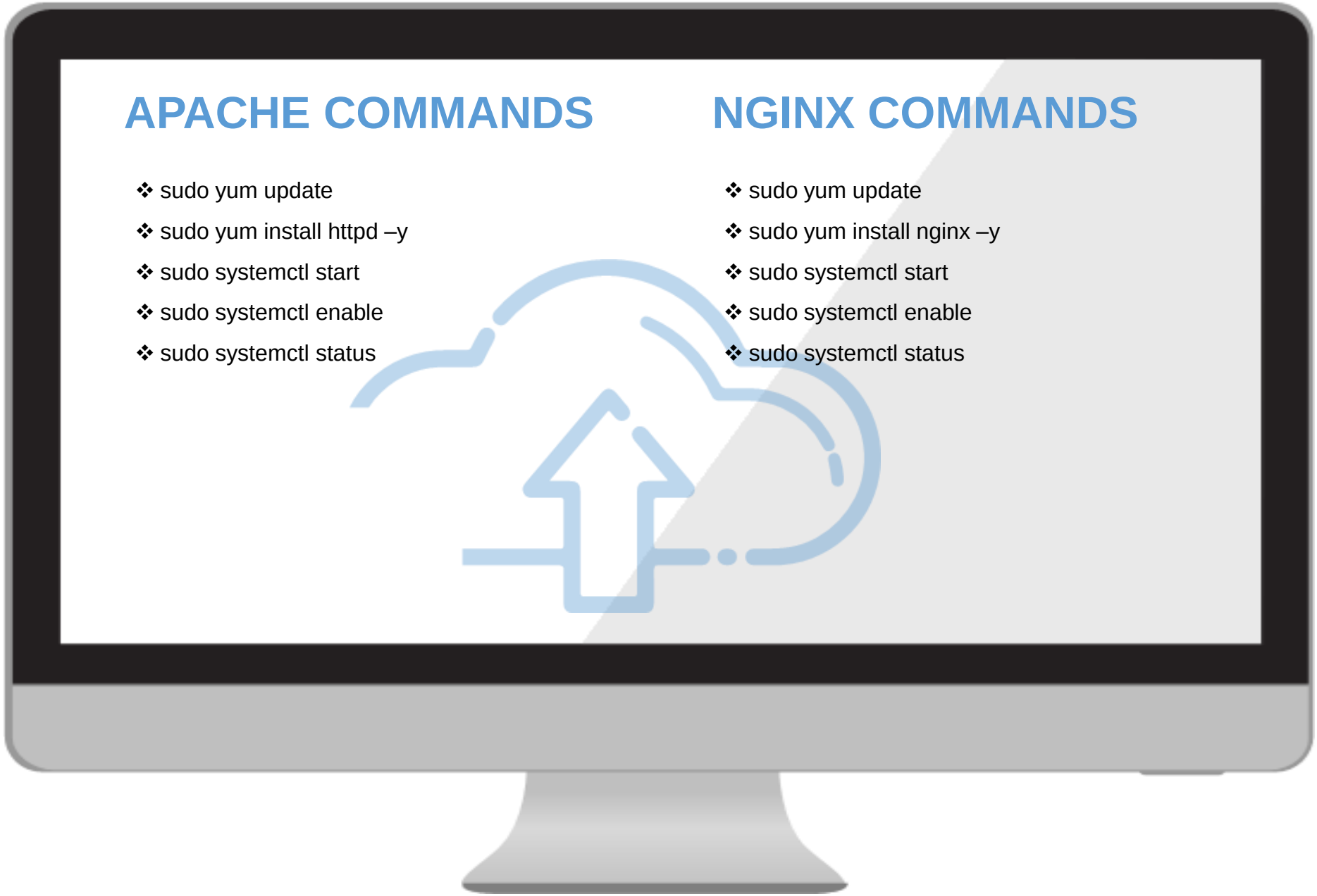
CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step9: Connected Instance

- ✓ Next, install an application or webserver on the instances created via the CLI you have “ssh’ed” into
- ✓ The commands are provided in the next slide



Step10: Install Apache Or Nginx Servers on instances



Step10: Installed Apache Server

```
  Verifying        : httpd-filesystem-2.4.62-1.amzn2023.noarch                                                                                                  7/12
  Verifying        : httpd-tools-2.4.62-1.amzn2023.x86_64                                                                                                       8/12
  Verifying        : libbrotli-1.0.9-4.amzn2023.0.2.x86_64                                                                                                      9/12
  Verifying        : mailcap-2.1.49-3.amzn2023.0.3.noarch                                                                                                      10/12
  Verifying        : mod_http2-2.0.27-1.amzn2023.0.3.x86_64                                                                                                    11/12
  Verifying        : mod_lua-2.4.62-1.amzn2023.x86_64                                                                                                          12/12
================================================================================================================================================================
WARNING:
  A newer release of "Amazon Linux" is available.

  Available Versions:

  Version 2023.7.20250623:
    Run the following command to upgrade to 2023.7.20250623:

      dnf upgrade --releasever=2023.7.20250623

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.7.20250623.html

================================================================================================================================================================

Installed:
  apr-1.7.5-1.amzn2023.0.4.x86_64                    apr-util-1.6.3-1.amzn2023.0.1.x86_64       apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch  httpd-2.4.62-1.amzn2023.x86_64             httpd-core-2.4.62-1.amzn2023.x86_64
  httpd-filesystem-2.4.62-1.amzn2023.noarch          httpd-tools-2.4.62-1.amzn2023.x86_64       libbrotli-1.0.9-4.amzn2023.0.2.x86_64
  mailcap-2.1.49-3.amzn2023.0.3.noarch               mod_http2-2.0.27-1.amzn2023.0.3.x86_64     mod_lua-2.4.62-1.amzn2023.x86_64

Complete!
[ec2-user@ip-10-0-1-100 ~]$ sudo systemctl start httpd
[ec2-user@ip-10-0-1-100 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-10-0-1-100 ~]$
```

✓ Install Apache server on instance one.

i-0fa52ea8b5bfa31d4 (Room9-EC2-Instance)

PublicIPs: 100.25.98.17 PrivateIPs: 10.0.1.100

Step10: Installed Nginx on instance 2.

```
  A newer release of "Amazon Linux" is available.

  Available Versions:

  Version 2023.7.20250623:
    Run the following command to upgrade to 2023.7.20250623:

      dnf upgrade --releasever=2023.7.20250623

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.7.20250623.html

================================================================================================================================

Installed:
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch   gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64   libunwind-1.4.0-5.amzn2023.0.2.x86_64
  nginx-1:1.26.3-1.amzn2023.0.1.x86_64                nginx-core-1:1.26.3-1.amzn2023.0.1.x86_64     nginx-filesystem-1:1.26.3-1.amzn2023.0.1.noarch
  nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
[ec2-user@ip-10-0-3-113 ~]$ sudo systemctl start nginx
[ec2-user@ip-10-0-3-113 ~]$ sudo systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
[ec2-user@ip-10-0-3-113 ~]$
```

i-0e36aed23e0be0f65 (Room9-EC2-Instance2)

PublicIPs: 44.201.131.109 PrivateIPs: 10.0.3.113

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

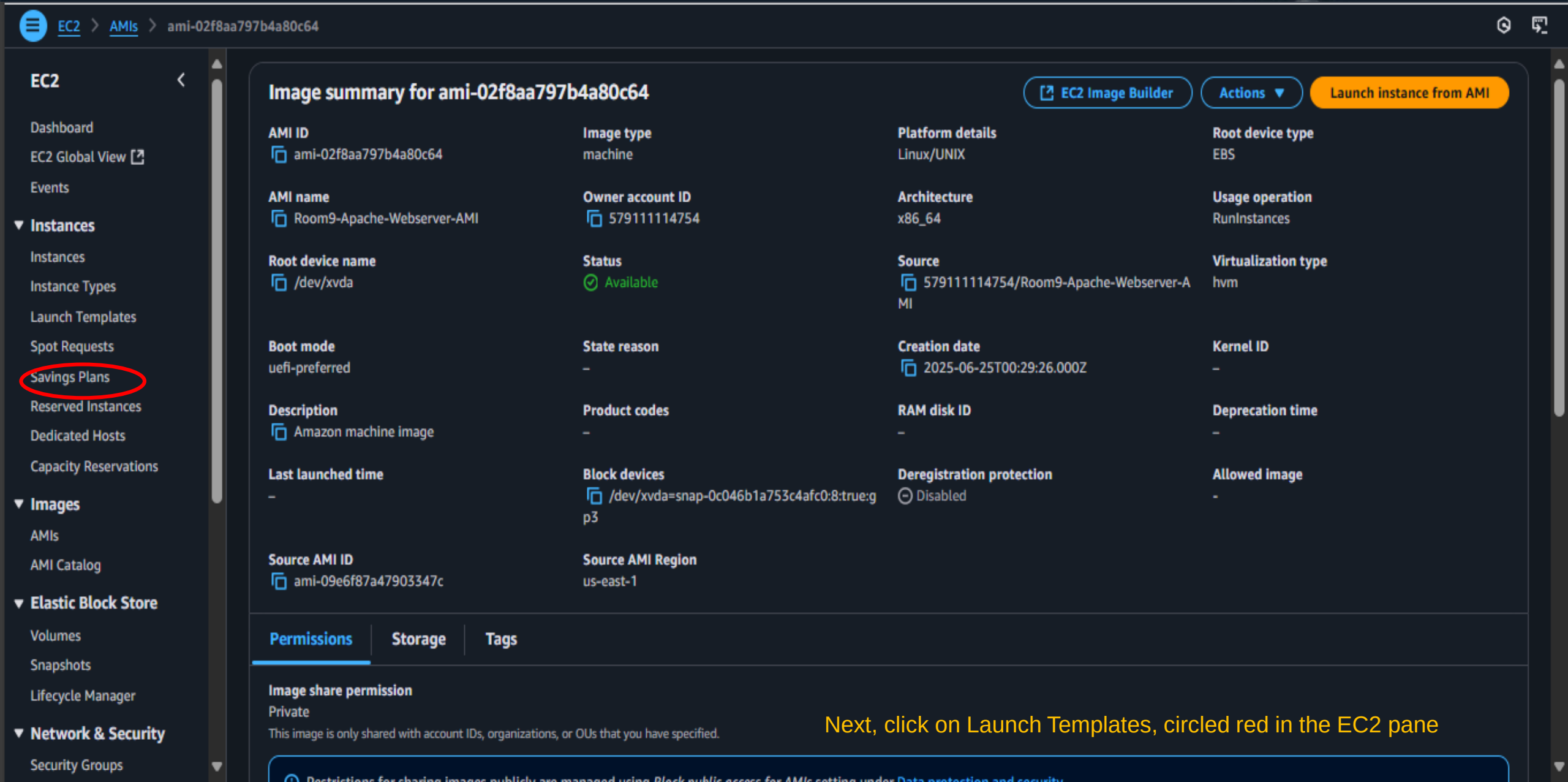
Step11: Create and configure an AMI

- Select the instance you want to create the AMI from.
- Fill in the details: Name and description
- **No reboot**: Leave unchecked (or check it if you don’t want the instance rebooted)
- Click **Create image**
- Go to **AMIs** under **Images** in the EC2 sidebar
- Find your newly created AMI

NOTES

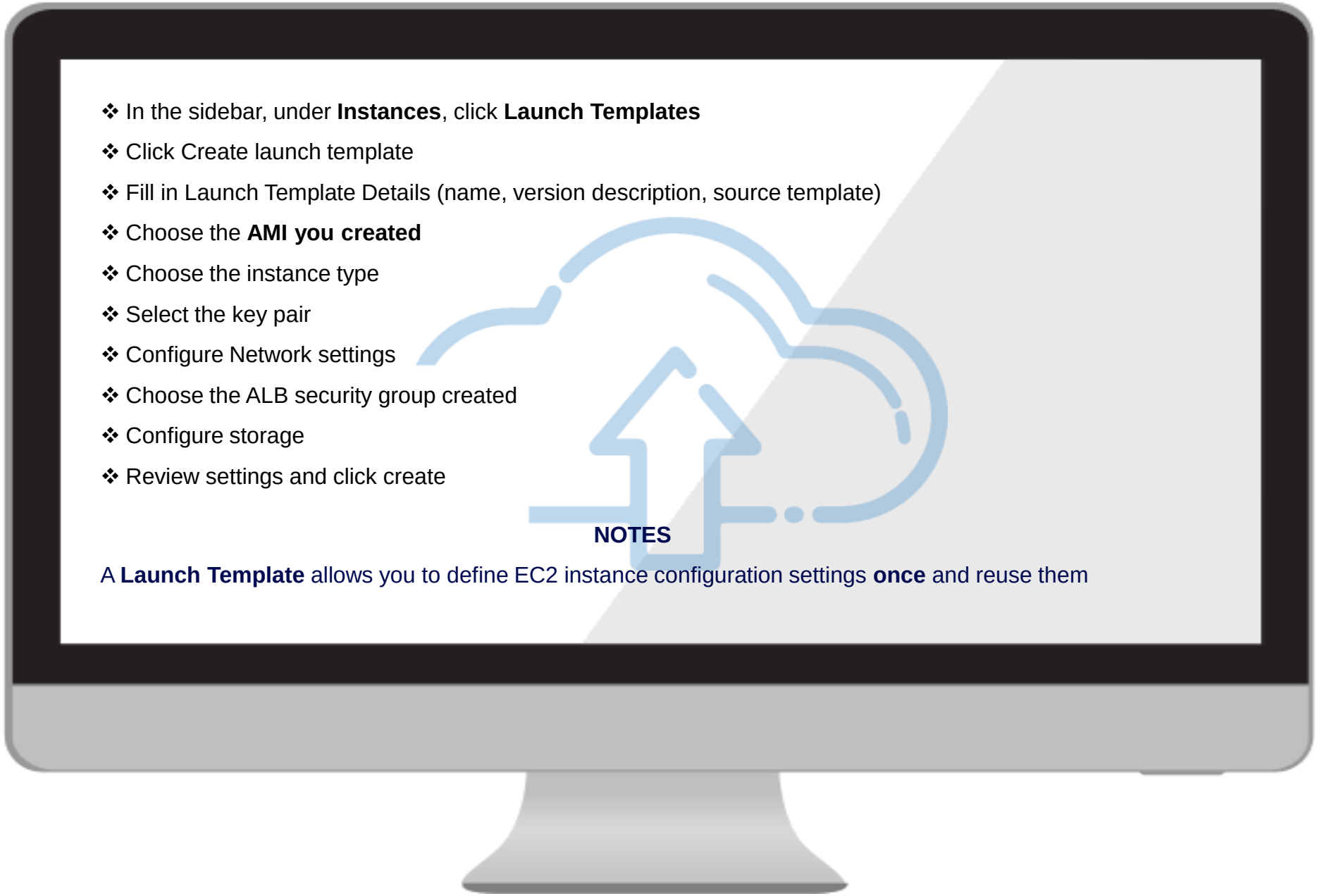
An **Amazon Machine Image (AMI)** is a snapshot of an EC2 instance's configuration — including its OS, installed software, settings, and data — that you can **reuse to launch new EC2 instances** with the exact same setup.

Step11: Created and configured AMI



Next, click on Launch Templates, circled red in the EC2 pane

Step12: Create and launch template

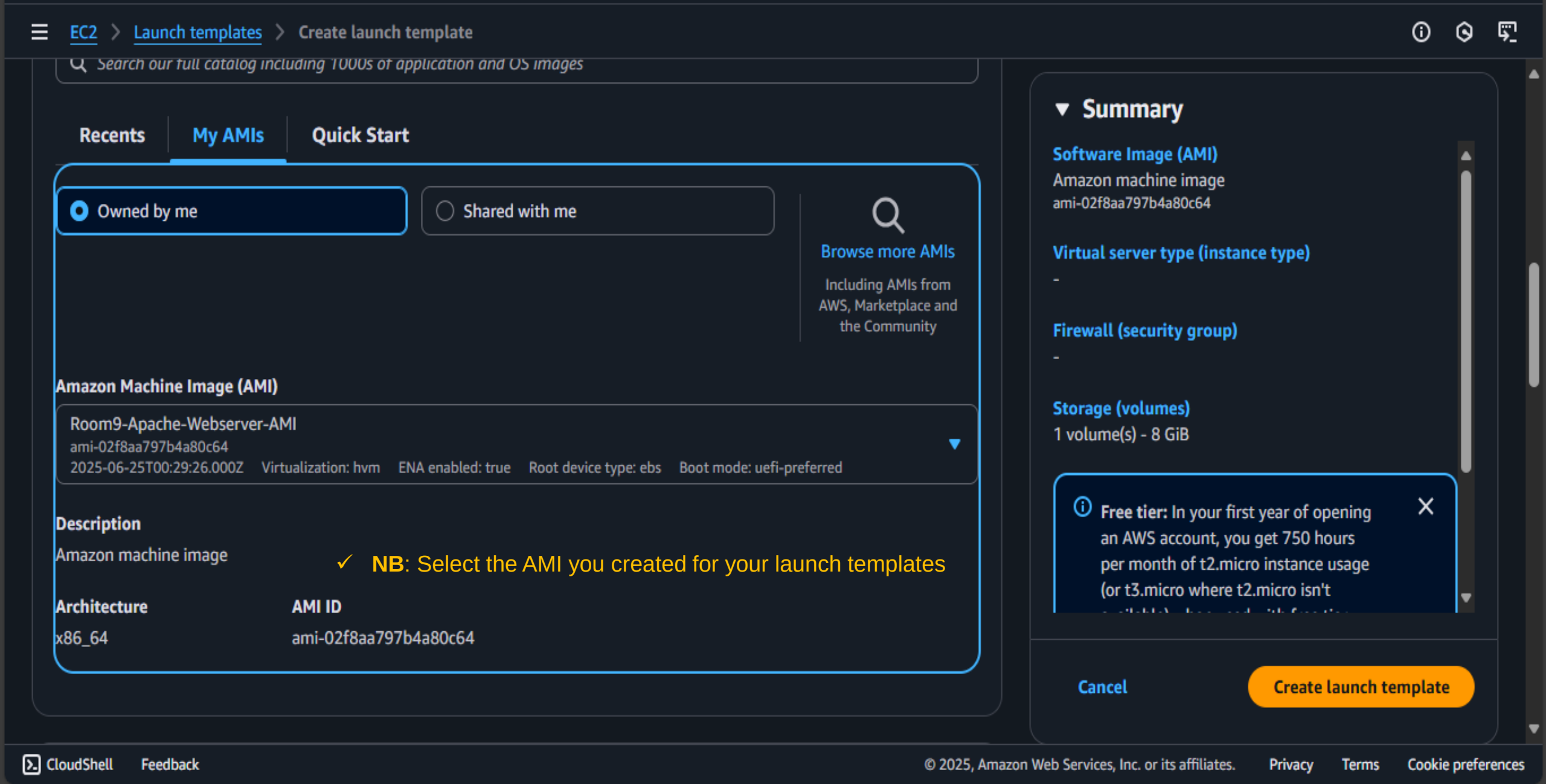


- In the sidebar, under **Instances**, click **Launch Templates**
- Click Create launch template
- Fill in Launch Template Details (name, version description, source template)
- Choose the **AMI you created**
- Choose the instance type
- Select the key pair
- Configure Network settings
- Choose the ALB security group created
- Configure storage
- Review settings and click create

NOTES

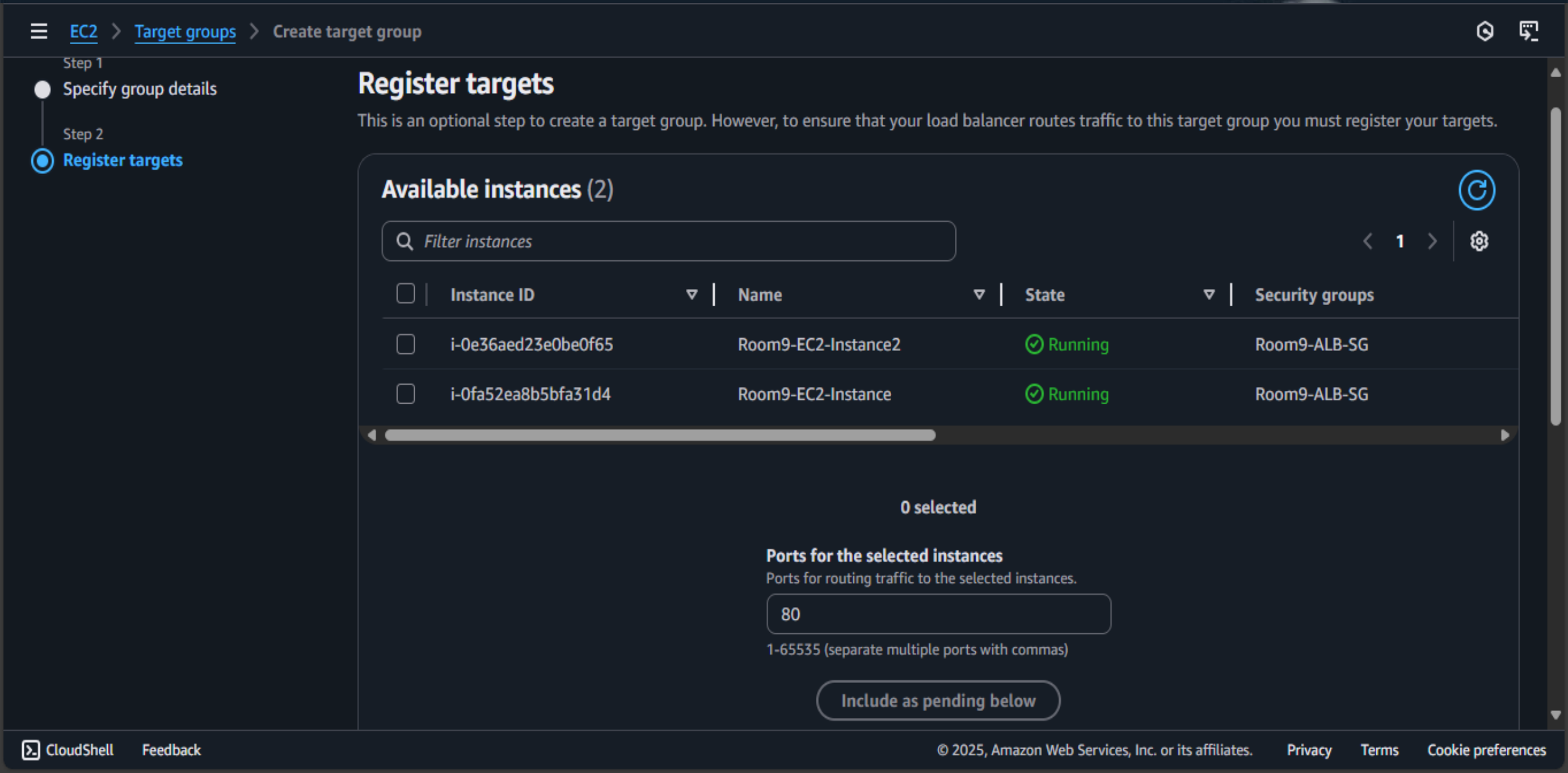
A **Launch Template** allows you to define EC2 instance configuration settings **once** and reuse them

Step 12: Create launch templates

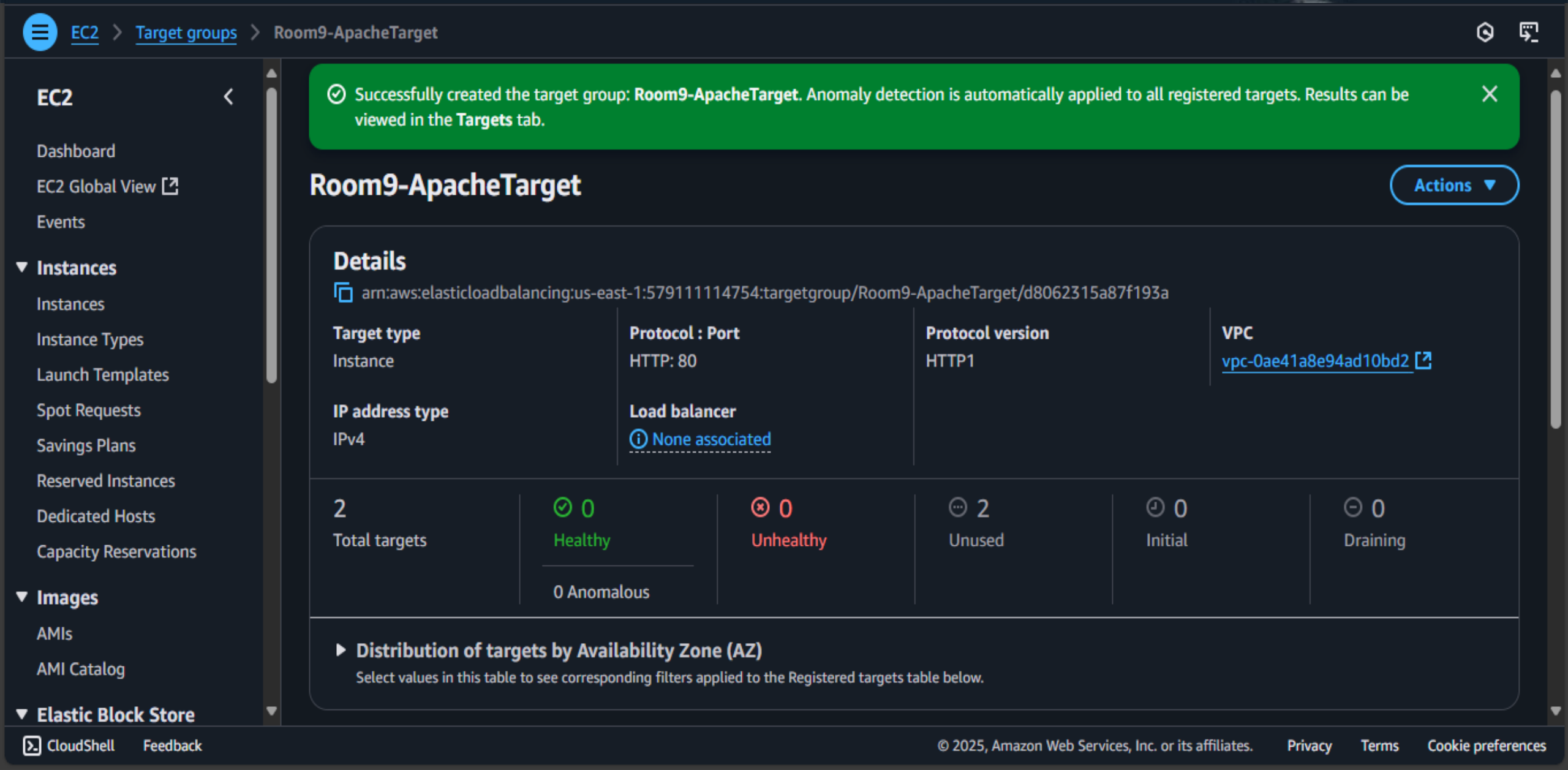


Step13: Create Target groups for the Application Load Balancer

Step 13: Create target group for application load balancer



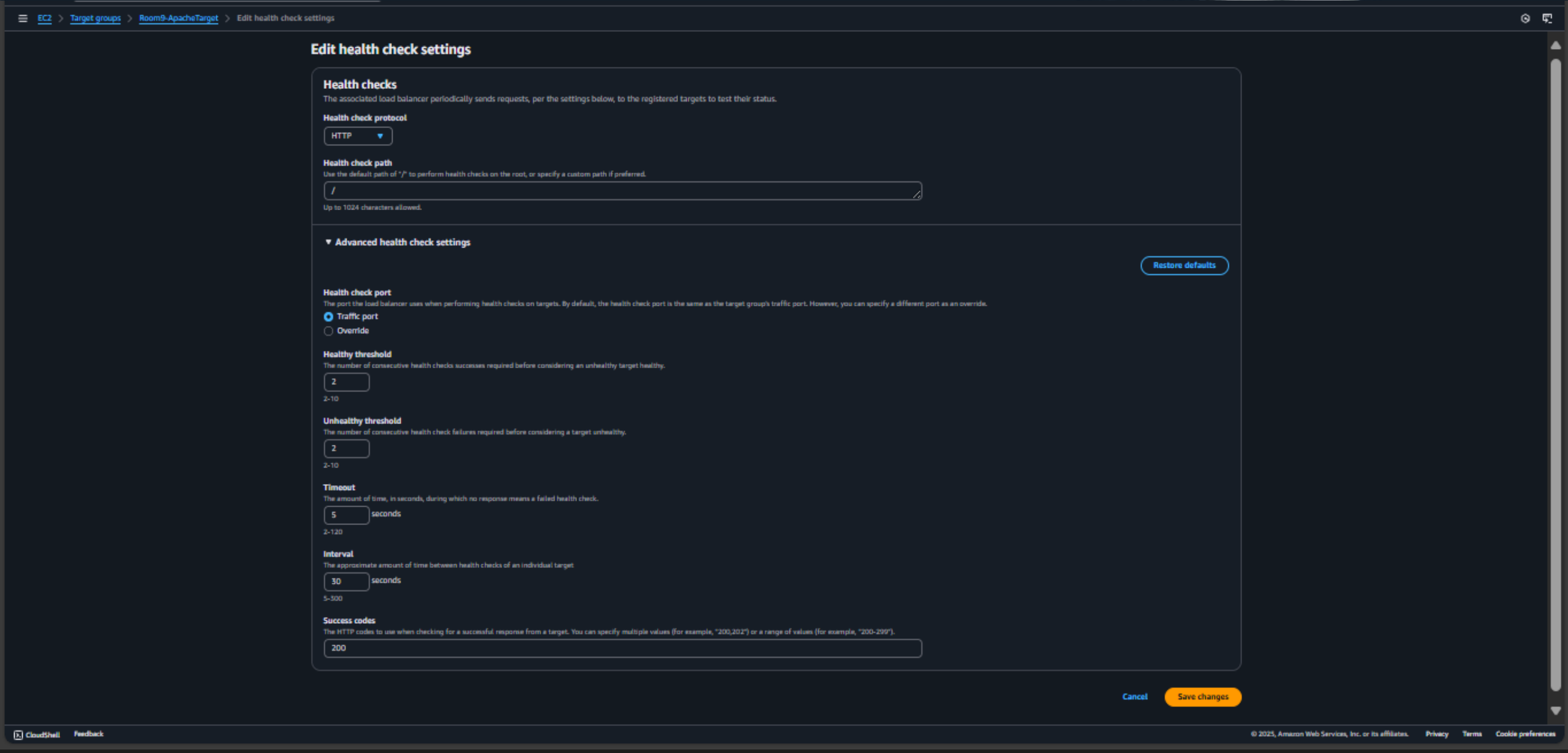
Step 13: Created target for load balancer



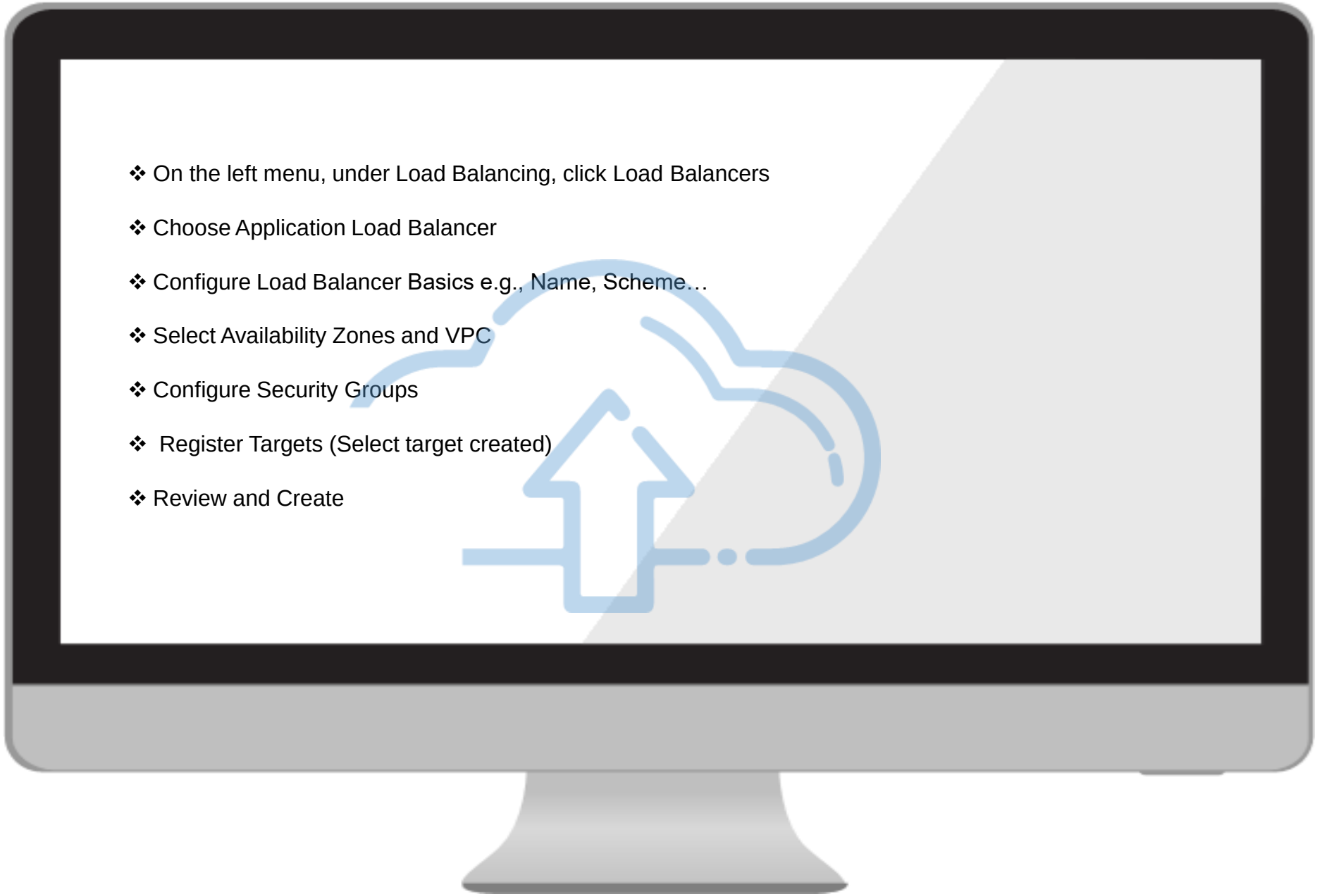
Step14: Configure Health Checks for Target Group

Step 14: Set up health checks

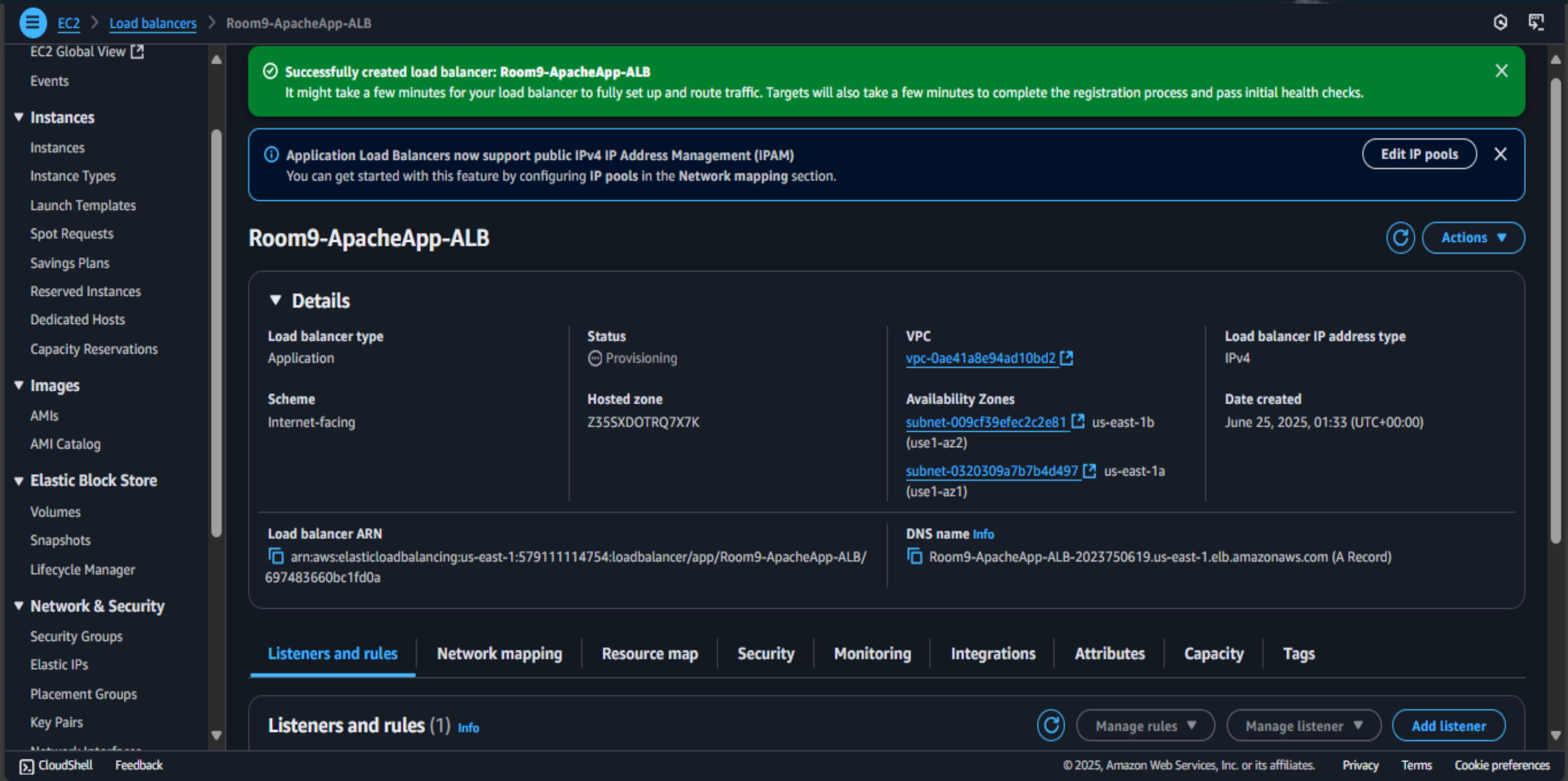
✓ Set up health checks to monitor the status of instances and route traffic only to healthy instances



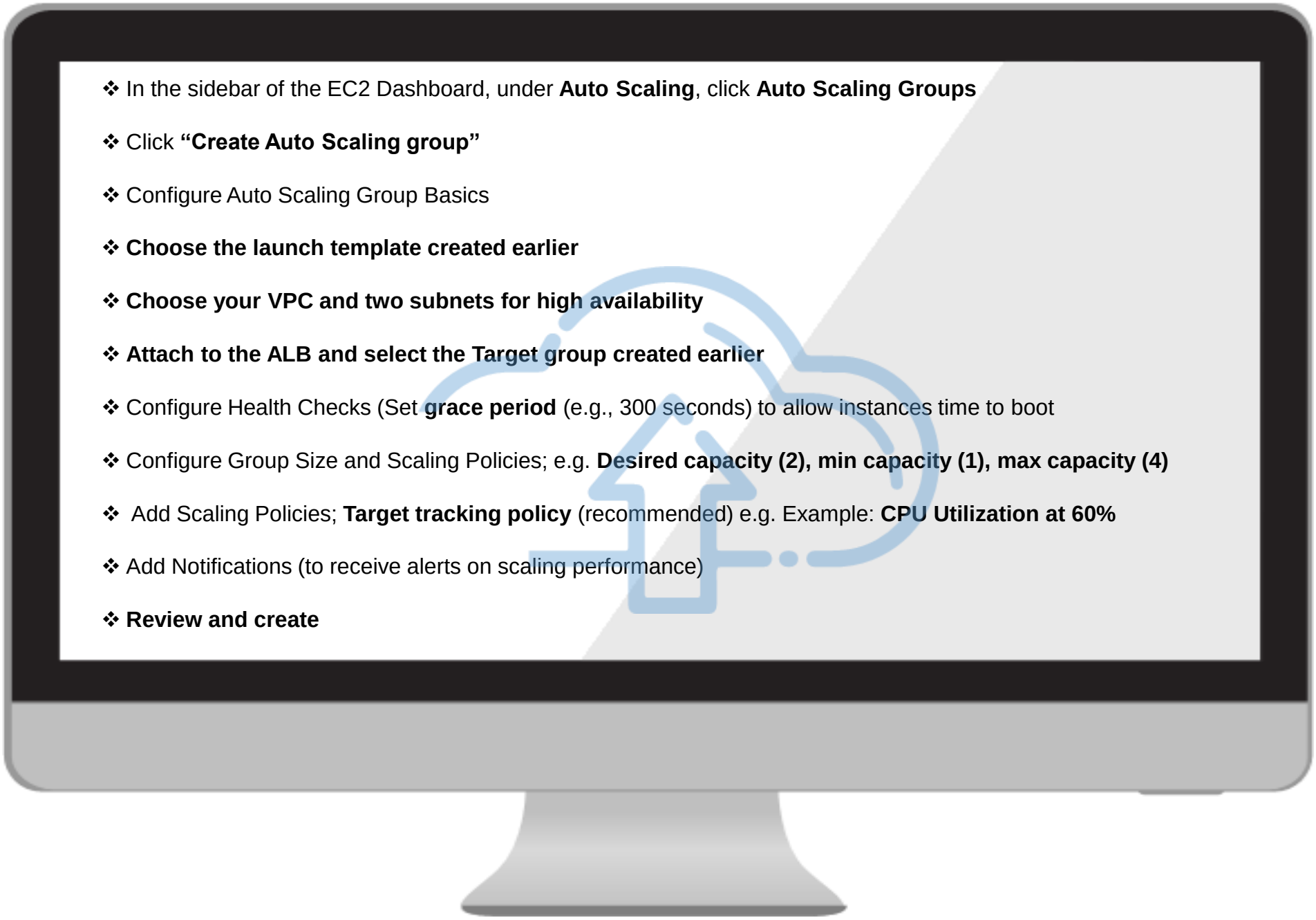
Step15: Create the application load balancer



Step15: Create your application load balancer



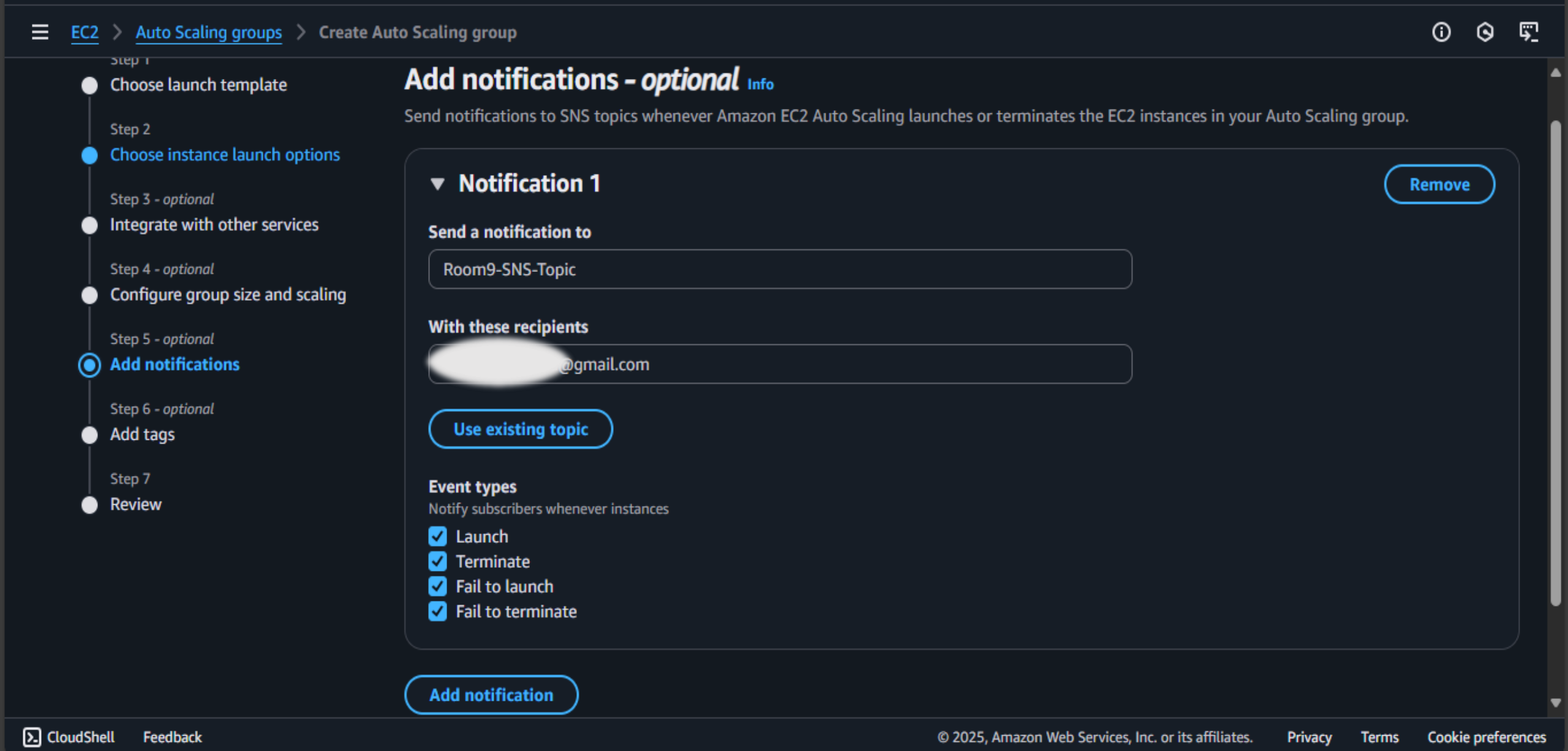
Step16: Create the auto scaling group



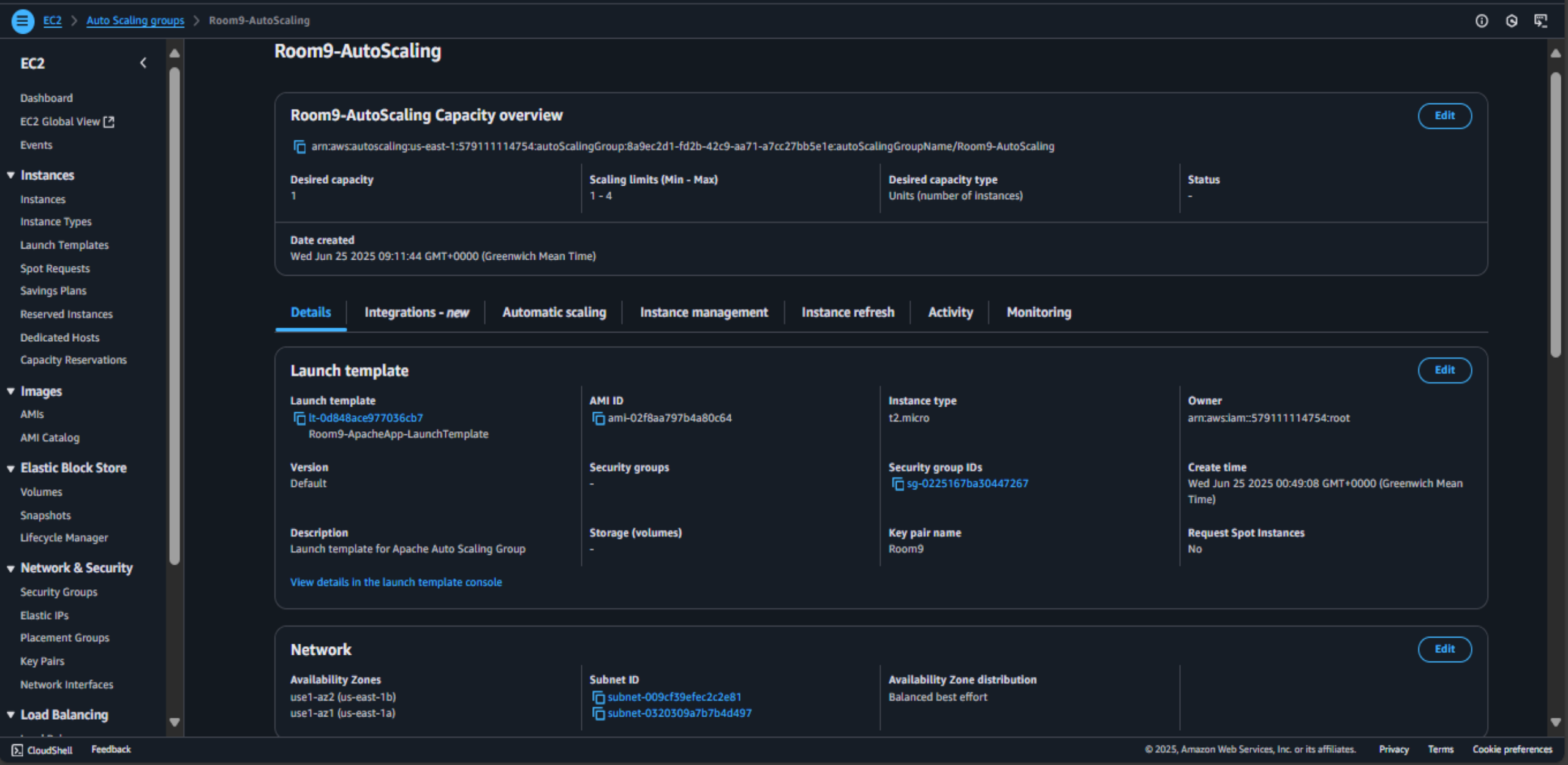
- In the sidebar of the EC2 Dashboard, under **Auto Scaling**, click **Auto Scaling Groups**
- Click **“Create Auto Scaling group”**
- Configure Auto Scaling Group Basics
- **Choose the launch template created earlier**
- **Choose your VPC and two subnets for high availability**
- **Attach to the ALB and select the Target group created earlier**
- Configure Health Checks (Set **grace period** (e.g., 300 seconds) to allow instances time to boot
- Configure Group Size and Scaling Policies; e.g. **Desired capacity (2), min capacity (1), max capacity (4)**
- Add Scaling Policies; **Target tracking policy** (recommended) e.g. Example: **CPU Utilization at 60%**
- Add Notifications (to receive alerts on scaling performance)
- **Review and create**

Step 16: Create your auto scaling group

✓ Subscribe to a notification service to receive updates on your scaling



Step 16: Created auto scaling group



Step 21: Confirmed notification settings

Simple Notification Service

Subscription confirmed!

You have successfully subscribed.

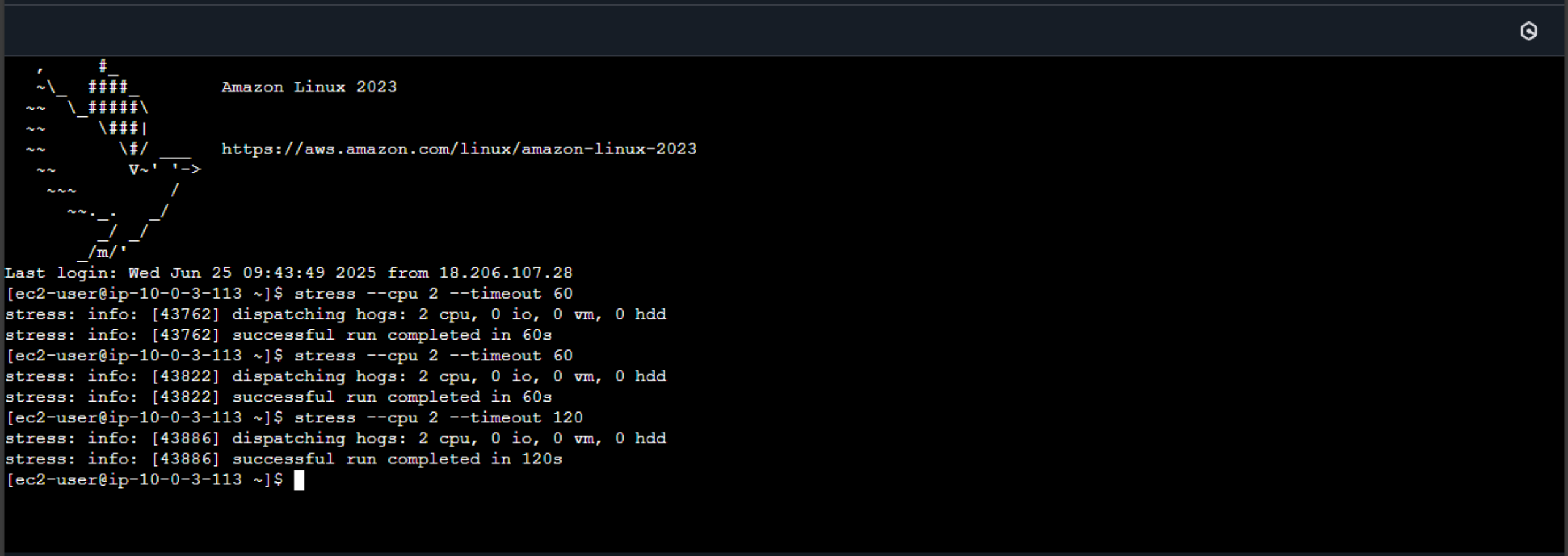
Your subscription's id is:
arn:aws:sns:us-east-1:579111114754:Room9-SNS-Topic:03dd79c9-4b3b-431f-868e-6324db05284d

If it was not your intention to subscribe, click here to unsubscribe.

Step17: Monitoring, Testing and Optimization

Step 17: Monitoring, Testing And Optimization

✓ Simulate CPU-Intensive workload with the stress command

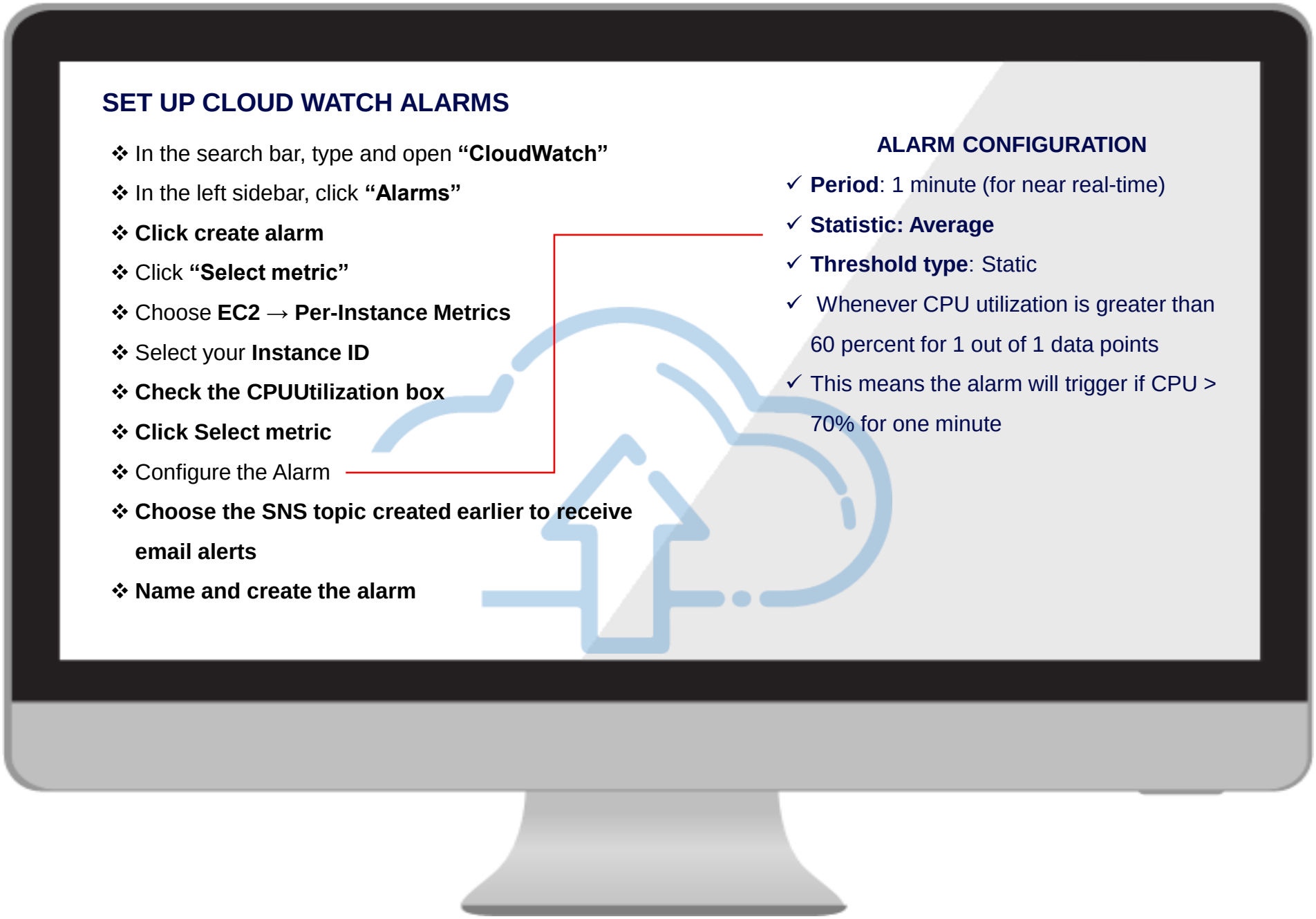


i-0e36aed23e0be0f65 (Room9-EC2-Instance2)

PublicIPs: 44.201.131.109 PrivateIPs: 10.0.3.113

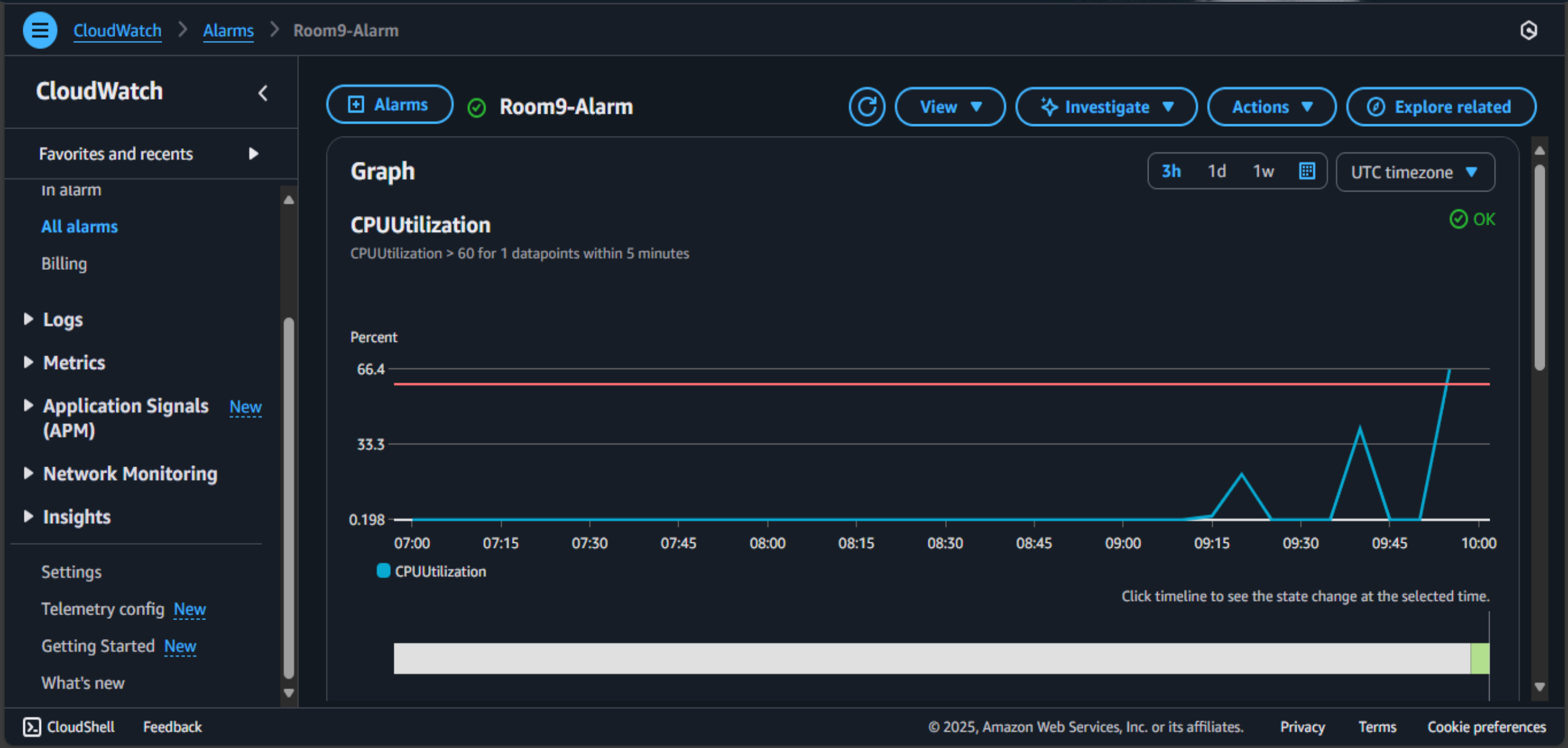
CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step17: Monitoring, Testing and Optimization



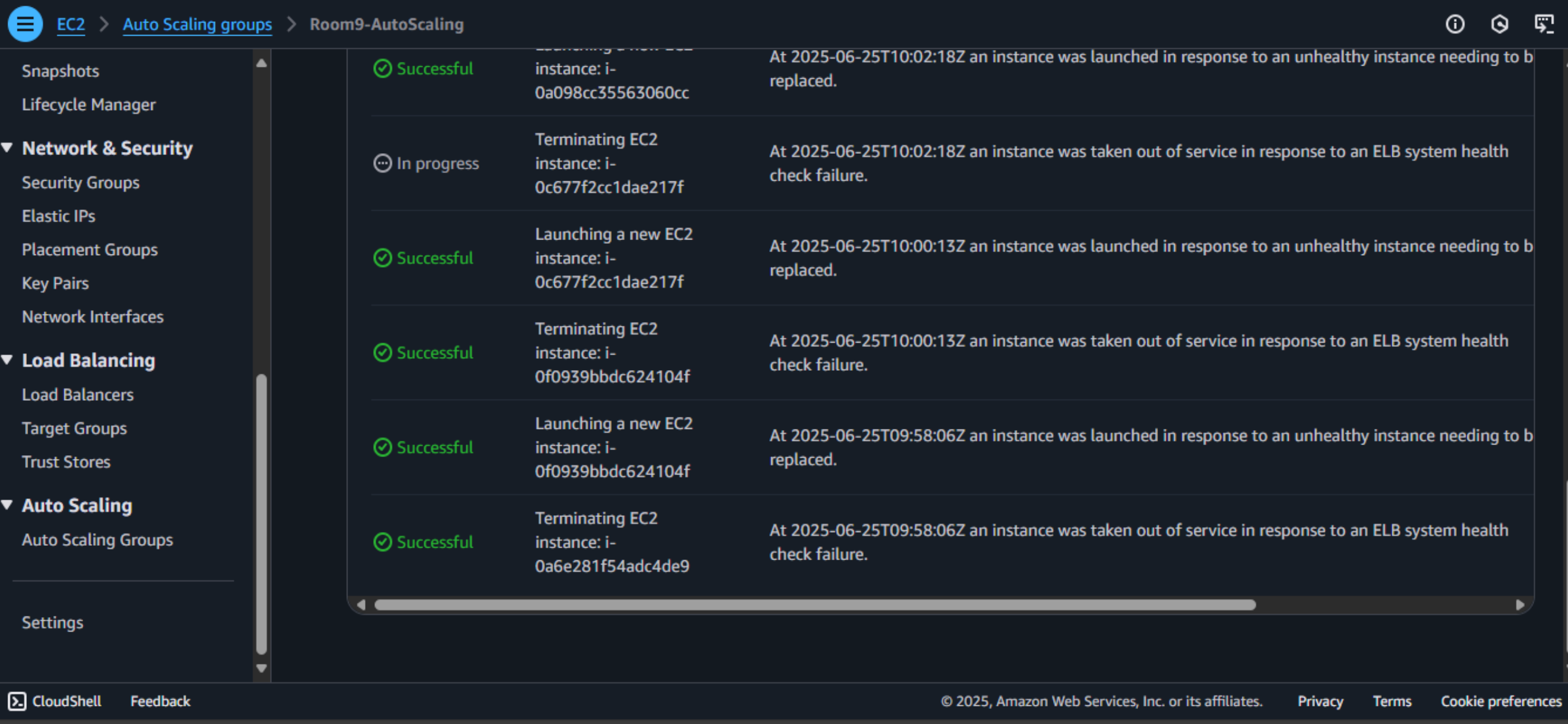
Step17: Testing And Optimization

✓ Real time monitoring in cloud watch



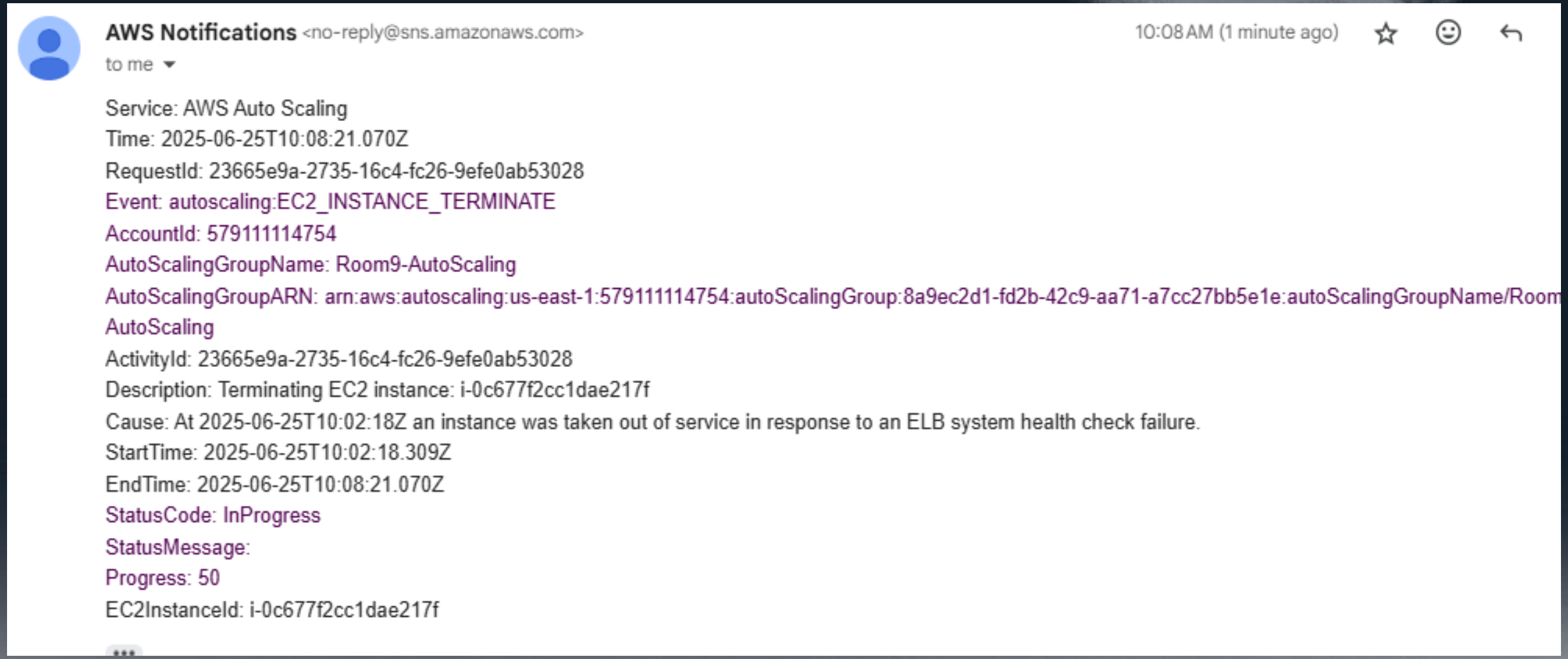
Step17: Testing And Optimization

✓ **Traffic simulation triggered auto-scaling events, resulting in new instances being launched and later terminated in line with defined scaling policies.**



Step17: Testing And Optimization

- ✓ Email Notification : Additionally, the integrated SNS configuration successfully sent email notifications, confirming each event in real time

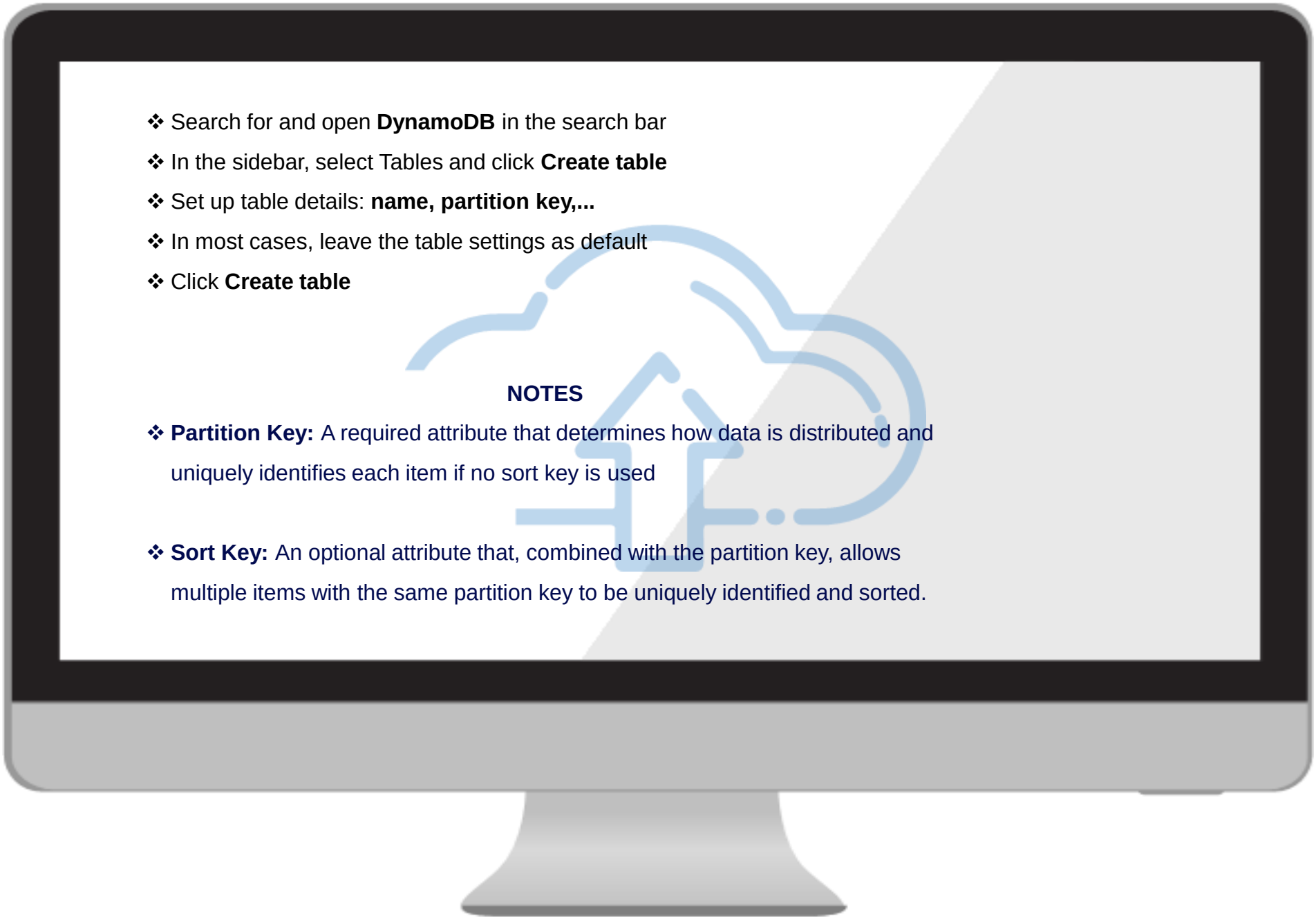


AWS Notifications <no-reply@sns.amazonaws.com> 10:08 AM (1 minute ago)

to me

Service: AWS Auto Scaling
Time: 2025-06-25T10:08:21.070Z
RequestId: 23665e9a-2735-16c4-fc26-9efe0ab53028
Event: autoscaling:EC2_INSTANCE_TERMINATE
AccountId: 579111114754
AutoScalingGroupName: Room9-AutoScaling
AutoScalingGroupARN: arn:aws:autoscaling:us-east-1:579111114754:autoScalingGroup:8a9ec2d1-fd2b-42c9-aa71-a7cc27bb5e1e:autoScalingGroupName/Room
AutoScaling
ActivityId: 23665e9a-2735-16c4-fc26-9efe0ab53028
Description: Terminating EC2 instance: i-0c677f2cc1dae217f
Cause: At 2025-06-25T10:02:18Z an instance was taken out of service in response to an ELB system health check failure.
StartTime: 2025-06-25T10:02:18.309Z
EndTime: 2025-06-25T10:08:21.070Z
StatusCode: InProgress
StatusMessage:
Progress: 50
EC2InstanceId: i-0c677f2cc1dae217f

Step18: Create and configure DynamoDB



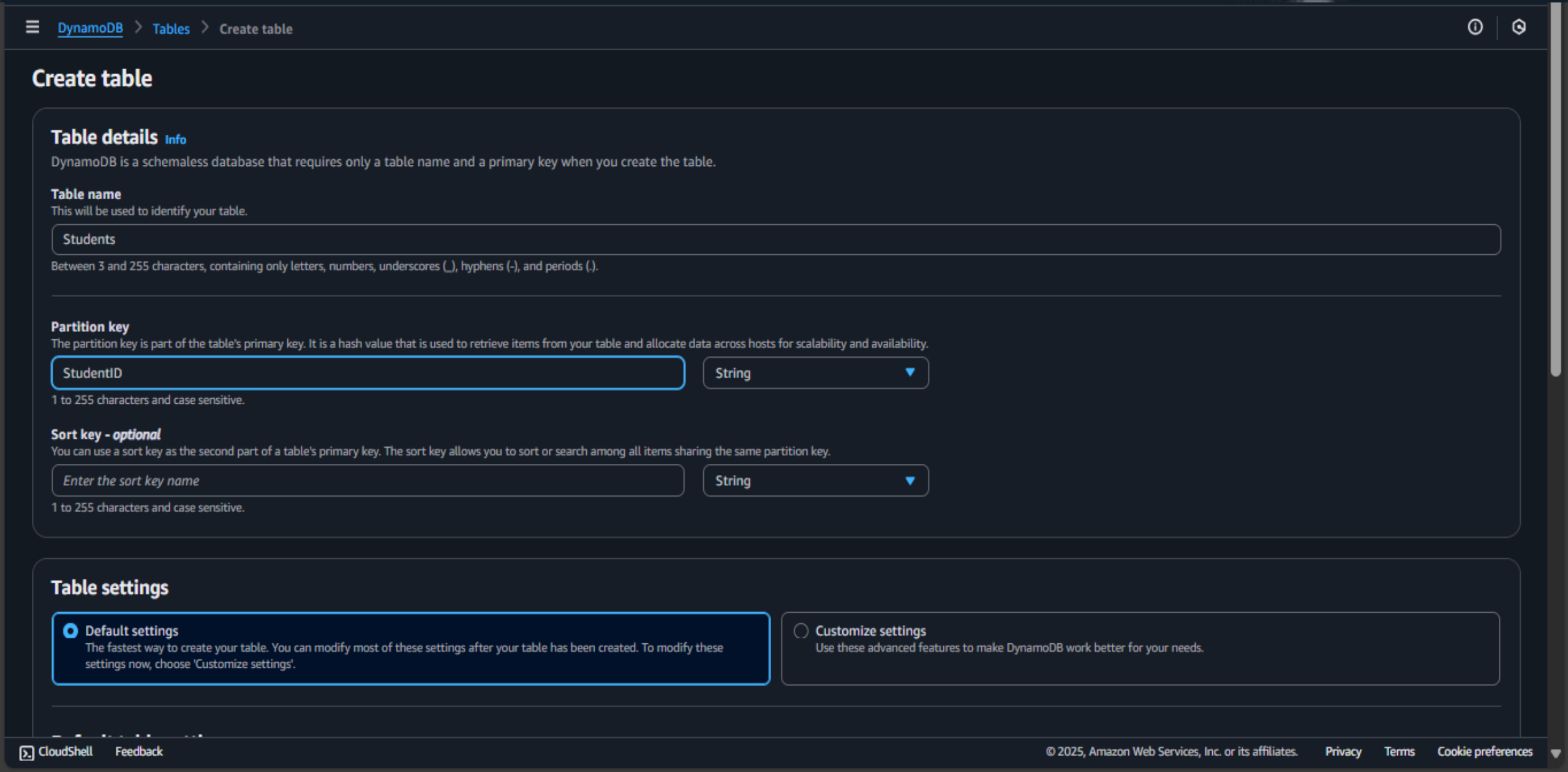
- Search for and open **DynamoDB** in the search bar
- In the sidebar, select Tables and click **Create table**
- Set up table details: **name, partition key,...**
- In most cases, leave the table settings as default
- Click **Create table**

NOTES

- **Partition Key:** A required attribute that determines how data is distributed and uniquely identifies each item if no sort key is used

- **Sort Key:** An optional attribute that, combined with the partition key, allows multiple items with the same partition key to be uniquely identified and sorted.

Step18: Create and configure DynamoDB

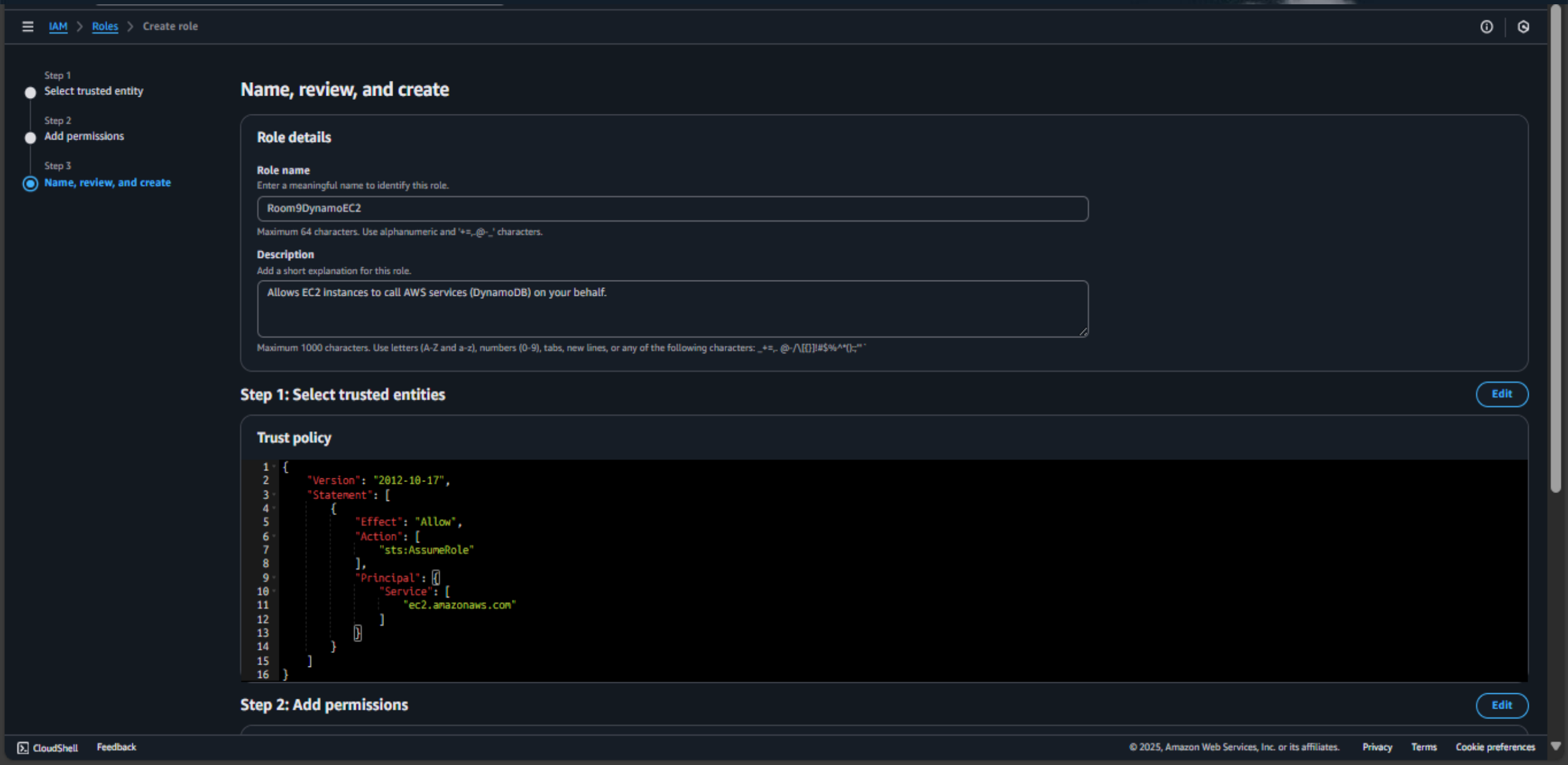


Step19: Create an IAM role and attach to instance

Step19: Create IAM Role

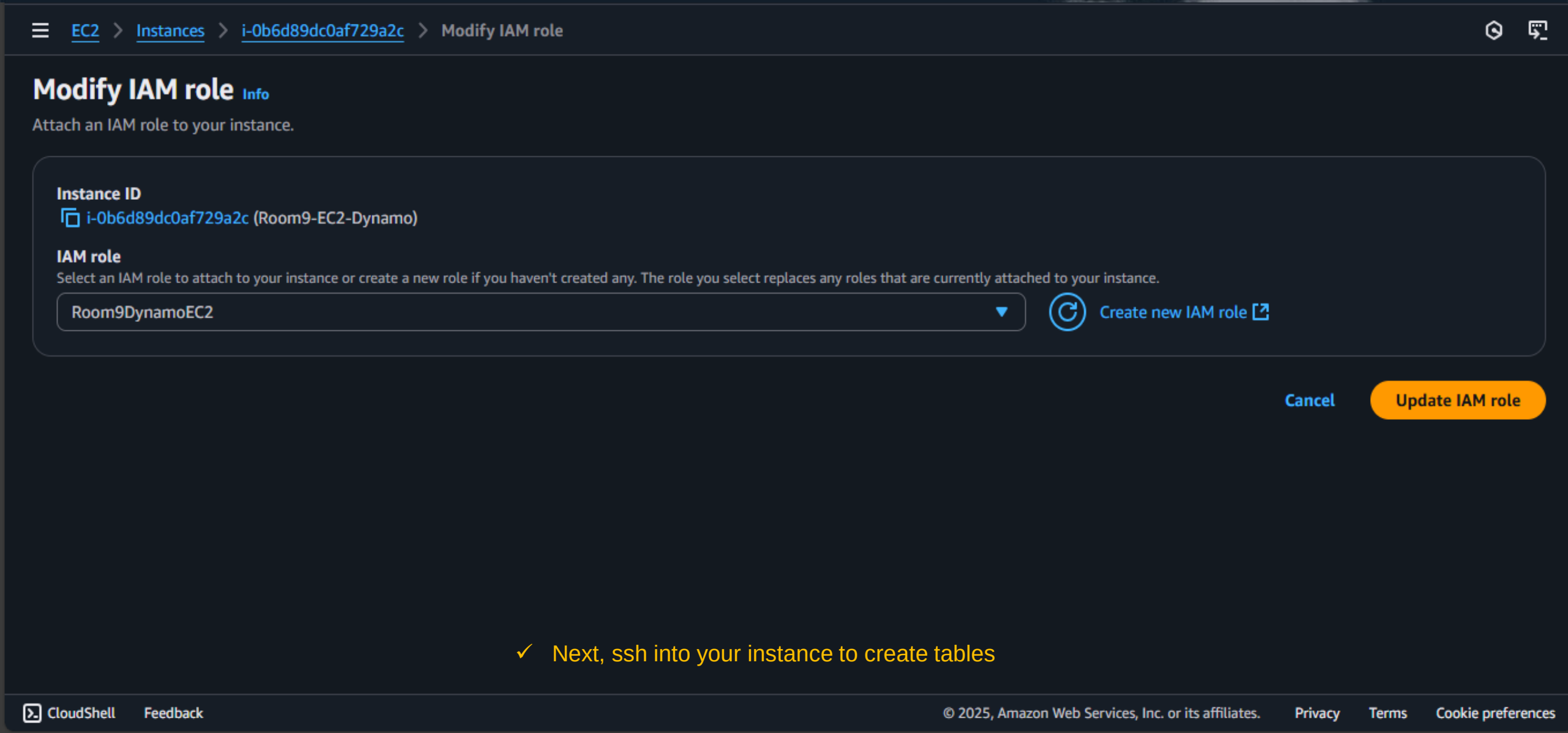
✓ Grant a full DynamoDB access policy for the role

✓ Next, attach the IAM role to an EC2 instance



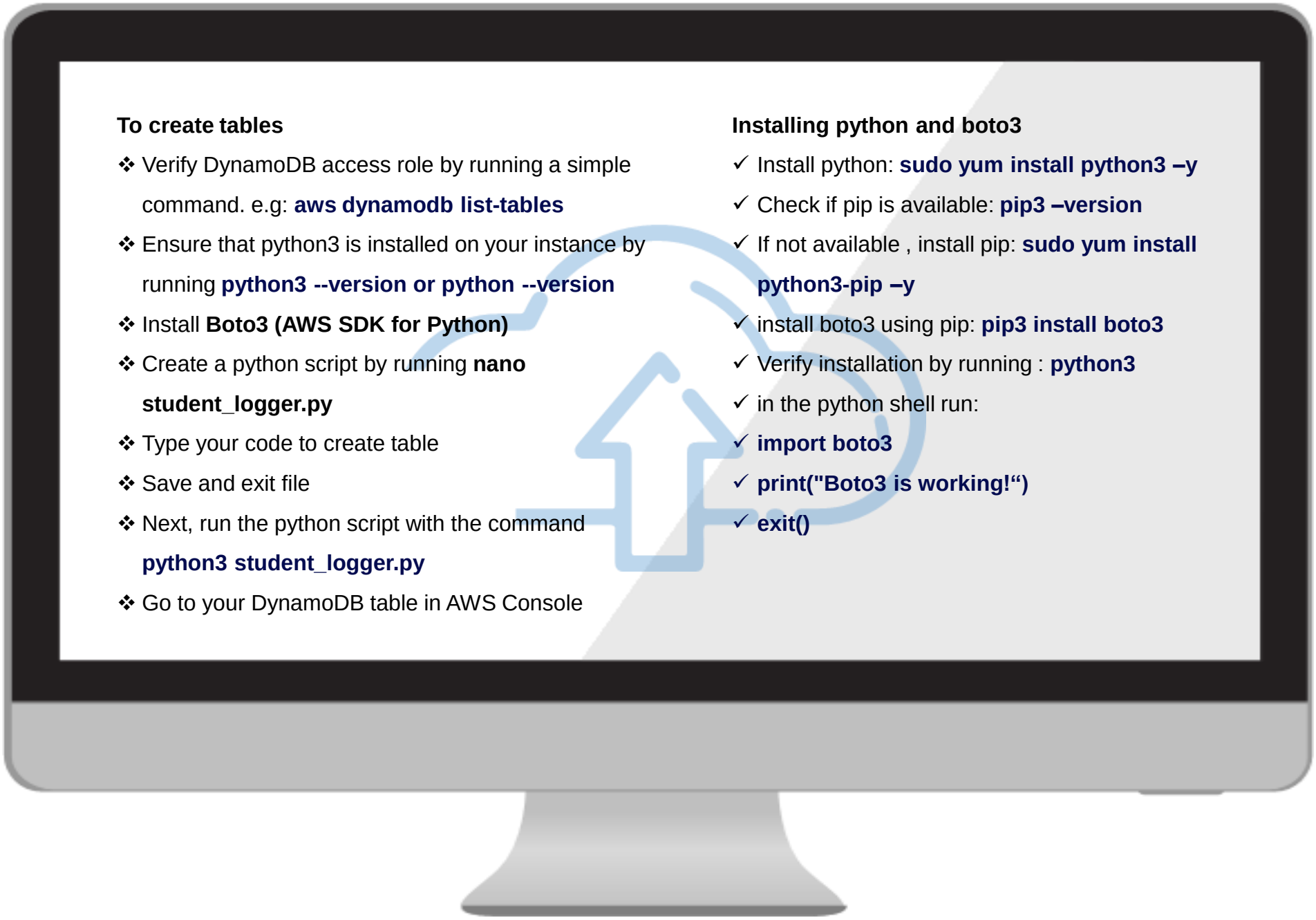
Step19: Attach IAM Role to your EC2 instance

✓ Attach the IAM role to your EC2 instance to grant access to your DB.



✓ Next, ssh into your instance to create tables

Step19: Create tables for your DB



To create tables

- ❖ Verify DynamoDB access role by running a simple command. e.g: **aws dynamodb list-tables**
- ❖ Ensure that python3 is installed on your instance by running **python3 --version or python --version**
- ❖ Install **Boto3 (AWS SDK for Python)**
- ❖ Create a python script by running **nano student_logger.py**
- ❖ Type your code to create table
- ❖ Save and exit file
- ❖ Next, run the python script with the command **python3 student_logger.py**
- ❖ Go to your DynamoDB table in AWS Console

Installing python and boto3

- ✓ Install python: **sudo yum install python3 –y**
- ✓ Check if pip is available: **pip3 –version**
- ✓ If not available , install pip: **sudo yum install python3-pip –y**
- ✓ install boto3 using pip: **pip3 install boto3**
- ✓ Verify installation by running : **python3**
- ✓ in the python shell run:
- ✓ **import boto3**
- ✓ **print("Boto3 is working!“)**
- ✓ **exit()**

Step19: Create tables for your DB

✓ Set up Boto3 (the AWS SDK for Python) on the EC2 instance to enable seamless interaction with AWS services through Python scripts.

Step19: Create tables for your DB

✓ Use a python script to create a table in your DB

```
import boto3

# Create a DynamoDB resource using the us-east-1 region
dynamodb= boto3.resource('dynamodb', region_name='us-east-1')

# Reference your DynamoDB table
table= dynamodb.Table('Students')

# List of students to add
students = [
        {'StudentID': '001', 'name': 'Harrison Adom', 'course': 'AWS Cloud Fundamentals'},
        {'StudentID': '002', 'name': 'Fadilatu Suhuyini Mahamud', 'course': 'Linux Basics'},
        {'StudentID': '003', 'name': 'Richmond Nyarko', 'course': 'AMZ CloudFront'},
        {'StudentID': '004', 'name': 'Genevieve Kankam Mensah', 'course': 'Quantum Engineering'},
        {'StudentID': '005', 'name': 'Clifford Mills', 'course': 'Statistics'},]

# Loop through and insert each studentre Is     ^K Cut          ^T Execute0 line^C Location     M-U Undo        M-A Set Mark    M-] T
for student in students:File    ^\ Replace      ^U Paste        ^J Justify      ^/ Go To Line   M-E Redo        M-6 Copy        ^B Wh
ere Was print(f"Inserted: {student['name']} | Status: {response['ResponseMetadata']['HTTPStatusCode']}")
```

i-0502ca591120bdbff (Room9-Dynamo-EC2)

PublicIPs: 54.152.23.128 PrivateIPs: 10.0.3.203

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step19: Created tables

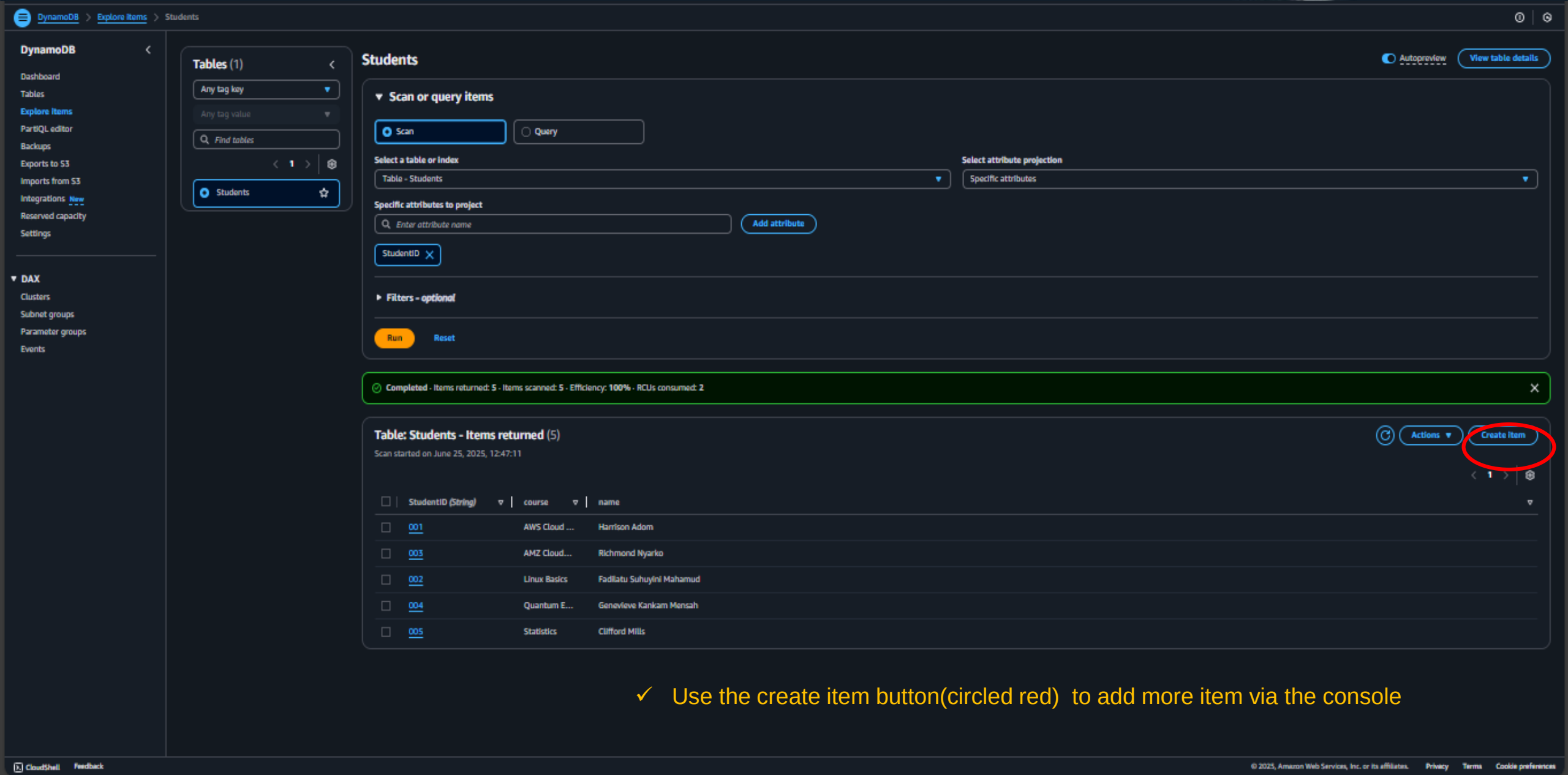
```
[ec2-user@ip-10-0-3-203 ~]$ ^C
[ec2-user@ip-10-0-3-203 ~]$ nano student_logger.py
[ec2-user@ip-10-0-3-203 ~]$ > student_logger.py
[ec2-user@ip-10-0-3-203 ~]$ nano student_logger.py
[ec2-user@ip-10-0-3-203 ~]$ > student_logger.py
[ec2-user@ip-10-0-3-203 ~]$ nano student_logger.py
[ec2-user@ip-10-0-3-203 ~]$ python3 student_logger.py
Inserted: Harrison Adom | Status: 200
Inserted: Fadilatu Suhuyini Mahamud | Status: 200
Inserted: Richmond Nyarko | Status: 200
Inserted: Genevieve Kankam Mensah | Status: 200
Inserted: Clifford Mills | Status: 200
[ec2-user@ip-10-0-3-203 ~]$ nano student_logger.py
[ec2-user@ip-10-0-3-203 ~]$ python3 student_logger.py
Inserted: Harrison Adom | Status: 200
Inserted: Fadilatu Suhuyini Mahamud | Status: 200
Inserted: Richmond Nyarko | Status: 200
Inserted: Genevieve Kankam Mensah | Status: 200
Inserted: Clifford Mills | Status: 200
[ec2-user@ip-10-0-3-203 ~]$
```

i-0502ca591120bdbff (Room9-Dynamo-EC2)

PublicIPs: 54.152.23.128 PrivateIPs: 10.0.3.203

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

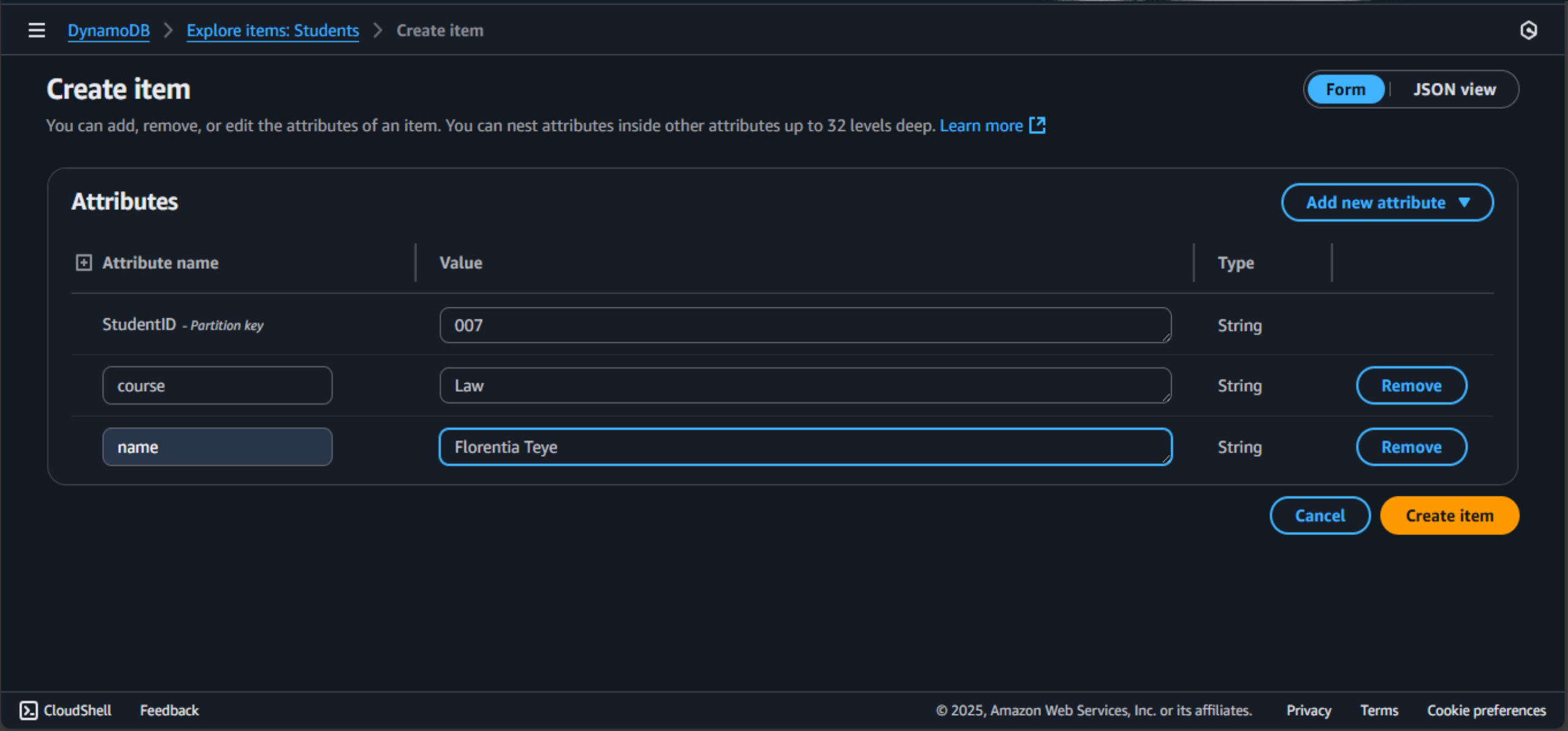
Step20: View Tables in DynamoDB



✓ Use the create item button(circled red) to add more item via the console

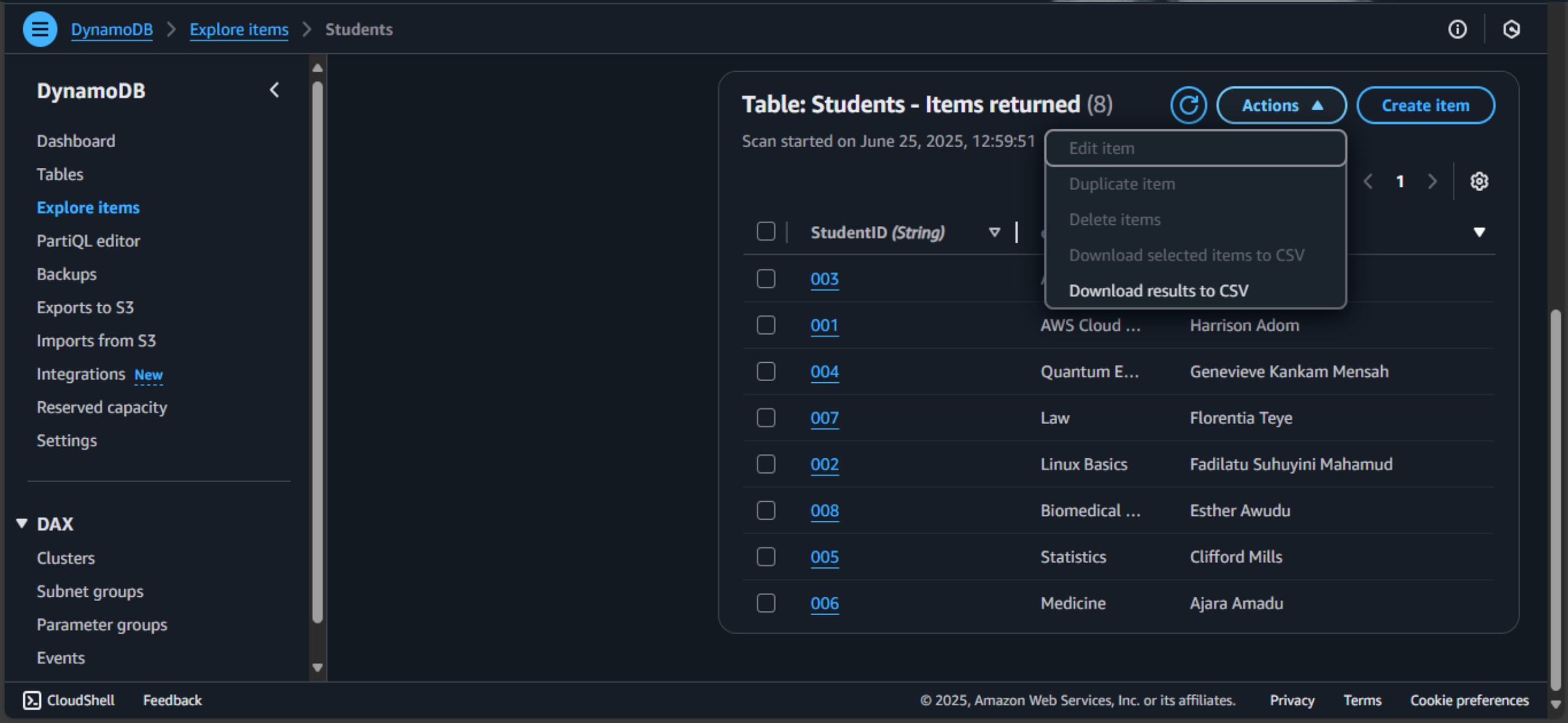
Step21: Add more records via the AWS Console

✓ Manually add new records to the DynamoDB ‘students’ table using the AWS Console by entering attribute names and values through the Items view.



Added records via the AWS Console

✓ NB: Florentia, Ajara, and Esther have been added to the students table.



CONCLUSION

Project summary

- ✓ Architected and deployed a secure, scalable, and highly available web application infrastructure on AWS.
- ✓ Leveraged services such as EC2, VPC, DynamoDB, IAM, Auto Scaling, Load Balancing, and CloudWatch.
- ✓ Simulate a production-grade environment and gain practical experience in designing and securing cloud-native systems.

Challenges

- ✓ A region mismatch between the EC2 instance and the DynamoDB table led to connection failures, which was resolved by recreating the database in the correct region.
- ✓ Additionally, misconfigured inbound and outbound rules in security groups and network ACLs initially blocked essential traffic, requiring careful review and alignment with application needs.
- ✓ CloudWatch alarms failed to trigger as expected due to insufficient CPU stress during testing, highlighting the need for properly calibrated performance simulations.

These issues reinforced the importance of precise configuration, region awareness, and effective monitoring in cloud infrastructure design.

DESIGNING AND MAINTAINING A SCALABLE, WEB APPLICATION IN AWS