



**St. JOSEPH'S**  
GROUP OF INSTITUTIONS  
OMR, CHENNAI - 119



## Placement Empowerment Program

*Cloud Computing and DevOps Centre*

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

Name: Samiya Afreen J

Department: CSE



**St. JOSEPH'S**  
COLLEGE OF ENGINEERING



**St. JOSEPH'S**  
INSTITUTE OF TECHNOLOGY

AUTONOMOUS INSTITUTIONS, AFFILIATED TO ANNA UNIVERSITY

# 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.

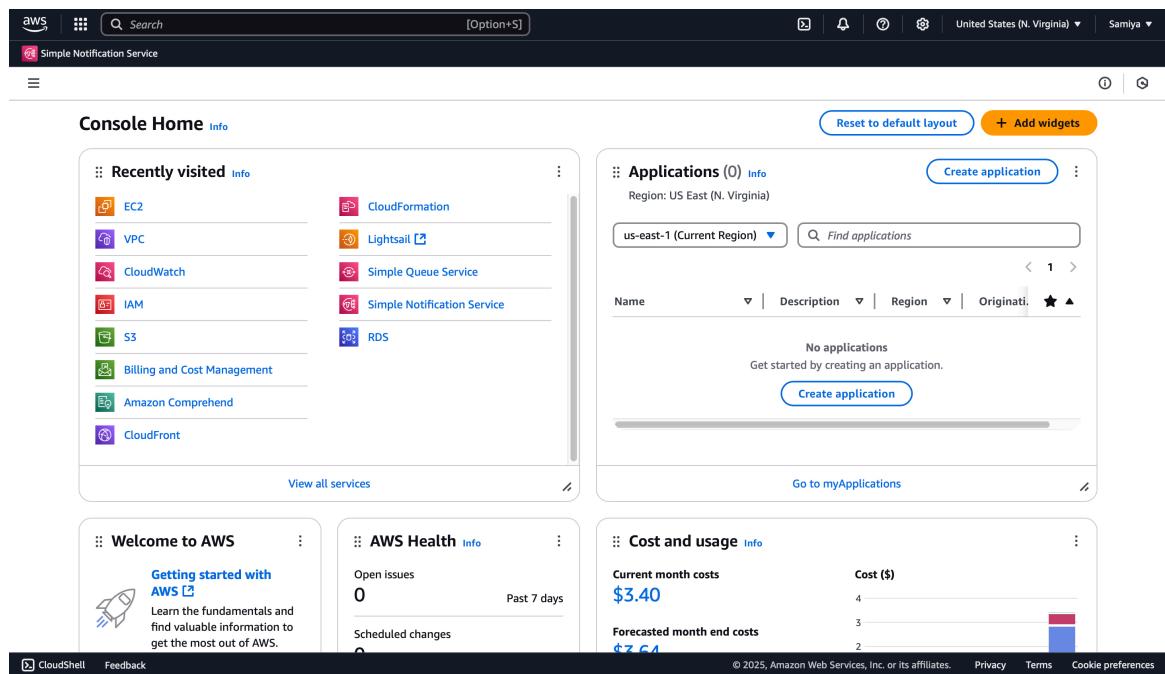
# Importance

- 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

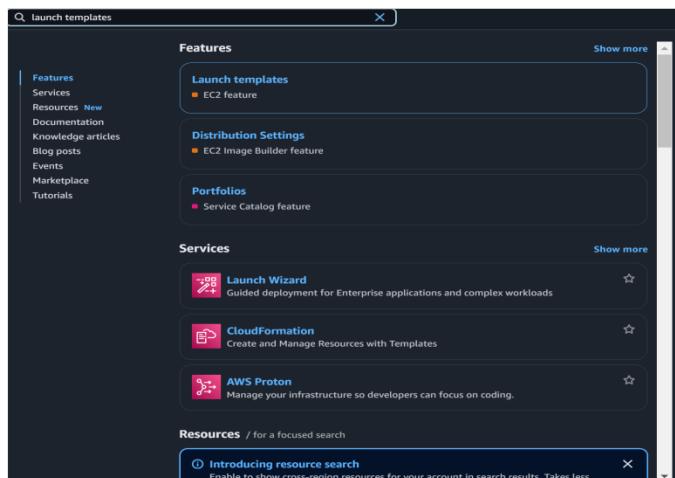
## Step 1:

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



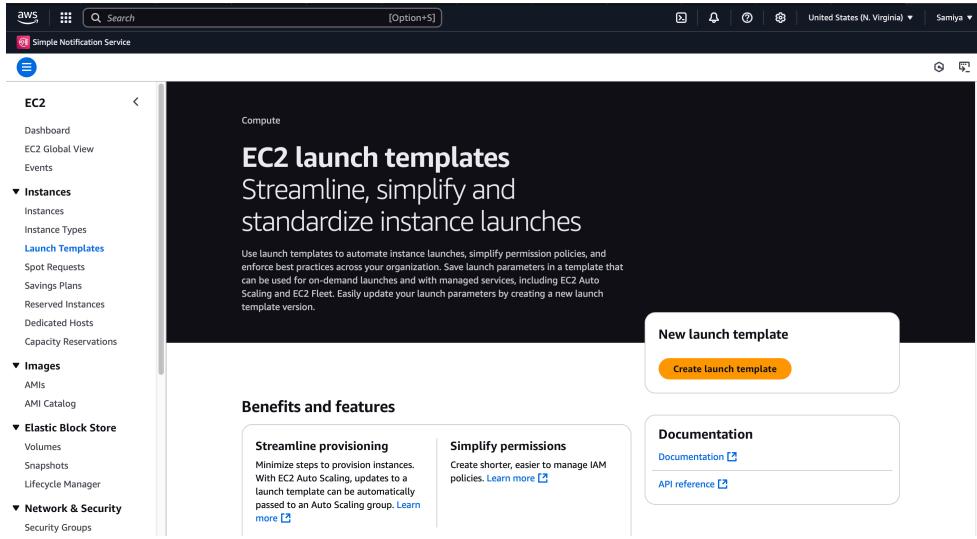
## Step 2:

Search for Launch Templates.



## Step 3:

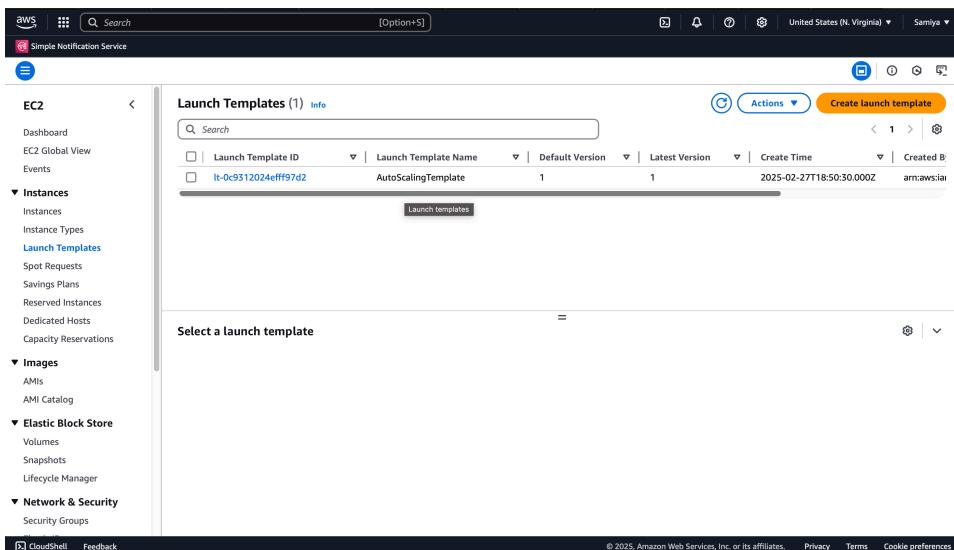
Click on the Create launch template.



The screenshot shows the AWS EC2 Launch Templates page. On the left, there's a navigation sidebar with options like Dashboard, EC2 Global View, Events, Instances, Instance Types, Launch Templates (which is selected), Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, and more. The main content area has a title 'EC2 launch templates' and a subtitle 'Streamline, simplify and standardize instance launches'. It includes a paragraph about using launch templates to automate instance launches. Below this, there's a 'New launch template' button with an orange 'Create launch template' button underneath it. To the right, there are sections for 'Benefits and features' (Streamline provisioning, Simplify permissions), 'Documentation' (Documentation, API reference), and a 'Launch templates' table with one entry: Launch Template ID: lt-0x9512024effff97d2, Launch Template Name: AutoScalingTemplate, Default Version: 1, Latest Version: 1, Create Time: 2025-02-27T18:50:30.000Z, and Created By: arnaws:sa: [redacted].

## 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.



The screenshot shows the AWS EC2 Launch Templates list page. The left sidebar is identical to the previous screenshot. The main area displays a table titled 'Launch Templates (1) Info' with one row. The columns are: Launch Template ID (lt-0x9512024effff97d2), Launch Template Name (AutoScalingTemplate), Default Version (1), Latest Version (1), Create Time (2025-02-27T18:50:30.000Z), and Created By (arnaws:sa: [redacted]). There are also 'Actions' and 'Create launch template' buttons at the top of the table. Below the table, there's a section titled 'Select a launch template' with a dropdown menu.

## Step 5:

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

The screenshot shows the AWS EC2 Auto Scaling Groups page. The left sidebar has a navigation menu with sections like Images, AMIs, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling (with Auto Scaling Groups selected). The main content area features a large banner with the text "Amazon EC2 Auto Scaling helps maintain the availability of your applications". Below the banner, there's a diagram titled "How it works" showing an "Auto Scaling group" containing four squares labeled "Desired capacity". One square is solid, one is dashed, and two are dotted, with arrows indicating scaling. To the right of the diagram are sections for "Pricing" and "Getting started". The "Pricing" section notes no additional fees beyond service fees. The "Getting started" section includes links for "What is Amazon EC2 Auto Scaling?", "Getting started with Amazon EC2 Auto Scaling", and "Set up a scaled and load-balanced application". At the bottom right, there's a "Create Auto Scaling group" button.

## Step 6:

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

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

The screenshot shows the "Create Auto Scaling group" wizard at Step 3. The left sidebar lists optional steps: 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), and Step 7 (Review). The main form has two sections: "Name" and "Launch template". In the "Name" section, the "Auto Scaling group name" is set to "MyAutoScalingGroup". In the "Launch template" section, the "Launch template" dropdown is set to "AutoScalingTemplate". Other fields include "Version" (Default (1)), "Description" (empty), "AMI ID" (ami-02a53b0d62d37a757), "Key pair name" (new), "Launch template" (AutoScalingTemplate), "Security groups" (empty), "Instance type" (t2.micro), and "Request Spot Instances" (No). A note at the bottom of the "Launch template" section states: "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."

## 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).

The screenshot shows the 'Network' configuration step of the AWS EC2 Auto Scaling group creation wizard. It includes sections for VPC selection, Availability Zones and subnets, and Availability Zone distribution.

**VPC:** Choose the VPC that defines the virtual network for your Auto Scaling group. A dropdown menu shows 'vpc-06260ec9a9ae0a454' (172.31.0.0/16 Default).

**Availability Zones and subnets:** Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC. A dropdown menu shows 'us-east-1a | subnet-0e433751635b9ed6' (172.31.32.0/20 Default) and 'us-east-1b | subnet-0fce724e20759806' (172.31.0.0/20 Default). A 'Create a subnet' button is also present.

**Availability Zone distribution - new:** Auto Scaling automatically balances instances across Availability Zones. If launch failures occur in a zone, select a strategy.

- Balanced best effort**: If launches fail in one Availability Zone, Auto Scaling will attempt to launch in another healthy Availability Zone.
- Balanced only**: If launches fail in one Availability Zone, Auto Scaling will continue to attempt to launch in the unhealthy Availability Zone to preserve balanced distribution.

At the bottom, there are buttons for 'Cancel', 'Skip to review', 'Previous', and 'Next'. The 'Next' button is highlighted in orange.

## Step 8:

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

The screenshot shows the 'Integrate with other services - optional' step of the AWS EC2 Auto Scaling group creation wizard.

**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**: 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.

**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.

**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 redirected towards healthy Availability Zones until the zonal shift is canceled.

At the bottom, there are buttons for 'CloudShell', 'Feedback', '© 2025, Amazon Web Services, Inc. or its affiliates.', 'Privacy', 'Terms', and 'Cookie preferences'.

The screenshot shows the 'Configure group size and scaling - optional' step of the Auto Scaling group creation wizard. On the left, a vertical navigation bar lists steps 1 through 7, with step 4 highlighted. The main area contains sections for 'Group size' and 'Scaling'. In the 'Group size' section, 'Desired capacity type' is set to 'Units (number of instances)', and the 'Desired capacity' field is set to 1. In the 'Scaling' section, 'Min desired capacity' and 'Max desired capacity' are both set to 1. Below these, there's a note about automatic scaling and options for target tracking policies.

The screenshot shows the 'Add notifications - optional' step of the Auto Scaling group creation wizard. The left navigation bar highlights step 5. The main area has a heading 'Add notification' with a note about sending SNS topics. It includes a 'Skip to review' button and navigation buttons for 'Cancel', 'Skip to review', 'Previous', and 'Next'.

## Step 9:

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

AWS | Search [Option+S] | United States (N. Virginia) | Samiya

Simple Notification Service

EC2 > Auto Scaling groups > Create Auto Scaling group

**Step 1**

- Choose launch template
- Choose instance launch options
- Integrate with other services
- Configure group size and scaling
- Add notifications
- Add tags
- Review

**Review** [Info](#)

**Step 1: Choose launch template**

**Group details**

Auto Scaling group name: MyAutoScalingGroup

**Launch template**

Launch template	Version	Description
AutoScalingTemplate <a href="#">Edit</a>	Default	

**Step 2: Choose instance launch options**

**Network**

VPC: vpc-06260e9a9ae0a454 [Edit](#)

**Availability Zones and subnets**

Availability Zone	Subnet	Subnet CIDR range
us-east-1a	subnet-0e433751635b9ed6 <a href="#">Edit</a>	172.31.32.0/20

**Availability Zone distribution**

Balanced load across us-east-1a

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

AWS | Search [Option+S] | United States (N. Virginia) | Samiya

Simple Notification Service

EC2 > Auto Scaling groups

**Auto Scaling groups (1) [Info](#)**

[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...
MyAutoScalingGroup	AutoScalingTemplate   Version Default	0	Updating capacity...	1	1	1	us-east-1a

0 Auto Scaling groups selected

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

AWS | Search [Option+S] | United States (N. Virginia) | Samiya

Simple Notification Service

EC2 > Auto Scaling groups > MyAutoScalingGroup

### MyAutoScalingGroup

**MyAutoScalingGroup Capacity overview**

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

Date created: Fri Feb 28 2025 00:27:42 GMT+0530 (India Standard Time)

[Edit](#)

**Details** [Integrations - new](#) [Automatic scaling](#) [Instance management](#) [Instance refresh](#) [Activity](#) [Monitoring](#)

**Launch template**

Launch template <a href="#">Edit</a> AutoScalingTemplate	AMI ID <a href="#">Edit</a> ami-02a53b0d62d37a757	Instance type t2.micro	Owner arn:aws:iam::762233734761:root
Version Default	Security groups -	Security group IDs <a href="#">Edit</a> sg-00242fc031917c3c3	Create time Fri Feb 28 2025 00:20:30 GMT+0530 (India Standard Time)
Description -	Storage (volumes) -	Key pair name new	Request Spot Instances No

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

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

## 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.