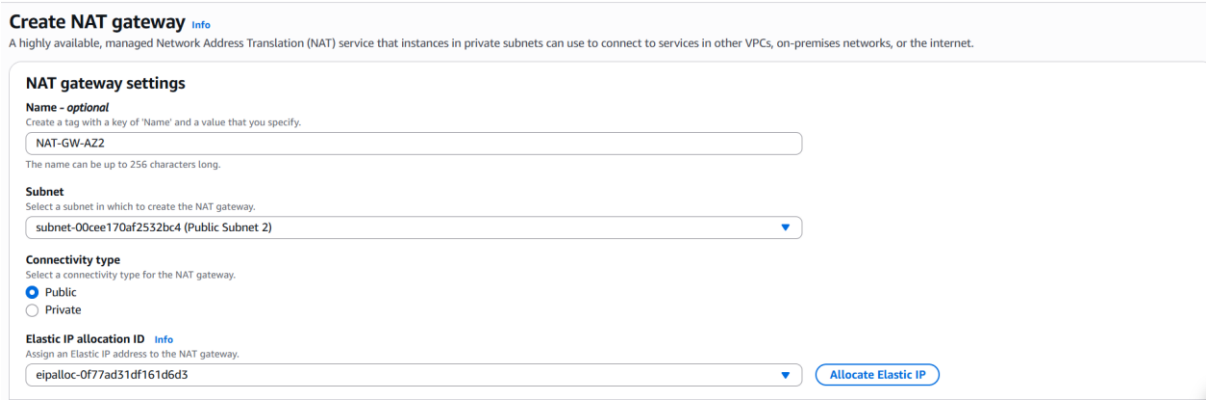
Creating a Scalable and Highly Available Environment for the Café

TASK 1 — Update Network for Multi-AZ High Availability

Task 1.1 — Create NAT Gateway in Public Subnet 2

Steps

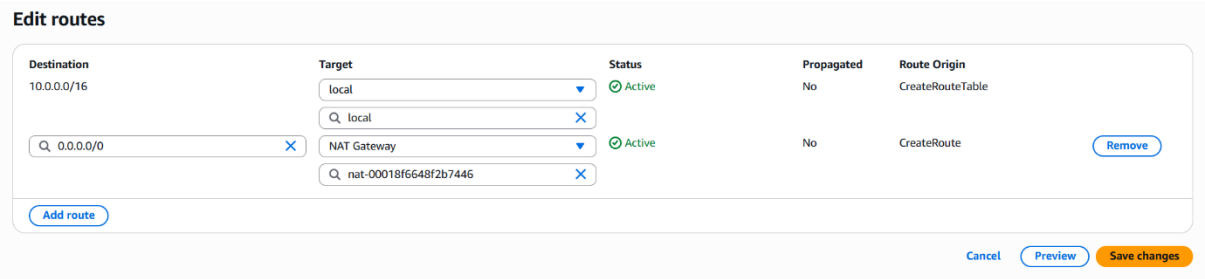
1. Open **VPC Console**.
2. Left menu → **NAT Gateways**.
3. Click **Create NAT Gateway**.
4. Configure:
    - **Subnet:** Public Subnet 2
    - **Elastic IP:** Allocate Elastic IP
    - **Name:** NAT-GW-AZ2
5. Click **Create NAT Gateway**.



Update Private Subnet 2 Route Table

1. VPC console → **Route Tables**.
2. Find the route table for **Private Subnet 2**
3. Select it → **Routes** tab → **Edit routes**.
4. Add route:
    - **Destination:** 0.0.0.0/0
    - **Target:** NAT Gateway → NAT-GW-AZ2

5. Click **Save changes**.



TASK 2 — Create a Launch Template

1. Open Launch Templates

EC2 Console → left menu → **Launch Templates** → **Create launch template**

2. Template Information

- Name: CafeWebServerTemplate

3. AMI

- Application & OS Images → **My AMIs**
- Select: **Cafe WebServer Image**

4. Instance Type

- Choose: **t2.micro**

5. Key Pair

- vokey

6. Network Settings

- Security Group: **CafeSG**

7. Tags

Add tag:

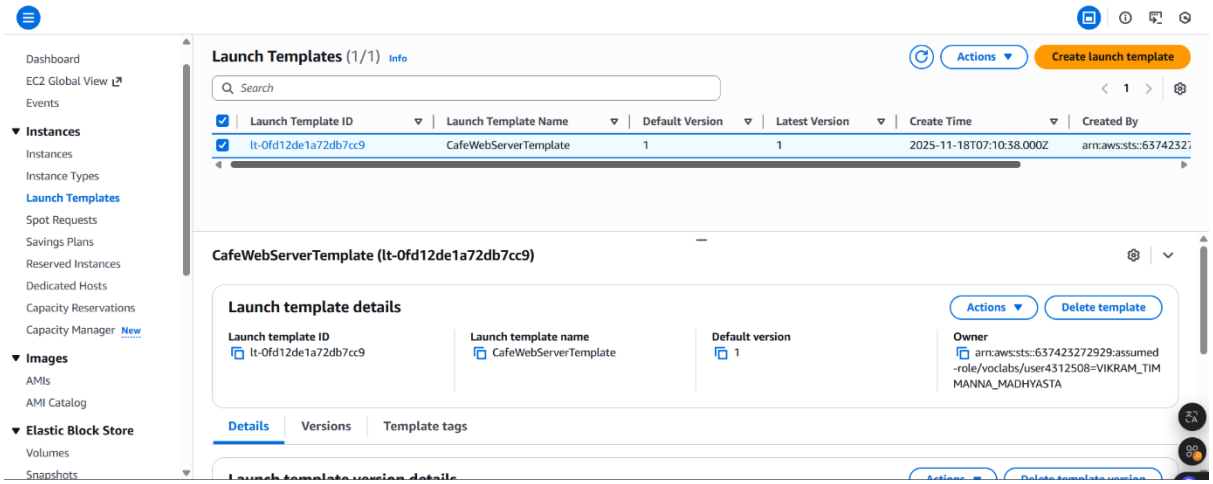
- **Key:** Name
- **Value:** webserver
- **Resource:** Instances

8. IAM Role

- Under Advanced details:
    - **IAM Instance Profile:** CafeRole

9. Create

Click **Create launch template**.



TASK 3 — Create Application Load Balancer

1. Open Load Balancer Console

EC2 → Load Balancers → **Create Load Balancer** → Application Load Balancer

2. ALB Configuration

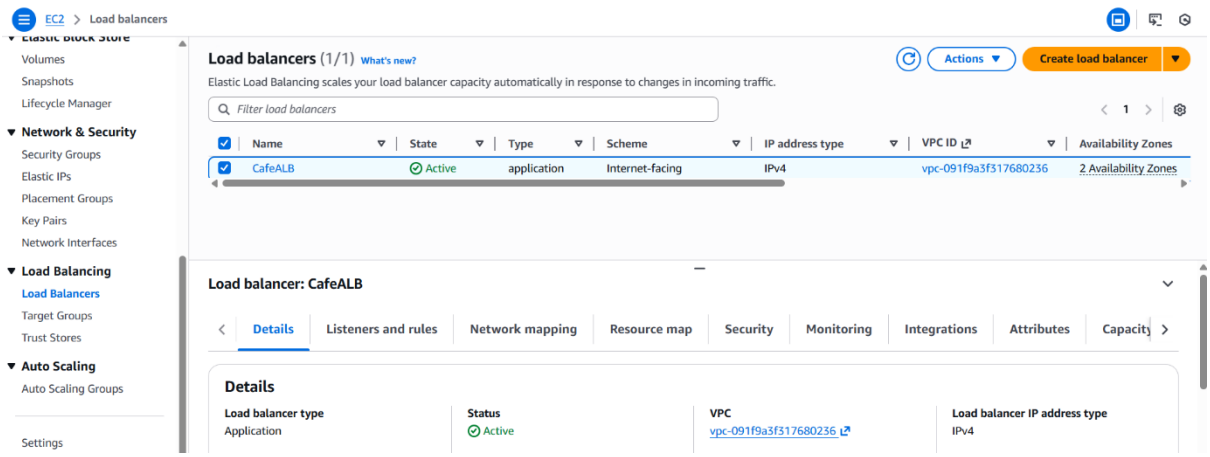
- Name: CafeALB
- Scheme: **Internet-facing**
- IP type: IPv4
- VPC: *lab VPC*
- Subnets:
  Public Subnet 1
  Public Subnet 2
- Select security group

3. Create Target Group

- Target type: **Instances**
- Name: CafeTargetGroup
- Protocol: **HTTP**
- Health check path: /cafe
- Click **Create target group**

4.Create

Create Load Balancer



TASK 4 — Create Auto Scaling Group

1. Open ASG Console

EC2 → **Auto Scaling Groups** → **Create Auto Scaling group**

2. Basic Details

- ASG name: CafeWebServerASG
- Launch template: CafeWebServerTemplate

3. Network

- VPC: *your lab VPC*
- Subnets (select both):
  Private Subnet 1
  Private Subnet 2

4. Group Size

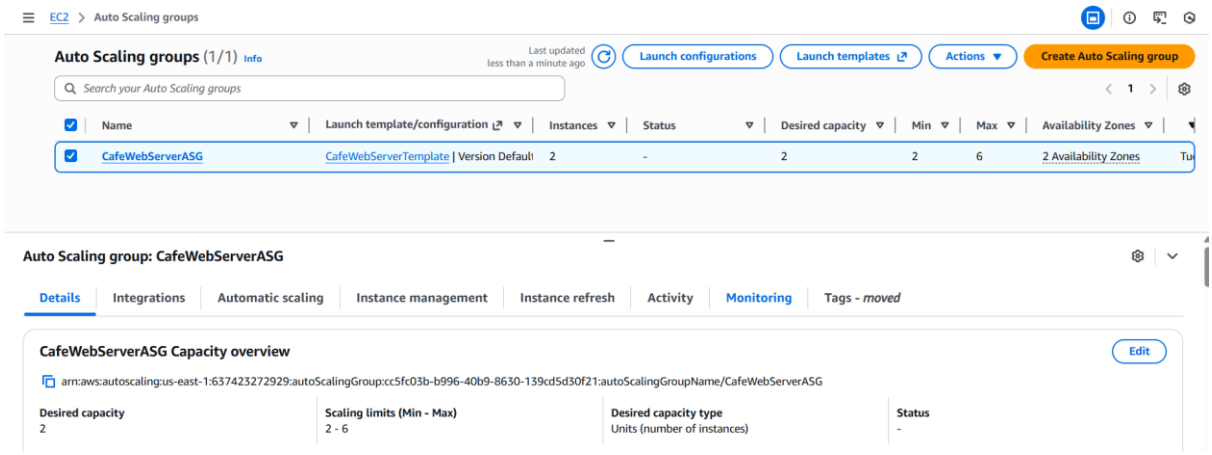
- **Desired:** 2
- **Min:** 2
- **Max:** 6

5. Scaling Policy

- Select **Target tracking scaling policy**
    - Metric: **Average CPU Utilization**
    - Target: **25%**
    - Instance warmup: **60 sec**

6. Create ASG

Click **Create Auto Scaling group**.



Task 5 — Testing and Validation

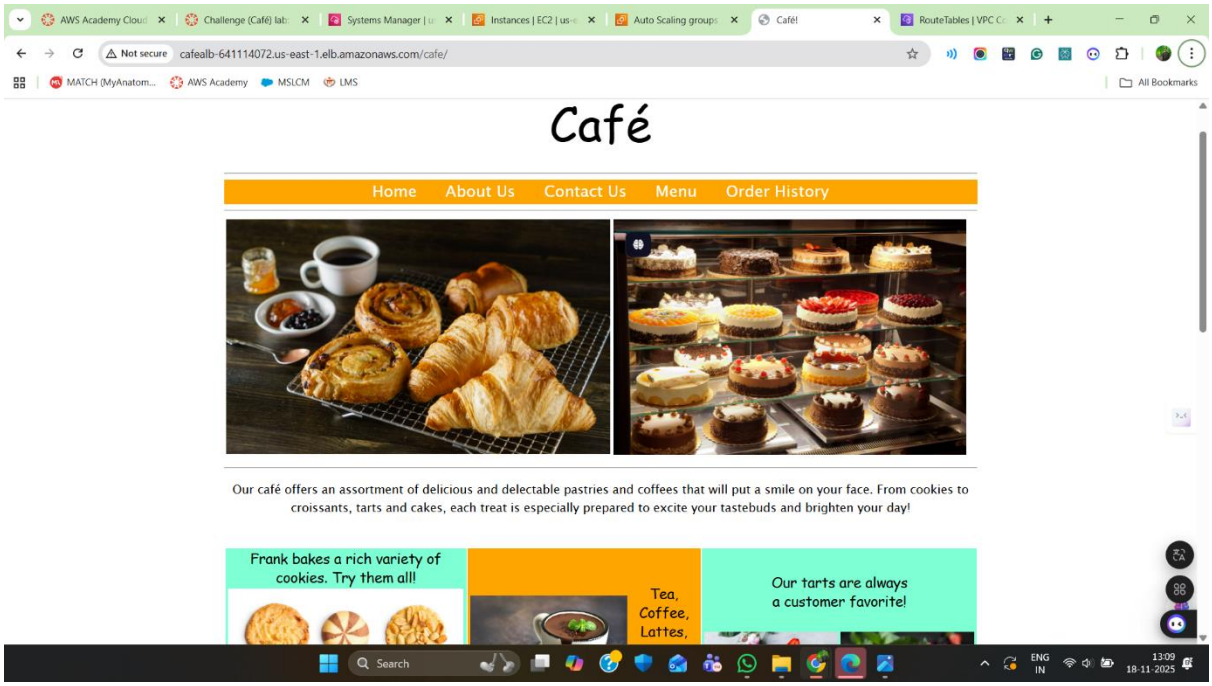
5.1 Application Reachability Test

The ALB DNS name was used to verify that the café application was accessible:

http://cafealb-641114072.us-east-1.elb.amazonaws.com/cafe/

This confirmed:

- Instances were healthy
- Target group was functioning
- ALB routing was operational



5.2 Auto Scaling Stress Test

To validate autoscaling behavior, a stress test was performed using **AWS Systems Manager Session Manager**.

Commands to Run:

sudo amazon-linux-extras install epel

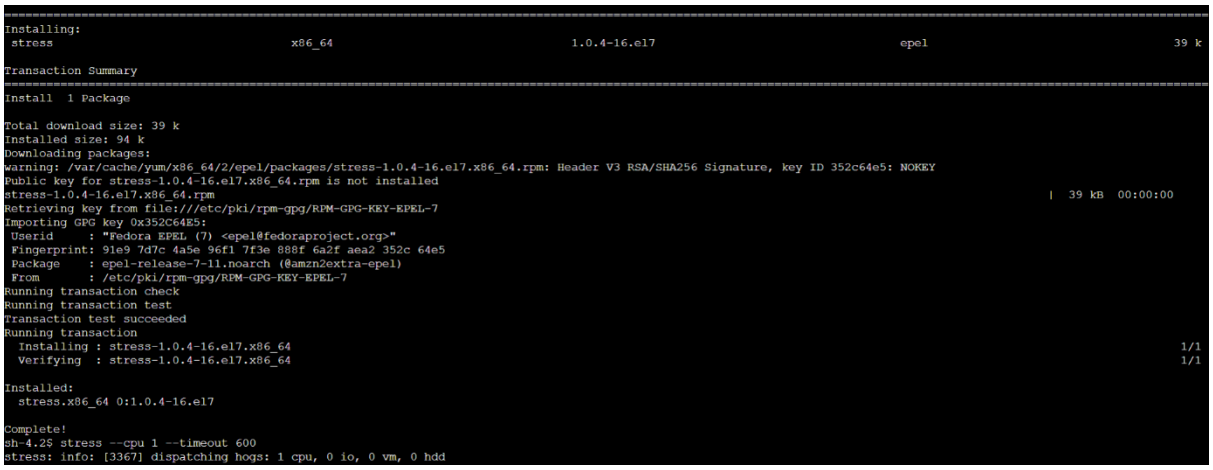
sudo yum install stress -y

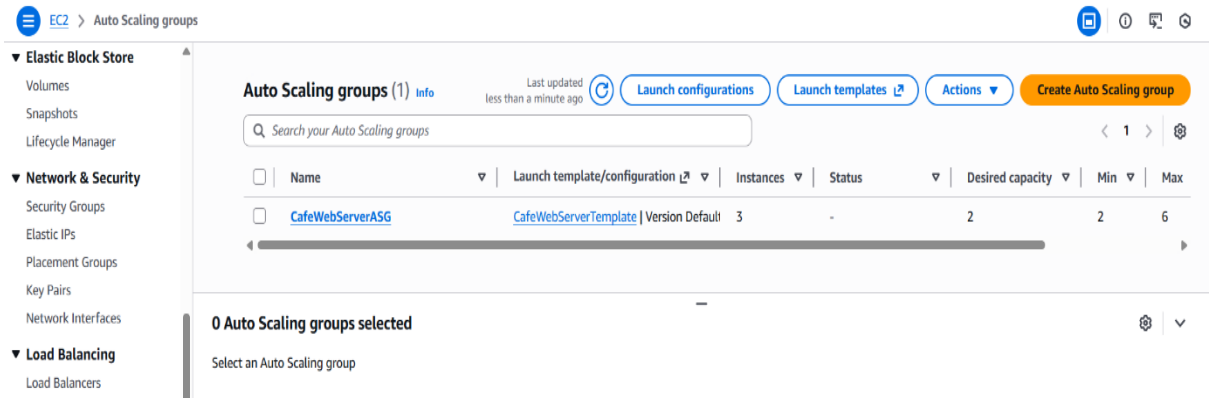
stress --cpu 1 --timeout 600

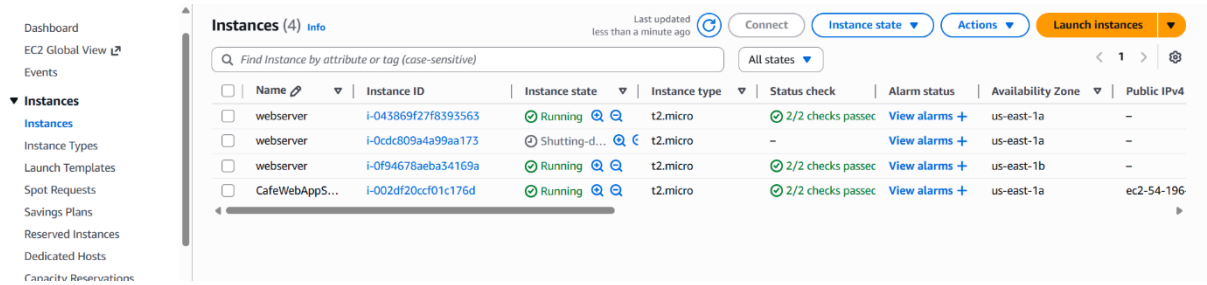
This generated artificial CPU load.
The ASG detected CPU >25% and initiated **scale-out**, launching additional EC2 instances.

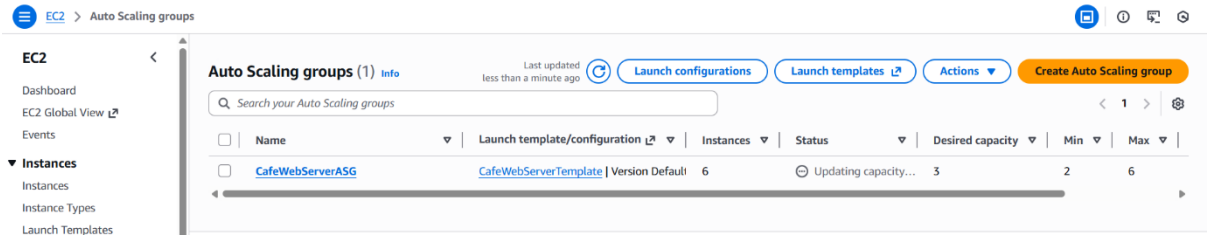
Monitoring the ASG dashboard confirmed:

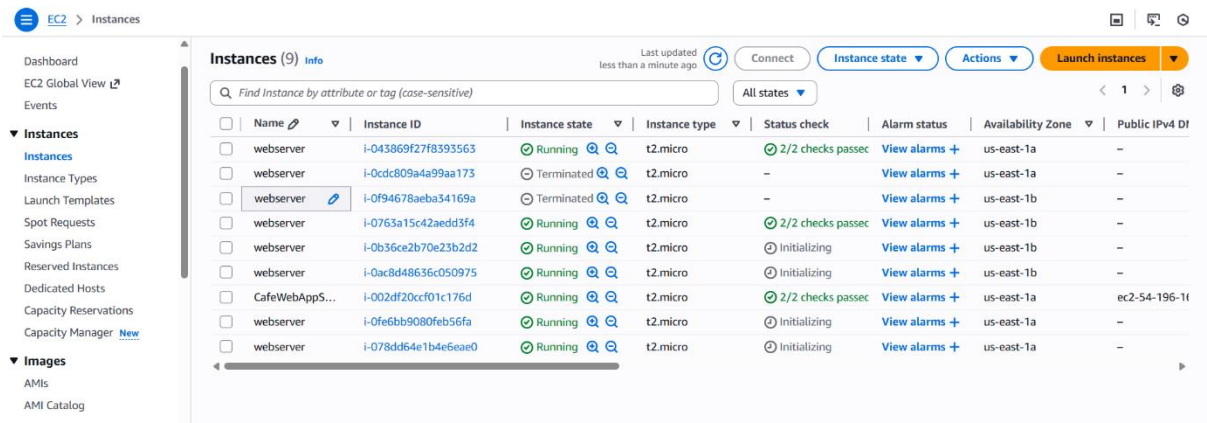
- Scaling policies were applied correctly
- New instances launched in both private subnets
- Load gradually balanced across instances











Conclusion

This lab successfully demonstrated the deployment of a highly available, fault tolerant, and scalable web application architecture using core AWS services. A multi-AZ design combined with NAT Gateways, private subnets, an ALB, and an Auto Scaling Group ensures robust performance and resiliency.