

## Task 5.

# Load Balancer & Auto Scaling

## Configure Application Load Balancer (ALB)

### Attach Auto Scaling Group (ASG)

### Scale based on CPU utilization

## Steps:

### Step 1: Create Target Group mytg

#### 1. EC2 → Target Groups → clicking on Create

The screenshot shows the 'Create target group' wizard in the AWS EC2 console. The title bar says 'Step 1 Create target group | EC2'. The left sidebar has three steps: 'Create target group' (selected), 'Register targets', and 'Review and create'. The main content area is titled 'Create target group' and explains that a target group can be made up of one or more targets. It shows four target type options: 'Instances' (selected), 'IP addresses', 'Lambda function', and 'Application Load Balancer'. Under 'Instances', it says 'Supports load balancing to instances in a VPC. Integrate with Auto Scaling Groups or ECS services for automatic management.' and 'Suitable for: ALB NLB GWLB'. Under 'IP addresses', it says 'Supports load balancing to VPC and on-premises resources. Facilitates routing to IP addresses and network interfaces on the same instance. Supports IPv6 targets.' and 'Suitable for: ALB NLB GWLB'. Under 'Lambda function', it says 'Supports load balancing to a single Lambda function. ALB required as traffic source.' and 'Suitable for: ALB'. Under 'Application Load Balancer', it says 'Allows use of static IP addresses and PrivateLink with an Application Load Balancer. NLB required as traffic source.' and 'Suitable for: NLB'. Below this is a 'Target group name' field containing 'mytg'. A note at the bottom says 'Accepts: a-z, A-Z, 0-9, and hyphen (-). Can't begin or end with hyphen. 1-32 total characters; Count: 4/32'. The bottom of the page includes standard AWS footer links like CloudShell, Feedback, Console Mobile App, Privacy, Terms, and Cookie preferences.

The screenshot shows the 'mytg' target group details page in the AWS EC2 console. The left sidebar lists categories: AMI Catalog, Elastic Block Store, Network & Security, and Load Balancing. The main content area shows a success message: 'Successfully created the target group: mytg. Anomaly detection is automatically applied to all registered targets. Results can be viewed in the Targets tab.' Below this, the target group name 'mytg' is displayed. The 'Details' section shows the ARN: arn:aws:elasticloadbalancing:ap-south-1:814337589369:targetgroup/mytg/d9cb51c515141a5f, Target type: Instance, Protocol: Port (HTTP: 80), Protocol version: HTTP1, IP address type: IPv4, and Load balancer: None associated. A table below shows target status: 0 healthy, 0 unhealthy, 0 unused, 0 initial, and 0 draining. The bottom right corner has a 'Actions' dropdown.

## Step 2: Create Application Load Balancer

EC2 → Load Balancers → Create

The screenshot shows the AWS Cloud Console with the URL [ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#SelectCreateELBWizard](https://ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#SelectCreateELBWizard). The page title is "Compare and select load balancer type". It provides a comparison between three types of load balancers:

- Application Load Balancer (ALB)**: Handles HTTP and HTTPS traffic. Targets include Lambda functions, Amazon S3 buckets, and Amazon EC2 instances.
- Network Load Balancer (NLB)**: Handles TCP, UDP, and TLS traffic. Targets include VPC endpoints, ALBs, and NLBs.
- Gateway Load Balancer (GWLB)**: Handles traffic from external sources. Targets include AWS WAF rules, AWS Lambda functions, and AWS Step Functions.

Select application load balancer and click on create.

The screenshot shows the AWS Cloud Console with the URL [ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateALBWizard](https://ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#CreateALBWizard). The page title is "Create Application Load Balancer". The "Basic configuration" section includes:

- Load balancer name**: alb
- Scheme**: Internet-facing (selected)
- Load balancer IP address type**: IPv4 (selected)

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

You can optionally choose to configure an IPAM pool as the preferred source for your load balancers 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 node will 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.

ap-south-1a (aps1-az1)  
Subnet  
Only CIDR 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-04cff1c02b7d4f454  
IPv4 subnet CIDR: 172.31.32.0/20

ap-south-1b (aps1-az3)  
Subnet  
Only CIDR 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-0870f37a4647fd624  
IPv4 subnet CIDR: 172.31.0.0/20

ap-south-1c (aps1-az2)  
Subnet  
Only CIDR 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-0b794ec7dbb17441d  
IPv4 subnet CIDR: 172.31.16.0/20

**Security groups** | [Info](#)

CloudShell Feedback Console Mobile App © 2026, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

selected Availability Zones only

Security Group:

- Allow HTTP (80)

Listener:

- HTTP → 80 → Forward to **Target Group**

**Listeners and routing** | [Info](#)

A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

**Listener HTTP:80**

Protocol	Port
HTTP	80

**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 groups    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	Weight	Percent
Select a target group	1	100%

**Add target group**  
You can add up to 4 more target groups.

# Successfully created application load balancer

Introducing ALB target optimizer  
Target optimizer lets you enforce a maximum number of requests per target using an ALB-provided agent, improving success rates, latency, and efficiency. [Learn more](#)

**alb**

**Details**

Load balancer type	Status	Load balancer IP address type
Application	Provisioning	IPv4
Scheme	Hosted zone	Date created
Internet-facing	ZP97RAFLXTNZK	February 8, 2026, 03:16 (UTC+05:30)
Load balancer ARN	DNS name	
<a href="#">arn:aws:elasticloadbalancing:ap-south-1:814337389369:loadbalancer/app/alb/ed45da460130fec</a>	<a href="#">Info</a> <a href="#">alb-596315428.ap-south-1.elb.amazonaws.com (A Record)</a>	

## Create Auto Scaling Group (ASG)

### Step 1: Create Launch Template mytmp

**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

mytmp

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

**Template version description**

my

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

**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.

**Summary**

Software Image (AMI)

Virtual server type (instance type)

Firewall (security group)

Storage (volumes)

**Free tier:** In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of

[Cancel](#) [Create launch template](#)

## Write user data:

The screenshot shows the 'Create launch template' page in the AWS EC2 console. Under the 'User data - optional' section, a file input field contains the following shell script:

```
#!/bin/bash
yum install httpd -y
systemctl start httpd
systemctl enable httpd
echo "Hello from Auto Scaling" > /var/www/html/index.html
```

Launch Templates (1)						
<a href="#">Actions</a> <a href="#">Create launch template</a>						
<input type="text"/> Search						
Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By	
<a href="#">lt-047e6b94961862669</a>	mytmp	1	1	2026-02-07T21:57:23.000Z	arn:aws:iam::8143	

## Step 2: Create Auto Scaling Group myasg

The screenshot shows the 'Create Auto Scaling group' wizard in the AWS EC2 console, currently at Step 1: Choose launch template.

**Step 1**  
Choose launch template

**Name**  
Auto Scaling group name  
myasg

**Launch template**  
mytmp

## • Select VPC & subnets

The screenshot shows the 'Create Auto Scaling group' wizard on the AWS EC2 console. The current step is 'Step 1 - Choose launch template'. The 'VPC' section is highlighted, showing the selection of 'vpc-052859de1479c19a9' and its default subnet '172.31.0.0/16'. The 'Availability Zones and subnets' section lists three availability zones: 'aps1-az1 (ap-south-1a) | subnet-04cff1c02b7d4f454', 'aps1-az2 (ap-south-1c) | subnet-0b794ec7dbb17441d', and 'aps1-az3 (ap-south-1b) | subnet-0870f37a4647fd624'. The 'Availability Zone distribution' section shows 'Balanced best effort' selected.

## Attach to existing Load Balancer

### Select Target Group

The screenshot shows the 'Create Auto Scaling group' wizard on the AWS EC2 console, currently at 'Step 2 - Choose instance launch options'. The 'Integrate with other services - optional' section is selected. The 'Load balancing' section is highlighted, showing the 'Attach to an existing load balancer' option selected. Below it, the 'Select the load balancers to attach' section shows 'Choose from your load balancer target groups' selected. A dropdown menu displays a target group named 'mytg | HTTP'.

## Group size:

- Desired: 1
- Min: 1
- Max: 3

The screenshot shows the AWS Auto Scaling Groups creation wizard at Step 4: Configure group size and scaling. The left sidebar lists steps from 1 to 7. The main panel is titled 'Group size' with an info link. It explains setting the initial size of the Auto Scaling group. A dropdown menu for 'Units (number of instances)' is open, showing '1'. Below it, a 'Desired capacity' field contains the value '1'. The 'Scaling' section follows, with an info link. It discusses resizing the group manually or automatically. Under 'Scaling limits', 'Min desired capacity' is set to '1' and 'Max desired capacity' is set to '3'. An 'Automatic scaling - optional' section includes a link to choose a target tracking policy.

## Create Myasg

The screenshot shows the AWS Auto Scaling groups list page. At the top, there's a search bar and a table header with columns: Name, Launch template/configuration, Instances, Status, Desired capacity, Min, Max, and Availability Zones. One row is visible for the group 'myasg', which uses launch template 'mytmp' and has a status of 'Updating capacity...'. The 'Actions' button is highlighted in orange.

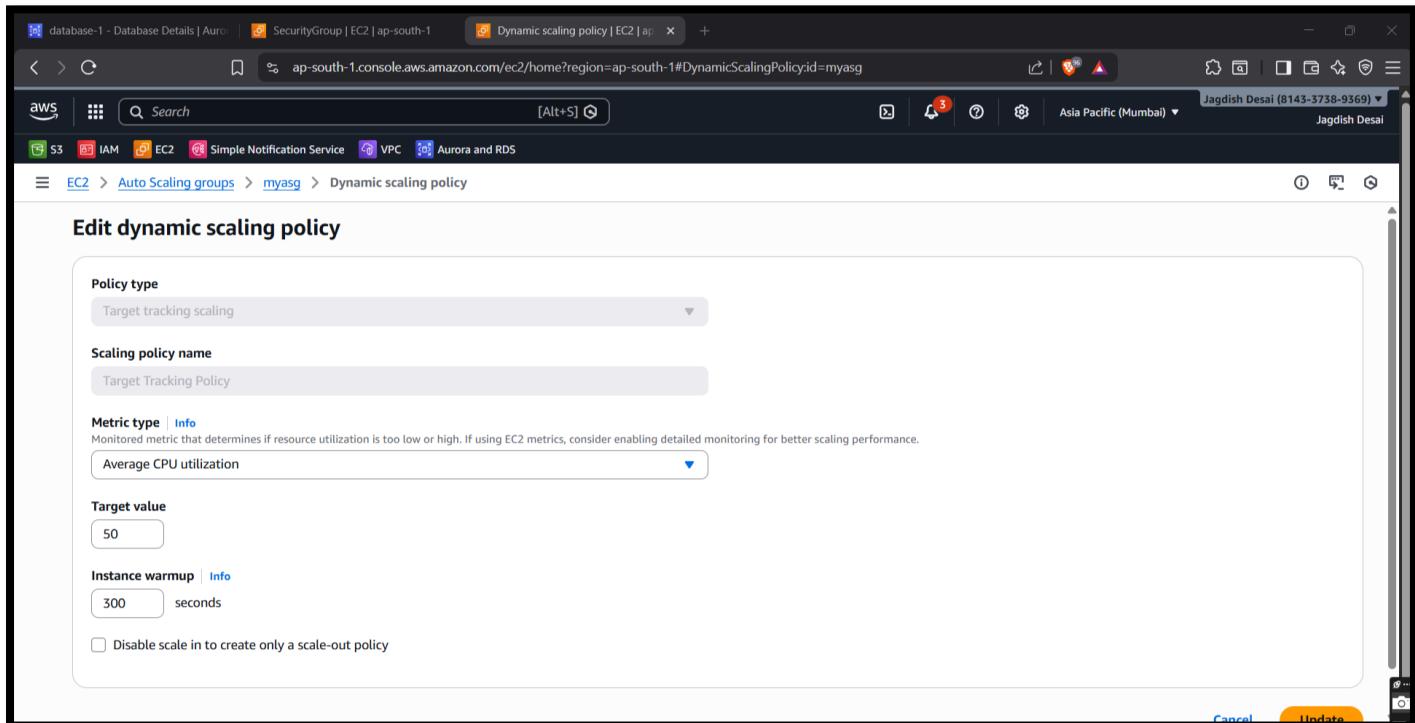
# Scale based on CPU Utilization

## Step 1: Adding Scaling Policy

1. ASG → Automatic scaling
2. Choose Target tracking
3. Metric: Average CPU utilization
4. Target value: 50%
5. Create

 If CPU > 50% → new EC2 added

 If CPU < 50% → EC2 removed



# Our Final work Flow

User → ALB → Target Group → EC2 (ASG)

## This Is created instance by us: ASG launched EC2:

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with categories like Instances, Images, Elastic Block Store, and Network & Security. The main area displays a table titled 'Instances (1) Info' with one row. The row contains the instance ID 'i-0f65fae86aab46c1a', its state 'Running', type 't2.micro', and other details like '2/2 checks passed'. A 'Launch instances' button is at the top right of the table. Below the table, there's a section labeled 'Select an instance'.

## Create CPU Load (SSH into EC2)

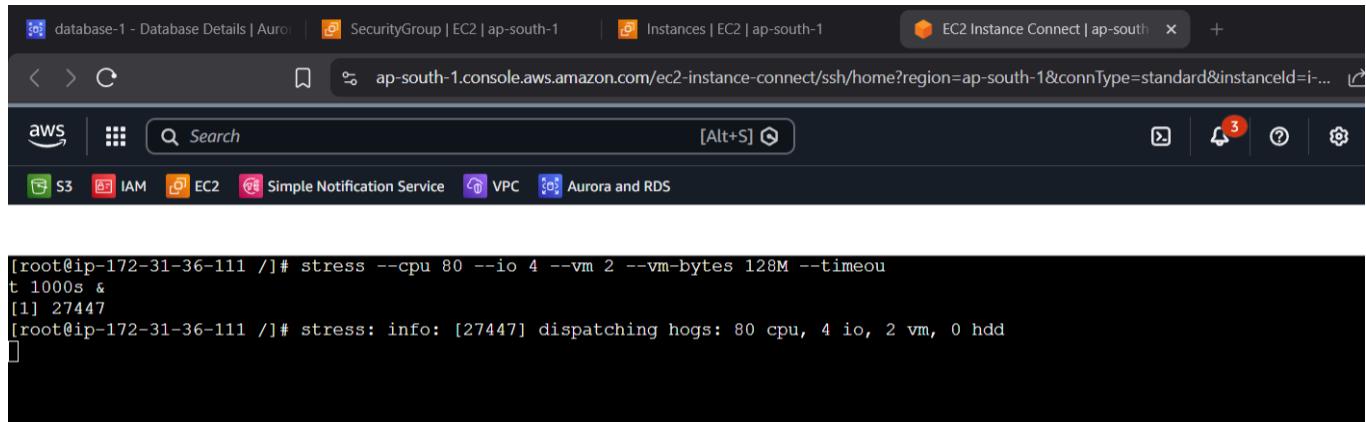
### Yum install stress -y

```
[root@ip-172-31-36-111 ~]# yum install stress
Last metadata expiration check: 0:13:42 ago on Sat Feb 7 22:12:37 2026.
Dependencies resolved.
=====
package      Architecture Version      Repository      Size
=====
Installing:
stress       x86_64      1.0.7-2.amzn2023.0.1      amazonlinux      34 k
=====
Transaction Summary
=====
Install 1 Package

Total download size: 34 k
Installed size: 68 k
Is this ok [y/N]: y
Downloading Packages:
stress-1.0.7-2.amzn2023.0.1.x86_64.rpm      437 kB/s | 34 kB   00:00
Total                                         280 kB/s | 34 kB   00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing :                                                 1/1
  Installing : stress-1.0.7-2.amzn2023.0.1.x86_64          1/1
  Running scriptlet: stress-1.0.7-2.amzn2023.0.1.x86_64    1/1
  Verifying  : stress-1.0.7-2.amzn2023.0.1.x86_64          1/1
=====
WARNING:
  A newer release of "Amazon Linux" is available.
```

# Increasing stress

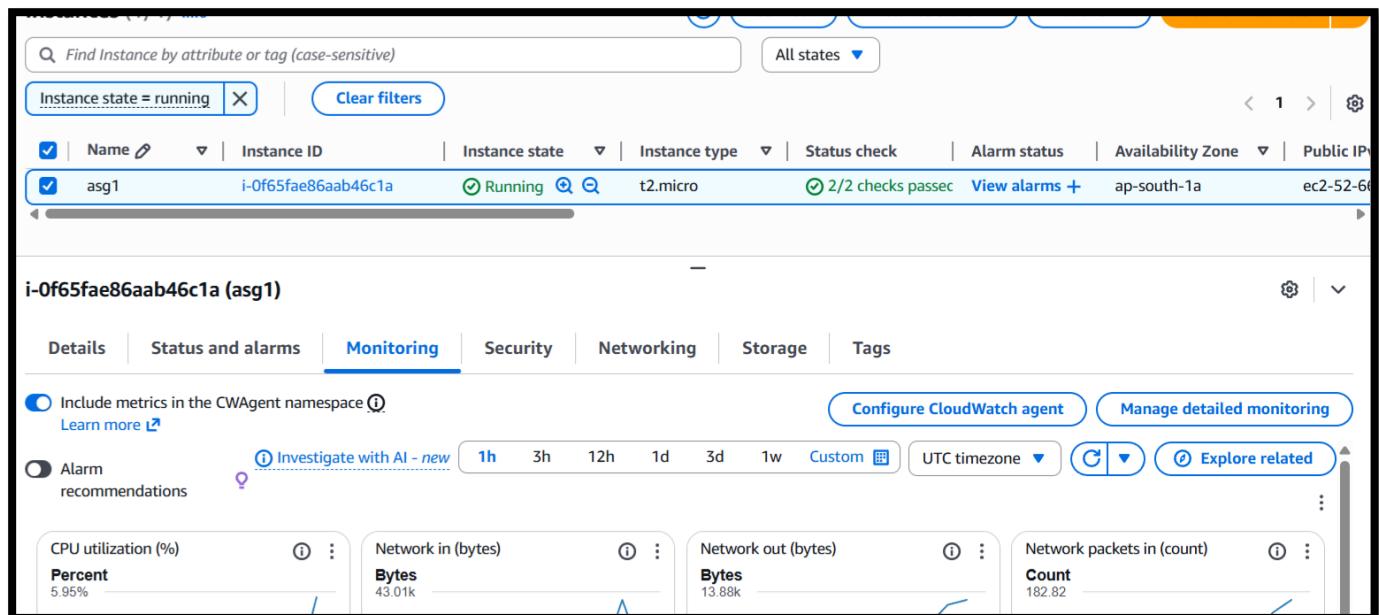
stress --cpu 80 --io 4 --vm 2 --vm-bytes 128M --timeout 1000s &



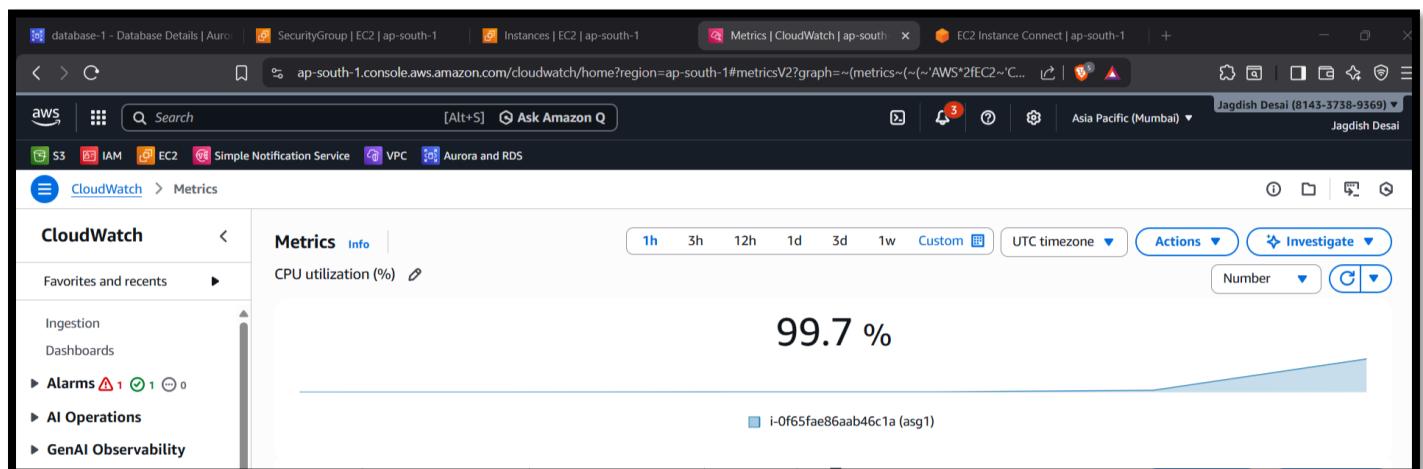
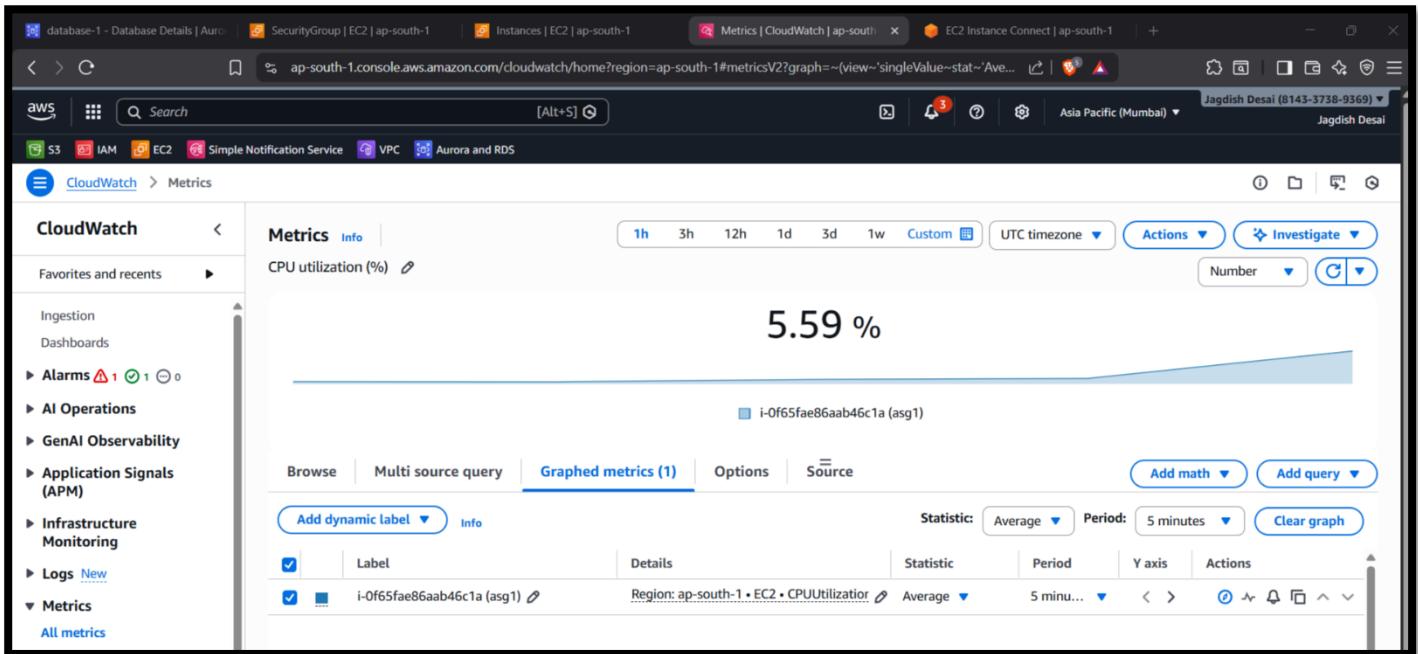
```
[root@ip-172-31-36-111 /]# stress --cpu 80 --io 4 --vm 2 --vm-bytes 128M --timeout 1000s &
[1] 27447
[root@ip-172-31-36-111 /]# stress: info: [27447] dispatching hogs: 80 cpu, 4 io, 2 vm, 0 hdd
```

# Increasing more stress

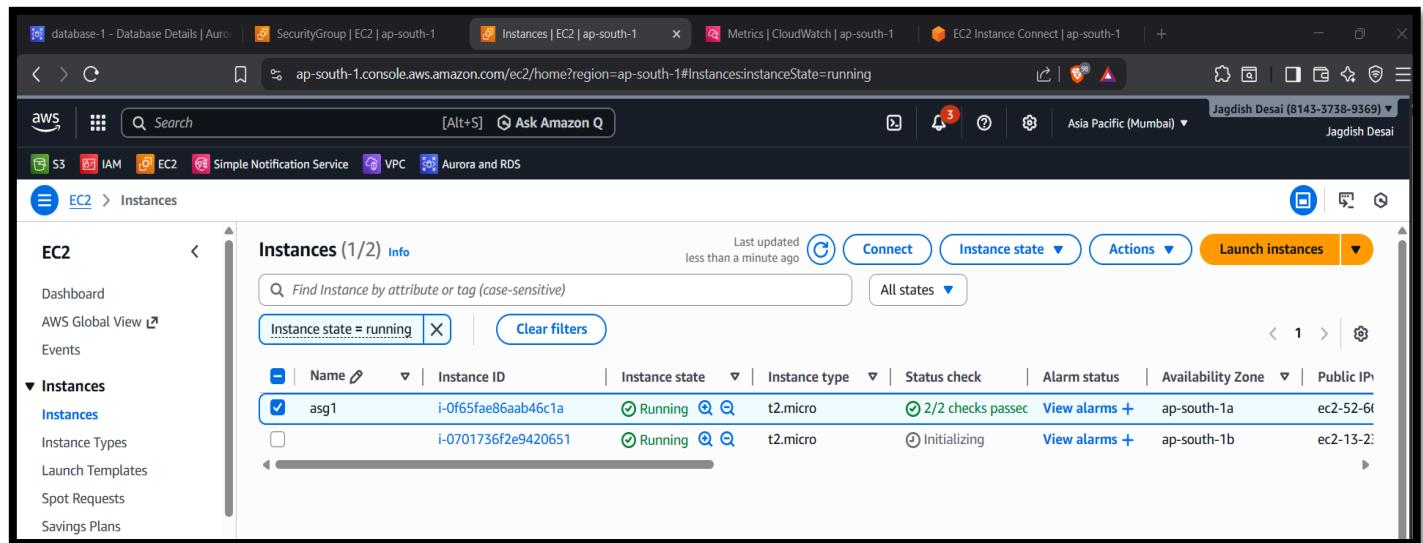
stress --cpu 80 --io 4 --vm 2 --vm-bytes 128M --timeout 1000s &



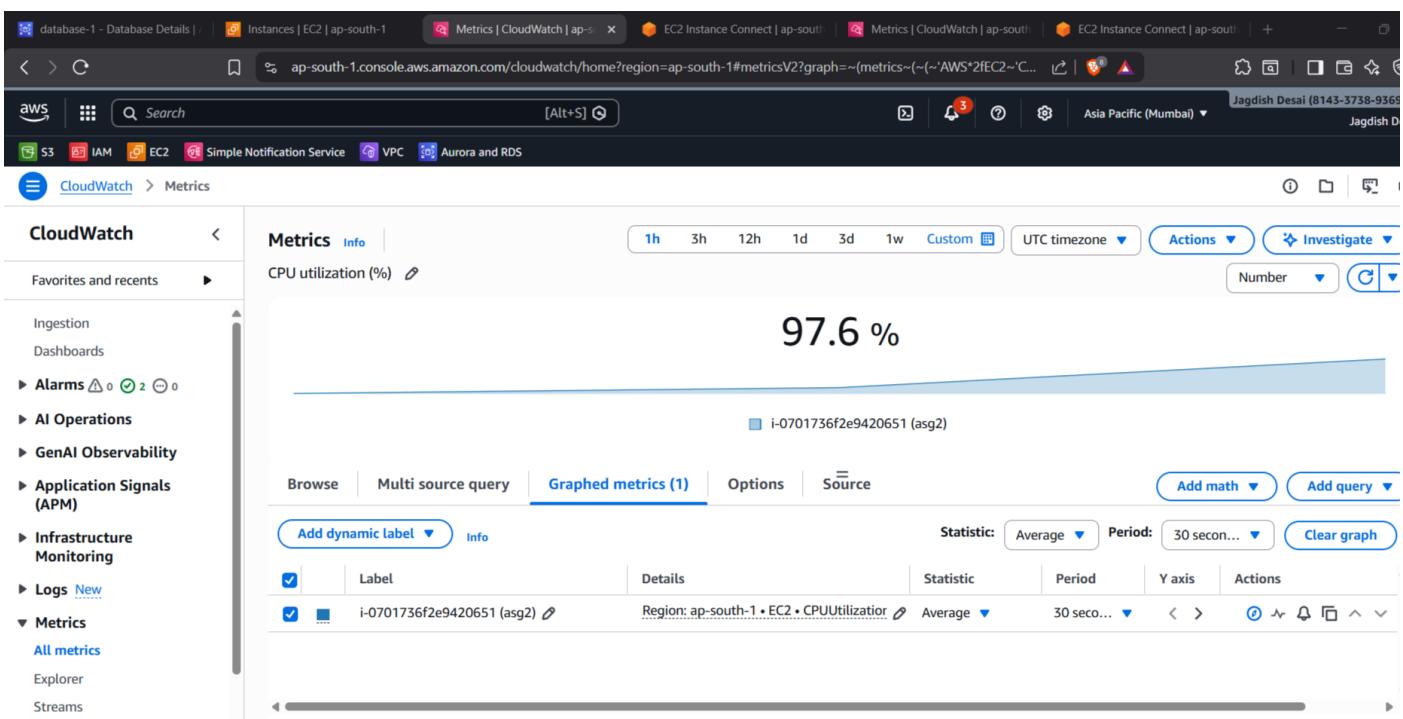
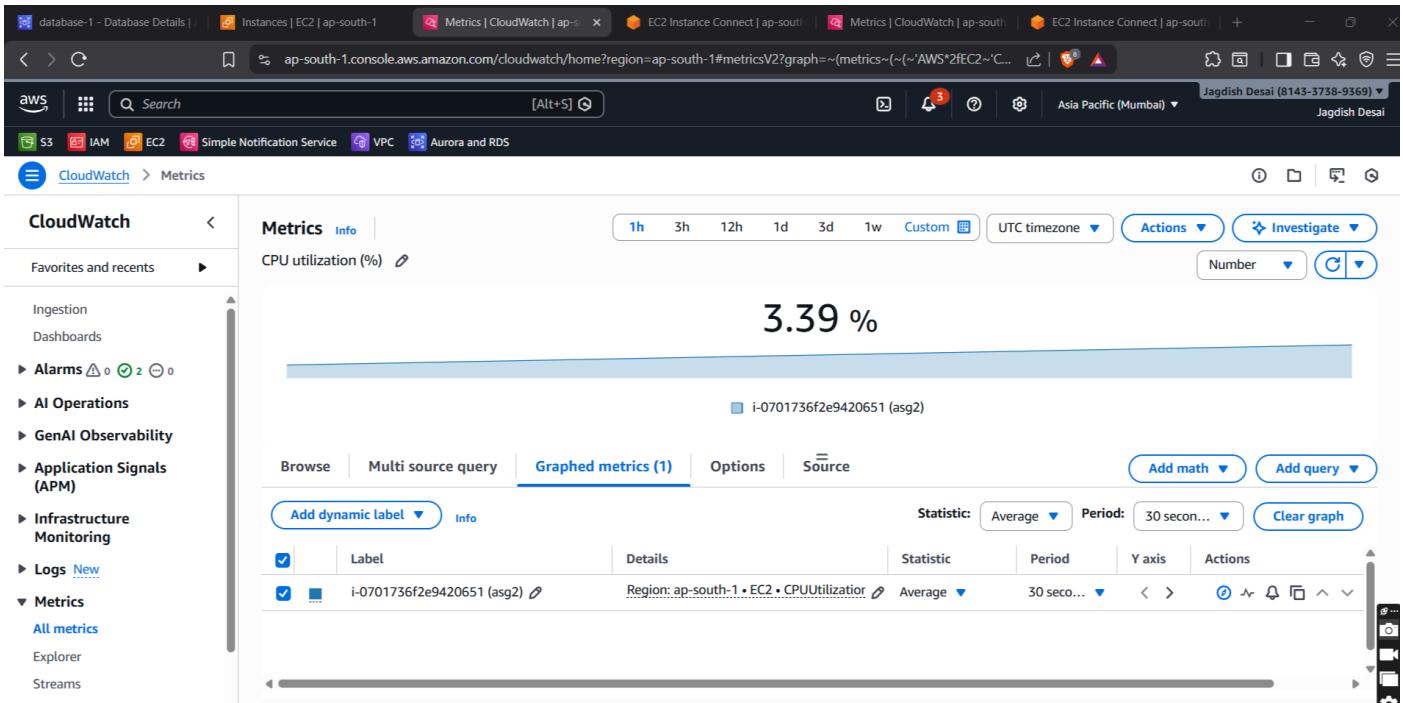
# View Cpu Load: ip- 52.66.118.136



After Increasing Load after above 50% It Crates another instance



# Also we give stress to that asg2 instance: ip- 13.232.12.57



After too much stress new instance will be created: ip-13.127.208.183

The screenshot shows the AWS Management Console with the EC2 Instances page open. The left sidebar shows navigation links for Database, Instances, Auto Scaling groups, Metrics, EC2 Instance Connect, Metrics, CloudWatch Metrics, EC2, VPC, Aurora and RDS, S3, IAM, EC2, Simple Notification Service, and VPC. The main content area displays the 'Instances (1/3) Info' table with the following data:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
asg1	i-0f65fae86aab46c1a	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	ap-south-1a	ec2-52-6t
<input checked="" type="checkbox"/> i-0e3eee16eac46e204	i-0e3eee16eac46e204	Running	t2.micro	Initializing	<a href="#">View alarms</a>	ap-south-1a	ec2-13-1t
asg2	i-0701736f2e9420651	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	ap-south-1b	ec2-13-2t

Below the table, the details for the selected instance (i-0e3eee16eac46e204) are shown. The 'Details' tab is active, displaying the instance summary. The Public IPv4 address is listed as 13.127.208.183.

Checking each instance private ip on browser:

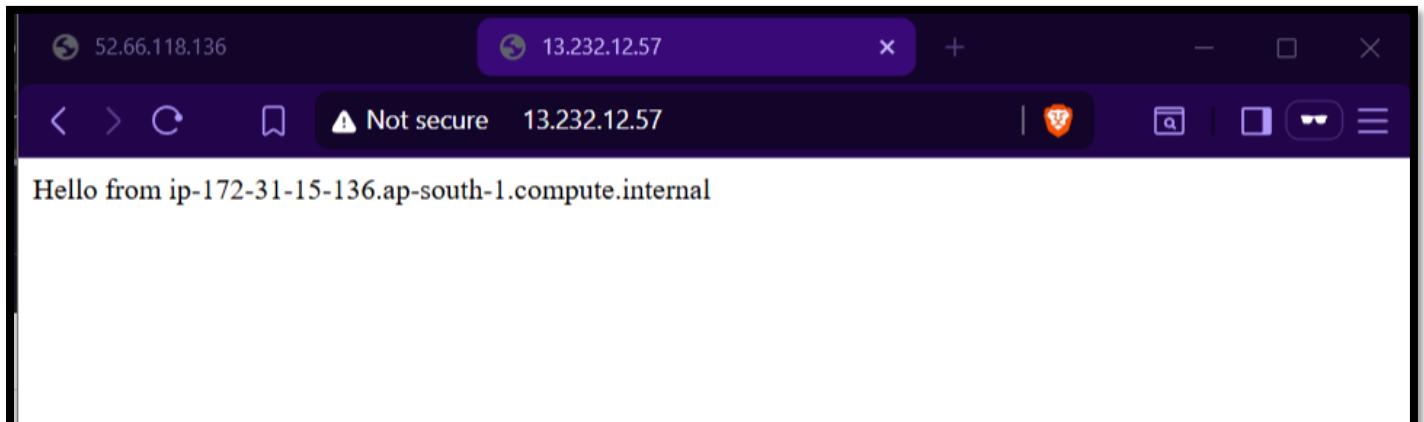
1.ASG-1- private-ip 172.31.36.111

echo "Hello from \$(hostname)" > /var/www/html/index.html

The screenshot shows a web browser window with the following details:

- Address bar: 52.66.118.136
- Content area: Hello from ip-172-31-36-111.ap-south-1.compute.internal
- Header bar: Public IP Viewer, 503 Service Temporarily Unavailable, Not secure, 52.66.118.136

## 2.ASG-2- private-ip 172.31.15.136



## 3.ASG- 3: private-ip 172.31.38.49

