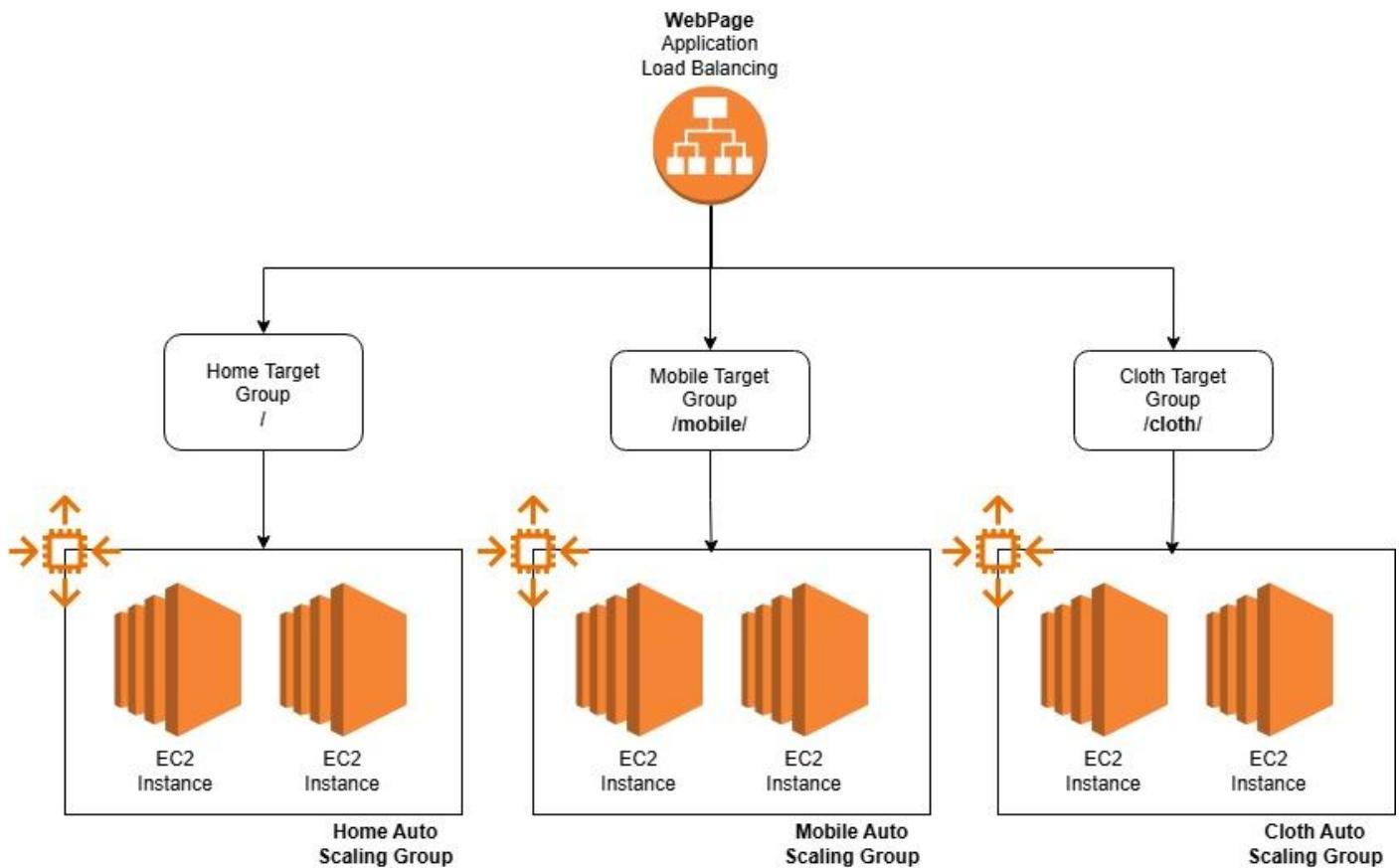


Path-Based Web Application on AWS using EC2 Auto Scaling and Application Load Balancer

By: Nikhilesh Sakhare

Overall layout

The entry point is an internet-facing Application Load Balancer (ALB) that exposes a single HTTP endpoint to users. Behind the ALB, there are three independent Auto Scaling Groups, each running a fleet of Amazon EC2 instances that serve a specific part of the website: home, mobile, and cloth. All resources are deployed in the same VPC across multiple subnets and Availability Zones to improve availability and fault tolerance.



Auto Scaling Groups and instances

Each Auto Scaling Group (home-as, mobile-as, cloth-as) is created from its own EC2 launch template, which defines the Amazon Linux AMI, instance type (t3.micro), security group, and user-data script. The user-data on each template installs Apache and writes a different HTML page so that the response clearly identifies the section being served (home, mobile, or cloth). The groups are configured with minimum, desired, and maximum capacities and can use target-tracking scaling based on average CPU utilization to automatically adjust the number of running instances under load.

Load balancer, target groups, and routing

The ALB is attached to the same VPC and spread across three public subnets, and it uses a shared security group (MyWebSG) that allows HTTP traffic on port 80. Three HTTP instance-target groups (home-tg, mobile-tg, cloth-tg) register the EC2 instances from their respective Auto Scaling Groups and handle health checks. The ALB's HTTP:80 listener uses a default rule that forwards root path '/' traffic to home-tg, and additional path-based rules that forward '/mobile/*' to mobile-tg and '/cloth/*' to cloth-tg. This allows a single DNS name from the load balancer to serve all three logical applications while keeping their compute layers isolated and independently scalable.

- Logged in to the AWS Management Console and verified the working region as Asia Pacific (Mumbai) from the console home page.

The screenshot shows the AWS Management Console Home page. At the top right, it displays "Account ID: 6735-8684-0000-0000" and "Asia Pacific (Mumbai)". The left sidebar has a "Recently visited" section with links to EC2, VPC, Billing and Cost Management, Aurora and RDS, CloudWatch, S3, IAM, and Simple Notification Service. Below this is a "View all services" link. On the right, there's a "Applications" section with a "Create application" button, a search bar for "Find applications", and a table header for "Name", "Description", "Region", and "Originati.". A message below says "No applications" and "Get started by creating an application." with a "Create application" button. At the bottom right is a "Go to myApplications" link.

- Opened the EC2 service from the search bar and reviewed the EC2 dashboard showing zero running instances and no existing load balancers or Auto Scaling Groups.

The screenshot shows the AWS Management Console EC2 service page. The search bar at the top contains "ec2". The left sidebar has a "Services" section with links to Features, Resources, Documentation, Knowledge articles, Marketplace, Blog posts, Events, and Tutorials. Below this is a "Were these results helpful?" section with "Yes" and "No" buttons. The main content area has a "Services" section with a "EC2" card (Virtual Servers in the Cloud) and a "Top features" section with links to Dashboard, Launch templates, Instances, Spot Instance requests, and Savings plans. It also has a "Features" section with "EC2 Instances" and "EC2 Resource Health" cards, both of which mention CloudWatch features. On the right side, there's a "Create application" button, a search bar for "Find applications", and a table header for "Name", "Description", "Region", and "Originati.". A message below says "No applications" and "Get started by creating an application." with a "Create application" button. At the bottom right is a "Go to myApplications" link.

3. From the EC2 left navigation pane, selected “Launch Templates” to start defining reusable instance configurations.

The screenshot shows the AWS EC2 console homepage. The left sidebar is titled 'EC2' and includes sections for Dashboard, Instances, and Images. The 'Instances' section is expanded, showing sub-options like Instances, Instance Types, Launch Templates, and Spot Requests. The main content area is titled 'Resources' and displays various Amazon EC2 resources in the Asia Pacific (Mumbai) Region. It includes tables for Instances (running), Auto Scaling Groups, Capacity Reservations, Dedicated Hosts, Elastic IPs, Instances, Key pairs, Load balancers, Placement groups, Security groups, Snapshots, and Volumes. To the right, there's a 'EC2 cost' summary with a date range of 'Past 6 months', credits remaining of '\$139.3 USD', and days remaining of '156 (May 11, 2026)'. Below that is a 'Cost (\$)' section with a graph and a link to 'AWS Health Dashboard'.

4. Clicked “Create launch template” and entered the template name homepage with an appropriate description, enabling Auto Scaling guidance.

The screenshot shows the 'EC2 launch templates' page. The left sidebar is identical to the previous screenshot. The main content area has a dark header 'Compute' and a large title 'EC2 launch templates: Streamline, simplify and standardize instance launches'. Below it is a description of how launch templates can automate instance launches and simplify permission policies. A prominent orange 'Create launch template' button is located in the top right. Below the button, there are sections for 'Benefits and features' (Streamline provisioning, Simplify permissions) and 'Documentation'.

The screenshot shows the 'Create launch template' wizard. The top navigation bar shows the user is on the 'Launch templates > Create launch template' page. The main form is titled 'Create launch template name and description'. It includes fields for 'Launch template name - required' (containing 'homepage'), 'Template version description' (containing 'A prod webserver for MyApp'), and 'Auto Scaling guidance' (with a checked checkbox for 'Provide guidance to help me set up a template that I can use with EC2 Auto Scaling'). On the right, the 'Summary' step is displayed, which includes sections for 'Software Image (AMI)', 'Virtual server type (instance type)', 'Firewall (security group)', and 'Storage (volumes)'. At the bottom right of the summary step is a 'Create launch template' button.

5. In the “Application and OS Images” section, selected the Amazon Linux 2023 AMI from the Quick Start list.

The screenshot shows the AWS EC2 console with the 'Launch templates' section open. Under the 'Launch template contents' heading, the 'Application and OS Images (Amazon Machine Image) - required' section is expanded. A search bar at the top of this section contains the placeholder 'Search our full catalog including 1000s of application and OS images'. Below the search bar, there are two tabs: 'Recents' and 'Quick Start', with 'Quick Start' being the active tab. A grid of recent AMIs is displayed, including Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. To the right of the grid, a button labeled 'Browse more AMIs' is visible, with the subtext 'Including AMIs from AWS, Marketplace and the Community'. At the bottom of this section, a specific AMI entry for 'Amazon Linux 2023 kernel-6.1 AMI' is shown, detailing its ID, creation date, and compatibility with Free tier eligible. On the far right of the interface, a 'Summary' panel is partially visible, containing sections for Software Image (AMI), Virtual server type (instance type), Firewall (security group), and Storage (volumes). A large orange 'Create launch template' button is located at the bottom right of the main content area.

6. Chose instance type t3.micro, which is free-tier eligible, for all web servers.

The screenshot shows the continuation of the AWS EC2 'Launch templates' configuration. The 'Description' section is present, followed by the 'Architecture' and 'AMI ID' details. The 'Instance type' section is expanded, showing the 't3.micro' option selected. The 'Free tier eligible' status is indicated. Below this, a detailed breakdown of the instance type's specifications and costs is provided, including family, vCPUs, memory, and various pricing options for On-Demand and On-Demand Windows usage. An 'Advanced' link is visible next to the instance type section. The 'Summary' panel on the right side of the screen remains largely the same, listing the chosen Software Image (Amazon Linux 2023 kernel-6.1 AMI), Virtual server type (t3.micro), and Storage (1 volume(s) - 8 GiB). The 'Create launch template' button is again prominent at the bottom right.

7. In the key pair section kept the default selection as per the account, and then moved to the security group configuration where “Create security group” was selected. Created a new security group named MyWebSG, added a description, associated it with the default VPC, and configured two inbound rules: SSH (port 22) from Anywhere and HTTP (port 80) from Anywhere so that the web pages are reachable over the internet.

The screenshot shows the AWS EC2 console under the 'Launch templates' section. On the left, there's a configuration pane for a security group. It has a 'Firewall (security groups)' section with a 'Create security group' button. Below it are fields for 'Security group name - required' (MyWebSG), 'Description - required' (Allows SSH access to developers), and 'VPC' (selected). Under 'Inbound Security Group Rules', there are two entries: 'Security group rule 1 (TCP, 22, 0.0.0.0/0)' and 'Security group rule 2 (TCP, 80, 0.0.0.0/0)'. Both rules have 'ssh' as the type, 'TCP' as the protocol, and specific port ranges (22 and 80 respectively). The source is set to 'Anywhere' for both. A note at the bottom of this section says: '⚠️ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to...' On the right, there are two summary panes. The top one is for the security group, showing the AMI (Amazon Linux 2023.9.2), instance type (t3.micro), and storage (1 volume(s) - 8 GiB). The bottom summary pane is for the launch template itself, showing the same AMI, instance type, and storage. Both panes have 'Cancel' and 'Create launch template' buttons.

8. Scrolled to the “User data” section and pasted a shell script to install and start Apache (httpd) and to create /var/www/html/index.html with a simple “This is home page: \$HOSTNAME” message.

The screenshot shows the 'User data - optional' section of the launch template creation page. A shell script is pasted into the text area:

```
#!/bin/bash
yum install httpd -y
systemctl start httpd
systemctl enable httpd
echo "<h1>This is home page: $HOSTNAME</h1>" > /var/www/html/index.html
```

Below the text area is a checkbox: 'User data has already been base64 encoded'. The right side of the screen shows the summary of the launch template, identical to the one in the previous screenshot.

9. Reviewed the summary on the right-hand side and clicked “Create launch template”, confirming that the homepage launch template and associated security group were created successfully.

The screenshot shows the AWS EC2 Launch Templates interface. At the top, there's a green success message box stating "Successfully created homepage(lt-0e29d745b71245a8d)." Below it, an "Actions log" table shows four successful steps: "Initializing requests", "Creating security groups", "Creating security group rules", and "Create Launch Template". Under "Next Steps", there are links for "Launch an instance", "Launch instance from this template", and "Create an Auto Scaling group from your template". The "Launch an instance" section includes a note about On-Demand Instances and a link to "Launch an On-Demand Instance from your launch template". The "Create an Auto Scaling group from your template" section includes a note about Auto Scaling and a link to "Create an Auto Scaling group from this template".

10. Repeated the “Create launch template” workflow to create a second launch template named mobilepage. Reused the same AMI, instance type, and security group MyWebSG but changed the user data script to create /var/www/html/mobile/index.html with “This is mobile page: \$HOSTNAME”, creating the /mobile directory first.

This screenshot shows the "Create launch template" page for the "mobilepage" launch template. On the left, the "User data - optional" section contains a user data script in a code editor:

```
#!/bin/bash
yum install httpd -y
systemctl start httpd
systemctl enable httpd
mkdir /var/www/html/mobile
echo "<h1>This is mobile page: $HOSTNAME</h1>" > /var/www/html/mobile/index.html
```

A checkbox below the editor says "User data has already been base64 encoded". On the right, the "Summary" section displays the configuration details:

- Software Image (AMI)**: Amazon Linux 2023.9.2... [read more](#)
- Virtual server type (instance type)**: t3.micro
- Firewall (security group)**: MyWebSG
- Storage (volumes)**: 1 volume(s) - 8 GiB

At the bottom right is a "Create launch template" button.

11. Performed the same steps again to create a third launch template named clothpage, with user data that installs Apache and creates /var/www/html/cloth/index.html to serve the “cloth” page.

After all three templates were created, the “Launch Templates” page showed homepage, mobilepage, and clothpage as separate launch templates ready to be used by Auto Scaling Groups.

The screenshot shows the AWS EC2 Launch Templates page. The left sidebar is titled 'EC2' and includes sections for Dashboard, AWS Global View, Events, Instances (selected), Launch Templates (selected), Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Capacity Manager. The main content area is titled 'Launch Templates (3) Info' and contains a table with three rows:

Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By
lt-01896de11dc151ee2	clothpage	1	1	2025-12-06T04:39:59.000Z	arn:aws:iam::6735
lt-0e29d745b71245a8d	homepage	1	1	2025-12-06T04:27:47.000Z	arn:aws:iam::6735
lt-02233c74e7d29a91f	mobilepage	1	1	2025-12-06T04:37:53.000Z	arn:aws:iam::6735

Below the table, there is a section titled 'Select a launch template' with a dropdown menu. The bottom of the page includes standard AWS navigation links like CloudShell, Feedback, and Console Mobile App, along with copyright information and cookie preferences.

12. From the EC2 navigation pane, opened “Auto Scaling groups” and clicked “Create Auto Scaling group”.

The screenshot shows the AWS EC2 Auto Scaling Groups page. The left sidebar is titled 'EC2' and includes sections for Snapshots, Lifecycle Manager, 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 selected). The main content area features a large banner with the heading 'Amazon EC2 Auto Scaling' and subtext 'helps maintain the availability of your applications'. Below the banner, a paragraph explains that Auto Scaling groups are collections of Amazon EC2 instances that enable automatic scaling and fleet management features. To the right of the banner is a callout box titled 'Create Auto Scaling group' with the subtext 'Get started with EC2 Auto Scaling by creating an Auto Scaling group.' and a prominent orange 'Create Auto Scaling group' button. At the bottom, there are two more callout boxes: 'How it works' (with a diagram showing an 'Auto Scaling group' box with arrows pointing in and out) and 'Pricing' (with text explaining that features have no additional fees beyond service fees for EC2, CloudWatch, and other AWS resources). The bottom of the page includes standard AWS navigation links and copyright information.

13. Provided the Auto Scaling group name home-as and selected the homepage launch template that was created earlier.

The screenshot shows the AWS Auto Scaling wizard at Step 1: Choose launch template. On the left, a vertical navigation bar lists steps from 1 to 7. Step 1 is highlighted. The main area is titled 'Choose launch template' with a 'Info' link. It says: 'Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group.' A 'Name' field is filled with 'home-as'. Below it, a note states: 'Must be unique to this account in the current Region and no more than 255 characters.' A 'Launch template' section follows, with a note: '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.' A dropdown menu shows 'homepage' selected. At the bottom, there's a 'Create a launch template' link.

14. On the “Choose instance launch options” step, selected the default VPC and enabled all three available subnets in different Availability Zones (ap-south-1a, ap-south-1b, ap-south-1c) to achieve high availability for the home servers.

The screenshot shows the AWS Auto Scaling wizard at Step 2: Choose instance launch options. The left sidebar shows steps 2 through 7. Step 2 is highlighted. The main area has a 'VPC' section where 'vpc-0f9cea248655056b6' is selected. An 'Availability Zones and subnets' section lists three subnets: 'aps1-az1 (ap-south-1a) | subnet-064a0ebe46a043a9c', 'aps1-az2 (ap-south-1c) | subnet-0b711be386d0c631e', and 'aps1-az3 (ap-south-1b) | subnet-02315315694c87bf0'. Each subnet entry includes its IP range (172.31.3.0/20, 172.31.16.0/20, 172.31.0.0/20) and a 'Default' label. A 'Create a subnet' link is present. Below these, an 'Availability Zone distribution - new' section allows selecting a strategy: 'Balanced best effort' (selected) or 'Balanced only'. The note under 'Balanced best effort' says: 'If launches fail in one Availability Zone, Auto Scaling will attempt to launch in another healthy Availability Zone.' The note under 'Balanced only' says: 'If launches fail in one Availability Zone, Auto Scaling will continue to attempt to launch in the unhealthy Availability Zone to preserve balanced distribution.'

On the same wizard, ensured that the instances launched by this group would inherit the security group MyWebSG from the launch template.

15. On the “Configure group size and scaling” step, set the desired capacity to 2, the minimum capacity to 1, and the maximum capacity to 3 so that the group can scale in and out within that range.

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

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.

Desired capacity
Specify your group size.
Units (number of instances) ▾
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.

Min desired capacity 1	Max desired capacity 3
Equal or less than desired capacity	
Equal or greater than desired capacity	

16. Enabled a target tracking scaling policy based on Average CPU utilization with a target value of 50 percent and a default instance warm-up time of 60 seconds, allowing the group to add or remove instances automatically based on CPU load.

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.

<input type="radio"/> No scaling policies Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.	<input checked="" type="radio"/> 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.
--	--

Scaling policy name
Target Tracking Policy

Metric type | Info
Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

Average CPU utilization

Target value
50

Instance warmup | Info
60 seconds

Disable scale in to create only a scale-out policy

17. Reviewed the configuration summary, confirming the launch template, network, subnets, and scaling settings, then created the home-as Auto Scaling Group.

Step 4 - optional
Configure group size and scaling

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Step 7
Review

Launch template
Launch template
homepage [lt-0e29d745b71245a8d](#)

Step 2: Choose instance launch options

Network
vpc
[vpc-0f9cea248655056b6](#)

Availability Zones and subnets

Availability Zone	Subnet	Subnet CIDR range
aps1-az1 (ap-south-1a)	subnet-064a0eb46a043a9c	172.31.32.0/20
aps1-az2 (ap-south-1c)	subnet-0b711be386d0c631e	172.31.16.0/20
aps1-az3 (ap-south-1b)	subnet-02315315694c87bf0	172.31.0.0/20

[Edit](#)

18. Repeated the Auto Scaling Group creation process to create mobile-as using the mobilepage launch template and cloth-as using the clothpage launch template, each with the same network, subnet, and scaling configuration.

Returned to the “Auto Scaling groups” page and verified that three groups (home-as, mobile-as, cloth-as) were present, each with a desired capacity of 2 instances.

Auto Scaling groups (3) [Info](#)

Last updated less than a minute ago [C](#) [Launch configurations](#) [Launch templates](#) [Actions](#) [Create Auto Scaling group](#)

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
cloth-as	clothpage Version Default	2	-	2	1	3
mobile-as	mobilepage Version Default	2	-	2	1	3
home-as	homepage Version Default	2	-	2	1	3

0 Auto Scaling groups selected

19. From the EC2 navigation pane under “Load Balancing”, selected “Target Groups” and confirmed that there were initially no target groups.

The screenshot shows the AWS EC2 console with the navigation pane open. Under the 'Load Balancing' section, 'Target Groups' is selected. The main area is titled 'Target groups' and displays a message: 'No target groups'. Below it, a button says 'Create target group'. On the left, a sidebar lists various EC2 features: Snapshots, Lifecycle Manager, 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, Settings). At the bottom, there are links for CloudShell, Feedback, and Console Mobile App, along with standard AWS footer links for Privacy, Terms, and Cookie preferences.

20. Clicked “Create target group”, kept the target type as “Instances”, and entered the first target group name home-tg.

The screenshot shows the 'Create target group' wizard at Step 2. The title is 'Create target group'. It says: 'A target group can be made up of one or more targets. Your load balancer routes requests to the targets in a target group and performs health checks on the targets.' The 'Settings - immutable' section is shown, with the 'Instances' option selected. Other options include 'IP addresses', 'Lambda function', and 'Application Load Balancer'. The 'Target group name' field is set to 'home-tg'. The wizard has three steps: Step 1 (Create target group), Step 2 - recommended (selected), Step 3 (Review and create). At the bottom, there are links for CloudShell, Feedback, and Console Mobile App, along with standard AWS footer links for Privacy, Terms, and Cookie preferences.

21. Completed the remaining settings (protocol HTTP, port 80, and default health check path) and moved to the “Register targets” step, where running EC2 instances created by the Auto Scaling Groups were visible.

Step 1
Create target group
Step 2 - recommended
Register targets
Step 3
Review and create

Register targets - recommended
This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (6)

<input type="checkbox"/>	Instance ID	Name	State	Security groups	Zone
<input type="checkbox"/>	i-05f40fd9a315d54d5		Running	MyWebSG	ap-south-1a
<input type="checkbox"/>	i-06804c1473df9b028		Running	MyWebSG	ap-south-1c
<input type="checkbox"/>	i-003adb6077bcec7c6		Running	MyWebSG	ap-south-1b
<input type="checkbox"/>	i-033e0679f05d43d8d		Running	MyWebSG	ap-south-1c
<input type="checkbox"/>	i-0e08fbbaa34806f194		Running	MyWebSG	ap-south-1a
<input type="checkbox"/>	i-0d70cce6ab84a2991		Running	MyWebSG	ap-south-1c

22. Don't select any instances for the home page target group now and then created additional target groups mobile-tg and cloth-tg by repeating the same steps.

After all three target groups were created, the Target Groups page showed home-tg, mobile-tg, and cloth-tg, each configured on HTTP port 80 and currently not yet associated with any load balancer.

Snapshots
Lifecycle Manager

Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces

Load Balancing
Load Balancers
Target Groups
Trust Stores

Auto Scaling
Auto Scaling Groups

Settings

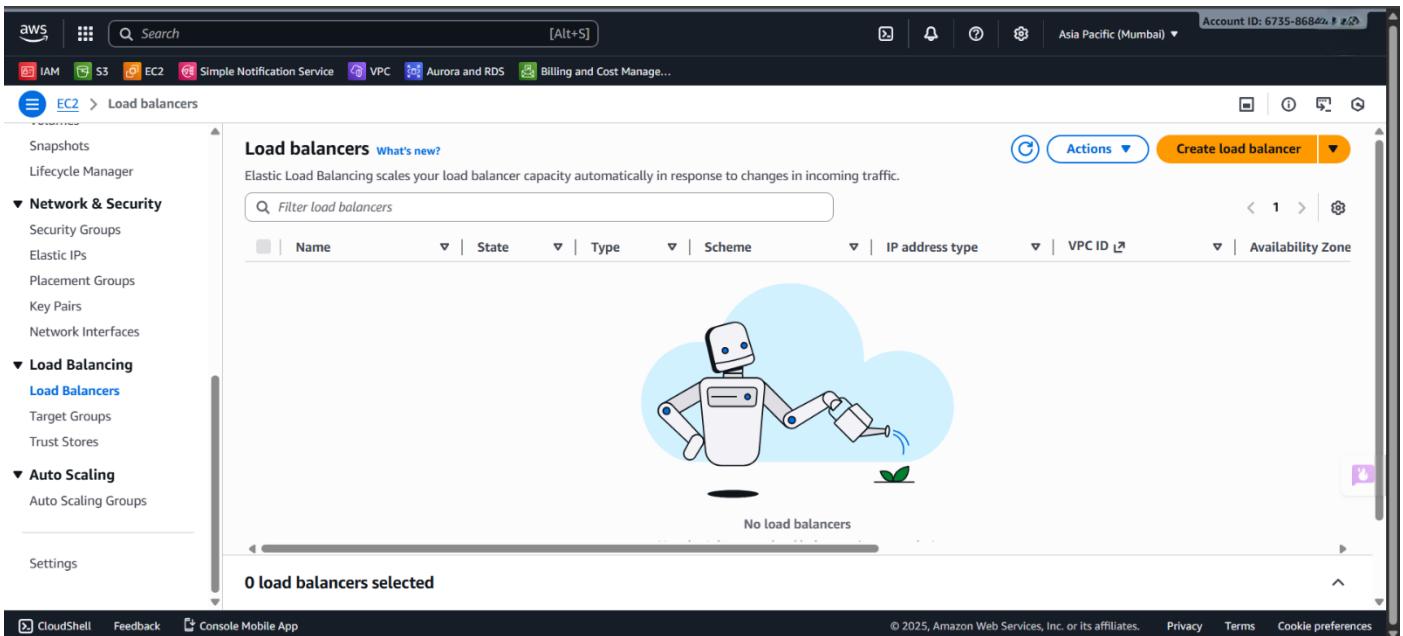
Target groups (3) [Info](#) | [What's new?](#)

Create target group

<input type="checkbox"/>	Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
<input type="checkbox"/>	cloth-tg	arn:aws:elasticloadbalancin...	80	HTTP	Instance	None associated	vpc-0f9c
<input type="checkbox"/>	home-tg	arn:aws:elasticloadbalancin...	80	HTTP	Instance	None associated	vpc-0f9c
<input type="checkbox"/>	mobile-tg	arn:aws:elasticloadbalancin...	80	HTTP	Instance	None associated	vpc-0f9c

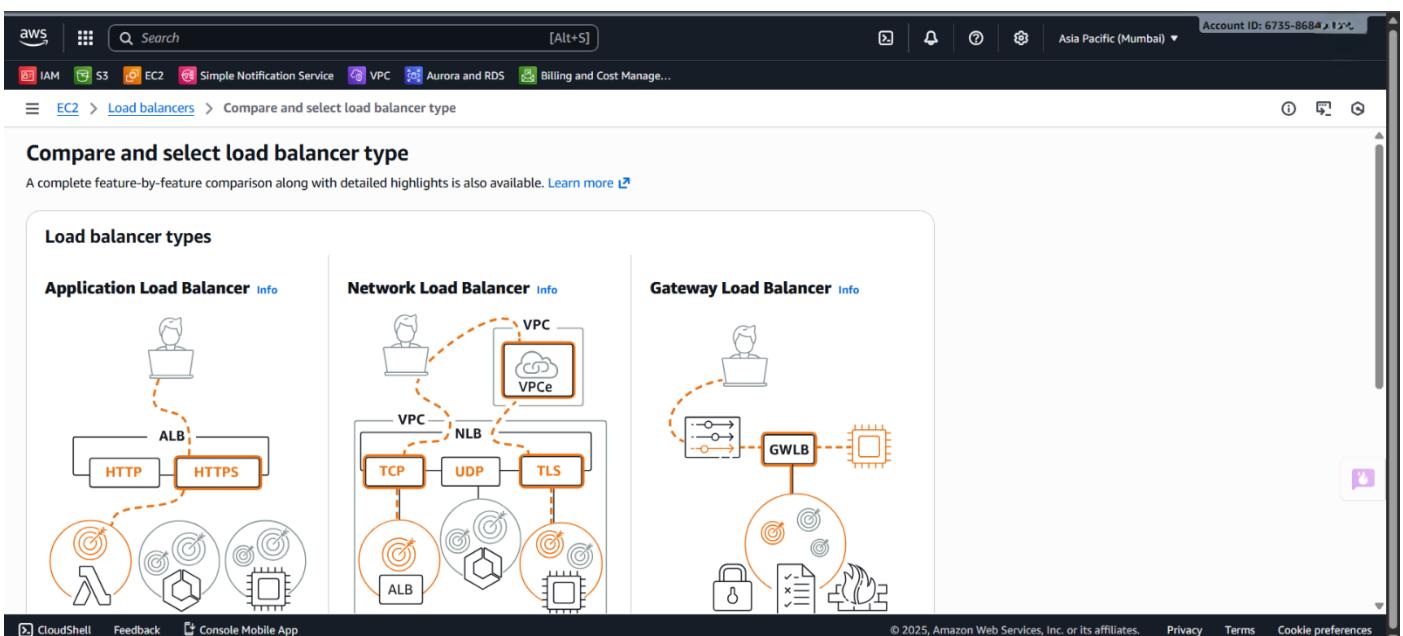
0 target groups selected

23. Navigated to “Load Balancers” under the EC2 “Load Balancing” section and verified that there was no existing load balancer.



The screenshot shows the AWS Cloud Console with the URL [https://console.aws.amazon.com/ec2/v2/home?#LoadBalancers](#). The top navigation bar includes links for IAM, S3, EC2, Simple Notification Service, VPC, Aurora and RDS, and Billing and Cost Management. The account ID is 6735-8684. The main content area is titled "Load balancers" and features a sub-header "Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic." Below this is a search bar labeled "Filter load balancers" and a table header with columns: Name, State, Type, Scheme, IP address type, VPC ID, and Availability Zone. A large, light blue cloud icon with a white robot holding a wrench is centered, accompanied by the text "No load balancers". At the bottom left, it says "0 load balancers selected". The bottom navigation bar includes CloudShell, Feedback, and Console Mobile App.

24. Clicked “Create load balancer” and chose “Application Load Balancer” from the list of load balancer types.



The screenshot shows the AWS Cloud Console with the URL [https://console.aws.amazon.com/ec2/v2/home?#LoadBalancers/Compare](#). The top navigation bar includes links for IAM, S3, EC2, Simple Notification Service, VPC, Aurora and RDS, and Billing and Cost Management. The account ID is 6735-8684. The main content area is titled "Compare and select load balancer type". It features three cards: "Load balancer types", "Application Load Balancer", and "Network Load Balancer". The "Application Load Balancer" card shows a diagram of a client connecting to an ALB (Application Load Balancer) which then routes traffic to three targets (Lambda function, API gateway, and Amazon RDS). The "Network Load Balancer" card shows a client connecting to an NLB (Network Load Balancer) which routes traffic via TCP, UDP, or TLS to three targets (ALB, VPC endpoint, and Lambda function). The "Gateway Load Balancer" card shows a client connecting to a GWLB (Gateway Load Balancer) which routes traffic to three targets (Amazon Kinesis, Amazon CloudWatch Metrics, and Amazon CloudFront). The bottom navigation bar includes CloudShell, Feedback, and Console Mobile App.

25. On the “Create Application Load Balancer” page, entered the load balancer name website-lb and selected the “Internet-facing” scheme so that the ALB can receive requests from the public internet.

The screenshot shows the AWS CloudFront Create Application Load Balancer page. In the 'Basic configuration' section, the 'Load balancer name' is set to 'website-lb'. Under 'Scheme', the 'Internet-facing' option is selected, which includes points such as serving internet-facing traffic and having public IP addresses. The 'Internal' scheme is also listed. In the 'Load balancer IP address type' section, 'IPv4' is selected. The bottom of the page shows standard AWS navigation links like CloudShell, Feedback, and Console Mobile App.

Selected IPv4 address type and kept the default VPC.

26. In the “Availability Zones and subnets” section, selected three public subnets across Availability Zones ap-south-1a, ap-south-1b, and ap-south-1c so that the ALB is distributed across all zones used by the Auto Scaling Groups.

The screenshot shows the 'Availability Zones and subnets' section of the AWS CloudFront Create Application Load Balancer page. Three subnets are selected: 'subnet-064a0beb46a043a9c' (IPv4 subnet CIDR: 172.31.32.0/20), 'subnet-02315315694c87bf0' (IPv4 subnet CIDR: 172.31.0.0/20), and 'subnet-0b711be386d0c631e' (IPv4 subnet CIDR: 172.31.16.0/20). The 'Security groups' section is visible at the bottom, showing a note about selecting an existing security group or creating a new one.

27. Under “Security groups”, attached the existing security group MyWebSG to the load balancer, allowing HTTP access to the ALB.

The screenshot shows the AWS CloudFormation interface for creating an Application Load Balancer. In the 'VPC' section, a VPC named 'MyWebVPC' is selected. In the 'Security groups' section, a security group named 'MyWebSG' is attached. The 'Listeners and routing' section is expanded, showing a default listener for port 80 configured to forward traffic to the 'home-tg' target group.

28. In the “Listeners and routing” section, configured the default HTTP listener on port 80 and set the default action to forward traffic to the home-tg target group so that root path / requests go to the home page.

The screenshot shows the AWS CloudFormation interface for creating an Application Load Balancer. In the 'Listeners and routing' section, a default listener for port 80 is configured to forward traffic to the 'home-tg' target group. The 'Default action' is set to 'Forward to target groups'. The 'Target group' is 'home-tg' with a weight of 1 and a percent of 100%. The 'Target group stickiness' option is turned off.

Reviewed all settings and created the Application Load Balancer website-lb, waiting until its state became “Active” and noting the DNS name shown on the Load Balancers page.

29. Returned to the “Auto Scaling groups” page, selected home-as, chose “Edit” from the Actions menu, and, in the “Load balancing – optional” section, attached the home-tg target group to this Auto Scaling Group.

The screenshot shows the AWS Auto Scaling Groups page. The navigation bar at the top includes links for IAM, S3, EC2, Simple Notification Service, VPC, Aurora and RDS, Billing and Cost Management, and the account ID. The main content area displays a table titled "Auto Scaling groups (1/3) Info". The table has columns for Name, Launch template/configuration, Instances, Status, Desired capacity, Min, and Max. Three rows are listed: "cloth-as" (Launch template: clothpage | Version Default, Instances: 2, Status: -), "mobile-as" (Launch template: mobilepage | Version Default, Instances: 2, Status: -), and "home-as" (Launch template: homepage | Version Default, Instances: 2, Status: -). The "home-as" row is selected, indicated by a blue border around the entire row. A "Actions" dropdown menu is open above the table, showing options "Edit" and "Delete". The left sidebar contains sections for 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, Settings).

The screenshot shows the "Edit" dialog for the "home-as" Auto Scaling Group. The title bar says "Auto Scaling group: home-as". The main content area is titled "Load balancing - optional". It instructs the user to attach the Auto Scaling group to an existing load balancer or define a new one. Under "Load balancers", there is a checked checkbox for "Application, Network or Gateway Load Balancer target groups". A dropdown menu labeled "Select target groups" is open, showing a single item: "home-tg | HTTP". Below this, a note states: "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](#). [Learn more](#)". There is also an unchecked checkbox for "Classic Load Balancers". At the bottom, there is a link "Create and attach new load balancers" and a button "Add a new load balancer". The left sidebar is identical to the previous screenshot, showing the "Auto Scaling Groups" section under "Auto Scaling".

Performed equivalent edits for mobile-as and cloth-as so that each Auto Scaling Group was registered with its respective target group.

30. Opened website-lb in the Load Balancers list and went to the “Listeners and rules” tab, which initially showed a single HTTP:80 listener forwarding all traffic to home-tg.

The screenshot shows the AWS CloudWatch Metrics Insights interface. The search bar at the top contains the query "CloudWatch Metrics Insights". Below the search bar, there are two main sections: "Metrics" and "Logs". The "Metrics" section displays a table with columns for "Metric Name", "Dimensions", "Unit", and "Value". One row is highlighted, showing "CloudWatch Metrics Insights" with dimensions "Region:us-east-1" and "MetricName:CloudWatch Metrics Insights". The "Logs" section shows a table with columns for "Log Stream", "Timestamp", and "Message". One log entry is visible, timestamped "2023-09-18T12:00:00+00:00" with the message "CloudWatch Metrics Insights". At the bottom of the page, there are navigation links for "Metrics", "Logs", and "Metrics + Logs".

Clicked “Manage rules” and selected “Add rule” to define additional listener rules for path-based routing.

31. In the “Add rule” wizard, set the rule condition to “Path” with the match value `/mobile/*`, using value matching syntax with wildcard support.

The screenshot shows the AWS CloudWatch Metrics Insights interface. The search bar at the top contains the query "CloudWatch Metrics Insights". Below the search bar, there are two main sections: "Metrics" and "Logs". The "Metrics" section displays a table with columns for "Metric Name", "Dimensions", "Unit", and "Value". One row is highlighted, showing "CloudWatch Metrics Insights" with dimensions "Region:us-east-1" and "MetricName:CloudWatch Metrics Insights". The "Logs" section shows a table with columns for "Log Stream", "Timestamp", and "Message". One log entry is visible, timestamped "2023-09-18T12:00:00+00:00" with the message "CloudWatch Metrics Insights". At the bottom of the page, there are navigation links for "Metrics", "Logs", and "Metrics + Logs".

32. For the same rule, configured the action “Forward to target groups” and selected mobile-tg as the target group, leaving weight at 1 (100 percent of traffic for that 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 Target type: Instance, IPv4 Target stickiness: Off	HTTP <input type="button" value="▼"/>	1 0-999

Add target group
You can add up to 4 more target groups.

Target group stickiness Info
Enables the load balancer to bind a user's session to a specific target group. To use stickiness the client must support cookies. If you want to bind a user's session to a specific target, turn on the Target Group attribute Stickiness.

Turn on target group stickiness

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

33. Assigned this rule a higher priority value (for example, priority 1) so that it is evaluated before the default rule, then reviewed the rule details showing the path condition /mobile/* and the forward action to mobile-tg, and created the rule.

Step 1 Add rule
Step 2 Set rule priority
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

Priority	Name tag	Conditions (If)	Transforms	Actions (Then)
1	-	Path (value) = [/mobile/*] Priority value must be 1-50,000.	-	<ul style="list-style-type: none"> Forward to target mobile-tg: 1 (100% Target group stickiness)
Last (default)	Default	If no other rule applies	-	<ul style="list-style-type: none"> Forward to target home-tg: 1 (100% Target group stickiness)

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

Review and create

Listener details: HTTP:80

Rule details

Priority 1	Rule ARN Pending
Conditions If request matches all: Path (value) = /mobile/*	Transforms -
Actions	
<ul style="list-style-type: none"> Forward to target group mobile-tg: 1 (100%) 	

Rule tags (0)

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

Key	Value
No tags found	

CloudShell Feedback Console Mobile App © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Similarly, created another rule with a path condition `/cloth/*` and an action forwarding to `cloth-tg`, giving it an appropriate priority, while keeping the default rule forwarding remaining traffic to `home-tg`.

After saving the rules, the `HTTP:80` listener on `website-lb` had multiple rules: one default rule forwarding to `home-tg` and additional rules routing `/mobile/*` and `/cloth/*` paths to `mobile-tg` and `cloth-tg` respectively.

34. From the Load Balancers page, copied the DNS name of `website-lb` once its status was Active.

Load balancers (1/1) [What's new?](#)

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
website-lb	Active	application	Internet-facing	IPv4	vpc-0f9cea248655056b6	3 Availability Zones

Load balancer: website-lb

Load balancer ARN
arn:aws:elasticloadbalancing:ap-south-1:673586849368:loadbalancer/app/website-lb/e8bca896ab315341

DNS name copied
website-lb-172332190.ap-south-1.elb.amazonaws.com (A Record)

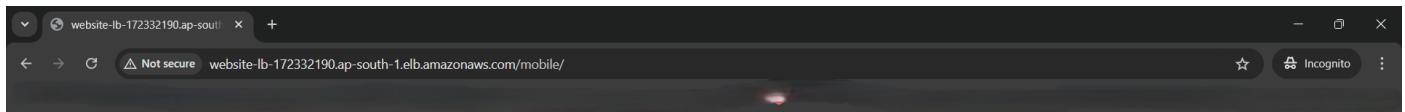
CloudShell Feedback Console Mobile App © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

35. In a browser, access `http://website-lb-.../mobile/` and verify that the “This is mobile page” HTML content is served through the ALB from one of the mobile Auto Scaling instances.

website-lb-172332190.ap-south-1.elb.amazonaws.com

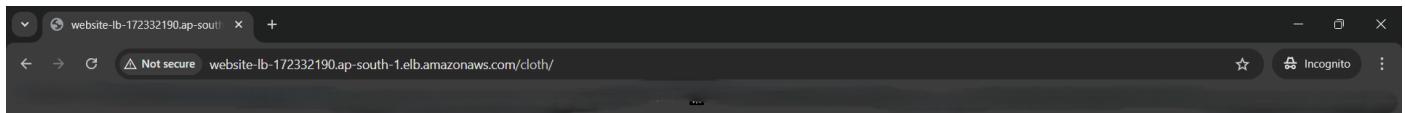
This is home page: ip-172-31-33-100.ap-south-1.compute.internal

36. In a browser, access <http://website-lb-.../> (root path) and confirm that the “This is home page” content is returned from the home Auto Scaling Group.



This is mobile page: ip-172-31-2-166.ap-south-1.compute.internal

37. In a browser, access <http://website-lb-.../cloth/> and verify that the “This is cloth page” content is served from the cloth Auto Scaling Group.



38. On the EC2 “Instances” page, view all running and terminated instances and confirm that multiple t3.micro instances across different Availability Zones are running as part of the three Auto Scaling Groups.

A screenshot of the AWS EC2 Instances page. The left sidebar shows navigation options like Dashboard, AWS Global View, Instances, Images, and CloudShell. The main area displays a table titled 'Instances (8) Info' with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 IP. The table lists 8 instances, mostly of type 't3.micro', running in various Availability Zones (ap-south-1a, ap-south-1b, ap-south-1c, ec2-13-204-68, ec2-13-203-78, ec2-13-203-79).

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 IP
i-003adb6077bcce7c6	i-003adb6077bcce7c6	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1b	ec2-13-204-68
i-0e08fbaa34806f194	i-0e08fbaa34806f194	Terminated	t3.micro	-	View alarms +	ap-south-1a	-
i-0d70cce6ab84a2991	i-0d70cce6ab84a2991	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1c	ec2-13-203-78
i-033e0679f05d43d8d	i-033e0679f05d43d8d	Terminated	t3.micro	-	View alarms +	ap-south-1c	-
i-05f40fd9a315d54d5	i-05f40fd9a315d54d5	Terminated	t3.micro	-	View alarms +	ap-south-1a	-
i-06804c1473df9b028	i-06804c1473df9b028	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1c	ec2-13-203-79
i-0c2650adfc3773299d	i-0c2650adfc3773299d	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1b	ec2-13-203-78
i-0b04fd2c1b3f9cce4	i-0b04fd2c1b3f9cce4	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1c	ec2-13-203-2