

# Assignment : 11

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

1. Login int AWS account and create a template. Select Ubuntu in quick start.

The screenshot shows the 'Create launch template' page in the AWS Management Console. The 'Launch template name' is 'cse112'. The 'Template version description' is also 'cse112'. The 'Application and OS Images (Amazon Machine Image) - required' section shows the 'Quick Start' tab selected, with 'Ubuntu' chosen from the 'Recent' list. The 'Summary' panel on the right shows the 'Software Image (AMI)' as 'Canonical, Ubuntu, 24.04, amd64...read more' and the 'Virtual server type (instance type)' as 't2.micro'. A 'Free tier' notification is visible at the bottom of the summary panel.

2. Select key pair and select existing security group in Firewall (security groups).

The screenshot shows the 'Create launch template' page in the AWS Management Console, Step 2. The 'Instance type' is 't2.micro'. The 'Key pair (login)' section shows 'key2' selected. The 'Network settings' section shows 'Subnet' as 'Don't include in launch template', 'Firewall (security groups)' as 'Select existing security group', and 'Security groups' as 'sg-407cc6fedf/vj34'. The 'Summary' panel on the right shows the 'Software Image (AMI)' as 'Canonical, Ubuntu, 24.04, amd64...read more' and the 'Virtual server type (instance type)' as 't2.micro'. A 'Free tier' notification is visible at the bottom of the summary panel.

3. After Selecting security group go to the user data and write given cmd. Click launch instance.

The screenshot shows the 'Create launch template' page in the AWS Management Console, Step 3. The 'User data - optional' section has a text area containing the following commands: 

```
#!/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 http://github.com/ruip7407/Repo1.git
cd Repo1
npm install
node index.js
```

 The 'Metadata accessible' section is set to 'Enabled'. The 'Summary' panel on the right shows the 'Software Image (AMI)' as 'Canonical, Ubuntu, 24.04, amd64...read more' and the 'Virtual server type (instance type)' as 't2.micro'. A 'Free tier' notification is visible at the bottom of the summary panel.

4. Create auto scaling group, enter name for auto scaling group, select template, click on next.

The screenshot shows the 'Choose launch template' step in the AWS Management Console. A progress bar on the left indicates the current step. The main content area has a 'Name' field with the value 'tanu03'. Below it, a 'Launch template' section shows a dropdown menu with 't2.micro' selected. To the right, a table lists details for the selected launch template: 'Launch template' (t2.micro), 'Version' (Default (1)), 'Description' (t2.micro), 'AMI ID' (ami-da38da6d5965cf57), 'Key pair name' (key2), 'Launch template' (t2.micro), 'Security groups' (sg-007c0a8a7b704704), 'Instance type' (t2.micro), and 'Request Spot Instances' (No).

5. Now select all availability zones and subnets then click on next.

The screenshot shows the 'Choose instance launch options' step in the AWS Management Console. The progress bar on the left is updated. The 'Instance type requirements' section shows 'Launch template' (t2.micro), 'Version' (Default), and 'Description' (t2.micro). The 'Network' section shows 'VPC' (vpc-0057a4403f7edac1) and 'Availability Zones and subnets' (ap-south-1a | subnet-057c0a8a7b704704, ap-south-1b | subnet-0d6659f0bb1f6c09d, ap-south-1c | subnet-0b7207a726c1a0016).

6. In load balancing select attach to new load balancer, in attach to new load balancer select application load balancer, in load balancer select internet-balancing.

The screenshot shows the 'Integrate with other services - optional' step in the AWS Management Console. The progress bar on the left is updated. The 'Load balancing' section shows 'Attach to a new load balancer' selected. The 'Attach to a new load balancer' section shows 'Application Load Balancer' selected. The 'Load balancer name' field has the value 'tanu2B-1'. The 'Load balancer scheme' section shows 'Internet-facing' selected. The 'Network mapping' section shows 'VPC' (vpc-0057a4403f7edac1) and 'Availability Zones and subnets' (ap-south-1a | subnet-0d6659f0bb1f6c09d).

7. In add 1000 in port, select new target group name. In health check period add 200 seconds.

**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: 1000

Default routing (forward to): Create a target group

New target group name: tanuCS-1

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  
VPC Lattice will not manage your Auto Scaling group's network access and connectivity with other services.

☐ Attach to VPC Lattice service  
Incoming requests associated with specified VPC Lattice target groups will be routed to your Auto Scaling group.

Create new VPC Lattice service

**Application Recovery Controller (ARC) zonal shift - new**  
During an Availability Zone impairment, target instance launches towards other healthy Availability Zones.

☐ Enable zonal shift  
New instance launches will be rerouted towards healthy Availability Zones until the zonal shift is cancelled.

**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.

8. In desired capacity write 2. In scaling limit give min 2 and max 3.

**Configure group size and scaling - optional**  
Define your group's desired capacity and scaling limits. You can optionally add automatic scaling to adjust the size of your group.

**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: Units (number of instances)

Desired capacity: 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  
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: Average CPU utilization

9. In target value as 50, instance warmup as 200 sec. Then click on create auto scaling group.

Target value: 50

Instance warmup: 200 seconds

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

**Instance maintenance policy**  
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.

Choose a replacement behavior depending on your availability requirements

**Minimal baseline**  
☒ No policy  
For rebalancing events, new instances will launch before terminating others. For all other events, instances terminate and launch at the same time.

**Flexible availability**  
☐ Launch before terminating  
Launch new instances and wait for them to be ready before terminating others. This allows you to go above your desired capacity by a given percentage and may temporarily increase costs.

**Control costs**  
☐ Terminate and launch  
Terminate and launch instances at the same time. This allows you to go below your desired capacity by a given percentage and may temporarily reduce availability.

**Flexible**  
☐ Custom behavior  
Set custom values for the minimum and maximum amount of available capacity. This gives you greater flexibility in setting how far below and over your desired capacity EC2 Auto Scaling goes when replacing instances.

**Additional capacity settings**  
Capacity Reservation preference  
Select whether you want Auto Scaling to launch instances into an existing Capacity Reservation or Capacity Reservation resource group.

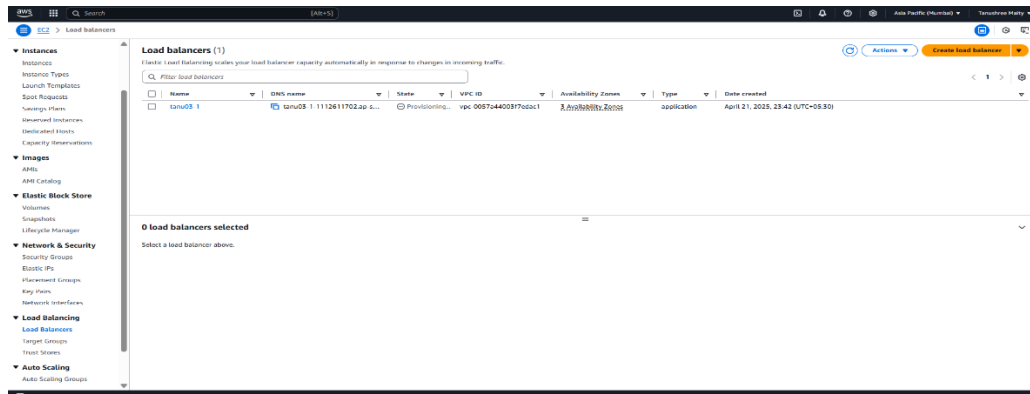
☒ Default  
Auto Scaling uses the Capacity Reservation preference from your launch template.

☐ None  
Instances will not be launched into a Capacity Reservation.

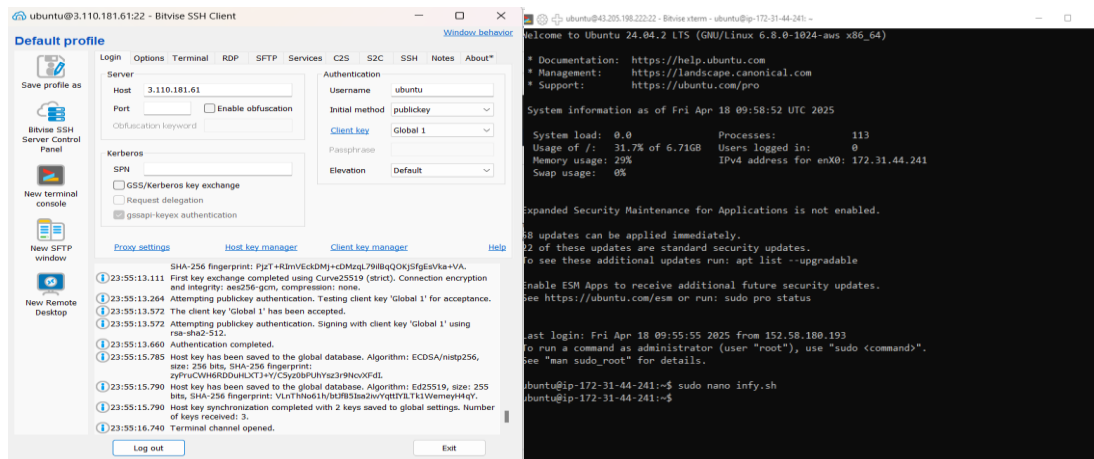
☐ Capacity Reservations only  
Instances will only be launched into a Capacity Reservation. If capacity isn't available, the instances fail to launch.

☐ Capacity Reservations first  
Instances will attempt to launch into a Capacity Reservation first. If capacity isn't available, instances will run in On-Demand capacity.

10. Go to load balance copy DNS name and run it in another tab.



11. Now open bitwise and copy Public IPv4 adress from any one instance and paste it in port. Open terminal and write the given command, it will start a loop.



12. In security we can see CPU utilization graph

