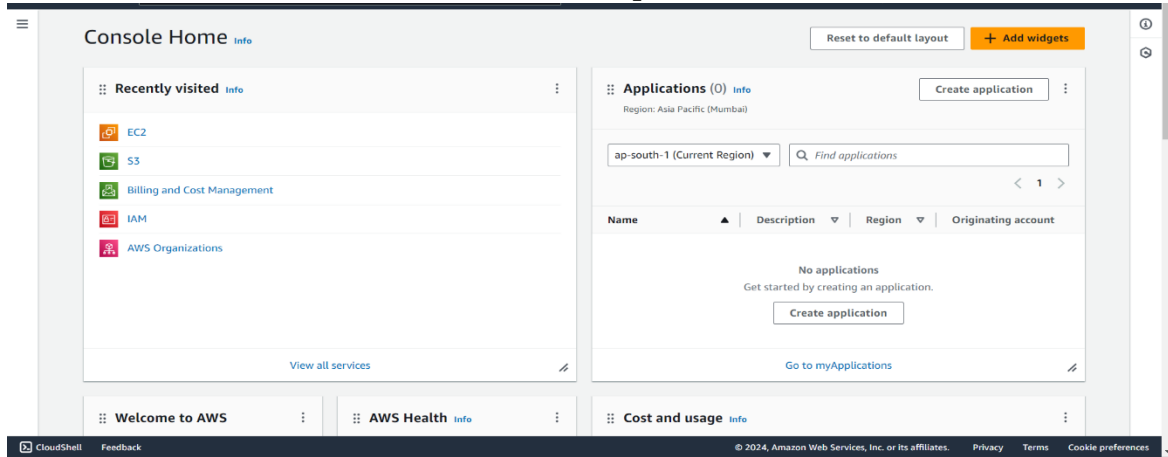


ASSIGNMENT -> 11

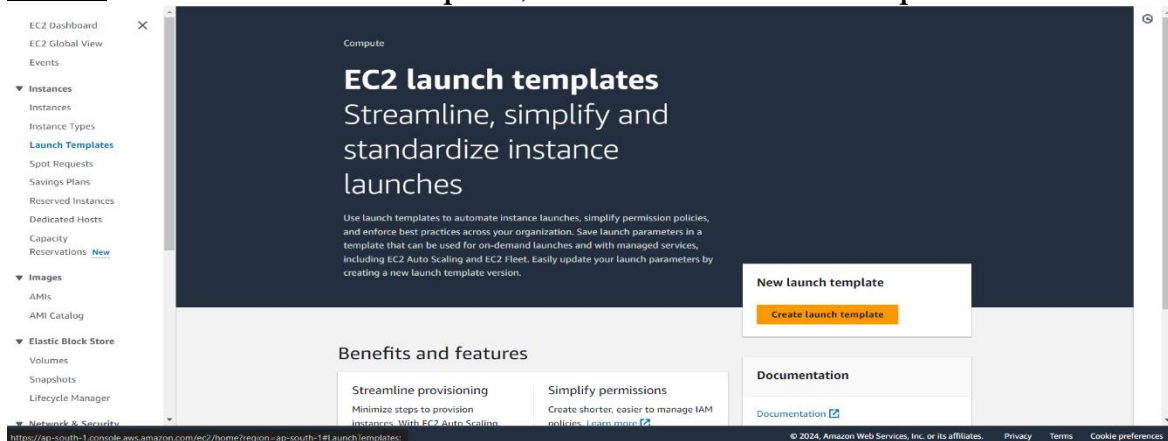
PROBLEM STATEMENT ->

Build scaling plans in AWS that balances the load on different EC2 instances.

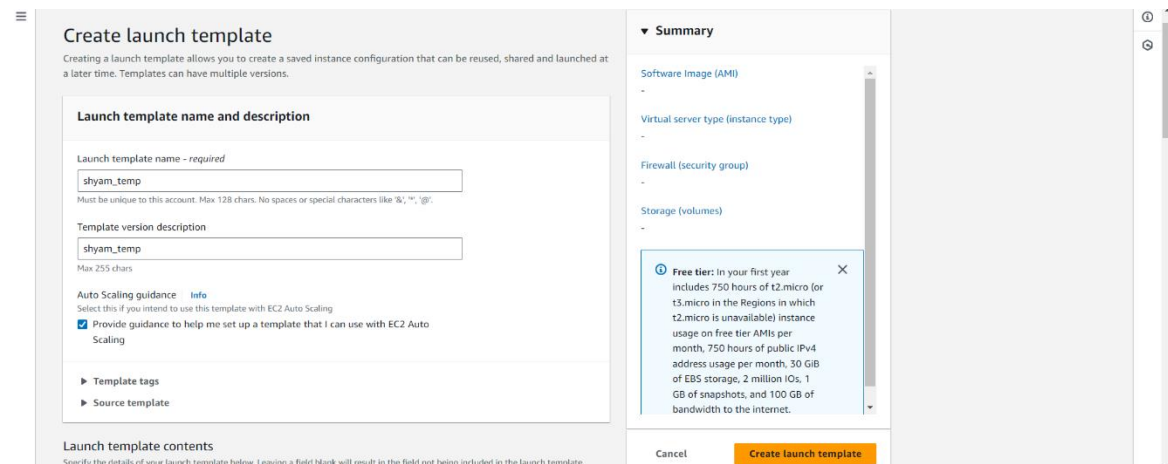
STEP 1-> From AWS home screen, select EC2 option.



STEP 2-> Under the Launch Templates, click on Create Launch Template



STEP 3-> Give a name and description to the template. Check the Auto Scaling Guidance checkbox.



STEP 7-> Scroll down to the bottom, in the bash console type the following commands: Then click on Create Launch Template .

Metadata response hop limit **Info**

2

Allow tags in metadata **Info**

Don't include in launch template

User data - optional **Info**

Upload a file with your user data or enter it in the field.

Choose file

```
#!/bin/bash
apt-get update
apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
apt-get install -y git
curl -SL https://deb.nodesource.com/setup_16.x | sudo -E bash -
apt-get install -y nodejs
git clone https://github.com/Shyam-SS/Assign8.git
cd Assign8
npm install
node index.js
```

☐ User data has already been base64 encoded

Summary

Software image (AMI)

Canonical, Ubuntu, 24.04 LTS, ...read more
ami-0f58b397bc5c1f2e8

Virtual server type (instance type)

t3.micro

Firewall (security group)

shyamsecurity

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, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of

Cancel **Create launch template**

STEP 8->Click on Auto Scaling Group .

Success

Successfully created shyam_temp(t-07048e838c1b05faf)

Actions log

Next Steps

Launch an Instance

With On-Demand Instances, you pay for compute capacity by the second (for Linux, with a minimum of 60 seconds) or by the hour (for all other operating systems) with no long-term commitments or upfront payments. Launch an On-Demand Instance from your launch template.

[Launch instance from this template](#)

Create an Auto Scaling group from your template

Amazon EC2 Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically according to conditions you define. You can use Auto Scaling to help ensure that you are running your desired number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs.

[Create Auto Scaling group](#)

Create Spot Fleet

A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance (of each instance type in each Availability Zone) is set by Amazon EC2, and adjusted gradually based on the long-term supply of and demand for Spot Instances. Spot instances are well-suited for data-analysis, batch jobs, background processing, and optional tasks.

[Create Spot Fleet](#)

STEP 9-> Click on Create Auto Scaling Group .

AMI Catalog

- ▼ Elastic Block Store
 - Volumes
 - Snapshots
 - Lifecycle Manager
- ▼ Network & Security
 - Security Groups
 - Elastic IPs
 - Placement Groups
 - Key Pairs
 - Network Interfaces
- ▼ Load Balancing
 - Load Balancers
 - Target Groups
 - Trust Stores [New](#)
- ▼ Auto Scaling
 - Auto Scaling Groups**
- Settings

Amazon EC2 Auto Scaling

helps maintain the availability of your applications

Create Auto Scaling group

Get started with EC2 Auto Scaling by creating an Auto Scaling group.

Create Auto Scaling group

Auto Scaling groups are collections of Amazon EC2 instances that enable automatic scaling and fleet management features. These features help you maintain the health and availability of your applications.

How it works

Auto Scaling group

Pricing

Amazon EC2 Auto Scaling features have no additional fees beyond the service fees for Amazon EC2, CloudWatch (for scaling policies), and the other AWS resources that you use. Visit the pricing page of each service to learn more.

STEP 10-> Give a name and select the newly created Template. Then, click on Next .

The screenshot shows the 'Launch template' step in the AWS Management Console. The left sidebar contains navigation links for Steps 4 through 7. The main content area is titled 'Launch template' and includes a warning box about account creation dates. Below this, there's a section for 'Launch template' with a dropdown menu set to 'shyam_temp'. A 'Create a launch template version' button is visible. The 'Additional details' section shows the description 'shyam_temp', AMI ID 'ami-0f58b397bdc1f2e8', key pair 'Key_123', and instance type 't3.micro'. At the bottom, there are 'Cancel' and 'Next' buttons.

STEP 11-> In Network tab select all the available zones. Then click on next.

The screenshot shows the 'Network' tab in the AWS Management Console. The left sidebar contains navigation links for Steps 4 through 7. The main content area is titled 'Network' and includes a warning box about Availability Zones. Below this, there's a section for 'VPC' with a dropdown menu set to 'vpc-0bc5549f9241a6455'. A 'Create a VPC' button is visible. The 'Availability Zones and subnets' section shows three subnets selected: 'ap-south-1c', 'ap-south-1b', and 'ap-south-1a'. At the bottom, there are 'Cancel', 'Skip to review', 'Previous', and 'Next' buttons.

STEP 12-> Select Attach a new load balancer , select Application Load Balancer & select InternetFacing .

The screenshot shows the 'Load balancing' step in the AWS Management Console. The left sidebar contains navigation links for Steps 3 through 7. The main content area is titled 'Load balancing' and includes a warning box about attaching a load balancer. Below this, there's a section for 'Attach to a new load balancer' with three radio buttons: 'No load balancer', 'Attach to an existing load balancer', and 'Attach to a new load balancer'. The 'Attach to a new load balancer' option is selected. Below this, there's a section for 'Load balancer type' with two radio buttons: 'Application Load Balancer' and 'Network Load Balancer'. The 'Application Load Balancer' option is selected. Below this, there's a section for 'Load balancer name' with a text input field containing 'shyam_scale-1'. Below this, there's a section for 'Load balancer scheme' with two radio buttons: 'Internal' and 'Internet-facing'. The 'Internet-facing' option is selected. At the bottom, there are 'Cancel' and 'Next' buttons.

STEP 13-> Give the port no. 4000 & select Create a target group. Then select No VPC Lattice Service.

Listeners and routing

If you require secure listeners, or multiple listeners, you can configure them from the [Load Balancing console](#) after your load balancer is created.

Protocol: HTTP Port: 4000

Default routing (forward to): Create a target group

New target group name: shyam_scale

Tags - optional

Consider adding tags to your load balancer. Tags enable you to categorize your AWS resources so you can more easily manage them.

Add tag

50 remaining

VPC Lattice integration options

To improve networking capabilities and scalability, integrate your Auto Scaling group with VPC Lattice. VPC Lattice facilitates communications between AWS services and helps you connect and manage your applications across compute services in AWS.

Select VPC Lattice service to attach:

☒ No VPC Lattice service

☐ Attach to VPC Lattice service

Create new VPC Lattice service

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.

STEP 14-> Check the Turn on Elastic Load Balancing Health checks checkbox. Give the Health Check Grace Period of 240 seconds. Click on NEXT .

EC2 health checks

[Always enabled](#)

Additional health check types - optional

☒ Turn on Elastic Load Balancing health checks

EC2 Auto Scaling will start to detect and act on health checks performed by Elastic Load Balancing. To avoid unexpected terminations, first verify the settings of these health checks in the [Load Balancer console](#).

Turn on VPC Lattice health checks

VPC Lattice can monitor whether instances are available to handle requests. If it considers a target as failed a health check, EC2 Auto Scaling replaces it after its next periodic check.

Health check grace period

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.

180 seconds

Additional settings

Monitoring

☐ Enable group metrics collection within CloudWatch

Default instance warmup

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 15->Under Desired capacity, give a size of 2.Under Scaling, give min capacity 2 & max capacity 3.

Group size

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 (GB) 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

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: 2

Max desired capacity: 3

Automatic scaling - optional

Choose whether to use a target tracking policy

☒ No scaling policies

☐ Target tracking scaling policy

Next

STEP 16->Select Target tracking scaling policy . And give the instance warmup time

of 240 seconds. Then click on Next .

4 3
Equal or less than desired capacity Equal or greater than desired capacity

Automatic scaling - optional [Info](#)
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
50

Instance warmup [Info](#)
180 seconds

☐ Disable scale in to create only a scale-out policy

Instance maintenance policy [Info](#)
Control your Auto Scaling group's availability during instance replacement events. This includes health checks, instance refreshes, maximum instance lifetime features and events that happen automatically to keep your group balanced, called rebalancing events.

STEP 17-> Click on Next .

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
[Choose launch template](#)

Step 2
[Choose instance launch options](#)

Step 3 - optional
[Configure advanced options](#)

Step 4 - optional
[Configure group size and scaling](#)

Step 5 - optional
Add notifications

Step 6 - optional
[Add tags](#)

Step 7
[Review](#)

Add notifications - optional [Info](#)
Send notifications to SNS topics whenever Amazon EC2 Auto Scaling launches or terminates the EC2 instances in your Auto Scaling group.

Add notification

Cancel Skip to review Previous Next

STEP 18-> Click on Next .

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
[Choose launch template](#)

Step 2
[Choose instance launch options](#)

Step 3 - optional
[Configure advanced options](#)

Step 4 - optional
[Configure group size and scaling](#)

Step 5 - optional
[Add notifications](#)

Step 6 - optional
Add tags

Step 7
[Review](#)

Add tags - optional [Info](#)
Add tags to help you search, filter, and track your Auto Scaling group across AWS. You can also choose to automatically add these tags to instances when they are launched.

☒ You can optionally choose to add tags to instances (and their attached EBS volumes) by specifying tags in your launch template. We recommend caution, however, because the tag values for instances from your launch template will be overridden if there are any duplicate keys specified for the Auto Scaling group.

Tags (0)

Add tag

50 remaining

Cancel Previous Next

STEP 19-> Review all the data of the group to be created and click on Create Auto Scaling Group .

Instance maintenance policy

Replacement behavior	Min healthy percentage	Max healthy percentage
No 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		

Cancel Previous Create Auto Scaling group

STEP 20-> After creating the scaling group, go back to Instances from the left side menu.

Auto Scaling groups (1) info

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
shyam_scale	shyam_template Version Default	0	Updating capacity...	2	2	3	ap-south-1c, ap-south-1b, ap-south-1a

0 Auto Scaling groups selected

STEP 21-> Since the capacity was given as 2, two instances are created. Now open any one of the the instance by clicking on its id.

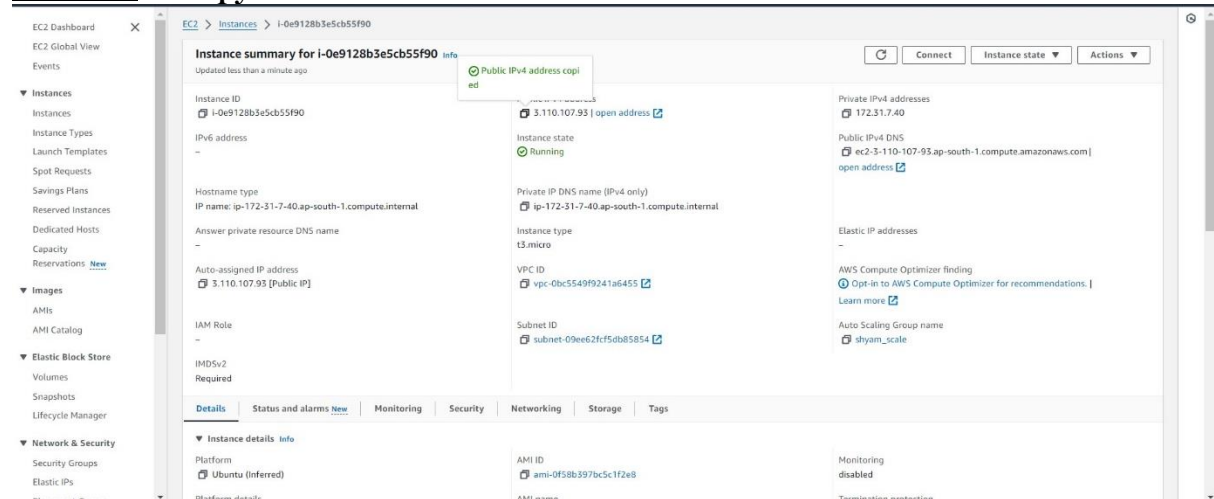
Instances (2) info

All states

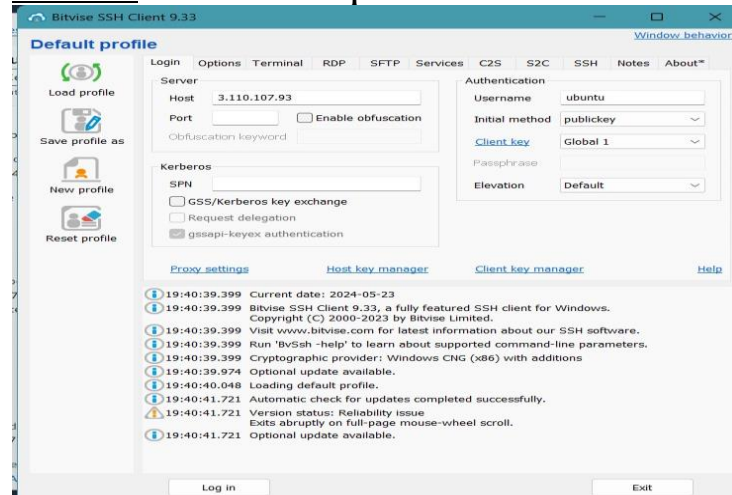
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
	i-0e9128b3e5cb55f90	Running	t3.micro	2/2 checks passed	View alarms	ap-south-1b	ec2-3-110-107-95.ap-s...	3.110.107.93	-
	i-03801ce5be9c3e248	Running	t3.micro	2/2 checks passed	View alarms	ap-south-1a	ec2-52-66-135-179.ap-...	52.66.135.179	-

Select an instance

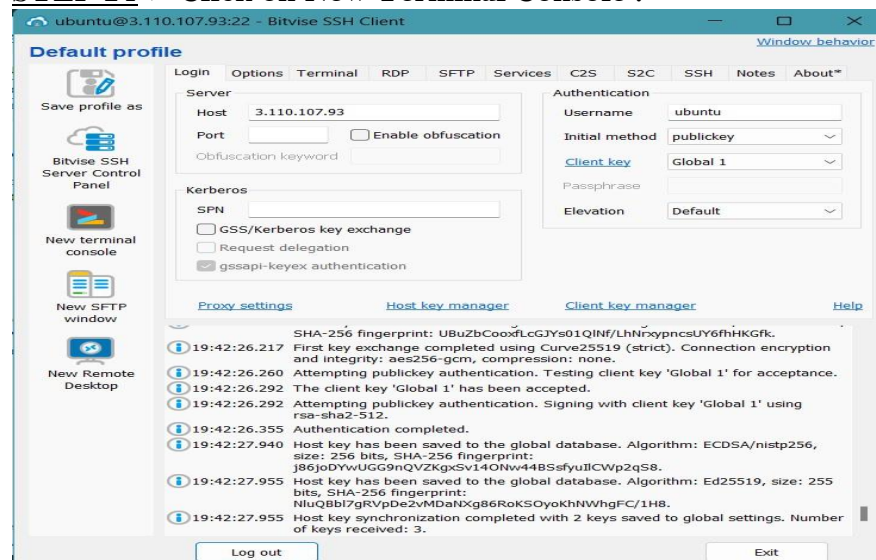
STEP 22-> Copy its Public IPv4 Address.



STEP 23-> Paste the copied address and click on Log in .



STEP 24-> Click on New Terminal Console .



STEP 25-> Type the command:

```
ubuntu@ip-172-31-7-40:~$ sudo nano infy.sh
```

STEP 26-> Write the following code for an infinite loop in the infy.sh file.

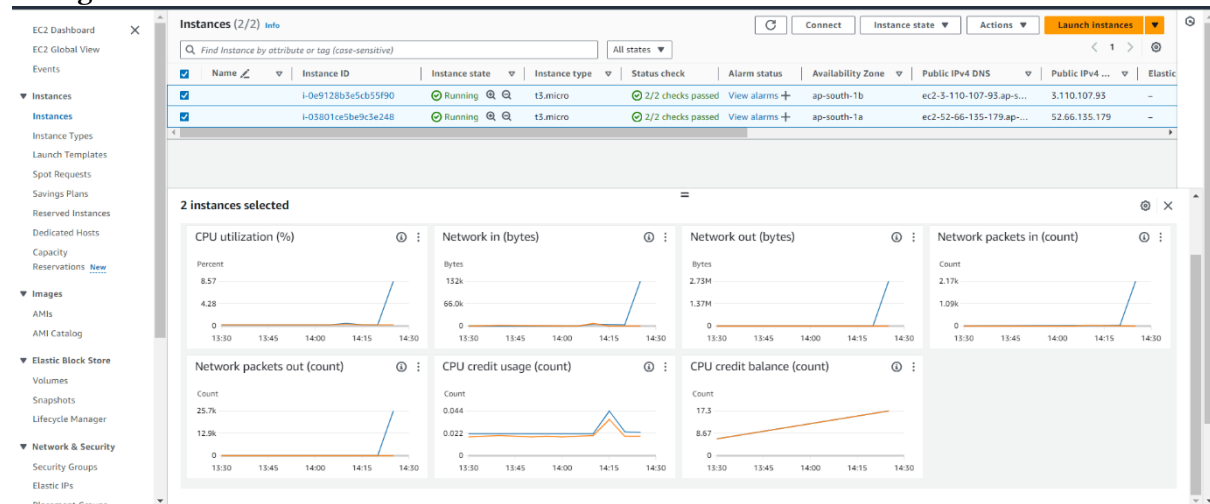
```
ubuntu@3.110.107.93:22 - Bitvise xterm - ubuntu@ip-172-31-7-40: ~
GNU nano 7.2 infy.sh
#!/in/bash
while(true)
do
echo "Inside loop"
done
```

STEP 27-> Write the following commands in the terminal:

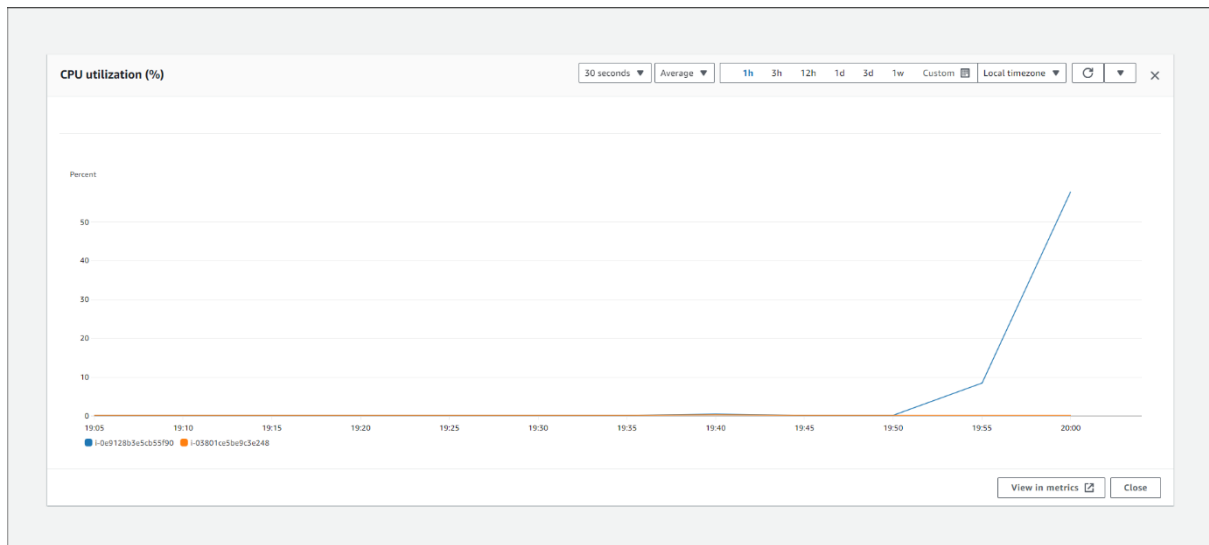
```
ubuntu@ip-172-31-30-92:~$ sudo chmod 777 infy.sh
ubuntu@ip-172-31-30-92:~$ sh infy.sh
```

```
ubuntu@3.110.107.93:22 - Bitvise xterm - ubuntu@ip-172-31-7-40: ~
Inside loop
Inside loop
Inside loop
Inside loop
Inside loop
Inside loop
Inside loop
Inside loop
Inside loop
Inside loop
Inside loop
```

STEP 28-> Select both the instances, then under monitoring go to CPU utilization and enlarge it.



STEP 29-> The graph shows the CPU Utilization for both the instances.



When the CPU utilization exceed the limit for both the instances, a new instance will be created.