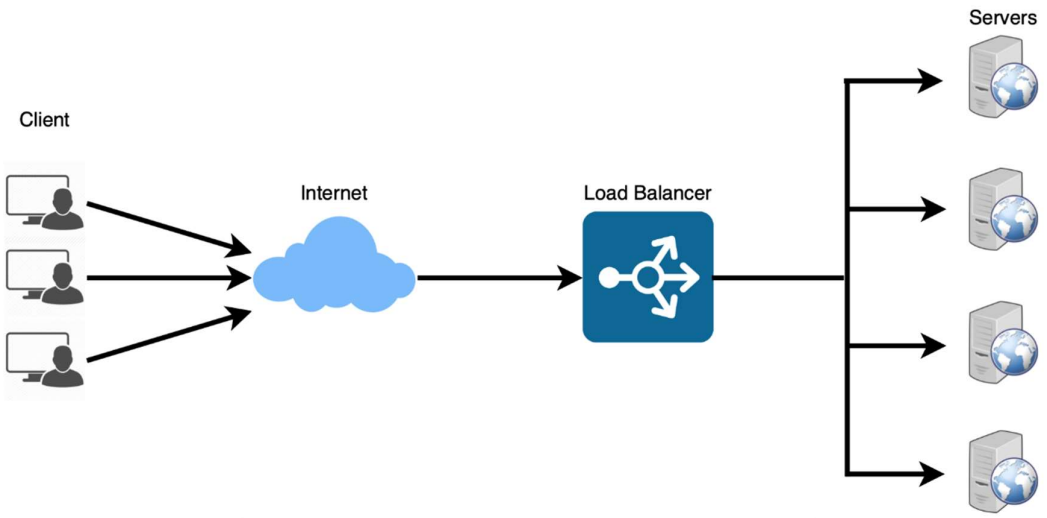
Elastic Load Balancing:

- Load balancing is the process of distributing workload evenly across multiple servers.
- This helps spread the traffic across a cluster of servers to enhance applications, websites responsiveness and availability.
- Load Balancers also detect the health of back end resource and do not send traffic to servers, which cannot achieve requests, along these lines load balancers can improve network and application performance by controlling and handling applications and network sessions automatically by using various algorithms.
- Load balancers can help Minimize the risk of denial-of-service attacks in addition to providing simple distributed service to multiple servers, grant legitimate users to access services without interruption, protect against single-point failures and avoid network traffic bottlenecks.
- Therefore, we can say that the LB is first line of defense against DDOS.
- ELB automatically distributes your incoming application traffic across all the EC2 instances that you are running.



Elastic Load Balancing provides four types of load balancers that can be used.

Application Load Balancer:

- Routes and load balances at the application layer (HTTP/HTTPS), and supports path-based routing.
- An Application Load Balancer can route requests to ports on one or more registered targets, such as EC2 instances, in your virtual private cloud (VPC).

Network Load Balancer:

- Routes and load balances at the transport layer (TCP/UDP Layer-4), based on address information extracted from the Layer-4 header.
- Network Load Balancers can handle traffic bursts, retain the source IP of the client, and use a fixed IP for the life of the load balancer.

Gateway Load Balancer:

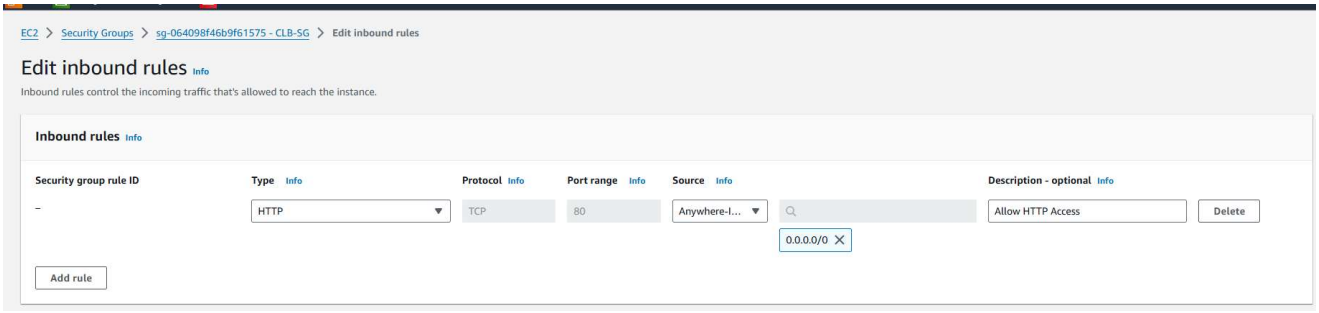
- Gateway Load Balancers work with virtual appliances that support the GENEVE protocol.

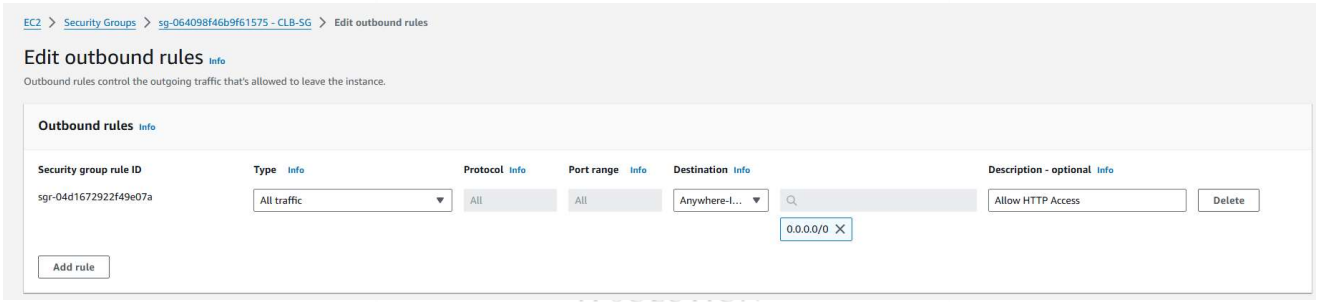
Classic Load Balancer:

- Routes and load balances either at the transport layer (TCP/SSL), or at the application layer (HTTP/HTTPS).

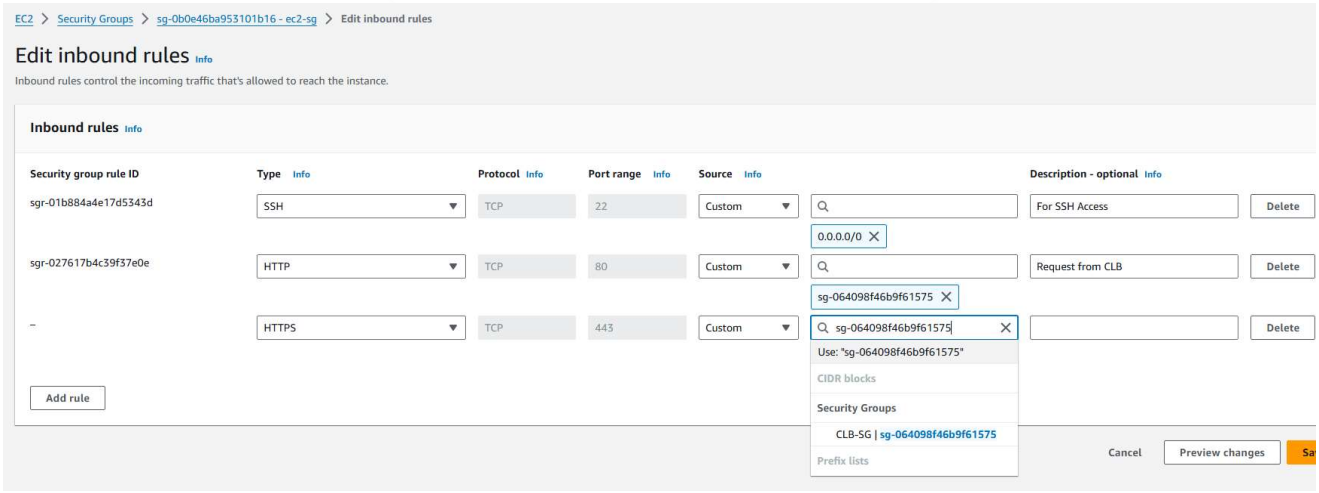
Create a Classic Load Balancer:

1. Login to AWS Console.
2. Go to EC2.
3. Create a security group for CLB. Here we should allow **only HTTP access** under inbound rule and allow **all traffic** under outbound rule.





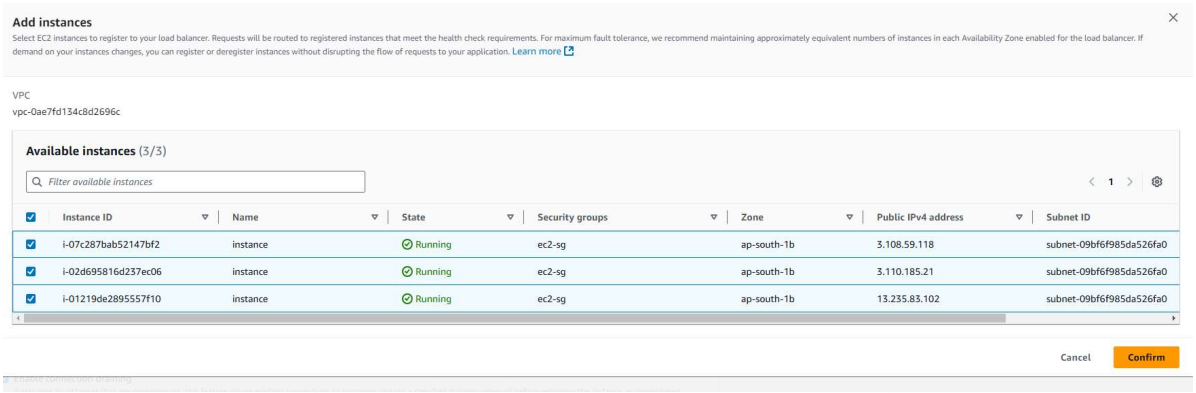
4. Before creating a CLB, let's create 3 EC2 instances, on top of which we will create a CLB.
5. Create a separate Security Group for EC2 instances. For the EC2 instances, we will allow both HTTP and HTTPS access.



6. While launching EC2 instances, add following script under user data section.

```
#!/bin/bash
apt-get update
apt-get install nginx -y
echo "<h1> This is $(hostname) </h1>" > /var/www/html/index.html
```

7. Go to **EC2 Dashboard** and click on **Load Balancers**.
8. Click on **Create Load Balancer**, select **Classic Load Balancer** and click on **Create**.
9. Give a name to Load Balancer.
10. Scheme – Internet-Facing.
11. Select all the default availability zones.
12. Assign the security group that we have created previously for CLB.
13. No changes required under **Listeners and routing** and **Health checks.**
14. Under Instances, add the 3 EC2 instances here.



15. Now click on **Create Load balancer,** Load Balancer will be created.
16. Copy the DNS Name of the CLB, and access in browser.

Delete a Load Balancer:

Select the Load Balancer, Click on Actions and Click on Delete Load balancer. After giving confirmation, it will get deleted.

Auto Scaling:

- AWS Auto Scaling monitors your applications and automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost.
- Using AWS Auto Scaling, it's easy to setup application scaling for multiple resources across multiple services in minutes.
- The service provides a simple, powerful user interface that lets you build scaling plans for AWS resources.
- Create a **Launch Template** before creating **Auto Scaling Group.**

Create a Launch Template:

1. Go to EC2 Dashboard and click on **Launch Templates** present under **Instances**.
2. Click on **Create New launch Template**.

Create launch template

Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.

Launch template name and description

Launch template name - *required*

EC2-Template

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

Template version description

v1

Max 255 chars

Auto Scaling guidance | Info

Select this if you intend to use this template with EC2 Auto Scaling

☐ Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

▶ Template tags

▶ Source template

Launch template contents

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

▼ 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

Don't include in launch template | Amazon Linux | macOS | Ubuntu | Windows | Red H

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible

ami-0dee22c13ea7a9a67 (64-bit (x86)) / ami-0c8eea98010057bd0 (64-bit (Arm))

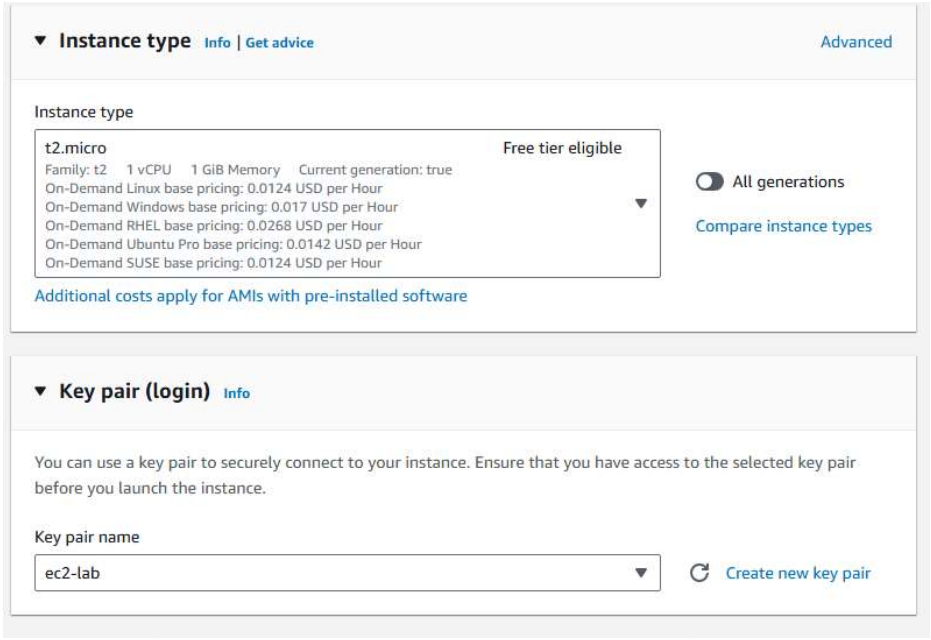
Virtualization: hvm ENA enabled: true Root device type: ebs

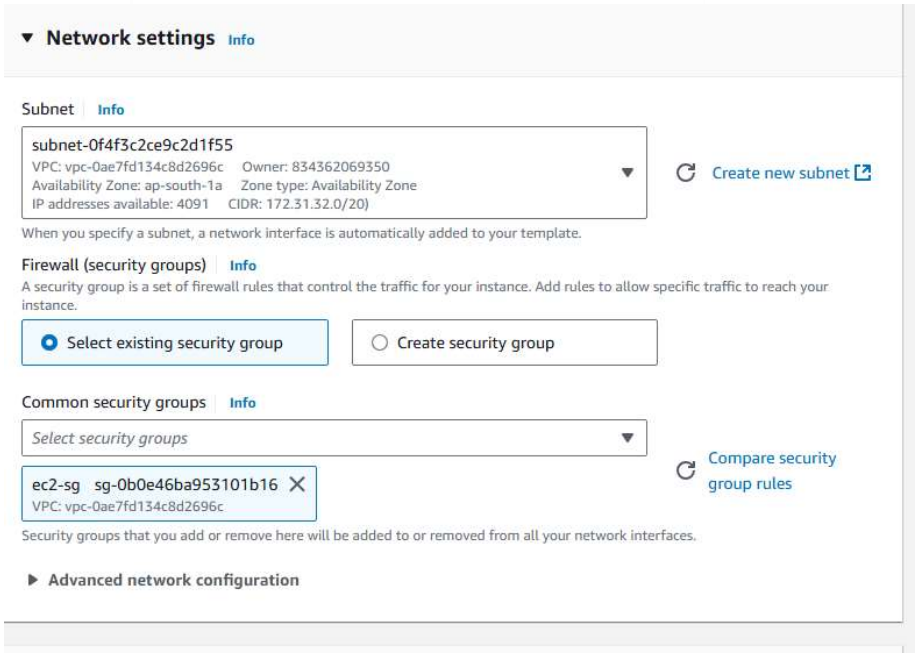
Description

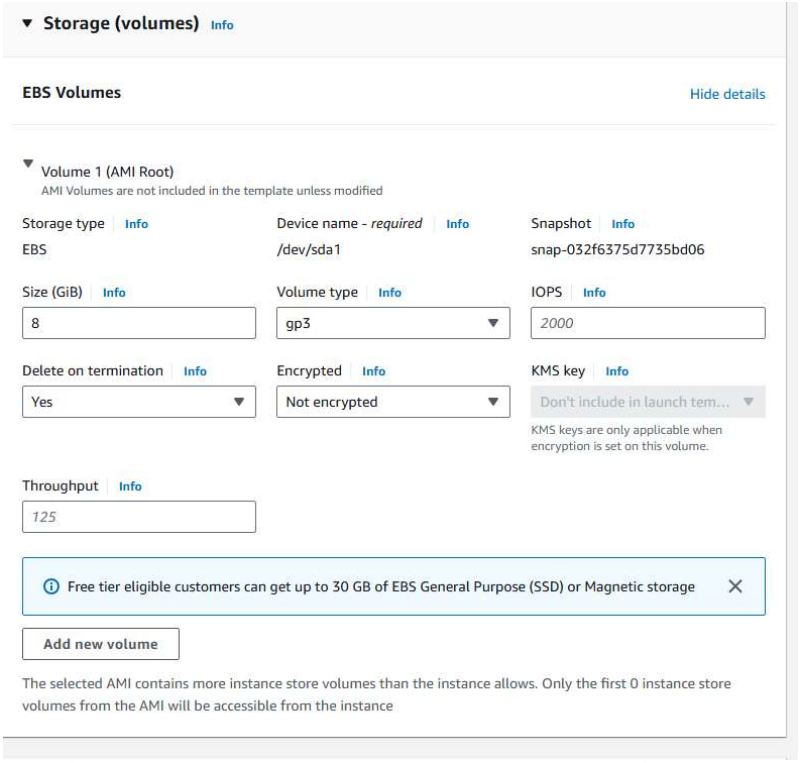
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Canonical, Ubuntu, 24.04, amd64 noble image

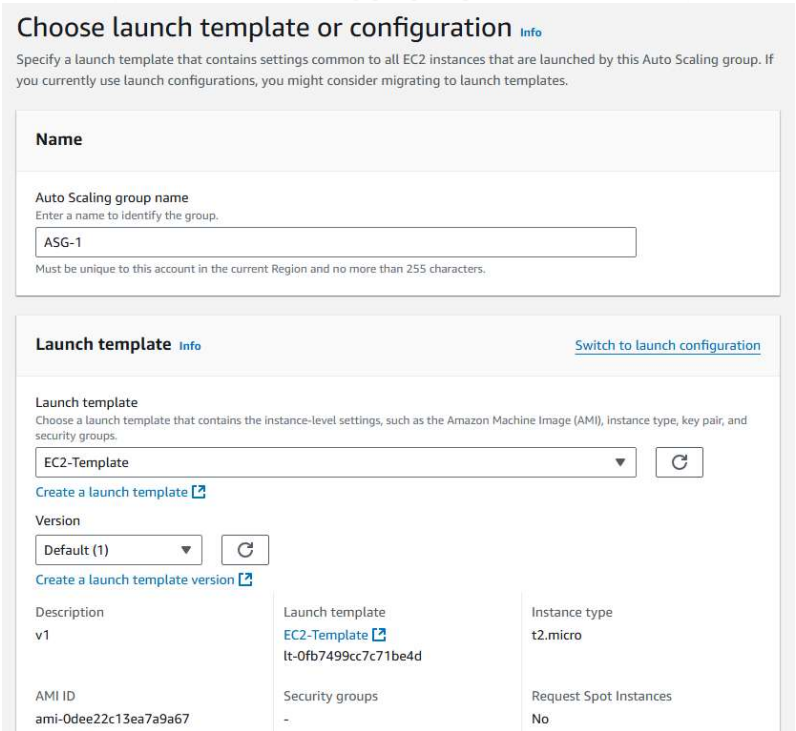
Architecture	AMI ID	Username
64-bit (x86)	ami-0dee22c13ea7a9a67	ubuntu Verified provider


Instance type Info | Get advice
Advanced
Instance type
t2.micro
Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0268 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0142 USD per Hour
On-Demand SUSE base pricing: 0.0124 USD per Hour
All generations
Compare instance types
Additional costs apply for AMIs with pre-installed software
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
ec2-lab
Create new key pair

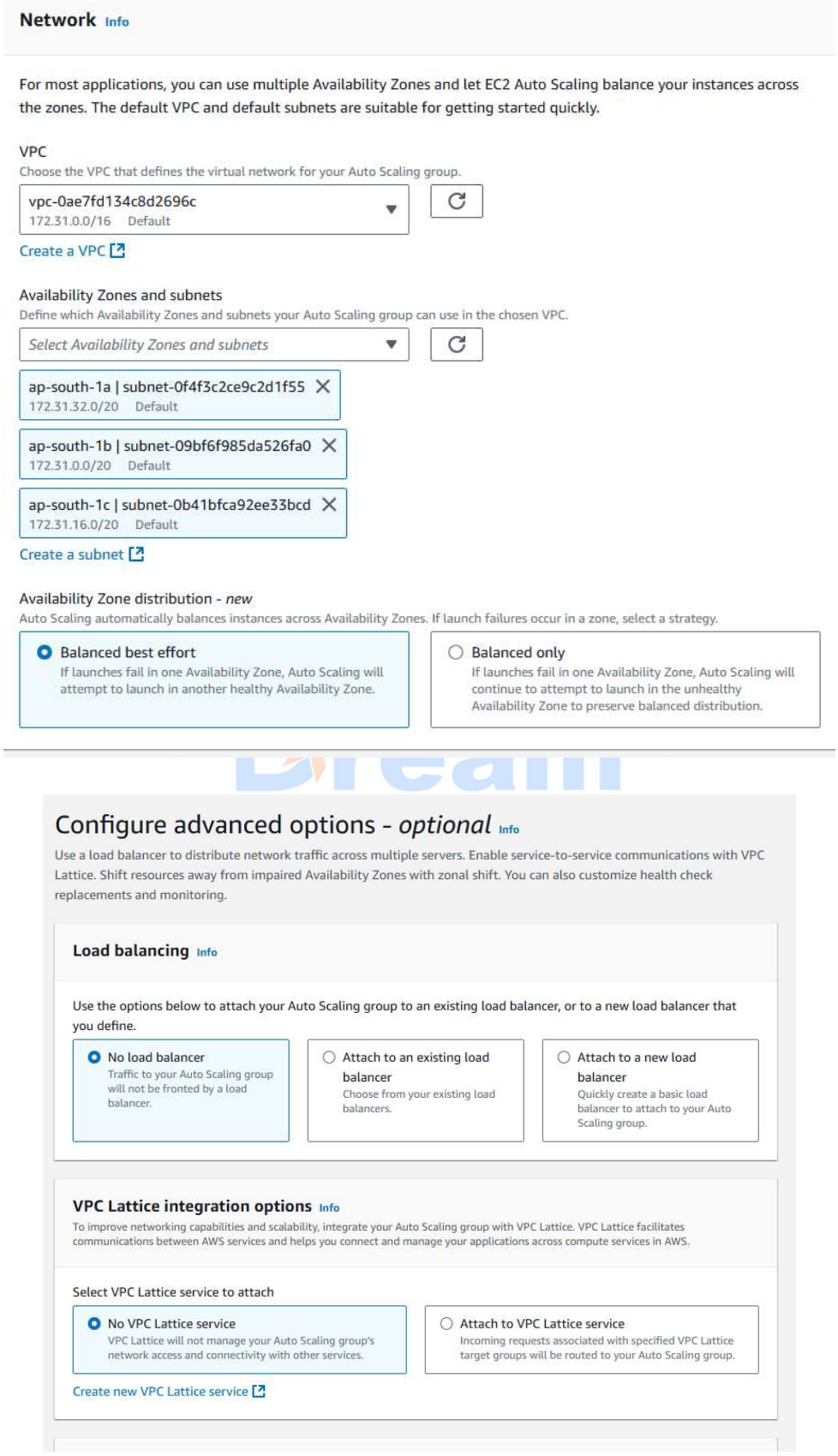

Network settings Info
Subnet Info
subnet-0f4f3c2ce9c2d1f55
VPC: vpc-0ae7fd134c8d2696c Owner: 834362069350
Availability Zone: ap-south-1a Zone type: Availability Zone
IP addresses available: 4091 CIDR: 172.31.32.0/20)
Create new subnet
When you specify a subnet, a network interface is automatically added to your template.
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Select existing security group
Create security group
Common security groups Info
Select security groups
ec2-sg sg-0b0e46ba953101b16
VPC: vpc-0ae7fd134c8d2696c
Compare security group rules
Security groups that you add or remove here will be added to or removed from all your network interfaces.
Advanced network configuration

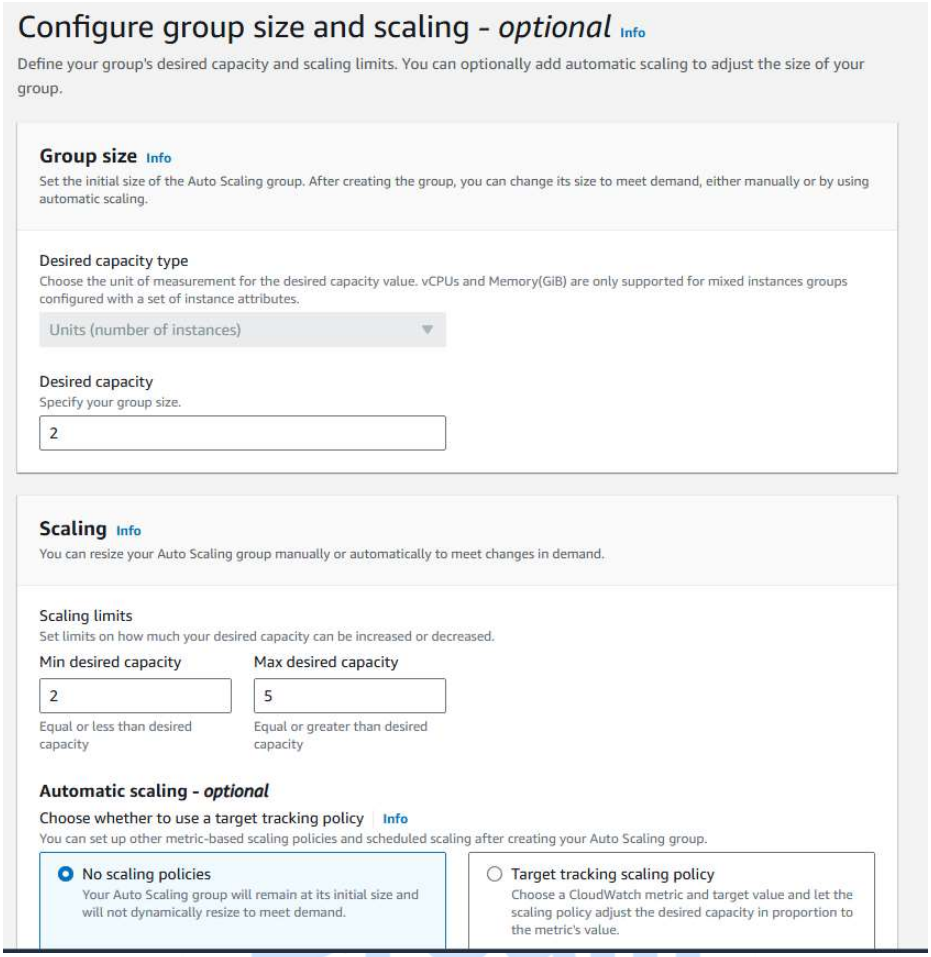


- Now click on create launch template.
- Go to EC2 Dashboard, click on **Auto Scaling Groups** present under **Auto Scaling**.
- Create Auto Scaling Groups.
- Provide a name and select launch template.



On the next screen, select the Availability Zones.





Next, Next, Next and create Auto Scaling Group.
Now you can check, 2 minimum instances must get created.

Delete Auto Scaling Group.
Go to Auto Scaling, Select Auto Scaling Groups, select the auto scaling group, and delete this.
If you delete the Auto Scaling Group, then associated EC2 instances will get deleted.