



COS20019

Cloud Computing Architecture

Week 10 – ACA Module 10:
Challenge Lab – Creating a Scalable and Highly Available
Environment for the Cafe

Truong Ngoc Gia Hieu
105565520

Challenge Lab: Creating a Scalable and Highly Available Environment for the Café

A. Scenario

The café will soon be featured on a famous TV food show. When the show airs, Sofia and Nikhil anticipate that the café's web server will experience a temporary spike in the number of users—perhaps even up to tens of thousands of users. Currently, the café's web server is deployed in one Availability Zone, and Sofia and Nikhil are worried that the website won't be able to handle the expected increase in traffic. They want to ensure that their customers have a great experience when they visit the website and that they don't experience any issues, such as lags or delays in placing orders.

To ensure this experience, the website must be responsive, able to scale both up and down to meet fluctuating customer demand, and be highly available. Instead of overloading a single server, the architecture must distribute customer order requests across multiple application servers so it can handle the increase in demand.

In this lab, you take on the role of Sofia to implement a scalable and highly available architecture for the café's web application.

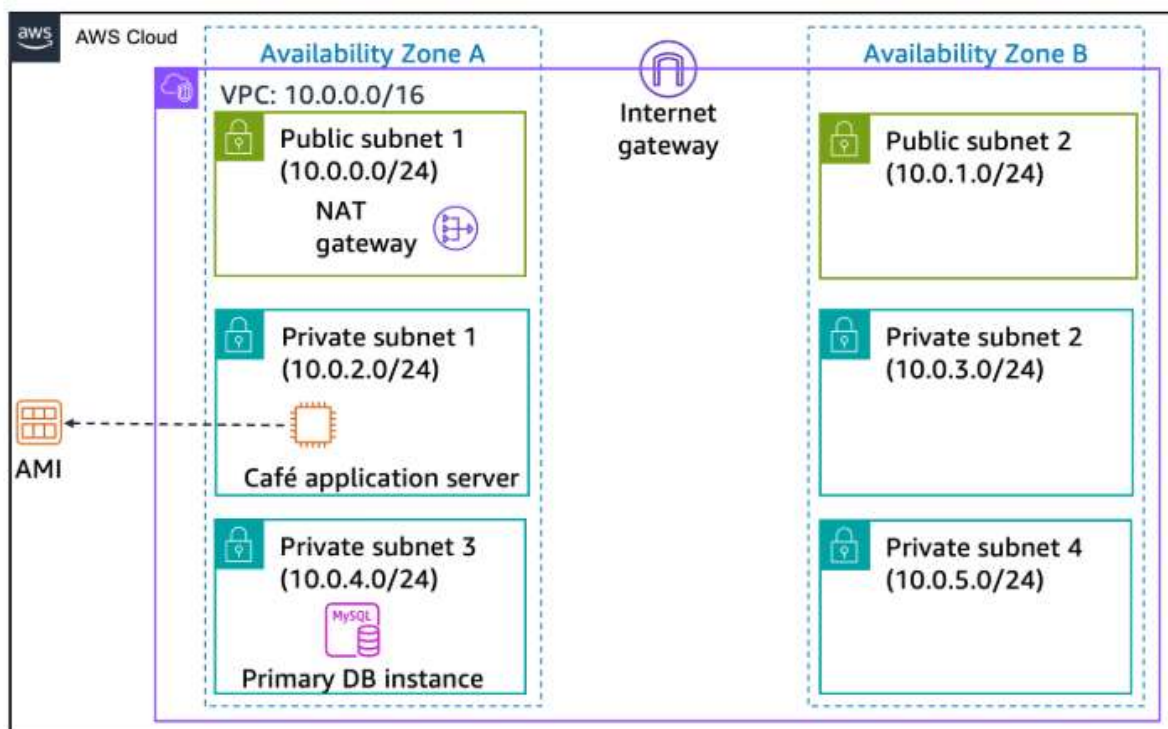
B. Lab overview and objectives

In this lab, you use Elastic Load Balancing (ELB) and Amazon EC2 Auto Scaling to create a scalable and highly available environment on Amazon Web Services (AWS).

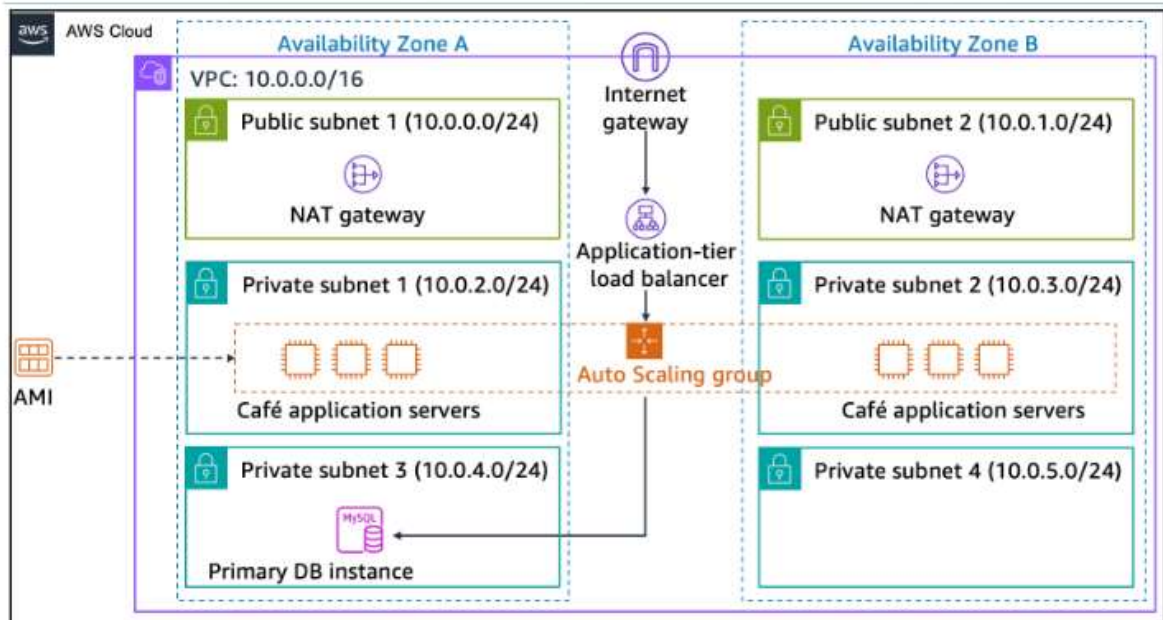
After completing this lab, you should be able to do the following:

- Inspect a virtual private cloud (VPC).
- Update a network to work across multiple Availability Zones.
- Create an Application Load Balancer.
- Create a launch template.
- Create an Auto Scaling group.
- Set an alarm on a system metric to initiate auto scaling.
- Test load balancing and automatic scaling.

When you start the lab, your architecture will look like the following example:



At the end of this lab, your architecture should look like the following example:



Note: In this challenge lab, step-by-step instructions are not provided for most tasks. You must figure out how to complete the tasks on your own.

Duration

This lab requires approximately **90 minutes** to complete..

AWS service restrictions

In this lab environment, access to AWS services and service actions might be restricted to the ones that are needed to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond the ones that are described in this lab.

C. Accessing the AWS Management Console

To access the AWS Management, click on the button Start Lab and wait the circle next to the text AWS turns from yellow to green.



Figure 1: AWS Activated

A business request for the café: Implementing a scalable and highly available environment (challenge)

Sofia understands that she must complete some tasks to implement high availability and scalability for the café's web application. However, before changing the café's application architecture, Sofia must evaluate its current state.

In the next several tasks, you work as Sofia to create and configure the resources that you need to implement a scalable and highly available application.

D. Task 1: Inspecting your environment

To answer required questions, click on the button **AWS Details**. Then, choose the **Access the multiple choice questions link** to load following questions and submit after finishing each question

Question 1: Which ports are open in the *CafeSG* security group?

- ☐ Ports 80 and 443
- ☒ Port 80
- ☐ Ports 80, 443, and 3899
- ☐ Ports 22, 80, and 443

Figure 2: Question 1

Question 2: Can you connect from the internet to instances in *Public Subnet 1*?

- ☒ Yes - If the instance has a public IP address, and the security group and network ACL allow it
- ☐ No - The public subnet has no internet gateway
- ☐ No - The public subnet has no NAT gateway configured for it
- ☐ No - The network access control list (network ACL) prevents any inbound traffic from the internet

Figure 3: Question 2

Question 3: Should an instance in *Private Subnet 1* be able to reach the internet?

- ☒ Yes
- ☐ No

Figure 4: Question 3

Question 4: Should an instance in *Private Subnet 2* be able to reach the internet?

- ☐ Yes
- ☒ No

Figure 5: Question 4

Question 5: Can you connect to the *CafeWebAppServer* instance from the internet?

- ☐ Yes
- ☒ No

Submit

Figure 6: Question 5

Question 6: What is the name of the Amazon Machine Image (AMI)?

- ☐ Amazon Linux
- ☐ WebServerAMI
- ☒ Cafe WebServer Image
- ☐ My Amazing Image

Figure 7: Question 6

E. Task 2: Updating a network to work across multiple Availability Zones (Creating a NAT gateway for the second Availability Zone)

Step 2.1: Search and select the NAT gateways service

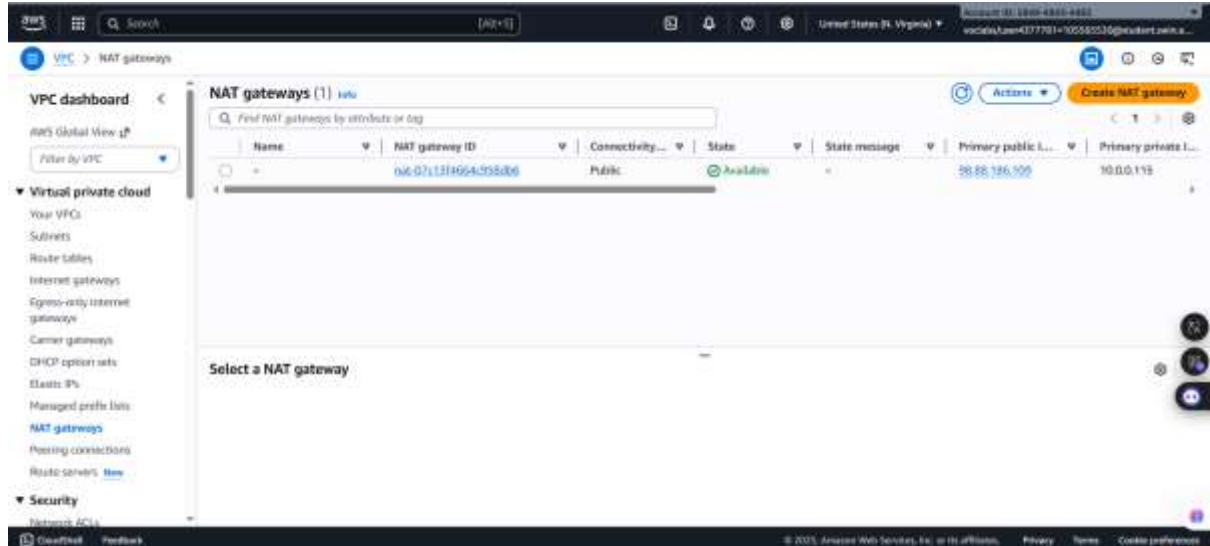


Figure 8: NAT Gateway page

Step 2.2: After clicking button “Create NAT gateway”, configure following settings:

- **Name** – *optional*: lab-natgateway
- **Subnet**: Public Subnet 2
- **Elastic IP allocation ID**: Choose **Allocate Elastic IP** button

Then, click button **Create NAT gateway** at the bottom of the page

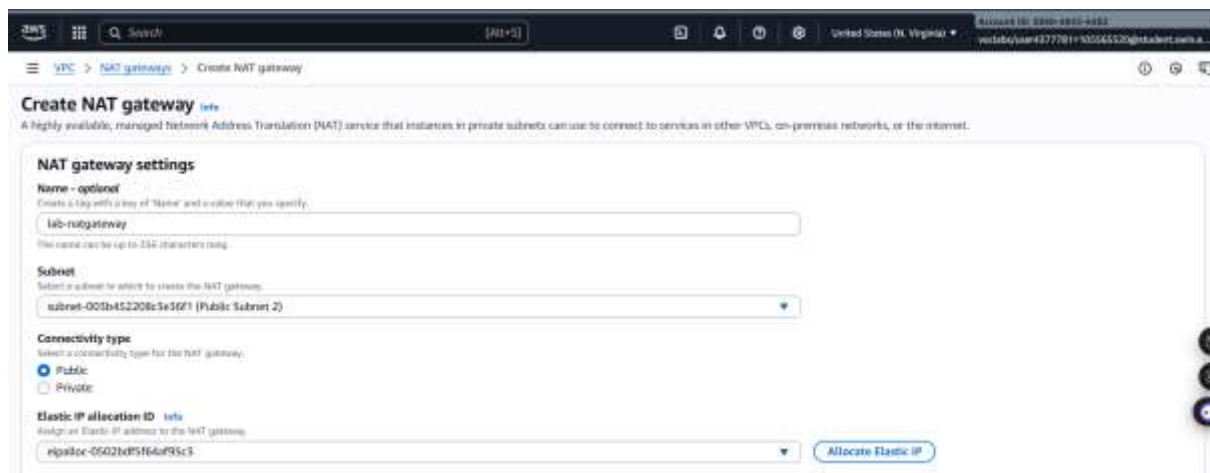


Figure 9: NAT Gateway configuration

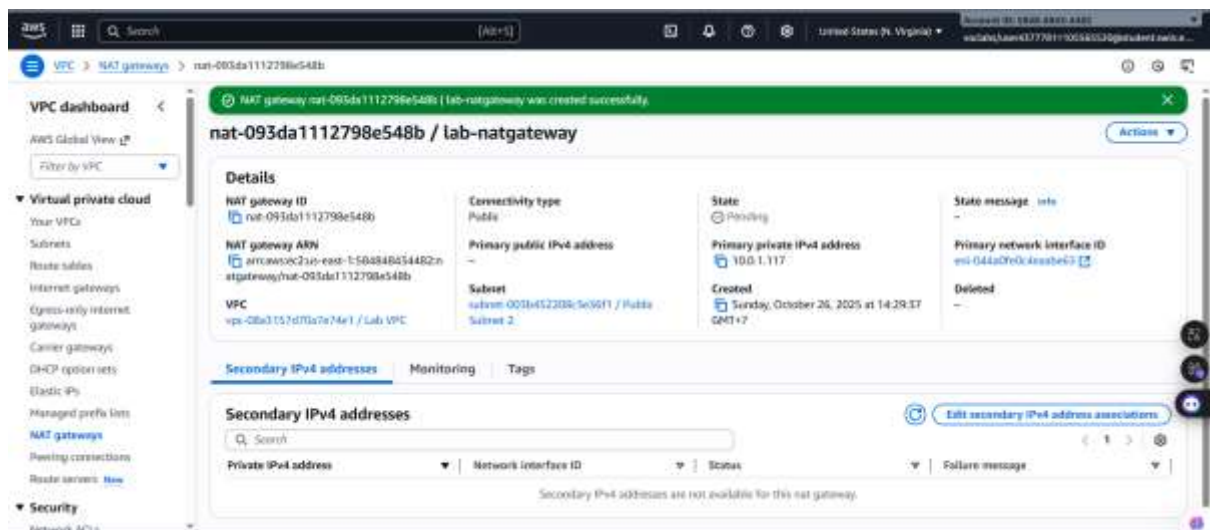


Figure 10: NAT Gateway created successfully

Step 2.3: In the left navigation pane, choose the **Route tables** and tick the box next to the **Private Route Table 2**

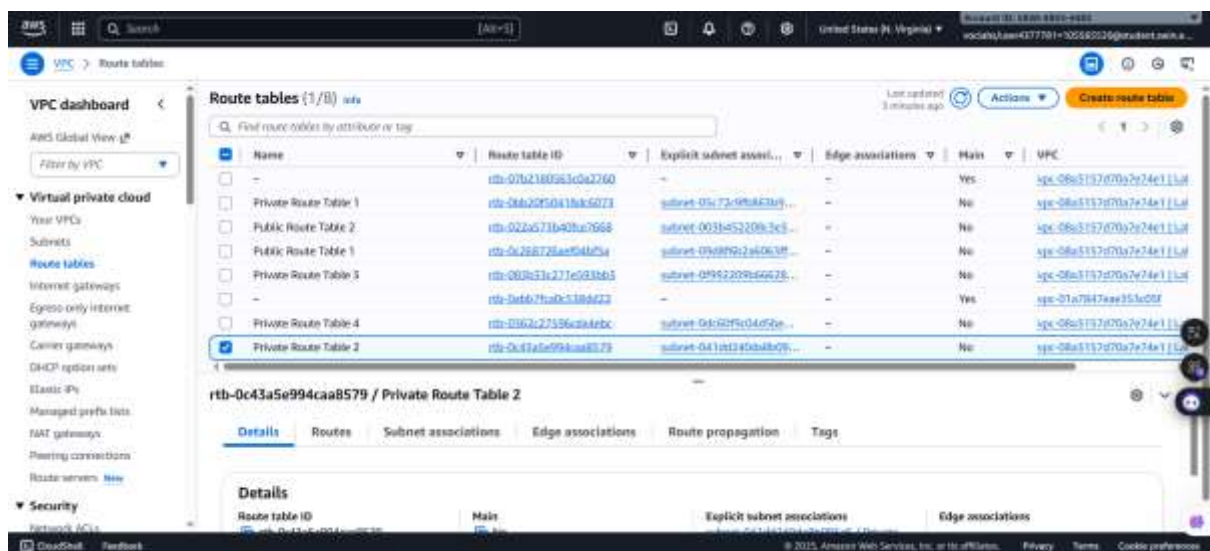


Figure 11: Private Route Table 2 selected

Step 2.4: In the **Routes** tab, choose **Edit routes**, click **Add route**, and then configure below settings:

+ **Destination:** 0.0.0.0/0

+ **Target:** NAT Gateway => lab-natgateway

Then, click **Save changes**

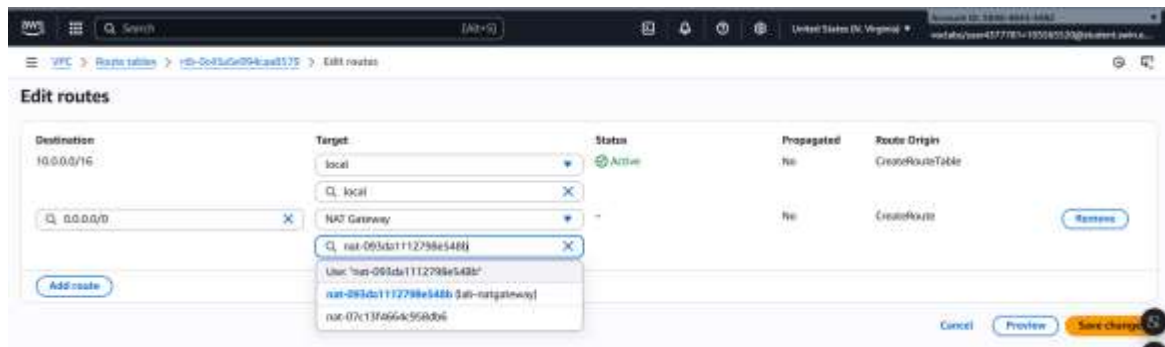


Figure 12: Edit route configuration

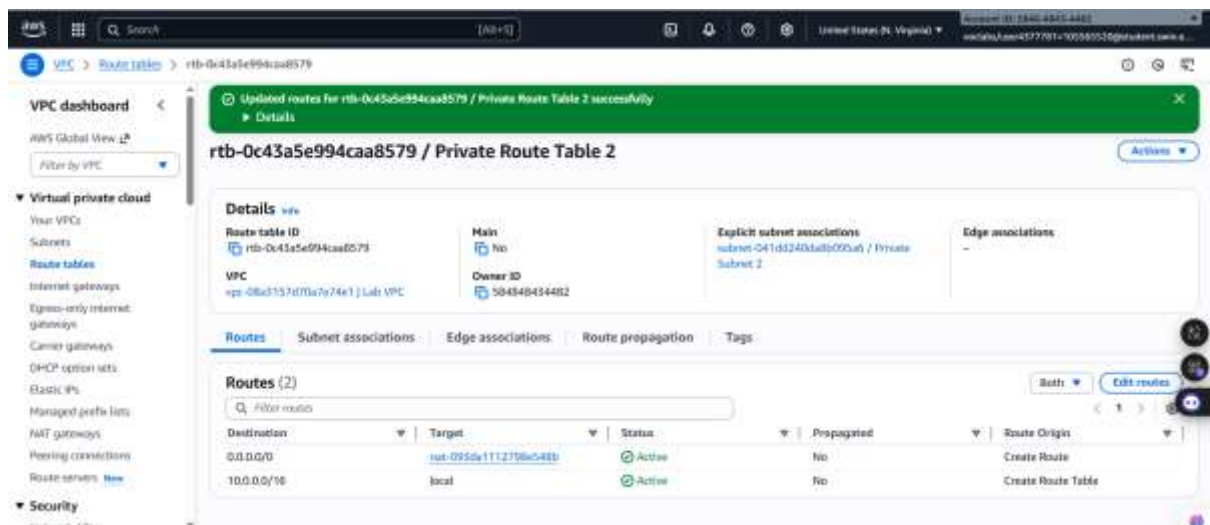


Figure 13: Edit route successfully

F. Task 3: Creating a launch template

During the lab setup, an AMI was created from the CafeWebAppServer instance. In this task, you create a launch template by using this AMI.

Step 3.1: Search and select **EC2**. Next, choose **Launch Templates** in left navigation pane and select button **Create launch template**

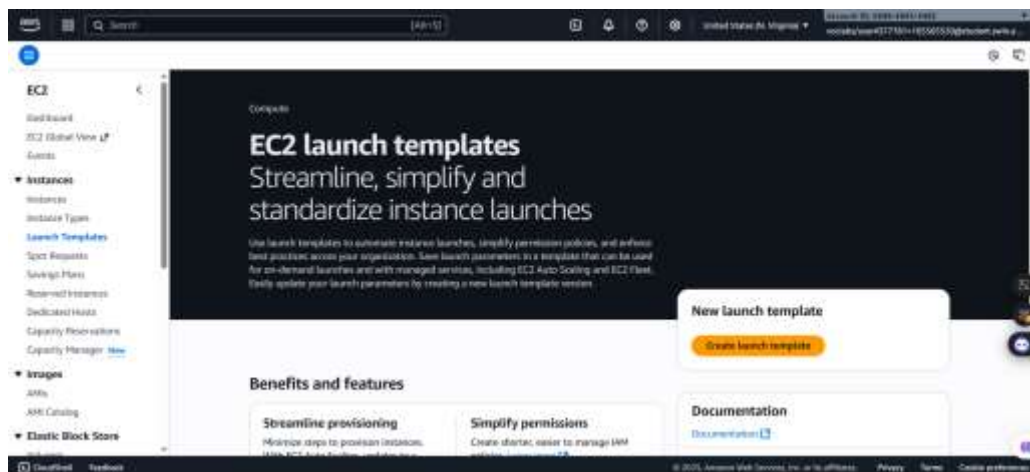


Figure 14: Launch template homepage

Step 3.2: Set the name for the template in the field **Launch template name** – **required** is *cafechallengetemplate*

Launch template name and description

Launch template name - *required*

cafechallengetemplate

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.

Figure 15: Template name configuration

Step 3.3: Application and OS Images (Amazon Machine Image): From My AMIs, choose **Cafe WebServer Image**.

▼ **Application and OS Images (Amazon Machine Image)** [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

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

Recents **My AMIs** Quick Start

☐ Don't include in launch template ☒ Owned by me ☐ Shared with me

[Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Cafe WebServer Image
ami-0b68a456ea5d226e4
2025-10-26T07:05:22.000Z Virtualization: hvm ENA enabled: true Root device type: ebs

Figure 16: AMI Configuration

Step 3.4: Instance type: Choose **t2.micro**.

▼ **Instance type** [Info](#) [Get advice](#) [Advanced](#)

Instance type

t2.micro
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

☒ All generations [Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

Figure 17: Instance configuration

Step 3.5: Key pair (login): Choose **Create new key pair**.



▼ **Key pair (login)** [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

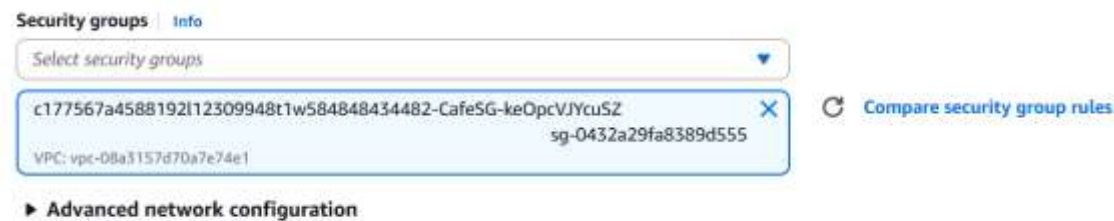
Key pair name

cafechallenge

Create new key pair

Figure 18: Key pair configuration

Step 3.6: Security groups: Enter CafeSG.



Security groups [Info](#)

Select security groups

c177567a4588192112309948t1w584848434482-CafeSG-keOpcVJYcuSZ
sg-0432a29fa8389d555

VPC: vpc-08a3157d70a7e74e1

Compare security group rules

► **Advanced network configuration**

Figure 19: Security group configuration

Step 3.7: Resource tags: Choose **Add new tag**, and configure the following options:

- Key: Enter Name.
- Value: Enter webserver.
- Resource types: Choose Instances.



▼ **Resource tags** [Info](#)

Key	Value	Resource types
Name	webserver	Instances

Add new tag

You can add up to 49 more tags.

Figure 20: Resource Tag configuration

Step 3.8: IAM instance profile: Choose CafeRole.



▼ **Advanced details** [Info](#)

IAM instance profile [Info](#)

CafeRole
arn:aws:iam::584848434482:instance-profile/CafeRole

Create new IAM profile

Hostname type [Info](#)

Figure 21: IAM instance profile configuration

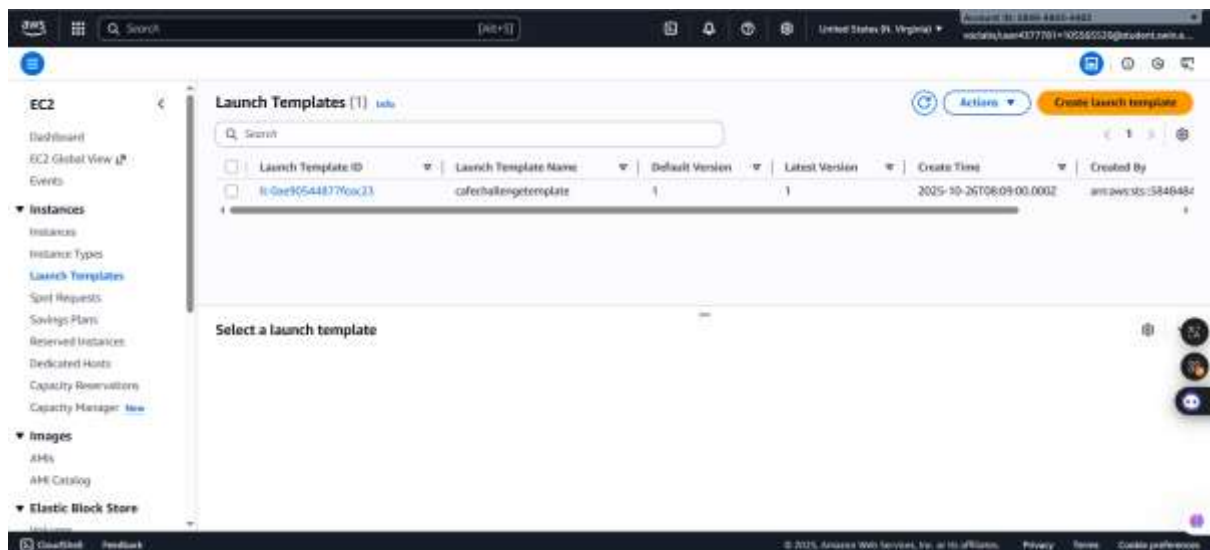


Figure 22: Template created successfully

G. Task 4: Creating an Auto Scaling group

Now that the launch template is defined, you create an Auto Scaling group for the instances. In this task, do not create a load balancer when you create the Auto Scaling group. You create a load balancer in the next task.

While creating an Auto Scaling group, you set a metric (CPU utilization) to initiate auto scaling events.\

Step 4.1: In left navigation pane, select **Auto Scaling Group** and choose button **Create Auto Scaling group**

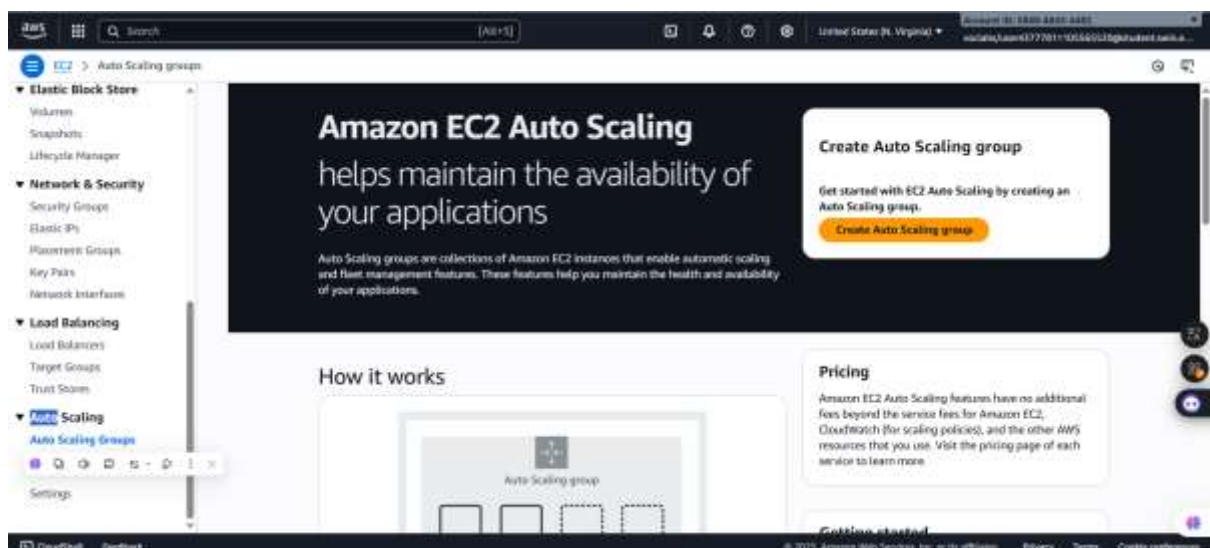


Figure 23: Auto Scaling Group homepage

Step 4.2: Set the **name** and **template** in **Step 1: Choose launch template or configuration**

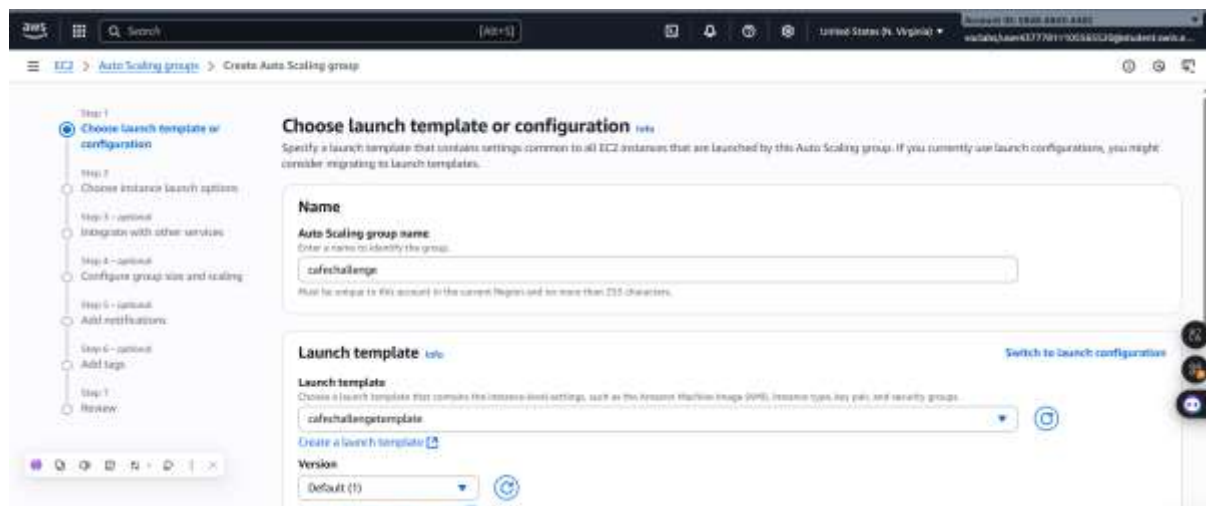
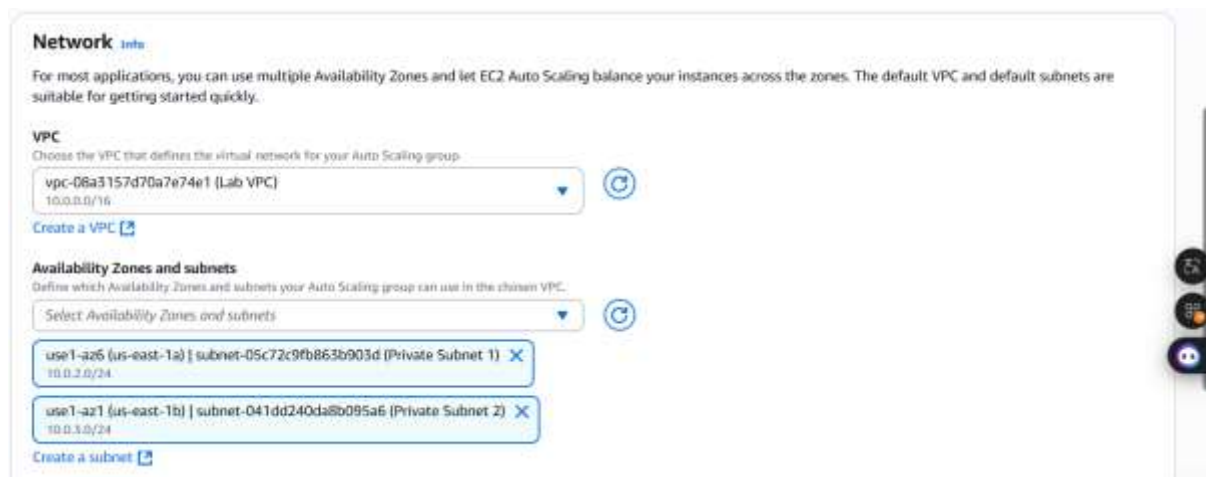


Figure 24: Name and template configuration

Step 4.3: Configure following settings in **Step 2: Create instance launch options**

- * **VPC:** Use the VPC that was configured for this lab.
- * **Availability Zones and subnets:** Choose **Private Subnet 1** and **Private Subnet 2**



2.

Figure 25: Network configuration

Step 4.4: In step 4 – **Optional**, configure:

- **Desired capacity:** Enter 2.
- **Min desired capacity:** Enter 2.
- **Max desired capacity:** Enter 6.

Group size [info](#)
Set the initial size of the Auto Scaling group. After creating the group, you can change its size to meet demand, either manually or by using automatic scaling.

Desired capacity type
Choose the unit of measurement for the desired capacity value. vCPUs and Memory(GiB) are only supported for mixed instances groups configured with a set of instance attributes.

Units (number of instances) ▼

Desired capacity
Specify your group size.

2

Scaling [info](#)
You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits
Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity **Max desired capacity**

2 6

Could not load these desired capacity. Could not load these desired capacity.

Figure 26: Group Size and Scaling configuration

Step 4.5: Automatic scaling - optional: Choose **Target tracking scaling policy**, and configure the following options:

- **Metric type:** Choose Average CPU utilization.
- **Target value:** Enter 25.
- **Instance warmup:** Enter 60.

Automatic scaling - optional
Choose whether to use a target tracking policy [info](#)
You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

☐ No scaling policies
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

☒ Target tracking scaling policy
Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

Scaling policy name

Target Tracking Policy

Metric type [info](#)
Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

Average CPU utilization ▼

Target value

25

Instance warmup [info](#)

60 seconds

Figure 27: Automatic scaling configuration

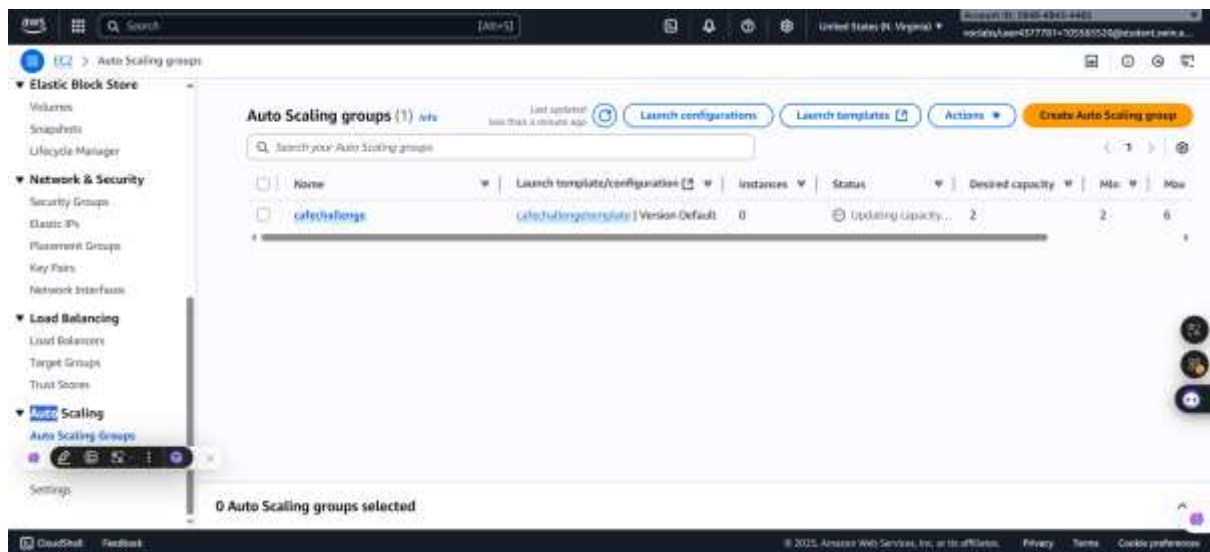


Figure 28: Auto Scaling Group created successfully

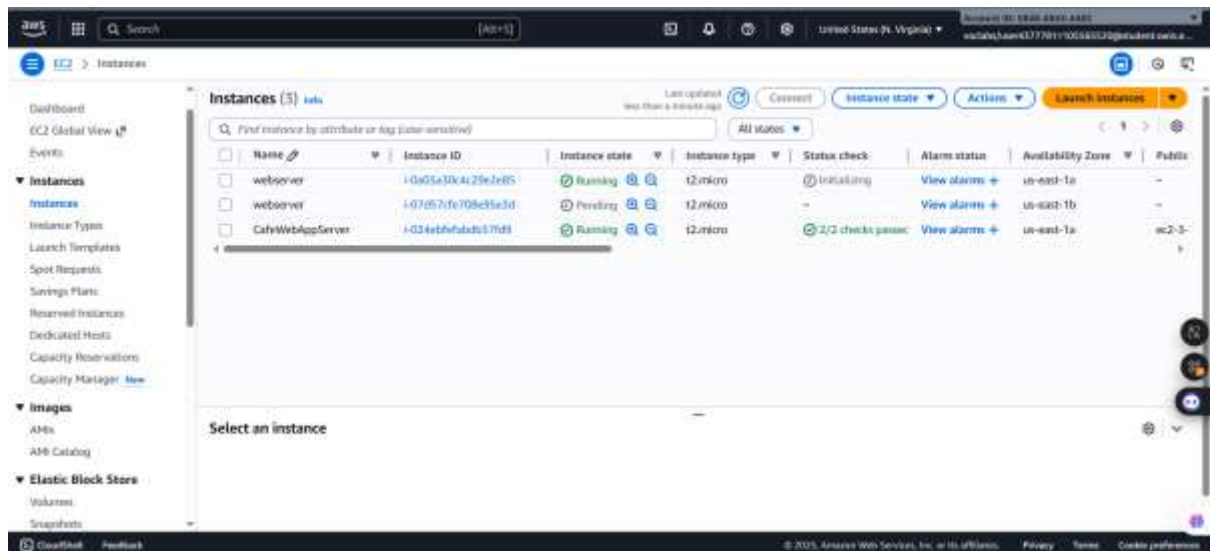


Figure 29: EC2 Confirmation

H. Task 5: Creating a load balancer

Now that your web application server instances are deployed in private subnets, you need a way for the outside world to connect to them. In this task, you create a load balancer to distribute traffic across your private instances.

Step 5.1: In left navigation pane, choose **Load balancer** and select **Create Load Balancer** button. Then, choose button **Create** in Application Load Balancer type

Step 5.2: For **Load balancer name**, type **cafechallenge**



Figure 30: Name configuration

Step 5.3: In Network mapping configuration:

- * **VPC:** Use the VPC configured for this lab.
- * **Subnets:** Use the two public subnets.

Figure 30: Name configuration

Network mapping [info](#)

The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC [info](#)

The load balancer will exist and route within the selected VPC. The selected VPC is also where the load balancer targets must be hosted unless routing to Lambda or on-premises targets, or if using VPC peering. To configure the VPC for your targets, view [target groups](#).

vpc-08a157d70a7e7ae1 (Lab VPC)

[Create VPC](#)

IP pools [info](#)

You can optionally choose to configure an IPAM pool as the preferred source for your load balancer IP addresses. Create or view Pools in the [Amazon VPC IP Address Manager console](#).

☒ Use IPAM pool for public IPv4 addresses

The IPAM pool you choose will be the preferred source of public IPv4 addresses. If the pool is depleted IPv4 addresses will be assigned by AWS.

Availability Zones and subnets [info](#)

Select at least two Availability Zones and a subnet for each zone. A load balancer needs to be placed in each selected zone and will automatically scale in response to traffic. The load balancer routes traffic to targets in the selected Availability Zones only.

☒ us-east-1a (us-east-1)

Subnet

Only IPv4 blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-05cd0f1ab2a60c1190f

IPv4 subnet CIDR: 10.0.0.0/24

Public Subnet 1

☒ us-east-1b (us-east-1)

Subnet

Only IPv4 blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-001ba5c220ba3c56f1

IPv4 subnet CIDR: 10.0.1.0/24

Public Subnet 2

*Figure 31: Network mapping configuration***Step 5.4: Security groups:** Create a new security group that allows HTTP traffic from anywhere.

Security groups [info](#)

A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can [create a new security group](#).

Security groups

Select up to 5 security groups.

sg-0a0f0c21e33680813 (cafechallenge)

sg-0a0f0c21e33680813 · VPC: vpc-08a157d70a7e7ae1

*Figure 32: Security group configuration***Step 5.5: Target group:** Create a new target group.

Default action [info](#)

The default action is used if no other rules apply. Choose the default action for traffic on this listener.

Routing action

☒ Forward to target group ☐ Redirect to URL ☐ Return fixed response

Forward to target group [info](#)

Choose a target group and specify routing weight or [create target group](#).

Target group

cafechallenge

Target type: Instance (IPv4) Target addresses: CIDR

HTTP

Weight: 1

Percent: 100%

Figure 33: Target group configuration

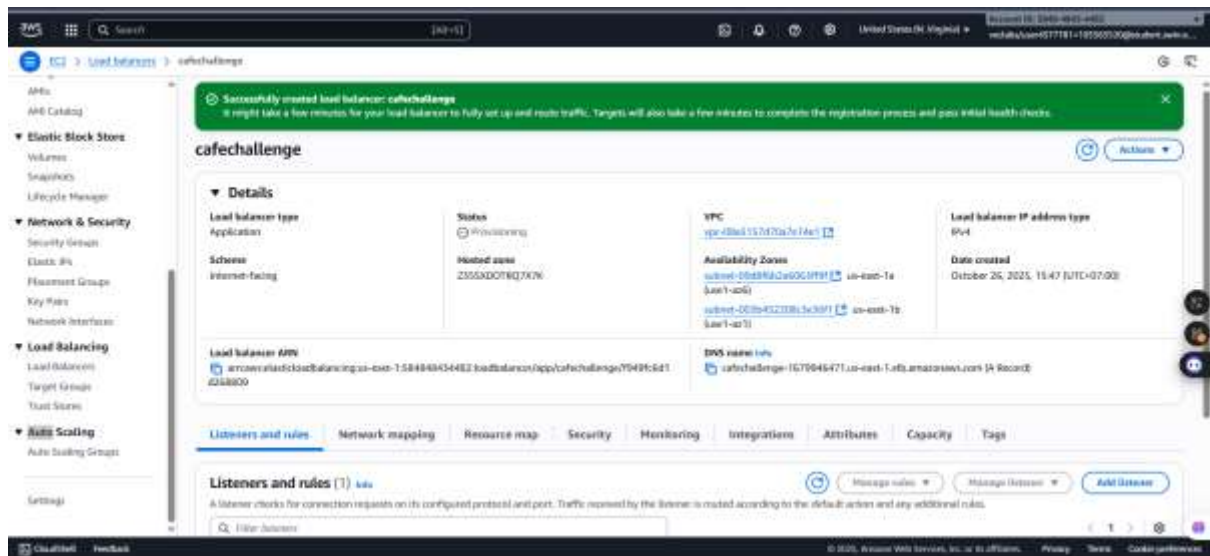


Figure 34: Created successfully

I. Task 6: Testing load balancing and automatic scaling

I.I Task 6.1: Testing the web application without load

Step 6.1: Paste the link <http://cafechallenge-1679946471.us-east-1.elb.amazonaws.com/cafe/> to new browser tab

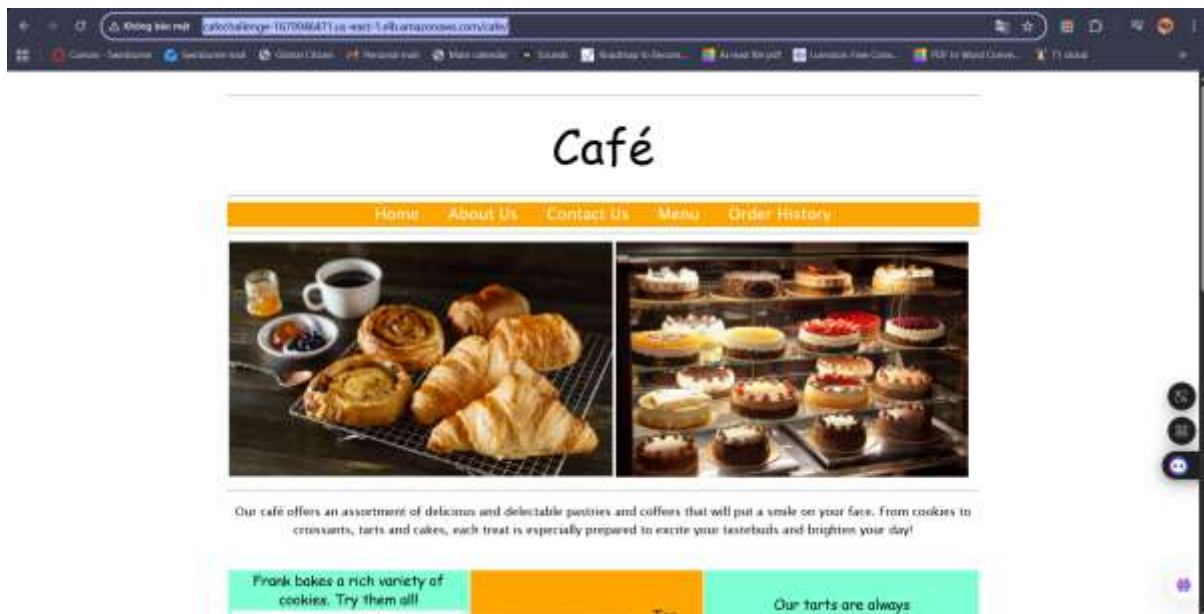


Figure 35: Load successfully

I.II Task 6.2: Testing automatic scaling under load

Step 6.2: Paste the command lines into EC2 console

```

Dependencies Resolved

Package Arch Version Repository Size
Installing:
stress x86_64 1.0.4-16.el7 rpm1 39 K

Transaction Summary
-----
Install 1 Package

Total download size: 39 K
Installed size: 94 K
Downloading packages:
warning: /var/tmp/rpm-tmp/s6_64/2/rpml/packages/stress-1.0.4-16.el7.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID 352684b1: NOKEY
Public key for stress-1.0.4-16.el7.x86_64.rpm is not installed
stress-1.0.4-16.el7.x86_64.rpm
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-RHEL-7
Importing GPG key from /etc/pki/rpm-gpg/RPM-GPG-KEY-RHEL-7
GPG key: "Fedora EPEL (P) epel@fedoraproject.org"
Fingerprint: 6161 1d7c 445e 04f1 729e 48f6 6a2f 8ea2 352c 8ba5
Package : rpm-release-7-11.noarch (@base/extra-rpml)
From : /etc/pki/rpm-gpg/RPM-GPG-KEY-RHEL-7
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
Installing : stress-1.0.4-16.el7.x86_64
Verifying : stress-1.0.4-16.el7.x86_64

Installed:
stress.x86_64 0:1.0.4-16.el7

Complete!
dnf-4.2.0 stress --cpu 1 --timeout 600
stress: info: 138763 dispatching hogs: 1 cpu 0 kb 0 mb 0 hdd
^C
dnf-4.2.0
  
```

Figure 36: Command lines

J. Completed task

Total score	31/31
[Answer 01]	1/1
[Answer 02]	1/1
[Answer 03]	1/1
[Answer 04]	1/1
[Answer 05]	1/1

Figure 37: Completed task

K. Conclusion

During this challenge lab, I learned how to transform a single-server, single-Availability Zone (AZ) web application architecture into a highly available and scalable one on AWS. I gained hands-on experience by extending the network across multiple AZs through the creation and configuration of a new NAT Gateway and updating its associated route table.

The core learning involved implementing elasticity and high availability by creating an EC2 Launch Template and integrating it with an Auto Scaling group. Finally, I learned how to distribute traffic to the application servers in private subnets by creating an Application Load Balancer (ALB), linking it to the Auto Scaling group via a target group, and successfully validating the system's ability to scale based on CPU utilization to handle a massive traffic spike.