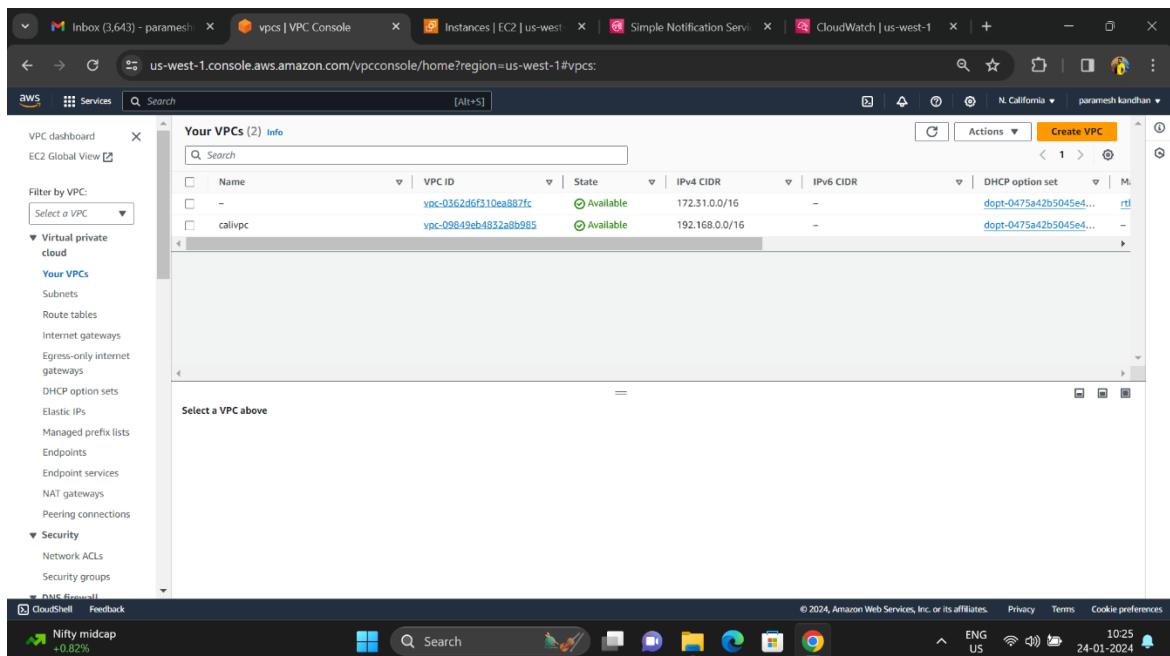
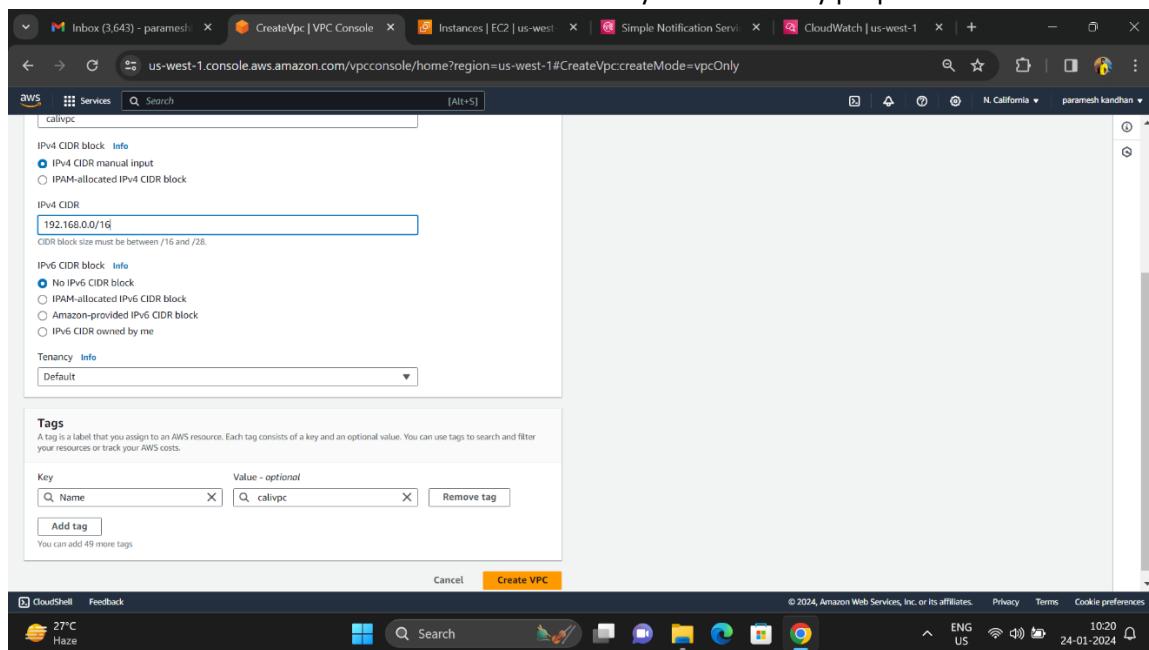


Implementing-Load-Balancing-with-Elastic-Load-Balancer

This project revolves around the implementation of Load Balancing with Elastic Load Balancer, employed to effectively regulate network traffic within a designated Virtual Private Cloud (VPC). In scenarios of heightened traffic, the system will dynamically spawn new instances through an auto-scaling mechanism, ensuring responsive and scalable infrastructure to accommodate varying workloads seamlessly. This strategic utilization of Elastic Load Balancer, coupled with auto-scaling capabilities, enhances the system's capacity to gracefully handle increased demand, thereby optimizing overall performance and resource allocation.

PROCESS:

- To create a aws account and then need to use the service of EC2 .
- In the webserver need to create a VPC for the security and the safety purpose.



The screenshot shows the AWS VPC Subnets console. A green banner at the top indicates "You have successfully created 1 subnet: subnet-083fe8cf90f439a22". The main table displays one subnet entry:

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
calipublic	subnet-083fe8cf90f439a22	Available	vpc-09849eb4832a8b985	192.168.0.0/24	-

Below the table, there is a section titled "Select a subnet". The bottom right corner of the screen shows the AWS CloudWatch Metrics dashboard with a value of "+0.51%".

- Then create a subnet in the class c . it will be start the IP adds: 192.168.0.10/24.

The screenshot shows the "CreateSubnet" wizard step. The first step, "Subnet 1 of 1", is displayed. The form fields are as follows:

- Subnet name:** calipublic
- Availability Zone:** No preference
- IPv4 VPC CIDR block:** 192.168.0.0/16
- IPv4 subnet CIDR block:** 192.168.0.10/24
- Tags - optional:** A single tag named "Name" with value "calipublic" is added.

The bottom right corner of the screen shows the AWS CloudWatch Metrics dashboard with a value of "28°C Haze".

- Sub netting allows organizations to divide their IP address space into smaller, more manageable segments. This is especially important in situations where there might be a limited pool of IP addresses available.
- Sub netting enables organizations to allocate IP addresses based on their specific needs for different departments, locations, or network segments.
- Sub netting can be used to isolate different parts of a network. This isolation can enhance security by controlling the flow of traffic between subnets and making it more challenging for unauthorized access or attacks to spread across the network.
- Network administrators can implement different security policies and access controls for each subnet, providing granular control over network resources.

The screenshot shows the AWS VPC console interface. On the left, a navigation pane lists various VPC-related services like EC2 Global View, Filter by VPC, Virtual private cloud, Your VPCs, Subnets, Route tables, Internet gateways, Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, Endpoints, Endpoint services, NAT gateways, Peering connections, Security, Network ACLs, Security groups, and DNS Firewall. The 'Internet gateways' section is currently selected. The main content area displays an Internet gateway named 'igw-0d5875ffffa0332a34'. It shows its ID as 'igw-0d5875ffffa0332a34', State as 'Attached', VPC ID as 'vpc-09849eb4832a8b985 | calivpc', and Owner as '714276899749'. A 'Tags' section contains a single tag 'Name: internet'. At the bottom right of the main area, there is a 'Manage tags' button. The browser's address bar shows the URL: `us-west-1.console.aws.amazon.com/vpcconsole/home?region=us-west-1#InternetGateway:id=igw-0d5875ffffa0332a34`. The status bar at the bottom indicates it's 11:08 on 24-01-2024.

- Need to enable the internet gateways for the purpose of connect the internet connectivity in the Public IP.

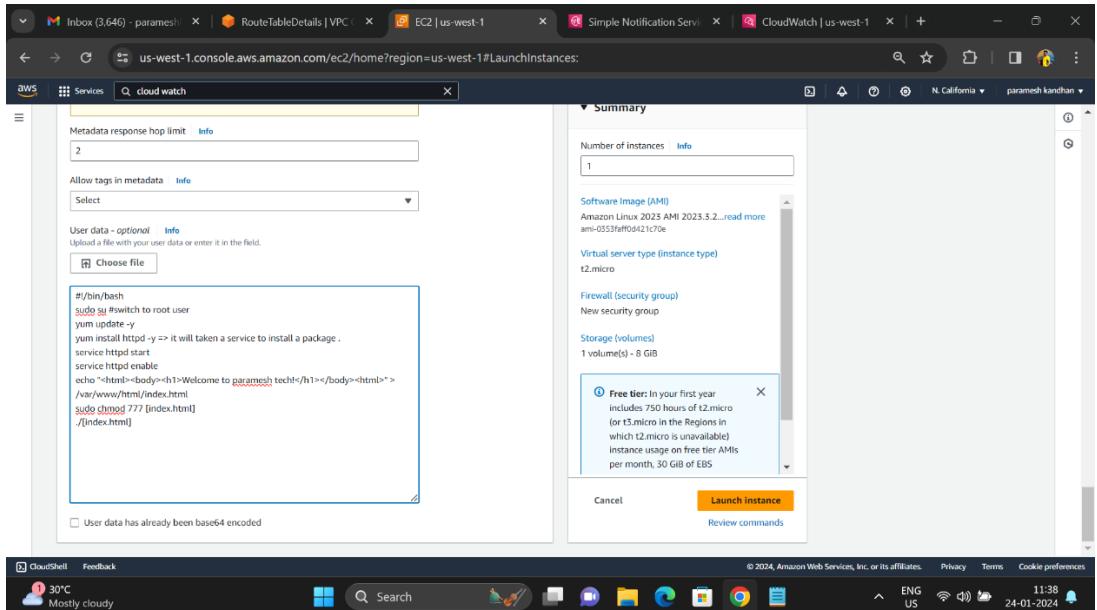
The screenshot shows the AWS VPC console interface. The navigation pane is identical to the previous one. The main content area is titled 'Edit routes' and shows a table for a route table named 'rtb-097089817956d040d'. The table has columns for Destination, Target, Status, and Propagated. One entry is visible: 'Destination: 192.168.0.0/16', 'Target: local', 'Status: Active', and 'Propagated: No'. Below this table is a search bar with 'Q: 0.0.0.0/0' and a dropdown menu showing 'Internet Gateway' and 'igw-0d5875ffffa0332a34'. At the bottom of the table are buttons for 'Add route', 'Cancel', 'Preview', and 'Save changes'. The browser's address bar shows the URL: `us-west-1.console.aws.amazon.com/vpcconsole/home?region=us-west-1#EditRoutes:RouteTableId=rtb-097089817956d040d`. The status bar at the bottom indicates it's 11:09 on 24-01-2024.

- Route tables can be populated through dynamic routing protocols (e.g., OSPF, EIGRP, BGP) or manually configured static routes. Dynamic routing protocols allow routers to exchange information about network reachability automatically, while static routes are manually configured by network administrators.
- Route tables are used in both IPv4 and IPv6 networks. They play a crucial role in directing traffic through the network based on the destination IP address.

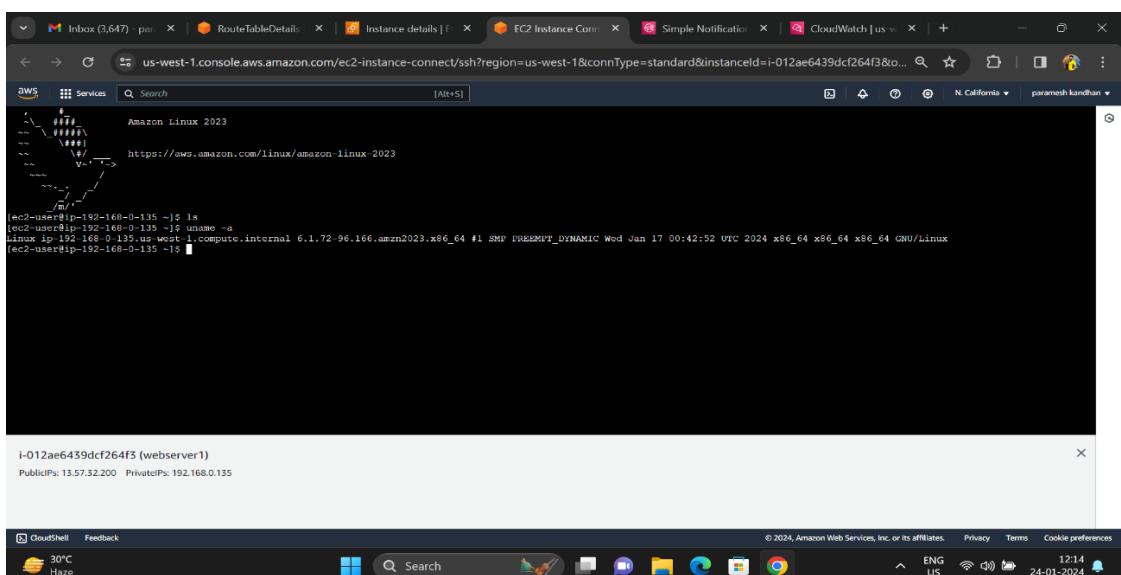
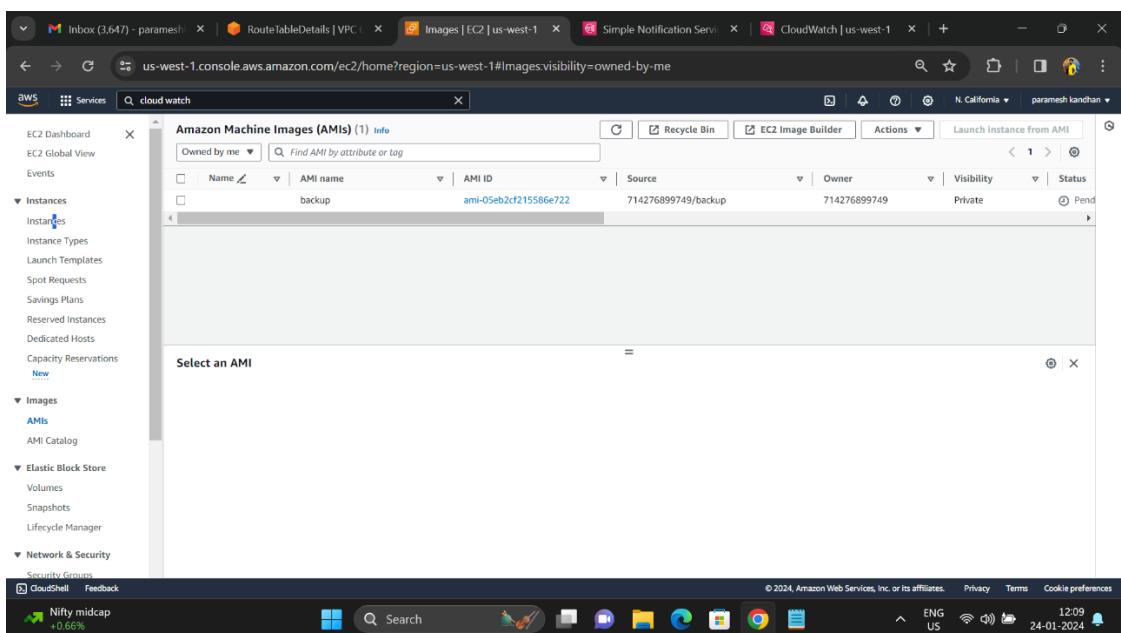
The screenshot shows the AWS VPC console with the URL us-west-1.console.aws.amazon.com/vpcconsole/home?region=us-west-1#RouteTableDetails:RouteTableId=rtb-097089817956d040d. The main title bar says "Updated routes for rtb-097089817956d040d / calipublicround successfully". The left sidebar is titled "Route tables" and lists various VPC components. The main content area shows the "Details" tab for the route table "rtb-097089817956d040d / calipublicround". It displays the Route table ID, Main status (No), Owner ID (714276899749), and VPC (vpc-09849eb4832a8b985 | calivpc). Under the "Routes" tab, there are two entries: "0.0.0.0/0" pointing to "igw-0d5875ffa0352a34" with "Status" as "Active" and "Propagated" as "No"; and "192.168.0.0/16" pointing to "local" with "Status" as "Active" and "Propagated" as "No". The bottom navigation bar includes links for CloudShell, Feedback, and a weather icon indicating 28°C Haze.

The screenshot shows the AWS EC2 console with the URL us-west-1.console.aws.amazon.com/ec2/home?region=us-west-1#Instances. The main title bar says "Instances (1/1) Info". The left sidebar is titled "Instances" and lists Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, and Network & Security. The main content area shows a table for "Instances (1/1) Info" with one row for "webserver1" (Instance ID: i-012ae6439dcf264f3, Status: Running, Type: t2.micro). Below the table, the "Instance: i-012ae6439dcf264f3 (webserver1)" details page is shown, providing comprehensive information about the instance's configuration and network settings. The bottom navigation bar includes links for CloudShell, Feedback, and a weather icon indicating 28°C Haze.

- Then need to create a ec2 instance under the VPC
- In the ec2 instance need to install the httpd package.
- # Yum install httpd* -y. Using a shell script enable the httpd on create a instance with the web sever.
- #!/bin/bash
- sudo su #switch to root user
- yum update -y
- yum install httpd -y => it will taken a service to install a package .
- service httpd start
- service httpd enable
- echo "<html><body><h1>Welcome to paramesh tech!</h1></body></html>" > /var/www/html/index.html
- sudo chmod 777 [index.html]
- ./[index.html]



- We are using the auto scaling concept for the purpose of taking a AMI as a backup.



- Now create a linux server for the web deployment.

```

<html>
  <body>
    <h1> welcome to the cybertech </h1>
  </body>
</html>

[ec2-user@ip-192-168-0-135 html]$ vim index.html
[ec2-user@ip-192-168-0-135 html]$ cat index.html
<html>
  <body>
    <h1> welcome to the cybertech 1 </h1>
  </body>
</html>

[ec2-user@ip-192-168-0-135 html]$ ^C
[ec2-user@ip-192-168-0-135 html]$ service httpd status
Redirecting to /bin/systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
     Active: active (running) since Wed 2024-01-24 06:48:04 UTC; 23min ago
       Docs: man:httpd(8)
   Main PID: 25879 (httpd)
     Status: "Total requests: 4; Idle/Busy workers 98/2;Requests/sec: 0.00284; Bytes served/sec: 2 B/sec"
        Tasks: 230 (limit: 1114)
      Memory: 16.7M
         CPU: 887ms
      CGroup: /system.slice/httpd.service
              ├─25879 /usr/sbin/httpd -DFOREGROUND
              ├─25880 /usr/sbin/httpd -DFOREGROUND
              ├─25881 /usr/sbin/httpd -DFOREGROUND
              ├─25882 /usr/sbin/httpd -DFOREGROUND
              ├─25883 /usr/sbin/httpd -DFOREGROUND
              └─26992 /usr/sbin/httpd -DFOREGROUND

Jan 24 06:48:04 ip-192-168-0-135.us-west-1.compute.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Jan 24 06:48:04 ip-192-168-0-135.us-west-1.compute.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Jan 24 06:48:04 ip-192-168-0-135.us-west-1.compute.internal httpd[25879]: Server configured, listening on port 80
[ec2-user@ip-192-168-0-135 html]$

```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG US 12:41 24-01-2024

- Setup a html code for the web hosting .

```

[ec2-user@ip-192-168-0-218 ~]$ hostname
ip-192-168-0-218.us-west-1.compute.internal
[ec2-user@ip-192-168-0-218 ~]$ sudo yum install httpd
Last metadata expiration check: 0:19:40 ago on Wed Jan 24 14:45:06 2024.
Dependencies resolved.

Transaction Summary
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Is this ok [y/N]: y

i-0c19694028e203a02 (webserver1)
PublicIPs: 54.67.95.223 PrivateIPs: 192.168.0.218

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG US 20:35 24-01-2024

```

```

[ec2-user@ip-192-168-0-218 ~]$ mod_lua-2.4.58-1.amzn2023.x86_64
[ec2-user@ip-192-168-0-218 ~]$ mod_http2-2.4.58-1.amzn2023.x86_64
[ec2-user@ip-192-168-0-218 ~]$ libbrotli-1.0.9-4.amzn2023.x86_64
[ec2-user@ip-192-168-0-218 ~]$ httpd-2.4.58-1.amzn2023.x86_64
Running scriptlet: httpd-2.4.58-1.amzn2023.x86_64
Verifying : apr-util-openssl-1.6.3-3.amzn2023.0.1.x86_64
Verifying : mod_lua-2.4.58-1.amzn2023.x86_64
Verifying : mod_http2-2.4.58-1.amzn2023.x86_64
Verifying : libbrotli-1.0.9-4.amzn2023.x86_64
Verifying : httpd-2.4.58-1.amzn2023.x86_64
Verifying : apr-1.7.2-2.amzn2023.0.2.x86_64
Verifying : apr-util-1.6.3-1.amzn2023.0.1.x86_64
Verifying : httpd-core-2.4.58-1.amzn2023.x86_64
Verifying : mailcap-2.1.49-3.amzn2023.0.3.noarch
Verifying : httpd-filesystem-2.4.58-1.amzn2023.noarch
Verifying : generic-logos-httpd-18.0.0-12.amzn2023.0.noarch

Installed:
apr-1.7.2-2.amzn2023.0.2.x86_64           apr-util-1.6.3-1.amzn2023.0.1.x86_64   apr-util-openssl-1.6.3-3.amzn2023.0.1.x86_64   generic-logos-httpd-18.0.0-12.amzn2023.0.noarch
httpd-2.4.58-1.amzn2023.x86_64            httpd-core-2.4.58-1.amzn2023.x86_64   httpd-filesystem-2.4.58-1.amzn2023.noarch          generic-tools-2.4.58-1.amzn2023.x86_64
libbrotli-1.0.9-4.amzn2023.0.2.x86_64      mailcap-2.1.49-3.amzn2023.0.3.noarch   mod_http2-2.4.58-1.amzn2023.x86_64             mod_lua-2.4.58-1.amzn2023.x86_64

Complete!
[ec2-user@ip-192-168-0-218 ~]$ service httpd start
Redirecting to /bin/systemctl start httpd.service
See system logs and 'systemctl status httpd.service' for details.
[ec2-user@ip-192-168-0-218 ~]$ sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
[ec2-user@ip-192-168-0-218 ~]$

i-0c19694028e203a02 (webserver1)
PublicIPs: 54.67.95.223 PrivateIPs: 192.168.0.218

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG US 20:35 24-01-2024

```

```

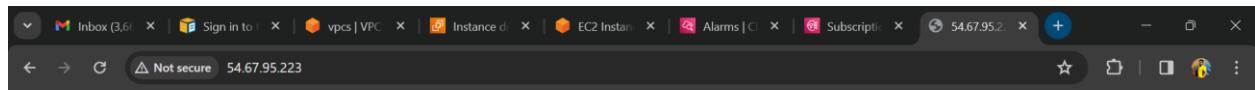
Complete!
[ec2-user@ip-192-168-0-218 ~]$ service httpd start
Redirecting to /bin/systemctl start httpd.service
Failed to start httpd.service: Access denied
See system logs and 'systemctl status httpd.service' for details.
[ec2-user@ip-192-168-0-218 ~]$ sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
[ec2-user@ip-192-168-0-218 ~]$ systemctl status httpd.service
● httpd.service - Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
     Active: active (running) since Wed 2024-01-24 15:05:53 UTC; 35s ago
       Docs: man:httpd.service(8)
    Main PID: 26302 (httpd)
      Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec: 0 B/sec"
        Tasks: 177 (limit: 1114)
       Memory: 13.2M
          CPU: 82ms
         CGroup: /system.slice/httpd.service
             ├─26302 /usr/sbin/httpd -DFOREGROUND
             ├─26303 /usr/sbin/httpd -DFOREGROUND
             ├─26304 /usr/sbin/httpd -DFOREGROUND
             ├─26305 /usr/sbin/httpd -DFOREGROUND
             └─26306 /usr/sbin/httpd -DFOREGROUND

Jan 24 15:05:53 ip-192-168-0-218.us-west-1.compute.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Jan 24 15:05:53 ip-192-168-0-218.us-west-1.compute.internal systemd[1]: Started httpd.service - The Apache HTTP Server...
Jan 24 15:05:53 ip-192-168-0-218.us-west-1.compute.internal httpd[26302]: Server configured, listening on: port 80
[ec2-user@ip-192-168-0-218 ~]$ i-0c19694028e203a02 (webserver1)
PublicIPs: 54.67.95.223 PrivateIPs: 192.168.0.218

```

The screenshot shows a terminal window within an AWS CloudShell interface. The user has run several commands to start the Apache HTTPD service on an EC2 instance. The output includes logs from the service's startup, its current status (active and running), and a list of its processes. At the bottom, the instance identifier (i-0c19694028e203a02) and its public IP address (54.67.95.223) are displayed.

- And the server 1 it will be up in the server IP ad 54.69.95.233



Welcome to the cyber tech 1

Learn about the future



- And then again create a server for the purpose of the load balancer.
- Before create a load balancer need to be create a target groups.
- The target group is combine of two ec2 instances.
- A load balancer is a critical component in computer networking and web hosting that distributes incoming network traffic or application requests across multiple servers. The primary purpose of a load balancer is to ensure that no single server bears too much load, preventing potential performance bottlenecks and improving the overall reliability and availability of the application or website.
- Load balancers distribute incoming traffic across multiple servers based on various algorithms, such as Round Robin, Least Connections, or Weighted Round Robin. This helps evenly distribute the load and prevent overloading of any single server.

- To create a target group.

The screenshot shows the 'Create Target Group' step in the AWS EC2 console. It lists three target types:

- IP addresses
 - Supports load balancing to VPC and on-premises resources.
 - Facilitates routing to multiple IP addresses and network interfaces on the same instance.
 - Offers flexibility with microservice based architectures, simplifying inter-application communication.
 - Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv4-to-IPv6 NAT.
- Lambda function
 - Facilitates routing to a single Lambda function.
 - Accessible to Application Load Balancers only.
- Application Load Balancer
 - Offers the 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.

Target group name: `caliserver`

Protocol : Port
Choose a protocol for your target group that corresponds to the Load Balancer type that will route traffic to it. Some protocols now include anomaly detection for the targets and you can set mitigation options once your target group is created. This choice cannot be changed 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

- And then create a load balancer.

The screenshot shows the 'Create Application Load Balancer' step in the AWS EC2 console. It includes sections for basic configuration and how application load balancers work.

Basic configuration

Load balancer name: `loadall`

Scheme: **Info**
Scheme can't be changed after the load balancer is created.

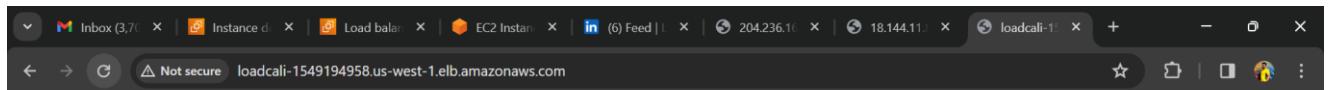
Internet-facing
An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. [Learn more](#)

Internal
An internal load balancer routes requests from clients to targets using private IP addresses.

IP address type: **Info**
Select the type of IP addresses that your subnets use.

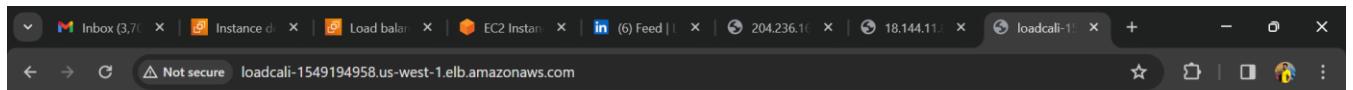
IPv4
Recommended for internal load balancers.

- It the output of load balancer,



welcome to cybertech 1

learn about future



Welcome to the cyber tech 2

Learn about the future



- In case the network traffic use a Auto Scaling helps you maintain application availability and allows you to scale your infrastructure dynamically based on predefined policies or schedules.
- The core component of Auto Scaling is the Auto Scaling Group. An Auto Scaling Group is a collection of Amazon EC2 instances that are created from a common Amazon Machine Image (AMI) and are designed to work together. The group automatically adjusts the number of instances in response to changes in demand.
- Scaling policies define when and how the Auto Scaling Group should adjust its capacity. There are two main types of scaling policies:

- **Scale Out (Add Capacity):** Triggered when a specified metric exceeds a certain threshold, leading to the addition of new instances.
- **Scale In (Remove Capacity):** Triggered when a specified metric falls below a certain threshold, resulting in the termination of instances.

The screenshot shows the AWS CloudShell interface. At the top, there are three tabs: 'Inbox (3,663) - parameshkandhan', 'RouteTables | VPC Console', and 'Auto Scaling groups | EC2 | us-west-1'. The main content area is titled 'Auto Scaling groups (1) Info' and shows a single entry for 'caliscale'. The table includes columns for Name, Launch template/configuration, Instances, Status, Desired capacity, Min, Max, and Availability Zones. The 'Availability Zones' column shows 'us-west-1a'. Below the table, it says '0 Auto Scaling groups selected'. The bottom of the screen shows the Windows taskbar with icons for File Explorer, Edge, Google Chrome, and others.

- Then given a artificial load to the using a command of "cat /dev/random > /dev/null"

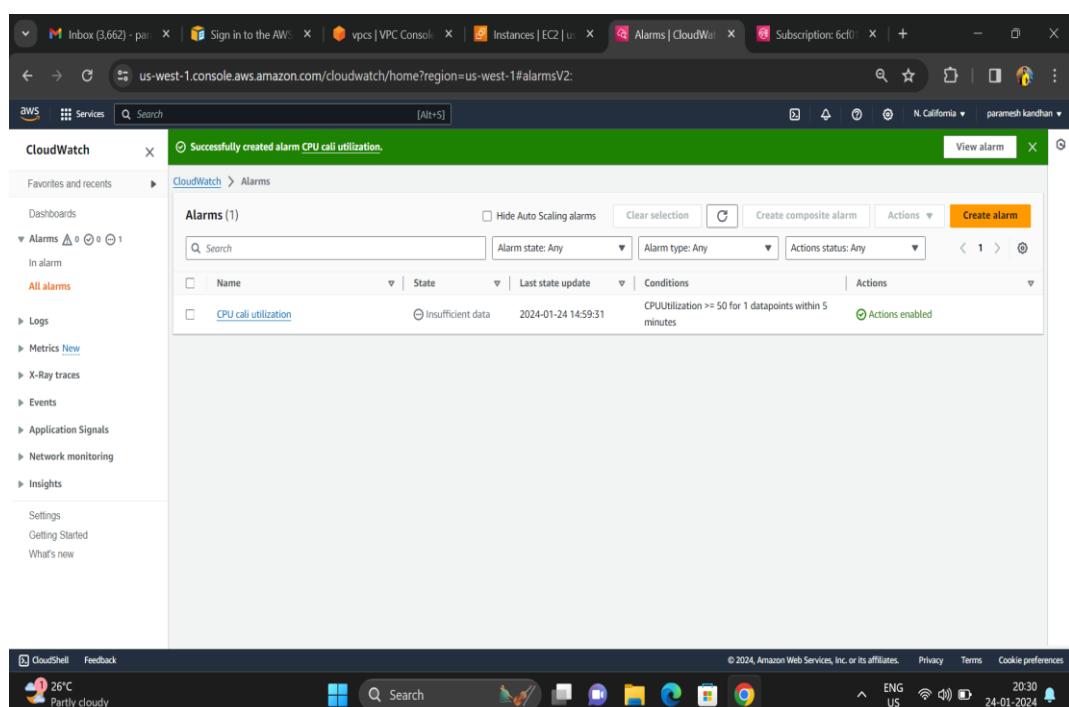
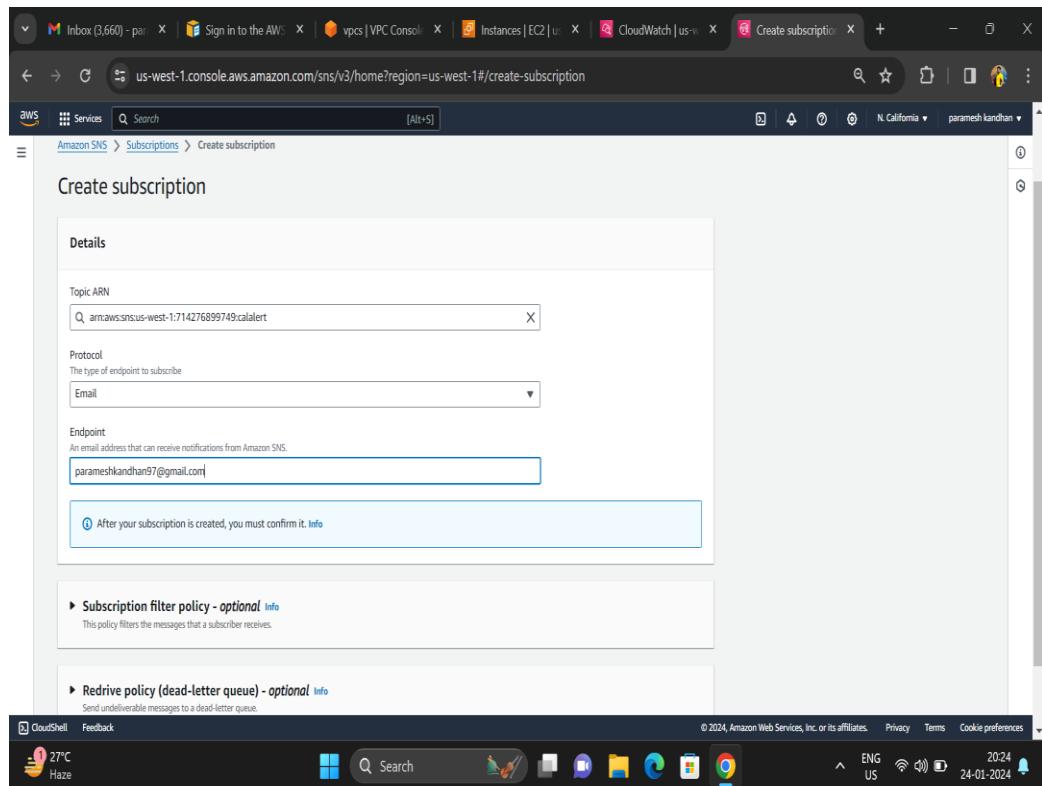
The screenshot shows the AWS CloudShell interface. At the top, there are five tabs: 'Inbox (3,662) - parameshkandhan', 'Recommended Jobs', 'RouteTables | VPC', 'Instances | EC2', and 'EC2 Instance Connect'. The main content area is a terminal window showing a session on an Amazon Linux 2023 instance. The user runs the command 'cat /dev/random > /dev/null', which generates a large amount of data. The terminal also shows the user's login information: 'Last login: Wed Jan 24 15:02:28 2024 from 13.52.6.115 (ec2-user@ip-192-168-0-218)\$. sudo cat /dev/random > /dev/null'. The bottom of the screen shows the Windows taskbar with icons for File Explorer, Edge, Google Chrome, and others.

- Then it automatically create a new ec2 instance.

The screenshot shows the AWS EC2 Instances page. The left sidebar contains navigation links for EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, CloudShell, and Feedback. The main content area displays a table titled 'Instances (4) Info' with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, and Public IP. The table lists four instances: webserver2 (terminated), webserver3 (running), and two other instances (t2.micro and t1.micro, both running). Below the table is a 'Select an instance' dropdown menu.

- For the monitoring purpose need to use the Amazon Cloud Watch and Amazon Simple Notification Service (SNS) are integral services in AWS that provide monitoring and notification capabilities, respectively. Combining these services can enable you to create professional and effective alerting and notification systems. Here's a guide on how to use CloudWatch and SNS in a professional way.
- Set up CloudWatch Alarms based on your custom metrics or predefined metrics for AWS resources. Alarms allow you to trigger actions when certain thresholds are breached.
- Integrate CloudWatch Logs to capture and analyze log data. Use CloudWatch Logs Insights for advanced querying and analysis of log data.
- To create a cloud watch need to be add the Organize subscribers using SNS Topics. Topics act as communication channels for sending messages to multiple endpoints.
- Subscribe your endpoints (email addresses, SMS numbers, HTTP endpoints, etc.) to relevant SNS Topics. This allows them to receive notifications.

The screenshot shows the AWS SNS Create topic page. The top navigation bar includes links for Inbox (3,662), Recommended Job, RouteTables | VPC, Instances | EC2, EC2 Instance Conn, How to give a load, and a search bar. The main content area has a 'Create topic' button and a 'Details' section. Under 'Type', 'Standard' is selected over 'FIFO (first-in, first-out)'. The 'Name' field is set to 'calalert'. The 'Display name - optional' field contains 'the cpu utilization is high'. The bottom of the page includes a CloudShell link, a feedback button, and a status bar showing 27°C Haze, 2023, 24-01-2024, and 21:54.



- Connect CloudWatch Alarms to SNS Topics for notification delivery. This ensures that when an alarm is triggered, relevant teams or individuals are notified via the chosen communication channels.
- If your application spans multiple AWS regions, use SNS for cross-region notifications. This ensures that teams in different regions are informed of any issues.
- The cpu utilization is noticed and send through the cloud watch tab and send through the Mail.

Screenshot of the AWS EC2 Instances page showing terminated instances webserver2 and webserver3.

Instances (4/4) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IP
webserver2	i-02905c68e143b6e84	Terminated	12.micro	-	View alarms +	us-west-1a	-	-
webserver3	i-04a82e15d79e65f6c	Shutting-d...	12.micro	2/2 checks passed	View alarms +	us-west-1a	-	54.176

Metrics Overview:

- Percent: 99.7 (blue), 49.8 (orange)
- Bytes: 560k (blue), 280k (orange)
- Bytes: 35.1k (blue), 17.6k (orange)
- Count: 619 (blue), 309 (orange)
- Network packets out (count): 229 (blue), 115 (orange)
- Disk reads (bytes): 1 (blue), 0.5 (orange)
- Disk read operations (operations): 1 (blue), 0.5 (orange)
- Disk writes (bytes): 1 (blue), 0.5 (orange)
- Disk write operations (operations): 0 (blue), 0 (orange)
- CPU credit usage (count): 0 (blue), 0 (orange)
- CPU credit balance (count): 0 (blue), 0 (orange)

CloudShell Feedback: 26°C Partly cloudy

Screenshot of a Gmail inbox showing an alarm notification from CloudWatch.

ALARM: "CPU cali utilization" in US West (N. California)

the cpu utilization is high <no-reply@sns.amazonaws.com>
to me ▾
9:55 PM (2 minutes ago)

You are receiving this email because your Amazon CloudWatch Alarm "CPU cali utilization" in the US West (N. California) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [99.67213114754097 (24/01/24 16:15:00)] was greater than or equal to the threshold (50.0) (minimum 1 datapoint for OK → ALARM transition)." at "Wednesday 24 January, 2024 16:25:26 UTC".

View this alarm in the AWS Management Console:
<https://us-west-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=us-west-1#alarmsV2:alarm%20CPU%20cali%20utilization>

Alarm Details:

- Name: CPU cali utilization
- Description:
- State Change: OK → ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [99.67213114754097 (24/01/24 16:15:00)] was greater than or equal to the threshold (50.0) (minimum 1 datapoint for OK → ALARM transition).
- Timestamp: Wednesday 24 January, 2024 16:25:26 UTC
- AWS Account: 714276899749
- Alarm Arn: arn:aws:cloudwatch:us-west-1:714276899749:alarm:CPU cali utilization

Threshold:

CloudShell Feedback: 21:58 24-01-2024