CLOUD COMPUTING LAB 3

1. Why is the Auto Scaling feature significant in cloud computing?

Auto scaling is a cloud computing feature that automatically adjusts the amount of computational resources allocated to an application based on its current demand. This means that the system can dynamically increase or decrease the number of active instances of services, virtual machines, or containers to maintain optimal performance and efficiency.

The Auto Scaling feature is significant in cloud computing for several key reasons:

1. Cost Efficiency:

- Optimized Resource Utilization: Auto scaling ensures that you are using only the necessary resources at any given time, scaling up during peak demands and scaling down during low demands. This prevents over-provisioning and reduces costs associated with idle resources.
- Pay-as-you-go Model: By dynamically adjusting the resources based on current needs, users can take full advantage of the pay-as-you-go pricing model offered by cloud providers, leading to significant cost savings.

2. Improved Performance and Availability:

- Handling Variable Loads: Auto scaling can respond to unexpected spikes in demand, ensuring that applications remain responsive and perform well under varying loads.
- High Availability: By automatically adding or removing instances based on demand, auto scaling helps maintain high availability and reliability of applications, minimizing the risk of downtime.

3. Enhanced Flexibility and Scalability:

- Elasticity: Auto scaling provides the elasticity needed to adapt to changing workloads without manual intervention. This is crucial for applications with unpredictable or highly variable usage patterns.
- Scalability: Businesses can seamlessly grow their infrastructure as needed, supporting business growth and new user requirements without significant upfront investments in hardware.

4. Operational Efficiency:

- Reduced Manual Intervention: Automating the scaling process reduces the need for manual monitoring and intervention, freeing up IT teams to focus on other critical tasks.

- Automated Management: Auto scaling can manage the entire lifecycle of instances, from provisioning to termination, ensuring that the infrastructure is always optimally configured.

5. Enhanced User Experience:

- Consistent Performance: By maintaining optimal resource levels, auto scaling helps ensure a consistent and positive user experience, even during peak usage times.
- Responsiveness: Quick response to changing demands means that users experience minimal latency and disruptions.

6. Disaster Recovery and Fault Tolerance:

- Resilience: Auto scaling can help create more resilient architectures by automatically replacing failed instances and distributing the load among healthy instances.
- Disaster Recovery: In the event of a failure, auto scaling can quickly provision new instances, helping to maintain service continuity.

2. Describe the following

1. Metric-based autoscaling
2. Schedule-based autoscaling

1. Metric Based Auto Scaling

Metric Based autoscaling, also known as dynamic or reactive autoscaling, automatically adjusts the number of active instances of a service or resource based on real time performance metrics. This ensures that the infrastructure adapts to varying workloads, providing sufficient resources during high demand and scaling down during low demand. Key metrics often used for this type of autoscaling include:

- CPU Utilization: Scaling up or down based on the percentage of CPU usage.
- Memory Usage: Adjusting instances according to memory consumption.
- Network Traffic: Scaling in response to the volume of inbound or outbound network traffic.
- Custom Metrics: Application Specific metrics such as request latency, error rates, or any other user defined parameters.

Metric Based auto scaling provides a responsive and efficient way to manage resources, ensuring optimal performance and cost efficiency by only using the necessary resources at any given time.

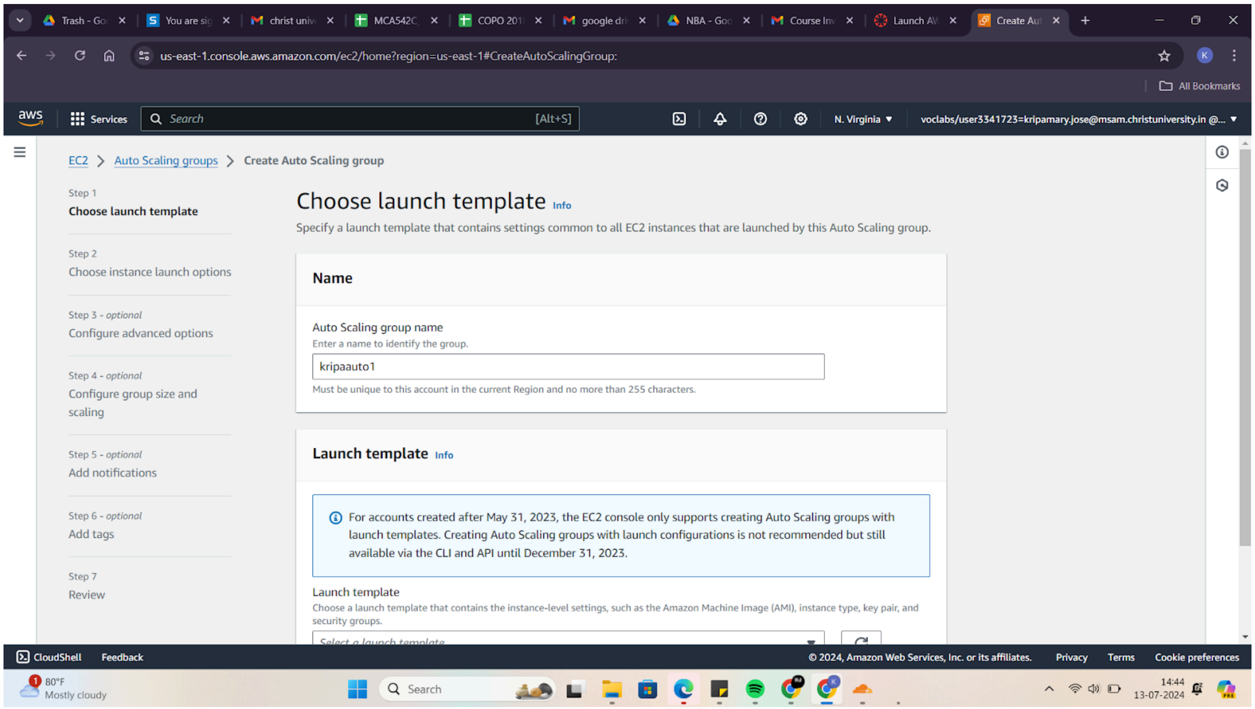
2. Schedule Based Auto Scaling

Schedule Based auto scaling, also known as time based or predictive autoscaling, adjusts the number of instances based on predefined schedules rather than real time metrics. This type of auto scaling is ideal for predictable workloads where resource needs are known to fluctuate at specific times. Examples of scenarios where schedules based auto scaling is useful include:

- Business Hours: Increasing resources during peak business hours and scaling down after hours or on weekends.
- Seasonal Trends: Adjusting resources for known seasonal spikes, such as holiday shopping periods.
- Regular Maintenance: Scaling down during scheduled maintenance windows or scaling up in anticipation of planned events.

Schedule Based auto scaling ensures that the necessary resources are available during known periods of high demand and helps to optimize costs by reducing resources during known periods of low activity.

3.Demonstrate Metric based auto scaling or Schedule based auto scaling to cater your organizations business requirements.



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateAutoScalingGroup:

voclabs/user3341723=kripamary.jose@msam.christuniversity.in @...

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Choose launch template

Step 2
Choose instance launch options

Step 3 - *optional*
Configure advanced options

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.

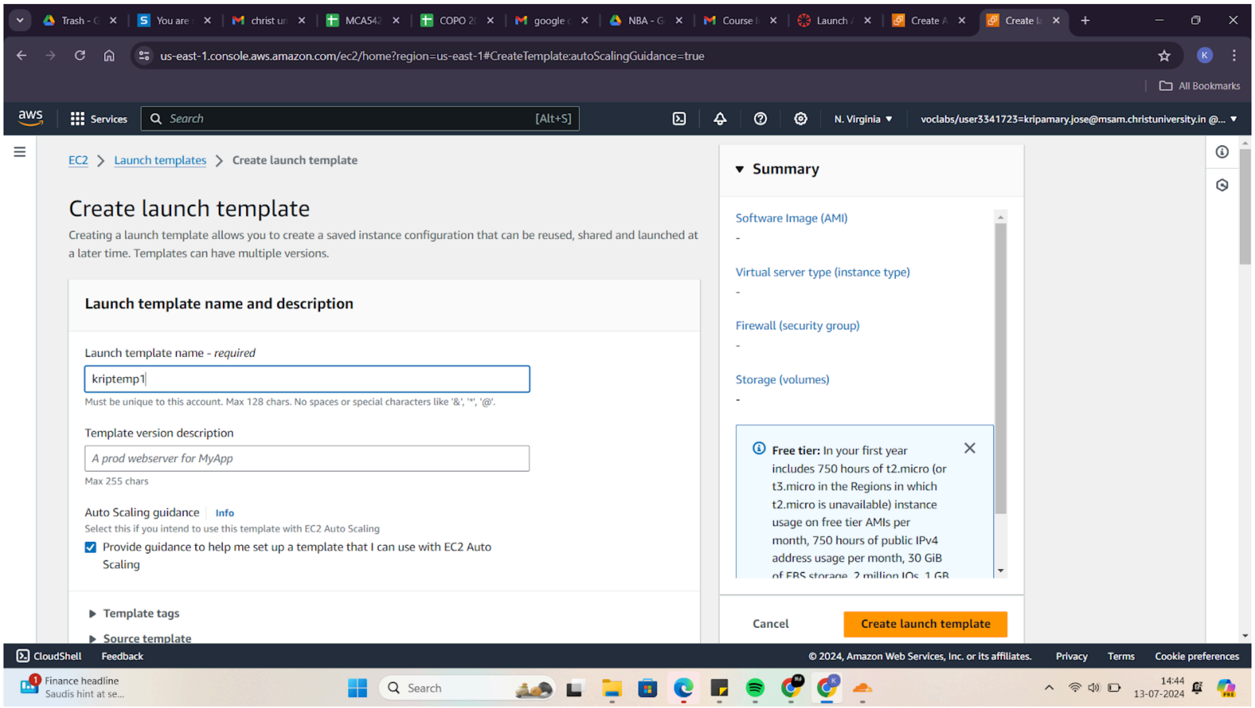
Name

Auto Scaling group name
Enter a name to identify the group.
kripaauto1
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.
Select a launch template


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateTemplate:autoScalingGuidance=true
Services
Search [Alt+S]
N. Virginia
voclabs/user3341723=kripamary.jose@msam.christuniversity.in @...
EC2 > Launch templates > Create launch template

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*

kriptemp1

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

Template version description

A prod webserver for MyApp

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

▸ Template tags

▸ Source template

Summary

Software Image (AMI)
-

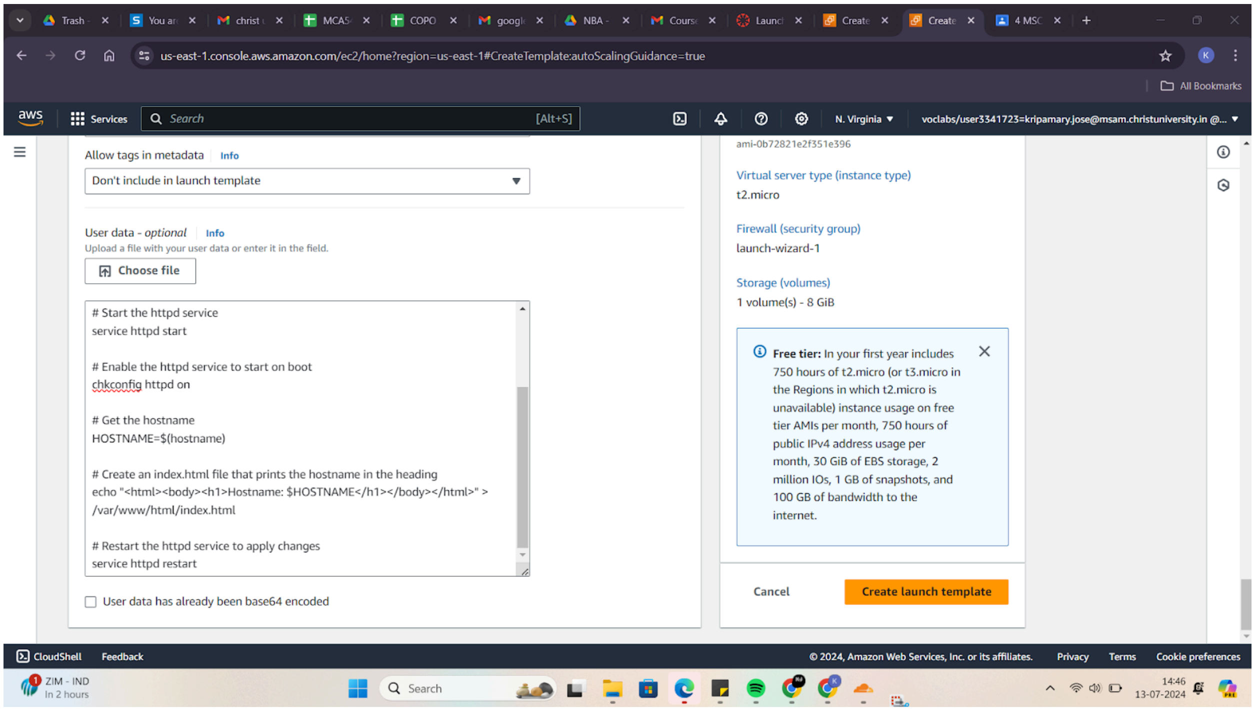
Virtual server type (instance type)
-

Firewall (security group)
-

Storage (volumes)
-

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB

Cancel
Create launch template
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
14:44
13-07-2024


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateTemplate:autoScalingGuidance=true
Services
Search [Alt+S]
N. Virginia
voclabs/user3341723=kripamary.jose@msam.christuniversity.in @...

Allow tags in metadata Info

Don't include in launch template

User data - *optional* Info

Upload a file with your user data or enter it in the field.

Choose file

```
# Start the httpd service
service httpd start

# Enable the httpd service to start on boot
chkconfig httpd on

# Get the hostname
HOSTNAME=$(hostname)

# Create an index.html file that prints the hostname in the heading
echo "<html><body><h1>Hostname: $HOSTNAME</h1></body></html>" >
/var/www/html/index.html

# Restart the httpd service to apply changes
service httpd restart
```

☐ User data has already been base64 encoded

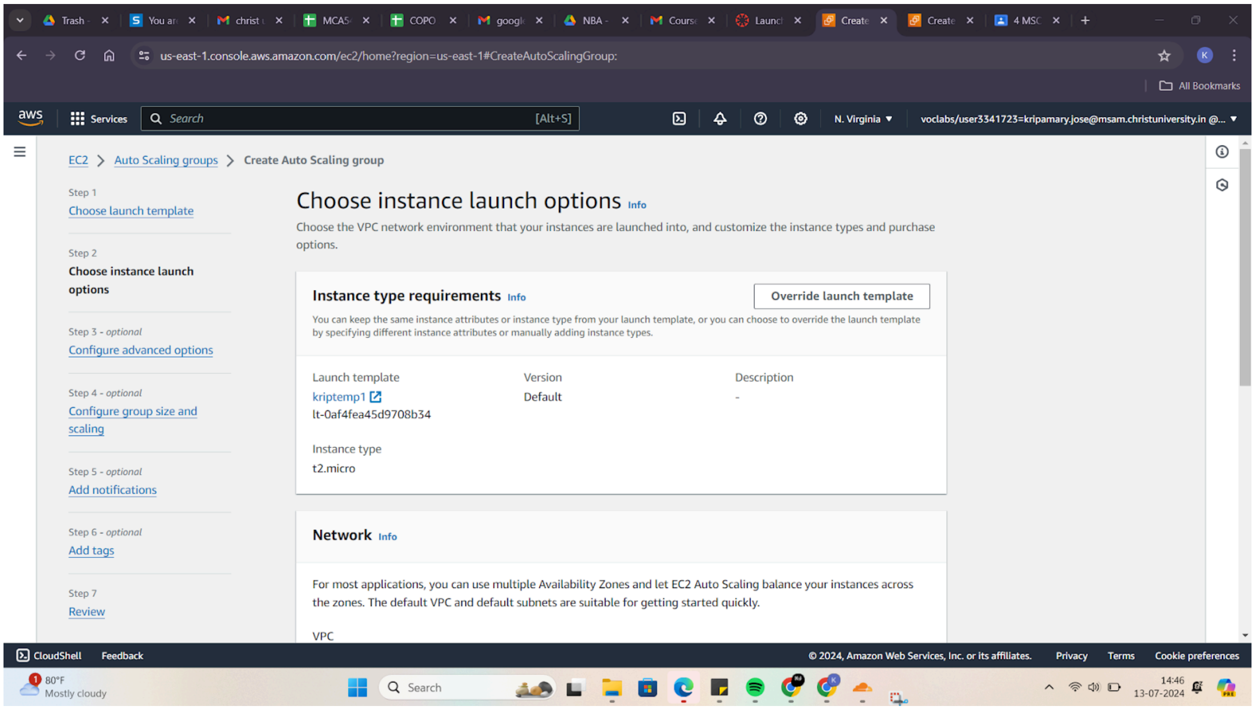
ami-0b72821e2f351e396

Virtual server type (instance type)
t2.micro

Firewall (security group)
launch-wizard-1

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateAutoScalingGroup:
Services
Search [Alt+S]
N. Virginia
voclabs/user3341723=kripamary.jose@msam.christuniversity.in @...
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
80°F Mostly cloudy
14:46 13-07-2024

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1 Choose launch template

Step 2 Choose instance launch options

Step 3 - *optional* Configure advanced options

Step 4 - *optional* Configure group size and scaling

Step 5 - *optional* Add notifications

Step 6 - *optional* Add tags

Step 7 Review

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.

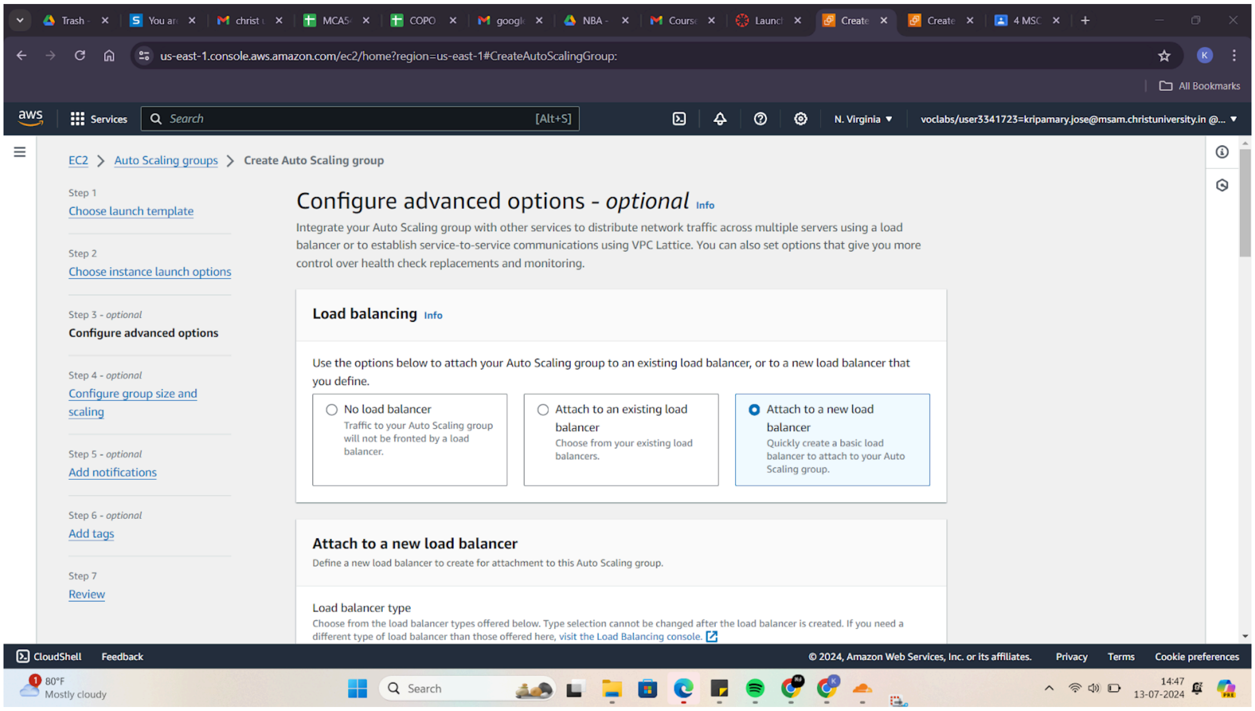
Launch template	Version	Description
kriptemp1 lt-0af4fea45d9708b34	Default	-

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


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#CreateAutoScalingGroup:
Services
Search [Alt+S]
N. Virginia
voclabs/user3341723=kripamary.jose@msam.christuniversity.in @...
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
80°F Mostly cloudy
14:47 13-07-2024

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1 Choose launch template

Step 2 Choose instance launch options

Step 3 - *optional* Configure advanced options

Step 4 - *optional* Configure group size and scaling

Step 5 - *optional* Add notifications

Step 6 - *optional* Add tags

Step 7 Review

Configure advanced options - *optional* Info

Integrate your Auto Scaling group with other services to distribute network traffic across multiple servers using a load balancer or to establish service-to-service communications using VPC Lattice. You can also set options that give you more control over 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.

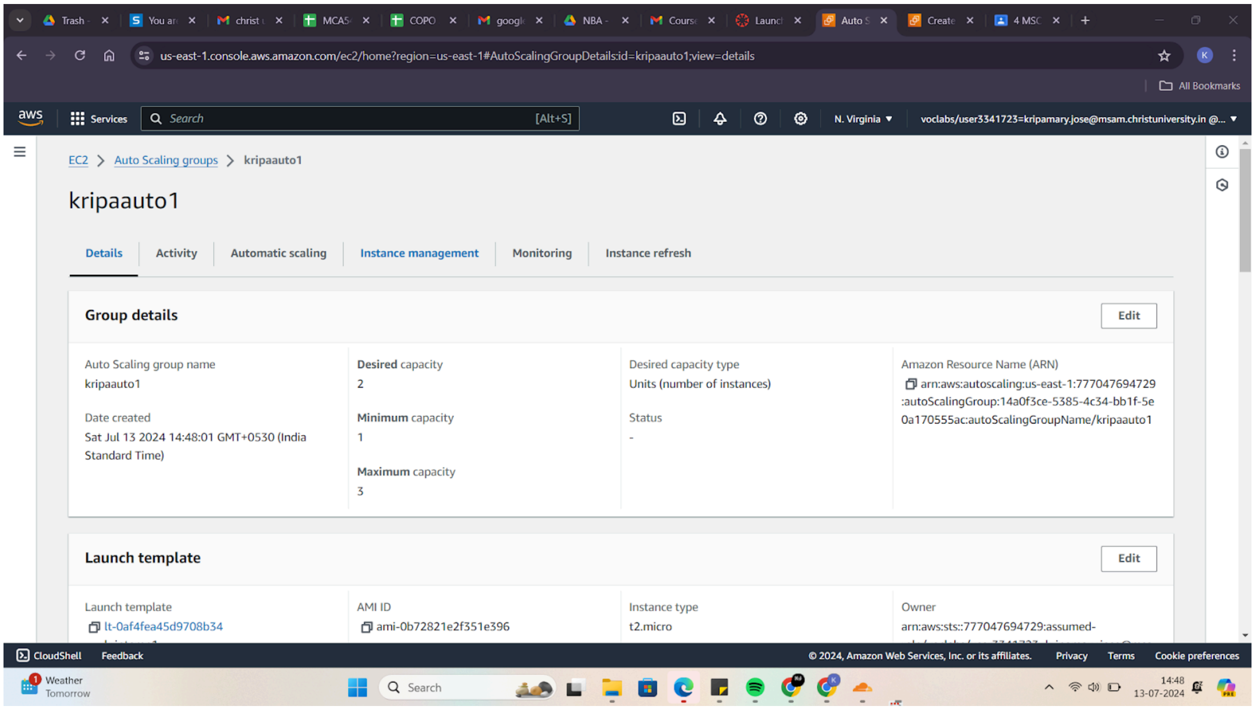
○ Attach to an existing load balancer
Choose from your existing load balancers.

● Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

Attach to a new load balancer

Define a new load balancer to create for attachment to this Auto Scaling group.

Load balancer type
Choose from the load balancer types offered below. Type selection cannot be changed after the load balancer is created. If you need a different type of load balancer than those offered here, visit the Load Balancing console.



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#AutoScalingGroupDetails:id=kripaauto1;view=details

EC2 > Auto Scaling groups > kripaauto1

kripaauto1

Group details

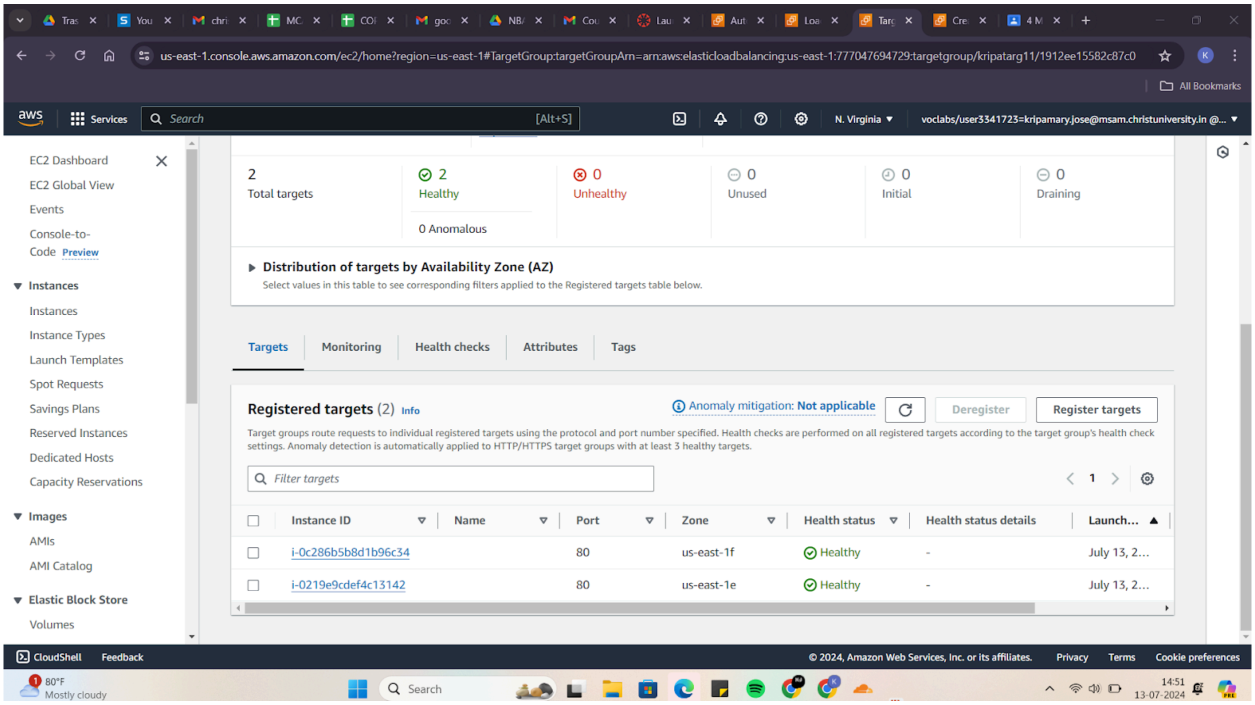
Auto Scaling group name kripaauto1	Desired capacity 2	Desired capacity type Units (number of instances)	Amazon Resource Name (ARN) arn:aws:autoscaling:us-east-1:777047694729:autoScalingGroup:14a0f3ce-5385-4c34-bb1f-5e0a170555ac:autoScalingGroupName/kripaauto1
Date created Sat Jul 13 2024 14:48:01 GMT+0530 (India Standard Time)	Minimum capacity 1	Status -	
	Maximum capacity 3		

Launch template

Launch template lt-0af4fea45d9708b34	AMI ID ami-0b72821e2f351e396	Instance type t2.micro	Owner arn:aws:sts::777047694729:assumed-

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

14:48 13-07-2024



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#TargetGroup:targetGroupArn=arn:aws:elasticloadbalancing:us-east-1:777047694729:targetgroup/kripatarg11/1912ee15582c87c0

2 Total targets	2 Healthy 0 Anomalous	0 Unhealthy	0 Unused	0 Initial	0 Draining

▶ Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

Registered targets (2) Info

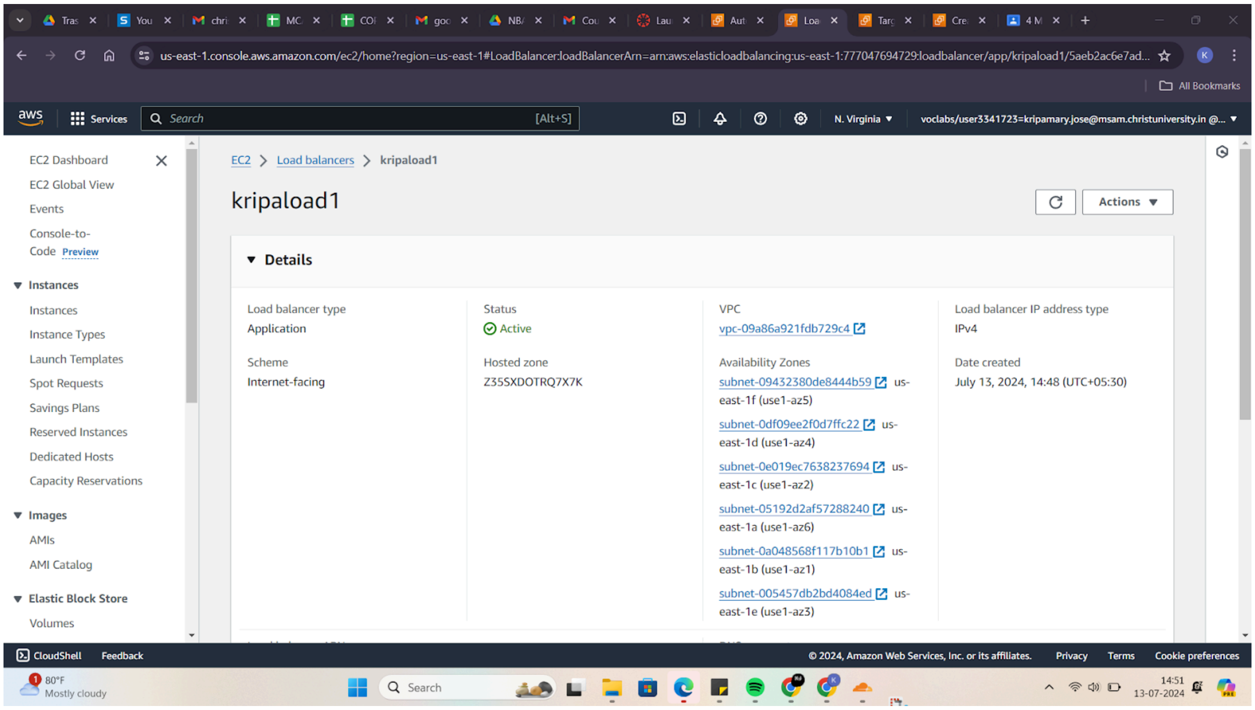
Anomaly mitigation: Not applicable

Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

□	Instance ID	Name	Port	Zone	Health status	Health status details	Launch...
□	i-0c286b5b8d1b96c34		80	us-east-1f	✓ Healthy	-	July 13, 2...
□	i-0219e9cdef4c13142		80	us-east-1e	✓ Healthy	-	July 13, 2...

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

14:51 13-07-2024



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:us-east-1:777047694729:loadbalancer/app/kripaload1/5aeb2ac6e7ad...

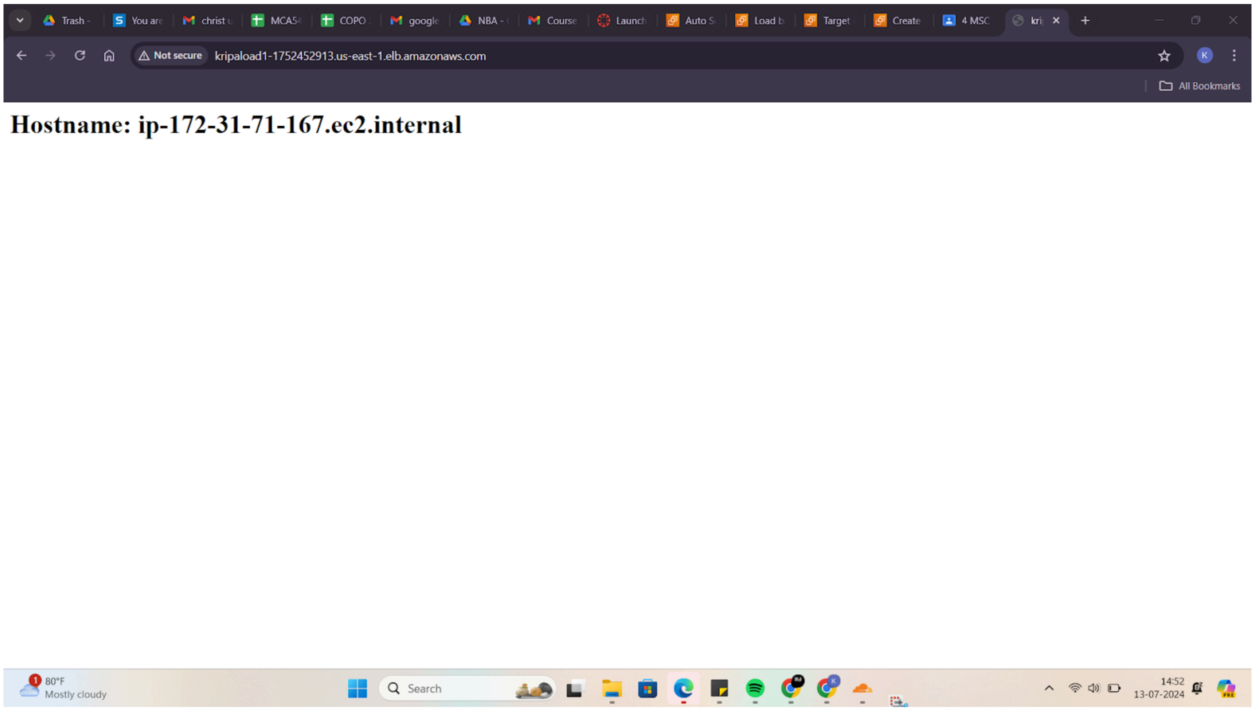
EC2 > Load balancers > kripaload1

kripaload1

Details

Load balancer type	Status	VPC	Load balancer IP address type
Application	✓ Active	vpc-09a86a921fdb729c4	IPv4

Scheme	Hosted zone	Availability Zones	Date created
Internet-facing	Z35SXDOTRQ7X7K	subnet-09432380de8444b59 us-east-1f (use1-az5) subnet-0df09ee2f0d7ffc22 us-east-1d (use1-az4) subnet-0e019ec7638237694 us-east-1c (use1-az2) subnet-05192d2af57288240 us-east-1a (use1-az6) subnet-0a048568f117b10b1 us-east-1b (use1-az1) subnet-005457db2bd4084ed us-east-1e (use1-az3)	July 13, 2024, 14:48 (UTC+05:30)



Not secure kripaload1-1752452913.us-east-1.elb.amazonaws.com

Hostname: ip-172-31-71-167.ec2.internal