



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: Hema S

Department: ECE

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 costefficiency 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

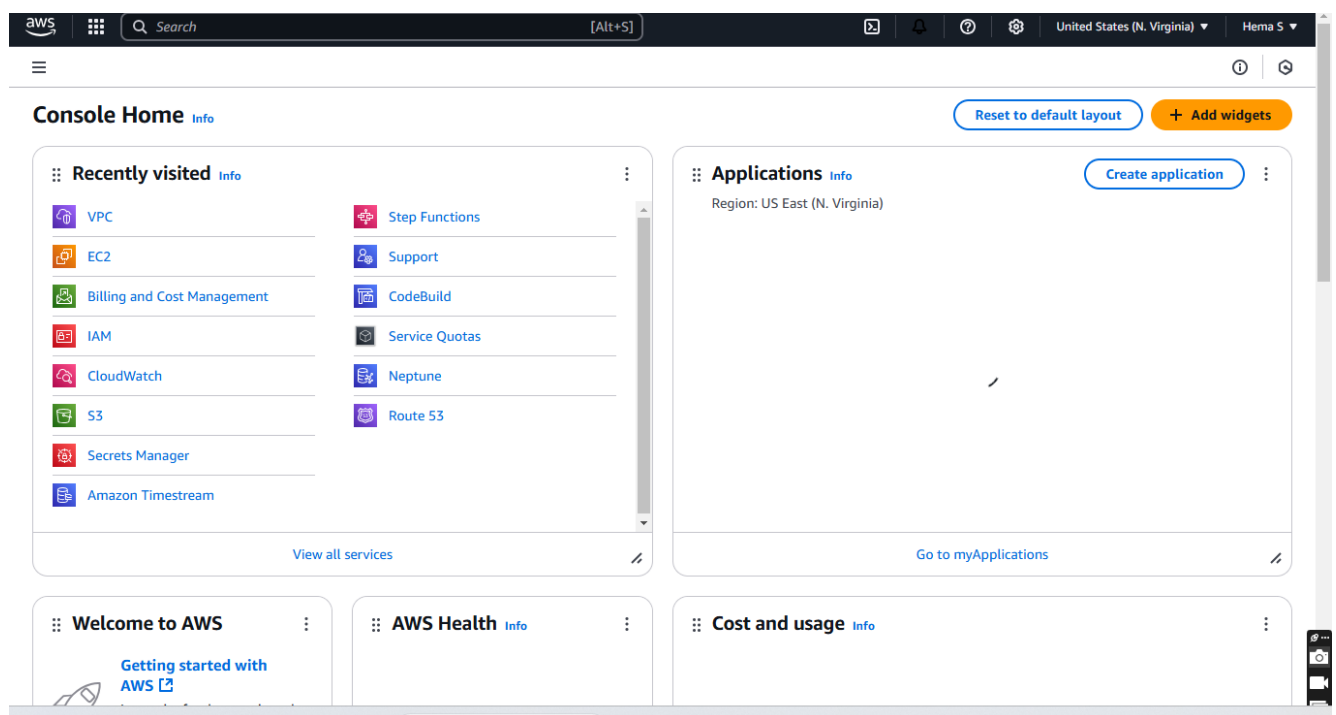
- 1. Improved Application Availability:** Auto Scaling ensures that applications remain available even during traffic spikes by automatically adding more VMs to meet demand.
- 2. Cost Optimization:** It dynamically reduces the number of VMs during low traffic periods, minimizing unnecessary costs.
-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

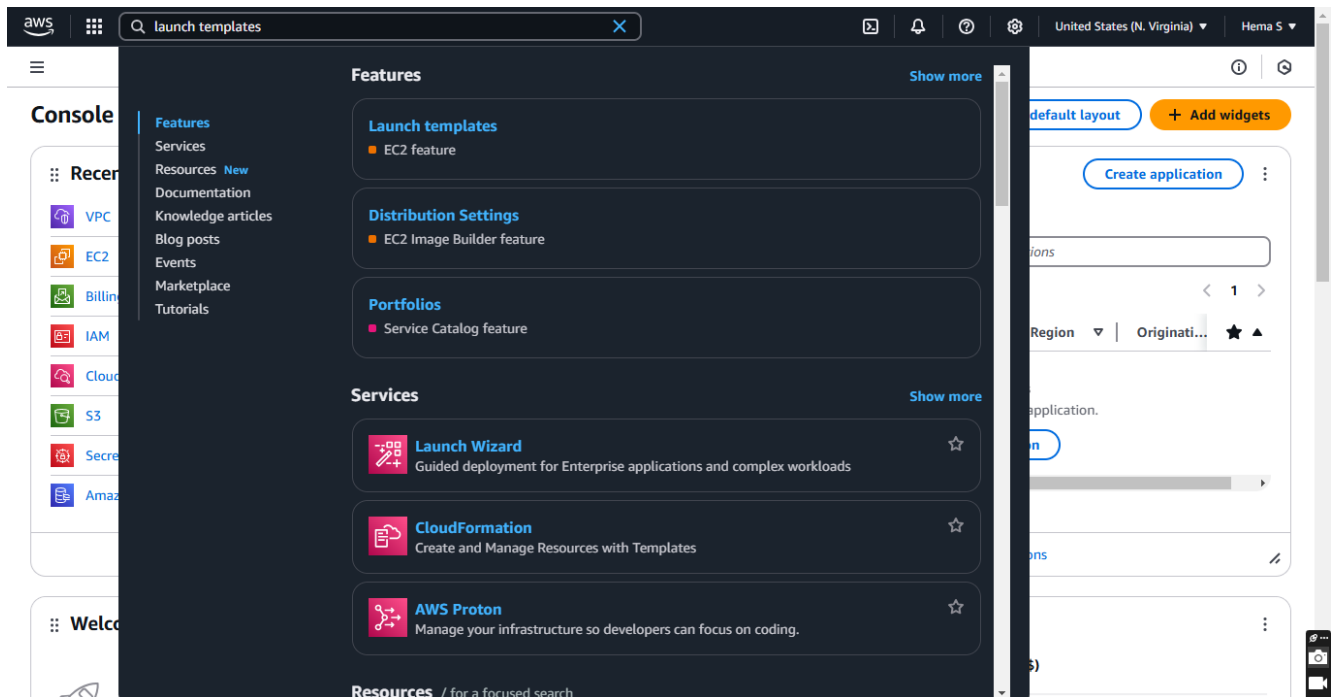
Step 1:

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



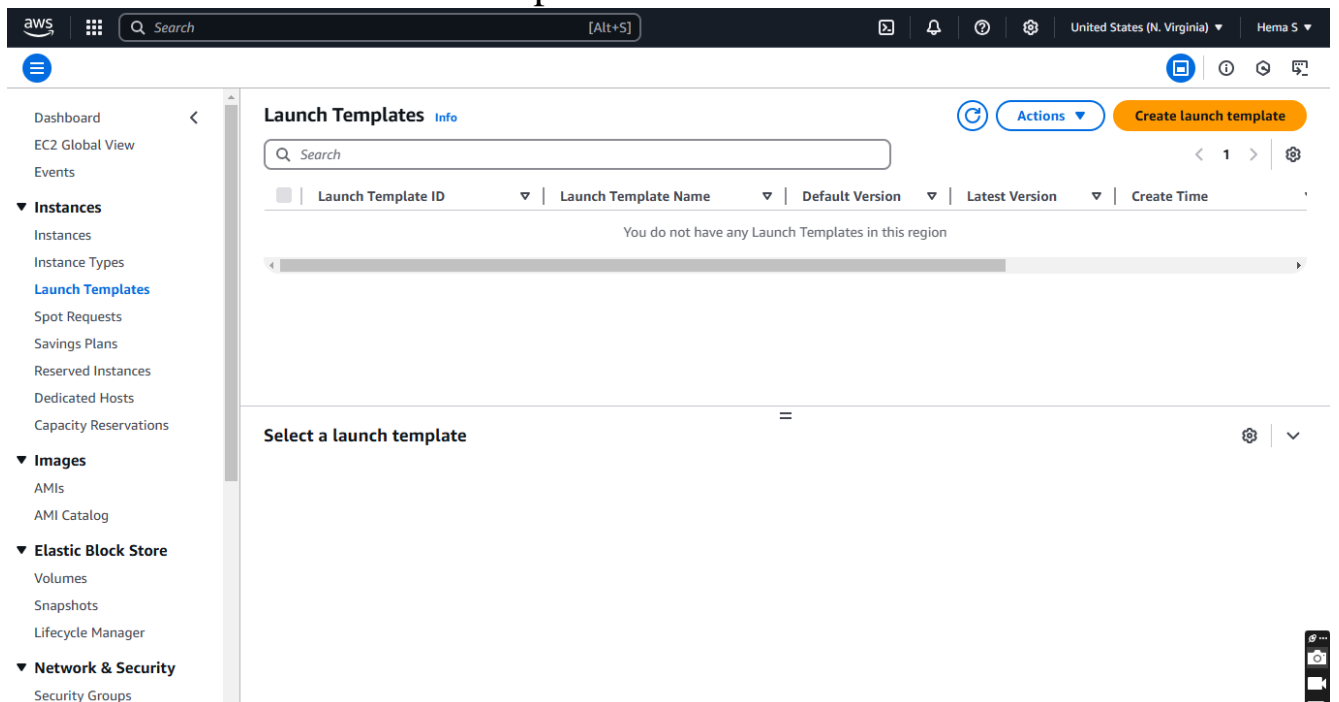
Step 2:

Search for Launch Templates.



Step 3:

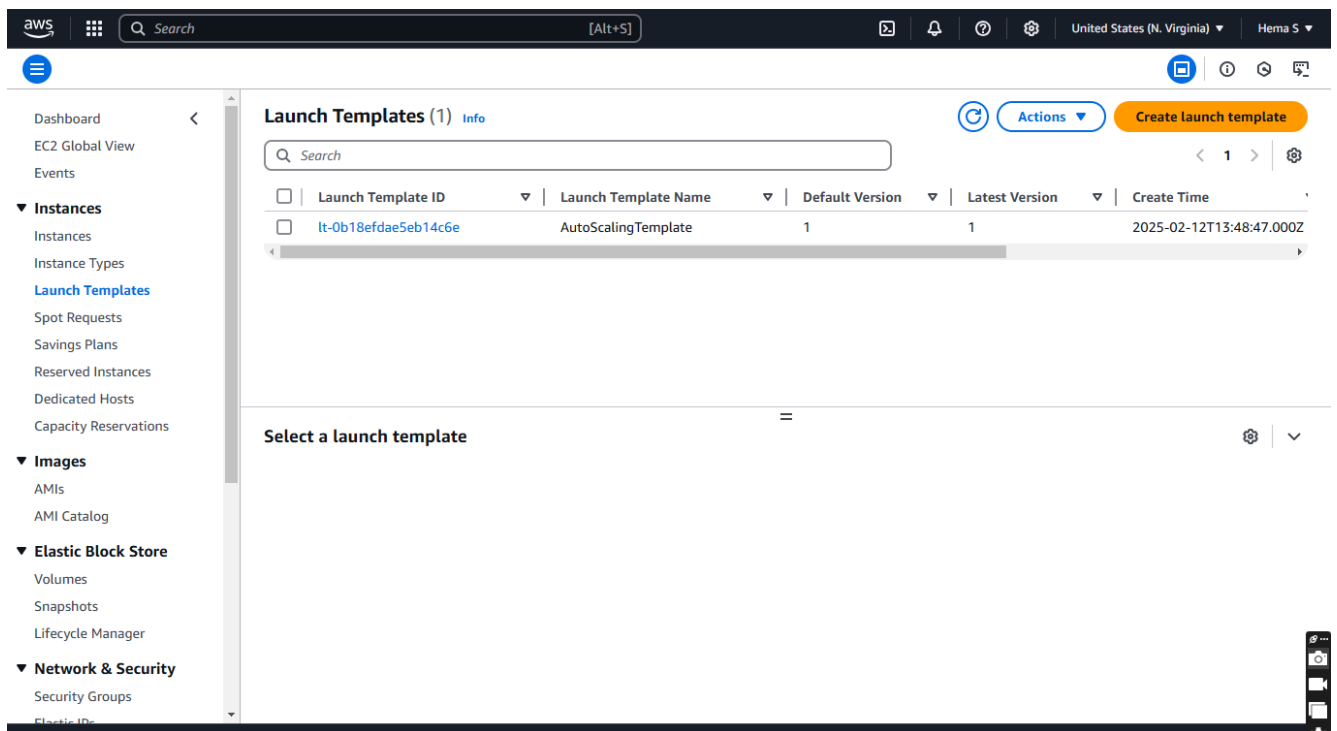
Click on the Create launch template.



Step 4:

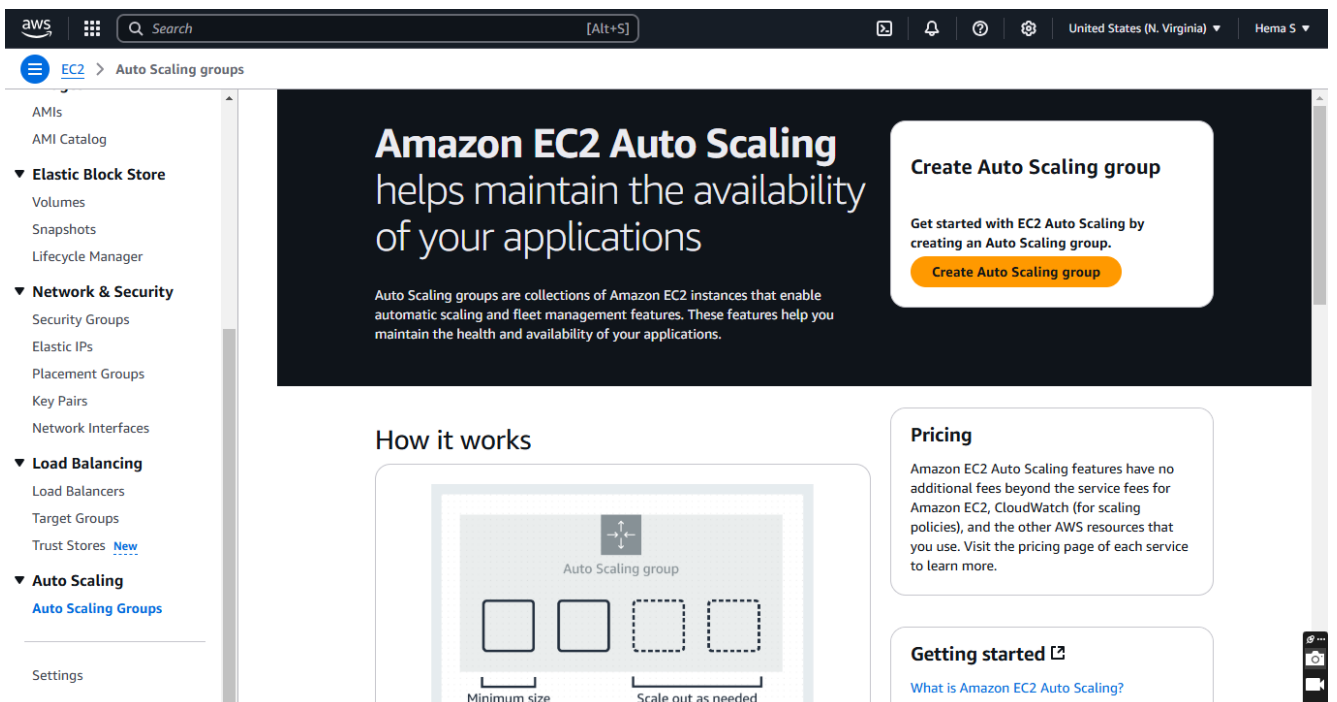
Create a **Launch Template** named **AutoScalingTemplate** using an **Amazon Machine Image (AMI)** like Amazon Linux 2 or any default image, and choose an **instance type** such as **t2.micro** for free-tier eligibility. Select an **existing key pair** (or create a new one) to enable

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).

The screenshot shows the 'Create Auto Scaling group' wizard in the AWS Management Console. The left sidebar lists seven steps: 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), and Step 7 (Review). Step 1 is currently selected.

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** | **Instance type**

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 'Create Auto Scaling group' wizard in the AWS Management Console, Step 2: Choose instance launch options. The left sidebar shows Step 2 is selected.

Choose instance launch options Info

Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

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	Version	Description
AutoScalingTemplate lt-0b18efdae5eb14c6e	Default	web

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

Step 8:

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

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

Integrate with other services - *optional* [Info](#)

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 [Info](#)

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 [Info](#)

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* [Info](#)

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

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 9:

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

aws

Search

[Alt+S]

United States (N. Virginia)

Hema S

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

Info

Step 1: Choose launch template

Edit

Group details

Auto Scaling group name

MyAutoScalingGroup

Launch template

AutoScalingTemplate

lt-0b18efdae5eb14c6e

Version

Default

Description

web

Step 2: Choose instance launch options

Edit

Network

VPC

vpc-0723525ba93a413d2

Availability Zones and subnets

Availability Zone	Subnet	Subnet CIDR range
us-east-1a	subnet-035f52afb9847abfa	172.31.0.0/20

aws

Search

[Alt+S]

United States (N. Virginia)

Hema S

EC2

>

Auto Scaling groups

>

MyAutoScalingGroup

AMIs

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

Auto Scaling

Auto Scaling Groups

Settings

MyAutoScalingGroup

MyAutoScalingGroup Capacity overview

Edit

arn:aws:autoscaling:us-east-1:711387100972:autoScalingGroup:8e4dabe5-50bf-4333-9135-f50458c8913b:autoScalingGroupName/MyAutoScalingGroup

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

Date created

Wed Feb 12 2025 19:42:36 GMT+0530 (India Standard Time)

<

Details

Integrations - new

Automatic scaling

Instance management

Instance refresh

Activity

>

Launch template

Edit

Launch template

lt-0b18efdae5eb14c6e

AutoScalingTemplate

Version

Default

Description

AMI ID

ami-085ad6ae776d8f09c

Security groups

-

Storage (volumes)

Instance type

t2.micro

Security group IDs

sg-0135c410e703566c4

Key pair name

Owner

arn:aws:iam::711387100972:root

Create time

Wed Feb 12 2025 19:18:47 GMT+0530 (India Standard Time)

Request Spot Instances

Auto Scaling groups (1)

Info

[Launch configurations](#)[Launch templates](#)[Actions](#)[Create Auto Scaling group](#)

Search your Auto Scaling groups

< 1 >

<input type="checkbox"/>	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	
<input type="checkbox"/>	MyAutoScalingGroup	AutoScalingTemplate Version Default	0	Updating capacity...	1	1	1	u...



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.