

# Implementing Autoscalling Group with Load Balancer

## Introduction

In this project, I designed and implemented an Application Load Balancer (ALB) in conjunction with Auto Scaling Groups (ASGs) to efficiently manage traffic distribution and the dynamic scaling of EC2 instances.

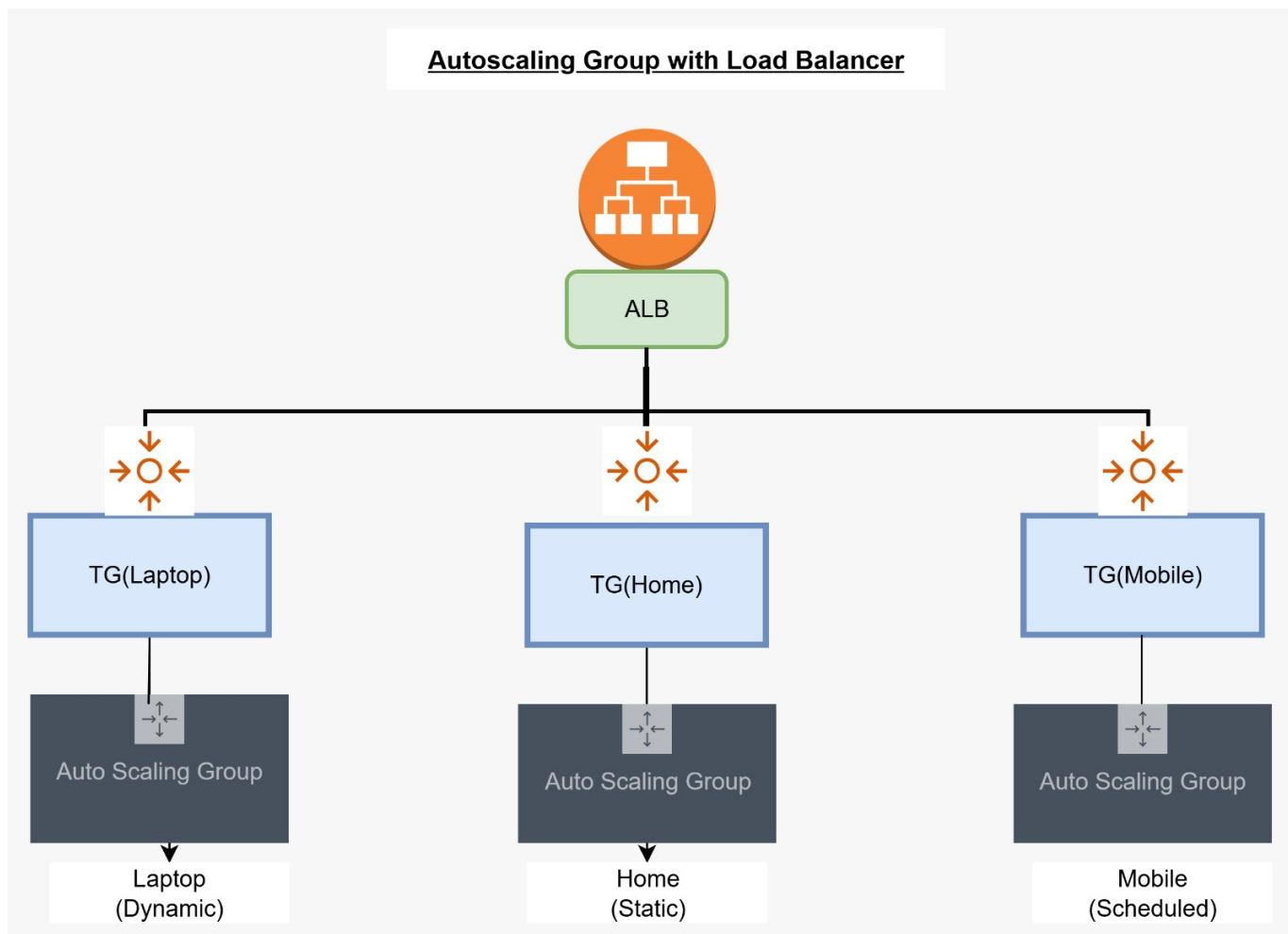
The architecture ensures high availability, fault tolerance, and scalability by routing incoming requests across multiple instances and automatically adjusting the number of instances based on real-time demand.

I configured three distinct Auto Scaling Groups to demonstrate different scaling strategies:

1. Home – a static scaling group with a fixed number of instances,
2. Laptop – a dynamic scaling group that reacts to usage metrics (e.g., CPU utilization) to scale in or out automatically.
3. Mobile – a scheduled scaling group that adjusts capacity based on predefined time-based schedules.

Each Auto Scaling Group is linked to its own Target Group, and all Target Groups are registered with the Application Load Balancer, enabling seamless request routing across the infrastructure.

## Deployment Architecture



# System Essentials

## 1. Application Load Balancer (ALB)

- Acts as the entry point for incoming traffic and intelligently distributes requests across multiple target groups.
- Enhances fault tolerance and ensures high availability by balancing the load among healthy instances.

## 2. Auto Scaling Groups (ASGs)

- To showcase various scaling techniques, I configured three distinct Auto Scaling Groups:
  - Home (Static Scaling): Maintains a fixed number of EC2 instances at all times.
  - Laptop (Dynamic Scaling): Automatically scales the number of instances up or down in response to real-time CPU utilization metrics.
  - Mobile (Scheduled Scaling): Adjusts the number of instances based on a predefined schedule (e.g., scaling out during peak traffic hours and scaling in during off-peak times).

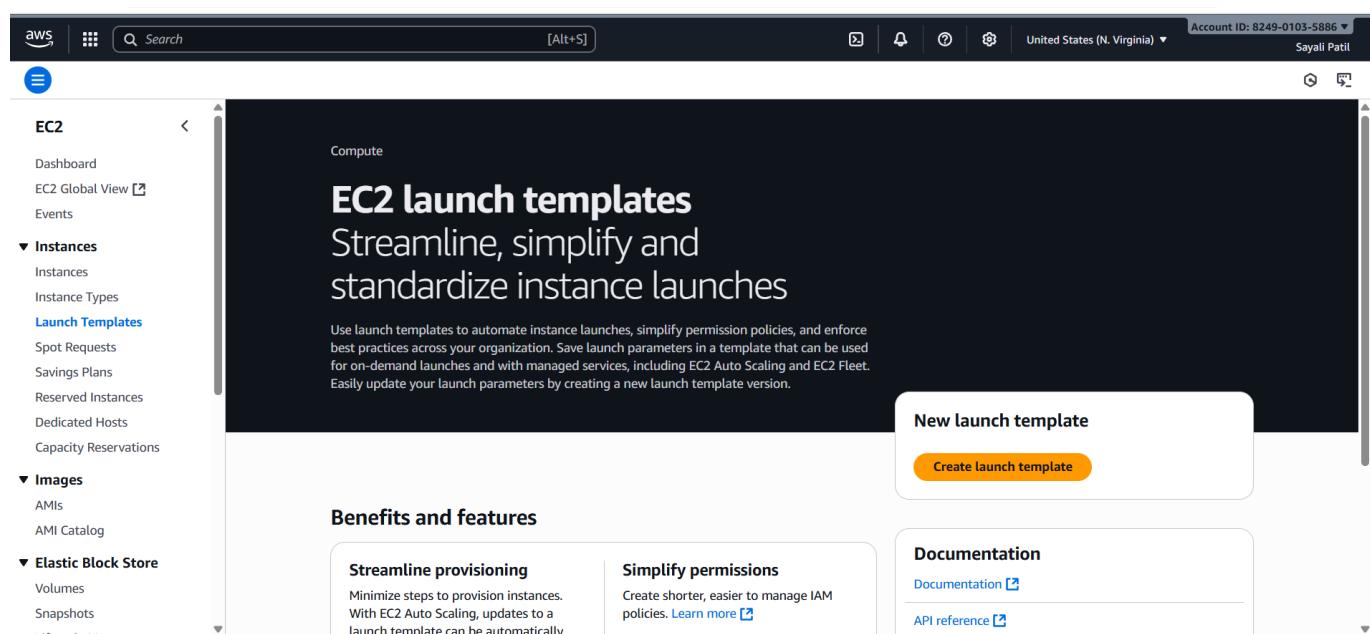
## 3. Target Groups

- Each ASG is linked to a dedicated target group.
- These target groups are integrated with the Application Load Balancer, enabling efficient routing of incoming requests to the appropriate set of instances.

# Steps to Setup

## Step 1: Launch Template

- Create 3 Launch Templates: home, laptop and mobile (each with unique User Data scripts)
- In an EC2 Instance, Navigate to the Launch Templates section in your AWS Management Console.



- Click on "Create Launch Template" to begin.
- Enter a Template Name (e.g., home, laptop, mobile)
- Write Discription in it.

**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*

home-LT

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

**Template version description**

This is my home server!

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**

**Launch template contents**

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

- Select an AMI from the Quick Start options (I selected Amazon Linux). Alternatively, you can use a custom AMI if you've previously created one.

**Application and OS Images (Amazon Machine Image)** [Info](#)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search catalog or choose [Browse more AMIs](#).

Search our full catalog including 1000s of application and OS images

**Recents** **Quick Start**

Don't include in launch template **Amazon Linux** AWS macOS Ubuntu Windows Red Hat SUSE Linux Debian

Browse more AMIs Including AMIs from AWS, Marketplace and the Community

**Amazon Machine Image (AMI)**

Amazon Linux 2023 kernel-6.1 AMI  
ami-08982f1c5bf93d976 (64-bit (x86), uefi-preferred) / ami-039f81f5ce6752b10 (64-bit (Arm), uefi)  
Virtualization: hvm ENA enabled: true Root device type: ebs

**Description**

Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.8.20250915.0 x86\_64 HVM kernel-6.1

- Select a Instance type and Key Pair

The screenshot shows the AWS EC2 'Create launch template' page. It includes sections for 'Instance type' (t3.micro), 'Key pair (login)', and 'Network settings'. The 'Summary' section on the right lists the selected AMI (Amazon Linux 2023), instance type (t3.micro), and storage (1 volume(s) - 8 GiB). A 'Create launch template' button is visible.

- Configure the Network Settings: Since this is a basic application, only port 80 (HTTP) and port 22 (SSH) are required. You can either create a new security group with these rules or select an existing security group that already allows this traffic.

The screenshot shows the AWS EC2 'Create launch template' page. It includes sections for 'Security groups' (selected: launch-wizard-9), 'Storage (volumes)' (EBS Volumes: Volume 1 (AMI Root) - 8 GiB, EBS, General purpose SSD (gp3), 3000 IOPS), and 'Resource tags'. A 'Create launch template' button is visible.

- Add user data script in advanced options (you don't need to add this script if you have selected your own AMI)

User data - optional | [Info](#)  
Upload a file with your user data or enter it in the field.

[Choose file](#)

```
#!/bin/bash
sudo yum update -y
sudo yum install httpd -y
sudo systemctl start httpd
sudo systemctl enable httpd
echo "<h1>This is a home page $(hostname -f)</h1>" > /var/www/html/index.html
```

User data has already been base64 encoded

**Summary**

**Software Image (AMI)**  
Amazon Linux 2023 AMI 2023.8.2... [read more](#)  
ami-08982f1c5bf93d976

**Virtual server type (instance type)**  
t3.micro

**Firewall (security group)**  
launch-wizard-9

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

[Cancel](#) [Create launch template](#)

- Click on Launch Template
- Similarly, Create other templates for Mobile and Laptop

User datascript for mobile-

```
#!/bin/bash
sudo yum update -y
sudo yum install -y httpd
sudo systemctl start httpd
sudo systemctl enable httpd

sudo mkdir /var/www/html/mobile
echo "<h1>This is the mobile page $(hostname -f)</h1>" >
/var/www/html/mobile/index.html
```

User datascript for laptop-

```
#!/bin/bash
sudo yum update -y
sudo yum install -y httpd
sudo systemctl start httpd
sudo systemctl enable httpd

sudo mkdir /var/www/html/laptop
echo "<h1>This is the laptop page $(hostname -f)</h1>" >
/var/www/html/laptop/index.html
```

**User data - optional**

```
#!/bin/bash
sudo yum update -y
sudo yum install httpd -y
sudo systemctl start httpd
sudo systemctl enable httpd

sudo mkdir /var/www/html/laptop
echo "<h1>This is a laptop page ${hostname -f}</h1>" >
/var/www/html/laptop/index.html
```

**Summary**

- Software Image (AMI)**: Amazon Linux 2023 AMI 2023.8.2...read more  
ami-08982f1c5bf93d976
- Virtual server type (instance type)**: t3.micro
- Firewall (security group)**: launch-wizard-9
- Storage (volumes)**: 1 volume(s) - 8 GiB

**Create launch template**

**User data - optional**

```
#!/bin/bash
sudo yum update -y
sudo yum install httpd -y
sudo systemctl start httpd
sudo systemctl enable httpd

sudo mkdir /var/www/html/mobile
echo "<h1>This is a mobile page ${hostname -f}</h1>" >
/var/www/html/mobile/index.html
```

User data has already been base64 encoded

**Summary**

- Software Image (AMI)**: Amazon Linux 2023 AMI 2023.8.2...read more  
ami-08982f1c5bf93d976
- Virtual server type (instance type)**: t3.micro
- Firewall (security group)**: launch-wizard-9
- Storage (volumes)**: 1 volume(s) - 8 GiB

**Create launch template**

- Successfully, we created a Launch Template

**Launch Templates (3)**

Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By
lt-0afbcad5feb599e13	mobile-LT	1	1	2025-09-29T09:08:28.000Z	arn:aws:iam::8249
lt-0c525ae7b611f0a1b	laptop-LT	1	1	2025-09-29T09:06:32.000Z	arn:aws:iam::8249
lt-07e168364a3cc1ede	home-LT	1	1	2025-09-29T09:03:52.000Z	arn:aws:iam::8249

**Select a launch template**

## Step 2: Creating AutoScaling Group (ASG)

Let's begin by creating the Auto Scaling Group for home, which uses static Scaling.

1. Navigate to the Auto Scaling Groups section in the AWS Management Console.

2. Click on "Create Auto Scaling Group" to get started.

3. You'll notice there are 7 setup steps listed on the left-hand side of the screen to guide you through the configuration process.

### #Choose Launch Template

- Give Name to autoscalling group.

- Select a Launch Template of home for home autoscalling group.
- click on next & go to next step.

## #Choose instance launch option

- Select Availability Zone. (Atleast 2)
- go to next step.

The screenshot shows the 'Create Auto Scaling group' wizard at Step 2. It lists several subnets under 'Availability Zones' and allows creating a new subnet. Under 'Balanced best effort' and 'Balanced only' strategies, it notes that launching in one zone will attempt to launch in another. A warning message states that the requested instance type (t3.micro) is not available in one availability zone and suggests changing the instance type or choosing other zones. Navigation buttons at the bottom include 'Cancel', 'Skip to review', 'Previous', and 'Next'.

## #Integrate with other services

- We'll integrate these Auto Scaling Groups with the Load Balancer after all three have been created. For now, let's proceed to the next step.

The screenshot shows the 'Create Auto Scaling group' wizard at Step 3. It includes a sidebar with steps 1-7. The current step, 'Step 3 - optional: Integrate with other services', is selected. It contains sections for 'Load balancing' and 'VPC Lattice integration options'. In 'Load balancing', the 'No load balancer' option is chosen. In 'VPC Lattice integration options', the 'No VPC Lattice service' option is chosen. Navigation buttons at the bottom include 'Create new VPC Lattice service' and 'Next'.

**Health checks**

Health checks increase availability by replacing unhealthy instances. When you use multiple health checks, all are evaluated, and if at least one fails, instance replacement occurs.

**EC2 health checks**

(i) Always enabled

**Additional health check types - optional** (Info)

- Turn on Elastic Load Balancing health checks  
Elastic Load Balancing monitors whether instances are available to handle requests. When it reports an unhealthy instance, EC2 Auto Scaling can replace it on its next periodic check.
- Turn on VPC Lattice health checks  
VPC Lattice can monitor whether instances are available to handle requests. If it considers a target as failed a health check, EC2 Auto Scaling replaces it after its next periodic check.
- Turn on Amazon EBS health checks  
EBS monitors whether an instance's root volume or attached volume stalls. When it reports an unhealthy volume, EC2 Auto Scaling can replace the instance on its next periodic health check.

**Health check grace period** (Info)

This time period delays the first health check until your instances finish initializing. It doesn't prevent an instance from terminating when placed into a non-running state.

300 seconds

Cancel Skip to review Previous Next

## #Configure Group size and scaling

- For Home (static) ASG select : Desired = 2 , min = 2 , max = 2.
- Here for mobile we are giving : Desired = 3 , min = 2 , max = 7. (After creating mobile ASG we will make it scheduled)
- For Laptop (Dynamic) ASG select : Desired = 3 , min = 2 , max = 7.

**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

**Configure group size and scaling - optional** (Info)

Define your group's desired capacity and scaling limits. You can optionally add automatic scaling to adjust the size of your group.

**Group size** (Info)

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.

2

**Scaling** (Info)

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.

<b>Min desired capacity</b>	<b>Max desired capacity</b>
2	2

Equal or less than desired capacity      Equal or greater than desired capacity

**Automatic scaling - optional**

- select target tracking scaling policy ( if your ASG is static you don't need policy so select No scaling policy)

**Automatic scaling - optional**

**Choose whether to use a target tracking policy** Info  
You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

No scaling policies  
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

Target tracking scaling policy  
Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

**Instance maintenance policy** Info  
Control your Auto Scaling group's availability during instance replacement events. This includes health checks, instance refreshes, maximum instance lifetime features and events that happen automatically to keep your group balanced, called rebalancing events.

**Choose a replacement behavior depending on your availability requirements**

- Mixed behavior**
- No policy**  
For rebalancing events, new instances will launch before terminating others. For all other events, instances terminate and launch at the same time.
- Prioritize availability**  
Launch new instances and wait for them to be ready before terminating others. This allows you to go above your desired capacity by a given percentage and may temporarily increase costs.
- Control costs**
- Terminate and launch**  
Terminate and launch instances at the same time. This allows you to go below your desired capacity by a given percentage and may temporarily reduce availability.
- Flexible**
- Custom behavior**  
Set custom values for the minimum and maximum amount of available capacity. This gives you greater flexibility in setting how far below and over your desired capacity EC2 Auto Scaling goes when replacing instances.

**Additional capacity settings**

**Capacity Reservation preference** Info  
Select whether you want Auto Scaling to launch instances into an existing Capacity Reservation or Capacity Reservation resource group.

- Go to the next

Select whether you want Auto Scaling to launch instances into an existing Capacity Reservation or Capacity Reservation resource group.

**Default**  
Auto Scaling uses the Capacity Reservation preference from your launch template.

**None**  
Instances will not be launched into a Capacity Reservation.

**Capacity Reservations only**  
Instances will only be launched into a Capacity Reservation. If capacity isn't available, the instances fail to launch.

**Capacity Reservations first**  
Instances will attempt to launch into a Capacity Reservation first. If capacity isn't available, instances will run in On-Demand capacity.

**Additional settings**

**Instance scale-in protection**  
If protect from scale in is enabled, newly launched instances will be protected from scale in by default.  
 Enable instance scale-in protection

**Monitoring** Info  
 Enable group metrics collection within CloudWatch

**Default instance warmup** Info  
The amount of time that CloudWatch metrics for new instances do not contribute to the group's aggregated instance metrics, as their usage data is not reliable yet.  
 Enable default instance warmup

**Cancel** **Skip to review** **Previous** **Next**

#Add notification

#Add tags

(you dont have to do anything in step 5 and 6 so just click on 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**

**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**

**Add tags - optional** Info

Add tags to help you search, filter, and track your Auto Scaling group across AWS. You can also choose to automatically add these tags to instances when they are launched.

ⓘ You can optionally choose to add tags to instances (and their attached EBS volumes) by specifying tags in your launch template. We recommend caution, however, because the tag values for instances from your launch template will be overridden if there are any duplicate keys specified for the Auto Scaling group.

**Tags (0)**

**Add tag**

50 remaining

**Cancel** **Previous** **Next**

## #Review

- select create autoscaling group

**Review** Info

**Step 1: Choose launch template**

**Group details**

Auto Scaling group name  
home-ASG

**Launch template**

Launch template  
home-LT Edit

Version  
Default

Description  
This is my home server

**Step 2: Choose instance launch options**

**Network**

VPC  
vpc-070d03d6e19cfe3b8 Edit

**Availability Zones and subnets**

Availability Zone | Subnet | Subnet CIDR range

**Capacity Reservation preference**

Preference  
Default

Capacity Reservation IDs  
-

Resource Groups  
-

**Step 5: Add notifications**

**Notifications**

No notifications Edit

**Step 6: Add tags**

**Tags (0)**

Key	Value	Tag new instances
No tags		

**Preview code** **Cancel** **Previous** **Create Auto Scaling group**

- Similarly create other 2 ASG according to their type dynamic and scheduling. Just need to change the group size as i mentioned.
- Only select target tracking scaling policy ( if your ASG is dynamic or scheduled you need policy so select target tracking scaling policy)
- Just need to change the group size as i mentioned.
  - Here, for mobile we are giving : Desired = 3 , min = 2 , max = 7. (After creating mobile ASG we will make it scheduled)
  - For Laptop (Dynamic) ASG select : Desired = 3 , min = 2 , max = 7.

The screenshot shows the AWS EC2 Auto Scaling groups page. At the top, there's a green success message: "mobile-ASG, 1 Scaling policy created successfully". Below it, the "Auto Scaling groups (3)" section is displayed. The table shows the following data:

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
mobile-ASG	mobile-LT   Version Default	3	-	3	2	7	6 Availability Zones
laptop-ASG	laptop-LT   Version Default	3	-	3	2	7	6 Availability Zones
home-ASG	home-LT   Version Default	2	-	2	2	2	6 Availability Zones

### Step 3: Now to make Mobile ASG Scheduled follow following steps

- Click on Mobile ASG and select Automatic scaling (You will see the following interface)

The screenshot shows the AWS EC2 Auto Scaling groups page with the sidebar expanded. The sidebar includes sections for Elastic Block Store, Network & Security, and Load Balancing. The main content area is identical to the previous screenshot, showing the three Auto Scaling groups: mobile-ASG, laptop-ASG, and home-ASG.

**mobile-ASG Capacity overview**

Desired capacity: 3    Scaling limits (Min - Max): 2 - 7    Desired capacity type: Units (number of instances)    Status: -

Date created: Mon Sep 29 2025 14:54:02 GMT+0530 (India Standard Time)

Automatic scaling (selected)

Scaling policies resize your Auto Scaling group to meet changes in demand. With reactive dynamic scaling policies, you can track specific CloudWatch metrics and take action when the CloudWatch alarm threshold is met. Use predictive scaling policies along with dynamic scaling policies in the following situations: when your application demand changes quickly, but with a recurring pattern, or when your EC2 instances require more time to initialize.

**Dynamic scaling policies (1) Info**

C Actions ▾ Create dynamic scaling policy

- Scroll down to bottom.
- Click on Create schedule action.

Evaluation period

Evaluation based on 2 days

No predictive scaling policies have been created

Predictive scaling policies use historical data to scale out your group ahead of forecasted hourly load.

**Scheduled actions (0) Info**

C Actions ▾ Create scheduled action

Filter scheduled actions

Name	Start time	End time	Recurrence	Time zone	Desired c...	Min	Max
------	------------	----------	------------	-----------	--------------	-----	-----

No scheduled actions are currently specified

Create scheduled action

- Give name to your schedule.
- Enter Desired , min and max values of instances.
- Choose Cron option in recurrence and give specific time in side box ( sequence is - minute hour date month and day of week)
- Enter the ending time with date
- Click on Create

The screenshot shows the AWS EC2 Auto Scaling Groups page for a group named 'mobile-ASG'. On the left, a sidebar lists various services like Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The main area is titled 'Evaluation period' and contains fields for 'Name' (set to 'Greateindiansale'), 'Desired capacity' (set to 3), 'Min' (set to 2), 'Max' (set to 7), 'Recurrence' (set to 'Cron' with the value '0 10 21 10 \*'), and 'Time zone' (set to 'Etc/UTC'). Below these are sections for 'Scheduled actions' and 'Specific start time'. A note says 'Current time in selected time zone is 2025-09-29/09:31 UTC'. Under 'Specific start time', there are fields for 'YYYY/MM/DD' (set to '2025/10/26') and '00:00' (under 'End by'). The 'Create' button is highlighted in orange at the bottom right.

The screenshot shows the same AWS EC2 Auto Scaling Groups page after the scheduled action has been created. A green success message at the top says 'Scheduled action created or edited successfully'. The main area now shows the 'Evaluation period' settings and a message stating 'No predictive scaling policies have been created'. Below this is a 'Create predictive scaling policy' button. At the bottom, a section titled 'Scheduled actions (1/1)' displays the newly created action with details: Name 'Greateindian...', Start time '2025 Octob...', End time '2025 Octob...', Recurrence '0 10 21 10 \*', Time zone 'Etc/UTC', Desired capacity '3', Min '2', and Max '7'. The 'Actions' and 'Create scheduled action' buttons are also visible.

- If you click on the Mobile Auto Scaling Group and scroll down to the Scheduled Actions section, you'll see the scheduled action that has been created.

#### Step 4: Creating Target Group For each Autoscalling Group

- Go to Target Groups
- Click on Create Target Group

The screenshot shows the AWS EC2 Target groups page. On the left, there's a navigation sidebar with sections for Elastic Block Store, Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), Load Balancing (Load Balancers, Target Groups, Trust Stores), and Auto Scaling (Auto Scaling Groups). The main area is titled "Target groups" and has a search bar at the top. It displays a message: "No target groups" and "You don't have any target groups in us-east-1". A blue "Create target group" button is located at the bottom right of this section. Below this, another message says "0 target groups selected" and "Select a target group above."

- Give a Name to your Target Group.

This screenshot shows the "Create target group" wizard. Step 1 is titled "Set target group properties". It includes a summary of Lambda functions (Facilitates routing to a single Lambda function, Accessible to Application Load Balancers only) and a section for an Application Load Balancer (Offers flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC, Facilitates using static IP addresses and PrivateLink with an Application Load Balancer). The "Target group name" field is filled with "home-TG". The "Protocol" dropdown is set to "HTTP" and the "Port" input is "80". The "IP address type" section shows "IPv4" selected (Each instance has a default network interface (eth0) that is assigned the primary private IPv4 address. The instance's primary private IPv4 address is the one that will be applied to the target). The "IPv6" option is also present (Each instance you register must have an assigned primary IPv6 address. This is configured on the instance's default network interface (eth0)).

- Scroll down and go to health check section.
- Give a path of your application in Health Check Path section. [Note :For home, i Gave "/" as a path . Here "/" represten the directry " /var/www/html/" which is defualt directry of httpd]
- click on next

The screenshot shows the 'Create target group' wizard on the AWS EC2 console. The current step is 'Set health check configuration'. It includes sections for 'Health checks', 'Attributes', and 'Tags - optional'. The 'Health checks' section is expanded, showing 'Health check protocol' set to 'HTTP' and 'Health check path' set to '/'. The 'Attributes' section contains a note about default attributes being applied. The 'Tags - optional' section suggests adding tags to categorize resources. At the bottom right are 'Cancel' and 'Next' buttons.

- Click on Create Target Group (don't need to change anything else)

The screenshot shows the 'Create target group' wizard on the AWS EC2 console, now at Step 2: 'Review targets'. It displays a summary of the target group configuration, including the selected instances and their ports. The 'Ports for the selected instances' section shows port 80. The 'Review targets' section lists 0 pending targets. At the bottom right are 'Cancel', 'Previous', and 'Create target group' buttons.

- Similarly Create Target Groups for Laptop and Mobile.
- You only need to change the path in Health check section as given below.

For Laptop :

**Target group name**  
laptop-TG  
A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

**Protocol**  
Protocol for load balancer-to-target communication. Can't be modified after creation.  
HTTP 80 1-65535

**IP address type**  
Only targets with the indicated IP address type can be registered to this target group.  
 **IPv4**  
 Each instance has a default network interface (eth0) that is assigned the primary private IPv4 address. The instance's primary private IPv4 address is the one that will be applied to the target.

**IPv6**  
 Each instance you register must have an assigned primary IPv6 address. This is configured on the instance's default network interface (eth0). [Learn more](#)

**VPC**  
Select the VPC with the instances that you want to include in the target group. Only VPCs that support the IP address type selected above are available in this list.  
vpc-070d03d6e19cfe3b8 (default) [Create VPC](#)

**Protocol version**  
 **HTTP1.1**  
 Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

**Health check protocol**  
HTTP

**Health check path**  
Use the default path of "/" to perform health checks on the root, or specify a custom path if preferred.  
/laptop/ Up to 1024 characters allowed.

**Advanced health check settings**

**Attributes**  
 Certain default attributes will be applied to your target group. You can view and edit them after creating the target group.

**Tags - optional**  
Consider adding tags to your target group. Tags enable you to categorize your AWS resources so you can more easily manage them.

[Cancel](#) [Next](#)

## For Mobile :

**Target group name**  
mobile-TG  
A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

**Protocol**  
Protocol for load balancer-to-target communication. Can't be modified after creation.  
HTTP 80 1-65535

**IP address type**  
Only targets with the indicated IP address type can be registered to this target group.  
 **IPv4**  
 Each instance has a default network interface (eth0) that is assigned the primary private IPv4 address. The instance's primary private IPv4 address is the one that will be applied to the target.

**IPv6**  
 Each instance you register must have an assigned primary IPv6 address. This is configured on the instance's default network interface (eth0). [Learn more](#)

**VPC**  
Select the VPC with the instances that you want to include in the target group. Only VPCs that support the IP address type selected above are available in this list.  
vpc-070d03d6e19cfe3b8 (default) [Create VPC](#)

**Protocol version**  
 **HTTP1.1**  
 Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

**Health check protocol**: HTTP

**Health check path**: /mobile/

**Advanced health check settings**

**Attributes**: Certain default attributes will be applied to your target group. You can view and edit them after creating the target group.

**Tags - optional**: Consider adding tags to your target group. Tags enable you to categorize your AWS resources so you can more easily manage them.

**Cancel** **Next**

- Now, three Target Groups are created

**Target groups (3) Info**

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
home-TG	arn:aws:elasticloadbalancing:us-east-1:8249-0103-5886:targetgroup/mobile-TG/53d1f3434e3a4a20	80	HTTP	Instance	None associated	vpc-070c
laptop-TG	arn:aws:elasticloadbalancing:us-east-1:8249-0103-5886:targetgroup/laptop-TG/53d1f3434e3a4a20	80	HTTP	Instance	None associated	vpc-070c
mobile-TG	arn:aws:elasticloadbalancing:us-east-1:8249-0103-5886:targetgroup/mobile-TG/53d1f3434e3a4a20	80	HTTP	Instance	None associated	vpc-070c

**0 target groups selected**

Select a target group above.

## Step 5: Connecting Autoscaling group to Target Groups

- Select an Autoscaling Group .
- Click on Action at top right corner.
- Click on Edit

**Scheduled action created or edited successfully**

**Auto Scaling groups (1/3) Info**

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
mobile-ASG	mobile-LT   Version Default	2	-	2	2	7
laptop-ASG	laptop-LT   Version Default	2	-	2	2	7
home-ASG	home-LT   Version Default	2	-	2	2	2

**Auto Scaling group: mobile-ASG**

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

\* Scroll down and you will see the Load balancing section.

- Click on Application load balancer
- Select a Target group for autoscalling group we selected. (for laptop ASG select Laptop target group, for mobile ASG select Mobile Target group and for home ASG select home target group.)

**Load balancing - optional**

**Load balancers**

Application, Network or Gateway Load Balancer target groups  
Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.  
Select target groups

mobile-TG | HTTP  
Load balancer: Not associated with any load balancer

**Note:** One of your target groups is not yet associated with any load balancer. In order for routing and scaling to occur, you will need to attach the target group to a load balancer. This can be done later in the [Load Balancing console](#).

Classic Load Balancers

**Create and attach new load balancers**

- click on update.

**Suspended processes** [Info](#)  
Select suspended processes

**Maximum instance lifetime**  
seconds

**Default cooldown** [Info](#)  
300 seconds

**Default instance warmup** [Info](#)  
The amount of time that CloudWatch metrics for new instances do not contribute to the group's aggregated instance metrics, as their usage data is not reliable yet.  
 Enable default instance warmup

**Tags (0)**  
  
50 remaining

- Successfully connected the Autoscalling group and target groups.
- Repeat process to connect each autoscalling group to its target group.

## Step 6: Setting up an Application Load Balancer and linking it to target groups.

- Go to Load Balancers and Click on Create Load Balancer

The screenshot shows the AWS EC2 Load Balancers page. On the left, there's a navigation sidebar with sections like Images, 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, and Trust Stores. The main area is titled "Load balancers" and contains a sub-section "Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic." It features a search bar "Filter load balancers" and a table header with columns: Name, State, Type, Scheme, IP address type, VPC ID, and Availability Zone. Below the table, a message says "0 load balancers selected" and "Select a load balancer above." There's also a small illustration of a robot standing next to a cloud.

- Select Application Load Balancer & Click on Create

This screenshot shows the "Compare and select load balancer type" page. It displays three options: Application Load Balancer, Network Load Balancer, and Gateway Load Balancer. Each option has a diagram and a brief description:

- Application Load Balancer Info:** Shows a client connecting to an ALB (Application Load Balancer) which then routes traffic to three targets (Lambda, Container, and EC2). It supports HTTP and HTTPS.
- Network Load Balancer Info:** Shows a client connecting to an NLB (Network Load Balancer) which then routes traffic via TCP, UDP, or TLS to three targets (ALB, Lambda, and Container). It supports VPC and VPCe.
- Gateway Load Balancer Info:** Shows a client connecting to a GWLB (Gateway Load Balancer) which then routes traffic to various third-party virtual appliances (represented by icons like a lock, file, and network card).

Each section includes a "Create" button at the bottom.

- Give Load balancer name Like ALB

This screenshot shows the "Create Application Load Balancer" page. It includes the following configuration fields:

- Load balancer name:** A text input field containing "ALB". A note says "Name must be unique within your AWS account and can't be changed after the load balancer is created."
- Scheme:** A dropdown menu where "Internet-facing" is selected. A note says "Scheme can't be changed after the load balancer is created." There are two options:
  - Internet-facing:** Described as serving internet-facing traffic, having public IP addresses, and requiring a public subnet.
  - Internal:** Described as serving internal traffic, having private IP addresses, and being compatible with IPv4 and Dualstack IP address types.
- Load balancer IP address type:** A dropdown menu where "IPv4" is selected. A note says "Includes only IPv4 addresses."

At the bottom, there's a note: "Select the front-end IP address type to assign to the load balancer. The VPC and subnets mapped to this load balancer must include the selected IP address types. Public IPv4 addresses have an additional cost."

- Select all AZ (choose AZ same as the AZ we selected while creating autoscalling group.)

**Availability Zones and subnets** | [Info](#)

Select at least two Availability Zones and a subnet for each zone. A load balancer node will be placed in each selected zone and will automatically scale in response to traffic. The load balancer routes traffic to targets in the selected Availability Zones only.

**us-east-1a (use1-az2)**

**Subnet**  
Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-022c5b89cb339215  
IPv4 subnet CIDR: 172.31.80.0/20

**us-east-1b (use1-az4)**

**Subnet**  
Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-042354815a750c8d0  
IPv4 subnet CIDR: 172.31.16.0/20

**us-east-1c (use1-az6)**

**Subnet**  
Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-05cd0e8e33324b316  
IPv4 subnet CIDR: 172.31.32.0/20

**us-east-1d (use1-az1)**

**Subnet**  
Only CIDR blocks corresponding to the load balancer IP address type are used. At least 8 available IP addresses are required for your load balancer to scale efficiently.

subnet-0a7312ab67e516365  
IPv4 subnet CIDR: 172.31.0.0/20

**us-east-1e (use1-az3)**

**Subnet**

- select a existing security group. (Select same security group we selected while creating autoscalling group)

**Security groups** | [Info](#)

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](#).

**Security groups**

Select up to 5 security groups

launch-wizard-1  
sg-018576c5e769beb0c VPC: vpc-070d03d6e19cfe3b8

launch-wizard-9  
sg-078882a10d21d386e VPC: vpc-070d03d6e19cfe3b8

default  
sg-095912d128d47e2c1 VPC: vpc-070d03d6e19cfe3b8

**Listener HTTP:80**

**Protocol**: HTTP | **Port**: 80 | **Range**: 1-65535 | [Remove](#)

**Default action** | [Info](#)

The default action is used if no other rules apply. Choose the default action for traffic on this listener.

**Routing action**

- In Listners and Routing section,
  - Select forward to target group
  - Choose a default target group. (Here default target group is home-TG)

**Default action** | [Info](#)  
The default action is used if no other rules apply. Choose the default action for traffic on this listener.

**Routing action**

- Forward to target groups
- Redirect to URL
- Return fixed response

**Forward to target group** | [Info](#)  
Choose a target group and specify routing weight or [create target group](#).

Target group	Weight	Percent
Select a target group	1	100%
Q	0-999	
home-TG Target type: Instance, IPv4   Target stickiness: Off	1	100%
laptop-TG Target type: Instance, IPv4   Target stickiness: Off		
mobile-TG Target type: Instance, IPv4   Target stickiness: Off		

Target must support cookies. If you want to bind a user's session to a specific target, turn on the Target Group attribute Stickiness.

**Listener tags - optional**  
Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

[Add listener tag](#)

- Click on Create Load Balancer

**Service integrations** [Edit](#)  
Amazon CloudFront + AWS Web Application Firewall (WAF): -  
AWS WAF: -  
AWS Global Accelerator: -

**Attributes**

Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.

**Creation workflow and status**

► **Server-side tasks and status**  
After completing and submitting the above steps, all server-side tasks and their statuses become available for monitoring.

[Cancel](#) [Create load balancer](#)

- Successfully created Application loadbalancer
- If you scroll down , you will see the HomeTG as defualt target group in Listners and Rules section.
- To add other Target groups..

\* Select HTTP80 i.e., home-TG

\* Click on Manage Rules

\* Click on Add Rule

The screenshot shows the AWS CloudFormation console with the following details:

- Stack Name:** MobileAppStack
- Outputs:**
  - MobileAppFunctionArn:** arn:aws:lambda:us-east-1:824901035886:function:MobileAppFunction
  - Description:** Lambda function for mobile application
- Resources:**
  - MobileAppFunction:** AWS::Lambda::Function
  - MobileAppLambdaLayer:** AWS::Lambda::LayerVersion
  - MobileAppLambdaLayerArn:** arn:aws:lambda:us-east-1:824901035886:layer:MobileAppLambdaLayer:1

- Give name to the rule and click on add condition and select path.

The screenshot shows the AWS CloudFormation console with the following details:

- Stack Name:** MobileAppStack
- Outputs:**
  - MobileAppFunctionArn:** arn:aws:lambda:us-east-1:824901035886:function:MobileAppFunction
  - Description:** Lambda function for mobile application
- Resources:**
  - MobileAppFunction:** AWS::Lambda::Function
  - MobileAppLambdaLayer:** AWS::Lambda::LayerVersion
  - MobileAppLambdaLayerArn:** arn:aws:lambda:us-east-1:824901035886:layer:MobileAppLambdaLayer:1

- Give path for mobile application

**Path = /mobile/\***

**Path condition value**  
Case sensitive.  
= /mobile/\*  
Valid characters are a-z, A-Z, 0-9 and special characters. Path must be 1-128 characters.

**Add OR condition value**

**Add condition ▾**  
You can add up to 4 more condition values for this rule.

**Actions Info**  
Requests matching all rule conditions route according to the rule actions.

**Routing action**

- Forward to target groups
- Redirect to URL
- Return fixed response

**Forward to target group | Info**  
Choose a target group and specify routing weight or [create target group](#).

Target group	Weight	Percent
mobile-TG	1	100%

- In Actions section

- select forward to target group
- select the target group i.e., mobile-TG
- Go to next step.

You can add up to 4 more condition values for this rule.

**Actions Info**  
Requests matching all rule conditions route according to the rule actions.

**Routing action**

- Forward to target groups
- Redirect to URL
- Return fixed response

**Forward to target group | Info**  
Choose a target group and specify routing weight or [create target group](#).

Target group	Weight	Percent
Select a target group	1	100%
mobile-TG	1	100%

0-999

stickiness the client must support cookies. If you want to bind a user's session to a specific target, turn on the Target

**Cancel** **Next**

- In Listners rule , give priority 1 for mobileTG
- Click on next

**Step 1 Add rule**

**Step 2 Set rule priority** **Selected**

**Step 3 Review and create**

**Set rule priority** Info

Each rule has a priority. The default rule is evaluated last. You can change the priority of a non-default rule at any time. You can't change the priority of the default rule.

**Listener details: HTTP:80**

**Listener rules (2)** Info

Traffic received by the listener is routed according to the default action and any additional rules. Rules are evaluated in priority order from the lowest value to the highest value.

Priority	Name tag	Conditions (If)	Actions (Then)	ARN
1	mobile-rule	Path = /mobile/*	<ul style="list-style-type: none"> <li>Forward to target group mobile-TG [2]: 1 (100%)</li> </ul> <small>Target group stickiness: Off</small>	Pending
Last (default)	Default	If no other rule applies	<ul style="list-style-type: none"> <li>Forward to target group home-TG [2]: 1 (100%)</li> </ul> <small>Target group stickiness: Off</small>	ARN

**Cancel** **Previous** **Next**

- Click on add rule

**Step 1 Set rule priority**

**Step 2 Review and create** **Selected**

**Listener details: HTTP:80**

**Rule details: mobile-rule**

Priority 1	Conditions If request matches all: Path = /mobile/*	Actions <ul style="list-style-type: none"> <li>Forward to target group mobile-TG [2]: 1 (100%)</li> </ul> <small>Target group stickiness: Off</small>
---------------	---	--

**Rule ARN**  
Pending

**Rule tags (1)** Edit

Tags can help you manage, identify, organize, search for and filter resources.

Key	Value
Name	mobile-rule

**Server-side tasks and status**

After completing and submitting the above steps, all server-side tasks and their statuses become available for monitoring.

**Cancel** **Previous** **Add rule**

- Successfully connected Mobile Target Group to Loadbalancer
- Similarly connect Laptop target group to load balancer with priority 2

The screenshot shows the AWS Lambda console interface for creating a new rule. The path is EC2 > Load balancers > ALB > HTTP:80 listener > Add rule. The first step, "Review and create", is selected. The "Name and tags" section has "laptop-rule" entered. The "Conditions" section contains one condition: "Path = /laptop/\*". The "Actions" section shows "Forward to target groups" selected, with a target group named "laptop-TG" assigned. The "Listener rules" table at the bottom lists three rules: "mobile-rule" (Priority 1), "laptop-rule" (Priority 2), and "Last (default)" (Priority 100). The "Next Step" button is visible.

This screenshot continues the rule creation process. The "Actions" section is shown again, with "Forward to target groups" selected. The "Target group" dropdown is set to "laptop-TG". The "Weight" field is set to 1, and the "Percent" field is set to 100%. The "Next Step" button is visible.

The screenshot shows the "Set rule priority" step. The "mobile-rule" is listed with Priority 1 and condition "Path = /mobile/\*". The "laptop-rule" is listed with Priority 2 and condition "Path = /laptop/\*". The "Last (default)" rule is listed with Priority 100 and condition "If no other rule applies". The "Next Step" button is visible.

- You can now see that the active instances have been automatically created in the instance dashboard

**Instances (8) Info**

Last updated less than a minute ago

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
i-0b0aecc3b69303809	i-0b0aecc3b69303809	Terminated	t3.micro	-	<a href="#">View alarms +</a>	us-east-1a	-
i-0ab67e346fe7052d2	i-0ab67e346fe7052d2	Running	t3.micro	3/3 checks passed	<a href="#">View alarms +</a>	us-east-1b	ec2-18-2
i-058b6f0d6651fa80f	i-058b6f0d6651fa80f	Running	t3.micro	3/3 checks passed	<a href="#">View alarms +</a>	us-east-1b	ec2-54-2
i-085b677b79d0380c8	i-085b677b79d0380c8	Running	t3.micro	3/3 checks passed	<a href="#">View alarms +</a>	us-east-1c	ec2-54-8
i-042012ad73bf79b1	i-042012ad73bf79b1	Terminated	t3.micro	-	<a href="#">View alarms +</a>	us-east-1f	-
i-0a77dfbc274a645	i-0a77dfbc274a645	Running	t3.micro	3/3 checks passed	<a href="#">View alarms +</a>	us-east-1c	ec2-23-2
i-0b5ef4313da10135	i-0b5ef4313da10135	Running	t3.micro	3/3 checks passed	<a href="#">View alarms +</a>	us-east-1d	ec2-3-23
i-05125e57840d673ff	i-05125e57840d673ff	Running	t3.micro	3/3 checks passed	<a href="#">View alarms +</a>	us-east-1b	ec2-184-

Select an instance

- To verify if it's functioning correctly:
  - Navigate to the Load Balancers section.
  - Select the load balancer you just created.
  - Scroll down to find and copy the DNS name.
  - Paste the DNS name into your web browser's address bar and press Enter

**Load balancers (1/1)**

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones
ALB	Active	application	Internet-facing	IPv4	vpc-070d03d6e19cf3b8	6 Availability Zones

**Load balancer: ALB**

east-1a (use1-az2)  
subnet-0aa92490e2e4cb085 us-east-1e (use1-az3)  
subnet-0ef33c861cf14bcae us-east-1f (use1-az5)

**Load balancer ARN**  
arn:aws:elasticloadbalancing:us-east-1:824901035886:loadbalancer/app/ALB/c6cf946992bf78e8

**DNS name**  
ALB-849233605.us-east-1.elb.amazonaws.com (A Record)

## Step 7: Result

- ALB DNS successfully routed traffic to all three target groups.

Output of home:

This is a home page ip-172-31-177.ec2.internal

This is a home page ip-172-31-42-38.ec2.internal

Output of laptop:



Output of mobile:



## Technologies Used

1. Amazon EC2 – Virtual machines used to run the application.
2. Application Load Balancer (ALB) – Distributes incoming traffic to different servers.
3. Launch Templates – Predefined settings used to quickly launch EC2 instances.
4. Auto Scaling Groups (ASG) – Automatically adds or removes instances based on demand (supports fixed, automatic, or scheduled scaling).
5. Target Groups – Directs traffic to the correct group of servers.
6. Amazon Linux – The operating system installed on the EC2 instances.
7. Security Groups – Act like a firewall to control incoming and outgoing traffic to the servers.

## Conclusion

This project showcases how integrating an Application Load Balancer (ALB) with multiple Auto Scaling Groups (ASGs) can effectively deliver high availability, fault tolerance, and scalability within the AWS environment.

By working with Static, Scheduled, and Dynamic scaling methods, I gained practical experience in configuring and managing various scaling techniques. The setup efficiently distributed incoming traffic across target groups, automatically adjusted the number of instances based on demand, and served application requests through a unified DNS endpoint.

In summary, this project strengthened my understanding of AWS load balancing, automated scaling, and cloud infrastructure management.