

# AWS LAB - Auto Scaling

Bharathi  
R22BPS10  
10

## 1. Create a VPC with details and ip as in the image

The screenshot shows the 'Create VPC' wizard in the AWS VPC console. The 'VPC settings' step is selected. Configuration includes:

- Resources to create:** VPC only (selected)
- Name tag - optional:** uv-test-vpc
- IPv4 CIDR block:** 12.0.0.0/16
- IPv6 CIDR block:** No IPv6 CIDR block selected
- Tenancy:** Default

**VPC created successfully**

The screenshot shows the 'VPC dashboard' for the newly created VPC 'vpc-0a12383cb7208ce9f / uv-test-vpc'. Key details include:

VPC ID	State	DNS hostnames	DNS resolution
vpc-0a12383cb7208ce9f	Available	Disabled	Enabled
Tenancy	DHCP option set	Main route table	Main network ACL
Default	dopt-3a4ef640	rtb-0828cf7b960c1e0cc	acl-09cfcb7db0d236596
Default VPC	IPv4 CIDR	IPv6 pool	IPv6 CIDR (Network border group)
No	12.0.0.0/16	-	-
Network Address Usage metrics	Route 53 Resolver DNS Firewall rule groups	Owner ID	
Disabled	-	904951670971	

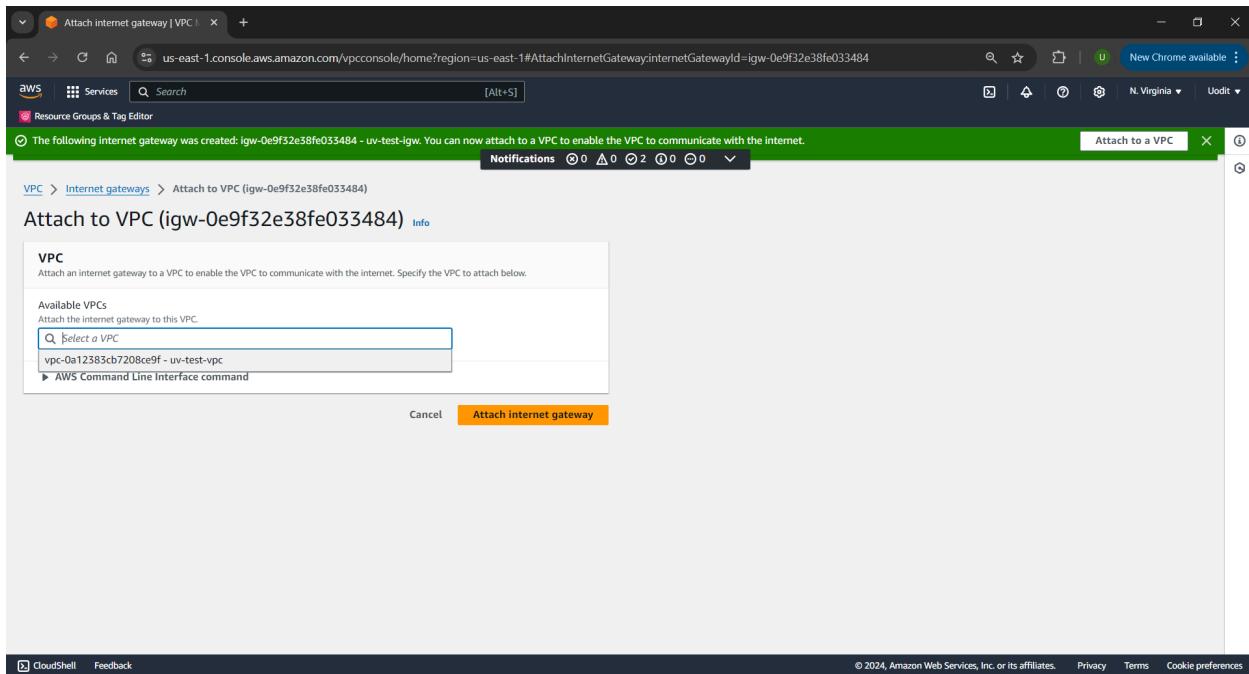
## 2. Create a new Internet Gateway

The screenshot shows the 'Create internet gateway' wizard in the AWS VPC console. The top navigation bar includes 'Services', 'Search', and 'Region: N. Virginia'. A success message at the top states 'You successfully created vpc-0a12383cb7208ce9f / uv-test-vpc'. The main section is titled 'Create internet gateway' with a 'Info' link. It explains that an internet gateway is a virtual router that connects a VPC to the internet. Below this, the 'Internet gateway settings' section contains a 'Name tag' field with the value 'uv-test-igw'. The 'Tags - optional' section shows one tag named 'Name' with value 'uv-test-igw'. At the bottom right are 'Cancel' and 'Create internet gateway' buttons.

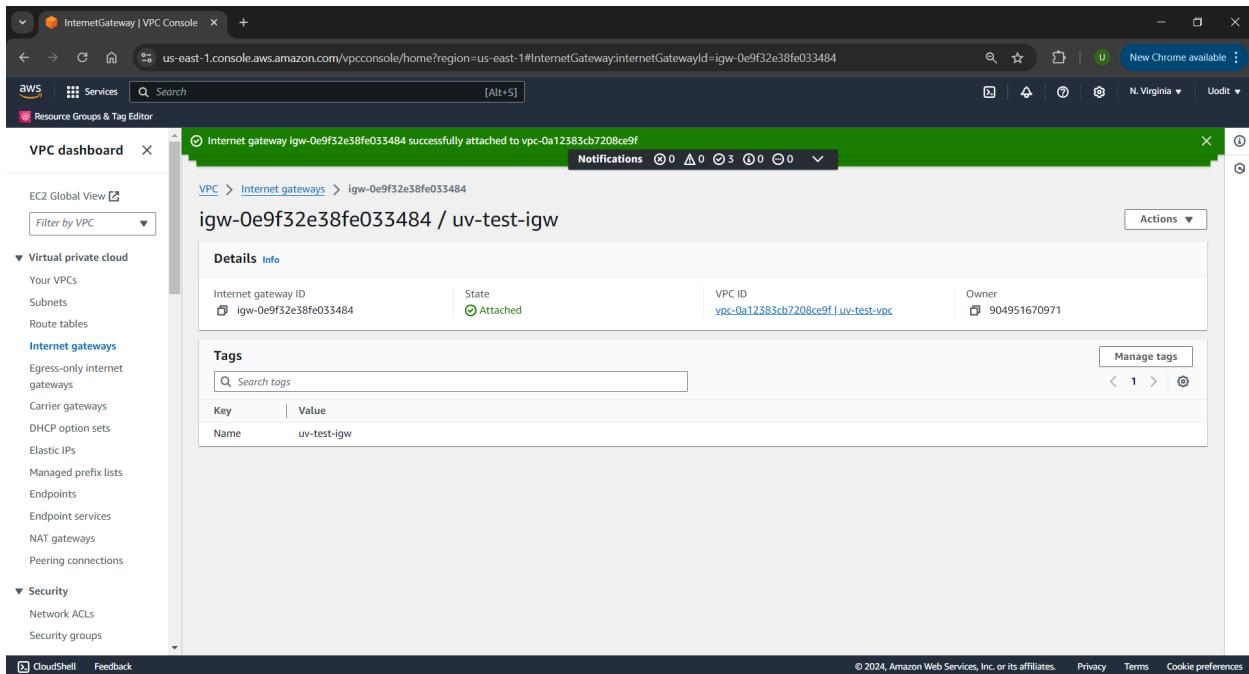
Internet gateway created successfully

The screenshot shows the 'InternetGateway | VPC Console' page. The top navigation bar includes 'Services', 'Search', and 'Region: N. Virginia'. A success message at the top states 'The following Internet gateway was created: igw-0e9f32e38fe033484 - uv-test-igw. You can now attach to a VPC to enable the VPC to communicate with the internet.' Below this, the 'VPC dashboard' sidebar lists 'EC2 Global View', 'Virtual private cloud', 'Internet gateways', 'Security', and 'CloudShell'. The main content area shows the 'igw-0e9f32e38fe033484 / uv-test-igw' internet gateway. The 'Details' tab is selected, showing the 'Internet gateway ID' as 'igw-0e9f32e38fe033484', 'State' as 'Detached', 'VPC ID' as '–', and 'Owner' as '904951670971'. The 'Tags' section shows one tag named 'Name' with value 'uv-test-igw'. At the bottom right are 'Actions' and 'Manage tags' buttons.

### 3. Attach the internet gateway to the VPC



**Internet gateway attached successfully**



## 4. Create a subnets

The screenshot shows the 'Create subnet' wizard in the AWS VPC console. In the 'VPC' section, a VPC ID is selected: 'vpc-0a12383cb7208ce9f (uv-test-vpc)'. Under 'Associated VPC CIDRs', the IPv4 CIDR is set to '12.0.0.0/16'. In the 'Subnet settings' section, 'Subnet 1 of 1' is being created. The 'Subnet name' is 'uv-test-sbnet-1'. The 'Availability Zone' is 'US East (N. Virginia) / us-east-1a'. The 'IPv4 VPC CIDR block' is '12.0.0.0/16'. The 'IPv4 subnet CIDR block' is '12.0.1.0/24'. A note indicates '256 IPs'. At the bottom, there are links for 'CloudShell' and 'Feedback'.

**Subnet-1: uv-test-sbnet-1 created successfully with ip-12.0.1.0/24**

The screenshot shows the AWS VPC dashboard. A green notification bar at the top states: 'You have successfully created 1 subnet: subnet-08748eee1f57a5805'. Below this, the 'Subnets (1)' section is displayed. A table lists the subnet details:

Name	Subnet ID	State	VPC	IPv4 CIDR
uv-test-sbnet-1	subnet-08748eee1f57a5805	Available	vpc-0a12383cb7208ce9f (uv-te...)	12.0.1.0/24

The left sidebar includes sections for EC2 Global View, Virtual private cloud (Your VPCs, Subnets, Route tables, Internet gateways, Egress-only internet gateways, Carrier gateways, DHCP option sets, Elastic IPs, Managed prefix lists, Endpoints, Endpoint services, NAT gateways, Peering connections), and Security. At the bottom, there are links for 'CloudShell' and 'Feedback'.

## 5. Create second subnet

The screenshot shows the 'Create subnet' wizard in the AWS VPC console. In the 'VPC' section, a VPC ID is selected: 'vpc-0a12383cb7208ce9f (uv-test-vpc)'. Under 'Associated VPC CIDRs', the IPv4 CIDR is set to '12.0.0.0/16'. In the 'Subnet settings' section, 'Subnet 1 of 1' is being created. The 'Subnet name' is 'uv-test-subnet-2'. The 'Availability Zone' is 'US East (N. Virginia) / us-east-1a'. The 'IPv4 VPC CIDR block' is '12.0.0.0/16' and the 'IPv4 subnet CIDR block' is '12.0.3.0/24'. The page includes standard AWS navigation and footer links.

**Subnet-1: uv-test-subnet-2 created successfully with ip-12.0.3.0/24**

The screenshot shows the AWS VPC dashboard. On the left, the 'Virtual private cloud' menu is expanded, with 'Subnets' selected. The main area displays a table titled 'Subnets (1) info' showing one subnet: 'uv-test-subnet-2' (Subnet ID: subnet-03e5f715a3c297671, State: Available, VPC: vpc-0a12383cb7208ce9f, IPv4 CIDR: 12.0.3.0/24). A success message at the top states: 'You have successfully created 1 subnet: subnet-03e5f715a3c297671'. The page includes standard AWS navigation and footer links.

Name	Subnet ID	State	VPC	IPv4 CIDR
uv-test-subnet-2	subnet-03e5f715a3c297671	Available	vpc-0a12383cb7208ce9f	12.0.3.0/24

## 6. Create a route table and select the above created VPC

The screenshot shows the 'Create route table' wizard in the AWS VPC console. In the 'Route table settings' section, a tag 'uv-test-rt' is added under 'Name - optional'. The 'VPC' dropdown is set to 'vpc-0a12383cb7208ce9f (uv-test-vpc)'. In the 'Tags' section, a tag 'uv-test-rt' is added under 'Value - optional'. The 'Create route table' button is highlighted in orange at the bottom.

Route table created successfully

The screenshot shows the 'RouteTableDetails' page for the route table 'rtb-0671b78d2e2058981'. A green banner at the top states 'Route table rtb-0671b78d2e2058981 | uv-test-rt was created successfully.' The main details table shows the route table ID, VPC, and other configurations. The 'Routes' tab is selected, displaying one route entry: 'Destination: 12.0.0.0/16, Target: local, Status: Active, Propagated: No'.

## 7. Go to subnet association section of route table and click edit subnet associations

The screenshot shows the AWS VPC Route Table Details page. The route table ID is rtb-0671b78d2e2058981 and it is associated with the VPC vpc-0a12383db7208ce9f. The Subnet associations tab is selected, showing a table with two rows of subnets: uv-test-sbnet-1 and uv-test-subnet-2, both associated with the main route table.

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
uv-test-sbnet-1	subnet-08748eee1f57a5805	12.0.1.0/24	-	Main (rtb-0828cf7b960c1e0cc)
uv-test-subnet-2	subnet-03e5f715a3c297671	12.0.3.0/24	-	Main (rtb-0828cf7b960c1e0cc)

## Select the subnets created and save association

The screenshot shows the EditRouteTableSubnetAssociations page. The subnets uv-test-sbnet-1 and uv-test-subnet-2 are selected in the Available subnets section. They are also listed in the Selected subnets section, which contains the same two entries. A Save associations button is visible at the bottom right.

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
uv-test-sbnet-1	subnet-08748eee1f57a5805	12.0.1.0/24	-	Main (rtb-0828cf7b960c1e0cc)
uv-test-subnet-2	subnet-03e5f715a3c297671	12.0.3.0/24	-	Main (rtb-0828cf7b960c1e0cc)

## 8. Go to routes section of route table and click edit routes

You have successfully updated subnet associations for rtb-0671b78d2e2058981 / uv-test-rt.

rtb-0671b78d2e2058981 / uv-test-rt

Details

Route table ID rtb-0671b78d2e2058981	Main No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-0a12385cb7208ce9f   uv-test-vpc	Owner ID 904951670971		

Routes (1)

Destination	Target	Status	Propagated
12.0.0.0/16	local	Active	No

Click add route and select internet gateway created and save changes

Edit routes

Destination 12.0.0.0/16	Target local	Status Active	Propagated No
0.0.0.0/0	Internet Gateway igw-0e9f32e38fe033484	-	No

Add route

Cancel Preview Save changes

## 9. Go to EC2 and create a Target group by selecting type as instance

The screenshot shows the 'Specify group details' step of the 'Create target group' wizard. Under 'Basic configuration', the 'Instances' option is selected. It lists two bullet points: 'Supports load balancing to instances within a specific VPC.' and 'Facilitates the use of [Amazon EC2 Auto Scaling](#) to manage and scale your EC2 capacity.' Below this, there are three other options: 'IP addresses', 'Lambda function', and 'Application Load Balancer'. The 'Target group name' field contains 'uv-tg-autoscaling-exp'.

Target group created successfully

The screenshot shows the 'uv-tg-autoscaling-exp' target group details page. A green banner at the top states 'Successfully created the target group: uv-tg-autoscaling-exp. Anomaly detection is automatically applied to all registered targets. Results can be viewed in the Targets tab.' The 'Details' section shows the ARN: 'arn:aws:elasticloadbalancing:us-east-1:904951670971:targetgroup/uv-tg-autoscaling-exp/c579a580fe781f6a'. The 'Targets' section shows 0 total targets, 0 healthy targets, 0 unhealthy targets, 0 unused targets, 0 initial targets, and 0 draining targets. The 'Registered targets (0)' section shows a table with columns: Instance ID, Name, Port, Zone, Health status, Health status details, and Launch time. A search bar 'Filter targets' is present. At the bottom right, there are buttons for 'Anomaly mitigation: Not applicable' (with a dropdown arrow), 'Deregister', and 'Register targets'.

## 10. Create a security group for HTTP requests

The screenshot shows the 'Create security group' wizard in the AWS Management Console. In the 'Basic details' section, the security group name is 'uv-sg-autoscaling-exp' and the description is 'allow http req.'. The VPC is set to 'vpc-0a12383cb7208ce9f (uv-test-vpc)'. In the 'Inbound rules' section, there is one rule: Type: HTTP, Protocol: TCP, Port range: 80, Source: Anywhere (0.0.0.0/0). An 'Add rule' button is visible.

**Security group created successfully**

The screenshot shows the 'sg-04b0dae5b7973583d - uv-sg-autoscaling-exp' security group details. It has a success message: 'Security group (sg-04b0dae5b7973583d | uv-sg-autoscaling-exp) was created successfully'. The 'Details' tab shows the security group name, ID, owner, and VPC ID. The 'Inbound rules' tab lists one rule: Name: sgr-0a005e219f9cc0606, IP version: IPv4, Type: HTTP, Protocol: TCP, Port range: 80, Source: 0.0.0.0/0. The 'Actions' dropdown menu is open.

## 11. Create another security group for HTTP and SSH request

The screenshot shows the 'Create security group' page in the AWS EC2 console. The 'Basic details' section includes:

- Security group name: `uv-sg-autoscaling-apache-exp`
- Description: `allow ssh and http req.`
- VPC Info: `vpc-0a12383cb7208ce9f (uv-test-vpc)`

The 'Inbound rules' section contains two rules:

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	Anywhere (0.0.0.0/0)	
SSH	TCP	22	Anywhere (0.0.0.0/0)	

At the bottom, there is a 'Add rule' button.

**Security group created successfully**

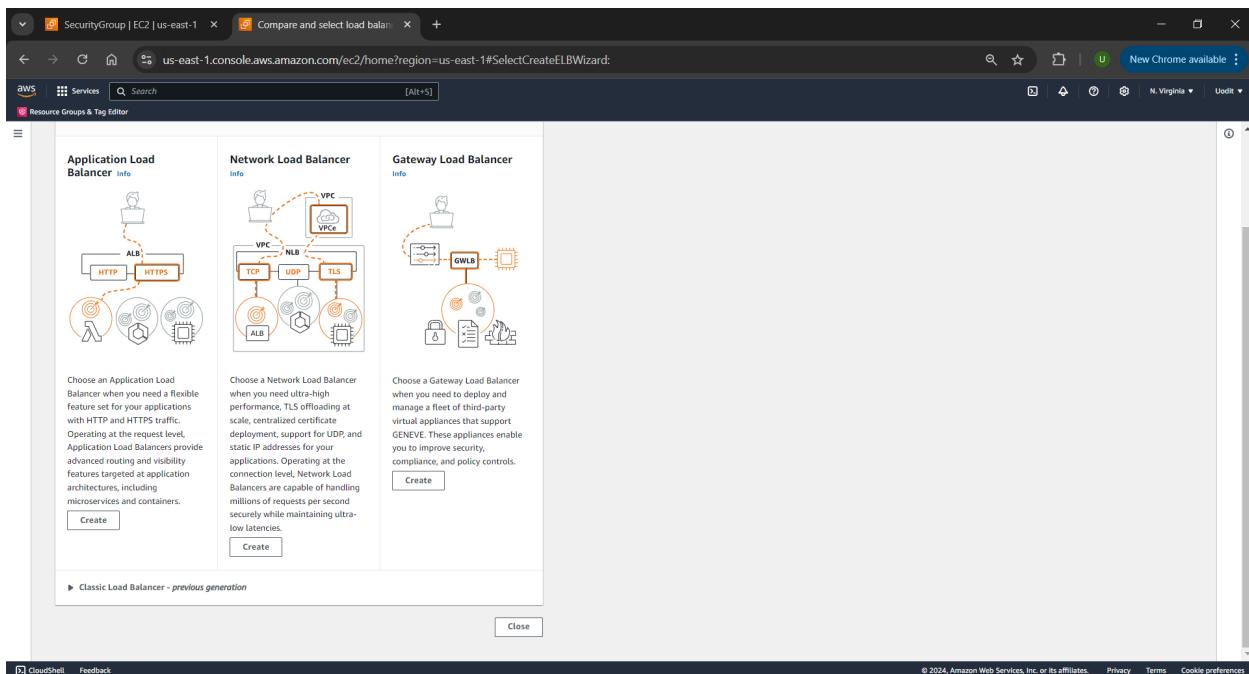
The screenshot shows the 'sg-0f12c4b76ed215a49 - uv-sg-autoscaling-apache-exp' security group details page. The 'Details' section includes:

Security group name: <code>uv-sg-autoscaling-apache-exp</code>	Security group ID: <code>sg-0f12c4b76ed215a49</code>	Description: <code>allow ssh and http req.</code>	VPC ID: <code>vpc-0a12383cb7208ce9f</code>
Owner: <code>904951670971</code>	Inbound rules count: <code>2 Permission entries</code>	Outbound rules count: <code>1 Permission entry</code>	

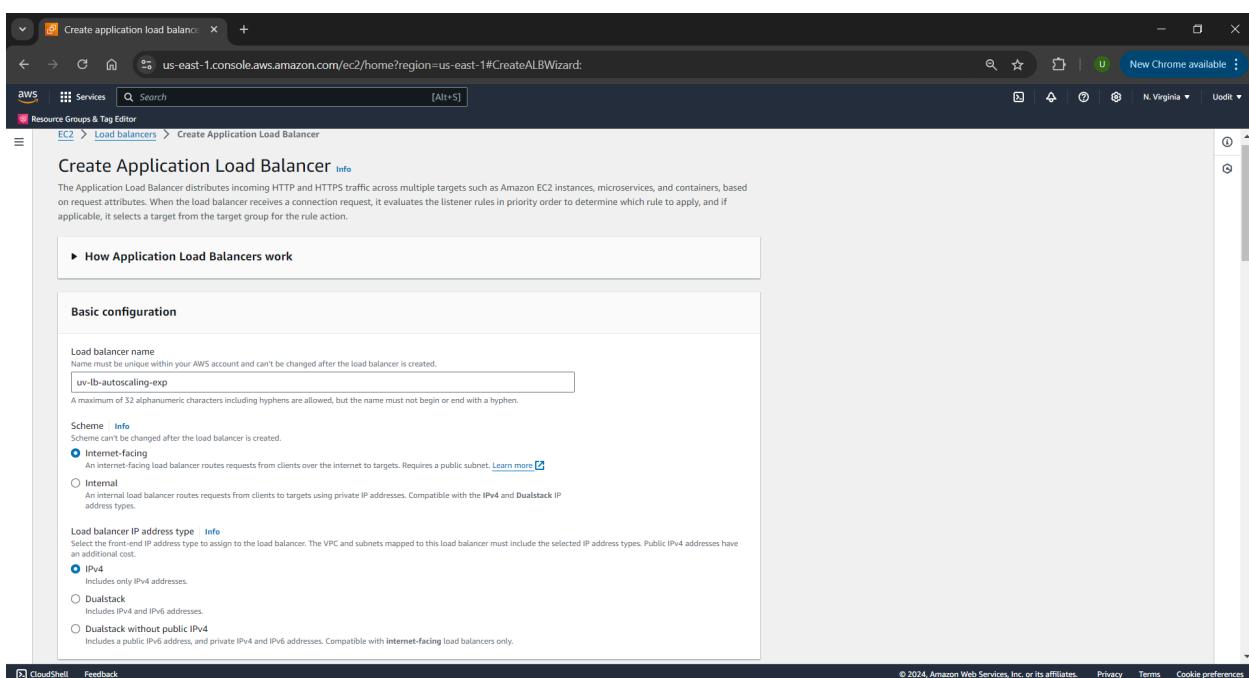
The 'Inbound rules' section lists the two rules defined earlier:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-01dc2018f22fa3d6e	IPv4	HTTP	TCP	80	0.0.0.0/0	-
-	sgr-0350474a8ba668...	IPv4	SSH	TCP	22	0.0.0.0/0	-

## 12. Now we need to create a load balancer. So go to load balance and select application load balancer



Fill and select the details of the load balancer as selected in the image



## Select the created VPC and its subnets

The screenshot shows the 'Create application load balancer' wizard in the AWS CloudFormation console. The current step is 'Network mapping'. In the 'VPC' section, 'uv-test-vpc' is selected. Under 'Availability Zones', 'us-east-1a (use1-az4)' and 'us-east-1b (use1-az6)' are checked. Subnets selected are 'subnet-08748eee1f57a5805' (IPv4 subnet CIDR: 12.0.1.0/24) and 'subnet-028bd75e6d92eb037' (IPv4 subnet CIDR: 12.0.3.0/24). Both are assigned by AWS. In the 'Security groups' section, it says 'A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can create a new security group.' A note at the bottom right indicates '© 2024, Amazon Web Services, Inc. or its affiliates.'

## Select the HTTP security group and the target group created

The screenshot shows the 'Create application load balancer' wizard in the AWS CloudFormation console. The current step is 'Security groups'. It shows two security groups selected: 'uv-sg-autoscaling-exp' (VPC: vpc-0a12383c7208ce9f) and 'default' (VPC: vpc-0a12385cb7208ce9f). Below this, the 'Listeners and routing' section is shown for the 'Listener HTTP:80'. It has one rule: 'Forward to: uv-tg-autoscaling-exp Target type: Instance, IPv4'. A note at the bottom right indicates '© 2024, Amazon Web Services, Inc. or its affiliates.'

## Create the load balancer

The screenshot shows the 'Create application load balancer' wizard in the AWS CloudFormation console. The 'Summary' step is selected, displaying the following configuration:

- Basic configuration:** uv-lb-autoscaling-exp
  - Internet-facing
  - IPv4
- Security groups:** uv-sg-autoscaling-exp (sg-04b0dade5b7973583d), default (sg-04eaae759e4580c47)
- Network mapping:** VPC vpc-0a123b3cb7208ce9f (uv-test-vpc)
  - us-east-1a: subnet-08748ee1f57a5805 (uv-test-subnet-1)
  - us-east-1b: subnet-028bd75e6d92eb037 (uv-test-subnet-2)
- Listeners and routing:** HTTP:80 (defaults to uv-tg-autoscaling-exp)
- Service integrations:** AWS WAF: None, AWS Global Accelerator: None
- Tags:** None
- Attributes:** A note states: "Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer."

At the bottom right, there are 'Cancel' and 'Create load balancer' buttons.

## Load balancer created successfully

The screenshot shows the 'Load balancer details' page for the load balancer 'uv-lb-autoscaling-exp'. The main message is: "Successfully created load balancer: uv-lb-autoscaling-exp. It might take a few minutes for your load balancer to fully set up and route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks." The 'Details' section shows the following information:

Load balancer type	Status	VPC	Load balancer IP address type
Application	Provisioning	vpc-0a123b3cb7208ce9f	IPv4
Scheme	Internet-facing	Hosted zone Z355XD0TRQ7XK	Availability Zones subnet-08748ee1f57a5805 (us-east-1a (use1-az2)) subnet-028bd75e6d92eb037 (us-east-1b (use1-az6))
Load balancer ARN	DNS name info am.awseslasticloadbalancing.us-east-1:904951670971:loadbalancer/app/uv-lb-autoscaling-exp/881cb24dc3c4c50a		

Below the details, there are tabs for 'Listeners and rules', 'Network mapping', 'Resource map - new', 'Security', 'Monitoring', 'Integrations', 'Attributes', and 'Tags'. The 'Listeners and rules' tab is selected, showing a table with one row: "Listeners and rules (1) Info". A note says: "A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules." There is a search bar for 'Filter listeners' and navigation buttons for pages 1 and 2.

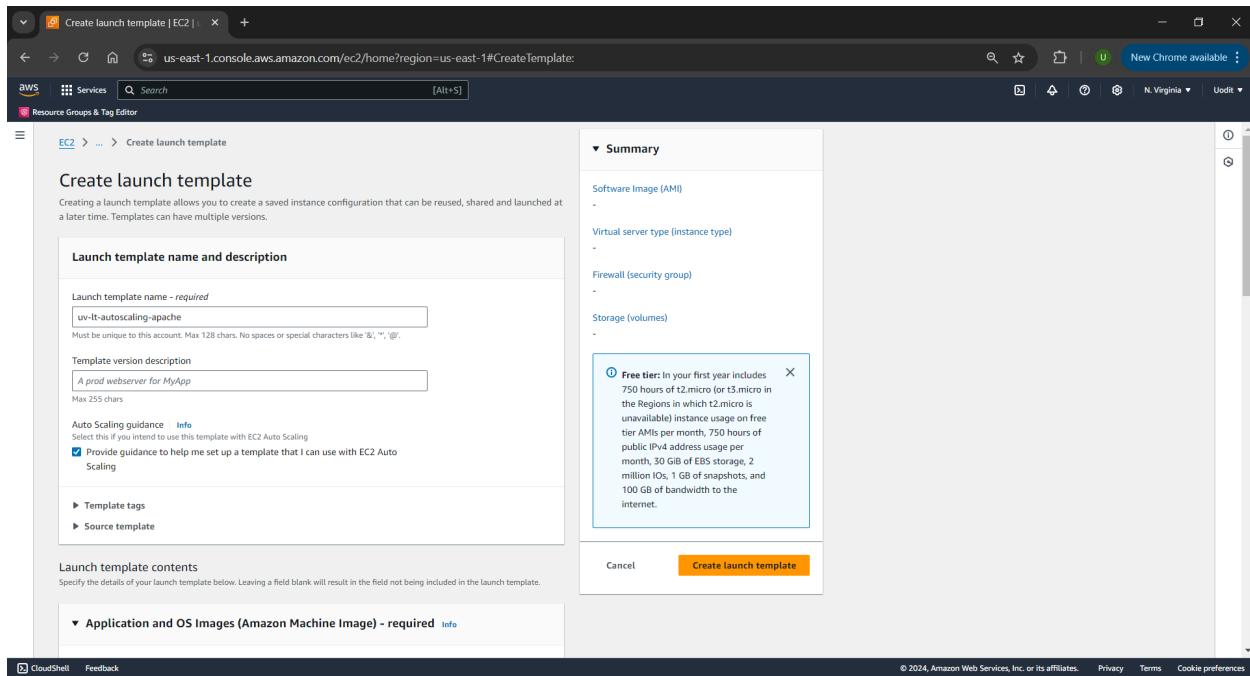
## Wait for few minutes for the status to become active

The screenshot shows the AWS Load Balancer details page. A green banner at the top indicates that the load balancer has been successfully created. The main section displays the load balancer configuration, including its type (Application), status (Active), VPC (vpc-0a12383cb7208ce9f), and IP address type (IPv4). It also shows the ARN and DNS name. Below this, there are tabs for 'Listeners and rules', 'Network mapping', 'Resource map - new', 'Security', 'Monitoring', 'Integrations', 'Attributes', and 'Tags'. The 'Listeners and rules' tab is selected, showing one rule. At the bottom right, there are links for 'Privacy', 'Terms', and 'Cookie preferences'.

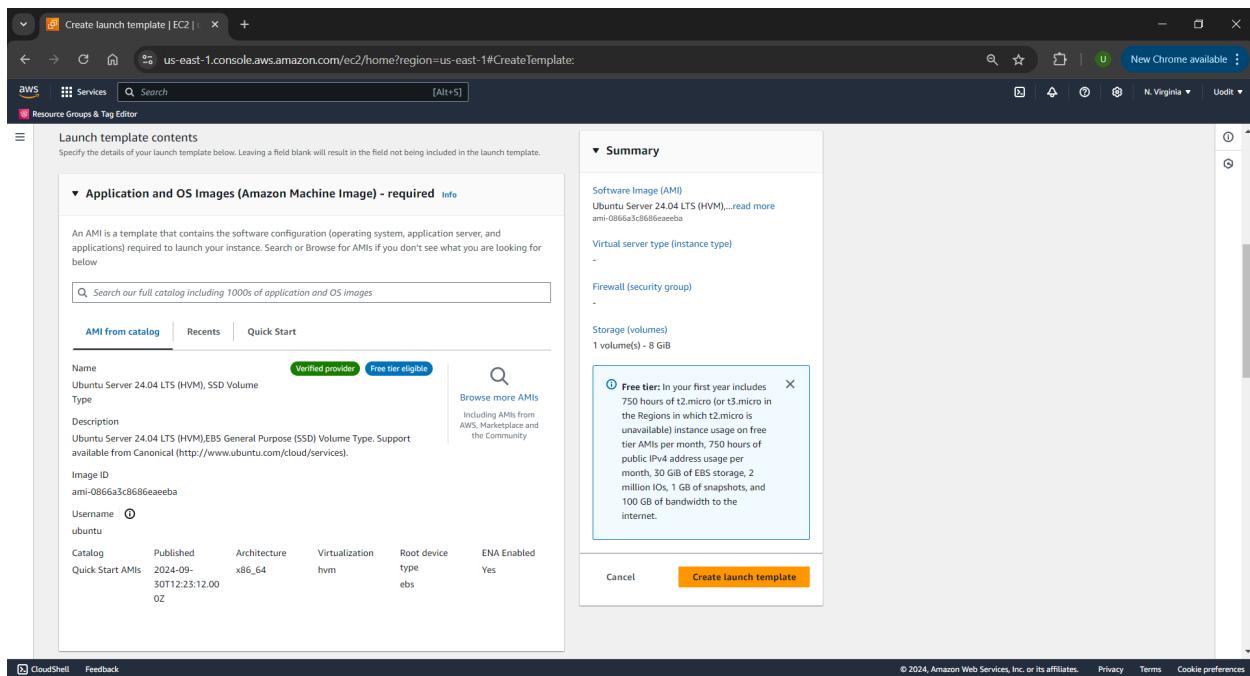
## 13. Go to EC2 launch template and click create

The screenshot shows the AWS EC2 Launch Templates page. The main heading is 'EC2 launch templates' with the subtext 'Streamline, simplify and standardize instance launches'. Below this, there is a brief description of what launch templates do. On the right side, there is a callout box titled 'New launch template' with a prominent orange 'Create launch template' button. To the right of this, there is a 'Documentation' section with links to 'Documentation' and 'API reference'. The left sidebar contains a navigation menu with options like 'Instances', 'Launch Templates', and 'Compute' (which is currently selected). The bottom right corner includes links for 'Privacy', 'Terms', and 'Cookie preferences'.

## Fill the details as per the images



## Select the Ubuntu AMI



## Select the t2.micro instance

The screenshot shows the 'Create launch template' wizard on the AWS EC2 service. In the 'Instance type' section, the 't2.micro' option is selected from the dropdown. A tooltip for 'Free tier eligible' provides details about the free tier benefits. Below the instance type, there's a 'Key pair (login)' section where the key pair 'Don't include in launch template' is chosen. Under 'Network settings', the 'Subnet' dropdown is set to 'Don't include in launch template'. The 'Firewall (security groups)' section shows the 'Select existing security group' button is selected, with the value 'uv-sg-autoscaling-apache-exp'. A tooltip for 'Free tier' is visible in the center-right of the screen. At the bottom right, the 'Create launch template' button is highlighted.

## Select the created security group and also enable auto assign public ip

This screenshot continues the 'Create launch template' process. In the 'Network settings' section, the 'Subnet' dropdown now shows 'vpc-0a1234567890c9f'. The 'Firewall (security groups)' section shows both 'Select existing security group' and 'Create security group' options. Under 'Common security groups', the 'uv-sg-autoscaling-apache-exp' group is listed. The 'Advanced network configuration' section includes a 'Network interface' table with one row (Device index 0) and a 'Subnet' dropdown set to 'Don't include in launch template'. The 'Security groups info' dropdown is set to 'Select security groups' and 'Enable' is selected for 'Auto-assign public IP'. A tooltip for 'Free tier' is visible. At the bottom right, the 'Create launch template' button is highlighted.

## And add this script to install apache web server

```
#!/bin/bash
yes | sudo apt update
yes | sudo apt install apache2
echo "<h1>Server Details</h1><p><strong>Hostname:</strong> $(hostname)</p><p><strong>IP Address:</strong> $(hostname -I | cut -d" " -f1)</p>" > /var/www/html/index.html
sudo systemctl restart apache2
```

The screenshot shows the AWS CloudFormation 'Create launch template' wizard. In Step 3, 'Configure launch template details', the 'User data - optional' field contains the provided shell script. The 'Summary' section on the right shows the configuration: Software Image (AMI) is Ubuntu Server 24.04 LTS (HVM), Virtual server type (instance type) is t2.micro, and Storage (volumes) is 1 volume(s) - 8 GiB. A tooltip for the free tier is displayed, stating: '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 I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.' The 'Create launch template' button is highlighted.

## Instance launched successfully

The screenshot shows the AWS CloudFormation 'Create launch template' wizard. In Step 4, 'Review and launch', a green 'Success' message states 'Successfully created uv-sg-autoscaling-apache(t-0e4530d3c349b6806)'. The 'Next Steps' section lists options: 'Launch an instance', 'Create an Auto Scaling group from your template', 'Create an Auto Scaling group', and 'Create Spot Fleet'. The 'View launch templates' button is highlighted.

## 14. Create an auto scaling group

The screenshot shows the AWS EC2 Auto Scaling landing page. The main content area features a large banner with the heading "Amazon EC2 Auto Scaling" and the subtext "helps maintain the availability of your applications". Below the banner is a diagram titled "How it works" which illustrates an "Auto Scaling group" containing four instances. The diagram labels include "Minimum size" (two solid boxes), "Desired capacity" (one solid box), "Scale out as needed" (one dashed box), and "Maximum size" (four boxes). To the right of the diagram is a box titled "Create Auto Scaling group" with a red "Create Auto Scaling group" button. The left sidebar contains a navigation menu with various AWS services listed, and "Auto Scaling Groups" is currently selected under the "Auto Scaling" section.

Fill the details as per the images

The screenshot shows the "Create Auto Scaling group" wizard, Step 1: Choose launch template or configuration. The "Name" field is set to "uv-esg-autoscaling-exp". The "Launch template" dropdown is set to "uv-it-autoscaling-apache". The "Version" dropdown is set to "Default (1)". The "Description" field shows "uv-it-autoscaling-apache" and "lt-be4550d5c349b6806". The "AMI ID" field shows "ami-0866a5c8686eaeaba". The "Instance type" is "t2.micro". The "Security groups" and "Request Spot Instances" fields are empty.

## Select the VPC and the subnets created

The screenshot shows the AWS Create Auto Scaling group wizard at Step 4: Network configuration. In the 'VPC' section, a dropdown menu is open, showing 'vpc-0x123abc7208ce9f (uv-test-vpc)'. Below it, two subnets are listed: 'us-east-1a | subnet-08748eeec1f57a5805 (uv-test-subnet-1)' and 'us-east-1b | subnet-028bd75ed92eb037 (uv-test-subnet-2)'. Both subnets are selected, indicated by blue outlines.

## Select attach an existing load balancer and also select the load balancer group created

The screenshot shows the AWS Create Auto Scaling group wizard at Step 5: Load balancing. In the 'Load balancing' section, the 'Attach to an existing load balancer' option is selected. Below it, under 'Attach to an existing load balancer', the 'Choose from your load balancer target groups' option is selected. A dropdown menu is open, showing 'uv-tg-autoscaling-exp | HTTP Application Load Balancer: uv-lb-autoscaling-exp'. This target group is highlighted with a blue outline.

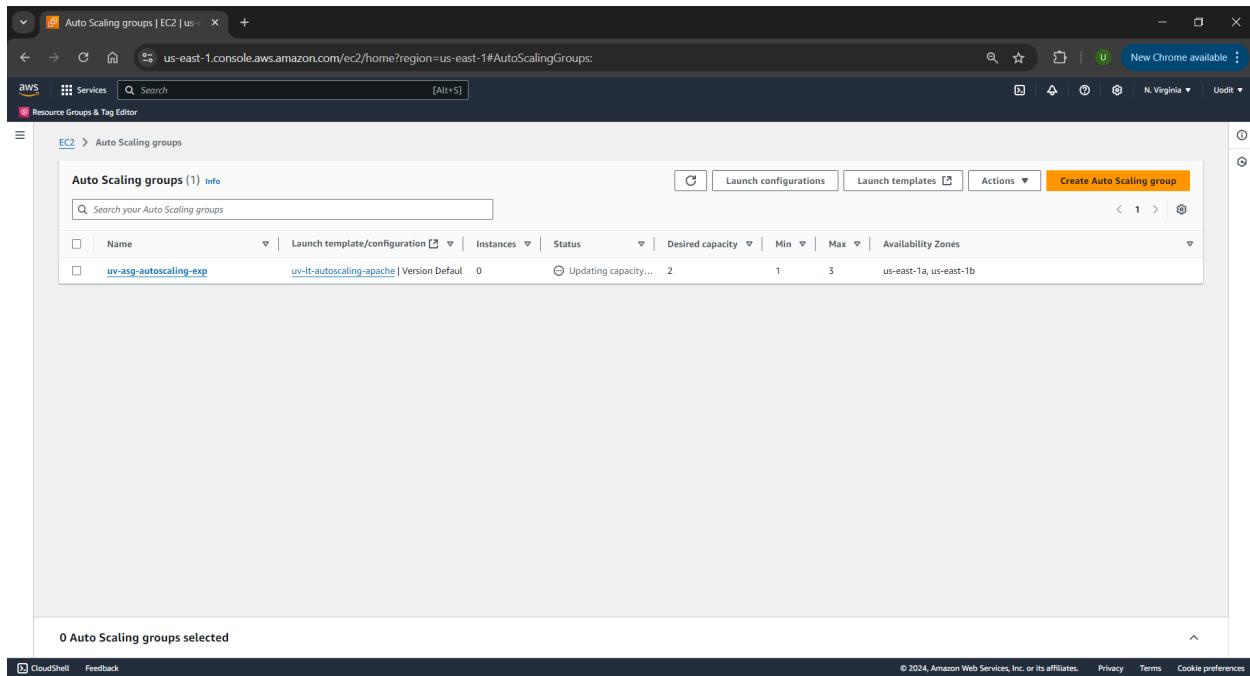
## Enable health check

The screenshot shows the 'Create Auto Scaling group | EC2' interface on the AWS console. The 'Health checks' section is open, displaying options for monitoring instance health. Under 'EC2 health checks', 'Always enabled' is selected. An optional note about turning on ELB health checks is present, with a callout box explaining it allows EC2 Auto Scaling to detect and act on health checks from ELB. Other optional health check types like VPC Lattice and Amazon EBS are listed. A 'Health check grace period' input field is set to 500 seconds. The 'Additional settings' section includes monitoring and CloudWatch metrics collection options.

## Set desired size to 2 and max size to 3

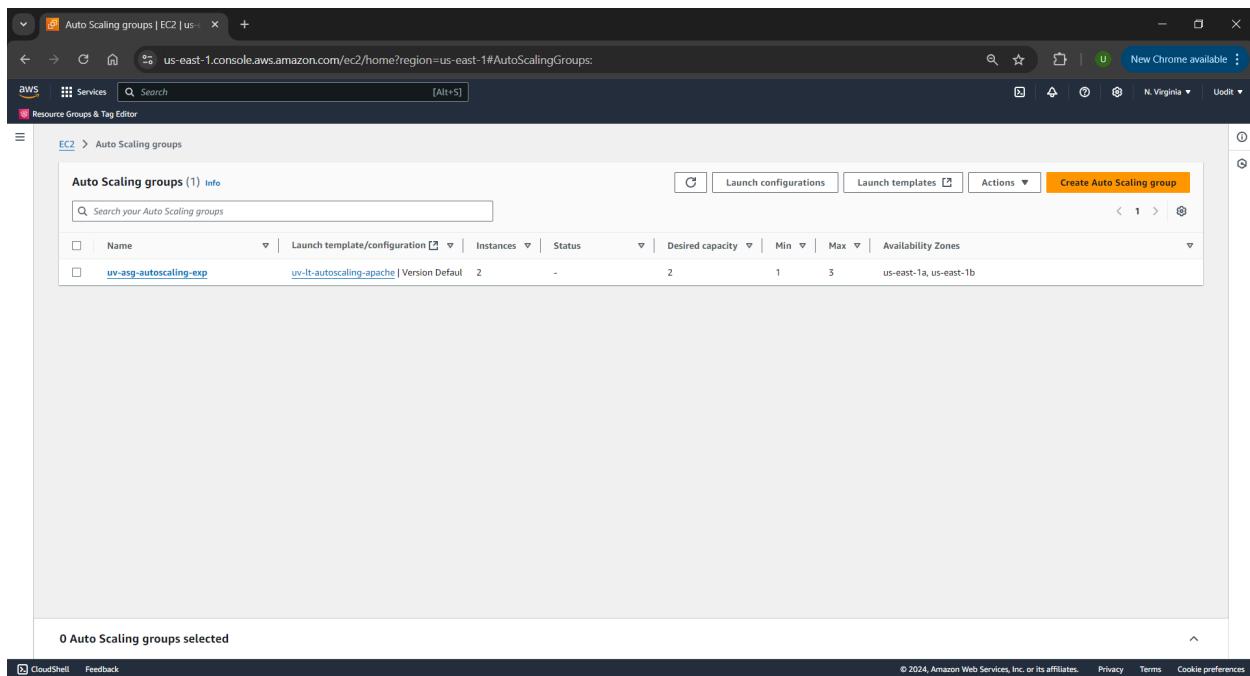
The screenshot shows the 'Create Auto Scaling group | EC2' interface, specifically the 'Configure group size and scaling - optional' step. On the left, a sidebar lists steps 1 through 7. Step 1 is 'Choose launch template or configuration', Step 2 is 'Choose instance launch options', Step 3 is 'optional Configure advanced options', Step 4 is 'optional Configure group size and scaling', Step 5 is 'optional Add notifications', Step 6 is 'optional Add tags', Step 7 is 'Review'. The main panel shows 'Group size' settings where 'Desired capacity' is set to 2. Below that, 'Scaling' settings show 'Min desired capacity' as 1 and 'Max desired capacity' as 3. A note at the bottom explains that scaling limits can be increased or decreased.

## Auto scaling group created successfully



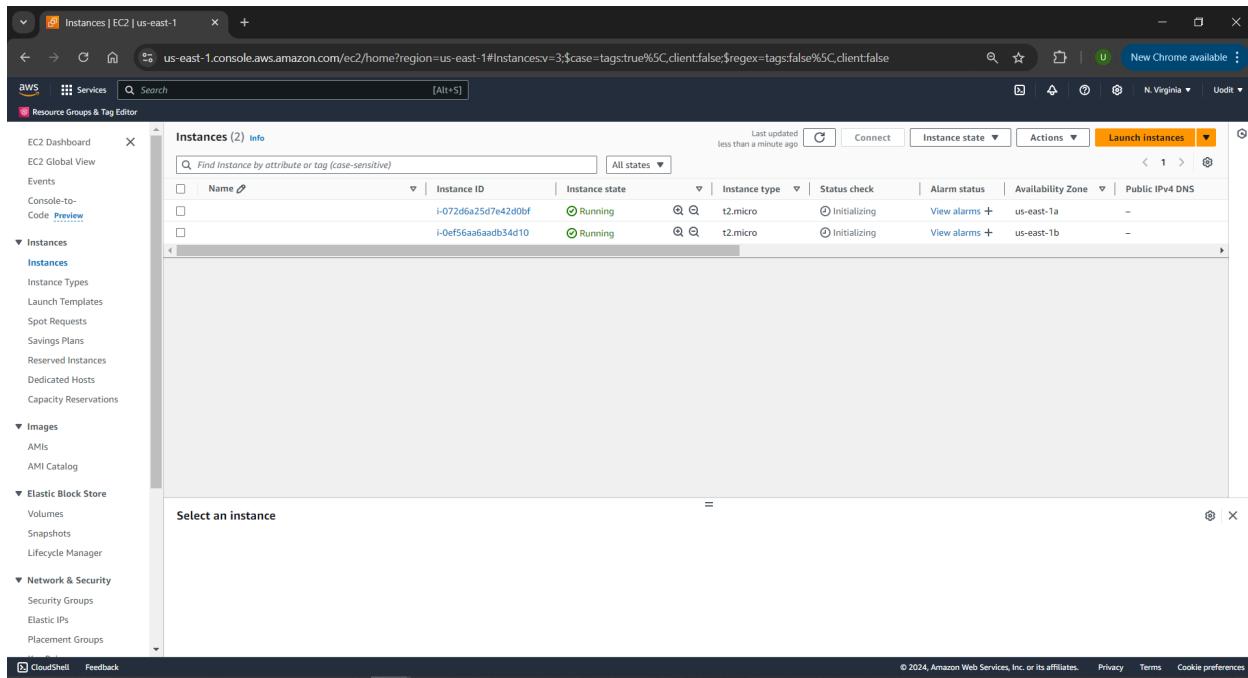
The screenshot shows the AWS Management Console interface for the Auto Scaling groups section. The URL is [us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#AutoScalingGroups](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#AutoScalingGroups). The page displays a table with one row for the auto scaling group 'uv-asg-autoscaling-exp'. The group has a launch template 'uv-it-autoscaling-apache' and a version 'Default'. The desired capacity is set to 2, with a minimum of 1 and a maximum of 3. The availability zones listed are us-east-1a and us-east-1b. The status shows 'Updating capacity...'. A search bar at the top allows searching for 'Auto Scaling groups'. A prominent orange 'Create Auto Scaling group' button is located at the top right.

Wait for some time to create the instance



This screenshot is from the same AWS session as the previous one, showing the same Auto Scaling group 'uv-asg-autoscaling-exp'. However, the status has changed to '2 instances running'. The other details remain the same: launch template 'uv-it-autoscaling-apache', version 'Default', desired capacity 2, min 1, max 3, and availability zones us-east-1a, us-east-1b.

## 15. Two instances has been created automatically by the auto scaling

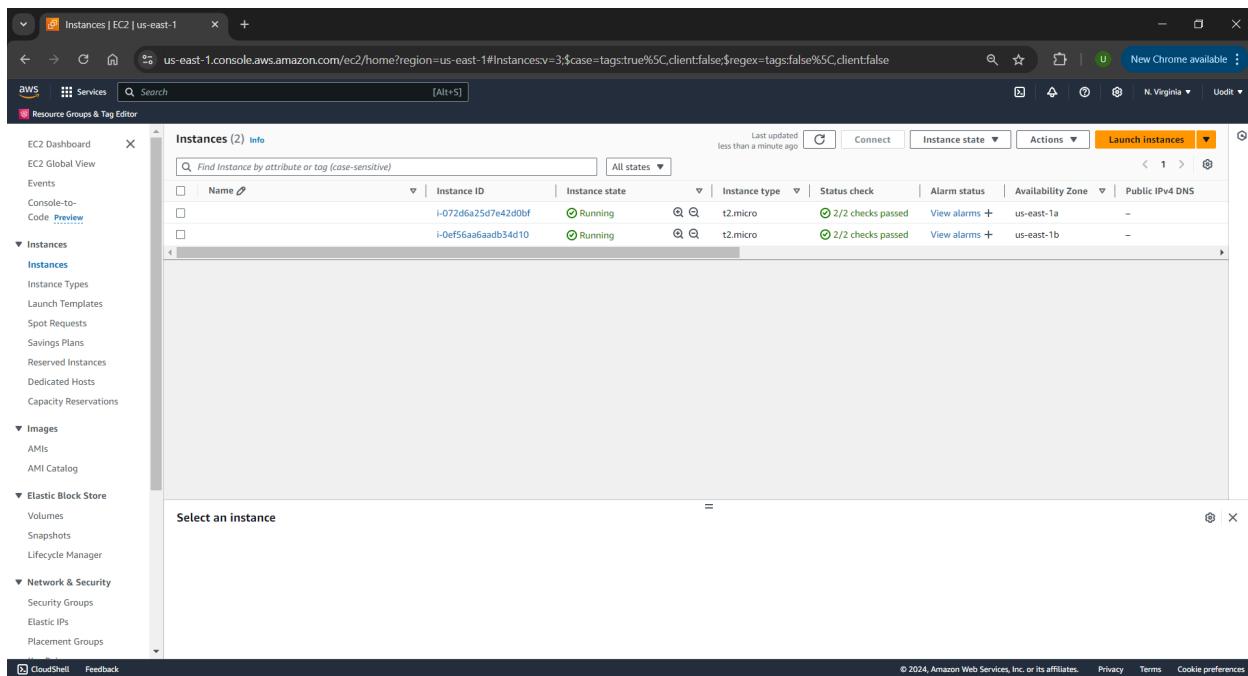


The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table titled "Instances (2) Info" with the following data:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
	i-072d6a25d7e42d0bf	Running	t2.micro	Initializing	View alarms +	us-east-1a	-
	i-0ef56aa6aadb34d10	Running	t2.micro	Initializing	View alarms +	us-east-1b	-

A modal window titled "Select an instance" is open at the bottom of the screen.

After few minutes the instance is running



The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table titled "Instances (2) Info" with the following data:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
	i-072d6a25d7e42d0bf	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
	i-0ef56aa6aadb34d10	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-

A modal window titled "Select an instance" is open at the bottom of the screen.

## 16. Go to the load balancer and copy the DNS and Paste the URL into a new tab to load the webpage

The screenshot shows the AWS CloudWatch Metrics console. In the top navigation bar, there are tabs for 'Metrics' (selected), 'Logs', 'CloudWatch Metrics Insights', and 'CloudWatch Metrics Insights (Preview)'. Below the navigation, there's a search bar with placeholder text 'Search metrics' and a 'Run' button. On the left, there's a sidebar with navigation links for 'Metrics Home', 'Metrics Overview', 'Metrics Data', 'Metrics Insights', 'Metrics Insights (Preview)', 'Metrics Metrics Insights', and 'Metrics Metrics Insights (Preview)'. The main content area displays a table with columns: Metric Name, Metric Type, and Last Value. One row is highlighted in blue, showing 'AWS Lambda Metrics' with 'Function Metrics' as the type and 'Last Value' as '1000'. At the bottom, there are buttons for 'Run' and 'Stop'.

The webpage is hosted from the 1st EC2

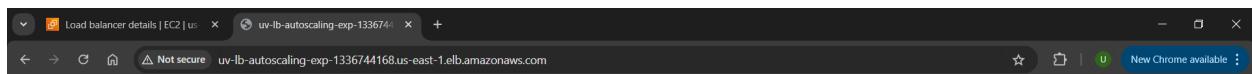
The screenshot shows a web browser window with the URL 'uv-lb-autoscaling-exp-1336744168.us-east-1.elb.amazonaws.com' in the address bar. The page content is mostly blank, indicating the website is currently unavailable or has not loaded fully. The browser interface includes standard controls like back, forward, and search.

### Server Details

Hostname: ip-12-0-1-64

IP Address: 12.0.1.64

**Reload the webpage so the load balancer switches the webpage hosted from the 1st EC2 to 2nd EC2**



### Server Details

Hostname: ip-12-0-3-36

IP Address: 12.0.3.36

---

**From the 2 above images it can be seen that the load balancer automatically switches between the 2 EC2 instances**

## 17. To verify the Auto scaling terminate one EC2 instance

A screenshot of the AWS Management Console EC2 Instances page. The left sidebar shows navigation options like Resource Groups &amp; Tag Editor, Instances, and Network &amp; Security. The main content shows a table of instances. A modal dialog box is open at the top right, stating "Successfully initiated termination (deletion) of i-072d6a25d7e42d0bf". The table lists three instances: one running, one terminated, and one running again. The terminated instance has a checkmark next to its name. Below the table, a detailed view for instance "i-072d6a25d7e42d0bf" is shown, including sections for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The instance summary shows it is terminated.

Since an instance is deleted we can see that 1 healthy and 1 unhealthy target

The screenshot shows the AWS EC2 Target groups page for a target group named "uv-tg-autoscaling-exp". The "Details" section displays the following information:

Target type	Protocol	Protocol version	VPC
Instance	HTTP: 80	HTTP1	vpc-0a12383cb7208ce9f
IP address type	Load balancer		
IPv4	uv-lb-autoscaling-exp		

Below this, a table shows the target status:

Total targets	Healthy	Unhealthy	Unused	Initial	Draining
2	1	1	0	0	0
0 Anomalous					

A note below the table says: "Select values in this table to see corresponding filters applied to the Registered targets table below."

18. After an EC2 instance is terminated automatically another one is created

The screenshot shows the AWS EC2 Instances page. A green banner at the top indicates: "Successfully initiated termination (deletion) of i-072d6a25d7e42d0bf". Below this, a table lists instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
i-05053135cf013dc7	i-072d6a25d7e42d0bf	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
<input checked="" type="checkbox"/>	i-072d6a25d7e42d0bf	Terminated	t2.micro	-	View alarms +	us-east-1a	-
i-0ef56aa6adbd34d10	i-0ef56aa6adbd34d10	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-

And we can see that instead of the deleted instance another has been automatically created and replaced

The screenshot shows the AWS EC2 Target groups page for the same target group "uv-tg-autoscaling-exp". The "Details" section is identical to the previous screenshot. The "Targets" section shows the following registered targets:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
i-05053135cf013dc7	i-072d6a25d7e42d0bf	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	-
i-0ef56aa6adbd34d10	i-0ef56aa6adbd34d10	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	-

At the bottom of the page, there is a note: "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."