



## **Placement Empowerment Program**

### ***Cloud Computing and DevOps Centre***

Implement Auto-scaling in the CloudSet up an autoscaling group for your cloud VMs to handle variable workloads.

Name: Jason Peniel Raj. S

Department: CSE

## Introduction

As modern applications face varying workloads, ensuring optimal performance and availability is critical. Auto Scaling, a feature provided by cloud platforms like AWS, dynamically adjusts computing resources in response to demand changes. This Proof of Concept (PoC) demonstrates how to set up an Auto Scaling Group (ASG) for virtual machines (VMs) to handle fluctuating workloads effectively. It explores defining launch configurations, setting scaling policies, and testing automatic scaling based on CPU usage.

## Overview

This PoC focuses on implementing a scalable architecture using AWS Auto Scaling Groups. The workflow includes:

- 1. Defining a Launch Template:** Configuring virtual machines (VMs) with required specifications like instance type, AMI, key pairs, and security groups.
- 2. Creating an Auto Scaling Group:** Setting initial group size and linking it to the launch template to manage instances dynamically.
- 3. Configuring Scaling Policies:** Setting up metrics like CPU utilization to trigger scaling actions (e.g., scaling up during high CPU usage).

**4. Testing Auto Scaling:** Simulating high CPU load to verify that the ASG launches additional instances to handle demand.

This PoC will demonstrate the reliability, flexibility, and cost efficiency of dynamic scaling in a cloud environment.

## Objective

The primary objective of this PoC is to:

1. Implement an **Auto Scaling Group (ASG)** to manage workloads effectively.
2. Define and configure a **Launch Template** for virtual machines.
3. Set up and test **scaling policies** based on predefined metrics, such as CPU utilization.
4. Validate the scaling process by simulating real-world scenarios (e.g., high CPU usage).

By completing this PoC, the goal is to gain hands-on experience with Auto Scaling and to understand its importance in ensuring application availability and cost management.

## Importance

- 1. Improved Application Availability:** Auto Scaling ensures that applications remain available even during traffic spikes by automatically adding more VMs to meet demand.
- 2. Cost Optimization:** It dynamically reduces the number of VMs during low traffic periods, minimizing unnecessary costs.

**3. Efficient Resource Utilization:** By scaling resources based on actual demand, Auto Scaling prevents over-provisioning and underutilization.

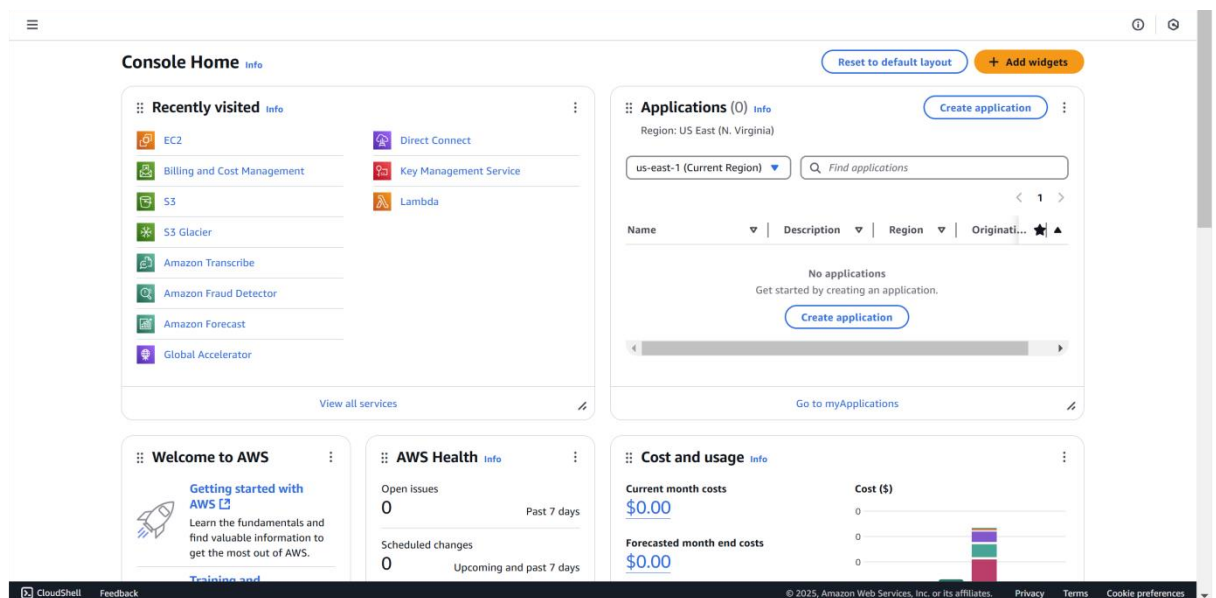
**4. Resilience to Failures:** Auto Scaling can replace unhealthy instances automatically, ensuring consistent application performance.

**5. Real-World Relevance:** The ability to manage variable workloads is a critical skill in cloud computing and aligns with industry practices.

## Step-by-Step Overview Step

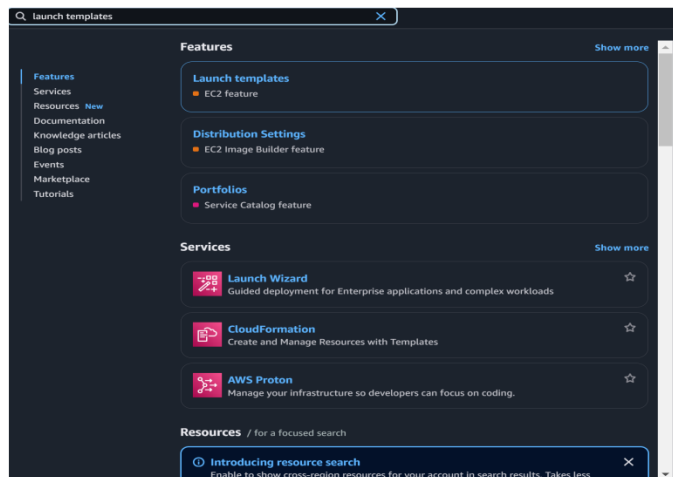
1:

1. Go to [AWS Management Console](#).
2. Enter your username and password to log in.



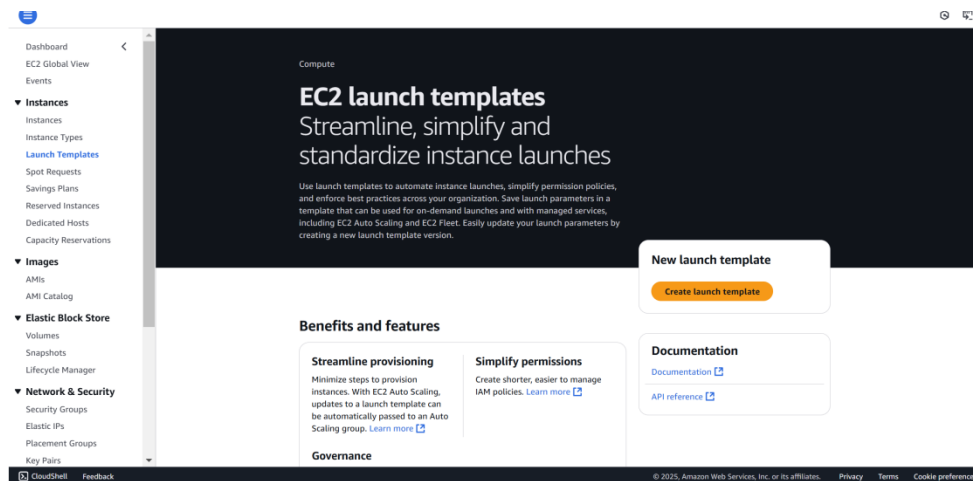
Step 2:

Search for Launch Templates.



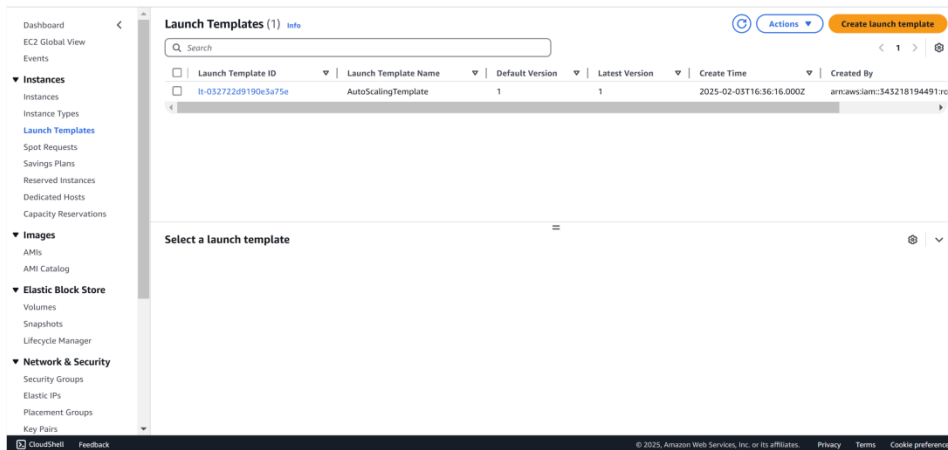
## Step 3:

Click on the Create launch template.



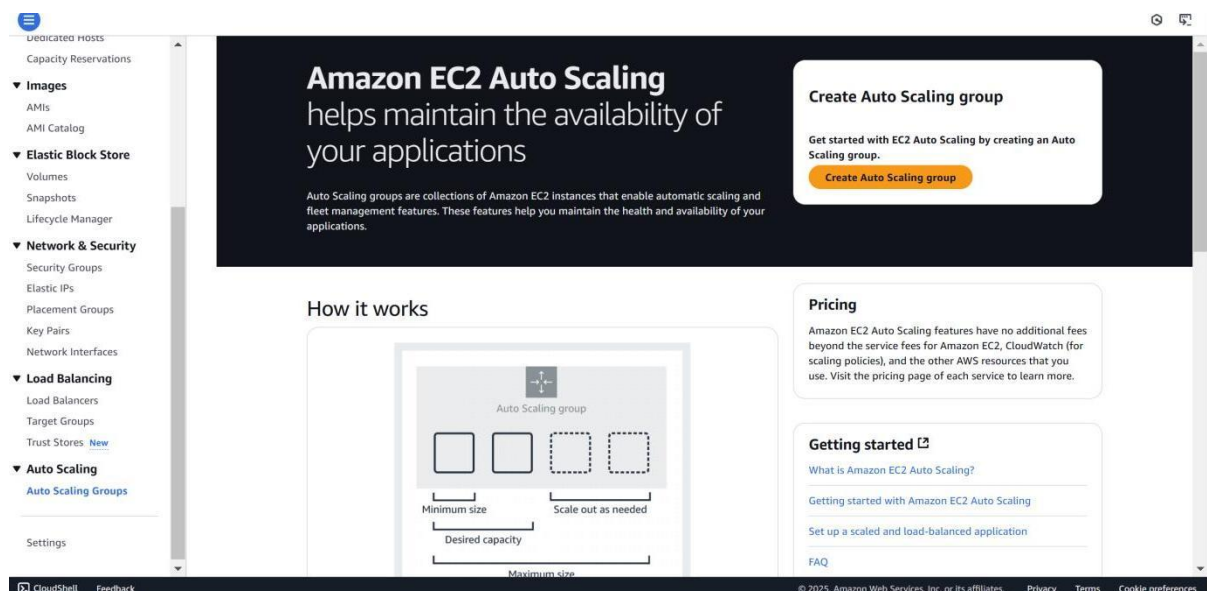
## Step 4:

Create a **Launch Template** named **AutoScalingTemplate** using an **Amazon Machine Image (AMI)** like Amazon Linux 2 or any default image, and choose an **instance type** such as **t2.micro** for free-tier eligibility. Select an **existing key pair** (or create a new one) to enable SSH access, and configure a **security group** that allows HTTP (port 80) and SSH (port 22). Once all details are filled out, click **Create launch template** to complete the setup.



## Step 5:

Go to the **EC2 Dashboard** . On the left sidebar, click on **Auto Scaling Groups**. Click on **Create an Auto Scaling group**.



## Step 6:

**Auto Scaling group name:** Give it a name (e.g., MyAutoScalingGroup).

**Launch Template:** Select the launch template you created earlier (AutoScalingTemplate).

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1: Choose launch template (selected)  
 Step 2: Choose instance launch options  
 Step 3 - optional: Integrate with other services  
 Step 4 - optional: Configure group size and scaling  
 Step 5 - optional: Add notifications  
 Step 6 - optional: Add tags  
 Step 7: Review

### Choose launch template [Info](#)

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group.

**Name**  
 Auto Scaling group name  
 Enter a name to identify the group.  
  
 Must be unique to this account in the current Region and no more than 255 characters.

**Launch template [Info](#)**  
 ⓘ For accounts created after May 31, 2023, the EC2 console only supports creating Auto Scaling groups with launch templates. Creating Auto Scaling groups with launch configurations is not recommended but still available via the CLI and API until December 31, 2023.

**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.  
 [Create a launch template](#)

**Version**  
 [Create a launch template version](#)

**Description**  
 -

**Launch template**  
[AutoScalingTemplate](#)  
 lt-032722d9190e3a75e

**Instance type**  
 t2.micro

**AMI ID**  
 -

**Security groups**  
 -

[Request Spot instances](#)

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

## Step 7:

**VPC and Subnets:** Choose your **VPC** (it's fine to use the default one). Select at least two subnets in different Availability Zones (this ensures high availability).

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 2: Choose instance launch options (selected)  
 Step 3 - optional: Integrate with other services  
 Step 4 - optional: Configure group size and scaling  
 Step 5 - optional: Add notifications  
 Step 6 - optional: Add tags  
 Step 7: Review

### Instance type requirements [Info](#)

[Override launch template](#)

You can keep the same instance attributes or instance type from your launch template, or you can choose to override the launch template by specifying different instance attributes or manually adding instance types.

**Launch template**  
[AutoScalingTemplate](#)  
 lt-032722d9190e3a75e

**Version**  
 Default

**Description**  
 -

**Instance type**  
 t2.micro

### 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.  
  
 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.  
 [Create a subnet](#)

172.31.80.0/20 Default

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

## Step 8:

For this PoC leave the next settings as default and click next .

EC2 > Auto Scaling groups > Create Auto Scaling group

CloudShellFeedback

© 2025, Amazon Web Services, Inc. or its affiliates. PrivacyTermsCookie preferences

Step 1Choose launch template

Step 2Choose instance launch options

Step 3 - optionalIntegrate with other services

Step 4 - optionalConfigure group size and scaling

Step 5 - optionalAdd notifications

Step 6 - optionalAdd tags

Step 7Review

Integrate with other services - optional

Use a load balancer to distribute network traffic across multiple servers. Enable service-to-service communications with VPC Lattice. Shift resources away from impaired Availability Zones with zonal shift. You can also customize health check replacements and monitoring.

Load balancing

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

Attach to an existing load balancer

Attach to a new load balancer

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.

No VPC Lattice service

Attach to 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

Step 1Choose launch template

Step 2Choose instance launch options

Step 3 - optionalIntegrate with other services

Step 4 - optionalConfigure group size and scaling

Step 5 - optionalAdd notifications

Step 6 - optionalAdd tags

Step 7Review

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

Choose the unit of measurement for the desired capacity value. vCPUs and Memory(GiB) are only supported for mixed instances groups configured with a set of instance attributes.

Units (number of instances)

Desired capacity

Specify your group size.

1

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

Max desired capacity

1

1

Equal or less than desired capacity

Equal or greater than desired capacity

Automatic scaling - optional

Choose whether to use a target tracking policy

No scaling policies

Target tracking scaling policy

Step 1Choose launch template

Step 2Choose instance launch options

Step 3 - optionalIntegrate with other services

Step 4 - optionalConfigure group size and scaling

Step 5 - optionalAdd notifications

Step 6 - optionalAdd tags

Step 7Review

Add notifications - optional

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

CloudShellFeedback

© 2025, Amazon Web Services, Inc. or its affiliates. PrivacyTermsCookie preferences

Step 9:



Review all the settings you've configured. Once satisfied, click **Create Auto Scaling Group**.

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1: Choose launch template  
Step 2: Choose instance launch options  
Step 3 - optional: Integrate with other services  
Step 4 - optional: Configure group size and scaling  
Step 5 - optional: Add notifications  
Step 6 - optional: Add tags  
Step 7: Review

### Review

**Step 1: Choose launch template**

**Group details**

**Auto Scaling group name**  
MyAutoScalingGroup

**Launch template**

Launch template	Version	Description
<a href="#">AutoScalingTemplate</a> lt-032722d9190e3a75e	Default	

**Step 2: Choose instance launch options**

**Network**

**VPC**  
[vpc-0f36f0944c12862e5](#)

**Availability Zones and subnets**

Availability Zone	Subnet	Subnet CIDR range
us-east-1a	<a href="#">subnet-0f01dae09f48d5f64</a>	172.31.80.0/20

**Availability Zone distribution**  
Balanced best effort

EC2 > Auto Scaling groups

Auto Scaling groups (1)

Search your Auto Scaling groups

Launch configurations Launch templates Actions Create Auto Scaling group

	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
	<a href="#">MyAutoScalingGroup</a>	<a href="#">AutoScalingTemplate</a>   Version Default	0	Updating capacity...	1	1	1	us-east-1a

0 Auto Scaling groups selected

EC2 > Auto Scaling groups > MyAutoScalingGroup

### MyAutoScalingGroup

**MyAutoScalingGroup Capacity overview**

[arn:aws:autoscaling:us-east-1:343218194491:autoScalingGroup:a52c8890-07bc-41c8-a7a8-2d6da899fa7d:autoScalingGroupName/MyAutoScalingGroup](#)

<b>Desired capacity</b> 1	<b>Scaling limits (Min - Max)</b> 1 - 1	<b>Desired capacity type</b> Units (number of instances)	<b>Status</b> -
------------------------------	--	---	--------------------

**Date created**  
Mon Feb 03 2025 22:12:12 GMT+0530 (India Standard Time)

Details Integrations - new Automatic scaling Instance management Instance refresh Activity Monitoring

**Launch template**

<b>Launch template</b> <a href="#">lt-032722d9190e3a75e</a> AutoScalingTemplate	<b>AMI ID</b> <a href="#">ami-0c614dee691cbbf37</a>	<b>Instance type</b> t2.micro	<b>Owner</b> arn:aws:iam::343218194491:root
<b>Version</b> Default	<b>Security groups</b> -	<b>Security group IDs</b> -	<b>Create time</b> Mon Feb 03 2025 22:06:16 GMT+0530 (India Standard Time)
<b>Description</b> -	<b>Storage (volumes)</b> -	<b>Key pair name</b> newkey	<b>Request Spot Instances</b> No

[View details in the launch template console](#)

## Step 10:

### Testing Auto Scaling :

#### Important Note

##### Do Not Perform This Test If You Want to Avoid Costs:

1. Launching and running additional EC2 instances will incur charges beyond the AWS Free Tier.
2. Simulating high CPU usage and triggering scaling may increase costs temporarily due to additional resource allocation.

#### 1. Simulate High CPU Usage on an EC2 Instance

Connect to one of your EC2 instances in the Auto Scaling Group using SSH.

Run a command to create artificial CPU load. For example:

```
sudo yum install -y stress
```

```
stress --cpu 2 --timeout 300
```

This command will utilize 2 CPU cores for 5 minutes, simulating high CPU usage.

#### 2. Monitor Scaling Activities

Navigate to the **AWS Management Console > EC2 Dashboard > Auto Scaling Groups**.

Select your Auto Scaling Group and go to the **Activity History** tab.

Check if a new instance is being launched based on your scaling policy (e.g., CPU utilization exceeding 50%).

### 3. Terminate the Stress Test

Once testing is done, stop the CPU load by pressing Ctrl+C in the terminal or by terminating the stress process.

### 4. Verify Scaling Down

After the CPU usage drops, monitor the Auto Scaling Group again to confirm that unnecessary instances are terminated, returning to the desired capacity.

## Outcome

This Proof of Concept (PoC) aimed to implement Auto Scaling in AWS to dynamically manage EC2 instances based on workload demand, ensuring efficient resource utilization and cost-effectiveness. Here's the outcome of the PoC:

1. **Launch Template and Auto Scaling Group Setup:** Successfully created a launch template and configured an Auto Scaling Group with scaling policies to dynamically manage EC2 instances based on workload.
2. **Dynamic Scaling and Monitoring:** Implemented scaling policies triggered by CPU utilization and verified automatic scaling actions using the Auto Scaling Group's Activity History.
3. **Cost Awareness:** Highlighted potential costs of running additional instances beyond the AWS Free Tier during testing and ensured resource usage was optimized.