Project 1
On

Creating Auto Scaling Group With 3 Different Servers
Appache-Server
Appache2-Server
Nginx-Server

Created By

Muzammil Peerzada

Guided By

Mittal Ashish

New EC2 Experience
Tell us what you think

EC2 Dashboard
EC2 Global View
Events
Tags
Limits

▼ Instances
Instances New
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances New
Dedicated Hosts
Scheduled Instances
Capacity Reservations

▼ Images
AMIs New
AMI Catalog

▼ Elastic Block Store

Resources

EC2 Global view

You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:

Instances (running)	0	Dedicated Hosts	0
Elastic IPs	0	Instances	1
Key pairs	3	Load balancers	0
Placement groups	0	Security groups	5
Snapshots	0	Volumes	

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud

Launch instance

Migrate a server

Note: Your instances will launch in the US East (N. Virginia) Region

AWS Health Dashboard

Region
US East (N. Virginia)

Status
This service is operating normally

Account attributes

Supported platforms
- VPC

Default VPC
vpc-0fbae6791ec5b61be

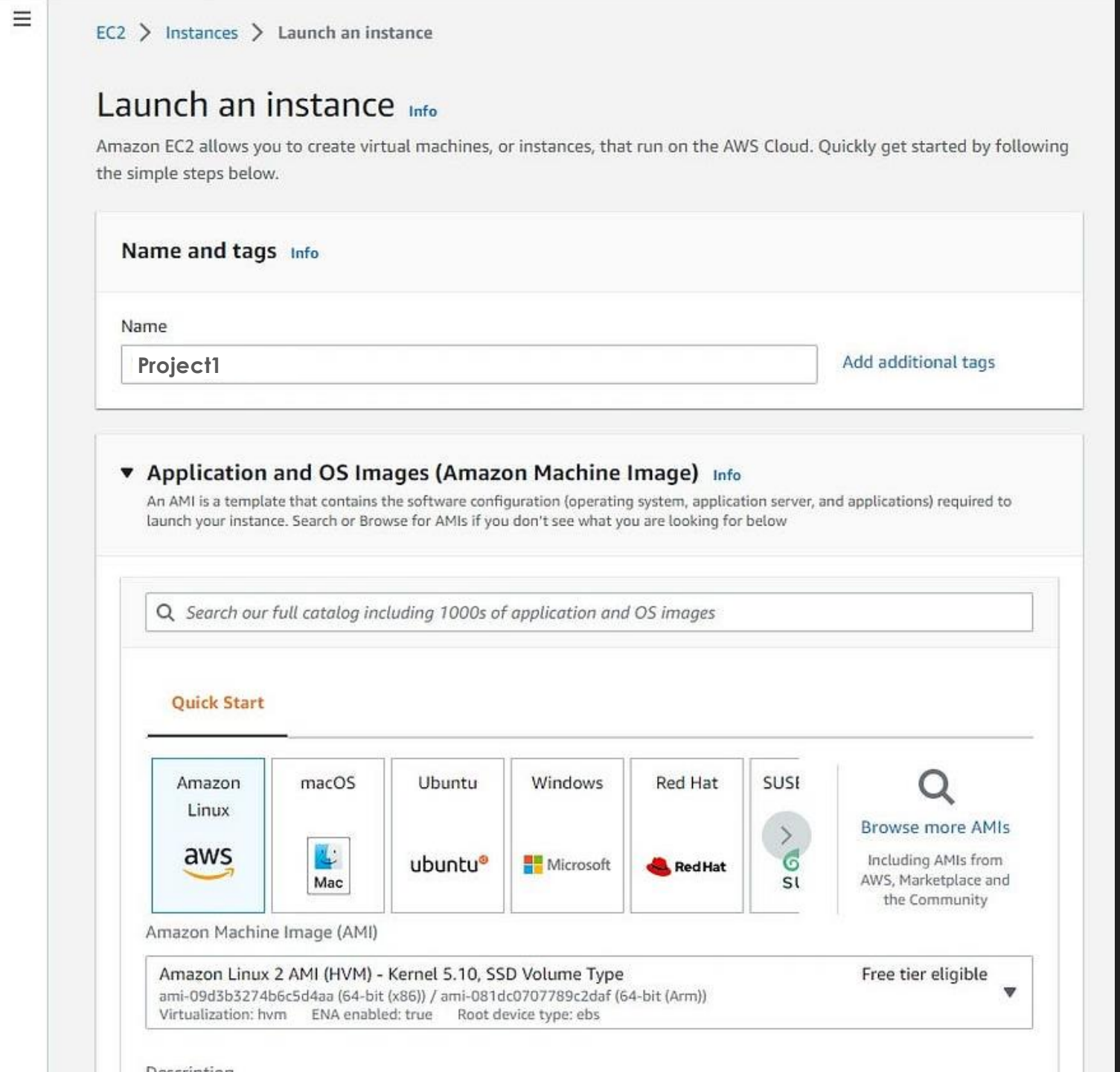
Settings
EBS encryption
Zones
EC2 Serial Console

Step 1

First step is you are going to need to log into your AWS Management Console, go to EC2 and Launch an instance.

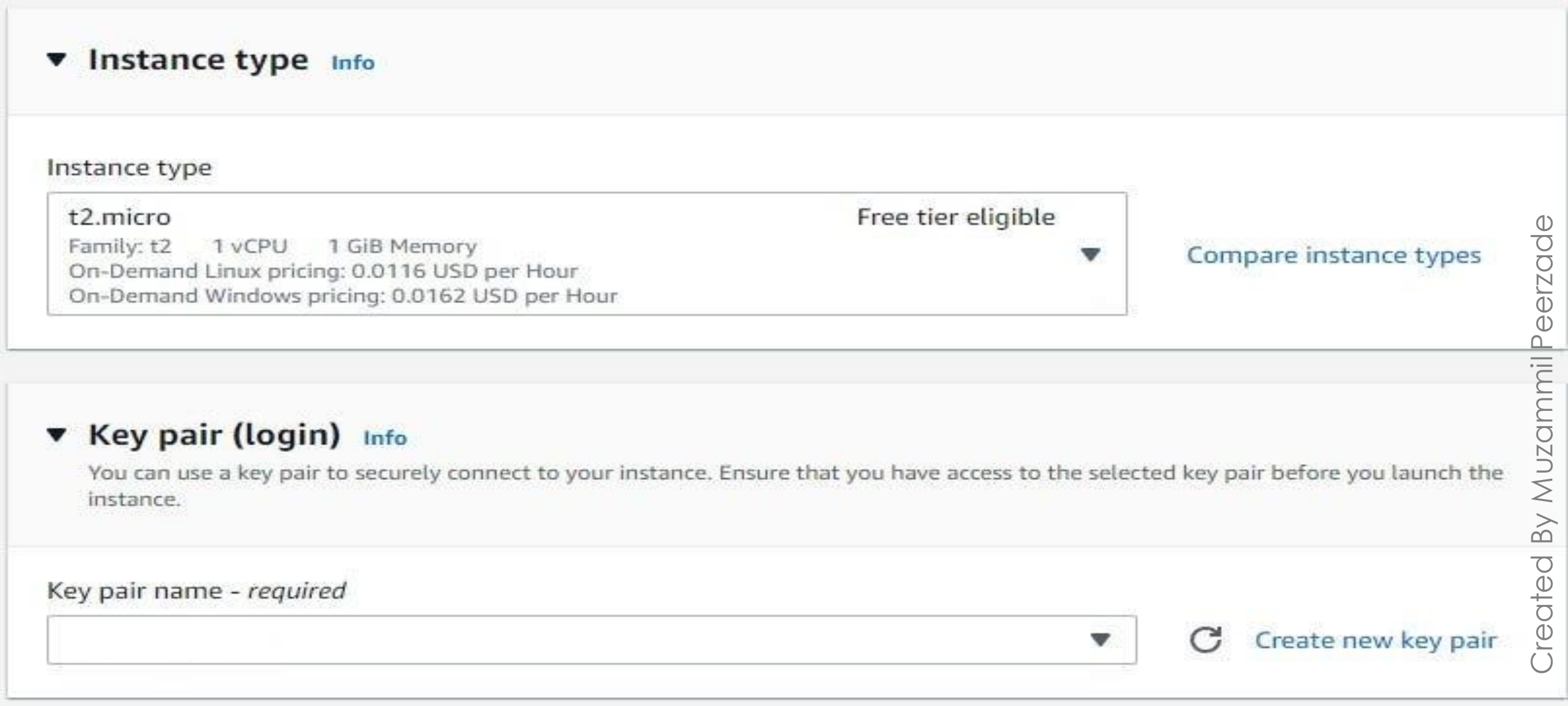
Enable Best Price-Performance with AWS Graviton2

AWS Graviton2 powered EC2 instances enable up to 40% better price performance for a broad spectrum of cloud workloads. Learn more



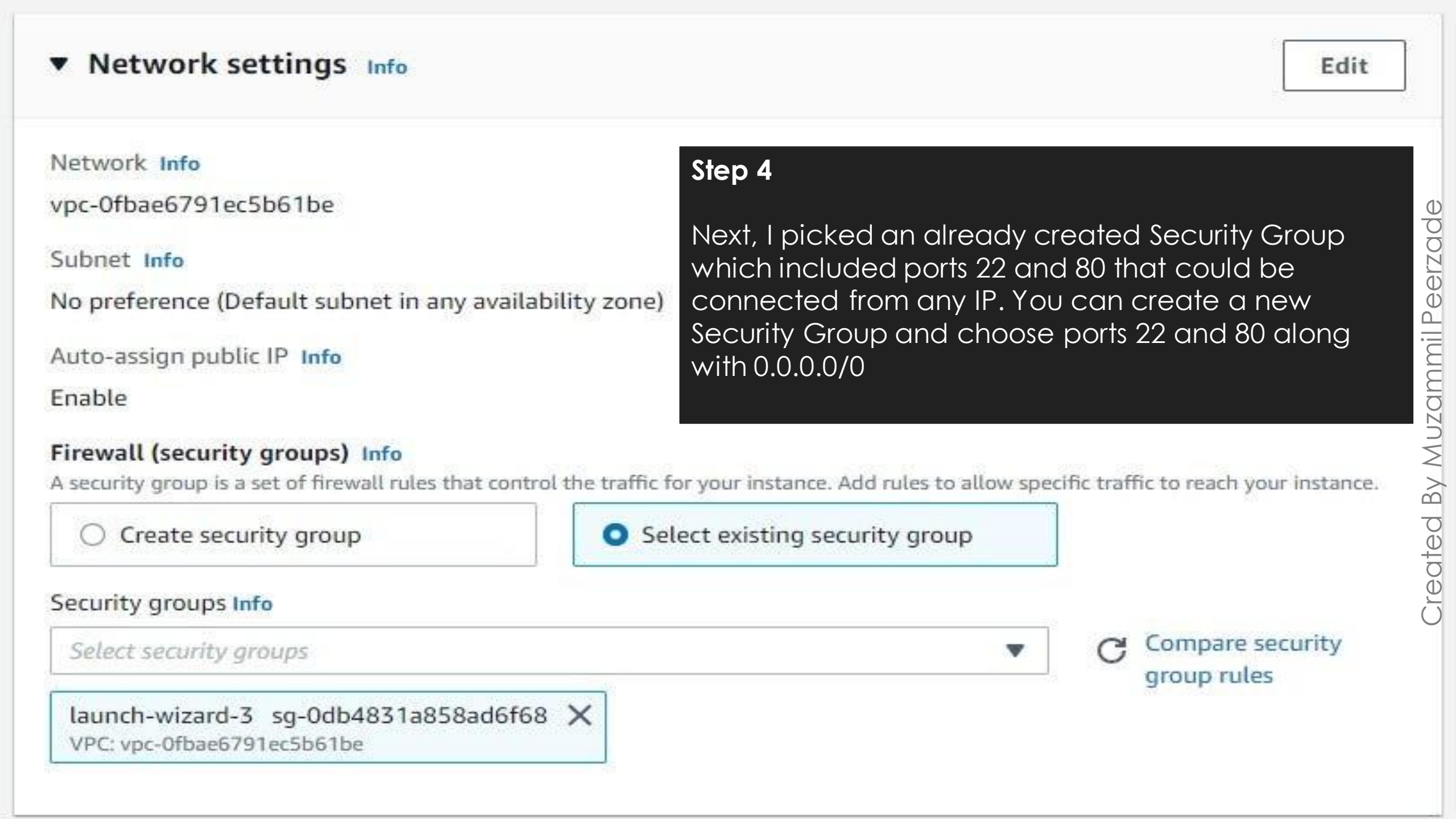
Step 2

Next, we will choose what we want to be included in the instance. Name the instance, my name is "Project". Pick the OS, I picked "Amazon Linux"



Step 3

On this step you will need to pick the instance type, I picked t2.micro which is a part of the free tier. Then either pick a previously created Key Pair or create a Key Pair and save it to your local machine.



Summary

Number of instances Info

1

Software Image (AMI)

Amazon Linux 2 Kernel 5.10 AMI...read more
ami-09d3b3274b6c5d4aa

Virtual server type (instance type)

t2.micro

Firewall (security group)

launch-wizard-3

Storage (volumes)

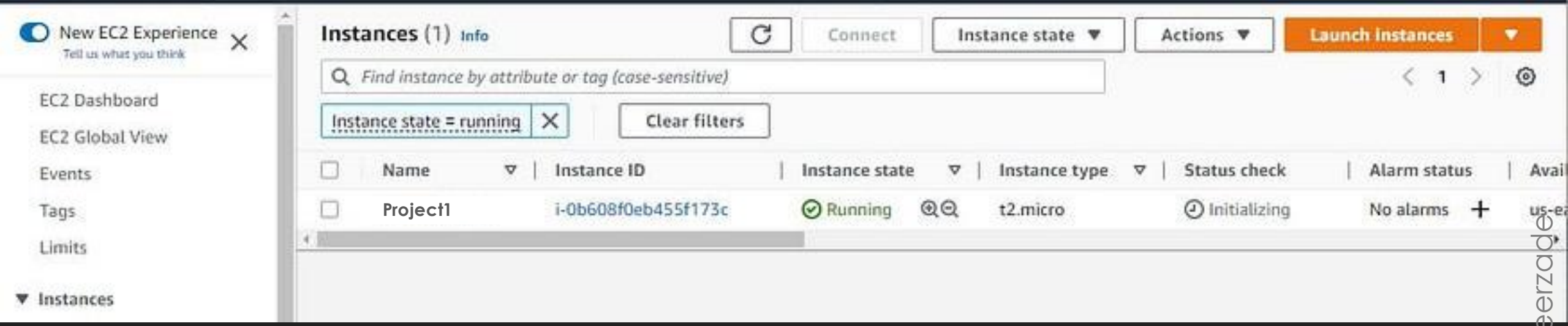
1 volume(s) - 8 GiB

Step 5

For this example I am only going to create one instance, and the rest is a summary of what I have chosen. If everything looks good click "Launch Instance"

Cancel

Launch instance



Step 6

The next screen will show the status of your instance, once it shows "Running" your instance is ready.

```
Microsoft Windows [Version 10.0.19045.3086]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>ssh -i Downloads/pm.pem ec2-user@3.111.188.11
The authenticity of host '3.111.188.11 (3.111.188.11)' can't be established.
ECDSA key fingerprint is SHA256:yhEAHKnHHBsFsn/3FuPfNSBmLjU6KrHHTr0OeIcnKuQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.111.188.11' (ECDSA) to the list of known hosts.

A newer release of "Amazon Linux" is available.
  Version 2023.1.20230705:
Run "/usr/bin/dnf check-release-update" for full release and version update info
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amaz
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-44-23 ~]$ sudo -i
```

Step 7

Now go into your command line, I am using Windows Terminal, to SSH into your newly created instance. Use the below command

ssh -i <yourkeypair.pem> ec2-user@<yourpublicip>

```
[root@ip-172-31-44-23 ~]# yum install httpd -y
Last metadata expiration check: 0:46:48 ago on Thu Jul  6 11:14:40 2023.
Package httpd-2.4.56-1.amzn2023.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-44-23 ~]# systemctl start httpd
[root@ip-172-31-44-23 ~]# systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[root@ip-172-31-44-23 ~]# systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
     Active: active (running) since Thu 2023-07-06 12:01:41 UTC; 20s ago
       Docs: man:httpd.service(8)
   Main PID: 27207 (httpd)
     Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec:   0 B/sec"
      Tasks: 177 (limit: 1114)
     Memory: 12.8M
        CPU: 76ms
     CGroup: /system.slice/httpd.service
             ├─27207 /usr/sbin/httpd -DFOREGROUND
             ├─27208 /usr/sbin/httpd -DFOREGROUND
             ├─27209 /usr/sbin/httpd -DFOREGROUND
             ├─27210 /usr/sbin/httpd -DFOREGROUND
             └─27211 /usr/sbin/httpd -DFOREGROUND

Jul 06 12:01:41 ip-172-31-44-23.ap-south-1.compute.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Jul 06 12:01:41 ip-172-31-44-23.ap-south-1.compute.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Jul 06 12:01:41 ip-172-31-44-23.ap-south-1.compute.internal httpd[27207]: Server configured, listening on: port 80
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
```

Step 8

Install Apache server. Start apache server. Enable apache services.

sudo yum install httpd –y
systemctl start httpd
systemctl enable httpd

Security group name	Security group ID	Description	VPC ID
launch-wizard-1	sg-0fd48576e71044b63	launch-wizard-1 created 2023-07-06T06:48:04.694Z	vpc-04f07ae04bb789ee9

Owner	Inbound rules count	Outbound rules count
457765629814	3 Permission entries	1 Permission entry

Step 9

Kindly Check the in security group of instance, port is allocated for http apache server

Inbound rules | Outbound rules

You can now check network connectivity with Reachability Analyzer — Run Reachability Analyzer

Inbound rules (3)

Manage tags | Edit inbound rules

Filter security group rules

1

Security group rule...	IP version	Type	Protocol	Port range	Source
sgr-0bf4216801ccd3fc3	IPv4	HTTPS	TCP	443	0.0.0.0/0
sgr-0dcd0e3373b02d3...	IPv4	HTTP	TCP	80	0.0.0.0/0
sgr-0b929531a5f1c8a02	IPv4	SSH	TCP	22	0.0.0.0/0

Step 10

Check the IP of your instance on search engine. If its show "**It works**" that means our apache server its working successfully.

Now we installing Docker to run multiple web server !

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime.

Step 11

Install docker, start docker, enable docker services.

```
[root@ip-172-31-44-23 ~]# yum install docker
Last metadata expiration check: 1:07:46 ago on Thu Jul  6 11:14:40 2023.
Package docker-20.10.23-1.amzn2023.0.1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-44-23 ~]# systemctl start docker
[root@ip-172-31-44-23 ~]# systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
     Active: active (running) since Thu 2023-07-06 12:22:41 UTC; 8s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
    Process: 29709 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
    Process: 29710 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
   Main PID: 29711 (dockerd)
      Tasks: 7 (limit: 1114)
     Memory: 31.4M
        CPU: 281ms
     CGroup: /system.slice/docker.service
             └─29711 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.396367586Z" level=info msg="ccResolverWrapper: sending update to cc:
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.396443608Z" level=info msg="ClientConn switching balancer to \"pick_f
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.398823558Z" level=error msg="Failed to built-in GetDriver graph btrfs
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.435164077Z" level=info msg="Loading containers: start."
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.673116948Z" level=info msg="Default bridge (docker0) is assigned with
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.797187685Z" level=info msg="Loading containers: done."
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.904518308Z" level=info msg="Docker daemon" commit=6051f14 graphdriver
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.904839830Z" level=info msg="Daemon has completed initialization"
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal systemd[1]: Started docker.service - Docker Application Container Engine.
Jul 06 12:22:41 ip-172-31-44-23.ap-south-1.compute.internal dockerd[29711]: time="2023-07-06T12:22:41.938298766Z" level=info msg="API listen on /run/docker.sock"
[root@ip-172-31-44-23 ~]# systemctl enable --now docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[root@ip-172-31-44-23 ~]# systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
     Active: active (running) since Thu 2023-07-06 12:22:41 UTC; 30s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 29711 (dockerd)
```

```
[root@ip-172-31-44-23 ~]# docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
[root@ip-172-31-44-23 ~]# docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
faef57eae888: Pull complete
76579e9ed380: Pull complete
cf707e233955: Pull complete
91bb7937700d: Pull complete
4b962717ba55: Pull complete
f46d7b05649a: Pull complete
103501419a0a: Pull complete
Digest: sha256:08bc36ad52474e528cc1ea3426b5e3f4bad8a130318e3140d6cfe29c8892c7ef
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
[root@ip-172-31-44-23 ~]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
faef57eae888: Already exists
7ebb04e7a9fb: Pull complete
50832d624967: Pull complete
efcbeeba1c88: Pull complete
90997b0c5be2: Pull complete
Digest: sha256:8059bdd0058510c03ae4c808de8c4fd2c1f3c1b6d9ea75487f1e5caa5ececa02
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-172-31-44-23 ~]# docker images
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
nginx        latest    021283c8eb95   44 hours ago   187MB
httpd        latest    d140b777a247   2 days ago     168MB
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
```

Step 12

Pull docker images - nginx , httpd with docker command

docker pull nginx
docker pull httpd

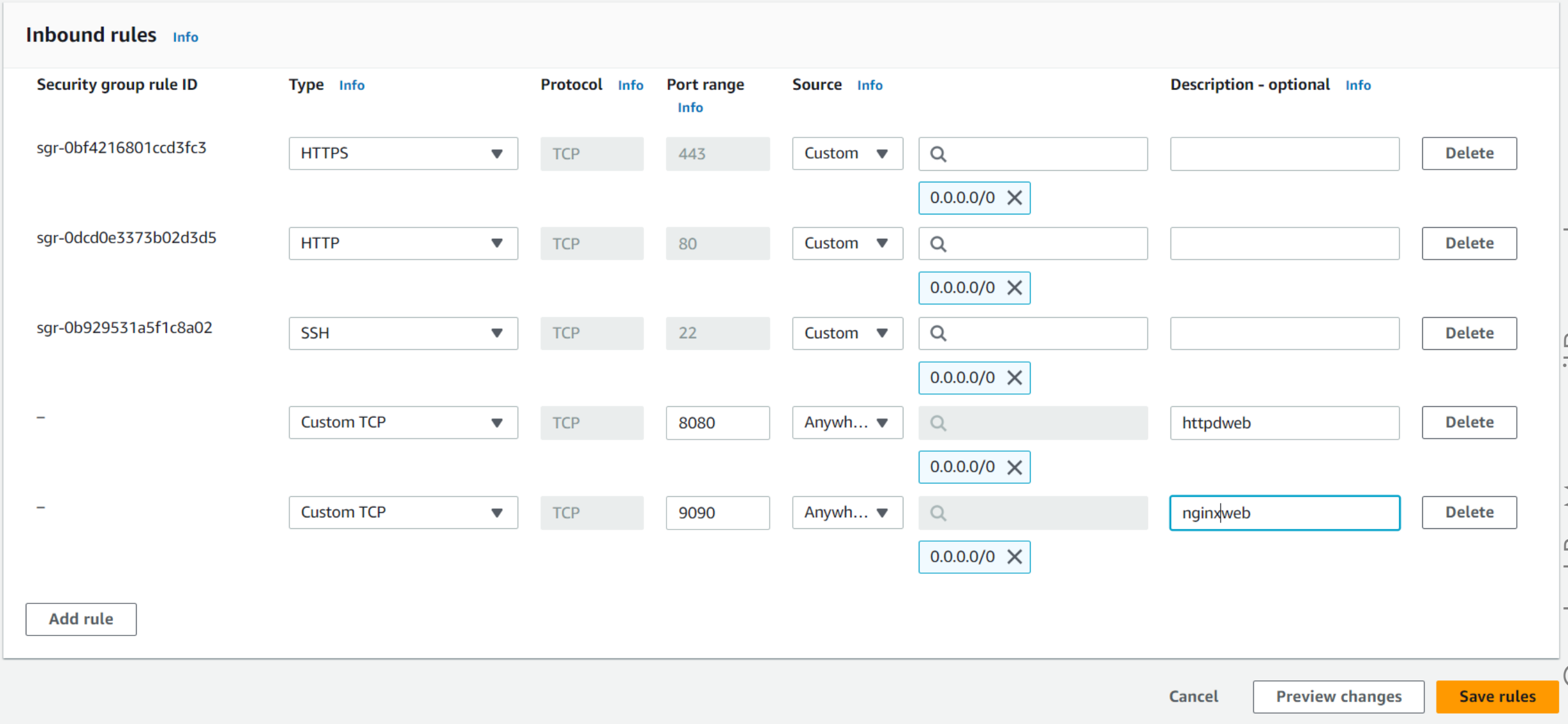
```
[ec2-user@ip-172-31-44-23 ~]$ sudo -i
[root@ip-172-31-44-23 ~]# docker run -itd --name httpdweb -p 8080:80 --hostname muzammil httpd
f875e80eb3b8975128282adf1ea16fd60ee6b4fe32ad40d2e1018d39369eb9ae
[root@ip-172-31-44-23 ~]# docker run -itd --name nginxweb -p 9090:80 --hostname muzammil nginx
37c813e86bd71b635f38e766c6d6bccdd7f17b0e6ed97b5ed1114d51ed047aa7
[root@ip-172-31-44-23 ~]# docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                     NAMES
37c813e86bd7   nginx     "/docker-entrypoint.…"   8 seconds ago    Up 7 seconds    0.0.0.0:9090->80/tcp, :::9090->80/tcp     nginxweb
f875e80eb3b8   httpd     "httpd-foreground"       42 seconds ago   Up 40 seconds   0.0.0.0:8080->80/tcp, :::8080->80/tcp     httpdweb
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
[root@ip-172-31-44-23 ~]#
```

Step 13

Now I am creating a docker container with image httpd and nginx with port 9090 and 8080 with hostname

8080 for httpd
9090 for nginx

With the command of
docker run –itd –name ContainerName –p 8080:80 –hostname HostName image



Step 14

Add the port number of container in instance security group and save.

It works!

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Step 15

Check the instance IP with container ports

All server is working properly.

Now we creating Auto-Scaling Group

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.

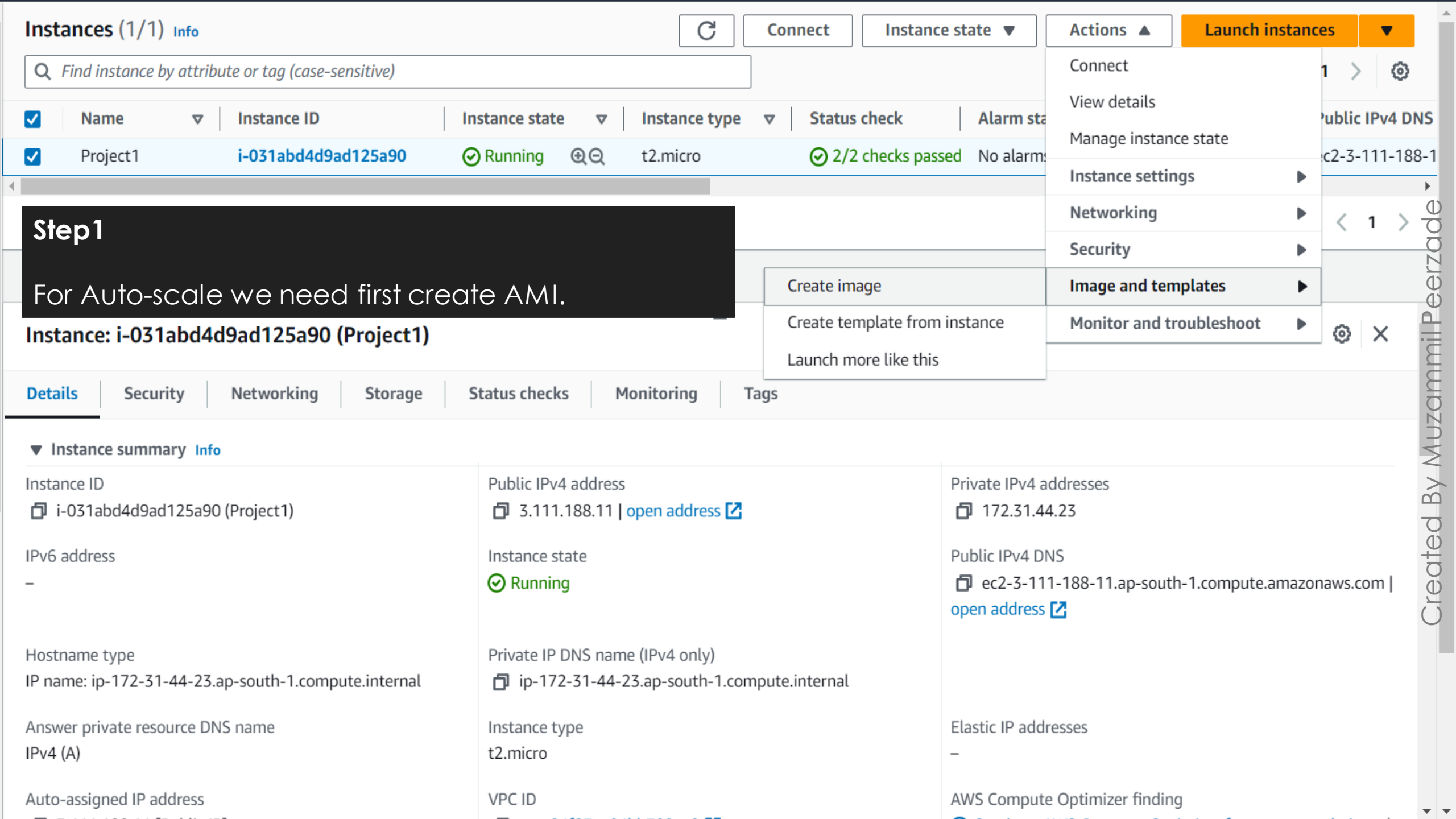

Instances (1/1) Info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
Name
Instance ID
Instance state
Instance type
Status check
Alarm sta
Public IPv4 DNS
Project1
i-031abd4d9ad125a90
Running
t2.micro
2/2 checks passed
No alarms
ec2-3-111-188-1
Connect
View details
Manage instance state
Instance settings
Networking
Security
Image and templates
Monitor and troubleshoot
Create image
Create template from instance
Launch more like this
Step1
For Auto-scale we need first create AMI.
Instance: i-031abd4d9ad125a90 (Project1)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance summary Info
Instance ID
i-031abd4d9ad125a90 (Project1)
Public IPv4 address
3.111.188.11 | open address
Private IPv4 addresses
172.31.44.23
IPv6 address
–
Instance state
Running
Public IPv4 DNS
ec2-3-111-188-11.ap-south-1.compute.amazonaws.com | open address
Hostname type
IP name: ip-172-31-44-23.ap-south-1.compute.internal
Private IP DNS name (IPv4 only)
ip-172-31-44-23.ap-south-1.compute.internal
Answer private resource DNS name
IPv4 (A)
Instance type
t2.micro
Elastic IP addresses
–
Auto-assigned IP address
VPC ID
AWS Compute Optimizer finding
Created By MuzammilPeerzade

Image name

Project1AMI

Maximum 127 characters. Can't be modified after creation.

Image description - *optional*

Image description

Maximum 255 characters

Step 2

Create image

No reboot

☐ Enable

Instance volumes

Storage type	Device	Snapshot	Size	Volume type	IOPS	Throughput	Delete on termination	Encrypted
EBS	/dev/...	Create new snapshot fr...	8	EBS General Purpose S...	3000		☑ Enable	☐ Enable

Add volume

During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.

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.

◉ Tag image and snapshots together
Tag the image and the snapshots with the same tag.

○ Tag image and snapshots separately
Tag the image and the snapshots with different tags.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel | Create image

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

Step 3

Create Template

Launch template name - *required*

Project1TEMPLATE

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

Template version description

A prod webserver for MyApp

Max 255 chars

Auto Scaling guidance **Info**

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

- [x] 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) - required 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 | My AMIs | Quick Start

(•) Owned by me ( ) Shared with me

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Project1AMI
ami-002c03cd14cd1e2fa
2023-07-06T13:50:28.000Z Virtualization: hvm ENA enabled: true Root device type: ebs boot mode: uefi-preferred

Description

-

Architecture	AMI ID
x86_64	ami-002c03cd14cd1e2fa

Step 5

Select AMI

Instance type Info

Advanced

Instance type

t2.micro — Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux pricing: 0.0124 USD per Hour
On-Demand Windows pricing: 0.017 USD per Hour
On-Demand RHEL pricing: 0.0724 USD per Hour
On-Demand SUSE pricing: 0.0124 USD per Hour

All generati

Compare instanc

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

pm

Create new key pair

Network settings Info

Subnet Info

Don't include in launch template

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

Summary

Step 5

Select Instance type key pair if you want to securely launch and create template

t2.micro

Firewall (security group)

-

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel

Create launch template

Step 1
Choose launch template or configuration

Step 2
Choose instance launch options

Step 3 - *optional*
Configure advanced options

Step 4 - *optional*
Configure group size and scaling policies

Step 5 - *optional*
Add notifications

Step 6 - *optional*
Add tags

Step 7
Review

Choose launch template or configuration Info

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group. If you currently use launch configurations, you might conside

Step 6

Add name, select template

Name

Auto Scaling group name
Enter a name to identify the group.

Project1A.S

Must be unique to this account in the current Region and no more than 255 characters.

Launch template Info

Switch to launch configuration

Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

Project1TEMPLATE

Create a launch template

Version

Default (1)

Create a launch template version

Description	Launch template	Instance type
-	Project1TEMPLATE lt-0bf9f5c91c369262d	t2.micro

Step 1
Choose launch template or configuration

Step 2
Choose instance launch options

Step 3 - *optional*
Configure advanced options

Step 4 - *optional*
Configure group size and scaling policies

Step 5 - *optional*
Add notifications

Step 6 - *optional*
Add tags

Step 7
Review

Choose instance launch options Info

Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

Step 7

Choose VPC , Add availability zone

Network Info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-04f07ae04bb789ee9
172.31.0.0/16 Default

Create a VPC

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets

ap-south-1a | subnet-0c1cce9e748283178
172.31.32.0/20 Default

ap-south-1b | subnet-0321e1b505be9ea55
172.31.0.0/20 Default

ap-south-1c | subnet-0afc48312ce2ef062
172.31.16.0/20 Default

Create a subnet

Step 1
Choose launch template or configuration

Step 2
Choose instance launch options

Step 3 - *optional*
Configure advanced options

Step 4 - *optional*
Configure group size and scaling policies

Step 5 - *optional*
Add notifications

Configure advanced options - *optional* Info

Choose a load balancer to distribute incoming traffic for your application across instances to make it more reliable and easily scalable. You can also set options that give you more control over health check replacements and monitoring.

Load balancing Info

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

- (●) No load balancer
  Traffic to your Auto Scaling group will not be fronted by a load balancer.
- ( ) Attach to an existing load balancer
  Choose from your existing load balancers.
- ( ) Attach to a new load balancer
  Quickly create a basic load balancer to attach to your Auto Scaling group.

Step 8

You can configure load balancer if you want run instance with load balancer.

Step 6 - *optional*
Add tags

Step 7
Review

Health checks

Health checks increase availability by replacing unhealthy instances. When you use multiple health checks, all are evaluated, and if at least one fails, instance replacement occurs.

EC2 health checks

Always enabled

Additional health check types - *optional* | Info

Turn on Elastic Load Balancing health checks

Elastic Load Balancing monitors whether instances are available to handle requests. Scaling can replace it on its next periodic check.

Health check grace period | Info

This time period delays the first health check until your instances finish initializing. It doesn't prevent an instance from terminating when placed into a non-running state.

300 seconds

Step 9

You can configure for health check also it will check the instance every 300 second if there is issue in instance it will create new once

Additional settings

Monitoring | Info

Enable group metrics collection within CloudWatch

Default instance warmup | Info

The amount of time that CloudWatch metrics for new instances do not contribute to the group's aggregated instance metrics, as their usage data is not reliable yet.

Enable default instance warmup

Cancel | Skip to review | Previous | Next

Step 1
Choose launch template or configuration

Step 2
Choose instance launch options

Step 3 - *optional*
Configure advanced options

Step 4 - *optional*
Configure group size and scaling policies

Step 5 - *optional*
Add notifications

Step 6 - *optional*
Add tags

Step 7

Configure group size and scaling policies - *optional* Info

Set the desired, minimum, and maximum capacity of your Auto Scaling group. You can optionally add a scaling policy to dynamically scale the number of instances in the group.

Group size - *optional* Info

Specify the size of the Auto Scaling group by changing the desired capacity. You can also specify minimum and maximum capacity limits. Your desired capacity must be within the limit range.

Desired capacity

3

Minimum capacity

3

Maximum capacity

3

Step 10

Add grouping size it will create 3 instance

No scaling policy

Instance scale-in protection

Instance scale-in protection

Enable instance protection from scale in

Step 5: Add notifications — Edit

Notifications

No notifications

Step 6: Add tags — Edit

Tags (0)

Key	Value	Tag new instances

No tags

Step 11

Create Auto Scaling Group

Cancel | Previous | Create Auto Scaling group

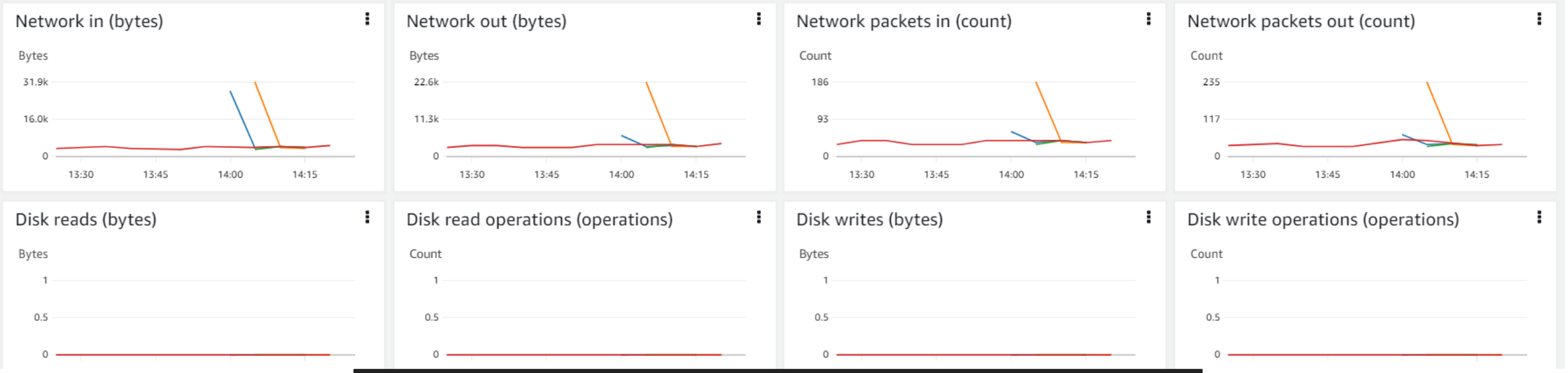
Instances (4/4) Info

Connect | Instance state | Actions | Launch instances

Find instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
Project1	i-031abd4d9ad125a90	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a	ec2-3-111-188-11.ap-s...	3.111.188.11	–
–	i-076c150464e120b59	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a	ec2-43-204-35-247.ap-...	43.204.35.247	–
–	i-0e2731dff4491b9aa	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1a	ec2-35-154-213-81.ap-...	35.154.213.81	–
–	i-07532badeb64b29e3	Running	t2.micro	2/2 checks passed	No alarms	ap-south-1b	ec2-3-109-121-13.ap-s...	3.109.121.13	–

Instances: i-07532badeb64b29e3, i-0e2731dff4491b9aa, i-076c150464e120b59, i-031abd4d9ad125a90 (Project1)



Step 12

We have been created scaling group successfully .
Auto-scale Created three more instance

Thank You